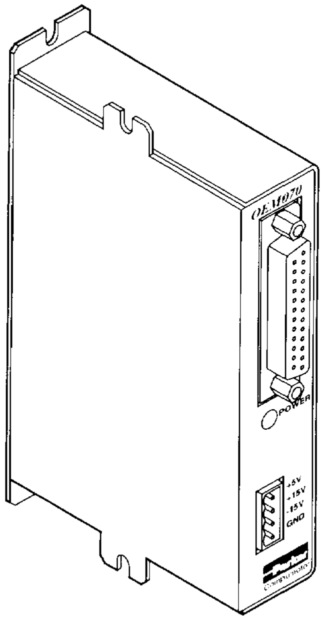
Compumotor

OEM070 Servo Controller User Guide



Compumotor Division
Parker Hannifin Corporation
p/n 88-014163-01 B

Parker

IMPORTANT

User Information

⚠ WARNING ⚠

OEM Series products are used to control electrical and mechanical components of motion control systems. You should test your motion system for safety under all potential conditions. Failure to do so can result in damage to equipment and/or serious injury to personnel.

Technical Assistance ⇨ *Contact your local automation technology center (ATC) or distributor, **or**...*

North America and Asia:
Compumotor,
Division of Parker Hannifin
5500 Business Park Drive
Rohnert Park, CA 94928

Telephone: (800) 358-9070
or (707) 584-7558
Fax: (707) 584-3793
FaxBack: (800) 936-6939
or (707) 586-8586
e-mail: tech_help@cmotor.com
Internet: http://www.compumotor.com

Europe *(non-German speaking)*:
Parker Digiplan
21 Balena Close
Poole, Dorset
England BH17 7DX
Telephone: +44 (0)1202 69 9000
Fax: +44 (0)1202 69 5750

Germany, Austria, Switzerland:
HAUSER Elektronik GmbH
Postfach: 77607-1720
Robert-Bosch-Str. 22
D-77656 Offenburg
Telephone: +49 (0)781 509-0
Fax: +49 (0)781 509-176

Parker
Automation

Technical Support Email
tech_help@cmotor.com

OEM070 Servo Controller User Guide

Revision B Change Summary

May 2000

The following is a summary of the primary technical changes to this document since the previous version was released. This document, part number 88-014163-01**B**, supersedes 88-014163-01**A**.

RS485 Communication Option Added
RS485 is now available as a custom option. Contact Compumotor's Custom Products Department for details. This feature is available only on OEM070 controllers with firmware 92-016637-01 or later.

Communication Error Checking Added
SSE (pg. 78) and % (pg. 92) are two new commands added to support communication error checking. These commands are only available on OEM070 controllers with firmware 92-016637-01 or later.

Firmware Upgrade
The new features and options listed above are available only for firmware 92-016637-01 or later. To upgrade older firmware, units must be returned for hardware and software upgrades at a nominal fee. Contact Compumotor's Customer Service department for details.

Homing Diagrams (pg. 99)
Homing diagrams have been added to aid in system setup.

Encoder Wiring (pg. 21)
Encoder wiring and color codes have been added for Compumotor's SM and NeoMetric Series motors. Wiring information for Compumotor's OEM Series motors, which are obsolete, has been removed.

C O N T E N T S

OEM070 USER GUIDE **7**
User Guide Chapter Overview 7
Installation Process Overview 8
Installation Preparation 8
Warnings & Cautions 9

1 INTRODUCTION **11**
OEM070 Description 11
  Features 11
  Block Diagram 12

2 INSTALLATION **13**
OEM070 Ship kit 13
OEM070 Mounting and Dimensions 14
  OEM070 Dimensions 14
  OEM070 Mounting 15
Power Supply Connections 15
OEM070 Inputs and Outputs 16
  General Purpose Inputs (Signal 1 – Signal 5) 16
  Output #1 (Signal 6) and Output #2 (Signal 8) 18
  CW (Signal 12) & CCW (Signal 13) Limit Inputs 18
  Dedicated Fault Input (Signal 9) 19
  RS-232C—Tx (Signal 14), Rx (Signal 15), and Ground (Signal 7) 19
  Enable Output (Signal 10) 20
  Encoder Inputs: +5V,A,B,Z ,Gnd (Signals 16 - 23) 20
  Daisy Chaining 22

C O N T E N T S

3 SPECIFICATIONS 23

4 TUNING 25

Tuning and Performance 25
PID Tuning 25
    CPG – Proportional Gain 27
    CDG – Derivative Gain; CTG – Derivative Sampling Period 27
    CIG – Integral Gain; CIL – Integral Limit 28
Tuning Procedure 28
    Tuning Parameters for Apex Series Motors 30
Configuring an In Position Window 31

5 SOFTWARE COMMANDS 33

Command Descriptions 33
Summary of Commands 94

6 TROUBLESHOOTING 95

Reducing Electrical Noise 95
Problem Isolation 95
RS-232C Problems 97
Software Debugging Tips 98
Encoder Problems 99
Homing Diagrams 99
Returning your System 101

INDEX 103

OEM070 User Guide

This user guide is designed to help you install, develop, and maintain your system. Each chapter begins with a list of specific objectives that should be met after you have read the chapter. This section will help you find and use the information in this guide.

User Guide Chapter Overview

CHAPTER ① INTRODUCTION

This chapter provides a description of the product and a brief account of its specific features.

CHAPTER ② INSTALLATION

This chapter contains a ship kit list of items you should have received with your OEM070. Instructions to mount and connect the system properly are included. Upon completion of this chapter, your system should be completely installed and ready to perform basic operations.

CHAPTER ③ SPECIFICATIONS

This chapter contains information on system performance specifications (environmental, etc.).

CHAPTER ④ TUNING

This chapter contains information on servo system gains and optimizing the performance of the system. A short procedure is included.

CHAPTER ⑤ SOFTWARE COMMANDS

This chapter contains descriptions off all software commands applicable to the OEM070. These commands are based on Compumotor's popular and easy-to-use X Series Language. The controller also provides additional I/O control and communication capabilities.

CHAPTER ⑥ TROUBLESHOOTING

This chapter contains information on identifying and resolving system problems. Descriptions of LED signals, debugging tools, problems/solutions table are included.

Installation Process Overview

To ensure trouble-free operation, pay special attention to the environment in which the equipment will operate, the layout and mounting, and the recommended wiring and grounding. These recommendations will help you easily and safely integrate the OEM070 into your manufacturing facility. If your environment contains conditions that may adversely affect solid-state equipment (electrical noise or atmospheric contamination), be sure to follow any special instructions to ensure the safety and long life of your equipment.

Installation Preparation

Before you install this product, complete the following steps:

1. Review this user guide. Become familiar with the user guide's contents so that you can quickly find the information you need.
2. Develop a basic understanding of all system components, their functions, and interrelationships.
3. Complete the basic system configuration and wiring instructions (in a simulated environment, not a permanent installation) provided in *Chapter ② Installation.*
4. Perform as many basic functions as you can with the preliminary configuration. Try to simulate the task(s) that you expect to perform when you permanently install your application (however, do not attach a load at this time). This will give you a realistic preview of what to expect from the complete configuration.
5. After you have tested the system's functions and become familiar with the system's basic features, carefully read *Chapter ② Installation.*

6. After you have read Chapter 2 and clearly understand what must be done to properly install the system, begin the installation process. Do not deviate from the instructions provided.
7. Before you customize your system, check all of the system functions and features to ensure that you have completed the installation process correctly.

The successful completion of these steps will prevent subsequent performance problems and allow you to isolate and resolve potential system difficulties before they affect your system's operation.

Warnings & Cautions

Warning and caution notes alert you to problems that may occur if you do not follow the instructions correctly. Situations that may cause bodily injury are presented as warnings. Situations that may cause system damage are presented as cautions.

CAUTION

System damage will occur if you power up the system improperly.

WARNING

During servo tuning, the system can undergo accidental and violent movement due to improper gain settings and programming errors. Please use extreme caution while prototyping

CHAPTER ①

Introduction

Section Objective

The information in this section will enable you to:

❒ Understand the product's basic functions and features

OEM070 Description

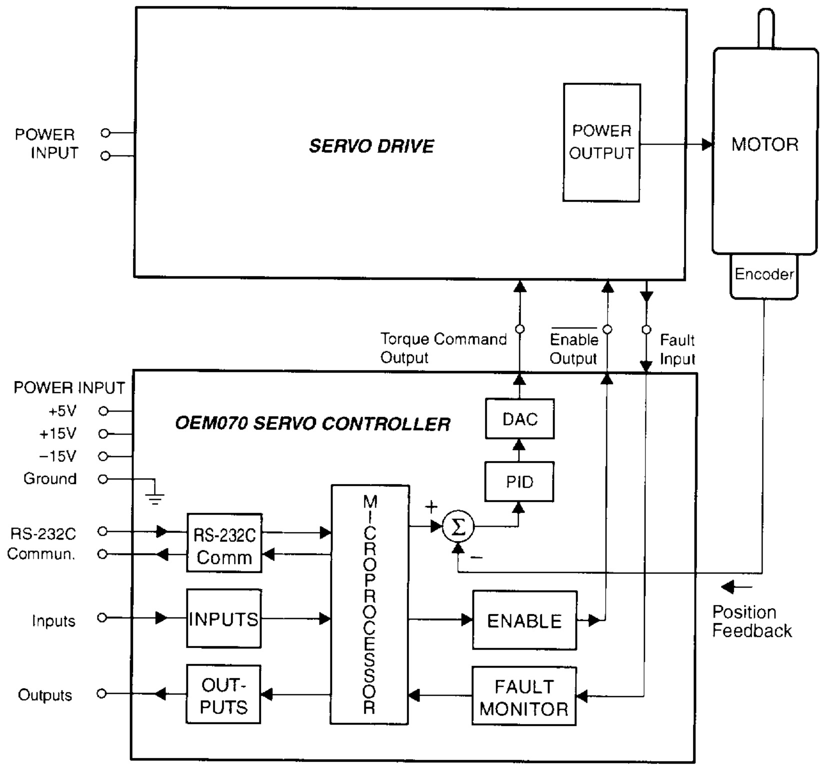
The OEM070 is a compact, stand-alone servo controller designed to operate with analog servo drives. Its small footprint, small depth housing contains a single board that controls motor torque or velocity with a ±10 VDC command output to servo drives operating in torque or velocity mode. The board also saves and executes stored programs, as well as interfaces with external devices such as incremental encoders, switches, host computers and Programmable Control Units.

Features

- ❒ The OEM070 is a single axis motion controller for applications such as :
  - ❍ Feed to Length
  - ❍ Step and Repeat
  - ❍ Linear Slide Positioning
- ❒ Full PID position servo control to maximize system performance
- ❒ 5 user-defined inputs
  - ❍ Trigger inputs
  - ❍ Sequence select
  - ❍ Home
  - ❍ Stop or Kill motion
  - ❍ Go
- ❒ 2 user-defined outputs
- ❒ 2K of BBRAM—Essential for program, RS232 address, tuning, and set-up storage

- ❐ Flexible operation modes
  - ❍ Stand Alone—Calls programs out of memory via remote inputs
  - ❍ On line—Connected via RS232 for continuous streaming of commands
- ❐ Uses standard differential or single-ended optical encoders for position information
- ❐ RS-232C communication for programming or direct operation
- ❐ Can daisy chain up to 255 units
- ❐ 960 kHz encoder input frequency
- ❐ Enable input for hardware disable
- ❐ Can store 7 programmed sequences in memory
- ❐ Dedicated CW, CCW limit inputs
- ❐ Fault Output can notify external system of servo fault

BLOCK DIAGRAM



OEM070 Servo Controller – Block Diagram

CHAPTER ②

Installation

Section Objectives

The information in this section will enable you to:

- ❐ Verify that each component of your system has been delivered safely and completely
- ❐ Become familiar with components and their interrelationships
- ❐ Understand the I/O functions and installation guidelines

OEM070 Ship kit

Inspect the OEM070 upon receipt for obvious damage to its shipping container. Report any such damage to the shipping company. Parker Compumotor cannot be held responsible for damage incurred in shipment. You should receive an OEM070 and a Servo Controller User Guide. Compare your order with the units shipped.

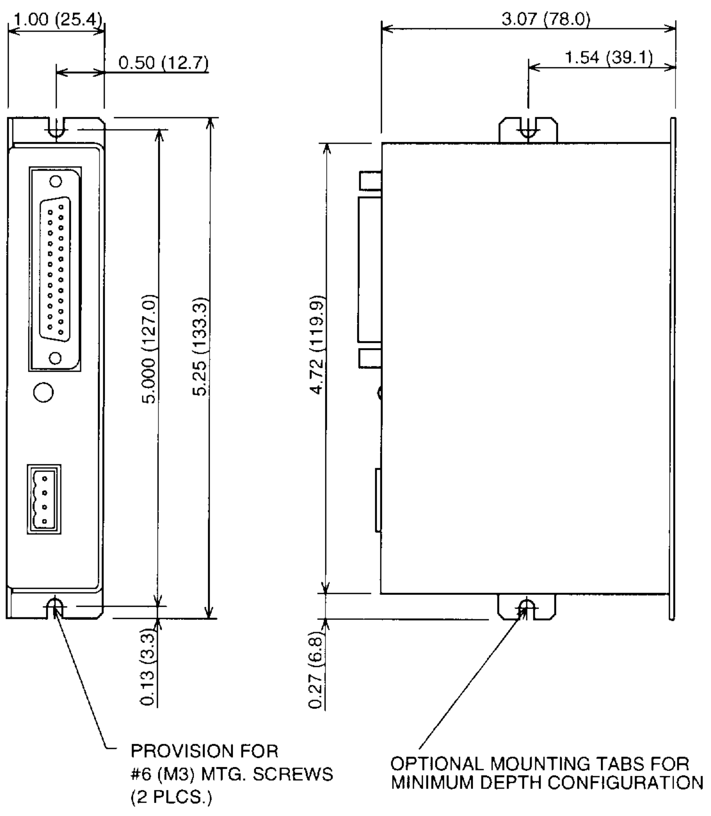
Part	**Part Number**
Controller	OEM070
OEM070 User Guide	*88-014163-01*

OEM070 Mounting and Dimensions

Refer to the instructions and diagrams in this section for specific mounting information.

OEM070 Dimensions

The OEM070 mounting is designed to minimize panel area and footprint. It can be mounted in a minimum area or a minimum depth configuration.



OEM070 Dimensions In Inches (Millimeters)

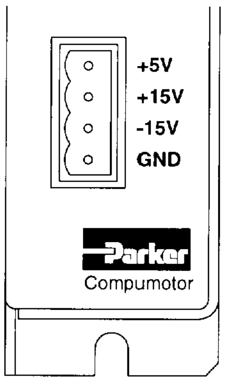
OEM070 Mounting

The OEM070 produces very little heat and can be mounted almost anywhere. The OEM070 is not water-proof, dust-proof, or splash proof, so please provide suitable controller protection. If you mount the OEM070 in an enclosure, observe the following guidelines:

❑ Do not mount large, heat-producing equipment directly beneath the OEM070.
❑ Do not mount the OEM070 directly above a drive (the drive produces more heat than the OEM070).

Power Supply Connections

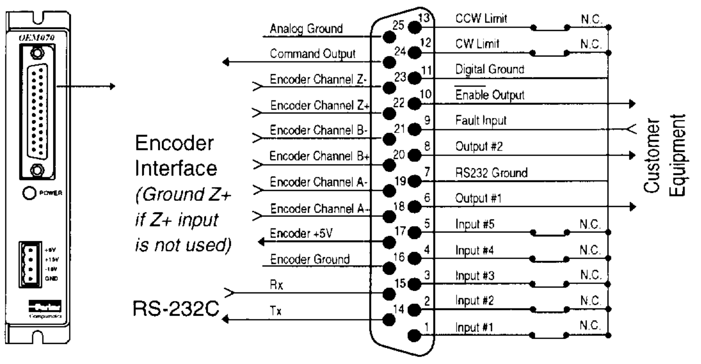
Connect a DC power supply to the 4-pin power input connector on the OEM070. Power supply requirements are:



Input Voltage		Input Current (minimum)
+15 VDC	at	15 mA
-15 VDC	at	15 mA
+5 VDC	at	300 ma

Be sure to connect the power supply ground to the OEM070 ground.

OEM070 Inputs and Outputs

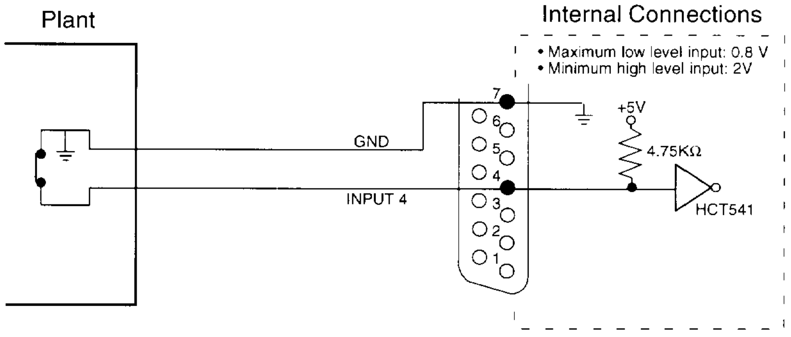


CAUTION

I/O is not OPTO isolated, I/O GND is common to GND. For greater noise immunity, we recommend the use of optical isolation modules. For added noise immunity, this controller has a digital filter; each input must be true for three successive cycles before recognizing a given state.

General Purpose Inputs (Signal 1 – Signal 5)

The OEM070 has 5 general purpose inputs. Each of these inputs may be configured to match the application needs. The figure represents a typical configuration of one of these inputs.


Plant
Internal Connections
• Maximum low level input: 0.8 V
• Minimum high level input: 2V
GND
INPUT 4
+5V
4.75KΩ
HCT541

Internal Connections for Pins 1-5 are identical

General Purpose Input Connected to a Switch

The **IN** command is used to configure the inputs to the following functions:

Trigger Input
The OEM070 can dedicate up to five Trigger inputs. These inputs are pulled up internally. These inputs are used with the Trigger (**TR**) command to control the OEM070's trigger function. Minimum pulse width is 1 ms.

Home Position Input
The OEM070 can dedicate up to one Home input. The Home input allows you to establish a home reference position. This input is not active during power-up. Refer to the Go Home (**GH**) command for more information on setting up and using this function. Minimum pulse width is 1 ms.

Sequence Select Input
The OEM070 can dedicate up to three Sequence Select inputs that allow you to control seven different sequences. Sequences are executed remotely by using one of the following logic patterns in conjunction with the **XP** command.

Sequence #	**Ø**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
SEQ Input #1	Ø	1	Ø	1	Ø	1	Ø	1
SEQ Input #2	Ø	Ø	1	1	Ø	Ø	1	1
SEQ Input #3	Ø	Ø	Ø	Ø	1	1	1	1

Ø = low, pulled to ground
1 = high, 5VDC

Stop or Kill Input
The OEM070 can dedicate up to one Stop and one Kill input. The active state is high. The Stop or Kill input is identical in function to the effect of the **S** or **K** command respectively.

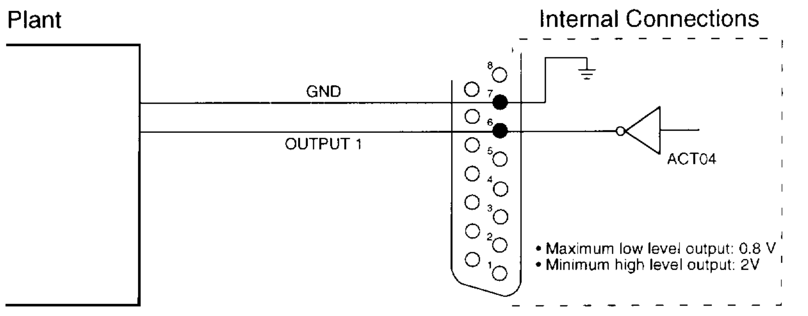
Go Input
The OEM070 can dedicate up to one Go input. The active state is high. The Go input is identical in function to the effect of the GO (**G**) command.

