ARTISAN®
TECHNOLOGY GROUP

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

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

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

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

Custom engineering
so your equipment works exactly as you specify.

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

Expert team | Trust guarantee | 100% satisfaction

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

Find the *Parker / Compumotor JSI* at our website: *Click HERE*

Compumotor

JSI Digital Servo Controller User Guide

Compumotor Division
Parker Hannifin Corporation
p/n 88-009040-01 A

Parker

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

User Guide Change Summary

The following is a summary of the primary changes to this user guide since the last version was released. This user guide, version 88-009040-01**A**, supersedes version 88-009040-01**Y**.

When a user guide is updated, the new or changed text is differentiated with a change bar in the right margin (this paragraph is an example). If an entire chapter is changed, the change bar is located to the right of the chapter title.

In general, this user guide was updated with new information related to operating the JSI with newly released Compumotor absolute encoders (AR23 with -1 decoder, AR-C, AL-C, and Photo-Trak). Technical changes to each chapter are synopsized below.

Chapters 1 & 2

No Changes

Chapter 3. Installation

Sections changed because of new absolute encoders:

- *Encoder Compatibility*
- *Encoders* (in *Electrical System Connections* section)
- *Using the JSI with an Absolute Encoder*

Chapter 4. Application Design

Updated *Closed-Loop Operation* section to include operation with new encoders and -1 version of AR23.

Chapter 5. Software Reference

Updated the CFB command to include operation with new encoders and -1 version of AR23.

Chapter 6. Hardware Reference

Updated *POSITION FEEDBACK* section to include operation with new encoders and -1 version of AR23.

Chapter 7. Maintenance & Troubleshooting

- Added procedure for returning product to affect repairs of upgrades
- Updated RS-232C Troubleshooting procedure

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

Table of Contents

Table of Contents i
List of Figures iii
List of Tables iii
How To Use This Manual iv
Assumptions iv
Contents of This Manual iv
Installation Process Overview iv
Developing Your Application v
Installation Preparation v
Conventions v
Commands vi
Warnings & Cautions vi
Levels vi
Related Publications vii

Chapter 1. Introduction 1
Product Description 1
Product Features 1
Theory of Operation 2

Chapter 2. Getting Started 3
What You Should Have 3
Ship Kit Table 3
Basic System Configuration 3
JSI Front Panel Diagram 4
Setting Controller Functions 5
Testing Output Levels 9

Chapter 3. Installation 13
Environmental Considerations 13
Complete System Configuration 13
Setting Drive and Motor Functions 13
Setting Controller Functions 13
Encoder Compatibility 14
Setting Incremental Encoder Resolution 15
Electrical System Connections 16
Wiring Guidelines 16
System Mounting 25
Controller Mounting 25
Encoder Mounting 25
Verifying Proper Installation 26
RS-232C Communication 26
Encoders 26
Inputs & Outputs 32

Chapter 4. Application Design 35
Motion Control Concepts 35
Move Profiles 35
Move Times 35
Incremental vs. Absolute Positioning 36
Application Considerations 38
Positional Accuracy vs. Repeatability 38
Closed-Loop Operation 38
Homing Function 39
Programmable Inputs 41
Delays 41
Branching 41
Subroutines 43
Loops 44
POBs (Programmable Output Bits) 45
Move Completion Signal 45
Sequence Commands 46
Stand-alone Operation 47
Host Computer Operation 52
PLC Operation 56
Tuning The JSI 58

Chapter 5. Software Reference 63
Definition & Description of Command Types 63
Command List 64

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

Chapter 6. Hardware Reference **167**
Factory Default Settings 167
Environmental Specifications 167
I/O Specifications & Descriptions 168
Connectors 168
POWER 169
FAULT OUT 170
COMMAND OUT 170
ANALOG IN 171
PROG INPUTS 171
PROG OUTPUTS 172
ENCODER (incremental encoder) 172
AMPL 172
POSITION FEEDBACK (absolute encoder) 172
RS-232 172
LEDS 173
Dimensional Drawings 174

Chapter 7. Maintenance & Troubleshooting **175**
Maintenance 175
Spare Parts Table 175
Battery 175
Troubleshooting 175
Problem Isolation 175
Reducing Electrical Noise 177
Returning the System 177

Appendices **179**
Command List 179
Glossary 181
Index 183

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

List of Figures

Figure 1-1. Typical Closed-Loop Servo System 2

Figure 2-1. JSI Front Panel Diagram 4
Figure 2-2. JSI Controller AC Power Wiring Diagram 5
Figure 2-3. JSI Controller DC Power Wiring Diagram 6
Figure 2-4. Confirm Device Address Setting 6
Figure 2-5. Wiring RS-232C Interface to a Computer or a Terminal 7
Figure 2-6. Display Baud Rate Setting 8
Figure 2-7. JSI/Voltmeter Wiring Diagram 10

Figure 3-1. Quadrature Signals From an Incremental Encoder 14
Figure 3-2. Analog Input and Command Output Connections 17
Figure 3-3. JSI Limit Switches 18
Figure 3-4. JSI/Incremental Encoder Connection 19
Figure 3-5. Typical JSI Encoder Input Circuit 20
Figure 3-6. Typical JSI/Absolute Encoder Connection & JSI/Drive Connection 20
Figure 3-7. JSI-to-Decoder Box Cable (for use with -4 Decoder Box and AR23) 21
Figure 3-8. JSI-to-Decoder Box Cable (for use with AR-C, AL-C, or Photo-Trak) 22
Figure 3-9. JSI-to-Decoder Box Cable (for use with -1 Decoder Box and AR23) 22
Figure 3-10. Programmable I/O Wiring Diagram 24
Figure 3-11. Panel Layout Guidelines and Mounting Options 25
Figure 3-12. JSI System Configuration 26

Figure 4-1. Triangular and Trapezoidal Profiles 36
Figure 4-2. JSI Thumbwheel Installation 50
Figure 4-3. Daisy-Chained JSIs 53
Figure 4-4. PLC Connection 57
Figure 4-5. JSI Controller Pushbuttons 59

Figure 6-1. JSI Front-Panel Diagram 168
Figure 6-2. JSI Controller AC Power Wiring Diagram 169
Figure 6-3. JSI Controller DC Power Wiring Diagram 170
Figure 6-4. Analog Input and Command Output Connections 171
Figure 6-5. Typical JSI Output Circuit 172
Figure 6-6. Wiring RS-232C Interface to a Computer or a Terminal 173
Figure 6-7. JSI Dimensional Diagrams 174

List of Tables

Table 2-1. Ship Kit Table 3

Table 3-1. Incremental Encoder Pin Connections 19

Table 4-1. Input BCD Values 51
Table 4-2. JSI Pushbutton Tuning Functions 60

Table 7-1. Spare Parts Table 175
Table 7-2. Diagnostic Codes Table 176

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

How To Use This Manual

This manual 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 the information in this manual.

Assumptions

This user guide assumes that you have a fundamental understanding of the information or are using the equipment listed below.

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

With this basic level of understanding, you will be able to effectively use this manual to install, develop, and maintain your system.

The examples in this user guide assume that you are using a DC servo motor

Contents of This Manual

This manual contains the following information:

Chapter 1: Introduction This chapter provides a description of the product and a brief account of its specific features.

Chapter 2. Getting Started This chapter helps you get your new system up and running. It includes a step-by-step procedure to test your system and ensure that it is operating properly.

Chapter 3. Installation This chapter contains a detailed list of the items you should have received with your JSI shipment. Figures (illustrations) and references that you will need to properly install the JSI are provided in this chapter.

Chapter 4. Application Design This chapter provides high-level information regarding encoder compatibility with the JSI, software limits, and sequence execution capabilities.

Chapter 5. Software Reference This chapter discusses X-Series commands for the JSI in detail. It is divided into two sections. The first section explains command syntax and system parameters that affect command usage and provides examples of how to use them. The second section contains an alphabetical listing of all commands, with a syntax and command description for each command.

Chapter 6. Hardware Reference The electrical, physical, environmental, and performance specifications of the JSI are described in this chapter.

Appendix The appendix is a quick-reference tool that contains a list of X Language commands associated with the product, a glossary of terms, an ASCII table, and an index.

Installation Process Overview

To ensure trouble-free operation, pay special attention to the following factors:

- The environment in which the JSI will operate
- The system layout and mounting
- The wiring and grounding practices used

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

Developing Your Application

Before you attempt to develop and implement your application, you should consider the following issues:

1. Define the requirements of your application. Clarify system expectations.
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 be ensured only if you complete all procedures in the proper sequence.

Installation Preparation

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

1. Review this entire manual. Become familiar with the manual'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 manual. 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 all of the system's functions and used or become familiar with all of 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 tune and 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 operation.

Conventions

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

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

Commands

The command examples in this user guide are presented vertically to help you read and understand them. When you actually type these commands at your computer keyboard, they will be displayed horizontally. All commands that you are instructed to enter are displayed as they should appear on your computer CRT. A one-line explanation of the command is provided next to each example. Be sure to add a delimiter (space or carriage return) after each command in a sequence. Refer to the example below.

Command	Description
> A1Ø	Sets acceleration to 10 rps^2
> V1Ø	Sets velocity to 10 rps
> D1ØØØ	Sets distance to 1,000 steps
> G	Executes the move (Go)

Responses are presented as the should appear on the CRT. An example is provided below.

Command	Response
> 1XU1	*SSJ1_A1Ø_V1Ø_LD3_MN_SN1Ø

In the interactive mode (default mode), the indexer responds with a prompt (>) when it receives a valid command and a question mark (?) when it does not receive a valid command. These interactive responses are preceded with a carriage return and a line feed. While defining loops or sequences, no responses are received until the loop or sequence is executed or until the XT command is executed. Refer to Chapter 5, *Software Reference*, for detailed command descriptions.

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.

WARNING

Do not touch the system 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.

Levels

The JSI has inputs and outputs that you can turn on or off. The voltage levels and terminology associated with these inputs and outputs are discussed below.

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

Inputs

With the JSI, you can configure the active input signal using the Input Level (**INL**) command. The JSI pulls the voltage level of these inputs up to +5VDC.

Command	Description
> **INLØ**	ON (1): 0V applied to the input (current flows) OFF (Ø): 5V - 30V applied to the input (no current flows)
> **INL1**	ON (1): 5V - 30V applied to the input (no current flows) OFF (Ø): 0V applied to the input (current flows)

Throughout this user guide, we will assume that the inputs are in **INLØ** mode (ON means that 0V are applied to the input and current is flowing). You can use 5VDC - 30VDC to control the input.

Programmable Output Bits

With the JSI, you can configure the active output signal using the Output Level (**OUTL**) command.

Command	Description
> **OUTLØ**	ON (1): 0V Output (current flows) OFF (Ø): 5V - 30V Output (no current flows)
> **OUTL1**	ON (1): 5V - 30V Output (no current flows) OFF (Ø): 0V Output (current flows)

Throughout this user guide, we will assume that the outputs are in **OUTLØ** mode (ON means that the voltage level is 0V and the current is flowing). The outputs are open-collector outputs. You can use 5VDC to 30VDC to power outputs **OUT1** - **OUT8** and sink up to 300mA of current.

Related Publications

The following publications may be helpful resources:

- Current Parker Compumotor Motion Control Catalog
- Seyer, Martin. *RS-232C Made Easy: Connecting Computers, Printers, Terminals and Modems.* Englewood Cliffs, NJ: Prentice Hall Inc., 1984
- Schram, Peter (editor). *The National Electric Code Handbook (Third Edition).* Quincy, MA: National Fire Protection Association

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

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

Chapter 1. Introduction

Chapter Objectives

The information in this chapter will enable you to:

- Understand the product's basic functions & features
- Understand basic motion control concepts and apply them to your application

Product Description

The JSI is a digital servo controller that utilizes a digital PID control algorithm. The versatility of the controller allows for different configurations. The output utilizes a 12-bit DAC and can be selected to be used either as a voltage command output for use with traditional servo amplifiers, or as a current command output to directly drive hydraulic servo valves. The position feedback information may be provided by incremental encoders, absolute encoders, acoustic transducers, or from an analog voltage signal.

The programs that control motor motion use the Compumotor standard motion control command language and are downloaded over a standard RS-232C interface. There may be up to 100 different motion programs stored in non-volatile memory. Ninety nine of these programs may be recalled from external BCD thumbwheel switches, programmable controller, or over the RS-232C port from a computer. The JSI motion programs are capable of complex motion control capabilities as well as interactive machine control using the 13 user-defined, programmable inputs and 8 programmable outputs.

The JSI is a fully self-contained package with built-in power supplies. All that is required is an input power source (VAC or VDC). The controller is easy to install, program, and operate.

Product Features

The JSI offers the following features:

- Sixteen-bit, 68000 microprocessor control
- Voltage/current selectable output for use with servo amplifiers, or for driving hydraulic servo valves
- Twelve-bit DAC output resolution
- Incremental and absolute encoder feedback is standard
- Analog input voltage command capability
- Servo loop update time 0.5 or 1.0 ms
- Non-volatile, battery-backed-up RAM
- Easy-to-use, push-button adjustment of PID servo parameters
- Two-digit LED to display diagnostic faults
- RS-232C command interface with selectable baud rate
- Utilizes the X-Series programming language
- Daisy-chaining up to 16 controllers is possible
- 8K of non-volatile memory
- Up to 100 motion programs may be stored in non-volatile memory
- Motion program execution may be initiated by external BCD switches, a programmable controller, or a computer
- Fully isolated dedicated inputs: Home and End-of-Travel Limits
- Thirteen user-defined, programmable inputs

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

- Eight user-defined, programmable outputs
- Fully self-contained package with built-in power supplies
- Simple installation, programming, and operation procedures
- Distance, velocity, sequence, and loop counts can be entered from parallel input
- Subroutine operations can be performed with this unit

Theory of Operation

The key operational procedures involved in running a closed- loop servo system with a JSI controller are described below.

- The JSI controller produces an analog signal (current or voltage) to control the analog servo control system (i.e., analog servo drive or servo valve).
- The JSI receives an encoder signal (incremental or absolute) to close the motor's position loop. This capability allows the analog servo system to accurately position the motor.
- You can operate the JSI from a host computer using RS-232C communications, pre-program the JSI for standalone operation, or run the unit with a programmable logic controller (PLC).
- The servo control (DC drive or hydraulic servo valve) must close the system's velocity loop.

Figure 1-1 is an example of a typical closed-loop servo system.

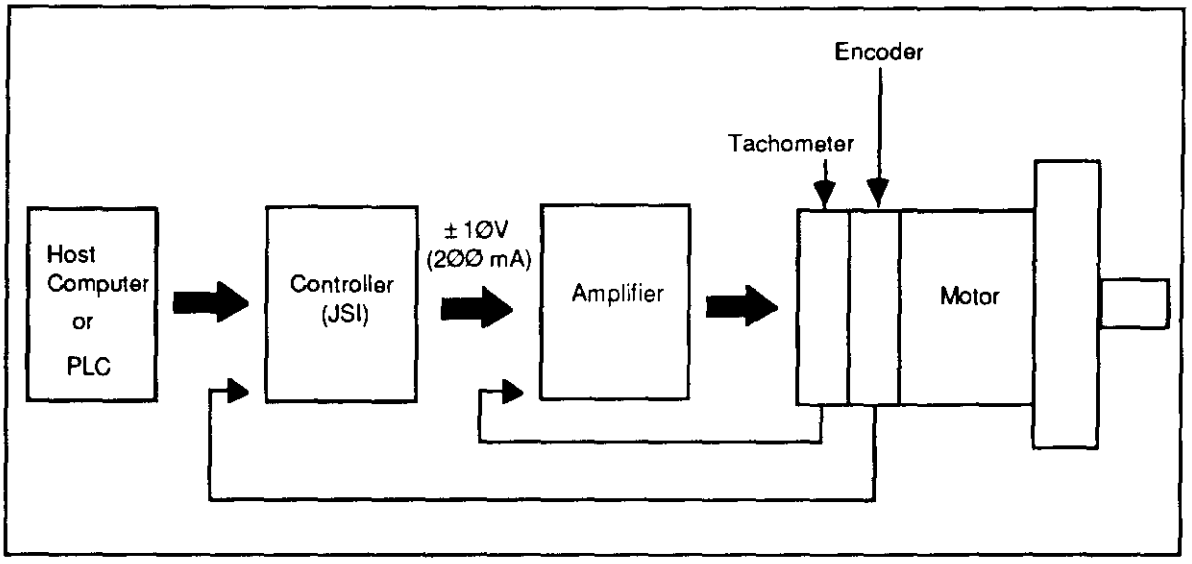


Figure 1-1. Typical Closed-Loop Servo System

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

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
- Ensure that each component functions properly
- Configure the system properly

What You Should Have

You should inspect your JSI shipment upon receipt for obvious damage to its shipping container. Report any such damage to the shipping company as soon as possible. Parker Compumotor cannot be held responsible for damage incurred in shipment. Carefully unpack and inspect your JSI shipment. In addition to the JSI controller, the items listed in Table 2-1 should be present and in good condition.

Ship Kit Table

Description	Quantity	Part Number
AC Cable Assembly	1	71-009039-01
JSI User Guide	1	88-009040-01

Table 2-1. Ship Kit Table

Basic System Configuration

This section provides step-by-step instructions to complete the basic configuration of your JSI system. Parker Compumotor recommends that you configure your system components in the following order. *(Remember, this is a temporary configuration. The permanent installation will be performed in Chapter 3, Installation).*

1. Refer to the manual that accompanied your drive and follow the instructions specified to set the drive's functions and attach it to the motor. Be sure to test the drive and the motor to ensure that they are operating properly.
2. Follow the instructions provided in this manual to ensure that the JSI servo controller is operating properly.
   a. You will establish RS-232C communications and ensure that the communication function operates properly. If you are using an IBM or IBM-compatible computer, you will have to convert the computer to a terminal to perform communications with the JSI. If you are using a terminal, you will have set the terminal's parameter's to match the JSI controller's settings.
   b. You will need an oscilloscope or a multimeter to check voltage and current levels.
3. Follow the instructions provided in this manual to connect your encoder to the JSI. Refer to the encoder's user guide for procedures to set the encoder's functions. Be sure to test the encoder to ensure that it is working properly.
4. Attach the drive to the JSI. The entire system is now fully configured. You should try to operate all of the components to ensure that they function properly in this basic configuration. Try to simulate the moves you intend to perform when you install your system permanently.

Figure 2-1 is an illustration of the JSI controller's front panel.

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

JSI Front Panel Diagram

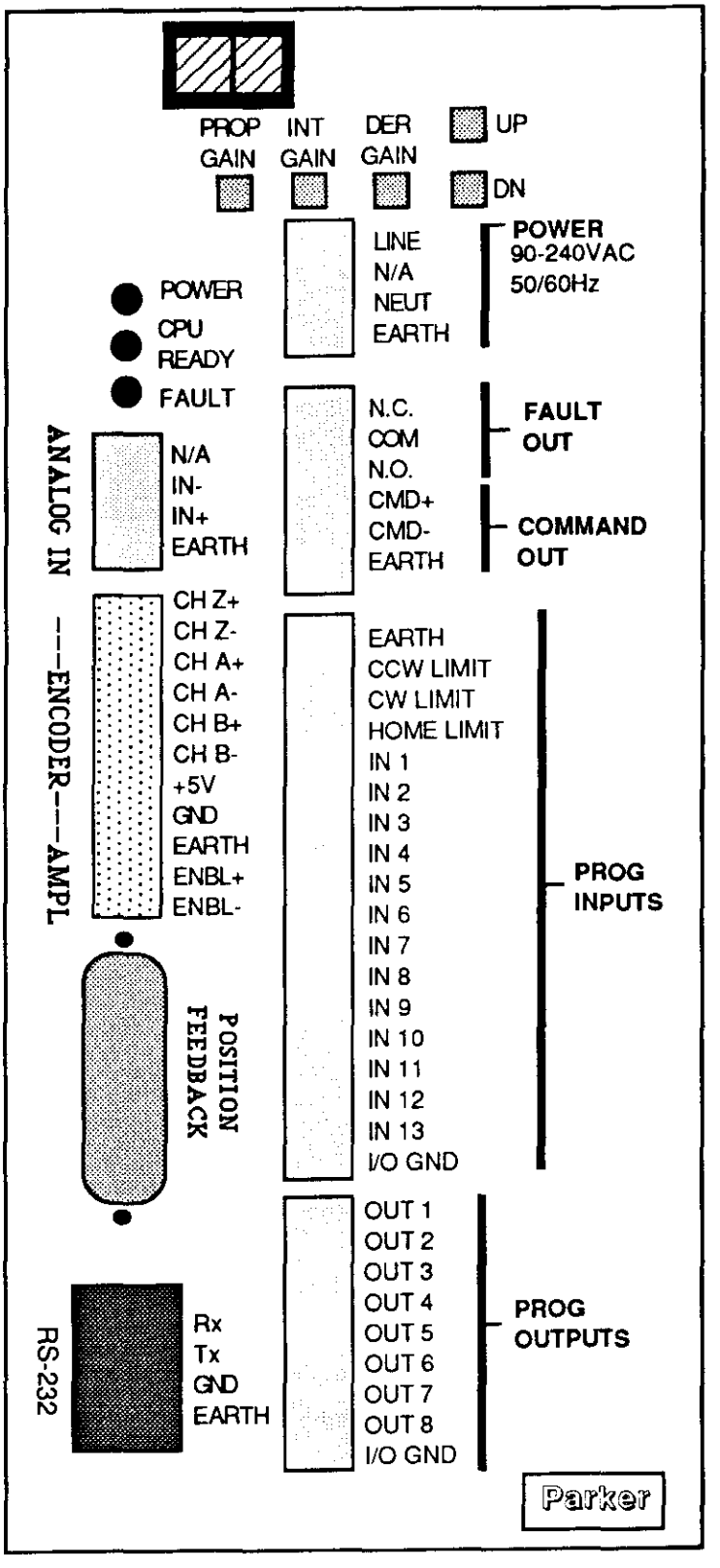


Figure 2-1. JSI Front Panel Diagram

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

Setting Controller Functions

This section discusses the various functions that must be properly set before you can operate the JSI controller.

Enable Input

Two enable input pins (ENBL+ and ENBL-) are located on the 11-pin screw terminal connector labeled ENCODER on the JSI's front panel. The outputs (CMD+ and CMD-) command will be disabled if you do not short-circuit this input. The input is optically isolated and single ended. To disable the command output, apply 5V to this input, or open the inputs between the **ENBL+** and **ENBL-** inputs.

Applying Power to the System

Once you have properly set the functions described above, you should prepare to wire and apply power to the JSI. You must apply power to test the JSI functions and features that are presented throughout the remainder of this chapter. You may power the JSI controller with AC or DC power. Figures 2-2 and 2-3 illustrate the proper power wiring connections for AC and DC power respectively.

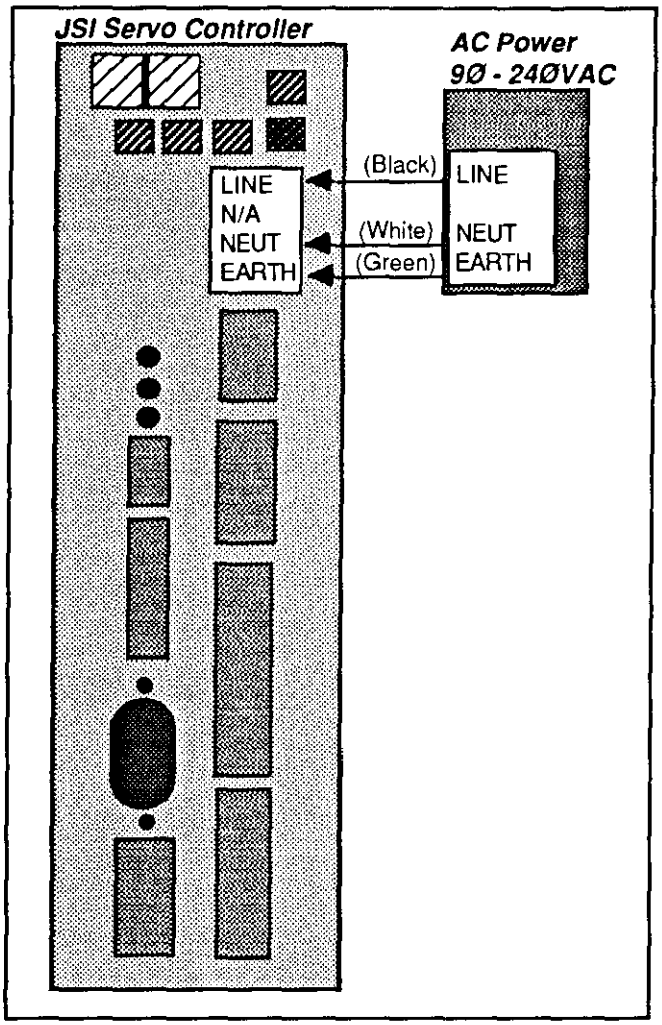


Figure 2-2. JSI Controller AC Power Wiring Diagram

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

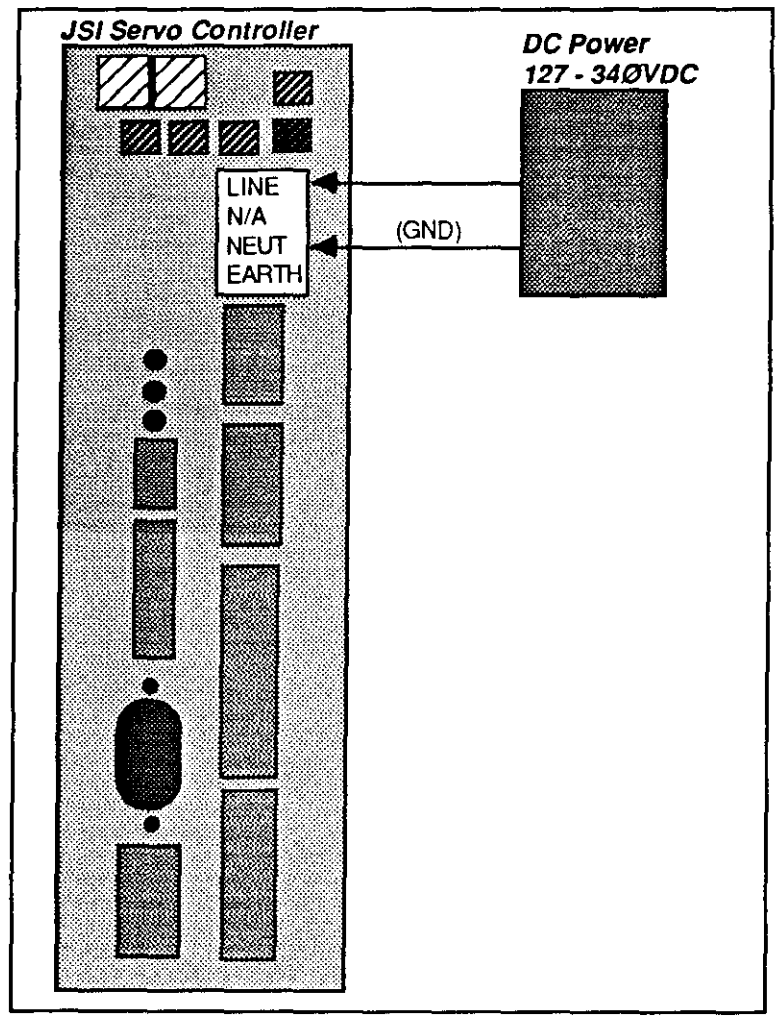


Figure 2-3. JSI Controller DC Power Wiring Diagram

Setting the Device Address

The default address setting for the JSI controller is **01**. To confirm that the JSI's device address is set to 1, push the PROP GAIN and DER GAIN button at the same time. The digits **01** will be displayed on the two-digit LED in the upper left corner of the JSI controller. Refer to Figure 2-4.

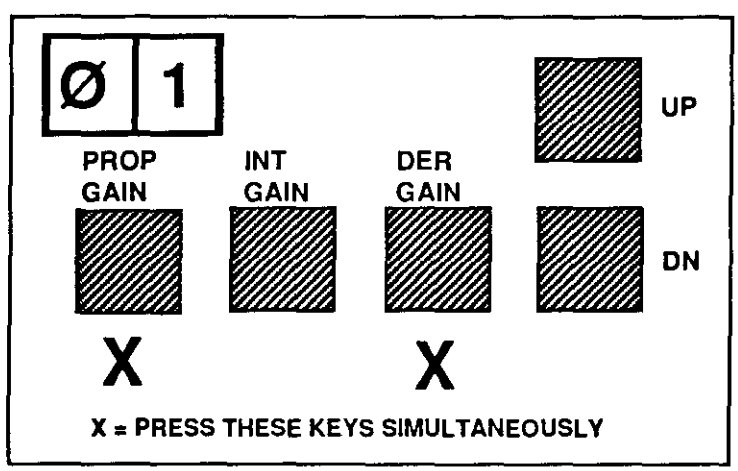


Figure 2-4. Confirm Device Address Setting

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

Changing the Address Setting

The LED display shown in Figure 2-5 verifies that the controller is set to **01**. If these digits (**01**) do not appear, use the following steps to set the device address to **01**.

1. Press and hold the PROP GAIN and DER GAIN buttons.
2. While holding the two buttons, push the DN button until the LED displays **01**. Refer to Figure 2-4.

If you need to increase the address setting, push the UP button instead of the DN button. You will need to modify the device address settings if you use several JSI units in your application. Each JSI controller must have a unique device address.

For testing purposes, the JSI's device address should always be set to ***01****.*

Establish Communications

To communicate with the JSI, your computer or terminal must have an RS-232C serial port. If your computer or terminal does not have a serial port, you can purchase one from your local computer dealer. The JSI uses a simple three-wire implementation of serial communication. The JSI uses RECEIVE DATA (Rx), TRANSMIT DATA (Tx), and GROUND(GND) signals on its auxiliary connector (as shown in Figure 2-5). *The ground (GND) connections on the auxiliary connector are signal ground or common as opposed to earth ground (EARTH).*

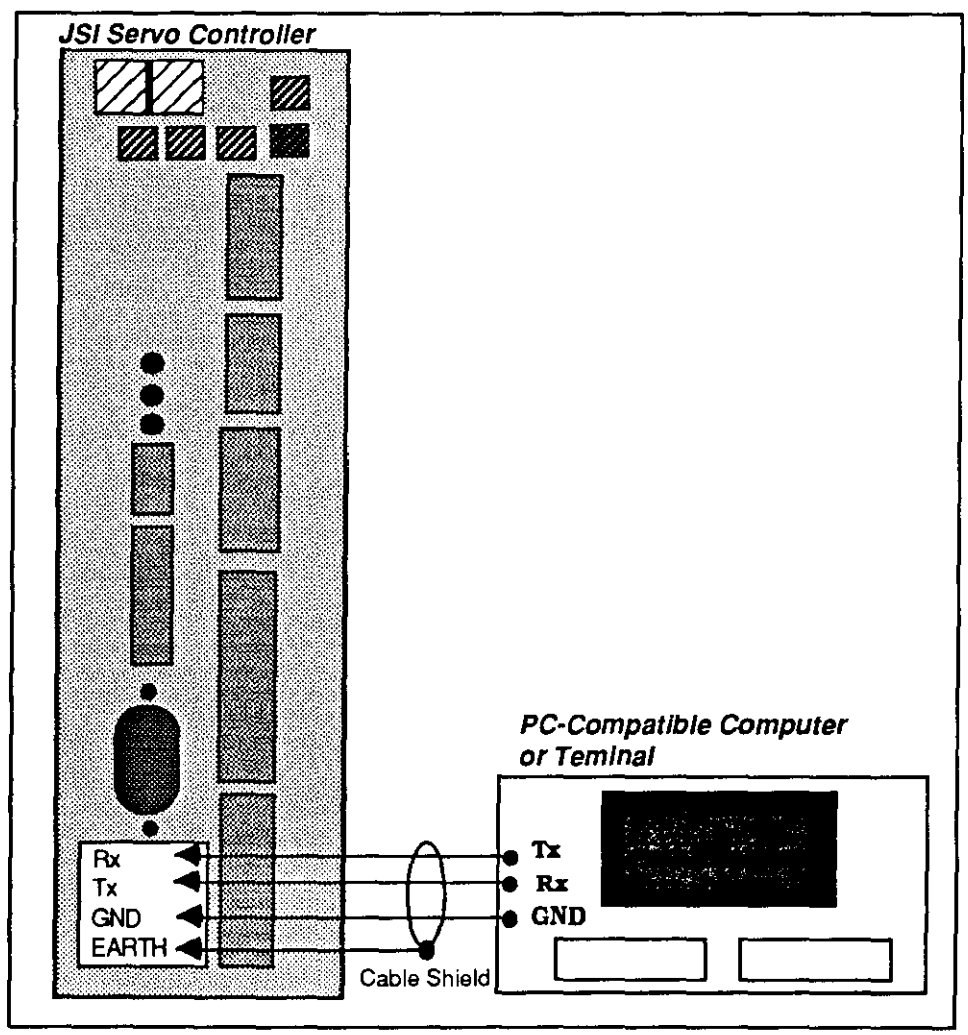


Figure 2-5. Wiring RS-232C Interface to a Computer or a Terminal.

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

You can change the baud rate communication parameter on the JSI. The other communication settings are fixed. The variable baud rates and fixed parameters are listed below.

- Baud Rates: 9,600 (default setting), 4,800, 2,400, 1,200, 600, 300
- Data Bits: 8
- Parity: None
- Stop Bits: 1
- Echo: Off (Full Duplex)

Selecting the Baud Rate Setting

You can use the pushbuttons and two-digit LED to display and change the baud rate. To display the current baud rate setting (*the first two digits of the baud rate are shown*), push the PROP GAIN and INT GAIN buttons at the same time. The default setting is 9,600. Refer to Figure 2-6.

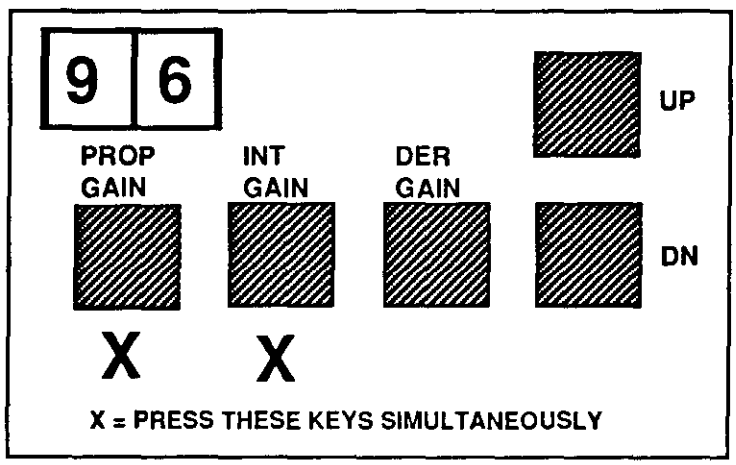


Figure 2-6. Display Baud Rate Setting

Changing the Baud Rate

The LED display shown in Figure 2-6 verifies that the baud rate is set to 9,600. If you want to change the current baud rate setting, use the following steps to set the baud rate.

1. Press and hold the PROP GAIN and INT GAIN buttons.
2. While holding the two buttons, push the UP or DN button until the LED displays the baud rate setting that you desire.

The baud rate setting is automatically saved to the JSI's non-volatile memory. Therefore, the last LED setting that is displayed before you release the PROP GAIN and INT GAIN buttons will be the baud rate that is implemented and saved until you change it again.

Once you have accurately set the baud rate and address settings, you can begin to test the RS-232C interface, which controls the JSI.

If you are using a terminal, you can start testing the RS-232C link by completing the steps outlined below (after you have properly set all of the parameters discussed above). If you are using an IBM or IBM-compatible computer, you will have to convert the computer into a terminal to prepare it for RS-232C communications. Several software vendors offer programs that you can use to convert your computer into a terminal.

- X-Talk
- PC-Talk
- ProCom

You can also refer to the sample programs provided inChapter 4, *Application Design*, of this manual. These programs can help you set up your computer to send commands to the JSI controller.

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

Verify RS-232C Communications

After you enable your computer to serve as a terminal, ensure that you have properly set your communications parameters.

- Baud Rates: 9,600 (the default setting), 4,800, 2,400, 1,200, 600, 300
- Data Bits: 8
- Parity: None
- Stop Bits: 1
- Echo: Off (Full Duplex)

To test your RS-232C communications link and ensure that it is operating properly, follow the steps outlined below.

Step 1 Be sure that the Command outputs (**CMD+** and **CMD-**) are ***not connected*** to the drive.

Step 2 If your connections are correct, and the device address is set to **01**, you should see the following message on your screen when you apply power to your unit.

```
* READY
>
```

*If the device address is not set to **01**, you will not receive this message.*

Step 3 Issue the command shown below. The proper response is also shown below.

Command	Response
> 1PR	*+ØØØØØØØØØØ

The Position Report (**PR**) command is a status request command (the **1** refers the the JSI controller's device address). The response given is the current encoder position. This response confirms that you have proper two-way communication between your computer and the controller. If you do not receive a response, return to the *Setting Controller Functions* section earlier in this chapter. Check your connections, and try to issue the **1PR** command again. If the problem persists, refer to Chapter 7, *Maintenance & Troubleshooting*.

Testing Output Levels

Now that you have tested the JSI's RS-232C communication link, you should test the functionality of the voltage output of the command outputs (**CMD+** and **CMD-**). These outputs will be used to move a DC motor. *Do not connect the **CMD+** and **CMD-** outputs to the drive until you are instructed to do so in these procedures. You will need these outputs to attach a voltmeter and perform voltage and current output testing.*

JSI LEDs

When an AC or DC power source is properly wired to the JSI, the green Power LED and the green CPU Ready LED will illuminate when you apply power to the unit. If you do not have the **ENBL+** and **ENBL-** inputs jumpered, the red Fault LED will also illuminate when you apply power to the JSI (the LED will display the digits **23**). If the Fault LED is lit and **23** is flashing, jumper the **ENBL+** and **ENBL-**inputs. Error code **16** will flash in the LED display. Error **16** indicates that the JSI **COMMAND OUTPUT** is disabled. Cycle power to the JSI. You can also use the **ON** command to remove error code **16** from the Fault LED). Only the Power and CPU Ready LEDs will remain on after you cycle the power.

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

Testing Voltage Output

If the LED display presents any other fault/error conditions, refer to Chapter 7. If no fault/error conditions exist, check the JSI's voltage output. *Do not connect the* **CMD+** *and* **CMD-** *outputs to the drive at this time. You will need these outputs to attach a voltmeter and perform voltage testing.* Refer to Figure 2-7 for an illustration of the proper voltmeter connections—the **CMD+** and **CMD-** outputs).

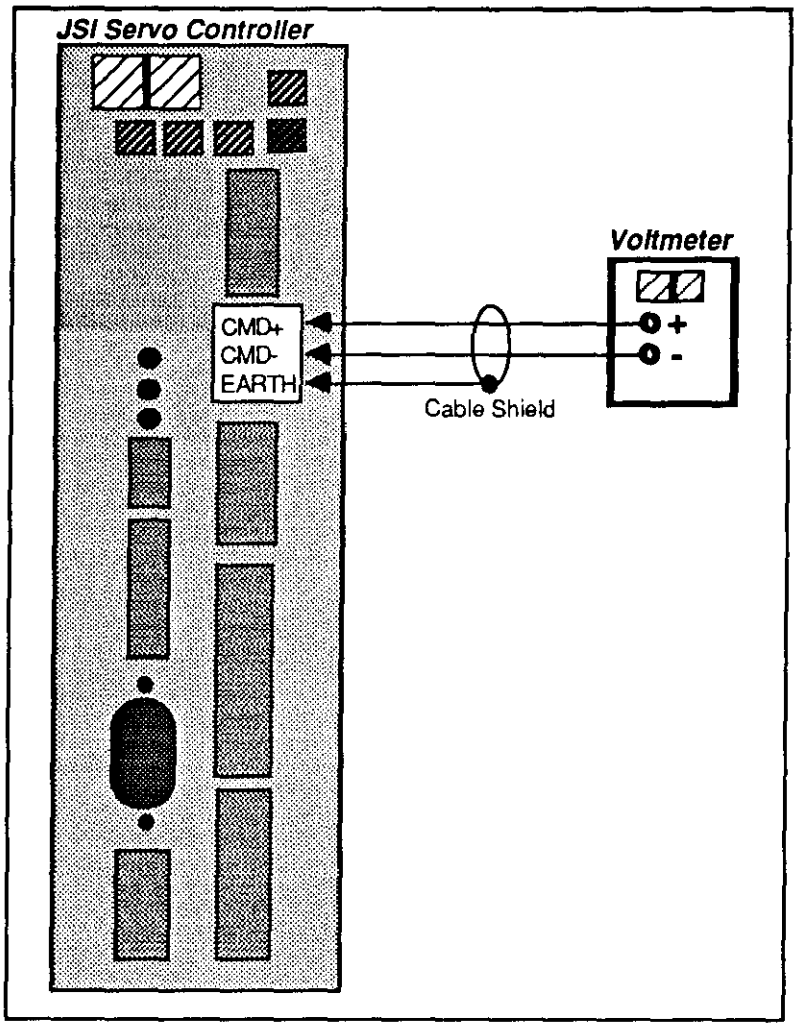


Figure 2-7. JSI/Voltmeter Wiring Diagram

Step 1 Use a voltmeter to measure the DC voltage between the **CMD+** and **CMD-** outputs The JSI controller should provide approximately ØV. The voltage level will appear on the voltmeter's LED.

Step 2 Enter the following commands:

Command	Description
> LD3	Disables CW and CCW limits
> CPE1ØØØØØØØØ	Defines the allowable position error
> A1	Sets acceleration to 1 rps^2
> V5	Sets velocity to 5 rps
> MC	Sets unit to Continuous mode
> G	Executes the move (Go)

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

Step 3 Measure the DC voltage between the **CMD+** and **CMD-** outputs. These outputs should generate approximately 7 to 8VDC. If the voltmeter does not indicate any voltage change from the previous test, check the wiring and try the test again. If the problem persists and you are unable to get the expected voltage reading (7 to 8VDC), refer to Chapter 7, *Maintenance & Troubleshooting.*

Step 4 Enter the following command:

Command	Description
`> 1DOI`	Instructs the system to report the voltage level continuously until you press any key. The controller should generate approximately 8.5VDC.

Step 5

Command	Description
`> K`	Kills motor motion. The JSI controller will generate approximately ØVDC.

The successful completion of this test confirms that the proper output voltages are being generated from the JSI controller's **CMD+** and **CMD-** outputs.

Testing Current Output

You should now prepare to test the JSI controller current output levels. *(proceed with this test only if no error conditions are present).* The JSI is configured at the factory to serve as a voltage output controller. Therefore, to perform this test, you will have to reconfigure the JSI as a current output controller. *Do not connect the **CMD+** and **CMD-** outputs to the drive at this time. You will need these outputs to attach a voltmeter and perform current output testing.*

Step 1 Remove power to the JSI controller.

Step 2 Connect a 20Ω, two-watt resistor from the **CMD+** and **CMD-** outputs.

Step 3 Apply power to the JSI controller

Step 4 Enter the following command:

Command	Description
`> COCH`	Configures the JSI controller as a current output controller. The JSI can generate between 0 and 200 mA of current.

Step 5

Command	Description
`> LD3`	Disables CW and CCW limits
`> CPE1ØØØØØØØØ`	Defines the allowable position error
`> A1`	Sets acceleration to 1 rps^2
`> V5`	Sets velocity to 5 rps
`> MC`	Sets unit to Continuous mode
`> G`	Executes the move (Go)

Step 6 Measure the DC voltage across the 20Ω resistor. The JSI controller should generate approximately 3.2VDC.

Step 7 Enter the following command:

Command	Description
`> 1DOI`	Instructs the system to report the output current level continuously until you press any key. The controller should generate approximately 160mA.

Step 8

Command	Description
`> K`	Kills motor motion. The JSI controller will generate approximately ØVDC.

Step 9

Command	Description
`> COV`	Reconfigures the JSI as a voltage output controller.

The successful completion of this test confirms that the proper current output levels are being generated from the JSI controller's **CMD+** and **CMD-** outputs.

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

Chapter 3. Installation

Chapter Objectives

The information in this chapter will enable you to:

- Mount all system components properly
- Connect all electrical system inputs and outputs
- Ensure that the complete system is installed
- Perform basic system operations

Environmental Considerations

Parker Compumotor recommends that you operate and store your JSI servo controller under the following conditions.

- Operating Temperature: 32° to 104°F (0° to 40°C)
- Storage Temperature: -22° to 185°F (-30° to 85°C)
- Humidity: 0 to 95% non-condensing

Complete System Configuration

In this section, you will go through a complete setup procedure to set the controller's functions, and mount it with the other components in your system.

Setting Drive and Motor Functions

Before you attempt to configure the JSI controller, you must configure and test the motor and drive. Refer to the user guides that accompanied your motor and drive. Follow the instructions in these manuals to wire and attach these units. Be sure that the motor and drive are operating properly before you set the JSI controller's functions and attach the drive to the controller. Run the motor with a variable supply to ensure proper installation.

Enabling the Controller

Before you can install the controller and begin programming the system, you must enable the controller by putting a jumper (link) between the enable inputs (**ENBL+** and **ENBL+**). If you apply power to the JSI without closing the enable inputs, the Fault LED will illuminate and flash error code **23.** The enable inputs allow the JSI to output voltage or current from the command output (**CMD+** and **CMD-**).

Setting Controller Functions

The following controller functions must be properly set before you can proceed with the permanent configuration and installation of the JSI system.

- JSI Device Address
- RS-232C Communications

JSI Device Address

If you are using more than 1 JSI servo controller on a machine, you should set up different addresses for each controller so you can distinguish them when you are programming. As many as 16 JSI units can be connected on a daisy chain configuration. Refer to Chapter 2, *Getting Started*, for detailed instructions on displaying and changing the JSI controller's device address.

Communication

You can change the baud rate communication parameter on the JSI. The other communication settings must be from your computer/terminal. The variable baud rates and fixed parameters are listed below.

- Baud Rates: 9,600 (the default setting), 4,800, 2,400, 1,200, 600, 300
- Data Bits: 8
- Parity: None
- Stop Bits: 1
- Echo: Off (Full Duplex)

Refer to Chapter 2, *Getting Started*, for detailed instructions on displaying and changing the JSI controller's baud rate setting.

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

Encoder Compatibility

The JSI is a position servo controller and thus requires position information or position feedback. The position command is generated internally. You define the position command with a motion profile. The transducers for position feedback that the JSI is compatible with are listed below.

- An incremental encoder
- Compumotor's line of absolute encoders (ABS8, AR23 phase II, AL phase II, Photo-Trak, AR-C, and AL-C)

Incremental Encoder

To use the incremental encoder feedback as the position transducer, the 11-pin screw terminal on the front panel provides the hardware interface. The JSI requires the standard phase A+, phase A-, phase B+, phase B-, phase Z+ and Phase Z- inputs. You can use single-ended encoders by not connecting the encoder's A-, B-, and Z- outputs. The phase A and phase B inputs provide quadrature input signals. The JSI has an internal quadrature detection circuit. The phase Z inputs provide one pulse per revolution. You can use this input to find the final home position when you execute a Go Home (**GH**) command.

The resolution of the encoder is determined by the particular encoder that you choose. If the encoder that you choose provides 1,000 pulses per revolution (a 1,000-line encoder), it will have 4,000 distinct locations per revolution after going through the quadrature detection circuit in the JSI. The 1,000 lines per revolution is known as the pre-quadrature resolution. This means that the phase A input provides 1,000 pulses in one revolution. The phase B input also provides 1,000 pulses per revolution. These pulses are offset from each other by 90 degrees, as shown in Figure 3-1.

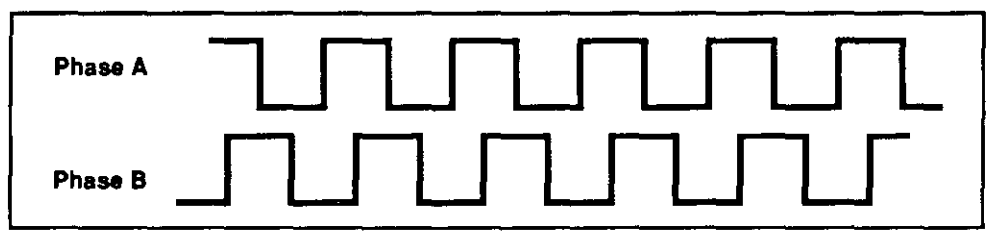


Figure 3-1. Quadrature Signals From an Incremental Encoder

The two phases go through a quadrature detection circuit. They provide four times the pre-quadrature resolution (1,000 pulses per revolution) or 4,000 counts per revolution. This is known as the post-quadrature resolution. The Z channel provides a single pulse every time the encoder passes its zero point. Therefore, the Z channel provides one pulse per revolution.

The maximum rate at which the phase A and phase B pulse can be detected determines the maximum speed that the motor can be achieve. This is limited by the bandwidth of the encoder interface circuit. The JSI has a bandwidth of 60 kHz pre-quadrature. With the incremental encoder in this example, 1,000 pulses per revolution are received. If the encoder is moving at a rate or 50 revolutions per second, the JSI will receive 1,000 X 50 pulses per revolution. This is less than the maximum bandwidth of the interface. A speed of 50 revolutions per second is more than adequate for most applications. If your application requires higher speeds, use a lower resolution encoder. The inputs to the JSI incremental encoder interface are optically isolated. The interface also provides a 5V output to power the incremental encoder.

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

Absolute Encoder

The JSI allows you to use encoders as position transducers (from Compumotor's line of absolute parallel encoders). On the front panel of the JSI, this encoder interface corresponds to the 15-pin D connector labeled **POSITION FEEDBACK**. The absolute encoder requires a separate decoder box for converting the encoder signals to the parallel data that is transmitted using the 15 pin D-connector interface. You can only use the ABS8, AR23 Phase II, AL Phase II, AR-C, AL-C, and Photo-Trak encoders with the JSI controller. You will receive this encoder box when you purchase one of Compumotor's absolute encoders. The encoder inputs are optically isolated. With absolute encoders, the maximum speed achievable is 4,000 rpm.

Encoder Resolution

The base resolution of the encoder is the number of distinct locations that are available in one revolution of the encoder. For incremental encoders, this base resolution is 4 times the *pre-quadrature* resolution. Thus, with a 1,000-line encoder you will have a resolution of 4,000 steps/revolution. With absolute parallel feedback encoders, the base resolution will vary depending on the encoder you choose. A typical resolution is 16,384 steps/revolution. The base resolution indicates that if the servo loop receives a position command of one revolution, it will command the motor or servo valve to move one revolution. The servo loop will expect to receive the number of counts or steps that correspond to one revolution from the position transducer. In the case of the absolute parallel encoder described above, the controller will command the motor to move 16,384 steps and will expect to receive 16,384 steps from the position transducer.

Setting Incremental Encoder Resolution

To perform preset moves with the JSI controller, you must use an incremental or absolute encoder (serves as a position feedback device that provides motor position information to the JSI). When you instruct the controller to make a move, the distance is measured in encoder steps. The encoder's resolution defines the number of pulses that the encoder generates per 1 revolution of movement.

Since there are many different encoders with different resolutions, you must specify for the JSI what type of encoder you have connected to the system. You can specify the various encoders and encoder resolutions to the JSI with the Configure Feedback Device (**CFB**) command

Command	Description
`> OFF`	Turn off JSI amplifier
`> CFB1,2000`	Tells JSI that the feedback device being used is an incremental encoder that has an post-quadrature resolution of 2,000 steps/rev (500 line).

To ensure that your incremental encoder is operating properly, follow the steps described in the *Making a Move with an Incremental Encoder and a DC Motor* section of Chapter 2, *Getting Started.*

Disabling the Limits

If you want to test the controller without connecting the CW and CCW limit switches, you will have to disable the limit inputs with the Limit Disable (**LD3**) command. If you attempt to perform a move without disabling the inputs or using the **LD3** command, the Fault LED will illuminate and flash error code **41** or **42.** These error codes indicate that the motor has reached the hardware limit.

