SMAC

LAC-26

TECHNICAL REFERENCE MANUAL

Revision 0, April 2013

SMAC CORP.
5807 Van Allen Way
Carlsbad, CA 92008

S.M.A.C.
5807 VAN ALLEN WAY
CARLSBAD, CA 92008

PHONE: 1-760-929-7575 / FAX: 1-760-929-7588

FOR TECHNICAL ASSISTANCE CALL : 1-760-929-7575

TABLE OF CONTENTS

1. Introduction

The LAC-26 is a two axis stand-alone integrated controller / driver, with input / output (I/O) capabilities similar to the LAC-25. The first axis is for the control of DC brush type motors or actuators with its integrated driver. The second axis features a high power driver for 3 phase brushless, single phase bushed, and voice coil inductive load motors.

The LAC-26 implements a mnemonic type command instruction set via a standard RS-232 serial communications interface. These commands can be executed directly or used to create command macros which are stored in the onboard nonvolatile RAM (NVRAM).

The LAC-26 can interface to the real world via the onboard motor drivers, 2 channels of
quadrature type encoder interface, 4 channels of optoisolated digital input and 2 channels of optoisolated digital output, with additional optoisolated inputs serving for limit, home and fault functions, 5 channels of 10-bit analog to digital (A/D) conversion (2 of which are reserved for monitoring amplifier output current), and an RS-232 serial communications link. A proprietary RS-422 interface is provided for future I/O expansion modules.

The LAC-25 controller card embedded within can control Digi Flex servo driver through the following I/O ports. Digital Output 2 – starting auto commutation signal, Digital Output 3 – enable driver, Digital Input 3 – commutation successful signal from driver, and axis 2 analog output – commutation.

1.1 Specifications

Description	**Stand-Alone 2 Axis Servo Motor Controller / Driver**
Operating Modes	Position, Velocity, Torque, Electronic Gearing
Control Algorithm	PID
Max. Servo Loop Rate	100 µS per enabled axis
Servo Position Feedback	Incremental Encoder with Index
Output (Standard)	PWM Motor Drive, Single Phase (brushed), 3 Amps Cont. and 6 Amps Peak at 50 VDC Max.
PWM Frequency	Approximately 19.531 KHz
Encoder and Index Input	5V Single-ended or Differential
Encoder Supply Voltage	5 VDC
Encoder Count Rate	2 Million Quadrature Counts per Second
Position Range	32 Bits
Velocity Range	32 Bits
Acceleration Range	32 Bits
General Purpose Digital I/O	3 Optoisolated inputs, 2 Optoisolated outputs
Dedicated Digital Inputs	Limit+, Limit-, Home and Fault, for each axis
Analog Inputs	5 Channels With 10-Bit Resolution, 3 are user accessible
Analog Output2	1 Channels, with 12-bits Resolution, ±10 VDC
Expansion I/O	Optional Expansion to 64 I/O
Communication Interface	RS-232
Supply Voltage	+11 To +50 VDC
Motor Voltage	+12 To +48 VDC
Dimensions	226mm Long by 88mm wide by 73mm Thick
Weight	Approximately 1.2 Kg

2nd Axis Driver Specification

Description	DigiFlex Servo Drive
Operation Mode	Current Loop(50 us), Velocity Loop(100 us), Position Loop(100 us)
Command Source	±10 V Analog, 5V Step and Direction, Encoder Following, PWM and Direction
Feed Back Support	Incremental Encoder (Max 20 MHz Quadrature), Halls, ±10VDC Position, Aux Incremental Encoder
Commutation Methods	Sinusodal, Trapezoidal
Output (Motor Supported)	Single Phase Brushed, Voice Coil, Inductive Load ; Three Phase (Brushless); 12A peak, 6A Continuous current
Communication Interface	RS-232, RS-485
Supply Voltage	+20 To +80 VDC
Logic Supply Voltage	5 VDC
Option	1st axis Configured

1.2 Digital I/O Interface

The LAC-26 includes 4 channels of general purpose digital input and 4 channels of general purpose digital output. Additionally, there are four channels of dedicated digital input for each of the two axis'. All of these I/O are protected through the use of optoisolators.

1.2.1 Dedicated Digital Inputs

Figure 1 illustrates one of the LAC-26's dedicated digital inputs. These inputs are Limit+, Limit-, Home and Fault for both axis of the LAC-26. The power to activate these inputs is provided by the LAC-26 so it is only necessary to complete the circuit to activate the input.

The Limit inputs are intended for signaling the LAC-26 that an axis has reached it's end of travel. When such an event occurs, the LAC-26 can ignore the event or stop the servo in some controlled fashion. The Home input is for detecting some sort of "home position" sensor. This can be used with the encoder index input to implement a very accurate homing method. A typical use for the Fault input is for an external device to signal a fault condition such as over-temperature.

Note: The external Fault input is tied to the internal over-temperature signal from the onboard drivers. When a fault condition occurs, that is either the internal over-temperature signal or external Fault signal go active, the 16-bit internal variable FCNT (see Internal Variables) begins to increment at 1 rate of once per millisecond. If the fault condition clears then the FCNT variable is also cleared. If the fault condition remains present long enough for the FCNT variable to count up to the value assigned to the FCMP variable, then the over Fault bit in the status word will be set and the servo will be disabled (assuming the Fault interrupt has not been enabled). The default value for FCMP is 10000 which will give a 10 second delay before causing the Fault bit to be set.

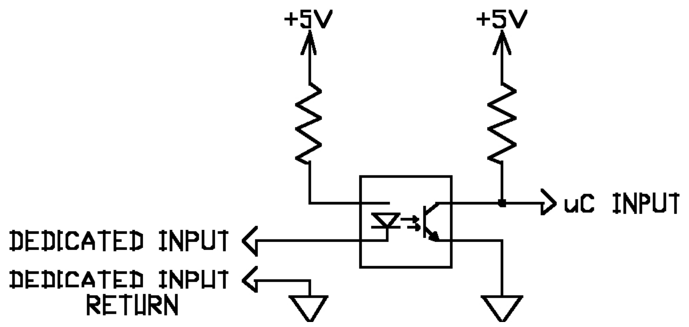


Figure 1. LAC-26 Dedicated Input.

1.2.2 General Purpose Inputs

Figure 2 illustrates one of the LAC-26's general purpose inputs. These inputs are galvanically isolated from the LAC-26. Current of the proper polarity must be supplied to the circuit to activate the input.

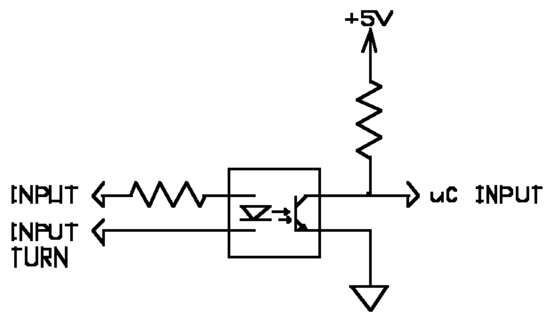


Figure 2. LAC-26 General Purpose Inputs

1.2.3 General Purpose Outputs

Figure 3 illustrates one of the LAC-26's general purpose outputs. These outputs are galvanically isolated from the LAC-26. When an output is activated, positive current will flow from the collector of the optocoupler transistor (the output pin) to it's emitter (the output return pin),

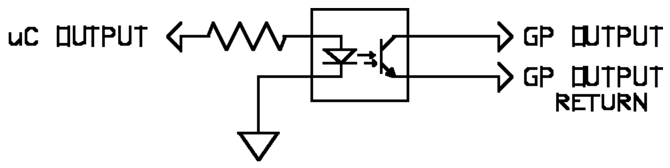


Figure 3. LAC-26 General Purpose Output

1.2.4 Digital I/O "States"

There are several commands that deal with controlling the digital I/O. All of these commands operate based on the following philosophy: With regard to an input, "active" means there is sufficient current flowing through that input and "inactive" means there is lack of sufficient current through that input. With regard to outputs, "active" means the ability for an output to pass current and "inactive" means the inability for an output to pass current.

The Channel High (CH) and Channel Low (CL) commands provide the user with the ability to determine whether a channel is active in the "on" state (CH) or active in the "off" state (CL). This is analogous to a switch and to whether it is normally open or normally closed. The Channel On (CN) and Channel Off (CF) commands do exactly as they imply in that they will turn a given output either on or off, which will make that output either active or inactive depending on the CH and CL commands as stated previously.

The (CH) command causes the following interpretation of the inputs and outputs:

An "activated" output is considered to be ON (e.g., Channel On "CN" command).
An "inactivated" output is considered to be OFF (e.g., Channel Off "CF" command).
An "activated" input is considered to be ON (e.g., Do If On "DN" command).
An "inactivated" input is considered to be OFF (e.g., Do If Off "DF" command).

The (CL) command causes the following interpretation of the inputs and outputs:

An "activated" output is considered to be OFF (e.g., Channel Off "CF" command).
An "inactivated" output is considered to be ON (e.g., Channel On "CN" command).
An "activated" input is considered to be OFF (e.g., Do If Off "DF" command).
An "inactivated" input is considered to be ON (e.g., Do If On "DN" command).

Input Current	"CH"	"CL"
Flowing	On	Off
No Flow	Off	On

Output Current	"CH"	"CL"
Flowing	CN	CF
No Flow	CF	CN

Table 2. I/O States.

Another feature of the digital input system is the ability for software input debouncing. All of the general purpose digital inputs are automatically sampled once every millisecond. Depending on the debounce delay set by the Input Debounce (ID) command, a given input must remain in the same state during one or more samplings before it is considered valid. If an input were to be found in a changed state during a sampling, the input would become invalid and the debounce delay would be restarted. If no or "0" debounce delay is used, then no input debouncing is performed. For example: if a "ID5" command has been issued, then a given input must remain in the same state for 5 samplings or for 5 milliseconds.

1.2.5 I/O Technical Specifications

1.2.5.1 General Purpose I/O Nominal Specifications.

	Unit	Specification
5	V	Minimum voltage to activate input.
0.83	mA	Input current at minimum activation voltage.
24	V	Maximum input voltage.
4.87	mA	Input current at maximum voltage.
1.1	V	Maximum voltage to deactivate input.
6	V	Absolute maximum reverse input voltage.
40	V	Maximum voltage output can sustain.
100	mA	Maximum current output can sustain.

Table 3: General Purpose I/O Specifications

1.2.5.2 Dedicated I/O Nominal Specifications.

	Unit	Specification
10	Ω	Maximum external circuit resistance to activate input.
1	K Ω	Minimum external circuit resistance to deactivate input.

Table 4: Dedicated I/O Specifications

1.3 Encoder Interface

The LAC-26 has two channels of quadrature type encoder interface with optional index signal input and the ability to supply +5 VDC at a minimum of 50 mA (or greater depending on other demands put on the internal 5 VDC power supply). The phase A+ and phase B+ inputs are pulled up to +5 VDC with 2.7K resistors, and the phase A- and phase B- inputs a biased at +2.5 VDC with 2.7K resistors. This arrangement which will accommodate both open collector and totem pole single-ended output encoders or differential output encoders. The phasing of the channels as well as the index signal sense can be changed via program command.

1.4 Output Driver Interface

The LAC-26 onboard output drivers are PWM switching amplifiers capable of supplying 3 Amps continuous and 6 Amps peak (for 200 mS minimum) at a switching frequency of approximately 19.531 KHz. These drivers are intended for driving DC brush type motors or actuators. Both drivers share the main power supply input and the peak voltage output to the motor will be nearly that of what is supplied.

The output drivers include an over-temperature sensor. If this sensor determines that the amplifier's temperature is greater than 140° C, the amplifier will then be disabled and the Over-Temperature bit will be set in that axis' status word.

For applications requiring capabilities above those of the onboard drivers, the ability to interface to external drivers is provided. This consists a 12-bit D/A ±10 VDC analog output signal for each axis. In applications where external drivers are not required, the analog outputs can be used for other purposes (e.g.,: oscilloscope monitoring of following error or output command).

1.5 Analog to Digital Conversion (A/D) Interface

The LAC-26 provides a 5 channel, 10 bit A/D conversion interface with a +10 VDC reference and analog ground. **For reverse compatibility purposes the A/D interface is actually ten channels but the user is only given access to channels 0, 1 and 2 while channels 6 and 7 are used internally for monitoring the output current of the onboard drivers. The other channels are unavailable and should be ignored.**

Whenever a Tell Analog "TA" or Get Analog "GA" command is issued, the specified A/D channel is converted and the result is either reported or stored for access by the user. Also, whenever the servo loop for an axis is executed, the "current monitoring" channel for that axis is converted and the result is stored for later access.

1.6 Serial Interface

The LAC-26 communicates with a host computer or a "dumb" terminal via an RS-232 serial interface. The baud rate is user selectable from 300 to 19,200 baud with **9600 baud being the default.** Characters are fixed at 8 bits in length with 1 stop bit and no parity. Software XON / XOFF handshaking is provided. Hardware handshaking is not supported at this time.

2. Macro Interrupt System

The LAC-26 employs a "Macro Interrupt System" to provide additional versatility in programming the LAC-26. This system comprises 32 interrupt sources with corresponding vectors. When an interrupt's source is enabled for operation and then becomes active, the current macro being executed is saved to a so called macro stack and execution of the macro specified by that interrupt's vector table entry begins. This happens to be similar procedure to that which the Macro Call (MC) command follows.

2.1 The Interrupt Vector Table

The Interrupt Vector Table consists of an entry for each interrupt source and each entry will correspond to that interrupt's level (level 0 = entry 0, level 1 = entry 1, etc.). A particular table entry must be loaded with the number of a valid macro to be executed should that interrupt source become active. The method for loading a vector table entry is provided by the Load Vector (LV) command. The user must first use the Accumulator Load (AL) command to set the number of the macro for a vector. The LV command is then used to transfer the low 8-bits of the accumulator to the vector table entry specified by the LV command. If an interrupt is generated and that vector table entry has not been defined (equal to 0) then the interrupt will not be executed. Note that this implies that macro "0" cannot be used as an interrupt macro. If an interrupt is generated and it's vector table entry has been defined but the macro it specifies has not, then an error will be reported.

2.2 Enabling and Disabling Interrupts

Loading a vector table entry will not enable an interrupt for operation. The Enable Vector (EV) command must be used for this purpose. When the EV command is used, it will enable the interrupt source (specified with the command) to function. In the event that it is necessary to disable an interrupt source, there is a Disable Vector (DV) command that functions in a similar manner as the EV command.

In order to prevent multiple or continuous interrupts, **as an interrupt is taken it is automatically disabled.** This means that the user must re-enable that interrupt using the EV command before it will occur again.

2.3 Interrupt Sources

The following table lists all the possible interrupt sources.

Interrupt Source	Level / Vector	Interrupt Source	Level/Vector
Axis 0 Error	31	Reserved	15
Axis 1 Error	30	Reserved	14
Reserved	29	Reserved	13
Reserved	28	Reserved	12
Axis 0 Fault	27	Reserved	11
Axis 1 Fault	26	Reserved	10
Reserved	25	Reserved	9
Reserved	24	Reserved	8
Axis 0 Limit	23	Reserved	7
Axis 1 Limit	22	Reserved	6
Reserved	21	Reserved	5
Reserved	20	Reserved	4
Axis 0 IP/IR	19	Digital Input 3	3
Axis 1 IP/IR	18	Digital Input 2	2
Reserved	17	Digital Input 1	1
Reserved	16	Digital Input 0	0

Table 5. Macro Interrupt Sources.

The Axis Error interrupts indicate that the position following error for a given axis has exceeded the limit set by the Set Error (SE) command. Normally, when this limit is exceeded, the servo is disabled and the "Error" bit in that axis' status word is set. **If the interrupt for this condition is enabled, the "Error" bit will still be set but the servo will not be disabled.**

The Axis Fault interrupts indicate that a fault condition (usually an over-temperature condition) has arisen. Normally, when this condition is detected, the servo is disabled and the "Fault" bit in that axis' status word is set. **If the interrupt for this condition is enabled, the "Fault" bit will still be set but the servo will not be disabled.**

The Axis Limit interrupts indicate that either a Limit+ or Limit- condition for an axis has been detected. Whether or not a limit input will be recognized is determined by the Limit On (LN) and Limit Off (LF) commands. The action taken is determined by the Limit Mode (LM) command.

Digital Inputs 00 - 03 provide 4 levels of undedicated, user definable interrupts. The interrupt for a given input will be active when that input is active.

2.4 Interrupt Priority

If more than one interrupt source becomes active at the same time, then the source with the higher level will be executed first. Level/vector 31 has the highest priority and level/vector 0 has the lowest priority.

2.5 Interrupt Completion

Once an interrupt macro (or set of macros) has finished executing, a Return from Call (RC) command or an undefined macro may be used to cause a return from the interrupting macro back to the interrupted macro where command execution will continue from where it was interrupted (see MS command). In cases where it is undesirable to return to the interrupted macro, the Unpush Macro (UM) command can be used to remove the previously pushed macro from the macro stack. This command can also be used to completely reset the macro stack in order that the user program can be restarted.

2.6 Interrupt Latency

Interrupt sources are sampled before each command in a macro is executed. This means that the amount of time that an interrupt is held off before execution (also known as interrupt latency) depends on how long it takes the previous command to complete. For most commands this delay will be imperceptible.

Commands such as Wait (WA), Wait for Edge (WE), Wait for Stop (WS), Wait for Off (WF), Wait for On (WN) and Wait for Index (WI) would normally be a source of unacceptable delay in that they can quite often be indeterminate in length. This problem has been avoided by making these instructions interruptable. For example, if a WA10000 command (a 10 second delay) is currently in progress and an interrupt comes along, the remaining delay period will be saved and then returned to after the interrupt has completed. If the interrupt were to take 3 seconds to execute, then the total wait time of the WA10000 command would be extended to 13 seconds.

The Position Mode (PM), Torque Mode (QM), Velocity Mode (VM), Wait for Position Absolute (WP) and Wait for Position Relative (WR) commands and any command that uses the serial communications link are all commands that could cause unacceptable interrupt latency. Therefore, their usage should be carefully considered where interrupts are possible.

3. Entering Commands

Immediately after power-up, the LAC-26 is ready to accept commands. To verify this, you can hit the ESC key. If everything is working properly, this should cause a greater than sign (">") prompt to appear on your display. If not, you need to verify that the power and communications connections are correct and verify the compatibility of the communications protocol.

Commands are entered via a "dumb" terminal or host computer such as a PC compatible. Commands sent to the LAC-26 should consist of standard ASCII characters, and the command lines should be followed by a carriage return. Linefeeds are not necessary since they are used for formatting and therefore they are ignored. As characters are entered at the keyboard, they should be echoed on your display. If your display echoes its own transmitted characters, you will want to issue the Echo Off (EF) command; otherwise, the Echo On (EN) command (which is the default mode) should be issued. If you enter an invalid command, the LAC-26 will respond with a question mark "?" followed by a code indicating the type of error and the Status LED will begin to blink. These codes are listed in Appendix A, LAC-26 Error Code Definitions.

If you make a mistake when entering a command, you can backspace to correct the error. If you are entering commands and change your mind, hitting the ESC key will cancel the line and give a new ">" prompt.

Once a command line has been entered and has finished executing, **hitting the RETURN key will cause the same command line to be re-executed.** While a set of commands are executing, hitting the space bar will cause command execution to pause until the space bar is hit again. Also, **if the ESC key is hit during execution or pause, command execution will be terminated,** and you will receive a new ">" prompt.