CAUTION
Unless configured otherwise (**SSH** command), the controller will dump the commands following the IN command in the buffer. Please pay special attention to the state of the inputs before entering the IN command.

OUTPUT #1 (SIGNAL 6) AND OUTPUT #2 (SIGNAL 8)

The OEM070 has two dedicated programmable +5 volt outputs. They may be used to signal peripheral devices upon the start or completion of a move. The default state for Outputs #1 and #2 is logic low. Refer to the Output (**O**) command for information on using these outputs.

The next drawing shows the schematic for one of the outputs.

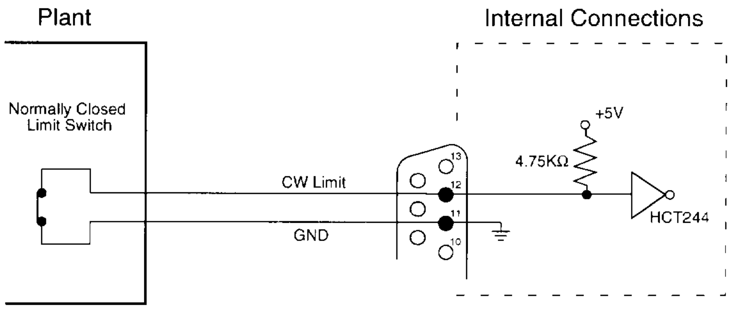

Plant
Internal Connections
GND
OUTPUT 1
8
7
6
5
4
3
2
1
ACT04
• Maximum low level output: 0.8 V
• Minimum high level output: 2V

Internal Connections for Pins 6 and 8 are identical

General Purpose Outputs

CW (SIGNAL 12) & CCW (SIGNAL 13) LIMIT INPUTS

The OEM070 has two dedicated hardware end-of-travel limits (CCW and CW). When you power up the OEM070, these inputs are enabled (high). To test the OEM070 without connecting the CCW and CW limits, you must disable the limits with the **LD3** command. You can use the Limit Switch Status Report (**RA**) and Input Status (**IS**) commands to monitor the limits' status. The following figure represents a typical configuration of these inputs. Minimum pulse width is 1 ms.

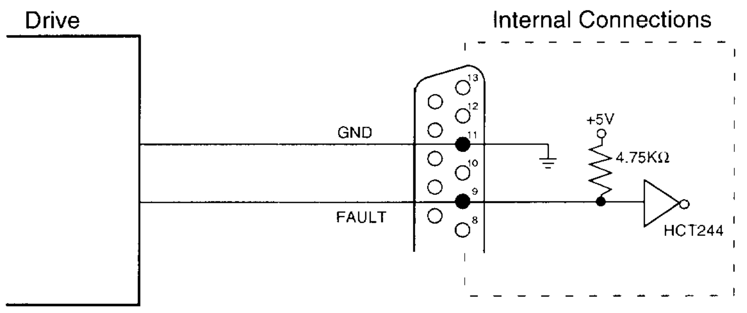


Internal Connections for Pins 12 and 13 are identical

Hardware Limit Switch Inputs

DEDICATED FAULT INPUT (SIGNAL 9)

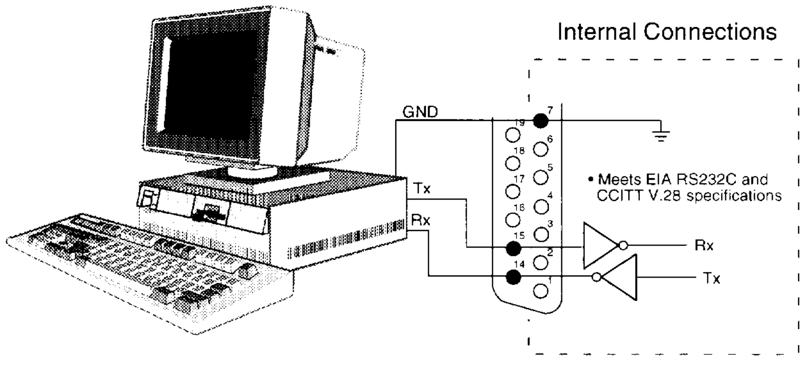
The OEM070 has one dedicated fault input. This input may be used to signal a drive fault to the OEM070. The fault input's active state is a logic high state. The **R** and **RSE** command will report the controller's error conditions. The next figure represents a typical configuration of this input.



Fault Input

RS-232C—TX (SIGNAL 14), RX (SIGNAL 15), AND GROUND (SIGNAL 7)

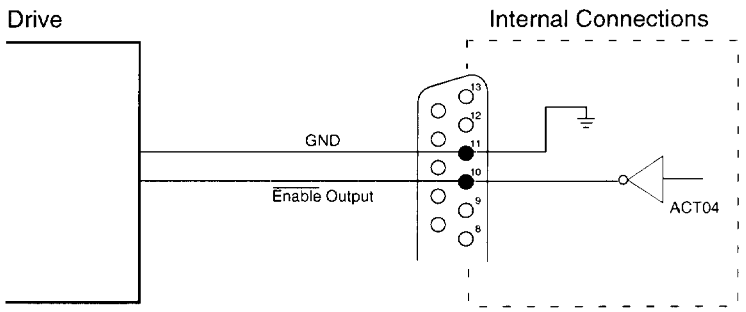
The OEM070 uses RS-232C as its communication medium. The controller does not support handshaking. A typical three-wire (Rx, Tx, and Signal Ground) configuration is used. The figure represents a typical RS-232C configuration.



RS232 Input

Enable Output (Signal 10)

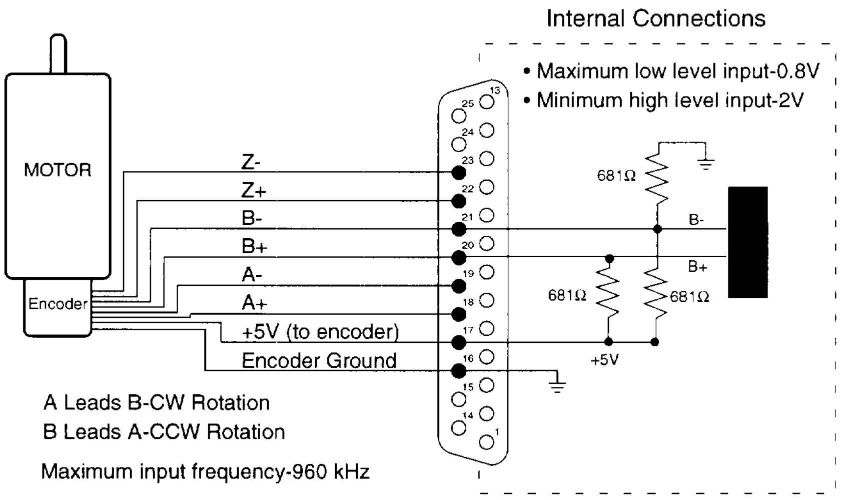
The OEM070 has an enable output that can be connected to the drive. This output can enable and disable the drive. The signal's active level is low. A position error, OFF command or an ST1 command will produce a high signal on this ouput.



Enable Output

Encoder Inputs: +5V,A,B,Z ,GND (Signals 16 - 23)

The OEM070 has six dedicated inputs for use with a differential incremental encoder. These inputs provide the position information for the servo loop.



Internal Connections for CHA (18,19), CHB (20,21) and CHZ (22,23) are identical

OEM070 Encoder Input

CAUTION

If you do not connect an encoder Z-channel output to the OEM070, then you must ground the Z+ input on the OEM070. To do this, connect a jumper wire between the Z+ input (pin 22), and the nearest available ground (pin 16 or pin 11).

ENCODER SPECIFICATIONS: SM & NEOMETRIC MOTORS

The same type of encoder is used on all SM and NeoMetric Series motors. Encoders have either 500 lines ("-D") or 1000 lines ("-E).

Mechanical	
Accuracy	±2 min of arc
Electrical	
Input power	5 VDC ±5%, 135 mA
Operating frequency	100 kHz max
Output device	26LS31
Sink/Source, nominal	20 mA
Suggested user interface	26LS32

SM Motors, Size 16 and Size 23 – Encoder Wiring

Designation	Pin No. MS14-18	Pin No. MS14-18	Not Applicable	Wire Color
Vcc	H	H	—	Red
Ground	G	G	—	Black
CH A+	A	A	—	White
CH A-	B	B	—	Yellow
CH B+	C	C	—	Green
CH B-	D	D	—	Blue
Index +	E	E	—	Orange
Index -	F	F	—	Brown
Shield	NC	NC	—	—

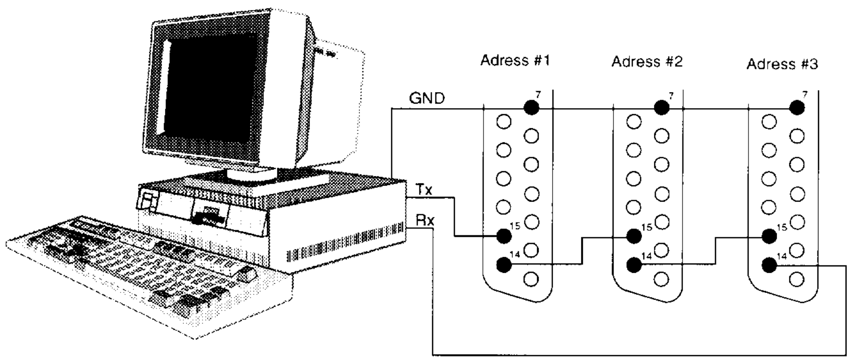
NeoMetric and J Series Motors, Size 070 (Size 031) – Encoder Wiring

Designation	Pin No. MS14-18	Wire Color
Encoder		
+5 VDC	H	Red
Ground	G	Black
CH A+	A	White
CH A-	B	Yellow
CH B+	C	Green
CH B-	D	Blue
Index +	E	Orange
Index -	F	Brown

DAISY CHAINING

You may daisy chain up to 255 OEM070s. Individual drive addresses are set with the # (Address Numbering) command. When daisy chained, the units may be addressed individually or simultaneously. You should establish a unique device address for each OEM070.

Refer to the figure below for OEM070 daisy chain wiring.



Daisy Chain of 3 OEM070s

Commands prefixed with a device address control only the unit specified. Commands without a device address control all units on the daisy chain. The general rule is: *Any command that causes the drive to transmit information from the RS-232C port (such as a status or report command), must be prefixed with a device address.* This prevents daisy chained units from all transmitting at the same time.

Attach device identifiers to the front of the command. The Go (**G**) command instructs all units on the daisy chain to go, while **1G** tells only unit 1 to go.

When you use a single communications port to control more than one OEM070, all units in a daisy chain receive and echo the same commands. Each device executes these commands, unless this command is preceded with an address that differs from the units on the daisy chain. This becomes critical if you instruct any controller/drive to transmit information. To prevent all of the units on the line from responding to a command, you must precede the command with the device address of the designated unit.

CHAPTER ③

Specifications

Section Objectives

The information in this section will enable you to:

❐ Verify and identify product specifications

Parameter	**Value**	
Performance		
Position Range	±1,073,741,823 encoder counts	
Velocity Range	0.01 to 200 rev/s	
Acceleration Range	0.01 to 9999 rev/s^2	
Velocity Accuracy	±0.02% of maximum rate	
Velocity Repeatability	±0.02% of set rate	
Digital Servo Loop		
Update Time	266 microseconds	
Output	12 bit DAC	
Servo Tuning	RS232C command interface	
Tuning Parameters	PID with digital filter	
Power		
DC Input	+15 VDC	15 mA
	-15 VDC	15mA
	+5 VDC	300mA
RS-232C Interface		
Connections	3-wire (Rx, Tx and GND) connection	
Maximum OEM070s daisy chained	255	
Address settings	Via # command	
Parameters	9600 baud, 8 data bits, 1 stop bit, no parity	
Inputs		
Encoder	Two phase differential (recommended) or single ended (+5VDC TTL compatible). 960 kHz maximum frequency.	
General Purpose	5 user defined, 2 hardware limit, fault (0-5V TTL)	

Outputs	
General Purpose	2 user defined, enable (buffered TTL)
Fault Output	Active HIGH:open collector, 24VDC max
Current monitor	1.2 V/A
Environmental	
Operating	0°C to 45°C (32°F - 113°F)
Storage	-30°C to 85°C (-22°F - 185°F)
Humidity	0 to 95% non condensing
Contaminants	The OEM070 is not water-proof, oil-proof, or dust-proof.

CHAPTER ④

Tuning

Section Objectives

The information in this section will enable you to:

❑ Understand the role and interaction of each gain parameter
❑ Tune your system for optimal performance
❑ Configure an In Position Window to determine move completion

Tuning and Performance

The OEM070 uses a digital *Proportional Integral Derivative* (PID) filter to compensate the control loop. For best performance, you must tune the filter's parameters. A properly tuned system will exhibit smooth motor rotation, accurate tracking, and fast settling time.

All tuning is performed via RS232-C communications.

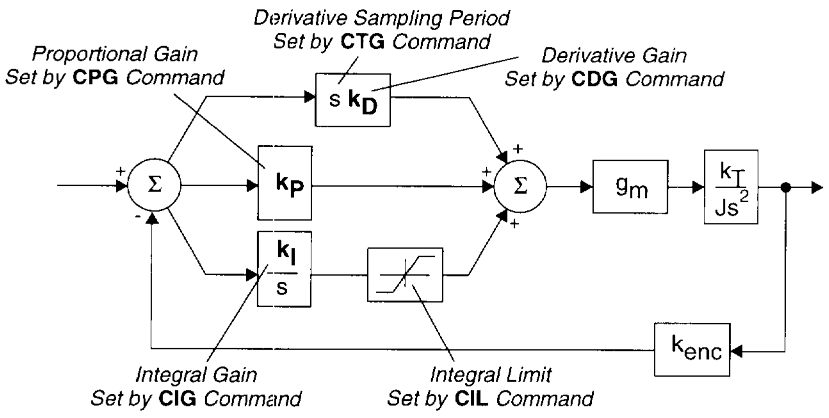
PID Tuning

In the procedure described below, you will systematically vary the tuning parameters until you achieve a move that meets your requirements for accuracy and response time.

The OEM070 generates a move profile based upon the user supplied acceleration, velocity, and distance commands (**A**, **V**, and **D**). At each servo sampling period (every 266 microseconds), the OEM070 calculates the position the motor should reach as it follows the move profile. This is called *commanded position*, and is one of two inputs to a summing node. Position information from the encoder, which is called *actual position*, is the other input to the summing node. During a typical move, actual position will differ from commanded position by at least a few encoder counts. When actual position is subtracted from commanded position at the summing node, an error signal is produced. The error signal is the input to the PID filter.

The position specified by the distance command (**D**) is called the *target position*. During the move, commanded position is not the same as target position. The commanded position is incremented each sampling period. When it finally matches the target position, the move is over.

The servo block diagram is shown below.



As the figure shows, you can adjust five different parameters to tune the PID filter. The relevant commands are:

CPG	Configure Proportional Gain
CDG	Configure Derivative Gain
CTG	Configure Derivative Sampling Period
CIG	Configure Integral Gain
CIL	Configure Integral Limit

To tune the system, you will iteratively increase **CDG** and **CTG** to their optimum values, then increase **CPG** to its optimum value. If necessary, you will also increase **CIG** and **CIL**.

In general, you will set **CDG** and **CPG** as high as possible, and **CIG** as low as possible. Trade-offs between response time, stability, and final position error will dictate the values you select. For loads that vary during operations, you can download new parameters by using buffered versions of the five tuning commands (**BCPG, BCDG, BCTG, BCIG, BCIL**).

WARNING

During servo tuning, the system can undergo accidental and violent movement due to improper gain settings and programming errors. Please use extreme caution while prototyping

Each tuning parameter is described in the following sections.

CPG – Proportional Gain

Proportional gain provides a torque that is directly proportional to the *magnitude* of the error signal. Proportional gain is similar to a spring—the larger the error, the larger the restoring force. It determines the stiffness of the system and affects the following error. High proportional gain gives a stiff, responsive system, but can result in overshoot and oscillation. Damping—provided by derivative gain—can reduce this overshoot and oscillation.

CDG – Derivative Gain; CTG – Derivative Sampling Period

Derivative gain provides a torque that is directly proportional to the *rate of change* of the error signal. The previous error is subtracted from the present error each sampling period. The difference represents the error's instantaneous rate of change, or *derivative*. The difference is multiplied by the value set by the **CDG** command, and the product contributes to the motor control output.

Derivative gain opposes rapid changes in velocity. It will dampen the resonance effects of proportional gain. With higher derivative gain, you can use higher proportional gain.

You can use the **CTG** command to make the derivative sampling period longer than the system's sampling period. The system sampling period—266 µsec—is the period between updates of position error, and cannot be changed. The derivative sampling period is an integer multiple of the system sampling period. It can range from 266 µsec to 68 msec, in increments of 266µsec (for example: CTG0 = 266 µs, CTG1 = 532 µs, CTG2 = 798 µs, etc.).

With a longer derivative sampling period, more time elapses between derivative error measurements. The difference between previous and present error is still multiplied by the CDG value. The product contributes to the motor control output every *system* sampling period, but is only updated every *derivative* sampling period. This gives a more constant derivative term and improves stability. Low velocity systems in particular can benefit from a longer sampling period.

Because of stability considerations, however, the derivative sampling period should be no longer than one tenth of the system mechanical time constant. This means many systems must have low values of **CTG**.

CIG – Integral Gain; CIL – Integral Limit

Integral gain provides a torque that is directly proportional to the sum, over time, of the error values—the *integral* of the error. The controller reads the error value every sampling period, and adds it to the sum of all previous error values. The sum is multiplied by the value set by the **CIG** command, and the product contributes to the motor control output every system sampling period.

Integral gain can remove steady state errors that are due to gravity or a constant static torque. Integral gain can also correct velocity lag that can occur in a constant velocity system.

If error persists during a move, the sum of the error values may be quite high at the end of the move. In this case, the torque provided by the integral gain can also be very high, and can cause an overshoot. This effect is called *integral windup*. You can use **CIL**, the integral limit command, to set a maximum value for integral gain. The integral limit constrains the integral term to values less than or equal to **CIL**, which will reduce the overshoot caused by integral windup.

Tuning Procedure

You can manually tune the OEM070 by varying the tuning parameters while you empirically evaluate the system response. This manual method works well in most applications.

Tuning Procedure

You will achieve best results by making a consistent, repetitive move that is representative of your application.

1 ***Issue a RETURN TO FACTORY SETTINGS command (RFS)***
The RFS command will reset the gains to their default values (CDG240, CTG0, CPG16, CIG2, CIL2, CPE4000)

2 ***Decrease CPG***
Decrease **CPG** to zero (**CPG0**). If your system has very little friction, internal offsets in the drive may cause the motor to run away (spin faster and faster) with **CPG** set to zero. If the motor runs away, issue an **OFF** command. When the motor stops, increase **CPG** by one unit (**CPG1**), and issue an **ON** command. If the motor continues to run away, repeat this procedure—incrementing **CPG** by one unit—until the motor remains stopped.

3 ***Decrease CIG***
Set the integral gain to zero (**CIG0**).

4 ***Increase Derivative Gain (CDG) and Derivative Sampling Period (CTG)***
Determine **CDG** and **CTG** iteratively. Increase **CDG** until the shaft begins high frequency oscillations, then increase **CTG** by one. With a higher **CTG**, the oscillations should be damped. Again increase **CDG** until oscillations occur, then increase **CTG** by one. Repeat this process until **CTG** reaches a value appropriate for the system.

In general, you will want values for **CDG** and **CTG** that are as large as possible, without producing unacceptably high motor vibrations. However, many systems will require a low **CTG** value, to ensure that the derivative sampling period is shorter than one tenth of the system mechanical time constant. Therefore, start with a low **CTG** and gradually increase it, rather than immediately trying a large **CTG** value.

5 ***Increase Proportional Gain (CPG)***
Determine the **CPG** value iteratively. Increase **CPG**, and evaluate the system damping. Repeat until the system is critically damped. You should increase **CPG** to the largest value that does not cause overshoot or ringing. Because proportional gain and derivative gain affect each other, you may need to repeat step 4 and step 5 several times to arrive at optimum values for **CPG** and **CDG**.

