Compumotor

Model 301 Indexer User Guide

Compumotor Division
Parker Hannifin Corporation
p/n 88-011728-01 A

Parker

Table Of Contents

List Of Figures

List Of Tables

How To Use This 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 is intended to help you find and use information in this user guide.

Assumptions

To use this user guide effectively, you should have a fundamental understanding of the following information.

- IBM (or IBM-compatible) computer experience
- Basic electronics concepts (voltage, switches, current, etc.)
- Basic motion control concepts (torque, velocity, distance, force, etc.)
- Basic serial communication concepts (e.g., RS-232C)
- PLC programming

With this level of understanding, you can effectively use this user guide to install, develop, and maintain your system.

Contents of This User Guide

This user guide contains the following information.

Chapter 1: Introduction

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

Chapter 2: Getting Started

This chapter contains a list of items you should have received with your shipment. It will help you become familiar with the system and ensure that each component functions properly. You will configure the system properly in this chapter.

Chapter 3: Installation

This chapter will help you properly mount the system and make all electrical connections. Upon completion of this chapter, your system should be completely installed and ready to perform basic operations.

Chapter 4: Application Design

This chapter will help you customize the system to meet your application's needs. Important application considerations are discussed. Sample applications are provided.

Chapter 5: Software Reference

This chapter explains Compumotor's X-Series programming language in detail. It describes command syntax and system parameters that affect command usage. An alphabetical list of all commands, with a syntax and command description for each command is included.

Chapter 6: Hardware Reference

This chapter contains information on system specifications (dimensions and performance).

Chapter 7: Maintenance & Troubleshooting

This chapter describes Compumotor's recommended system maintenance procedures. It also provides methods for isolating and resolving hardware and software problems.

Installation Process Overview

To ensure trouble-free operation, you should pay special attention to the environment in which the Model 301 will operate, the layout and mounting, and the wiring and grounding practices used. These recommendations are intended to help you easily and safely integrate the Model 301 into your manufacturing facility. Industrial environments often contain conditions that may adversely affect solid state equipment. Electrical noise or atmospheric contamination, may also affect the Model 301.

Developing Your Application

Before you develop and implement your application, there are several issues that you should consider and address.

1. Clarify the requirements of your application. Clearly define what you expect the system to do.
2. Assess your resources and limitations. This will help you find the most efficient and effective means of developing and implementing your application.
3. Follow the guidelines and instructions outlined in this user guide. Do not skip any steps or procedures. Proper installation and implementation can only be ensured if all procedures are completed in the proper sequence.

Installation Preparation

Before you attempt to install this product, you should complete the following steps:

1. Review this entire 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, Getting Started.*
4. Perform as many basic moves and functions as you can with the preliminary configuration. You can only perform this task if you have reviewed the entire user guide. You should 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 used or become familiar with the system's features, carefully read *Chapter 3, Installation.*
6. After you have read Chapter 3 and clearly understand what must be done to properly install the system, you should begin the installation process. Do not deviate from the sequence or installation methods provided.
7. Before you begin to 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 any potential system difficulties before they affect your system's operation.

Conventions

To help you understand and use this user guide effectively, the conventions used throughout this user guide are explained in this section.

Commands

All commands that you are instructed to enter are displayed in all capital letters, just as they appear on the terminal (vertically). A one-line explanation of the command is provided next to each example. The command is displayed in boldface. Be sure to separate each command with a space (press the space bar). Press the carriage return key to execute the commands on a specific line. In this user guide, commands are often shown in a vertical fashion so that a short explanation of each command can be provided. Refer to the example below.

Command	Description
> A5	Sets acceleration to 5 rps^2
> V5	Sets velocity to 5 rps
> D1ØØØ	Sets distance to 1,000 steps
> G	Executes the move (Go)

On your computer screen or terminal, the command string shown above would actually look like the example shown below.

```
> A5 V5 D1ØØØ G<cr>
```

Responses are set in all capital letters, as they are on the terminal. An example is provided below.

Command	Response
> RV	*92-011006-01A4

The system generally ignores command syntax that is not within the valid range for a specific command (valid ranges are provided in *Chapter 5, Software Reference*). Compumotor does not guarantee system performance when the system executes commands that contain invalid syntax or are outside of the valid range.

Warnings & Cautions

Warning and caution notes alert you to possible dangers that may occur if you do not follow instructions correctly. Situations that may cause bodily injury are presented as warnings. Situations that may cause system damage are presented as cautions. These notes will appear in bold face and the word warning or caution will be centered and in all capital letters. Refer to the examples shown below.

WARNING

Do not touch the motor immediately after it has been in use for an extended period of time. The unit will be hot.

CAUTION

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

Related Publications

The following publications may be helpful resources.

Seyer, Martin. *RS-232C Made Easy: Connecting Computers, Printers, Terminals and Modems.* Englewood Cliffs, New Jersey: Prentice Hall, Inc., 1984

Current Parker Compumotor Motion Control Catalog

Schram, Peter (editor). *The National Electric Code Handbook (Third Edition).* Quincy, MA: National Fire Protection Association

Operations manual for the PLC that you will use with the Model 301 Indexer

Chapter 1. Introduction

Chapter Objectives

The information in this chapter will enable you to understand the product's basic functions & features.

Product Description

The Model 301 Indexer is designed for plug-in compatibility with the Texas Instruments' Series 305™ PLC and the GE Fanuc Series One™ PLC. Conservative electrical design and complete optical isolation of external signals maintain the industrial ruggedness of the PLC. The Model 301 is a single axis controller that accepts quadrature encoder feedback. Figure 1-1 shows the Model 301's front panel.

With a standard 3-wire RS-232C interface, the Model 301 uses an extended form of Compumotor's X Series Language for ease of programming and flexibility of interactive control with the PLC rack. The Model 301 has an on-board editor that provides complete program creation, modification, and monitoring through a remote terminal. As programs are written, they are automatically stored in nonvolatile memory. Execution may begin at any point in the stored program as designated by the PLC program or through the RS-232C port. The point at which motors are commanded to move depends on PLC contacts, time, and position information. You can even program the Model 301 to turn on and off outputs of the PLC within the execution of its own program. This bus-compatible product provides complete backplane integration between the PLC and the motor.

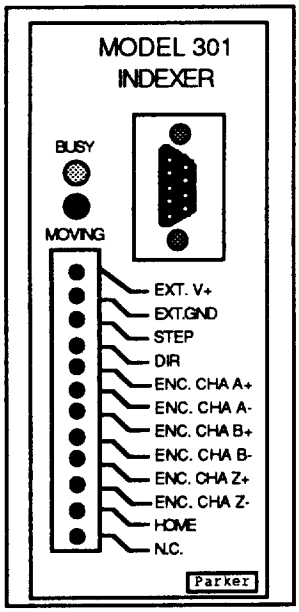

Figure 1-1. Model 301—Front Panel

There is a home input line for the axis to decouple the scan time of the PLC for sensing motor home positions.

The Model 301 controls a motor axis independent of the PLC's CPU. The indexer is not burdened by PLC scan time limitations. The scan time of the PLC is only pertinent in the communication between the Model 301 and the PLC through the backplane. Figure 1-2 is an example of a Model 301 configuration.

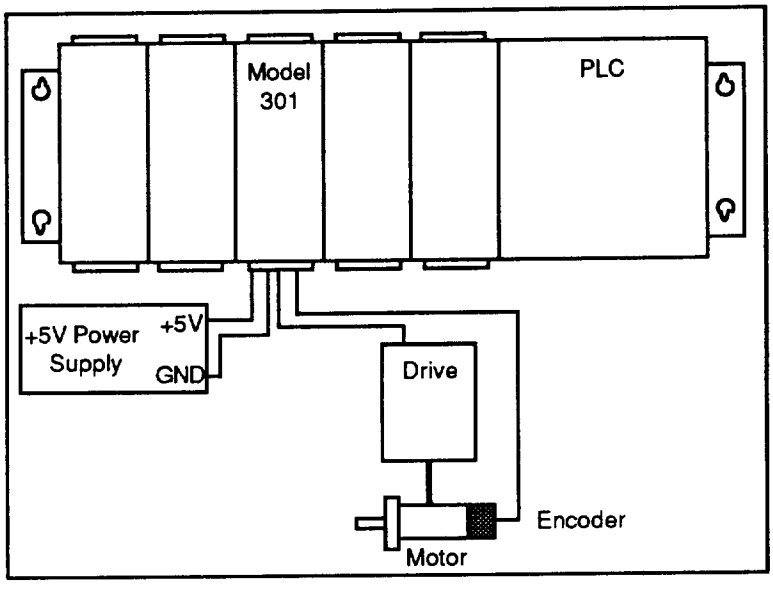

Figure 1-2. Sample Model 301 Configuration

Product Features

- Single-axis control with incremental encoder feedback
- Plug-in bus compatibility with the TI Series 305™ or GE Fanuc One™ PLC
- Standard RS-232C programming interface and complete online editing of the stored program
- Programmable position, direction, velocity, and acceleration for precise motion control
- 5VDC optically isolated inputs and outputs provide high electrical noise immunity
- 2K or 8K battery-backed RAM memory to store multiple programs
- Up to 63 separate indexer program entry points may be defined—complete flexibility of indexer program execution from the PLC program
- Integral high-speed inputs for accurate home sensor and sensor-interactive control
- Commands support complex move profiles; velocity changes on-the-fly triggered by time, position, or PLC contacts
- Conditional control of program flow with **IF** statements based on the state of PLC contacts
- On-line debugging with the Trace (**XTR**) command
- PLC output contacts may be set or cleared from the Model 301's programs

Chapter 2. Getting Started

Chapter Objectives

The information in this chapter will enable you to:

- Verify that each component of your system has been delivered safely
- Become familiar with system components and their interrelationships
- Bench test the system

What You Should Have

Inspect your Model 301 shipment upon receipt for damage to its shipping container. Report any damage to the shipping company immediately. Parker Compumotor cannot be held responsible for damage incurred in shipment. The items listed in Table 2-1 should be present and in good condition.

Part/Quantity	Part Number
Model 301	MODEL 301
Model 301 User Guide (1)	88-011728-01 A
Indexer/Drive Cables (1)	71-011159-10
Connector (1)	43-011058-01
RS-232C Cable (1)	71-011319-10

Table 2-1. Model 301 Ship Kit List

Basic System Configuration

Figure 2-1 provides an overview of the connections you will have to make to operate the Model 301. Each of the connections will be discussed in detail in this chapter.

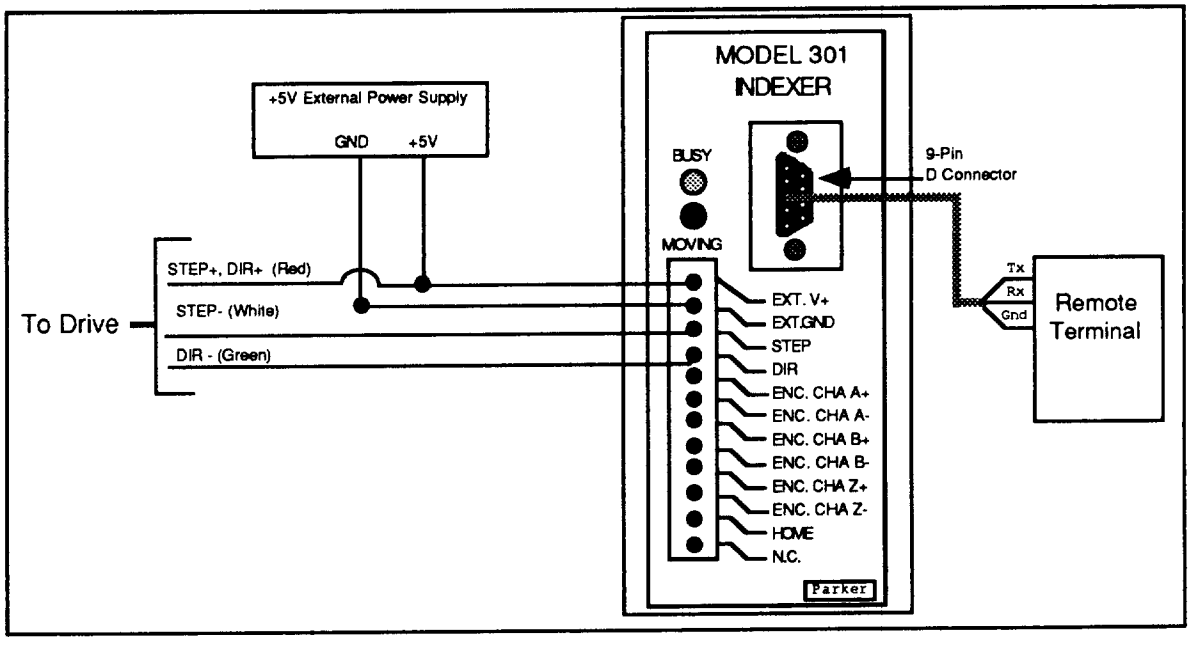

Figure 2-1. Basic System Wiring Diagram

Indexer Setting

Before you insert the Model 301 into one of the PLC's available ports, you must check the following indexer setting:

- In a standard rack system, the jumper (refer to Figure 2-2) must be placed over pins **1 and 2**.
- In an extended PLC cage, the jumper (refer to Figure 2-2) must be placed over pins **2 and 3**. The jumper disables the 1XX (octal) addresses on the card. Figure 2-2 shows the location of the jumper.

Push in the buttons at the top and bottom of the Model 301's front panel to insert/remove the unit from the rack.

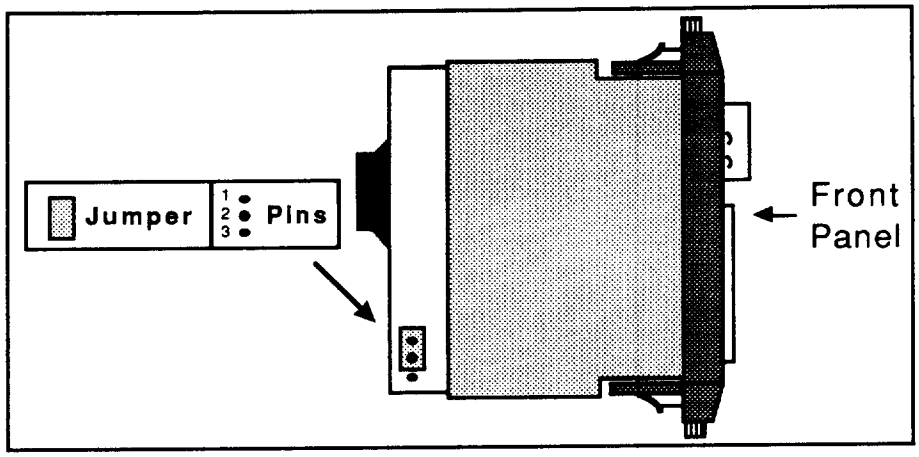

Figure 2-2. Location of Jumper for Standard Card Cage

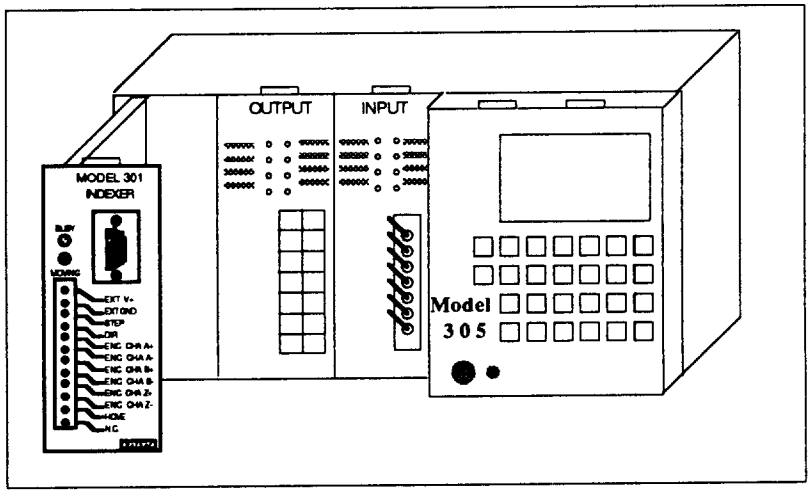

Figure 2-3. Inserting a Model 301 into a Rack

Establish Communications

You can program the Model 301 with any ASCII device that communicates via standard RS-232C. The terminal's parameters for RS-232C communications should be:

- Baud Rate: 9,600
- Stop Bit: 1
- Data Bits: 8
- Echo: Off

The Model 301's echo function is always on. Attach the RS-232C connection from your ASCII device to the 9-pin D connector on the front panel of the Model 301 (refer to Figure 2-4). If you are using an IBM PC, an RS-232C cable is provided.

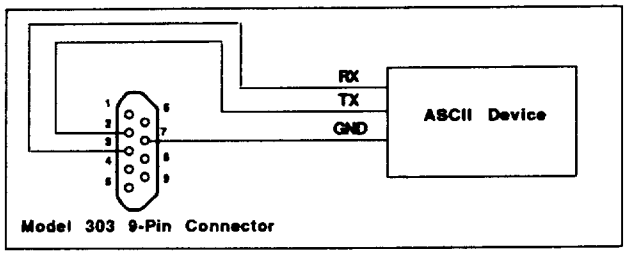

Figure 2-4. RS-232C Connection

Testing RS-232C

To ensure that the RS-232C connection is operating properly, complete the following steps.

1. Apply power to PLC. If your terminal is already on, you will see a message indicating that the indexer is ready. Below the message, a prompt (>) should be present. Press the Return key. A new prompt should appear. If you powered up your terminal after the Model 301, press the Return key. The terminal should display a prompt (>).