The software limits are disabled when you first apply power to the system. Therefore, you can make the motor move upon power up if you disable the hardware limits (**LD3**). If you do not have any hardware limit switches connected, or if you want to provide more system security, you can enable the software limits (refer to the Software Limits (**SL**) command) and define the limits. The software limits are factory set to position 0. Therefore, you must define a range of the limit with the **SL** command to make the motor move.

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

Electrical System Connections

This section contains directions and procedures to complete all of the JSI's electrical connections. Refer to the basic system wiring diagram (Figure 2-1) in Chapter 2, *Getting Started*, for specific I/O locations.

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, Massachusetts.

In general, all components and enclosures must be connected to earth ground through a grounding electrode conductor to provide a low impedance path for ground fault or noise-induced currents. All earth ground connections must be continuous and permanent. Compumotor recommends a single-point grounding setup. Prepare components and mounting surfaces prior to installation so that good electrical contact is made between mounting surfaces of equipment and enclosure. Remove the paint from equipment surfaces where the ground contact will be bolted to a panel and use star washers to ensure solid bare metal contact.

For temporary installation, or when you cannot implement the grounding method described above, you must connect the GROUND terminal on the AC power connector to the earth ground.

Drive and Motor Wiring

Refer to the manuals provided with the drive and motor you are using with the JSI servo controller to properly wire the motor to the drive. Ensure that you have performed this wiring step successfully.

Analog Input

Figure 3-2 represents an analog input. This configuration is not generally used . This input controls the command output voltage/current. With the Configure Velocity Scale (**CVS**) command. You can vary the command output (**CMD+**, **CMD-**) based on an analog input. It is useful when you need to track or follow another device. (This feature is only applicable if you intend to control motion with respect to input voltage.) Refer to Figure 3-2 for typical wiring of the analog inputs.

Command Output

Command outputs (**CMD+** and **CMD-**) control the velocity of the drive by outputting either a voltage (±10V) or current (±200mA or ±60mA). You can drive a DC servo drive or a hydraulic cylinder by applying this out to the drive input. You may configure the JSI as either a voltage output (with the Configure Output as Voltage Source (**COV**) command) device or a current output (with the Configure Output as Current Source Low (**COCL**) or Configure Output as Current Source High (**COCH**) commands) device. The controller is factory set as a voltage output device. Typically, the DC drives are driven by ±10VDC. However, many hydraulic cylinders require current input. See Figure 3-2 for typical wiring of the Command outputs.

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

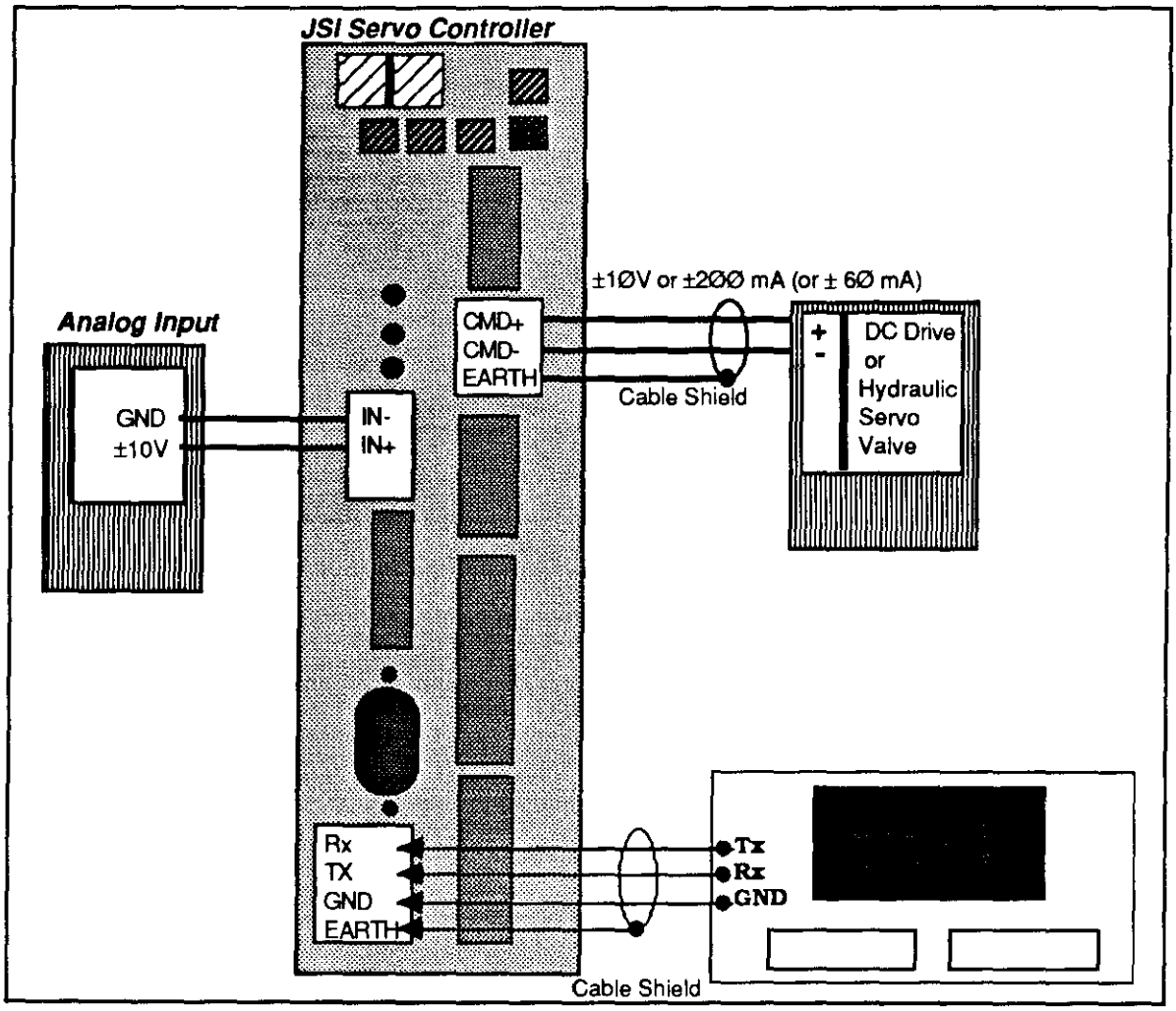


Figure 3-2. Analog Input and Command Output Connections

Limits

The **CW** and **CCW LIMITS** serve as emergency stops to prevent the motor from traveling beyond the operational travel limits. Refer to Figure 3-3 for a wiring diagram for these inputs.

These limit inputs are enabled. You must install or disable these limits to allow the motor to move. You can apply up to 30VDC to control the CW and CCW Limit inputs. You can also disable the hardware limit inputs with the RS-232C command (**LD3**).

Home Limit

The home limit input differs from the CW and CCW limit inputs and the **IN1** through **IN13** inputs. The active state of the home limit input is determined by the **OSC** command instead of the **INL** command. This input is useful if you want your machine to start its operation from a repeatable position. You can use this input in conjunction with the Go Home (**GH**) command or the Go Home input set up using the Input (**IN**) command. When the JSI executes a GH command, it scans the Home Limit input until the switch activates the Home Limit input. The Homing function is discussed in detail the *Homing the Motor* section of this chapter. Refer to Figure 3-3 for a wiring illustration of this limit switch.

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

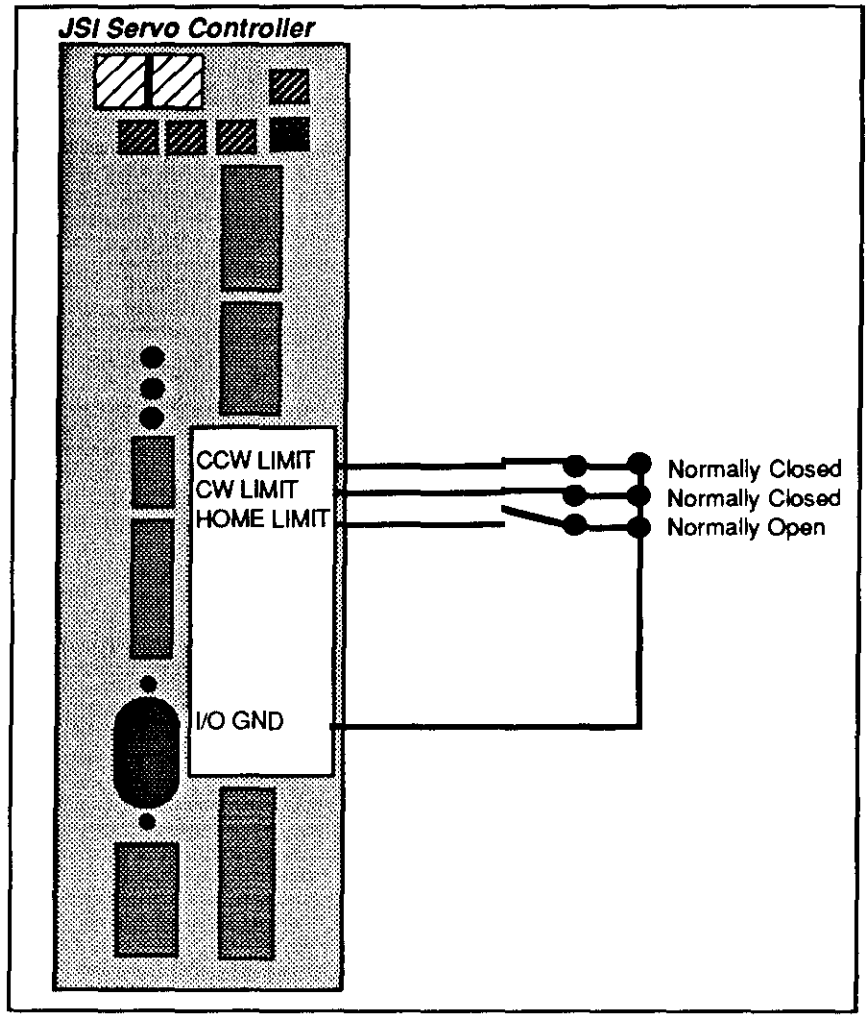


Figure 3-3. JSI Limit Switches

Encoders

You can use parallel output absolute encoders or incremental quadrature encoders as position feedback devices for the JSI controller. This section provides information to help to wire your encoder to the JSI controller.

Incremental Encoder

Figure 3-4 represents the proper wiring connections that you must make to operate the JSI and an incremental encoder.

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

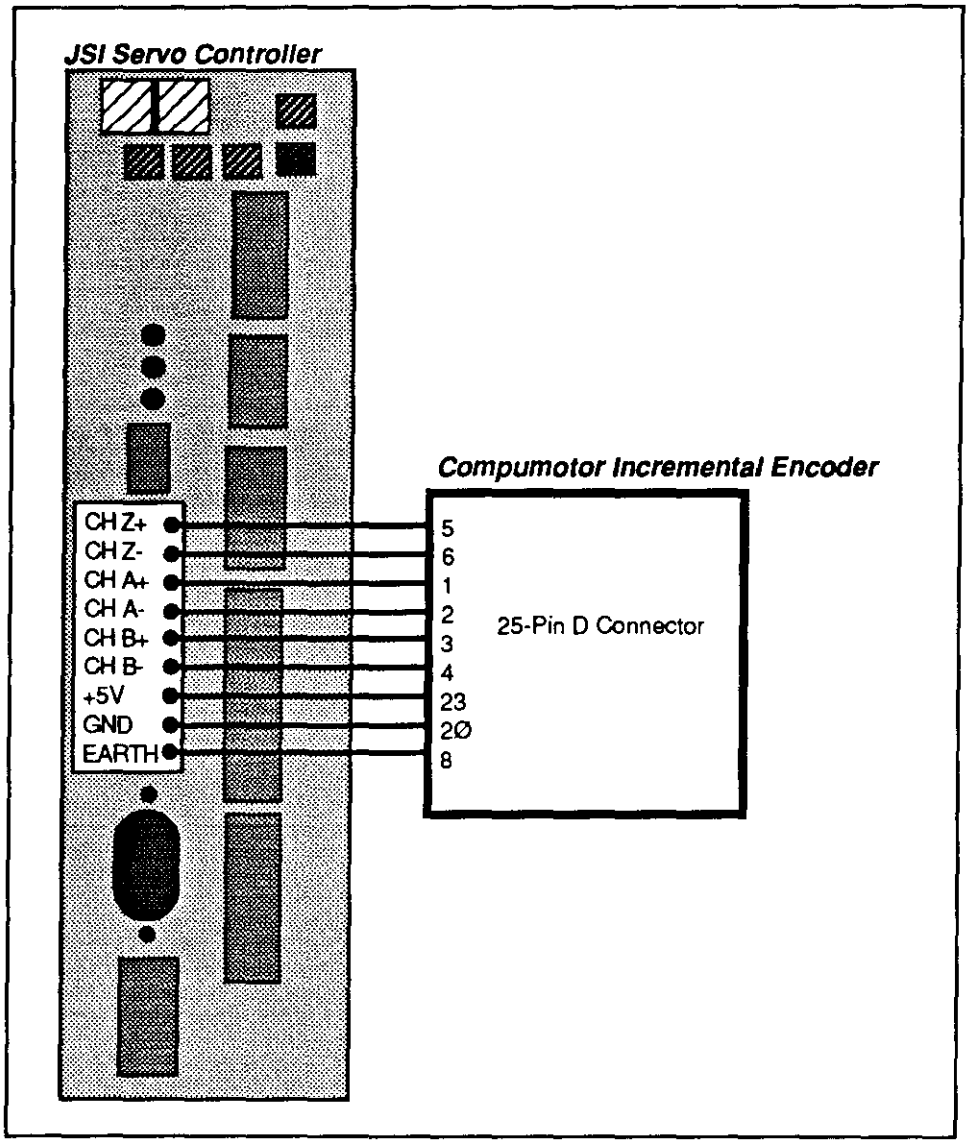


Figure 3-4. JSI/Incremental Encoder Connection

Pin #	57 Size	83 & 106 Size	Description
1	Brown	Yellow	Channel A
2	Brown/White	White/Yellow	Channel A Return
3	Green	Blue	Channel B
4	Green/White	White/Yellow	Channel B Return
5	Orange	Orange	Channel Z
6	Orange/White	White/Orange	Channel Z Return
7	———	---------	*Not connected*
8	Shield	Shield	Ground Case
9 - 13	———	---------	*Not connected*
14	Black	Black	Common
15 - 22	———	---------	*Not connected*
23	Red	Red	5VDC
24 - 25	———	———	*Not connected*

Table 3-1. Incremental Encoder Pin Connections

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

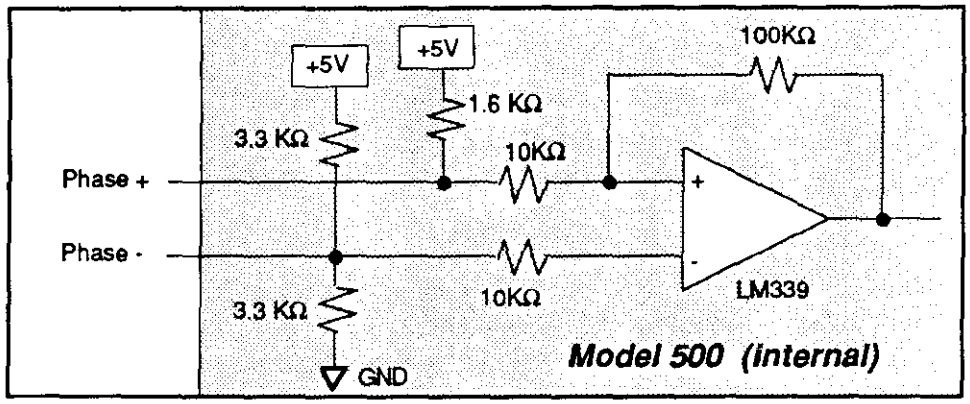


Figure 3-5. Typical JSI Encoder Input Circuit

If you use a single-ended encoder, do not connect the JSI's A-, B-, or Z- inputs. You must connect the encoder's ground connection to the JSI's incremental encoder GND.

Absolute Encoder

The absolute encoder behaves in the same manner as an incremental encoder as far as servo feedback. The difference is that absolute encoders will give the same unique position for each mechanical location in its absolute range. Refer to the absolute encoder's user guide to set the resolution for the encoder. Use this same resolution for the feedback resolution in the **CFB** command.

Figure 3-6 depicts the overall system configuration with an absolute encoder.

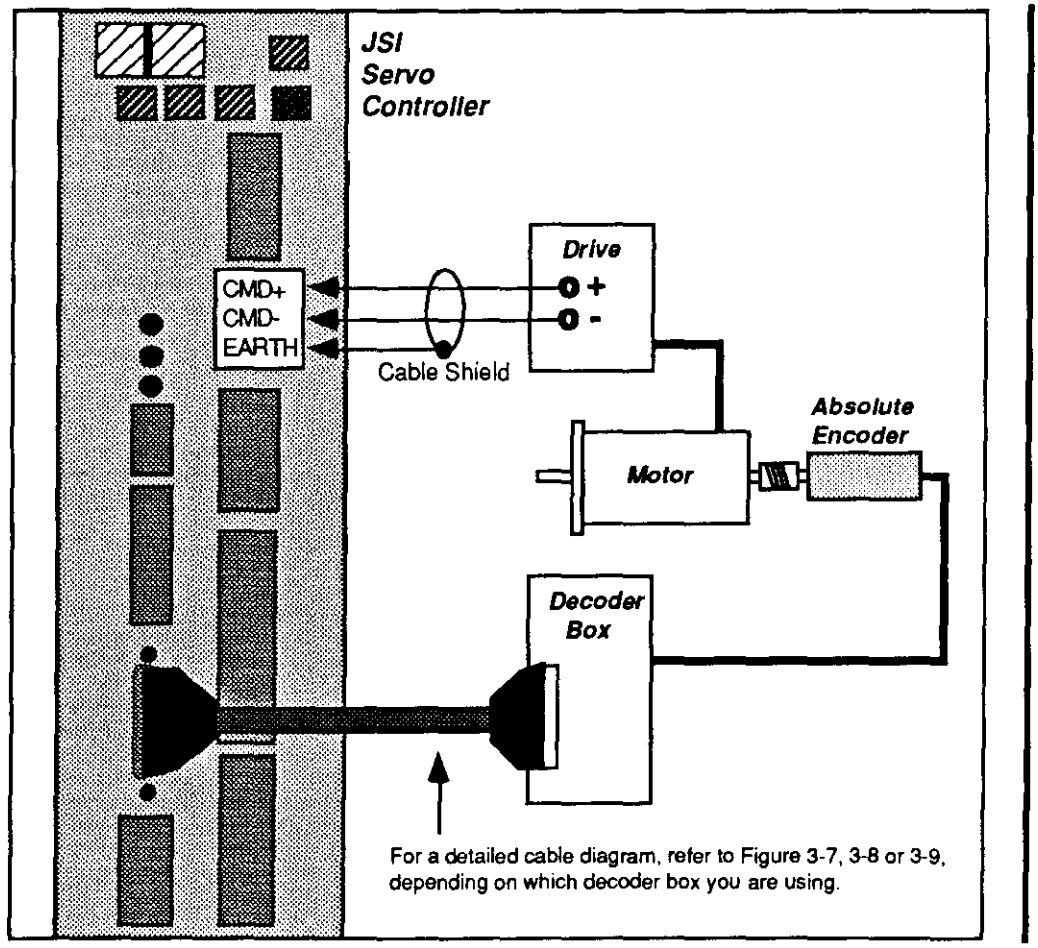


Figure 3-6. Typical JSI/Absolute Encoder Connection & JSI/Drive Connection

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

The JSI supports the ABS8 or -4 style decoder box and the -1 style decoder box with Compumotor's AR23 absolute encoders. It also supports the AR-C, AL-C and Photo-Trak absolute encoders. Use the **CFB** command to select one of these absolute devices for your feedback.

Command	Description
> **CFB1**	Selects an incremental encoder
> **CFB2**	Selects -4 style decoder box with the AR23 absolute encoder (Fig . 3-7)
> **CFB3**	Selects AR-C, AL-C, or Photo-Trak absolute encoders (Fig . 3-8)
> **CFB4**	Selects -1 style decoder box with the AR23 absolute encoder (Fig . 3-9)

To install the interconnect cables from an absolute encoder's decoder box to the JSI controller, complete the following steps:

1. Remove power from the motor/drive, JSI controller, and absolute encoder/decoder.
2. Fabricate and install a multi-conductor (at least 12) shielded cable from the absolute encoder/decoder to the absolute encoder input of the JSI controller (refer to Figure 3-7, 3-8, or 3-9, depending on which decoder box you are using).
3. Verify that the encoder cable is plugged into the decoder and that the JSI is connected to the motor/drive.

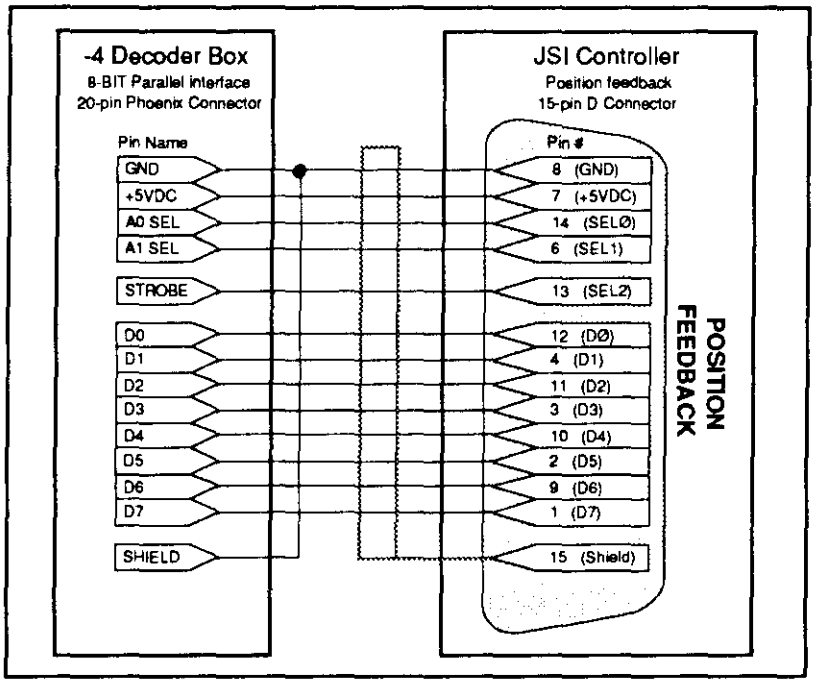


Figure 3-7. JSI-to-Decoder Box Cable (for use with -4 Decoder Box and AR23)

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

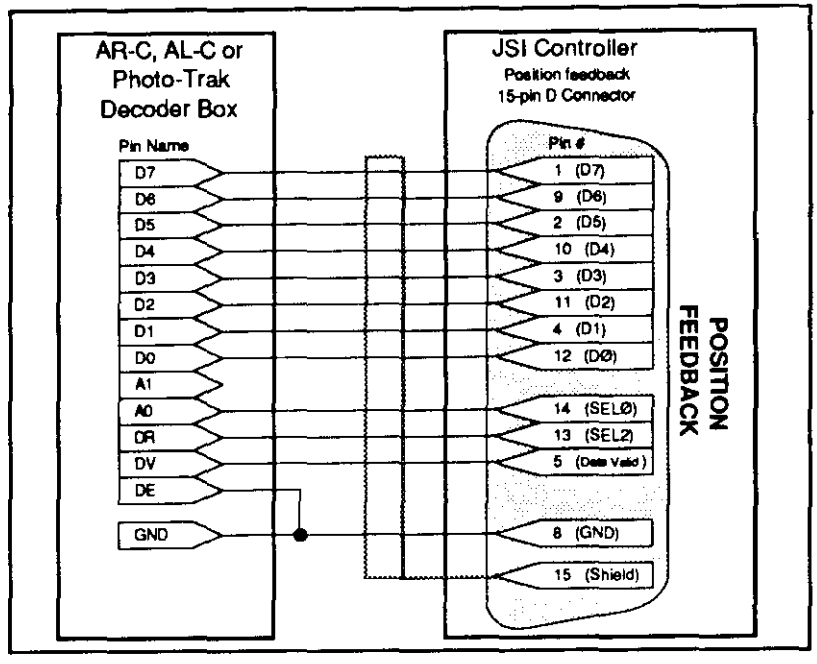


Figure 3-8. JSI-to-Decoder Box Cable (for use with AR-C, AL-C, or Photo-Trak)

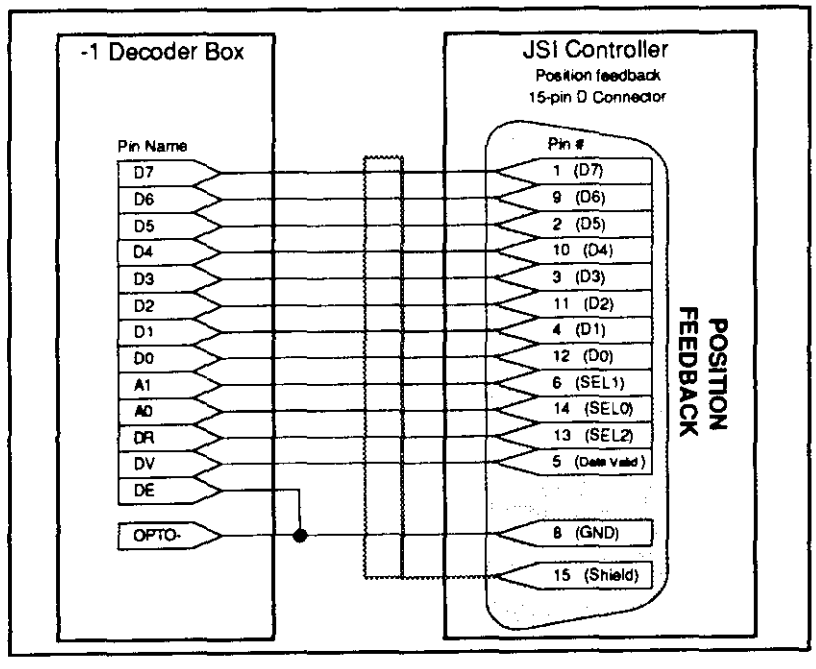


Figure 3-9. JSI-to-Decoder Box Cable (for use with -1 Decoder Box and AR23)

Power

The power input requirement is 90 - 240VAC or 127 - 340VDC at 1.2A root mean square (rms). If you are using a DC power input, place the positive input on the LINE connection and the negative input on the NEUT connection. Refer to Chapter 2, Getting Started, for specific information on wiring AC and DC power to the JSI controller.

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

Other Inputs and Outputs

Fault Outputs

The fault outputs are relay outputs. There are two types of outputs available: normally open and normally closed.

- Normally-Closed Outputs will **open** when a fault condition occurs
- Normally-Open Outputs will **close** when a fault condition occurs

Fault outputs will be activated when any of the following fault conditions occur:

16 — Amplifier Off
20 — Following Error Exceeded
23 — Drive Enable Not Activated
30 — Checksum Error
60 — Commanded the Drive to Shutdown (OFF Command)
66 — User Fault

These Fault outputs are located on the front panel of the JSI controller. Refer to Figure 2-1 for the exact location of these outputs. These relay fault outputs can handle up to 120VAC and 2A of current.

Programmable Inputs

The JSI controller pulls the voltage level of inputs **IN 1** - **IN 13** to 5VDC. If you are using a PLC, you can apply up to 30VDC to these inputs or connect them to the **I/O GND** to activate them. If you are using a relay to activate the inputs and outputs, use solid state relays to prevent electrical noise from impairing the performance of your application. You can program the programmable inputs to perform various functions (i.e., stop input, sequence-select input, etc.). You must use the Input (**IN**) command to change the function of a programmable input. You can also change the active level of the inputs with the Configure Input Level (**INL**) command. The default setting for these inputs is **INLØ** (Input Active Low). Refer to Figure 3-10.

Programmable Outputs

The JSI controller's programmable outputs, **OUT 1** - **OUT 8** are open collector outputs. You can run these outputs with 5 - 30VDC and sink up to 300 mA into them. You can interface these outputs with the PLC directly. If the PLC can directly interface with open-collector output. If not, you will have to use a pull-up resistor and an external power supply to use this outputs (refer to Figure 3-10). These outputs can be configured for different functions (i.e., moving/not moving) with the Configure Output (**OUT**) command. You can also configure the active level of the outputs with the Configure Output Level (**OUTL**) command. The default setting for these outputs is **OUTLØ** (Output Active Low).

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

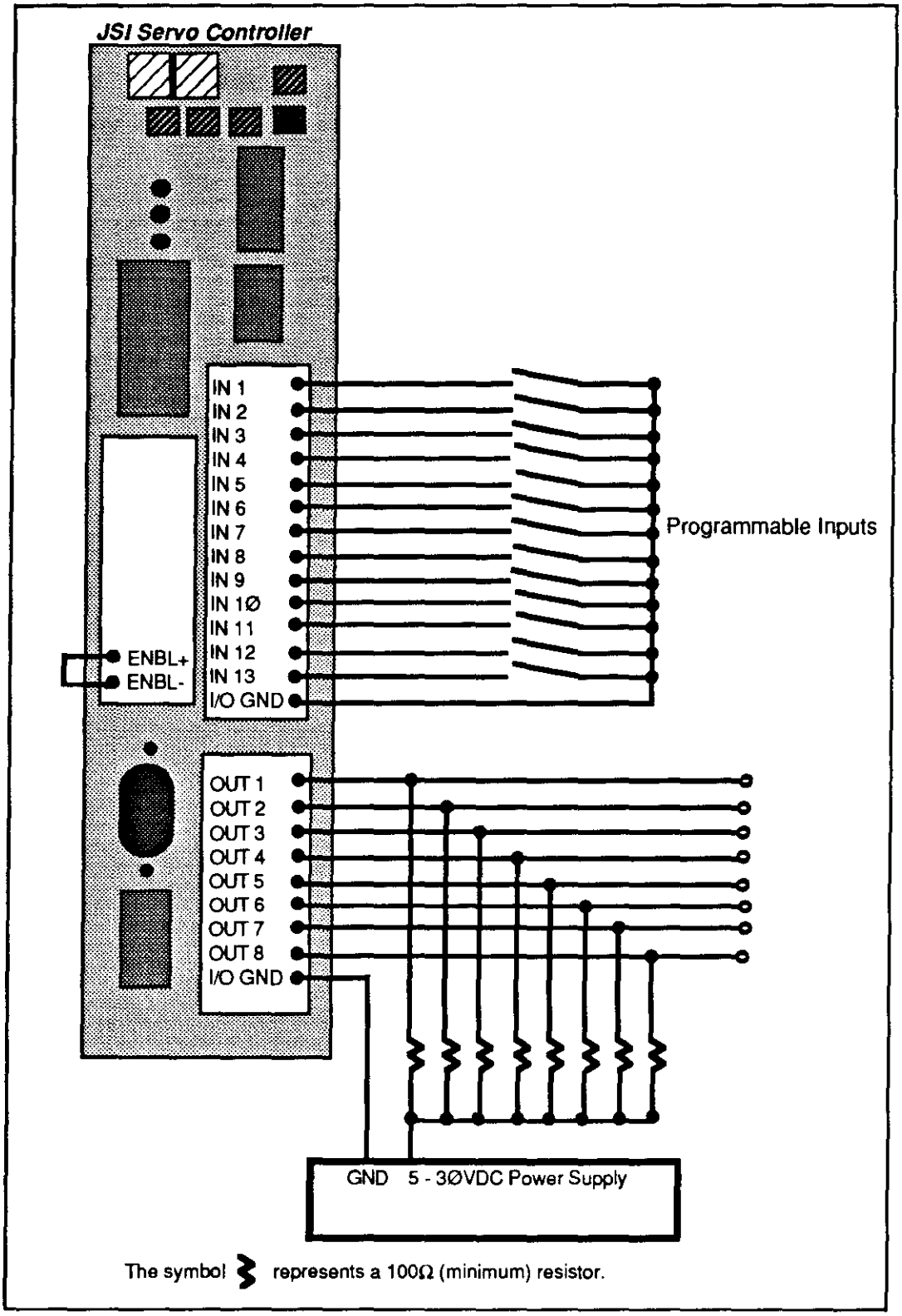


Figure 3-10. Programmable I/O Wiring Diagram

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

System Mounting

Proper mounting and panel layout are essential for trouble-free operation of the JSI. You should allow sufficient space for unrestricted air flow over the JSI to allow for efficient convection cooling of the unit.

Controller Mounting

If you mount the JSI in an enclosure, observe the following guidelines:

- Make sure there is at least 4 inches of unrestricted air-flow space around the JSI to allow for efficient convection cooling (refer to Figure 3-11). Fan cooling may be necessary if adequate air flow is not provided.
- Do not mount large, heat-producing equipment (such as a drive) directly beneath the JSI. The maximum allowable ambient temperature directly below the JSI is 40°C.

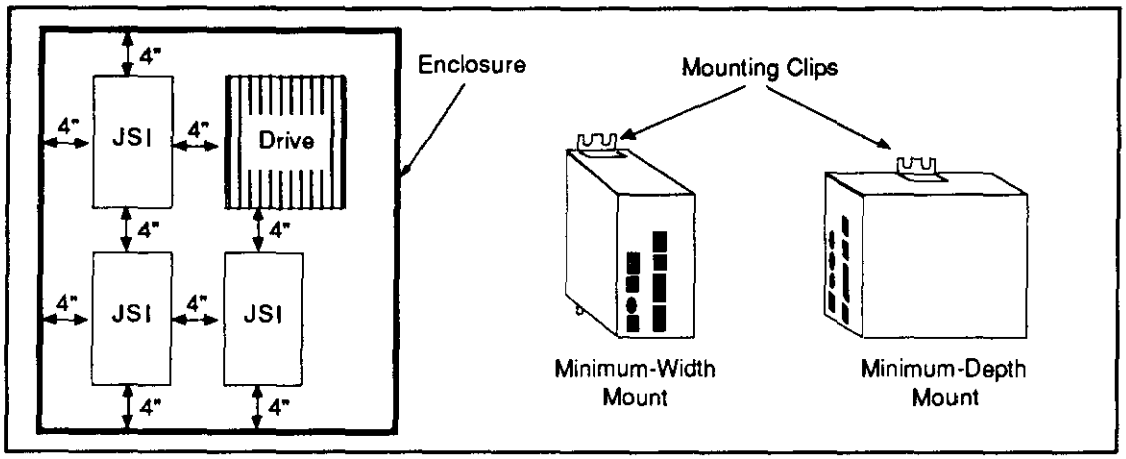


Figure 3-11. Panel Layout Guidelines and Mounting Options

Minimum Width or Depth

You can mount the JSI for minimum depth or width, depending on how you attach the mounting clips to the unit (see Figure 3-11 above).

The JSI is shipped with the two clips attached to the back of the indexer, opposite the power connectors. This allows the maximum amount of panel space possible.

For minimum depth, attach the two clips to the side of the indexer. This allows you to mount the drive in the shallowest possible enclosure.

Encoder Mounting

Refer to the operations manual that accompanied the encoder you are using. Follow the instructions provided to install the encoder to you JSI system. A discussion on encoder compatibility and performance is provided in Chapter 4, *Application Design*.

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

Verifying Proper Installation

Refer to Chapter 2, *Getting Started* (*Testing Output Levels* and *JSI LEDs* sections) for detailed instructions on verifying proper installation of your system. You should have completed these steps during the basic system configuration, but Compumotor recommends that you repeat these tests now to ensure that your system functions properly in its permanent environment.

RS-232C Communication

Refer to Chapter 2, *Getting Started* (*Verifying RS-232C Communications* section) for detailed instructions on verifying proper installation of your system. You should have completed these steps during the basic system configuration, but Compumotor recommends that you repeat these tests now to ensure that your system functions properly.

Making a Move with an Incremental Encoder and a DC Motor

Before you attempt to use the JSI controller to make a move with a DC motor, you must successfully complete the voltage or current tests. As stated at the outset of the user guide, these exercises and tests assume that you are using a DC motor. In this exercise, we are also assuming that you are running a DC servo drive that requires a ±10V control input. If you are running a drive that requires current input, you must use the Configure Current Output—Low (**COCL**) or Configure Current Output—High (**COCH**) commands to reconfigure the JSI controller as a current output controller. Figure 3-12 represents the components that you should have wired and mounted.

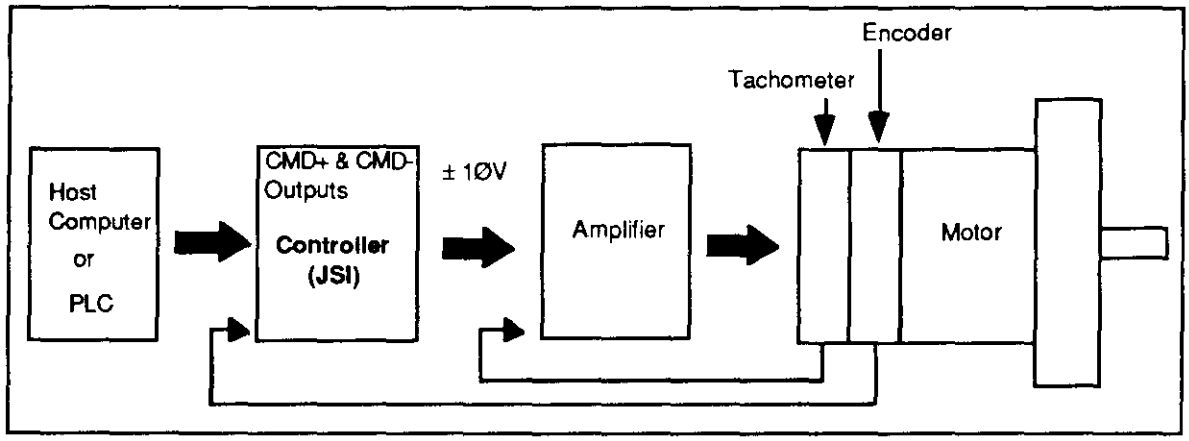


Figure 3-12. JSI System Configuration

After you have reconfigured the JSI, perform the following steps. ***For this exercise, do not attach the load to the motor.***

Step 1 Remove power from the JSI controller and the drive. (Make sure that the load is not attached to the motor.)

Step 2 Verify that the incremental encoder is installed properly as discussed in the beginning of this chapter. (Refer to Figure 3-4 for a detailed diagram of the JSI/Encoder wiring connection.)

Step 3 Apply power to the JSI controller only. Do not apply power to the drive.

Step 4 Type the following command to display the encoder position.

Command	Description
`> 1DPA1`	Display the actual position of the encoder

The response should read `*ØØØØØØØØØØ`. This indicates that the encoder is at absolute zero position.

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

Step 5 Grab the motor shaft and manually turn the shaft approximately 1/2 revolution in the CW direction. (If you have an encoder with a resolution greater than 4,000 steps/revolution, do not turn the motor more than 1 revolution because the JSI controller will fault out due to the following error.) If the encoder is wired properly, the actual encoder position report back will increase.

Step 6 See if the JSI controller read the encoder counts properly by entering the **1DPA1** command again.

Step 7 If the encoder counts did increase (a positive number), go to Step 11. If the encoder counts decreased, go to Step 8.

Step 8 Remove power from the JSI controller.

Step 9 Switch encoder wires A+ with B+. Switch encoder wires A- with B-, if they are wired to the JSI.

Step 10 Return to Step 3 and repeat these steps again. You should be able to complete them successfully now.

Step 11 Remove power from the JSI controller.

Step 12 Connect **CMD+** and **CMD-** outputs of the JSI controller to the DC Drive (**CMD+** and **CMD-** will output ±10V control signal to the drive).

Step 13 After you verify proper wiring between the drive and the JSI controller, power the JSI controller and the drive. (The motor should not move) If the motor moves and the JSI controller's fault LED is illuminated and error code **20** flashes , go to Step 14.

Step 14 Type the following command to attempt a preset move.

Command	Description
> OFF	Disables the drive using the shutdown output
> CFB1,4000	Configures the encoder feedback as an incremental encoder with post quadrature resolution of 4000 steps/revolution.
> ON	Enables the drive
> MN	Sets the system to normal mode
> A5	Sets acceleration to 5 rps^2
> V2	Sets velocity to 2 rps
> D4000	Sets distance to 4,000 steps
> LD3	Disables the CW and CCW limits
> G	Executes the 4,000-step move

Assuming the encoder resolution is 4,000 steps/revolution, the motor should move 1 revolution in the CW direction and stop. If the motor moved the proper distance in the CW direction, go to Step 18. If the motor moves in the wrong direction, or if the motor moves continuously , go to Step 15.

Step 15 Remove power from the JSI controller and the drive.

Step 16 Switch the **CMD+** and **CMD-** wires.

Step 17 Return to Step 13 and repeat these steps again. You should be able to complete them successfully now.

Step 18 Make the motor move in the opposite direction by issuing the following commands:

Command	Description
> H	Changes the direction of the move
> G	Executes 4,000-step move in CCW direction

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

Using the JSI with an Absolute Encoder

Before you attempt to use the JSI controller to make a move with a DC motor, you must successfully complete the voltage and current tests. As stated at the outset of the user guide, these exercises and tests assume that you are using a DC motor. In this exercise, we are also assuming that you are running a DC servo drive that requires a ±10V control input. If you are running a drive that requires current input, you must use the Configure Current Output—Low (**COCL**)and Configure Current Output—High (**COCH**) commands to reconfigure the JSI controller as a current output controller. Figure 3-11 represents the components that you should have wired and mounted.

This section explains how to test your absolute encoder/decoder and how to make a move with an absolute encoder and a DC motor. The following assumptions apply to this test and example:

- The JSI controller is functional (works properly).
- The drive is functional.
- The absolute encoder/decoder is a Compumotor product (ABS8 (-4), -1, AR-C, AL-C, or Photo-Trak).
- The encoder is properly attached to the motor.
- **The load is not coupled to the motor. The motor is secure.**
- The JSI controller, drive, and absolute encoder/decoder have not previously been connected.

Testing an Absolute Encoder/ Decoder

Use the following procedure to test the absolute encoder with the JSI.

Step 1 Remove power from the drive.

Step 2 Apply power to the JSI controller.

Step 3 Apply power to the absolute encoder/decoder.

Step 4 Measure the DC voltage at pins 1 and 2 of the phoenix connector of the decoder box. The voltmeter should provide a reading of approximately 5V.

Step 5 Issue a **1CFB2** command to the JSI controller to display the feedback for an absolute encoder that has a resolution of 16,384 steps per revolution.

Step 6 Send the **1CFB** command to the JSI controller to display the feedback configuration. You should be using an absolute encoder with a base resolution of 16,384 steps/inch. Issue the **1DPA** command to the JSI controller to continually display the absolute encoder's position. If your RS-232C interface is unable to provide the continuous display feature, use the **1DPA1** command to instruct the system to report the encoder position one time only. Then, turn the rotor shaft manually to verify that the absolute encoder/decoder is operating properly.

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

Making a Move with an Absolute Encoder and a DC Motor

This section explains how to make a move with the JSI controller, an absolute encoder, and a drive as a system. You should have verified that the absolute encoder is working properly with the JSI in the *Testing an Absolute Encoder/Decoder* section of this chapter.

The absolute encoder has a finite position coordinate system. The encoder begins with a count of zero and counts up to a finite number that is the end of the absolute coordinate system. It then rolls over and starts at zero again. Compumotor recommends that the JSI and the absolute encoder are not used in the Continuous mode (eventually the end or the roll-over point of the coordinate system will be reached.)

The following steps instruct the motor to rotate 4,000 steps in one direction and then rotate 4,000 steps in the opposite direction. This back-and-forth motion will constitute one loop. The loop will be performed 10 times. To complete this move, perform the following steps.

Step 1 Ensure that the drive, encoder, and JSI are properly wired.

Step 2 Apply power to the JSI and encoder *only*.

Step 3 Issue the **1DPA** command and ensure that the position is more than 8,000 steps from the point or the end of the absolute coordinate system. As a safety precaution in the absolute coordinate system, you should set software end-of-travel limits to prevent the motor from crossing the roll-over point.

Step 4 Issue the following commands:

Command	Description
> SL4ØØØ,9ØØØØØ	Sets the lower and upper end of travel limits
> SLDØ	Enables travel limits

Step 5 Issue the **ON** command to the JSI controller. The motor/ drive should now hold its position. If the motor runs away (does not hold its position) reverse the **CMD+** and **CMD-** connections on the JSI controller and try this step again.

Step 6 To perform an incremental move, issue the following commands:

Command	Description
> PS	Pauses the system until you issue a Continue (**C**) command
> MN	Sets the system to Normal mode
> MPI	Sets the unit to Incremental mode
> LD3	Disables the CCW and CW limits
> A1Ø	Sets the acceleration to 10 rps^2
> V1	Sets velocity to 1 rps
> D4ØØØ	Sets distance to 4,000 steps
> L1Ø	Loops commands 10 times
> G	Executes the move (Go)
> T2	The system waits two seconds after the move
> H	Changes the direction of the motor
> N	Ends the loop
> C	Continues execution

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

Limit Switches To test the CW and CCW Limit switches with the JSI controller, complete the following test procedures.

- Ensure that the CW and CCW limit switches are wired properly.
- Make sure that the load is not attached to the motor.
- Make sure that you can manually open and close the limit switches.

Step 1 Turn off (open) the **CW** and **CCW Limit** switches.

Step 2 Type **1IS.** Assuming that you have not turned on any other inputs, you should receive the following response `*11ØØ_ØØØØ_ØØØØ_ØØØØ`. This means that both CW and CCW limits are off (closed).

Step 3 Turn on (close) the **CW** and **CCW Limit** switches.

Step 4 To test the CW limit, enter the following commands:

Command	Description
> LDØ	Enables the **CW** and **CCW Limits**
> INLØ	Normally closed limit inputs
> MC	Sets controller to Continuous mode
> A1	Sets acceleration to 1 rps^2
> V1	Sets velocity to 1 rps
> H+	Changes the direction of the motor (CW)
> G	Executes the move (Go)

The motor should move in the CW direction.

Step 5 Turn off (open) the **CW Limit** switch. The motor should stop moving and error code **41** should be flashing in the two-digit LED. This verifies that the **CW Limit** switch is working properly.

Step 6 Turn on (close) the **CW Limit** switch. Error code **41** should no longer be displayed in the two-digit LED.

Step 7 To test the CCW limit, enter the following commands:

Command	Description
> H-	Changes the motor's direction (CCW)
> G	Executes the move (Go)

The motor should move in the CCW direction.

Step 8 Turn on (close) the **CCW Limit** switch. The motor should stop moving and error code 41 will be displayed in the two-digit LED. This verifies that the **CCW Limit** switch is working properly.

Step 9 Turn on (close) the **CCW Limit** switch.The motor should begin to move in the CCW direction. Error code **41** should no longer be displayed in the two-digit LED.

If you are not able to stop the limit switches, reverse your current wiring of the CW and CCW Limit inputs. Try to perform this test again. If you are still unsuccessful, refer to Chapter 7, Maintenance and Troubleshooting.

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

Homing The Motor

You can initiate the Go Home function from either the RS-232C interface or the Go Home input configured using the Configure Input (**IN**) command. You must also define the Go Home Velocity (**GHV**), Go Home Acceleration (**GHA**), and Final Go Home Velocity (**GHF**) to properly initiate the Go Home function. You can define the initial direction of the homing function with the Go Home Velocity (**GHV**) command. You can initiate the homing function by entering the Go Home (**GH**) command or enabling the Go Home input.

When you command the motor to go home, it begins to move in the direction and at the velocity you specified. It performs this move at the last defined acceleration rate (Go Home Acceleration), and looks for the home limit input to go active. If the motor encounters an end-of-travel limit while it searches for home, it will reverse direction and look for the **HOME LIMIT** input to go active in the opposite direction. If the motor encounters the other limit before it detects the home signal, the Go Home move will be aborted and the motor will stop.

To test the functionality of the **HOME LIMIT** switch, complete the following steps.

Step 1 Manually open the **HOME LIMIT** switch and type **1IS.** Assuming your end-of-limits are closed (off), the system should respond with `*11ØØ_ØØØØ_ØØØØ_ØØØØ`.

Step 2 Close the **HOME LIMIT** switch and type **1IS.** The system should respond with `*111Ø_ØØØØ_ØØØØ_ØØØØ`. This verifies that the home limit switch is functioning properly.

Step 3 Open the **HOME LIMIT** switch.

Step 4 To test the JSI's homing function, enter the following string of commands.

Command	Description
`> IN1S`	Sets up **IN1** (Input 1) as a remote Go Home input
`> 1GHA5`	Set go home acceleration to 5 rps^2
`> GHF.2`	Sets final go home velocity to 0.2 rps
`> GHV+1`	Sets the go home velocity to 1 rps in the CW direction
`> GH`	Executes the Go Home function

Step 5 Ensure that the home limit input switch is open (on).

Step 6 Close **IN1** (Input 1) and then open it. (The motor will start to search for the **HOME LIMIT** switch again.)

Step 7 Momentarily ground (turn off) the **HOME LIMIT** input. The motor decelerates to a stop.

The following events will occur as a result of these commands:

1. The motor moves in the CW direction at constant velocity (1 rps). You must momentarily ground the **HOME LIMIT** input to power the opto isolator.
2. The motor decelerates to a stop.

When you open the **HOME LIMIT** input after issuing a **GH** command, the system recognizes the location where the input became active as home. The drive decelerates the motor at the last rate specified. If during this deceleration period the home limit becomes inactive, the drive will move the motor back at a rate specified by the Go Home Final (**GHF**) command until the limit is again active. To reset the absolute counter to zero at the end of a go home move, enter the Position Zero (**PZ**) command.

The successful completion of this test verifies that the homing function is working properly with the JSI controller.

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

Inputs & Outputs

This section presents information about the JSI's inputs and outputs. The following inputs and outputs will be addressed:

- Programmable Inputs
- Programmable Outputs and/or Faults

Programmable Inputs

The input and output modes of the JSI controller offer flexibility and many function options for the input and output pins. All of the IM modes that are available in previous Compumotor products are achievable with the JSI (plus much more). You can program each input pin or designate them to correspond to a particular function. There are 19 functions that you can designate. To designate each input pin to a particular function, use the Set Input Functions (**IN**) command. The input pins are numbered **IN 1** - **IN 13** on the front panel. The three remaining inputs are dedicated to **CW**, **CCW**, and **HOME LIMIT** inputs. Each function has been assigned a letter:

A: Trigger
B: Sequence
C: Kill
D: Stop
E: Amplifier Shutdown
F: Pause/Continue
G: Go
H: Direction
J: Jog+ (CW)
K: Jog- (CCW)
L: Jog Speed Select
M: Terminate Loop
N: Data
P: Memory Lock
R: Reset
S: Go Home
T: Position Zero
U: User Fault
V: Data Valid

To program a particular input line to perform one of the designated functions, enter the **IN** command, followed by the pin number (1-13), followed by the letter of the particular function that you are designating the pin to perform. You may define the active input level with the Set Input Active Level (**INL**) command. The factory setting is **INLØ** (Active low).

Verifying Proper Input Wiring

The following steps show how you can configure Input 1 (**IN1**) as a Remote Go input and Inputs 2 and 3 (**IN2** and **IN3**) as trigger inputs.

Step 1

Command	Description
> MN	Sets to mode normal
> INL	Sets input level to be active low (off)
> LD3	Disables CW and CCW Limits
> IN1G	Sets up first input (IN1) as Go input.
> IN2A	Sets up second input (IN2) as trigger input bit 1
> IN3A	Sets up third input (IN3) as trigger input bit 2
> A2	Sets acceleration to 2 rps^2
> V.5	Sets velocity to 0.5 rps
> D4ØØØ	Sets distance to 4,000 steps
> TR1	Waits for Trigger bit 1 to go off and Trigger bit 2 to go on. (These commands defines the first three inputs and defines the velocity and acceleration)
> G	Executes the move (Go)

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

Step 2

Command	Response
> 1IN1	*Ø1_G_GO_INPUT_(STATUS_ON)
> 1IN2	*Ø2_A_TRIGGER_INPUT_(STATUS_ON)
> 1IN3	*Ø3_A_TRIGGER_INPUT_(STATUS_ON)

These commands verify that inputs (**IN1** through **IN3**) have been configured properly. If not, go back and try step 1 again.