6 ***Determine Integral Gain (CIG) and Integral Limit (CIL) Values***
High values for **CIG** will make the system respond quickly, but can cause other problems. In general, you should set **CIG** to the *lowest* value that will correct following errors and static position errors, but not increase overshoot or settling time. In a system without static torque loading, a **CIG** of zero may be appropriate.

CIL limits **CIG**—therefore, before you increase **CIG** to a particular value, you must first increase **CIL** to an equal or higher value.

Tuning Parameters for Apex Series Motors

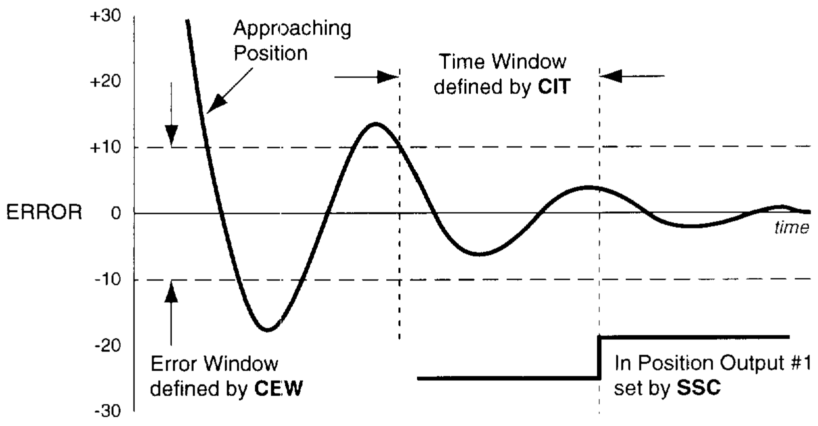
If you use Compumotor's APEX Series Motors, refer to the following table. Set the initial tuning parameters as indicated for your specific motor, then follow the above tuning procedure to fine tune your system.

INITIAL TUNING PARAMETERS FOR APEX MOTORS

MOTOR	*CPG*	*CDG*	*CTG*	*CIG*	*CIL*
APEX604	CPG30	CDG220	CTG2	CIG20	CIL40
APEX605	CPG40	CDG250	CTG3	CIG20	CIL40
APEX606	CPG60	CDG300	CTG5	CIG20	CIL40
APEX610	CPG100	CDG320	CTG4	CIG20	CIL40
APEX620	CPG110	CDG350	CTG3	CIG20	CIL40
APEX630	CPG120	CDG380	CTG4	CIG20	CIL40
APEX635	CPG150	CDG390	CTG5	CIG20	CIL40
APEX640	CPG150	CDG390	CTG5	CIG20	CIL40

Configuring an In Position Window

You can define an In Position Window, and use it to indicate that the preceding move is done. Two commands—**CEW** and **CIT**—determine the height and width of the window. A third command—**SSC**—can turn on output #1 when the In Positon criteria are met.


+30
+20
+10
ERROR 0
-10
-20
-30
Approaching Position
Time Window defined by CIT
time
Error Window defined by CEW
In Position Output #1 set by SSC

As the drawing shows, **CEW** defines the position error window at the end of a move. **CIT** specifies the length of time the motor must be within the error window. The motor is In Position when three conditions are satisfied:

① The controller algorithm is finished (no input position command)

② Position error is less than that specified by the **CEW** command

③ Condition ② above has been true for the length of time specified by the **CIT** command

If **SSC** has been set to 1, output #1 will turn on when these three conditions have been met. You can use output #1 to trigger external hardware from the In Positon condition. The output will stay on until the next move command is issued, such as **GO** or **GO HOME**.

(**NOTE**: If the motor is held (mechanically, or against an end stop), and **CPE** is greater than **CEW**, the motor may become "trapped" between **CPE** and **CEW**: it will not execute the next move. In this rare situation, two things are happening: 1.) **CPE** is not violated, and therefore no position error fault occurs; 2.) in position criteria are not met.

If you were to execute a **1R**, the response would be ***B**, which means the drive is "busy" waiting for the move to be over. Why doesn't the drive force the motor to finish the move? The motor is somehow held. To correct this situation, try touching the motor; this may complete the move, and the drive may execute the next move. Or, execute a **DPA** to read actual position, and verify that the move is not complete. You can also execute a **KILL** to reset the positions, and then do the next series of moves.)

C H A P T E R ⑤

Software Commands

Chapter Objective

❑ Use this chapter as a reference for the function, range, default, and sample use of each command for the OEM Controller.

Command Descriptions

① A—Acceleration *(Example Command)*

② Command Type: Motion	⑥ Valid Software Version: A
③ Syntax: <a>An	⑦ Units: revs/sec^2
④ Range: n = 0.01-999.99	⑧ Default Value: A = 100
⑤ Attributes: Buffered, Savable in Sequence	⑨ See Also: D, G, MR, V
	⑩ Response to aA is *An

① ***Command Mnemonic***
The beginning of each command entry contains the command's mnemonic value and full name.

② ***Command Type***
Set-Up—Set-up commands define application conditions. These commands establish the output data's format from the controller.

Motion—Motion commands affect motor motion, such as acceleration, velocity, distance, go home, stop, direction, mode, etc.

Programming—Programming commands affect programming and program flow for trigger, output, all sequence commands, time delays, pause and continue, enable and disable, loop and end-loop, line feed, carriage return, and backspace.

Status—Status commands respond (report back) with data. These commands instruct the system to send data out from the serial port for host computer use.

③ ***Syntax***
The proper syntax for the command is shown here. The specific parameters associated with the command are also shown. If any of

these parameters are shown in brackets, such as <a>, they are optional. The parameters are described below.

a—An *a* indicates that a device address must accompany the command. Only the device specified by this parameter will receive and execute the command. Valid addresses are 1-255.

n—An *n* represents an integer. An integer may be used to specify a variety of values (acceleration, velocity, etc.).

s—An *s* indicates that a sign character, either positive or negative (+ or -), is required.

x—An *x* represents any character or string of characters.

④ Range

This is the range of valid values that you can specify for n (or any other parameter specified).

⑤ Attributes

This first attribute indicates if the command is *immediate* or *buffered.* The system executes immediate commands as soon as it receives them. Buffered commands are executed in the order that they are received with other buffered commands. Buffered commands can be stored in a sequence.

The second attribute explains how you can save the command.

- Savable in Sequence
- Never Saved
- Automatically Saved

Savable in Sequence commands are saved when they are defined in a sequence (see **XT** command). *Savable in Sequence* commands can be stored in system memory (nonvolatile) and retained when power is removed from the system. A command that is *Never Saved* is executed without being saved into the system's permanent memory . *Automatically Saved* commands are automatically saved into memory upon execution.

⑥ Valid Software Version

This field contains the current revision of the software in which the command resides at the time this user guide was released.

⑦ Units

This field describes what unit of measurement the parameter in the command syntax represents.

⑧ Default Value

The default setting for the command is shown in this box. A command will perform its function with the default setting if you do not provide a value.

⑨ ***See Also***
Commands that are related or similar to the command described are listed here.

⑩ ***Response***
A sample status command and system response are shown. When the command has no response, this field is not shown.

A—Acceleration

❑ Command Type: Motion	❑ Valid Software Version: A
❑ Syntax: <a>An	❑ Units: revs/sec^2
❑ Range: n = 0.01-9999.99	❑ Default Value: A = 100
❑ Attributes: Buffered, Savable in Sequence	❑ See Also: D, G, MR, V
	❑ Response to aA is *An

The Acceleration command specifies the rotary acceleration rate to be used for the next Go (**G**) command. The acceleration remains set until you change it. You do not need to reissue this command for subsequent Go (**G**) commands. Accelerations outside the valid range cause the acceleration to remain at the previous valid **A** setting.

If the Acceleration command is entered with only a device address (**1A**), the controller will respond with the current acceleration value. If a move is commanded without specifying an acceleration rate, the previously commanded acceleration rate will be used. Acceleration cannot be changed on the fly. The minimum acceleration is

Min Accel = Encoder resolution (**ER** command) x .00465

Command	**Description**
A1Ø	Sets acceleration to 10 revs/sec^2
V1Ø	Sets velocity to 10 revs/sec
D2ØØØ	Sets distance to 2,000 encoder counts
G	Executes the move

B—Buffer Status

❑ Command Type: Status	❑ Valid Software Version: A
❑ Syntax: aB	❑ Units: N/A
❑ Range: N/A	❑ Default Value: N/A
❑ Attributes: Immediate, Never Saved	❑ Response to aB is *B or *R
	❑ See Also: BS

The buffer status command will report the status of the command buffer. If the command buffer is empty or less than 95% full, the controller will respond with a ***R**.

The command buffer is 512 bytes long. A ***B** response will be issued if less than 5% of the command buffer is free.

*R = More than 5% of the buffer is free
*B = Less than 5% of the buffer is free

This command is commonly used when a long series of commands will be loaded remotely via RS-232C interface. If the buffer size is exceeded, the extra commands will not be received by the controller until more than 5% of the command buffer is free.

Command	Response
1B	***B** (less than 5% of the command buffer is free)

BCDG—Buffered Configure Derivative Gain

- ❑ Command Type: Set-up
- ❑ Syntax: <a>BCDGn
- ❑ Range: n = 0-32,767
- ❑ Attributes: Buffered Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: 240
- ❑ Response to aBCDG is *CDGn
- ❑ See Also: BCIG, BCPG, BCTG, CIG, CPG, CTG

This buffered command is used for system tuning. This term represents the gain applied to the derivative of the position error—in other words, the rate at which the position error is changing. This gain produces a damping effect similar to velocity feedback.

Refer to *Chapter ④ Tuning* for more information.

Command	Description
BCDG400	Set the derivative gain term to 400
1BCDG	Reports derivative gain term (*CDG400)

BCIG—Buffered Configure Integral Gain

- ❑ Command Type: Set-up
- ❑ Syntax: <a>BCIGn
- ❑ Range: n = 0-32,767
- ❑ Attributes: Buffered Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: 2
- ❑ Response to aBCIG is *CIGn
- ❑ See Also: BCDG, BCPG, BCTG, CDG, CPG, CTG

This buffered command is used for system tuning. This term represents the gain applied to the integral of the position error—the net accumulation of the position error over time. Thus integral gain will contribute when a position error is not being reduced over time, as may be caused by the effects of friction or gravity. This gain will improve overall accuracy but may increase settling time and, if excessive, may cause a low frequency oscillation around the commanded position.

Before you increase **BCIG**, you must first increase the integral limit (**BCIL**) to an equal or higher value.

Refer to *Chapter ④Tuning* for more information.

Command	Description
BCIL40	Set the integral limit term to 40
BCIG10	Set the integral gain term to 10
1BCIG	Reports integral gain term (*CIG10)

BCIL—Buffered Configure Integral Limit

❑ Command Type: Set-up
❑ Syntax: <a>BCILn
❑ Range: n = 0-32,767
❑ Attributes: Buffered Savable in Sequence
❑ Valid Software Version: A
❑ Units: N/A
❑ Default Value: 2
❑ Response to aBCIL is *CILn
❑ See Also: BCDG, BCIL, BCPG, BCTG, CDG, CIL,CPG, CTG

This buffered command is used for system tuning. This term represents the limit applied to the integral gain contribution of the PID equation. A high integral gain with a large inertial load can cause a non-ringing overshoot of the commanded position. By limiting the contribution of integral action, this overshoot can be minimized.

Refer to *Chapter* ④ *Tuning* for more information.

Command	Description
BCIG10	Set the integral gain term to 10
BCIL40	Set the integral limit to 40
1BCIL	Reports integral limit term (*CIL40)

BCPE—Buffered Configure Position Error

❏ Command Type: Set-up	❏ Valid Software Version: A
❏ Syntax: <a>BCPEn	❏ Units: N/A
❏ Range: n = 0-32,767	❏ Default Value: 4000
❏ Attributes: Buffered Savable in Sequence	❏ Response to aBCPE is *CPEn
	❏ See Also: DPE, CPE

This buffered command defines the maximum allowable position or following error. If the actual position error ever exceeds the allowable position error, the controller will generate a fault condition and shut down the drive. If the maximum allowable position error is set to Ø, the function is disabled and no amount of position error will generate the fault condition.

Refer to *Chapter* ④ *Tuning* for more information.

Command	Description
BCPE400	Set the maximum allowable position error to 400 encoder counts
BCPE0	Disable fault generation due to position error
1BCPE	Reports maximum position error setting (*CPE0)

BCPG—Buffered Configure Proportional Gain

❏ Command Type: Set-up	❏ Valid Software Version: A
❏ Syntax: <a>BCPGn	❏ Units: N/A
❏ Range: n = 0-32,767	❏ Default Value: 16
❏ Attributes: Buffered Savable in Sequence	❏ Response to aBCPG is *CPGn
	❏ See Also: BCDG, BCIG, BCTG, CDG, CIG, CTG

This buffered command is used for system tuning. This term represents the gain applied directly to the position error. The proportional gain sets how actively the system will respond to position error. High proportional gain will give a stiff, responsive system, but may result in overshoot and oscillation.

Refer to *Chapter* ④ *Tuning* for more information.

Command	Description
BCPG50	Set the proportional gain term to 50
1BCPG	Reports proportional gain term (*CPG50)

BCTG—Buffered Configure Derivative Sampling Period

- ❑ Command Type: Set-up
- ❑ Syntax: <a>BCTGn
- ❑ Range: n = 0-255
- ❑ Attributes: Buffered Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: 266 µsec
- ❑ Default Value: 0
- ❑ Response to aBCTG is *CTGn
- ❑ See Also: BCDG, BCIG, BCPG, CDG, CIG, CPG

This buffered command is used for system tuning. Use **BCTG** to adjust the derivative sampling period. The *system* sampling period—266 µsec—is the period between updates of position error. The *derivative* sampling period is an integer multiple of the system sampling period. In general, a longer derivative sampling period gives a more constant derivative term and improves stability. Many systems require a low **BCTG** value to prevent oscillations, however. Therefore, start with a low value and increase it incrementally.

Refer to *Chapter ④ Tuning* for more information.

Command	Description
BCTG0	Set the derivative sampling period to 266 µsec
BCTG1	Set the derivative sampling period to 532 µsec
BCTG2	Set the derivative sampling period to 798 µsec
BCTG3	Set the derivative sampling period to 1064 µsec
1BCTG	Reports derivative sampling period (*CTG3)

BS—Buffer Size Status

- ❑ Command Type: Status
- ❑ Syntax: aBS
- ❑ Range: N/A
- ❑ Attributes: Immediate, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ Response to aBS is *n
- ❑ See Also: B

This command reports the number of bytes remaining in the command buffer. When entering long string commands, check the buffer status to be sure that there is enough room in the buffer. Otherwise, commands may be lost. Each character (including delimiters) uses one byte. The range for the response is 0 - 512 bytes.

Command	Response
1BS	***122** (122 bytes available in the buffer)

C—Continue

❑ Command Type: Motion	❑ Valid Software Version: A
❑ Syntax: <a>C	❑ Units: N/A
❑ Range: N/A	❑ Default Value: N/A
❑ Attributes: Immediate, Never Saved	❑ See Also: PS, U

The Continue (**C**) command ends a pause state. It enables your controller to continue executing buffered commands. After you enter a Pause (**PS**) or the Pause and Wait for Continue (**U**) command, you can clear it with a Continue (**C**) command. This command is useful when you want to transmit a string of commands to the buffer before you actually execute them.

Command	Description
MC	Sets move to continuous mode
A1Ø	Sets acceleration to 10 revs/sec^2
V1Ø	Sets velocity to 10 revs/sec
PS	Pauses system until controller receives **C** command
G	Accelerates the motor to 10 revs/sec
C	Continues executing commands in the buffer

The motor will not execute the **G** command until the **C** command is issued

CDG—Configure Derivative Gain

❑ Command Type: Set-up	❑ Valid Software Version: A
❑ Syntax: <a>CDGn	❑ Units: N/A
❑ Range: n = 0-32,767	❑ Default Value: 240
❑ Attributes: Immediate Automatically Saved	❑ Response to aCDG is *CDGn
	❑ See Also: CIG, CPG, CTG, RFS

This command is used for system tuning. This term represents the gain applied to the derivative of the position error, in other words, the rate at which the position error is changing. This gain produces a damping effect similar to velocity feedback.

Refer to *Chapter* ④ *Tuning* for more information.

Command	Description
CDG400	Set the derivative gain term to 400
1CDG	Reports derivative gain term (*CDG400)

CEW—Configure In Position Error Window

❑ Command Type: Set-up	❑ Valid Software Version: A
❑ Syntax: <a>CEWn	❑ Units: encoder counts
❑ Range: n = 0-32,767	❑ Default Value: 50
❑ Attributes: Buffered Savable in Sequence	❑ Response to aCEW is *CEWn
	❑ See Also: CIT, SSC

This command, together with the **CIT** command, can be used to configure an In Position window, which can be used to indicate that the preceding move has terminated.

The In Position condition is met when:

- The controller algorithm has finished (no input position command)
- The CEW condition is met (the position error is less than that specified by the **CEW** command).
- The above condition has been true for the length of time specified by the **CIT** command

The position error range, specified by *n* in **CEW*n***, is the maximum number of encoder counts allowed on either side of the desired position. For example, if *n* = 10, then the In Position window is 20 encoder counts wide.

Output 1 can be configured with the **SSC** command to show the state of the In Position detector. This allows the user to trigger external hardware from the In Position condition.

Refer to *Chapter ④ Tuning* for more information.

Command	**Description**
CEW10	Configure an In Position Error Window ±10 encoder counts either side of desired position
1CEW	Reports ± number of encoder counts (*CEW10)

CIG—Configure Integral Gain

❑ Command Type: Set-up	❑ Valid Software Version: A
❑ Syntax: <a>CIGn	❑ Units: N/A
❑ Range: n = 0-32,767	❑ Default Value: 2
❑ Attributes: Immediate Automatically Saved	❑ Response to aCIG is *CIGn
	❑ See Also: CIL, CDG, CPG, CTG, RFS

This command is used for system tuning. This term represents the gain applied to the integral of the position error—the net accumulation of the position error over time. Thus integral gain will contribute when a position error is not being reduced over time, as may be caused by the effects of friction or gravity. This gain will improve overall accuracy but may increase settling time and, if excessive, may cause a low frequency oscillation around the commanded position.

Before you increase **CIG**, you must first increase the integral limit (**CIL**) to an equal or higher value.

Refer to *Chapter* ④ *Tuning* for more information.

Command	Description
CIL40	Set the integral limit term to 40
CIG10	Set the integral gain term to 10
1CIG	Reports integral gain term (*CIG10)

CIL—Configure Integral Limit

❑ Command Type: Set-up	❑ Valid Software Version: A
❑ Syntax: <a>CILn	❑ Units: N/A
❑ Range: n = 0-32,767	❑ Default Value: 2
❑ Attributes: Immediate Automatically Saved	❑ Response to aCIL is *CILn
	❑ See Also: CIG,CDG, CPG, CTG, RFS

This command is used for system tuning. This term represents the limit applied to the integral gain contribution of the PID equation. A high integral gain with a large inertial load can cause a non-ringing overshoot of the commanded position. By limiting the contribution of integral action, this overshoot can be minimized.

Refer to *Chapter* ④ *Tuning* for more information.

Command	Description
CIG10	Set the integral gain term to 10
CIL40	Set the integral limit to 40
1CIL	Reports integral limit term (*CIL40)

CIT—Configure In Position Time

❑ Command Type: Setup	❑ Valid Software Version: A
❑ Syntax: <a>CITn	❑ Units: Milliseconds
❑ Range: n = 0-32,767	❑ Default Value: 20
❑ Attributes: Buffered Savable in Sequence	❑ Response to aCIT is *CITn
	❑ See Also: CEW, SSC

This command is used to specify the time period that the servo is to be within the In Position window before the "In Position" signal is generated. The range is 0 to 32,767, and is the number of milliseconds to be used as the testing time frame.

Refer to *Chapter* ④ *Tuning* for more information.