2. Type R and press the Return key. The Model 301 will display the Dynamic Data, Registered Data, and Active Parameters. A sample Status Report (R) command's response is shown below.

```
*DYNAMIC DATA
*   INPUT BYTE  (BØ - B7) = ØØØØØØØØ
*   OUTPUT BYTE (BØ - B7) = ØØØØØØØØ
*   HOME: = 1
*REGISTERED DATA
*   Inputs:  I1=Ø I2=Ø I3=Ø I4=Ø I5=Ø
*   Outputs: O1=Ø O2=Ø O3=Ø O4=Ø O5=Ø O6=Ø
*   Motor Position =+Ø
*   Encoder Position =+Ø
*ACTIVE PARAMETERS:
*   MR25ØØØ GHV 1 GHF Ø.1
*   FRØØØØØØØØ
*   ER 4ØØØ DB5 DW25Ø CG8
*   VSØ VØ AØ
*   MPI D+Ø TØ.5 LØ
```

If you receive the data listed above, your RS-232C connection is working properly. If you do not receive any response, check your wiring, and perform the steps again. Some set-up values may be different. **Before proceeding, remove power from the PLC.**

Drive/Indexer Connection

Connect the external power (EXT. V+), external ground (EXT. GND), and drive outputs as shown in Figure 2-5. **Compumotor recommends that you make multiple connections to a terminal block and a single connection to the Model 301.**

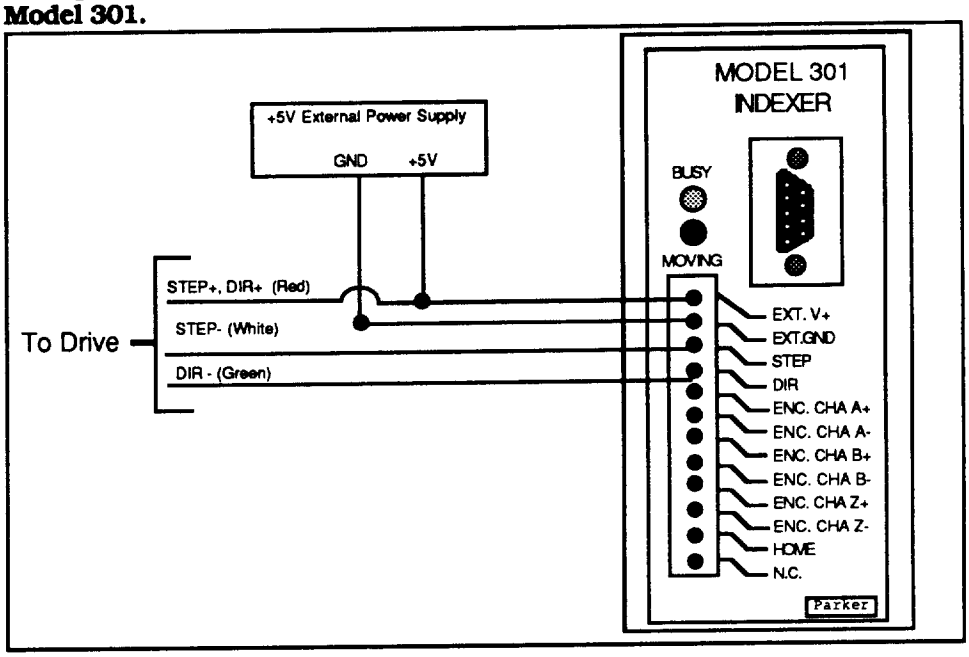

Figure 2-5. Model 301/Drive Connections

Setting Drive Functions

Refer to the manual that accompanied the drive you are using with the Model 301 Indexer. ***Follow the instructions provided in the manual to configure the motor and drive and complete any settings*** (e.g., motor current). Before proceeding, be sure that you have properly made all connections and settings:

- Indexer Settings
- RS-232C Connection
- Indexer/Drive Connection
- Drive Functions
- Drive/Motor Connections

Making an Open Loop Move

To ensure that you have wired the Model 301 and the other components of your system properly, use the following instructions to perform a move.

Step 1

Apply power to the PLC, external power supply, and the drive.

Step 2

The motor resolution must be set to the same resolution as the drive. This example assumes 25,000 steps per rev. Use the **MR** command to set the motor resolution to 25,000 motor steps/rev (**MR25ØØØ**). To ensure that the motor resolution is properly set, issue the Status Report (**R**) command. Under the Active Parameters portion of this report, the current motor resolution is shown. The motor resolution is highlighted in the example below. This is only part of the report. Refer to the **R** command description in *Chapter 5, Software Reference* for the entire report.

```
*ACTIVE PARAMETERS:
*   MR25ØØØ  GHV1        GHFØ.1
*   FR ØØØØØØØØ
*   ER 4ØØØ DB 5 DW 25Ø CG 8
*   VSØ         VØ   AØ
*   MPI         D+   TØ.5   LØ
```

Step 3

Using the terminal, enter the following commands:

```
> MPI
> FRØØØØØØØØØ
> A5 V5 D25000 G<cr>
```

Please note the spaces between the commands and the carriage return after the Go (G) command. A description of each command is given below.

Command	Description
> MPI	Sets to Incremental mode
> FRØØØØØØØØØ	Clear any saved FS parameters
> A5	Sets acceleration to 5 rps^2
> V5	Sets velocity to 5 rps
> D25000	Sets distance to 25,000 steps
> G	Executes the command (Go)

The motor should move 25,000 steps in the CW direction. The moving (Red) LED will be on while steps are being sent. If the motor does not move, check the wiring and refer to *Chapter 7, Troubleshooting.*

To make the motor move in the CCW direction, enter the following commands:

Command	Description
> H	Changes the direction of movement
> G	Executes the command (Go)

The previous acceleration, velocity, and distance parameters are used in this move. It is simply performed in the CCW direction.

Making a Closed Loop Move

To make a closed-loop move a 1,000 line (4,000 pulse post-quadrature) encoder must be wired to the Model 301. When ready, enter the following commands.

Command	Description
> PZ	Zero motor and encoder positions
> FSB1	Enter closed loop mode
> FSC1	Enable position maintenance
> FSD1	Enable stop on stall
> FSH1	Enable stall detect
> ER4ØØØ	Set encoder resolution to 4,000 pulses/rev
> A5 V5 D4ØØØ G	Motor should move 1 rev CW

When the move is complete enter R.

The partial response from the terminal should be:

```
REGISTERED DATA
     Inputs:
     Outputs:
     Motor Position = +25ØØØ
     Encoder Position = +4ØØØ
```

If the encoder position is Ø or a message is sent indicating that the motor stalled, the encoder is not properly connected (refer to *Chapter 7, Maintenance and Troubleshooting*).

If the encoder position is negative (-), the encoder phases are reversed (refer to *Chapter 7, Maintenance and Troubleshooting*).

When the motor is finished moving enter:

```
> H G
> R
```

The motor and encoder positions should both be at 0 again.

8-Bit Outputs

The Model 301 Indexer has outputs to the PLC. These outputs are transmitted on the *upper octal addresses.* If the card is in the slot to the left of the Series One™ CPU, the outputs from the indexer card (which are inputs to the PLC) would occupy an address space from 100 to 107. The address 100 corresponds to BØ and 107 corresponds to B7. Refer to the example below (refer to Figure 3-1 for more information on rack addresses). *If you are using an extended card cage, the output bit addresses are not available (refer to the* ***Indexer Setting*** *section earlier in this chapter).*

The PLC may use the most significant two output bits (B7 & B6) to determine indexer status. These two bits indicate whether the indexer is executing a user program (i.e., **`Program Busy`**) or if the indexer is sending pulses (i.e., **`Motor Busy`**). The other bits are general-purpose outputs that are controlled by the indexer and its program. The protocol of these outputs is shown below:

B7	B6	B5	B4	B3	B2	B1	BØ
Busy	Program Busy	O6	O5	O4	O3	O2	O1

`B7 = 1:` The motor is moving, the indexer is sending pulses.
`B6 = 1:` A sequence is being executed.
`B5 - BØ = 1:` You can set these general outputs to any logic level in immediate mode or under program control. *Outputs are labeled from 1 to 6. Zero (Ø) is not used.*

The outputs are initialized to a logic zero on power up. If the PLC is turned from RUN mode to PROGRAM mode or LOAD mode, all of the outputs will go to a logic one (1) and the execution of any commands will be disengaged. When the unit is returned to RUN mode, the outputs will be reset to a zero state.

8-Bit Inputs

The Model 301 Indexer card looks like an 8-bit output card to the Series One™ PLC. If the card is in the slot to the left of the Series One™ CPU, the eight inputs (which are outputs from the PLC) would have the address space from ØØ to Ø7. The PLC output at address ØØ is the BØ input bit to the indexer card and address Ø7 corresponds to B7. The input addresses that correspond to BØ - B4 are shown below.

B4	B3	B2	B1	BØ
I5	I4	I3	I2	I1

You must use the command structure shown in Table 2-2 to issue commands to the indexer card. The most significant bit (MSB)—B7—is the command valid strobe line. When this line is toggled from low to high, the other 7 bits have valid data. The strobe line must stay high for at least 1 ms.

COMMAND	B7	B6	B5	B4	B3	B2	B1	BØ
XG#	Ø—>1	Ø	A5	A4	A3	A2	A1	AØ
KILL	Ø—>1	1	Ø	Ø	Ø	Ø	Ø	1
KILL & RESET OUTPUTS	Ø—>1	1	Ø	1	Ø	Ø	Ø	1
STOP	Ø—>1	1	Ø	Ø	Ø	Ø	1	Ø
STOP & RESET OUTPUTS	Ø—>1	1	Ø	1	Ø	Ø	1	Ø
HOME (+)	Ø—>1	1	Ø	Ø	Ø	1	x	x
HOME (-)	Ø—>1	1	Ø	Ø	1	Ø	x	x
PAUSE	Ø—>1	1	Ø	Ø	1	1	1	1
RESUME	Ø—>1	1	Ø	Ø	1	1	Ø	Ø
R	Ø—>1	1	Ø	1	1	Ø	Ø	Ø
PR	Ø—>1	1	Ø	1	1	Ø	Ø	1
PR	Ø—>1	1	Ø	1	1	Ø	1	Ø
PR	Ø—>1	1	Ø	1	1	Ø	1	1
RESET OUTPUTS	Ø—>1	1	Ø	1	Ø	Ø	x	x
GENERAL INPUTS	x	1	1	I5	I4	I3	I2	I1
—> = logic transition x = don't care								

Table 2-2. Input Bit Command Structure

The following section defines each of the commands shown in Table 2-2.

XG# This command executes a user program beginning from the sequence that you define (#0 - #63). The least six bits (AØ - A5) are the program sequence pointer.

The address lines have the following weights:

A5	A4	A3	A2	A1	AØ
32	16	8	4	2	1

To run sequence #35; AØ, A1, and A5 (1 + 2 + 32 = 35) must be active. If you instruct the indexer to execute a program that does not exist, it will respond with a question mark (?). Instructing the indexer to execute sequence Ø will run the first program in its memory.

KILL This command allows you to terminate an output pulse train immediately, with no deceleration.

KILL & RESET OUTPUTS This command allows you to terminate an output pulse train immediately (with no deceleration) and reset all six of the general programmable outputs to a logic low. Outputs are cleared at start of deceleration.

STOP This command allows you to decelerate the motor to a stop.

STOP & RESET OUTPUTS This command allows you to decelerate the motor to a stop and reset all six of the general programmable outputs to a logic low. Outputs are cleared at the start of deceleration.

HOME+ The Home+ command searches for the home switch in the CW direction. When the home switch for the axis goes low, the indexer searches for the CW edge of the home switch (see Figure 3-5).

HOME- The Home- command searches for the home switch in the CCW direction. When the home switch for the motor goes low, the indexer searches for the CW edge of the home switch (see Figure 3-5).

PAUSE This command allows you to interrupt program execution. Motion will be decelerated (the same as an S command).

RESUME The Resume command continues the execution of an interrupted sequence.

PR — POSITION REPORT This is a Position Report command. It provides axis encoder and motor position information. The information to be transmitted is selected by setting appropriate bits:

- BØ - Motor Position
- B1 - Encoder Position

If both positions are requested, motor position is sent first.

R — DISPLAY STATUS Report current indexer status over RS-232C port

RESET OUTPUTS This command allows you to reset all six of the general programmable outputs to a logic low.

GENERAL INPUTS

You can use these inputs as end-of-travel limits or for program conditional branching. Note that the I's are labeled from 1 to 5. Zero (Ø) is not used. Upon power up, the inputs are initialized low (logic Ø). Information from the PLC may be dynamically transmitted to the indexer card via the general inputs when B5 and B6 are both high (logic 1). If either B5 or B6 go low, the last state of the inputs are saved in the indexer card. The strobe line is not used to latch the state of these inputs.

When you switch the PLC from Run mode to the Program or Load modes, the general inputs are reset to logic Ø until the PLC re-programs a specific input.

Sample Program for 5-Slot Rack

If you use the Model 301 in a 5-slot rack, you can use the following program example. The program can be used with an input simulator, output module, the Model 301, and a PLC programmer. The program allows the Model 301 to control the output module's outputs or enables external devices to send commands to the Model 301 via the input module. The modules must be arranged in the rack as shown in Figure 2-6.

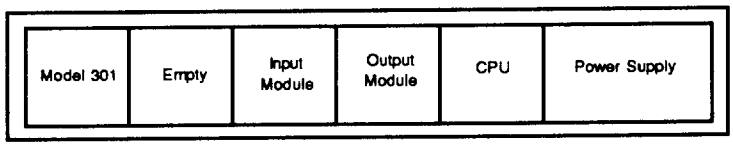

Figure 2-6. 5-Slot Rack Component Arrangement

Enter the following program with the PLC programmer.
Be sure the PLC's key is set to PROG.

Line	Program	Description
1	CLR, SHF, 348, DEL, NXT	Clears PLC memory
2	STR, SHF, 10, ENT, OUT, SHF, 30, ENT	Writing from input module to indexer module. The PLC reads the status of the input module and sends the command inputs to the Model 301.
4	STR, SHF, 11, ENT, OUT, SHF, 31, ENT	
6	STR, SHF, 12, ENT, OUT, SHF, 32, ENT	
8	STR, SHF, 13, ENT, OUT, SHF, 33, ENT	
1Ø	STR, SHF, 14, ENT, OUT, SHF, 34, ENT	
12	STR, SHF, 15, ENT, OUT, SHF, 35, ENT	
14	STR, SHF, 16, ENT, OUT, SHF, 36, ENT	
16	STR, SHF, 17, ENT, OUT, SHF, 37, ENT	
18	STR, SHF, 130, ENT, OUT, SHF, 0, ENT	Writing from indexer module to output module. The PLC reads the Model 301's outputs and sets the appropriate outputs on the output module.
2Ø	STR, SHF, 131, ENT, OUT, SHF, 1, ENT	
22	STR, SHF, 132, ENT, OUT, SHF, 2, ENT	
24	STR, SHF, 133, ENT, OUT, SHF, 3, ENT	
26	STR, SHF, 134, ENT, OUT, SHF, 4, ENT	
28	STR, SHF, 135, ENT, OUT, SHF, 5, ENT	
3Ø	STR, SHF, 136, ENT, OUT, SHF, 6, ENT	
32	STR, SHF, 137, ENT, OUT, SHF, 7, ENT	

Turning On Outputs

Turn the PLC's key to the RUN position.
To turn on outputs 1, 3, and 5, enter: **> O1Ø1Ø1Ø**
To turn on outputs 2, 4, and 6, enter: **> OØ1Ø1Ø1**

Controlling the Indexer With Remote Inputs

This exercise will teach you how to program and store motion sequences and activate the sequences from a remote input. First, you must use the terminal to create the sequences. *You should enter the **boldface** and **underlined** instructions.*

```
> FRØØØØØØØØ    (Turn off closed loop functions)
> CLR
> Are You Sure (Y/N)? Y
> EXR1
  * (.1)  1: ...

Inserting Sequence 1

   * 1: >A5 V5 D25ØØØ G XT    (Press Enter Key)
*  >(Press Enter Key Again to Exit Edit Mode)
> EXR5
  * (.1)  5: ...

Inserting Sequence 5

   * 5: >A1 V1 D-25ØØØ G XT (Press Enter Key)
*  >(Press Enter Key Again to Exit Edit Mode)
> LST
  1:  A5 V5 D25ØØØ G XT
  5:  A1 V1 D-25ØØØ G XT
  * 1259 BYTES FREE. *
> PZ              Zero Position
> MPI             Incremental move mode
```

The commands are described in detail below.

Command	Description
> CLR	Clears the indexer's memory
> EXR1	Begins definition of Sequence #1
A5	Sets acceleration to 5 rps^2
V5	Sets velocity to 5 rps
D25ØØØ	Sets distance to 25,000 steps
G	Executes the sequence
XT	Ends Sequence #1 definition
> EXR5	Begins definition of Sequence #1
A1	Sets acceleration to 1 rps^2
V1	Sets velocity to 1 rps
D-25ØØØ	Sets distance to 25,000 steps
G	Executes the sequence
XT	Ends Sequence #5 definition
> LST	Lists current sequences #1 & #5
> PZ	Sets the current position to zero
> MPI	Sets the indexer to Incremental mode

We will now use the input simulator to execute these two sequences and to execute other commands. Refer to Table 2-2 for a complete list of the input bit command structure.

First, you will ensure that the motor is set to position Ø. This should have been done with the Set Position Zero (PZ) command you issued earlier. Set the switches on the input simulator to the following settings to execute the Position Report (PR) command:

BØ	1
B1	Ø
B2	Ø
B3	1
B4	1
B5	Ø
B6	1
B7	Ø

Toggle B7 input (turn the input on and then off). The screen should display the axis' motor position as +Ø. Now you can execute sequence #1. Remember the weights of the address lines:

A5	A4	A3	A2	A1	AØ
32	16	8	4	2	1

Set the switches on the input simulator to the following settings to execute sequence #1 (XG1):

BØ	1
B1	Ø
B2	Ø
B3	Ø
B4	Ø
B5	Ø
B6	Ø
B7	Ø

Toggle B7 input. The X axis should turn 25,000 steps in the **CW** direction when sequence #1 is run. Now you will check the axis' position again to determine if it made the move properly. Since it started at **Ø**, it should be at position **25,000** now. Set the input simulator to the following settings to execute the Position Report (PR) command:

BØ	1
B1	Ø
B2	Ø
B3	1
B4	1
B5	Ø
B6	1
B7	Ø

Toggle B7 input. The axis's position should be displayed on the screen as +25ØØØ steps.

You will now execute sequence #5. Set the switches on the input simulator to the following settings to execute sequence #5 (XG5):

BØ	1
B1	Ø
B2	1
B3	Ø
B4	Ø
B5	Ø
B6	Ø
B7	Ø

Toggle B7 input. Toggle Switch B7. The axis should turn 25,000 steps in the **CCW** direction when sequence #5 is executed. Now check the axis' position again. Since it moved 25,000 steps in the **CCW** direction, it should be at position Ø again. Set the input simulator to the following settings to execute the Position Report (PR) command:

BØ	1
B1	Ø
B2	Ø
B3	1
B4	1
B5	Ø
B6	1
B7	Ø

Toggle B7 input. Axis X's position should be +Ø steps. Enter the following commands through the terminal:

Command	Description
> MC	Sets the indexer to Continuous mode
> G	Executes the move (Go)

The axis should begin moving **CCW.** The indexer executes the command parameters that were last used—*sequence #5.* The motor continues to move beyond the -25,000 distance defined in sequence #5 because you are operating in Continuous mode (the distance value has no meaning in this mode). To stop the axis, you will set the input simulator to perform the Stop (S) command, which will decelerate the motor to a stop. Set the input simulator as follows:

BØ	Ø
B1	1
B2	Ø
B3	Ø
B4	Ø
B5	Ø
B6	1
B7	Ø

Toggle B7 input. Toggle Switch B7. The axis should stop. You can try other patterns with the input simulator. Refer to Table 2-2 for additional remote input commands.

Chapter 3. Installation

Chapter Objectives

The information in this chapter will enable you to:

- Insert the unit into the PLC properly
- Connect all electrical system inputs and outputs properly
- Ensure that the complete system is installed properly
- Perform basic system operations

Environmental Considerations

Parker Compumotor recommends that you operate and store your Model 301 under the following conditions:

- Ambient Operating Temperature: 32°F - 122°F (0°C - 50°C)
- Storage Temperature: -22°F - 185°F (-30°C - 85°C)
- Humidity: 0 to 95% non-condensing

The Model 301 is protected against short circuit and over temperature. Compumotor does not recommend that you test these features or operate your system in such a way as to induce short circuiting or overtemperature situations.