Step 3 Turn on trigger bit 1(**IN2**) and turn off trigger bit 2 (**IN3**). This will satisfy the wait for trigger condition and proceed with command execution.

Step 4 Turn on the Go input (**IN1**). The motor will make a preset move of 4,000 steps again.

This verifies that the inputs can be configured as programmable inputs and start inputs. Other **IN** features are discussed in Chapter 4, Application Design.

Programmable Outputs

You can program the JSI controller's outputs in the same way that you program the product's inputs. There are 17 possible functions for which the outputs can be defined with the **OUT** command. Refer to the **OUT** command in Chapter 5, Software Reference for a detailed explanation of command. The possible output configurations are listed below.

A: Programmable Output
B: Moving/Not Moving
C: Sequence in Progress
D: At Limits
E: At Position Zero
F: Fault
G: Dead Band
H: Amp Off
J: Strobe
K: Command Error
L: Position Error Fault
M: Check Sum Error
N: CW Software Limit Reached
P: CCW Software Limit Reached
R: CW Hardware Limit Reached
S: CCW Hardware Limit Reached
T: Output Based on Position

The outputs are labeled on the front panel of the JSI as **OUT 1**-**OUT 8**. They are assigned in the same manner as the input pins. You can set the active level of the output with the Set Active Output Level (**OUTL**) command. The factory setting is **OUTLØ** (current flows when turned on). All of the outputs (**OUT 1** - **OUT 8**) are set up as programmable outputs as the factory default setting. The outputs are open collector outputs, so you must connect the outputs using a pull up resistor and a power supply (5 - 30VDC).

The following steps show how you can set up (**OUT 1**) as a moving/not moving output and **OUT 2** as a programmable output.

Step 1 Make sure the outputs are wired up properly with pull up resistors and a power supply.

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

Step 2

Command	Description
> OUT1B	Sets up **OUT1** as a moving/Not moving output.
> OUT2A	Sets up **OUT2** as a programmable output.
> A1	Sets acceleration to 1 rps^2
> V5	Sets velocity to 5 rps
> D-4ØØØ	Sets distance to 4000 steps
> G	Executes the move
> T3	Waits three seconds
> O1	Turns on output 2 (**OUT2**)
> T3	Waits three seconds
> OØ	Turns off output 2 (**OUT2**)

While the motor is making the first move, **OUT1** should stay on. When the first move is finished, **OUT1** should go off. After 3 seconds, **OUT2** should come on and stay on for four seconds before turning off.

This verifies that you can configure the controller's outputs to work as a programmable output and Moving/Not Moving outputs. In Chapter 4, *Application Design*, we will explain the use of other functions that you can program with the Configure Output (**OUT**) command.

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

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
- Customize the system to meet your requirements
- 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. The JSI can perform all of the profiles discussed in this chapter.

Triangular and Trapezoidal Profiles

For constant acceleration control systems, velocity, acceleration, and distance parameters are defined before the system can execute a preset move. The value of these parameters determines the type of motion profile as either triangular or trapezoidal.

A triangular profile results when the velocity and acceleration are set so that the motor does not attain the defined velocity before the motor travels half of the distance that you specify. This results from either a relatively low acceleration, a relatively high velocity, or both. The motion profile for this move is shown in Figure 4-1.

A trapezoidal profile (see Figure 4-1) results when the motor reaches the defined velocity before the motor moves half of the specified distance. A trapezoidal profile may occur if you specify a low velocity with a high acceleration or a long distance.

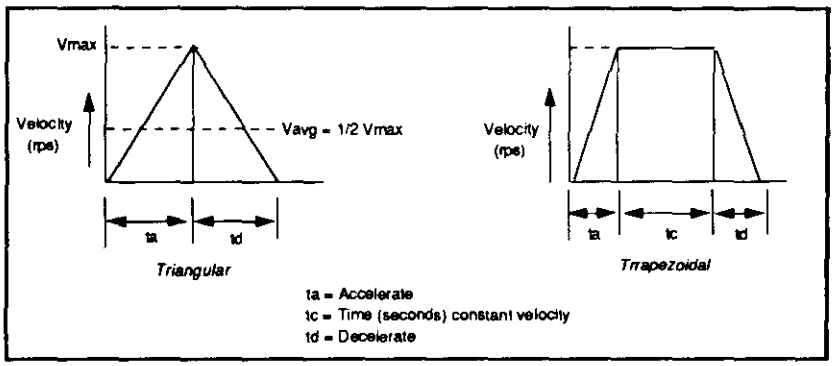


Figure 4-1. Triangular and Trapezoidal Profiles

Move Times: Calculated vs Actual

You can calculate the time it takes to complete a move by using the acceleration, velocity, and distance values that you define. However, you should not assume that this value is the actual move time. There is calculation delay and motor settling time that makes your move longer. You should also expect some time for the motor to settle into position. You can decrease the motor's settling time by tuning the JSI controller and the drive. Tuning is discussed later in this chapter.

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

Incremental vs. Absolute Positioning

A preset move is a move distance that you specify (in encoder steps). You can select preset moves by putting the JSI into normal mode using the Mode Normal (**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 by using the Mode Position Incremental (**MPI**) command. You can select absolute moves using the Mode Position Absolute (**MPA**) command

Continuous Mode Moves

The Mode Continuous (**MC**) command accelerates the motor to the velocity that you last specified with the Velocity (**V**) command. The motor continues to move at the specified velocity until you issue the Stop (**S**) command or specify a velocity change. To change velocity while the motor is moving, use the **V** command followed by the **G** command. The Continuous mode is useful for applications that require constant movement of the load, when the motor must stop after a period of time has elapsed (rather than after a fixed distance), or when the motor must be synchronized to external events such as trigger input signals.

Command	Description
> PS	Waits to Continue (**C**) command to execute next command
> IN1A	Sets up **IN 1** (Input 1) as trigger bit 1
> IN2A	Sets up **IN 2** (Input 2) as trigger bit 2
> LD3	Disables the CW and CCW limits (***this command is not necessary if you have installed the limits***)
> MC	Sets unit to the Continuous mode
> A2	Sets acceleration to 2 rps^2
> V.1	Sets velocity to 0.1 rps
> G	Executes the move (Go)
> T1	Waits 1 second after the motor reaches constant velocity
> V.5	Sets velocity to 0.5 rps
> G	Changes velocity (Go)
> TR1Ø	Waits for trigger bit 1 to go on and bit 2 to go off
> VØ	Sets velocity to 0 rps
> G	Changes velocity to 0 rps
> MN	Changes mode to preset mode
> C	Instructs the unit to continue executing commands

This sequence sets the JSI to the Continuous mode. The motor reaches 0.1 rps, waits one second, changes velocity to 0.5 rps, waits for you to turn **IN 1** (Input 1) on and turn **IN 2** (Input 2) off, and then stops. Note that the commands **VØ** and **G** stop the motor (the **S** command is not a buffered command and cannot be used in this situation, unless you wish to halt the operation in the middle of the program).

Incremental Mode Preset Moves

When you are in the Incremental mode (**MPI**), a preset move moves the motor the specified distance from its starting position. You can specify the direction of the move in two ways. You can specify the direction by using the optional sign (D+8,000 or D-8,000), or you can define it separately with the Set Direction (**H**) command (H+ or H-).

Command	Description
> LD3	Disables the CW and CCW Limits (***this command is not necessary if you have installed the limits***)
> MPI	Sets unit to Incremental Position Mode
> A2	Sets acceleration to 2 rps^2
> V5	Sets velocity to 5 rps
> D8ØØØ	Sets distance to 8,000 steps (two revolutions)
> G	Executes the move (Go)
> G	Repeats the move (Go)
> H	Reverses direction of next move
> G	Executes the move (Go)

The motor moves two revolutions and stops. It then moves another two revolutions in the same direction and stops. The motor changes direction and moves two revolutions.

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

Absolute Mode Preset Moves

A preset move in the absolute mode (**MPA**) moves the motor the distance that you specify (in encoder steps) from the absolute zero position. You can set the absolute position to zero with the Position Zero (**PZ**) command or by cycling the power to the controller. The absolute zero position is initially the power-up position.

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.

The JSI powers up in Incremental mode. 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 JSI retains the absolute position, even while the unit is in the Incremental mode. You can use the Position Report (**PR**) command to read the absolute position.

In the following example, the motor makes its moves with respect to the absolute zero position. The motor makes a final move to return to the absolute zero position.

If you are using an absolute encoder, you must ensure that the encoder is operating within the valid range. If you move the encoder beyond the roll-over point, the JSI will lose track of its position. Refer to the manual that accompanied your absolute encoder for a more detailed discussion of roll-over points.

If you use absolute encoder feedback (**CFB2**) upon powering up the JSI controller, the JSI will read the actual position of the absolute encoder. You can use the **1DPA** command to determine the absolute encoder's position. In Absolute mode, all the moves you make will be with respect to this absolute position. If you use the Position Zero (**PZ**) command, the absolute position (**DPA**) does not change. The distance that you define for motor movement, however, will be measured from the point specified as zero.

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 as zero position
> D1ØØØØ	Sets distance to 10,000 steps
> G	Moves to absolute position 10,000 (Go)
> D2ØØØØ	Sets distance to 20,000 steps
> G	Moves the motor to absolute position 20,000 (Go)
> DØ	Sets the move distance to absolute position 0
> G	Moves to absolute position zero
> MPI	Sets controller to Incremental Position mode

The absolute positioning function described in this section refers to the controller's ability to retain the position of the motor while the JSI is powered up. However, once power to the JSI is lost, the motor position information is also lost.

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

Application Considerations

This section contains information that you should consider and evaluate when designing and developing your system.

Positional Accuracy vs. Repeatability

Some applications require high absolute accuracy. Others require repeatability. You should clearly define and distinguish these two concepts when you address the issue of system performance.

If the positioning system is taken to a fixed place and the coordinates of that point are recorded. The only concern is how well the system repeats when you command it to go back to the same point. For many systems, what is meant by accuracy is really repeatability. Repeatability measures how accurately you can repeat moves to the same position.

Accuracy on the other hand, is the error in finding a random position. For example, suppose the job is to measure the size of an object. The size of the object is determined by moving the positioning system to a point on the object and using the move distance required to get there as the measurement value. In this situation, basic system accuracy is important. The system accuracy must be better than the tolerance on the measurement that is desired.

Consult the technical data section of The Compumotor Catalog for more information on accuracy and repeatability.

Achieving Maximum Motor/Drive Performance with the JSI

When you use the JSI controller with a DC motor or any other analog devices, you can expect to achieve the maximum velocity that is specified in the DC drive/motor user guide. Certain factors may prevent the motor from achieving its specified maximum velocity. The drive you are using is the main factor that may prevent your motor from achieving its maximum velocity. You must tune your drive with the load attached to the motor to get the best response from the drive. After achieving the peak performance from the drive, you can tune the JSI (PID tuning) to obtain the optimum performance from your entire system.

When you tune your drive and the JSI controller, consider the type of load that you are moving. The load will affect the system's response and settling time. For detailed information on tuning, refer to *Tuning the JSI* at the end of this chapter.

Closed-Loop Operation

The JSI uses position feedback to position the load accurately. The drive, or the servo valve you are controlling must control the velocity feedback. To close the position loop of the JSI controller, you can use either an incremental encoder or a Compumotor absolute encoder. You can configure the JSI to accept different feedback devices with the Configure Feedback Device (**CFB**) command.

- **CFB1**: Indicates feedback device is incremental encoder
- **CFB2**: Indicates feedback device is AR23 absolute encoder with -4 decoder box
- **CFB3**: Indicates feedback device is AR-C, AL-C, or Photo-Trak absolute encoder
- **CFB4**: Indicates feedback device is AR23 absolute encoder with -1 decoder box

Once you define the feedback device and the resolution of the feedback device, the JSI controller will scan either the POSITION FEEDBACK Input for absolute encoder feedback (**CFB2**, **CFB3**, or **CFB4**) or the ENCODER input for incremental encoder feedback (**CFB1**). You must turn off the JSI with the **OFF** command before changing the encoder type or resolution.

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

In incremental encoder mode (**CFB1**), the JSI can accept TTL Square Wave quadrature pulses in either a differential or single-ended fashion. Compumotor's incremental encoders are differential quadrature encoders (refer to Figure 3-4 in Chapter 3 for a wiring diagram of this encoder). You can also define encoder resolution with the **CFB1** command. The default setting is 4,000 steps per revolution. The maximum frequency of pulses that the JSI can read is 60 KHz before quadrature. Therefore, after quadrature, the maximum frequency the JSI can read is 240 KHz.

In the absolute encoder mode (**CFB2**, **CFB3**, or **CFB4**), the JSI reads the parallel data that the absolute encoder provides. Because it takes longer to process parallel positional information, the servo sample period for the PID loop will be slowed to 1,024 microseconds from 512 microseconds. However, the I/O functions are still performed every 512 microseconds. Refer to Chapter 3, *Installation*, for connection diagrams for the absolute encoders.

Incremental Encoder Example

This example sets up the JSI to receive feedback from the incremental encoder and make a closed loop move.

Command	Description
> LD3	Disables CW and CCW limits
> OFF	Shutdown JSI Output
> CFB1,4000	Defines the feedback device as incremental encoder with resolution of 4,000 steps/rev after quadrature
> ON	Turn on JSI output
> A10	Sets acceleration at 10 rps^2
> V5	Sets velocity at 5 rps
> D40000	Sets move distance to 1 revolution
> G	Executes the move (Go)

Absolute Encoder Example

Command	Description
> LD3	Disables CW and CCW limits
> OFF	Disables **COMMAND OUT**
> CFB2,16384	Defines the feedback device as AR23 -4 absolute encoder with a resolution of 16,384 steps/rev after quadrature
> ON	Enables **COMMAND OUT**
> A10	Sets acceleration at 10 rps^2
> V5	Sets velocity at 5 rps
> D16384	Sets move distance to 1 revolution
> G	Executes the move (Go)

Homing Function

You can use this function to establish a home reference position. You can instruct the JSI to move the motor to a home position with the Go Home (**GH**) command. This home position is typically established by mounting a load-activated switch to the home limit input so that the switch turns on when the load is in the desired position.

When you use an incremental encoder, the Z Channel or Index Channel, if any, may be used in conjunction with the home limit switch to establish the home position. The home position is located where the edge selected with the OS command of the Home Limit input occurs. (i.e., the controller recognizes the home position as the position where the home limit signal makes a transition from on to off, or from off to on, depending on the selected edge and the initial direction of the go home move.) Once it recognizes the selected edge , the motor decelerates to a stop. After coming to a stop, the JSI positions the motor away from the selected edge of home limit signal in the opposite direction of the initial direction of the go home move. After coming to a stop a second time, the JSI creeps the motor towards the selected edge at Go Home Final Velocity (**GHF**) until the home limit input becomes active again or the home limit and the encoder Z Channel pulse are active. The homing function is designed to accommodate encoder index signals that are typically of short duration. The function works best with this kind of home signal.

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

You must ensure that the final approach starts from the opposite side of the signal from the selected edge. If the final approach direction is positive , the final move must start from the negative side of the selected edge. If there is significant backlash and friction in the system, and the indexer is instructed to go home in the clockwise direction, the motor may end up on the wrong side of the signal and execute its final approach in the wrong direction.

This problem can also occur if the motor's go home speed is high, and the Home limit signal is delayed, (by a relay or Programmable Controller for example). In such a situation, you should initiate homing operations from the opposite side of the selected edge of home.

When you conclude the homing operation, the indexer resets its internal position counter. You can determine the true indexer position referenced to an encoder with the Report Absolute Encoder Position (**DPA**)command.

You can use the home limit input in conjunction with the encoder's Channel Z input to select a final home position. To activate homing **Z** bit, you must activate the **OSD1** command. In this situation, a load-activated switch connected to the Home Limit input locates the general home position area, and the indexer channel signal from the encoder is used for final Home positioning. The Channel Z and Home Enable inputs must both be active to mark the home position.

Under interface control, the Go Home (**GH**) command has the form **GH**<direction><velocity>. This indicates which direction to move, and at what velocity. For example, the command GH-2 sends the motor in the negative direction at 2 rps in search of the home signal. The acceleration parameter for this move is the home value for acceleration (**GHA** command). If an end-of-travel limit is activated before home is found, the controller reverses direction and attempts to find the home position again. If the other limit is activated before the controller finds home, the controller will stop trying to go Home. The controller can indicate whether or not the homing process was successful by responding to the Request Indexer Status (**R**) and Go Home Status (**RG**) commands.

Command	Description
> LD3	Disables Positive and Negative limit inputs
> OSB1	Back up to Home limit
> OSD1	Enables Go Home to Z bit
> GHA1	Sets go home acceleration to 1 rps^2
> GH.5	Sets Go Home in the positive direction at 0.5 rps

The motor starts to move toward the home position. Upon encountering the home switch, the motor comes to a stop. The motor changes direction and searches for the home switch again. When it sees the switch, the motor looks for the Z bit (if **OSB1** and **OSD1** are enabled) to turn on after the home limit switch is closed. In encoder mode, home limit switch and Z bit must be on at the same time for the motor to Go Home successfully.

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

Programmable Inputs

You can use the Trigger Pause (**TR**) command to cause a sequence of buffered commands to wait for one or more inputs to come to a preferred state. Inputs **IN 1 - IN 13** are set at the factory (default setting) to function as programmable inputs.

You may also configure programmable inputs to perform other functions. Refer to the Configure Input Mode (**IN**) command for a description of the other functions that are available.

Command	Description
> IN1C	Sets up **IN 1** as Kill (K) input
> IN2G	Sets up **IN 2** as Go (G) input.
> IN3A	Sets up **IN 3** as trigger input 1
> IN4A	Sets up **IN 4** as trigger input 2
> MN	Sets the unit to Preset mode
> A2	Sets acceleration to 2 rps^2
> V5	Sets velocity to 5 rps
> D25ØØØ	Sets distance to 25,000 steps
> TRØ1	Waits for trigger input 1(**IN 3**)to be off and trigger input 1(**IN 4**) to be off
> G	Executes 25,000-step move (Go)

This example program configures **IN 1** as a Kill input and **IN 2** as a Start input. **IN 3** and **IN 4** are set up as programmable trigger inputs. If you activate the Start input (**IN 2**), the motor will move 25,000 steps. The rest of the program waits for the trigger inputs to match before executing the move. At any time during the operation of the JSI, if you activate **IN 1**, the controller immediately stops the operation and clears the command buffer.

Delays

You can use the Time (**T**) command to halt the operation of the indexer function for a preset time. If you are in the Continuous mode, you may use the Time (**T**) command to run the motor at continuous velocity for a set time, then change to a different velocity.

In the Preset mode, the motor finishes the move before the indexer executes the time delay.

Command	Description
> PS	Waits for the controller to receive a Continue (**C**) command before executing the next command
> G	Moves motor 25,000 steps
> T5	Waits 5 seconds after the move ends
> H	Changes motor direction
> G	Moves motor 25,000 steps in the opposite direction
> C	Continues execution

Branching

You can perform conditional branching with the Compare Error Flag (**IFER**), Compare User Flag (**IFFL**), and Compare Input Status (**IFIN**) commands. All three of these commands are very similar to IF_THEN statement in BASIC programming. If the condition matches, the JSI will perform the commands that immediately follow the **IF** command. If the condition does not match, the JSI will skip all of the commands that follow the IF command until it reaches the End of IF Statement (**NIF**) command.

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

IFER Command

This command checks to see if any error condition exists to execute a conditional command. This command is useful if you wish to trap different error conditions (Drive Disabled, User Fault Input Activated, Excessive Position Error, etc). For a detailed description of this command, refer to Chapter 5, *Software Reference*.

Command	Description
> XD1Ø	Defines Sequence 10. When a fault occurs, this sequence 10 will execute. The sequence is defined with the Set Fault or Kill Sequence (**XFK1Ø**) command.
> IFER1	If hardware **CCW LIMIT** switch is reached, perform the following commands:
> A1Ø	Sets acceleration to 10 rps^2
> V5	Sets velocity to 5 rps
> D1ØØØØ	Sets distance to 10,000 steps
> G	Executes the move (Go)
> NIF	Ends IF statement
> IFERX1	If hardware **CW LIMIT** switch is reached, perform the following commands:
> A1Ø	Sets acceleration to 10 rps^2
> V5	Sets velocity to 5 rps
> D-1ØØØØ	Sets distance to 10,000 steps in the opposite direction
> G	Executes the move (Go)
> NIF	Ends IF statement
> IFERXXXXXXX1	If Drive Enable (**ENBL+**) input is opened, perform the following commands:
> 1" CONNECT_	Writes to the RS-232C port to inform the user
> 1"DRIVE_	that drive enable is disconnected.
> 1"ENABLE_	
> 1"INPUT_	
> NIF	Ends **IF** statement
> XT1Ø	Ends Sequence loop
> XFK	Sets Sequence 10 as the Fault sequence.

IFFL Command

This command uses the pattern set by the User Flag (**SFL**) command to execute the conditional commands. This command is useful if you wish to make a decision based on previous sequence executions that will set or clear the user flag bits. For example, in an application with several sequences, at the end of each sequence, you can assign different bit patterns with the **SFL** command. If you select these sequences from the host computer, you may wish to make different moves depending on the sequence you ran. For a detailed description of this command, refer to Chapter 5, *Software Reference*.

Command	Description
> PS	Waits for the controller to receive a Continue (**C**) command before executing the next command
> SFL1Ø1Ø	Sets user flag bits 7 and 5 and clears bits 6 and 4, the remaining bits are not altered
> IFFL1Ø1Ø	If user flag bits 5 and 7 are set, and bits 6 and 4 are clear, perform the following commands
> A1Ø	Sets acceleration to 10 rps^2
> V5	Sets velocity to 5 rps
> D25ØØØ	Sets distance to 25,000 steps
> G	Executes the move (Go)
> NIF	Ends IF statement
> C	Continues execution

The **IFFL** pattern matches the **SFL** setting. The motor moves 25,000 steps.

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

IFIN Command This command compares the input pattern (**CW**, **CCW**, and **HOME LIMIT** outputs and **IN 1** - **IN 13** inputs) to execute the conditional commands. This command is useful for branching and performing conditional moves using the programmable inputs. For a detailed description of this command, refer to Chapter 5, *Software Reference*.

Command	Description
> IFINXXX1Ø	If **IN 1** is active and **IN 2** is not active, issue the following commands:
> A1Ø	Sets acceleration to 10 rps^2
> V5	Sets velocity to 5 rps
> D25ØØØ	Sets distance to 25,000 steps
> G	Executes the move (Go)
> NIF	Ends IF statement
> IFINXXXØ1	If **IN 1** is not active (open)and **IN 2** is active (closed), issue the following commands:
> A1Ø	Sets acceleration to 10 rps^2
> V5	Sets velocity to 5 rps
> D5ØØØ	Sets distance to 5,000 steps in the opposite direction
> G	Executes the move (Go)
> NIF	Ends IF statement
> IFINXXX1	If **IN 1** is active, do the following command.
> 1"DONE	Ends message saying done
> NIF	Ends IF statement

Subroutines When you use the Goto Sequence (**XG**) and the Execute a Sequence (**XR**) command sequences, you can execute different sequences from within a sequence. These commands are similar to GOTO and GOSUB commands in BASIC programming. If you use an **XG** command, the program will move to the sequence that you specified in the **XG** command. After executing the specified sequence, the system will not return to the original sequence. It will remain in the current sequence, unless it receives another execution command (**XG** or **XR**). However, if you use the **XR** command, the program will return control to the original sequence that contained the **XR** command. Program control will return to the original sequence when a Terminate Sequence (**XT**) command is reached in a sequence. This prompts the program to return to the sequence that initiated the move to another sequence.

You can nest as many as 16 different levels of sequences within one program. For example, when you exit Sequence 1 to execute Sequence 2 with the **XR2** command, you can execute to Sequence 3 from Sequence 2. This procedure of nesting **XR** commands can be repeated 16 times. The **XG** command has no limit since the program will not return control to the original sequence.

Command	Description
> XE2	Erases Sequence 2
> XD2	Defines Sequence 2
A1Ø	Sets acceleration to 10 rps^2
V5	Sets velocity to 10 rps
D2ØØØ	Sets distance to 2,000 steps
G	Executes the move (Go)
> XT	Ends Sequence 2 definition
> XE3	Erases Sequence 3
> XD3	Defines Sequence 3
A1Ø	Sets acceleration to 10 rps^2
V5	Sets velocity to 10 rps
D-2ØØØ	Sets distance to 2,000 steps
G	Executes the move (Go)
> XT	Ends Sequence 3 definition
> XE1	Erases Sequence 1
> XD1	Defines Sequence 1
XR2	Executes Sequence 2
XR3	Executes Sequence 3
> XT	Ends Sequence 1 definition
> XR1	Executes Sequence 1

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

In the previous example, when you execute Sequence 1, the program moves to Sequence 2. After executing Sequence 2, the program returns to Sequence 1. The program then moves to execute Sequence 3.

Command	Description
> XE1	Erases sequence 1
> XD1	Defines sequence
IFINXXX1	If **IN 1** is active (Closed)
XG2	Executes Sequence 2
NIF	Ends If statement
A1	Sets acceleration to 1 rps^2
V5	Sets velocity to 5 rps
D25ØØØ	Sets distance to 25,000 steps
G	Executes move.
> XT	Ends Sequence 1 definition
> XR1	Executes Sequence 1

In the following example, when you execute Sequence 1, the program checks the input pattern. If **IN 1** is on, the program moves to execute Sequence 2. After executing sequence 2, program does not return to Sequence 1. If **IN 1** is off, the program ignores the **XG2** command and makes the 25,000-step move.

Loops

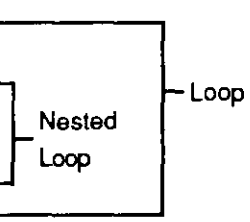
You may use the Loop (**L**) command to repeat certain programs. You can nest Loop commands up to 16 levels deep.

Command	Description
> PS	Pauses command execution until the controller receives a Continue (**C**) command
> MPI	Sets unit to Incremental mode
> A5	Sets acceleration to 5 rps^2
> V5	Sets velocity to 5 rps
> L5	Loops 5 times
D2ØØØ	Sets distance to 2,000 steps
G	Executes the move (Go)
T2	Delays 2 seconds after the move
> N	Ends loop
> C	Initiates command execution to resume

The motor moves a total of 10,000 steps

The example below shows how you can nest a small loop inside a major loop. In this example, the motor makes 2 moves and returns a line feed. The unit repeats these procedures and will continue to repeated until you instruct the unit to stop.

Command	Description
> PS	Pauses command execution
> L	Loops indefinitely
> 1LF	Sends a line feed
> L2	Loops twice
> G	Executes 2,000-sstep move
> T.5	Waits 0.5 seconds
> N	Ends loop
> N	Ends loop
> C	Continues command execution



This command execution continues until you issue the Stop (**S**) or Kill (**K**) commands.

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

POBs (Programmable Output Bits)

You can turn the programmable outputs (**OUT 1** - **OUT 8**) on and off with the Output (**O**) and Immediate Output (**IO**) commands. Outputs **OUT 1** through **OUT 8** are factory set as programmable outputs. However, you can configure all of the outputs to perform different functions (Moving/Not Moving, Amp Off, Strobe, etc.) with the Configure Output(**OUT**) command. Refer to the **OUT** command in the Chapter 5, Software Reference for descriptions of the available functions. You can use these outputs to turn on and off other devices (i.e., lights, switches, etc.).

Command	Description
> PS	Pauses command execution until JSI receives Continue (**C**) command
> A1Ø	Sets acceleration to 10 rps^2
> V5	Sets velocity to 5 rps
> D25ØØØ	Sets distance to 25,000 steps
> OUT1A	Sets **OUT 1** as a programmable output
> OUT2A	Sets **OUT 2** as a programmable output
> OUT3B	Sets **OUT 3** as a Moving/Not Moving output
> O1Ø	Turns **OUT 1** on and **OUT 2** off
> G	Executes the move
> OØ1	Turn **OUT 1** off and **OUT 1** on
> C	Initiates command execution to resume

This example above defines **OUT1** and **OUT2** as programmable outputs and **OUT3** as a Moving/Not Moving output. Before the motor moves 25,000 steps, **OUT1** is turned on and **OUT2** is turned off. These outputs will remain in this state until the move is completed, then **OUT1** will turn off and **OUT2** will be turned on. While the motor is moving, **OUT3** remains on.

Move Completion Signal

When you complete a move, you may use the JSI's programming capability to signal the end of the current move. In a preset move, you may use one of the following commands:

- **LF** Line feed
- **CR** Carriage return
- **O** Output command
- **"** Quote command

Example# 1

Command	Description
> A2	Sets acceleration to 2 rps^2
> V.5	Sets velocity to 0.5 rps
> D125ØØ	Sets distance to 12,500 steps
> G	Executes the move (Go)
> 1LF	Sends a line feed over the RS-232C interface

The motor moves 12,500 steps. When you complete the move, the unit issues a line feed from the JSI to the host over the RS-232C interface.

Example #2

Command	Description
> A2	Sets acceleration to 2 rps^2
> V.5	Sets velocity to 0.5 rps
> D125ØØ	Sets distance to 12,500 steps
> G	Executes the move (Go)
> 1CR	Sends a carriage return

The motor moves 12,500 steps. When the JSI completes the move, the unit issues a carriage return from the JSI to the host over the RS-232C interface.

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

Example #3

Command	Description
> OUT1A	Sets Output 1 (**OUT 1**) as a programmable output
> A2	Sets acceleration to 2 rps^2
> V.5	Sets velocity to 0.5 rps
> D125ØØ	Sets distance to 12,500 steps
> G	Executes the move (Go)
> O1	Turns on Output 1

The motor moves 12,500 steps. When the move is completed, Output 1 is turned on.

Example #4

Command	Description
> A2	Sets acceleration to 2 rps^2
> V.5	Sets velocity to 0.5 rps
> D125ØØ	Sets distance to 12,500 steps
> G	Executes the move (Go)
> 1"DONE	Sets the Quote DONE message

The motor moves 12,500 steps. When you complete the move, the unit issues the DONE message from the JSI to the host over the RS-232C interface.

Sequence Commands

Use the following commands to define, erase, and run sequences. Refer to Chapter 5, *Software Reference*, for detailed descriptions and syntax of the following commands.

Command	Description
> XBS	Reports the number of bytes available for sequence programming
> XD	Starts sequence definition
> XE	Deletes sequence from EEPROM
> XQ	Sets/resets interrupted Run mode
> XRP	Runs a sequence with a pause
> XT	Ends sequence definition
> XU	Uploads sequence
> XR	Runs a sequence
> SSJ1	Runs a sequence defined by BCD sequence inputs
> XG	Exits current sequence and moves to execute another sequence

A sequence is a series of commands. These commands are executed in order whenever the sequence is run. Immediate commands cannot be stored in a sequence, just as they cannot be stored in the command buffer. Only buffered commands may be used in a sequence.

The JSI has 8,000 bytes of non-volatile memory to store 100 sequences. You can use the **XBS** command to determine how many bytes are available in the sequence buffer. The sequence buffers may have variable lengths, so you may have one long sequence or several short ones, as long as the total length does not exceed the 8,000 bytes of allocated space.

The commands that you enter to define a sequence are presented vertically in the examples below. This was done to help you read and understand the commands. When you are actually typing these commands into your terminal, they will be displayed horizontally.

To begin the definition of a sequence, enter the Define Sequence (**XD**) command immediately followed by sequence identifier number (1 to 100) and a delimiter. The Terminate Sequence (**XT**) command ends the sequence definition. All commands that you enter after the **XD** command and before the **XT** command will be executed when the sequence is run. An example is provided below.

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

Command	Description
> XE1	Erases Sequence 1
> XD1	Begins definition of Sequence 1
A2	Sets acceleration to 2 rps^2
V1Ø	Sets velocity to 10 rps
D5ØØØ	Sets distance to 5000 steps
G	Executes the move (Go)
H	Reverses direction
G	Executes the move (Go)
XT	Ends definition of sequence
> XR1	Runs Sequence 1

You can run a sequence by entering the **XR** command immediately followed by a sequence identifier number (1 to 100) and a delimiter. If you have enabled the Continuous Sequence Scan mode (**SSJ1**), you can also execute a sequence by switching the Sequence Select inputs, after you define the inputs as sequence-select inputs with the Configure Inputs (**IN**) command.

Once you define a sequence, it cannot be redefined until you delete it. You can delete a sequence by entering the **XE** command immediately followed by a sequence identifier (1 to 100) and a delimiter. You may then redefine that sequence.

Sequence 100 (if you have defined it) is always run when you power up the system or when you reset the indexer with the Reset (**Z**) command. For convenience, you may find it advantageous to place all of your set-up commands in Sequence 100.

Sequences that you define are automatically saved into the JSI's non-volatile memory. The only way to erase these sequences is by using the Erase Sequence (**XE**) command.

Selecting Sequences

After you define the sequences from the RS-232C interface, you can execute the sequences by using one of the following modes of operation:

- Stand-alone: Use thumbwheel switches to select and run the sequence.
- Computer Interface: Use the Run Sequence (**XR**) command to run the sequences.
- PLC (Programmable Logic Controller): Use the sequence select inputs to run a sequence.

Stand-alone Operation

This section explains and provides examples of how to store programs and run them with remote switches, and run them automatically when you power up the system. First, you will need to enter the programs into the JSI. You will need a terminal or a computer with RS-232C communication capabilities for programming the JSI controller.

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

Power-Up Sequence Execution

You can program the JSI to execute a sequence of commands on power up (sequences can be used as subroutines).

Sequence 100 always runs on power up. To run another sequence on power up, put an **XR**<num> (or **XG**<num>) at the end of sequence 100. If sequence 100 is empty, nothing happens on power up. Refer to Chapter 5, Software Reference, for Detailed descriptions and syntax of the following commands.

Command	Description
> XE1ØØ	Erases Sequence 100
> XD1ØØ	Begins definition of sequence 100
LD3	Disables limits if they are not connected
A2	Sets acceleration to 2 rps^2
V5	Sets velocity to 5 rps
D125ØØ	Sets distance to 12,500 steps
G	Executes the move (Go)
XT	Ends sequence definition
> Z	Resets the controller and runs Sequence 100

A power-up sequence is typically used to store set-up or initialization parameters that your application requires. Some of these commands are listed below.

Command	Description
> SSJ1	Continuous Sequence Scan Mode
> SN	Scan time
> JA	Jog acceleration
> JVL	Jog velocity low
> JVH	Jog velocity high

You can put any commands that are not immediate (buffered) into Sequence 100 (if you want to executed them during power up).

Using Remote Jogs and Sequence

In some applications, you may want to move the motor manually. You can configure the JSI to allow you move the motor manually with the Configure Input (**IN**) command. You must define the jogging velocity with the Jog Velocity High (**JVH**) and Jog Velocity Low (**JVL**). You can define three different inputs for jogging: CW Jog input (**IN#J**), CCW Jog Input (**IN#K**), and Jog Speed Select High/Low (**IN#L**). You must also enable the jogging feature with the **OSE1** command. Once you set up all of these parameters, you can attach a switch to the jog inputs that you defined and perform jogging. (**#** represents digits 1 - 13, which you enter.)

The following example shows how you can define power-up Sequence 100 to set up jogging.

Step 1 Define a power up sequence.

Command	Description
> XE1ØØ	Erase sequence 100
> XD1ØØ	Define sequence 100
JA2	Set Jog Acceleration to 2 rps^2
OSE1	Enables Jog function
JVL.1	Sets low-speed jog velocity to 0.1 rps
JVH5	Sets high-speed jog velocity to 5 rps
IN1J	Sets **IN 1** as a CW jog input
IN2K	Sets **IN 2** as a CCW jog input
IN3L	Sets **IN 3** as a speed-select input
IN4B	Sets **IN4** as a sequence-select input
IN5B	Sets **IN5** as a sequence-select input
> XT	Ends Sequence Definition

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

Step 2 Define all sequences that your application may require.

Command	Description
> XE1	Erases Sequence 1
> XD1	Defines Sequence 1
A1	Sets acceleration to 1 rps^2
V2	Sets velocity to 2 rps
D1ØØØ	Sets distance to 1,000 steps CW
G	Executes the move (Go)
> XT	Ends sequence definition

Command	Description
> XE2	Erase Sequence 2
> XD2	Defines Sequence 2
A1	Sets acceleration to 1 rps^2
V2	Sets velocity to 2 rps
D-1ØØØ	Sets distance to 1,000 steps CCW
G	Executes the move (Go)
> XT	Ends sequence definition

Step 3 Reset the JSI controller

Command	Description
> Z	Resets the JSI controller

Step 4 Turn on the **IN 1** input to move the motor in the CW direction at 0.1 rps (until you turn off **IN 1**).

Step 5 Turn on the **IN 2** input to move the motor in the CCW direction at 0.1 rps (until you turn off **IN 2**).

Step 6 Turn on the **IN 3** input to switch to high-speed jogging.

Step 7 Repeat steps 4 and 5 to perform high-speed jogging.

Step 7 Enter the **XR1** command to run the motor 1,000 steps CW.

Step 7 Enter the **XR2** command to run the motor 1,000 steps CCW.

Selecting Sequences with Thumbwheels

The following example shows how the JSI is typically used with remote thumbwheels. In this example only 3 sequences are entered. As many as 100 sequences may be defined and up to 99 may be executed using remote thumbwheels. Sequence 100 is automatically executed during power up or reset (refer to the Reset (**Z**) command.

You must install your thumbwheels as shown in Figure 4-2.

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

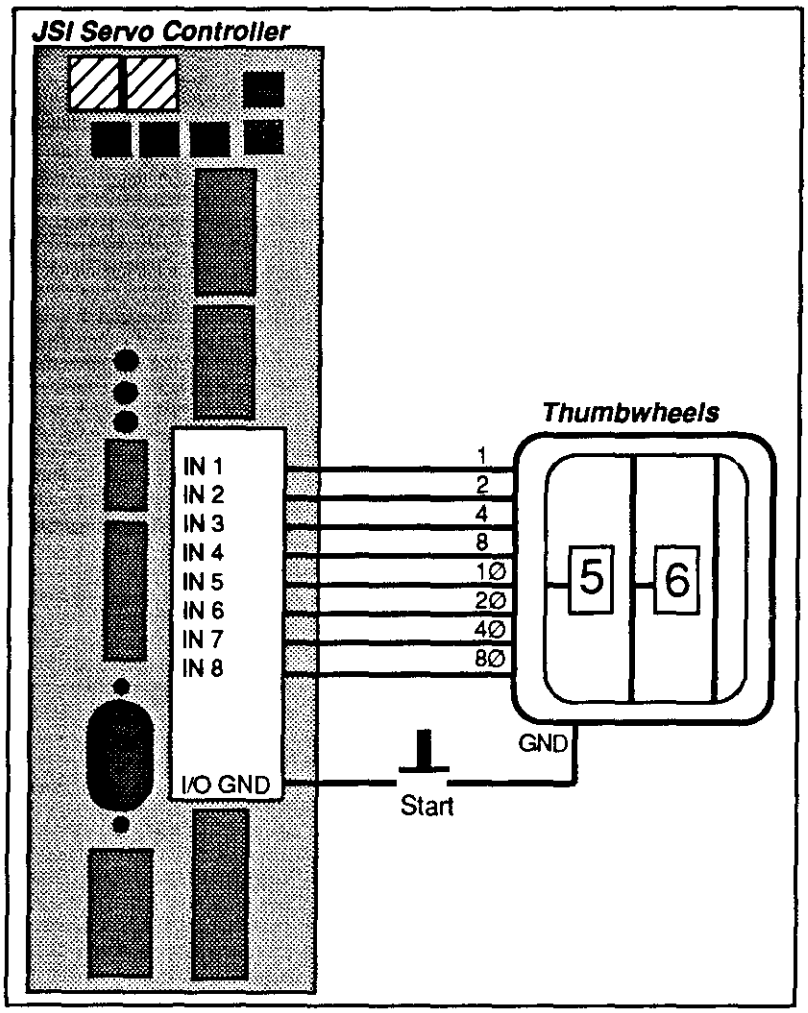


Figure 4-2. JSI Thumbwheel Installation

Step 1 Define a power-up sequence. Set up inputs **IN 1** - **IN 8** as sequence-select inputs using the Configure Input (**IN**) commands. The **IN1** command will be the least significant bit of the 1's (ones) digit. The **IN8** command will be the most significant digit of the 10's (tens) digit. Refer to Table 4-1.

Command	Description
> XE1ØØ	Erases sequence 100
> XD1ØØ	Defines sequence 100
IN1B	Sets up Input 1 as the sequence input
IN2B	Sets up Input 2 as the sequence input
IN3B	Sets up Input 3 as the sequence input
IN4B	Sets up Input 4 as the sequence input
IN5B	Sets up Input 5 as the sequence input
IN6B	Sets up Input 6 as the sequence input
IN7B	Sets up Input 7 as the sequence input
IN8B	Sets up Input 8 as the sequence input
INLØ	Sets active input level to active low (ON = 0V applied to the input)
SSJ1	Set JSI to execute programs from remote interface
SN1ØØ	Set scan time to 100 msec
XQ1	Enable sequence hold
> XT	Ends sequence definition

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

The BCD values of inputs **IN 1** - **IN 8** are shown in Table 4-1.

Inputs	IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
BCD Value	80	40	20	10	8	4	2	1
	10's Digit				*1's Digit*			

Table 4-1. Input BCD Values

Step 2 Define any any sequences that your application may need.

Command	Description
> XE1	Erases Sequence 1
> XD1	Starts Sequence 1 definition
MN	Sets mode to normal
A2	Sets acceleration to 2 rps
V5	Sets velocity to 5 rps
D25ØØØ	Sets distance to 25,000 steps
G	Executes the move (Go)
> XT	Ends sequence definition

Command	Description
> XE5	Erases Sequence 5
> XD5	Starts Sequence 5 definition
MN	Sets mode to normal
A2	Sets acceleration to 2 rps^2
V5	Sets velocity to 5 rps
D1ØØØØ	Sets distance to 10,000 steps
G	Executes the move (Go)
> XT	Ends sequence definition

Command	Description
> XE99	Erases Sequence 99
> XD99	Starts Sequence 99 definition
MN	Sets mode to normal
A2	Sets acceleration to 2 rps^2
V5	Sets velocity to 5 rps
D-35ØØØ	Sets distance to -35,000 steps
G	Executes the move (Go)
> XT	Ends sequence definition

Step 3 Connect the thumbwheels to the **IN 1** - **IN 8** inputs as shown in Figure 4-2.

Step 4 Reset the JSI controller. This will prepare the power up (Sequence 100) to be executed when you apply power to the controller.

Command	Description
> Z	Resets the JSI controller

Step 5 Set your thumbwheel to 1 and push start to move the motor 25,000 steps in the positive direction.

Step 6 Set your thumbwheel to 5 and push start to move the motor 10,000 steps in the positive direction.

Step 7 Set your thumbwheel to 99 and push start to move the motor 35,000 steps in the negative direction.

Note: If you select (with the thumbwheel) an invalid or unprogrammed sequence, nothing will happen.

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

Host Computer Operation

This section assumes that you have successfully communicated with the JSI controller. If you have not verified that your communications link is functioning properly, return to Chapter 2, *Getting Started*, and complete this check before attempting to complete the next section. (*Be sure that your communications parameters are set as specified in Chapter 2, Getting Started.*) The following program example runs one JSI controller to make two different moves.

```
1 '                    JSI.BAS PROGRAM
2 '
3 ' ********************************************************************************************
4 ' *                                                                                          *
5 ' * This program controls the RS232 Communication line to execute 2 different moves         *
6 ' * using the JSI.                                                                           *
7 ' *                                                                                          *
********************************************************************************************
15 OPEN "COM1:9600,N,8,1,RS,CS,DS,CD" AS #1        ' Open Communication port
20 V$ = "": Q$ = "": ECHO$ = "": LF$ = "":         ' Initialize variables
90 CLS
100 LOCATE 12,15
105 PRINT " PRESS ANY KEY TO START THE PROGRAM "
107 V$ = INKEY$: IF LEN(V$) = 0 THEN 100           ' Wait for input from user
120 Z$ = "K K LD3"                                 ' Reset and disable limits on the JSI
122 PRINT #1,Z$;
124 Q$ = INPUT$(8,1)
900 ' ********************************************************************************************
901 ' *                                                                                          *
902 ' *  Lines 1000-1060 sends a move down to the first JSI.  Computer waits for the Line        *
903 ' *  Feed from the JSI indicating that the motor has finished its move.  Computer            *
904 ' *  will not command second JSI to move until the first move is completed.                  *
905 ' *                                                                                          *
********************************************************************************************
1000 MOVE1$ = "MN A1 V2 D40000 G 1LF "             ' Define move 1
1005 CLS
1007 LOCATE 12,15: PRINT " DOING MOVE 1 "
1010 PRINT #1,MOVE1$                               ' Perform move 1.
1015 ECHO$ = INPUT$(23,1)                          ' Read echoes from JSI.
1020 LF$ = INPUT$(1,1)                             ' Wait for line feed from JSI
1040 IF LF$ <> CHR$(10) GOTO 1020                  ' indicating end of move.
1045 CLS
1047 LOCATE 12,15
1050 PRINT "MOVE 1 DONE"                           ' Let user know that move 1 is done
1060 LOCATE 15,15: PRINT " PRESS ANY KEY TO GO ON TO SECOND MOVE "
1070 V$ = INKEY$: IF LEN(V$) = 0 THEN 1060
1900 '
********************************************************************************************
1901 ' *                                                                                      *
1902 ' * After axis one is done, we request that you hit any key to go on to second move.    *
1903 ' * In real application, we would expect you to go ahead with the process and work on   *
1904 ' * on the part before going on to next move.  (i.e., Activate a punch)                  *
1905 ' * Now that first move is finished go on to move 2.  The JSI also prints a line feed   *
1906 ' * after finishing the second move.                                                    *
1907 ' * As soon as the computer receives the line feed from JSI, the program will go back   *
1908 ' * to the first move.                                                                  *
1909 ' *                                                                                      *
********************************************************************************************
2000 MOVE2$ = "H G 1LF "
2005 CLS
2007 LOCATE 12,15: PRINT " DOING MOVE 2 "
2010 PRINT #1,MOVE2$
2015 ECHO$ = INPUT$(8,1)
2020 LF$ = INPUT$(1,1)
2040 IF LF$ <> CHR$(10) GOTO 2020
2045 CLS
2047 LOCATE 12,15
2050 PRINT "MOVE 2 DONE "
2060 FOR I = 1 TO 1000: NEXT I
2070 GOTO 20                                       ' Go back to beginning of program.
```

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

Multi-Axis Control (Daisy-Chaining)

You may daisy chain up to 16 JSI controllers. Individual drive addresses are set with the JSI's push buttons. When daisy-chained, the units may be addressed individually or simultaneously. You should establish a unique device address for each JSI. Refer to Figure 4-4 for JSI daisy chain wiring information.

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

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

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

No JSI executes a device-specific command unless the unit number specified with the command matches the JSI's unit number. Device-specific commands include both buffered and immediate commands.

You must use status-request commands in an orderly fashion. Commands should only be issued when the host is ready to read the response. You should not send new commands until you receive a response from the previous status-request command. In particular, you should not issue a immediate-status command until the host receives a buffered command status response. If this is not followed, the command responses will be intertwined, rendering the information useless.

If you enable the Interactive mode (**SSI1**), only the JSI that is set to Address 1 will respond with the **>**. This prevents all the JSIs from sending out **>** in a daisy chain. Typically, you should disable the Interactive mode when you use a host computer with the JSI controller.

JSI Daisy-Chain Wiring

Figure 4-3 illustrates how JSIs might be wired in a daisy-chain configuration.

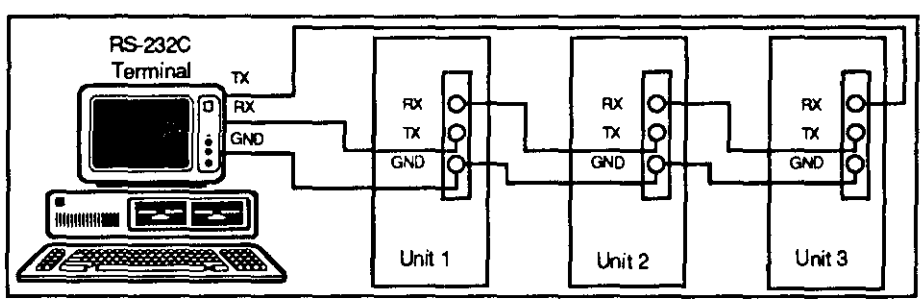


Figure 4-3. Daisy-Chained JSIs

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

Sample Applications and Commands

Example: Three indexers are on an RS-232C daisy-chain. Send the following commands:

Command	Description
> MN	Sets unit to Preset mode
> A5	Sets acceleration to 5 rps^2 for all three controllers
> V1Ø	Sets velocity to 5 rps for all three controllers
> LD3	Disables limits (if they are not connected)
> 1D25ØØØ	Sets Axis 1 distance to 25,000 steps
> 2D5ØØØØ	Sets Axis 2 distance to 50,000 steps
> 3D1ØØØØØ	Sets Axis 3 distance to 100,000 steps
> G	Moves all axes.

Unit 1 moves 25,000 steps, unit 2 moves 50,000 steps, and unit 3 moves 100,000 steps. All three units use the same acceleration and velocity rates. Units 1 and 2 move at about the same time.

Multi-Axis Interface Program Example

The following program is very similar to JSI.BAS, except this program controls two JSIs on a daisy-chain. This program assumes the device address of two JSIs to be #1 and #2 respectively. The program does the following.

1. Executes the first move upon user input
2. Waits for a line feed from the JSI controller, which indicates the end of the move.
3. Executes the second move upon user input.
4. Waits for a line feed from the JSI controller, which indicates the end of the move. It then begins the process again.