Command	**Description**
1SSC1	Set output 1 as "In Position"
1CIT30	Set "In Position" time to 30ms
1CEW20	Set allowable position error to ±20 encoder counts

CPE—Configure Position Error

❑ Command Type: Set-up	❑ Valid Software Version: A
❑ Syntax: <a>CPEn	❑ Units: N/A
❑ Range: n = 0-32,767	❑ Default Value: 4000
❑ Attributes: Immediate Automatically Saved	❑ Response to aCPE is *CPEn
	❑ See Also: DPE, RFS

This command defines the maximum allowable position or following error. If the actual position error ever exceeds the allowable position error, the controller will generate a fault condition and shut down the drive. If the maximum allowable position error is set to Ø, the function is disabled and no amount of position error will generate the fault condition.

Refer to *Chapter* ④ *Tuning* for more information.

Command	**Description**
CPE400	Set the maximum allowable position error to 400 encoder counts
CPE0	Disable fault generation due to position error
1CPE	Reports maximum position error setting (*CPE0)

CPG—Configure Proportional Gain

- ❏ Command Type: Set-up
- ❏ Syntax: <a>CPGn
- ❏ Range: n = 0-32,767
- ❏ Attributes: Immediate
  Automatically Saved
- ❏ Valid Software Version: A
- ❏ Units: N/A
- ❏ Default Value: 16
- ❏ Response to aCPG is *CPGn
- ❏ See Also: CDG, CIG, CTG, RFS

This command is used for system tuning. This term represents the gain applied directly to the position error. The proportional gain sets how active the system will be to position error. High proportional gain will give a stiff, responsive system, but may result in overshoot and oscillation.

Refer to *Chapter* ④ *Tuning* for more information.

Command	**Description**
CPG50	Set the proportional gain term to 50
1CPG	Reports proportional gain term (*CPG50)

CR—Carriage Return

- ❏ Command Type: Programming
- ❏ Syntax: <a>CR
- ❏ Range: N/A
- ❏ Attributes: Buffered
  Savable in Sequence
- ❏ Valid Software Version: A
- ❏ Units: N/A
- ❏ Default Value: N/A
- ❏ See Also: LF

The Carriage Return (**CR**) command determines when the controller has reached a particular point in the execution buffer. When the controller reaches this command in the buffer, it responds by issuing a carriage return (**ASCII 13**) over its interface back to the host computer or terminal. If you place the CR command after a Go (**G**) command, it indicates when a move is complete. If you place the CR command after a Trigger (**TR**) command, it indicates when the trigger condition is met.

You can use Carriage Return (**CR**) and Line Feed (**LF**) commands with the Quote (") command to display multiple-line messages via the RS-232C interface.

Command	**Description**
MN	Sets mode to preset mode
A5Ø	Sets acceleration to 50 revs/sec^2
V5	Sets Velocity to 5 revs/sec

D5ØØØ	Sets distance to 5,000 encoder counts
G	Executes the move (Go)
1CR	Sends a carriage return after move is completed

The motor moves 5,000 encoder counts. When the motor stops, the controller sends a carriage return over its interface.

CTG—Configure Derivative Sampling Period

- ❑ Command Type: Set-up
- ❑ Syntax: <a>CTGn
- ❑ Range: n = 0-255
- ❑ Attributes: Immediate Automatically Saved
- ❑ Valid Software Version: A
- ❑ Units: 266 μsec
- ❑ Default Value: 0
- ❑ Response to aCTG is *CTGn
- ❑ See Also: CDG, CIG, CPG, RFS

This command is used for system tuning. Use **CTG** to adjust the derivative sampling period. The *system* sampling period—266 μsec—is the period between updates of position error. The *derivative* sampling period is an integer multiple of the system sampling period. In general, a longer derivative sampling period gives a more constant derivative term and improves stability. Many systems require a low **CTG** value to prevent oscillations, however. Therefore, start with a low value and increase it incrementally.

Refer to *Chapter ④ Tuning* for more information.

Command	Description
CTG0	Set the derivative sampling period to 266 μsec
CTG1	Set the derivative sampling period to 532 μsec
CTG2	Set the derivative sampling period to 798 μsec
CTG3	Set the derivative sampling period to 1064 μsec
1CTG	Reports derivative sampling period (*CTG3)

D—Distance

- ❑ Command Type: Motion
- ❑ Syntax: <a>Dn
- ❑ Range: n = ±1,073,741,823
- ❑ Attributes: Buffered Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: encoder counts
- ❑ Default Value: 4000
- ❑ Response to aD is *Dn
- ❑ See Also: A, G, MN, MPA, MPI, V, H

The Distance (**D**) command defines either the number of encoder counts the motor will move or the absolute position it will seek after a Go (**G**) command is entered. In incremental mode (**MPI**), the value set with the Distance (**D**) command will be the distance (in encoder counts) the motor will travel on all subsequent Go (**G**) commands. In absolute mode (**MPA**), the distance moved by the motor will be the difference between the present motor position and the position (referenced to the zero position) set with the **D** command. In either mode, the direction is controlled by the direction (+ or -) that precedes the distance value. The **D** command has no effect on continuous moves (**MC**).

In Mode Normal (**MN**) the position may not exceed the maximum distance range of 1,073,741,823 encoder counts. If the motor approaches the absolute maximum (plus or minus), the controller will not execute any **GO** commands that would cause the distance to exceed the absolute maximum. To proceed further, use the **PZ** command to reset the absolute counter to zero, and then resume operations.

If **D** is entered with only a device address (**1D**), the controller will respond with the current distance value. If a move is commanded without specifying a distance, the previously commanded distance will be applied to the move.

Command	Description
MN	Sets controller to Normal mode
MPI	Sets controller to Incremental Position mode
A1Ø	Sets acceleration to 10 revs/sec^2
V1Ø	Sets velocity to 10 revs/sec
D4ØØØ	Sets distance to 4000 encoder counts
G	Executes the move

A servo motor with a 4000 count encoder will travel 1 rev (CW) after G is issued.

DPA—Display Position Actual

- ❑ Command Type: Status
- ❑ Syntax: aDPA
- ❑ Range: n = ±1,073,741,823
- ❑ Attributes: Immediate Device Specific, Never saved
- ❑ Valid Software Version: A
- ❑ Units: Encoder counts
- ❑ Default Value: NA
- ❑ Response to aDPA is *DPAn
- ❑ See Also: D, PZ

Single display of actual motor position as measured by the encoder. This command is functionally identical to the **PX** command. The response is the position in encoder counts as referenced to the last power reset or position zero command (**PZ**).

Command	Description
1DPA	Report the actual motor position of axis 1 (*+0004000000)

DPE—Display Position Error

- ❑ Command Type: Status
- ❑ Syntax: aDPE
- ❑ Range: n = ±2,147,483,646
- ❑ Attributes: Immediate Device Specific, Never saved
- ❑ Valid Software Version: A
- ❑ Units: Encoder counts
- ❑ Default Value: NA
- ❑ Response to aDPE is *DPEn
- ❑ See Also: D, DPA

Single display of position error. The response is the difference in encoder counts of the actual motor position and the commanded motor position. This information is used by the control algorithm to control the torque command to the motor. It is normal for the position error to be present during a move but large position errors are usually caused by inappropriate gain settings. Issuing a DPE command before the motor has settled into final position will also result in larger position errors.

Command	Description
1DPE	Report the position error of axis 1 (*+0000000005).

DVA—Display Velocity Actual

- ❑ Command Type: Status
- ❑ Syntax: aDVA
- ❑ Range: n = 0 – 20000
- ❑ Attributes: Immediate Device Specific, Never saved
- ❑ Valid Software Version: A
- ❑ Units: (rev/sec)*100
- ❑ Default Value: NA
- ❑ Response to aDVA is *DVAn
- ❑ See Also: DPA, V

Single display of actual motor velocity in revolutions per sec. There is an implied decimal point before the last two digits.

Command	Description
MC	Mode continuous
V2	Velocity of 2 rev/sec
G	Go
1DVA	Report the actual motor velocity of axis 1 (*00200)

E—Enable Communications

❑ Command Type: Programming
❑ Syntax: <a>E
❑ Range: N/A
❑ Attributes: Immediate
Never Saved
❑ Valid Software Version: A
❑ Units: N/A
❑ Default Value: Enabled
❑ See Also: F

The Enable Communications (**E**) command allows the controller to accept commands over the serial communications interface. You can re-enable the communications interface with this command if you had previously disabled the RS-232C interface with the Disable Communications Interface (**F**) command. If several units are using the same communications interface, the **E** and **F** commands can help streamline programming.

Command	Description
F	Disables all units (axes) on the communications interface
1E	Enables serial interface on Device 1
4E	Enables serial interface on Device 4
A1Ø	Set acceleration to 10 revs/sec^2
V5	Set velocity to 5 revs/sec
D5ØØØ	Sets distance to 5000 encoder counts
G	Executes the move (Go—only axes 1 & 4 will move)

ER—Encoder Resolution

❑ Command Type: Set-up
❑ Syntax: <a>ERn
❑ Range: n = 400 - 65,532
❑ Attributes: Buffered,
Savable in Sequence
❑ Valid Software Version: A
❑ Units: n = encoder counts/rev
❑ Default Value: 4000
❑ Response to aER is *ERn
❑ See Also: CPE

The encoder resolution defines the number of encoder counts the controller will see per revolution of the motor. The number of lines on an encoder should be multiplied by 4 to arrive at the correct ER value per revolution of the motor. (In other words, one line of an encoder will produce 4 encoder counts due to quadrature detection)

Command	Description
ER4ØØØ	Sets encoder resolution to 4000 encoder counts per 1 motor revolution
1ER	Reports Encoder Resolution (*ER4000)

F—Disable Communications

- ❑ Command Type: Programming
- ❑ Syntax: <a>F
- ❑ Range: N/A
- ❑ Attributes: Immediate
  Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: None
- ❑ See Also: E

The Disable Communications (**F**) command is useful when you are programming multiple units on a single interface. Axes that are not intended to process global commands should be disabled using device specific **F** commands. This allows you to program other units without specifying a device identifier on every command. If you do not disable other units in a daisy chain, uploading programs may cause other units on the daisy chain to perform uploaded commands.

Command	Description
1F	Disables the communications interface on unit #1
3F	Disables the communications interface on unit #3
G	All controllers (except 1 & 3) will execute a move

G—Go

- ❑ Command Type: Motion
- ❑ Syntax: <a>G
- ❑ Range: N/A
- ❑ Attributes: Buffered
  Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: None
- ❑ See Also: A, D, MC, MN, S, V

The Go (**G**) command instructs the motor to make a move using motion parameters that you have previously entered. You do not have to re-enter Acceleration (**A**), Velocity (**V**), Distance (**D**), or the current mode (**MN** or **MC**) commands with each **G** (if you do not need to change them).

In the Normal mode (**MN**), moves can be either incremental or absolute. In the Incremental Preset mode (**MPI**), a **G** will initiate the move distance you specified with the **D** command. A **G** command in the Absolute Preset mode (**MPA**) will not cause motion unless the distance (**D**) value is different from the present motor position (**PR**) .

In Continuous mode (**MC**), you only need to enter the Acceleration (**A**) and Velocity (**V**) commands prior to **G**. The system ignores the Distance (**D**) command in this mode.

No motor motion will occur until you enter **G** in either the Normal (**MN**) or Continuous (**MC**) modes.If motion does not occur with **G**, an activated end-of-travel limit switch may be on. Check the hard limit switches or use the limit disable command (**LD3**—see **RA** command also). The next buffered command will not be executed until after the move is completed.

Command	Description
MN	Sets Normal mode (preset)
A5	Sets acceleration to 5 revs/sec^2
V1Ø	Sets velocity to 10 revs/sec
D2ØØØ	Sets distance to 2,000 encoder counts
G	Executes the move (Go)
A1	Sets acceleration to 1 rev/sec^2
G	Executes the move (Go)

Assuming the controller is in Incremental Preset mode, the motor turns 2,000 encoder counts and repeats the 2,000 count move using the new acceleration value of 1 rev/sec^2 (Total distance moved = 4,000 encoder counts).

GH—Go Home

- ❑ Command Type: Motion
- ❑ Syntax: <a>GHn
- ❑ Range: n = ± .01 - 200
- ❑ Attributes: Buffered Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: Revs/sec
- ❑ Default Value: n = 0
- ❑ See Also: OS, RC, V,IN

The Go Home (**GH**) command instructs the Controller to search for an absolute home position in the positive or negative (+ or -) direction. It defines home as the position where the home limit signal changes states. To use the GH command, one of the general purpose inputs (pins 1-5) must be configured as a home input (**IN** command).

Homing can be as simple as decelerating to a stop when the edge of the home limit is detected. By using the **OS** commands, the homing process can be tailored to meet the application needs.

OSB—*Back up to home* makes homing more repeatable by backing off the home switch and re-approaching at low speed. The final approach to home switch is always form the CW direction.

OSC—*Define active state of home* allows the use of a normally closed or normally open limit switch.

OSD—*Enable Z Channel for home* uses the Z channel of the encoder, in conjunction with a home switch, to determine the final home position. The Z channel is a more accurate home position than the edge of a switch.

OSH—*Reference edge of home switch* allows either edge of the home switch to be used as the final edge position.

The Controller will reverse direction if an end-of-travel limit is activated while searching for home. However, if a second end-of-travel limit is encountered in the new direction, the Go Home procedure will stop and the operation will be aborted. The Homing Status (**RC**) command will indicate if the homing operation was successful.

The Go Home command will use acceleration set by the A command. The Go Home velocity will not affect the standard velocity (**V**) value.

Command	Description
INE1	Configure input #1 as home input
OSB1	Back up the home switch
OSD1	Reference Z channel as final home
GH-2	The motor moves CCW at 2 revs/sec and looks for the Home Limit input to go active.

Since the motor is turning CCW, it will see the CW edge of the limit first. It will decelerate to a stop and turn at 0.1 rev/sec in the CW direction until it detects the Z channel.

^H—Delete

- ❑ Command Type: Programming
- ❑ Syntax: ^H
- ❑ Range: N/A
- ❑ Attributes: Immediate
  Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: None

This command allows you to delete the last character that you entered. The **^H** command will not prevent execution of an immediate command. A new character may be entered at that position to replace the existing character. (**^H** indicates that the Ctrl key is held down when the H key is pressed.) This command prompts the controller to backup one character in the command buffer, regardless of what appears on the terminal. On some terminals, the Ctrl and the left arrow (<—) keys produce the same character.

This command will *not* delete characters beyond the last delimiter issued. Pressing the delete key will not delete the previous character.

H—Set Direction

❑ Command Type: Programming
❑ Syntax: <a>H(s)
❑ Range: s = + or -
❑ Attributes: Buffered
Savable in Sequence
❑ Valid Software Version: A
❑ Units: N/A
❑ Default Value: +
❑ See Also: D

The Set Direction (**H**) command changes or defines the direction of the next move that the system will execute. This command does not affect moves already in progress.

H+ = Sets move to CW direction
H– = Sets move to CCW direction
H = Changes direction from the previous setting

In preset moves, a Distance (**D**) command entered after the **H** command overrides the direction set by the **H** command. In Continuous mode (**MC**), only the **H** command can set the direction of motion.

Command	Description
MN	Sets Normal mode
A5	Sets acceleration to 5 revs/sec^2
V5	Sets velocity to 5 revs/sec
D4ØØØ	Sets distance to 4,000 encoder counts
G	Executes the move (Go) in CW direction
H	Reverses direction
G	Executes the move (Go) in CCW direction
MC	Sets mode to continuous
H+	Sets direction to CW
G	Moves continuously in CW direction

IN—Set Input Function

❑ Command Type: Set-up
❑ Syntax: aINxn
❑ Range: x = A – F, n = 1 – 5
❑ Attributes: Buffered
Savable in sequence
❑ Valid Software Version: A
❑ Units: NA
❑ Default Value: AAAAA
❑ Response to aIN is *xxxxx
❑ See Also: IS, TR, XP, #, K, GH

This command configures the function of each of the 5 general purpose inputs. You can configure each input to perform one of the following functions:

Function A—*Trigger Input*
Used with the **TR** command as a comparison input. The TR command defines active high, low, or "don't care." Up to 5 inputs can be configured as trigger inputs.

Function B—*Sequence Select Input*
Executes predefined sequences from remote inputs based on the **XP** command. Active state (sequence selected) is high. Up to 3 inputs can be configured as sequence select inputs.

Function C—*Kill Input*
Immediately halts execution of the move. Same as kill (**K**) command. Active state (kill initiated) is high. Up to one input can be configured as a kill input.

Function D—*Stop Input*
Decelerates the motor to a stop using the value specified in the Acceleration (**A**) command (dumps the sequence or command buffer if configured by **SSHØ**). Same as Stop (**S**) command. Active state (stop initiated) is high. Up to one input can be configured as a stop input.

Function E—*Home Input*
Defines the motor home or origin position as executed by the **GH** command and configures with the Homing Function (**OS**) commands. **OSC** determines the active state of the input. Up to one input can be configured as a home input.

Function F—*Go Input*
Accelerates the motor to a velocity and distance specified in the Acceleration (**A**), Velocity (**V**), and Distance (**D**) commands. Same as Go (**G**) command. Active state (go initiated) is high. Up to one input can be configured as a go input.

Some of the functions (stop, kill, home, go) can only have one input configured to that function. If you try to configure another input to that function, the controller will not recognize the new function and revert back to the previous definition. For example, if input 1 is a kill function, and you want input 5 to be the kill function, you must first change input 1 to another function.

Function *B* (sequence select) can configure up to three inputs as sequence select inputs. If you try to configure more than three inputs, the controller will only recognize the first three. It will revert back to the previous definitions for the additional inputs.

Command	Description
1INA1	Configure input one as trigger input 1
1INB2	Configure input two as sequence select input 1
1INB3	Configure input three as sequence select input 2
1INC4	Configure input four as kill input
1INF5	Configure input five as go input
1IN	Reports input configuration (*ABBCF)

IS—Input Status

❑ Command Type: Status	❑ Valid Software Version: A
❑ Syntax: aIS	❑ Units: N/A
❑ Range: N/A	❑ Default Value: N/A
❑ Attributes: Immediate Never Saved	❑ See Also: IN, LD, RSE
	❑ Response to aIS is *nnnnnnnn

This command reports the status of all hardware inputs. The response is 8 ASCII digits (Ø or 1) corresponding to the following I/O bits:

1—IN1 (Ø = Low, 1 = High)
2—IN2 (Ø = Low, 1 = High)
3—IN3 (Ø = Low, 1 = High)
4—IN4 (Ø = Low, 1 = High)
5—IN5 (Ø = Low, 1 = High)
6—CW limit (Ø = Low, 1 = High)
7—CCW limit (Ø = Low, 1 = High)
8—Fault (Ø = Low, 1 = High)

This is <u>not</u> a software status. It will report the actual hardware status of the inputs. **IS** can help you troubleshoot an application, to verify that limit switches, trigger inputs and home switches work.

Command	Response
2IS	***ØØØ1ØØØ1** (The input status of device 2 is reported: I/O bits 1-3 and 5-7 are low (grounded), and I/O bits 4 (IN4), and 8 (Fault), are high)

K—Kill

❑ Command Type: Motion	❑ Valid Software Version: A
❑ Syntax: <a>K	❑ Units: N/A
❑ Range: N/A	❑ Default Value: N/A
❑ Attributes: Immediate Never Saved	❑ See Also: IN,S

This command causes commanded motion to cease immediately. There is *NO* deceleration of the motor. The motor will servo around the position where the kill was entered. The load could be driven past limit switches and cause damage to the mechanism and possibly to the operation. In addition to stopping the motor, the **K** command will terminate a loop, end a time delay, abort down-loading a sequence **(XD)**, and clear the command buffer.

WARNING

The Kill (**K**) command is not an emergency stop. The motor is not disabled. Motion caused by instability or incorrect wiring will not be stopped. An emergency stop should cut power to the amplifier or interrupt the hardware enable input (pin 10), and mechanically prevent the motor from turning.