Complete System Configuration

Before you proceed with this section, you should have completed all of the steps and procedures contained in *Chapter 2, Getting Started.* You should already be familiar with the set-up procedures for communications, and power.

PLC Port Addressing

Each PLC port has a unique device address. The Model 301 will assume the device address of the port in which it is inserted. You can insert the Model 301 Indexer into any available port. Figure 3-1 shows the standard addresses given to PLC ports.

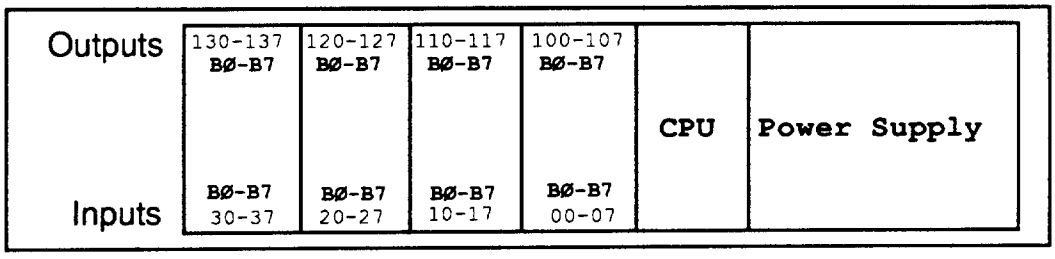

Figure 3-1. PLC Port Addresses

Extended PLC Cages

- In a standard rack system, the jumper (refer to Figure 3-2) must be placed over pins **1 and 2**.
- In an extended PLC cage, the jumper (refer to Figure 3-2) must be placed over pins **2 and 3**.

If you intend to insert the Model 301 into a PLC with an extended cage, you must place the jumper on pins 2 and 3 of the indexer board before inserting the indexer into the port. Figure 3-2 shows the location the jumper. Push in the buttons at the top and bottom of the Model 301's front panel to remove the unit from the rack.

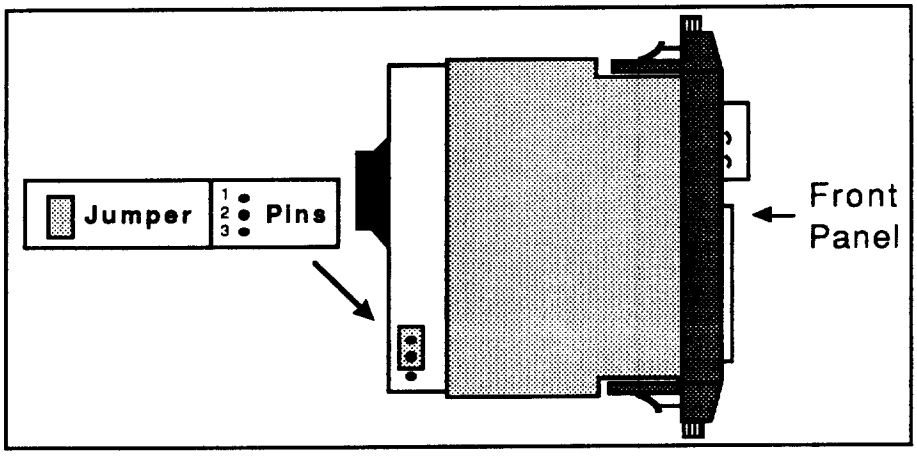

Figure 3-2. Location of Jumper for Standard Card Cage

Indexer Insertion

After you have properly set the jumper (***if necessary***), you can insert the Model 301 Indexer into any available port of the PLC. You may now begin the system connections.

System Connections

This section will help you properly wire the Model 301. Specifically, the following procedures and information will be addressed:

- Wiring Guidelines
- Establishing communications (RS-232C)
- Wiring the external 5VDC power supply
- Wiring the indexer to the drive

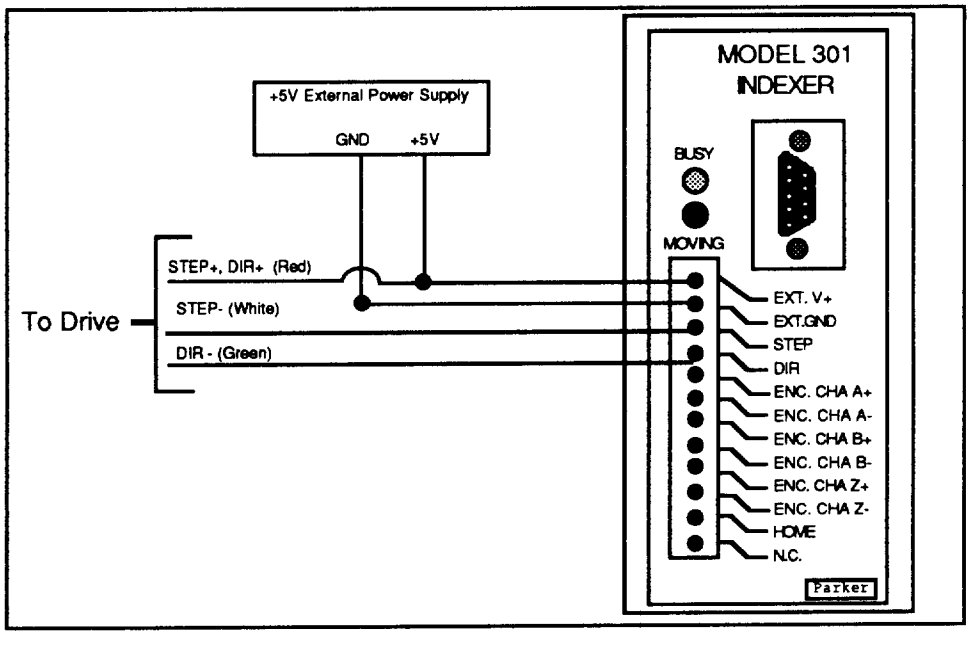

Figure 3-3. Complete Configuration

Wiring Guidelines

Proper grounding of electrical equipment is essential to ensure the safety of personnel. You can reduce the effects of electrical noise due to electromagnetic interference (EMI) by grounding. All Compumotor equipment should be properly grounded. A good source of information on grounding requirements is the National Electrical Code published by the National Fire Protection Association of Boston, MA.

For grounding follow PLC manufacturers recommendations.

Communications

You can program the Model 301 with any ASCII device that communicates via standard RS-232C. The terminal's parameters for RS-232C communications should be:

- Baud Rate: 9,600 (fixed)
- Stop Bit: 1
- Data Bits: 8
- Echo: Off

The Model 301's echo function is always on. The 9-pin D connector on the Model 301's front panel provides the RS-232C connections. The pin out for this connector is defined in Table 3-1.

Pin #	Function
Pin 1	Not Used
Pin 2	TXD, Transmit Signal
Pin 3	RXD, Receive Signal
Pin 4	DTR/CTS, Always set at +10VDC
Pin 5	Signal Ground
Pin 6	Not Used
Pin 7	Signal Ground
Pin 8	Not Used
Pin 9	Not Used

Table 3-1. RS-232C Pin-Out

Refer to *Chapter 2, Getting Started* for communications testing procedures that you can use to ensure proper operation.

External Power Supply

The indexer card is powered by the PLC's rack power supply. The indexer card uses a maximum of 150 mA of the PLC's +9V supply. This is equal to 15 units of load as described in the Series One™ Programmable Controllers Manual (distributed by GE/Fanuc).

To use the Model 301's inputs and outputs (on external connector), you must provide an external +5V power supply. Figure 3-3 illustrates the +5V wiring configuration.

Wiring the Indexer to the Drive

Connect the external power (EXT. V+), external ground (EXT. GND), and drive outputs as shown in Figure 3-3. **Compumotor recommends that you make multiple connections to a terminal block and a single connection to the Model 301.**

Verifying Proper Installation

You should have completely configured your system. This section will help you verify that you have wired the system properly and ensure that it is fully operational. You should have completed testing the RS-232C Communications already (the steps for this test were first discussed in *Chapter 2, Getting Started*).

Homing The Motor

You can initiate the Go Home function by issuing the Go Home (GH) command. When you issue the GH command, you must include the direction that the motor should use to search for home. The home limit input on the Model 301 is optically isolated, and is normally off. You must use a normally open, load-activated switch to ground to determine the home position.

When you command the motor to go home, it begins to move in the direction you specified. It performs this move at the last defined acceleration and velocity rates, and looks for the home limit input to go active. The indexer searches for home to the CW edge. The CW edge of the home switch is defined as the first switch transition that occurs if it is traveling in the CCW direction).

To test the Model 301's homing function, connect a N.O. switch between HOME and EXT GND. Enter the following command string.

Command	Description
> GHV5	Set go home velocity to 5 rps
> GHF.2	Sets final go home velocity to 0.2 rps
> GH+	Instructs the motor to go home in the CW direction

The following events occur when you go home in the CW direction (refer to Figure 3-4):

1. The motor moves in the CW direction at 5 rps.
2. When the home switch is closed and opened, the motor decelerates to a stop, then moves in the CCW direction at the velocity you specified with the Go Home Final Speed (GHF) command.
3. Momentarily close the home switch again to stop the motor.

The following events occur when you go home in the CCW direction (refer to Figure 3-5):

1. The motor moves in the CCW direction until the home switch becomes active.
2. The motor decelerates to a stop and moves in the CW direction until the home switch becomes inactive.
3. The motor creeps to the CW edge of the switch at the velocity you set with the GHF command. The motor stops when the switch becomes active.

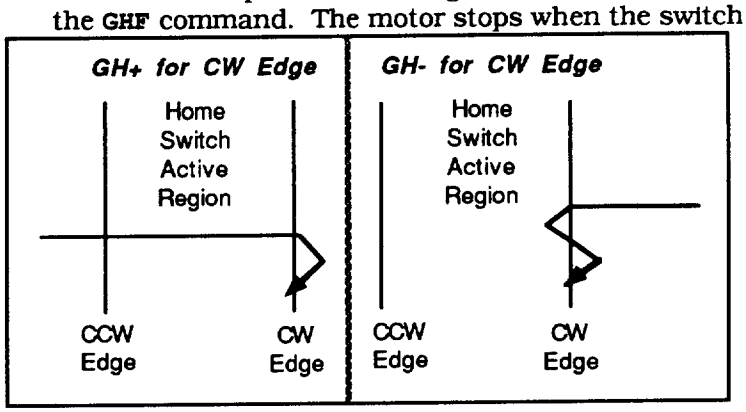

Figure 3-4. Homing Operation

Inputs & Outputs

This section discusses the Model 301's inputs and outputs.

Inputs

The Model 301 has eight inputs, five of which can be used for program control. *Chapter 2, Getting Started* contains a functional description of these inputs. General programmable inputs may be used for conditional branching. The inputs are labeled from 1 to 5 (Ø is not used). Upon power-up, the inputs are initialized to a logic zero state until the PLC reprograms an input specifically. Data from the PLC may be dynamically transmitted to the indexer card via the general inputs when B5 and B6 are both high (logic 1). If either B5 or B6 go low, the last state of the inputs are saved to the indexer card. The strobe line (B7) is not used to latch the state of these inputs. You can also set the state of the inputs with the **TEST** command through the RS-232C port.

Programmable Outputs

The Model 301 has eight output bits. The indexer card's outputs to the PLC are transmitted on the upper octal address (1XX addresses). If the card is in the slot to the left of the Series One™ CPU, the outputs from the indexer card (which are inputs to the PLC) will occupy addresses 100 to 107. Address 100 corresponds to BØ and address 107 is B7. *If you are using an extended rack system, the output bits are not usable.*

The PLC uses the most significant two bits of the outputs to determine indexer status (B6 & B7). ***You cannot define or program these bits.*** These two bits indicate whether the indexer is busy executing a user program (*Program Busy*), or whether it is currently sending out pulses (*Motor Busy*). The protocol of these outputs is shown below:

Address	B7	B6	B5	B4	B3	B2	B1	BØ
Status	Busy	P.Busy	O6	O5	O4	O3	O2	O1

Chapter 4. Application Design

Chapter Objectives

The information in this chapter will enable you to:

- Recognize and understand important considerations that must be addressed before you implement your application
- Understand the capabilities of the system
- Use examples to help you develop your application

Motion Control Concepts

This section discusses basic motion control concepts that you should be familiar with as you develop your application.

Move Profiles

In any motion control application, the most important requirement is precise position, whether it be with respect to time or velocity. A motion profile represents the velocity of the motor during a period of time in which the motor changes position. The type of motion profile that you need depends upon the motion control requirement that you specify. The basic types of motion profiles are described below.

Triangular and Trapezoidal Profiles

For indexing systems, you must define velocity, acceleration, and distance parameters before the system can execute a preset move. The value of these parameters determines the motion profile as either triangular or trapezoidal. A triangular profile results when the velocity and acceleration are set such that the defined velocity is not attained before the motor travels half of the specified distance. This results from either a relatively low acceleration, a relatively high velocity, or both. A triangular profile is shown in Figure 4-1.

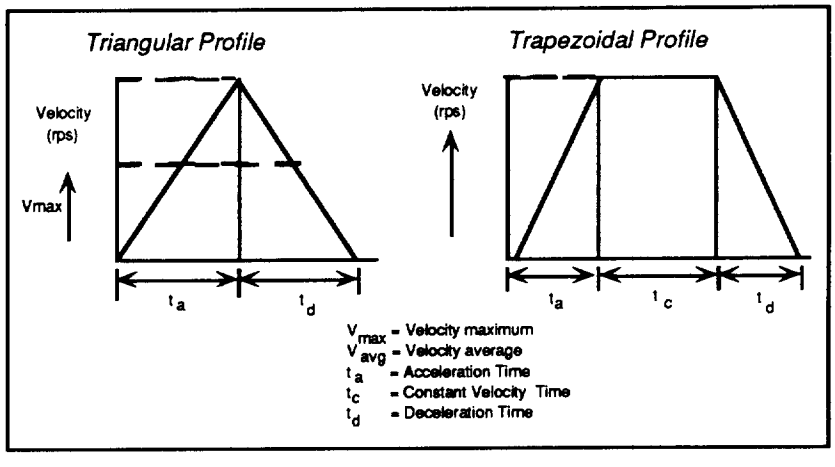

Figure 4-1. Triangular and Trapezoidal Profiles

A trapezoidal move profile results when the defined velocity is attained before the motor has moved half of the specified distance. A trapezoidal move may occur if you specify a low velocity with a high acceleration or a long distance. The resulting motion profile will resemble the profile shown in Figure 4-1.

Incremental vs. Absolute Positioning

A preset move is a move in which you specify the distance (in motor steps). You can select preset moves by putting the indexer into Normal mode (MN command). Preset moves allow you to position the motor in relation to the motor's previous stopped position (incremental moves) or in relation to a defined zero reference position (absolute moves). You can select incremental moves with the Mode Position Incremental (MPI) command. You can select absolute moves with the Mode Position Absolute (MPA) command.

Incremental Preset Mode Moves

When you are in the Incremental mode (MPI), a preset move moves the motor the specified distance from its starting position. You specify the direction with the optional (±) sign (D+2ØØØØ or D-1ØØØØ), or you can define it separately with the Set Direction (H+ or H-) command.

Command	Description
> MPI	Sets unit to Incremental Position Mode
> A2	Sets acceleration to 2 rps^2
> V5	Sets velocity to 5 rps
> D25ØØØ	Sets distance to 25,000 steps
> G	Executes the move (Go)
> G	Repeats the move (Go)
> H	Reverses direction of next move
> G	Executes the move (Go)

Absolute Preset Mode Moves

A preset move in the Absolute mode (MPA) moves the motor the distance that you specify from the absolute zero position. You can set the absolute position to zero with the Position Zero (PZ) command, successfully completing a go home move, or by cycling the power to the drive.

The direction of an absolute preset move depends upon the motor position at the beginning of the move and the position you command it to move to. For example, if the motor is at absolute position +12,800, and you instruct the motor to move to position +5,000, the motor will move in the negative direction a distance of 7,800 steps to reach the absolute position of +5,000.

When you issue the Mode Position Absolute (MPA) command, it sets the mode to absolute. When you issue the Mode Position Incremental (MPI) command, the unit switches to Incremental mode. The indexer retains the absolute position, even while the unit is in Incremental mode. You can use the Position Report (PRA) or Status Request (R) commands to read the absolute position.

Command	Description
> MPA	Sets unit to Absolute Position mode
> A2	Sets acceleration to 2 rps^2
> V1Ø	Sets velocity to 10 rps
> PZ	Sets the current position to zero
> D1ØØØØ	Sets position to 10,000 steps
> G	Executes the move (Go)
> D2ØØØØ	Sets position to 20,000 steps
> H	Reverses the position of next move to -20,000 steps
> G	Moves the motor to absolute position -20,000 (Go)
> DØ	Sets the move position to 0
> G	Executes the move (Go)
> MPI	Sets indexer to Incremental Position mode

Continuous Mode Moves

In the Continuous mode (MC), the motor will accelerate to its constant velocity when you issue a G (Go) command, the distance command is ignored. The motor will run at constant velocity until you issue a Stop or Kill command (a command that interrupts motion).

Command	Description
> FSBØ	Sets unit to motor step mode
> MC	Sets unit to Continuous mode
> A1Ø	Sets acceleration to 10 rps^2
> V1Ø	Sets velocity to 10 rps
> G	Executes the move (Go)

In the example above, the motor will ramp up to 10^2 rps and continue to run. You can command a new velocity while the motor is running with the following commands.

Command	Description
> V5	Sets velocity to 5 rps
> G	Decelerates to 5 rps

The motor will decelerate from 10 rps to 5 rps using the previously specified acceleration rate.

Modes Of Operation

This section discusses the three modes of operation that are applicable to the Model 301:

- Immediate RS-232C
- Interactive Editing
- PLC Operation

Immediate RS-232C Mode

The Model 301's RS-232C interface port allows you to send motion commands for immediate execution. You can also use this port to interactively edit motion programs and sequences that are stored in the Model 301's internal, nonvolatile memory. You can enter and edit sequences from any RS-232C terminal or computer.

Being able to execute commands as soon as they are received is especially useful during set-up and debugging when you are installing the system or if an application requires data from a remote computer or programmable controller. All commands are composed of simple ASCII characters.

In Immediate mode, the indexer responds with a prompt (>) when it receives a valid command and a question mark (?) when it receives an invalid command. If you enter a valid command, but enter an invalid range (e.g., A20), the Model 301 will respond with a question mark (?) . The interactive responses are preceded with a carriage return and a line feed.

In Interactive Edit mode, the Model 301 does not check syntax, command validity, or ranges. You must ***execute*** a defined sequence to determine if it is interpreted properly. Use the Trace (XTR) command to see where question mark (?) appears to find invalid commands.

Sending Characters

When the Model 301 is connected to a terminal, and you issue a carriage return <cr>, a prompt will be provided (>). The Model 301 is now ready to receive commands. The following commands demonstrate what you would type to perform an incremental move.

Command	Description
> MPI	Sets unit to incremental mode
> A1Ø	Sets acceleration to 10 rps^2
> V1	Sets velocity to 1 rps
> D25ØØØ	Sets distance to 25,000 steps
> G	Executes the move (Go)

All commands listed in *Chapter 5, Software Reference* that are categorized as immediate can be executed in this fashion.