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

```
1 '                    JSI-2.BAS PROGRAM
2 '
3 ' ********************************************************************************************
4 ' *                                                                                          *
5 ' * This program controls the RS232 Communication line to execute 2 different moves using *
6 ' * two JSI drives.                                                                          *
7 ' *                                                                                          *
********************************************************************************************
15 OPEN "COM2:9600,N,8,1,RS,CS,DS,CD" AS #1        ' Open Communication port
20 V$ = "": Q$ = "": ECHO$ = "": LF$ = "":         ' Initialize variables
90 CLS
100 LOCATE 12,15
105 PRINT " PRESS ANY KEY TO START THE PROGRAM "
107 V$ = INKEY$: IF LEN(V$) = 0 THEN 100           ' Wait for input from user
120 Z$ = "K LD3"                                   ' Reset the JSI controller
122 PRINT #1,Z$;
124 Q$ = INPUT$(8,1)
900 ' ********************************************************************************************
901 ' *                                                                                          *
902 ' *  Line 1000-1060 sends a move down to the first JSI.  Computer                            *
903 ' *  waits for the Line Feed from the JSI indicating that the motor                          *
904 ' *  has finished its move.   Computer will not command second JSI                           *
905 ' *  until axis 1 has completed its move.                                                    *
906 ' *                                                                                          *
********************************************************************************************
1000 MOVE1$ = "1A1 1V2 1D40000 1G 1LF " ' Define move for Axis 1
1005 CLS
1007 LOCATE 12,15: PRINT " MOVING AXIS 1 "
1010 PRINT #1,MOVE1$                               ' Move axis 1.
1015 ECHO$ = INPUT$(23,1)                          ' Read echoes from JSI
1020 LF$ = INPUT$(1,1)                             ' Wait for line feed from JSI
1040 IF LF$ <> CHR$(10) GOTO 1020                  ' indicating end of move.
1045 CLS
1047 LOCATE 12,15
1050 PRINT "AXIS 1 FINISHED ITS MOVE "             ' Let user know axis 1 done
1060 LOCATE 15,15: PRINT " PRESS ANY KEY TO MOVE SECOND AXIS "
1070 V$ = INKEY$: IF LEN(V$) = 0 THEN 1060
1900 '
********************************************************************************************
1901 ' *                                                                                          *
1902 ' * After axis one is done, we request that you press any key to go on                       *
1903 ' * to second move.  In real application, we would expect you to                            *
1904 ' * go ahead with the process and work on the part before going on to                       *
1905 ' * next move.  (i.e.,  Activate a punch)                                                   *
1906 ' *                                                                                          *
1907 ' * Now that first axis finished its move, we go on to move axis 2.                         *
1908 ' * Second JSI also prints a line feed after finishing the move.                           *
1909 ' * As soon as computer receives the line feed from JSI, program will                       *
1910 ' * go back to the first move.                                                              *
1911 ' *                                                                                          *
********************************************************************************************
2000 MOVE2$ = "2A1 2V2 2D10000 2G 2LF "
2005 CLS
2007 LOCATE 12,15: PRINT " AXIS 2 MOVING "
2010 PRINT #1,MOVE2$
2015 ECHO$ = INPUT$(23,1)
2020 LF$ = INPUT$(1,1)
2040 IF LF$ <> CHR$(10) GOTO 2020
2045 CLS
2047 LOCATE 12,15
2050 PRINT "AXIS 2 FINISHED ITS MOVE "
2060 FOR I = 1 TO 1000: NEXT I
2070 GOTO 20                                       ' Go back to beginning of program.
```

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

PLC Operation

You can use a PLC to execute 99 different sequences that are stored in the JSI controller. You can configure up to eight inputs as sequence-select inputs. This capability allows you to use eight outputs from the PLC to select as many as 99 sequences. You can accomplish this by changing the BCD values of the PLC outputs.

Scanning for Sequence Execution

Changing the BCD values of sequence input lines results in a new sequence being run that corresponds to the new value. The sum of the values issued determines which sequence the indexer will run. For example, if you set up inputs 1 - 8 as sequence-select inputs, the lowest input (Input 1)will have the least significant value and the highest input (Input 8) will have the most significant value. Refer to Table 4-1.

You must use the Input (**IN**) command to set up the inputs as sequence-select inputs. You must also configure the JSI controller to read sequences via BCD input (use the **SSJ1** command).

After you issue the **SSJ1** command, the JSI controller scans the sequence select inputs to find the sequence that you have specified for execution. After the system completes the execution of the desired sequence, the controller scans the sequence inputs again to find the next sequence to be executed. If you use the Interrupted Run Mode (**XQ1**) command. In the Interactive Run mode (**XQ1**), the JSI controller waits until all of the sequence inputs are turned off before selecting the next sequence to execute. Use the Input Level (**INL**) command to set up the controller's active level.

The Scan (**SN**) command determines how long the sequence-select input must be maintained before the controller executes the program. This is a debounce time.

Sample Applications and Commands

This section provides step-by-step procedures to run sequences from your PLC. First, you need to enter the programs into the JSI. You will need a terminal or a computer with RS-232C communication capability. You need to define the sequences before you can execute them with your PLC's BCD outputs. The JSI controller automatically saves these sequences (battery-backed-up system).

Using a terminal or a computer, key in the following commands. *The commands that you enter to define a sequence are presented vertically in the example below. This was done to help you read and understand the commands. When you are actually typing these commands into your terminal, they will be displayed horizontally.*

Step 1

Define a power-up sequence. *Every time you power up the JSI controller, it executes these commands and enables the JSI to read up to 99 sequences from the sequence-select inputs.*

Command	Definition
> XE1ØØ	Erases sequence 100
> XD1ØØ	Defines sequence 100
SSJ1	Executes sequences via PLC input
SN2Ø	Sets scan time to 20 msec
XQ1	Sets JSI to interrupted run mode
A1Ø	Sets acceleration to 10 rps^2
V2	Sets velocity to 2 rps
IN1B	Sets up Input 1 as a sequence-select input
IN2B	Sets up Input 2 as a sequence-select input
IN3B	Sets up Input 3 as a sequence-select input
IN4B	Sets up Input 4 as a sequence-select input
IN5B	Sets up Input 5 as a sequence-select input
IN6B	Sets up Input 6 as a sequence-select input
IN7B	Sets up Input 7 as a sequence-select input
IN8B	Sets up Input 8 as a sequence-select input
OUT1C	Sets up Output 1 as sequence-in-progress output
LD3	Disables the limits (if they are not connected)
> XT	Ends the sequence definition

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

Step 2 Define any sequences that your application may require.

Command	Description
> XE1	Erases Sequence 1
> XD1	Defines Sequence 1
D2ØØØ	Sets distance to 2,000 steps
G	Executes the move (Go)
> XT	Ends Sequence 1 definition

Command	Description
> XE2	Erases Sequence 2
> XD2	Defines Sequence 2
D4ØØØ	Sets distance to 4,000 steps
G	Executes the move (Go)
> XT	Ends Sequence 2 definition

Command	Description
> XE3	Erases Sequence 3
> XD3	Defines Sequence 3
D8ØØØ	Sets distance to 8,000 steps
G	Executes the move (Go)
> XT	Ends Sequence 1 definition

Command	Description
> XE99	Erases Sequence 99
> XD99	Defines Sequence 99
D-14ØØØ	Sets distance to -14,000 steps
G	Executes the move (Go)
> XT	Ends Sequence 99 definition

Step 3 Verify that your programs were stored properly by uploading each entered sequence (**XU**). If you receive responses that differ from what you programmed, re-enter those sequences.

Step 4 Run each program from the RS-232C interface with the Run Sequence (**XR**) command. Make sure that the motor moves the distance that you specify.

Step 5 Assuming your PLC accepts open-collector outputs connect the inputs and outputs as shown in Figure 4-4. If not, you will need to add pull up resistors to the outputs.

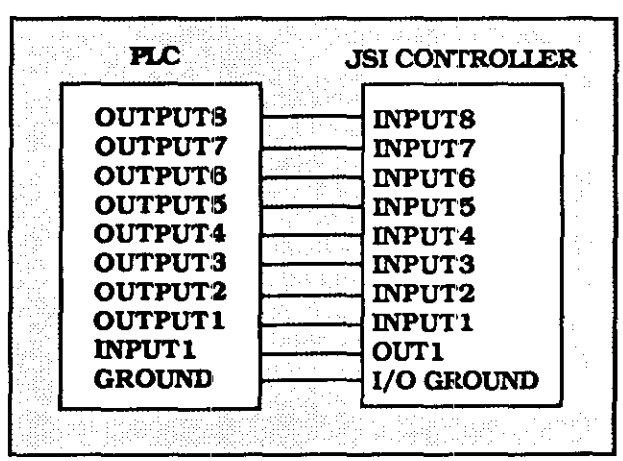


Figure 4-4. PLC Connection

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

Step 6 Refer to the user guide that accompanied you PLC unit to turn on the proper combination of outputs to execute one of the four sequences programmed and stored in the JSI controller. You should program the PLC to search for sequence completion output from the JSI controller before the controller begins searching for the next sequence to execute (this type of interaction is performed through a *handshake* between the JSI and the PLC—the signal transmitted indicates that another sequence is ready to be executed.

- Turning on Output 1 only will execute Sequence 1.
- Turning on only the PLC's Output 2 will execute Sequence 2.
- Turning on only the PLC's Outputs 1 and 2 will execute Sequence 3.
- Turning on only the PLC's Outputs 1, 4, 5, and 8 will execute Sequence 99.

Step 7 Cycle down power to the JSI. The system will execute Sequence 100.

Step 8 Turn on the appropriate sequence-select input [set for the least Scan Time(**SN**)] to execute the proper sequences. Since the sequence hold feature (refer to the **XQ1** command) was enabled during the power-up sequence, your PLC program must turn off all of the sequence-select inputs before you can select another sequence.

Tuning The JSI

Before you begin to tune the JSI, you should ensure that the analog servo valve or drive will exceed the maximum speed desired when the maximum analog output from the JSI is applied. The JSI can command ±10V or ±200mA. If this is not enough to exceed the maximum speed desired, the system will not be able to provide the desired performance.

The most important aspect of a servo system is setting the controller's gains (described in the previous sections). The controller in a conventional continuous-time (i.e., analog) servo system is an active or passive filter network, usually an op-amp with resistive and capacitive feedback. The values and configuration of the resistors and capacitors in the filter network are determined by the compensation necessary to stabilize the servo system.

The values of the resistors and capacitors can be thought of as being the gains of the controller. To stabilize the servo system, you must adjust the gains (potentiometers) to match the system being controlled (the system being the motor, its load, and the amplifier). In the case of the JSI, the gains of the controller are the constant coefficients of the recursive equation (software). The form of the recursive equation determines how many of these gains must be adjusted in order to stabilize the system.

Fortunately, there are only three user-changeable values in the equation and, better yet, it is not necessary to know the equation to tune a JSI servo system. Tuning a JSI servo system usually requires adjusting only one controller gain. The other gains are predefined and, in most cases, require no further adjustment.

There are two methods available to adjust a JSI's servo compensation network.

- The five pushbuttons on the JSI's front panel
- The RS-232C communications port

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

PushButton Tuning

The JSI has five pushbuttons on the front panel that provide a simple push-button method of fine-tuning the system's performance to a specific attached load. The JSI system is factory preset with only a low proportional gain (for safety reasons). You must tune the JSI to achieve desired performance levels.

Once you have the system installed and the motor connected to its intended load, you can determine whether any fine tuning is required by observing the response of the system to commands from your indexer and by observing how stiff the system is when at rest.

When the motor is at rest, try to deflect the shaft. You should not be able to easily turn the shaft away from its rest position. If it feels very soft, the system gains probably need to be increased. A *soft* system will not respond very quickly to move commands. If the shaft feels stiff, be certain to check that the system is not vibrating. If it is vibrating, the gain may be too high. In extreme cases, the vibration grows in amplitude, producing violent motions that cause the drive to fault or break something. For this reason, you should tune the JSI and servo with some caution.

Select one of the P, I, or D buttons and hold it down. When you do, the two-digit display lights up to indicate the present value for the term you have selected from (0 - 99%). You should note this value in case you wish to return to it. While holding the button for the term you selected, push the UP button to increase the value of the term. Push the DOWN button to decrease the value of the term. You must hold a button down while using the UP and DOWN buttons. If you do not select one of the term buttons, the display will not be lighted, and no value changes occur from holding the UP and DOWN buttons.

The value that will be used is the one displayed when you release the UP or DOWN button. This value will be used by the JSI immediately. Push-button tuning allows rapid adjustments to be made to tuning parameters.

To store values in non-volatile memory, push the PROP GAIN, INT GAIN, DER GAIN and DN buttons. If you ever wish to retrieve the factory default values, all you need to do is push both the P, I, D and UP term buttons together (refer to Figure 4-5). This causes the factory values for each term to be reloaded. Table 4-2 provides a complete listing of the JSI's push-button tuning functions.

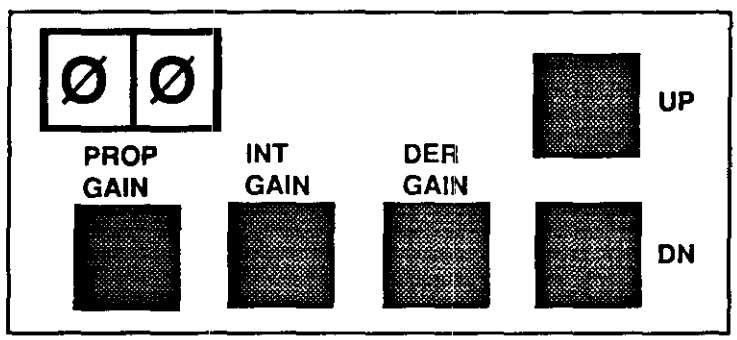


Figure 4-5. JSI Controller Push Buttons

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

Set integral gain: *Press I to show values. Press I and UP or DN to increase/decrease the percent gain.*

Set proportional gain: *Press P to show values. Press P and UP or DN to increase/decrease the percent gain.*

Set derivative gain: *Press D to show values. Press D and UP or DN to increase/decrease the percent gain.*

Set device address: *Press P and D together to show values. Press P and D and UP or DN to increase/decrease the device address.*

Set baud rate: *Press P and I together to show values. Press P and I and UP or DN to increase/decrease the first two digits of the baud rate.*

PROP GAIN	INT GAIN	DER GAIN	UP	DOWN	Function
Ø	Ø	Ø	Ø	Ø	Display Errors
X	Ø	Ø	Ø	Ø	Display Proportional Gain
X	Ø	Ø	X	Ø	Increase Proportional Gain
X	Ø	Ø	Ø	X	Decrease Proportional Gain
Ø	X	Ø	Ø	Ø	Display Integral Gain
Ø	X	Ø	X	Ø	Increase Integral Gain
X	X	Ø	Ø	X	Decrease Integral Gain
Ø	Ø	X	Ø	Ø	Display Derivative Gain
Ø	Ø	X	X	Ø	Increase Derivative Gain
Ø	Ø	X	Ø	X	Decrease Derivative Gain
X	X	Ø	Ø	Ø	Display RS-232 Baud Rate
X	X	Ø	X	Ø	Increase RS-232 Baud Rate
X	Ø	X	Ø	X	Decrease RS-232 Baud Rate
X	Ø	X	Ø	Ø	Display RS-232 Device Address
X	Ø	X	X	Ø	Increase RS-232 Device Address
X	Ø	X	Ø	X	Decrease RS-232 Device Address
Ø	X	X	Ø	Ø	Reserved
Ø	X	X	X	Ø	Reserved
Ø	X	X	Ø	X	Reserved
X	X	X	Ø	Ø	Save Values to Non-volatile Memory
X	X	X	X	Ø	Return to Factory Settings
Ø	Ø	Ø	X	X	Reset
X	X	X	X	Ø	Display Software Rev ision Level

Table 4-2. JSI PushButton Tuning Functions

Miscellaneous Functions

Return to factory settings: P, I, D, and UP together
Save tuning values to non-volatile memory: P, I, D, and DN together

- Reset: UP and DN together
- Display Software revision level: P, I, and D together

General Tuning Considerations

You should remember that all of the terms are interactive and it may take considerable experimentation to find the exact combination of values to get the best performance in your application. In a very general sense, the proportional term should be set first, followed by the integral term, and finally the derivative term.

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

Proportional Gain

Proportional gain affects system stiffness and accuracy. As the proportional gain is adjusted higher, the influence of the error between the commanded position and the actual feedback position becomes greater. If the gain is sufficiently high, the system will oscillate. This happens because very small position changes are amplified into very large error signals, and the inertia of the servo and load will not allow the system to follow the electronic commands fast enough. The system lag time eventually reaches a point where the feedback and the command signals are in phase. At this point, oscillation results. When this occurs, you must reduce the gain below the oscillation point.

Integral Gain

Integral gain allows the system to compensate for position errors (due, for example, to friction). Integral gain reduces velocity ripple. It does this by slowing down the electronic response time so that it more closely resembles the response of the mechanical components of the loop. The integral gain factor is applied to a summation of all errors that occur from the time zero (when you first enable the command output **CMD+** & **CMD-**). Hence, this gain reduces the steady-state error in the system.

Derivative Gain

Derivative gain adds damping effects to the system. In the case of a system oscillating at the end of a move, or around a change in velocity, increasing the derivative gain will reduce the oscillation. The derivative gain adds phase lead to compensate for the systems natural phase lag. The derivative gain is applied to the rate of change of the position error. Thus, if a constant error occurs, the derivative gain will not affect the system. A Changing position error will allow the derivative gain to affect the system.

Position Errors Due to Load Inertia

Load is the inertia (mass times the radius of rotation squared) seen by the shaft of the motor measured in (Oz-In-In). The amount of inertia affects the torque required. The torque required is a function of acceleration and inertia. If you find that your load inertia is larger than 10 times the motor inertia, you may want to trade some speed performance for accuracy at the final position. In this case, increase the integral gain.

Slow Response Due to Load Inertia

In a system where the load inertia is larger than the factory setting allowed, it may be possible to increase the system response time by allowing the overshoot to increase. In this case, increase the proportional gain, and decrease the integral gain.

Position Errors Due to Friction

Significant friction in the load may cause the end-of-move position to be unacceptably in error. In this case, you can trade off system response for final positional accuracy. To do this, increase the integral gain, and increase the derivative gain.

Shaft and Coupling Vibration

If the load is coupled to the shaft through a non-rigid coupler, it is possible for the shaft and coupler to oscillate at a frequency greater than would be possible for the system as a whole. In this case, it may be possible to trade system response for system stability. To do this increase the integral and derivative gains, while decreasing the proportional gain.

Inadequate Response Time (Frequency Response)

Some systems may involve very little inertia, but need the torque for very high acceleration. In this case, you may want to narrow the range of values that the system can handle to optimize the move profile for a light load. To do this, increase the proportional gain while decreasing the integral term.

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

Overshoot If the system exhibits excessive overshoot (when making the transition from acceleration to constant velocity or from deceleration to a stop), it may be caused by a commanded acceleration that is much higher than what the system can actually produce. If this occurs, reduce the commanded acceleration [refer to the Acceleration (**A**) command in Chapter 5, Software Reference] until this overshoot is minimized. Reducing the integral gain will also limit overshoot in most cases.

Integral Limit The integral portion of the PID control algorithm provides an additional factor for tuning purposes. The integrator portion of the control is the integral gain of the position error from the time you enable the servo to the present time. Digitally, this is a summation of all the position errors from each sample period from the time you enable the servo up to the present time. You can limit the maximum value that the summation of errors can achieve with the Define Integral Sum Maximum Sum Limit (**CIL**) command. If you prevent the summation of the errors to become too large, you can reduce the overshoot that is caused by the integral gain *and* achieve steady-state accuracy.

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

Chapter 5. Software Reference

Chapter Objectives

The information in this chapter will enable you to:

- Identify the four 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

Definition & Description of Command Types

Command Syntax

NAME: The name of the command is shown here.

SYNTAX: The proper syntax for the command is shown here. The specific parameters that are associated with the command are also shown. Definitions of the parameters are shown below.

- a This indicates that a device address must accompany the command. Only the device specified by this parameter will receive and execute the command. If this parameter , or any other parameter appears within brackets <>, providing a value for the parameter is optional.
- n An integer (n) may be used to specify a variety of values (acceleration, velocity, etc.)
- x This represents a letter that you must supply to distinguish which command function you want to apply.
- x, y This represents a letter that you must supply to distinguish which command functions you want to apply.
- s This indicates that a sign character, either positive or negative (+ or -) is required.

DEFAULT: The factory default setting

RANGE: The range of values for the parameter(s)

VALID: Revision history. The revision of software when the command was added or modified.

RESPONSE: In this field, a sample of the command is provided and followed by its response. The response includes the number of characters returned, their format, and any delimiters.

TYPE:

Motion: Motion commands affect motor motion. For example, accelerate, velocity, distance, go home, stop, direction, mode, etc.

Status: Status commands respond (report back) with information.

Setup: Setup commands define setup conditions for the application. Setup commands include the following types of commands:

- Homing (Go home acceleration and velocity, etc.)
- Input/Output (Limits, scan time, in-position time, etc.)
- Tuning (Servo or position tracking)
- General (Set switches, return to factory settings, etc.)

Programming: Programming commands affect programming and program flow. For example, trigger, output, all sequence commands, quote, time delays, pause and continue, enable and front-panel, loop and end loop, line feed, carriage return, and backspace.

ATTRIBUTES: Immediate or Buffered

- Device Address Optional, Device Address Required, or Never Saved
- Savable in Sequence or Independently Saved

DESCRIPTION: A description of the command is provided here. The description defines the command's purpose and explains when you might use it.

SEE ALSO: Commands that are related or similar to the command described are listed here.

EXAMPLES: An example of how you might use the command is provided.

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

# #

NAME: Single Step
SYNTAX: <a>#
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: This command controls he execution of a sequence when the **XST** (Single Step Mode) is enabled. Each time you enter the **#** command followed by a delimiter (carriage return or space), one command in the sequence buffer will be executed. You can run in Single Step mode only if you have RS-232C interface connected to a host. If you issue a Kill (**K**) command, while you are in Single Step mode, the sequence execution will be aborted, but the Single Step mode is retained. When you cycle power, the controller will no longer be in single step mode.

SEE ALSO: **XST**
EXAMPLE:

Command	Description
> **XE1**	Erases sequence 1
> **XD1**	Defines sequence 1
> **A5**	Sets acceleration to 5 rps^2
> **V2**	Sets velocity to 2 rps
> **D1ØØØØ**	Sets distance to 10,000 steps
> **G**	Executes the move (Go)
> **XT**	Ends sequence definition
> **XST1**	Enables Single Step mode
> **XR1**	Executes Sequence 1
> **XTR1**	Enables Trace mode
> **#**	Execute first command

\# Execute the first command
*SEQUENCE_ØØ1_COMMAND_A5
Displays the first command executed.

\# Execute the second command
*SEQUENCE_ØØ1_COMMAND_V2
Displays the second command executed.

\# Execute the third command
*SEQUENCE_ØØ1_COMMAND_D1ØØØØ
Displays the third command executed.

Execute the fourth command.
*SEQUENCE_ØØ1_COMMAND_G
Displays the fourth command executed. Motor should have moved 10,000 steps.

Execute the fifth command
*SEQUENCE_ØØ1_COMMAND_XT
Displays the last command .

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

A

NAME: Acceleration
SYNTAX: <a>**An**
DEFAULT: n = 10 rps^2
RANGE: n= 0.001 to 9999.999 rps^2
VALID:
RESPONSE: a**A** ***An**[cr]
TYPE: Motion
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
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 and is stored in nonvolatile memory. You do not need to reissue this command for subsequent Go (**G**) commands.

SEE ALSO: **D, V, G**
EXAMPLES:

Command	Description
> **MN**	Sets the moves to mode normal (preset moves)
> **A5**	Sets Acceleration to 10 rps^2
> **V1Ø**	Sets Velocity to 10 rps
> **D1ØØØØ**	Sets Distance to 10,000 steps
> **G**	Executes the move (Go)

B

NAME: Buffer Status Report
SYNTAX: a**B**
DEFAULT n = R
RANGE: n = B, R
VALID:
RESPONSE: a**B** *n[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: The buffer status command reports the status of the command buffer. If the command buffer is empty or less than 90% full, the controller will respond with a *R[cr]. A *B[cr] response will be issued if less than 10% of the command buffer is free. The command buffer is 2,000 bytes long.

This command is commonly used when a long series of commands will be loaded remotely. If the buffer size is exceeded, the extra commands will never be received by the controller.

*R = More than 10% of the buffer is free
*B = Less than 10% of the buffer is free

SEE ALSO: **BS**
EXAMPLES:

Command	Response
> **1B**	***R** [cr] (more than 10% of the Buffer is free)

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

BCDB

NAME: Buffered Configure Dead Band
SYNTAX: <a>**BCDB**n
DEFAULT: n = 0
RANGE: n = 0 - 32,767
RESPONSE: a**BCDB** *SLIP_FAULT_DEADBAND=n_STEPS[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The buffered configure dead band command defines the dead band value. If a parameter follows the command, the value will become the new dead band value. An Output may be configured to indicate when the absolute value of the following error is outside of the dead band region (the **OUT** command). When the following error exceeds the dead band, the exceed dead band indicator output will be turned ON. When the following error is within the dead band region set with this command, the exceed dead band indicator output will be OFF.

SEE ALSO: **OUT, CDB**
EXAMPLES:

Command	Description
> A1Ø	Sets acceleration to 10 rps2
> V5	Sets velocity to 5 rps
> D5ØØØØ	Sets distance to 10,000 steps
> BCDB5Ø	Configure dead to 50 steps
> OUT1G	Configures Output 1 as Dead band Output Output 1 will go on when the Position Error exceeds the deadband Execute the Move (Go)

The Exceed Dead band Indicator output indicates whether or not the following error exceeds 50 motor steps

BCDG

NAME: Buffered Configure Differential Gain
SYNTAX: <a>**BCDG**n
DEFAULT: n = 0
RANGE: n = 0-99
RESPONSE: a**BCDG** *DIFFERENTIAL_GAIN=N_PERCENT [cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Independently Saved
DESCRIPTION: This command defines the differential gain to be used for tuning. When you tune the drive using the Buffered Differential Gain (**BCDB**) command, you are actually setting a percentage of the Differential Gain Maximum (**CDM**). The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **CDG, CDM, BCDM, SV, Tuning Section**
EXAMPLES:

Command	Description
> **A5**	Sets Acceleration to 5 rps^2
> **V1Ø**	Sets Velocity to 10 rps
> **D32ØØØ**	Sets Distance to 32,000 steps
> **BCDG44**	Sets the differential gain to 44 percent of the maximum value
> **G**	Executes the move (Go)
> **BCDG2Ø**	Sets the differential gain to 20 percent of the maximum value
> **G**	Executes the move (Go)

The derivative gain is set to 44 percent of the maximum value before the first 32,000 step move, and to 20 percent of the maximum value before the second 32,000-step move.

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

BCDM

NAME: Buffered Configure Differential Maximum
SYNTAX: <a>**BCDM**n
DEFAULT: n = 100
RANGE: n = 0 - 32,767
VALID:
RESPONSE: a**BCDM*** *DIFFERENTIAL_GAIN_MAXIMUM=N[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Independently Saved
DESCRIPTION: This command defines the maximum differential gain you may use for tuning purposes. If a valid number is entered, it will become the new maximum differential gain. This command is identical to **CDM** except that it is buffered. The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **CDM, BCDG, CDG, SV, Tuning Section**
EXAMPLE:

Command	Description
> **A5**	Sets acceleration to 5 rps^2
> **V1Ø**	Sets velocity to 10 rps
> **D32ØØØ**	Sets distance to 32,000 steps
> **BCDM15ØØØ**	Sets maximum derivative gain to 15,000
> **G**	Executes the Move (Go)
> **BCDM5ØØØ**	Sets derivative maximum gain to 5,000
> **G**	Executes the Move (Go)

The differential gain is set to 15,000 before the first 32,000-step move, and to 5,000 before the second 32,000-step move.

BCIG

NAME: Buffered Configure Integral Gain
SYNTAX: <a>**BCIG**n
DEFAULT: n = 0
RANGE: n = 0-99
VALID:
RESPONSE: a**BCIG** *INTEGRAL_GAIN_=n_PERCENT[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Independently Saved
DESCRIPTION: If you supply a value, the specified value will become the new value for the integral gain. A new value for the **BCIG** will be calculated using the entered value as a percent of maximum Integral value. The **CIG** command differs from the **BCIG** command only in that it can be executed sequentially within a sequence. The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **CIG, CIM, BCIM, SV, Tuning Section**
EXAMPLE:

Command	Description
> **A5**	Sets acceleration to 5 rps^2
> **V1Ø**	Sets velocity to 10 rps
> **D32ØØØ**	Sets distance to 32,000 steps
> **G**	Executes the move (Go)
> **BCIG5Ø**	Sets integral gain to 50% of maximum value
> **G**	Executes the move (Go)

The first 32,000-step move is made, then the integral gain is changed to 50 percent of maximum value before executing the second 32,000-step move.

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

BCIL

NAME: Buffered Configure Maximum Integral Sum Limit
SYNTAX: <a>**BCIL**n
DEFAULT: n = 65535
VALID: n = 0 - 214748347
RESPONSE: a**BCIL** *INTEGRATOR_LIMIT=n[cr]
TYPE: Setup
ATTRIBUTES: Buffered , Device Address Optional, Independently Saved
DESCRIPTION: This command defines the maximum integral sum limit. The value for the sum of the integral value will not be able to exceed the value set by the **BCIL** command. This command is useful if you want to move into a final position quickly. If you have a small **BCIL** value, the system will not allow the sum of the errors to exceed the n value resulting in the motor moving into position faster. If you make the **BCIL** value very large, the response of the final move will be slow. The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **CIG, CIL, BCIM, BCIG, SV, CIM, Tuning Section**
EXAMPLES:

Command	Description
> **BCIL7ØØØØ**	Sets the maximum sum of the integral error to 70,000

BCIM

NAME: Buffered Configure Integral Maximum
SYNTAX: <a>**BCIM**n
DEFAULT: n = 1000
RANGE: n = 0-32,767
VALID:
RESPONSE: a**BCIM** *INTEGRAL_GAIN_MAXIMUM=n[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Independently Saved
DESCRIPTION: This command is identical to the **CIM** command, except that it is buffered. This command defines the maximum integral gain you may use for tuning purposes. If you enter a valid number, it will become the new maximum integral gain. When you tune the drive using the Configure Integral Gain (**CIG**) command, you are actually setting the percentage of the maximum integral gain. The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **CIM, CIG, BCIM, SV, CIL, BCIL, Tuning Section**
EXAMPLE:

Command	Description
> **A5**	Sets acceleration to 5 rps^2
> **V1Ø**	Sets velocity to 10 rps
> **D32ØØØ**	Sets distance to 32,000 steps
> **BCIM2ØØØ**	Sets maximum integral gain to 2000
> **G**	Executes the move (Go)
> **BCIM5ØØØ**	Sets maximum integral gain to 5000
> **G**	Executes the move (Go)

The maximum integral gain is set to 2000 before the 32,000-step move and to 5000 after the first move.

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

BCPE

NAME: Buffered Configure Position Error
SYNTAX: <a>**BCPE**n
DEFAULT: n = 4000
RANGE: n = 0 - 999,999,999
RESPONSE: a**BCPE** *MAXIMUM_POSITION_ERROR=n_STEPS[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: If you supply a value, the specified value will become the new value for the maximum position (or following) error. If the actual following error exceeds this number (in Encoder Steps), the JSI will shutdown the drive and flash an error code 20 (following error exceeded).

This command is identical to **CPE** except that it is buffered. It is useful when the position error must be changed within a sequence.

SEE ALSO: **CPE, CDB, Tuning Section**
EXAMPLE:

Command	Description
> **A5**	Sets acceleration to 5 rps^2
> **V1Ø**	Sets velocity to 10 rps
> **BCPE4ØØØ**	Sets maximum following error to 4,000
> **D32ØØØ**	Sets distance to 32,000
> **G**	Executes the move (Go)
> **BCPE6ØØØ**	Sets maximum following error to 6,000
> **G**	Executes the move (Go)

The maximum following error is set to 4,000 before the first 32,000-step move and to 6,000 before the second 32,000-step move.

BCPG

NAME: Buffered Configure Proportional Gain
SYNTAX: <a>**BCPG**n
DEFAULT: n = 10
RANGE: n = 0 - 99
RESPONSE: a**BCPG** *PROPORTIONAL_GAIN=n_PERCENT[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Independently Saved
DESCRIPTION: This command defines the proportional gain to be used for tuning. When you tune the drive using the Buffered Configure Proportional Gain (**BCPG**) command, you are actually setting a percentage of the proportional gain maximum (**CPM**). This command is useful in tuning out the proportional error during a move. This command is identical to **CPG** except that it is buffered. It is useful when the proportional gain needs to be changed in a sequence. The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **CPG, CPM, Tuning Section, BCPM, SV**
EXAMPLES:

Command	Description
> **A5**	Sets acceleration to 5 rps^2
> **V1Ø**	Sets velocity to 10 rps
> **D32ØØØ**	Sets distance to 32,000
> **BCPG2Ø**	Sets proportional gain to 20
> **G**	Executes the move (Go)
> **BCPG5Ø**	Sets proportional gain to 50
> **G**	Executes the move (Go)

The proportional gain is set to 20 percent of the maximum value before the first 32,000 step move and to 50 percent of the maximum value before the second 32,000 step move.

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

BCPM

NAME: Buffered Configure Proportional Maximum
SYNTAX: <a>**BCPM**n
DEFAULT: n = 500
RANGE: n = 0 - 32,767
VALID:
RESPONSE: a**BCPM** *PROPORTIONAL_GAIN_MAXIMUM=n[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command defines the maximum proportional gain you can use for tuning purposes. If a valid number is entered, it will become the new maximum proportional gain. When you tune the drive using the proportional gain (**CPG**), you will actually set a percentage of the Maximum Proportional gain. This command is identical to **CPM** except that it is buffered. It is useful when the proportional maximum must be changed within a sequence. The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **CPM, CPG, BCPG, SV, Tuning Section**
EXAMPLES:

Command	Description
> **A5**	Sets acceleration to 5 rps^2
> **V1Ø**	Sets velocity to 10 rps
> **D25ØØØ**	Sets distance to 25,000 steps
> **BCPM5ØØØ**	Sets maximum proportional gain to 5000
> **G**	Executes the move (Go)
> **BCPM6ØØØ**	Sets maximum proportional gain to 6,000
> **G**	Executes the move (Go)

BS

NAME: Buffer Status Report
SYNTAX: a**BS**
DEFAULT: n = 2000
RANGE: n = 0 to 2000
VALID:
RESPONSE: a**BS** *n[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: Reports the number of characters (including delimiters) that can be loaded into the command buffer. When entering long string commands, check the buffer status to be sure that buffer space is available.

If you are using a computer interface, you can load the program lines to the controller faster than the controller can execute them. Therefore, the commands would get stored in the buffer. This command tells you how many more characters can be downloaded to the controller.

SEE ALSO: **B**
EXAMPLES:

Command	Description
> **1BS *1ØØ**	Space for 100 characters remain in the command buffer

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

C

NAME: Continue
SYNTAX: <a>**C**
DEFAULT: None
RANGE: None
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: The Continue (**C**) command ends a pause state. It enables your controller to continue executing buffered commands. After you initiate a pause with the Pause (**PS**) command or the Pause and Wait for Continue (**U**) command, you can clear it with a Continue command. This command is useful when you want to transmit a string of commands before you actually need to execute them.

If you are using the resume function (**SSL1**), the Continue (**C**) command can also be used resume execution of a move or a sequence after the program is halted by the Stop (**S**) command. Upon entering the C command, the program or the move will be started from the point where it stopped.

SEE ALSO: **PS, U, SSL**
EXAMPLES:

Command	Description
> **PS**	Pauses execution until the indexer receives a **C** command
> **MC**	Sets move to Continuous mode
> **A5**	Sets acceleration to 5 rps^2
> **V5**	Sets velocity to 5 rps
> **G**	Executes the move (Go)
> **T1Ø**	Waits 10 seconds after the move
> **VØ**	Sets velocity to zero
> **G**	Decelerates the motor to zero velocity
> **C**	Starts executing commands in buffer

CDB

NAME: Configure Dead Band
SYNTAX: <a>**CDB**n
DEFAULT: n = 0
RANGE: n = 0-32,767
RESPONSE: a**CDB** *SLIP_FAULT_DEADBAND=nSTEPS[cr]
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional, Savable in Sequence
DESCRIPTION: The **CDB** command defines the new dead-band value in motor steps. If dead-band is exceeded the slip fault output (if defined using the **OUT** command) will be active. When the exceeded deadband indicator output is off it indicates that the absolute value of the following error is within the dead band region. This is useful when you need to know if the motor rotor is within a certain tolerance range with respect to the commanded position.

SEE ALSO: **OUT , BCDB**
EXAMPLES:

Command	Description
> **1CDB5Ø**	Sets the dead-band to 50 steps
> **1CDB**	(Response) *SLIP_FAULT_DEADBAND=50 STEPS
> **OUT1G**	Sets Output 1 as a Exceeded Deadband Indicator Output
> **A10**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 10 rps
> **D25ØØØ**	Sets distance to 25,000
> **G**	Executes the move (Go)

If the motor is more than 50 encoder steps off, the JSI will activate Output 1.

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

CDG

NAME: Configure Differential Gain
SYNTAX: <a>**CDG**n
DEFAULT: n = 0
RANGE: n = 0 - 99
VALID:
RESPONSE: a**CDG** *DIFFERENTIAL_GAIN=n_PERCENT[cr]
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional, Independently Saved
DESCRIPTION: This command defines the differential gain to be used for tuning. When you tune the drive using the Differential Gain (**CDG**) command, you are actually setting a percentage of the Differential Gain Maximum (**CDM**). The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **BCDG, CDM, BCDM, SV, Tuning Section**
EXAMPLES:

Command	Description
> **1CDG3Ø**	Sets the differential gain to 30% of the maximum value

CDM

NAME: Configure Differential Maximum
SYNTAX: <a>**CDM**n
DEFAULT: n = 100
RANGE: n = 0 - 32,767
VALID:
RESPONSE: a**CDM** *DIFFERENTIAL_GAIN_MAXIMUM=n[cr]
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional, Independently Saved
DESCRIPTION: This command defines the maximum differential gain you may use for tuning purposes. If you enter a valid number, it will become the new maximum differential gain. When you tune the drive using the Configure Derivative Gain (**CDG**) command, you will actually set a percentage of the maximum differential gain.(**CDM**). The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **BCDM, CDG, BCDG, SV, Tuning Section**
EXAMPLES:

Command	Description
> **CDM5ØØ**	Sets the maximum differential gain to 500
> **CDG35**	Sets the differential gain to 35% of the maximum value set by **CDM**.

CFB

NAME: Configure Feedback Device
SYNTAX: <a>**CFB**n,x
DEFAULT: n = 1
RANGE: n = 1, 2, x = 200-32768
RESPONSE: *DEVICE_NUMBER_n[cr]*DEVICE_RESOLUTION_x[cr]
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional
DESCRIPTION: This command configures the JSI Controller to accept position feedback from the feedback device number entered, and also the feedback device resolution. The controller output must be off to use the **CFB** command.

CFB1: Selects an incremental encoder
CFB2: Selects an AR23 absolute encoder with -4 decoder box (ABS8)
CFB3: Selects an AR-C, AL-C, or Photo-Trak absolute encoder
CFB4: Selects an AR23 absolute encoder with -1 decoder box

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

Once you define the feedback device and the resolution of the feedback device, the JSI controller will scan either the POSITION FEEDBACK Input for absolute encoder feedback (**CFB2**, **CFB3**, or **CFB4**) or the ENCODER input for incremental encoder feedback (**CFB1**). You must turn off the JSI with the **OFF** command before changing the encoder type or resolution.

In incremental encoder mode (**CFB1**), the JSI can accept TTL Square Wave quadrature pulses in either a differential or single-ended fashion. Compumotor's incremental encoders are differential quadrature encoders (refer to Figure 3-4 in Chapter 3 for a wiring diagram of this encoder). You can also define encoder resolution with the **CFB1** command. The default setting is 4,000 steps per revolution. The maximum frequency of pulses that the JSI can read is 60 KHz before quadrature. Therefore, after quadrature, the maximum frequency the JSI can read is 240 KHz.

In the absolute encoder mode (**CFB2**, **CFB3**, or **CFB4**), the JSI reads the parallel data that the absolute encoder provides. Because it takes longer to process parallel positional information, the servo sample period for the PID loop will be slowed to 1,024 microseconds from 512 microseconds. However, the I/O functions are still performed every 512 microseconds. Refer to Chapter 3, *Installation*, for connection diagrams for the absolute encoders.

Entering the resolution and leaving the device number blank (e.g., **CFB,5ØØØ**) only changes the resolution, leaving the feedback device the same. When you change the feedback device, some other parameters are affected.

In **CFB1, 4ØØØ**, the Motor Resolution (**CMR**) command and Configure Position Error (**CPE**) both become 4,000. In **CFB2, 16,384**, both **CMR** and **CPE** both default to 16,384. In incremental mode (**CFB1**) the controller sample period time is 512 usec. If you use absolute encoder mode (**CFB2**), the sample period goes up to 1024 usec. You must set the JSI to **CFB2** if you are using an absolute encoder, otherwise the sample period will get the controller and the encoder out of synchronisity.

SEE ALSO: **CMR, CPE**

EXAMPLE:

Command	Description
> **CFB1,8ØØØ**	Define the feedback device as incremental encoder with the resolution of 8,000 steps per revolution.
> **CFB**	*DEVICE_NUMBER = 1_INCREMETAL ENCODER *DEVICE_RESOLUTION = 8000

CIG

NAME: Configure Integral Gain

SYNTAX: <a>**CIG**n

DEFAULT: n = 0

RANGE: n = 0 - 99

VALID:

RESPONSE: a**CIG** *INTEGRAL_GAIN= n_PERCENT[cr]

TYPE: Setup

ATTRIBUTES: Immediate, Device Address Optional, Independently Saved

DESCRIPTION: This command configures the integral gain to be used for tuning. When you tune the drive using Configure Integral Gain (**CIG**),you are actually setting a percentage of the Maximum Integral Gain (**CIM**). The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **BCIG, CIM, BCIM, SV, CIL, BCIL, Tuning Section**

EXAMPLES:

Command	Description
> **CIG7Ø**	Sets the integral gain to 70% of the maximum integral gain.

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

CIL

NAME: Configure Integral Sum Maximum Limit
SYNTAX: <a>**CIL**n
DEFAULT: n = 65535
RANGE: n = 0 - 2147483647
VALID:
RESPONSE: a**CIL** *INTEGRATOR_LIMIT=n[cr]
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional, Independently Saved
DESCRIPTION: This command defines the maximum integral sum limit. The value for the sum of the integral value is set by the **CIL** command. This command is useful if you want to move into a final position quickly. If you have a small **CIL** value, the system will not allow the sum of the errors to exceed the n value. This will result in the motor moving into position faster. If you make the **CIL** value very large, the response of final move will be slow. The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **CIG, BCIG, CIM, BCIM, SV, BCIL, Tuning Section**
EXAMPLE:

Command	Description
CIL7ØØØØ	Set the maximum sum of the integral error to 70,000

CIM

NAME: Configure Integral Maximum
SYNTAX: <a>**CIM**n
DEFAULT: n = 1000
RANGE: n = 0 - 32,767
VALID:
RESPONSE: a**CIM** *INTEGRAL_GAIN_MAXIMUM=n[cr]
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional, Independently Saved
DESCRIPTION: This command defines the maximum integral gain you may use for tuning purposes. If you enter a valid number, it will become the new maximum integral gain. When you tune the drive using the Configure Integral Gain (**CIG**) command, you are actually setting the percentage of the maximum integral gain. The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **BCIM, CIG, SV, CIL, BCIG, BCIL Tuning Section**
EXAMPLES:

Command	Description
> **CIM1ØØØ**	Sets the maximum integral gain to 1,000
> **CIG45**	Sets the integral gain to 45% of the maximum integral gain

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

COCH

NAME: Configure Output as Current Source (High)
SYNTAX: <a>**COCH**
DEFAULT: Controller is factory set to ±10 volt voltage source
RANGE: None
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional
DESCRIPTION: This command configures the JSI controller's output stage as a 200 mAmp current source. You may use the **COP** command to limit the maximum current output (Make it lower than 200 mAmp). From the factory setting, the controller is set as a voltage output source (**COV**). You must issue this command to change your output from voltage to current. All Compumotor's Analog Motor/Drive's are voltage input drives. You must not configure the JSI controller as a current source if the drive is voltage controlled. You may also use the **COCL** command to configure the output as a current source output with a maximum current of 60mA.

SEE ALSO: **COV, COCL, COP, DCC**
EXAMPLE:

Command	Description
COCH	Configure the output as a 200 mAmp current source.

COCL

NAME: Configure Output as Current Source (Low)
SYNTAX: <a>**COCL**
DEFAULT: Controller is factory set to ±10 volt voltage source
RANGE: None
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional
DESCRIPTION: This command configures the JSI Controller's output stage as a 60 mAmp current source. You may use the **COP** command to limit the maximum current output (Make it lower than 60 mAmp). The advantage of the **COCL** command over the **COCH** (High output current to 200 mAmp) is that the resolution is higher for **COCL** than for **COCH.** The JSI controller is set as voltage output source (**COV**) from the factory. You must issue the **COCL** command to change your output from a voltage to current source. You must not configure the JSI Controller as a current source if the drive is voltage controlled. You may also use the **COCH** command to configure the output as current source output with a maximum current of 200 mAmps.

SEE ALSO: **COV, COCH, COP, DCC**
EXAMPLE:

Command	Description
COCL	Configure the output of the JSI controller as a 60 mAmp current source.

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

COP

NAME: Configure Output Peak
SYNTAX: <a>**COP**n
DEFAULT: n = 100
RANGE: n = 0 - 100
VALID:
RESPONSE: See Example
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional
DESCRIPTION: This command sets the maximum value for the output command (**COV, COCL, COCH**) can be. The command specifies the maximum value of the control output as the percentage of the maximum control output. The JSI Controller uses the 12 bit DAC (Digital to Analog Converter) to determine the output current or output voltage. When the output peak is set at 100% (**COP1ØØ**), all of the DAC is used. At 50% (**CO5Ø**), only half of the DAC range is used. If the output is configured as a 200mA current source (**COCH**), at 100% the output will be 200mAmp, and the full DAC range is used. If the output peak is set at 50% (**COP5Ø**), the output current maximum will be 100 mAmp. For a 60 mAmp current source (**COCL**), 100% corresponds to 60mAmp output, and 50% corresponds to 30mAmp. The voltage source (**COV**) at 100% will provide 10 volts maximum output, and at 50% it will provide 5 volts maximum output. Use the DCC (Display current configuration command) to see if you are in either current or voltage mode.

SEE ALSO: **COV, COCH, COCL, DCC**
EXAMPLE:.

Command	Description
COCH	Configures output as current source (200mAmp)
COP1Ø	limit the maximum output as 10% of the maximum (+ 20mAmp)
1COP	The response should be *CURRENT_OUTPUT_LIMIT = 19.93_MILLIAMPS[cr]

COV

NAME: Configure Output as Voltage Source
SYNTAX: <a>**COV**
DEFAULT: Controller is factory set to ±10 volt voltage source
RANGE: None
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional
DESCRIPTION: This command configures the JSI controller's output stage as a ±10 volt voltage source . The JSI is set as a voltage output source. You may use the **COP** command to limit the maximum voltage output (Make it lower than 10 Volts). If you need to use the current output, you must use either the **COCH** or **COCL** commands to change the controller from voltage output to current output. The voltage output uses a 12 bit DAC (Digital to Analog Converter). When you use the **COP** command to limit the output voltage, you will not affect the resolution of the output voltage.

SEE ALSO: **COCH, COCL, COP, DCC**
EXAMPLE:

Command	Description
COV	Sets the output as voltage source
COP5Ø	Set the maximum output as 50% of + 10V.

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

CMR

NAME: Configure Motion Resolution
SYNTAX: <a>**CMR**n
DEFAULT: n = 4,000
RANGE: n = 200 - 32768
VALID:
RESPONSE: aCMR *MOTION_RESOLUTION=n_STEPS/REV[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Independently Saved
DESCRIPTION: You can define the motion resolution using this command. You must shut down the drive with the **OFF** command before defining a new resolution.

Changing motion resolution with the **CMR** command does not affect the system's dynamic performance. The microprocessor converts the commanded position to the appropriate absolute feedback position mathematically.

Before you define the new resolution, you must issue either the **OFF** or **ST1** command to shut down the drive. After the **CMR** command is executed, you must issue either the **ON** or **STØ** command to re-enable the drive. The new resolution that you issue will not take effect until you issue a distance command after the **CMR** command.

It should be noted that this command changes the values and responses of any other command that uses motion steps. You should configure motion resolution before you use any of the other commands that are defined in motion steps.

SEE ALSO: **OFF, ON, ST1, STØ**
EXAMPLE:

Command	Description
OFF	Turns Amplifier off
CMR4Ø96	Defines a motion resolution of 4,096 steps per revolution
ON	Turns amplifier on

This example demonstrates the procedures required to change the motion's resolution.

CPB

NAME: Configure Push Button
SYNTAX: <a>**CPB**n
DEFAULT: n = 1
RANGE: n = 0, 1
VALID:
RESPONSE: a**CPB** *1_FRONT_PANEL_ENABLED_[cr] or
*Ø_FRONT_PANEL_DISABLED_[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command allows you to control user access to the front panel push buttons. Since the PID tuning parameters can be modified via the Front Control Panel, and changing these values can effect move time and final position accuracy, it may be desirable to disable this interface during normal operations. The **CPB1** command enables the front panel push buttons and **CPBØ** will disable the front panel.

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

SEE ALSO:
EXAMPLES:

Command	Description
> **L**	Loop infinite times
> **A1Ø**	Sets acceleration to 10 rps^2
> **V1**	Sets velocity to 1 rps
> **D5ØØØ**	Sets distance to 5,000 steps
> **G**	Executes the move (Go)
> **CPB1**	Enable push buttons
> **T1Ø**	Wait 100 seconds
> **CPBØ**	Disable push buttons
> **N**	End loop

This sequence will only allow the user to access the front panel push buttons for 10 seconds after the move is complete. During the move, the front panel push buttons for tuning are disabled.

CPE

NAME: Buffered Configure Position Error
SYNTAX: <a>**CPE**n
DEFAULT: n = 4000
RANGE: n = 0 - 999,999,999
VALID:
RESPONSE: a**CPE** *MAXIMUM_POSITION_ERROR=n_STEPS[cr]
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional
DESCRIPTION: If a value is supplied, the specified value will become the new value for the maximum position (or following) error.

Depending on your load and the move parameters you will have different following errors: Higher acceleration will result in greater following error. This command defines the acceptable amount of following error for the system.

SEE ALSO: **BCPE, CDB, BCDB, Tuning Section**
EXAMPLE:

Command	Description
> **A5**	Sets Acceleration to 5 rps^2
> **V1Ø**	Sets Velocity to 10 rps
> **CPE4ØØØ**	Sets maximum following error to 4,000
> **D32ØØØ**	Sets Distance to 32,000
> **G**	Executes the move (Go)
> **CPE6ØØØ**	Sets maximum following error to 6,000
> **G**	Executes the move (Go)

The maximum following error is set to 4,000 before the first 32,000 step move and to 6,000 before the second 32,000 step move.

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

CPG

NAME: Configure Proportional Gain
SYNTAX: <a>**CPG**n
DEFAULT: n = 10
RANGE: n = 0 - 99
VALID:
RESPONSE: a**CPG** *PROPORTIONAL GAIN=n_PERCENT[cr]
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional, Independently Saved
DESCRIPTION: This command defines the proportional gain to be used for tuning. When you tune the drive using the Configure Proportional Gain (**CPG**) command, you are actually setting a percentage of the Proportional Gain Maximum (**CPM**). The Save (**SV**) command must be used to retain the value in nonvolatile memory.

This command is useful in tuning out the positional error during moves.

SEE ALSO: **BCPG, CPM, BCPM, Tuning Section, SV**
EXAMPLES:

Command	Description
> **1CPG4**	Configure the proportional gain to four percent of the maximum value.

CPM

NAME: Configure Proportional Maximum
SYNTAX: <a>**CPM**n
DEFAULT: n = 500
RANGE: n = 0 - 32,767
VALID:
RESPONSE: a**CPM** *PROPORTIONAL_GAIN_MAXIMUM = n[cr]
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional, Independently Saved
DESCRIPTION: This command defines the maximum Proportional Gain you can use for tuning purposes. If a valid number is entered, it will become the new maximum proportional gain. When you tune the drive using the proportional gain (**CPG**), you will actually set a percentage of the Maximum Proportional Gain. The Save (**SV**) command must be used to retain the value in nonvolatile memory.

SEE ALSO: **BCPM, CPG, BCPG, SV, Tuning Section**
EXAMPLES:

Command	Description
> **A5**	Sets acceleration to 5 rps^2
> **V1Ø**	Sets velocity to 10 rps
> **D25ØØØ**	Sets Distance to 25,000 steps
> **CPM5ØØØ**	Sets maximum proportional gain to 5000
> **G**	Executes the move (Go)
> **CPM6ØØØ**	Sets maximum proportional gain to 6,000
> **G**	Executes the move (Go)

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

CR

NAME: Carriage Return
SYNTAX: a**CR**
DEFAULT: None
RANGE: None
VALID: None
RESPONSE: a**CR** [cr]
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: The Carriage Return (**CR**) command can determine when the indexer has reached a particular point in the execution buffer. When the indexer reaches this command in the buffer, it responds by issuing a carriage return (ASCII 13) over its interface back to the host computer. If you place the **CR** command after a Go (**G**) command, it will indicate when a move is complete. If you place the **CR** command after a Trigger (**TR**) command, it will indicate when the trigger condition is met.