Command	Description
A5	Sets acceleration to 5 revs/sec^2
V2	Sets velocity to 2 revs/sec
MC	Sets mode to continuous
G	Executes the move (Go)
K	Stops the motor instantly

L—Loop

- ❑ Command Type: Programming
- ❑ Syntax: <a>Ln
- ❑ Range: n = 0 - 65,535
- ❑ Attributes: Buffered Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: number of loops
- ❑ Default Value: None
- ❑ See Also: C, N, U, Y

When you combine the Loop (**L**) command with the End-of-Loop (**N**) command, all of the commands between **L** and **N** will be repeated the number of times indicated by n. If you enter **L** without a value specified for n, or with a Ø, subsequent commands will be repeated continuously. If you specify a value greater than 65,535, the loop will be repeated continuously.

The **N** command prompts the controller to proceed with further commands after the designated number of loops have been executed. The **Y** command stops loop execution after completing the current loop cycle. The Immediate Pause (**U**) command allows you to temporarily halt loop execution after completing the current loop cycle. You can use the Continue (**C**) command to resume loop execution.

Command	Description
L5	Loop 5 times
A5	Sets acceleration to 5 revs/sec^2
V1Ø	Sets velocity to 10 revs/sec
D1ØØØØ	Sets distance to 10,000 encoder counts
G	Executes the move (Go)
N	End of loop

The commands in the loop will be executed 5 times.

LD—Limit Disable

❑ Command Type: Set-Up	❑ Valid Software Version: A
❑ Syntax: <a>LDn	❑ Units: See Below
❑ Range: n = 0 - 3	❑ Default Value: Ø
❑ Attributes: Buffered Savable in Sequence	❑ See Also: RA

The Limit Disable (**LD**) command allows you to enable/disable the end-of-travel limit switch protection. The **LDØ** condition does not allow the motor to turn without properly installing the limit inputs. If you want motion without wiring the limits, you must issue **LD3**. For machine and operator safety, hardware limits are highly recommended.

- Enable CCW and CW limits—***n = Ø (Default)***
- Disable CW limit—n = 1
- Disable CCW limit—n = 2
- Disable CCW and CW limits—n = 3

Command	**Description**
1LDØ	Enables CW and CCW limits. The motor will move only if the limit inputs are bypassed or connected to normally-closed limit switches.
1LD3	Allows you to make any move, regardless of the limit input state.

LF—Line Feed

- ❑ Command Type: Programming
- ❑ Syntax: <a>LF
- ❑ Range: N/A
- ❑ Attributes: Buffered
  Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: CR

When you issue the Line Feed (**LF**) command, the system transmits a line feed character over the communications link. When the controller reaches this command in the buffer, it responds by issuing a line feed (ASCII 10) over its interface back to the host computer. If you place the **LF** command after a Go (**G**) command, it indicates when a move is complete. If you place the **LF** command after a Trigger (**TR**) command, it indicates when the trigger condition is met.

You can use the Carriage Return (**CR**) and **LF** commands with the Quote (") command to display multiple-line messages via the RS-232C interface.

Command	Description
A5	Sets acceleration to 5 revs/sec^2
V5	Sets velocity to 5 revs/sec
D15ØØØ	Sets distance to 15,000 encoder counts
G	Executes the move (Go)
1LF	Transmits a line feed character over the communications interface after the move is completed

MC—Mode Continuous

- ❑ Command Type: Motion
- ❑ Syntax: <a>MC
- ❑ Range: N/A
- ❑ Attributes: Buffered
  Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Status: Inactive
- ❑ See Also: MN, T, TR, V

The Mode Continuous (**MC**) command causes subsequent moves to ignore any distance parameter and move continuously. You can clear the **MC** command with the Move Normal (**MN**) command.

The controller uses the previously defined Acceleration (**A**) and Velocity (**V**) commands to reach continuous velocity. Using the Time Delay (**T**), Trigger (**TR**), and Velocity (**V**) commands, you can achieve basic velocity profiling. After a new parameter is entered a Go (**G**) command is required. Acceleration (**A**) cannot be changed on the fly.

Command	Description
MC	Sets mode to continuous
A5	Sets acceleration to 5 revs/sec^2
V5	Sets velocity to 5 revs/sec
G	Executes the move (Go)
T1Ø	Move at 5 revs/sec for 10 seconds
V7	Set velocity to 7 revs/sec
G	Change velocity to 7 revs/sec
T1Ø	Move at 7 revs/sec for 10 seconds
VØ	Set velocity to 0 rev/sec (stop)
G	Executes the **VØ** command

The motor turns at 5 revs/sec for 10 seconds, then moves at 7 revs/sec for 10 seconds before decelerating to a stop.

MN—Mode Normal

- ❑ Command Type: Motion
- ❑ Syntax: <a>MN
- ❑ Range: N/A
- ❑ Attributes: Buffered
  Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Status: Active
- ❑ See Also: A, D, G, MC, MPA, MPI

The Mode Normal (**MN**) command sets the positioning mode to preset. In Mode Normal, the motor will move the distance specified with the last distance (**D**) command. To define the complete move profile, you must define Acceleration (**A**), Velocity (**V**), and the Distance (**D**). The **MN** command is used to change the mode of operation from Mode Continuous (**MC**) back to normal or preset. To use the **MPA** or **MPI** command, you must be in Mode Normal (**MN**).

Command	Description
MN	Set positioning mode to preset
A5	Set acceleration to 5 revs/sec^2
V5	Set velocity to 5 revs/sec
D1ØØØ	Set distance to 1,000 encoder counts
G	Executes the move (Go)

Motor turns 1,000 encoder counts CW after the **G** command is issued.

MPA—Mode Position Absolute

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>MPA
- ❑ Range: N/A
- ❑ Attributes: Buffered
  Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Status: Inactive
- ❑ See Also: D, MN, MPI, PZ

This command sets the positioning mode to absolute. In this mode all move distances are referenced to absolute zero. In Mode Position Absolute (**MPA**), giving two consecutive Go (**G**) commands will cause the motor to move only once, since the motor will have achieved its desired absolute position at the end of the first move.

MPA is most useful in applications that require moves to specific locations while keeping track of the beginning position.

You can set the absolute counter to zero by cycling power or issuing a Position Zero (**PZ**) command. You must be in Normal mode (**MN**) to use this command. In continuous mode (**MC**), **MPA** is ignored.

Command	Description
MN	Sets Normal mode (preset)
PZ	Resets absolute counter to zero
MPA	Sets position mode absolute
A5	Sets acceleration to 5 revs/sec^2
V1Ø	Sets velocity to 10 revs/sec
D4ØØØØ	Sets destination to absolute position 40,000
G	Motor will move to absolute position 40,000
D1ØØØØ	Sets destination to absolute position +10,000
G	Motor will move to absolute position +10,000

The motor will move 40,000 encoder counts in the CW direction (if starting from position Ø) and then move 30,000 encoder counts in the CCW direction to reach the absolute position 10,000.

MPI—Mode Position Incremental

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>MPI
- ❑ Range: N/A
- ❑ Attributes: Buffered Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Status: Active
- ❑ See Also: D, MN, MPA

This command sets the positioning mode to incremental. In incremental mode all move distances specified with the Distance (**D**) command will be referenced to the current position. Mode Position Incremental (**MPI**) is most useful in applications that require repetitive movements, such as feed to length applications.

You must be in normal mode (**MN**) to use this command. In continuous mode (**MC**), this command is ignored.

Command	Description
MN	Set positioning mode normal (preset)
MPI	Set positioning mode incremental
A5	Sets acceleration to 5 revs/sec^2

V1Ø	Sets velocity to 10 revs/sec
D1ØØØØ	Sets distance of move to 10,000 encoder counts
G	Move 10,000 encoder counts CW
G	Move another 10,000 encoder counts CW

The motor moves 10,000 encoder counts CW after each **G** command (total move is 20,000 encoder counts).

N—End of Loop

- ❑ Command Type: Programming
- ❑ Syntax: <a>N
- ❑ Range: N/A
- ❑ Attributes: Buffered Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: C, L, PS, U

This command marks the end of a loop. You must use this command in conjunction with the Loop (**L**) command. All buffered commands that you enter between the **L** and **N** commands are executed as many times as the number that you enter following the **L**

Command	Description
MN	Sets move to Normal mode
A5	Sets acceleration to 5 revs/sec^2
V5	Sets velocity to 5 revs/sec
D1ØØØØ	Sets move distance to 10,000 encoder counts
L5	Loops the following commands five times
G	Executes the move (Go)
N	Ends the loop

The move will be executed five times.

OFF—De-Energize Drive

- ❑ Command Type: Programming
- ❑ Syntax: <a>OFF
- ❑ Range: N/A
- ❑ Attributes: Immediate Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: ST, ON

The OFF command immmediately disables the drive through the enable output, which would be connected to the enable input of the drive. This command can be used to shut down the drive in an emergency. This command is functionally identical to the ST1 command.

Command	Description
OFF	De-energize the drive
1IS	*00000001 (fault bit active)

ON—Energize Drive

- ❑ Command Type: Programming
- ❑ Syntax: <a>OFF
- ❑ Range: N/A
- ❑ Attributes: Immediate
  Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: ST, OFF

The ON command immediately re-enables the drive. This command is used to re-enable the drive after a commanded shutdown or after a fault condition such as excessive position error. This command is functionally identical to the ST0 command.

For details on the enable output see *Chapter ② Installation*. Also refer to the drive user guide for proper use of enable input.

Command	Description
ON	Energize the drive
1IS	*00000000 (fault bit inactive)

O—Output

- ❑ Command Type: Programming
- ❑ Syntax: <a>Onn
- ❑ Range: Ø, 1 or X (See Below)
- ❑ Attributes: Buffered
  Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: on, off, or unchanged
- ❑ Default Value: ØØ
- ❑ See Also: OS, S, TR

The Output (**O**) command turns the programmable output bits on and off. This is used for signaling remote controllers, turning on LEDs, or sounding whistles. The output can indicate that the motor is in position, about to begin its move, or is at constant velocity, etc.

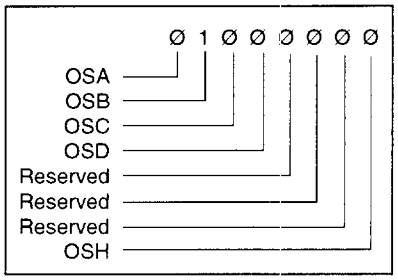
n=1 = Turns output bits on
n=Ø = Turns output bits off
n=X = Leaves output bits unchanged

Command	Description
MN	Set to Mode Normal
A1Ø	Set acceleration to 10 revs/sec^2
V5	Sets velocity to 5 revs/sec
D2ØØØØ	Set move distance to 20,000 encoder counts
OØ1	Set programmable output 1 off and output 2 on
G	Executes the move (Go)
OØØ	After the move ends, turn off both outputs

OS—Report Homing Function Set-Ups

- ❑ Command Type: Status
- ❑ Syntax: <a>OS
- ❑ Range: N/A
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: OS(A-H)
- ❑ Response to aOS is nnnnnnnn

This command results in a report of which software switches have been set by **OS** commands. The reply is eight digits. This command reports **OSA** through **OSH** Set-up status in binary format. The digit 1 represents ON (enabled), the digit Ø represents OFF (disabled). The default response is ***Ø1ØØØØØØ**.



OSA—Define Active State of End-of-Travel Limits

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>OSAn
- ❑ Range: n = Ø, 1
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: NA
- ❑ Default Value: Ø
- ❑ See Also: LD, OSC

OSAØ: Normally Closed Contacts
OSA1: Normally Open Contacts
This command sets the active state of the CW and CCW end-of-travel limit inputs. It enables you to use either normally closed or normally open switches.

Command	Description
OSA1	Sets active state for normally open limit switches
OSCØ	Sets active state of home input closed (low)
OSH1	Selects the CCW side of the home signal as the edge on which the final approach will stop

OSB—Back Up To Home

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>OSBn
- ❑ Range: n = Ø, 1
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: See Below
- ❑ Default Value: 1
- ❑ See Also: GH, OSC, OSD, OSH

OSBØ: Back up to home
OSB1: Back up to selected edge

This command is used to make homing more repeatable. With Back Up to Selected Home (**OSB**) command enabled, homing is a two step process. First it approaches the home switch at high speed (set by the GH command) until it sees the switch. Then it decelerates and returns to the switch at slow speed. Since it always makes its final approach from the same direction, the system will act differently whether the active edge of the switch is the first or second edge encountered.

If the selected edge for final home position is the first edge encountered the motor will decelerate to 0 velocity, when that edge is detected. The motor will then reverse direction and stop on the selected edge.

If the selected edge for the final home position is the second edge encountered the motor will travel until that edge is detected. The motor will decelerate to a 0 velocity. The controller will then position the motor 1/2 of a revolution on the outside of the selected edge. Finally the motor will creep at 0.1 rev/sec in the direction of the active home region, until home is detected.

With **OSB** disabled, the motor will decelerate to 0 velocity after encountering the active home region, and will be considered to be at home if the home limit input is still active. If the deceleration overshoots the active home region the motor will reverse direction and travel back to the active home region.

Command	Description
OSB1	Sets back up to home switch active
OSCØ	Sets active state of home input closed (low)
OSH1	Selects the CCW side of the home signal as the edge on which the final approach will stop

OSC—Define Active State of Home Switch

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>OSCn
- ❑ Range: n = Ø, 1
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: NA
- ❑ Default Value: Ø
- ❑ See Also: GH, OSB, OSD, OSH

OSCØ: Active state of home input is n = Ø (closed)
OSC1: Active state of home input is n=1 (open)

OSCØ requires that a normally open (high) switch be connected to the home limit input. **OSC1** requires that a normally closed (low) switch be connected to the home limit input.

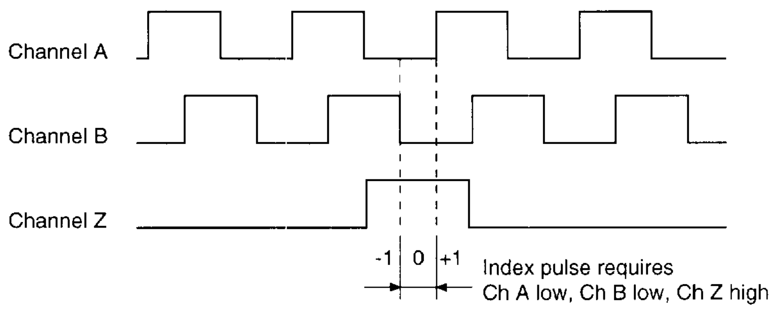
Command	Description
OSC1	Sets the active state of the home input to open

OSD—Enable Encoder Z Channel for Home

- ❑ Command Type: Set-up
- ❑ Syntax: <a>OSDn
- ❑ Range: n = 0, 1
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: 0
- ❑ See Also: OSB,OSC,OSH,GH

OSDØ = Do not reference Z Channel during homing
OSD1 = Reference Z Channel during homing

The encoder Z channel is used (in conjunction with a load activated switch connected to the home limit) to determine the home position. The switch determines the home region, and the Z channel determines the exact and final home position inside the home region. As the next drawing shows, the final home position occurs when Channel A is low, Channel B is low, and Channel Z is high.



For OSD1 to be selected, OSB1 must also be selected.

Command	Description
OSD1	Recognizes Z channel as final home reference

OSH—Reference Edge of Home Switch

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>OSHn
- ❑ Range: n = Ø, 1
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: NA
- ❑ Default Value: Ø
- ❑ See Also: GH, OSB, OSC, OSD

OSHØ: Selects the CW side of the Home signal as the edge on which the final approach will stop

OSH1: Selects the CCW side of the home signal as the edge on which the final approach will stop

The CW edge of the Home switch is the first switch transition seen by the controller when traveling from the CW limit in the CCW direction. If n = 1, the CCW edge of the Home switch will be referenced as the Home position. The CCW edge of the Home switch is the first switch transition seen by the controller when traveling from the CCW limit in the CW direction. If n = Ø, the CW edge of the Home switch will be referenced as the Home position.

Command	Description
OSB1	Sets back up to home switch active
OSCØ	Sets active state of home input closed (low)
OSH1	Selects the CCW side of the home signal as the edge on which the final approach will stop

The home limit becomes active when the home limit input is closed. The controller recognizes the CCW edge of the switch as the home limit and backs up to that edge to complete the Go Home move.

PR—Absolute Position Report

- ❑ Command Type: Status
- ❑ Syntax: aPR
- ❑ Range: ±1,073,741,820
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: Encoder counts
- ❑ Default Value: N/A
- ❑ See Also: D, MPA, MPI, MN, PZ, PX
- ❑ Response to aPR is *±nnnnnnnnnn

This command reports the commanded motor position relative to the power-up position. When a **D** command is issued, the distance is relative to the value of **PR**. The difference between the commanded position (**PR**) and the actual encoder position (**PX**) is the position error (**DPE**). The controller is always trying to minimize the position error. You can reset the encoder position counter to zero by using the position zero (**PZ**) command or reset (**Z**). After **PZ** the encoder position (**PX**) will be set to zero. If there was a position error before the **PZ** was issued, the value of **PR** will differ from the value of **PX** by the amount of the position error. Increasing integral gain (**CIG**) can help reduce the position error at rest thus insuring the value of **PR** equals the value of **PX**.

Command	Description
1PR	Commanded position report. (*+ØØØØØØ2ØØØ)
1PX	Encoder position report (*+ØØØØØØ2ØØ5)

The actual motor position is 5 encoder counts from the commanded position.

PS—Pause

- ❑ Command Type: Programming
- ❑ Syntax: <a>PS
- ❑ Range: N/A
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: C, U

This command pauses execution of a command string or sequence until the controller receives a Continue (**C**) command. **PS** lets you enter a complete command string before running other commands. **PS** is also useful for interactive tests and synchronizing multiple controllers that have long command strings.

Command	Description
PS	Pauses execution of commands until the controller receives the Continue (C) command
A5	Sets acceleration to 5 revs/sec^2
V5	Sets velocity to 5 revs/sec
D4ØØØ	Sets move distance to 4,000 encoder counts
G	Executes the move (Go)
T2	Delays the move for 2 seconds
G	Executes the move (Go)
C	Continues Execution

When the controller receives the **C** command, the motor moves 4,000 encoder counts twice with a 2 second delay between moves.

PX—Report Absolute Encoder Position

- ❑ Command Type: Status
- ❑ Syntax: aPX
- ❑ Range: ±1,073,741,820
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: Encoder counts
- ❑ Default Value: N/A
- ❑ See Also: PR, PZ
- ❑ Response to aPX *±nnnnnnnnn

This command reports the actual motor position as measured by the encoder. When a **D** command is issued, the distance is relative to the value of **PR** not the value of **PX**. The difference between the commanded position (**PR**) and the actual encoder position (**PX**) is the position error (**DPE**). The controller is always trying to minimize the position error. You can reset the encoder position counter to zero by using the position zero (**PZ**) command or reset (**Z**). After **PZ** the encoder position (**PX**) will be set to zero. If there was a position error before the **PZ** was issued, the value of **PR** will differ from the value of **PX** by the amount of the position error. Increasing integral gain (**CIG**) can help reduce the position error at rest thus insuring the value of **PR** equals the value of **PX**.

Command	Description
MN	Set to mode normal
PZ	Sets the absolute counter to zero
A1Ø	Sets acceleration to 10 rev/sec2
V5	Sets velocity to 5 rev/sec
D46ØØ	Sets move distance to 4,600 encoder counts
G	Executes the move (Go)
1PX	After the motor executes the move, the encoder position is reported: The response is *+0000004600.

PZ—Set Absolute Counter to Zero

- ❑ Command Type: Programming
- ❑ Syntax: <a>PZ
- ❑ Range: N/A
- ❑ Attributes: Buffered, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: D, MN, PR, PX

This command sets the absolute encoder position counter to zero. If there was a position error before the **PZ** was issued, the new value of **PR** will differ from the value of **PX** by the amount of the position error. Absolute counter will also be set to zero when you cycle power or when you successfully execute a homing (**GH**) function.