Requesting Status

There are several commands that you can use to request status information from the Model 301's RS-232C port. You can also obtain this information from a terminal or computer and use the data to debug the system. One example of such a command is the Status Report (**R**) command. A sample response from the **R** command is shown below.

```
*DYNAMIC DATA
*   INPUT BYTE  (B7 - BØ) = ØØØØØ11Ø
*   OUTPUT BYTE (B7 - BØ) = ØØØØØØØØ
*   HOME: = 1
*REGISTERED DATA
*   Inputs:  I1=Ø I2=Ø I3=Ø I4=Ø I5=Ø
*   Outputs: O1=Ø O2=Ø O3=Ø O4=Ø O5=Ø O6=Ø
*   Motor Position =+Ø
*   Encoder Position =+Ø
*ACTIVE PARAMETERS:
*   MR25ØØØ GHV 1 GHF Ø.1
*   FRØ11ØØØØØ
*   ER 4ØØØ DB5 DW25Ø CG8
*   VSØ VØ AØ
*   MPI D+Ø TØ.5 LØ
```

Refer to Chapter 5, Software Reference for more information on status commands (i.e., **PRA**, **PRX**, **PX**, and **FR**).

Interactive Edit Mode

You can also use the Model 301's RS-232C interface to enter and edit sequences. A sequence consists of several Model 301 commands. You should be sure to enter the commands in the order that you intend them to be executed. When the sequence is run, the system executes the commands in exactly the order that they appear in the sequence.

You can store up to 63 sequences in the Model 301's battery-backed RAM memory. There is no limit to the size of each sequence as long as the combined total of all sequences does not exceed the available memory. All stored sequences do not have to be the same size (e.g., two 50-byte sequences and four 250-byte sequences). For applications that require additional memory storage capacity, the Model 301-M offers 8K of battery-backed RAM memory.

To begin entering a sequence, you must issue the Edit Sequence (**EXR**) command. At the prompt, enter **EXR** followed by the sequence # that you want to create. Refer to the following example. The commands that you enter are shown in **boldface** and **underlined**. The interactive responses from the system are shown in plain type.

```
> EXR1Ø
  *(.1)  1Ø:...

  Inserting Sequence 1Ø

  *.10 >A1Ø
  *    >V1Ø
  *    >D25ØØØ
  *    >G
  *    >XT
  >
```

At this point, you can begin to enter the commands for sequence #10. Notice that the Model 301 prompts you with an asterisk (*) and a bracket (>) in the Interactive Edit mode. To exit the Edit mode press the carriage return key on an enpty line [cr] or press the [esc] key.

Within the Interactive Edit mode, there are two editing sub-modes:

- Fill mode
- Edit mode

Fill Mode

This mode is used when no sequence exists—you are creating the sequence. You can ***fill*** line after line, just as in the example above

Edit Mode

You will automatically enter this mode whenever you edit an existing sequence. The sequence and its line numbers will be displayed. You must use the line-editor commands that allow you to insert, edit, or delete a line.

The following example demonstrates how to edit an existing sequence. When you issue the **EXR** command, the Model 301 lists the sequence along with the line numbers. You may now edit (**E**), insert (**I**), or delete(**D**).

```
> EXR1Ø
  * (.1)  10: A1Ø
  * (.2)      V1Ø
  * (.3)      D25ØØØ
  * (.4)      G
  * (.5)      XT

Editing Sequence 10
  * >
```

Edit a Line

To edit a line, enter **E.** followed by the line number that you want to modify.

```
* >E.3
  *(.3)  D25ØØØØ

  * >>  >D5ØØØØØ
```

The Model 301 lists the line to be edited directly above the asterisk prompt. This allows you to see what is currently stored in the line as you prepare to edit it. To edit a line, you must re-enter the entire line (including the change you want to make). When the entire line is re-written press the return key. The Model 301 will automatically re-list the entire sequence so that you can review the changes.

```
  * (.1)  10: A1Ø
  * (.2)      V1Ø
  * (.3)      D5ØØØØ
  * (.4)      G
  * (.5)      XT

  Editing Sequence 1Ø
  * >
```

Exiting Edit Mode

If you press the Enter key while the cursor is on a blank line, the currently stored line will remain unchanged. You can press the Escape <ESC> key at any time to abort the editing session without changing the current line.

You can also exit the Edit mode by typing Q (Quit Editing Mode) on a blank line and pressing the Return key or by pressing the return on a blank line.

Listing Sequences

You can list sequences with the **LST** command. This command lets you list the entire contents of memory, specified sequences, or ranges of sequences.

`LST` This command lists the entire nonvolatile sequence memory.

`LSTnn` This command allows you to list the designated sequence.

`LSTnn-nnn` This command allows you to list all sequences within a specified block (e.g., 15 - 30).

`LSTnn-` This command allows you to list all sequences from a specified sequence to the end of the program.

`LST-nn` This command allows you to list all sequences from the beginning of the program to a specified sequence of the program.

When you list multiple sequences or enter **LST63** the number of bytes of program memory that are available will be displayed after the contents of the sequences are displayed.

Inserting and Deleting Lines

When you are in the Edit mode, you can insert and delete lines in a sequence. To insert a line, type the I command followed by the line number that follows the point where you want to insert a new line. For example, if you wanted to insert a line between lines #3 and #4, you would specify line #4 as the point of insertion. Refer to the following example.

```
> EXR1Ø
  * (.1)  10: A1Ø
  * (.2)      V1Ø
  * (.3)      D25ØØØ
  * (.4)      G
  * (.5)      XT

  Editing Sequence 1Ø

  * >I.4
  * (.4) Inserting
  * >> >H+

  * (.1)  10: A1Ø
  * (.2)      V1Ø
  * (.3)      D5ØØØØ
  * (.4)      H+
  * (.5)      G
  * (.6)      XT

  Editing Sequence 1Ø
  * >D.4
  * (.4)  H+ Deleted

  * (.1)  10: A1Ø
  * (.2)      V1Ø
  * (.3)      D5ØØØØ
  * (.4)      G
  * (.5)      XT

  > Editing Sequence 1Ø
  * >
```

Deleting A Sequence

If you want to delete an entire sequence from memory, use the following steps.

1. Enter the Edit Sequence (**EXRnn**) command.
2. At the edit prompt (*** >**), type **D** and press the carriage return key.
3. The system will ask you to verify your request before the sequence is deleted.

```
* >D
Are You Sure (Y/N)? Y
Sequence nn Deleted
```

nn refers to the sequence # to be deleted

Clearing Memory

Occasionally, you may want to clear the entire contents of the battery-backed RAM memory. To do this, use the Clear (**CLR**) command. The system will ask you to verify your request to clear the memory before performing the task. Refer to the example below.

```
> CLR
  Are You Sure (Y/N)? N (Enter Y to clear)
>
```

PLC Operation

The Model 301 also communicates with the PLC processor (GE Fanuc One™ and TI Series 305™) over the backplane communication bus. The PLC program can instruct the Model 301 to execute a pre-programmed sequence, monitor the status of the indexer, and synchronize the motion program with the rest of the machine it is controlling. (Table 2-2 contains a list of PLC backplane commands.)

Program Design

This section discusses the basic elements and issues of program design for the Model 301. The issues addressed are:

- Sequences
- Trigger Inputs
- Programmable Outputs
- Time Delays
- Branching
- Limits

Sequences

Sequences are the building blocks of motion programs in the Model 301. You can store up to 63 individual sequences in the indexer's nonvolatile battery-backed RAM. A sequences can be as small as a single command or as large as the available memory.

Sequences can also be thought of as subroutines within a larger program. A sequence is a list of commands that are executed one at a time when you run the sequence.

The Model 301 has commands that allow you to branch to sequences based on conditions within a sequence. The indexer also provides the ability to **GOTO** a different sequence, or **GOSUB** to another sequence, returning to the original point after execution is complete. Refer to *Chapter 5, Software Reference* for detailed descriptions of the following commands.

Sequence Programming/Editing Commands	
EXR	Edit a sequence
I	Insert a line
E	Edit a line
D	Delete a line
Q	Quit Edit mode
LST	List sequence(s)
CLR	Clear memory

Sequence Execution Commands	
XR	Runs a sequence. When used within a sequence, it jumps to execute another sequence, then returns to the original point, like a GOSUB command.
XG	Exits the current sequence and executes the specified sequence, like a GOTO command.

Sequence Debugging Commands	
XTR	Enables/Disables the Sequence Trace mode
TEST	Simulates the PLC inputs
R	Provides a status report of the indexer

Special Sequence Commands	
XT	Ends the definition of a sequence

A sequence is a series of commands. These commands are executed in the order in which they are programmed (entered). Refer to the *Interactive Edit Mode* section earlier in this chapter for an explanation of how to enter and edit sequences. Two examples sequences are shown below.

Command	Description
> LST1-2	Lists sequences 1 & 2
1: A1Ø	Sets acceleration to 10 rps^2
V1Ø	Sets velocity to 10 rps
D25ØØØ	Sets the distance to 25,000 distance
G	Executes the move (Go)
2: D5ØØØØ	Sets the distance to 50,000 distance
H-	Sets axis to the CCW direction
G	Executes the move (Go)
XT	Ends the sequence

The commands that you enter to define a sequence are presented vertically in the previous example above. This was done to provide descriptions of each command. You can actually enter as many commands as you wish on each line, separated by a space:

```
> LST1-2
   1: A1Ø V1Ø D25ØØØ G
   2: D5ØØØØ H- G XT
```

In the two example sequences, only sequence #2 has an XT command at the end. In this example, if you execute sequence #1, the Model 301 will execute sequence #2 after sequence #1 is completed. If you execute only sequence #2, the indexer will stop after the sequence is completed.

Trigger Inputs

The Model 301 has 5 trigger inputs that are controlled by the host PLC. To recognize trigger inputs, the eight inputs from the PLC bus must be set by the PLC program as follows.

B7	B6	B5	B4	B3	B2	B1	BØ
X	1	1	I5	I4	I3	I2	I1

X = Don't Care

Bits #5 & #6 must be ON for the Model 301 to recognize bits Ø - 4 as trigger inputs 1 - 5 respectively.

You can use the Trigger (TR) command to instruct your program to wait for the correct combination of inputs to be turned on before it proceeds with the next command. This is one way in which you can synchronize the Model 301's motion program with the PLC program. Refer to the following example.

Command	Description
> A1Ø	Sets acceleration to 10 rps^2
> V1Ø	Sets velocity to 10 rps
> D25ØØØ	Sets the distance to 25,000 steps
> TRØØØØ1	The move cannot be made until inputs #1 - #4 are off, and #5 is on
> G	Executes the move (Go)

Programmable Outputs

The Model 301 has eight programmable outputs to the PLC bus. Two are dedicated — Busy and Program Busy. With the Output (O) command, you can use the Model 301 to turn the other six general programmable outputs on and off within a sequence. The O command signals the PLC when some part of your motion program is or has been executed.

Command	Description
> MN	Sets indexer to normal preset mode
> A1Ø	Sets acceleration to 10 rps^2
> V1Ø	Sets velocity to 10 rps
> D25ØØØ	Sets distance to 25,000 steps
> TRØØØØ1	The move cannot be made until inputs #1 - #4 are off, and #5 is on
> O1ØØØØ	Turns output #1 on
> G	Executes the move (Go)

Time Delays

You can use the Time (T) command to delay execution of a sequence for a preset period of time.

Command	Description
> MN	Sets indexer to normal preset mode
> A1Ø V1Ø D25ØØØ	Sets the move parameters
> G	Executes the move (Go)
> T1Ø	Sets a time delay of 10 seconds
> G	Executes the move (Go)
> XT	Ends the sequence

Branching

You can use the Conditional If (IF) command for conditional branching within a program. This command tests the input conditions. If the condition is true, all commands that follow the IF statement are executed. If the conditions are not true, the Model 301 will skip all of the commands associated with the condition, until it reaches an End of If Statement (NIF) command.

You can use the Conditional If statement in conjunction with the XG (GOTO sequence) and the XR (GOSUB) sequence commands for flexible program development.

Limits

The Model 301 does not have dedicated limit inputs. These can be implemented by using a separate input card and moving the input status to the Model 301 input data registers. The Model 301 can act upon the inputs using the IF(IXXXXX) . . . NIF procedure or activating the Stop-on-input3 (FSF1) feature.

Chapter 5. Software Reference

Chapter Objectives

The information in this chapter will enable you to:

- Identify the five types of commands in Compumotor's X-Series Language
- Use this chapter as a reference for the function, range, default, and sample use of each command

Description of Format

①PRA ②Status	③Position Request			④VALID Software Version A
⑤SYNTAX PRA	⑥UNITS X = steps	⑦RANGE None	⑧DEFAULT None	⑨ATTRIBUTES Sequence/Immediate
⑩EXECUTION TIME <10 ms		⑪SEE ALSO PZ, SP, R, PRX		
⑫RESPONSE TO PRA IS See Below				

1. ***Mnemonic Code*** This box contains the command's mnemonic code and the command type. The command types are described below.

2. ***Type*** This portion of the box contains the command's type. The five command types are listed below.

   Edit You can use edit commands to create or modify sequences.

   Motion Motion commands affect motor motion (i.e., accelerate, velocity, distance, go home, stop, direction, mode, etc.).

   Programming Programming commands affect programming and program flow. For example, output, all sequence commands, time delays, loop and end loop, and triggers.

   Set-Up Set-up commands define set-up conditions for the application. Set-up commands include the following types of commands:

   - **FSA**, **FSB**, etc.

   Status Status commands respond (report back) with information.

3. ***Full Name*** This field contains the full command name.

4. ***Valid Revision Level*** — This field contains the revision history of the command. It includes the revision of software when the command was added or modified. If the revision level of the software you are using is equal to or greater than the revision level listed in this field, your software will contain the commands as described.

5. ***Syntax*** — The proper syntax for the command is shown here. The specific parameters associated with the command are also shown. Definitions of the parameters are described below.

   - **n** This represents an integer. You may use an integer to specify a variety of values (acceleration, velocity, etc.).
   - **s** This represents a sign character (+ or -). This variable allows you to specify direction (CCW or CW) or a positive or negative value.

6. ***Units*** — This field describes what unit of measurement the parameter is using.

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

8. ***Default*** — 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. ***Attributes*** — This box indicates if the command is ***immediate*** , ***sequence*** or ***sequence/immediate.*** The system executes immediate commands as soon as it receives them. You will enter **immediate** commands via an RS-232C terminal (you must enter a carriage return after these commands to execute them). With the Model 301, sequence commands are only executable in a sequence, and in the order that they are received.

   Commands that are classified as **sequence** can only be executed in a sequence. Commands that are classified as **sequence/immediate** may be executed in the Immediate mode as well as within sequences.

10. ***Execution Time*** — The execution time is the span of time that passes from the moment you issue a command to the moment the system begins to execute it.

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

12. ***Response*** — A sample status command is given (next to **RESPONSE TO**) and the system response is shown. ***This box will only be provided if the system provides a response to the command. If no response is provided, this box will not be included with the description.***

Special Commands

You can use the following special commands with the Model 301.

ESC Edit/Motion	ESC Key			VALID Software Version A
SYNTAX ESC	**UNITS** None	**RANGE** None	**DEFAULT** None	**ATTRIBUTES** Immediate
EXECUTION TIME <10 ms		**SEE ALSO** S, K		

Description

The ESC (escape) key terminates motion. During a move, this key will terminate motion. The user program, in addition to the motion, is terminated with the ESC key.

If you press the ESC key while you are editing a program, you will exit from the Edit mode without changing the current line. If you press the ESC key two times, the terminal screen will clear (VT100 must be emulated).

@ Edit	Comments Delimiter			VALID Software Version A
SYNTAX @	**UNITS** None	**RANGE** None	**DEFAULT** None	**ATTRIBUTES** Sequence
EXECUTION TIME <10 ms		**SEE ALSO** D, E, EXR, I, Q		

Description

When you edit a program in the Edit mode, you may place comments on any line after you enter the @ delimiter. Comments can be useful when you need to briefly describe the procedure that a statement line will perform.

Example

```
L
IF(IØØXXX)  XR5  NIF  @  Run  Set-Up
IF(IØ1XXX)  XR10  NIF  @  Index  1
IF(I1ØXXX)  XR15  NIF  @  Index  2
IF(I11XXX)  XR20  NIF  @  Go-Home
N
```

General Command Listing

A Motion	Set Acceleration			VALID Software Version A
SYNTAX Annn.nn	**UNITS** n = rps^2	**RANGE** 0.03 to 999.99	**DEFAULT** 0 (motor dependent)	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO** D, V, G, R		

Description The Acceleration command specifies the acceleration rate to be used upon executing the next Go (G) command. The acceleration remains set until you change it. You do not need to reissue this command for subsequent G commands. If you enter an acceleration rate that is outside the valid range, the indexer will maintain the previous valid setting.

Example

Command	Description
> MN	Sets the moves to mode normal (preset moves)
> A5	Sets acceleration to 5 rps^2
> V1Ø	Sets velocity to 10 rps
> D1ØØØØ	Sets distance to 10,000 steps
> G	Executes the move (Go)

CG Set-Up	Correction Gain			VALID Software Version A
SYNTAX <a>CGn	**UNITS** n = amount of error to correct	**RANGE** 1-8	**DEFAULT** saved in non-volitle RAM	**ATTRIBUTES** Sequence/ Immediate
EXECUTION TIME <10ms		**SEE ALSO** FSB, FSC, DB, R		

Description This command allows you to set the amount of error (steps) that should be corrected on the initial position maintenance (FSC1) command correction move (which takes place whenever the motor is stationary and outside the dead-band [DB] region).

The percentage of error that the Position Maintenance function will attempt to correct on its correction moves is n/8 • 100%. If you set n to 1, the system will correct the error slowly (1/8 of the error is corrected on the first try). This type of correction is performed smoothly. If you set n to 8, the system will correct the error faster. However, there may be more overshoot and ringing at the end of this type of correction move.

Example

Command	Description
CG3	The system corrects 3/8 of the final-position error on the initial correction move

CLR Edit	Clear Entire Program			VALID Software Version A
SYNTAX CLR	UNITS None	RANGE None	DEFAULT None	ATTRIBUTES Immediate
EXECUTION TIME <10 ms		SEE ALSO D, E, EXR, I, LST		

Description This command clears (erases) the entire user program. The Model 301 will prompt you to enter **Y** or **N** to verify your intentions before it erases the memory. *You can use the Delete (D) command to delete individual sequences.*

D Motion	Set Direction & Distance			VALID Software Version A
SYNTAX D±nnnnnnnn	UNITS n = steps	RANGE ±99,999,999	DEFAULT 0	ATTRIBUTES Sequence/Immediate
EXECUTION TIME <10 ms		SEE ALSO A, G, V, MN, MPA, MPI, SP, FSB, R		

Description The Distance (D) command defines either the number of steps the motor will move or the absolute position it will seek after you enter a Go (G) command. In Incremental mode (MPI), the value set with the D command will be the distance (in steps) that the motor will travel on all subsequent G commands.