SEE ALSO:
EXAMPLE:

Command	Description
> **MPA**	Sets mode for absolute position
> **A5Ø**	Sets acceleration to 50 rps^2
> **V5**	Sets Velocity to 5 rps
> **D5ØØØ**	Sets distance to 5,000 steps
> **G**	Executes the move (Go)
> **1CR**	Sends a carriage return

The motor moves 5,000 steps. When the motor stops, the indexer sends a carriage return over its interface.

CVS

NAME: Configure Velocity Scale mode
SYNTAX: <a>**CVS**n, x
DEFAULT: n = 0, x=100
RANGE: n = 0 -2, x=0-5000
VALID:
RESPONSE: a**CVS** *n_ANALOG_MODE_DISABLED *ANALOG_GAIN_x if n=1, response is *n_ANALOG_FOLLOWING_MODE *ANALOG_GAIN_x if n=2, response is *n_ANALOG_SCALING_MODE *ANALOG_GAIN_x
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional
DESCRIPTION: This command configures the JSI for using the analog input interface in one of two possible modes. The two modes are the velocity following mode and the velocity scaling mode. The modes are selectable by choosing a number from 0 to 2 for n in the command above. To set up the JSI for one of the possible modes the unit must have the output command disabled by issuing an OFF command. The the command is entered as follows; type in CVS followed by a number to select the desired mode,then a comma (,) followed by the analog gain.

VELOCITY FOLLOWING mode *n=1*
Type in:
CVS 1,100

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

n=1 chooses the velocity following mode. In the velocity following mode the JSI will command a velocity profile which is proportional to the velocity signal applied at the analog input connector. The proportion factor is determined by the analog gain which is entered after the mode number. In this example a factor of 100 is entered. The velocity that the JSI will command is determined by the following equation.

$$\frac{\text{Analog input}}{\text{10 volts}} * \frac{\text{analog gain}}{100} * V = \text{velocity commanded}$$

Analog input: This is the voltage level present at the analog input. A 10 bit analog to digital converter coverts the analog signal into binary information. Typically in the velocity following mode the analog input signal will be the analog signal from a velocity tachometer attached to the servo whose velocity is to be followed.

10 volts: This is the maximum absolute value that the analog to digital converter can accept.

analog gain: This is the gain factor that is entered with the command after the mode number. In this case the analog gain is 100.

100: The analog gain is entered as a scale factor. Thus if the analog gain were 100 then the scale factor would be 1, and if the analog input were 10 volts then the factor which the velocity is multiplied by will be 1.

V: This is the velocity that the user enters using the V command.

In this mode a value for the velocity is chosen, then the velocity following mode is chosen using the number 1, and the analog gain is chosen to be between 0 and 5000. It should be noted that this velocity is the velocity of the velocity command profile. From this velocity profile a position command is determined for each sample period. The actual position and the commanded position create an error which goes through the PID algorithm and forms the analog velocity output command at the command output interface. This velocity is not the same as the velocity produced by the above equation. It is the velocity after the commanded velocity has gone through the PID filter.

VELOCITY SCALING mode *n=2*

Selects the velocity scaling mode. In this mode the voltage at the analog input is used to scale the commanded velocity. The JSI must be used in the continuous mode (MC command) to use the velocity scaling mode. The velocity is scaled according to the following equation:

$$\frac{\text{Analog input}}{\text{10 volts}} * \frac{\text{analog gain}}{100} * V + V = \text{velocity commanded}$$

Analog input: This is the voltage level present at the analog input. A 10 bit analog to digital converter coverts the analog signal into binary information. Typically in the velocity following mode the analog input signal will be the analog signal from a velocity tachometer attached to the servo whose velocity is to be followed.

10 volts: This is the maximum absolute value that the analog to digital converter can accept.

analog gain: This is the gain factor that is entered with the command after the mode number. In this case the analog gain is 100.

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

100: The analog gain is entered as a scale factor. Thus if the analog gain were 100 then the scale factor would be 1, and if the analog input were 10 volts then the factor which the velocity is multiplied by will be 1.

V: This is the velocity that the user enters using the V command.

The JSI is placed in the continuous mode using the MC command. The CVS command is then used with n=2 for the scaling mode followed by a comma and the analog gain. The analog gain can be from 0 to 5000. The voltage at the analog input is divided by the full scale voltage of 10 volts then multiplied by the analog gain divided by 100. This is then multiplied by the velocity and added to the velocity. Multiplying the number times the velocity maintains the units of revs/second. Typically the analog input will be an external error signal which, based on its value will slow down or speed up the servo. The continuous velocity is adjusted by the analog input according to the above equation.

To disable the analog input mode which is selected by either a 1 or a 2 type in n=0. n=0 will disable the analog input signal from having any affect.

D

NAME: Distance
SYNTAX: <a>**D**n
DEFAULT: n = 4000
RANGE: n = ± 838,860,800
VALID:
RESPONSE: a**D** *Dn[cr]
TYPE: Motion
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The Distance (**D**) command defines motion in either the number of steps it will move or the absolute position it will seek after a Go (**G**) command is entered. In incremental mode (**MPI**), the value set with the Distance (**D**) command will be the distance motion will travel on all subsequent Go (**G**) commands. In absolute mode (**MPA**), the distance moved by the motor will be the difference between the current motor position and the position (referenced to the zero position) set with the **D** command. A valid distance must be defined with the **D** command before a preset move can be executed. The Distance (**D**) command has no effect on continuous moves (**MC**). The Distance (**D**) value is stored in nonvolatile memory.

SEE ALSO: **A, G, V, MN, MPA, MPI**
EXAMPLES:

Command	Description
> **MN**	Set mode to normal (preset)
> **A5**	Set Acceleration to 5 rps^2
> **V1Ø**	Set velocity to 10 rps
> **D5ØØØØ**	Set Distance to 50,000 steps
> **G**	Executes the move (Go)

Motor will travel 50,000 steps in the clockwise direction after the **G** command is issued.

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

DAI

NAME: Display Analog Input
SYNTAX: a**DAI**
DEFAULT: None
RANGE: n = ± 10.00
RESPONSE: a**DAI** *ANALOG_INPUT=n_VOLTS
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: This command reports the analog voltage input to the JSI Controller. It can report the value either continuously or one time. This input can be used in conjunction with the Velocity Scaling (**CVS**) command to perform velocity tracking.

a**DAI** Continually reports the analog input value until any key is pressed
a**DAI1** Report analog input value 1 time only without descriptive text.

SEE ALSO: **CVS**
EXAMPLES:

Command	Description
> **1DAI1**	Report analog input 1 time only without descriptive text
> **1DAI**	Continually report the analog input value

DCC

NAME: Display Current Control Configuration
SYNTAX: a**DCC**
DEFAULT: None
RANGE: None
RESPONSE: See Example
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required
DESCRIPTION: This command reports the current configuration of the JSI Controller. It reports the following information:

- Feedback Device Number (Defined by **CFB**)
- Sample Period Time (Defined by **CFB** either as 512 or 1024 usec)
- Output Amplifier Type (Defined by **COV, COCL, COCH** to determine if the controller output should be a voltage or current source)
- Output Limit (Defined by **COP** to limit the maximum output voltage or current)
- Feedback Device Resolution (Defined by **CFB** command)

SEE ALSO: **CFB, COV, COCL, COCH, COP**
EXAMPLE: >**1DCC** *CONTROLLER_CONFIGURATION:

*FEEDBACK_DEVICE_NUMBER_=1_INCREMENTAL_ENCODER
The feedback device is an incremental encoder

*SERVO_SAMPLE_PERIOD_=512_MICROSECONDS
Controller sampling period is 512 microseconds

*OUTPUT_AMPLIFIER_IS_A_1Ø_VOLT_VOLTAGE_SOURCE
The JSI is using 10 volts voltage source
*VOLTAGE_OUTPUT_LIMIT=1Ø.ØØ_VOLTS
The voltage output limit is set to 100% of the maximum

*DEVICE_RESOLUTION_=4ØØØ
Feedback device resolution is 4000 steps/rev after quadrature.

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

DFS

NAME: Display Flags for Controller Status
SYNTAX: a**DFX**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: a**DFX** *bbbb_bbbb_bbbb_bbbb_bbbb_bbbb_bbbb_bbbb*[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: The display flags for Controller Command returns the Controller status flag as a 32 bit response. Each bit refers to a status flag as shown on the chart below. The associated number for each flag corresponds to each binary (1 or 0) response in the following order:
31,3Ø,29,28,....3,2,1,Ø

Bit
31 27 23 19 15 — these bits are all reserved for
3Ø 26 22 18 14 — future use they all return zero's
29 25 2Ø 17 13
29 24 21 16 12
1Ø 9 7 6 4 2 1

Bit
14 User Fault no = Ø yes = 1
11 Enable Circuit Ø = enable 1 = disabled
8 Failed nonvolatile memory checksum no= Ø yes = 1
5 max position error exceeded no = Ø yes = 1
3 command out disabled Ø = no 1 = yes
Ø commanded shutdown Ø = no 1 = yes

DFX

NAME: Display Flags for Controller Status
SYNTAX: a**DFX**
DEFAULT: None
RANGE: b = 0, 1
VALID:
RESPONSE: a**DFX** *bbbb_bbbb_bbbb_bbbb_bbbb_bbbb_bbbb_bbbb*[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: The display flags of Controller Command returns the controller status flag as a 32 bit response. Each bit refers to a status flag as shown on the chart below. The associated number for each flag corresponds to each binary (1 or 0) response is in the following order:
31,3Ø,29,28,....3,2,1,Ø

Bit
31 27 23 — these bits are all reserved for
3Ø 26 22 — future use they all return zero's
29 25 13
29 24

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

Bit	
21	hit a software CCW limit no= Ø, yes = 1
2Ø	hit a software CW limit no = Ø, yes= 1
19	home limit not found = Ø, found = 1
18	jog disable = Ø, enable = 1
17	queued for RM mode no = Ø, yes = 1
16	run sequence on power up no=Ø yes=1
15	U command Ø=not waiting, 1=waiting for continue
14	waiting for a trigger no = Ø yes = 1
13	home to resolver position no = Ø yes = 1
12	back up to home limit Ø = no yes = 1
11	high speed portion of home move no = Ø 1 = in process
1Ø	execute a sequence no = - yes = 1
9	wait on a timer no = Ø yes = 1
8	hit a CCW limit no = Ø yes = 1
7	hit a CW limit no = Ø yes = 1
6	**PS** command Ø=not waiting 1 = waiting for continue
5	absolute move direction no = CW yes = CCW
4	Incremental/absolute Ø = MPI 1 = MPA
3	Mode preset = Ø continuous = 1
2	Commanded move direction Ø = CW 1 = CCW
1	preset move in progress Ø = not moving 1 = moving
Ø	continuous move during acceleration/deceleration Ø=not moving 1=moving

DMO

NAME: Display Maximum Overshoot
SYNTAX: a**DMO**
DEFAULT: None
RANGE:
VALID:
RESPONSE: a**DMO** *MAXIMUM_OVERSHOOT_= n_STEPS
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: This command reports the number of steps the motor moves past the designated distance. This command is valid in preset moves only.

a**DMO1** Gives overshoot report once, without descriptive text
a**DMO** Continually reports the position overshoot until any key is pressed.

This command is useful for tuning out overshoots in the system. When you tell the motor to execute a move, the overshoot value is set to zero and the overshoot is measured while the move is being made. When a new move is started the error is set to zero again.

SEE ALSO: None
EXAMPLE:

Command	Description
> **L**	Loop forever until stopped
> **A1Ø**	Sets acceleration to 10 rps^2
> **V2**	Sets velocity to 2 rps
> **D25ØØØ**	Sets distance to 25,000 steps
> **G**	Executes the move (Go)
> **T2**	Waits 2 seconds after the move
> **1DMO1**	Reports maximum overshoot once without descriptive text
> **H**	Changes direction of the move.
> **G**	Executes the move (Go)
> **N**	End Loop

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

DOI

NAME: Display Output Instantaneous
SYNTAX: a**DOI**
DEFAULT: None
RANGE: n = ± 10.00VOLTS, ± 200 MILLIAMPS, ± 60 MILLIAMPS
VALID;
RESPONSE: a**DOI** *COMMAND_OUT=n
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: This command displays the Analog output being applied to the drive. It will constantly update and display the output to the drive. If you are in Voltage Output (**COV**) mode , the information range will be ± 10 volts. In you are in Lower Current Output (**COCL**) mode, the information range will be ± 60 milliamps. If you are in Higher Current Output (**COCH**) mode, the information range will be ± 200 milliamps. Analog output is continuously reported until any key is pressed.

a**DOI** Continually reports the analog output value until any key is pressed
a**DOI1** Report analog input value 1 time only without descriptive text.

SEE ALSO: **COV, COCL, COCH**
EXAMPLES:

Command	Description
> **COV**	Change JSI to + 10 Volts Output Mode
> **1DOI1**	Report output voltage once
> **1DOI**	Reports the output voltage continuously

DPA

NAME: Display Position Actual
SYNTAX: a**DPA**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: a**DPA** *ACTUAL_POSITION=n_STEPS
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: The display position actual command displays the actual position in feedback steps. The function is canceled by sending any single ASCII character to the JSI.

a**DPA1** reports the actual position 1 time without descriptive text
a**DPA** reports the actual position continuously until a character is sent to the JSI

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

DPE

NAME: Display Position Error
SYNTAX: a**DPE**
DEFAULT:
RANGE: None
VALID:
RESPONSE: a**DPE** *POSITION_ERROR_=n_STEPS
TYPE: Status
ATTRIBUTES: Buffered, Device Address Required, Never Saved
DESCRIPTION: This command reports the difference between the position setpoint and the current position. The **PID** algorithm uses this number to determine the sign and magnitude of the command from the JSI. The larger the position error, the larger the control signal output to correct the error. This value is also used by the controller to see if the motor is within the deadband specified by the **CDB** command.

a**DPE** Continually reports the position error value until any key is pressed
a**DPE1** Report Position error value 1 time only without descriptive text.

SEE ALSO: **CDB**
EXAMPLES:

Command	Description
a**DPE**	Continually reports the position error value until any key is pressed
a**DPE1**	Report position error value 1 time only without descriptive text.

DPS

NAME: Display Setpoint Position
SYNTAX: a**DPS**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: a**DPS** *POSITION_SETPOINT=n_STEPSn[cr]
TYPE: Status
ATTRIBUTES: Buffered, Device Address Required, Never Saved
DESCRIPTION: This command reports the absolute position setpoint. This reporting is continued until any character is sent to the indexer through an RS-232C interface.

a**DPS** Continually reports the setpoint position value until any key is pressed

a**DPS1** Report[position setpoint value 1 time only without descriptive text.

The position setpoint indicates the JSI's commanded position for **PID** computation.

SEE ALSO: None
EXAMPLES:

Command	Response
> **DPS**	Reports the setpoint position continuously
> **DPS1**	Report setpoint position 1 time only without descriptive text.

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

DRD

NAME: Read Distance via Parallel Input/Output
SYNTAX: <a>**DRD**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command initiates the controller to read distance value from the parallel inputs. You must set up the inputs as Data input using the **IN** command. You must also set up outputs as strobe outputs using t the **OUT** command. Typically, you would set up 8 inputs as DATA Inputs and 5 Outputs as strobe outputs using the OUT command. If an input is designated as a Data Sign, then if it is active during any byte read, the sign becomes negative, otherwise the inactive condition (default) is a positive sign.

You must also set up the strobe output delay time (typically greater than a **PLC** Scan Time, if using a PLC, or a minimal debounce time if using thumbwheel switches) so that the device you are getting data from will know when the JSI is ready for data. Sequence of events take place when we enter the DRD command:

Step 1 The lowest Output bit designated as strobe output will go on and stay on for strobe output delay time defined by **STR** command.

Step 1A If an input is designated as a data valid input, then the strobe output will stay active until the data valid input is active.

Step 2 The JSI will read the parallel inputs designated as DATA inputs. The lowest inputs will be the least significant bit. The JSI reads 8 bits (2 BCD digits) and stores them into two digits on the distance register.

Step 3 Next Strobe output bit designated as strobe output will go on and stay on for delay time defined by **STR** command.

Step 3A If an input is designated as a data valid input, then the strobe output will stay active until the data valid input is active.

Step 4 The JSI will read the parallel inputs designated as DATA inputs are placed in the lowest digits on the position register. Previously entered values are shifted two positions to the left.

Step 5 Steps 3 through 4 are repeated as many times as the strobe outputs are available. You can use up to 5 different outputs as strobe outputs. The last two digits read are the least significant digit of the distance value.

SEE ALSO: **STR, IN, OUT**
EXAMPLE: The following example shows how the JSI reads the distance value of 3589721 steps. We will use inputs 1 through 8 for DATA inputs and outputs 1 through 4 for Strobe Outputs.

Step 1 Type the following:

Command	Description
OUT1J	Set Output 1 as Strobe Output
OUT2J	Set Output 2 as Strobe Output
OUT3J	Set Output 3 as Strobe Output
OUT4J	Set Output 4 as Strobe Output
IN1N	Set Input 1 as a Least Significant DATA input
IN2N	Set input 2 as a DATA input
IN3N	Set input 3 as a DATA input
IN4N	Set input 4 as a DATA input
IN5N	Set input 5 as a DATA input
IN6N	Set input 6 as a DATA input
IN7N	Set input 7 as a DATA input
IN8N	Set input 8 as a the Most significant DATA input
STR5ØØ	Set strobe output delay time to 500 msec.
A1Ø	Sets acceleration to 10 rps^2
V5	Set velocity to 5 rps

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

Step 2 As soon as you enter the Read Distance via Parallel Input (**DRD**) command , the following should happen

Type the following:

Command	Description
DRD	Start reading from Parallel input

Step 3 The JSI will turn on OUT1 for 500 msec.

Step 4 You must have your inputs 1 through 8 to the following settings.

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	OFF	OFF	OFF	OFF	OFF	ON	ON
	Ø				3		

The current setting of the distance is
ØØØØØØØØØ3

Step 5 After the JSI reads the input, OUT1 turns off and OUT2 will automatically go on for 500 msec.

Step 6 You must have your inputs 1 through 8 set to the following settings.

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	ON	OFF	ON	ON	OFF	OFF	OFF
	5				8		

Previously entered value of 03 shifts 2 locations to the left and 58 is placed on the 2 lowest positions. The current setting of the distance value is:
ØØØØØØØ358

Step 7 OUT 2 turns off and OUT3 automatically goes on for 500 msec.

Step 8 You must have your inputs 1 through 8 set to the following settings:

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
ON	OFF	OFF	ON	OFF	ON	ON	ON
	9				7		

Previously entered values 0358 shifts 2 locations to the left and 97 is placed on the two lowest positions. The current setting of the distance value is:
ØØØØ35897

Step 9 OUT 3 turns off and OUT4 automatically goes on for 500 msec.

Step 10 You must have your inputs 1 through 8 set to the following settings.

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	OFF	ON	OFF	OFF	OFF	OFF	ON
	2				1		

Previously entered values 035897 shifts two locations to the left and 21 is placed on the two lowest positions. The final setting of the distance value becomes 3,589,721.

Step 11 Since the fifth strobe output is not defined, the JSI now goes to the next command in the buffer.

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

DTP

NAME: Display Tuning Parameters
SYNTAX: a**DTP**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: See Description
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required
DESCRIPTION: This command displays the servo tuning parameters in block format.

P Proportional Gain
I Integral Gain
D Derivative Gain

The first row displays the percentage of the maximum gain being used by the JSI, and the second row displays the maximum gain set for each parameter.

This command also reports
- The sum of integrator Limit set by **CIL** command
- The maximum position error set by **CPE** command
- The motion resolution set by **CMR** command
- Status of the Hardware limit enabled/not enabled

SEE ALSO: **CIL, CPE, CMR, LD, CIG, CPG, CDG, CIM, CPM, CDM**
EXAMPLE: **1DTP**

	P	I	D	
PERCENT	1Ø	Ø	Ø	Proportional gain is set to 1Ø% of the maximum (5ØØ)
MAXIMUM	5ØØ	1ØØØ	1ØØØ	Integral and Derivative gains are turned off (Ø%)

*INTEGRATOR_LIMIT=65535
CIL is set to 65535
*MAXIMUM_POSITION_ERROR=4ØØØ_STEPS
CPE is set to 4000
*MOTION_RESOLUTION=4ØØØ_STEPS/REV
CFB resolution is set to 4000 steps
*LIMITS_ENABLED
Limits are enabled by **LDØ**

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

DVA

NAME: Display Velocity Actual
SYNTAX: a**DVA**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: a**DVA** *ACTUAL_VELOCITY=n_RPM
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: This number is reported in revolutions per minute. This value is the actual velocity being read from the feedback device.

aDVA Continually reports the actual velocity continuously until any key is pressed
aDVA1 Report the actual velocity 1 time only without descriptive text.

DVS

NAME: Display Velocity Setpoint
SYNTAX: a**DVS**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: a**DVS** *VELOCITY_SETPOINT=n
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: This number is reported in steps per second and can be reported either once or continually. This value is the velocity being calculated by the PID loop.

aDVS Continually reports velocity setpoint until any key is pressed
aDVS1 Report the velocity setpoint 1 time only without descriptive text.

SEE ALSO: None
EXAMPLES:

Command	Description
> **1DVS**	Velocity setpoint is reported until a number key is pressed.

E

NAME: Enable RS-232C Communication Interface
SYNTAX: <a>**E**
DEFAULT: Communications Enabled
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: The Enable Communications Interface (**E**) command allows the controller to accept commands over the RS-232C interface. You can re-enable the RS-232C interface with this command if you had previously disabled the interface with the Disable Communications Interface (**F**) command. If several units are using the same communications interface, the **E** and **F** commands can help streamline programming.

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

SEE ALSO: **F**

EXAMPLES:

Command	Description
> **F**	Disables all units (axes) on the communications interface
> **1E**	Enables Device 1
> **4E**	Enables Device 4
> **G**	Executes the move (only axes 1 and 4 will move)
> **E**	Enable TS-232C communications on all devices.

Only Axes 1 and 4 will make the preset move. Then all the others will be enabled.

F

NAME: Disable RS-232C Communications Interface
SYNTAX: <a>**F**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: The Disable Communications Interface (**F**) command is useful when you are programming multiple units on a single interface. Axes that are not intended to process global commands receive device specific **F** commands. This allows you to program other units without specifying a device identifier on every command. I
SEE ALSO: **E**
EXAMPLES:

Command	Description
> **1F**	Disables the communications interface on units with device address 1
> **3F**	Disables the communications interface on units with device address 3
> **G**	All of the indexers (except devices 1 and 3) will execute the move (Go)

G

NAME: Go
SYNTAX: <a>**G**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Motion
ATTRIBUTES: Buffered, Device Address Optional, Saveable in Sequence
DESCRIPTION: The Go (**G**) command instructs motion to occur using motion parameters that you have previously entered. You do not have to re-enter Acceleration (**A**), Velocity (**V**), Distance (**D**), or the current mode (**MN** or **MC**) commands with each Go (**G**) command . The acceleration (**A**), Velocity (**V**), and Distance (**D**) values are saved in nonvolatile memory. In the Incremental Preset mode (**MPI**), a Go (**G**) command will move the steps you specified with the Distance (**D**) command.

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

A Go (**G**) command in the Absolute Preset mode (**MPA**) will not cause motion unless you enter a change in Distance (**D**) command first, from where you are.

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

No motion will occur until you enter the Go (**G**) command in both the Normal (**MN**) and Continuous (**MC**) modes.

If motion does not occur with the Go (**G**) command, an activated software travel limit or end-of-travel limit switch may be on. Check the limit switches.

SEE ALSO: **S, MN, MC, A, V, D, MPI, MPA**

EXAMPLES:

Command	Description
> **MN**	Sets mode to normal (preset)
> **A5**	Sets acceleration to 5 rps^2
> **V1Ø**	Sets velocity to 10 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)

Assuming the indexer is in Incremental Preset mode, a move of 25,000 steps executes and then repeats the 25,000-step move using the new acceleration value of 1 rps^2 (Total distance moved = 50,000 steps).

GH

NAME: Go Home

SYNTAX: <a>**GH** <n>

DEFAULT: n = 1

RANGE: n = ± .01 - 99.99 rps

VALID:

RESPONSE: None

TYPE: Motion

ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence

DESCRIPTION: The Go Home (**GH**) command instructs the JSI to search for an the home limit switch in the positive or negative direction. It will define Home as the position where the Home Limit signal changed states nearest the edge selected with the OS command. As soon as the selected edge is detected (usually via a load activated switch) the motor decelerates to 0 velocity.

The controller will then position the motor on the outside of the selected edge. Finally, the motor will creep at Final Go Home velocity defined by **GHF** command in the direction of the home active region.

The process is the same in encoder step mode, except the controller also looks for the Z Channel input as well as the Home Limit input to be active. This means that the Z Channel pulse should be *enveloped* by the active region of the Home Limit input.

The controller will reverse direction if an End-Of-Travel limit is activated while searching for Home; however, if a second End-Of-Travel limit is encountered in the new direction the Go Home procedure will stop and the operation will be aborted. The **RG** command will indicate if the operation

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

was successful. If the Backup To Home Switch (**OSB**) function is disabled by the **OS** command, the the motor will be considered at Home if the Home limit input is still active at the end of the deceleration, following the encounter of the selected edge of the home switch. See the Application Design Section for a detailed description of the homing function.

If you set the Go Home velocity using the **GH** command, the motor will move in the direction and velocity specified by the **GH** command and the value is stored in the **GHV** command. If you do not specify the velocity, the homing will be executed using the velocity and direction defined by the **GHV** command.

SEE ALSO: **GHA, GHV, IN, INL, GHF, OSB, OSC, OSD, OSG, OSH**

EXAMPLE:

Command	Description
> **GHA1Ø**	Sets go home acceleration to 10 rps^2
> **GHV2**	Sets go home velocity to 2 rps
> **IN1S**	Sets input 1 as the remote go home input
> **INLØ**	The remote go home will be executed when the Go Home input is low (Closed)
> **GH**	Go Home in the direction and velocity specified by **GHV** command using the **GHA** acceleration.

GHA

NAME: Go Home Acceleration

SYNTAX: <a>**GHA**n

DEFAULT: n = 99.000 rps^2

RANGE: n = 0.0001 - 9999.999 rps^2

VALID:

RESPONSE: a**GHA** *n[cr]

TYPE: Motion

ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence

DESCRIPTION: The Go Home Acceleration (**GHA**) command sets the linear acceleration value that will be used during any subsequent Go Home (**GH**) moves. The value is stored in nonvolatile memory.

The Go Home velocity is set using the Go Home Velocity (**GHV**) command. You must use the **IN** command to specify the remote Go Home input.

SEE ALSO: **GH, GHV, IN, INL**

EXAMPLE:

Command	Description
> **GHA1ØØ**	Sets Go Home acceleration to 100 rps^2
> **GHV2**	Sets Go Home velocity to 2 rps
> **IN1S**	Sets input 1 as the remote go home input
> **INLØ**	The remote Go Home will be executed when Go Home is low (closed)
> **GH**	Go Home in the direction and velocity specified by **GHV** command, using the **GHA** acceleration

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

GHF

NAME: Go Home Final Velocity
SYNTAX: <a>**GHF**n
DEFAULT: n = 0.1 rps
RANGE: n = 0-9.99 rps
VALID:
RESPONSE: a**GHF** *GFHn[cr]
TYPE: Motion
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command set the velocity for the final approach in the go home sequence. The value is stored in nonvolatile memory.

SEE ALSO: **GH, OS, GA, GHA, GHV**
EXAMPLE:

Command	Description
> **GHF.1**	The velocity of the final approach of the next Go Home move will be .1 rps
> **GH2**	Execute Go Home at 2 rps in CW direction.

GHV

NAME: Go Home Velocity
SYNTAX: <a>**GHV**n
DEFAULT: n = ± 10
RANGE: n = ± .01 - 99.99
VALID:
RESPONSE: a**GHV** *GHVn[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command sets the velocity and direction to be used for remote homing or for subsequent Go Home (**GH**) commands. This command is stored in nonvolatile memory. When you turn on the remote Go Home input, or issue a Go Home (**GH**) command over RS-232C interface, the motor will start moving in the direction and velocity specified by **GHV** command.

The acceleration of homing process can be defined using the **GHA** command. The remote Go Home input must be specified using the **IN** command.

SEE ALSO: **GH, GHF, GHA, IN, INL, OS**
EXAMPLE:

Command	Description
> **GHA1Ø**	Sets Go Home acceleration to 10 rps^2
> **GHV2**	Sets Go Home velocity to 2 rps
> **IN1S**	Sets input 1 as the remote Go Home input
> **INLØ**	The remote go home will be executed when the Go Home input is low (closed)
> **GH**	Go home in the direction and velocity specified by the **GHV** command, using the **GHA** Acceleration

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

^H

NAME: Backspace
SYNTAX: **^H** (Control **H**, ASCII 8)
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Immediate, Not Addressable, Never Saved
DESCRIPTION: This command allows you to delete the last character that you entered. The **^H** command will not prevent execution of an immediate command. A new character may be entered at that position to replace the existing character. (**^H** indicates that the Ctrl key is held down when the **H** key is pressed.) This command prompts the JSI to backup one character in the command buffer, regardless of what appears on the terminal. On some terminals, the Ctrl and the left arrow (←) keys produce the same character.

H

NAME: Set Direction
SYNTAX: <a>**H**<s>
DEFAULT: s = **+**
RANGE: s = +, -
VALID:
RESPONSE: None
TYPE: Motion
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The Set Direction (**H**) command changes or defines the direction of the next move that the system will execute. This command does not effect moves already in progress.

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

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

SEE ALSO: **D**
EXAMPLE:

Command	Description
> **MN**	Sets Normal mode
> **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) in CW direction
> **H**	Reverses direction
> **G**	Executes the move (Go) in CCW direction
> **MC**	Sets mode to continuous
> **H+**	Sets direction to CW
> **G**	Moves continuously in CW direction

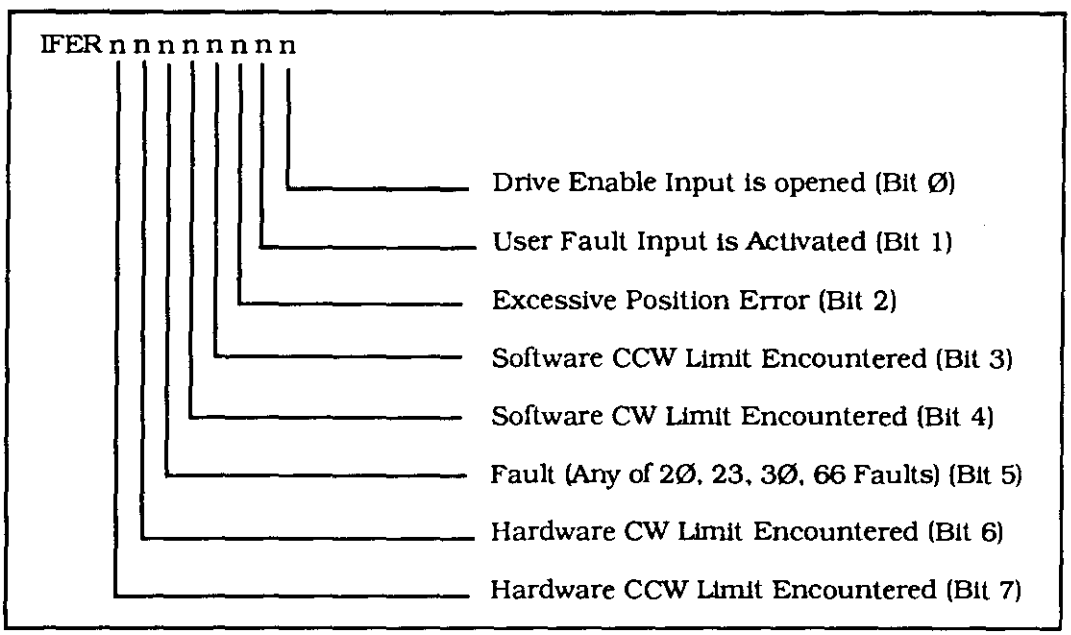
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

IFER

NAME: Compare Error Flag
SYNTAX: <a>**IFER**nnnnnnnn
DEFAULT: None
RANGE: n = 0, 1, x
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command checks to see if any fault condition exists to execute a conditional command. This command is very similar to IF_THEN statement in Basic Programming. The difference is, the 8 bits set by different fault conditions determine the next command to be executed. This command compares the individual bit pattern of the error flag. A 0 means the error does not exist, and a 1 means the error does exist. An X means the comparison doesn't care whether that particular error exists or not, so if you put 8 X's after the **IFER** command, the commands between **IFER** and **NIF** will be executed no matter what the error condition of the commands may be. An **IF** command (**IFER, IFFL, IFIN**) cannot be used within another **IF** command.

This command is useful to trap different error conditions. For example, you may want to move the motor in the CW direction if the motor reached the CCW limit. The command should be used in the **XFK** sequence to recover from errors or to set the JSI (especially the outputs) to a safe state.

Refer to the following error bits.



For this command, we will refer to 1 as set, and Ø as not set.

Bit Ø This condition becomes set when you remove the Enable jumper from the JSI connector. This will also create an error 23.

Bit 1 This condition becomes set when you use one of the 13 inputs as a user fault input and activate the input. This will also create an error 66.

Bit 2 This condition becomes set when the position error set by the **CPE** value is exceeded. This will also create an error 20.

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

Bit 3 This condition becomes set when the software CCW limit is encountered. This will also create an error 44.

Bit 4 This condition becomes set when the software CW limit is encountered. This will also create an error 43.

Bit 5 This condition becomes set if any one of the following conditions exist; Excessive Position Error (Error 20), Drive Enable Opened (Error 23), EEPROM checksum error (error 30), or User Fault (error 66) becomes active.

Bit 6 This condition becomes set when the hardware CW limit is encountered. This will also create an error 41.

Bit 7 This condition becomes set when the hardware CCW limit is encountered. This will also create an error 42.

SEE ALSO: **DFX, NIF, IFIN, IFFL, XFK**

EXAMPLE:

Command	Description
> L	Loop forever
> IFER1	If hardware CCW limit switch is encountered do the following.
> A1Ø	Sets acceleration to 10 rps^2
> V5	Sets velocity to 5 rps
> D1ØØØØ	Sets distance to 10,000 steps
> G	Executes the move (Go)
> NIF	End of statement
> IFERX1	If hardware CW limit is encountered do the following.
> A1Ø	Sets acceleration to 10 rps^2
> V5	Sets velocity to 5 rps
> D-1ØØØØ	Sets distance to 10,000 steps in the opposite direction
> G	Executes the move (Go)
> NIF	End of statement
> IFERXXXXXXX1	If Drive enable input is opened do the following
> 1"CONNECT_	
> 1"DRIVE_	
> 1"ENABLE_	Write to the RS-232C port letting the user
> 1"INPUT_	know that drive enable is disconnected.
> N	End of Loop

There are three IF_THEN statements. The first checks if the hardware CCW limit has been encountered. Motion will occur in the CW direction if CCW limit has been encountered. Then it checks for the CW limit to move in the CCW direction in case CW limit has been encountered. Lastly, the controller checks for the Drive enable.

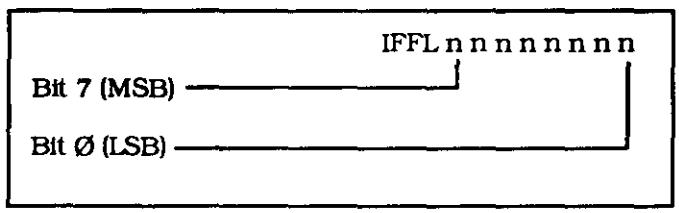
IFFL

NAME: Compare User Flag
SYNTAX: <a>**IFFL**nnnnnnnn
DEFAULT: None
RANGE: n = 0, 1, X
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command uses the pattern set by the Set User Flag (**SFL**) command to execute the conditional commands. This command is very similar to the

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

IF_THEN statement in Basic Programming. The difference is, the 8 bits set by the **SFL** command are compared to the 8 bits set by the **IFFL** command. If the bit patterns match, the commands following the **IFFL** command up to **NIF** will be executed. If the bit patterns do not match the program will skip to a command immediately following the **NIF** command. For this command, we will refer to 1 as set, and 0 as not set. An X represents a don't care condition, the bit will not be used in the evaluation of the condition. An **IF** command (**IFFL, IFER, IFIN**) cannot be used within another **IF** command.

This command is useful if you wish to make a decision based on previous sequence executions than will set or clear user flag bits. Also you can use a host computer to generate the **SFL** (User Flag) pattern depending on the condition, then you can use IFFL to perform branching type of operations.



SEE ALSO: **NIF, SFL, IFER, IFIN**

EXAMPLE 1:

Command	Description
> **SFL1Ø1**	Set user flag bits 7 and 5 and clear bit 6, the remaining bits are left unaltered
> **IFFL1Ø1Ø**	If user flag bits 5 and 7 are set and bits 6 and 4 clear, do the following commands
> **A1Ø**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D25ØØØ**	Sets distance to 25,000 steps
> **G**	Executes the move (Go)
> **NIF**	End of statement

If the **IFFL** pattern matches the **SFL** setting the motor will move 25,000 steps.

Example 2:

Command	Description
> **SFLX1X1**	Set bits 4 and 6
> **IFFL1Ø11XX1Ø**	If bits 1, 4, 5, and 7 are set and bits 0 and 6 are not set do the following commands
> **A1Ø**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D5ØØØ**	Sets distance to 5,000 steps
> **G**	Executes the move (Go)
> **NIF**	End of statement
> **IFFLXXX1**	If bit 4 on SFL is set, do the following commands.
> **A1Ø**	Sets acceleration to 10 rps^2
> **V2**	Sets velocity to 2 rps
> **D-25ØØØ**	Sets distance to 25,000 steps in the opposite direction
> **G**	Executes the move (Go)
> **NIF**	End of statement

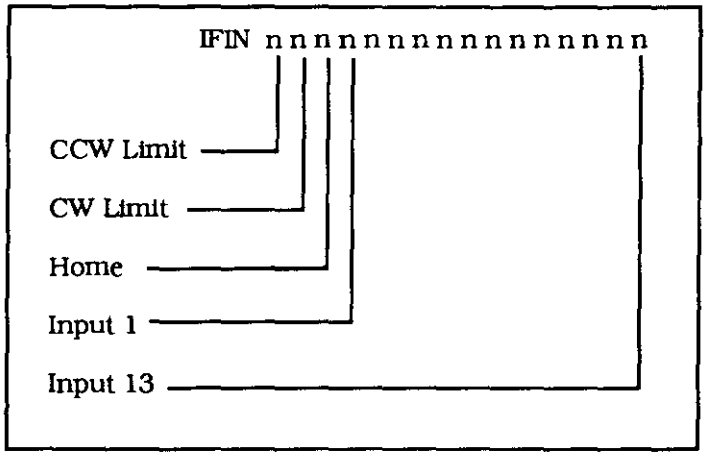
There are 2 IF_THEN statements on Example 2. The first one does not match the user flag, and skips to the command after **NIF**. The second **IFFL** pattern does match the SFL (User flag) pattern, therefore the motor moves -25,000 steps.

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

IFIN

NAME: Compare Input Status
SYNTAX: <a>**IFIN**nnnnnnnnnnnnnnnn
DEFAULT: None
RANGE: n = 0, 1, X
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command used the input pattern (16 inputs) of the JSI Controller to execute the conditional commands. This command is very similar to the IF_THEN statement in Basic Programming. The difference is, the 16 input bits set by various conditions will determine the next command to be executed. If the hardware input patterns match the **IFIN** input pattern, then the commands between IFIN and **NIF** will be executed. If patterns do not match, the command following the **NIF** will be executed. For this command, 1 means active and 0 means not active. You can use the **INL** command to set the active state (High or Low) of your inputs. An X means the JSI doesn't care whether that particular input is active or not. An **IF** command (**IFIN, IFER, IFFL**) cannot be used within another **IF** command.

This command is useful for branching and performing conditional moves. For example, depending on where you are, you may want to perform different moves.



SEE ALSO: **IN, INL, NIF**
EXAMPLE:

Command	Description
> **IFINXXX1Ø**	If input 1 is active and input 2 is not active issue the following commands
> **A1Ø**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D25ØØØ**	Sets distance to 25,000 steps
> **G**	Executes the move (Go)
> **NIF**	End of statement
> **IFINXXXØ1**	If input 1 is not active and input 2 is active, issue the following commands.
> **A1Ø**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D5ØØØ**	Sets distance to 5,000 steps in the opposite direction
> **G**	Executes the move (Go)
> **NIF**	End of statement
> **IFINXXX1**	If input 1 is active, do the following command.
>**1"DONE**	End message saying done
> **NIF**	End of statement

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

IN

NAME: Set Input Functions
SYNTAX: <a>**IN**nx
DEFAULT: n = 1 to 13, x = A
RANGE: n = 1 - 13, x = A-Z
VALID:
RESPONSE: See Example
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command configures the function for each of the 13 inputs on the JSI Controller. Each input may be configured to function as one of the following.

A *Trigger Input* - Used with **TR** command as a comparison input.

INPUT ON - Trigger Active
INPUT OFF - Trigger Inactive

B *Sequence Select Input* - Used to execute predefined sequences from remote inputs.

INPUT ON - Sequence Input Selected
INPUT OFF - Sequence Input not Selected

C *Kill Input* - Used to immediately halt execution of a move and to dump the sequence or command buffer. Same as executing a Kill (**K**) command.

INPUT ON - Kill initiated
INPUT OFF - Kill not initiated

D *Stop Input* - Used to decelerate the motor to a stop using the preset acceleration (**A**) rate. Same as executing a Stop (**S**) command

INPUT ON - Stop initiated
INPUT OFF - Stop not initiated

E *Amplifier Shutdown Input* - Used to enable and disable the amplifier. Same as ON and OFF command

INPUT ON - Shutdown active (Disable Command Out)
INPUT OFF - Drive Activated (Enable Command Out)

F *Pause/Continue Input* - Used to pause and continue the commands being executed. Same as Pause (**U**) and Continue (**C**) commands.

INPUT ON - Pause Execution
INPUT OFF - Continue Execution

G *Go Input* - Used to initiate a move. This is entered as a Go (**G**) command. To make the motor move you must bring the input from the OFF to the On state.

INPUT ON - Execute the move (Go)
INPUT OFF - Do not execute the move

H *Direction Input* - Used to toggle the direction of the motor. Same as entering the **H** command. You must toggle your input from OFF to On to change the direction.

I *Reserved*

J *Jog CW Input* - This input initiates jogging in the CW or positive direction. To use this input, you must enable jogging using the **OSE1** command and set both High Jog Velocity (**JVH**) and Low Jog Velocity (**JVL**)

INPUT ON - Start Jogging in CW direction
INPUT OFF - Stop Jogging

K *Jog CCW Input* - This input initiates jogging in the CCW or negative direction. To use this input, you must enable jogging using the **OSE1** command and define both High Jog Velocity (**JVH**) and Low Jog Velocity (**JVL**)

INPUT ON - Start Jogging in CCW direction
INPUT OFF - Stop Jogging

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

L *Jog Speed Select Input* - Used to select 2 different velocities for jogging. High Jog Velocity is defined by JVH command and Log Jog Velocity defined by JVL command. If not used, then jogging defaults to the low jog velocity.

INPUT ON - High Jog Velocity selected
INPUT OFF - Low Jog Velocity selected.

M *Terminate Loop Input* - Used to terminate the loop after finishing the current pass. Same as executing a **Y** command

INPUT ON - Terminate Loop
INPUT OFF - Terminate Loop not activated

N *Data Input* - Used to load parallel bytes of data into the JSI Controller. You can use this input to strobe data for distance, velocity, loop counts, and sequence inputs. You can only define up to 8 inputs as data inputs. See **DRD, LRD, XRD,** and **VRD** for detailed explanation of the usage of the data input.

INPUT ON - Data input true
INPUT OFF - Data input false

O *Reserved*

P *Memory Lock Input* - Used to prevent editing of sequence and lock out the usage of some of the commands. This input is useful if you do not want anyone to get access to your program or tuning parameters.

INPUT ON - Lock out sequence editing and commands
INPUT OFF - Do not lock out commands

Commands Locked Out:
CDB, CDG, CDM, CFB, CIG, CIL, CIM, CIP, COCL, COCH, COP, COV, CPB, CPE, CPG, CPM, CVS, CMR, RFS, RIFS, XE, XD, SV

The controller will report its setting, but you cannot change or use those commands while you are in a locked out mode.

Q *Reserved*

R *Reset Input* - Used to invoke a software reset of the controller. This is the same as sending a Reset (Z) to the JSI.

INPUT ON - Reset the Controller
INPUT OFF - Do not reset the Controller

S *Go Home Input* - Used to initiate a Go Home command. This input is the same as executing a GH command. You must define Go Home Velocity (**GHV**) and Go Home Acceleration (**GHA**) before executing homing

INPUT ON - Execute Go Home
INPUT OFF - Do not execute Go Home

T *Position Zero Input* - Used to set the present position as absolute zero position. The same as executing a **PZ** command.

INPUT ON - Zero the Absolute Position Counter
INPUT OFF - Do not zero the Absolute Position Counter

U *User Fault Input* - Input used to indicate from other devices that a fault condition exists. This input might be a pushbutton, PLC, or a Sensor Input.

INPUT ON - Fault condition exists
INPUT OFF - No fault condition exists.

When user fault is activated, the error code 66 will flash on and off. This fault output is latched, to clear the fault you must either reset the drive, cycle, power, or issue an **ON** command through the RS-232C interface.

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

V *Data Valid Input* - Input used with **STR** command to the handshake the data transfer from the interface to JSI Input line. It is used to input Distance (**DRD**). Velocity (**VRD**), Loop count (**LRD**), and the sequence number (**XRD**). As soon as we enable one of the inputs as data valid, the data will not be read until a data valid line becomes active. If an input is not assigned to data valid, then the data loading occurs synchronously according to the Strobe (**STR**) command.

INPUT ON - Data Valid
INPUT OFF - Data not Valid

W *Data Sign Input* - Input used with the **DRD** command to indicate the sign of the distance value being loaded. In an input is not set to a **W** function, then the sign defaults to a plus sign.

INPUT ON - Sign is negative
INPUT OFF - Sign is positive.

Z *No Function* - The input serves no assigned function.

If you use any other letters that particular input will set to Z. The ON and OFF state of all 13 inputs can be defined using the INL (Input Polarity) command. Using this command, you can set your input to be either active high, or active low.

SEE ALSO: **TR, K, S, A, ON, OFF, PS, C, G, H, JVH, JVL, OSE, Y, Z, GH, PZ, DRD, VRD, LRD, XRD, STR**

EXAMPLE:

Command	Description
> **IN1C**	Set up input 1 as Kill (K) input
> **IN2G**	Set up input 2 as Go (G) input
> **IN3J**	Set up input 3 as CW Jog input
> **IN4K**	Set up input 4 as CCW Jog input
> **IN5L**	Set up input 5 as jog velocity Select input
> **OSE1**	Enable Jogging
> **JVH1Ø**	Set high speed jog velocity to 10 rps
> **JVL1**	Set low speed jog velocity to 1 rps
> **1IN**	Status of all the inputs are reported

```
*ØA__CCW_LIMIT_(STATUS_ON)
*ØB__CW_LIMIT_(STATUS_ON)
*ØC__HOME_LIMIT_(STATUS_OFF)
*Ø1_C_KILL_INPUT_(STATUS_OFF)
*Ø2_G_GO_INPUT_(STATUS_OFF)
*Ø3_J_JOG_CW_INPUT_(STATUS_OFF)
*Ø4_K_JOG_CCW_INPUT_(STATUS_ON)
*Ø5_L_JOG_SPEED_SELECT_INPUT_(STATUS_ON)
*Ø6_A_TRIGGER_INPUT_(STATUS_ON)
*Ø7_A_TRIGGER_INPUT_(STATUS_ON)
*Ø8_A_TRIGGER_INPUT_(STATUS_OFF)
*Ø9_A_TRIGGER_INPUT_(STATUS_ON)
*1Ø_A_TRIGGER_INPUT_(STATUS_OFF)
*11_A_TRIGGER_INPUT_(STATUS_ON)
*12_A_TRIGGER_INPUT_(STATUS_ON)
*13_A_TRIGGER_INPUT_(STATUS_OFF)
```

1IN1 *Ø1_C_KILL_INPUT_(STATUS_OFF)

Status of only input 1 is reported back.

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

INL

NAME: Set Active Input Level
SYNTAX: <a>**INL**n
DEFAULT: n = 0
RANGE: n = 0 or 1, x - LOW or HIGH
VALID:
RESPONSE: a**INL** *n_INPUT_SIGNAL_LEVEL_ACTIVE_X[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command configures the voltage level that the JSI considers to be an active input signal.

INLØ Sets a low level (closed) as an active signal
INL1 Sets a high level (Open) as an active signal

This command allows you to use either normally closed (INL1) or normally open (**INLØ**) switches.
The maximum voltage for inputs are 30VDC, and it is current limited.

SEE ALSO: **IN**
EXAMPLE:

Command	Description
> **INL1**	Set a high level (open) as an active signal
> **AIØ**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D5ØØØ**	Sets distance to 5,000 steps

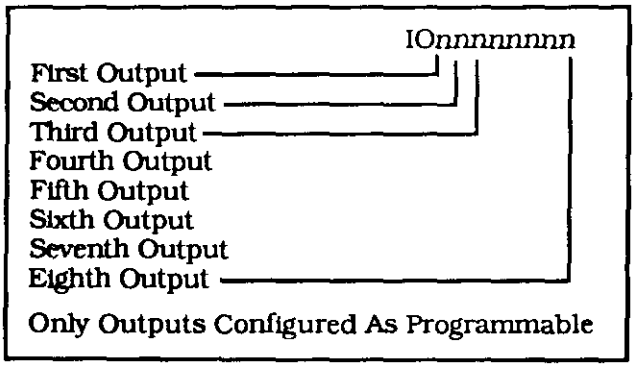
> **1INL** *1_INPUT_SIGNAL_LEVEL_ACTIVE_HIGH[cr]

The JSI responds with the active level of the inputs after entering these commands.

IO

NAME: Immediate Output
SYNTAX: <a>**IO**<n>
DEFAULT: None
RANGE: n = 0, 1, or X
VALID:
RESPONSE: a**IO** *nnnn_nnnn[cr]
TYPE: Programming
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: This command immediately sets the output bits configured as programmable outputs as specified in the **IO**.command. It does not affect the output state of outputs not configured as programmable.This command can only be used by having a host with a RS-232C interface on-line with the indexer. Typically, this command is used to turn a peripheral device on or off any time during the machine process.

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



Note that programmable output bits are configured by the **OUT** command. The voltage level represented as an active state is defined with the **OUTL** command. The order of the output pattern is programmable outputs from Output 1 to Output 8 (as labeled on the unit).