Command	Description
MN	Enter position mode (mode normal)
MPA	Makes preset moves from absolute zero position
PZ	Sets absolute position counter to zero
A1Ø	Sets acceleration to 10 rev/sec^2
V5	Sets velocity to 5 rev/sec
D4ØØØØ	Sets move distance to 40000 encoder counts
G	Executes the move (Go)

1PX	Reports absolute encoder position (*+0000040000)
PZ	Sets the absolute counter to zero
1PX	Reports absolute encoder position (*+0000000000)

"—Quote

- ❑ Command Type: Programming
- ❑ Syntax: "x
- ❑ Range: x = up to 17 ASCII characters
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: CR, LF
- ❑ Response to "x is x

Up to 17 characters entered after the quotation marks (") will be transmitted, exactly as they are entered, over the RS-232C link. A space entered by the space bar indicates the end of the command. A space is always sent after the last character in the string. This command is used during buffered moves or sequences to command other devices to move, or to send the message to a remote display. On a daisy chain of multiple units, if more than one unit is reporting a message at once, the messages will overlap and be garbled.

Command	Description
PS	Pause execution until Continue (C) is entered
A5	Set acceleration to 5 revs/sec^2
V5	Set velocity to 5 revs/sec
D2ØØØ	Set distance to 2,000 encoder counts
G	Executes the move (Go)
"MOVE_DONE	Transmits message
C	Continue move

After the move, the OEM Controller will send the message MOVE_DONE via the RS-232C port

QØ—Exit Velocity Profiling Mode

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>QØ
- ❑ Range: N/A
- ❑ Attributes: Immediate, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: Q1, RM

The **QØ** command exits the Velocity Profiling mode. The motor will stop when **QØ** is issued. Entering this command will cause the OEM Controller to enter Normal mode (**MN**).

Q1—Enter Velocity Profiling Mode

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>Q1
- ❑ Range: N/A
- ❑ Attributes: Immediate, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: QØ, RM

Q1 activates Velocity Profiling mode. Subsequent **RM** commands will immediately change motor velocity. **QØ** exits this mode.

Command	Description
ER2ØØØ	Set encoder resolution to 2000
Q1	Enter Velocity Streaming mode
RMØØØØ22ØC	Accelerate to 0.25 revs/sec^2
RMØØØØ4418	Accelerate to 0.5 revs/sec^2
RMØØØØ8831	Accelerate to 1 revs/sec^2
RMØØØ11Ø62	Accelerate to 2 revs/sec^2
RMØØØØ8831	Decelerate to 1 revs/sec^2
RMØØØØ4418	Decelerate to 0.5 revs/sec^2
RMØØØØ22ØC	Decelerate to 0.25 revs/sec^2
RMØØØØØØØØ	Decelerate to 0 revs/sec^2
QØ	Exit Velocity Streaming mode

R—Request Controller Status

- ❑ Command Type: Status
- ❑ Syntax: aR
- ❑ Range: N/A
- ❑ Attributes: Immediate, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: RA, RB, RC, XSR, XSS
- ❑ Response to aR is *x

The Request Controller Status (**R**) command can be used to indicate the general status of the controller. Possible responses are:

Character	Definition
*R	Ready
*S	Ready, Attention Needed
*B	Busy
*C	Busy, Attention Needed

When the controller is not prepared to accept another command, the following conditions will cause a controller is busy (***B**) response:

* Performing a move
* Accelerating/decelerating during a continuous move
* A time delay is in progress. (T command)
* In RM mode
* Paused
* Waiting on a Trigger
* Going Home
* In Power-on sequence mode
* Running a sequence
* Executing a loop

The following conditions will cause an error (***S** or ***C**) response:

* Go home failed
* Limit has been encountered
* Sequence execution was unsuccessful
* Sequence memory checksum error
* Undervoltage
* Drive recently enabled

When the response indicates that attention is required, the **RA**, **RB**, **RC**, **XSR**, or **XSS** commands can provide details about the error.

It is not recommended that this command be used in tight polling loops that could result in microprocessor overload. Time delays can alleviate this problem.

This command is not intended to be used to determine if a move is complete. It should be used after a move is complete to determine if errors or faults exist. Use a buffered status request (**CR** or **LF**) command or a programmable output to indicate move completion.

Command	Response
1R	***R** (Controller ready to accept a command, and no error conditions exist.)

RA—Limit Switch Status Request

- ❑ Command Type: Status
- ❑ Syntax: aRA
- ❑ Range: N/A
- ❑ Attributes: Immediate, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: R, RB
- ❑ Response to aRA is *x

The **RA** command responds with the status of the end-of-travel limits during the last move as well as the present condition. This is done by responding with one of 12 characters representing the conditions listed below.

Response Character	***Last Move Terminated By***		***Current Limit Status***	
	CW Limit	***CCW Limit***	***CW Limit***	***CCW Limit***
*@	No	No	Off	Off
*A	Yes	No	Off	Off
*B	No	Yes	Off	Off
*D	No	No	On	Off
*E	Yes	No	On	Off
*F	No	Yes	On	Off
*H	No	No	Off	On
*I	Yes	No	Off	On
*J	No	Yes	Off	On
*L	No	No	On	On
*M	Yes	No	On	On
*N	No	Yes	On	On

The **RA** command is useful when the motor will not move in either or both directions. The report back will indicate if the last move was terminated by one or both end-of-travel limits. This command is not intended to be used to determine if a move is complete. It should be used after a move to determine if errors or faults exist. If you are hitting a limit switch, the Ready Status (**R**) will return a ***S**.

Command	Response
1RA	*@ (the last move was not terminated by a limit and no limits are currently active.)

RB—Loop, Pause, Shutdown, Trigger Status Request

- ❑ Command Type: Status
- ❑ Syntax: aRB
- ❑ Range: N/A
- ❑ Attributes: Immediate, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: L, PS, R, RA, ST, TR
- ❑ Response to aRB is *x

This command receives a response from *@ to ***O**, as defined below. The four conditions for which status is indicated are as follows:

Loop Active: A loop is in progress.
Pause Active: Buffered commands waiting for a Continue (**C**).
Shutdown Active: The motor is shutdown by the **ST1** command.
Trigger Active: At least one trigger is active.

Response Character	*Loop Active*	*Pause Active*	*Shutdown Active*	*Trigger Active*
*@	No	No	No	No
*A	Yes	No	No	No
*B	No	Yes	No	No
*C	Yes	Yes	No	No
*D	No	No	Yes	No
*E	Yes	No	Yes	No
*H	No	No	No	Yes
*I	Yes	No	No	Yes
*J	No	Yes	No	Yes
*K	Yes	Yes	No	Yes
*L	No	No	Yes	Yes
*M	Yes	No	Yes	Yes
*N	No	Yes	Yes	Yes
*O	Yes	Yes	Yes	Yes

This command is not intended to be used to determine if a move is complete. It should be used after the move is complete to determine if errors or faults exist.

Command	Response
1RB	*A (After issuing a 1RB command, the response came back as *A. This means that the controller is currently executing a loop.)

RC—Homing Status Request

- ❑ Command Type: Status
- ❑ Syntax: aRC
- ❑ Range: N/A
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ Response to aRC IS *x
- ❑ See Also: R, RA, RB, FS, GH

The RC command has the same response format of RA and RB. The condition for which status is indicated is:

Homing Function Failure:
In this condition, the controller has encountered both End-of-Travel limits or one of several possible Stop commands or conditions. Go Home motion was concluded, but not at Home.

Response Character	**Go Home Successful?**
*@	YES
*B	NO

Command	Description
1RC	*B Go home was unsuccessful.

RFS—Return Servo Gains to Factory Settings

- ❑ Command Type: Status
- ❑ Syntax: aRFS
- ❑ Range: N/A
- ❑ Attributes: Immediate, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: RA, RB, RC, XSR, XSS
- ❑ Response to aRFS is *x

This command can be used to return all the servo gains to the factory defaults immediately. This command is useful when the current servo gain values are inappropriate and re-tuning is required. By returning to factory defaults the gains will be in proper relationship to one another allowing easy fine tuning of the system. The factory defaults are:

Proportional Gain	(**CPG**)	16
Integral Gain	(**CIG**)	2
Derivative Gain	(**CDG**)	240
Derivative Sampling Period	(**CTG**)	0

Position Error	(**CPE**)	4000
Integral Limit	(**CIL**)	2

Command	Description
1RFS	All servo gains returned to the factory defaults

RM—Rate Multiplier in Velocity Streaming

- ❑ Command Type: Motion
- ❑ Syntax: <a>RMn
- ❑ Range: n = Ø - FFFFFFFF
- ❑ Attributes: Immediate, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: revs/sec
- ❑ Default Value: None
- ❑ See Also: D, H, QØ, Q1

The **RM** command sets an immediate velocity where n represents an 8-digit hexadecimal value. The value for n is determined with the following formula:

(desired revs/sec) • *(encoder resolution)* * *17.432576* = **decimal # for velocity value to be rounded off to the closest whole number.**

The resulting decimal number must be converted to a hexadecimal number to obtain the value for n.

The velocity change is instant—there is no acceleration/deceleration ramp between velocities. A limit switch closure will stop movement in Velocity Profiling mode, but does not cause the OEM Controller to exit Velocity Streaming mode. To recover from a limit stop in **RM** mode, **QØ** must be issued and the direction must be changed. Velocity Profiling mode is uni-directional. The last direction set either from a move or from a Distance (**D**) or Direction (**H**) command will be used. Bi-directional moves can be made in this mode by returning to velocity zero (Ø), turning **RM** mode off, changing the direction, and re-enabling **RM** mode. Exiting **RM** mode with **QØ** causes the OEM Controller to enter Normal mode (**MN**).

Command	Response
ER2ØØØ	Set encoder resolution to 2000
Q1	Enter Velocity Streaming mode
RMØØØØ22ØC	Accelerate to 0.25 revs/sec^2
RMØØØØ4418	Accelerate to 0.5 revs/sec^2
RMØØØØ8831	Accelerate to 1 revs/sec^2
RMØØØ11Ø62	Accelerate to 2 revs/sec^2
RMØØØØ8831	Decelerate to 1 revs/sec^2
RMØØØØ4418	Decelerate to 0.5 revs/sec^2
RMØØØØ22ØC	Decelerate to 0.25 revs/sec^2
RMØØØØØØØØ	Decelerate to 0 revs/sec^2
QØ	Exit Velocity Streaming mode

RSE—Report Servo Errors

- ❑ Command Type: Status
- ❑ Syntax: aRSE
- ❑ Range: N/A
- ❑ Attributes: Immediate, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: R,RA, RB, RC
- ❑ Response to aRSE is *x

The Report Servo Errors (**RSE**) command can be used to indicate the general status of the controller. Possible responses are:

Character	Definition
*0	No errors
*2	Excessive position error
*4	Drive fault
*6	Commanded shutdown
*8	Undervoltage, or drive was recently enabled

During a fault condition, the RSE command can be used to interrogate the controller for the reason of the fault. The following can cause a latched fault condition—the red LED will be illuminated, and the fault output will be active:

-Drive commanded shutdown via the **OFF** or **ST1** commands

-Position error exceeded the value set in the **CPE** command

-Drive fault (overvoltage, overtemperature, etc.)

The next two conditions are not latched, and do not illuminate the red LED. They will cause a ***8** response to **RSE**, until the next move command is issued.

-Undervoltage (voltage at drive's DC input drops below 21.5VDC)

-Drive was recently enabled

Command	Response
1RSE	*0 (Controller ready to accept a command, and no error conditions exist.)

RV—Revision Level

❑ Command Type: Status	❑ Valid Software Version: A
❑ Syntax: aRV	❑ Units: N/A
❑ Range: N/A	❑ Default Value: N/A
❑ Attributes: Buffered, Savable in Sequence	❑ See Also:
	❑ Response to aRV is nn-nnnnnn-nnnn

The Revision (**RV**) command responds with the software part number and its revision level. The response is in the form shown below:

***92-nnnn-nn<*xn*>**[cr]
(part number, ***revision level***)

The part number identifies which product the software is written for, as well as any special features that the software may include. The revision level identifies when the software was written. You may want to record this information in your own records for future use. This type of information is useful when you consult Parker Compumotor's Applications Department.

Command	Response
1RV	92-016637-01A

The product is identified by *92-016637-01A, and the revision level is identified by A.

S—Stop

❑ Command Type: Motion	❑ Valid Software Version: A
❑ Syntax: <a>S	❑ Units: N/A
❑ Range: N/A	❑ Default Value: N/A
❑ Attributes: Immediate, Never Saved	❑ See Also: A, K, QØ, SSH, ST

This command decelerates the motor to a stop using the last defined Acceleration (**A**) command. This command clears the command buffer (at the end of a move, if one is in progress). The Sequence Definition (**XD**) command is aborted and a time delay is terminated. If **SSH1** is set the controller will stop the current move but it will not clear the command buffer.

Command	Description
MC	Sets move in continuous mode
A1	Sets acceleration to 1 revs/sec^2
V1Ø	Sets velocity to 10 revs/sec
G	Executes the move (Go)
S	Stops motor (motor comes to a stop at a deceleration rate of 1 revs/sec^{2})

The **S** command is not buffered. As soon as the controller receives the **S** command, it stops motion.

SN—Scan

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>SNn
- ❑ Range: 1 - 1000
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: n = mS
- ❑ Default Value: 50
- ❑ See Also: XP

The Scan (**SN**) command allows you to define the *debounce time* (in milliseconds) for external sequence selection inputs. The debounce time is the amount of time that the sequence inputs must remain constant for a proper reading from a remote controller, such as a programmable logic controller (PLC). If you are using a PLC you should change the debounce time to match the *on time* of the PLC outputs.

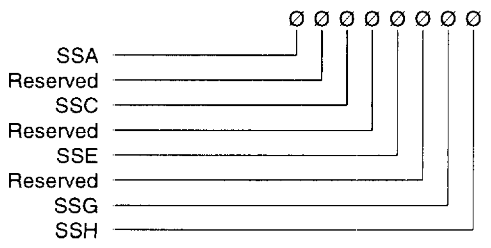
This command allows you to select the best possible trade-off between noise immunity and speed for a given application. If you make your scan time too short, the OEM070 may respond to an electrical glitch. If you issue the Scan command with only a device address (**1SN**), the controller will respond with the current debounce time (***SNn**).

Command	**Description**
SN1Ø	Sets scan time of sequence select inputs to 10 ms

SS—Software Switch Function Status

- ❑ Command Type: Status
- ❑ Syntax: aSS
- ❑ Range: N/A
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: SSA, SSC, SSG, SSH
- ❑ Response to aSS is *nnnnnnnn

This command reports the status of the **SS** commands. From left to right, the 8-character response corresponds to **SSA** through **SSH**.



SSA—RS-232C Echo Control

❑ Command Type: Set-Up	❑ Valid Software Version: A
❑ Syntax: <a>SSAn	❑ Units: See Below
❑ Range: n = Ø, 1	❑ Default Value: Ø
❑ Attributes: Buffered, Savable In Sequence	❑ See Also:

This command turns the RS-232C echo (transmission of characters received from the remote device by the OEM070) on and off.

SSAØ = Echo on
SSA1 = Echo off

In the Echo On (**SSAØ**) mode, characters that are received by the controller are echoed automatically. In the Echo Off (**SSA1**) mode, characters are not echoed from the OEM070. This command is useful if your computer cannot handle echoes. In a daisy chain, you must have the echo on (**SSAØ**) to allow controllers further down the chain to receive commands. *Status commands do not echo the command sent, but transmit the requested status report.*

Command	Description
SSA1	Turns echo off (Characters sent to the controller are not echoed back to the host.)

SSC—Output #1 on In Position

❑ Command Type: Set-up	❑ Valid Software Version: A
❑ Syntax: <a>SSCn	❑ Units: N/A
❑ Range: n = 0 or 1	❑ Default Value: 0
❑ Attributes: Buffered, Savable in Sequence	❑ See Also: CEW, CIT

With **SSC** set to 1, output 1 will turn on when the motor is within the In Position window for the specified time defined by the **CEW** and **CIT** commands

b = 0:	Normal
b = 1:	Output 1 is configured as an "In Position" output

Command	Response
SSC1	Set output 1 as an In Position output

SSE Enable/Disable Communication Error Checking

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>SSEn
- ❑ Range: 0 (disable), 1 (enable)
- ❑ Attributes: Buffered, Savavble in Sequence
- ❑ Valid Software Version: E
- ❑ Units: N/A
- ❑ Default Value: 0 (disable)
- ❑ See Also: %

This command setting determines whether or not each byte received at the controller is checked for communication errors. **SSE1** enables error checking for all bytes received at the controller, and **SSE0** disables error checking. See the **%** command for the types of errors detected.

SSG—Clear/Save the Command Buffer on Limit

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>SSGn
- ❑ Range: n = Ø, 1
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: See Below
- ❑ Default Value: Ø
- ❑ See Also: LD

SSGØ = Clears command buffer on limit
SSG1 = Saves command buffer on limit

In most cases, it is desirable that upon activating an end-of-travel limit input all motion should cease until the problem causing the over-travel is rectified. This will be assured if all commands pending execution in the command buffer are cleared when hitting a limit. This is the case if **SSGØ** is specified. If **SSG1** is specified and a limit is activated, the current move is aborted, but the remaining commands in the buffer continue to be executed.

Command	Description
SSG1	Saves buffer on limit
A1Ø	Sets acceleration to 10 revs/sec^2
V5	Sets velocity to 5 revs/sec
D4ØØØ	Sets distance to 4,000 encoder counts
G	Executes the move (Go)
O11	Turn on outputs 1 and 2

If a limit switch is encountered while executing the move, outputs 1 and 2 will still go on.

SSH—Clear/Save Command Buffer on Stop

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>SSHn
- ❑ Range: n = Ø, 1
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: See Below
- ❑ Default Value: Ø
- ❑ See Also: S

SSHØ = Clears command buffer on stop
SSH1 = Saves command buffer on stop

In Normal Operation (**SSHØ**) the Stop (**S**) command or a dedicated stop input will cause any commands in the command buffer to be cleared. If you select the Save Command Buffer On Stop (**SSH1**) command, a Stop (**S**) command will only stop execution of a move in progress. It will not stop execution of any commands that remain in the buffer. However, it will terminate a loop in the current pass.

Command	Description
SSHØ	Clears buffer on stop
A1Ø	Sets acceleration to 10 revs/sec^2
V5	Sets velocity to 5 revs/sec
D4ØØØ	Sets distance to 4,000 encoder counts
L5Ø	Loops 50 times
G	Executes the move (Go)
T.5	Pauses the motor 500 ms
N	Ends Loop
S	Stops motion

When **S** is issued, the controller will clear the buffer and stop the move.

ST—Shutdown

- ❑ Command Type: Programming
- ❑ Syntax: <a>STn
- ❑ Range: n = Ø, 1
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: See Below
- ❑ Default Value: Ø
- ❑ See also: OFF, ON

The ST1 command immediately disables the drive through the enable output, which would be connected to the enable input of the drive. This command can be used to shut down the drive in an emergency. This command is functionally identical to the OFF command.

The STO command immediately re-enables the drive. This command is used to re-enable the drive after a commanded shutdown or after a fault condition such as excessive position error. This command is functionally identical to the ON command.

For details on the enable output see *Chapter ② Installation*. Also, refer to the drive product user guide for the proper use of enable input.

Command	Description
ST1	Disable the drive product
ST0	Enable the drive product

T—Time Delay

- ❑ Command Type: Programming
- ❑ Syntax: <a>Tn
- ❑ Range: n = 0.01 - 99999.99
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: seconds
- ❑ Default Value: None

The Time (**T**) command causes the controller to wait the number of seconds that you specify before it executes the next command in the buffer. This command is useful whenever you need to delay the motor's actions or when you wish to move the motor in continuous velocity for preset time.