In Absolute mode (MPA), the distance that the motor moves is the difference between the current position and the zero position. The D command does not affect continuous mode moves (MC). The D command specifies either motor steps or encoder steps, depending on the mode (FSB).

Example #1

Command	Description
MN	Sets unit to Normal mode (preset)
A5	Sets acceleration to 5 rps^2
V10	Sets velocity to 10 rps
D5ØØØØ	Sets distance to 50,000 steps
G	Executes the move (Go)
G	Executes the move, moves another 50,000 steps

Example #2

Command	Description
MPA	Sets unit to Absolute Position mode
PZ	Sets current axis position as zero
A5	Sets acceleration to 5 rps^2
V5	Sets velocity to 5 rps
D5ØØØØ	Sets distance to 50,000 steps
G	Executes the move (Go)
G	**No motion**, because the axis has already traveled the 50,000 steps commanded with the 1st Go command

D Edit	Delete			VALID Software Version A
SYNTAX D.nnn	**UNITS** n = line number	**RANGE** 1 - 999	**DEFAULT** None	**ATTRIBUTES** Immediate
EXECUTION TIME <10 ms		**SEE ALSO** E, EXR, I, Q, @		

Description

In the Edit mode, the D command lets you delete an entire sequence. You can also use a variation of this command (D.nn) to delete one line in a sequence. The stored sequence is:

```
10:  V5  A5  D50000
     O111111
     G
     T2  O111110
     XT
```

You can enter the Edit mode for sequence #10 with the following command:

```
>  EXR10<cr>
```

Sequence #10 will be listed.

```
*(.1)  10:  V5  A5  D50000
*(.2)       O111111
*(.3)       G
*(.4)       T2  O111110
*(.5)       XT
Editing  Sequence  10
```

To delete just one *line* in the sequence, you can use the D.nnn command in the manner shown below. This command will remove the third line from the sequence.

```
*  D.3<cr>
```

The Model 301 will respond with the following prompt:

```
*(.3)       G..  Deleted

*(.1)  10:  V5  A5  D50000
*(.2)       O111111
*(.3)       T2  O111110
*(.4)       XT
Editing  Sequence  10
```

To delete the *entire* sequence, you can use the D<cr> command. The following prompt will appear:

```
Are you sure (Y/N)  Enter  Y
Sequence  10..  DELETED
```

DB Set-Up	Deadband			VALID Software Version A
SYNTAX	**UNITS**	**RANGE**	**DEFAULT**	**ATTRIBUTES**
<a>DBn	n = motor steps	n 0 - 999,999	saved in non-volitle RAM	Immediate/ Sequence
EXECUTION TIME < 10ms		**SEE ALSO** FS, CG, R, DW		

Description

This command specifies a positioning range (in encoder steps) that the motor may not exceed after completing a move. If the motor's position is closer to the desired position than the number specified, no position maintenance correction will be performed. If the motor's position is not within the allowable range, position maintenance is performed (if enabled by the Enable Position Maintenance (**FSC1**) command). The purpose of the **DB** command is to prevent the motor from searching for a set position when it is within an allowable dead band range.

Example

Command	Description
DB1ØØ	Sets Position Maintenance to activate if the motor's end-of-move position is off by more than 100 encoder steps.

DLY Motion	Delay Before Changing Output or Velocity			VALID Software Version A
SYNTAX	**UNITS**	**RANGE**	**DEFAULT**	**ATTRIBUTES**
DLYnnnnnnnn	n = motor steps	0 - 99,999,999 0,1, or X	0 0	Sequence
EXECUTION TIME <10 ms		**SEE ALSO** G, T		

Description

The **DLY** command delays program execution based on position counts that you specify. A **DLY** command must follow a **G** command in a sequence. The number of steps that you specify as the **DLY** variable represents the count of relative steps (from the initiation of the previous Go command) that program execution will be delayed. This command is similar to the Time Delay (**T**) command, except that the **T** command delays execution according to a specified time. *This command only works in Continuous mode.*

Changing Velocity After Position Delay

Command	Description
5 : **A5Ø V1 MC G**	Begins continuous move at 1 rps
DLY1ØØØØØ V3 G	Ramps motor to 3 rps after 100,000 steps
DLY1ØØØØØ V5 G	Ramps motor to 5 rps after 100,000 more steps
DLY1ØØØØØ S	Stops the motor after 100,000 more steps
XT	End sequence

Changing Outputs After Position Delay

Command	Description
6 : **A5Ø V1 MC G**	Begins continuous move at 1 rps
DLY1ØØØØØ O111ØØØ G	Ramps motor to 3 rps after 100,000 steps
DLY1ØØØØØ OØØØ111 G	Ramps motor to 5 rps after 100,000 more steps
DLY1ØØØØØ S	Stops the motor after 100,000 more steps
XT	End sequence

DW Set-Up	Deadband Window			VALID Software Version A
SYNTAX <a>DWn	**UNITS** n = motor steps	**RANGE** 0 - 999,999	**DEFAULT** saved in non-volatile RAM	**ATTRIBUTES** Buffered Savable in Sequence
EXECUTION TIME <10ms		**SEE ALSO** FS commands, R, DB		

Description

This command allows precise deadband specification in motor pulses. The backlash deadband allows systems with backlash to use stall detect features. Stall detection will not occur until the error exceeds the deadband window.

Compumotor recommends a **DW** value of 250 if you are using a Motor Resolution (**MR**) of 25,000 and an Encoder Resolution(**ER**) of 4,000. If using low acceleration values this value may be reduced. There will always be some lag between the motor steps sent out and encoder steps received. This value should never be zero.

If you set **DW** to 50 a position maintenance move of less than 50 steps the Model 301will not be able to detect a stall

Example

Command	Description
DW1ØØ	100 motor steps of Backlash are expected by the indexer. A stall will not be detected until the encoder lags the motor position by more then 100 motor steps.

E Edit	Edit a Line in a Sequence			VALID Software Version A
SYNTAX E.nnn	**UNITS** nnn = line number	**RANGE** 1 - 999	**DEFAULT** None	**ATTRIBUTES** Immediate
EXECUTION TIME <10 ms		**SEE ALSO** D, EXR, I, Q		

Description

This command allows you to edit a single line within a sequence. The value that you specify with the **E.nn** command is the **nth** line counted from the beginning of that block. You can only use this command in the Edit mode. After you edit the line, enter a carriage return <cr> to mark the end of the line. The entire block is rewritten to include the change. You will be prompted to enter your next Edit mode command by an asterisk (*).

To exit the Edit mode, you must enter **Q** (Quit Edit Mode) and press the carriage return <cr>. This keystroke combination terminates the editing session. You may also press <cr> on an empty line or the ESC key while you are in the Edit mode to terminate an editing session. Pressing the return key on an empty line also exits the Edit mode.

ER Set-Up	Encoder Resolution			VALID Software Version A
SYNTAX <a>**ER**n	**UNITS** n = post-quadrature encoder steps	**RANGE** 1 - 5000	**DEFAULT** saved in non-volatile RAM	**ATTRIBUTES** Immediate/ Sequence
EXECUTION TIME <10ms		**SEE ALSO** FS, DW, R		

Description

The encoder resolution defines the number of encoder steps the indexer will see per revolution (or per inch for linear motors) of the motor. The number of lines on an encoder should be multiplied by 4 to arrive at the correct **ER** value per revolution (per inch) of the motor .

In other words one line of an encoder will product 4 encoder steps.

NOTE: A 4:1 ratio of motor steps to encoder steps or a 16:1 ratio of motor step: encoder lines is proper closed loop operation. If a lower ratio is used it may be difficult to tune the position maintenance (Servoing) feature of your indexer.

Example

Command	Description
ER4000	(Encoder resolution is set to 4000 steps per motor revolution)

EXR Edit	Edit Sequence in a Program			VALID Software Version A
SYNTAX **EXR**n n	**UNITS** n = sequence #	**RANGE** 1 - 63	**DEFAULT** None	**ATTRIBUTES** Immediate
EXECUTION TIME <10 ms		**SEE ALSO** D, E, I, Q, XG, XR, XT, @		

Description

This command enables you to edit an existing sequence of commands or write a new sequence of commands into the user program. A sequence may contain many single-line commands. An asterisk (*) will appear before the command line prompt when you are in Edit mode.

If you specify a new sequence, enter the commands after the indented prompt appears. If you specify an existing sequence, the sequence is listed and you may only enter line-edit commands. Line-edit commands are **E.nn**, **I.nn**, and **D.nn**, which you may use to create, insert, or delete single lines within an existing block.

A single line may not exceed 40 characters. You must use the ENTER key or <cr> (carriage return) to terminate the line. The number of lines within a program block is limited only by the size of the nonvolatile memory.

To exit the Edit mode, enter Q (Quit Edit Mode) and press the carriage return <cr>. You may also press <cr> on an empty line or the ESC key to end an editing session.

If you do not complete the edited sequence with an End Sequence (XT) command, the sequence that follows the edited sequence in the program's memory will be executed when you run the edited sequence. Sequence numbers can serve as labels for subroutines or branches. Refer to the following example.

```
Ø5: A5 V5 D25ØØØ
    G
    T2
Ø6: A1Ø V2 D5ØØØ
    G
    XT
```

When sequence #6 is called, a move of 5,000 steps will be performed. When sequence #5 is called, a move of 25,000 steps will be executed, followed by a wait (delay) of two seconds. After the two seconds have elapsed, the 5,000-step move of sequence #6 will be executed.

FR Status/ Set-Up	**Report/Change Set-Up Status**			**VALID** Software Version A
SYNTAX <a>**FR**n	**UNITS** n = status	**RANGE** 0, 1	**DEFAULT** Saved in non-volatile RAM	**ATTRIBUTES** Immediate/ Sequence
EXECUTION TIME <10ms		**SEE ALSO** FS commands, R, MPI, MPA		
RESPONSE TO aFR **IS** a**FR** nnnnnnnn				

Description

This command allows you to request the status of functions set by **FS** commands. The response contains one ASCII digit per function set by the **FS** command, each of which is a zero or a one. The digits correspond to the functions, left to right, A through H. The digit 1 corresponds to a function that is on. A Ø corresponds to an **FS** command that is off. The command can also be used to modify the **FS** set-up commands (See Example).

A Incremental = OFF (Ø), Absolute = ON (1):Defines the move distance (D) as either incremental from current position, or as absolute (referenced to the absolute zero position).

B Motor step mode = OFF (Ø)/Encoder step mode = ON (1):Defines the distance (D) parameter in units of motor steps or encoder steps

C Position Maintenance Ø = OFF, 1 = ON:Enables position maintenance. This will cause the indexer to servo the motor to the desired position if not in the correct position at the end of a move, or, if the motor is forced out of position while at rest.

D Terminate move on Stall Detect Ø = OFF, 1 = ON:Instructs the indexer to abort any move if a stall is detected.

E Turn on Output #1 on Stall Detect Ø = OFF, 1 = ON:Instructs the indexer to set output #1 on if a stall is detected.

F Terminate move if Input 3 is active Ø = OFF, 1 = ON: Instructs the indexer to abort any move if a signal is received on the Imput #3. If output on stall is enabled (**FSE1**), the indexer will also turn on an output when the input is active.

G Turn on Output #2 when in deadband

H Enable Stall Detect Ø = disabled, 1 = enabled

Example

Command	Description
FR<cr>	Indexer response = 11000000. The indexer is in absolute encoder step mode with all other **FS** encoder functions turned OFF.
FR111Ø0000	Position Maintenance (FSC1) is enabled
FR<cr>	Indexer response is 11100000

FSA Set-Up	**Set Indexer to Incremental/Absolute Mode**			**VALID** Software Version A
SYNTAX <a>**FSA**n	**UNITS** n = mode	**RANGE** 0, 1	**DEFAULT** Saved in non-volatile RAM	**ATTRIBUTES** Immediate/ Sequence
EXECUTION TIME <10ms		**SEE ALSO** MPI, MPA, FR, F		
RESPONSE TO aFSA **IS** a**FSA** *nnn				

Description

This command sets the indexer to perform its moves in either absolute or incremental positioning mode.

FSAØ (Incremental mode)
FSA1 (Absolute mode)

In Incremental mode (**FSA1**), all moves are made with respect to the position at the beginning of the move. This mode is useful for repeating moves of the same distance.

In Absolute mode (**FSA1**), all moves are made with respect to the absolute zero position. The absolute zero position is set to zero when you power up the indexer, successfully complete a go home, or execute the Position Zero (**PZ**) command.

FSAØ is the same as the **MPI** command
FSA1 is the same as the **MPA** command

The Absolute mode is useful when you need to move to specific locations.

Example

Command	Description
FSA1	Sets Indexer to Absolute mode
PZ	Resets the absolute counter to zero
A1Ø	Sets acceleration to 10 units
V5	Sets velocity to 5 units
D25ØØØ	Moves motor to absolute position 25,000
G	Executes the move (Go)
D5ØØØØ	Move motor to absolute position 50,000
G	Executes the move (Go)

The motor moves 25,000 steps, then moves an additional 25,000 steps to reach the absolute position of 50,000.

FSB Set-Up	**Set Indexer to Motor/Encoder Step Mode**			**VALID** Software Version A
SYNTAX <a>**FSB**n	**UNITS** n = mode	**RANGE** 0, 1	**DEFAULT** saved in non-volatile RAM	**ATTRIBUTES** Immediate/ Sequence
EXECUTION TIME <10ms		**SEE ALSO** ER, D, FR		

Description

This command sets up the indexer to perform moves in either motor steps or encoder steps.

FSBØ: (Motor steps)
FSB1: (Encoder steps)

In Motor Step mode, the distance command (D) defines moves in motor steps.

In Encoder Step mode, the distance command defines moves in encoder steps.

You must set up the indexer for the correct encoder resolution The Encoder Resolution (**ER**) command is used to define the encoder resolution.

Enabling Encoder Step Moves does not guarantee that your moves will position to the exact encoder step commanded. Position maintenance (**FSC**) must be enabled to activate closed loop servoing.

Example

Command	Description
ER4ØØØ	Set up encoder. 4,000 encoder pulses (1,000 lines) are produced per unit of the motor.
FSB1	Set moves to encoder step mode
A1Ø	Set acceleration to 10 units/sec^2
V5	Set velocity to 5 units/sec
D4ØØØ	Set distance to 4,000 encoder steps
G	Executes the move (Go)

The motor will turn in the CW direction until 4,000 encoder pulses are received.

FSC Set-Up	Enable/Disable Position Maintenance			VALID Software Version
SYNTAX <a>FSCn	**UNITS** n = on/off	**RANGE** 0, 1	**DEFAULT** Saved in non-volatile RAM	**ATTRIBUTES** Immediate/ Sequence
EXECUTION TIME <10 ms		**SEE ALSO** FSB, ER, DB, FR, FSH, FSD, DW, CG		

Description

FSC1 = Enable Position Maintenance
FSCØ = Disable Position Maintenance

Enabling position maintenance will cause the indexer to servo the motor until the correct encoder position is achieved. This occurs at the end of a move (if the final position is incorrect) or any time the indexer senses a change in position while the motor is at zero velocity. You must have an encoder connected in order to enable position maintenance. The indexer will maintain position in both Encoder and Motor Step mode.

Position maintenance will be disabled (turned OFF) (**FSCØ**) automatically if a stall is detected while running position maintenance.

When position maintenance is on a correction at the end of the move may be executed. If running a sequence including the commands **L5 G H N** there may seem to be delays between the moves.

Example

Command	Description
ER4ØØØ	Set encoder resolution to 4,000.
FSB1	Set encoder step mode.
FSC1	Enable position maintenance

FSD Set-Up	Stop on Stall			VALID Software Version
SYNTAX <a>FSDn	**UNITS** n = on/off	**RANGE** 0, 1	**DEFAULT** Savable in non-volatile RAM	**ATTRIBUTES** Immediate/ Sequence
EXECUTION TIME <10 ms		**SEE ALSO** DW, ER, FR, FSH, RE		

Description

Entering **FSDØ** will cause the indexer to attempt to finish the move when a stall is detected, even if the load is jammed.

Entering **FSD1** will cause the indexer to stop the move in progress when a stall is detected. This command is only valid if stall detection (**SSH1**) has been enabled. It will have no effect otherwise. A move stopped by a stall can be resumed.

Example

Command	Description
DW1ØØ	Set backlash value to 100 steps.
ER4ØØØ	Set encoder resolution to 4,000 steps/rev.
FSH1	Enable stall detect.
FSD1	Enable stop on stall.

FSE Set-Up	Turn on Output #1 on Stall			VALID Software Version
SYNTAX <a>FSEn	**UNITS** n = output	**RANGE** 0, 1	**DEFAULT** Saved in non-volatile RAM	**ATTRIBUTES** Immediate/ Sequence
EXECUTION TIME <10 ms		**SEE ALSO** DW, ER, FSF, FR		

Description

FSEØ - Do not turn on output #1 on stall
FSE1 - Turn on output #1 on stall

Entering **FSE1** will cause the indexer to turn on output number 1 when a stall is detected. This is useful for signaling other components in your system that a stall has occurred. This command will only be valid if Stall Detect (**FSH1**) has been enabled.

Output number 1 is uneffected by a stall when **FSEØ** is entered.

This output will also turn on if **FSE1** and Terminate Motion On Input (**FSF**) is enabled.

Example

Command	Description
ER4ØØØ	Set encoder resolution to 4,000 steps/rev.
DW2ØØ	Set backlash deadband to 200 motor steps.
FSH1	Enable stall detect.
FSE1	Turn on output number 1 when a stall is detected.

FSF Set-Up	Stop Motion on Input #3			VALID Software Version
SYNTAX <a>FSFn	**UNITS** n = on/off	**RANGE** 0, 1	**DEFAULT** Saved in non-volatile RAM	**ATTRIBUTES** Immediate/ Sequence
EXECUTION TIME < 10ms		**SEE ALSO** TR, FR, S, K		

Description

FSFØ - Do not terminate move on Input #3
FSF1 - Terminate move when Input #3 is active.

Entering **FSF1** will cause any move in progress to be stopped whenever Input #3 (Bit #2) is active. Setting up another unit to turn on Output #1 when it detects a stall with the Turn on Output on Stall (**FSE**) command, enables the user to implement a multi-axes stop on stall by connecting the output of one axis to the input on the other. The move will be decelerated at the rate last set with the acceleration (**A**) command. A terminated move can be resumed (RE). If the move is in a sequence the sequence will also be stopped.

Enabling this function dedicates input #3 to a stop input. Thus it will disallow many of the 8-bit PLC interface commands.

Example

Command	Description
FSF1	Input #3 is now dedicated as a remote terminate input.

FSG Set-Up	Turn on Output 2 when within Deadband			VALID Software Version A
SYNTAX <a>FSGn	**UNITS** n = on/off	**RANGE** 0, 1	**DEFAULT** Saved in non-volatile RAM	**ATTRIBUTES** Immediate/ Sequence
EXECUTION TIME < 10ms		**SEE ALSO** DB, FR		

Description

FSGØ - Do not turn on output #2 when the motor is within deadband.
FSG1 - Turn on output #2 when the motor is within deadband.

This output is cleared at the beginning of a move and updated at the end of the move.

Example

Command	Description
DB2Ø	Define deadband as 20 steps
FSG1	Turn on output 2 while within deadband

FSH Set-Up	Enable Stall Detect			VALID Software Version
SYNTAX <a>FSHn	**UNITS** n = on/off	**RANGE** 0, 1	**DEFAULT** Saved in non-volatile RAM	**ATTRIBUTES** Immediate/ Sequence
EXECUTION TIME <10ms		**SEE ALSO** DW, ER, FR, FSD		

Description

FSHØ Disable Stall Detect
FSH1 Enable Stall Detect

This command must be used to detect a stall condition. After enabling stall detection, stop on stall (**FSD1**) and output on stall (**FSE1**) can be used.

It is necessary to define the Deadband Window (**DW**) command and the Encoder Resolution (**ER**) command before this feature will operate properly. Stall Detection is only possible when an encoder is being used.

Example

Command	Description
DW1ØØØ	Set deadband window to 1,000 steps
ER4ØØØ	Set encoder resolution to 4,000 steps (500 lines)
FSH1	Enable stall detection
FSD1	Stop motor movement if stall detected.

G Motion	Go			VALID Software Version A
SYNTAX G	**UNITS** None	**RANGE** None	**DEFAULT** None	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO** A, D, K, MC, MPA, MPI, MN, S, V		

Description

The Go (G) command instructs the axis 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 mode (MN or MC) commands with each G command.

A G command in the Absolute mode (MPA) will cause motion to the position you specify with the Distance (D) command.

In Continuous mode (MC), you only need to enter the acceleration and velocity commands prior to the G command. The system ignores the distance command in this mode. No motor motion occurs until you enter the G command.

Example

Command	Description
MN	Sets mode to Normal (preset)
A5	Sets acceleration to 5 rps^2
V5	Sets velocity to 5 rps
D25ØØØ	Sets distance to 25,000 steps
G	Executes the move (Go)
A1	Sets acceleration to 1 rps^2
G	Executes the move (Go)

GH Motion	Go Home			VALID Software Version A
SYNTAX GH±(n)	**UNITS** n = direction (velocity)	**RANGE** + or - (.01 - 99.99)	**DEFAULT** +	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO** GHF, GHV, FSB		

Description

This command instructs the indexer to search for home in the CW or CCW direction, depending on your instruction (±). The speed is the previously entered Go Home Velocity (GHV).

The home limit input on the Model 301 is optically isolated, and is normally off. You must use a normally open, load-activated switch to ground to determine the home position.

The indexer searches for home to the CW edge. The CW edge of the home switch is defined as the first switch transition that occurs when the motor reaches the home switch if it is traveling in the CCW direction.

If the indexer is in encoder mode (FSB1) it will search for the Z-channel as the home position. **A home switch is required to home to the Z channel. The home switch must be active in order for the Z-channel pulse to be recoganized.**

You can also execute the Go Home commands with the 8-bit interface (refer to *Chapter 2, Getting Started*).

The Model 301 will zero the motor and encoder position at the end of the go home move. To test the Model 301's homing function, enter the following command string.

Command	Description
> GHV5	Set go home velocity to 5 rps^2
> GHF.2	Sets final go home velocity to 0.2 rps
> GH+	Instructs the motor to go home at 5rps

The **GHV** and **GH±** commands can be combined if desired. **GH+3** instructs the motor to search for the home limit in the CW direction at 3 rps.

The following events occur when you go home in the CW direction (refer to Figure 5-1):

Open-Loop Go-Home Move

1. The motor moves in the CW direction until the home switch becomes active.
2. When the home switch is closed, the motor decelerates to the velocity you specified with the **GHF** command.
3. When the switch becomes inactive the direction is reversed. The motor creeps in the CW direction until the home switch becomes active.

The following events occur when you go home in the CCW direction (refer to Figure 5-1):

1. The motor moves in the CCW direction until the home switch becomes active.
2. The motor decelerates to a stop and moves in the CW direction at the **GHF** velocity until the home switch becomes inactive.
3. The motor creeps to the CW edge of the switch at the velocity you set with the **GHF** command. The motor stops when the switch becomes active.

The absolute position counters are set to zero at the end of a go home move.

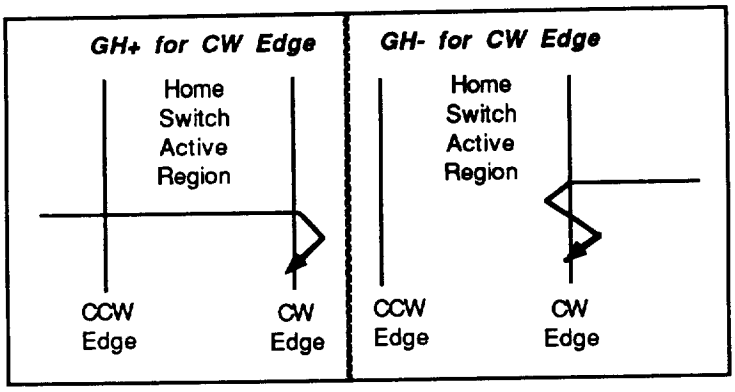

Figure 5-1. Open-Loop Homing Operation

Closed-Loop Go-Home Move

1. The motor moves in the CW direction until the home switch becomes active.
2. When the home switch is closed, the motor decelerates to the velocity you specified with the **GHF** command.
3. When the switch becomes inactive the direction is reversed. The motor creeps in the CW direction until the Z channel becomes active.

The following events occur when you go home in the CCW direction (refer to Figure 5-2):

1. The motor moves in the CCW direction until the home switch becomes active.
2. The motor decelerates to a stop and moves in the CW direction at the **GHF** velocity until the home switch becomes inactive.
3. The motor creeps at the velocity you set with the **GHF** command. The motor stops when the Z channel becomes active.

The absolute position counters are set to zero at the end of a go home move.

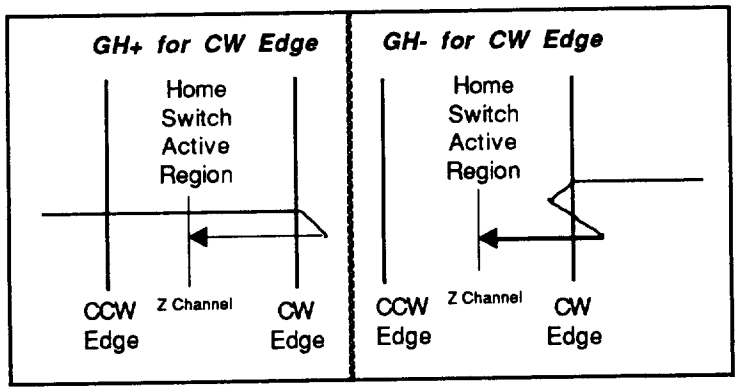

Figure 5-2. Closed-Loop Homing Operation

If the Home switch becomes inactive without finding the Z-channel, the Go-Home will stop and the Model 301 will send the following message:

```
* Marker not in home active region
```

GHF Motion	Go Home Final Speed			VALID Software Version A
SYNTAX GHF±nn.nn	**UNITS** n = rps	**RANGE** 0.01 - 99.99	**DEFAULT** Saved in non-volatile RAM	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO** GH, GHV, R		

Description This command allows you to set the speed at which the indexer instructs an axis to creep to a home switch during the final part of the HOME routine. The value (nn.nn) that you specify should be small (see GHV command) to ensure a repeatable and accurate homing routine. This is the portion of the HOME move that finds the precise edge of the limit switch that defines home in the system.

Example

Command	Description
GHF.04	Sets velocity for go home final to 0.04 rps

GHV Motion	Go Home Velocity			VALID Software Version A
SYNTAX GHV±nn.nn	**UNITS** n = rps	**RANGE** 0.01 - 99.99	**DEFAULT** Saved in non-volatile RAM	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO** GH, GHF, R		

Description This command allows you to set the speed at which an axis will seek its home position. After the switch is detected, the system will use the final go home speed (refer to the GHF command) to find the final home switch edge.

H± Motion	Toggle Direction			VALID Software Version A
SYNTAX Hx	**UNITS** x = direction	**RANGE** + or -	**DEFAULT** None	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO**		

Description The commanded direction will be used when the indexer receives this command. A plus sign (+) represents a CW motion. A minus sign (-) represents a CCW motion. If you do not specify a + or -, the direction will be toggled.

Example

Command	Description
MN	Sets unit to Normal mode
A5	Sets acceleration to 5 rps^2
V5	Sets velocity to 5 rps
D25000	Sets distance to 25,000 steps
G	Executes the move (Go)
H	Changes the direction of the move
G	Executes the move in opposite direction (Go)

I Edit	Insert a Line Within a Sequence			VALID Software Version A
SYNTAX I.nnn	**UNITS** nnn = line number	**RANGE** 1 - 999	**DEFAULT** None	**ATTRIBUTES** Immediate
EXECUTION TIME <10 ms		**SEE ALSO** D, E, EXR, Q, @		

Description

The **I** command is an Edit mode command that is valid only after you have entered an **EXR** (Edit a Sequence in a Program) command. The **I** command variable (**nn**) allows you to insert a new line (counted from the beginning of the block). After you insert the line and press the carriage return to mark the end of the line, the entire block is rewritten to include the change. You will be prompted to enter your next edit mode command by an asterisk (*).

To exit the Edit mode, you must enter **Q** (Quit Edit Mode) and press the carriage return <cr>. You may also press <cr> on an empty line or the ESC key while you are in the Edit mode to terminate an editing session.

IF Programming	Conditional IF			VALID Software Version A
SYNTAX **IF**(Innnnn)	**UNITS** n = input bit pattern	**RANGE** 0, 1, or X	**DEFAULT** None	**ATTRIBUTES** Sequence
EXECUTION TIME <10 ms		**SEE ALSO** NIF, R, TEST		

Description

The **IF** command tests the input bit pattern. If the statement is true, the commands between **IF** and **NIF** will be executed. If the statement is not true, the indexer resumes executing at the line that follows the **NIF** statement. The condition is checked against the saved input state (registered input data of **R** report). Refer to *Chapter 2, Getting Started* or the **TEST** command description in this chapter. The conditional **IF** command can be used to implement limits.

Example

Command	Description
IF(I**Ø**Ø11X)	Sets IF condition
G	Executes a move when inputs #1 and #2 are low and #3 and #4 are high
NIF	Ends IF condition

K Motion	Kill Program Execution			VALID Software Version A
SYNTAX K	**UNITS** None	**RANGE** None	**DEFAULT** None	**ATTRIBUTES** Immediate
EXECUTION TIME <10 ms		**SEE ALSO** S, ESC Key		

Description

The K command is an emergency stop command and should only be used as such. This command stops indexing immediately. There is no deceleration of the motor. If the Kill command causes the motor to slip (i.e., large loads at high speed), the load could be driven past limit switches and cause damage to the mechanism and possibly the operation.

In addition to stopping the motor, the K command will terminate a loop, end a time delay, terminate program execution, and kill a trigger. You can also execute the Kill command with the 8-bit interface (refer to *Chapter 2, Getting Started*).

Example

Command	Description
A5	Sets acceleration to 5 rps^2
V2	Sets velocity to 2 rps
MC	Sets unit to Continuous mode
G	Executes the move (Go)
•	
•	
K	Stops the motor instantly

L Programming	Loop			VALID Software Version
SYNTAX Ln	**UNITS** n = # of loops	**RANGE** 0 - 65,535	**DEFAULT** 0	**ATTRIBUTES** Sequence
EXECUTION TIME <10 ms		**SEE ALSO** N, Y, XG, R, K		

Description

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. The L command, without a value specified for n, or with a Ø, will create an infinite loop.

The End-of-Loop command prompts the controller to proceed with further commands after the designated number of loops have been executed. The Stop Loop (Y) command ends execution of the loop.

Example

Command	Description
> MN	Sets indexer to Normal mode
> L5	Loops 5 times
> A5	Sets acceleration to 5 rps^2
> V1Ø	Sets velocity to 10 rps
> D1ØØØØ	Sets distance to 10,000 steps
> G	Executes the move (Go)
> N	Specifies the above 10,000-step move to be repeated five times
> XT	End of sequence

LST Edit	List Program			VALID Software Version A
SYNTAX See Below	UNITS n = line #	RANGE 1 - 63	DEFAULT None	ATTRIBUTES Immediate
EXECUTION TIME <10 ms		SEE ALSO CLR, EXR		

Description You can use the List command in a variety of ways. This command allows you to display all of the sequences in the unit's memory. Listing multiple sequences also shows how much memory is available for further sequences.

`LST` This command lists the entire nonvolatile sequence memory.

`LSTnn` This command allows you to list the designated sequence.

`LSTnn-nnn` This command allows you to list all sequences within a specified block (e.g., 15 - 30).

`LSTnn-` This command allows you to list all sequences from a specified sequence to the end of the program.

`LST-nn` This command allows you to list all sequences from the beginning of the program to a specified sequence of the program.

Example

Command	Description
`> LST`	Lists all current sequences

```
  5: MPI MN D+10000 V11 A80
  6: IF (I0000000) S NIF
     O111111
     L10 T.5 G H N
     T1
     RG7
     XT
```

MC Motion	Mode Continuous			VALID Software Version A
SYNTAX MC	UNITS None	RANGE None	DEFAULT None	ATTRIBUTES Sequence/Immediate
EXECUTION TIME <10 ms		SEE ALSO A, G, MN, T, TR, V, DLY, R		

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

The controller uses the Acceleration (**A**) and Velocity (**V**) commands to reach continuous velocity. Using the Time Delay (**T**), Trigger (**TR**), or Delay (**DLY**) with Velocity (**V**) commands, you can achieve basic velocity profiling.

Example

Command	Description
`> MC`	Sets mode to continuous
`> A5`	Sets acceleration to 5 rps^2
`> V5`	Sets velocity to 5 rps
`> G`	Executes the move (Go)

The motor turns at 5 rps until it is halted by the Stop (**S**) command, Kill (**K**) command, a limit switch, or by a new velocity specification.

To change velocity after a certain time has passed, use the following procedure.

Example

Command	Description
> MC	Sets mode to continuous
> A1	Sets acceleration to 1 rps^2
> V5	Sets velocity to 5 rps
> G	Executes the move (Go)
> T5	Wait 5 seconds after velocity of 5rps is reached
> V6	Sets velocity to 6 rps
> G	Executes the move to 6rps
> T5	Wait 5 seconds after velocity of 5rps is reached
> V4	Sets velocity to 4 rps
> G	Executes the move to 4rps
> T1Ø	Wait 10 seconds after velocity of 5rps is reached
> VØ	Sets velocity to 0 rps
> G	Decelerates to a stop

To change velocities after a certain distance see the Delay (DLY) command.

MN Motion	**Mode Normal**			**VALID** Software Version
SYNTAX MN	**UNITS** None	**RANGE** None	**DEFAULT** None	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO** A, D, G, MC, MPA, MPI, V, R		

Description

The Mode Normal (MN) command sets the Positioning mode to preset. In Mode Normal, the motor will move the distance specified with the distance (D) command. To define the complete move profile, you must define Acceleration (A), Velocity (V), and the Distance (D). The MN command changes the mode of operation from Mode Continuous (MC) to Preset mode.

Example

Command	Description
> MN	Set positioning mode to preset
> A5	Set acceleration to 5 rps^2
> V5	Set velocity to 5 rps
> D1ØØØ	Set distance to 1,000 steps
> G	Executes the move (Go)

The motor turns 1,000 steps in the CW direction after the **G** command is issued. The motor comes to a stop after the move. **Normal mode is the default operating mode.** It is in effect upon power up.

MPA Motion	**Absolute Position Mode**			**VALID** Software Version A
SYNTAX MPA	**UNITS** None	**RANGE** None	**DEFAULT** Saved in non-volatile RAM	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO** MC, MN, MPI, FSB, FSA, FR, R		

Description

The MPA command sets the indexer to Absolute Position mode. In this mode, the Distance command (D) serves as an absolute position command. To return to Incremental mode, use the MPI command. MPA is the same as the FSA1 command.

MPI Motion	Incremental Position Mode			VALID Software Version A
SYNTAX MPI	UNITS None	RANGE None	DEFAULT Saved in non-volatile RAM	ATTRIBUTES Sequence/Immediate
EXECUTION TIME <10 ms		SEE ALSO MC, MN, MPA, FSA, FR, R		

Description The **MPI** command sets the indexer to the Incremental Position mode. In this mode, the Distance command (**D**) serves as an incremental position command. **MPI** is the same as the **FSAØ** command.

Example

Command	Description
> MN	Sets indexer to Normal mode
> MPI	Sets positioning mode Incremental
> A5	Sets acceleration to 5 rps^2
> V5	Sets velocity to 5 rps
> D5ØØØØ	Sets distance to 50,000 steps
> G	Executes the move (Go)

MR Set-Up	Motor Resolution			VALID Software Version
SYNTAX MRn	UNITS n = steps/rev.	RANGE 100 - 65,535	DEFAULT Saved in non-volatile RAM	ATTRIBUTES Sequence/Immediate
EXECUTION TIME <10 ms		SEE ALSO A, V, ER, R		

Description The Motor Resolution (**MR**) command sets the number of steps per revolution. This command allows the indexer to control drives of different resolutions while maintaining the commanded acceleration and velocity. *This variable is not reset with the Software Reset (Z) command.* The **MR** command will be ignored is the **A** or **V** parameters are invalid for the new resolution.

Example

Command	Description
> MN	Sets positioning mode to preset
> MR4ØØ	Sets motor resolution to 400 steps/rev
> A5	Sets acceleration to 5 rps^2
> V1Ø	Sets velocity to 10 rps
> D8ØØ	Sets distance of move to 800 steps
> G	Executes the move (Go)

A 400-step-per-revolution motor/drive will turn 800 steps (two revs) CW at an acceleration of 10 rps^2 and a velocity of 10 rps after the **G** command.

If this same command set is sent to a motor/drive with a resolution of 4,000, the motor will still turn 800 steps (1/5 of a revolution). However, the actual acceleration would only be 0.5 rps^2 and the actual velocity would only be 1 rps. The controller resolution and motor/drive resolution must match to get the commanded velocity and acceleration. *This command does not affect distance. This command does not set the drive resolution.*

N Programming	End of Loop			VALID Software Version A
SYNTAX N	UNITS None	RANGE None	DEFAULT None	ATTRIBUTES Sequence
EXECUTION TIME <10 ms		SEE ALSO L, Y		

Description The **N** command marks the end of a loop. You can use this command in conjunction with the Loop (**L**)command. All *sequence* commands that you enter between the **L** and the **N** commands are executed as many times as you specify with the **L** command's variable (**nn**).

Example

Command	Description
> **MN**	Sets the unit to Normal mode
> **A5**	Sets acceleration to 5 rps^2
> **V5**	Sets velocity to 5 rps
> **D1ØØØØ**	Sets distance to 10,000 steps
> **L5**	Loops or repeats the move 5 times
> **G**	Executes the move (Go)
> **N**	Ends the loop
> **XT**	Ends sequence definition

NIF Programming	End of IF Commands			VALID Software Version A
SYNTAX NIF	UNITS None	RANGE None	DEFAULT None	ATTRIBUTES Sequence
EXECUTION TIME <10 ms		SEE ALSO IF		

Description The **NIF** command marks the end of an **IF** statement.

Example

Command	Description
> **IF(IØØØØ1)**	Perform the following sequence when input values match the IF statement
MN	Sets to mode normal
A5	Sets acceleration to 5 rps^2
V1	Sets velocity to 1rps
D1ØØØØ	Sets distance to 10,000 steps
L1Ø	Loops 10 times
T.5	Pause for 5 seconds
G	Execute the move (Go)
H	Change the direction of the move
N	End the loop
NIF	End the IF condition

O Programming	Set Programmable Outputs			VALID Software Version A
SYNTAX Onnnnnn	**UNITS** n = output on or off	**RANGE** 0 = off 1 = on X = don't care	**DEFAULT** None	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO** DLY, I, O, FSG, FSE		

Description

The Output (O) command turns the programmable output bits on and off. The output can indicate that the motor is in position, about to begin its move, or is at constant velocity, etc. The PLC can read the 6 outputs as the least significant bits of the upper address. If the Model 301 is in slot address 00 to 07, the 6 outputs can be read at address 100 to 105. The O command cannot be used with an extended card cage.

Example

Command	Description
> A5	Set acceleration to 15 rps^2
> V5	Sets velocity to 5 rps
> D2ØØØØ	Set move distance to 20,000 steps
> OØ1	Set programmable output 1 off and output 2 on
> G	Executes the move (Go)
> OØØ	After the move ends, turn off outputs 1 and 2—outputs 3 - 6 are not affected

PR Status	Position Request			VALID Software Version A3
SYNTAX PR	**UNITS** n = motor steps	**RANGE** None	**DEFAULT** None	**ATTRIBUTES** Immediate
EXECUTION TIME <10 ms		**SEE ALSO** PZ, R, PX, FSB, SP		
RESPONSE TO PR IS See Below				

Description

The PR command is a status request command that provides current motor absolute position information. You can also execute position requests with the 8-bit interface (refer to *Chapter 2, Getting Started*).

Example

Command	Response
PR	+5ØØ—The position is 500 motor steps

PX Status	Report Absolute Encoder Position			VALID Software Version A
SYNTAX <a>PX	UNITS n = encoder steps	RANGE None	DEFAULT None	ATTRIBUTES Immediate/ Sequence
EXECUTION TIME < 10 ms		SEE ALSO PX, R, PR		
RESPONSE TO aPX IS aPXn				

Description This command returns a decimal value indicating the absolute position of the incremental encoder. The absolute position is based on the zero position. The zero position is established when you power up the system. The zero position can also be established after the indexer performs a Go Home (GH) command or a Position Zero (PZ) command. Whether in Motor Step mode or Encoder Step mode, the position is reported in encoder steps.

Example

Command	Description
1PX +4000	The encoder step position is 4,000

PZ Programming	Set Position Zero			VALID Software Version A
SYNTAX PZ	UNITS None	RANGE None	DEFAULT None	ATTRIBUTES Sequence/Immediate
EXECUTION TIME <10 ms		SEE ALSO SP, GH, PR, PX0		

Description This command allows you to set the absolute position register for the motor and encoder to zero.

Example

Command	Description
PR +25,000	Set motor position to +25,000
PX +4,000	Sets motor to +4,000
PZ	Sets motor to zero
PR 0+	Requests absolute positions
PX +0	

Q Edit	Quit Editing Mode			VALID Software Version
SYNTAX Q	UNITS None	RANGE None	DEFAULT None	ATTRIBUTES Immediate
EXECUTION TIME <10 ms		SEE ALSO D, E, EXR, I, ESC Key		

Description You can use this command to exit the Editing mode. Pressing the carriage return key <cr> on an empty line also exits from the Editing mode. You can also press the Escape key (**esc**) to exit from the editing mode.

R Status	Status Report			VALID Software Version A
SYNTAX R	UNITS None	RANGE None	DEFAULT None	ATTRIBUTES Immediate
EXECUTION TIME <10 ms		SEE ALSO PRA, PRX, FR		
RESPONSE TO R IS See Below				

Description This command provides you with a report of the indexer's current status. The status report includes the following information:

- PLC input values
- Indexer output values
- Current position count
- Last operating parameters

A sample response from the **R** command is shown below.

```
*DYNAMIC DATA
*   INPUT BYTE  (BØ - B7) = ØØØØØ11Ø
*   OUTPUT BYTE (BØ - B7) = ØØØØØØØØ
*   HOME: = 1
*REGISTERED DATA
*   Inputs:  I1=Ø I2=Ø I3=Ø I4=Ø I5=Ø
*   Outputs: O1=Ø O2=Ø O3=Ø O4=Ø O5=Ø O6=Ø
*   Motor Position =+Ø
*   Encoder Position =+Ø
*ACTIVE PARAMETERS:
*   MR25ØØØ GHV 1 GHF Ø.1
*   FRØ11ØØØØØ
*   ER 4ØØØ DB5 DW25Ø CG8
*   VSØ VØ AØ
*   MPI D+Ø TØ.5 LØ
```

RE Motion	Resume			VALID Software Version A
SYNTAX RE	UNITS None	RANGE None	DEFAULT None	ATTRIBUTES Immediate
EXECUTION TIME <10 ms		SEE ALSO S, FSF, FSD		

Description The Resume (RE) command continues the execution of an interrupted sequence. The indexer recognizes this command after you issue a Stop (S) command. It enables the Model 301 to complete a move that was stopped. After you initiate a pause, you can clear it with an RE command. You can also execute Pause and Resume commands with the 8-bit interface (refer to *Chapter 2, Getting Started*).

Example

Command	Description
MN	Sets move to Normal mode
D5ØØØØØ	Sets distance to 500,000 steps
A5	Sets acceleration to 5 rps^2
V5	Sets velocity to 5 rps
G	Executes the move (Go)
S	Interrupts execution of move
RE	Resumes execution of move

RV Status	Report Software Revision			VALID Software Version A
SYNTAX RV	UNITS None	RANGE None	DEFAULT None	ATTRIBUTES Immediate
EXECUTION TIME <10 ms		SEE ALSO		
RESPONSE TO RV IS *92-Ø11ØØ6-Ø1A4				

Description The RV command reports the version of software in the indexer.

S Motion	Stop			VALID Software Version A
SYNTAX S	UNITS None	RANGE None	DEFAULT None	ATTRIBUTES Sequence/Immediate
EXECUTION TIME <10 ms		SEE ALSO K, ESC Key, RE		

Description This command allows you to stop axis motion. When the indexer receives the S command, it immediately initiates a ramped deceleration. You can also execute a stop with the 8-bit interface (refer to *Chapter 2, Getting Started*).

SP Programming	Set Position Counter			VALID Software Version A
SYNTAX SP±nnnnnnnn	**UNITS** n = motor steps	**RANGE** 0 - ±nnnnnnnn	**DEFAULT** None	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO** PRX, PZ, PR		

Description This command allows you to set the motor position counter to the specified value.

Example

Command	Description
SP-5ØØØ	Sets the motor position to -5,000 steps

T Programming	Time Delay			VALID Software Version A
SYNTAX Tnnn.nn	**UNITS** n = seconds	**RANGE** 0.01 -999.99	**DEFAULT** None	**ATTRIBUTES** Sequence
EXECUTION TIME <10 ms		**SEE ALSO** MC, DLY, R		

Description The T command delays program execution for a number of seconds based on the value that you specify.

Example

Command	Description
> T5.5	Delays motion for 5.5 seconds
> G	Executes the move (Go)

TEST Programming	Test Simulate			VALID Software Version A
SYNTAX TESTnnnnn	**UNITS** n = bit pattern	**RANGE** 0 or 1	**DEFAULT** None	**ATTRIBUTES** Immediate
EXECUTION TIME <10 ms		**SEE ALSO** IF, R, TR		

Description The **TEST** command allows you to simulate the inputs from the PLC bus. This command is useful when you want to test a sequence, but the PLC program is not yet finished. You can also use this command to debug sequence commands. Assume the inputs are all Ø's. Use the Status Request (R) command to review *registered data* and determine current input status.

The **TEST** command will be ignored if bits 5 and 6 of the PLC 8-bit interface are active.

Example

Command	Description
5 : V5 A5 MC G	Begins continuous motion at 5 rps
TRØØØØ1	Waits until inputs match
V1Ø G	Slew to 10 rps
TR1ØØØØ	Waits until inputs match
S	Stops the motor
XT	Ends sequence #5 definition

Use XG5 to execute sequence #5. The motor will begin to move at 5 rps. Use the TEST command to force the inputs to match the TR command values.

```
> TEST0000 1
```

This will allow the motor to continue program execution. The motor will accelerate to 10 rps. Now use the TEST command to force the inputs to match the values of the second TR command.

```
> TEST1000
```

This will stop the motor. You can also see the impact of the TEST command by using the R command to review the state of the *registered data*.

TR Programming	Trigger			VALID Software Version A
SYNTAX	UNITS	RANGE	DEFAULT	ATTRIBUTES
TRnnnnn	n = input bit pattern	0, 1, or X	0	Sequence
EXECUTION TIME <10 ms		SEE ALSO IF, R (check reg. data), TEST, K		

Description

The TR command is only valid in a sequence. Commands that follow the TR command will not be executed until the PLC or the TEST command make the inputs match the TR command's value (nnnnn). Refer to *Chapter 2, Getting Started* for more information on hardware latching of inputs. Refer to the TEST command description for more information on setting inputs through the serial port.

Example

Command	Description
5 : V5 A5 MC G	Begins continuous motion at 5 rps
TR10111	Sequence execution stops here until the registered inputs match the TR command's values
V10 G	Accelerates to 10 rps

V Motion	Velocity			VALID Software Version A
SYNTAX	UNITS	RANGE	DEFAULT	ATTRIBUTES
Vnn.nn	n = rps	0.01 - 99.99	0 rps	Sequence/Immediate
EXECUTION TIME <10 ms		SEE ALSO A, D, G, VS, R		

Description

This command allows you to set the maximum speed that an axis may reach during a move. The maximum step output rate is 325 KHz.

Example

Command	Description
MC	Sets unit to Continuous mode
A1	Sets acceleration to 1 rps^2
V5	Sets velocity to 5 rps
D25000	Sets distance to 25,000 steps
G	Executes the move (Go)

VS Motion	Initial Velocity			VALID Software Version A
SYNTAX VSnn.nn	**UNITS** n = rps	**RANGE** 0.01 - 99.99	**DEFAULT** 0 rps	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO** A, D, G, V, R		

Description At the outset of a move, this command sets the initial velocity. This command allows you to avoid specific low-frequency ranges that might stall or resonate step motors.

Example

Command	Description
MC	Sets unit to Continuous mode
A1	Sets acceleration to 1 rps^2
V5	Sets velocity to 1 rps
VS1	Sets initial velocity at 1 rps
G	Executes the move (Go)

The initial velocity will be 1rps and will ramp up to 5rps.

XG Programming	Execute Sequence			VALID Software Version A
SYNTAX XGnn	**UNITS** n = sequence #	**RANGE** 0 - 63	**DEFAULT** None	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO** EXR, XR, XT		

Description The XG command begins the execution of a program starting at the sequence # that you specify (nn). You may also use this command within a sequence to begin executing another sequence. You can execute a sequence with the 8-bit interface (refer to *Chapter 2, Getting Started*). Executing sequence Ø will cause execution to begin at the first program in memory.

Example

Command	Description
XG7	Executes the commands in sequence #7

XR Programming	Execute Sequence w/Return			VALID Software Version A
SYNTAX XRnn	**UNITS** n = sequence #	**RANGE** 0 - 63	**DEFAULT** None	**ATTRIBUTES** Sequence/Immediate
EXECUTION TIME <10 ms		**SEE ALSO** EXR, XG , XT		

Description

When you use the **XR** command within a sequence, it begins the execution of a program starting at the sequence # that you specify (**nn**). When the indexer reaches the End Sequence (**XT**) command, execution is returned to the command line that follows the **XRnn** command. This command is especially useful when you want to use subroutine sequences. This command works just like the **XG** command in Immediate mode. Executing sequence 0 will start program execution at the beginning of memory

Example

Command

```
1 : D1ØØØ  G  XT
2 : D2ØØØ  G  XT
3 : D3ØØØ  G  XT
4 : D4ØØØ  G  XT
20:A5  V5  MN
    IF (IØØØØ1) XR1 NIF
    IF (IØØØ1Ø) XR2 NIF
    IF (IØØØ11) XR3 NIF
    IF (IØØ1ØØ) XR4 NIF
    XG2Ø
    XT
```

Enter **XG20**. Motor will move different distances based upon inputs.

XT Programming	End Sequence			VALID Software Version A
SYNTAX XT	**UNITS** None	**RANGE** None	**DEFAULT** None	**ATTRIBUTES** Sequence
EXECUTION TIME <10 ms		**SEE ALSO** EXR, XG, XR		

Description

The **XT** command ends a sequence. If the sequence is called from an **XG** command, program execution stops when the indexer reaches an **XT** command. If the sequence is initiated with an **XR** command, program execution returns to the line that follows the **XRnn** command when the indexer reaches an **XT** command. If the sequence does not contain an **XT** command, program execution will move on to the next sequence.

XTR Programming	Enable/Disable Trace Mode			VALID Software Version A
SYNTAX XTRn	**UNITS** n = enable/ disable	**RANGE** 1 = Trace mode On 0 = Trace mode Off	**DEFAULT** 0	**ATTRIBUTES** Immediate
EXECUTION TIME <10 ms		**SEE ALSO**		

Description When you set the XTR command to 1, the Trace mode is enabled. As a debugging tool, the Trace mode sends user programs one character at a time (as the indexer reads it). This mode tends to slow down program execution slightly. The XTR command is displayed before it is executed.

Example

Command	Description
XTR1	Instructs the Model 301 to send the executed command to the terminal

Y Programming	Terminate Loop			VALID Software Version A
SYNTAX Y	**UNITS** None	**RANGE** None	**DEFAULT** None	**ATTRIBUTES** Immediate
EXECUTION TIME <10 ms		**SEE ALSO** L, N		

Description The Y command terminates loop execution at the end of the loop that is currently being executed. The sequence being run will be terminated.

Z Set-Up	Software Reset			VALID Software Version A
SYNTAX Z	**UNITS** None	**RANGE** None	**DEFAULT** None	**ATTRIBUTES** Immediate
EXECUTION TIME <10 ms		**SEE ALSO** None		

Description The Z command resets the Model 301. Many parameters are reset to their default values.

Chapter 6. Hardware Reference

Chapter Objectives

The information in this chapter will enable you to:

- Use this chapter as a quick-reference tool for most system specifications
- Use this chapter as a quick-reference tool for proper I/O connections

Environmental Specifications

Ambient Operating Temperature: 32°F - 140°F (0°C - 60°C)
Storage Temperature: 40°F - 185°F (-40°C - 85°C).
Humidity: 5 - 95%

Electrical Specifications

This section summarizes the power supply requirements and electrical characteristics of the available interfaces.

Power Supply Requirements

The Model 301 card requires only the +9V supply provided by the PLC rack power supply. The Model 301 uses a maximum of 150 mA of the +9V supply. This is equivalent to 15 units of the load described in the PLC user manual.

To interface the Model 301 to the drive and encoder a separate +5V supply is required.

Serial Communications (RS-232C)

The 9-pin female D-shell connector on the Model 301 front panel provides the connection for RS-232C communications. Figure 6-1 shows the pin assignments for the serial communications port. The indexer's serial communications parameters are listed below:

Baud Rate: 9,600
Data Bits: 8
Stop Bits: 1
Parity: None
XON/XOFF: Not used
Echo: On

SuperPaint!Pat:User Guides:Model 303 Indexer:Model 303 Figures:Fig. 6-1!Draw(154,88:381,186)

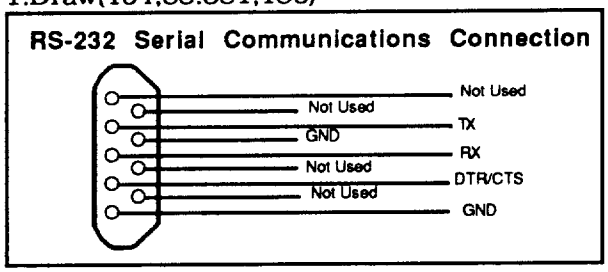

Figure 6-1. RS-232C Serial Communications

Cable Pin Outs

The pinout for the RS-232C 9-pin D connector on the Model 301 is provided below. Compumotor provides a cable for this connector (**part number 71-011319-10**).

Model 301 9-Pin Connector	Function	Color	Terminal 25-Pin Connector
Pin #2	Tx	RED	Pin #3
Pin #3	Rx	GREEN	Pin #2
Pin #5	Shield	-	No Connection
Pin #7	GND	WHITE	Pin #7

The pin out for the 25-pin D motor/driver cable (**part number 71-011159-10**) for connection between the Model 301 and a *Compumotor drive* is provided below.

Compumotor Indexer 25-Pin Connector	Color	Function
Pin #1	RED	+5V
Pin #2	RED	+5V
Pin #14	WHITE	STEP-
Pin #15	GREEN	DIR-
Pin #16	RED	+5V
Pin #17	WHITE	SHUTDOWN-

10-Pin Screw Terminal Connections

The pin connections for the inputs and outputs on the 12-pin screw terminal are shown in Figure 6-2.

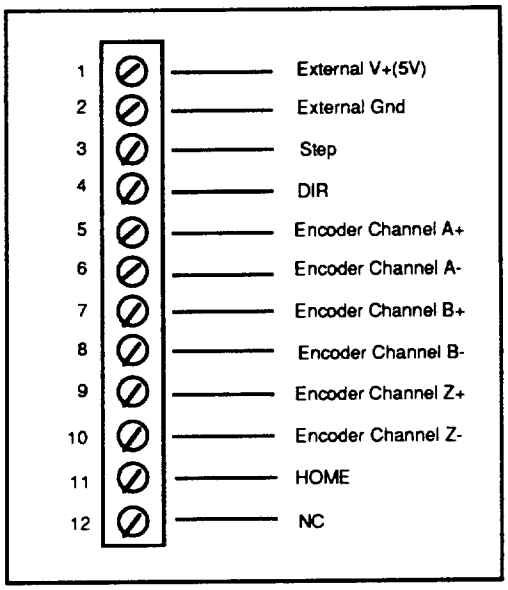

Figure 6-2. 12 Pin I/O Connector

The typical output circuit is used for the step & direction signals (Figure 6-3).