SEE ALSO: **O, OUT, OUTL**

EXAMPLES:

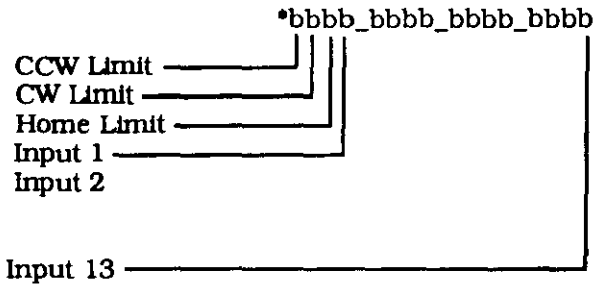
Command	Description
> **OUT5**	Sets output 5 as a programmable output
> **OUT6**	Sets output 6 as a programmable output
> **MN**	Sets to preset mode
> **A5**	Sets acceleration to 5 rps^2
> **V1Ø**	Sets velocity to 10 rps
> **D25ØØØ**	Sets distance to 25,000 steps
> **IO1X**	Sets Programmable Output 1 (labeled as Output 5) ON and leaves Programmable Output 2 (labeled as output 6) unaffected
> **G**	Executes the move (Go)
> **IOØ1**	Sets Programmable Output 1 (Output 6 label) OFF and leaves Programmable Output 2 (output 6 label) ON

If the **IOØ1** command is issued while the motor is moving, Output 1 turns off, and Output 2 turns on. This is different from the Output (**O**) command. The **O** command is buffered and executes after the move is finished.

IS

NAME: Input Status Report
SYNTAX: a**IS**
DEFAULT:
RANGE: b = 0, 1
VALID:
RESPONSE: a**IS** *bbbb_bbbb_bbbb_bbbb[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: Reports the status of the 16 discrete inputs. The actual function of the input is determined by the **IN** command. The active voltage level is determined by the **INL** command.

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



This is not a software status. It will report the actual hardware status of the inputs. This command is useful for troubleshooting an application.

SEE ALSO: **IN, INL**

EXAMPLES:

Command	Description
> 1IS	*1ØØØ_1ØØØ_ØØØ1_Ø11Ø[cr]

The CCW limit and inputs 2, 9, 11 and 12 are active.

JA

NAME: Jog Acceleration
SYNTAX: <a>**JA**n
DEFAULT: n = 99.000 rps^2
RANGE: n = .001 - 999.999
VALID:
RESPONSE: a**JA** ***JA**n[cr]
TYPE: Motion
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The jog acceleration command allows you to specify the linear acceleration rate that will be applied to reach constant jog velocity. The value is stored in nonvolatile memory.

SEE ALSO: **JVL, JVH, OSE**

EXAMPLES:

Command	Description
> JA3Ø	Sets jog acceleration to 30 rps^2
> JVL2	Sets low jog velocity to 2 rps

Motion ramps to a velocity of 2 rps at an acceleration rate of 30 rps when jog the input is activated.

JVH

NAME: Jog Velocity (High)
SYNTAX: <a>**JV**n
DEFAULT: n = 10
RANGE: n = .001 to 99.999 rps
VALID:
RESPONSE: a**JVH*JVH**n[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional

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

DESCRIPTION: This command defines the speed at which the motor will run when Jog input has been turned on and the high jog velocity is selected. You can define 2 different Jogging velocities:

JVL Low speed jog
JVH High speed jog

Typically you will use high speed jog to move the load to the vicinity of where you want to be. Then you can use the low speed jog to position your load accurately.

The jog speed (low or High) and the jog inputs (Jog CW or Jog CCW) must be defined using the IN command. The acceleration rate of the jog can be specified using the JA command. The default for the jog speed is low jog velocity (**JVL**).

SEE ALSO: **IN, INL, JA, JVL**
EXAMPLE:

Command	Description
> **IN1L**	Set input 1 as the jog velocity select (high or low)
> **IN2J**	Set input 2 as Jog input in the positive direction
> **IN3K**	Set input 3 as Jog input in the negative direction
> **JA1Ø**	Jog Acceleration is set to 10 rps^2
> **JVL.1**	Set Low jog velocity to .1 rps
> **JVH1Ø**	Set high jog velocity to 10 rps
> **INLØ**	Set the active level of the jog input to low

JVL

NAME: Jog Velocity (Low)
SYNTAX: <a>**JVL**n
DEFAULT: n = 1
RANGE: n = .01 to 99.999 rps
VALID:
RESPONSE: a**JV L *JVL**n[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional
DESCRIPTION: This command defines the speed at which the motor will run when Jog input has been turned on. You can define 2 different Jogging velocities:

JVL Low speed jog
JVH High speed jog

Typically you will use high speed jog to move the load to the vicinity of where you want to be. Then you can use the low speed jog to position your load accurately.

The jog speed (low or High) and the jog inputs (Jog CW or Jog CCW) must be defined using the IN command. The acceleration rate of the jog can be specified using the JA command. The default for the jog speed is the **JVL** value.

SEE ALSO: **IN, INL, JA, JVH**
EXAMPLE:

Command	Description
> **IN1L**	Set input 1 as the jog velocity select (high or low)
> **IN2J**	Set input 2 as Jog input in the positive direction
> **IN3K**	Set input 3 as Jog input in the negative direction
> **JA1Ø**	Jog Acceleration is set to 10 rps^2
> **JVL.1**	Set low jog velocity to .1 rps
> **JVH1Ø**	Set high jog velocity to 10 rps
> **INLØ**	Set the active level of the jog input to low

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

K

NAME: Kill
SYNTAX: <a>**K**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Motion
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: The Kill (**K**) command is an emergency stop command and should only be used as such. This command causes indexing to cease immediately. There is *NO* deceleration. If Kill causes the motor to lose torque (i.e., large loads at high speed), the load could be driven past limit switches and cause damage to the mechanism and possibly to the operation.

In addition to stopping , the **K** command will terminate a loop, end a time delay and clear the command buffer.

SEE ALSO: **S**
EXAMPLES:

Command	Description
> **A5**	Sets acceleration to 5 rps^2
> **V2**	Sets velocity to 2 rps
> **MC**	Sets mode to continuous
> **G**	Executes the move (Go)
•	
•	
•	
> **K**	Stops the motion instantly

L

NAME: Loop
SYNTAX: <a>**L**<n>
DEFAULT: None
RANGE: n = 0-99,999,999
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
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. If you enter the **L** command without a value specified for n, or with a 0, subsequent commands will be repeated continuously. Loops may be nested up to 16 levels deep.

The Stop Loop (**Y**) command terminates a loop at the end of the current repetition. The Immediate Pause (**U**) command allows you to temporarily halt loop execution. You can use the Continue (**C**) command to resume loop execution.

SEE ALSO: **Y, N, U, C**
EXAMPLES:

Command	Description
> **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

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

LA

NAME: Limit Acceleration
SYNTAX: <a>**LA**n
DEFAULT: n = 900.000 rps^2
RANGE: n = 0.001 - 9999.999 rps^2
VALID:
RESPONSE: a**LA** *LAn[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The Limit Acceleration (**LA**) command allows you to define the deceleration rate that should be used when an end-of-travel limit or software travel limit is encountered. The value is stored in nonvolatile memory. This command is useful if you do not want an abrupt stop upon encountering a limit. However, you should be careful to specify a deceleration rate that will stop the load before it can do any damage. Normally, limit switches are placed so that the load safely decelerates to a stop.

SEE ALSO: **A, LD, SLD**
EXAMPLES:

Command	Description
> **LA5Ø**	Motion decelerates at 50 rps^2 when it encounters an end of travel limit.

LD

NAME: Limit Disable
SYNTAX: <a>**LD**n
DEFAULT: n = 0
RANGE: n = 0 - 3
VALID:
RESPONSE: a**LD** *Ø_CW_AND_CCW_LIMITS_ENABLED[cr] or
*1_CCW_LIMITS_ENABLED[cr] or
*2_CW_LIMITS_ENABLED[cr] or
*3_NO_LIMITS_ENABLED[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Saveable in Sequence
DESCRIPTION: The Limit Disable (**LD**) command allows you to enable/disable the end-of-travel limit switch protection. The **LDØ** condition does not allow motion without properly installing the limit inputs. If you want motion without wiring the limits, you must issue the **LD3** command.

Enable CCW and CW limits	n = Ø (Default)
Disable CW limits	n = 1
Disable CCW limits	n = 2
Disable CCW and CW limits	n = 3

SEE ALSO: **RA, SLD**
EXAMPLES:

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

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

LED

NAME: Illuminate Number on LED Display
SYNTAX: <a>**LED**<n>
DEFAULT:
RANGE: n = 0-99
VALID:
RESPONSE:
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional
DESCRIPTION: This command allows the user to set a number to the 2 digit LED display on the JSI Drive. This command is useful when you want the JSI to interact with the operator. For example, you might define number 90 as the moving status, or number 95 as stopped status. The LED command without a value will cancel a previous LED command as the display in blanked.

SEE ALSO:
EXAMPLE:

Command	Description
> **PS**	Pause execution of following commands until the Continue (C) command is issued
> **A1Ø**	Set acceleration to 10 rps^2
> **V5**	Set velocity to 5 rps
> **D25ØØØ**	Set move distance to 25,000 steps
> **LED9Ø**	Turn on 2 digit LEDs to 90 (Moving)
> **G**	Execute the move (Go)
> **LED95**	Turn on 2 digit LEDs to 95 (stopped)
> **C**	Continue execution

LF

NAME: Line Feed
SYNTAX: a**LF**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: a**LF** [lf]
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: When you issue the Line Feed (**LF**) command, the system transmits a line feed character (ASCII Code 10) over the communications link. This command is useful when you send messages to control a display screen, or to signal a move completion.

SEE ALSO: **CR**
EXAMPLES:

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

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

LRD

NAME: Read Loop Count via Parallel Input/Output
SYNTAX: <a>**LRD**
DEFAULT: None
RANGE:
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command initiates the controller to read Loop Count values from the parallel inputs. You must set up the inputs as Data Inputs using the **IN** command. You must also set up outputs as strobe outputs. Typically, you would set up 8 inputs as DATA Inputs and 5 Outputs as strobe outputs using the **OUT** command. You must also set up the strobe output delay time (typically greater than a PLC Scan Time if using a PLC, or a minimal debounce time if using thumbwheel switches) so that the device you are getting data from will know when the JSI is ready for data. Sequence of events take place when we enter the **LRD** command:

Step 1 The lowest Output bit designated as strobe output will go on and stay on for strobe output delay time defined by **STR** command.

Step 1A If an input is designated as a data valid input, then the strobe output will stay active until the data valid input is active.

Step 2 The JSI will read the parallel inputs designated as DATA inputs. The lowest inputs will be the least significant bit. The JSI reads 8 bits (2 BCD digits) and stores them into two digits on the loop register.

Step 3 Next Strobe output bit designated as strobe output will go on and stay on for delay time defined by **STR** command.

Step 3A If an input is designated as a data valid input, then the strobe output will stay active until the data valid input is active

Step 4 The JSI will read the parallel inputs designated as DATA inputs are placed in the lowest digits on the loop register. Previously entered values are shifted two positions to the left.

Step 5 Steps 3 through 4 are repeated as many times as the strobe outputs are available. You can use up to 5 different outputs as strobe outputs. The last two digits read are the least significant digit of the loop value.

SEE ALSO: **STR, IN, OUT**
EXAMPLE: The following example shows how the JSI reads the loop value of 721. We will use inputs 1 through 8 for DATA inputs and outputs 1 through 4 for Strobe Outputs.

Step 1 Type the following:

Command	Description
OUT1J	Set Output 1 as Strobe Output
OUT2J	Set Output 2 as Strobe Output
OUT3J	Set Output 3 as Strobe Output
OUT4J	Set Output 4 as Strobe Output
IN1N	Set Input 1 as a Least Significant DATA input
IN2N	Set input 2 as a DATA input
IN3N	Set input 3 as a DATA input
IN4N	Set input 4 as a DATA input
IN5N	Set input 5 as a DATA input
IN6N	Set input 6 as a DATA input
IN7N	Set input 7 as a DATA input
IN8N	Set input 8 as a the Most significant DATA input
STR5ØØ	Set strobe output delay time to 500 msec.
A1Ø	Sets acceleration to 10 rps^2
V5	Set velocity to 5 rps
D2ØØØ	Set Distance to 2,000 steps

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

Step 2 As soon as you enter the Read Loop count via Parallel Input **(LRD)** command , the following should happen

Type the following:

Command	Description
LRD	Start reading from Parallel input
G	Execute the move
T2	Wait 2 seconds
N	End of loop

Step 3 The JSI will turn on OUT1 for 500 msec.

Step 4 You must have your inputs 1 through 8 to the following settings.

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
	Ø					Ø	

The current setting of the loop count is
ØØØØØØØØØØ

Step 5 After the JSI reads the input, OUT1 turns off and OUT2 will automatically go on for 500 msec.

Step 6 You must have your inputs 1 through 8 set to the following settings.

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
	Ø					Ø	

Previously entered value of ØØ shifts 2 locations to the left and ØØ is placed on the 2 lowest positions. The current setting of the Loop count value is:
ØØØØØØØØØØ

Step 7 OUTZ automatically goes on for 500 msec.

Step 8 You must have your inputs 1 through 8 set to the following settings:

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	OFF	OFF	OFF	OFF	ON	ON	ON
	Ø					7	

Previously entered values ØØØØ shifts 2 locations to the left and Ø7 is placed on the two lowest positions. The current setting of the Loop count value is:
ØØØØØØØØØ7

Step 9 OUT4 automatically goes on for 500 msec.

Step 10 You must have your inputs 1 through 8 set to the following settings.

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	OFF	ON	OFF	OFF	OFF	OFF	ON
	2					1	

Previously entered values ØØØØØ7 shifts two locations to the left and 21 is placed on the two lowest positions. The final setting of the loop count value becomes
ØØØØ721

Step 11 Since the fifth strobe output is not defined, the JSI now goes to the next command in the buffer and executes the move 721 times with 2 second delay between each move.

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

MC

NAME: Mode Continuous
SYNTAX: <a>**MC**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Motion
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
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 Move Normal (**MN**) command.

The JSI uses the Acceleration (**A**) and Velocity (**V**) commands to reach continuous velocity.

Using the Time Delay (**T**), Trigger (**TR**), and Velocity (**V**) commands, you can achieve basic velocity profiling.

SEE ALSO: **MN, A, V, TR, T**
EXAMPLES:

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.

MN

NAME: Mode Normal
SYNTAX: <a>**MN**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Motion
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
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 is used to change the mode of operation from Mode Continuous (**MC**).

SEE ALSO: **MC, D, A, V, MPA, MPI**
EXAMPLES:

Command	Description
> **MN**	Set positioning mode to preset
> **MPI**	Set to incremental positioning mode
> **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)

Motor turns 1,000 steps in the CW direction after the **G** command is issued.

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

MPA

NAME: Mode Position Absolute
SYNTAX: <a>**MPA**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command sets the positioning mode to absolute. In this mode all move distances are referenced to absolute zero. Note that in Mode Normal (**MN**) and Mode Position Absolute (**MPA**), giving two consecutive go (**G**) commands will cause the motor to move once, since the motor will have achieved its desired absolute position at the end of the first move.

Mode Position Absolute (**MPA**) is most useful in applications that require moves to specific locations.

You can set the absolute counter to zero by cycling power or issuing a Position Zero (**PZ**) command.

SEE ALSO: **MN, MPI, D, PZ, SP**
EXAMPLES:

Command	Description
> **MN**	Set mode normal (preset).
> **MPA**	Set position mode absolute.
> **A5**	Set acceleration to 5 rps^2.
> **V1Ø**	Set velocity to 10 rps.
> **D25,ØØØ**	Set distance to 25,000 steps.
> **G**	Motor will move to absolute position 25,000.
> **D12,5ØØ**	Set distance to +12,500 steps
> **G**	Motor will move to absolute position +12,500 steps.

MPI

NAME: Mode Position Incremental
SYNTAX: <a>**MPI**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence.
DESCRIPTION: This command sets the positioning mode to incremental. In incremental mode all move distances specified with the Distance (**D**) command will be referenced to the current position. Mode Position Incremental (**MPI**) is most useful in applications that require repetitive movements, such as feed to length applications.

SEE ALSO: **MN, MPA, D**
EXAMPLES:

Command	Description
> **MN**	Set positioning mode normal (preset)
> **MPI**	Set positioning mode incremental
> **A5**	Sets acceleration to 5 rps^2
> **V1Ø**	Sets velocity to 10 rps
> **D1Ø,ØØØ**	Sets distance of move to 10,000 steps
> **G**	Move 10,000 steps CW
> **G**	Move another 10,000 steps CW

The motor moves 10,000 steps CW after each **G** command.

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

N

NAME: End of Loop
SYNTAX: <a>**N**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command marks the end of loop. You can use this command in conjunction with the Loop (**L**) command. All buffered commands that you enter between the **L** and **N** commands are executed as many times as the number that you enter following the **L** command.

SEE ALSO: **L, PS, C, Y, L**
EXAMPLES:

Command	Description
> **PS**	Pauses the execution of buffered commands until the JSI receives a Continue (**C**) command
> **MN**	Sets move to mode normal
> **A5**	Sets acceleration to 5 rps^2
> **V5**	Sets velocity to 5 rps
> **D1ØØØØ**	Sets move distance to 10,000 steps
> **L5**	Loops the above series of commands five times
> **G**	Executes the move (Go)
> **N**	Ends the loop
> **C**	Clears pause and executes all the buffered commands

The motor makes a 10,000 step move 5 times.

NIF

NAME: End of IF commands
SYNTAX: <a>**NIF**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command marks the end of an **IF** command. The JSI has three **IF** commands.

IFFL If the User Flag pattern matches, execute the following commands
IFER If certain error conditions exist, execute the following commands
IFIN If the 16 input pattern matches, execute the following command.

When you use any of the three commands, listed above, you would end the commands, following the IF command with the NIF.

In case the IF condition does not match, the commands between IF and NIF will not be executed, and the program will skip to the command following the NIF command.

SEE ALSO: **IFFL, IFER, IFIN**

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

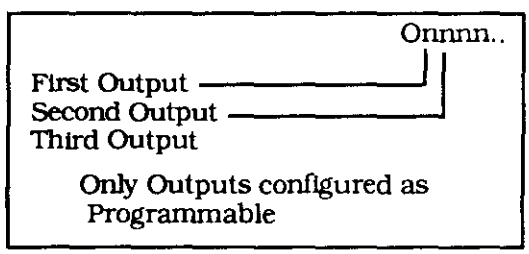
EXAMPLE:

Command	Description
> **INLØ**	Set active input level low.
> **IF1NXXX1Ø**	If input 1 is active and input 2 is not active issue the following commands then skip to the command following the **NIF** command
> **A1Ø**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D25ØØØ**	Sets distance to 25,000 steps
> **G**	Executes the move (Go)
> **NIF**	End of statement
> **IFINXXXØ1**	If input 1 is not active and input 2 is active, issue the following commands then skip to the command following the **NIF** command.
> **A1Ø**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D5ØØØ**	Sets distance to 5,000 steps in the opposite direction
> **G**	Executes the move (Go)
> **NIF**	End of statement
> **IFINXXX1**	If input 1 is active, do the following command, then skip to the command following the **NIF** command
> **1"DONE**	Transmit message saying done
> **NIF**	End of statement

O

NAME: Output
SYNTAX: <a>**O** <n>
DEFAULT: None
RANGE: n = 0, 1 or x
VALID:
RESPONSE: a**O** *nnnn_nnnn[cr]
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The Output (**O**) command turns the output bits configured as programmable outputs on and off. It does not affect the output state of the outputs not configured as programmable. This is used for signaling remote controllers, turning on LED's, or sounding alarms. The output can indicate that the motor is in position, about to begin its move, or is at constant velocity, etc.

The programmable output bits are configured by the **OUT** command. The voltage level represented as an active state is defined with the **OUTL** command. The order of the output pattern is programmable outputs from output 1 to output 8 (as labeled on the unit).



SEE ALSO: **OUT, IO, OUTL**
EXAMPLES:

Command	Description
> **OUT2A**	Set output 3 as a programmable output
> **OUT4A**	Set output 4 as a programmable output
> **A1Ø**	Set acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D2ØØØØ**	Set move distance to 20,000 steps
> **OØ1**	Set programmable output 1 (labeled as Output 3) off and output 2 (labeled as Output 4) on
> **G**	Executes the move (Go)
> **OXØ**	After the move ends, turn off programmable output 2 (labeled as Output 4)

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

OFF

NAME: Off
SYNTAX: <a>**OFF**
DEFAULT: **ON**
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Immediate, Device Address Optional, Never Saved.
DESCRIPTION: This command allows you to disable the JSI command OUT (voltage or current output). When you issue an **OFF** command, the fault LED light will be on to indicate that the command OUT output is shut down. You must issue an **ON** command to enable the command OUT output and clear the fault. The contents of the buffer will be cleared when you execute this command.

This command is similar to the **ST1** command. However, this command is immediate, and **ST1** is buffered.

SEE ALSO: **ON, ST1, STØ**
EXAMPLES:

Command	Description
> **OFF**	Turns off the JSI command OUT output

ON

NAME: On
SYNTAX: <a>**ON**
DEFAULT:
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: This command turns enables the JSI command OUT (voltage or current output). If you issue the **OFF** command to disable the JSI command OUT output, issuing the **ON** command re-enables the output (voltage or current). The command out configuration is set with the **COV** command to a voltage source, and to a current source with the **COCL** and **COCH** commands.

The **ON** command stops existing motion. This command is similar to the **STØ** command. The **ON** command, however, is immediate and **STØ** is buffered.

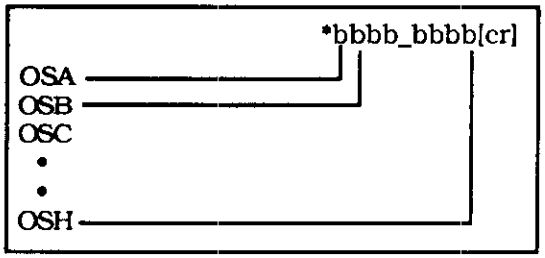
SEE ALSO: **OFF, ST1, STØ**
EXAMPLES:

Command	Description
> **ON**	Turns on the JSI command OUT output

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

OS

NAME: Function Setup Report
SYNTAX: a**OS**
DEFAULT: None
RANGE: b = 0, 1
VALID:
RESPONSE: a**OS** *bbbb_bbbb[cr]
TYPE: Status
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: Reports the status of **OS** command (**OSA-OSH**) settings.



SEE ALSO: **OSB, OSC, OSD, OSE, OSG, OSH**

OSB

NAME: Back up to Home Switch
SYNTAX: <a>**OSB**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION:
OSBØ Do not back up to home switch
OSB1 Back up to home switch

If you do not enabled function, the JSI will consider the motor at Home if the home input is active at the end of deceleration after encountering the active edge of the Home region. If you enable this function, the JSI will decelerate the motor to a stop after encountering the active edge of the Home region, and then move the motor in the opposite direction of the initial Go Home move with the GHF velocity until the active edge of the Home region is encountered. The JSI will then consider the motor at Home. This occurs regardless of whether or not the home input is active at the end of the deceleration of the initial Go Home move.

SEE ALSO: **GH, OSC, OSD, OSG, OSH, OS**

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

OSC

NAME: Define Active State of Home Switch
SYNTAX: <a>**OSC**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION:

OSCØ	Active state of home input is a low signal level
OSC1	Active state of home input is a high signal level

This command sets the active state of the home input. It enables you to use either a normally closed or a normally open switch for homing. **OSCØ** requires a low voltage signal to activate the home limit input.

OSC1 requires a high-voltage signal to activate the home limit input.

SEE ALSO: **GH, OS**
EXAMPLES:

Command	Description
OSC1	Sets home input's active state to a high state

OSD

NAME: Enable Encoder Z Channel Input
SYNTAX: <a>**OSD**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION:

OSDØ	Disable Z channel function during home
OSD1	Recognizes Z channel after encountering the home

The Z channel is used (in conjunction with a load activated switch connected to home limit) to determine the home position. The switch determines the home region. The Z channel determines the exact position in that region that will be used as the home reference.

SEE ALSO: **OSB, OSC, OSE, OSF, GH, GHA, GHF, GHV**
EXAMPLES:

Command	Description
OSDØ	Disable Z channel function

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

OSE

NAME: Jog Enable
SYNTAX: <a>**OSE**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The jogging functions are configured with the **IN** command, and are enabled with the **OSE** command.

OSEØ Jog Disable
OSE1 Jog Enable

SEE ALSO: **IN, JVL, JA, INL**

OSG

NAME: Final homing direction
SYNTAX: <a>**OSG**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: **OSGØ** Sets the final Home approach direction to be clockwise
OSG1 Sets the final home approach direction to be counterclockwise

This enables you to approach home from any direction yet stop at home repeatedly in the same edge of the switch

SEE ALSO: **OSH**

OSH

NAME: Reference Edge of Home Switch
SYNTAX: <a>**OSH**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: **OSHØ** Selects the clockwise side of the Home signal as the "edge" on which the final approach will stop

OSH1 Selects the counterclockwise side of the home signal as the "edge" on which the final approach will stop

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

The clockwise edge of the Home switch is defined as the first switch transition seen by the JSI when traveling off of the CW limit in the CCW direction. If n = 1, the counterclockwise edge of the Home switch will be referenced as the Home position. The counterclockwise edge of the Home switch is defined as the first switch transition seen by the JSI when traveling off of the CCW limit in the CW direction.

SEE ALSO: **OSG**

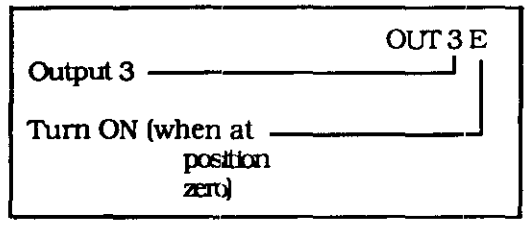
OUT

NAME: Output Functions
SYNTAX: <a>**OUT**nx
DEFAULT: x = A
RANGE: n = 1-8, x = A-n
VALID:
RESPONSE: a**OUT**n See Example
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command sets the functions for each of the eight outputs. All the outputs can be configured for different functions. The factory setting for all eight outputs are programmable output bits. The following table lists the functions available for each output bit.

A	*Programmable Output* - Used with **O** and **IO** commands
B	*Moving/Not Moving* - On when motion is occurring
C	*Sequence in Progress* - On when a sequence is being executed.
D	*At Limits* - On when the motor has reached either a software or hardware limit
E	*At Position Zero* - On when the absolute position counter is zero
F	*Fault* - On when a fault has occurred. See **RSE** command for possible fault conditions
G	*Deadband* - On when position error exceeds the deadband set by **CDB** command
H	*Amp Off* - Indicates when you turn off the drive using **OFF** or **ST1** command
I	*Reserved*
J	*Strobe* - Turns on to load bytes of parallel data. Indicates to the interface that the JSI is ready for data.
K	*Command Error* - On when RS-232C or BCD input has bad data
L	*Position Error Fault* - On when following error set by **CPE** command is exceeded. This situation also creates an error 20
M	*Reserved*
N	*CW Software Limit Reached* - CW Software limit set with **SL** command has activated
O	*Reserved*
P	*CCW Software limit reached* - CCW Software limit set with **SL** command has activated
Q	*Reserved*
R	*CW Hardware limit activated*
S	*CCW hardware limit activated*
T	*Output based on position* - This output goes on when a condition specified by **OUTP** command exists.
Z	*No Function*
OTHERS:	*No Function*

Outputs 1 through 8 can be configured as any one of the functions described above. For example, output 3 is set as an at position zero indicator::

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



SEE ALSO: **IN, OUTP, OUTL**

EXAMPLE 1: The OUT command also reports the status of each output bit.
a**OUT1** *1_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)

EXAMPLE 2: By not specifying the output bit number, JSI reports the status of all the output bits.
a**OUT**
*1_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*2_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*3_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*4_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*5_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*6_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*7_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*8_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)

OUTL

NAME: Set Active Output Level
SYNTAX: <a>**OUTL**n
DEFAULT: n = 0
RANGE: n = 0, 1, x = Low or High
VALID:
RESPONSE: a**OUTL** *n_OUTPUT_SIGNAL_LEVEL_ACTIVE_x[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command configures the voltage level that the JSI considers to be an active output signal.

OUTLØ Sets a low level (0V) as an active signal
OUTL1 Sets a high level (5-30V) as an active signal

Outputs 1 through 8 are open collector outputs. Therefore, you must supply power (5-30VDC) to run the outputs. The output is able to sink up to 300 mAmps of current. You must configure the outputs using the **OUT** command if you wish to use the outputs for anything other than programmable outputs.

SEE ALSO: **OUT**
EXAMPLE:

Command	Description
> **OUTLØ**	Set a low level (0V) active signal
> **OUT1A**	Set output 1 as a programmable output

> **1OUTL** *Ø_OUTPUT_SIGNAL_LEVEL_ACTIVE_LOW[cr]

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

OUTP

NAME: Output on Position
SYNTAX: <a>**OUTP**xn,<t>
DEFAULT: x = A, n = 0, t = 5
VALID:
RESPONSE: a**OUTP** *OUTPUT_ON_x_POSITION_n[cr]
*POSITION_OUTPUT_ON_TIME_t_MILLISECONDS_[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: An output can be configured to activate based on the position with the OUT command. The comparison position is entered as either an incremental or absolute position.

The command **OUTP** is configured as **OUTP**xn,<t> where x is A for Absolute or I for incremental mode, n is distance, and is the time in milliseconds the output is active (only in incremental mode).

In absolute mode, the output turns on when the current position is greater than the entered distance. The time is ignored in this mode.

In incremental mode, the output turns on each time the distance is traversed and stays activated for the entered time. The sign for the distance will convert to a plus sign as the distance is an absolute value.

SEE ALSO: **OUT**
EXAMPLE 1:

Command	Description
> **OUT2T**	Set Output 2 to activate based on position
> **OUTPA1ØØØ**	When the absolute position is greater than +1000, the output turns on

EXAMPLE 2:

Command	Description
> **OUT2T**	Set output 2 to activate based on position
> **OUTPI,5ØØØ,5ØØ**	The output will activate for 500 milliseconds every 5000 steps

PR

NAME: Absolute Position Report
SYNTAX: a**PR**
DEFAULT: None
RANGE: n = ±2,147,483,647
VALID:
RESPONSE: a**PR** *±n[cr]
TYPE: Status
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: Absolute position request; response is *snnnnnnnnnn[cr]. Reports motor position with respect to power up position or last time a **SP** command was issued. The units of distance are scaled by the **CMR** command. The absolute position counter can track up to +/- 2^{31} -1, or 2,147,483,647 steps. If the counter is overrun in the relative position mode (by running the motor continuously for long periods of time, 24 hours at 20 RPS and 5,000 steps per rev), the absolute position will be invalid.

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

In preset mode, response to this command will be reported after the move is done. In continuous mode, the response to this command will be reported after motor reaches constant velocity.

SEE ALSO: **MPI, MPA, MN, PZ, D**

EXAMPLES:

Command	Description
> **PZ**	Resets the absolute counter to zero
> **LD3**	Disable both CW & CCW limits
> **A1Ø**	Set Acceleration to 10 rps^2
> **V5**	Set velocity to 5 rps
> **D5ØØØ**	Set move distance to 5000 steps
> **G**	Executes the move (Go)
> **1PR**	Request absolute position report. Response should be (*+ØØØØØØ5ØØØ[cr])

PS

NAME: Pause

SYNTAX: <a>**PS**

DEFAULT: None

RANGE: None

VALID:

RESPONSE: None

TYPE: Programming

ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence

DESCRIPTION: This command pauses execution of a command string or sequence following the Pause (**PS**) command until the JSI receives a Continue (**C**) command. This command is useful if you need to enter a complete string of commands before you can execute your other commands.

This command is useful for interactive tests and in synchronizing multiple indexes that have long command strings.

SEE ALSO: **C**

EXAMPLES:

Command	Description
> **PS**	Pauses execution of following commands until the JSI receives the Continue (**C**) command
> **A5**	Sets acceleration to 5 rps^2
> **V5**	Sets velocity to 5 rps
> **D25ØØØ**	Sets move distance to 25,000 steps
> **G**	Executes the move (Go)
> **T2**	Delays the next move for 2 sec
> **G**	Executes the move (Go)
> **C**	Continues Execution

When the JSI receives the **C** command, the motor moves 25,000 steps twice with a 2 second delay.

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

PZ

NAME: Set Absolute Counter to Zero
SYNTAX: <a>**PZ**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command sets the absolute position counter to zero.

SEE ALSO: **MN, MPI, MPA, PR, D**
EXAMPLES:

Command	Description
> **MPA**	Make all preset moves with respect to absolute zero position
> **A1Ø**	Set Acceleration to 10 rps^2
> **V5**	Set Velocity to 5 rps
> **D25ØØ**	Set move distance to 2500 steps
> **G**	Executes the move (Go)
> **1PR**	Report Absolute Position (Response = ***+25ØØ**)
> **PZ**	Zero the absolute counter
> **1PR**	Report absolute position (Response = ***Ø**)

"

NAME: Quote
SYNTAX: a"x
DEFAULT: None
RANGE: X = Any ASCII character or number (in the range of 33–126 decimal), with a maximum string length of 18 characters.

VALID:
RESPONSE: **X** = The ASCII range of characters accepted by the command
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: Any characters entered after the quotation marks(") will be transmitted, exactly as they were entered, over the RS-232C link. A space entered by the space bar indicates the end of the command. A space is always sent after the last character in the string. This command is used during buffered moves or sequences, or to command other Compumotor devices to move. The ASCII range of characters accepted by the command is 33 - 126 (decimal).

SEE ALSO: **CR, LF**
EXAMPLE 1:

Command	Description
> **MN**	Set to mode normal (Preset Moves)
> **A1Ø**	Set acceleration to 10 rps^2
> **V5**	Set velocity to 5 rps
> **D125ØØ**	Set distance to 12,500 steps
> **G**	Executes the move (Go)
> **1"MOVE_DONE**	After motor finished the move, the Compumotor JSI will send the message MOVE_DONE out from the RS-232C port.

EXAMPLE 2:

Command	Description
> **MN**	Set to mode normal (Preset Moves)
> **A1Ø**	Set acceleration to 10 rps2
> **V5**	Set velocity to 5 rps
> **D125ØØ**	Set distance to 12,500 steps
> **G**	Executes the move (Go)
> **1"2XR1**	Once the move is done, Run Sequence 1 is commanded on a unit with device address 2.

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

QØ

NAME: Exit Velocity Profiling Mode
SYNTAX: <a>QØ
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The QØ command exits the velocity profiling mode. Motion will stop when QØ is issued.

SEE ALSO: Q1, RM
EXAMPLES: See Q1 command

Q1

NAME: Enter Velocity Profiling Mode
SYNTAX: <a>Q1
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The Q1 command enters the JSI in velocity profiling mode. Subsequent RM commands will cause an immediate change in motion velocity. Use QØ to exit this mode.

SEE ALSO: QØ, RM
EXAMPLES:

Command	Description
> Q1	Enter Velocity Profiling mode
> RMØ1ØØ	Go to RM velocity of (100 hex) RM rps
> RMØ5ØØ	Go to RM velocity of (500 hex) RM rps
> RM1ØØØ	Go to RM velocity of (1000 hex) RM rps
> RMØ5ØØ	Go to RM velocity of (500 hex) RM rps
> RMØ1ØØ	Go to RM velocity of (100 hex) RM rps
> QØ	Exit velocity profiling mode

Motion will stop when QØ command is entered.

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

R

NAME: Report JSI Status
SYNTAX: **aR**
DEFAULT: None
RANGE: x=R, B, S, C
VALID:
RESPONSE: aR *x[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: The Request JSI Status (**R**) command can be used to indicate the general status of the JSI. Possible responses are:

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

The following conditions will cause a response indicating that the JSI is busy:

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

The following conditions will cause a response indicating than an error exists.

* A feedback error condition exits.
* Go home failed
* Limit has been encountered
* Sequence execution was unsuccessful

When the response indicates that attention is required, more details on the error condition are available by using the **RA** or **RB** commands.

It is not recommended that this command be used in tight polling loops which could result in microprocessor over load. Inserting time delays can alleviate this problem.

This command is not intended to be used to determine if a move is complete. Rather it should be used after the move is complete to determine if there might be errors or faults.

Use a buffered status request command or a programmable output to indicate move completion.

SEE ALSO: **RA, RB**
EXAMPLE:

Command	Description
> 1R	(Response) = *R (JSI ready to accept a command and no error conditions require attention.)

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

RA

NAME: Limit Switch Status Report
SYNTAX: a**RA**
DEFAULT:
RANGE: x = @, A, B, C, D, E, F, G, H, I, J, K, L, M, N, O
VALID:
RESPONSE: a**RA** *x[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: The Limit Switch Status Report (**RA**) command responds with the status of the end of travel limits during the last move as well as the present condition. This is done by responding with one of 16 characters representing the conditions listed below.

Response	Last Move Terminated By		Limit Switch Active	
Character	CW Limit	CCW Limit	CCW Limit	CW Limit
*@	NO	NO	NO	NO
*A	YES	NO	NO	NO
*B	NO	YES	NO	NO
*C	YES	YES	NO	NO
*D	NO	NO	YES	NO
*E	YES	NO	YES	NO
*F	NO	YES	YES	NO
*G	YES	YES	YES	NO
*H	NO	NO	NO	YES
*I	YES	NO	NO	YES
*J	NO	YES	NO	YES
*K	YES	YES	NO	YES
*L	NO	NO	YES	YES
*M	YES	NO	YES	YES
*N	NO	YES	YES	YES
*O	YES	YES	YES	YES

The **RA** command is useful when the motor will not move in either or both directions. The report back will indicate whether or not the last move was terminated by one or both end of travel limits.

RB

NAME: Loop, Pause, Shutdown, Trigger Status Report
SYNTAX: a**RB**
DEFAULT: None
RANGE: x = @, A, B, C, D, E, F, G, H, I, J, K, L, M, N, O
VALID:
RESPONSE: a**RB** *x[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: Responds with the form *x[cr] where x = "@" - "0"

This command reports the status of four functions within the JSI: Loop in Progress, Pause in Progress, Trigger Wait Active and Shutdown active.

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

Response Character	Loop Active	Pause Active	Shutdown Active	Trigger Active
*@	NO	NO	NO	NO
*A	YES	NO	NO	NO
*B	NO	YES	NO	NO
*C	YES	YES	NO	NO
*D	NO	NO	YES	NO
*E	YES	NO	YES	NO
*F	NO	YES	YES	NO
*G	YES	YES	YES	NO
*H	NO	NO	NO	YES
*I	YES	NO	NO	YES
*J	NO	YES	NO	YES
*K	YES	YES	NO	YES
*L	NO	NO	YES	YES
*M	YES	NO	YES	YES
*N	NO	YES	YES	YES
*O	YES	YES	YES	YES

SEE ALSO: **R, RA, TR, PS, L, ST**

EXAMPLES:

Command	Description
> **1RB**	After using 1RB, response came back as ***A.** This means that the JSI is currently executing a loop .

RFS

NAME: Return Drive Parameters to Factory Setting

SYNTAX: <a>**RFS**

DEFAULT: None

RANGE: None

VALID:

RESPONSE: None

TYPE: Setup

ATTRIBUTES: Immediate, Device Address Optional, Never Saved

DESCRIPTION: Return the drive to factory setting. The Save (**SV**) command must be used to retain the values in nonvolatile memory. The following settings are changed:

Proportional Gain (**PG**) set to 10
Proportional Maximum (**CPM**) set to 500
Integral Gain (**CIG**) set to 0
Integral Maximum (**CIM**) set to 1000
Differential Gain (**CDG**) set to 0
Differential Maximum (**CDM**) set to 100
Dead Band (**CDB**) set to 0
Motion Resolution (**CMR**) set to 4000
Maximum Position Error (**CPE**) set to 4000
Integrator Sum Limit (**CIL**) set to 65,535

SEE ALSO: **RIFS, SV**

EXAMPLES: None

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

RG

NAME: Go Home Status Report
SYNTAX: a**RG**
DEFAULT: None
RANGE: x = @,A
VALID:
RESPONSE: a**RG** *x[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: Go Home Status request; responds with either *@ or *A indicating success or failure of last go home attempt.

Response	Go Home Successful
*@	NO
*A	YES

SEE ALSO: **GH, R, RA, RB**

RIFS

NAME: Return JSI to Factory Settings
SYNTAX: <a>**RIFS**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: Execution of this command will cause the following JSI parameters to return to factory default settings.

Configure Feed Back (**CFB**) set to 1 (incremental encoder)
4000 (Feedback resolution)
Software Limits (**SL**) set to 0, 0
Jog Velocity High (**JVH**) set to 10 rps
Jog Velocity Low (**JVL**) set to 1 rps
Jog Acceleration (**JA**) set to 99 rps^2
Limit Acceleration (**LA**) set to 900 rps^2
Go Home Velocity (**GHV**) set to 1 rps
Go Home Final (**GHF**) set to 0.1 rps
Go Home Acceleration (**GHA**) set to 99 rps^2
Velocity (**V**) set to 1 rps
Acceleration (**A**) set to 10 rps^2
Output Peak (**COP**) set to 100
SS flags set to 0
OS flags set to 0
Configurable Inputs (**IN**) set to trigger inputs (A)
Configurable Outputs (**OUT**) set to programmable outputs (A)
All Sequences (1 - 100) are erased.

The command out signal must be disabled (**OFF** or **STØ**) to execute the **RIFS** command.

SEE ALSO: **RFS**
EXAMPLES: None

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

RM

NAME: Rate Multiplier in Velocity Streaming Mode
SYNTAX: <A>**RM**h
DEFAULT: **RM**ØØØØØ
RANGE: h = ØØØØØ to FFFFF
VALID:
RESPONSE: None
TYPE: Motion
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: RM command followed by 5 hexadecimal digits represent a velocity

The velocity change is essentially instantaneous; there is no acceleration/deceleration ramp between velocities. A limit switch-closure will stop movement while in velocity profiling mode, but does not cause the JSI to exit velocity streaming mode. **RM** (profiling) mode is unidirectional. The direction will be the last activated direction either from an actual move or from a **D** or **H** command. Bidirectional moves using this mode can be made by returning to velocity zero, switching off **RM** mode, changing the direction, and re-enabling **RM** mode. This extra overhead should be acceptable given the need to attend to going to velocity zero when changing directions in real situations.

To generate the hexadecimal for any particular speed:

ddddd = (feedback resolution(steps/rv)*256*(servo sampling time)

Where the feedback resolution is from the **CFB** command and the servo sampling time is either 0.000512 **(CFB1)** or 0.001024 **(CFB2)** . Then convert ddddd to hex

SEE ALSO: **Q1, QØ**
EXAMPLE:

Command	Description
> **CFB1,4ØØØ**	Set to feedback resolution of 4000 and servo sampling time to 0.0000512
> **Q1**	Enter Velocity streaming mode
> **RMØØ418**	Accelerate to 2 rps
> **RMØØ83Ø**	Accelerate to 4 rps
> **T1**	Run at 4 rps for 1 sec
> **RMØØ418**	Decelerate to 2 rps
> **RMØØØØØ**	Decelerate to 0 rps
> **QØ**	Exit velocity streaming mode

Notes on Velocity Profiling Mode:

Situations requiring non-linear accelerations may use the **QØ**, **Q1** and **RM** commands. **Q1** is used to enter the velocity profiling mode, and **QØ** is used to exit. While in this mode the **RM** command is used to generate velocity values that are immediately implemented while the motor is moving. This means that the **RM** command must be sent to the JSI at the time the change in velocity is required. This creates a stair-step effect in velocity change. By implementing a large number of very small instantaneous velocity changes, a smooth, non-linear acceleration ramp can be achieved.

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

RS

NAME: Report Status of Sequence Execution
SYNTAX: a**RS**
DEFAULT:
RANGE: x = @, A, B, C, D
VALID:
RESPONSE: a**RS** *x[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: Responds in the form *x[cr] where X = "@" - "D".
Bit 0: Sequence started; Bit 1: Sequence Ended

Response Character	Sequence Started	Sequence Ended	Bad Loop
*@	NO	NO	NO
*A	YES	NO	NO
*B	NO	YES	NO
*C	YES	YES	NO
*D	N/A	N/A	YES

Whenever a sequence is started, the sequence start bit is set and the sequence end bit is cleared (this only occurs if the sequence is valid and is actually run). Whenever a sequence is ended, the start bit is cleared and the end bit is set. Any abrupt move termination (e.g., limit activation), or a **K** or **S** command clears both bits.

***D** is reported when there is an unbalanced number of loops and loop terminators inside a sequence. Starting a loop in one sequence and terminating it in another sequence is not allowed. Nested loops require complete closure before execution will begin.

Sequence started is true when: An **XR**, **XRP** or a power-up successfully starts a sequence.

It is false when: A STOP or a KILL command is received, or limits are hit.

Sequence Ended is true when: An **XT** is encountered, when a STOP or KILL is executed, or when End-of-Travel limit is encountered.

Sequence ended is false when: A sequence is successfully started.

SEE ALSO: **XR**, **XRP**

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

RSE

NAME: Report Servo Errors
SYNTAX: a**RSE**
DEFAULT: None
RANGE: See Below
VALID:
RESPONSE: See Below
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: You can find out what error condition exists in the JSI using this command. If you see the two digit LED flashing a code, you should start troubleshooting the unit using the **RSE** command. The error number and its description are reported. The possible error messages are:

16	Amplifier off
20	Following error exceeded
23	Enable plug not inserted
30	CRC error in EEPROM
41	CW Hardware Limit
42	CCW Hardware Limit
43	CW Software Limit
44	CCW Software Limit
60	Commanded Off
66	User Fault

POSSIBLE RESPONSES:

Command	Response
a**RSE**	*NO_ERRORS[cr]
a**RSE**	*COMMAND_OUT_DISABLED_BY_x[cr] x = 16,20, 23, 30, 60, 66
a**RSE**	*JSI_DISABLED_BY_x[cr] x = 41, 42, 43, 44

SEE ALSO: None

RV

NAME: Revision Level Report
SYNTAX: a**RV**
DEFAULT:
RANGE: n = 0 - 9, x = A - Z
VALID:
RESPONSE: a**RV** *92-nnnnnn-nn<xn>[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: The Revision (**RV**) command responds with the software part number and its revision level. The response is in the form shown below:

*92-nnnnnn-nn<xn>[cr]
part number revision level

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

SEE ALSO:
EXAMPLES:

Command	Response
1RV	Response = *92-ØØ783Ø-Ø1Y1

The product is identified by 92-ØØ783Ø. The revision level is identified by Ø1Y1.

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

S

NAME: Stop
SYNTAX: <a>**S**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE:
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: This command decelerates motion to a stop using the last defined Acceleration (**A**) command. This command normally clears any remaining commands in the command buffer, unless prevented from doing so by the Clear/Save The Command Buffer On Stop (**SSH1**) command. When the **SSH1** command is present, the **S** command stops only the current move. The JSI executes the next command in the buffer. The Stop (**S**) command does not stop motion in Velocity Streaming or Rate Multiplier (**RM**) mode. If you are in the **RM** mode, issue an Exit Velocity Profiling Mode (**QØ**) command to stop the motion.

If the **SSL** command is enabled, issuing a Continue (**C**) command after a **S** command will resume execution by completing the interrupted move and continuing sequence or buffer execution.

SEE ALSO: **K, SSH, QØ, A, SSL, C**
EXAMPLE:

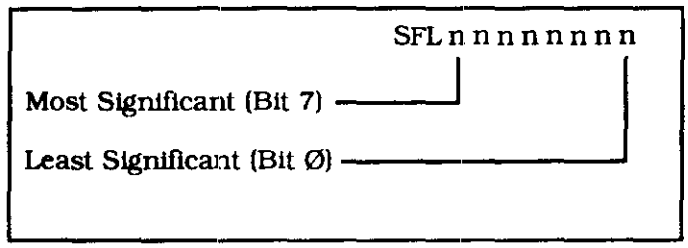
Command	Description
> **MC**	Sets move in continuous mode
> **A1**	Sets acceleration to 1 rps^2
> **V1Ø**	Sets velocity to 10 rps
> **G**	Executes the move (Go)
> **A5**	Sets Acceleration to 5 rps^2
> **S**	Stops (motor decelerates) to 0 rps at a rate of 5 rps^2

The **S** command is not buffered since it is an immediate command. As soon as the JSI receives the **S** command, it stops motion.

SFL

NAME: Set User Flag
SYNTAX: <a>**SFL**nnnnnnnn
DEFAULT: n = 0
RANGE: n = 0, 1 or X
VALID:
RESPONSE: <a>**SFL** *nnnn_nnnn[cr]
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional
DESCRIPTION: This command sets a condition of 8 different bits. You can define the state of bits 1 through 8 using this command to be either a 0, 1, or X. A 0 clears the corresponding bit, as 1 sets the corresponding bit, and an x retains the bit's present value. Not all the bits need to be defined during an **SFL** command; **SFL11** sets bits 6 and 7 while leaving the remaining bits unaltered.

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



Once you set bits 1 though 8, you can use the **IFFL** (If the flags match) command to do an IF_THEN operation.

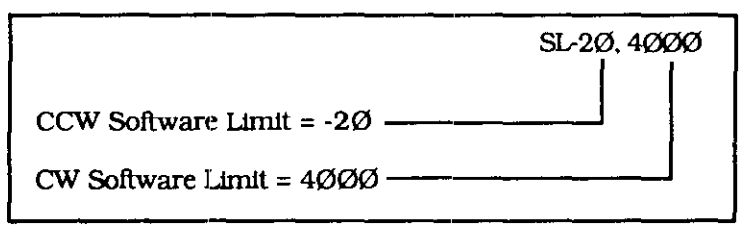
SEE ALSO: **IFFL, NIF**
EXAMPLE:

Command	Description
> **SFL1Ø1Ø**	Set bits 5 and 7, and clear bits 6 and 4
> **IFFL1Ø1Ø**	If user flag bits 5 and 7 are set, then issue the following commands
> **A1Ø**	Set acceleration to 10 rps^2
> **V5**	Set velocity to 5 rps
> **D25ØØØ**	Set distance to 25,000 steps
> **G**	Executes the move (Go)
> **NIF**	End of **IFFL**

If **IFFL** pattern matches the **SFL** pattern, the motor will move 25,000 steps.

SL

NAME: Software Limits
SYNTAX: <a>**SL**n, x
DEFAULT: n = 0, x = 0
RANGE: n, x = ±2,147,483,647
VALID:
RESPONSE: a**SL** *CCW_SOFTWARE_LIMIT_n[cr] *CW_SOFTWARE_LIMIT_x[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command defines the counterclockwise (CCW) and Clockwise (CW) software end of travel limits respectively. Once you define the CCW and CW software limits, the motor will not be able to exceed those positions, unless the software limits are disabled by the **SLD3** command. You may use the OUT command to configure 1 or more outputs to signal if a software end of travel limit has been encountered. When the motor reaches the software limit, it will come to an immediate stop, using the acceleration specified.



SEE ALSO: **OUT, SLD, LA**
EXAMPLE:

Command	Description
> **SL-1ØØØØ,26944**	Set CCW software limit to -1000 steps. Set CW software limit to 26944 steps.
> **SLDØ**	Enable both CW and CCW software limits
> **PZ**	Set absolute position to zero
> **A1Ø**	Set acceleration to 10 rps^2
> **V5**	Set velocity to 5 rps
> **D4ØØØØ**	Set distance to 40,000 steps
> **G**	Executes the move (Go)

Motor will start moving and starts decelerating when it reaches 26944 steps.

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

SLD

NAME: Software Limit Disable
SYNTAX: <a>**SLD**n
DEFAULT: n = 3
RANGE: n = 0 - 3
VALID:
RESPONSE: a**SLD** *Ø_CW_AND_CCW_SOFTWARE_TRAVEL_LIMITS_ENABLED[cr] or
*CCW_SOFTWARE_TRAVEL_LIMIT_ENABLED[cr] or
*2_CW_SOFTWARE_TRAVEL_LIMIT_ENABLED[cr] or
*3_NO_SOFTWARE_TRAVEL_LIMITS_ENABLED[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command enables or disables the software end of travel limits defined by the **SL** command

SLDØ Enables both CW and CCW Software limits. motion will not be allowed to go past the software limits
SLD1 Disables CW Software limit. Can travel past CW Software limit
SLD2 Disables CCW Software Limit. Can travel past CCW Software Limit.
SLD3 Disable both CW and CCW Software limits. Motor will ignore both CW and CCW software limit.