Command instructions are intended for use with the following syntax:

```
[Axis#]Command[Argument]<CR>
```

or...

```
[Axis#]Command[Argument],[Axis#]Command[Argument],...etc.
```

The axis number is normally specified as being from "1" to "2" with "0" being used to refer to both axis' at the same time. Once an axis has been specified, the same one will remain in effect until another is specified. For example, if the following were entered:

```
1SG100,SD500,SV1000000<CR>
```

or...

```
1SG100<CR>
SD500<CR>
SV1000000<CR>
```

the SG command specifies the axis number, so the subsequent SD and SV commands are performed on the same axis. If the following command were issued:

```
0TP<CR>
```

it would have the same affect as issuing these commands:

```
1TP<CR>
2TP<CR>
```

During interrupts and macro calls, the axis number is saved and then later restored. **After an interrupt macro entered or a macro call is taken, it is a good idea for the user to make sure an axis number (if necessary) gets set during one of the first commands encountered.**

The numerical range of an argument will vary depending on the command with which it is used. The mathematical interpretation of the argument will depend on whether the Decimal Mode (DM) or Hexadecimal Mode (HM) was the last issued (DM is the power on default). Both decimal and hexadecimal numbers less than zero should be entered with a preceding minus "-" sign. If no argument is given, then it will be assumed as "0". The exceptions to this are the Macro Define (MD), Macro Jump (MJ), Macro Call (MC),Macro Sequence (MS), Reset Macro (RM) and Tell Macro (TM), commands. It should be noted that commands can be strung together by using commas, up to a maximum line length of 127 characters.

If a command line is ended by a ";" and a comment, i.e...

```
>SG1000,SD5000  ;  Set  filter  gains.<CR>
```

then the ";" and anything following it to the end of the line will be ignored. This feature is not particularly useful if you are entering commands manually, as comments are not retained by the LAC-26. However, if commands are downloaded to the LAC-26 from a host computer, the ability for line comments can make program documentation possible and desirable.

3.1 Downloading Commands

In many cases, it is more convenient to enter commands using a text editor on a host computer and then download that text file to the LAC-26 using a communications program such as ProComm or the Microsoft Windows Terminal program. Whatever communications software is used, it must have the ability to provide a short delay (approx. 100 mS) after transmitting each line to give the LAC-26 time to interpret and store the commands that were just sent.

4. Introduction to Commands

The LAC-26 command instructions are varied and consist of several categories of purpose. The command descriptions will be detailed by these categories.

4.1 Parameter Commands

The parameter setting commands are considered to be those for setting the operating conditions of the servo system (i.e. PID filter gains, velocity, acceleration and etc.).

Command: aDBn -- Dead-Band --

Argument:	0 <= n <= 16383
Default:	0

This command sets the position following error dead-band for servo axis 'a'. The purpose for the DB command is to allow an acceptable static position error for which there will be no restoring force. This has the affect of reducing or eliminating "hunting" which is the continuous movement at or about a position in trying to seek that position. This is useful for applications that cannot tolerate this condition. Please note that the DB command is only in effect when the servo is not in motion (or when the Trajectory Complete bit is set in the servo status word).

Related Commands: TF

Command: aFAn -- Feed-forward, Acceleration --

Argument: 0 <= n <= 32767
Default: 0

This command allows for the adjustment of the PID digital filter acceleration feed-forward term for servo axis 'a'.

During the course of a Position Mode (PM) or Velocity Mode (VM) move, at any point during acceleration or deceleration (with a consistent load), the ideal required value of the servo output is fairly consistent and somewhat predictable.

During acceleration or deceleration:

OUTPUT = (VELOCITY * FV_CONSTANT) + (ACCELERATION * FA_CONSTANT)

During constant velocity:

OUTPUT = (VELOCITY * FV_CONSTANT)

If this value can be dynamically predicted and summed with the output of the PID digital filter, in effect, it reduces the burden of the PID filter to make lead/lag corrections based of the following error, thereby enhancing performance.

Related Commands: FV, OM, OO

Command: FF -- Fail Input Off --

Default: Off

This command has no effect but is retained for backward compatibility purposes.

Command: FN -- Fail Input On --

Default: Off

This command has no effect but is retained for backward compatibility purposes.

Command: aFRn -- Set Derivative Sampling Period --

Argument: 0 <= n <= 127
Default: 0

This command allows for the adjustment of the derivative sampling interval for servo axis 'a'. The period of this interval can be calculated by the following:

T = (n+1) * S * 0.000100

where "T" is the period in seconds, "n" is the FR command argument and "S" is the sample period count specified by the Servo Speed (SS) command. For example, if the value previously set by the SS command is 10 and the value set by the FR command is 1, then the derivative sample period will be:

(1+1) * 10 * 0.000100 = .002000 S or 2 mS

This command is useful in tuning the PID servo loop to the inertial properties of the system.

Related Commands: RI, SS

Command: aFVn -- Feed-forward, Velocity --

Argument: 0 <= n <= 32767
Default: 0

This command allows for the adjustment of the PID digital filter velocity feed-forward term for servo axis 'a'.

During the course of a Position Mode (PM) or Velocity Mode (VM) move, at any point along the way (with a consistent load), the ideal required value of the servo output is fairly consistent and somewhat predictable.

During acceleration or deceleration:

OUTPUT = (VELOCITY * FV_CONSTANT) + (ACCELERATION * FA_CONSTANT)

During constant velocity:

OUTPUT = (VELOCITY * FV_CONSTANT)

If this value can be dynamically predicted and summed with the output of the PID digital filter, in effect, it reduces the burden of the PID filter to make lead/lag corrections based of the following error, thereby enhancing performance.

Related Commands: FA, OM, OO

Command: aGRn -- Gear Ratio --

Argument: -8388607 <= n <= 8388607
Default: 0

This command sets the electronic gearing ratio for axis 'n'. A negative argument will cause a direction reversal in the electronic gearing. The argument to this command is the desired gearing ratio scaled by 65536.

Examples:

```
GR65536     ; Slave gear ratio is 1:1
GR131072    ; Slave gear ratio is 2:1
GR32768     ; Slave gear ratio is .5:1
GR6554      ; Slave gear ratio is .1:1 (actual gear ratio is
            ; .100006103516:1).
GR-65536    ; Slave ratio is 1:1 with direction reversal
```

Related Commands: EG

Command: aILn -- Set Integration Limit --

Argument: 0 <= n <= 16,383
Default: 0

This command clamps the level of influence that the PID digital filter integral term can use to reduce the static position error of servo axis 'a'. When properly adjusted, this can enhance loop stability and operation. The Integral Limit (IL) and Set Integral Gain (SI) must both be set to a non-zero value in order for the integral term to have any effect.

Related Commands: SI

Command: **aLFn -- Limit Switch Input Off --**

Argument: 0 <= n <= 3
Default: 0

This command disables one or more of the limit switch inputs for servo axis 'a'. The valid arguments to this command determine which inputs will be disabled and are as follows:

n	Limit Switch Inputs Disabled
0, 3 or no argument	Limit+ and Limit-
1	Limit+
2	Limit-

Related Commands: LM, LN

Command: **aLMn -- Limit Switch Input Mode --**

Argument: 0 <= n <= 3
Default: 0

This command is used to select how the LAC-26 will react when a limit switch is activated for servo axis 'a'. The valid arguments for this command are as follows:

n	Action
0	Turn servo off, continue commands
1	Stop abruptly, continue commands
2	Decelerate smoothly, continue commands
3	Interrupt only

In all cases, the Error flag in the status word will be set. This will prevent the LAC-26 from moving the servo until the flag is cleared by issuing the Motor On (MN) command. Before this command will have any effect, the limit switch must be enabled with the Limit Switch On command (LN).

Related Commands: LF, LN

Command: **aLNn -- Limit Switch Input On --**

Argument: 0 <= n <= 3

This command is used to enable one or both of the limit switch inputs for servo axis ‘a’. Once enabled, the servo will be stopped or turned off if a limit switch input goes active. At the same time the Limit Switch Tripped and Error Flags will be set in the status word. These flags will remain set until the servo is turned back on with the Motor On (MN) command. Once the servo is turned back on, it can be moved out of the limit switch region with any of the standard motion commands. The argument to this command determines which of the limit switch inputs will be enabled. The coding is as follows:

n	Limit Switch Inputs Enabled
0,3 or no argument	Limit+ and Limit-
1	Limit+
2	Limit-

Related Commands: LF, LM

Command: aOMn -- Output Mode --

Argument: 0 <= n <= 255
Default: 0

This command allows the user to determine what data gets sent to the D/A analog output for a given axis. The upper four bits of the argument are for redirection of data and **determine from which axis the D/A channel will get it's data**. This allows both D/A channels to output data from the same axis. If no redirection is specified, the default data used is that of the current axis.

Note: When outputting the servo output command, if no redirection is specified, then the output command as phase adjusted by the PH command will be output. If redirection is specified, then the normalized data as reported by the TQ command will be output.

'n'	Value Output
0	Servo Output Command
1	Servo Following Error
2	Servo Following Error * 64
3	Variable USER1
4	Low Bits of Encoder Position
5	Reserved
6	Reserved
7	Reserved

'n+'	Redirect Channel
0	Default
16	1
32	2

Examples:

```
1OM0        ; Send axis 1 servo output command to D/A channel 1.
1OM16       ; Send axis 1 servo output command to D/A channel 1.

2OM0        ; Send axis 2 servo output command to D/A channel 2.
2OM32       ; Send axis 2 servo output command to D/A channel 2.

1OM0        ; Send axis 1 servo output command to D/A channel 1.
2OM17       ; Send axis 1 following error to D/A channel 2.
```

Command: aOOn -- Output Offset --

Argument: -32767 <= n <= 32767
Default: 0

This command allows the user to set a continuous output for servo axis 'a'. In certain applications, such as an overhanging load, there will be a continuous burden placed upon a servo axis. In cases like these, where there is a predictable load, the OO command can be used to provide a continuous restoring force that will be combined with the output of the PID digital filter. This has the affect of improving the performance of the PID digital filter in that because it is not saturated with static load, it has a better dynamic response to load disturbances.

Related Commands: FA, FV, OM

Command: aPHn -- Set Servo Phasing --

Argument: 0 <= n <= 63
Default: 0

This command is used to set the output polarity, encoder phasing, Index input sense, Home input sense, Limit+ and Limit- input sense for servo axis 'a'. The polarity of the output will determine whether the servo is driven in a direction that reduces or increases position error. The encoder phase will determine whether the position count will increase or decrease for a valid encoder input sequence. The Index sense determines what logic edge will cause the Index input to be active. The Limit+, Limit- and Home sense determines whether these signals are active "on" or active "off.

To determine the argument to be used with the PH command, use the follow table and add the required values together.

	Add to 'n'
Output Phase Normal	0
Output Phase Reversed	1
Encoder Phase Normal	0
Encoder Phase Reversed	2
Index Active Level Low	0
Index Active Level High	4
Home Sense Active "ON"	0
Home Sense Active "OFF"	8
Limit+ Sense Active "ON"	0
Limit+ Sense Active "OFF"	16
Limit- Sense Active "ON"	0
Limit- Sense Active "OFF"	32

For example, if it were necessary to reverse the encoder phasing and to set the Limit+ and Limit- inputs to active "OFF", then 'n' would be (2 + 16 + 32) or 50. The default phasing and sense is equivalent to issuing this command with a argument of 0.

Command: aRIn -- Sampling Rate of Integral --

Argument: 0 <= n <= 127
Default: 0

This command allows for the adjustment of the PID digital filter integral sampling interval for servo axis 'a'. The period of this interval can be calculated by the following:

T = (n+1) * S * 0.000100

where "T" is the period in seconds, "n" is the RI command argument and "S" is the sample period count specified by the Servo Speed (SS) command. For example, if the value previously set by the SS command is 10 and the value set by the RI command is 1, then the integral sample period will be:

(1+1) * 10 * 0.000100 = .002000 S or 2 mS

This command is useful in tuning the PID servo loop to the inertial properties of the system.

Related Commands: FR

Command: aSAn -- Set Acceleration --

Argument: 0 <= n < 1,073,741,823
Default: 0

This command sets the acceleration rate for servo axis 'a'. The 32 bit argument to this command is scaled by 65536. This number determines how much the servo's velocity will be altered by each servo loop interval (determined by the Servo Speed "SS" command) while it is accelerating or decelerating. If this command is executed during a Position Mode move, it will be ignored.

Example:

Encoder: 500 lines or 2000 Counts/Rev
Desired Acceleration: 75 Rev/Sec2
Servo Loop Interval: 1,000 Hz

9830.4 = ((75 Rev/Sec * 2000 Counts/Rev) / 1000 Hz2) * 65536

To achieve an acceleration of 75 Rev/Sec2, the command SA9830 would be issued. A simpler way to calculate the acceleration argument would be to determine a constant for your application by which to calculate desired acceleration.

131.072 = K = ((1 Rev/Sec * 2000 Counts/Rev) / 1000 Hz2) * 65536

9830.4 = 75 Rev/Sec * K

Please note that if the Set Acceleration (SA) command is used with an argument of "0", then you have commanded the velocity to change in steps of zero which means if the servo is stopped it will not be able to move, and if the servo is moving it will not be able to change velocity.

Related Commands: SS, SV

Command: aSCn -- Set Current Mode Gain --

Argument: 0 <= n <= 32,767
Default: 0

This command sets the "current mode" gain for servo axis 'a' used by Torque Mode (QM1 only, see QM command). This allows the response of the current error integrator to be adjusted to suit a given application. A low SC value will provide slow response to load changes while a high SC value will provide quick response to load changes. If SC is set is too low then inadequate control may occur. If the SC setting is too high then the system may be unstable. A good "general" value for SC is about 8000.

Related Commands: QM, SQ

Command: aSDn -- Set Derivative Gain --

Argument: 0 <= n <= 32,767
Default: 0

This command sets the derivative gain term of the PID digital filter loop for servo axis 'a'.

Related Commands: SG, SI, IL

Command: aSGn -- Set Proportional Gain --

Argument: 0 <= n <= 32,767
Default: 0

This command sets the proportional gain term of the PID digital filter loop for servo axis 'a'.

Related Commands: SI, SD, IL

Command: aSIn -- Set Integral Gain --

Argument: 0 <= n <= 32,767
Default: 0

This command sets the integral gain term of the PID digital filter loop for servo axis 'a'. The Set Integral Gain (SI) and Integral Limit (IL) must both be set to a non-zero value in order for the integral term to have any effect.

Related Commands: SG, SD, IL

Command: aSQn -- Set [Maximum] Torque Level --

Argument: -32,767 <= n <= 32,767 in Torque Mode 0 (QM0)
-1,023 <= n <= 1,023 in Torque Mode 1 (QM1)
0 <= n <= 32,767 in Position Mode (PM)
0 <= n <= 32,767 in Velocity Mode (VM)
Default: 32,767

For servo axis 'a', this command sets the maximum output level when in Position Mode (PM) or Velocity Mode (VM) and sets the desired output level when in Torque Mode (QM). When this command is issued in Position Mode (PM) or Velocity Mode (VM), its limiting effect will remain, in all modes of operation, until again changed. Note that an argument less than zero is allowed only in Torque Mode (QM).

Examples:

```
>PM,SQ20000<CR>      ; Set maximum output to +-20000.

>VM,SQ10000<CR>      ; Set maximum output to +-10000.

>QM,SQ20000<CR>      ; Set output to forward 20000.

>QM,SQ-10000<CR>    ; Set output reverse 10000.

>PM,SQ10000<CR>      ; Set maximum output to +-10000.
>QM0,SQ15000<CR>    ; Set output to forward 15000 but
                   ; will only achieve output of 10000
                   ; due to previous command.
```

Related Commands: PM, QM, SC, VM

Command: SSn -- Set Servo Speed --

Argument: 1 <= n <= 255
Default: 2

This command sets the servo loop interval, which is the same for both axis' of the LAC-26. The period of this interval can be calculated by the following:

T = (n) * 0.000100

where "T" is the period in seconds and "n" is the SS command argument. For example, if the value set by the SS command is 10 then the servo loop period will be:

10 * 0.000100 = .001000 S or 1 mS

The minimum servo loop rate is 100 µS per enabled axis. The following table shows the minimum servo loop times:

# Axis Enabled	Minimum Loop Time
0	100µS
1	100µS
2	200µS

Using the SS command affects the Set Velocity (SV) and Set Acceleration (SA) commands in that they both are specified in terms of servo loop intervals. For example, if the servo loop rate is doubled, the velocity and acceleration would appear to have been halved. It should also be noted that the "SS" command will most likely affect the required settings of the PID digital filter.

Related Commands: SA, SV

Command: aSVn -- Set Maximum Velocity --

Argument: 0 <= n < 1,073,741,823
Default: 0

This command sets the desired velocity (in quadrature counts per servo loop interval) for servo axis 'a'. The 32 bit argument to this command consists of a 16 bit integer part and a 16 bit fractional part.

Example:

Encoder: 500 lines, 2000 Counts/Rev
Desired Velocity: 40 Rev/Sec
Servo Loop Interval: 1,000 Hz

5242880 = ((40 Rev/Sec * 2000 Counts/Rev) / 1000 Hz) * 65536

To achieve a velocity of 40 Rev/Sec, the command SV5242880 would be issued. A simpler way to calculate the velocity would be to determine a constant for your application by which to calculate desired velocity.

131072 = K = ((1 Rev/Sec * 2000 Counts/Rev) / 1000 Hz) * 65536

5242880 = 40 Rev/Sec * K

Related Commands: SA, SS

4.2 Reporting Commands

The reporting commands output data relevant to the operating status of the LAC-26. Numerical output will be in the mathematical base determined by the Decimal Mode (DM) / Hexadecimal Mode (HM) commands and output will be followed by a carriage return and linefeed.

Command: TAn -- Tell Analog to Digital Channel 'n' --

Argument: 0 <= n <= 9

This command will cause a conversion on A/D channel 'n' and will display the result.

Related Commands: GA

Command: aTB -- Tell Breakpoint --

Default: "None"

This command reports the last programmed breakpoint value specified by the Interrupt on Position Absolute (IP) or the Interrupt on Position Relative (IR) commands for servo axis 'a'. If no breakpoint value has yet been specified then the word "NONE is returned.

Related Commands: IP, IR

Command: TCn -- Tell State of I/O Channel 'n' --

Argument: 0 <= n <= 63
Default: "Off"

This command reports the current state of the I/O channel specified by 'n' and what type of logic the channel is (either active "on" or active "off"). The display is formatted as follows:

/xx = aaa

The presence of the "/" character indicates that a channel is active in the "off" state. A lack of the "/" character indicates that the channel is active in the "on" state. The "xx" indicates the channel number and the "aaa" indicates the channel's current state, either "ON" or "OFF". The following examples show all the possible combinations:

```
 01 = ON ; Channel 1 is "ON" in the active "ON” state. /01
= ON ; Channel 1 is "ON" in the active "OFF" state. 01 =
OFF ; Channel 1 is "OFF" in the active "ON" state. /01 =
OFF ; Channel 1 is "OFF" in the active "OFF" state.
```

Related Commands: CF, CH, CL, CN

Command: aTD -- Tell Derivative Gain --

This command reports the derivative gain value of the PID digital filter for servo axis 'a'.

Related Commands: TG, TI, TL

Command: TE -- Tell Error --

This command reports the last error code caused by any command or macro. If no error has occurred then a "0" will be reported. Once the TE command has been issued, then the error code will be reset to zero. The TE command is useful for determining what error occurred in the case that no display was connected at the time.

Command: aTF -- Tell Following Error --

This command reports the following error for servo axis 'a'. This value is the difference between the current desired temporal position (or that which is reported by the Tell Optimal (TO) command) of the trajectory generator and the servo's current real position (or that which is reported by the Tell Position (TP) command).

Related Commands: DB

Command: aTG -- Tell Proportional Gain --

This command reports the proportional gain value of the PID digital filter for servo axis 'a'.

Related Commands: TI, TD, TL

Command: aTI -- Tell Integral Gain --

This command reports the integral gain value of the PID digital filter for servo axis 'a'.

Related Commands: TG, TD, IL

Command: aTKn -- Tell (K) Constants --

Argument: 0 <= n <= 1

This command will display a number of internal settings depending on the value specified by 'n'. If 'n' is "0" or is not specified, then various parametric values for the servo axis specified by 'a' will be displayed. If 'n' is "1", then various system parameters will be displayed.

Example display for the command "1TK0":

```
>1TK0

Parameter Values for Axis [1]

Proportional Gain ----------(SG) = 0
Integral Gain --------------(SI) = 0
Derivative Gain ------------(SD) = 0
Integral Limit -------------(IL) = 0
Current Gain ---------------(SC) = 0
Velocity Feed-forward Gain - (FV) = 0
Accel. Feed-forward Gain --- (FA) = 0
Output Offset --------------(OO) = 0
Position Error Dead-Band --- (DB) = 0
Maximum Following Error ---- (SE) = 16383
Integral Sample Rate -------(RI) = 0
Derivative Sample Rate ----- (FR) = 0
Phase and Sense Settings --- (PH) = 0
Maximum Velocity -----------(SV) = 0
Acceleration ---------------(SA) = 0
Desired Direction ----------(DI) = 0
Torque (output) Limit ------(SQ) = 32767
Axis Type ------------------(OM) = 0
```

Example display for the command "TK1":

```
>TK1

System Parameter Settings (group 1).

Axis 1 Enabled ------------      (EA) = Yes
Axis 2 Enabled ------------      (EA) = Yes
Base 16 Input & Output --     (HM/DM) = Off
Character Echo ---------- (EN/EF) = On
Handshake --------------- (HN/HF) = Off
Fail -------------------- (FN/FF) = Off
Servo Loop Rate -----------      (SS) = 2
Input Debounce/Delay -------     (ID) = 0
Phase and Sense Settings --- (CV) = 0
Intr. Vector Enable, HIGH (EV/DV) = 0
Intr. Vector Enable, LOW (EV/DV) = 0
Firmware Revision ----------     (VE) = 3.30
```

Command: aTL -- Tell Integration Limit --

This command reports the integration limit value of the PID digital filter for servo axis 'a'.

Related Commands: TG, TI, TD

Command: TMn -- Tell Macros --

Argument: -2 <= n <= 255

The Tell Macro (TM) command will display the commands which make up any macros that have been defined. If 'n' >= 0 and 'n' <= 255, then the macro specified by 'n' be displayed. If 'n' = - 1, then all the macros will be displayed preceded by the individual macro number. If 'n' =-2, then all macros will be displayed and will be preceded by the letters "MD" and the individual macro number. This is useful when downloading programs from the LAC-26 to a computer in that you need not re-enter the Macro Define (MD) command and number at the beginning of each macro.

Related Commands: MD, RM

Command: aTO -- Tell Optimal Position --

This command reports the desired position for servo axis 'a'. This value may be different from the position reported by the TP command if the servo is moving or the controller is unable to drive the servo.

Related Commands: TT, TP

Command: aTP - Tell Real Position --

This command reports the absolute position of servo axis 'a'. It may be used to monitor motion during both the Motor On (MN) and the Motor Off (MF) states.

Related Commands: TT, TO

Command: aTQ -- Tell Torque --

his command reports the current output torque being commanded for servo axis 'a'. This value will be either that which is generated by the PID output or which is set by the user via the Torque Mode (QM).

Command: TRn -- Tell Contents of Register 'n' --

Argument: 0 <= n <= 511

This command reports the value contained by Register 'n'.

Related Commands: Register Commands

Command: aTS -- Tell Status Word --

This command reports the operating status word of servo axis 'a'. The response is coded into a single 32 bit value. The meaning of each bit is listed below:

Bit	Purpose
0	**Servo Enabled.** This bit will be set to "1" when the servo is enabled via the Motor On (MN) command. A "0" indicates that the servo has been disabled via one of the following reasons: a Motor Off (MF) command, a servo error due to excessive following error or over temperature condition, a limit event, an external fault.
1	**Servo Error.** A "1" in this bit position indicates that the maximum servo following error or over-temperature condition has occurred, or the external fault has been activated.
2	**Over Temperature.** A "1" in this bit position indicates that an over-temperature condition has occurred or the external fault input has been activated.
3	**Breakpoint Reached.** A "1" in this bit position indicates that a previously defined breakpoint has been reached. This bit will be reset by any of the following commands: Motor On (MN), Interrupt on Position (IR), Interrupt on Relative Position (IR).
4	**Trajectory Complete.** A "1" in this bit position indicates that the servo has completed a commanded move. A "0" indicates that the servo is busy executing a commanded move.
5	**Servo Stopping.** A "1" in this bit position indicated that the servo has been commanded to stop. Upon stopping, this bit is then cleared.
6	**Current Direction.** This bit indicated the current direction of travel for the servo. (0 = Positive, 1 = Negative)
7	**Desired Direction.** This bit indicates the direction commanded by the Direction (DI) command. (0 = Positive, 1 = Negative)
8	**Reserved.**
9	**Reserved.**
10	**Looking for Index.** A "1" in this bit position indicates that the IMCSA is currently watching for an index pulse to occur as commanded by the Find Index (FI) command.
11	**Looking for Edge.** A "1" in this bit position indicates that the IMCSA is currently watching for the Coarse Home input to go active as commanded by the Find Edge (FE) command.
12	**Reserved.**
13	**Coarse Home Input Active.** A "1" in this bit position indicates that the Coarse Home input is active.
14	**Capture Index Flag.** A "1" in this bit position indicates that the Find Index (FI) command is trying to capture the position on occurrence of an index pulse as opposed to initializing the position.
15	**Reserved.**
16	**Accelerating.?** A "1" in this bit position indicates that the servo is currently accelerating.
17	**Position Mode.** A "1" in this bit position indicates that the servo is currently operating in position mode.
18	**Velocity Mode.** A "1" in this bit position indicates that the servo is currently operating in velocity mode.
19	**Torque Mode.** A "1" in this bit position indicates that the servo is currently operating in (voltage) torque mode.
20	**Current Mode.** A "1" in this bit position indicates that the servo is currently operating in (current) torque mode).
21	**Reserved.**

22 **Electronic Gearing Mode.** A "1" in this bit position indicates that the servo is currently operating in electronic gearing mode.

23	**Reserved.**
24	**Limit Mode Abort.** Used by Limit Mode (LM) to determine what action to take upon the occurrence of a limit switch event.
25	**Limit Mode Stop.** Used by Limit Mode (LM) to determine what action to take upon the occurrence of a limit switch event.
26	**Limit- Tripped.** A “1” in this bit position indicates that a Limit- input event has occurred and has been acted upon as set by the Limit Mode (LM) command.
27	**Limit- Enabled.** A “1” in this bit position indicates that the Limit- input has been enabled for action via the Limit Mode (LM) command.
28	**Limit- Active.** A “1” in this bit position indicates that the Limit- input is currently active.
29	**Limit+ Tripped.** A “1” in this bit position indicates that a Limit+ input event has occurred and has been acted upon as set by the Limit Mode (LM) command.
30	**Limit+ Enabled.** A “1” in this bit position indicates that the Limit+ input has been enabled for action via the Limit Mode (LM) command.
31	**Limit+ Active.** A “1” in this bit position indicates that the Limit+ input is currently active.

Command: aTT -- Tell Target Position --

This command reports the current target position of servo axis 'a'. This is the absolute position to which the servo was last commanded to move. It may have been specified directly with the Move to Position (MP) or Move Absolute (MA) commands or indirectly with the Move Relative (MR) command. If the servo axis is in Velocity Mode (VM), then the target position will track the current optimal position (or that which is reported by the Tell Optimal (TO) command).

Related Commands: TO, TP

Command: aTV -- Tell Current Velocity --

This command reports the trajectory generator current velocity for servo axis 'a'. The value reported has the same units as the Set Velocity (SV) command. See the description of that command for further details.

Related Commands: SV, SS

Command: VE -- Tell Version --

This command reports the firmware revision level. This revision level exists as a code in the internal RAM memory (see Register Commands). This code should be interpreted as the first byte being the major revision number and second byte being the minor revision number. This allows user programs to determine the firmware revision for compatibility purposes.

Related Commands: Register Commands

4.3 Motion Commands

Motion Commands are those commands that involve or cause the actual movement of a servo. Any position mode (PM) based motion will be carried out using a trapezoidal velocity profile with the maximum velocity determined by the Set Velocity (SV) command. The acceleration and deceleration are the same and both are determined by the Set Acceleration (SA) command.

During the execution of a Position Mode based motion, **the target position (MA or MR) and the maximum velocity (SV) may be changed at any time, however, changes to the acceleration (SA) will be ignored.** If a velocity mode (VM) based motion is under way, the maximum velocity (SV) and the acceleration (SA) may be changed at any time. During this type of motion, the target position will continuously be set equal to the current optimal position. If a torque mode (QM) based motion is under way, the torque setting (SQ) may be changed at any time. During this type of motion, the target and optimal positions will continuously be set equal to the current real position.
In either PM, VM or QM, a Stop (ST) or Abort (AB) command will cause motion for the specified servo axis to cease. If a servo axis is in Position Mode (PM) and is commanded into Velocity Mode (VM), the motion will continue (only with no target destination) at the current maximum velocity set by the Set Velocity (SV) command. If a servo axis is in PM or VM and commanded into Torque Mode (QM), the servo output will be reduced to zero, and the unit will be placed in QM. If a servo axis is in VM and commanded into PM, the servo will be stopped at the current acceleration rate and placed into the PM mode. If a servo axis is in QM and commanded into VM, it will attempt to instantly assume the last known maximum velocity and remain in VM. If a servo axis is in QM and commanded into PM, the output will be reduced to zero, and the servo will begin station keeping in PM.

There is another mode of operation referred to as "Electronic Gearing Mode". This mode is a supplement to the position based modes of operation. Electronic gearing causes the servo axis for which it is in effect (or the slave axis), to move proportionally to the servo axis it is tracking (or the master axis) in conjunction with it's own motion profile. The proportion of the slave axis it set by the Gear Ratio (GR) command.

When the EG command is issued, the relative position of the master axis (from the time of the EG command), is monitored by the slave axis. This relative motion is multiplied by the value set with the GR command and the product of which is added to the optimal temporal position of the slave axis. The result of this is to allow the motion profile of the master servo axis to be superimposed on the slave axis' motion profile. While the EG mode is primarily intended for use with axis' that are operating in position mode (PM), it will also work with axis' that are operating in velocity mode (VM).

Command: aAB -- Abort Motion --

This command causes an emergency stop on servo axis 'a'. The servo stops abruptly but leaves the servo control loop enabled. The target position is changed to be equal to the present position. The command is used for stopping an undesired motion.

Related Commands: ST

Command: aCI -- Capture on Index --

This command causes the position counter of servo axis 'a' to be stored in the internal variable HREG upon receipt of an index pulse. For more information on HREG, see the Register Commands and the listing of the internal variables.

Related Commands: FI, WI

Command: aDA -- Enable Axis for Operation --

Default: Axis 1 and 2 enabled.

This command will disable operation of servo axis 'a'. Because no operation may be made on a disabled axis, the equivalent of a Motor Off (MF) command is executed before the axis is disabled.

Related Commands: EA

Command: aDHn -- Define Home --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command causes the current position of the servo axis 'a' to be defined as 'n'. From then on, all positions used and reported for the servo axis will be relative to that physical position.

Related Commands: FE, FI

Command: aDIn -- Set Direction --

Argument: n = 0 for positive, n = 1 for negative
Default: 0

This command sets the move direction for servo axis 'a' when in velocity mode. Issuing this command with an argument of "0" will cause the servo to move in a positive direction while an argument of "1" will cause it to move in a negative direction.

Related Commands: SA, SV, VM

Command: aEA -- Enable Axis for Operation --

Default: Axis 1 and 2 enabled.

This command will enable operation of servo axis 'a'. If an axis is not enabled for operation, no system resources will be expended on it. For example, this means that if an axis is going to be used for it's encoder interface only, it must still be enabled for operation via the EA command, however, the undedicated encoder interface is always enabled and does not apply here.

Before using the EA command, one should be sure that the Set Servo Speed (SS) command has been used such that the servo loop time is long enabled enough to service the newly axis as well as any previously enabled axis'.

Related Commands: DA

Command: aEGn -- Electronic Gearing Mode --

Argument: -2 <= n <= 2
Default: Position Mode

This command causes (slave) servo axis 'a' to begin to move proportionally to (master) servo axis 'n'. If 'n' is "0" then the electronic gearing mode for axis 'a' will be disabled. If 'n' is positive then the slave axis will track the master axis' optimal position (or that which is reported by the TO command). If 'n' is negative then the slave axis will track the master axis' real position (or that which is reported by the TP command). It is preferable that the EG command only be used when the target (or slave) axis is not in motion.

Related Commands: GR

Command: aFEn -- Find Edge of Coarse Home Input --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command is used to initialize servo axis 'a' at a given position. It will remain in effect until the Home input goes active. At that time, the current position of the servo will be defined as 'n'. This command will neither start nor stop any servo motion. It is up to the user to initiate servo motion before issuing the command and to stop any motion after the command is completed.

Related Commands: WE

Command: aFIn -- Find Edge of Index --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command is used to initialize servo axis 'a' at a given position. It will remain in effect until the Index input goes active. At that time, the current position of the servo will be defined as 'n'. Like the Find Edge (FE) command, this command will not start or stop any servo motions; that is up to the user. Since an index pulse may occur at numerous points of the servo's travel (once per revolution in rotary encoders), a typical application will require a home signal to establish a coarse position reference before the index pulse can be used to fine tune that reference.

Related Commands: CI, WI

Command: aGH -- Go Home --

This command causes servo axis 'a' to move to absolute position 0. This is equivalent to a Move Absolute (MA) command when the destination is 0. This command will initiate motion; therefore, a Go (GO) command is not required.

Related Commands: MA, GO

Command: aGO -- Go (start motion) --

This command causes servo axis 'a' to begin motion. When in the Velocity Mode (VM), the axis will accelerate to a constant velocity. When in the Position Mode (PM) the axis will begin to seek the specified target position. The servo must be in the "on" state for motion to occur.

Related Commands: MA, MR, PM, VM

Command: aMAn -- Move Target Absolute --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command sets the target position of servo axis 'a' to absolute position 'n'. The absolute position is relative to the point of initialization (home or 0). As the Move Absolute (MA) command will not initiate a motion, a Go (GO) command must be used. The servo's target position will be adjusted whether the servo is on or off.

Related Commands: GO, MR, PM

Command: aMF -- Motor Off --

This command is used to place servo axis 'a' in the "off" state. The servo's output will go to the null level. This command can be used to prevent unwanted motion or to allow manual positioning of the unit. When the servo is turned off, the target and the optimal positions will follow the real position.

Related Commands: MN

Command: aMN -- Motor On --

This command is used to place servo axis 'a' in the "on" state. When the servo is turned on, its target position is set to its current position so that the servo is not inclined to move. When this command is issued, it will disable the Home and Index interrupt sources and will reset the Error, Fault, Breakpoint reached, Looking for Edge, Looking for Index and Limit Tripped bits in the status word for axis 'a'.

Related Commands: LM, LN, MF

Command: aMRn -- Move Target Relative --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command generates a relative target position of 'n' counts for servo axis 'a'. Since the Move Relative (MR) command will not initiate a motion, a Go (GO) command must be used. The servo's target position will be adjusted whether the servo is on or off.

Related Commands: GO, MA, PM

Command: aPM -- Enter Position Mode --

Default: Position mode

This command causes servo axis 'a' to enter the position mode of operation. In this mode, the servo can be commanded to move to a specific target position. The moves will be executed using a trapezoidal velocity profile based upon parameters set by the Set Velocity (SV) and the Set Acceleration (SA) commands.

Related Commands: EG, QM, VM

Command: aQMn -- Enter Torque Mode --

Argument: 0 for voltage mode, 1 for current mode
Default: Position mode

This command places servo axis 'a' in the Torque Mode of operation. For mode 0 (or QM0), the output PWM duty cycle (or analog output for appropriate models) can be manually controlled by the user program. Once QM0 has been entered, the Set Torque (SQ) command can be used to set or change the servo output (see the SQ command). Bear in mind, that if the output was being limited by an SQ command before entering torque mode (while in PM or VM), it will remain limited while in QM and cannot be changed until the unit is in PM or VM.

For mode 1 (or QM1), the LAC-26 will use one of its A/D channels to monitor servo output current and attempt to maintain that output current at a steady level as commanded by the user program (see the SQ and SC commands). In that the A/D converter has 10-bit resolution, this will result in a value from 0 to 1023 which represents the instantaneous output current with 0 being approximately 0 Amps and 1023 being approximately 5 Amps with intermediate values being somewhat linear.

In the case of a motor (for example), when an output command of 512 is set (or about 2.5 amps), the PWM output begins to ramp up until the desired output current is reached. If the motor is not loaded enough to require 2.5 amps then the PWM will ramp up to 100% putting the maximum supply voltage to the motor. If at this point the motor is suddenly loaded, the PWM will ramp down to a level sufficient to supply 2.5 amps to the motor.

Mode QM1 will not function with the D/A module option.

Related Commands: EG, PM, VM

Command: aSEn -- Set Maximum Following Error --

Argument:	0 <= n <= 16,383
Default:	16,383

This command is used to set the maximum allowable following error (difference between the actual position and the optimal position) for servo axis 'a'. If the following error exceeds the programmed value, the servo will be turned off (except in the case of the related interrupt being enabled) and the Error bit in the status word for axis 'a' will be set. This bit will remain set until the servo is turned back on with the Motor On (MN) command.

Related Commands: DB, TF

Command: aST -- Stop Motion --

This command is used to terminate motion on servo axis 'a'. This command differs from the Abort (AB) command in that the servo will decelerate at its preset rate instead of stopping abruptly. When in Torque Mode, the outputs will be set to null.

Related Commands: AB, WS

Command: aVM -- Enter Velocity Mode --

Default: Position mode

This command places servo axis 'a' in the Velocity Mode of operation. In this mode, the servo can be commanded to move in either direction to a maximum velocity. The servo will move in that direction until commanded to stop. When using Velocity Mode, the user must specify the direction for the servo to move using the Direction (DI) command. After specifying the desired maximum velocity and the desired direction and placing the servo in velocity mode, the servo can be started by issuing the Go (GO) command. While the servo is moving in Velocity Mode, the user can change the velocity by issuing new Direction (DI) and/or Set Velocity (SV) commands. The acceleration rate at which the servo's velocity will change, is determined by the Set Acceleration (SA) command. The acceleration can also be changed at any time.

Related Commands: EG, QM, PM

4.4 Register Commands

The LAC-26 uses part of its nonvolatile RAM (NVRAM) to create a 512 by 32 bit general purpose register space with Register "0" being referred to as the Accumulator.

The registers have many uses including storing data and parameters, performing mathematical operations and controlling command execution. The registers can be manipulated by several commands and can also be used to replace the argument in most commands. For example, if register "6" contains the value "-12000" and the following command is used...

```
MA@6,GO
```

it would use the contents of register "6" as the argument thus giving the same result as if the following command was issued...

```
MA-12000
```

Note that the use of the "@" character is what caused the command to assume a register argument (or indirect argument) instead of a direct argument. If the value following the "@" is not in the range of the 512 registers (0 to 511), an error will be reported. If the value contained by the register of the indirect argument is out of range, an error will also be reported.

As stated earlier, because the registers are within the NVRAM, they are non-volatile and can be used as such. For example, if a register is incremented once every user program cycle, it can be used as an ongoing cycle counter for maintenance purposes, production accounting and etc.

4.4.1 Internal Variables

In certain applications, the user may find it necessary to use data pertinent to the internal operation of the LAC-26. This may be accomplished via the use of the Read Byte (RB), the Read Word (RW) and the Read Long (RL) commands which copy the LAC-26's internal RAM memory to the accumulator and the Write Byte (WB), the Write Word (WW) and the Write Long (WL) commands which copy the accumulator to the LAC-26's internal RAM memory. The listings below tell the location of and describe the internal variables that may be of use to the user.

WARNING: Randomly modifying these variables or other internal RAM not listed will most likely affect the operation of the LAC-26 resulting in unpredictable behavior or possible damage.

Status	Status word (TS command).
PV	Peak velocity (SV command).
MPV	Negative peak velocity (SV command).
V	Temporal velocity (TV command).
Desp	Desired position (TT command).
Carp	Temporal desired position (TO command).
Ack	Acceleration constant (SA command).
Curp	Current real position (TP command).
HREG	Holding register (CI,IP,IR commands).
IPPOS	Interrupt on position reference.
QI	Integral of current mode error (QM1 command).
PGAIN	Proportional term of PID filter (SG command).
IGAIN	Integral term of PID filter (SI command).
DGAIN	Derivative term of PID filter (SD command).
IL	Integral limit of PID filter (IL command).
CGAIN	Current mode gain (QM1 command).
FVGAIN	Velocity feed-forward of PID filter (FV command).
BIAS	Output offset (OO command).
THRO	Current servo output (TQ,SQ command).
QCMD	Current command (QM1,SQ command).
TLMTPL	Maximum positive servo output (SQ command).
FAGAIN	Acceleration feed-forward of PID filter (FA command).
PERR	Last calculated servo position error (TF command).
OERR	Old servo position error.
MAXERR	Maximum servo position error (SE command).
I	Servo error integral.
DERIV	Servo error derivative.
IMON	Servo output current monitor.
INTRVL	PID derivative sample interval (FR command).
SMPCNT	PID derivative sample counter.
IINTRVL	PID integral sample interval (RI command).
ISMPCNT	PID integral sample counter.
AXIS	Axis number lookup index.
ATYPE	Axis output type (OM command).
PHASE	Phase, sense and polarity information (PH command).
GMAxis	Gear mode "gear to" axis (EG command).
DBAND	Position error dead-band (DB command).
DERR	Position error with dead-band.
GMOFF	Gear mode initial position offset (EG command).
RATIO	Gear mode gear ratio (GR command).
USER1	User variable #1 (See OM command).
FCNT	Fault system counter.
FCMP	Fault system comparator.
TLMTMI	Maximum negative servo output (SQ command).
RecRate	Data recorder sample rate.
RecAddr	Data recorder sample address source.
RecSize	Data recorder sample data size.
AIN0	Analog Input 0. (After TA0,GA0 command).
AIN1	Analog Input 1. (After TA1,GA1 command).
AIN2	Analog Input 2. (After TA2,GA2 command).
AIN3	Analog Input 3. (After TA3,GA3 command).
AIN4	Analog Input 4. (After TA4,GA4 command).
AIN5	Analog Input 5. (After TA5,GA5 command).
AIN6	Analog Input 6. (After TA6,GA6 command).
AIN7	Analog Input 7. (After TA7,GA7 command).
AIN8	Analog Input 8. (After TA8,GA8 command).

AIN9	Analog Input 9. (After TA9,GA9 command).
VERSION	Version number for program use. (VE command).
LST_ERR	Last error code (TE command).
LTIMER	1 Hz LED timer.
ADSEMA	A/D Interrupt system semaphores flags.
SYSSTAT	System status word.
IPEND0	User interrupt pending level 0 -15 (EV,DV command).
IPEND1	User interrupt pending level 16 - 31 (EV,DV command).
SCLOCK	Servo loop counter.
RCLOCK	1 mS Real time clock/counter.
IO_DELAY	User digital input delay (ID command).

SYSSTAT Variable Bit Definitions

Bit	Purpose
0	**Axis 0 Enabled.** This bit is set to “1” when axis 0 is enabled.
1	**Axis 1 Enabled.** This bit is set to “1” when axis 1 is enabled.
2	**Reserved.**
3	**Reserved.**
4	**Reserved.**
5	**Pause.** This bit is set to “1” when the space bar is used to pause a user program.
6	**SXOff.** This bit is set to “1” when the LAC-26 receives an XOFF code.
7	**HexMod.** This bit is set to “1” by the HM command and set to “0” by the DM command.
8	**Echo.** This bit is set to “1” by the EN command and set to “0” by the EF command.
9	**HandSh.** This bit is set to “1” by the HN command and set to “0” by the HF command.
10	**Reserved.**
11	**Reserved.**
12	**Reserved.**
13	**Reserved.**
14	**Fail.** This bit is set to “1” by the FN command and set to “0” by the FF command.
15	**BadInp.** This bit is set to “1” by the VI command to indicate an error in user input.

Variable Name	Type	Axis #1 Address	Axis #2 Address
Status	LONG	01C0H/0448	0250H/0592
PV	LONG	01C6H/0454	0256H/0598
MPV	LONG	01CAH/0458	025AH/0602
V	LONG	01CEH/0462	025EH/0606
Desp	LONG	01E0H/0480	0270H/0624
Carp	LONG	01E6H/0486	0276H/0630
Ack	LONG	01EAH/0490	027AH/0634
Curp	LONG	01EEH/0494	027EH/0638
HREG	LONG	01F8H/0504	0288H/0648
IPPOS	LONG	01FCH/0508	028CH/0652
QI	LONG	0200H/0512	0290H/0656
PGAIN	WORD	0204H/0516	0294H/0660
IGAIN	WORD	0206H/0518	0296H/0662
DGAIN	WORD	0208H/0520	0298H/0664
IL	WORD	020AH/0522	029AH/0666
CGAIN	WORD	020CH/0524	029CH/0668
FVGAIN	WORD	020EH/0526	029EH/0670
BIAS	WORD	0210H/0528	02A0H/0672
THRO	WORD	0212H/0530	02A2H/0674
QCMD	WORD	0214H/0532	02A4H/0676
TLMTPL	WORD	0216H/0534	02A6H/0678
FAGAIN	WORD	0218H/0536	02A8H/0680
PERR	WORD	021AH/0538	02AAH/0682
OERR	WORD	021CH/0540	02ACH/0684
MAXERR	WORD	021EH/0542	02AEH/0686
I	WORD	0220H/0544	02B0H/0688
DERIV	WORD	0222H/0546	02B2H/0690
IMON	WORD	0224H/0548	02B4H/0692
INTRVL	BYTE	0226H/0550	02B6H/0694
SMPCNT	BYTE	0227H/0551	02B7H/0695
IINTRVL	BYTE	0228H/0552	02B8H/0696
ISMPCNT	BYTE	0229H/0553	02B9H/0697
AXIS	WORD	022AH/0554	02BAH/0698
ATYPE	BYTE	022CH/0556	02BCH/0700
PHASE	BYTE	022EH/0558	02BEH/0702
GMAxis	BYTE	022FH/0559	02BFH/0703
DBAND	WORD	0230H/0560	02C0H/0704
DERR	WORD	0232H/0562	02C2H/0706
GMOFF	LONG	0234H/0564	02C4H/0708
RATIO	LONG	0238H/0568	02C8H/0712
USER1	WORD	0240H/0576	02D0H/0720
FCNT	WORD	0242H/0578	02D2H/0722
FCMP	WORD	0244H/0580	02D4H/0724
TLMTMI	WORD	0246H/0582	02D6H/0726

System Variable Locations

Variable Name	Type	Address
RecRate	WORD	01A6H/422
RecAddr	WORD	01A8H/424
RecSize	WORD	01AAH/426
AIN0	WORD	0600H/1536
AIN1	WORD	0602H/1538
AIN2	WORD	0604H/1540
AIN3	WORD	0606H/1542
AIN4	WORD	0608H/1544
AIN5	WORD	060AH/1546
AIN6	WORD	060CH/1548
AIN7	WORD	060EH/1550

AIN8	WORD	0610H/1552
AIN9	WORD	0612H/1554
VERSION	WORD	0616H/1558
LST_ERR	BYTE	0619H/1561
LTIMER	WORD	061AH/1562
ADSEMA	WORD	061CH/1564
SYSSTAT	WORD	0712H/1810
IPEND0	WORD	0718H/1816
IPEND1	WORD	071AH/1818
SCLOCK	LONG	0722H/1826
RCLOCK	LONG	0726H/1830
IO_DELAY	BYTE	073EH/1854

Command: AAn -- Add 'n' to Accumulator (Signed) --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command adds the value 'n' to the accumulator.

Command: AC -- Accumulator Complement --

This command causes a (bit-wise) 1's complement of the accumulator.

Command: ADn -- Accumulator Divide by 'n' (Signed) --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command performs a 64 bit by 32 bit signed division with a 64 bit quotient and a 32 bit remainder. **The low order 32 bits of the numerator must be in the accumulator and the high order 32 bits must be in Register 1.** The divisor is specified by 'n'. The lower 32 bits of the quotient will be stored in the accumulator and the upper 32 bits will be stored in Register 1. The remainder will be in Register 2.

Related Commands: AM

Command: AEn -- Exclusive-Or Accumulator with 'n' --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command causes the result of an exclusive-OR of the accumulator with 'n' to reside in the accumulator.

Command: ALn -- Load Accumulator with 'n' (Signed) --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command causes the accumulator to be loaded with the value 'n'.

Command: AMn -- Multiply Accumulator by 'n' (Signed) --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command does a signed multiply of the 32 bit contents of the accumulator and 'n', **leaving the lower 32 bits of the 64 bit product in the accumulator and the upper 32 bits in Register 1.**

Related Commands: AD

Command: ANn -- And Accumulator with 'n' --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command causes the result of the accumulator ANDed with 'n' to reside in the accumulator.

Command: AOn -- Or Accumulator with 'n' --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command causes the result of the accumulator ORed with 'n' to reside in the accumulator.

Command: ARn -- Copy Accumulator to Register 'n' --

Argument: 0 <= n <= 511

This command causes the value in the accumulator to be copied to Register 'n'.

Related Commands: RA

Command: ASn -- Subtract 'n' from Accumulator (Signed)

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command causes the value 'n' to be subtracted from the accumulator.

Command: RAn -- Copy Register 'n' to Accumulator --

Argument: 0 <= n <= 511

This command causes the value in Register 'n' to be copied to the accumulator.

Related Commands: AR

Command: RBn -- Read Byte (8 bits) at RAM location 'n'

Argument: 0 <= n <= 2047

This command reads the 8 bit value at internal RAM location 'n' and copies it to the low 8-bits of the accumulator while clearing the upper 24 bits.

Related Commands: RW, RL, WB, WL, WW

Command: RLn -- Read Long (32 bits) at RAM location 'n' --

Argument: 0 <= n <= 2046

This command reads the 32 bit value at internal RAM location 'n' and copies it to the accumulator. The value specified by 'n' must be evenly divisible by 2.

Related Commands: RB, RW, WB, WL, WW

Command: RWn -- Read Word (16 bits) at RAM location 'n' --

Argument: 0 <= n <= 2046

This command reads the 16 bit value at internal RAM location 'n' and copies it to the low 16-bits of the accumulator while clearing the upper 16 bits. The value specified by 'n' must b e evenly divisible by 2.

Related Commands: RB, RL, WB, WL, WW

Command: SLn -- Shift Accumulator Left 'n' bits --

Argument: 0 <= n <= 31

This command causes the accumulator to be shifted left 'n' bits while filling the low order bits with zero.

Related Commands: SR

Command: SRn -- Shift Accumulator Right 'n' bits --

Argument: 0 <= n <= 31

This command causes the accumulator to be shifted right 'n' bits while filling the high order bits with zero.

Related Commands: SL

Command: TRn -- Tell Contents of Register 'n' --

Argument: 0 <= n <= 511

This command reports the value contained by Register 'n'.

Command: WBn -- Write Byte (8 bits) to RAM location 'n' --

Argument: 0 <= n <= 2047

This command copies the low 8-bits of the accumulator to the internal RAM location 'n'.

Related Commands: RB, RL, RW, WL, WW

Command: WLn -- Write Long (32 bits) to RAM location 'n' --

Argument: 0 <= n <= 2046

This command copies all 32 bits of the accumulator to the internal RAM location 'n'. The value of 'n' must be evenly divisible by two.

Related Commands: RB, RL, RW, WB, WW

Command: WWn -- Write Word (16 bits) to RAM location 'n' --

Argument: 0 <= n <= 2046

This command copies the low 16-bits of the accumulator to the internal RAM location 'n'. The value of 'n' must be evenly divisible by two.

Related Commands: RB, RL, RW, WB, WL

4.5 Sequence Commands

The LAC-26 includes commands that provide for conditional sequence command execution based on the register data, I/O state and etc. These Sequence Commands are illustrated by the following general forms:

- If the condition is true, command execution will continue normally. If the condition is false, the next two commands in the command line or the macro will be skipped.
- If the condition is true, command execution will continue normally. If the condition is false, the rest of the command line or the macro will be skipped.
- If the condition is true, command execution will continue normally. If the condition is false, command execution will be suspended until the condition becomes true.

Command: DFn -- Do if I/O Channel is "Off" --

Argument: 0 <= n <= 63

This command will cause command execution to continue if I/O channel 'n' is "Off"; otherwise, the rest of the command line or the macro will be skipped.

Related Commands: CF, CN, DN

Command: DNn -- Do if I/O Channel is "On" --

Argument: 0 <= n <= 63

This command will cause command execution to continue if I/O channel 'n' is "On"; otherwise, the rest of the command line or the macro will be skipped.

Related Commands: CF, CN, DF

Command: EP -- End Program --

This command will cause program execution to cease and return the ">" prompt.

Related Commands: RC

Command: IBn -- If Accumulator is Below 'n' (Signed) -

Argument: -2,147,483,647 <= n <= 2,147,483,647

If the contents of the accumulator is less than (signed comparison) the value 'n', command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Command: ICn -- If Bit 'n' of Accumulator is Clear (0) --

Argument: 0 <= n <= 31

If bit 'n' of the accumulator is clear (equals 0), command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Command: IEn -- If Accumulator Equal to 'n' --

Argument: -2,147,483,647 <= n <= 2,147,483,647

If the accumulator is equal to the value 'n', command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Related Commands: IU

Command: IFn -- If I/O Channel is "Off" --

Argument: 0 <= n <= 63

If the I/O channel specified by 'n' is in the "off" state, command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Related Commands: CF, CN, IN

Command: IGn -- If Accumulator is ">" 'n' (Signed) --

Argument: -2,147,483,647 <= n <= 2,147,483,647

If the contents of the accumulator is greater than (signed comparison) the value 'n', command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Command: INn -- If I/O Channel is "On" --

Argument: 0 <= n <= 63

If the I/O channel specified by 'n' is in the "on" state, command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Related Commands: CF, CN, IF

Command: aIPn -- Interrupt on Absolute Position 'n' --

Argument: -2,147,483,647 <= n <= ,147,483,647

This command is used to determine when servo axis 'a' has achieved the specified position 'n' referenced by the home position (or zero). When that position has been reached, the "breakpoint reached" flag in the status register will be set. This command can be issued before or after the servo has been commanded to move.

Related Commands: IR,TB

Command: aIRn -- Interrupt on Relative Position 'n' --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command is used to determine when servo axis 'a' has achieved the specified position 'n' referenced by the current position when this command is executed. When that position has been reached, the "breakpoint reached" flag in the status register will be set. This command can b e issued before or after the servo has been commanded to move.

Related Commands: IP, TB

Command: ISn -- If Bit 'n' of Accumulator is Set (1) --

Argument: 0 <= n <= 31

If bit 'n' of the accumulator is set (equals 1), command execution will continue; otherwise, the next two commands in the command line the or the macro will be skipped.

Related Commands: IC

Command: IUn -- If Accumulator Unequal to 'n' --

Argument: -2,147,483,647 <= n <= 2,147,483,647

If the accumulator is unequal to the value 'n', command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Related Commands: IE

Command: RPn -- Repeat --

Argument: 0 <= n <= 65,535

This command causes the command line to repeat 'n' more times. If 'n' is not specified or is "0", the command line is repeated indefinitely.

Command: WAn -- Wait ÔnÕ milliseconds --

Argument: 0 <= n <= 65,535

This command causes a wait period of 'n' milliseconds before going on to the next command.

Command: aWEn -- Wait for Edge -

Argument: 0 or 1

This command waits until the Coarse Home Input Active bit in the status word of servo axis 'a' is at the logic state specified by 'n' before continuing command execution. If 'n' is not specified or is 0, it will wait for the Home input to go inactive. If 'n' is 1, it will wait for the Home input to go active.

Note: The argument for this command works in reverse as compared to that of the IMCSA controller
.

Related Commands: FE

Command: WFn -- Wait for I/O Channel "Off" --

Argument: 0 <= n <= 63
Default: "Off"

This command waits until I/O channel 'n' is in the "off" state before continuing commandexecution.

Related Commands: CF, CN, WN

Command: aWI -- Wait for Index --

This command will wait for the occurrence of an Index input signal on servo axis 'a' before continuing to the next command. If a Find Index (FI) command has not been issued prior to this command, no waiting will occur.

Related Commands: CI, FI

Command: WNn -- Wait for I/O Channel "On" --

Argument: 0 <= n <= 63
Default: "Off"

This command waits until I/O channel 'n' is in the "on" state before continuing commandexecution.

Related Commands: CF, CN, WF

Command: aWPn -- Wait for Absolute Position 'n' --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command is used to determine when servo axis 'a' has achieved the specified position 'n' referenced by the home position (or zero). Until that position has been reached, commandexecution will be suspended. This command can be issued before or after the servo has been commanded to move; however, if the servo is not moving, then command execution may be suspended indefinitely.

Related Commands: WR

Command: aWRn -- Wait for Relative Position 'n' --

Argument: -2,147,483,647 <= n <= 2,147,483,647

This command is used to determine when servo axis 'a' has achieved the specified position 'n' referenced by the current position when this command is executed. Until that position has been reached, command execution will be suspended. This command can be issued before or after the servo has been commanded to move; however, if the servo is not moving, then commandexecution might be suspended indefinitely.

Related Commands: WP

Command: aWSn -- Wait for Stop --

Argument: 0 <= n <= 65535

This command will wait until servo axis 'a' has stopped moving for 'n' milliseconds before continuing to the next command. So that there is only one delay period, in the case where 'n' is specified as "0", the command will wait for all servo axis' to stop moving before the delay is executed.

Related Commands: ST

4.6 Learned Position Storage (LPS) Commands

The LAC-26 uses part of its nonvolatile RAM (NVRAM) to create a 256 item table for storing positions. **This table overlaps the last 256 registers of the register table.**

The LPS table is normally accessed by three commands: Learn Position (LP), Learn Target (LT) and Move to Position (MP). The purpose for the LPS table is to allow the user to store pre- determined positions for later use (such as in contouring) as the NVRAM will retain data even when powered down. In other cases, this table can be built through calculations using the register commands.

Command: aLPn -- Learn Current Position --

Argument: 0 <= n <= 255

This command causes the current position of servo axis 'a' (that position reported by the Tell Position (TP) command) to be stored in location 'n' of the Learned Position Storage table. This command is useful for "teaching" target positions to the LAC-26.

Related Commands: LT, MP

Command: aLTn -- Learn Target Position --

Argument: 0 <= n <= 255

This command causes the current target position of servo axis 'a' (that position reported by the Tell Target (TT) command) to be stored in the location 'n' of the Learned Position Storage table. This command is useful for setting up a table of target positions via a downloading procedure.

Related Commands: LP, MP

Command: aMPn -- Move to Index Table Position 'n' --

Argument: 0 <= n <= 255

This command causes the position contained by the Learned Storage Position table location 'n' to become the new target position. This command will not initiate a motion; therefore, a Go (GO) command may be required.

Related Commands: LP, LT

4.7 Macro Commands

Command instructions can be entered and executed immediately, but the LAC-26 also has the capability of using commands to form other commands called "macros". These macros are stored in the nonvolatile RAM (NVRAM) and can be executed automatically. Macros are created by stringing together one or more commands (with arguments), separated by commas, and indicating where they are to be stored. It should be noted that macros can use indirect as well as direct arguments. The LAC-26 allows for the creation of 256 macros consisting of a total of nearly 2,300 command instructions. Macro calls via the Macro Call (MC) command or the interrupt system, may be nested up to 25 calls deep.

An example of a macro might be...

```
>MD5,SV1000000,SA10000,MA25000,GO,WS100<CR>
```

Once this command line is entered, it will become the definition for macro 5. When macro 5 is used via the Macro Call (MC), Macro Sequence (MS) or Macro Jump (MJ) commands, the commands contained within the macro will be executed automatically.

There are a few necessary restrictions when creating macros. When using the Macro Define (MD) command, it must be placed first in the command line and it may be used only once. This also implies that it cannot be used as part of a macro. The Reset Macro (RM) command is similar in that it too cannot be used as part of a macro.

Another feature of the LAC-26 is the ability to automatically execute macro "0" on power- up or after a Reset (RT) command. If macro "0" is not defined, the user will receive the ">" prompt, and the LAC-26 will wait for manual input.

Command: DVn -- Disable Interrupt Vector 'n' --

Argument: 0 <= n <= 31

This command disables interrupt vector 'n'. This prevents any possibility of an interrupt being caused by that source.

Related Commands: EV, LV

Command: EVn -- Enable Interrupt Vector 'n' --

Argument: 0 <= n <= 31

This command enables interrupt vector 'n'. This allows the possibility of an interrupt being caused by that source.

Related Commands: DV, LV

Command: JPn -- Jump to Command, Absolute --

Argument: 0 <= n <= 31

This command causes execution of a macro to jump to the absolute command specified by 'nÕ. Commands are numbered sequentially starting from 0. If 'n' is specified whereas the command would attempt to jump past the end of the macro, command execution will resume as if the macro had ended.

Related Commands: JR

Command: JRn -- Jump to Command, Relative --

Argument: 0 <= n <= 31

This command causes execution of a macro to jump to the command specified by 'n' relative to the current command location. If 'n' is specified as zero, then this command will jump to itself. If 'n' is specified such that the command would attempt to jump past the end of the macro, command execution will resume as if the macro had ended. If 'n' is specified such that the command would attempt to jump to a point before the beginning of the macro, an error will be reported.

Related Commands: JP

Command: LVn -- Load Interrupt Vector 'n' --

Argument: 0 <= n <= 31

This command loads interrupt vector table entry 'n' with the contents of the low 8-bits of the accumulator. The following program fragment will cause execution of macro "10" upon the activation of digital input 0 (while a macro program is running).

```
MD1,AL10    ; Load accumulator with number for macro "10".
MD2,LV0     ; Load that number to interrupt vector 0.
MD3,EV0     ; Enable interrupt vector 0.

MD10,MG"Input 0 was just activated.",RC
```

When input 0 is activated, the line "Input 0 was just activated." will be displayed.

Related Commands: DV, EV

Command: MCn -- Macro Call --

Argument: 0 <= n <= 255

This command allows a previously defined macro specified by 'n' to be called like a subroutine. When this command is used, the current macro being executed is pushed to the macro stack and execution of macro 'n' begins. If macro 'n' has not been defined, then an error will b e reported. After execution of the defined macro, command execution will continue immediately after the Macro Call (MC) command. The Macro Call (MC) command can be used any place in a macro. Macro calls may be nested up to 25 times; however, it is NOT advisable for a macro to callitself.

Related Commands: RC, UM

Command: MDn -- Macro Definition --

Argument: 0 <= n <= 255

This command is used to define a new macro. Any duplication of numbers will simply result in the loss of the previously defined macro using that number and the loss of the memory that i t used. The Macro Define (MD) command must be the first command in the command line or an error will be reported.

Related Commands: RM, TM

Command: MJn -- Macro Jump --

Argument: 0 <= n <= 255

This command may be used to "Jump" to a previously defined macro command. Once the LAC-26 begins executing the new macro, it has no record of how it got there. This means that any commands that appear after the Macro Jump (MJ) command will not be executed. If there is no macro defined by the number 'n', an error will be reported. Once the end of the macro is encountered, macro execution stops (See the Macro Sequence (MS) command). The Macro Jump command can be used any place in a command string or macro command. It is also acceptablefor
a macro to jump to itself.

Related Commands: MC, MS

Command: MSn -- Begin Macro Sequence --

Argument: 0 <= n <= 255

This command will cause macros to be executed sequentially beginning with macro 'n' until an undefined macro or an End Program (EP) command is encountered. This command ca n be used anywhere in a command string or macro command. If the Macro Jump (MJ) or Macro Call (MC) commands are encountered, macro execution will still continue to execute sequentially.

Related Commands: MC, MJ

Command: RC -- Return from Macro Call --

When executed, this command will cause immediate return to the calling macro (assuming there was one).

Related Commands: MC, interrupts

Command: RMn -- Reset Macro(s) --

Argument: 0 <= n < 255

This command is used to delete one or more macros. If an argument is used, the macro specified by 'n' will be deleted and the memory it used will be lost. If no argument is used, all macros will be deleted, and all macro memory will be recovered. A Reset Macro command (RM) should be used before entering or downloading a new set of macro commands.

Related Commands: MD

Command: TMn -- Tell Macros --

Argument: -2 <= n <= 255

The Tell Macro (TM) command will display the commands which make up any macros that have been defined. If 'n' >= 0 and 'n' <= 255, then the macro specified by 'n' be displayed. If 'n' = -1, then all the macros will be displayed preceded by the individual macro number. If 'n' =-2, then all macros will be displayed and will be preceded by the letters "MD" and the individual macro number. This is useful when downloading programs from the LAC-26 to a computer in that you need not re-enter the Macro Define (MD) command and number at the beginning of each macro.

Related Commands: MD

Command: UMn -- Un-push Macro(s) --

Argument: 0 or 1

This command is used for controlling the macro subroutine call and return stack. If this command is used with no argument or an argument of "0" then one macro will be removed from the macro stack. If there is no macro to be removed then a MACRO_STACK_UNDERFLOW error will be reported. If the argument is "1" then the macro stack will be completely reset. The UM command is used in the event that a macro is paused due to a program interrupt and it is not desirable to return to that macro, or under some circumstances, a program may need to completely reset itself.

Related Commands: MC, interrupts

4.8 Input / Output (I/O) Commands

The user is able to manipulate the LAC-26's I/O channels via the use of several commands. These include setting or clearing outputs, reading inputs and altering the logic type of both.

Command: BIn -- Bulk Input from I/O Port 'n' --

Argument: 0 <= n <= 7

This command reads the value at the 8-bit digital input port and copies it to the low 8-bits of the accumulator with the lower channel being the lowest bit in the accumulator. The unused bits of the accumulator will be set to 0. The state of the inputs will be determined by the Channel High (CH) and Channel Low (CL) commands and the inputs will have been debounced if the Input Debounce (ID) command is in effect.

Related Commands: BO

Command: BOn -- Bulk Output to I/O Port 'n' --

Argument: 0 <= n <= 7

This command copies the low 8-bits of the accumulator to the digital output port. The state of the outputs will be determined by the Channel High (CH) and Channel Low (CL) commands. Unused bits in the accumulator are ignored.

Related Commands: BI

Command: CFn -- Turn I/O Channel 'n' "Off" --

Argument: 0 <= n <= 63
Default: "Off"

This command will cause I/O channel 'n' to assume the "off" state. The actual output state will depend on whether the channel is set active "on" (CH command) or active "off" (CL command).

Related Commands: CH, CL, CN

Command: CHn -- Make I/O Channel 'n' Active High --

Argument: 0 <= n <= 63
Default: Active high or active "on"

This command causes I/O channel 'n' to assume an active "on" mode.

Related Commands: CL

Command: CLn -- Make I/O Channel 'n' Active Low --

Argument: 0 <= n <= 63
Default: Active high or active "on"

This command causes I/O channel 'n' to assume an active "off" mode.

Related Commands: CH

Command: CNn -- Turn I/O Channel 'n' "On" --

Argument: 0 <= n <= 63
Default: "Off"

This command will cause I/O channel 'n' to assume the "on" state. The actual output state will depend on whether the channel is set active "on" or active "off".

Related Commands: CF, CH, CL

Command: IDn -- Input Debounce 'n' milliseconds --

Argument: 0 <= n <= 7
Default: 0

This command determines the length of debounce time, if any, that is applied to the digital inputs. The digital inputs are sampled once every millisecond. What this means is that an input must remain in a given state for 'n' samples before it is considered valid. If the argument 'n' is "0", then no input debouncing in performed. This debouncing applies to the following commands: BI, DF, DN, IF, IN, WF, WN.

4.9 Future Expansion Interface

For future expansion, the LAC-26 provides a proprietary high speed serial data bus. Such uses might include various external devices such as thumb-wheel switches, LED or LCD displays, switch panels and additional number or types of input and output. This provides custom expansion flexibility in certain qualified OEM applications.

4.10 Serial Communications and Miscellaneous Commands

These commands control operation of the serial communication's interface and cover the balance of the LAC-26 functions not fitting in the other categories.

Command: BK -- Break --

This command will cause the rest of the command line or macro to be skipped. This command is used along with the Sequence Commands for conditional command execution.

Command: BRn -- Set Baud Rate --

Argument:	300, 600, 1200, 2400, 4800, 9600 or 19,200
Default:	**9600**

This command allows the user to change the baud rate at which the serial communication's interface operates. Once this command has been issued with a valid argument, the communication's interface will then immediately operate at the specified baud rate. This baud rate will remain in place, even after power cycling the unit, until it is changed again. **Please note that the default baud rate is 9600.**

Command: CDn -- Capture Data --

Argument: 0 <= n <= 16383

This command allows for the capture and recording of data from an internal variable with the user being able to later download and plot this data for analysis. The number of samples that can be recorded depends on the amount of macro memory available at the time that the Capture Storage (CS) command is issued.

The argument determines the number of samples to record. If an argument is used that is greater than the allocated storage space, then the argument will be truncated to fit that space. All data sizes are extended to 32 bits before they are stored.

Before using the CD command, a few things must be taken care of. The first of these is t o define an area to store the data. The Capture Storage (CS) command is used for this in that i t allocates a storage area in the macro memory to where the data can be recorded. The next thing to do is to specify the location and size of the data to be recorded. The internal 16-bit variable ÒRecAddrÓ (see Register commands) is used to specify the location (or address) of the variable to be recorded. The internal 16-bit variable ÒRecSizeÓ is used to specify the data size. If bit 0 of ÒRecSizeÓ is equal to Ò1Ó then 8 bit data samples are recorded. If bit 1 of ÒRecSizeÓ is equal to Ò1Óthen 16 bit data samples are recorded. If bits 0 and 1 of ÒRecSizeÓ are both Ò0Ó then 32 data samples are recorded. The last thing that needs to be specified is the sample rate at which the data is captured. This is set by writing the appropriate value to the 16-bit variable ÒRecRateÓ. The sample rate will be whatever the servo sample rate is (see SS command) divided by ÒRecRateÓ.

RecSize	Recorded Variable Size
1	8 Bits
2	16 Bits
0	32 Bits

After using the CD command, the Dump Data (DD) command can be used to display the data that was captured.

Example:

To make 2000 records of the actual position of axis 2 you would issue the following commands:

```
CS2000            ; Reserve space for 2000 samples.
;
;
;
AL4,WW422         ; Servo rate = 1mS so Data rate = 4mS.
AL638,WW424       ; Set address of axis 2 Curp.
AL0,WW426         ; Set for 32-bit variable size.
```

The CS command should be used only once. After that, the CD and DD commands can be used repeatedly.

Related Commands: CS, DD

Command: CSn -- Capture Storage --

Argument: 0 <= n <= 16383

This command is used to allocate storage space for the data recorder commands. Th e argument determines the maximum number of samples that can be recorded. If an argument is used that is greater than the macro memory available, then an error will be reported. Because information pertaining to this command is stored in non-volatile memory, once this command is issued, the storage space will remain, even after power-cycling. To recover the macro memory used by this command, use the Reset Macros (RM) command.

Related Commands: CD, DD

Command: DDn -- Dump Data --

Argument: 0 <= n <= 16383

This command is used to dump data that has been previously recorded by the Capture Data (CD) command to the display. The argument determines the number of recorded samples to display. If an argument is used that is greater than the allocated storage space, then the argument will be truncated to fit that space.

Related Commands: CD, CS

Command: DM -- Decimal Mode --

Default: Decimal Mode

This command causes all numerical input and output to be interpreted as decimal or base 10. Numbers will be output with a leading "-" if they are less than zero.

Related Commands: HM

Command: EF -- Echo Off --

Default: Echo On

This command suppresses the echoing of characters received by the serial communication's interface. It is normally used in the half-duplex mode of operation in serial communications.

Related Commands: EN

Command: EN -- Echo On --

Default: Echo On

This command causes characters received from the serial communication's interface to b e echoed as they are received. It is normally used in the full-duplex mode of operation in serial communications.

Related Commands: EF

Command: GAn -- Get A/D Channel

Argument: 0 <= n <= 9

This command will cause a conversion on A/D channel 'n' and will store the result in the bottom 10 bits of the accumulator. Unused bits will be set to 0.

Related Commands: TA

Command: HF -- Hardware Handshaking Off --

Default: Hardware handshake off

This command does not function and is retained for backward compatibility reasons.

Command: HM -- Hexadecimal Mode --

Default: Decimal Mode

This command causes all numerical input and output to be interpreted as hexadecimal or base 16. Numbers will be output as 2, 4 or 8 digits with leading 0's if they are positive and leadingF's if they are negative.

Related Commands: DM

Command: HN -- Hardware Handshaking On --

Default: Hardware handshake off

This command does not function and is retained for backward compatibility purposes.

Command: MG[[""][:n][:N]] -- Display Message --

Argument: 0 <= n <= 511

This command allows for the display of an optional text string and / or an optional register variable with the additional option of inhibiting the carriage return / linefeed (CRLF) at the end.

The text string must be enclosed by quotes """ and may be up to 127 characters long. Additional parameters must be separated by a colon ":" .

The following are all the valid examples. Note that the "N" option inhibits a CRLF. Parameters must be given in specific order.

```
>MG                           ; Display CRLF only.

>MG"THE BLACK BEAR IS "       ; Display "THE BLACK BEAR IS " followed by;
CRLF.

>MG0                          ; Display contents of register 0 followed
                              ; by CRLF.

>MGN                          ; This command displays nothing followed
                              ; by nothing.

>MG"THE BLACK BEAR IS ":0     ; Display "THE BLACK BEAR IS " followed by
                              ; the contents of register 0 followed by
                              ; CRLF.

>MG"THE BLACK BEAR IS ":N     ; Display "THE BLACK BEAR IS " followed by
                              ; nothing.

>MG0:N                        ; Display the contents of register 0
                              ; followed by nothing.

>MG"THE BLACK BEAR IS ":0:N ; Display "THE BLACK BEAR IS " followed by
                              ; the contents of register 0 followed by
                              ; nothing.
```

Command: NO -- No Operation --

This command does nothing. It can be used to cause short delays in command lineexecutions or to fill out commands (see Sequence Commands).

Command: RT -- Reset --

This command performs a complete restart of the LAC-26, including restoration of al l default conditions, such as acceleration and velocity, and leaves all servo axis' in the "off" state.

Command: **VI[[""][:n][:N]] -- Variable Input --**

Argument: 0 <= n <= 255

This command allows for the display of an optional text string, an optional carriage return / linefeed (CRLF) and the entry of integer numeric operator input to a register variable.

The text string must be enclosed by quotes """ and may be up to 127 characters long. Additional parameters must be separated by a colon ":". Entered numbers will be interpreted (as decimal or hexadecimal) as determined by the current mode set by the DM or HM commands. If the value entered by the operator is in error (indeterminate), then the Bad Input bit (bit 15 of the variable SYSSTAT) is set, otherwise it is cleared. If no operator input is entered (only a carriage return is received), then no register will be altered. If no argument is given and operator input is received,
then register 0 will be the default recipient of the input value.

The following are all the valid examples. Note that the "N" option causes a CRLF. Parameters must be given in specific order.

```
>VI                    ; Wait for operator input (if any) to register 0.

>VI"ENTER NUMBER: "    ; Display "ENTER NUMBER: " and wait for
                       ; operator input (if any) to register 0.

>VI5                   ; Wait for operator input (if any) to
                       ; register 5.

>VIN                   ; Display a CRLF and wait for operator input
                       ; (if any) to register 0.

>VI"ENTER NUMBER: ":5  ; Display "ENTER NUMBER: " and wait for
                       ; operator input (if any) to register 5.

>VI"ENTER NUMBER: ":N  ; Display "ENTER NUMBER: " followed by a
                       ; CRLF and wait for operator input (if any)
                       ; to register 0.

>VI5:N                 ; Display a CRLF and wait for operator input
                       ; (if any) to register 5.

>VI"ENTER NUMBER: ":5:N ; Display "ENTER NUMBER: " followed by a CRLF
                       ; and wait for operator input (if any) to
                       ; register 5.
```

Command: **XFn -- Set XOFF Code --**

Argument: 0 <= n <= 255
Default: 19

This command does not function and is retained for backward compatibility purposes.

Command: XNn -- Set XON Code --

Argument: 0 <= n <= 255
Default: 17

This command does not function and is retained for backward compatibility purposes.

Command: ZF -- Format NVRAM --

Argument: n = 123

This command re-formats and re-initializes the LAC-26's non-volatile static ram (NVRAM). This means that any macros will be deleted and their space recovered, register contents will be set to 0 and the baud rate (among other things) will be set to their default values. Evidence of the baud rate change will not be immediate but will occur when the unit is either power cycled or the Reset (RT) command is issued. This command is intended for use by the manufacturer to place the unitin a known state after testing but is sometimes useful as a "when all else fails" means under certain circumstances. This command must include an argument "key code" of "123" in order to function. This is to reduce the possibility of accidental execution.

Command: ZZ -- Dump Memory --

Argument: 0 <= n <= $3FFFF

This command does a display dump of the memory starting at address 'n'. This command is intended for use by the manufacturer for diagnosing purposes.

---

5. Appendix A, LAC-26 Error Code Definitions

1 - ARGUMENT ERROR.

This error indicates that a command argument was not given or was specified out of the permissible numerical range.

2 - INVALID COMMAND.

This error indicates that an invalid or unrecognized command was specified in the command line.

3 - INVALID MACRO COMMAND.

This error indicates that an invalid or unrecognized command was used in defining a macro.

4 - MACRO ARGUMENT ERROR.

This error indicates that a command argument was not given or was specified out of the permissible numerical range in defining a macro.

5 - MACRO NOT DEFINED

This error indicates that execution of an undefined macro was attempted.

6 - MACRO OUT OF RANGE.

The LAC-26 allows for a maximum of 256 macros (numbered 0 to 255). This error indicates that an attempt was made to access a macro out of this boundary.

7 - OUT OF MACRO SPACE.

The LAC-26 allows for a maximum of 256 macros with up to 256 commands per macro a n d approximately 15800 bytes of macro storage space. Each macro command requires 6 bytes o f macro storage memory and each macro requires an overhead of 1 byte. This error indicates that so many macros/macro commands have been defined that there is no remaining memory to define more.

8 - CAN'T DEFINE MACRO IN A MACRO.

This error indicates that an attempt was made to define a macro from another macro that is currently being executed. This is not allowed on the LAC-26 as to prevent loss of program control due to possible "nesting".

9 - CAN'T DEFINE MACRO WHILE SERVO ENABLED.

This error indicates that attempt was made to define a macro while a servo axis the was enabled (i.e.: "MN"). This is not allowed on the LAC-26 as to prevent loss of program and/or servo control due to macro memory space definition contention.

10 - MACRO JUMP ERROR.

This command indicates that an attempt was made to jump ("MJ" command) to a command within a macro that does not exist.

11 - OUT OF MACRO STACK SPACE.

When a macro is called by another macro (via the "MC" command or a macro interrupt), the return macro and macro command number must be saved along with other internal variables. This error indicates that
the memory space set aside for this purpose has been exhausted and no more "calls" may b e attempted. The LAC-26 is capable of macro calls nested 25 deep.

12 - MACRO MUST BE FIRST COMMAND.

When defining a macro, the "MD" command must be the first command in the commandline. This error indicates that this requirement was not met.

13 - STRING ERROR

When using a MG or VI command, no closing quote was encountered.

14 - MACRO STRING ERROR

When using a MG or VI command in defining a macro, no closing quote was encountered.

15 - SYNTAX ERROR

Indicates the improper usage of the MG or VI commands.

16 - MACRO SYNTAX ERROR

Indicates the improper usage of the MG or VI commands while defining a macro.

17 - AXIS RANGE ERROR

The axis specified is out the possible numerical range.

18 - INTERRUPT MACRO NOT DEFINED

During the course of interrupt processing, at attempt was made to go to an undefined macro.

19 - INTERRUPT MACRO STACK ERROR

Indicates that the macro stack has run out of space during interrupt processing.

20 - MACRO STACK OVERFLOW

The macro stack has run out of space.

21 - MACRO STACK UNDERFLOW

An attempt was made to "pop" a macro off of the macro stack when there was no macro t pop.

5. AMC driver setup

5.1 Communication Protocol

DZ digital drives offer networking capability through either CANopen or RS-485 communication: CANopen is used with DZC drives, and RS-485 is used with DZR drives. Both DZC and DZR drives include an auxiliary RS- 232 serial port used for configuring the drive through DriveWare. DZR drives can also connect to DriveWare through RS-485.

5.1.1 CANopen

CANopen is an open standard embedded machine control protocol that operates through the CAN communication interface on DZC digital drives. The CANopen protocol is developed for the CAN physical layer. The CAN interface for ADVANCED Motion Controls DZ drives follows the CiA (CAN in Automation) DS301 communications profile and the CiA DSP402 device profile. CiA is the non-profit organization that governs the CANopen standard. More information can be found at www.can-cia.org.
On DZC drives, a CAN interface is provided through a transmit pin and a receive pin. A user - supplied external transceiver which meets a CAN physical layer standard (ex. ISO 11898 -2) is required for CAN communication. This transceiver acts as a medium between chip-level CAN signals and bus-level CAN signals. When choosing a transceiver, make sure it matches with the physical layer standard of the CAN bus being used. It is also recommended to isolate the transceiver from the DZ drive. See "CAN Transceiver" on page 31 for more information on interfacing with a CAN transceiver.

CAN communication works by exchanging messages between a CANopen "host" and CANopen "nodes". The messages contain information on specific drive functions, each of which is defined by a group of objects. An object is roughly equivalent to a memory location that holds a certain value. The values stored in the drive's objects are used to perform the drive functions (current loop, velocity loop, position loop, I/O functions, etc.). For more detailed information on CANopen communication with DZC drives and a complete list of CAN objects, consult the ADVANCED Motion Controls CANopen Communication Manual, available for download at www.a-m -c.com.

5.1.2 RS-485 Communication

ADVANCED Motion Controls' proprietary serial protocol is a byte-based, binary, master-slave standard to access drive "commands" used on DZR drives. The drive commands provide read or write access to drive parameters, with each command containing one or more parameters. Each command is assigned a unique index number, and parameters within a command are given offset values. As a result, parameters are referenced using a combination of the command index and parameter offset values. The serial protocol utilizes variable length commands to access one or more parameters within an index. On DZR drives, the RS-485 interface is provided through a transmit pin and a receive pin. These pins should be connected to the appropriate locations on a serial cable connector, as specified by the serial protocol. The reference point for the RS-485 signals is common with the signal ground of the drive. See "RS-485/232 Interface" on page 32 for more information on the DZ drive serial interface connection. For more detailed information on RS-485 communication with DZ drives, consult the ADVANCED Motion Controls Serial Communication Manual.

5.2 Control Modes

The DZ family of digital drives operate in a variety of control modes. The setup and configuration parameters for these modes are commissioned through DriveWare. See the DriveWare Software Guide for mode configuration information.

The name of the mode refers to which servo loop is being closed in the drive, not the end-result of the application. For instance, a drive operating in Current (Torque) Mode may be used for a positioning application if the external controller is closing the position loop. Oftentimes, mode selection will be dependent on the requirements and capabilities of the controller being used with the drive as well as the end-result application.

5.2.1 Current (Torque) or Profile Current (Torque)

In these modes, the input command voltage controls the output current. The drive will adjust the output duty cycle to maintain the commanded output current. Current modes are used to control torque for rotary motors (force for linear motors), but the motor speed is not controlled. The output current and other parameters can be monitored in DriveWare through the digital oscilloscope function. DriveWare also offers configuration of maximum and continuous current limit values.

While in Current (Torque) Mode, the drive will maintain a commanded torque output to the motor based on the input reference command. Sudden changes in the motor load may cause the drive to output a high torque command with little load resistance, causing the motor to spin

Note rapidly. Therefore, Current (Torque) Mode is recommended for applications using a digital position controller to maintain system stability.

5.2.2 Velocity or Profile Velocity

In these modes, the input command voltage controls the motor velocity. Velocity modes require the use of a feedback element to provide information to the drive about the motor velocity. DZ drives allow velocity control with either Hall Sensors or an Encoder as the feedback element. The motor velocity and other parameters can be monitored in DriveWare through the digital oscilloscope function. The feedback element being used for velocity control must be specified in DriveWare, which also offers configuration of velocity limits. See “Feedback Supported” on page 8 for more information on feedback devices.

5.2.3 Position or Profile Position

In these modes, the input command voltage controls the actual motor position. Position modes require the use of a feedback element to provide information to the drive about the physical motor location. DZ drives allow position control with either an Encoder or ±10V Position feedback. The motor position and other parameters can be monitored in DriveWare through the digital oscilloscope function. The feedback element being used for position control must be specified in DriveWare, which also offers configuration of position limits. See “Feedback Supported” on page 8 for more information on feedback devices.

5.2.4 Cyclic Synchronous Modes

Cyclic Synchronous Modes (DZC drive models only) give responsibility of trajectory control to the host. The drive interpolates between command points, defining the rate by dividing the change in command by the interpolation time period. This allows the drive to respond smoothly to each step in command.

Cyclic Synchronous Current In Cyclic Synchronous Current Mode, the drive closes the current loop. The host is allowed more control by having the ability to instantly add current feedforward values. This allows for gain compensation in applications with varying loads.

Cyclic Synchronous Velocity In Cyclic Synchronous Velocity Mode, the drive closes two control loops: velocity and current. The host is allowed more control by having the ability to instantly add velocity and current feedforward values. This allows for gain compensation in applications with varying loads.

Cyclic Synchronous Position In Cyclic Synchronous Position Mode, the drive closes three control loops: position, velocity, and current. The host can send target position, velocity feedforward, and current feedforward values to the drive. This allows for gain compensation in applications with varying loads.

5.2.5 Feedback Supported

There are a number of different feedback options available in the DZ family of digital drives. The feedback element must be capable of generating a voltage signal proportional to current, velocity, position, or any parameter of interest. Such signals can be provided directly by a potentiometer or indirectly by other feedback devices such as Hall Sensors or Encoders. These latter devices must have their signals converted to a DC voltage, a task performed by the DZ drive circuitry and configuration software.

5.2.6 Feedback Polarity

The feedback element must be connected for negative feedback. This will cause a difference between the command signal and the feedback signal, called the error signal. The drive compares the feedback signal to the command signal to produce the required output to the load by continually reducing the error signal to zero. For DZ drives, this becomes important when using “Encoder Feedback” and “Hall Sensors”, as connecting these feedback elements for positive feedback will lead to a motor "run- away" condition. In a case where the feedback lines are connected to the drive with the wrong polarity, the drive will attempt to correct the "error signal" by applying more command to the motor. With the wrong feedback polarity, this will result in a positive feedback run- away condition. To correct this, either change the order that the feedback lines are connected to the drive, or use DriveWare to reverse the internal velocity feedback polarity setting. The AutoCommutation routine in DriveWare will typically determine the proper feedback polarity setting.

5.2.7 Hall Sensors

DZ drives can use single-ended Hall Sensors for commutation and/or velocity control. The Hall Sensors (typically three) are built into the motor to detect the position of the rotor magnetic field. With Hall Sensors being used as the feedback element, the input command voltage controls the motor velocity, with the Hall Sensor frequency closing the velocity loop. The Hall Sensor frequency is converted into velocity feedback that the drive uses to control the motor speed and direction. The actual motor speed can be monitored in DriveWare through the digital oscilloscope function.

Due to the inherent low resolution of motor mounted Hall Sensors, using Hall Sensors for velocity feedback is not recommended for low-speed applications below 300 rpm for a 6-pole motor, 600 rpm for a 4-pole motor, or 900 rpm for a 2-pole motor. Hall Velocity Mode is better suited

Note for velocity control applications where the motor will be spinning at higher speeds.

Yes youFor more information on using Hall Sensors for trapezoidal commutation, see “Trapezoidal Commutation” on page 74.

Note
Users designing their own PCB interface may also design the appropriate circuitry on their PCB interface to allow differential inputs

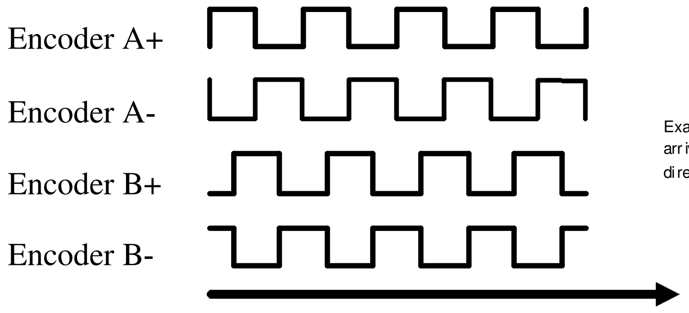
5.2.8 Encoder Feedback

DZ drives can utilize differential encoder inputs for velocity or position control, with the option of also using the encoder to commutate the motor. The encoder provides incremental position feedback that can be extrapolated into very precise velocity or position information. With an Encoder being used as the feedback element, the input command controls the motor velocity or motor position, with the frequency of the encoder signal closing either the velocity or position loop. The encoder signals are read as "pulses" that the DZ drive uses to essentially keep track of the motor’s speed, position and direction of rotation. Based on the speed and order in which these pulses are received from the encoder, the drive can interpret the motor velocity and physical location. The actual motor speed and physical location can be monitored in DriveWare through the digital oscilloscope function. Both the "A" and "B" channels of the encoder are required for operation. DZ drives also accept an optional differential "index" channel that can be used for absolute position reference.

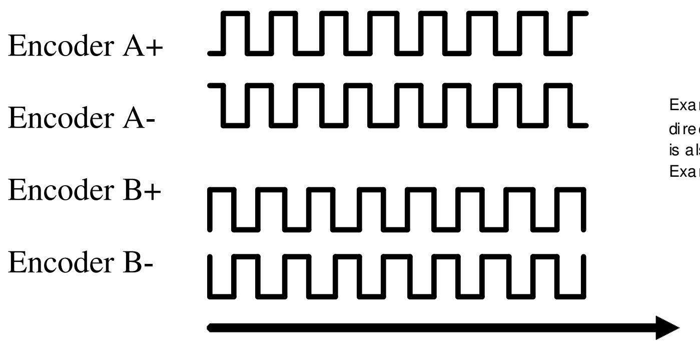
The MC1XDZ_02 mounting cards provide the option of using single-ended encoder inputs. Users designing their own PCB interface may also design the appropriate circuitry on their PCB interface to allow single-ended inputs.

Figure 2.2 below represents differential encoder "pulse" signals, showing how dependent on which signal is read first and at what frequency the "pulses" arrive, the speed and direction of the motor shaft can be extrapolated. By keeping track of the number of encoder "pulses" with

FIGURE 2.2 Encoder Feedback Signals



Example 1: Encoder-A precedes Encoder-B. The pulses arrive at a certain frequency, providing speed and directional information to the drive.



Example 2: Encoder-B precedes Encoder-A, meaning the direction is opposite from Example 1. The signal frequency is also higher, meaning the speed is greater than in Example 1.

DZ drives can also use encoder feedback for sinusoidal commutation by using the AutoCommutation routine in DriveWare. Encoder feedback is also used in the "Phase Detect" procedure in DriveWare, which is necessary when using a three phase (brushless) motor without Hall Sensors. The Phase Detect routine will have to be run before AutoCommutation. Phase Detect works by sending a small current signal to the motor, prompting the motor to vibrate slightly for a few seconds. The encoder feedback from this movement provides a starting position for the motor, allowing the drive to then be properly commutated. See the Driveware Software Guide for more information on Phase Detect.

The high resolution of motor mounted encoders allows for excellent velocity and position control and smooth motion at all speeds. Encoder feedback should be used for applications requiring precise and accurate velocity and position control, and is especially useful in

Note applications where low-speed smoothness is the objective.

5.2.8 ±10 VDC Position

DZ drives accept an analog ±10 VDC Position feedback, typically in the form of a load-mounted potentiometer. The feedback signal must be conditioned so that the voltage does not exceed ±10 V, and is connected to the DZ drive through the Programmable Analog Input. In

DriveWare, the connection method that is used must be selected under the Position Loop

Feedback options. See the DriveWare Software Guide for more information.

When using the Programmable Analog Input for ±10 VDC Position feedback, the drive cannot use a ±10V Analog input command through the reference inputs. See the Command Sources section below for other available types of command inputs.

5.3 Command Sources

The input command source for DZ drives can be provided by one of the following options.

5.3.1 ±10V Analog

Both DZC and DZR drives accept a differential or single-ended analog signal with a range of ±10 V from an external source. The input command signals should be connected to pins P1-3 and P1-4. See “Programmable Analog Input” on page 42 for more information.

5.3.2 Encoder Following

Both DZC and DZR drives can utilize Encoder Following as a form of input command. In Encoder Following mode, an auxiliary differential encoder signal can be used to command the drive in a master/slave configuration. The gearing ratio (input counts to output counts ratio) can be configured in DriveWare by the user. Encoder Following is only a valid option when the DZ drive is operated in position mode. The auxiliary encoder signal input should be connected to pins P1- 17, P1- 18, P1-19, and P1-20. See “Auxiliary Encoder Input” on page 42 for more information.

5.3.3 Over the Network

Both DZC and DZR drives can utilize network communication as a form of input command. DZC drives can provide an input reference command through the CAN interface, and DZR drives can provide an input reference command through the RS-485 interface. For more information on CANopen and RS- 485, see “Communication Protocol” on page 6.

5.3.4 PWM and Direction

DZ drives accept either a PWM and Direction or a Single Input PWM signal as the command source input type. The Direction inputs commands the direction of rotation, while the PWM input duty cycle commands the drive output. The PWM input is connected to P1-17, while the Direction input is connected to P1-19. Scaling, offset, and command inversion may be configured for customized control. The PWM and Direction command source supports broken wire detection for cases when the PWM command reaches 0% or 100% duty cycle. The frequency range of the PWM and Direction command input is 1kHz - 125kHz.

5.3.5 Indexing

DZ drives allow configuration of up to 16 separately defined Index tasks in DriveWare. Indexes can be either Absolute (commands a pre-defined move to an absolute position) or Relative (commands a pre-defined move relative to the current position).

5.3.6 Jogging

DZ drives allow configuration of two separate Jog velocities in DriveWare, commanding motion at a defined constant velocity with infinite distance.

5.3.7 5V TTL Step and Direction (DZR Drives Only)

DZR drives accept a differential Step and Direction input command from an external source. The Direction input commands the direction of rotation, while each pulse of the Step input commands the motor to "step" in that direction. Since the input is directly controlling the actual position of the motor, the physical motor location can be determined without any other feedback element. The differential Step input signal should be connected to pins P1-17 and P1-18, and the differential Direction input signal should be connected to pins P1-19 and P1-20. See “Step and Direction Input (DZR drives only)” on page 42 for more information.

5.4 System Requirements

To successfully incorporate a DZ digital servo drive into your system, you must be sure it will operate properly based on electrical, mechanical, and environmental specifications while anticipating impacts on performance.

5.4.1 Specifications Check

Before selecting a DZ digital servo drive, a user should consider the requirements of their system. This involves calculating the voltage, current, torque, and power requirements of the system, as well as considering the operating environment and any other equipment the drive will be interfacing with. Before attempting to install or operate a DZ servo drive, be sure all the following items are available:

- DZ Digital Servo Drive
- DZ Servo Drive Datasheet (specific to your model)
- DZ Series Digital Hardware Installation Manual
- DriveWare Software Guide

DZ servo drives are shipped with no other connectors or mounting components other than the signal and power header pins on the drive PCB itself. However, mounting cards and mating connectors are readily available. See “Mounting Card” on page 27 for the ADVANCED Motion

Note Controls DZ mounting card. Customized mounting options are also available for orders with sufficient volume.

5.4.2 Motor Specifications

DZ digital servo drives have a given current and voltage rating unique to each drive. Based on the necessary application requirements and the information from the datasheet of the motor being used, a DZ drive may be selected that will best suit the motor capabilities. Some general guidelines that are useful when pairing a DZ servo drive with a motor:

- The motor current IM is the required motor current in amps DC, and is related to the torque needed to move the load by the following equation:

$$M = \frac{\text{Torque}}{KT}$$

Where: KT -motor torque constant

The motor current will need to be calculated for both continuous and peak operation. The peak torque will be during the acceleration portion of the move profile. The continuous torque is the average torque required by the system during the move profile, including dwell times.

- The system voltage requirement is based on the motor properties and how fast and hard the motor is driven. The system voltage requirement is equal to the motor voltage, VM, required to achieve the move profile.

$$V_M = (K_E \cdot S_M) + (I_M \cdot R_M)$$

Where:

K_E -motor back EMF constant
S_M -motor speed (use the maximum speed expected for the application)
I_M -motor current (use the maximum current expected for the application)
R_M -motor line-to-line resistance

- The motor inductance is vital to the operation of DZ servo drives, as it ensures that the DC motor current is properly filtered.

A motor that does not meet the rated minimum inductance value of the DZ drive may damage the drive! If the motor inductance value is less than the minimum required for the selected drive, use of an external filter card is necessary.

A minimum motor inductance rating for each specific DZ drive can be found in the drive datasheet. If the drive is operated below the maximum rated voltage, the minimum load inductance requirement may be reduced.

5.4.3 Power Supply Specifications

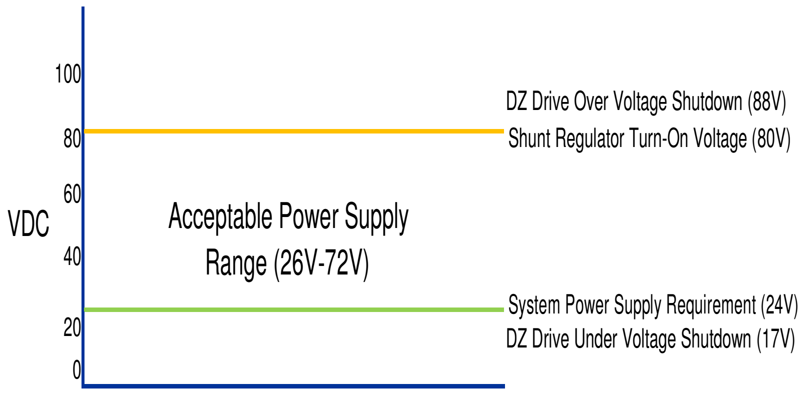
DZ servo drives operate off an isolated unregulated DC Power Supply (see Table 2.2 for drive model power supply ranges and over-voltage shutdown values). To avoid nuisance over- or under-voltage errors caused by fluctuations in the power supply, the system power supply voltage should be at least 10% above the entire system voltage requirement, and at least 10% below the lowest value of the following:

- Drive over voltage
- External shunt regulator turn-on voltage

Use of a shunt regulator is necessary in systems where motor deceleration or a downward motion of the motor load will cause the system's mechanical energy to be regenerated via the drive back onto the power supply. This regenerated energy can charge the power supply capacitors to levels above that of the DZ drive over - voltage shutdown level. If the power supply capacitance is unable to handle this excess energy, or if it is impractical to supply enough capacitance, then an external shunt regulator must be used to dissipate the regenerated energy. The shunt regulator will "turn-on" at a certain voltage level (set below the drive over-voltage shutdown level) and discharge the regenerated electric energy in the form of heat.

The diagram below provides a possible example of an appropriate system power supply voltage for a DZ-012L080 drive using an external shunt regulator.

FIGURE 2.6 Sample DZ-012L080 Power Supply Selection



The power supply current rating is based on the maximum current that will be required by the system. If the power supply powers more than one drive, then the current requirements for each drive should be added together. Due to the nature of servo drives, the current into the drive does not always equal the current out of the drive. However, the power in is equal to the power out. Use the following equation to calculate the power supply output current, IPS, based on the motor current requirements.

$$I_{PS} = \frac{V_M ? I_M}{V_{PS} ? (0.98)}$$

Where: PS

V_M	-nominal power supply voltage
I_M	-motor current
V	-motor voltage

Use values of V and I at the point of maximum power in the move profile (when VMIM = max). This will usually be at the end of a hard acceleration when both the torque and speed of the motor is high.

5.4.4 Environment

To ensure proper operation of a DZ servo drive, it is important to evaluate the operating environment prior to installing the drive.

Environmental Specifications

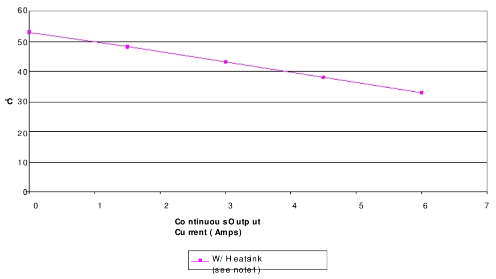
Parameter	Description
Ambient Temperature Range	See Figure 2.7
Baseplate Temperature Range [1]	0 - 65 ºC (DZ-012L080) ; 0 - 75 ºC (DZ-020L080 / DZ-040L080 / DZ-060L080 / DZ-010L200/DZ-025L200)
Humidity	90%, non-condensing
Mechanical Shock	15g, 11ms, Half-sine
Vibration	2 - 2000 Hz @ 2.5g
Altitude	0-3000m

Altitude 0-3000m

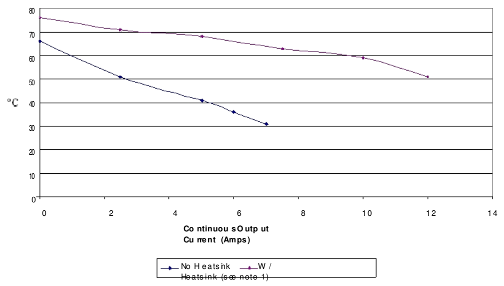
1.Thermal shutdown occurs when PCB temperature reaches this value. For DZ-060L080 and DZ-025L200 drives, the base plate temperature at this point may be between 60ºC and 75ºC depending on rate of base plate cooling (additional heat sinking), ambient temperature, and output current.
Ambient Temperature Range and Thermal Data DZ drives contain a built-in over-temperature disabling feature if the baseplate temperature rises above the maximum baseplate temperature value, specified in Table 2.6. When operating at a specific output current and with maximum DC supply voltage, Figure 2.7 below specifies an upper limit to the ambient temperature range DZ drives can operate within while keeping the baseplate temperature below the maximum baseplate temperature. It is recommended to mount the baseplate of the DZ drive to a heatsink for best thermal management results. Note that DZ-012L080 drives operated at supply voltages approaching 80VDC will require additional heatsinking to achieve rated characteristics, so no natural convection data is given.

FIGURE 2.7 DZ Drives Maximum Ambient Temperature Range

Maximum Ambient °C
DZ-012L080 Drive Models at 80VDC



Maximum Ambient °C
Z-020L080 Drive Models at 80VDC



1.The heatsink used in the above tests is 15" x 22" x 0.65"
aluminum plate.
2 For DZ-060L080 models at 80VDC, operating at continuous currents above 20 amps will require additional forced air cooling.

Shock/Vibrations While DZ drives are designed to withstand a high degree of mechanical shock and vibration, too much physical abuse can cause erratic behavior, or cause the drive to cease operation entirely. Be sure the drive is securely mounted in the system to reduce the shock and vibration the drive will be exposed to. The best way to secure the drive against mechanical vibration is to use screws to mount the DZ drive against its baseplate. For information on mounting options and procedures, see “Mounting” on page 27 .

Care should be taken to ensure the drive is securely mounted in a location where no moving parts will come in contact with the drive.

6. Operation and Features

This chapter will present a brief introduction on how to test and operate a DZ servo drive. Read through this entire section before attempting to test the drive or make any connections.

6.1 Features and Getting Started

To begin operation with your DZ drive, be sure to read and understand the previous chapters in this manual as well as the drive datasheet and the DriveWare Software Manual. Ensure that all system specifications and requirements have been met, and become familiar with the capabilities and functions of the DZ drive. Also, be aware of the "Troubleshooting" section at the end of this manual for solutions to basic operation issues.

6.1.1 Initial Setup and Configuration

Carefully follow the grounding and wiring instructions in the previous chapters to make sure your system is safely and properly set up. For initial testing purposes, it is not necessary to use a controller to provide a command input, or to have any load attached to the motor. The items required will be:

- DZ Servo Drive attached to Mounting Card or PCB Interface
- Motor
- DC Power Supply and Logic Power Supply for supplying power to system
- DriveWare Setup Software and Software Manual for detailed instructions on how to setup, tune and configure a DZ drive in DriveWare

The following steps outline the general procedure to follow when commissioning a DZ drive for the first time. The DriveWare Software Manual contains more detailed information on each step.

1. Check System Wiring: Before beginning, check the wiring throughout the system to ensure proper connections and that all grounding and safety regulations have been followed appropriately for the system.

Do not apply power to the system until certain all wiring and grounding has been setup safely and properly!

2. Apply Power: Power must be applied to the drive before any communication or configuration can take place. Turn on the +5 VDC Logic supply, then turn on the main DC Power supply. Use a multimeter or voltmeter to check that both power supply levels are within their specified ranges.

3. Establish Connection: Open DriveWare on the PC. The DZ drive should be attached to a mounting card or PCB interface, and connected to the PC with a serial cable. Choose the "Connect to a drive" option when DriveWare starts, and enter the appropriate communication settings in the options window that appears. See the DriveWare Software Guide for more information on connecting to a drive. For connection issues, see "Connection Problems" on page 52.

4. Input Motor Data: Once DriveWare has connected to the DZ drive, the motor and feedback information must be entered in DriveWare. This information is required for the drive to be configured properly.
5. Set User Units: User Units allows the user to set the general units that will be used in DriveWare. A variety of unit types are available, and DriveWare also offers the option of using custom units.

6. Configure Drive Limits and Events: DriveWare allows the user to manually configure system parameters and limits, and assign "actions" to specific events. The limits and their corresponding actions are used as both safety measures to avoid system damage, as well as parameter observation tools for drive configuration and troubleshooting.
7. Tune the Current Loop: Once the drive parameters are configured properly, the current loop must be tuned. This is the innermost loop and forms the basis of all motion. The current loop gains can be calculated based on the motor and application data entered in the previous two steps, or they can be set manually. The DriveWare Software Manual contains instructions on current loop tuning.

8. Commutate the Motor: For Three Phase (Brushless) motors, commutation is necessary to maintain the optimal torque generation at any motor position. Typically this is accomplished by running the AutoCommutation routing in DriveWare. Also see "Commutation" on page 44.

Once the drive has been commutated successfully, the drive is ready for further use, such as velocity or position loop tuning. Consult the DriveWare Software Manual for the correct procedure.

6.1.2 Commutation

Motor commutation is the process that maintains an optimal angle between the magnetic field created by the permanent magnets in the motor and the electromagnetic field created by the currents running through the motor windings. This process ensures optimal torque or force generation at any motor position. Single phase (brushed) motors accomplish this process with internal commutators built into the motor housing. Three phase (brushless) motors require a correctly configured drive to commutate properly, however. There are two ways to configure a DZ digital drive to commutate a three phase (brushless) motor in DriveWare:

- AutoCommutation: Most applications can use the AutoCommutation routine in DriveWare for configuring a drive to a specific motor. This routine will automatically detect the feedback devices attached to the motor and ask the user to verify them against the motor's data sheet.
- Manual Commutation: This process requires more time, and may not be as accurate as AutoCommutation. This method will have to be performed if:

— The motor is mechanically restrained such that it is unable to rotate (AutoCommutation requires the motor to rotate 2 revolutions + 1 electrical cycle in both directions for a rotary motor, and 3 electrical cycles for a linear motor).

— The motor or load has a significant amount of inertia.

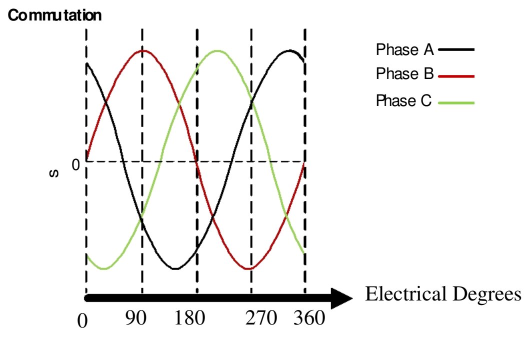
For three phase (brushless) drives that use encoder feedback but no Hall Sensors, it is required to run the "Phase Detect" routine in DriveWare before AutoCommutation. Phase Detect works by sending a small current signal to the motor, prompting the motor to vibrate slightly for a few seconds. The encoder feedback from this movement provides a starting position for the motor, allowing the drive to then be properly commutated.

See the DriveWare Software Manual for more information on AutoCommutation, Manual Commutation, and Phase Detect.

DZ drives allow either sinusoidal or trapezoidal commutation.

Sinusoidal Commutation Sinusoidal commutation provides greater performance and efficiency than trapezoidal commutation. DZ drives can commutate sinusoidally when connected to a motor- mounted encoder. Sinusoidal Commutation works by supplying current to each of the three motor phases smoothly in a sinusoidal pattern. The flow of current through each phase is shifted by 120 degrees. The sum of the current flowing through all three phases adds up to zero. Figure 4.2 shows one electrical cycle of the motor phase currents.

FIGURE 4.2 Sinusoidal Commutation Motor Phase Currents



Trapezoidal Commutation Trapezoidal commutation is accomplished with the use of Hall Sensors on three phase (brushless) motors. DZ drives can commutate trapezoidally when used with properly spaced Hall Sensors. Unlike sinusoidal commutation, current flows through only two motor phases at a time with trapezoidal commutation. The Hall Sensors each generate a square wave with a certain phase difference (either 120- or 60 - degrees) over one electrical cycle of the motor. This results in six distinct Hall states for each electrical cycle. Depending on the motor pole count, there may be more than one electrical cycle per motor revolution. The number of electrical cycles in one motor revolution is equal to the number of motor poles divided by 2. For example:

- a 6-pole motor contains 3 electrical cycles per motor revolution
- a 4-pole motor contains 2 electrical cycles per motor revolution
- a 2-pole motor contains 1 electrical cycle per motor revolution

The drive powers two of the three motor phases with DC current during each specific Hall Sensor state as shown in Figure 4.3.

FIGURE 4.3 Hall Sensor Commutation and Motor Phase Current for 120-Degree Phasing

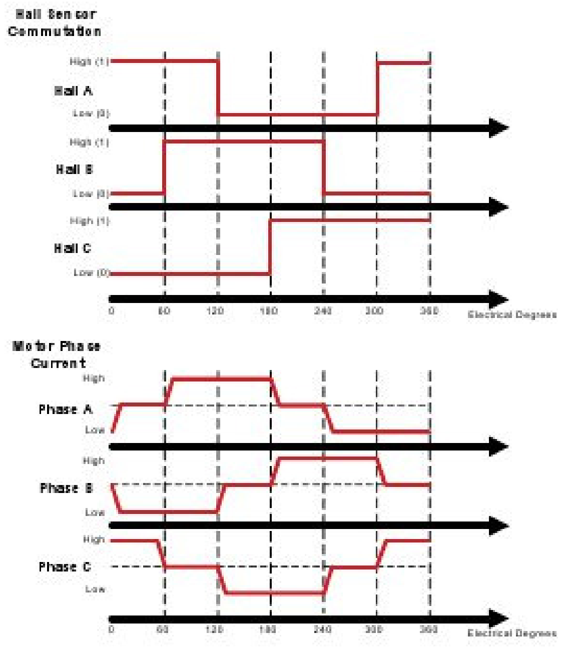


Table 4.5 shows the default commutation states for 120-degree and 60-degree phasing.

TABLE 4.5 DigitalDependingDriveCommutationonthespecificSequencesetup,theTablesequences may change after running AutoCommutation.

		60 Degree			120 Degree			Motor	
	Hall 1	Hall 2	Hall 3	Hall 1	Hall 2	Hall 3	Phase A	Phase B	Phase C
	1	0	0	1	0	0	-	LOW	HIGH
	1	1	0	1	1	0	HIGH	LOW	-
Valid	1	1	1	0	1	0	HIGH	-	LOW
	0	1	1	0	1	1	-	HIGH	LOW
	0	0	1	0	0	1	LOW	HIGH	-
	0	0	0	1	0	1	LOW	-	HIGH
Invalid	1	0	1	1	1	1	-	-	-
	0	1	0	0	0	0	-	-	-

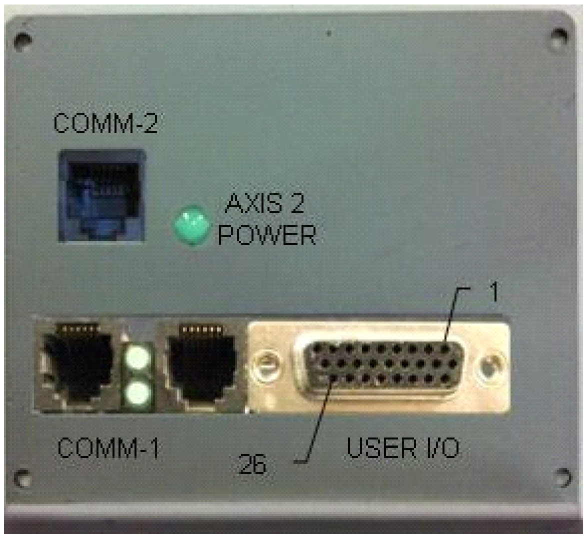
Appendix 1. Specification

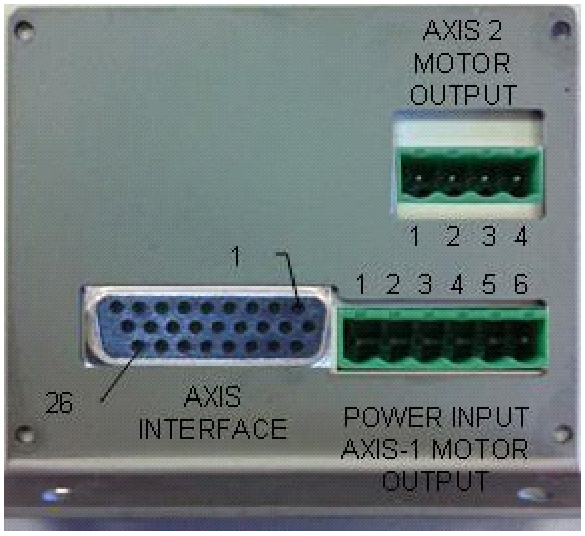
Description	**Stand-Alone 2 Axis Servo Motor Controller / Driver**
Operating Modes	Position, Velocity, Torque, Electronic Gearing
Control Algorithm	PID
Max. Servo Loop Rate	100 µS per enabled axis
Servo Position Feedback	Incremental Encoder with Index
Output (Standard)	PWM Motor Drive, Single Phase (brushed), 3 Amps Cont. and 6 Amps Peak at 50 VDC Max.
PWM Frequency	Approximately 19.531 KHz
Encoder and Index Input	5V Single-ended or Differential
Encoder Supply Voltage	5 VDC
Encoder Count Rate	2 Million Quadrature Counts per Second
Position Range	32 Bits
Velocity Range	32 Bits
Acceleration Range	32 Bits
General Purpose Digital I/O	3 Optoisolated inputs, 2 Optoisolated outputs
Dedicated Digital Inputs	Limit+, Limit-, Home and Fault, for each axis
Analog Inputs	5 Channels With 10-Bit Resolution, 3 are user accessible
Analog Output2	1 Channels, with 12-bits Resolution, ±10 VDC
Expansion I/O	Optional Expansion to 64 I/O
Communication Interface	RS-232
Supply Voltage	+11 To +50 VDC
Motor Voltage	+12 To +48 VDC
Dimensions	226mm Long by 88mm wide by 73mm Thick
Weight	Approximately 1.2 Kg

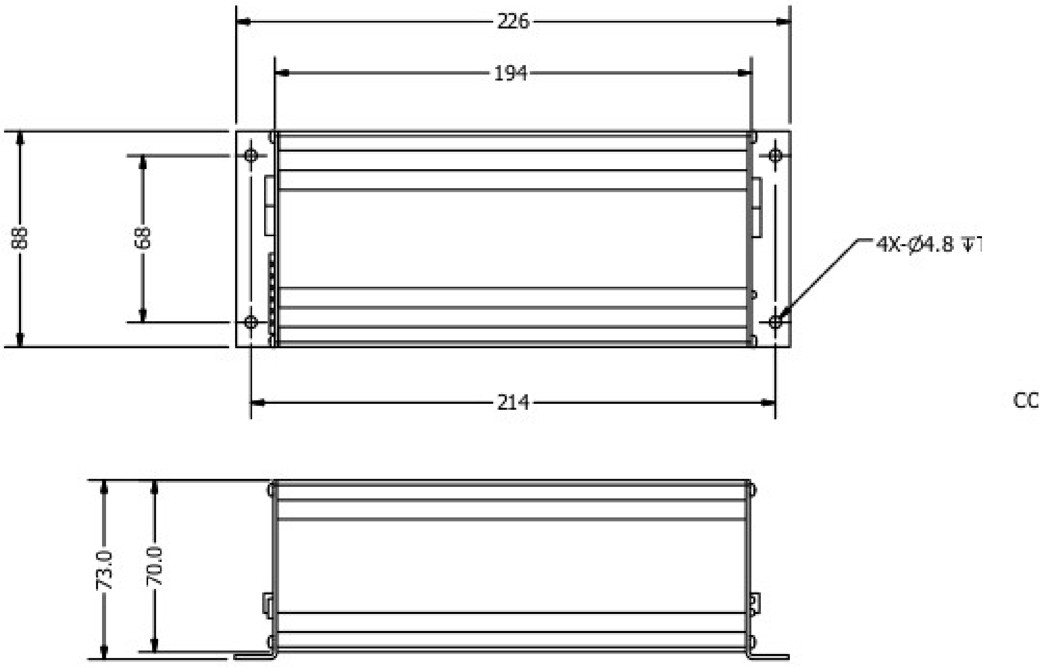
2nd Axis Driver Specification

Description	DigiFlex Servo Drive
Operation Mode	Current Loop(50 us), Velocity Loop(100 us), Position Loop(100 us)
Command Source	±10 V Analog, 5V Step and Direction, Encoder Following, PWM and Direction
Feed Back Support	Incremental Encoder (Max 20 MHz Quadrature), Halls, ±10VDC Position, Aux Incremental Encoder
Commutation Methods	Sinusodal, Trapezoidal
Output (Motor Supported)	Single Phase Brushed, Voice Coil, Inductive Load ; Three Phase (Brushless); 12A peak, 6A Continuous current
Communication Interface	RS-232, RS-485
Supply Voltage	+20 To +80 VDC
Logic Supply Voltage	5 VDC
Option	1st axis Configured

Appendix 2 Outline drawing







Aooendix 3. Connector Pin Definitions

J2 - Power Interface
8-Pin 5.08mm Centers Phoenix
Digi-Key# ED1721-ND

1. Axis 1 Motor- output
2. Axis 1 Motor+ output
3. NC
4. NC
5. Main V+ power input
6. Main power return

J3 – Axis 2 Motor Output
4-Pin 5.08 Centers Phoenix
Digi-Key # ED1719-ND

1.Motor Phase A
2.Motor Phase C
3.Motor Phase B
4.Power Return

J1 - Axis Interface
26-Pin High density female D-Sub
NorComp# HDT26P : Digi-Key# T815M-ND

1. Axis 2 Limit- input
2. Axis 1 Fault input
3. Axis 1 Limit+ input
4. +5 VDC
5. +5 VDC
6. Axis 2 Encoder phase B-
7. Axis 2 Encoder phase B+
8. Axis 1 Encoder phase B-
9. Axis 1 Encoder phase B+
10. Axis 2 Fault input
11. Axis 2 Limit+ Input
12. Axis 1 Home Input
13. Common
14. Common
15. Axis 2 Encoder phase A-
16. Axis 2 Encoder phase A+
17. Axis 1 Encoder phase A-
18. Axis 1 Encoder phase A+
19. Axis 2 Home Input
20. Axis 1 Limit- Input
21. Common
22. Common
23. Axis 2 Encoder Index-
24. Axis 2 Encoder Index+
25. Axis 1 Encoder Index-
26. Axis 1 Encoder Index+

J4 – User I/O Interface
26-Pin High density female D-Sub
NorComp# HDT26P : Digi-Key# T815M-ND

1. NC
2. Analog input 2
3. Analog input 1
4. NC
5. NC
6. Opto-input 1
7. Opto-input 1 return
8. Opto-output 1
9. Opto-output 1 return
10. Axis 1 Analog output
11. NC
12. Analog input 0
13. NC
14. NC
15. NC
16. NC
17. Opto-input 0
18. Opto-input 0 return
19. Axis 1 Analog output return
20. 10 VDC Analog refence output
21. Analog reference return
22. Common
23. Opto-input 2
24. Opto-input 2 return
25. Opto-output 0
26. Opto-output 0 return

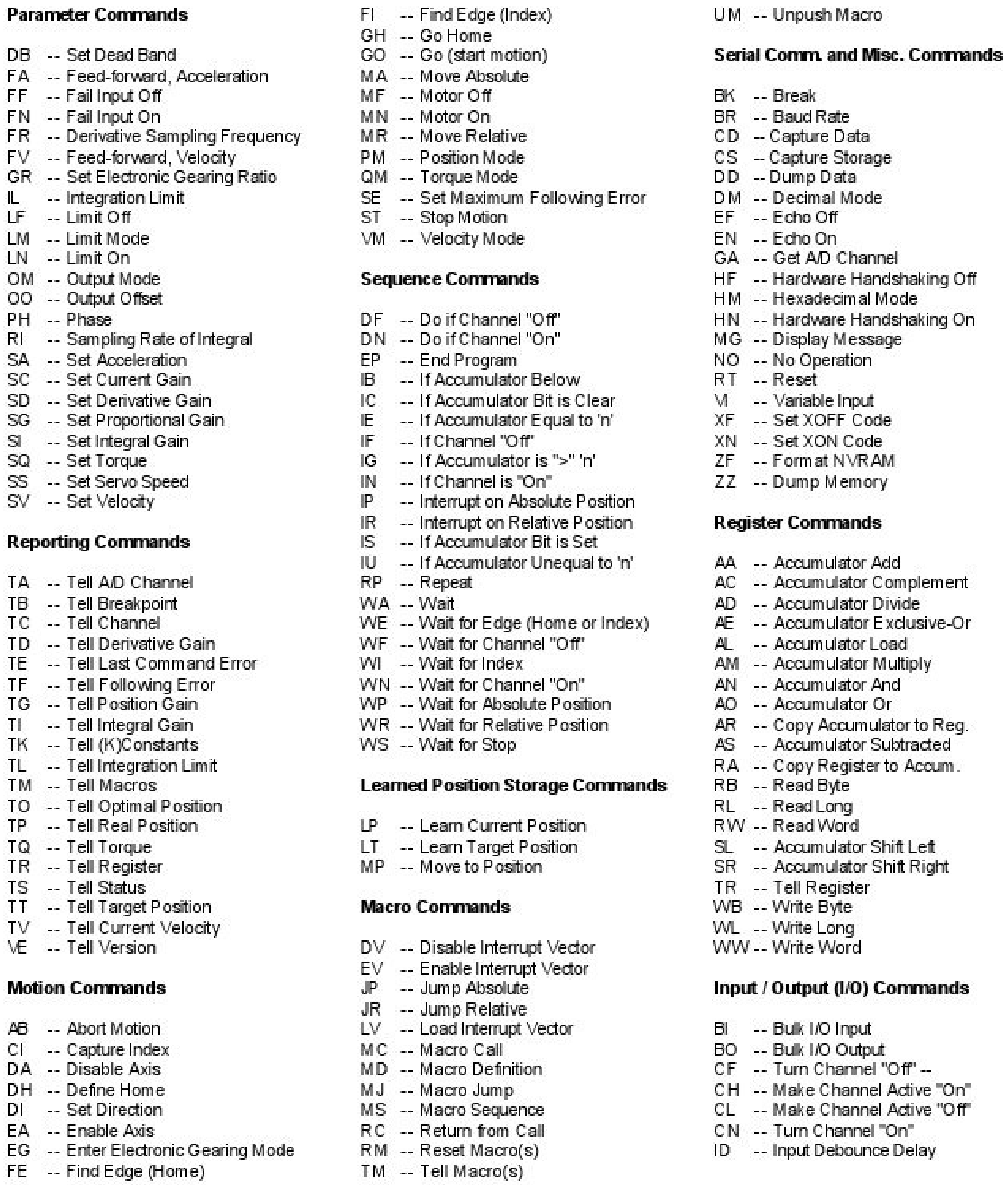
COMM-1 - RS-232 Communication Interface
6-Pin Modular Type
AMP# 5-641337-3 : Digi-Key# A9093-ND

1. Handshake output
2. Handshake input
3. Receive data input
4. Transmit data output
5. Common
6. +5 VDC

COMM-2 – RS232 Communication Interface
Axis 2 Servo Drive
6-Pin Modular Type
AMP# 5-641337-3 : Digi-Key# A9093-ND

1. NC
2. NC
3. Receive data input
4. Transmit data output
5. Common
6 NC

Appendix 4 – Summary of LAC-26 Commands



Parameter Commands

DB	-- Set Dead Band
FA	-- Feed-forward, Acceleration
FF	-- Fail Input Off
FN	-- Fail Input On
FR	-- Derivative Sampling Frequency
FV	-- Feed-forward, Velocity
GR	-- Set Electronic Gearing Ratio
IL	-- Integration Limit
LF	-- Limit Off
LM	-- Limit Mode
LN	-- Limit On
OM	-- Output Mode
OO	-- Output Offset
PH	-- Phase
RI	-- Sampling Rate of Integral
SA	-- Set Acceleration
SC	-- Set Current Gain
SD	-- Set Derivative Gain
SG	-- Set Proportional Gain
SI	-- Set Integral Gain
SQ	-- Set Torque
SS	-- Set Servo Speed
SV	-- Set Velocity

Reporting Commands

TA	-- Tell A/D Channel
TB	-- Tell Breakpoint
TC	-- Tell Channel
TD	-- Tell Derivative Gain
TE	-- Tell Last Command Error
TF	-- Tell Following Error
TG	-- Tell Position Gain
TI	-- Tell Integral Gain
TK	-- Tell (K)Constants
TL	-- Tell Integration Limit
TM	-- Tell Macros
TO	-- Tell Optimal Position
TP	-- Tell Real Position
TQ	-- Tell Torque
TR	-- Tell Register
TS	-- Tell Status
TT	-- Tell Target Position
TV	-- Tell Current Velocity
VE	-- Tell Version

Motion Commands

AB	-- Abort Motion
CI	-- Capture Index
DA	-- Disable Axis
DH	-- Define Home
DI	-- Set Direction
EA	-- Enable Axis
EG	-- Enter Electronic Gearing Mode
FE	-- Find Edge (Home)
FI	-- Find Edge (Index)
GH	-- Go Home
GO	-- Go (start motion)
MA	-- Move Absolute
MF	-- Motor Off
MN	-- Motor On
MR	-- Move Relative
PM	-- Position Mode
QM	-- Torque Mode
SE	-- Set Maximum Following Error
ST	-- Stop Motion
VM	-- Velocity Mode

Sequence Commands

DF	-- Do if Channel "Off"
DN	-- Do if Channel "On"
EP	-- End Program
IB	-- If Accumulator Below
IC	-- If Accumulator Bit is Clear
IE	-- If Accumulator Equal to 'n'
IF	-- If Channel "Off"
IG	-- If Accumulator is ">" 'n'
IN	-- If Channel is "On"
IP	-- Interrupt on Absolute Position
IR	-- Interrupt on Relative Position
IS	-- If Accumulator Bit is Set
IU	-- If Accumulator Unequal to 'n'
RP	-- Repeat
WA	-- Wait
WE	-- Wait for Edge (Home or Index)
WF	-- Wait for Channel "Off"
WI	-- Wait for Index
WN	-- Wait for Channel "On"
WP	-- Wait for Absolute Position
WR	-- Wait for Relative Position
WS	-- Wait for Stop

Learned Position Storage Commands

LP	-- Learn Current Position
LT	-- Learn Target Position
MP	-- Move to Position

Macro Commands

DV	-- Disable Interrupt Vector
EV	-- Enable Interrupt Vector
JP	-- Jump Absolute
JR	-- Jump Relative
LV	-- Load Interrupt Vector
MC	-- Macro Call
MD	-- Macro Definition
MJ	-- Macro Jump
MS	-- Macro Sequence
RC	-- Return from Call
RM	-- Reset Macro(s)
TM	-- Tell Macro(s)
UM	-- Unpush Macro

Serial Comm. and Misc. Commands

BK	-- Break
BR	-- Baud Rate
CD	-- Capture Data
CS	-- Capture Storage
DD	-- Dump Data
DM	-- Decimal Mode
EF	-- Echo Off
EN	-- Echo On
GA	-- Get A/D Channel
HF	-- Hardware Handshaking Off
HM	-- Hexadecimal Mode
HN	-- Hardware Handshaking On
MG	-- Display Message
NO	-- No Operation
RT	-- Reset
VI	-- Variable Input
XF	-- Set XOFF Code
XN	-- Set XON Code
ZF	-- Format NVRAM
ZZ	-- Dump Memory

Register Commands

AA	-- Accumulator Add
AC	-- Accumulator Complement
AD	-- Accumulator Divide
AE	-- Accumulator Exclusive-Or
AL	-- Accumulator Load
AM	-- Accumulator Multiply
AN	-- Accumulator And
AO	-- Accumulator Or
AR	-- Copy Accumulator to Reg.
AS	-- Accumulator Subtracted
RA	-- Copy Register to Accum.
RB	-- Read Byte
RL	-- Read Long
RW	-- Read Word
SL	-- Accumulator Shift Left
SR	-- Accumulator Shift Right
TR	-- Tell Register
WB	-- Write Byte
WL	-- Write Long
WW	-- Write Word

Input / Output (I/O) Commands

BI	-- Bulk I/O Input
BO	-- Bulk I/O Output
CF	-- Turn Channel "Off" --
CH	-- Make Channel Active "On"
CL	-- Make Channel Active "Off"
CN	-- Turn Channel "On"
ID	-- Input Debounce Delay

Appendix 5 AMC Driver Troubleshooting

B.1 Fault Conditions and Symptoms

An inoperative drive is typically an indication of a disabling fault condition. The fault condition can either be caused by a system parameter in excess of software or hardware limits, or by an event that has been user-configured to disable the drive upon occurrence.

To determine whether the drive is in a fault state, use the Drive Status function in DriveWare to view active and history event items and drive fault conditions. See the DriveWare Software Guide for more information on reading the Drive Status window. Some common fault conditions caused by hardware issues are listed below.

Over-Temperature Verify that the baseplate temperature is less than the maximum allowable baseplate temperature value. The drive remains disabled until the temperature at the drive baseplate falls below this threshold.

Over-Voltage Shutdown

1. Check the DC power supply voltage for a value above the drive over-voltage shutdown limit. If the DC bus voltage is above this limit, check the AC power line connected to the DC power supply for proper value.

2. Check the regenerative energy absorbed during deceleration. This is done by monitoring the DC bus voltage with a voltmeter or oscilloscope. If the DC bus voltage increases above the drive over-voltage shutdown limit during deceleration or regeneration, a shunt regulator may be necessary. See “Power Supply Specifications” on page 17 for more information.

Under-Voltage Shutdown Verify power supply voltage for minimum conditions per specifications. Also note that the drive will pull the power supply voltage down if the power supply cannot provide the required current for the drive. This could occur when high current is demanded and the power supply is pulled below the minimum operating voltage required by the drive.

Short Circuit Fault

1. Check each motor lead for shorts with respect to motor housing, power ground, and also phase-to-phase. If the motor is shorted it will not rotate freely when no power is applied while it is uncoupled from the load.

2. Disconnect the motor leads to see if the drive will enable without the motor connected.

3. Measure motor armature resistance between motor leads with the drive disconnected.

Invalid Hall Sensor State See the “Commutation Sequence” table in “Hall Sensors” on page 9 for valid commutation states. If the drive is disabled check the following:

1. Check the voltage levels for all the Hall sensor inputs.

2. Make sure all Hall Sensor lines are connected properly.

B.1.1 Software Limits

Because DriveWare allows user configuration of many system parameters such as current, velocity, and position limits, as well as an associated "event action" for DriveWare to take when the system reaches this limit, it is possible for a drive to appear to be inoperative when in actuality it is simply in an assigned disable state.

For example, the motor velocity can be limited by giving a value to the Motor Over Speed selection in DriveWare. An "event action", such as "Disable the Power Bridge", can also be assigned for DriveWare to take if the motor reaches this speed. If the motor does happen to reach this velocity limit, DriveWare will automatically cut power to the drive's output in this particular case, and the drive will be disabled. In the Drive Status window, "Motor Over Speed" will be shown as a "history" event, and "Commanded Disable" will be shown as an "Action" event.

Depending on each specific system and application, there are many different options available for assigning system limits and associated actions. See the DriveWare Software Guide for more information.

B.1.2 Connection Problems

Connection problems are oftentimes caused by incorrect communication settings in DriveWare. The default factory setting for DZ drives is a Drive Address of 63 and 115200 Baud Rate. When connecting to the drive with DriveWare for the first time, these default factory settings will have to be used along with the appropriate serial port being used with the PC. Once the connection has been established, the Drive Address and Baud Rate may be changed. Check all communications settings to be sure that the Drive Address, Baud Rate, and serial port are correct. If unable to determine the appropriate settings, the Auto Detect routine will automatically scan for serial port and Baud Rate settings.

Faulty connection cables are also a possible cause of connection problems. Check all cables for any shorts or intermittent connections.

For network communication over CAN or RS-485, the DZ drive must be configured for the appropriate communication protocol. See “RS-485 Selection Jumper” on page 43 for more information.

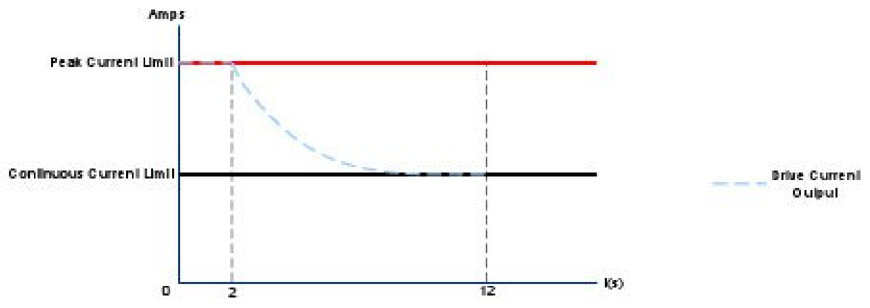
B.1.3 Overload

Verify that the minimum inductance requirement is met. If the inductance is too low it could appear like a short circuit to the drive and thus it might cause the short circuit fault to trip. Excessive heating of the drive and motor is also characteristic of the minimum inductance requirement not being met. See drive data sheets for minimum inductance requirements.

B.1.4 Current Limiting

All drives incorporate a “fold-back” circuit for protection against over -current. This “fold-back” circuit uses an approximate “I2t” algorithm to protect the drive. All drives can run at peak current for a maximum of 2 second (each direction) . Currents below this peak current but above the continuous current can be sustained for a longer time period, and the drive will automatically fold back at an approximate rate of "I2t" to the continuous current limit within a time frame of less than 10 seconds. An over-current condition will not cause the drive to become disabled unless configured to do so in DriveWare.

FIGURE B.1 Peak Current Fold-Back



B.1.5 Motor Problems

A motor run-away condition is when the motor spins rapidly with no control from the command input. The most likely cause of this error comes from having the feedback element connected for positive feedback. This can be solved by changing the order that the feedback element lines are connected to the drive, or by using DriveWare to reverse the internal velocity feedback polarity setting.

Another common motor issue is when the motor spins faster in one direction than in the other. This is typically caused by improper motor commutation or poor loop tuning. Follow the steps in the DriveWare Software Guide to properly commutate and tune the motor.

B.1.6 Causes of Erratic Operation

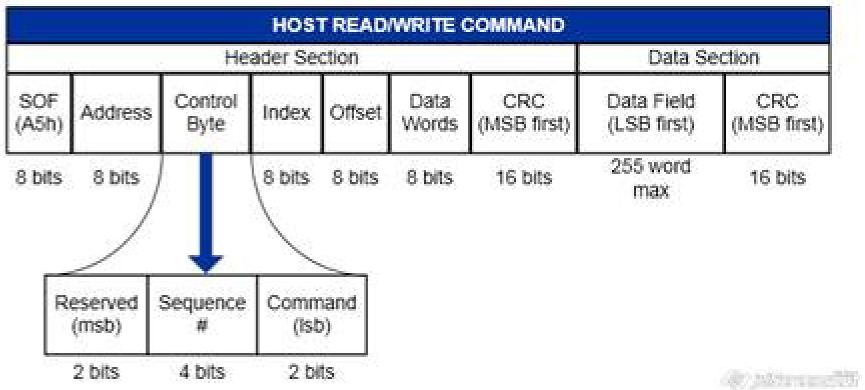
- Improper grounding (i.e., drive signal ground is not connected to source signal ground).
- Noisy command signal. Check for system ground loops.
- Mechanical backlash, dead-band, slippage, etc.
- Noisy inhibit input line.
- Excessive voltage spikes on bus

B.1.7 Phase detect problems on startup

Under some circumstances the phase detect fault signal can be set at power up which will show a phase detect error. If this condition occurs there is an index that controls which events can be reset using the reset events function via digital input (normally the phase detect fault can not be reset). Below is the procedure for setting the index bit to allow the phase detect fault to be reset after power on:

An rs232 program is required to write the hex information directly to the amplifier. Here is an example program: http://www.compuphase.com/software_termite.htm

The index that controls the reset events function is index 105, offset 2. Bit 5 should be set to 1 to enable phase detection fault a re-settable event. The command would look like this:



Send

SF	DA	CB	Ind.Off		L	CRC		Data		CRC	
A5	3F	02	69	02	01	55	EF	FC	0F	A7	43

This example shows the data written to index 105 (69h) as 0FFC. The drive previously had a value of 0FDC. Bit 5 was enabled to activate the Phase Detection Fault Bit in the reset event action index.