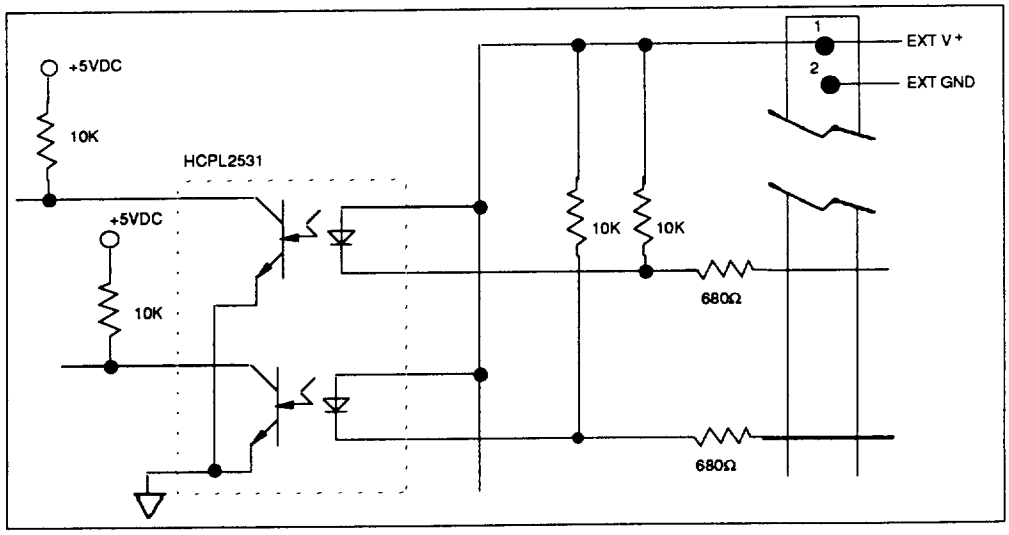

Figure 6-3. Typical Input Circuit

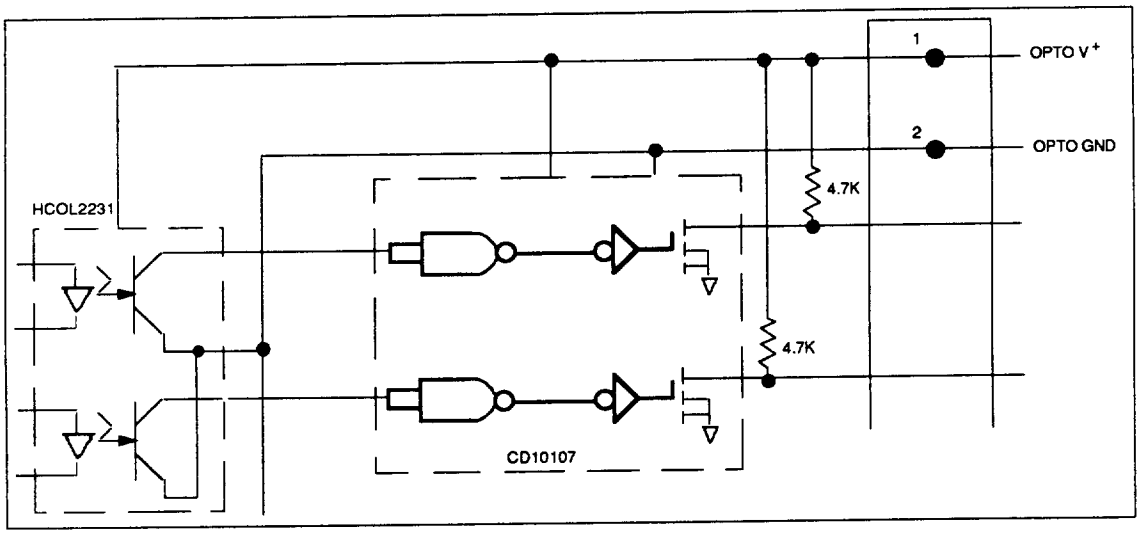

Figure 6-4. Typical Output Circuits

This input circuit is used for the home input. The encoder inputs use 26LS32 line driver receivers.

All of the inputs and outputs as well as the RS-232C interface are optically isolated.

LEDs

The busy LED indicates that a sequence is currently running. The moving LED indicates that steps are being sent to the drive.

System Specifications

The following performance specifications apply to the Model 301 indexer.

I/O Specifications

Maximum Step Output Frequency: 325,000 pulses/second

Memory

Nonvolatile Memory: 2 K bytes
Expanded Memory Option: 8 K bytes

Chapter 7. Troubleshooting

Chapter Objectives

The information in this chapter will enable you to:

- Maintain the system's components to ensure smooth, efficient operation
- Isolate and resolve system hardware problems
- Isolate and resolve system software problems

Troubleshooting

This section discusses methods to identify, isolate, and resolve problems that may occur with the Model 301.

Problem Isolation

If your system malfunctions, you must identify and isolate the problem. When you accomplish this, you can begin to resolve 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.

Try to determine if the problem is mechanical, electrical, or software-related. Can you repeat or re-create the problem? Do not attempt to make quick rationalizations about problems. Random events may appear to be related, but they are not necessarily contributing factors to your problem. You must carefully investigate and decipher the events that occurred before the subsequent system problem.

You may be experiencing more than one problem. You must solve one problem at a time. Log (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. If your system's problem persists, contact Compumotor's Applications Department.

Reducing Electrical Noise

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

RS-232C Communications

If you are having problems communicating with the Model 301 indexer, use the following procedure to troubleshoot the RS-232C communications interface.

1. Configure the terminal and the Model 301 to the same baud rate, number of data bits, number of stop bits, and parity.

2. Ensure that the transmit connection (Tx) of the terminal is wired to the receive connection (Rx) of the Model 301, and that the receive connection (Rx) of the terminal is wired to transmit connection (Tx) of the Model 301.

   Try switching the receive and transmit wires on either the terminal or the Model 301 if communication fails.

   Verify that you have wired RS-232C on the Model 301 to the port selected.

3. Some serial ports require handshaking. If so, you may establish three-wire communication by connecting RTS to CTS (usually pins #4 and #5) and DSR to DTR (usually pins #6 to #20) at the Model 301 end.

4. If you receive double characters, for instance typing **A** and receiving **AA**, your computer is set for half duplex.

5. Use DC common or signal ground as your reference, ***not earth ground***.

6. 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, ***shield the cable to earth ground at one end only***.

7. To test your terminal for proper three wire communication, unhook your peripheral device and transmit a character. You should not receive an echoed character. If you do, you are in half duplex mode. Change the setup to full duplex. Connect the host's transmit and receive lines and send another character. You should receive the echoed character. If you do not receive the echoed character, consult the terminal manufacturer for the unit's serial interface and proper pin outs.

Motor Fails to Move

Test the motor to see if it has holding torque. If there is no holding torque, here are some probable causes.

- There is no AC power.
- There are bad connections or bad cables. Disconnect the power connector, then use an ohm meter to monitor continuity between the motor-to-drive cable.

If the unit has holding torque and the motor shaft still fails to move, here are some probable causes:

- The load is jammed. You should hear the drive attempting to move the motor. Remove AC power from the drive and verify that you can move the load manually away from the point of the jam.
- The +5V supply is improperly connected
- Indexer parameters are incorrectly set up. If certain parameters are out of range or are missing, the motor will not move when you issue the Go (**G**) command.

Use the R status command to determine what is preventing the move. Check **A**, **V**, and **D** commands to make sure that all the parameters are set properly. *The Model 301's busy (Green) LED will be on if steps are being sent out.*

The following are additional troubleshooting techniques:

- Check the motor for damage. Also check the motor cable to see if it is damaged or shortened. These conditions may cause the drive to fault.
- Ohm the motor and cables to make sure that short-circuits do not exist between phases or to earth ground. On your most sensitive scale, the resistance across each motor phase should be consistently low (but not zero) and similar to each other. On your highest scale, the resistance between motor phases and between each phase and earth ground should be infinite.

Encoder Feedback

Check wiring between Model 301 and the Encoder. If no encoder pulses are received, there may be no power to the encoder or the phases are not wired properly.

If the encoder counts are in the wrong direction, the phases need to be reversed.

Connect the following:

- Phase A+ to Phase B+
- Phase A- to Phase B-
- Phase B+ to Phase A+
- Phase B- to Phase A-

Appendices

Command Listing

ESC (Escape Key)
@ (Comments Delimiter—*Edit Mode*)

A (Acceleration)

CG (Correction Gain)
CLR (Clear)

D (Distance)
D (Delete—*Edit Mode*)
DB (Deadband)
DLY (Delay)
DW (Deadband Window)

E (Edit)
ER (Encoder Resolution)
EXR (Edit a Sequence in a Program)

FR (Report Setup Status)
FSA (Set Indexer to Incremental/Absolute Mode)
FSB (Set Indexer to Motor/Encoder Step Mode)
FSC (Enable/Disable Position Maintenance)
FSD (Stop on Stall)
FSE ()Turn on Output #1 on Stall)
FSF (Kill Motion on Trigger)
FSG (Turn on Output 2 When Within Deadband)
FSH (Enable Stall Detection)

G (Go)
GH (Go Home)
GHF (Go Home Final Speed)
GHV (Go Home Velocity)

H± (Set Direction)

I (Insert a Line in a Sequence)
IF (Conditional IF)

K (Kill)

L (Loop)
LST (List Program)

MC (Mode Continuous)
MN (Mode Normal)
MPA (Mode Position Absolute)
MPI (Mode Position Incremental)
MR (Motor Resolution)

N (End of Loop)
NIF (End of IF Commands)

O (Set Programmable Outputs)

PR (Position Request)
PX (Report Absolute Encoder Position)
PZ (Set Position Zero)

Q (Quit Editing Mode)

R (Status Report)
RE (Resume)
RV (Revision)

S (Stop)
SP (Set Position Counter)

T (Time Delay)
TEST (Test Simulate)
TR (Trigger)

V (Velocity)
VS (Initial Velocity)

XG (Execute Sequence)
XR (Execute Sequence w/Return)
XT (End Sequence)
XTR (Enable/Disable Trace Mode)

Y (Terminate Loop)

Z (Reset Software)

Glossary

Absolute Positioning
Refers to a motion control system employing position feedback devices (absolute encoders) to maintain a given mechanical location.

Absolute Programming
A positioning coordinate reference wherein all positions are specified relative to some reference, or *home* position. This is different from incremental programming, where distances are specified relative to the current position.

Acceleration
The change in velocity as a function of time. Acceleration usually refers to increasing velocity and deceleration describes decreasing velocity.

Accuracy
A measure of the difference between expected position and actual position of a motor or mechanical system. Motor accuracy is usually specified as an angle representing the maximum deviation from expected position.

Address
Multiple devices, each with a separate address or unit number, can be controlled on the same bus. The address allows the host to communicate individually to each device.

Ambient Temperature
The temperature of the cooling medium, usually air, immediately surrounding the motor or another device.

ASCII
American Standard Code for Information Interchange. This code assigns a number to each numeral and letter of the alphabet. In this manner, information can be transmitted between machines as a series of binary numbers.

Bandwidth
The frequency range in which the magnitude of the system gain expressed in dB is greater than -3 dB.

Baud Rate
The number of bits transmitted per second. Typical rates include 300; 600;1,200; 2,400; 4,800; 9,600; 19,200. This means at 9,600 baud, one character can be sent nearly every millisecond.

BCD
Binary Coded Decimal is an encoding technique used to describe the numbers 0 - 9 with four digital (on or off) signal lines. Popular in machine tool equipment, BCD interfaces are now giving way to interfaces requiring fewer wires—such as RS-232C.

Bit
Abbreviation of Binary Digit, the smallest unit of memory equal to 1 or 0.

Block Diagram
A simplified schematic representing components and signal flow through a system.

Bode Plot
A graph of system gain and phase versus input frequency that graphically illustrates the steady state characteristics of the system.

Break Frequency
Frequency(ies) at which the gain changes slope on a Bode plot. (Break frequencies correspond to the poles and zeroes of the system.)

Byte
A group of 8 bits treated as a whole, with 256 possible combinations of ones and zeros, each combination representing a unique piece of information.

Closed Loop
A term relating to any system where the output is measured and compared to the input. The output is adjusted to reach the desired condition. In motion control, the term is used to describe a system wherein a velocity or position (or both) transducer is used to generate correction signals by comparison to desired parameters.

Critical Damping
A system is critically damped when the response to a step change in desired velocity or position is achieved in the minimum possible time with little or no overshoot.

Crossover Frequency
The frequency at which the gain intercepts the 0 dB point on a Bode Plot. (Used in reference to the open-loop gain plot.)

Daisy-Chain
A term used to describe the linking of several RS-232C devices in sequence such that a single data stream flows through one device and on to the next. Daisy-chained devices usually are distinguished by device addresses, which serve to indicate the desired destination for data in the stream.

Damping
An indication of the rate of decay of a signal to its steady state value. Related to settling time.

Damping Ratio
Ratio of actual damping to critical damping. Less than one is an underdamped system and greater than one is an overdamped system.

Data Bits
Since the ASCII character set consists of 128 characters, computers may transmit only seven bits of data. However, most computers support an eight bit extended ASCII character set.

Dead Band
A range of input signals for which there is no system response.

Decibel
A logarithmic measurement of gain. If G is a system gain (ratio of output to input), then 20 log G equals gain in decibels (dB).

Detent Torque
The minimal torque present in an unenergized motor. The detent torque of a Compumotor or step motor is typically about one percent of its static energized torque.

DTE
Data Communications Equipment transmits on pin #2 and receives on pin #3.

Duty Cycle
For a repetitive cycle, the ratio of on time to total cycle time.
Duty Cycle = *On Time* ÷ *On Time + Off Time*

Efficiency
The ratio of power output to power input.

Encoder
A device that translates mechanical motion into electronic signals used for monitoring position or velocity.

Friction
A resistance to motion caused by surfaces rubbing together. Friction can be constant with varying speed (Coulomb friction) or proportional to speed (viscous friction).

Full Duplex
The terminal will display only received or echoed characters.

Gain
The ratio of system output signal to system input signal.

Half Duplex
In half duplex mode, a terminal will display every character transmitted. It may also display the received character.

Hand Shaking Signals
RST: Request To Send
CTS: Clear To Send
DSR: Data Set Ready
DTR: Data Terminal Ready
IDB: Input Data Buffer
ODB: Output Data Buffer

Holding Torque
Sometimes called static torque, it specifies the maximum external force or torque that can be applied to a stopped, energized motor without causing the rotor to rotate continuously.

Home
A reference position in a motion control system, usually derived from a mechanical datum. Often designated as the ***zero*** position.

Hysteresis
The difference in response of a system to an increasing or a decreasing input signal.

IEEE-488
A digital data communications standard popular in instrumentation electronics. This parallel interface is also known as GPIB, or General Purpose Interface Bus.

Incremental Motion
One step of motion for each step command (usually a pulse) received.

Incremental Programming
A coordinated system where position or distances are specified relative to the current position.

Inertia
A measure of an object's resistance to a change in velocity. The larger an object's inertia, the larger the torque that is required to accelerate or decelerate it. Inertia is a function of an object's mass and its shape.

Inertial Match
For most efficient operation, the system coupling ratio should be selected so that the reflected inertia of the load is equal to the rotor inertia of the motor.

Lead Compensation Algorithm
A mathematical equation implemented by a computer to decrease the delay between the input and output of a system.

Limits
Properly designed motion control systems have sensors called limits that alert the control electronics that the physical end of travel is being approached and that motion should stop.

Logic Ground
An electrical potential to which all control signals in a particular system are referenced.

Microstepping
An electronic control technique that proportions the current in a step motor's windings to provide additional intermediate positions between poles. Produces smooth rotation over a wide speed range and high positional resolution.

Null Modem
A simple device or set of connectors which switches the receive and transmit lines of a three wire RS-232C connector.

Open Collector
A term used to describe a signal output that is performed with a transistor. An open collector output acts like a switch closure with one end of the switch at ground potential and the other end of the switch accessible.

Open Loop
Refers to a motion control system where no external sensors are used to provide position or velocity correction signals.

OPTO-isolated
A method of sending a signal from one piece of equipment to another without the usual requirement of common ground potentials. The signal is transmitted optically with a light source (usually a Light Emitting Diode) and a light sensor (usually a photosensitive transistor). These optical components provide electrical isolation.

Parallel
Refers to a data communication format wherein many signal lines communicate more than one piece of data at the same time.

Parity
An RS-232C error detection scheme that can detect an odd number of transmission errors.

Phase Angle
The angle at which the steady state input signal to a system leads the output signal.

Phase Margin
The difference between 180° and the phase angle of a system at its crossover frequency.

Pole
A frequency at which the transfer function of a system goes to infinity.

Pulse Rate
The frequency of the step pulses applied to a motor driver. The pulse rate multiplied by the resolution of the motor/drive combination (in steps per revolution) yields the rotational speed in rps.

Ramping
The acceleration and deceleration of a motor. May also refer to the change in frequency of the applied step pulse train.

Rated Torque
The torque producing capacity of a motor at a given speed. This is the maximum torque the motor can deliver to a load and is usually specified with a torque/speed curve.

Relative Accuracy
Also referred to as *Step-to-Step Accuracy*. This specification tells how microsteps can change in size. In a perfect system, microsteps would all be exactly the same size, but drive characteristics and the absolute accuracy of the motor cause the steps to expand and contract by an amount up to the relative accuracy figure. The error is not cumulative.

Repeatability
The degree to which the positioning accuracy for a given move performed repetitively can be duplicated.

Resolution
The smallest positioning increment that can be achieved. Frequently defined as the number of steps required for a motor's shaft to rotate one complete revolution.

Ringing
Oscillation of a system following a sudden change in state.

RMS Torque
For an intermittent duty cycle application, the RMS Torque is equal to the steady state torque which would produce the same amount of motor heating over long periods of time.

Where:
Ti = Torque during interval i
t = Time of interval i

RS-232C
A data communications standard that encodes a string of data on one line in a time sequential format. The standard specifies the proper voltage and timing requirements so that different manufacturers' devices are compatible.

Slew
In motion control, the portion of a move made at a constant non-zero velocity.

Speed
Describes the linear or rotational velocity of a motor or other object in motion.

Start Bits
RS-232C character transmissions begin with a bit which signals the receiver that data is now being transmitted.

Static Torque
The maximum torque available at zero speed.

Step Angle
The angle the shaft rotates upon receipt of a single step command.

Stiffness
The ability to resist movement induced by an applied torque. Is often specified as a torque displacement curve, indicating the amount a motor shaft will rotate upon application of a known external force when stopped.

Stop Bits
When using RS-232C, one or two bits are added to every character to signal the end of a character.

Synchronism
A motor rotating at a speed correctly corresponding to the applied step pulse frequency is in *synchronism*. Load torques in excess of the motor's capacity (rated torque) cause a loss of synchronism. This condition does not damage step motors.

Text/Echo (Off/On)
This setup allows received characters to be re-transmitted back to the original sending device. Echoing characters can be used to verify or ***close the loop*** on a transmission.

Torque
Force tending to produce rotation.

Torque-to Inertia Ratio
Defined as a motor's holding torque divided by the inertia of its rotor. The higher the ratio, the higher a motor's maximum acceleration capability will be.

Transfer Function
A mathematical means of expressing the output to input relationship of a system.

TTL
Transistor-Transistor Logic. Describes a common digital logic device family that is used in most modern digital electronics. TTL signals have two distinct states that are described with a voltage—a logical ***zero*** or ***low*** is represented by a voltage of less than 0.8V and a logical ***one*** or ***high*** is represented by a voltage from 2.5V to 5V.

XON/XOFF
Two ASCII characters supported in some serial communication programs. If supported, the receiving device transmits an XOFF character to the host when its character buffer is full. The XOFF character directs the host to stop transmitting characters to the device. Once the buffer empties the device will transmit an XON character to signal the host to resume transmission.

Zero
A frequency at which the transfer function of a system goes to zero.

Index