This command is very similar to the Hardware Limit Disable (**LD**) command, except it uses software limits.

You can use the **OUT** command to configure an output to indicate that the software limit has been reached. However, if you disable the limits, the output will not be activated even if the motion goes past the software limit.

SEE ALSO: **SL, OUT**
EXAMPLE:

Command	Description
> **SLØ,Ø**	Set CCW software limit to 0 and CW software limit to 0
> **PZ**	Set absolute position to zero
> **A1Ø**	Set acceleration to 10 rps^2
> **V5**	Set velocity to 5 rps
> **D25ØØØ**	Set distance to 25,000 steps
> **SLD3**	Disable both CW and CCW software limits
> **G**	Executes the move (Go)

If **SLDØ** was entered in place of **SLD3**, motion would not have occurred, because the motor is sitting on the software limit.

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

SN

NAME: Scan Time Delay
SYNTAX: <a>**SN**n
DEFAULT: n = 50
RANGE: n = 1 to 1000
VALID:
RESPONSE: a**SN** *SCAN_TIME=n_MILLISECONDS[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The Scan (**SN**) command allows you to define the debounce time (in milliseconds) for external sequence selection inputs. The debounce time is the amount of time that the sequence inputs must remain constant for a proper reading from a remote controller, such as a programmable logic controller (PLC). If you are using a PLC you should change the debounce time to be greater than the scan time for the PLC to assure that the correct data is present when the data is read.

SEE ALSO: None
EXAMPLES:

Command	Description
> **SN15Ø**	Sets scan time of sequence select inputs to 150 milliseconds.

SP

NAME: Set Position Absolute
SYNTAX: <a>**SP**n
DEFAULT: n = 0
RANGE: n = ± 1,999,999,999
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command allows you to set the absolute counter (value n). The **SP** command is useful for labeling a position value at a certain point. For example, you can set the zero reference point (home) to some location other than that of the physical hardware home. If you have a cut-off saw, for example, you may not be able to mount the home switch at the cut point. However, by mounting the home switch at a known distance away, and resetting the reference point with the **SP** command, you can make the system function as if the home switch were at the cut point.

Note that the units of the **SP** command are scaled by the **CMR** command. You can get a position report in the same units by using the **PR** command.

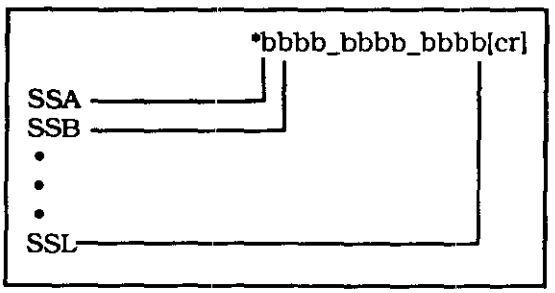
SEE ALSO: **MPI, MPA, PR, D, PZ, CMR**
EXAMPLES:

Command	Description
> **MN**	Sets mode to normal
> **A2**	Sets acceleration to 2 rps^2
> **GH2**	Instructs the motor to go home at 2 rps
> **PZ**	Sets the absolute position counter to zero units
> **1PR**	***+ØØØØØØØØØØ** Reads the absolute position counter
> **SP5ØØØ**	Sets absolute counter to 5,000 units
> **1PR**	***+ØØØØØØ5ØØØ** Reads the absolute position counter to verify the absolute position

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

SS

NAME: Function Setup Report
SYNTAX: a**SS**
DEFAULT: None
RANGE: b = 0, 1
VALID:
RESPONSE: a**SS** *bbbb_bbbb_bbbb[cr]
TYPE: Status
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: Reports the status of the **SS** commands (**SSA - SSL**) settings.



SEE ALSO: **SSA, SSG, SSH, SSI, SSJ, SSL**

SSA

NAME: RS-232C Echo Control
SYNTAX: <a>**SSA**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: a**SSA** *n[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command turns the RS-232C echo the re-transmission to the host device of characters received from the remote device by the JSI on and off.

SSAØ = Echo on
SSA1 = Echo off

In the Echo On (**SSAØ**) mode, characters that are received by the JSI are echoed automatically. In the Echo Off (**SSA1**) mode, characters are not echoed from the JSI. This command is useful only if your computer can handle echoes. In a daisy chain, you must have the echo turned on (**SSAØ**) to allow JSI's further down the chain to receive commands.

SEE ALSO: **SS**
EXAMPLES:

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

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

SSG

NAME: Clear/Save The Command Buffer on Limit
SYNTAX: <a>**SSG**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: In most cases, it is desirable that upon activating an end of travel limit input, all motion should cease until the problem causing the over-travel is rectified. This will be assured if all commands pending execution in the command buffer are cleared when hitting a limit. This is the case if **SSGØ** is specified. If **SSG1** is specified and a limit is activated, the current move is aborted, but the remaining commands in the buffer continue to be executed.

SEE ALSO: **LD, LA, SS**
EXAMPLES:

Command	Description
> **SSG1**	Save buffer on limit
> **A1Ø**	Set acceleration to 10 rps^2
> **V5**	Set velocity to 5 rps
> **D25ØØØ**	Set distance to 25,000 steps
> **G**	Executes the move (Go)
> **Ø11**	Activate programmable outputs 1 and 2

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

SSH

NAME: Clear/Save The Command Buffer on Stop
SYNTAX: <a>**SSH**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: a**SSH** *n[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION:
SSHØ = Clears command buffer
SSH1 = Saves command buffer

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

SEE ALSO: **LD, SS**
EXAMPLES:

Command	Description
> **SSHØ**	Clears buffer on stop
> **A1Ø**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D25ØØØ**	Sets distance to 25,000 steps
> **L5Ø**	Loops 50 times
> **G**	Executes the move (Go)
> **T.5**	Pauses for 500 msec
> **N**	Ends Loop
> **S**	Stops motion

When you issue the **S** command, the JSI will clear the buffer and stop the move.

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

SSI

NAME: Enable/Disable Interactive Mode
SYNTAX: <a>**SSI**b
DEFAULT: b = 0
RANGE: b= 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: When **SSIØ** is enabled and the device address is set to one, the JSI responds with a ">" when it understands a command and a "**?**" when it doesn't. Both responses are preceded with a line feed, carriage return sequence. If in Interactive Mode, the JSI will transmit a "*READY" and a ">" when energized or when a Reset (**Z**) command is executed.

If you try to define a loop command, you will not get back a ">" until you finish defining the loop.

SSI1 command disables the interactive mode. You will not receive ">" or "**?**" from the JSI.

SEE ALSO: **SS, SSA**

SSJ

NAME: Enable/Disable Continuous Scan Mode
SYNTAX: <a>**SSJ**b
DEFAULT: **b = 0**
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: **SSJ1** enabled: The JSI continuously scans the inputs designated as sequence select inputs by the **IN** command and executes the sequence represented by the BCD number presented on the inputs. If Interrupted Run Mode (**XQ**) is active, then all the sequence input lines must go inactive prior to scanning the next sequence. A stop command discontinues continuous sequence scanning.

SSJØ disabled: Does not scan the BCD numbers for sequence execution. In this mode, you could execute sequences using the RS-232C interface.

SEE ALSO: **XR, XD, XT, IN, INL, XQ, SS**

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

SSL

NAME: Resume Execution Enable
SYNTAX: <a>**SSL**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command will enable the controller to stop execution of commands, then resume execution using the Continue (**C**) command.

SSL1 Resume execution of the sequence or commands in the buffer when Continue (**C**) command is entered
SSLØ Disable resume feature

You can stop a program or move using the Stop (**S**) or Remote Stop input. If **SSL1** is enabled, the move will resume as soon as you enter the Continue (**C**) command.

SEE ALSO: **S, C**
EXAMPLE:

Command	Description
> **SSL1**	Enable resume function
> **XE1**	Erase sequence 1
> **XD1**	Define sequence 1
> **A1**	Set acceleration to 1 rps2
> **V1**	Set velocity to 1 rps
> **D2ØØØØØØ**	Set distance to 200,000 steps
> **G**	Execute the move (Go)
> **T2**	Wait 2 seconds
> **G**	Execute the move (Go)
> **XT**	End sequence definition
> **XR1**	Run sequence 1

While the motor is moving, enter a Stop (**S**) command. The motor will come to a stop. Now enter Continue (**C**) command, the motor would pick up where it stopped and complete the moves.

ST

NAME: Shutdown
SYNTAX: <a>**ST**n
DEFAULT: n = 0
RANGE: n = 0, 1
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The Shutdown (**ST1**) command disables the JSI command OUT (voltage or current) output. The system ignores move commands that you issue after the **ST1** command.

The **STØ** command enables the command OUT output. Once you enable the command OUT, you can execute moves.

This command allows you to manually position the load. The absolute position counter is set to zero when you enter an **STØ** command.

SEE ALSO: **OFF, ON**
EXAMPLES:

Command	Description
> **ST1**	Turns off the JSI command OUT output

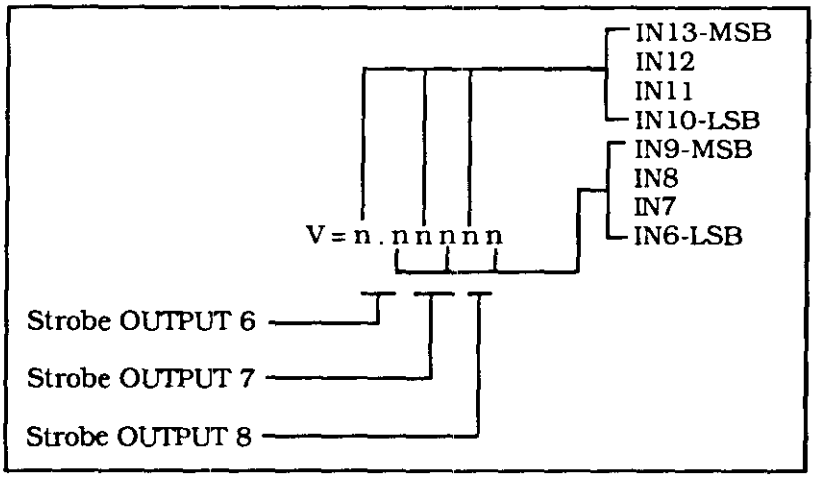
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

STR

NAME: Set Strobe Output Delay Time
SYNTAX: <a>**STR**n
DEFAULT: n = 1000 *msec*
RANGE: n = 5000
VALID:
RESPONSE: a**STR** *STROBE_TIME_n_MILLISECONDS[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command defines the amount of time each strobe output (defined by OUT) stays active. This delay and strobe outputs are used when loading parallel BCD data via remote inputs. The data transferred from the remote inputs are, Velocity (VRD), Distance (DRD), Loop Counts (LRD), and Sequence Numbers (XRD)

The STR command would typically be greater than a PLC scan time (to ensure that the data is present during a read), if used with a PLC or would set to a minimal debounce time if using thumbwheel switches. The strobe output will indicate that the JSI is ready for parallel input.

The data read by the inputs goes from the least significant digit to the most significant digit. The lowest input numbers are the least significant bit of the lower digit. For example, if we specify inputs 6 through 13 as the inputs, the inputs would automatically receive the following values during a **VRD** operation:



If the data VALID input is used, then when the strobe delay times out, the output strobe will remain active until the data valid input becomes active.

When you strobe the output, the first set of data read by inputs 6 through 13 will become the least significant digits. The next time the output is STROBED, the previously read data will be shifted 2 digits to the left, and the new value read will be come the least significant digits. This process continues until you turned on and off the number of strobe outputs you have configured (maximum of 5 strobe outputs available).

If you are using a thumbwheel to set the data, or using a PLC, be sure to have a diode connected backwards between the inputs and the logic ground to prevent current from flowing. This will prevent the JSI Controller from reading false information.

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

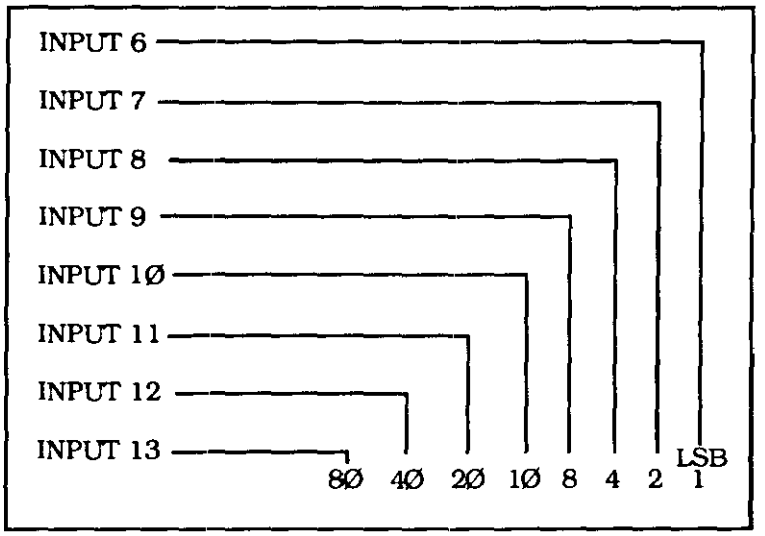
SEE ALSO: **VRD, LRD, DRD, XRD, IN, OUT, Application Design Section**

EXAMPLE:

Command	Description
OUT6J	Set output 6 as strobe output
OUT7J	Set output 7 as strobe output
OUT8J	Set output 8 as strobe output
IN6N	Set input 6 as least significant bit data input
IN7N	Set input 7 as 2nd least significant bit data input
IN8N	Set input 8 as 3rd least significant bit data input
IN9N	Set input 9 as 4th least significant bit data input
IN1ØN	Set input 10 as 5th least significant bit data input
IN11N	Set input 11 as 6th least significant bit data input
IN12N	Set input 12 as 7th least significant bit data input
IN13N	Set input 13 as 8th least significant bit data input
STR5ØØ	Set strobe output delay time to 500 mseconds
VRD	Read parallel inputs as velocity data

The example command would do the following as soon as the controller sees the **VRD** command.

Step 1 JSI would turn on Output 6 for 500 milliseconds
Step 2 Then it would read the inputs 6 thought 13 for the two digits
Step 3 JSI would then turn off Output 6 and turn on Output 7 for 500 milliseconds
Step 4 Then it would read two more digits
Step 5 JSI would then turn off Output 7 and turn on output 8 for 500 milliseconds
Step 6 Then the JSI would read two more digits.
Step 7 Output 8 would then turn off



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

SV

NAME: Save Tuning Parameters
SYNTAX: <a>**SV**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPES: Programming
ATTRIBUTES: Immediate, Device Address Optional, Not Saved
DESCRIPTION: The **SV** command saves only the tuning parameters in nonvolatile memory.

Those parameters requiring the Save (**SV**) command are:

Integral Gain (**CIG, BCIG**)
Integral Maximum (**CIM, BCIM**)
Proportional Gain (**CPG, BCPG**)
Proportional Maximum (**CPM, BCPM**)
Differential Gain (**CDG, BCDG**)
Differential Maximum (**CDM, BCDM**)
Integral Sum Limit (**CIL, BCIL**)

SEE ALSO: None
EXAMPLES:

Command	Description
> **CIG5Ø**	Set integral gain to 50% of the maximum value
> **SV**	Unit saved all the changes in nonvolatile memory

T

NAME: Time
SYNTAX: <a>**Tn**
DEFAULT: None
RANGE: n = 0.01 to 999.99 seconds
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The Time (**T**) command causes the JSI to wait the number of seconds that you specify before it executes the next command in the buffer. This command is useful whenever you need to dwell within a sequence.

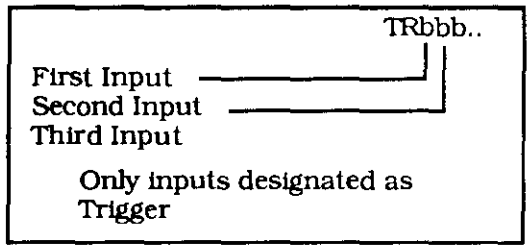
SEE ALSO: None
EXAMPLES:

Command	Description
> **MN**	Sets mode to normal
> **A5**	Sets acceleration to 5 rps^2
> **V5**	Sets velocity to 5 rps
> **D25ØØØ**	Sets distance to 25,000 steps
> **T1Ø**	Pauses for 10 seconds
> **G**	Executes the move (Go)
> **T5**	Pauses for 5 seconds after the move ends
> **G**	Executes the move (Go)

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

TR

NAME: Wait for Trigger
SYNTAX: <a>**TR**b<b>
DEFAULT: None
RANGE: b = 0, 1, or x
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Never Saved
DESCRIPTION: When the **TR** command is used, the JSI will get to this command and wait until the input (configured as triggers with the **IN** command) match the pattern,. It will not scrutinize those inputs not designated as triggers, before going on to the next command. The voltage level that represents an active state is defined with the **INL** command. The order of comparison as designated trigger inputs from Input 1 to Input 13 (as labeled on the unit).



Triggers are used to synchronize JSI operations with external events. They can be used to implement a handshaking function with other devices. Three characters are used for bbbbbb variables are listed below:

1 = Input active(ON)
0 = Input inactive (OFF)
x = Don't care

SEE ALSO: **D, TS, IN, INL**
EXAMPLES:

Command	Description
> **IN2A**	Configure input 2 as a trigger
> **IN3A**	Configure input 3 as a trigger
> **IN7A**	Configure input 7 as a trigger
> **TR1XØ**	Wait for input 2 to be active and input 7 to be inactive before going on to the next command. Input 3 is ignored.
> **AIØ**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D25ØØØ**	Sets distance to 25,000 steps
> **G**	Executes the move (Go)

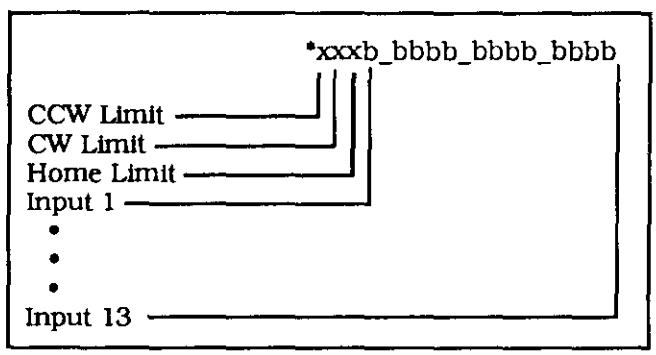
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

TS

NAME: Trigger Input Status
SYNTAX: a**TS**
DEFAULT: None
RANGE: b = 0, 1, x
VALID:
RESPONSE: a**TS** *xxxb_bbbb_bbbb_bbbb[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Not Saved
DESCRIPTION: This command reports the state of the trigger inputs. Response is in the form "xxxb_bbbb_bbbb_bbbb[cr]" where

b= 1 (Input Active)
b = 0 (Input Inactive)
b = x (Input not configured as a trigger)

TS command is useful for checking the status of the trigger inputs when it appears as though execution is being halted by a **TR** command. To make sure that your trigger pattern is met, you can check with **TS** command.



SEE ALSO: **TR, IN, INL**
EXAMPLES:

Command	Description
> **1TS**	*xxx1_xxxx_11xx_xxØØ[cr] Inputs 1, 6, 7, 12 and 13 are configured as triggers with 1, 6 and 7 active and 12 and 13 inactive.

U

NAME: Pause and Wait for Continue
SYNTAX: <a>**U**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: This command causes the JSI to complete the command in progress, then wait until it receives a Continue (**C**) to resume processing. Since the buffer is saved, the JSI continues to execute the program (at the point where it was interrupted). The JSI continues processing when it receives the **C** command. This command is typically used to stop a machine while it is unattended.

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

SEE ALSO: **PS, C**

EXAMPLES:

Command	Description
> **MN**	Sets move to Normal mode
> **A5**	Sets acceleration to 5 rps^2
> **V5**	Sets velocity to 5 rps
> **LØ**	Loops indefinitely
> **D256ØØ**	Sets distance to 25,600 steps
> **G**	Executes the move (G)
> **T1Ø**	Waits 10 seconds after the move
> **N**	Ends loop
> **U**	Halts execution until the JSI receives the Continue command.

This command string pauses at the point where the **U** command is entered. A Continue (**C**) command causes execution to resume at the point where it was paused. In this example, the loop stops at the end of a move, and resumes when the JSI receives the **C** command. There may be a 10-second delay before motion resumes after the **C** command is executed, depending on when the Pause and Wait for Continue (**U**) command is completed.

V

NAME: Velocity
SYNTAX: <a>**Vn**
DEFAULT: n = 1 rps
RANGE: n = 0 to 99.999 rps
VALID:
RESPONSE: a**V** *__Vn__[cr]
TYPE: Motion
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The velocity (**V**) command defines the maximum speed at which the motion will occur when given the Go (**G**) command. The value is stored in nonvolatile memory. Entering a velocity that is beyond the capabilities of your system will generate a following error. If the following error exceeds the **CPE** value, the JSI will fault and disable its control output.

SEE ALSO: **A, D, G**

EXAMPLES:

Command	Description
> **MC**	Sets move to continuous
> **A5**	Sets acceleration to 5 rps^2
> **V5**	Sets velocity to 5 rps
> **G**	Go (Begin motion)

In preset mode, Mode Normal (**MN**) the maximum velocity may also be limited when the resulting move profile is triangular. In Mode Continuous (**MC**), a Go (**G**) command is completed—the JSI moves on to the next command in the buffer—once the specified velocity is reached.

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

VRD

NAME: Read Velocity Value from Parallel Input/Output
SYNTAX: <a>**VRD**
DEFAULT: None
RANGE:
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command initiates the controller to read velocity values from the parallel inputs. You must set up the inputs as Data Inputs using the **IN** command. You must also set up outputs as strobe outputs. Typically, you would set up 8 inputs as DATA Inputs and 5 Outputs as strobe outputs using the **OUT** command. You must also set up the strobe output delay time (typically greater than a PLC Scan Time if using a PLC, or a minimal debounce time if using thumbwheel switches) so that the device you are getting data from will know when the JSI is ready for data. Sequence of events take place when we enter the **VRD** command:

Step 1 The lowest Output bit designated as strobe output will go on and stay on for strobe output delay time defined by **STR** command.

Step 1A If an input is designated as a data valid input, then the strobe output will stay active until the data valid input is active.

Step 2 The JSI will read the parallel inputs designated as DATA inputs. The lowest inputs will be the least significant bit. The JSI reads 8 bits (2 BCD digits) and stores them into two digits on the velocity register.

Step 3 Next Strobe output bit designated as strobe output will go on and stay on for delay time defined by **STR** command.

Step 3A If an input is designated as a data valid input, then the strobe output will stay active until the data valid input is active.

Step 4 The JSI will read the parallel inputs designated as DATA inputs are placed in the lowest digits on the velocity register. Previously entered values are shifted two positions to the left.

Step 5 Steps 3 through 4 are repeated as many times as the strobe outputs are available. You can use up to 5 different outputs as strobe outputs. The last two digits read are the least significant digit of the loop value.

The velocity range you can enter via parallel interface is from .00001 to 99.99999.

SEE ALSO: **STR, IN, OUT**
EXAMPLE: The following example shows how the JSI reads the velocity value of 35.89721. We will use inputs 1 through 8 for DATA inputs and outputs 1 through 4 for Strobe Outputs.

Step 1 Type the following:

Command	Description
OUT1J	Set Output 1 as Strobe Output
OUT2J	Set Output 2 as Strobe Output
OUT3J	Set Output 3 as Strobe Output
OUT4J	Set Output 4 as Strobe Output
IN1N	Set Input 1 as a Least Significant DATA input
IN2N	Set input 2 as a DATA input
IN3N	Set input 3 as a DATA input
IN4N	Set input 4 as a DATA input
IN5N	Set input 5 as a DATA input
IN6N	Set input 6 as a DATA input
IN7N	Set input 7 as a DATA input
IN8N	Set input 8 as a the Most significant DATA input
STR5ØØ	Set strobe output delay time to 500 msec.
A1Ø	Sets acceleration to 10 rps^2
V5	Set velocity to 5 rps
D2ØØØ	Set Distance to 2,000 steps

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

Step 2 As soon as you enter the Read Velocity via Parallel Input **(VRD)** command , the following should happen

Type the following:

Command	Description
> **VRD**	Start reading velocity from Parallel Input
> **A2Ø**	Set acceleration to 20 rps^2
> **D1ØØØØØ**	Set distance to 100,000
> **G**	Execute the move (Go)

Step 3 The JSI will turn on OUT1 for 500 msec.

Step 4 You must have your inputs 1 through 8 to the following settings.

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	OFF	OFF	OFF	OFF	OFF	ON	ON
Ø					3		

The current setting of the velocity is
ØØ.ØØØØ3 rps

Step 5 After the JSI reads the input, OUT1 turns off and OUT2 will automatically go on for 500 msec.

Step 6 You must have your inputs 1 through 8 set to the following settings.

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	ON	OFF	ON	ON	OFF	OFF	OFF
5					8		

Previously entered value of Ø3 shifts 2 locations to the left and 58 is placed on the 2 lowest position. The current setting of the velocity value is:
ØØ.ØØ358 rps

Step 7 OUT2 turns off and OUT3 automatically goes on for 500 msec.

Step 8 You must have your inputs 1 through 8 set to the following settings:

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
ON	OFF	OFF	ON	OFF	ON	ON	ON
9					7		

Previously entered values Ø358 shifts 2 locations to the left and 97 is placed on the two lowest positions. The current setting of the velocity value is:
ØØ.35897 rps

Step 9 OUT 3 turns off and OUT4 automatically goes on for 500 msec.

Step 11 You must have your inputs 1 through 8 set to the following settings.

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	OFF	ON	OFF	OFF	OFF	OFF	ON
2					1		

Previously entered values Ø35897 shifts two locations tooth left and 21 is placed on the two lowest positions. The final setting of the velocity value becomes
35.89721 rps

Step 12 Since the fifth strobe output is not defined, the JSI now goes to the next command in the buffer and executes the move at a velocity of 35.89721 rps.

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

W1

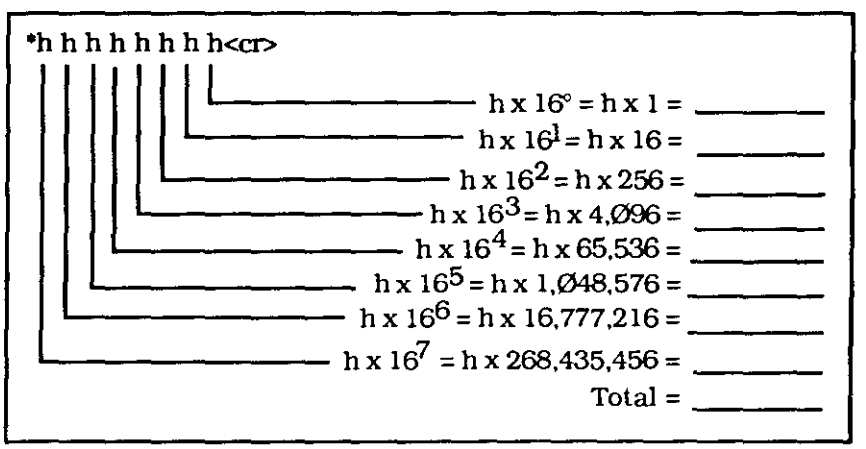
NAME: Signed Binary Position Report
SYNTAX: **aW1**
DEFAULT: None
RANGE: b = 8 bit binary number
VALID:
RESPONSE: **aW1** bbbb[space]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: Report back gives immediate binary representation of position relative to start of the current move. The format of the response is a four character response (bbbb) that is interpreted as a 32-bit binary number. The number must then be interpreted by the host device to give a numerical position in steps. The format is in 2's complement. Move in negative direction will report back negative number (bit 31 is set to 1).

The transmission is not preceded with an asterisk (*) to maximize response time. *Do not use the **W1** command when multiple JSI units are daisy-chained.*

SEE ALSO: **W2, W3, PR**

W2

NAME: Hexadecimal Position Report
SYNTAX: aW2
DEFAULT: None
RANGE: h = hexadecimal number Ø - 9 and A - F
VALID:
RESPONSE: aW2 *hhhhhhhh<cr>
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: The immediate hexadecimal character position report back (while motion is occurring) indicates position relative to start of current move. The format of the response is eight (8) hexadecimal characters. Your host computer needs to convert these characters into useable format. This command does not indicate the direction of the movement; you will only receive an unsigned number.



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

where h can be one of the following values:

Decimal Value	Hexadecimal Value (h)
Ø	Ø
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F

SEE ALSO: W1, W3, PR

EXAMPLES:

Command	Description
> **1W2 *ØØØØFAØ4**	A distance of FAØ4 (64ØØ4 steps) has occurred since the start of the last move.

W3

NAME: Signed (+ or -) Position Report

SYNTAX: a**W3**

DEFAULT: None

RANGE: H = Ø - 9 or A - F

VALID:

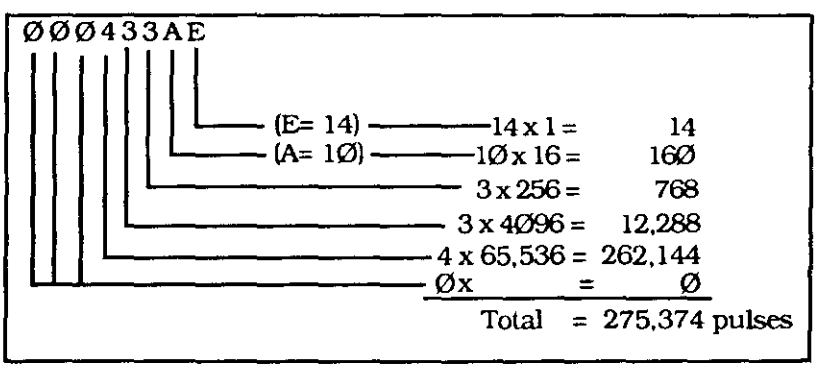
RESPONSE: a**W3** *hhhhhhhh[cr]

TYPE: Status

ATTRIBUTES: Immediate, Device Address Required, Never Saved

DESCRIPTION: This command will respond with an immediate hexadecimal character position report back in 2's complement format. The position response indicates the motor position relative to the current move. The format of the response is an eight hexadecimal number.

Assume the response was *ØØØ433AE. The decimal value would be:



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

If the first digit of the response is an "F"

Fhhhhhhh

then response represents a "two's complement" negative number Any response but Fhhhhhhh should be interpreted per the **W2** command.

To interpret a negative number (starting with Fhhhhhhh)

The binary approach:

1. Convert the hexadecimal response to binary form.
2. Complement the binary number
3. Add 1 to the binary result
4. Convert the binary result to decimal value with a minus sign placed ahead of the decimal value.

The computer approach: Subtract the hexadecimal number from 16^8 (2^{32}) (4,294,967,296).

The easy way:

1. Chop off all the leading "F"'s, and convert to decimal
2. Convert and subtract the next largest power of 16.

Example: the JSI responds to W3 as follows:

*FFFF9E58

1. Chop off the F's:	9E58 hex	=	40,536
2. Subtract from 16^4	10000 hex	=	65,536
	Results	=	-25,000

SEE ALSO: **W1, PR**

EXAMPLES:

Command	Description
> **1W3 *FFFFA19C**	In the current move, you are at -24163 steps from initiation of move.

XBS

NAME: Sequence Memory Available Report

SYNTAX: a**XBS**

DEFAULT: None

RANGE: n = 0 - 6000 x = 0 - 100

VALID:

RESPONSE: a**XBS** *n_OF_8ØØØ_BYTES_(X%)_SEQUENCE_MEMORY_ REMAINING[CR]

TYPE: Status

ATTRIBUTES: Immediate, Device Address Required

DESCRIPTION: This command reports the remaining amount of memory that can be used for sequence storage. The total space available for sequence storage is 8000 bytes (characters). This command is useful to find out how much more programming can be done on the JSI, after defining several programs.

This command reports both number of bytes available, and the percentage (%) of memory available. The status of the memory using the **B** and **BS** commands do not tell you the status of the sequence buffer.

SEE ALSO: **B, BS**

EXAMPLES:

Command	Description
1XBS	*4ØØØ_of_8ØØØ_Bytes_(5Ø%)_Sequence_Memory_ Remaining

There are 4000 bytes (50%) of the sequence buffer remaining for you to program.

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

XC

NAME: Sequence Checksum Report
SYNTAX: a**XC**
DEFAULT: None
RANGE: n = 0 - 255
VALID:
RESPONSE: a**XC** *n[cr]
TYPE: Status
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: This command reports the nonvolatile memory checksum. After the unit has been programmed, the response can be used for system error checking. The number reported does not indicate the number of bytes programmed. This response is designed to be used for comparison. As long as the sequences are not reprogrammed, the checksum response should always be the same.

SEE ALSO: **XD, XE**
EXAMPLES:

Command	Description
> **1XC**	Response = *ØØ149

XD

NAME: Sequence Definition
SYNTAX: <a>**XD**n
DEFAULT: None
RANGE: n = 1 to 100
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command begins sequence definition for a specific sequence. All the commands between the **XD** command and the Sequence Termination (**XT**) command will be defined as a sequence. If a sequence you are trying to define already exists, you must erase that sequence before defining it.

Immediate commands cannot be entered into a sequence.

SEE ALSO: **XT, XE, XR**
EXAMPLES:

Command	Description
> **XE1**	Erase sequence #1
> **XD1**	Define sequence #1
> **MN**	Sets mode normal
> **A1Ø**	Sets acceleration to 10 rps
> **V5**	Sets velocity to 5 rps
> **D1ØØØØ**	Sets distance to 10,000 steps
> **G**	Executes the move (Go)
> **XT**	End defining sequence #1
> **XR1**	Execute sequence #1

The commands in sequence 1 are defined and executed.

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

XDIR

NAME: Sequence Directory
SYNTAX: a**XDIR**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: See Description
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: This command reports the sequence number (sequence 1 to 100), and the amount of memory used by each sequence. The response is in the following format:

1XDIR *No_Sequences_Defines - If there are no sequences defined
1XDIR *Sequence_n_uses_x_bytes
*Sequence_n_uses_x_bytes

If sequence exists, it would list the sequence number and the number of bytes used by the sequence.

SEE ALSO: **XT, XE, XD, XBS**
EXAMPLE:

Command	Description
1XDIR	*SEQUENCE_5_USES_251_BYTES *SEQUENCE_39_USES_45_BYTES

Reports the number of bytes used by sequences 5 and 39. No other sequences are defined.

XE

NAME: Sequence Erase
SYNTAX: <a>**XE**n
DEFAULT: None
RANGE: n = 1-100
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command allows you to delete a sequence.
The sequence that you specify (n) will be deleted when you issue the command. Caution should be used when executing this command as the sequence is irretrievable

SEE ALSO: **XD, XT, XR**
EXAMPLES:

Command	Description
> **XE1**	Deletes Sequence 1

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

XFK

NAME: Set Fault or Kill Sequence
SYNTAX: <a>**XFK**n
DEFAULT: n = 0
RANGE: n = 0-99
VALID:
RESPONSE: a**XFK** *nn[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional;
DESCRIPTION: This command selects the sequence that will be executed is a fault or Kill condition occurs. A selection of Ø causes no sequences to be executed. This command is useful if you want to reset an output in case a fault or kill condition exists. You can also use this command to send a message through the RS-232C interface when a fault or kill condition exists. The fault conditions are listed below:

LED Display

Code	Condition
20	EXCESSIVE POSITION ERROR
23	DRIVE ENABLE NOT ACTIVE
30	EEPROM CHECKSUM ERROR
41	CW LIMIT SWITCH ENGAGED
42	CCW LIMIT SWITCH ENGAGED
43	CW SOFTWARE LIMIT ENGAGED
44	CCW SOFTWARE LIMIT ENGAGED
66	USER FAULT

The Kill condition exists is you issue a Kill (**K**) command over the RS-232C interface or parallel interface.

SEE ALSO; **XR, K**
EXAMPLE:

Command	Description
> **XFK5**	Execute sequence 5 when fault or kill condition exists
> **XE5**	Erase Sequence 5
> **XD5**	Define Sequence 5
"**FAULT_OR_KILL**	Send the message
> **XT**	End Sequence Definition

Whenever a Fault or Kill occurs, the Controller will display "FAULT_OR_KILL"

XG

NAME: Goto Sequence
SYNTAX: <a>**XG**n
DEFAULT: None
RANGE: n = 1-99
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command will jump to a designated sequence for execution. Once you jump to a sequence using the **XG** command, you can not return to the sequence from which the **XG** originated (unless another **XG** command is

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

executed). To jump to a sequence and return (**GOSUB** operation), you must use the **XR** command. There are no limitations on the number of **XG** commands as there is no nesting involved.

SEE ALSO; **XR**

EXAMPLES:

Command	Description
> **XE1**	Erase sequence #1
> **XD1**	Define sequence #1
> **A2**	Sets acceleration to 2 rps^2
> **V5**	Sets velocity to 5 rps
> **D1ØØØØ**	Sets distance to 10,000 steps
> **G**	Executes the move (Go)
> **XG5**	Go to sequence 5
> **XT**	End defining sequence #1
> **XE5**	Erase Sequence 5
> **XD5**	Define sequence #5
> **1PR**	Absolute Position Report
> **XT**	End sequence 5 definition
> **XR1**	Execute sequence #1

The motor will move 10000 steps, then the controller will jump to sequence 5 and indicate the absolute position.

XQ

NAME: Sequence Interrupted Run Mode
SYNTAX: <a>**XQ**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: a**XQ** *Ø__INTERRUPTED_RUN_MODE_OFF[cr] or
*1__INTERRUPTED_RUN_MODE_ON[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: Set interrupted run mode (b = 1)
Clear interrupted run mode (b = 0)
This command can only be used in conjunction with continuous mode (**SSJ**), and external sequence select lines. If **XQ1** is executed, the JSI will not accept a sequence select inputs, until all sequence select lines have been brought inactive. After all lines have simultaneously been brought inactive, the JSI will then read the sequence select lines and execute the sequence whose number appears there. This interrupted mode will continue until an **XQØ** command is executed. You may use **S** or **K** command to stop sequence execution.

SEE ALSO: **XD, XE, XT, XZ, XP, SSJ, IN**

EXAMPLES:

Command	Description
> **XE1ØØ**	Erase sequence #100
> **XD1ØØ**	Define sequence #100
> **LD3**	Disable CW & CCW limits
> **SSJ1**	Sets continuous mode
> **XQ1**	Sets interrupted mode
> **XT**	End Sequence #100

When the JSI powers up, sequence 100 will be executed. Interrupted run mode will be set. Sequence select input lines all need to go inactive before selecting any sequences

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

XR

NAME: Run a Sequence
SYNTAX; <a>**XR**n
DEFAULT: None
RANGE: n = 1 to 100
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command executes the commands contained within the designated sequence.

An **XR** command can be used within one sequence to start execution of another sequence. In this respect an **XR** acts like a GOSUB as sequence execution will return to the calling sequence. The maximum number of nested **XR** commands is 16.

SEE ALSO: **XE, XD, XT, XRP, XG**
EXAMPLES:

Command	Description
> **XE1**	Erase sequence #1
> **XD1**	Define sequence #1
> **A1Ø**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D1ØØØØ**	Sets distance to 10,000 steps
> **G**	Executes the move (Go)
> **XT**	End defining sequence #1
> **XR1**	Execute sequence #1

Sequence 1 is defined and executed using **XD1** and **XR1** commands respectively

XRD

NAME: Read Sequence via Parallel Input/Output
SYNTAX: <a>**XRD**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command initiates the controller to read the sequence value from the parallel inputs from within a sequence. You must set up the inputs as Data input using the **IN** command. You must also set up outputs as strobe outputs. Typically, you would set up 8 inputs as DATA Inputs and 5 Outputs as strobe outputs using the OUT command. However if only **XRD** is used, then only 1 strobe is required.

You must also set up the strobe output delay time (typically greater than a **PLC** Scan Time if using a PLC, or a minimal debounce time if using thumbwheel switches) so that the device you are getting data from will know when the JSI is ready form data. Sequence of events take place when we enter the **XRD** command

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

Step 1 The lowest Output bit designated as strobe output will go on and stay on for strobe output delay time defined by **STR** command.

Step 1A If an input is designated as a data valid input, then the strobe output will stay active until the data valid input is active.

Step 2 The JSI will read the parallel inputs designated as DATA inputs. The lowest inputs will be the least significant bit. The JSI reads 8 bits (2 BCD digits) and stores them into two digits on the sequence register.

Step 3 Next Strobe output bit designated as strobe output will go on and stay on for delay time defined by **STR** command.

Step 3A If an input is designated as a data valid input, then the strobe output will stay active until the data valid input is active.

Step 4 The JSI will read the parallel inputs designated as DATA inputs are placed in the lowest digits on the sequence register. Previously entered values are shifted two positions to the left.

Step 5 Steps 3 through 4 are repeated as many times as the strobe outputs are available. You can use up to 5 different outputs as strobe outputs. The last two digits read are the least significant digit of the sequence register.

In most cases you would use **SSJ** command to execute sequences from remote inputs. However, SSJ lets you select sequences only when the controller is not currently executing a sequence. You must use **XRD** command if you wish to execute a sequence remotely from within a sequence.

SEE ALSO: **STR, IN, OUT**

EXAMPLE: The following example shows how the JSI reads the sequence value of 21. We will use inputs 1 through 8 for DATA inputs and outputs 1 through 4 for Strobe Outputs.

Step 1 Type the following:

Command	Description
> **XE21**	Erase sequence #21
> **XD21**	Define sequence #21
A2Ø	Sets acceleration to 20 rps^2
V5	Sets acceleration to 5 rps
D25ØØØ	Sets distance to 25,000 steps
G	Executes the move (Go)
> **XT**	End defining sequence #1
OUT1J	Set Output 1 as Strobe Output
OUT2J	Set Output 2 as Strobe Output
OUT3J	Set Output 3 as Strobe Output
OUT4J	Set Output 4 as Strobe Output
IN1N	Set Input 1 as a Least Significant DATA input
IN2N	Set input 2 as a DATA input
IN3N	Set input 3 as a DATA input
IN4N	Set input 4 as a DATA input
IN5N	Set input 5 as a DATA input
IN6N	Set input 6 as a DATA input
IN7N	Set input 7 as a DATA input
IN8N	Set input 8 as a the Most significant DATA input
STR5ØØ	Set strobe output delay time to 500 msec.
A1Ø	Sets acceleration to 10 rps^2
V5	Set velocity to 5 rps

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

Step 2 As soon as you enter the Read Distance via Parallel Input (**XRD**) command, the following should happen

Type the following:

Command	Description
XRD	Start reading from Parallel input

Step 3 The JSI will turn on OUT1 for 500 msec.

Step 4 You must have your inputs 1 through 8 to the following settings.

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
	Ø				Ø		

The current setting of the sequence is
ØØØØØØØØØØ

Step 5 After the JSI reads the input, OUT1 turns off and OUT2 will automatically go on for 500 msec.

Step 6 You must have your inputs 1 through 8 set to the following settings.

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
	Ø				Ø		

Previously entered value of ØØØØ shifts 2 locations to the left and ØØ is placed on the 2 lowest positions. The current setting of the sequence value is still
ØØØØØØØØØØ

Step 7 OUT2 turns off and OUT3 automatically goes on for 500 msec.

Step 8 You must have your inputs 1 through 8 set to the following settings:

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
	Ø				Ø		

Previously entered values Ø shifts 2 locations to the left and another Ø is placed on the two lowest positions. The current setting of the sequence value is still:
ØØØØØØØØØØ

Step 9 OUT3 turns off and OUT4 automatically goes on for 500 msec.

Step 10 You must have your inputs 1 through 8 set to the following settings.

IN8	IN7	IN6	IN5	IN4	IN3	IN2	IN1
OFF	OFF	ON	OFF	OFF	OFF	OFF	ON
	2				1		

Previously entered values ØØØØØØ shifts two locations to the left and 21 is placed on the two lowest positions. The final setting of the sequence value becomes
21

Step 11 The JSI now checks to see if there is a fifth strobe output. Since we do not have a fifth output, the JSI controller executes sequence 21 by moving the motor 25,000 steps.

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

XRP

NAME: Sequence Run With Pause
SYNTAX: <a>**XRP**n
DEFAULT: None
RANGE: n = 1 to 100
VALID:
RESPONSE; None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command is identical to the Sequence Run (**XR**) command, except that it automatically generates a pause condition. You muse clear this condition with the Continue (**C**) command before the JSI executes the command buffer. The pause condition is asserted only if the sequence is valid. This allows you to execute a sequence without the delay of buffering that sequence. An **XRP** command can be used within one sequence to start execution of another sequence (in this respect an **XRP** acts like a **GOSUB**). The maximum number of nested **XRP** commands is 16.

SEE ALSO: **XR, XD, XT, XE, C**
EXAMPLES:

Command	Description
> **XE5**	Erases Sequence #5
> **XD5**	Defines Sequence #5
> **A1Ø**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D1ØØØØ**	Sets distance to 10,000 steps
> **G**	Executes the move (Go)
> **XT**	Ends defining Sequence #5
> **XRP5**	Runs Sequence #5 with a pause
> **C**	JSI executes Sequence #5

Upon issuing **XRP5**, Sequence #5 is entered into the command buffer, but is not executed. You must issue a Continue (**C**) command to execute Sequence #5.

XSD

NAME: Sequence Status Definition Report
SYNTAX: a**XSD**
DEFAULT: None
RANGE: n = 0, 1, 2
VALID:
RESPONSE: a**XSD** *n[cr]
TYPE: Status
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: This command reports back the status of previous sequence definition (**XD...XT**). The response is in the form *n[cr]. The valid values and descriptions of *n* are shown below:

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

The **XSD** command is useful for verifying that the last sequence definition attempt was successful.

SEE ALSO: **XD, XE, XT**
EXAMPLES:

Command	Description
> **1XSD**	*1 When you issued the XSD command to device #1, the response was 1. This means that you need to erase the existing sequence to define that specific sequence.

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

XSR

NAME: Sequence Status Run Report
SYNTAX:
DEFAULT: None
RANGE: n = 0, 1, 2
VALID:
RESPONSE: a**XSR** *n[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: This command allows you to check whether or not the last sequence you issued was started successfully. The valid values and descriptions for *n* are shown below.

0 = Last attempt to run the sequence was
2 = Invalid sequence number was requested

SEE ALSO: **XR, XRP**
EXAMPLES:

Command	Description
> **XR2**	Runs Sequence 2
> **1XSR**	*Ø[cr] Sequence started OK

XSS

NAME: Sequence Status Report
SYNTAX: a**XSS**n
DEFAULT: None
RANGE: n = 1-100, x = 0, 1, 2
VALID:
RESPONSE: a**XSS**n *x[cr]
TYPE: Status
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: This command reports whether the sequence is empty, has a bad checksum, or is OK.

The valid values and descriptions of x are shown below:

0 = Empty
1 = Bad Checksum
2 = O.K.

This command is useful to see if the particular sequence exists and if that portion of memory has been corrupted or not.

SEE ALSO: **XD, XT, XE**
EXAMPLES:

Command	Description
> **1XSS1**	*Ø[cr] Nothing programmed in sequence 1

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

XST

NAME: Single Step Mode
SYNTAX: <a>**XST**n
DEFAULT: n = 0
RANGE: n = 0, 1, x = ACTIVE, INACTIVE
VALID:
RESPONSE: a**XST** *n_STEP_MODE_X[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command sets the controller into a single step mode. This command can only be used with the Step (#) command. When you are running a sequence with the single step mode active, every time you issue a Step (#) command, the controller will execute one command in the sequence buffer.

XST1 Single Step Mode active
XSTØ Single Step Mode inactive

Since you need to send a # command over the RS-232C interface, this command cannot be run in stand alone mode. You must be executing the sequence in RS-232C mode. You must enter a delimiter after the Step (#) command to execute the command. If you are in the Trace (**XTR**) mode, the controller will display one command every time you enter the # command.

This command is useful for troubleshooting your program to see where you are in the program and what takes place with each command. You can use the Kill (**K**) command to abort the sequence execution.

SEE ALSO: **XR, 3, XTR**
EXAMPLES:

Command	Description
> **XE1**	Erase sequence #1
> **XD1**	Define sequence #1
> **A5**	Sets acceleration to 5 rps^2
> **V2**	Sets velocity to 2 rps
> **D1ØØØØ**	Sets distance to 10,000 steps
> **G**	Executes the move (Go)
> **XT**	End defining sequence #1
> **XST1**	Enable single step mode
> **X1TR1**	Enable trace mode
> **XR1**	Execute sequence 1
> #	Execute the first command
*SEQUENCE_ØØ1_COMMAND_A5	Displays the first command executed
> #	Execute the second command
*SEQUENCE_ØØ1_COMMAND_V2	Displays the second command executed
> #	Execute the third command
*SEQUENCE_ØØ1_COMMAND_D1ØØØØ	Displays the third command executed
> #	Execute the fourth command
*SEQUENCE_ØØ1_COMMAND_G	Displays the fourth command executed Motor should have moved 10,000 steps.
> #	Execute the fifth command
*SEQUENCE_ØØ1_COMMAND_XT	Displays the last command executed

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

XT

NAME: Sequence Termination
SYNTAX: <a>**XT**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The **XT** command is a sequence terminator. This command flags the end of the sequence currently being defined. Sequence definition is not complete until this command is issued.

SEE ALSO: **XD, XE, XR**
EXAMPLES:

Command	Description
> **XD1**	Define sequence #1
> **MN**	Sets move to mode normal
> **A1Ø**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D25ØØØ**	Sets distance to 25,000 steps
> **G**	Executes the move (Go)
> **XT**	End sequence definition

XTR

NAME: Set Trace Mode
SYNTAX: <a>**XTR**n
DEFAULT: n = 0
RANGE: n = 0, 1, x=ACTIVE, INACTIVE
VALID:
RESPONSE: a**XTR** *n_TRACE_MODE_x
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: This command transmits the command that is being executed from the JSI to the host via RS-232C interface. This command only works if you are running a sequence.

a**XTR1** Enables Trace Mode
a**XTRØ** Disables Trace Mode

Enabling trace mode transmits the commands and the sequence number being executed. If you have a Loop (**L**) command in a sequence, it will also display the loop count.

This command is useful if the user wishes to see where you are in the program as the program is being executed.

SEE ALSO: **XR, L**
EXAMPLES:

Command	Description
> **XE1**	Erase sequence #1
> **XD1**	Define sequence #1
> **A1Ø**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D25ØØØ**	Sets distance to 25,000 steps
> **L2**	Loop 2 times
> **G**	Executes the move (Go)
> **N**	End Loop
> **XT**	End defining sequence #1
> **X1TR1**	Enable trace mode
> **XR1**	Execute sequence 1

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

After turning on the trace mode, as you run the sequence (**XR1**), the controller will display the current command being executed. The trace mode output is shown below:

*SEQUENCE_ØØ1_COMMAND_A1Ø
*SEQUENCE_ØØ1_COMMAND_V5
*SEQUENCE_ØØ1_COMMAND_D25ØØØ
*SEQUENCE_ØØ1_COMMAND_L2
*SEQUENCE_ØØ1_COMMAND_G_LOOP_COUNT_ØØ1
*SEQUENCE_ØØ1_COMMAND_N_LOOP_COUNT_ØØ1
*SEQUENCE_ØØ1_COMMAND_G_LOOP_COUNT_ØØ2
*SEQUENCE_ØØ1_COMMAND_N_LOOP_COUNT_ØØ2
*SEQUENCE_ØØ1_COMMAND_XT

XU

NAME: Upload Sequence
SYNTAX: a**XU**n
DEFAULT: None
RANGE: n = 1 to 100
VALID:
RESPONSE: a**XU**n uploads the sequence stored
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: Upload sequence. The sequence *n* will be sent to the host, via RS-232C interface preceded with an asterisk (*) and terminated by an extra [cr]. All command delimiters in the sequence will be sent out as underscores.

If the sequence is empty, then *_[cr] is transmitted to the host.