Command	Description
MN	Sets Normal mode
A5	Sets acceleration to 5 revs/sec^2
V5	Sets velocity to 5 revs/sec
D4ØØØ	Sets distance to 4,000 encoder counts
T1Ø	Pauses motor movement 10 seconds
G	Executes the move (Go)
T5	Pauses the motor for 5 seconds after the move
G	Executes the move (Go)

TR—Wait For Trigger

- ❑ Command Type: Programming
- ❑ Syntax: <a>TRnnnnn
- ❑ Range: n = Ø, 1, or X
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: See Below
- ❑ Default Value: None
- ❑ See Also: IN

This command allows you to specify a trigger configuration to be matched before continuing execution of the move, where *nnnnn* corresponds to triggers 1, 2, 3, 4 and 5 respectively. The possible values for *n* are as follows:

n = 1	Wait for the trigger input to be high (opened)
n = Ø	Wait for the trigger input to be low (grounded)
n = X	Ignore the trigger input

The lowest numbered input will be the first trigger. For example, if input 3,4 and 5 are configured with the IN command as triggers, they will be trigger 1,2 and 3 respectively.

Command	Description
1IN3A	Configure input 3 as trigger input 1
1IN5A	Configure input 5 as trigger input 2
TR1Ø	Wait for input 1 to be opened and input 2 to be grounded before going on to the next command
A1Ø	Sets acceleration to 10 revs/sec^2
V5	Sets velocity to 5 revs/sec
D4ØØØ	Sets distance to 4,000 encoder counts
G	Executes the move (Go)

Motion will not occur until trigger conditions are true.

U—Pause and Wait for Continue

- ❑ Command Type: Programming
- ❑ Syntax: <a>U
- ❑ Range: N/A
- ❑ Attributes: Immediate, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: C, PS

This command causes the indexer to complete the move in progress, then wait until it receives a Continue (**C**) to resume processing. Since the buffer is saved, the controller continues to execute the program (at the point where it was interrupted). The controller continues processing when it receives the **C** command. This command is typically used to stop a machine while it is unattended.

Command	Description
MN	Sets move to Normal mode
A5	Sets acceleration to 5 revs/sec^2
V5	Sets velocity to 5 revs/sec
LØ	Loops indefinitely
D46ØØ	Sets distance to 4,600 encoder counts
G	Executes the move (G)
T1Ø	Waits 10 seconds after the move
N	Ends loop
U	Halts execution until the controller receives the Continue command **(C)**

This command string pauses when the **U** command is entered. A **C** command resumes execution where it was paused. In this example, the loop stops at the end of a move, and resumes when the controller receives the **C** command. In reaction to the **T1Ø** command in the loop, there may be a 10 second delay before motion resumes after the **C** is executed, depending on when the **U** command is completed.

V—Velocity

- ❑ Command Type: Motion
- ❑ Syntax: <a>Vn
- ❑ Range: n = 0.01 - 200.00
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: revs/sec
- ❑ Default Value: 1
- ❑ See Also: A, D, G, GH, MR

The **V** command defines the maximum speed at which the motor will run when given the Go (**G**) command. The maximum encoder frequency the controller can accept is 960 kHz. In preset Mode Normal (**MN**), the maximum velocity may be limited when the resulting move profile is triangular. In Mode Continuous (**MC**), when a Go (**G**) command is issued the controller moves to the next command in the buffer.

Once you define the velocity, that value will be valid until you define another velocity, cycle DC power, or issue a **Z** (Reset) command.

*If the value specified for the **V** command is not valid, the OEM070 ignores that value and defaults to the value specified in the last **V** command. If **V** is issued with only a device address (**1V**), the controller will respond with the current velocity value (***Vn**).*

Command	Description
MC	Sets move to continuous
A5	Sets acceleration to 5 revs/sec^2
V5	Sets velocity to 5 revs/sec
G	Go (Begin motion)

XC—Sequence Checksum

- ❑ Command Type: Status
- ❑ Syntax: aXC
- ❑ Range: N/A
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: None
- ❑ See Also: XD, XE

XC computes the BBRAM checksum. After the unit is programmed, the response can be used for system error checking. The three-decimal response (ØØØ - 255) is followed by a [cr]. The response does not indicate the number of bytes programmed. This response is designed to be used for comparison. As long as the OEM070 is not re-programmed, the checksum response should always be the same.

Command	Response
1XC	*149

XD—Sequence Definition

- ❑ Command Type: Programming
- ❑ Syntax: <a>XDn
- ❑ Range: n = 1 - 7
- ❑ Attributes: Buffered, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: Sequences
- ❑ Default Value: None
- ❑ See Also: XE, XR, XRP, XT

This command begins sequence definition. All commands between the **XD** command and Sequence Termination (**XT**) command are defined as a sequence. The sequences will automatically be defined when **XT** is issued. If a sequence you are trying to define already exists, you must erase that sequence before defining it using the Erase Sequence (**XE**) command. A sequence cannot be longer than 255 characters. Immediate commands cannot be entered into a sequence. Sequences can only be permanently saved with the -M2 (BBRAM) option. Without the -M2 option sequences can be saved in operating RAM, and *will* be retained after a reset (Z) but not after a power cycle.

Command	Description
XE1	Erases sequence #1
XD1	Defines sequence #1
MN	Sets to Normal mode
A1Ø	Sets acceleration to 10 revs/sec^2
V5	Sets acceleration to 5 revs/sec
D1ØØØØ	Sets distance to 10,000 encoder counts
G	Executes the move (Go)
XT	Ends definition of Sequence #1
XR1	Executes Sequence #1

XE—Sequence Erase

- ❑ Command Type: Programming
- ❑ Syntax: <a>XEn
- ❑ Range: n = 1 - 7
- ❑ Attributes: Buffered, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: Sequences
- ❑ Default Value: None
- ❑ See Also: XD, XR, XRP, XT

This command allows you to delete a sequence. The sequence that you specify (n) will be deleted when you issue the command. *Compumotor recommends that you delete a sequence before re-defining it.*

Command	Description
XE1	Deletes Sequence 1
XD1	Defines Sequence 1

XP—Set Power-up Sequence Mode

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>XPn
- ❑ Range: n = 0 - 9
- ❑ Attributes: Buffered, Automatically Saved
- ❑ Valid Software Version: A
- ❑ Units: Sequences
- ❑ Default Value: 0
- ❑ See Also: IN, XQ, XSP, XSR

This command executes a single sequence or multiple sequences on power-up. If *n* = 1-7, the sequence whose value = *n* will be executed on power up. Control will then be passed to the RS-232C interface.

If *n* = 8, the sequence whose number appears on the sequence select inputs (configured with the **IN** command) will be executed on power-up. Control will then be passed to the RS-232C interface.

If *n* = 9, the sequence whose number appears on the Sequence Select inputs (configured with the **IN** command) will be executed on power-up. When the first sequence is finished in **XP9** mode, the

OEM070 will scan the Sequence Select inputs again and execute the next sequence. This cycle will continue until a Stop (**S**) or Kill (**K**) command is issued, a limit is encountered, or the unit is powered down. The possible settings for this command are as follows:

n = Ø: No sequence is executed on power-up
n =1-7: Sequence 1 - 7 is executed on power-up
n = 8: Sequence select inputs are read (single run) on power-up
n = 9: Sequence select inputs are read (continuous run) on power-up

In **XP9** mode, you can use the **XQ1**command to stop the OEM070 from selecting the next sequence until all the sequence select inputs are first opened.

Sequences can only be permanently saved with the -M2 (BBRAM) option. Without the -M2 option sequences can be saved in operating RAM, and *will* be retained after a reset (Z) but not after a power cycle.

Command	Description
XP1	Executes Sequence #1 on power-up
XE1	Erases Sequence #1
XD1	Defines Sequence #1
LD3	Disables CW & CCW limits
A1Ø	Sets acceleration to 10 revs/sec2
V5	Sets velocity to 5 revs/sec
D4ØØØ	Sets distance to 4,000 encoder counts
G	Executes the move (Go)
XT	Ends definition of Sequence #1
Z	Resets the controller

The motor moves 4,000 encoder counts during power-up (with -M2 option only) or reset (Z).

XQ—Sequence Interrupted Run Mode

- ❑ Command Type: Set-Up
- ❑ Syntax: <a>XQn
- ❑ Range: n = Ø, 1
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: Sequences
- ❑ Default Value: Ø
- ❑ See Also: XP

n = 1: Interrupted Run mode is set (on)
n = Ø: Interrupted Run mode is reset (off)

This command can be used only when the OEM070 is stand-alone power-up sequencing in **XP9** mode. In **XP9** mode, if **XQ1** is executed, the OEM070 will not accept a sequence select input until all sequence select inputs are OFF (closed). After all lines have simultaneously been brought to a low state (OFF), the controller will then read the sequence select lines and execute the sequence whose number appears there. This paused mode will continue until an **XQØ** command is executed. You may use **S** or **K** command to stop sequence execution. **XQ1** must be the first command entered in the sequence.

Command	Description
XE1	Erases sequence #1
XD1	Defines sequence #1
XQ1	Turns Interrupted Run mode on
LD3	Disables CW & CCW limits
XT	Ends Sequence #1
XP9	Sets power-up sequences as sequence select inputs
Z	Resets the controller to start sequence scanning

If you execute Sequence #1 during power up by setting the sequence select inputs (configured with the **IN** command) inputs properly, Interrupted Run mode will be set.

XR—Run a Sequence

- ❑ Command Type: Programming
- ❑ Syntax: <a>XRn
- ❑ Range: n = 1 - 7
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: Sequence
- ❑ Default Value: 0
- ❑ See Also: XD, XE, XRP, XT

This command loads a pre-defined sequence into the command buffer (clears the buffer first) and executes these commands as a normal set of commands. **XR** automatically recalls the sequence from BBRAM.

XR can be used within one sequence to start execution of another sequence; however, all commands in the first sequence following **XR** will be ignored (in this respect an XR acts like a GOTO not a GO-SUB). An **XR** command placed within a loop will be ignored.

Sequences can only be permanently saved with the -M2 (BBRAM) option. Without the -M2 option sequences can be saved in operating RAM, and will be retained after a reset (Z) but not after a power cycle.

Command	Description
XE1	Erases sequence #1
XD1	Defines sequence #1
A1Ø	Sets acceleration to 10 revs/sec2
V5	Sets velocity to 5 revs/sec
D1ØØØØ	Sets distance to 10,000 encoder counts
G	Executes the move (Go)
XT	Ends Sequence #1 definition
XR1	Executes Sequence #1

Sequence #1 is defined (XD1) and executed (**XR1**).

XRP—Sequence Run With Pause

- ❑ Command Type: Programming
- ❑ Syntax: <a>XRPn
- ❑ Range: n = 1 - 7
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: Sequence
- ❑ Default Value: 0
- ❑ See Also: XD, XE, XR, XT

This command is identical to the Sequence Run (**XR**) command, except that it automatically generates a pause condition. You must clear this condition with the Continue (**C**) command before the controller executes the command buffer. The pause condition is invoked only if the sequence is valid. This allows you to execute a sequence without the delay of buffering that sequence.

Command	Description
XE5	Erases Sequence #5
XD5	Defines Sequence #5
A1Ø	Sets acceleration to 10 revs/sec2
V5	Sets velocity to 5 revs/sec
D1ØØØØ	Sets distance to 10,000 encoder counts
G	Executes the move (Go)
XT	Ends definition of Sequence #5
XRP5	Runs Sequence #5 with a pause
C	Indexer executes Sequence #5

Upon issuing XRP5, Sequence #5 is entered in the command buffer, but is not executed. Issue a C command to execute Sequence #5.

XSD—Sequence Status Definition

- ❑ Command Type: Programming
- ❑ Syntax: aXSD
- ❑ Range: N/A
- ❑ Attributes: Buffered, Savable in Sequence
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: XD, XE, XT
- ❑ Response to aXSD is *n

This command reports the status of the previous sequence definition (**XD**...**XT**). The response is 0 - 2. The valid values and descriptions of possible responses are shown below:

n = Ø: Download O.K.
n = 1: A sequence already exists with the number you have specified.
n = 2: Out of memory. The sequence buffer is full.

XSD verifies that the last sequence definition was successful.

Command	Response
1XSD	*1 (A sequence already exists as sequence 1)

XSP—Sequence Status Power-up

- ❏ Command Type: Status
- ❏ Syntax: aXSP
- ❏ Range: N/A
- ❏ Attributes: Buffered, Never Saved
- ❏ Valid Software Version: A
- ❏ Units: N/A
- ❏ Default Value: N/A
- ❏ See Also: XP, XQ, XSR
- ❏ Response to aXSP is *n

The Sequence Status Power-up (**XSP**) determines which, if any, sequence will be executed on power-up. After setting a power-up sequence using the Sequence Power-up (**XP**) command, you can check to make sure that proper sequence will be executed on power-up with **XSP**. The command reports which sequence the system will execute during power-up. The range of sequences is Ø - 9.

Command	Description
1XSP	*3 (Indicates that sequence #3. If it exists, will be executed upon power-up or reset.)

XSR—Sequence Status Run

- ❏ Command Type: Status
- ❏ Syntax: aXSR
- ❏ Range: N/A
- ❏ Attributes: Immediate, Never Saved
- ❏ Valid Software Version: A
- ❏ Units: N/A
- ❏ Default Value: N/A
- ❏ See Also: XR, XRP
- ❏ Response to aXSR is *n

This command allows you to check whether or not the last sequence issued was executed successfully without hitting limits, Stop (**S**), or Kill (**K**). The valid values and responses are shown below.

- * **Ø** = Last sequence was successful
- * **1** = In a loop
- * **2** = Invalid sequence
- * **3** = Erased
- * **4** = Bad checksum
- * **5** = Running
- * **6** = Killed, stopped

Command	Response
1XSR	*Ø (Sequence ran O.K.)

XSS—Sequence Status

- ❑ Command Type: Status
- ❑ Syntax: aXSSn
- ❑ Range: n = 1 - 7
- ❑ Attributes: Buffered, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: Sequences
- ❑ Default Value: None
- ❑ See Also: XD, XE, XT
- ❑ Response to aXSSn is *x

XSS reports whether the sequence specified by n (representing one of the sequences 1 - 7) is empty, has bad checksum, or is OK.

Ø = Empty
1 = Bad Checksum
3 = O.K.

XSS verifies the existence of sequences and if that portion of memory has been corrupted.

Command	Response
1XSS1	*Ø (Sequence #1 of device 1 is not defined.)

XT—Sequence Termination

- ❑ Command Type: Programming
- ❑ Syntax: <a>XT
- ❑ Range: N/A
- ❑ Attributes: Buffered, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: XD, XE, XR, XRP

XT is a sequence terminator. This command flags the end of the sequence currently being defined. Sequence definition is not complete until this command is issued. Properly defined sequences are saved into BBRAM (*-M2 Option Only*) automatically by issuing this command.

NOTE: In your communication program, use sufficient time delays after downloading a sequence before you send more commands to the OEM070. In particular, after sending the **XT** command, wait at least 12.5 msec before sending a **Z** command.

Command	Description
XE1	Erases Sequence #1
XD1	Defines Sequence #1
MN	Sets to Normal mode
A1Ø	Sets acceleration to 10 revs/sec^2
V5	Sets velocity to 5 revs/sec
D4ØØØ	Sets distance to 4,000 encoder counts
G	Executes the move (Go)
XT	Ends sequence definition

XU—Upload Sequence

❑ Command Type: Status
❑ Syntax: aXUn
❑ Range: n = 1 - 7
❑ Attributes: Buffered, Never Saved
❑ Valid Software Version: A
❑ Units: Sequences
❑ Default Value: N/A
❑ See Also: F, XD, XE, XT
❑ Response to aXUn is contents of sequence n

This command sends the contents of sequence n to the host computer via the RS-232C interface, terminated by a carriage return [cr]. The contents of that sequence will appear on the computer CRT. All command delimiters in the sequence will be shown as spaces (2ØH). Any device identifiers that were included in the original sequence will also be eliminated (they are not stored in the sequence).

When using a daisy-chain, **XU** must be used cautiously as the contents of the sequence will go to all controllers in the loop between the controller that is uploading and the host. The **F** command may be used to turn off communication on units you are not uploading from.

Command	Description
2F	Turns off communication to unit #2
3F	Turns off communication to unit #3
1XU1	Uploads sequence #1 from unit #1

Y—Stop Loop

❑ Command Type: Programming
❑ Syntax: <a>Y
❑ Range: N/A
❑ Attributes: Immediate, Never Saved
❑ Valid Software Version: A
❑ Units: N/A
❑ Default Value: N/A
❑ See Also: L, N

The Stop Loop (**Y**) command takes you out of a loop when the loop completes its current pass. This command does not halt processing of the commands in the loop until the controller processes reach the last command of the current loop. At that time, the controller executes the command that follows the End Loop (**N**) command. You cannot restart the command loop unless you enter the entire command structure, including the Loop (**L**) and End Loop (**N**) commands.

Command	Description
L	Loops indefinitely
A1Ø	Sets acceleration to 10 revs/sec^2
V5	Sets velocity to 5 revs/sec
D4ØØØ	Sets distance to 4,000 encoder counts
T2	Waits 2 seconds
G	Executes the move (Go)
N	Ends loop
Y	Stops loop

The loop requires the motor to move 4,000 encoder counts CW and then wait for 2 seconds. The loop terminates at the end of the loop cycle it is executing when it receives the **Y** command.

Z—Reset

- ❑ Command Type: Programming
- ❑ Syntax: <a>Z
- ❑ Range: N/A
- ❑ Attributes: Immediate, Never Saved
- ❑ Valid Software Version: A
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: K, S

The Reset (**Z**) command is equivalent to cycling DC power to the controller. This command returns all internal settings to their power-up values. It clears the command buffer. Like the Kill (**K**) command, the **Z** command immediately stops output pulses to the motor.

When you use the **Z** command, the controller is busy for 1,000 ms and ignores all commands. This command sets all position counters to zero and returns all values except the **XP** command to factory defaults.

Command	Description
1Z	Resets controller with address 1

#—Address Numbering

- ❑ Command Type: Set-up
- ❑ Syntax: <a>#n
- ❑ Range: n=1-255
- ❑ Attributes: Immediate, Automatically Saved
- ❑ Valid Software Version: A
- ❑ Units: Address number
- ❑ Default Value: 1
- ❑ See Also: E,F

This command sets the individual unit address for each OEM070, allowing addresses up to 255. Upon receipt of the command, the OEM070 will assign itself the address in the command and will pass on the daisy chain the address *plus one*, thus enabling automatic addressing of all units on the daisy chain. The address may also be set individually if preferred.

#1 - Automatic addressing of all units
Response - #(number of units plus one)

If the unit addresses exceed 255 , then the response will be #?. A <CR> or LF must be used with this command.

<u>Command</u>	<u>Description</u>
#1	#5 (for a daisy chain of 4 units, the units will assign themselves addresses 1 through 4 and return #5 as confirmation).

%—Reset Communication

- ❑ Command Type: Status
- ❑ Syntax: %
- ❑ Range: N/A
- ❑ Attributes: Immediate, Never Saved
- ❑ Valid Software Version: E
- ❑ Units: N/A
- ❑ Default Value: N/A
- ❑ See Also: E,F,SSE

When the OEM070 detects a communication error, it ignores all *external* commands and echoes an **&** for each byte it receives from the host. You can use this **%** command to re-establish communication, and to identify the cause of the communication error.

In a daisy-chained environment, units located downstream from the unit detecting a communication error will also disable external command processing. Units upstream in a daisy chain are not affected.

NOTE: Error detection will only occur if SSE1 is enabled. Detection of a communication error has no effect on internal command processing, or sequence execution. A communication error will not stop motion.

Possible responses are:

Character	Definition
*	No errors
***0**	Unit upstream (daisy chained)
***1**	Overrun, data received too fast
***2**	Framing error

Command	Repsonse
%	***2** (Either host or controller has lost synchronization.)

NOTE: For daisy chained environments, the response values are in reverse order.

%	***0*0*0*1******* (First 5 units report no error, 6th unit detected an overrun error, and the last 3 units turned communication off because of unit 6)

Summary of Commands

A—Acceleration
B—Buffer Status
BCDG—Buffered Configure Derivative Gain
BCIG—Buffered Configure Integral Gain
BCIL—Buffered Configure Integral Limit
BCPE—Buffered Configure Position Error
BCPG—Buffered Configure Proportional Gain
BCTG—Buffered Configure Derivative Sampling Period
BS—Buffer Size Status
CDG—Configure Derivative Gain
CEW—Configure In Position Error Window
CIG—Configure Integral Gain
CIL—Configure Integral Limit
CIT—Configure In Position Time
CPE—Configure Maximum Position Error
CPG—Configure Proportional Gain
CR—Carriage Return
CTG—Configure Filter Time Constant
D—Distance
DPA—Display Position Actual
DPE—Display Position Error
DVA—Display Velocity Actual
E—Enable Communications
ER—Encoder Resolution
F—Disable Communications
G—Go
GH—Go Home
^H—Delete
H—Set Direction
IN—Set Input Functions
IS —Input Status
K—Kill
L—Loop
LD—Limit Disable
LF—Line Feed
MC—Mode Continuous
MN—Mode Normal
MPA—Mode Position Absolute
MPI—Mode Position Incremental
N—End of Loop
O—Output
OFF—Servo Disable
ON—Servo Enable
OS—Report Homing Function Set-Ups
OSA—Define Active State of End-of-Travel Limits
OSB—Back Up To Home
OSC—Define Active State of Home Switch
OSD—Enable Encoder Z Channel for Home
OSH—Reference Edge of Home Switch
PR—Position Report
PS—Pause
PX—Report Absolute Encoder Position
PZ—Set Absolute Counter to Zero
“—Quote
Q1—Enter Velocity Profiling Mode
QØ—Exit Velocity Profiling Mode
R—Request Controller Status
RA—Limit Switch Status Report
RB—Loop, Pause, Shutdown, Trigger Status Report
RC—Homing Status Report
RFS—Return Servo Gains to Factory Settings
RM—Rate Multiplier in Velocity Streaming Mode
RSE—Report Servo Erros
RV—Revision Level
S—Stop
SN—Scan
SS—Software Switch Function Status
SSA—RS-232C Echo Control
SSC—Output #1 on In Position
SSE—Enable/Disable Communication Error Checking
SSG—Clear/Save the Command Buffer on Limit
SSH—Clear/Save Command Buffer on Stop
ST—Shutdown
T—Time Delay
TR—Wait For Trigger
U—Pause and Wait for Continue
V—Velocity
XC—Sequence Checksum
XD—Sequence Definition
XE—Sequence Erase
XP—Set Power-up Sequence Mode
XQ—Sequence Interrupted Run Mode
XR—Run a Sequence
XRP—Sequence Run With Pause
XSD—Sequence Status Definition
XSP—Sequence Status Power-up
XSR—Sequence Status Run
XSS—Sequence Status
XT—Sequence Termination
XU—Upload Sequence
Y—Stop Loop
Z—Reset
#—Address Numbering
%—Reset Communication

C H A P T E R ⑥

Troubleshooting

Section Objectives

The information in this section will enable you to:

❏ Maintain the system to ensure smooth, efficient operation
❏ Isolate and resolve system problems

Reducing Electrical Noise

For detailed information on reducing electrical noise in your system, refer to the current Compumotor Catalog.

Problem Isolation

When your system does not function properly (or as you expect it to operate), the first thing that you must do is identify and isolate the problem. When you accomplish this, you can effectively begin to resolve and eradicate the problem.

The first step is to isolate each system component and ensure that each component functions properly when it is run independently. You may have to dismantle your system and put it back together piece by piece to detect the problem. If you have additional units available, you may want to use them to replace existing components in your system to help identify the source of the problem.

Determine if the problem is mechanical, electrical, or software-related. Can you repeat or re-create the problem? Random events may appear to be related, but they may not be contributing factors to your problem. Investigate the events that occur before the subsequent system problem.

Symptoms	Possible Causes	Solutions	Useful Commands
No motion	Limits active	Check hard wiring Disable limits	R,RA,LD
	Exceeded maximum distance range	Reset absolute counter with PZ command	PZ, D, MPI, PR
	Position error	CPE command too small Check encoder wiring	R,RSE,CPE,DP E
	Already executing a command	Check status commands	R,RB
	Servo gains too small	Increase gains	CPG,CDG,CIL, CIG
	Stop, kill, trigger inputs active	Check wiring to inputs Check input setup	R,RB,IN,IS
	Loop, pause active	Check status commands	R,RB
	Incorrect unit address	Verify unit address Re-issue unit address	R,#
	In absolute mode (already at position)	Issue new or different absolute distance Issue MPI command	R,MPI,MPA
	Unit shutdown	Enable controller	ST, ON R
No communications	RS-232 miswired	Check wiring for RS-232	
	Wrong RS232 settings	9600 Baud, 8 data bits, 1 stop bit, no parity	
	F command enabled	Issue an E command	E,F
Motor runs away	Encoder miswired	Check encoder wiring Verify power to encoder	
	Controller not receiving encoder pulses	With drive disabled turn shaft clockwise (front of motor) PX should read positive counts	ST0,OFF,ST1, ON,DPA,PX
Enable Output Inactive	Position error	Increase allowable position error Decrease move parameters	CPE,DPE, RSE A,V
	Unit disabled	Enable unit	ST0,ON
Unit not responding to commands	Defined a sequence and didn't close with an XT	Issue an XT at the end of a sequence	XD,XT
	Communications disabled	Enable communications	E,F

You may be experiencing more than one problem. You must solve one problem at a time. Document all testing and problem isolation procedures. You may need to review and consult these notes later. This will also prevent you from duplicating your testing efforts.

Once you isolate the problem, take the necessary steps to resolve it. Use the solutions in this chapter. If your system's problem persists, call Compumotor at 800-358-9070.

RS-232C Problems

Use the following procedure to troubleshoot communication problems that you may have with the OEM070.

1. Be sure the host computer's transmit (Tx) wire is wired to the controller's receive (Rx) connection, and the host computer's receive (Rx) wire is wired to the controller's transmit (Tx) connection. Switch the receive and transmit wires on either the host or controller if the problem persists.

CAUTION

OEM070 Rx, Tx, and GND pin outs are not 2, 3, and 7 like most devices.

2. Confirm that the host and peripheral are configured for the same baud rate, 8 data bits, 1 stop bit, and no parity.
3. Use DC common or signal ground as a reference, not earth ground.
4. Cable lengths should not exceed 50 ft. unless you are using some form of line driver, optical coupler, or shield. As with any control signal, be sure to shield the cable-to-earth ground at one end only.
5. To test the terminal or terminal emulation software and the RS-232C cable for proper three-wire communication, unhook the OEM070 and enter a character. You should not receive an echoed character. If you do, you are in half duplex mode. Connect the host's transmit and receive lines together and send another character. You should receive the echoed character. If not, consult the manufacturer of the host's serial interface for proper pinouts.
6. *The following applies only to firmware 92-016637-01 or higher*: If the OEM070 echoes back **&** for each byte sent to

it, a data communication error has occured. To re-establish communication, see the **%** command.

7. *The following applies only to firmware 92-016637-01 or higher*: To extend cable length and/or improve noise immunity, an RS-485 option is available as a custom product. Contact Compumotor's Custom Products department for details.

Software Debugging Tips

This section offers helpful tips for debugging programs and understanding errors and fault conditions. The OEM070 has several tools that you can use to debug a problem in the system. The software tools are listed below:

R—Report Status
RA—Limit Switch Status Report
RB—Loop, Pause, Shutdown, Trigger Status report
RSE—Report Servo Errors
IS—Input Status Report
BS—Buffer Status Report
B—Buffer Status Report

The troubleshooting table also offers possible causes for typical symptoms.

Encoder Problems

Since the OEM070 relies on feedback information, the encoder connections are critical for the unit to operate properly. If you suspect the controller is not receiving good position data, use the following procedure to verify.

1. Disable the drive (**ST1** or **OFF**)
2. Enter a **PZ** command
3. Rotate the motor CW by hand approximately one revolution
4. Enter a PX. It should read approximately 4000 (for a 1000 line encoder)
5. Rotate CCW one revolution.
6. Enter PX command. It should read approximately 0

If your controller did not respond with similar values, the controller is not receiving encoder information. Either the encoder is mis-wired or in need of repair.

Homing Diagrams

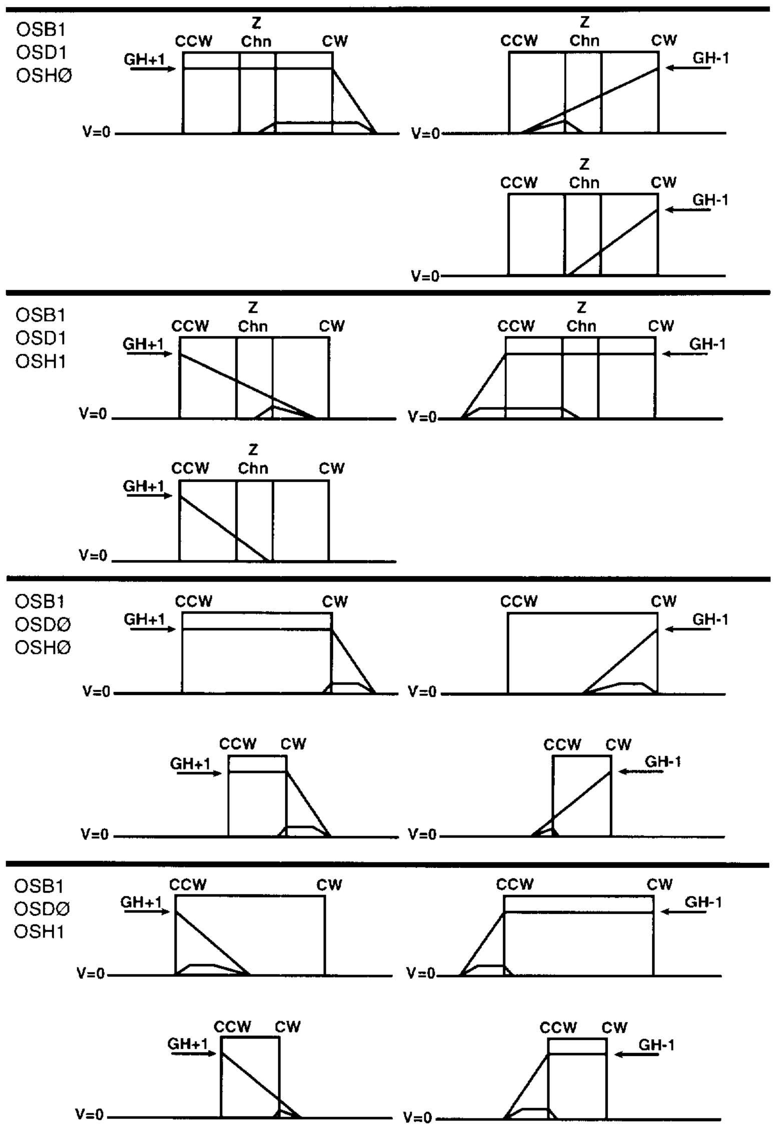
The following diagrams are examples of the many possible homing set-ups. Your parameters may vary and the results may vary slightly depending on your settings.

The CW side of the home pulse is the side closest to the CW limit. The CCW side of the home pulse is the side closest to the CCW limit.

The long pulse diagrams are indicative of situations where the motor decelerates while remaining inside the home pulse width due to the rapid homing deceleration or a very wide home pulse. The short pulse diagrams are indicative of situations where the motor decelerates through the home pulse width due to slow deceleration or a very narrow pulse width.

If an end-of-travel limit is hit during the initial homing, refer to the homing diagram for the opposite direction of travel.

The diagrams are drawn as a general guide. Velocity levels and slopes are drawn to indicate the general move profile the motor will make during the go home move. The vertical axis is velocity and the horizontal axis the position in relation to the home input transitions. Some lines are drawn as closely as possible together to indicate identical velocities, yet remain discernible.



Homing Diagrams

Returning your System

If you must return the OEM070 for repairs, use the following steps:

1. Get the serial number and the model number of the defective unit, and a purchase order number to cover repair costs in the event the unit is determined to be out of warranty.

2. In the USA, call your Automation Technology Center (ATC) for a Return Material Authorization (RMA) number. Returned products cannot be accepted without an RMA number. If you cannot obtain an RMA number from your ATC, call Parker Compumotor's Customer Service Department at (800) 722-2282.

Ship the unit to:

> Parker Hannifin Corporation
> Compumotor Division
> 5500 Business Park Drive, Suite D
> Rohnert Park, CA 94928
> Attn: RMA # xxxxxxxxx

3. In the UK, call Parker Digiplan for a GRA (Goods Returned Authorization) number. Returned products cannot be accepted without a GRA number. The phone number for Parker Digiplan Repair Department is 0202-690911. The phone number for Parker Digiplan Service/Applications Department is 0202-699000.

Ship the unit to:

> Parker Digiplan Ltd.
> 21, Balena Close,
> Poole, Dorset,
> England. BH17 7DX

4. Elsewhere: Contact the distributor who supplied the equipment.

Index

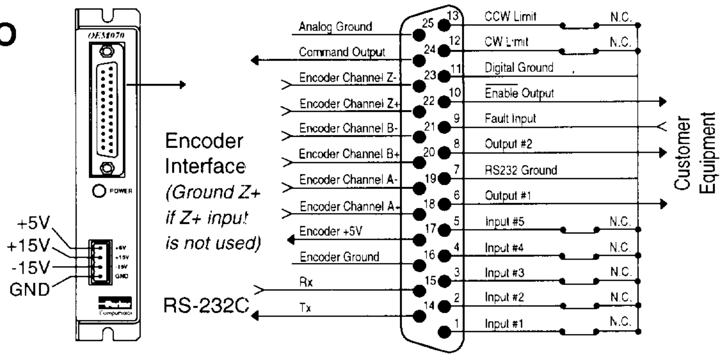
Symbols
"—Quote *68*
22, 92
% *92*
^H—Delete *51*
A
Acceleration *35*
Acceleration Range *23*
Actual position *26*
Address *92*
B
Buffer status *35*
C
Carriage return *44*
Checksum *83*
Command mnemonic *33*
Command type *33*
Commanded position *25*
D
Daisy chain *22, 92*
Debugging *98*
Default setting *34*
Derivative gain *27, 36, 40*
Derivative sampling period *27, 39*
Digital servo loop–update time *23*
Direction *52*
Disable communications *49*
Distance *45*
E
Enable communications *48*
Enable input *20*
Encoder
  inputs *20*
  specifications *21*
Encoder problems *99*
Encoder resolution *48*
End of loop *55, 60*
End-of-travel limits *18, 56, 70, 78*
Error window *31*
F
Fault input *19*
Filter time constant *45*
Front panel LEDs *97*
G
General purpose inputs *16*
Go *49, 53*
Go home *50, 72*
Go input *17*
H
Home limit *17, 50, 53, 64, 65*
Homing Diagrams *99*
Homing function set-ups *62*
I
In Position window *31*
  output #1 on In Position *77*
  position criteria *41*
  time criteria *43*
Input functions *52*
Input status *54*
Inputs *16*
Installation preparation *8*
Integral gain *28, 36, 37, 42*
Integral limit *28, 37, 42*
K
Kill *17, 53, 54*
L
Limit disable *56*
Limits *18*
Loop *55, 60, 71, 91*
M
Mode continuous *57*
Mode normal *58*
Mode position absolute *58*
Mode position incremental *59*

O
OEM070
  Block Diagram *12*
  Description *11*
  Features *11*
OFF *60*
ON *61*
Outputs *16, 18, 61*
P
Pause *40, 66, 71, 81, 87*
PID tuning *25*
Position *25*
Position, actual *46*
Position error *31, 38, 43, 47*
Position range *23*
Position report *65*
Position reset *67*
Power supply connections *15*
Power supply requirements *15*
Power-up *88*
Power-up sequence *84*
Proportional gain *27, 38, 44*
Proportional Integral Derivative *25*
R
Report servo errors *74*
Reset *91*
Reset Communication *92*
Return servo gains to factory settings *72*
Revision level *75*
RS-232C *19, 77*
  daisy chaining *22, 92*
  problems *97*
  specifications *23*
Run a sequence *86*
S
Sampling period *39, 45*
Scanning inputs *76*
Sequence definition *83*
Sequence erase *84*
Sequence select inputs *17, 84*
Sequence termination *89*
Sequence upload *90*
Sequences *53, 83*
Ship kit *13*
Shutdown *71, 79*
Software switch function status *76*
Status *69, 70, 71, 72, 87, 88*
Stop *17, 53, 75, 79*
T
Target position *26*
Time delay *80*
Triggers *17, 52, 71, 80*
Troubleshooting *95*
Tuning *25*
Tuning Procedure *28*
U
Update Time–Digital Servo Loop *23*
V
Velocity *82*
  accuracy *23*
  range *23*
  repeatability *23*
Velocity streaming mode *68, 73*
W
Warnings & Cautions *9*
Z
Z channel *64*

Summary of Commands

A–Acceleration
B–Buffer Status
BCDG–Buffered Configure Derivative Gain
BCIG–Buffered Configure Integral Gain
BCIL–Buffered Configure Integral Limit
BCPE–Buffered Configure Position Error
BCTG–Buffered Configure Derivative Sampling Period
BS–Buffer Size Status
CDG–Configure Derivative Gain
CEW–Configure In Position Error Window
CIG–Configure Integral Gain
CIL–Configure Integral Limit
CIT–Configure In Position Time
CPE–Configure Maximum Position Error
CPG–Configure Proportional Gain
CR–Carriage Return
CTG–Configure Filter Time Constant
D–Distance
DPA–Display Position Actual
DPE–Display Position Error
DVA–Display Velocity Actual
E–Enable Communications
ER–Encoder Resolution
F–Disable Communications
G–Go
GH–Go Home
^H–Delete
H–Set Direction
IN–Set Input Functions
IS –Input Status
K–Kill
L–Loop
LD–Limit Disable
LF–Line Feed
MC–Mode Continuous
MN–Mode Normal
MPA–Mode Position Absolute
MPI–Mode Position Incremental
N–End of Loop
O–Output
OFF–Servo Disable
ON–Servo Enable
OS–Report Homing Function Set-Ups
OSA–Define Active State of End-of-Travel Limits
OSB–Back Up To Home
OSC–Define Active State of Home Switch
OSD–Enable Encoder Z Channel for Home
OSH–Reference Edge of Home Switch
PR–Report Commanded Position
PS–Pause
PX–Report Absolute Encoder Position
PZ–Set Absolute Counter to Zero
"–Quote
Q1–Enter Velocity Profiling Mode
QØ–Exit Velocity Profiling Mode
R–Request Indexer Status
RA–Limit Switch Status Report
RB–Loop, Pause, Shutdown, Trigger Status Report
RC–Homing Status Report
RFS–Return Servo Gains to Factory Settings
RM–Rate Multiplier in Velocity Streaming Mode
RSE–Report Servo Errors
RV–Revision Level
S–Stop
SN–Scan
SS–Software Switch Function Status
SSA–RS-232C Echo Control
SSC–Output #1 on In Position
SSE–Enable/Disable Communication Error Checking
SSG–Clear/Save the Command Buffer on Limit
SSH–Clear/Save Command Buffer on Stop
ST–Shutdown
T–Time Delay
TR–Wait For Trigger
U–Pause and Wait for Continue
V–Velocity
XC–Sequence Checksum
XD–Sequence Definition
XE–Sequence Erase
XP–Set Power-up Sequence Mode
XQ–Sequence Interrupted Run Mode
XR–Run a Sequence
XRP–Sequence Run With Pause
XSD–Sequence Status Definition
XSP–Sequence Status Power-up
XSR–Sequence Status Run
XSS–Sequence Status
XT–Sequence Termination
XU–Upload Sequence
Y–Stop Loop
Z–Reset
#–Address Numbering
%–Reset Communication

I/O


OEM070
POWER
+5V
+15V
-15V
GND
Encoder Interface
(Ground Z+ if Z+ input is not used)
RS-232C
Analog Ground
Command Output
Encoder Channel Z-
Encoder Channel Z+
Encoder Channel B-
Encoder Channel B+
Encoder Channel A-
Encoder Channel A+
Encoder +5V
Encoder Ground
Rx
Tx
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
CCW Limit
CW Limit
Digital Ground
Enable Output
Fault Input
Output #2
RS232 Ground
Output #1
Input #5
Input #4
Input #3
Input #2
Input #1
N.C.
Customer Equipment