SEE ALSO: **XD, XE, XT**
EXAMPLES:

Command	Description
> **1XU1**	Upload sequence #1 from unit #1

Y

NAME: Stop Loop
SYNTAX: <a>**Y**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: The Stop Loop (**Y**) command takes you out of a loop when the loop completes its current pass. This command does not halt processing of the commands in the loop until the JSI reaches the last command of the current loop. At that time, the JSI executes the command that follows the End Loop (**N**) command. You cannot restart the command loop unless you enter the entire command structure, including the Loop (**L**) and End Loop (**N**) commands.

SEE ALSO: **L, N, LRD**
EXAMPLES:

Command	Description
> **L**	Loops indefinitely
> **A1Ø**	Sets acceleration to 10 rps^2
> **V5**	Sets velocity to 5 rps
> **D25ØØØ**	Sets distance to 25,000 steps
> **T2**	Waits 2 seconds
> **G**	Executes the move (Go)
> **N**	Ends loop
> **Y**	Stops loop

The loop requires the motor to move 25,000 steps CW and then wait for 2 seconds. The loop terminates at the end of the loop cycle it is executing when it receives the **Y** command.

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

Z

NAME: Reset
SYNTAX: <a>**Z**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: The Reset (**Z**) command is equivalent to cycling AC power to the JSI. This command returns all internal settings to their power-up values. It clears the command buffer. Like the Kill (**K**) command, the **Z** command *immediately* terminates motion.

When you use the Reset command, the JSI is busy for 2,500 milliseconds and ignores all commands.

Any changes to the tuning parameters that you do not save before issuing this command will be lost.

This command sets all position counters to zero.

SEE ALSO: **S, K**
EXAMPLES: None

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

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

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

Factory Default Settings

RS-232C Communications

- Baud Rates: 9,600 (default setting), 4,800, 2,400, 1,200, 600, 300
- Data Bits: 8
- Parity: None
- Stop Bits: 1
- Echo: Off (Full Duplex)

JSI Device Address

The default address setting for the JSI controller is **01**.
Refer to Chapter 2, Getting Started for information on displaying and displaying the address setting with the push buttons.

Hardware Interfaces

Output command: ±10V analog voltage source

Feedback Device Number: 1, Incremental encoder interface enabled;
Incremental encoder base resolution = 4000 steps/revolution

Absolute encoder interface is enabled

Front-panel push buttons are enabled

Analog input interface is disabled

RS-232C is enabled

Inputs active low, and configured as triggers, with dedicated home, clockwise, and counter-clockwise limits.

Outputs active low, configured as programmable outputs.

Limits

Hardware limits enabled
Software limits disabled

Tuning Parameters

Proportional gain = 10% of 500
Integral gain = 0% of 1,000
Derivative gain = 0% of 100

Motion Parameters

Maximum following error = 4000 steps
Motion resolution = 4000 steps/revolution

Environmental Specifications

Parker Compumotor recommends that you operate and store your JSI servo controller under the following conditions.

- Operating Temperature: 32° to 104°F (0° to 40°C)
- Storage Temperature: -22° to 185°F (-30° to 85°C)
- Humidity: 0 to 95% non-condensing

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

I/O Specifications & Descriptions

Connectors

This section refers to the Inputs and outputs that are on the JSI controller's front panel. Refer to Figure 6-1 for the location of the inputs and outputs described in this section.

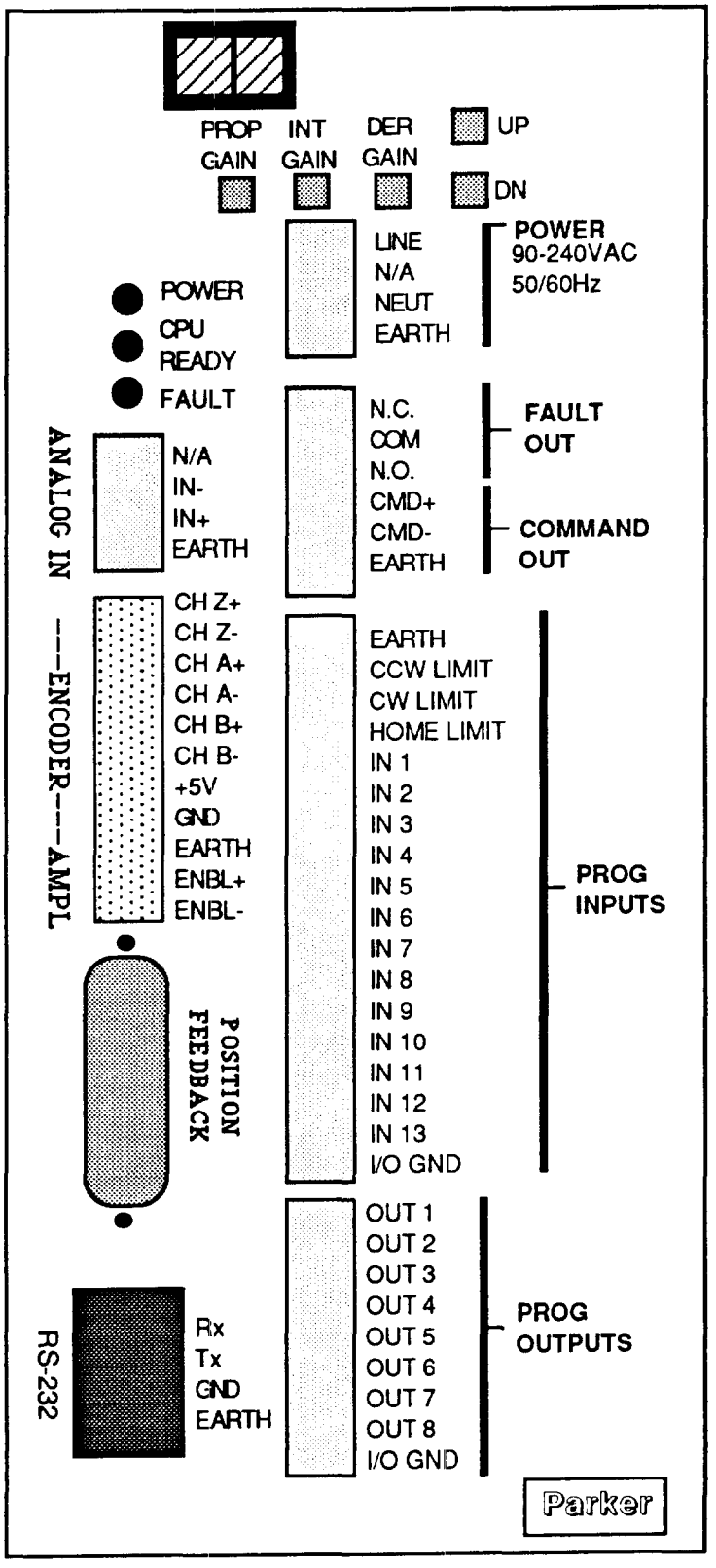


Figure 6-1. JSI Front-Panel Diagram

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

POWER

You can apply from 90 - 240VAC or 127 - 340VDC to this input. Figures 6-2 and 6-3 represent proper wiring for AC and DC power sources respectively.

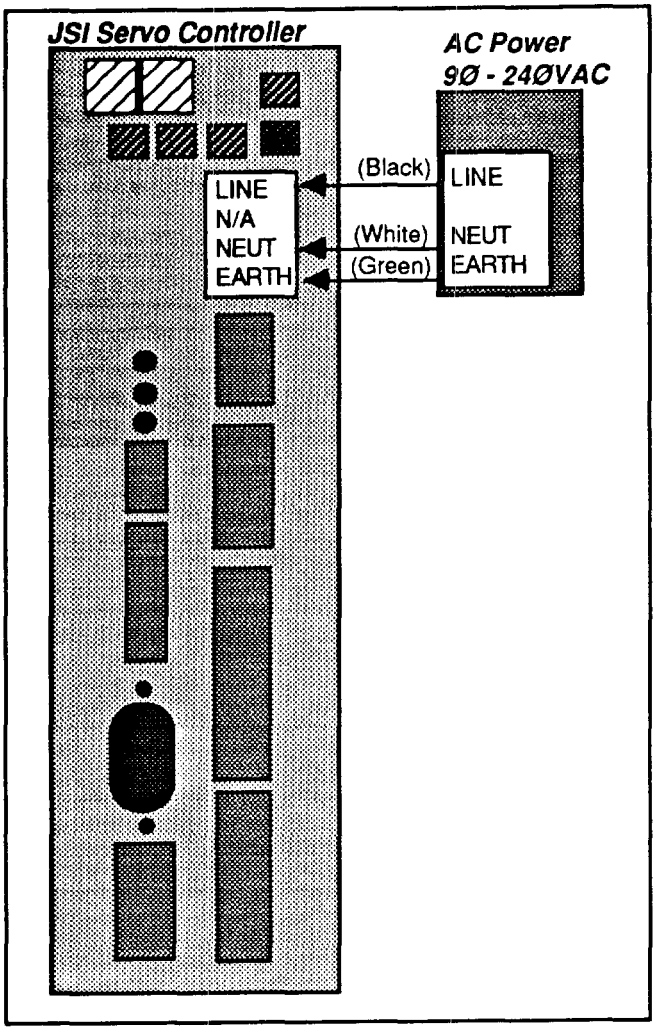


Figure 6-2. JSI Controller AC Power Wiring Diagram

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

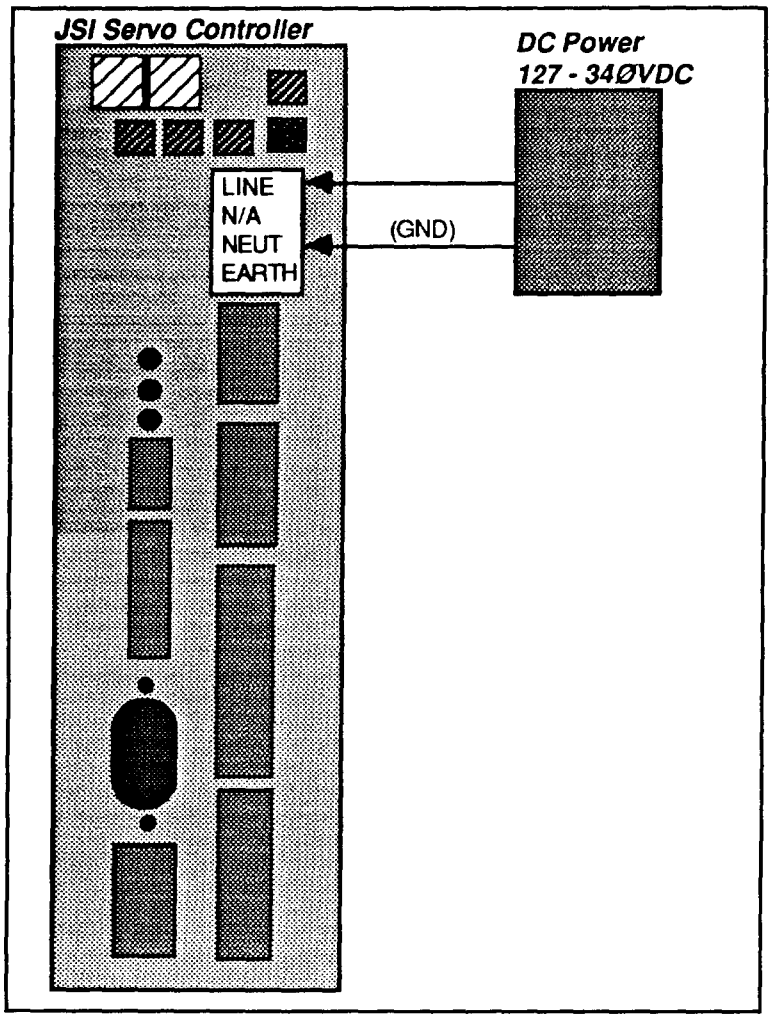


Figure 6-3. JSI Controller DC Power Wiring Diagram

FAULT OUT Normally-closed and normally-open fault outputs are relay outputs that have a contact rating of 2A at 120VAC.

- NC = Normally-closed relay output
- NO = Normally-open relay output.

COMMAND OUT This output is an analog current or voltage signal from the 12-bit DAC (digital-to-Analog converter). You can configure the output (**CMD+** and **CMD-**) as either a current output source or a voltage output source depending on your application's needs. The JSI is factory set as voltage output source controller. With the Configure Current Source—Low (**COCL**) or Configure Current Source—High (**COCH**) commands, you can convert the JSI into a current source controller.

- Configure Voltage Source (**COV**): ±10V Output
- Configure Current Source—Low (**COCL**): ± 60 mA output
- Configure Current Source—High (**COCH**): ± 200 mA output

This output drives either an analog servo drive or a hydraulic servo valve. Figure 6-4 provides an example of proper wiring for this output.

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

ANALOG IN

The JSI controller can receive analog voltage information from external devices through analog inputs (**IN+** and **IN-**). The analog input goes through a 10-bit analog-to-digital converter. You can use this analog input in two ways. You can use it either as a velocity input for a Velocity Following mode or as an error input for scaling the command velocity. Figure 6-4 provides an example of proper wiring for this input.

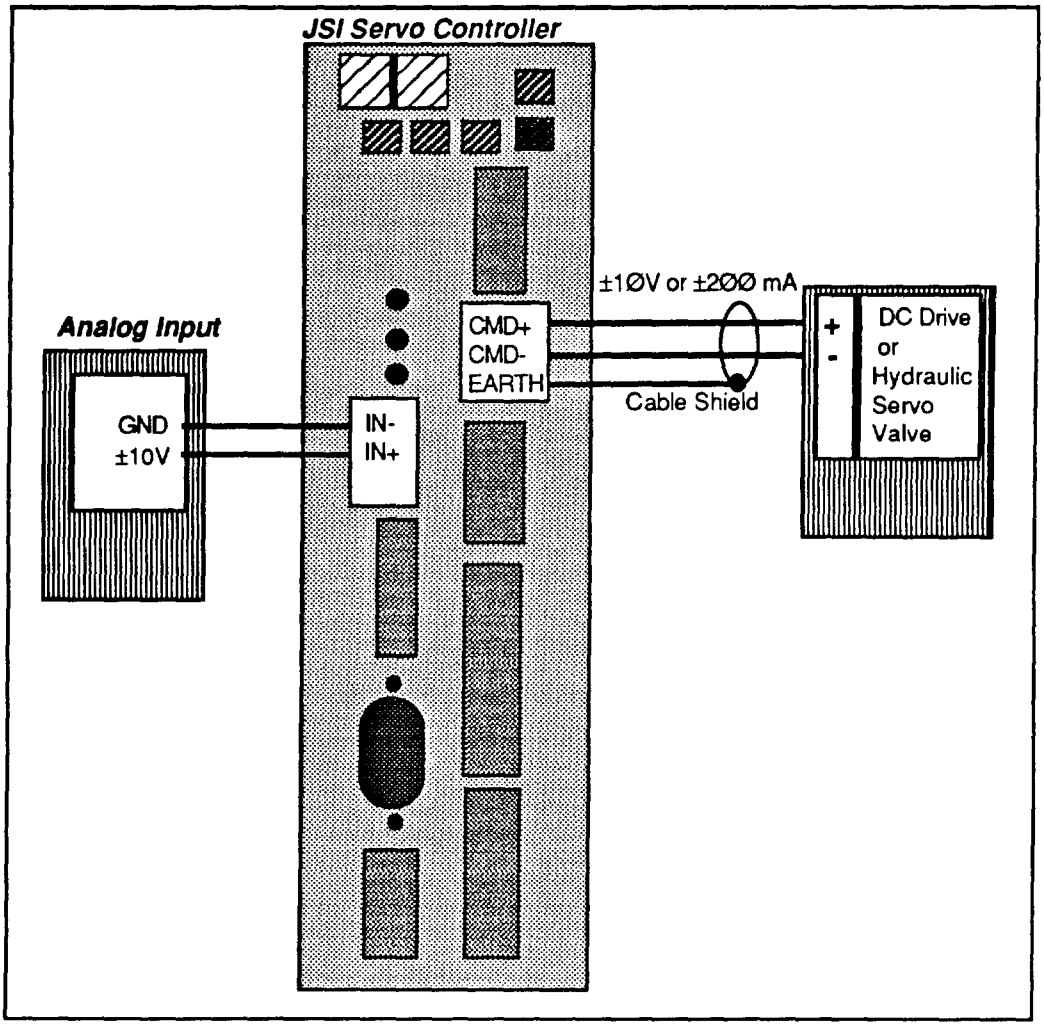


Figure 6-4. Analog Input and Command Output Connections

PROG INPUTS

There are 16 inputs, 1 I/O ground and one earth ground pin associated with **PROG INPUTS**. The first three inputs are the **CCW**, **CW** and **HOME LIMITs**. These inputs are electrically identical to inputs **IN1** through **IN13**. The primary difference between these inputs is that the **CCW**, **CW** and **HOME LIMIT** inputs are dedicated inputs, while **IN 1** through **IN 13** have programmable input functions. The inputs have an internal isolated 5V supply. Figure 6-5 represents a typical input circuit.

Input Voltage: 5 - 30 VDC. Current Limiting Resistor not needed

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

PROG OUTPUTS

There are eight programmable outputs available on JSI controller. These outputs are open collector outputs and you must supply between 5 to 30VDC and a pull up resistor with a minimum resistance of 100 ohms. Refer to Figure 6-5.

Output: 5 - 30V and can sink up to 300 mA.

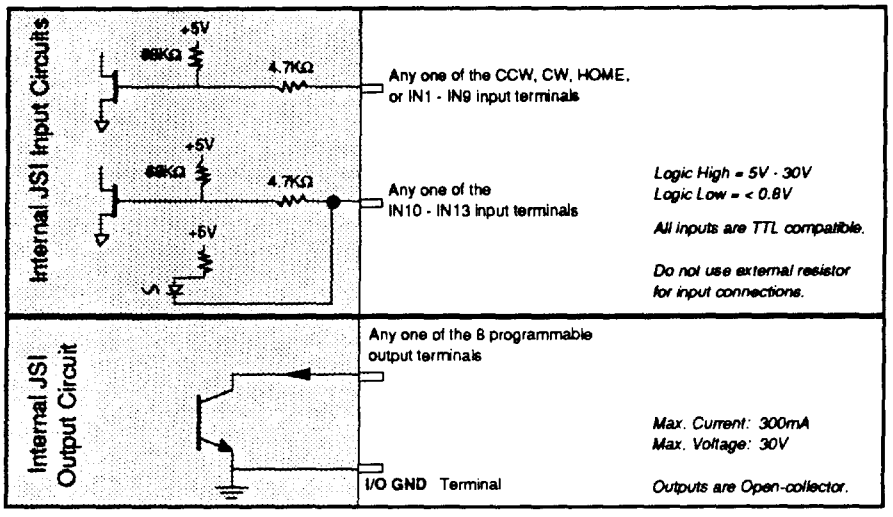


Figure 6-5. Typical JSI Output Circuit

ENCODER (Incremental encoder)

This input reads incoming pulses from an incremental encoder. The JSI can accept quadrature pulses in a differential or single-ended fashion. If you are using a single-ended encoder, do not connect the A-, B-, and Z- inputs. The JSI controller provides the 5VDC at 500 mA supply for the encoder.

Max Frequency Input: 60 kHz pre-quadrature.

AMPL

This input enables the JSI's command output (**CMD+** and **CMD-**). If you place a jumper (short) between **ENBL+** and **ENBL-** inputs, the JSI command output will be enabled. If you leave the **ENBL+** and **ENBL-** inputs open, the command output will be disabled. If you power the JSI without jumpering the enable input (**ENBL+** and **ENBL-**), the fault LED will illuminate and error code **23** will flash.

POSITION FEEDBACK (absolute encoder)

This input receives feedback from Compumotor's absolute encoders. The JSI supports the ABS8 (AR23 with -4 decoder), AR23 with -1 decoder, AL-C, AR-C, and Photo-Trak encoders. Refer to Chapter 3, *Installation*, for connection details.

RS-232

You can communicate with the JSI controller with your computer or terminal through RS-232C inputs. You must wire the **Tx** (Transmit), **Rx** (Receive), and **GND** (Logic Ground) inputs as shown on Figure 6-6. The controller accepts standard EIA RS-232C signals from +15 to -15VDC. You can change the baud rate, but the other parameters are fixed. You must configure your computer or terminal to the proper setting.

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

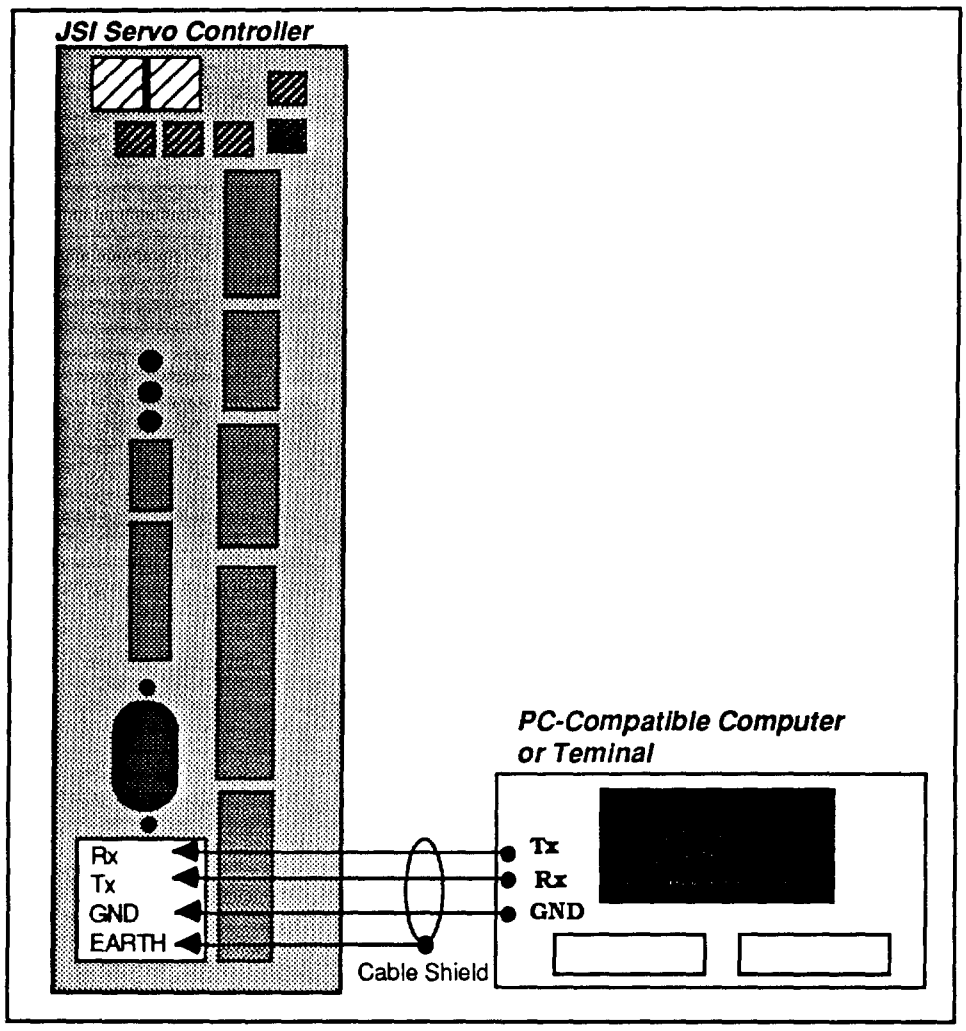


Figure 6-6. Wiring RS-232C Interface to a Computer or a Terminal.

LEDS

The two-digit LED flashes an error code if any type of error occurs. You can also light the LEDs with the LED command.

The Power LED will be green if you power up the JSI microprocessor properly.

The On-line LED will be green if the microprocessor and its peripherals are functioning properly. If this LED is off, a problem exists within the processor.

The Fault LED will be off under normal operations. If a hardware or software error exists, this LED will turn red. Refer to the *Diagnostic Codes* section of Chapter 7, *Maintenance & Troubleshooting*, for a complete listing of these codes.

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

Dimensional Drawings

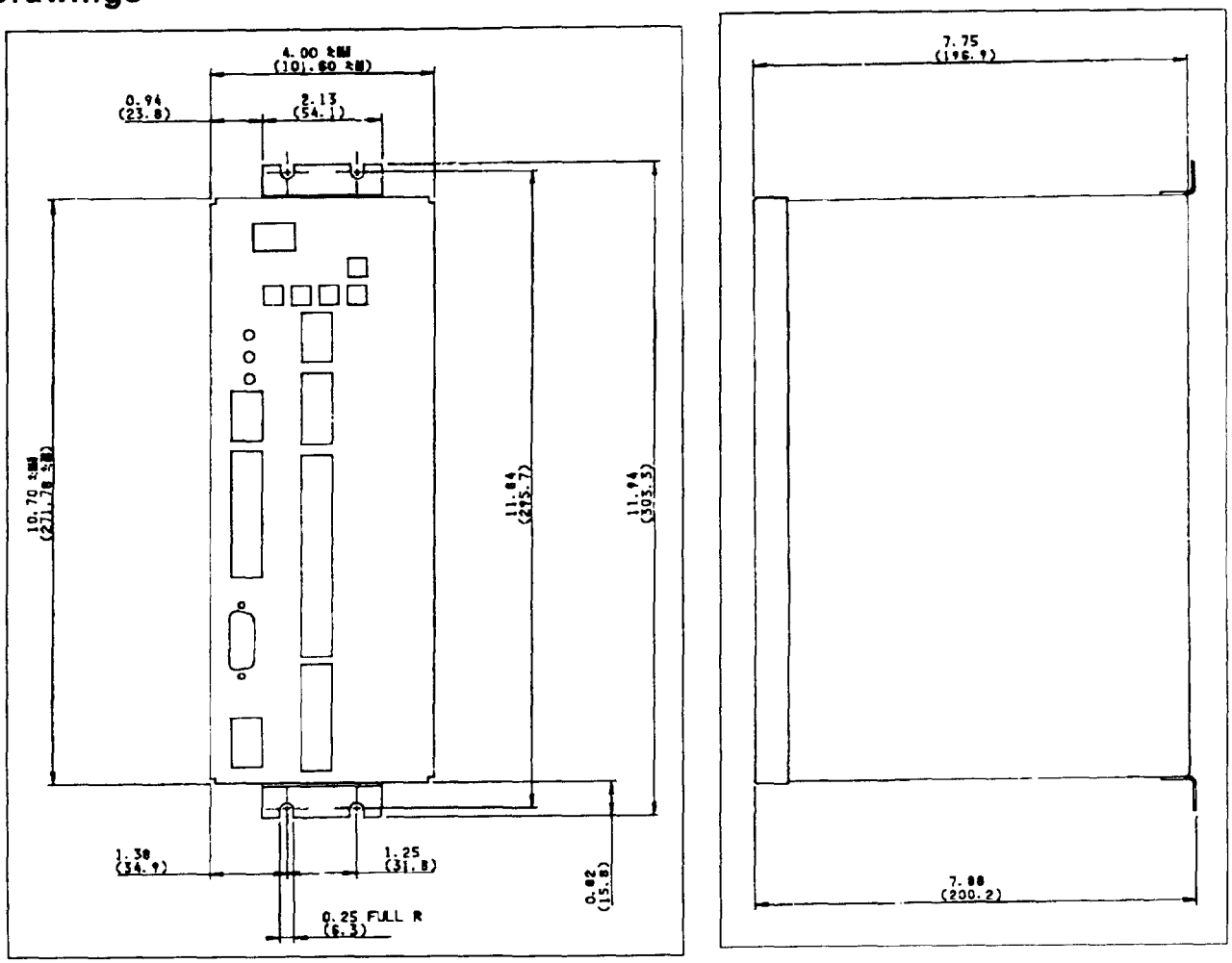


Figure 6-7. JSI Dimensional Diagrams

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

Chapter 7. Maintenance & 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 and software problems
- Use this chapter as a quick reference tool for a description of system error codes

Maintenance

Spare Parts Table

Table 7-1 is a list of recommended spare parts for the JSI system.

Description	Function	Part Number
4-Pin Connector	Power Connection	43-005560-01
6-Pin Connector	Command Output & Fault Output	43-006006-01
9-Pin Connector	Programmable Output	43-008755-01
11-Pin Connector	Inc. Encoder & Enable Input	43-008885-01
Battery	Maintains RAM memory	47-007709-01

Table 7-1. Spare Parts Table

Battery

The nonvolatile memory of the JSI Controller is a Battery-Backed RAM (Random Access Memory). The battery is used to keep the RAM powered at all times. The life of this battery is approximately 10 years. When the battery runs low, the JSI will be unable to hold a program in nonvolatile memory. If the battery back up is not functioning properly, error **30** will be displayed when you cycle power. To verify that the battery is OK, type **SV** (Save) then cycle power to the JSI controller. If the battery is OK, the error 30 (Check Sum error) should be gone. If the message is still flashing after the Save (**SV**) and power cycle, you should replace the battery (Compumotor Part Number 47-007709-01). Call your local distributor to order the battery.

Troubleshooting

This section provides methods to identify and resolve possible indexer-related hardware and software problems. You should also refer to the drive's user guide for troubleshooting procedures specific to the drive you are using.

Problem Isolation

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

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

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 have isolated the problem, take the necessary steps to resolve it. Refer to the problem solutions contained in this chapter. If your system's problem persists, contact Parker Compumotor's Applications Department at (800) 358-9070.

WARNING

Be sure to remove power before disconnecting JSI system components or changing wiring.

CW, CCW & Home Switches

If you are having problems using the Trigger (**TR**), Home (**GH**), CW, CCW and Sequence Select inputs, you must first check your wiring for proper installation. Use an Ohm meter for proper connection of the switches and inputs. If the hardware connection seems correct, you must use the Input Status (**IS**) command to report the hardware status of each input and see if the JSI recognizes the input change. You may do this by changing the input state manually and issue the **IS** command. If the status does not change, check the hardware settings.

Remote Sequencing (BCD Inputs)

If you are trying to run sequences from BCD interfaces, the first thing you must verify is the hardware interface. Use the Ohm meter to verify proper wiring. Then use the **IS** command to read the status of the inputs. Change the input setting and check the Input Status (**IS**) again to make sure that the JSI recognized the change in the sequence select input. Make sure that your BCD input is calling the proper sequences. Check Chapter 4, *Application Design*, for the Sequence Select Table. If you have a problem running a sequence from the remote input, try running the sequence using the **XR** command before attempting to run it using BCD input.

Diagnostic Codes Table

The two-digit LED flashes an error code if any type of error occurs. You can also light the LEDs with the **LED** command.

The following table lists the error codes and the explanation of each error code. The symptoms and solutions table shows how you can try to correct these error conditions.

CONDITION	LED DISPLAY CODE
BLANK	NO ERRORS
16	AMPLIFIER OFF
20	EXCESSIVE POSITION ERROR
23	DRIVE ENABLE NOT ACTIVE
30	EEPROM CHECKSUM ERROR
41	CW LIMIT SWITCH ENGAGED
42	CCW LIMIT SWITCH ENGAGED
43	CW SOFTWARE LIMIT ENGAGED
44	CCW SOFTWARE LIMIT ENGAGED
60	COMMANDED SHUTDOWN

Table 7-2. Diagnostic Codes Table

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

RS-232C Problems

If you are having problems communicating with the JSI, try the following procedure to troubleshoot the communications interface.

1. Power-up your computer or terminal *and then* power-up the JSI.
2. The serial port of your computer/terminal may require handshaking. If so, you must disable handshaking with your terminal emulator software package. You can also disable handshaking by connecting the computer's/terminal's RTS to CTS (usually pins 4 and 5) and DSR to DTR (usually pins 6 to 20).
3. Verify that the computer/terminal and 500 are configured to the same baud rate, number of data bits, number of stop bits, and parity.
4. Check to make sure you are using DC common or signal ground as your reference, *not* earth ground.
5. Cable lengths should not exceed 50 ft. unless you are using some form of line driver, optical coupler, or shield. As with any control signal, be sure to shield the cable to earth ground at one end only.
6. Press the space bar several times. The cursor should move one space each time you press the space bar. If your terminal displays garbled characters, check the terminal's protocol set-up. The baud rate setting probably does not match the JSI's setting.
7. If the terminal does not display garbled characters, press the space bar several times. If the cursor does not move, disconnect the **RS232** plug from the 500 and measure the cable's continuity . If you do not measure > -3V on Pin 1 (**Rx**) to Pin 3 (**GND**), your cable is wired incorrectly or your terminal is broken. If you measure > -3V on Pin 2 (**Tx**) to Pin 3 (**GND**), then switch the JSI's transmit (**Tx**) and receive (**Rx**) wires and try this test again.
8. Once you are able to make the cursor move, enter some characters. These characters should appear on the computer or terminal display. If each character appears twice, your host is set to half-duplex. It should be set to full-duplex.

Reducing Electrical Noise

Try to eliminate sources of possible noise interference. Potential noise sources include inductive devices such as solenoids, relays, motors, and motor starters operated by a hard contact.

For more information on identifying and suppressing electrical noise, refer to the Technical Data section of the *Compumotor Programmable Motion Control Catalog*.

Returning the System

If you must return your JSI system to affect repairs or upgrades, use the following steps:

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

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

2. Before you return the unit, have someone from your organization with a technical understanding of the JSI system and its application include answers to the following questions:
   - What is the extent of the failure/reason for return?
   - How long did it operate?
   - Did any other items fail at the same time?
   - What was happening when the unit failed (i.e., installing the unit, cycling power, starting other equipment, etc)?
   - How was the product configured (in detail)?
   - What, if any, cables were modified and how?
   - With what equipment is the unit interfaced?
   - What was the application?
   - What was the system environment (temperature, enclosure, spacing, unit orientation, contaminants, etc.)?
   - What upgrades, if any, are required (hardware, software, user guide)?

3. In the USA, call Parker Compumotor for a Return Material Authorization (RMA) number. Returned products cannot be accepted without an RMA number. The phone number for Parker Compumotor Applications Department is (800) 358-9070.

   Ship the unit to:
   Parker Hannifin Corporation
   Compumotor Division
   5500 Business Park Drive
   Rohnert Park, CA 94928
   Attn: RMA # xxxxxxx

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

   Ship the unit to:
   Parker Digiplan Ltd.,
   21, Balena Close,
   Poole,
   Dorset,
   England.
   BH17 7DX

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

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

Appendices

Command Listing

#	Single Step
A	Acceleration
B	Buffer Status Report
BCDB	Buffered Configure Dead Band
BCDG	Buffered Configure Differential Gain
BCDM	Buffered Configure Differential Maximum
BCIG	Buffered Configure Integral Gain
BCIL	Buffered Configure Maximum Integral Sum Limit
BCIM	Buffered Configure Integral Maximum
BCPE	Buffered Configure Position Error
BCPG	Buffered Configure Proportional Gain
BCPM	Buffered Configure Proportional Maximum
BS	Buffer Status Report
C	Continue
CDB	Configure Dead Band
CDG	Configure Differential Gain
CDM	Configure Differential Maximum
CFB	Configure Feedback Device
CIG	Configure Integral Gain
CIL	Configure Integral Maximum Sum Limit
CIM	Configure Integral Maximum
CMR	Configure Motion Resolution
COCH	Configure Output as Current Source (High)
COCL	Configure Output as Current Source (Low)
COP	Configure Output Peak
COV	Configure Output as Voltage Source
CMR	Configure Motion Resolution
CPB	Configure Push Button
CPE	Configure Position Error
CPG	Configure Proportional Gain
CPM	Configure Proportional Maximum
CR	Carriage Return
D	Distance
DAI	Display Analog Input
DCC	Display Current Control Configuration
DFS	Display Flags for Servo Status
DFX	Display Flags for Indexer Status
DMO	Display Maximum Overshoot
DOI	Display Output Instantaneous
DPA	Display Position Actual
DPE	Display Position Error
DPS	Display Setpoint Position
DRD	Read Distance via Parallel Input/Output
DTP	Display Tuning Parameters
DVA	Display Velocity Actual
DVS	Display Velocity Setpoint
E	Enable RS-232C Communication Interface
F	Disable RS-232C Communication Interface
G	Go
GH	Go Home
GHA	Go Home Acceleration
GHF	Go Home Final Velocity
GHV	Go Home Velocity
^H	Backspace
H	Set Direction
IFER	Compare Error Flag
IFFL	Compare User Flag
IFIN	Compare Input Status
IN	Set Input Functions
INL	Set Active Input Level
IO	Immediate Output
IS	Input Status Report
JA	Jog Acceleration
JVL	Jog Velocity (Low)
JVH	Jog Velocity (High)
K	Kill
L	Loop
LA	Limit Acceleration
LD	Limit Disable
LED	Illuminate Number on LED Display
LF	Line Feed
LRD	Read Loop Count via Parallel Input/Output
MC	Mode Continuous
MN	Mode Normal
MPA	Mode Position Absolute
MPI	Mode Position Incremental
N	End of Loop
NIF	End of IF Commands
O	Output
OFF	Off
ON	On
OS	Function Setup Report
OSB	Back up to Home Switch
OSC	Define Active State of Home Switch
OSD	Enable Encoder Z Channel Input
OSE	Jog Enable
OSG	Final Homing Direction
OSH	Reference Edge of Home Switch
OUT	Set Output Functions
OUTL	Set Active Output Level
OUTP	Output on Position
PR	Absolute Position Report
PS	Pause
PZ	Set Absolute Counter to Zero
"	Quote
Q0	Exit Velocity Profiling Mode
Q1	Enter Velocity Profiling Mode
R	Indexer Status Report
RA	Limit Switch Status Report
RB	Loop, Pause, Shutdown, Trigger Status Report
RFS	Return Drive Parameters to Factory Setting
RG	Go Home Status Report
RIFS	Return Indexer to Factory Settings
RM	Rate Multiplier in Velocity Streaming Mode
RS	Sequence Execution Report
RSE	Report Servo Errors
RV	Revision Level Report
S	Stop
SFL	Set User Flag
SL	Software Limits
SLD	Software Limit Disable
SN	Scan Delay Time
SP	Set Position Absolute
SS	Function Setup Report
SSA	RS-232C Echo Control

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

SSG Clear/Save the Command Buffer on Limit
SSH Clear/Save the Command Buffer on Stop
SSI Enable/Disable Interactive Mode
SSJ Enable/Disable Continuous Scan Mode
SSL Enable/Disable Resume Execution
ST Shutdown
STR Set Strobe Output Delay Time
SV Save Tuning Parameters

T Time
TR Wait for Trigger
TS Trigger Input Status

U Pause and Wait for Continue

V Velocity
VRD Read Velocity from Parallel Input/Output

W1 Signed Binary Position Report
W2 Hexadecimal Position Report
W3 Signed (+ or -) Position Report

XBS Report Sequence Memory Available
XC Sequence Checksum Report
XD Sequence Definition
XDIR Sequence Directory
XE Sequence Erase
XFK Set Fault or Kill Sequence
XG Goto Sequence
XQ Sequence Interrupted Run Mode
XR Run Sequence
XRD Read Sequence via Parallel Input/Output
XRP Sequence Run with Pause
XSD Sequence Status Definition Report
XSR Sequence Status Run Report
XSS Sequence Status Report
XST Single Step Mode
XT Sequence Termination
XTR Set Trace Mode
XU Upload Sequence

Y Stop Loop

Z Reset

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

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.

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, 600,1200, 2400, 4800, 9600, 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 through 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.

Bode Plot
A graph of system gain and phase versus input frequency which 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 broadly applied term relating to any system where the output is measured and compared to the input. The output is then 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.

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.)

DAC
Digital-to-analog converter

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.

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.

DCE
Data Communications Equipment transmits on pin three and receives on pin two.

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).

Delimiter
A character (space or carriage return) used to separate fields in a command.

Drive
This is the electronics portion of the motion control system that controls power to the motor.

DTE
Data Communications Equipment transmits on pin two and receives on pin three.

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 which translates mechanical motion into electronic signals used for monitoring position or velocity.

Following
The ability to make one axis perform motion based on the motion of a second, or *master*, axis.

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

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

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
A motion control term that is used to describe a device that produces 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.

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

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.

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 are used to communicate more than one piece of data at the same time.

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

PID Tuning
Refers to proportional, integral, and derivative gain tuning.

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 revolutions per second.

Quadrature
A type of incremental encoder output in which the two square wave outputs are offset by 90°.

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 in 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.

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 information on a single line in a time sequential format. The standard specifies the proper voltage and timing requirements so that different manufacturers' devices are compatible.

Sequence
A series of motion control commands. These commands are created, stored, and executed from the indexer's/controller's EEPROM memory.

Short-circuit
A defect in a winding that causes part of the normal electrical circuit to be bypassed. This frequently results in reducing the resistance or impedance to such an extent (near zero) as to cause overheating of the winding, and subsequent burnout.

Speed
Used to describe 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.

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.

Synchonism
A motor rotating at a speed correctly corresponding to the applied step pulse frequency is said to be in synchronism. Load torques in excess of the motor's capacity (rated torque) will cause a loss of synchronism. This condition is not damaging to a step motor.

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 0.8Vand a logical *one* or *high* is represented by a voltage from 2.5 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.

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

Index

ABSOLUTE MODE PRESET MOVES 37, 114
ABSOLUTE ENCODER 15, 37, 39, 72, 73, 172
  CONNECTIONS 20
  TEST 28
ABSOLUTE POSITION REPORT 123
ABSOLUTE ZERO POSITION 37, 125
ACCELERATION 65
ACCURACY 38
ADDRESS 13, 167
  DISPLAY & CHANGE 6
ANALOG INPUTS 16
ANALOG OUTPUT DISPLAY 86
ANALOG VOLTAGE INPUT 83
ASSUMPTIONS IV

BACKSPACE 96
BASIC ELECTRONICS IV
BASIC MOTION CONTROL IV
BASIC SERIAL COMMUNICATION IV
BATTERY 175
BAUD RATE DISPLAY & CHANGE 8
BCD INTERFACE PROBLEMS 176
BRANCHING 41
BUFFER STATUS 65, 70

COMMAND BUFFER 139
COMMAND LIST 179
COMMAND OUTPUTS 16
CONFIGURE CURRENT OUTPUT 26
CONFIGURE FEEDBACK DEVICE 15, 38
CONFIGURE INPUT MODE (IN) COMMAND
CONNECTIONS
  ANALOG INPUTS 16, 171
  COMMAND OUTPUTS 16, 170
  DAISY-CHAINING JSIS 53
  DRIVE & MOTOR 16
  ENABLE INPUTS 5, 172
  ENCODERS
    ABSOLUTE 20, 172
    INCREMENTAL 18, 172
  END-OF-TRAVEL & HOME 17
  FAULT OUTPUTS 23, 170
  PLC 57
  POWER 5, 169
    AC 5
    DC 6
  PROGRAMMABLE INPUTS 23, 171
  PROGRAMMABLE OUTPUTS 23, 172
  RS-232C 7, 172
  THUMBWHEELS 49
CONTINUOUS MODE 36, 41, 113
CONTINUOUS SCAN MODE 140
CONTROLLER STATUS 84
CONVENTIONS V
COUPLER 61
CURRENT OUTPUT 11
  CONFIGURE 26

DAISY-CHAIN 13, 53
DC MOTOR 26, 29
DEADBAND 66, 71
DEBOUNCE TIME 56
DELAYS 41
DELIMITER VI
DERIVATIVE GAIN 61
DIAGNOSTIC CODES 176
DIMENSIONAL DRAWINGS 174
DIRECTION 96
DISABLING THE LIMITS 15
DISPLAY
  POSITION (ACTUAL) 86
  POSITION ERROR 87
  SETPOINT POSITION 87
  TUNING PARAMETERS 90
  VELOCITY (ACTUAL) 91
  VELOCITY SETPOINT 91

EEPROM 46
ELECTRICAL NOISE 16, 177
ENABLE INPUTS 5, 13
ENCODER
  COMPATIBILITY 14
  CONNECTIONS (SEE *CONNECTIONS:ENCODER*)
  INPUT CIRCUIT 20
  MOUNTING 25
  RESOLUTION 15, 20, 39, 73
ENVIRONMENTAL CONSIDERATIONS 13
ERROR CODES 176
ERROR FLAG 97
ESTABLISH COMMUNICATIONS 7

FACTORY DEFAULT SETTINGS 129
FAULT OUTPUTS 23
FEEDBACK DEVICE 72
FEEDBACK RESOLUTION 20
FRICTION 61
FRONT PANEL DIAGRAM 4

GAIN 90
  DIFFERENTIAL 66
  INTEGRAL 67
  PROPORTIONAL 69
GO (G) COMMAND 53
GO HOME 14, 31, 39, 40, 93, 118, 121
GO HOME STATUS 40, 130
GOSUB 156
GOTO SEQUENCE 155
GRA (GOODS RETURNED AUTHORIZATION) NUMBER 178
GROUNDING 7, 16

HARDWARE INTERFACES 167
HEXADECIMAL POSITION REPORT 150
HOME ENABLE 40
HOME LIMIT SWITCH 93, 118
  PROBLEMS 176
  TEST 31
HOMING FUNCTION 31, 39
HOST COMPUTER OPERATION 52

IBM (OR IBM-COMPATIBLE) COMPUTER IV
IF 115
IMMEDIATE COMMANDS 46
IMMEDIATE OUTPUT (IO) 45, 104
INCREMENTAL ENCODER 14, 39, 72, 73
  CONNECTIONS 18
INCREMENTAL MODE (MPI) 36, 114
INDEX CHANNEL 39
INDEXER STATUS (R) 40
INPUTS vi
  FUNCTIONS 101
  LEVEL 104
  STATUS 100, 105, 176
  TEST 32
INSTALLATION PROCESS OVERVIEW iv
INTEGRAL GAIN 61
INTEGRAL LIMIT 62
INTERACTIVE MODE vi, 53, 140

JOG
  ACCELERATION 106
  ENABLE 120
  VELOCITY 106

KILL 108

LED DISPLAY 110, 176
LEDS 9, 173
LIMIT SWITCHES 135, 167
  ACCELERATION 109
  CONNECTIONS 17
  DISABLE 15, 109
  PROBLEMS 176
  STATUS 128
  TEST 30
LINE FEED 110
LOAD INERTIA 61
LOOPS 44, 108, 115, 140, 164

MODE CONTINUOUS (MC) 36
MODE POSITION ABSOLUTE (MPA) 36, 37
MODE POSITION INCREMENTAL (MPI) 36
MODE POSITION INCREMENTAL (MPI) 37
MOTION
  MOTION CONTROL CONCEPTS 35
  PARAMETERS 167
  PROFILES
    TRAPEZOIDAL 35
    TRIANGULAR 35
MOTION RESOLUTION 77
MOTOR INERTIA 61
MOUNTING 25
  ENCODER 25
  MINIMUM DEPTH 25
  MINIMUM WIDTH 25
MOVE COMPLETION SIGNAL 45
MOVE PROFILES 35
MOVE TIMES
  CALCULATED VS ACTUAL 35
MULTI-AXIS CONTROL
  DAISY-CHAINING 53
  INTERFACE PROGRAM 54

OUTPUTS vi
  CONFIGURATION 75, 116, 121
  CIRCUIT 172
  LEVEL 122
  OUTPUT (O) COMMAND 45
OVERSHOOT 62, 85

PANEL LAYOUT 25
PAUSE 71, 124, 146
PERFORMANCE 38
PID TUNING 38, 39, 58, 73
PLC 23, 137
  CONNECTIONS 57
  OPERATION 56
POSITION
  ACCURACY 38
  DISPLAY 86
  ERRORS 61, 69
  REPORT
    ABSOLUTE 123
    HEXADECIMAL 150
    PR COMMAND 37
    PZ COMMAND 31, 37
    SIGNED 151
    SIGNED BINARY 150
POWER CONNECTIONS 5, 22, 169
POWER-UP SEQUENCE 48, 50, 56
PRESET MOVE 36
PROBLEM ISOLATION 175
PROFILES
  TRAPEZOIDAL 35
  TRIANGULAR 35
PROGRAMMABLE INPUTS 23, 32, 41
PROGRAMMABLE OUTPUTS 23, 45
  TEST 33
PROPORTIONAL GAIN 61
PUSHBUTTONS
  CONFIGURATION 77
  FEATURE SUMMARY 60
  FUNCTIONS 60
  TUNING 59

READ DISTANCE VIA PARALLEL I/O 88
READ LOOP COUNT VIA PARALLEL I/O 111
READ SEQUENCE VIA PARALLEL I/O 157
READ VELOCITY VIA PARALLEL I/O 148
RECEIVE DATA 7
RELATED PUBLICATIONS VII
REMOTE SEQUENCING 176
REPEATABILITY 38
RESET 140, 165
RESOLUTION 14, 77
RESPONSE TIME 61

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

RETURN DRIVE PARAMETERS TO FACTORY SETTING 129
RETURN JSI TO FACTORY SETTINGS 130
RETURN MATERIAL AUTHORIZATION (RMA) NUMBER 178
RETURN PROCEDURE 177
RS-232C 7, 8, 53, 56, 164, 167, 172
  DISABLE 92
  ECHO CONTROL 138
  ENABLE 91
  TROUBLESHOOTING 177
  VERIFICATION 9

SEQUENCE
  CHECKSUM REPORT 153
  COMMANDS 46
  DEFINITION 153
  DIRECTORY 154
  ERASE 154
  FAULT 155
  INPUT LINES 56
  INTERRUPTED RUN MODE 156
  KILL 155
  MEMORY AVAILABLE 152
  READ VIA PARALLEL I/O 157
  RUN 157
  RUN WITH PAUSE 160
  SCANNING 48, 56, 137, 140
    SCAN (SN) COMMAND 56
    SCAN TIME DELAY 137
  SELECTION 47
  STATUS REPORT 132
  STATUS DEFINITION REPORT 160
  STATUS RUN REPORT 161
  SEQUENCE SELECT INPUTS 56
  TERMINATION 163
  UPLOAD 164
SERVO ERRORS REPORT 133
SERVO FEEDBACK 20
SET DIRECTION (H) COMMAND 36
SETUP REPORT 118, 138
SHAFT AND COUPLING VIBRATION 61
SHIP KIT 3
SHUTDOWN 141
SIGNED BINARY POSITION REPORT 150
SINGLE STEP MODE 64, 162
SOFTWARE LIMIT DISABLE 136
SOFTWARE LIMITS 135
SPARE PARTS 175
STAND-ALONE OPERATION 47
STATUS
  BUFFER 65, 70
  CONTROLLER (JSI) 84, 127
  GO HOME 40, 130
  INPUTS 100, 105, 106, 176
  LIMITS 128
  SEQUENCE 132
STOP (S) COMMAND 36
STROBE OUTPUT DELAY 142

TEST
  ABSOLUTE ENCODER 28
  CURRENT OUTPUT 11
  END-OF-TRAVEL LIMIT SWITCHES 30
  HOME LIMIT SWITCH 31
  INCREMENTAL ENCODER 26
  PROGRAMMABLE INPUTS 32
  PROGRAMMABLE OUTPUTS 33
  RS-232C LINK 9
  VOLTAGE OUTPUT LEVELS 10
THUMBWHEELS 49
TIME DELAY (T) 41, 144
TRACE MODE 163
TRANSMIT DATA 7
TRIGGER INPUTS 36, 145
  STATUS 146
  WAIT FOR TRIGGER 145
TRIGGER PAUSE (TR) COMMAND 41
TROUBLESHOOTING 175
TUNING 38, 58
  PARAMETERS 90, 144, 167

UPLOAD SEQUENCE 164
USER FLAG 98, 134
USING REMOTE JOGS AND SEQUENCE 48

VELOCITY
  VELOCITY (V) COMMAND 36
  VELOCITY PROFILING MODE 126, 131
  VELOCITY SCALING MODE 80
  VELOCITY STREAMING MODE 131
VOLTAGE LEVELS vi
VOLTAGE OUTPUT 10
VOLTMETER 10

WAIT FOR TRIGGER 145

Z CHANNEL 14, 39, 93, 119
ZERO POSITION 37, 125, 137

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

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP