PrecisionAire™

PrecisionAire Software, Hardware and Controller

USER'S MANUAL

TOL-O-MATIC, INC

Excellence in Motion®

3604-4104_03.0

DMS 01/04

Table of Contents

Technical Reference

Communication Protocol (continued)

Technical Reference *Communication Protocol (continued)*

List of figures:

List of figures (continued):

1.1 PrecisionAire Overview

PrecisionAire was developed to be a low cost position control system relative to electric motion systems, without the setup and control challenges of traditional proportional valve pneumatic servo systems. PrecisionAire is intended for applications not requiring positional repeatability better than +/-0.010 inches up to a 120" stroke. Longer strokes up to the maximum stroke length are capable of repeatability better than +/-0.025 inches. Motion profiles are user defined by programming the move distance, the maximum speed, acceleration time, and deceleration time. System variables, including supply pressure, valve Cv, the load, and how well the system is tuned, do not effect the final repeatability of the system. However, they will influence how well the actual motion follows the theoretical profile.

PrecisionAire significantly reduces the concerns of changing loads, system friction, vertical operation, setup, tuning, instabilities, and long stroke lengths typically experienced with pneumatic servo systems. The Tol-O-Matic patent-pending approach used in PrecisionAire accomplishes this using the muscle of air to provide thrust and an electric current-controlled magnetic particle brake to provide proportional braking for position control. This approach significantly reduces the effects of directly attempting to control a compressible fluid through proportional valves, or trying to predict when to activate an on/off brake to achieve a desired position.

The PrecisionAire system requires pressure regulators. To minimize overshooting past the desired position it is recommended to operate at an air pressure no greater then 10 psi above the pressure required to achieve the desired speed or force. Limiting the air pressure will also reduce the brake regulation at constant speeds, which can reduce system ‘hesitations’ caused by the brake trying to overcompensate for higher air pressures. PrecisionAire software has an easy to use Data Acquisition that can insure the proper air pressure is selected based upon the applications loads and speeds. In a vertical application a dual pressure system must be used. Two separate external pressure regulators (or valves with a sub-base mounted pressure regulator) are required to provide a dual pressure system to the actuator. The brake end of the PrecisionAire actuator must be mounted at the top.

The thrust limitations experienced in a traditional end-of-stroke open loop pneumatic system are also present in PrecisionAire. These limitations are due to constraints including differential air pressure, bore size, system Cv, and friction in the system. These constraints will limit the maximum speed, thrust to overcome an external force, and acceleration of a load.

Tol-O-Matic provides Tol-O-Motion Sizing and Selection software to provide guidance regarding these constraints as they apply to specific applications.

Tol-O-Motion Sizing and Selection software also considers duty cycle limitations due to heat dissipation requirements of the brake. Heat generation takes place not only during deceleration of a load, but also during regulation at a constant speed. It is desirable to operate at a minimum supply pressure necessary for the application. This will minimize the heat generated by the brake, therefore, maximizing the application's duty cycle. There may also be a minimum speed that can be achieved with a given air pressure, determined by the duty cycle desired. Data Acquisition within the PrecisionAire software can help determine the minimum supply pressure required for your loads and speeds.

The PrecisionAire system is packaged to be a completely independent system or part of a higher level system through communications with inputs and outputs. A system consists of an actuator with integral brake and encoder, and a programmable controller/drive. As shown below, the PrecisionAire system utilizes a completely enclosed, internal timing belt to provide the linkage between the cylinder carrier, the brake, and an optical encoder. Recommended valves are discussed in Chapter 5, however, they are not included as part of the PrecisionAire system.

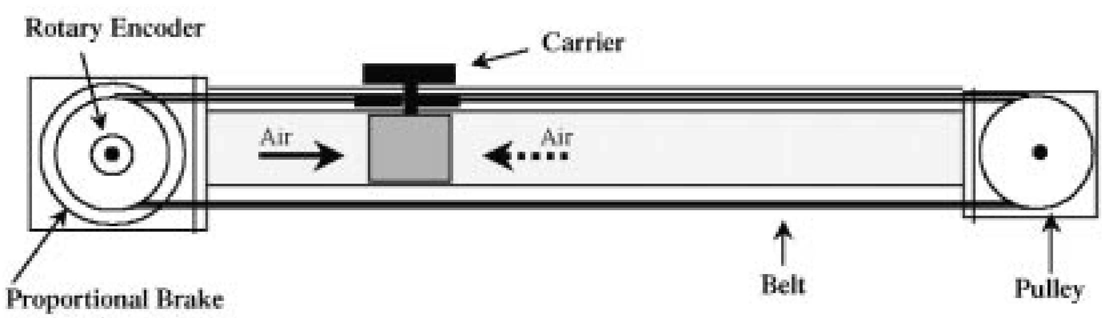


Figure 1.1 PrecisionAire actuator

The controller operates from line voltage, and therefore, does not require additional external power supplies. Setup is quick and easy using the Windows-based Icon programming through the RS232 serial port or with the optional onboard keypad and LCD screen. Teach modes are available in either mode of programming. Default servo gains are supplied with the PrecisionAire system. Depending on loads and the desired motion profile, a manual adjustment of these gains may be required. Data Acquisition can help determine proper servo gain adjustments by comparing the actual versus the commanded move profile. The controller has internal EEPROM capable of storing up to 10 programs (each 100 lines) that can be activated directly, on power-up, or with an input.

1.2 Features

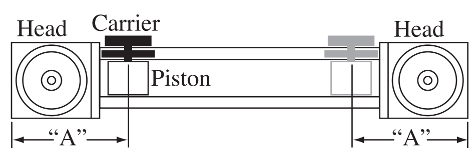
1. Position control.
2. Programmable position repeatability of +/- 0.010 in up to 120" stroke. For longer strokes repeatability can be expected better than +/- 0.025 in.
3. Repeatability independent of supply pressure, valve Cv and load changes
4. Programmable user units, motion profile and holding torque.
5. Jog and teach functions
6. One RS232 serial port for optional Windows-based programming
7. 1024 line rotary encoder for position feedback
8. 7 General purpose optically-isolated inputs and 4 optically-isolated outputs
9. Dedicated enable input, fault/error and in-position outputs
10. Two 24Vdc valve outputs. (250mA max. for each output)
11. Power ON, Fault/Error, and In-Position LED indicators.
12. Optional 4x20 character LCD display and an embedded keypad for easy set-up and programming also functions as embedded HMI
13. 64K byte EEPROM for saving up to 10 motion programs (each up to 100 command lines).
14. Uses standard directional air valves (NO SERVO VALVES required)
15. Data collection for tuning (available with Windows-based software)
16. Pluggable screw terminals (no breakout terminals required)
17. Short-circuit protection, current fault, position fault, configurable software limits protection.
18. Actuator available in 1" and 1½" bore sizes.
19. Re-circulating ball-bearing load support system provides, high direct load capacity, high moment load capacity, and wear resistance.
20. Wedge design guarantees that the raceways are parallel, which ensures a pre-load that is consistent throughout the length of the cylinder.
21. Compact package size (compared to electric systems)

1.3 Controller/Drive Specifications

Power	**1-in brake (PAS10)**	**1.5 in brake (PAS15)**
Continuous current	1 Amp	2 Amp
Peak current (1 sec)	2 Amp	4 Amp
Input Voltage (single/3 phase)	95-130 Vac (190-250Vac) (Voltage range is switch selectable)	
Input frequency	47-63 Hz	
Serial communication port		
Type	RS-232	
Settings	19200 baud, 8 data bits, no parity, 1 stop bit, no flow control	
Inputs and outputs		
Dedicated optically isolated input ENABLE	5-24Vdc, 15mA max Can be configured to source or sink current	
Dedicated optically isolated outputs 2 valve solenoid outputs in-position & fault outputs	 24Vdc max, 250mA max 24Vdc max, 20mA max	
General-purpose optically isolated inputs seven inputs	5-24Vdc, 15mA max Can be configured to source or sink current	
General-purpose optically isolated outputs four outputs	5-24Vdc, 20mA max Can be configured to source or sink current	
Encoder Feedback	1024 lines, Incremental, 5Vdc, differential, A/B channels	
Connectors		
Serial	9 pin D-sub.	
All others	Pluggable screw terminal blocks	
Environmental		
Storage temperature	-4F to 158F (-20C to 70C)	
Operating temperature	32F to 104F (0 to 40C)	
Humidity	5% to 95% non-condensing	
Mechanical		
Dimensions	5.8" height x 10.1" wide x 3.3" deep	
Weight	8 lbs (3.7 kgs)	

1.4 Actuator Specifications

Specifications:	PAS10 (1 inch bore)	*metric equiv.*	PAS15 (1.5 inch bore)	*metric equiv.*
Base Weight (incl. carrier)	14.82 lbs.	(6.72 kgs.)	23 lbs.	(10.43 kgs.)
Weight per inch of stroke	0.308 lbs.	(0.0055 kgs./mm)	0.504 lbs.	(0.01114 kgs./mm)
Maximum Stroke Length	18'	(5.5 m)	16' 5"	(5.0 m)
Dead Length (2 x "A")	15.7"	(399 mm)	22.64"	(575.1 mm)
Maximum Load at Maximum Speed	75 lbs. @ max. speed of 100 in./sec.	(34 kgs. @ max. speed) (of 2.54 m/sec.)	150 lbs. @ max. speed of 100 in./sec.	(68 kgs. @ max. speed) (of 2.54 m/sec.)
Positional Repeatability	± 0.010" up to 120" strk	(± 0.25 mm up to 2540mm strk)	± 0.010" up to 120"strk	(± 0.25 mm up to 2540mm strk)
Operating Temperature	32°F to 104°F	(0° to 40°C)	32°F to 104°F	(0° to 40°C)
Recommended Belt Tension	60 lbs.	(267 N)	122 lbs.	(543 N)
Maximum Air Pressure	100 PSI	(6.89 Bar)	100 PSI	(6.89 Bar)
Maximum Load Fz	591 lbs.	(268 kgs.)	1,454 lbs.	(660 kgs.)
Fy	341 lbs.	(155 kgs.)	840 lbs.	(381 kgs.)
Maximum Bending Moments				
Mx	250 in.-lbs.	(28.25 N-m)	859 in.-lbs.	(97.06 N-m)
My	269 in.-lbs.	(30.39 N-m)	1033 in.-lbs.	(116.72 N-m)
Mz	156 in.-lbs.	(17.63 N-m)	596 in.-lbs.	(67.34 N-m)
Pulley Pitch DIA.	1.88"	(47.7 mm)	2.506"	(63.7 mm)
Belt Width	0.75"	(19 mm)	1.00"	(25 mm)
Length of Air Cushion Spear	1.0"	(25.4 mm)	1.7"	(43.1 mm)
Brake Coil Resistance	8 ohms	8 ohms	4 ohms	4 ohms



Dead length = 2 x "A"

1.5 Agency Approvals

Refer to approval agency marks on the controller/drive or contact
Tol-O-Matic for latest information on agency approvals.

Before you begin…

To reduce risk of injury and equipment damage and eliminate wasted time and effort, please read this manual in its entirety before attempting to install or operate the controller.

NOTES:

Chapter 2 Safety

2.1 Potential Hazards

The equipment described in this manual is intended for use in industrial control/drive systems. This equipment can endanger life through moving machinery and high voltage, therefore it is essential that guards for both electrical and mechanical parts be in place.

Hazards which can be encountered in the use of this equipment include:

- Electrical shock
- Electrical fire
- Mechanical
- Stored energy

These hazards must be controlled by suitable machine design, using the safety guidelines which follow.

2.2 Voltage Potentials

Voltage potentials for the internal drive circuitry vary from 325 volts above to 325 volts below earth ground for a 230 volt input. Voltages can reach 88 Vdc within the controller. All circuits, including the connections on the front panel, should be considered “hot” when power is connected.

2.3 Installer Responsibilities

As the user or person installing the controller/drive, you are responsible for determining the suitability of the product for the intended application. ***Tol-O-Matic is neither responsible nor liable for indirect or consequential damage resulting from the use of this product.***

A *qualified person* is someone who is familiar with all safety codes and established safety practices pertaining to the installation, operation and maintenance of this equipment and the hazards involved. For more detailed definition, refer to IEC 364.

It is recommended that anyone who operates or maintains electrical or mechanical equipment should have a basic knowledge of First Aid as a minimum, they should know where the First Aid kit is kept and the identity of the official First Aid personnel.

These safety notes do not represent a complete list of the steps necessary to ensure safe operation of the equipment. For further information, please contact the nearest Tol-O-Matic distributor.

2.4 Safety Guidelines

Electrical shock and fire hazard can be avoided by using normal installation procedures for electrical power equipment in an industrial environment. Installation must be undertaken by suitably qualified personnel.

Mechanical hazards are associated with potential uncontrolled movement of the actuator. If this poses a risk in the machine, appropriate precautions must be made to disconnect the air source when personnel have access to moving parts of the machine. Note also that the brake must be securely mounted at all times.

Storage energy hazards are both electrical and mechanical.

1. Electrical hazards can be avoided by disconnecting the controller/drive from its power source and waiting for at least 1 minute prior to removing protective covers or touching any connections.

2. Mechanical hazards require a risk analysis on the effects of stored mechanical energy when the machine is running at speed, as well as the potential for the disconnection of the brake while air source is applied.

The following points should be observed for the safety of personnel:

- Only qualified personnel familiar with the equipment are permitted to install, operate and maintain the device.
- System document must be available and observed at all times.
- All non-qualified personnel should maintain a safe distance from the equipment.
- The system must be installed in accordance with local regulations.
- The equipment is intended for permanent connection to a main power input. It is NOT intended for use with a portable power input.
- DO NOT power up the unit without all guards and covers in place.
- DO NOT operate the unit without connecting the brake conductors to the appropriate terminals on the controller/drive.
- Always remove power before making or removing any connection on the unit. Failure to observe this condition could result in injury or damage to equipment.
- DO NOT remove cover from unit while in operation.

- DO NOT make any connections to the internal circuitry. Connections on the side panels are the only points where users should make connections.
- Be careful of the line voltage input and brake output terminals. High voltage is present when power is applied to the controller/drive.
- DO NOT use the enable input as a safety shutdown. Always remove power to a controller/drive before maintaining or repairing the unit.

NOTES:

Chapter 3 Unpacking, Inspection and Storage

3.1 Unpacking the Controller/Drive

Remove the PrecisionAire™ controller from the shipping carton. Retain the shipping materials for storage or in case the unit needs to be returned. Check contents against the packing list. Model, part number and related information appear on a label on the bottom of the controller/drive.

3.2 Inspection Procedure

To protect your investment and ensure applicable warranty rights, Tol-O-Matic recommends the unit be carefully inspected for any signs of physical damage. If any damage is detected, contact the purchasing agent to make a claim with the shipper.

If any improper performance is detected while testing the unit, contact your Tol-O-Matic distributor to obtain a Return Material Authorization (RMA). Do this as soon as possible after receipt of the unit.

For specific warranty information, refer to Appendix A in this manual.

3.3 Storage

Return the controller/drive to its original shipping carton using the original packing materials. Store in a clean dry place with humidity within 5% and 95%, non-condensing. Make sure the temperature is between -20C and 70C (-4F and 158F).

NOTES:

4.1 Actuator

For intermediate support, tube supports or mounting plates can be mounted to the PrecisionAire actuator. The number of tube support brackets or mounting plates required and their placement depends on the overall length of the actuator and the total weight being moved and supported. Refer to the tube support data chart Fig. 4.1 below.

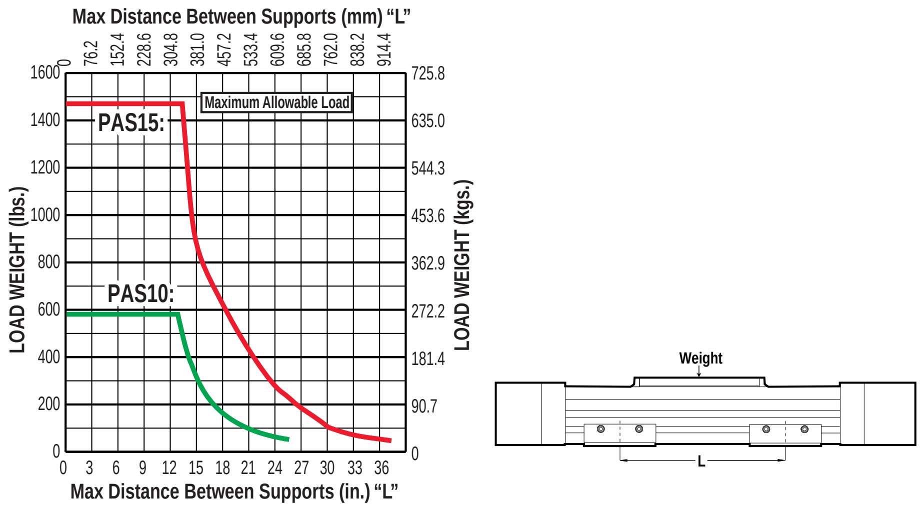


Figure 4.1 Tube support requirements

Recommended belt pretension:

1" bore	60 lbs.	267N
1½" bore	122 lbs.	543N

When optional shock absorber is ordered for heavier load homing, a shock plate is mounted on top of the carrier that will change the mounting pattern on the carriage. Please refer to the PrecisionAire catalog for shock performance charts and plate mounting dimensions. The shock absorber will be mounted on the non-brake end at the factory.

4.2 Controller/Drive

1. The controller/drive unit must be mounted in a proper electrical enclosure providing protection to IP54 (protected against dust and splashing water), or IP65 (dust free and protected against water jets) where the environment is poor. Many NEMA (National Electrical Manufacturers Association) Type 4 cabinets provide this level of protection.

2. Size the enclosure to provide the following spacing around the controller/drive:

Above and below:	7.6 cm (3 in)
Sides:	5.1 cm (2 in)
Front:	1.25 cm (0.5 in)

Caution! If the cabinet is ventilated, use filtered or conditioned air to prevent accumulation of dust and dirt inside the controller/drive. The air must be free of oil, corrosive or electrically conductive contaminants.

3. Position the controller/drive on a flat, solid surface capable of supporting the controller/drive's weight

4. Bolt the unit to the cabinet using the mounting slots on the controller/drive. Use M5 metric or #10 standard screw for mounting.

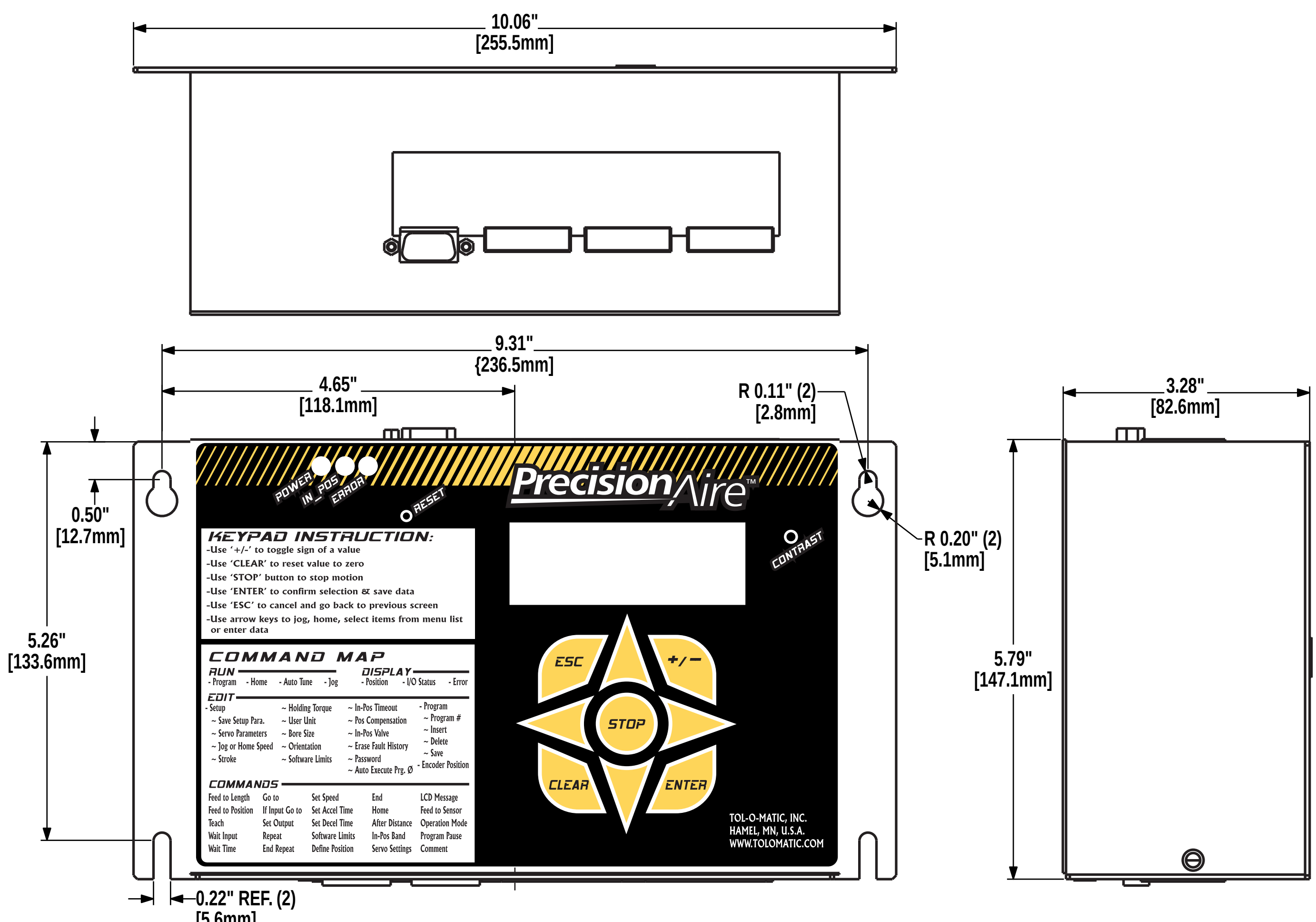


Figure 4.2 Controller/Drive dimensions

5.1 Setting Up the Air System

Pressure regulation is necessary to achieve optimum PrecisionAire system performance. An external pressure regulator or a valve with a sub-base mounted pressure regulator can be used.

It is recommended to operate at the minimum supply pressure necessary to achieve the applications speed or force. Operating at the minimum air pressure will help to reduce overshoot/undershoot, give more consistent operation, and ease the tuning of the system. Typically this air pressure will be no more then 10 PSI above what is required for the desired acceleration, velocity, or force. Limiting the air pressure will reduce the brake regulation at constant speeds, which can reduce system 'hesitations' caused by the brake trying to overcompensate for higher air pressures. It will also reduce the likelihood of overshooting/undershooting or servoing into position. PrecisionAire software has an easy to use Data Acquisition that can insure the proper air pressure is selected for the applications loads or speeds.

See 5.4 for air pressure considerations for Vertical Applications.

See chapter 7 Tuning for selecting air pressure example.

5.2 Valve Connections

Caution! **Do NOT use relays or PLC outputs to control solenoid valves. Use valve outputs directly from the PrecisionAire controller. Valves operated by 24Vdc solenoids are required for PrecisionAire systems.**

Cv is a number expressing the ability of a fluid to flow under pressure difference or pressure drop. It is also referred to as flow capacity or flow coefficient. The greater the Cv value, the better the flow. Cv is analogous to electrical conductance.

The required Cv for valves used in PrecisionAire systems is based on the motion profile. Cv may affect maximum speed achieved and response time.

> **Note:** Valve Cv can affect the performance of the Precision-Aire system. Recommended Cv ratings are: PAS10 = Cv 1.2 or higher; PAS15 = Cv 1.8 or higher. Too much flow is never a problem with PrecisionAire. Not enough flow can severely hinder performance.

The following three types of valve configurations (Figures 5.1 to, 5.3) can be used to operate the PrecisionAire system. **However, the solenoid coil response time will affect the in-position settling time of the system. A solenoid response time of less than 20 ms is required.**

The use of two 2-position, 3-way valves directly plumbed to the PrecisionAire cylinder ports is the **preferred** method of valving for **optimum** performance and required for vertical or long-stroke applications. Using a 3-position 4-way valve is an optional method for short stroke horizontal installations.

When plumbing the air valves to the PrecisionAire actuator, optimal system performance can be obtained by mounting the valves in close proximity to the PrecisionAire actuator. By maintaining short air-line lengths between valve(s) and PrecisionAire actuator, the response time will be optimized. If valves cannot be mounted in close proximity to the PrecisionAire actuator, it is recommended to maintain the same length of air-line for each valve.

A. Using two 2-position 3-way normally open valves with 24Vdc solenoids: Connection is shown in Fig. 5.1. When using this valve configuration Tol-O-Matic recommends **"De-Energized"** for the "In Position Valves" selection in the Set-up portion of the PAS controller software. (see Figure 6.1, pg. 6-4)
*Valve A is connected to the actuator non-brake end and valve **B** is connected to the brake end.*

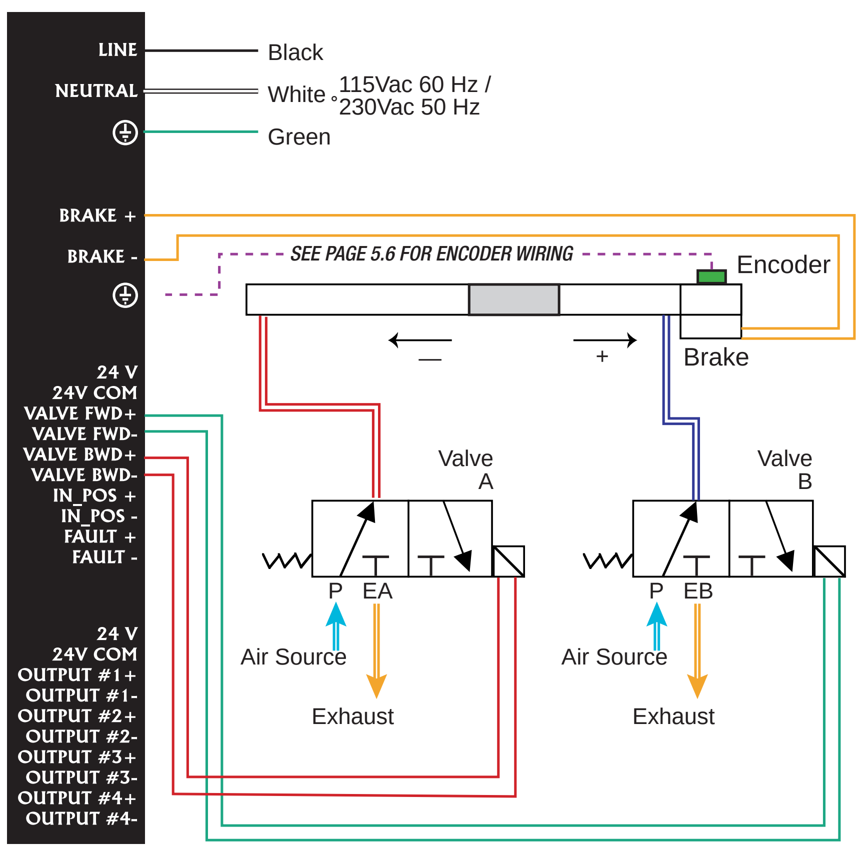


Figure 5.1
3-Way Normally Open Valve Connections

B. Using two 2-position 3-way normally closed valves with 24Vdc solenoids: Connection is shown in Fig. 5.2. When using this valve configuration Tol-O-Matic recommends **"Energized"** for the "In Position Valves" selection in the Set-up portion of the PAS controller software. (see Figure 6.1, pg. 6-4)

Valve ***A*** *is connected to the actuator non-brake end and valve* ***B*** *is connected to the brake end.*

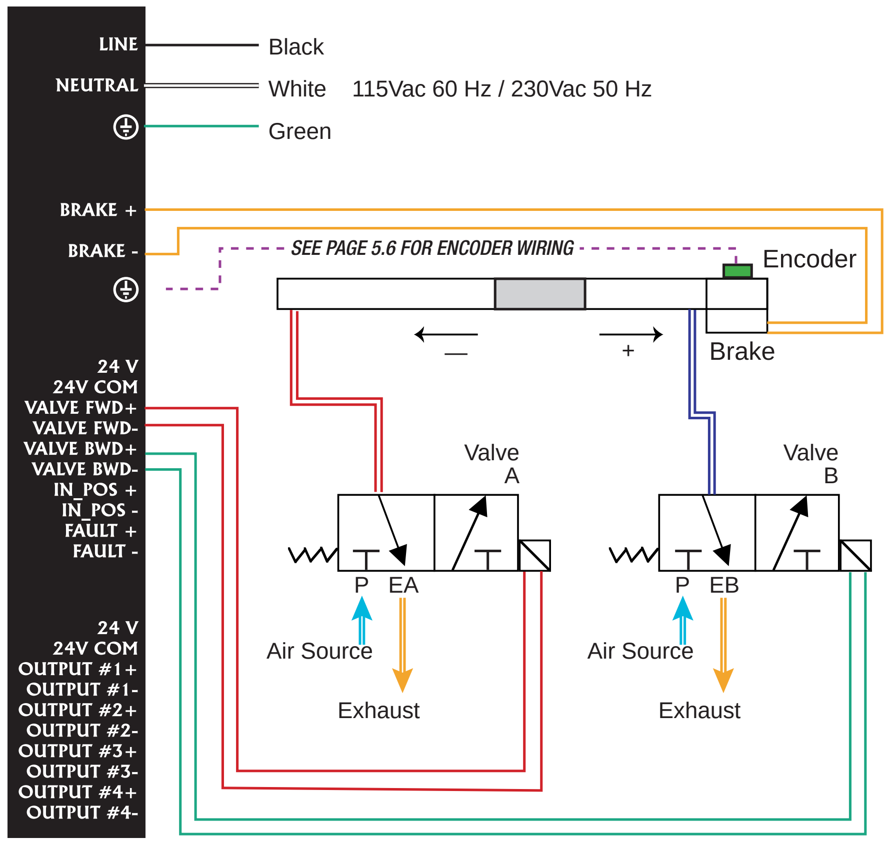


Figure 5.2
3-Way Normally Closed Valve Connections

C. Using a 3-position 4-way, spring centered valve with the center configured with pressure (P) to cylinder ports A and B, with dual 24Vdc solenoids: Connection is shown in Fig. 5.3. Select valve "In Position Valve" (default is "De-Energized")(see Figure 6.1, pg. 6-4)
*Port A is connected to the actuator non-brake end and port **B** is connected to the brake end.*

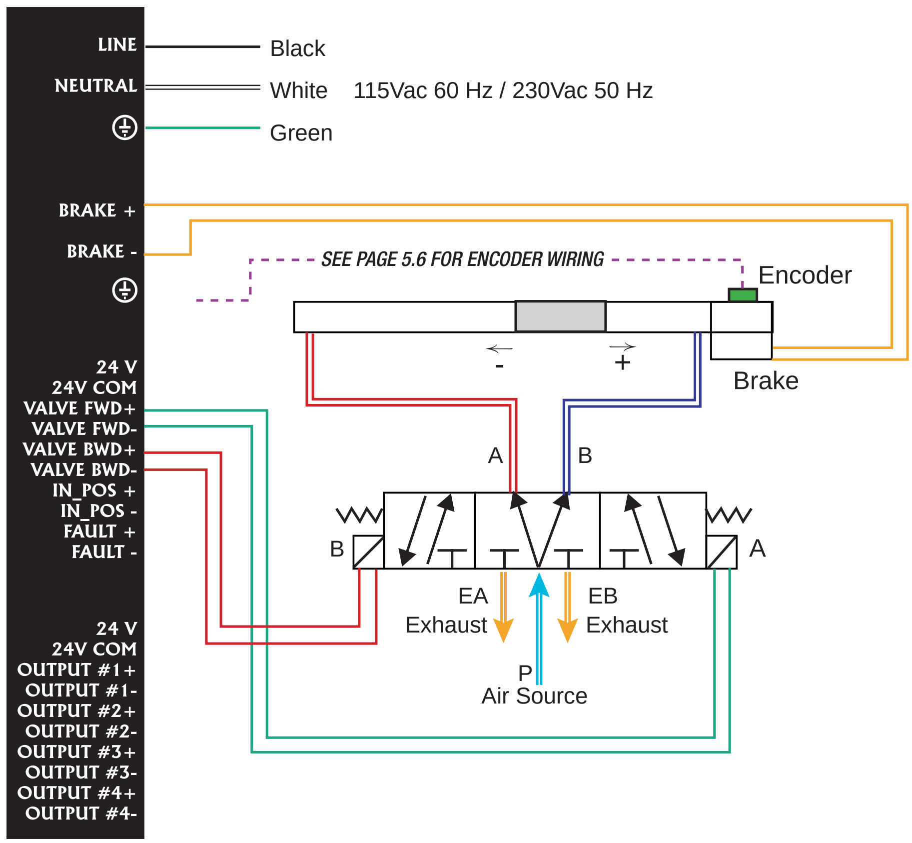


Fig. 5.3
3 Position 4-Way, Spring Centered Valve Connections

For short stroke, horizontal applications only.

5.3 Horizontal Applications

For horizontal applications, a two valve (See Figures 5.1and 5.2 system is preferred for optimal performance. However, a single valve (See figure 5.3) can work for short stroke applications. When plumbing the PrecisionAire actuator, keep the valve(s) as close to the actuator as possible. If operating with a single valve or if the valves cannot be mounted close to the actuator, use air lines of the same length to optimize the system performance.

5.4 Vertical Applications

For vertical applications, a dual-pressure system (similar to figure 5.1 or 5.2) must be used. Using two separate external pressure regulators or a valve with a sub-base mounted pressure regulator will be required to achieve a dual pressure system. Vertical applications should always be mounted with the brake end at the top. This will minimize the chance of any belt slack at the brake. Typically the downward motion will require less then half the pressure of the upward motion.

Mounting the valve as close as possible to the port of the PrecisionAire actuator and using equal length air lines will optimize the PrecisionAire system performance.

5.5 Brake Wiring

Connect the two brake wires to the 'BRAKE +' and 'BRAKE -' terminals on the controller.

Brake Coil Resistance	
PAS10	about 8 Ohms
PAS15	about 4 Ohms

NOTE:
Do NOT connect brake wires across 'BRAKE +' and 'GND', or 'BRAKE -' and 'GND' at brake terminal. Do not connect any wires to 'GND' Applying AC power to brake terminal will damage controller permanently. Brake coil may be damaged or shorted when controller is damaged. None of the brake wire leads are shorted to the brake housing.

WARNING:
Prior to applying power to controller, test the brake coil resistance. A shorted brake could permanently damage the controller. Operating at a brake coil resistance below 7.2 ohms for a PAS10 or 3.6 ohms for a PAS15 could permanently damage the controller.

5.6 Encoder Wiring

Encoder pin assignment and color codes are shown in Fig. 5.4. Tol-O-Matic provides 15 feet of encoder cable.

NOTE:

Do NOT combine encoder wires and brake wires in a single cable. The encoder may pick up line noise from the brake power signals.

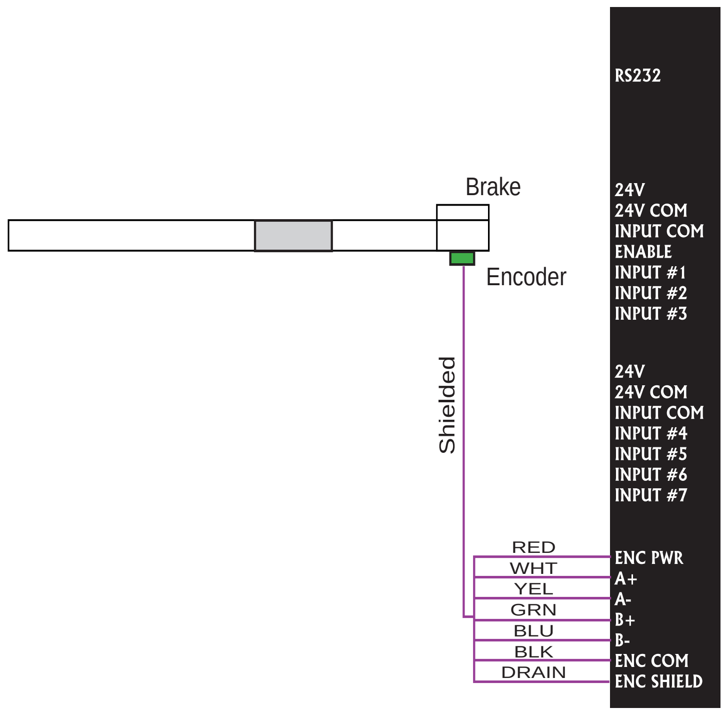


Figure 5.4 Encoder to PrecisionAire Controller Connections

5.7 115/230 ac Power Wiring

Be sure to set the ac power switch to the correct value (115 or 230 Vac) and connect the line, neutral and ground wires to the terminal as labeled (Figure 5.5). The factory setting for the controller/drive is 230 Vac.

Do NOT connect ac power to brake terminal.

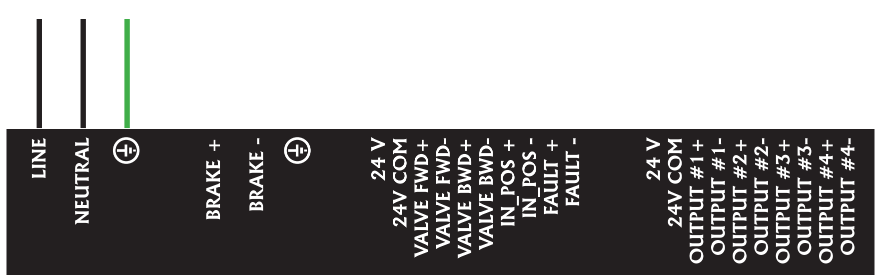


Figure 5.5 Connect ac power to PrecisionAire controller

5.8 Input Wiring

The PrecisionAire controller/drive has an ENABLE input that enables the controller/drive for operation when active.

NOTE 1: If the current is flowing through an input, the logic state of that input is LOW (i.e., circuit is CLOSED). I/O display on the LCD should indicate '0' for that input. If no current is flowing the logic state is HIGH (i.e., circuit is OPEN). I/O display on the LCD will indicate '1' for the input.

Warning! ***The ENABLE input should never be used for an emergency stop or to put the system into a "safe" condition. Always remove air from the actuator and power from the controller/drive before servicing the system.***

All the inputs are optically isolated. Each input channel can either sink or source up to 15 mA. The input channels can be connected in different ways as illustrated in Fig. 5.6. Tol-O-Matic can supply both Hall-effect and Reed switches designed to work with PrecisionAire actuators. Figures 5.7 to 5.9 are sample connection diagrams for all types of Tol-O-Matic switches.

NOTE 2: In order to accommodate different switch types the INPUT COM for INPUTS #1, 2, 3 is not internally connected to the INPUT COM for INPUTS #4, 5, 6, & 7. If using both blocks of inputs, both INPUT COMs must be hooked up.

NOTE 3: When an input is used for the program pause or feed to sensor command, the input scanning rate is about 10 ms.

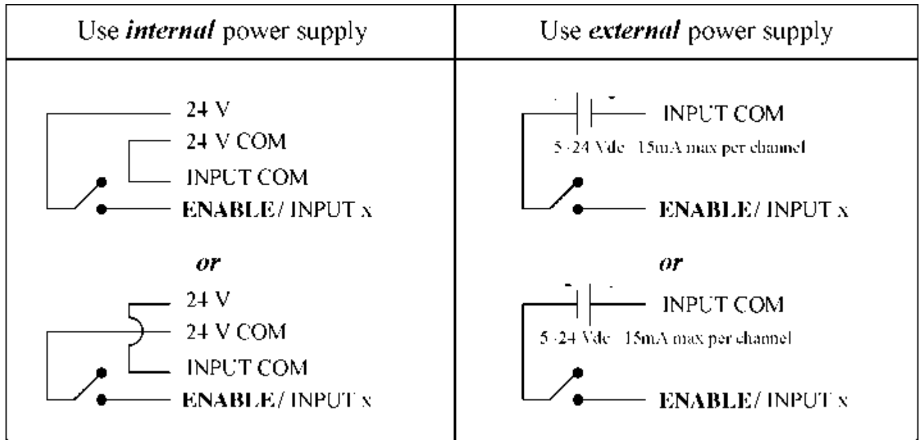


Figure 5.6 Controller Input Connections

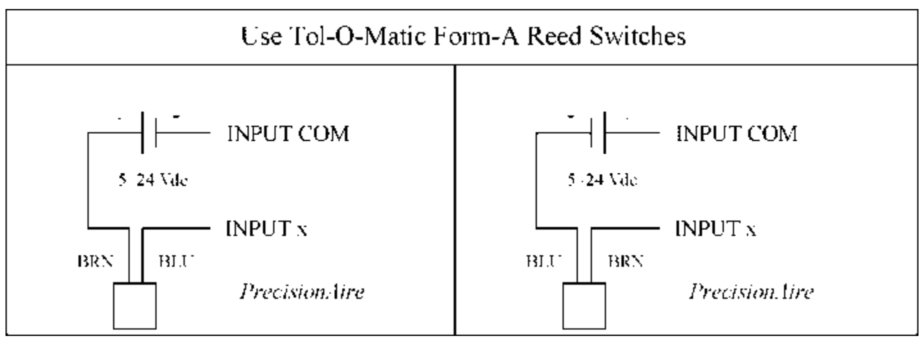


Figure 5.7 Tol-O-Matic Form-A Reed Switch Connections

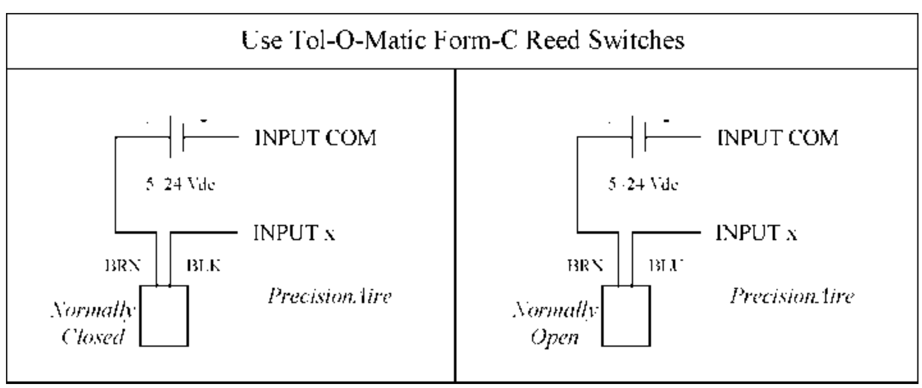


Figure 5.8 Tol-O-Matic Form-C Reed Switch Connections

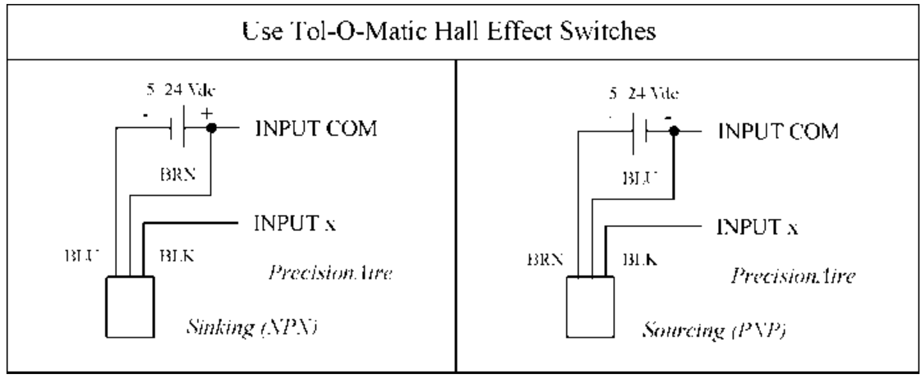


Figure 5.9 Tol-O-Matic Hall-effect Switch Connections

5.9 Output Wiring

The PrecisionAire controller/drive provides two dedicated outputs to allow the controlling PLC or motion controller to monitor the controller's status. They are IN_POSITION and FAULT connections. All outputs are optically-isolated. Each output channel can either sink or source up to 20 mA maximum. The output channels can be connected in different ways as illustrated in Fig. 5.10. There is about a 100 ms delay while the encoder physically moves into position before the IN_POS output light turns ON. This delay is necessary to avoid any false signal when an overshoot occurs.

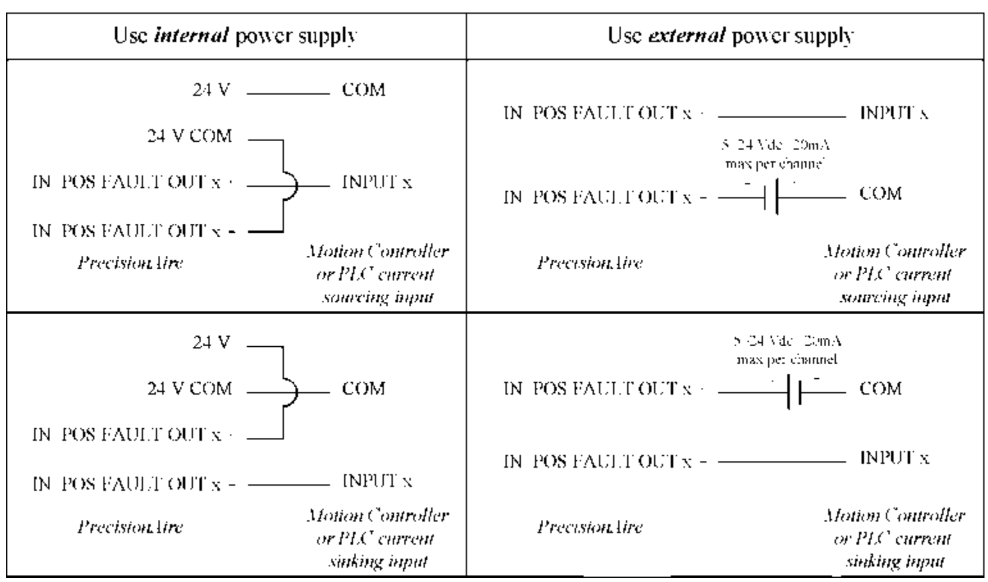


Figure 5.10 Controller Output Connections

5.10 Cushions and Shock Absorbers

Unlike the cushions on a regular pneumatic cylinder, the cushions on a PrecisionAire actuator are only utilized when homing .

Servo positioning within the cushion region is not recommended. 1.0" of stroke on each end should be added to a PAS10 • 1.7" on each end for a PAS15. Positioning within the cushion region will typically add more cycle time to a move.

NOTE: Programming speed for a home move is not allowed above 12 in./sec.. However, actual speed will be based upon air pressure, System Cv, servo parameters, and the load. It is recommended to use the Data Acquisition to properly set air pressure and servo parameters prior to homing.

The maximum speed allowed for homing is 12 in/sec (305 mm/sec). The cushions will handle a load of up to 60 lb (27 kg) for a 1-inch bore and up to 150 lb (68 kg) for a 1.5-inch bore. If the load exceeds this limit, a shock will need to be used. When using shock absorbers, always home towards the shock end. This is especially important with factory installed shock absorbers as the actuators internal cushions are removed. Homing away from the shock absorber can cause damage to the actuator.

Chapter 6 Software Setup and Programming

6.1 General Considerations

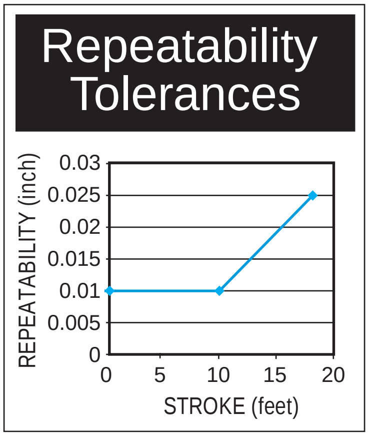


Figure 6.0 Repeatability vs. Stroke

A. Positional repeatability

The positional repeatability of the PrecisionAire system can be programmed to +/- 0.010" (+/- 0.254 mm) with increments of 0.006" (0.15 mm). Due to belt stretches, piston and bearing frictions, repeatability at carriage will be greater than +/- 0.010" for stroke lengths longer than 120 inches. Repeatability at maximum stroke of 18 feet could reach a maximum of +/- 0.025".

Note: Positioning time can be decreased if the In-Position Band is increased. The lower the In-Position Band is set the greater likelihood overshoot or servoing into position could occur affecting the cycle time. It is recommended to operate at the highest In-Position Band possible, this will improve cycle time and reduce hunting for position.

B. Holding Torque

The in-position holding torque can be set from 0 to 100% of the maximum brake torque. During dwell time (especially in horizontal applications), it is often possible to reduce the torque necessary to hold a load. By reducing the holding torque (as in stepper systems), the efficiency is improved and the holding device is not required to dissipate as much heat. This improves the allowable duty cycle. The factory default holding torque setting is 25%. The maximum holding torque for a PAS10 is 94 lbs. For a PAS15 251 lbs.

C. Motion profile

Motion profiles are user defined by programming the move distance, the maximum speed, acceleration time, and deceleration time. How well the actual motion follows the theoretical profile is based on the system supply pressure, system Cv, the load, and how well the system is tuned. During acceleration the brake is released and the ramp rate is limited by the traditional pneumatic cylinder limitations mentioned above. During the constant velocity portion of the profile, encoder information is evaluated to supply a brake current necessary to maintain the programmed speed. Deceleration is the most tuning critical portion of the profile. In order to reduce the high speed thrust requirement of the belt and torque of the brake, both valves are de-energized to allow air circulation through the valve(s) during the initial part of the deceleration. The gain settings will control how linear the deceleration is over the programmed deceleration time. If the ratio of KT to KI is not appropriately set, a rapid deceleration will be experienced with a short slow move into final position. Users will experience positioning overshoot on PAS systems. However, overshoot can be minimized by adjusting servo settings or deceleration time in the motion profile. Decreasing the air pressure can also help minimize overshoot as well as increasing the in-position band. Increasing the in-position band will reduce in-position settling time, if a larger position window is acceptable. Refer to the Chapter 7 (Tuning) for more detail.

D. Air Cushions

Positioning within the air cushion is *not* recommended. Doing so may cause damage to the cylinder. Cushion spears are approximately 1-inch long for the PAS10 and 1.7 inches for the PAS 15. Ordering an additional 1" of stroke is recommended on each end of the actuator to avoid positioning within the air cushions. Positioning within the cushion region will typically add more cycle time to a move.

6.2 PC Hardware Requirements

- An IBM compatible computer running Microsoft Windows® 95, 98, 2000 or NT.
- A hard disk with 10 MB of free disk space.
- 32MB of RAM minimum

6.3 PC Software Installation

- Close all Windows® programs.
- Insert the Tol-O-Matic CD and open.
- Under 'Programming Software' select PrecisionAire software.
- Double click to open.
- Follow the on-screen instructions that appear.
- After installing, run the PrecisionAire software to start motion control & programming.

6.4 System Setup

Following are the recommended steps to use once the PrecisionAire system has been mounted and wired according to the previous sections.

1. Install software (see 6.3) or if using the integral keypad and LCD, see Section 6.6.
2. Program the following initial setup parameters (see 6.5 for setup parameter descriptions)
   - Bore size (1 or 1.5 in bore)
   - Actuator orientation (horizontal or vertical)
   - User unit (inches or mm)
   - Stroke length (Specify available stroke length before tuning)
   - In-position holding torque (25% as default)
   - Select valve "In Position Valve" (default is "De-Energized")
   - In-position time out (default to 60 seconds)
   - In-position band (default to 0.010" as minimum)
   - Software limits (default to OFF)
3. Enable Controller
4. Determine proper air pressure to achieve desired acceleration/velocity using the Data Acquisitions actual versus commanded speed. See air system considerations (5.1) for velocity control. See 7.1 Data Acquisitions for selecting air pressure example.
5. Determine proper servo gains to achieve desired motion. See Servo Parameter Gains (6.5) for overshoot/undershoot considerations. See chapter 7 Tuning for more details.
6. Create and run program (see 6.5)
7. Air pressure, servo gains, deceleration values, and in-position band may still require adjustment in the setup window or within the program to optimize performance.

NOTE: If using a replacement controller test the brake coil resistance prior to applying power to the controller. See 5.5 brake wiring.

6.5 Set-up Parameter Definitions

Once the PrecisionAire system has been mounted and wired, set-up parameters can be entered by installing the PrecisionAire software on a PC or by using the Keypad and LCD display. For procedures on using the keypad and LCD to set up and program, refer to section 6.7 Setup and Programming from Keypad and LCD.

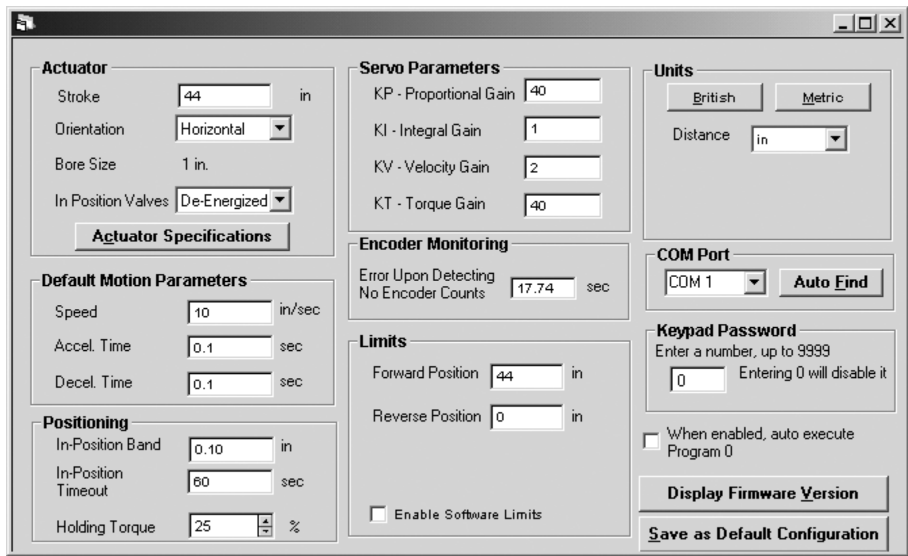
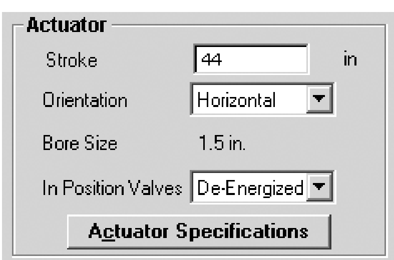


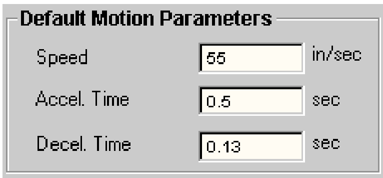
Figure 6.1 Software Set-up Screen

ACTUATOR PARAMETERS



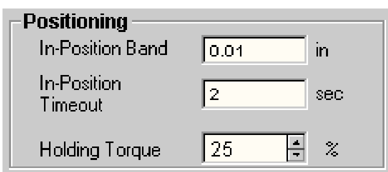
Actuator stroke, orientation, bore size and valve configuration need to be assigned before tuning and programming.
To change bore size, select Tools and Advanced Settings.

DEFAULT MOTION PARAMETERS



Default motion parameters for maximum speed, accel and decel times will be used for each new move created in a program. Parameters can be changed for specific moves in the programming screen. Performance of the move profile is based on the system supply of air pressure, valve CV , load and tuning. Data Acquisition files can help determine achievable acceleration rates and speeds of your system as well as necessary deceleration rates for desired performance.

POSITIONING



In-Position Band

The in-position band refers to the repeatability of the system. The system can be programmed to +/- 0.010" (for strokes of 120" or less) in 0.006" increments. Repeatability at the carriage will be greater than +/- 0.010" for strokes longer than 120" due to belt stretches, piston and bearing frictions. Repeatability for stroke lengths over 120" will automatically be calculated by the controller and displayed in the appropriate software window.
Note: Positioning time can be decreased if the In-Position Band is increased. The lower the In-Position Band is set the greater likelihood overshoot or servoing into position could occur affecting the cycle time. It is recommended for applications where cycle time is critical to operate at the highest In-Position Band possible.

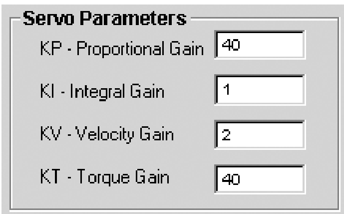
In-Position Time Out

In-position time out is the time allocated for the executed move to reach its desired position before a system error occurs. This is adjustable from 1 sec to 60 sec.

In-Position Holding Torque

In-position holding torque is the amount of brake pressure applied to maintain position. Torque can be set from 0 to 100% of maximum brake torque. During dwell time (especially in horizontal applications), the amount of brake torque required to hold a load can often be reduced for improved efficiency. The maximum holding torque for a PAS10 is 94 lbs. For a PAS15 251 lbs.

SERVO PARAMETER GAINS



Servo parameters are used for tuning.

KP Gain

The KP gain determines how sensitive the controller will respond to a programmed move. Default is set at 40. For most applications, no adjustment should be necessary.

KI Gain

The KI gain is the controller response to achieving a position. Default is set at 1. Increase when making small moves of 0.5 inches or less.

KV Gain

The KV gain is the controller's response to the commanded velocity. Default is set at 3. Increase when the actual velocity needs to be as close to the commanded velocity as possible.
Note: See air system considerations 5.1 for velocity control.

KT Gain

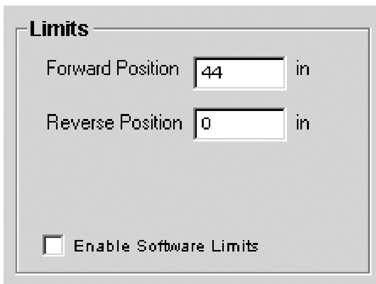
The KT gain is the brake response to acceleration. Default is set at 40. Increase if overshooting is present, which will increase the torque applied to the brake.

Note: Decreasing air pressure, increasing in-position band, or increasing deceleration time can all reduce overshoot.

ENCODER MONITORING

The encoder monitoring continually checks the encoder for feedback while a move is taking place. If the programmed time elapses with no feedback a position fault 03 will occur. Depending on load, air pressure and tuning the minimum encoder monitoring time can vary. It is generally recommended to have a time greater than 300 msec.

SOFTWARE LIMITS



The position software limits are system "checks".

Forward and Reverse Positions

The forward and reverse position limits of the system are relative to the defined home position. In (Figure 6.2) the 26" stroke is utilizing 24" of actuator travel with 1" of extra stroke (as recommended) on each end. The forward position limit determines how far the carrier will move in a positive direction (toward the brake) before the system will fault out. If the home position of "0" is determined 4" from the actuator end, the maximum forward position limit for this actuator would be 21". The Reverse Position limit determines how far the carrier will move in a negative direction (away from the brake) before a fault occurs. The maximum negative direction in this case would be –3".

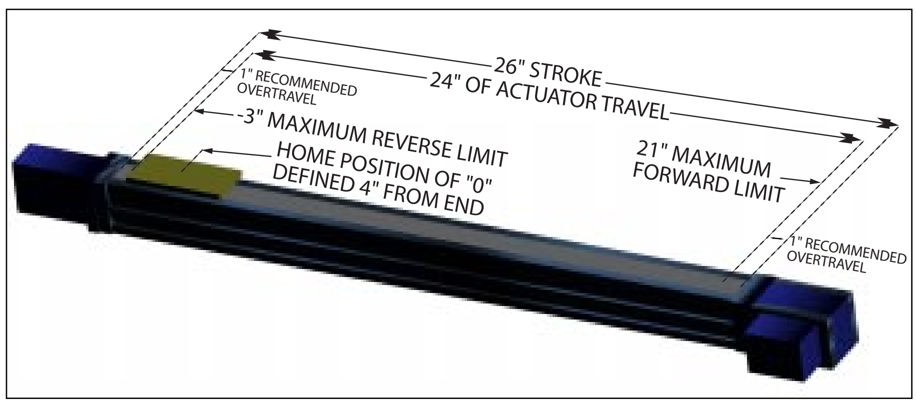
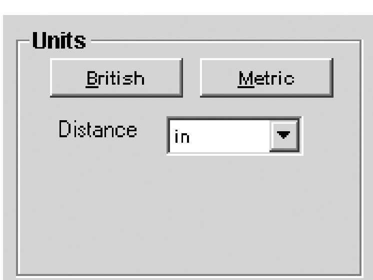


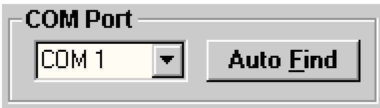
Figure 6.2 Forward and Reverse Position Limits

UNITS



Units can be specified in either a British or Metric unit display.

COM PORT



The software accommodates 8 different COM port settings. Clicking the Auto Find button will automatically find the current COM port setting.

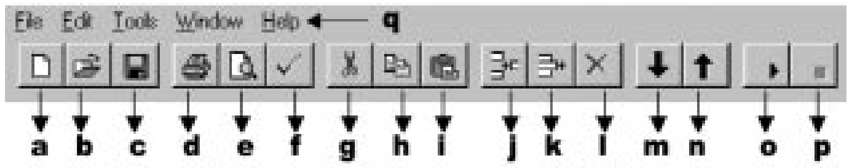
PASSWORD PROTECTION

For use with the keypad interface, a 4-digit numerical password can be entered (up to 9999) in order to access program functionality. The keypad password can be uploaded when uploading the setup parameters from the controller. A new password will be saved in the controller EEPROM when setup parameter are downloaded and saved. Specifying a password of "0" will disable keypad password protection

6.6 Programming from Windows-based PC software

The software contains four main screens: setup, programming, display & diagnostics and data acquisition.

A. Toolbar Descriptions



a. New program
b. Open setup, program or data acquisition file
c. Save setup, program or data acquisition file
d. Print
e. View program source code
f. Program syntax check
g. Cut program line(s)
h. Copy program line(s)
i. Paste program line(s)
j. Insert program line
k. Delete program line(s)
l. Clear program line(s)
m. Download setup settings or motion program to controller
n. Upload controller settings or program to PC
o. Run program or start display screen update
p. Stop program or terminate display screen update
q. Press F1 for the Help menu

B. Programming

Click on the ‘PROGRAMMING’ (See Figure 6.3) button to switch to the programming window. There are 20 commands for PrecisionAire programming. Users can program or edit multiple programs at a time. Click on ‘File’ then ‘New’ to start a new program. Click on a blank program line to display command icons.

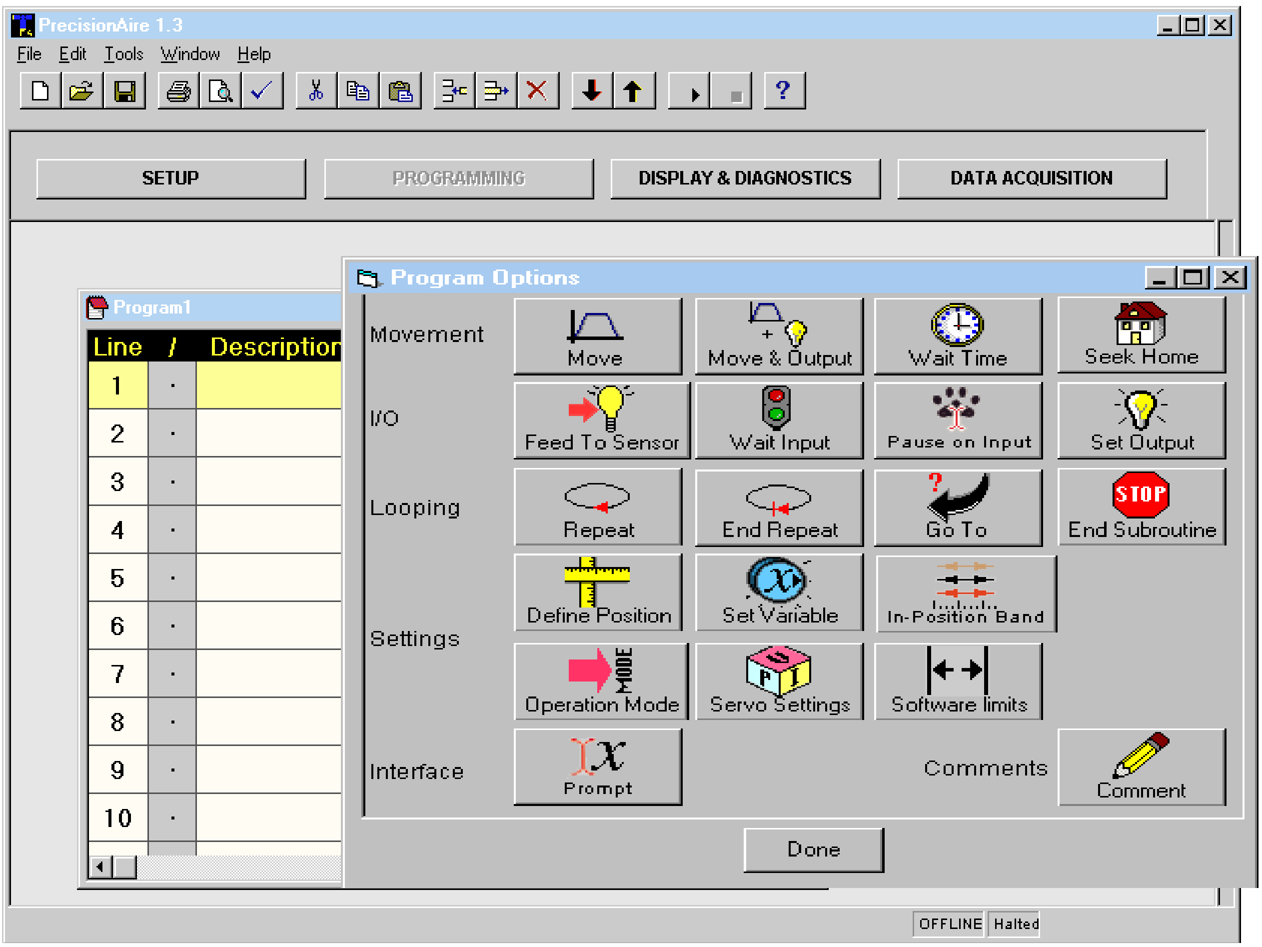
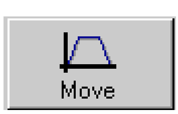


Figure 6.3 PrecisionAire software – Programming Options

MOVE OPTIONS



1. MOVE — Move the carrier

Users can specify an absolute or incremental move by using the default move parameters or entering new values (Figure 6.4). Or, click on the "Teach Position" (See Figure 6.5) button to jog or manually move the carrier to a desired position for programming. Enter the desired move distance or position in the distance text box and edit the speed, accel time and decel time text boxes appropriately for the move.

Note: The absolute direction is always positive toward the brake end, regardless of which end is used for the home position. If the brake end is defined as the zero position, negative absolute values must be used.

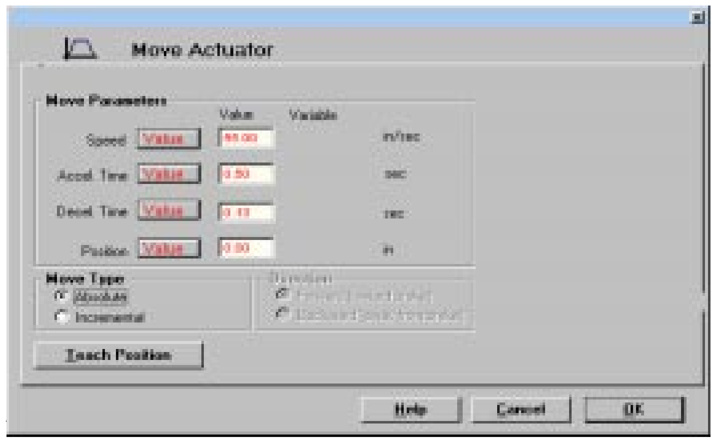
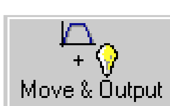


Figure 6.4 Move Command

Figure 6.5 Teach Command



2. MOVE & OUTPUT — Move carrier and set output

This is similar to the move command, but gives the option of setting an output during or at the end of a move. (See figures 6.6-6.7)

Note: If setting an output at the end of a move profile, the distance entered in 'after distance traveled' must be equal to the move

command distance entered. The output will be set when the actuator servos into position and the in-position light comes on. This accommodates for any possible overshoot. If the distance entered in the 'after distance traveled' is not equal to the move command distance the output will be set when the encoder sees this position reached regardless of possible overshoot. This could create a condition where the output is set while the actuator overshoots and servos back to the position.
When programming with the LCD and keypad interface, combine "AFTER DISTANCE" and "FEED TO POSITION" (or "FEED TO DISTANCE") commands for Move & Output.
TO DISTANCE") commands for Move & Output.

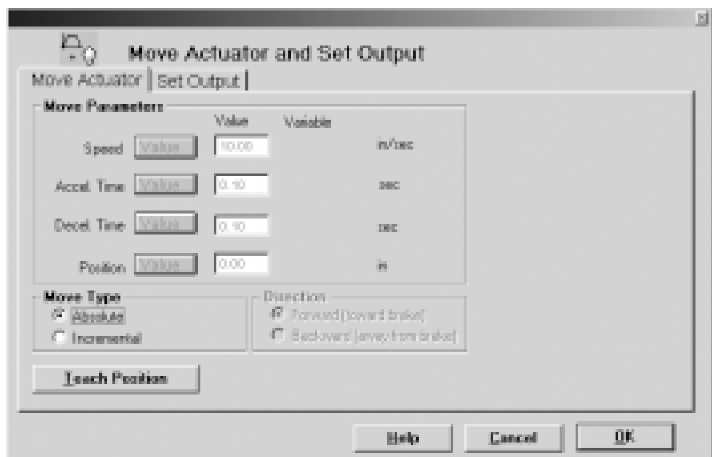


Figure 6.6 Move and Set Output Command (move)

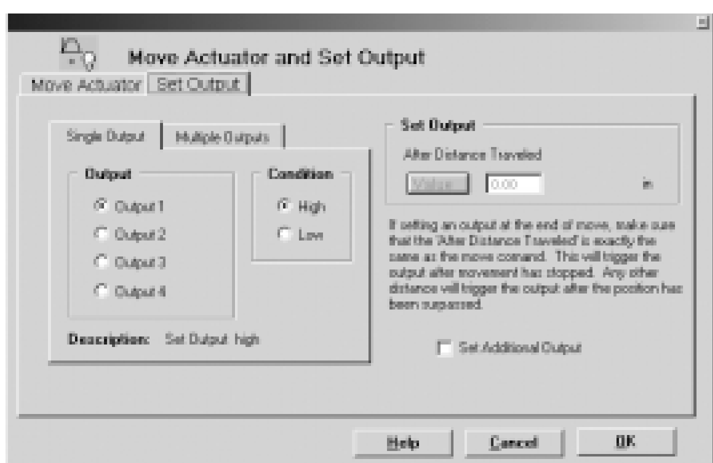


Figure 6.7 Move and Set Output Command (set output)

3. WAIT TIME — Delay for a period of time

Specify the desired delay time in the program. The time unit is determined from the setup window. (Figure 6.8)

Note: Wait times using a variable are set in msec.

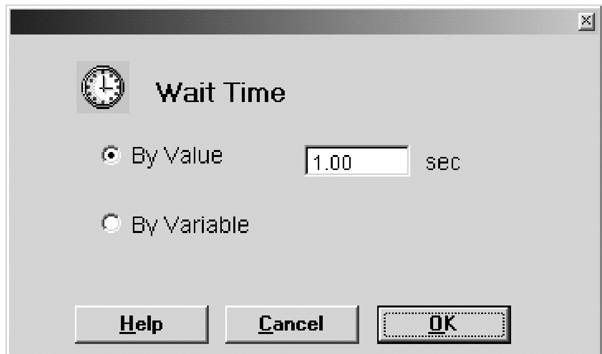


Figure 6.8 Wait Time Command

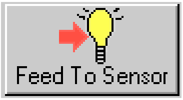
4. SEEK HOME – homing the carrier

Select the homing direction. (See Figure 6.9) The controller will use the default speed of a maximum limit of 12 in./sec. Users can select a lower speed if desired. Homing will move the carrier to the end of the actuator you specified and then set the position to absolute zero. Please refer to section 5.9: Cushions and shocks for load weight requirements when homing.

Warning: Speeds greater than 12 in./sec. could occur on a homing routine potentially damaging the actuator. Speeds are based upon air pressure available, Cv of the valve, and servo gains set for the move. A system must be properly tuned to insure 12 in./sec. speed is achieved on a homing routine.

Figure 6.9 Seek Home Command

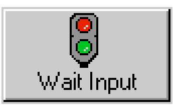
I/O OPTIONS



5. FEED TO SENSOR — Moves the carrier until the given input condition is reached

Allows stopping at any point during a programmed move based on controller input(s). (Figure 6.10) Single or multiple inputs can be selected. Feed to Sensor can be positioned two ways, at the point of deceleration or at the input signal. When positioning at point of deceleration the controller decelerates to a stop after the input condition is met and registers in position. When positioning at the input signal the controller servos to the position where the input condition was seen and registers in position.

Figure 6.10 Feed to Sensor Command



6. WAIT INPUT – Wait for one or multiple inputs

For a single input, choose the input channel and specify its status (circuit open or circuit closed) (Figure 6.11). For multiple inputs, (Figure 6.12) the controller will wait until the exact input status matches the binary value specified by the user.
NOTE: For Binary value a checked box designates circuit closed.

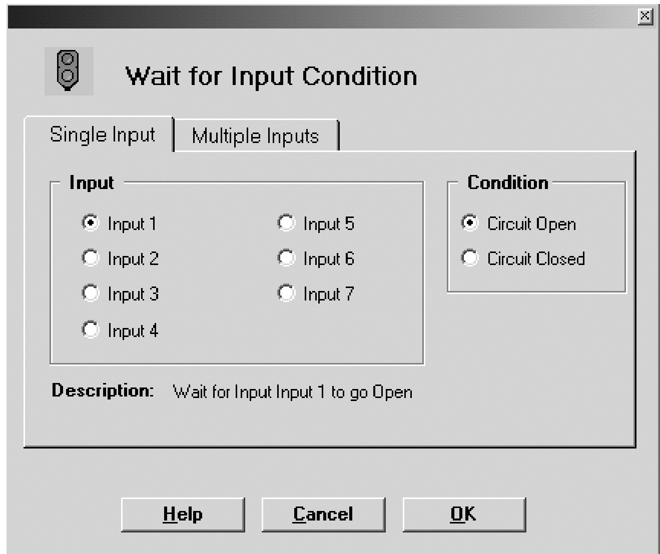


Figure 6.11 Wait for Single Input

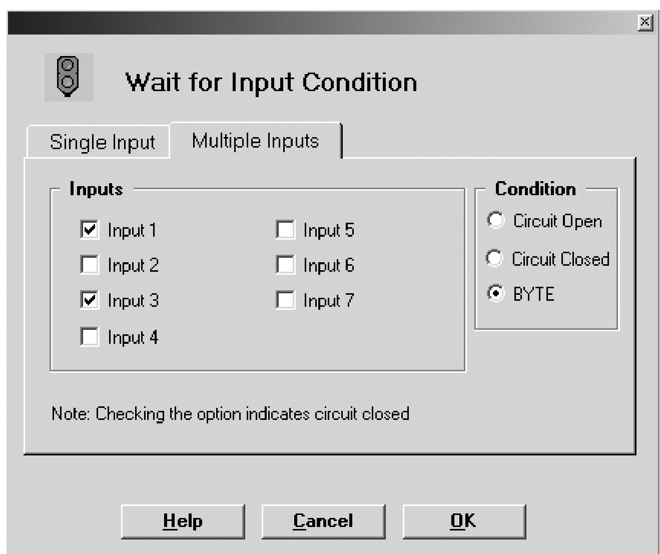
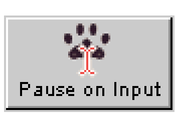


Figure 6.12 Wait for Multiple Inputs



7. PAUSE ON INPUT – Pauses a program at any time

The controller will scan the specified inputs and motion will stop, pausing the motion when input is triggered. Move will resume once input it cleared. (Figure 6.13) Stop time for pause on input is proportional to the deceleration time setting in program. This needs to be set only once at the beginning of the program or where needed. It will stay in effect until turned off.

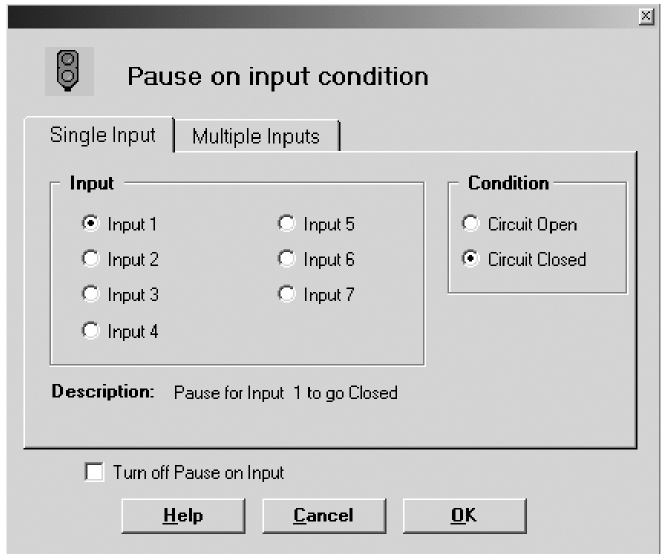
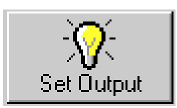


Figure 6.13 Pause on Input Command



8. SET OUTPUT – Set one or multiple outputs

For setting a single output, as shown in Fig. 6.14, select the desired output channel and choose its status (high or low) and click ‘OK’ to continue. For setting multiple outputs, as shown in Fig. 6.15, check the output channels to set high or uncheck them to set low.

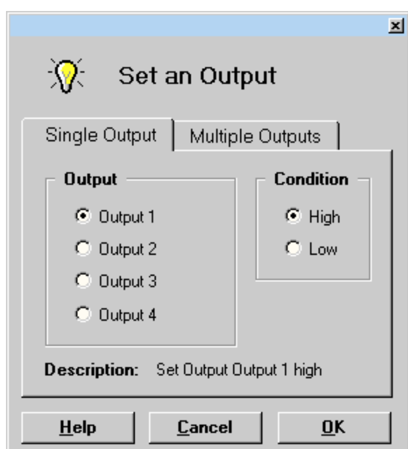


Figure 6.14 Setting a Single Output

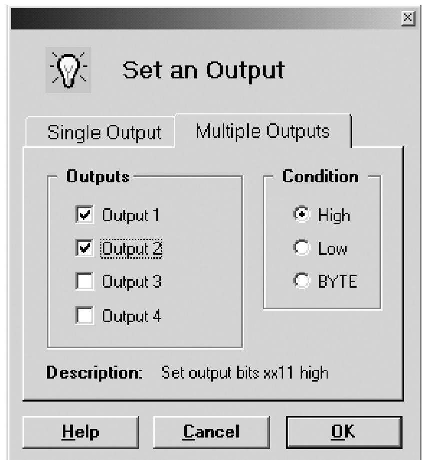
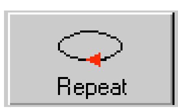


Figure 6.15 Setting Multiple Outputs

LOOPING OPTIONS



9. REPEAT — Repeats a Loop

Specify the number of loops to repeat. Be sure to end the repeat section by the end repeat command. (Figure 6.16) The PrecisionAire controller can allow up to 16 levels deep of nested repeat loop

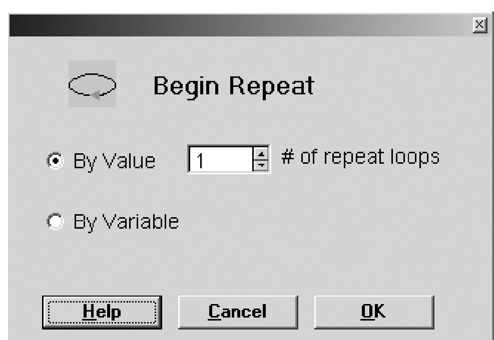
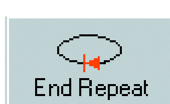


Figure 6.16 Repeat Loop Command



10. End Repeat – end of repeat loop

Controller will branch to the beginning of the repeat loop until specified number of loops has been fulfilled.

11. Go To – branch program to different location

The line number specifies which line the program should branch. A "GoTo line 1" would jump back to the beginning of the program. You can also go to a program number and call a subroutine. When using a subroutine call, the program it calls must end in an end of subroutine command. The controller will allow up to 8-levels deep of nested subroutine calls. For a single input, (Figure 6.17) choose the input and its status Circuit Open or Circuit Closed. For multiple inputs, (Figure 6.18) the controller will branch to the specified location when the exact input status matches the binary value specified by the user. A conditional jump may also be specified when branching is dependent on an input or a variable.

It is not recommended that the GoTo command be used within a repeat or used to jump into a repeat loop unless the user has considerable programming experience. Doing so can cause unexpected carrier motion and/or system faults if not programmed properly.

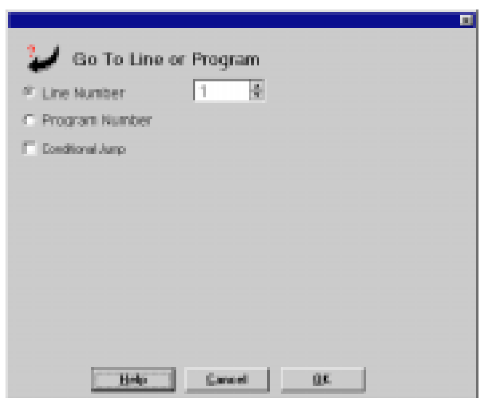
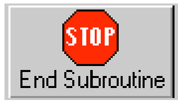


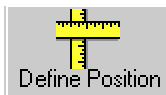
Figure 6.17 Go To Line Number

Figure 6.18 Go To Program Number



12. END SUBROUTINE – Used after a subroutine call to end the subroutine

SETTING OPTIONS



13. DEFINE POSITION — Define current encoder position as specified by user.

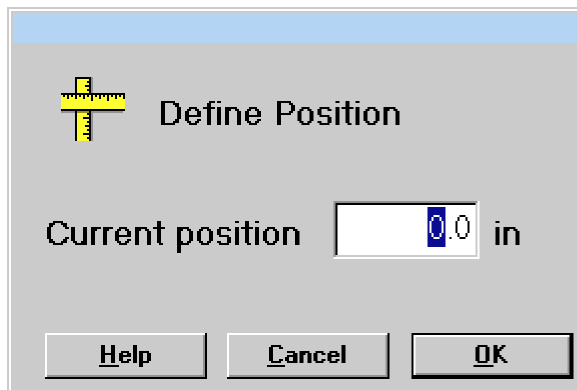


Figure 6.19 Define encoder position command

14. SET VARIABLE

There are 64 variables available for more complex programming needs. Numerical values, encoder position, commanded position, commanded speed, maximum speed and move times can be specified for any variable. (See Figure 6.20) Clicking on a red "Value" button in any of the applicable command options will allow a variable to be set for that function.

An 8 character alpha and/or numeric description can be applied to any variable. To change the name of a variable, select "Tools" from the menu and select "View Variable Names".

It is necessary, when selecting, to enter a numerical value to specify if the value entered is a position or speed. This tells the program to convert the value to encoder counts.

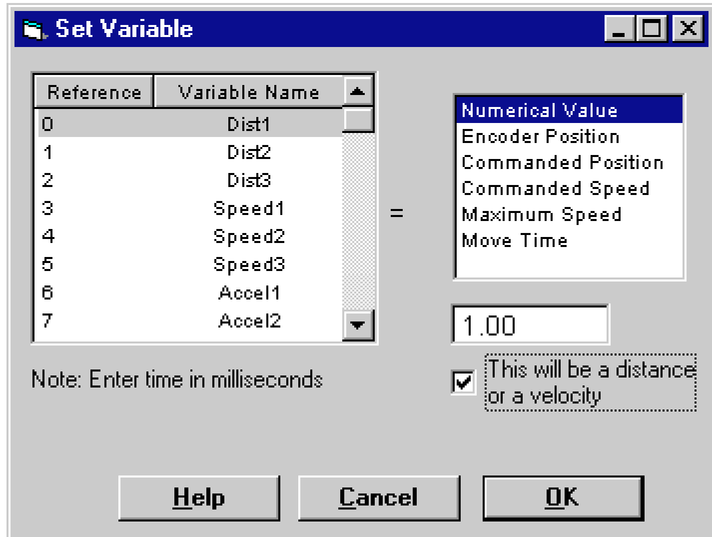
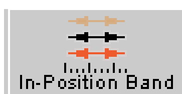


Figure 6.20 Set Variable Command



15. IN-POSITION BAND — Desired positional repeatability

The in-position band can be changed within a program if a different repeatability is required for a specific move. (Figure 6.21)

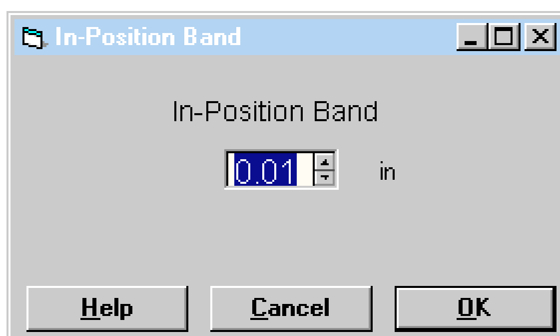
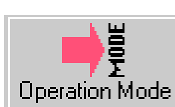


Figure 6.21 In-Position Band Command



16. OPERATION MODE — Selection of servo or thrust mode. In-position holding torque can be changed in thrust mode.

Servo mode is used for positioning. By switching to thrust mode, instead of looking for a fixed position, a direction is selected and the holding torque placed on the brake becomes the amount of thrust applied to the load. A wait time or wait for input is typically used in conjunction with thrust mode, then it is switched back into servo mode in order to complete the next move. (Figure 6.22 and 6.23)

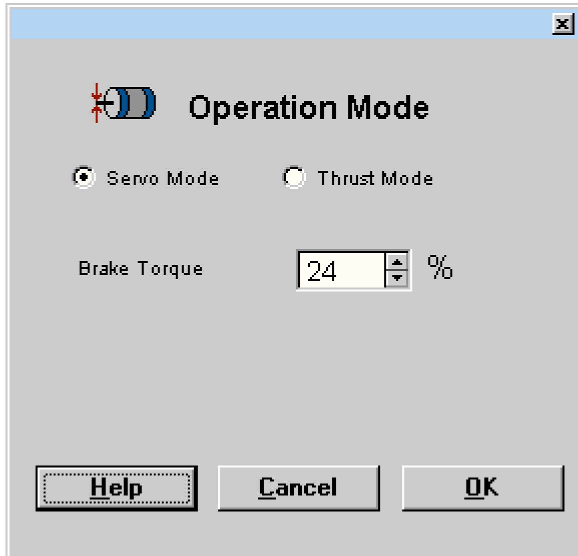


Figure 6.22 Operation Mode Command: Servo Mode

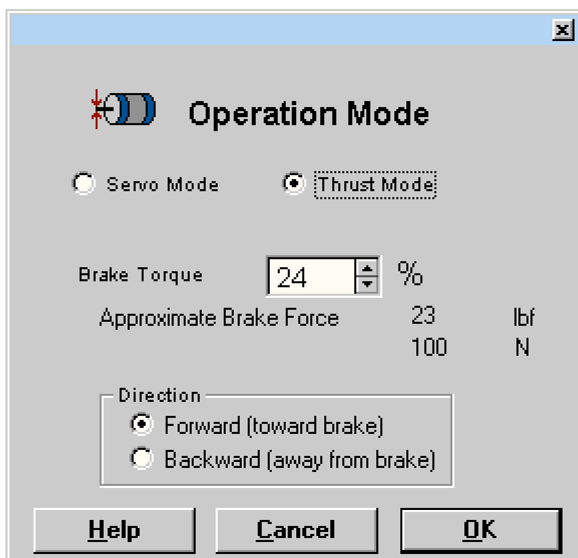


Figure 6.23 Operation Mode Command: Thrust Mode

17. SERVO SETTINGS — Changes the default settings

Default settings made in the set up screen can be changed at any time in the program for individual moves, resulting in optimal carrier performance to minimize carrier over and undershoot. (Figure 6.24)

NOTE: This may be necessary if programming moves of different speeds and/or forces, or if load weight changes.

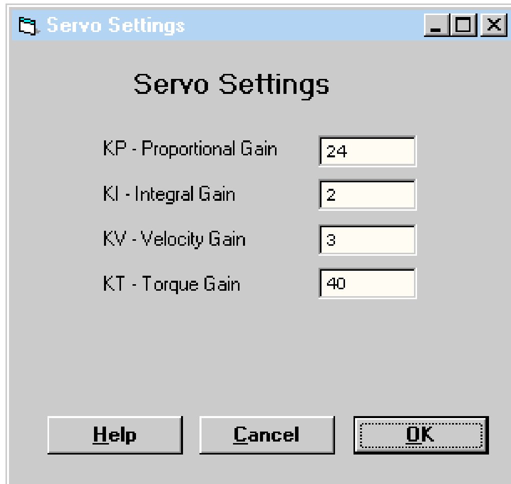
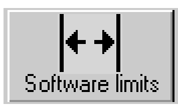


Figure 6.24 Servo Setting Command



18. SOFTWARE LIMITS — Enables or disables previously set software limits

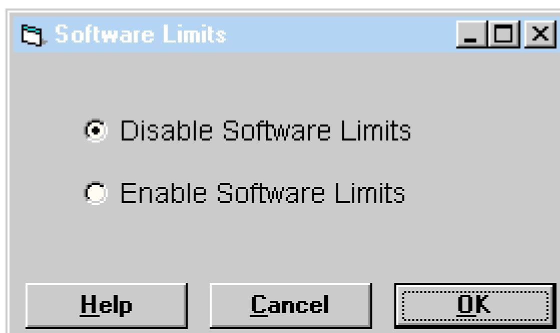
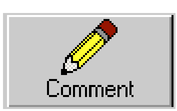


Figure 6.25 Software Limits Command

COMMENT OPTION



20. COMMENT — Insert text information into a program

Note: Blank comment lines will be stored as a blank line with no data.

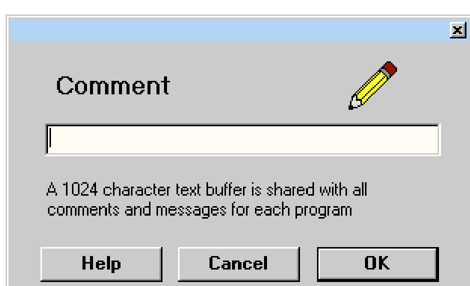
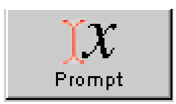


Figure 6.27 Comment Command

C. User interface option

The PrecisionAire controller has the option of a keypad and LCD display panel. (See 6.7 for programming with keypad.) The keypad and LCD panel can be used as an embedded operator interface by using the prompt command in the programming options. This command can be used to prompt a message or variable value at any point in a program for other users to enter job values or operator alerts. Move distances, speeds, acceleration and deceleration rates can be entered or modified. As well as repeat count values, wait times and operator messages.



PROMPT — Prompt a message or variable

If using a PrecisionAire controller with the LCD panel, this command may be used to prompt a message or a variable value at any point in a program for other users to enter job values or operator alert.

Figure 6.26 Prompt Command

D. View/edit variable

Choose 'tools' from the file menu bar and select 'view variable names' to open the variable screen as shown in Fig. 6.28. Users can define variable values or change variable names (up to 8 characters each) by clicking on the desired cell on the spread sheet. Variable values are in encoder counts (if position or speed related), or milliseconds (if time related). If position or speed related enter distance or speed in inches then use 'Convert to Encoder Counts' button to determine counts. To find position or speed enter the counts and use 'Convert to Length' button to display the distance or speed in inches. Variable names and values will automatically be saved in EEPROM after clicking on 'done'. 'Reset variables' button will zero all variable values.

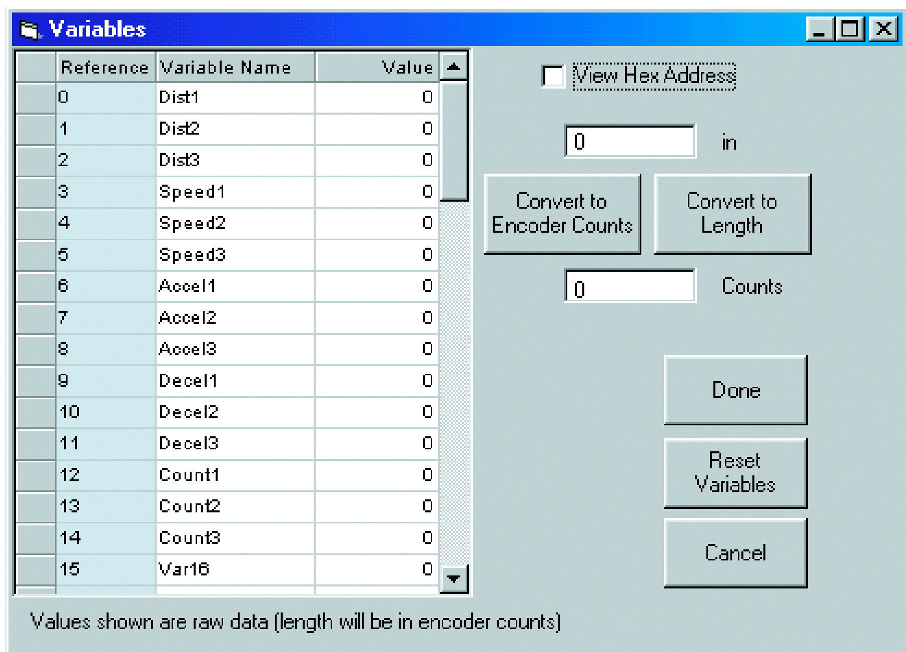


Figure 6.28 Variable Screen

E. Terminal Screen

Choose 'tools' from the file menu bar and select 'terminal windows' to open a terminal as shown in Figure 6.29. Two-letter terminal commands can be entered through the command line, and corresponding reply will be shown in the response text box. This screen is helpful for users who are familiar with 2-letter commands to perform higher level system troubleshooting.

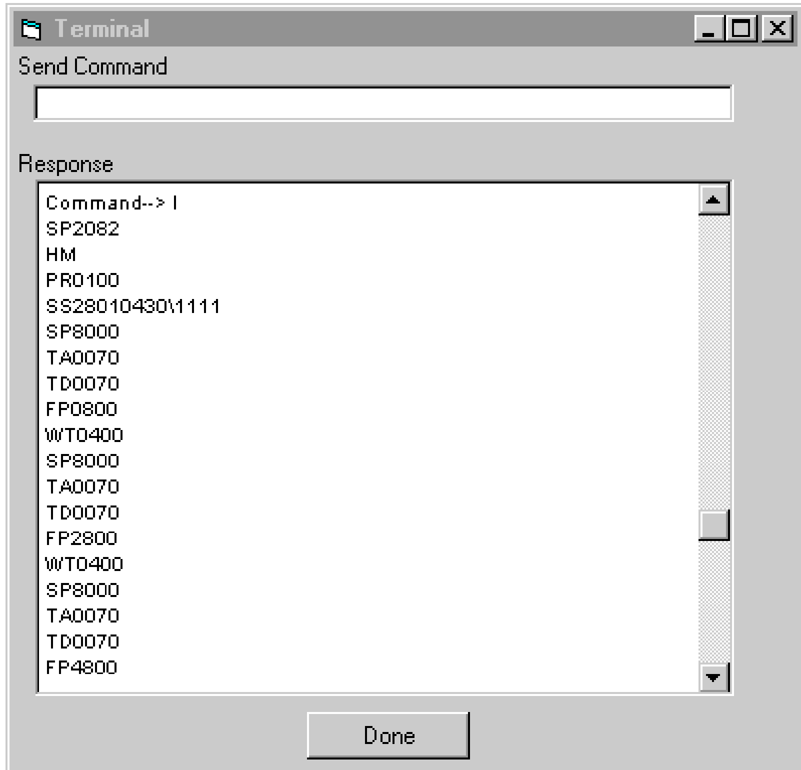


Figure 6.29 Terminal Window

6.7 Setup and Programming from Keypad & LCD

A 3x3 switch array and a 4x20 character LCD display are used for system setup and programming. The left/right arrow key is used to jog the carrier backward/forward, select a motion command, change a decimal place or select the next data entry. The up/down arrow key is used to scroll the menu, select the next data entry, or change a value. Use the 'ENTER' key to confirm a selection and 'ESC' to go back to last menu without saving changes. The command map for the keypad and LCD interface is shown next to the keypad as in Fig. 6.31. Please refer to Chapter 9.2 for more information about the LCD screens.

A. Screen header:

The LCD screen header includes the screen name and helpful function key information that is shown at the first line of LCD display. The screen name is helpful for users when referring to the command map. Available left/right or up/down arrow keys are prompted at the beginning and end of the header.

Screen Header →

```
↓← PROGRAM #00 →↑
000>HOME
001>SET SPEED
002>SET ACCEL. TIME
```

Screen #1

B. Data entry:

Use the up or down arrow key to change the value of data. Use the left or right arrow keys to change a decimal place or the increment or decrement by factors of 10.

For example:

1. To increase data value by 20 at initial increment of 1.
   Press the left arrow key once, then the up arrow key twice to increase the value by 20.

2. To decrease the data value by 0.020 at initial increment of 1.
   Press the right arrow key twice (decrement 0.01) then the down arrow key twice to decrease the value by 0.020.

C. Save:

The PrecisionAire controller/drive will NOT save any changes automatically. After editing, select "SAVE PROGRAM" and "SAVE SETUP PARA." to save the motion program and system settings to the EEPROM.

D. Program Single I/O:

Under the "SET OUTPUT", "WAIT FOR INPUT", or "IF INPUT GOTO" command, select the desired I/O channel (using <, or >) and use down arrow to select the I/O status (HIGH/LOW) for the command. In screen #2, the program will wait until input channel #6 goes low or closed. In screen #3, the program will set output #4 high.

```
↓← PROG#06 L003 ↓←
WAIT UNTIL INPUT 006
      GOES LOW
```

Screen #2

```
↓←PROG#06 L007↓←
  SET OUTPUT 04
     TO HIGH
```

Screen #3

E. Program Multiple I/O:

When programming multiple I/O for the "SET OUTPUT," "WAIT FOR INPUT", or "IF INPUT GOTO" command, the I/O value is considered as **BINARY** and the I/O status options are **BYTE, &_HI, or &_LO.** The least-significant-bit (LSB) of the binary code represents input channel #1 and the most-significant-bit (MSB) represents input #7. For example, the binary code of value 10 is 0001010 ($2^3+2^1=8+2=10$), as shown in table 6.30. I/O channel 2 and 4 are selected. Please refer to Appendix B for a list of binary code up to decimal value 127.

Output channel	N/A	N/A	N/A	4	3	2	1
Input channel	7	6	5	4	3	2	1
Binary bit number	6	5	4	3	2	1	0
Binary code (10)	0	0	0	1	0	1	0

Figure 6.30 Binary code of decimal value10 for multiple I/O programming

- **BYTE** status: In screen #4, the program will wait until input 2 and 4 are HIGH and the other inputs are LOW. The exact I/O status has to be matched for each channel.
- **&_HI** status: In screen #5, the program will set output 2 & 4 HIGH only. Status of output 1 & 3 is not affected.
- **&_LO** status: In screen #6, the program will jump to line #123 if

input 2 & 4 are LOW. Status of other inputs is NOT considered.

Screen #4	Screen #5	Screen #6
↓← PROG#06 L003 ↓← WAIT UNTIL INPUT 010 GOES BYTE	↓← PROG#06 L003 ↓← SET OUTPUT 10 TO &_HI	↓← PROG#06 L003 ↓← IF INPUT 010 &_LO GO TO LINE 123

F. Select Variable Input:

Variable input is available for 'feed to length', 'feed to position', 'wait time', 'repeat', 'set speed', 'accel time', and 'set decel time' commands. To toggle between numerical and variable data entries, press the 'CLEAR' key to zero integer value and press 'DOWN' arrow key.

G. LCD Message & Comment Command:

LCD message prompt and command text can NOT be edited in the keypad interface. Please use PC software to edit message text.

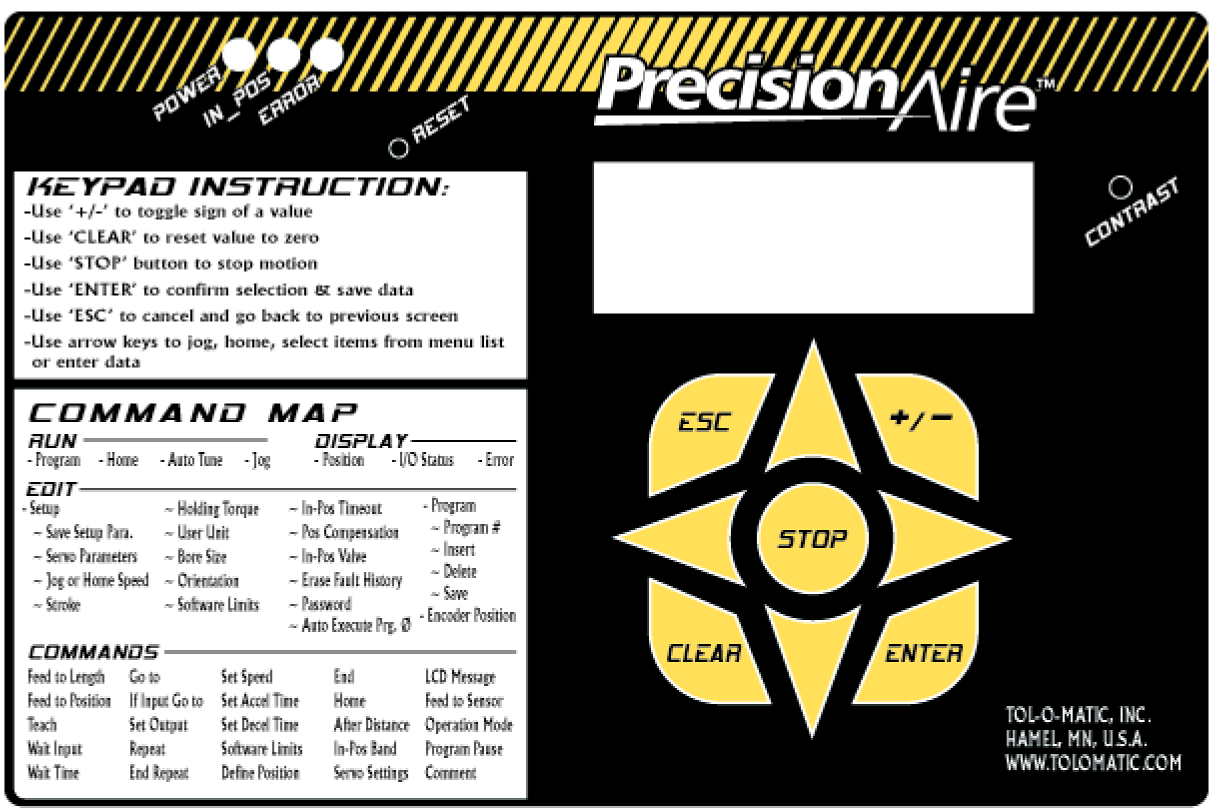


Figure 6.31 PrecisionAire Keypad Interface

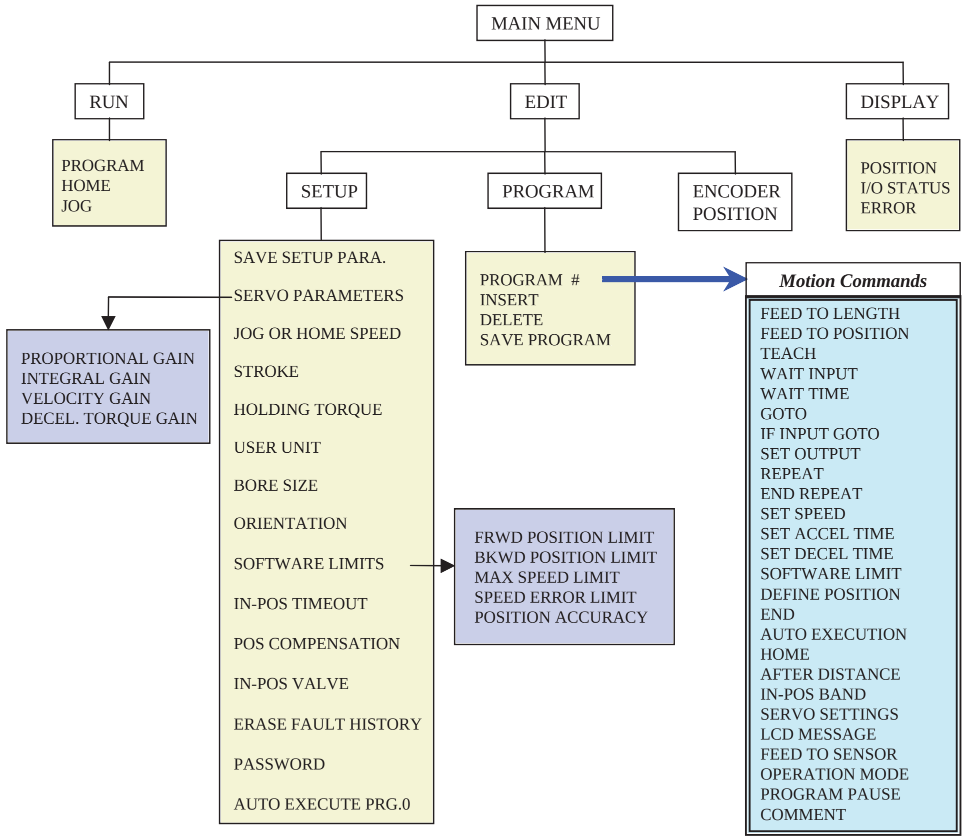


Figure 6.32 Command map of PrecisionAire LCD/keypad interface

6.8 Programming Examples

The following examples are based on an actuator with a stroke length longer than 75 inches using English units with setup parameters specified and entered beforehand. The controller will home backward first and move forward to an absolute position of 70.050-in. Five 10-inch incremental moves will then be made in the backward direction. Set output 4 HIGH for 50 ms then LOW when motion completes.

A. Program an example from keypad

1. Go to ‘EDIT’ then ‘PROGRAM’ menu.
2. Press the right arrow key once to select program #01 and press the ‘ENTER’ key.
3. Press the left arrow key 9 times to display the ‘HOME’ command at line #000 and press ‘ENTER’
4. The home menu is displayed. Leave the default setting to home the carrier away from brake and press ‘ENTER’ to continue.
5. Press the down arrow key once to select line #001 and press the right arrow key 11 times to display ‘SET SPEED’ command and press the ’ENTER’ key.
6. Use the arrow keys to set the speed at 0071 in/s. Press the ‘ENTER’ key. Press up arrow key to increase value & left arrow key to change increment.
7. Press the down arrow key once to select line #002 and press the right arrow key 12 times to display ‘SET ACCEL. TIME’ command and press ‘ENTER’ key.
8. Use the arrow keys to set the acceleration time at 0250 ms and press the ’ENTER’ key.
9. Press the down the arrow key once to select line #003 and press the right arrow key 13 times to display the ‘SET DECEL. TIME’ command and press the ’ENTER’ key.
10. Use the arrow keys to set the speed at 0250 ms and press the ‘ENTER’ key.
11. Press the down arrow key once to select line #004. Press the right arrow key twice to display ‘FEED TO POSITION’ command and press the ’ENTER’ key.
12. The feed to position screen is displayed. Use the arrow keys to set the integer value to +70.
13. Press the right arrow key to switch to the decimal field. Use the up or down arrow key to set the value to +0070.050 and press ‘ENTER’ to confirm the setting.
14. Press the down arrow key once to select line #005 and

press the right arrow key 9 times to display the 'REPEAT' command. Press the 'ENTER' key.

15. The repeat menu is displayed. Press the up key 5 times to set the value to 00005 and press 'ENTER' to confirm the setting.
16. Press the down arrow key once to scroll the program menu down to line #006 and press right arrow key once to display 'FEED TO LENGTH' command' and press 'ENTER' key.
17. The feed to length screen is displayed and use the up arrow key to set the integer value to +0010.000 and press the '+/-' key once to toggle the sign to show – 0010.00 and press 'ENTER' key.
18. Press the down arrow key once to select line #007 and press the right arrow key 10 times to show 'REPEAT END' command. Press 'ENTER'.
19. The end of repeat screen is displayed. Press the 'ENTER' key to go back.
20. Press the down arrow key once to select line #008 and press the right arrow key 8 times to display 'SET OUTPUT' command. Press 'ENTER' command.
21. The set output screen is shown. Press the up arrow key 3 times to select output 4 HIGH and press 'ENTER' to continue.
22. Press the down arrow key once to select line #009 and press the right arrow key 5 times to display 'WAIT TIME' command and press the 'ENTER' command.
23. The wait time menu is displayed. Use the arrow keys to set the time to 0050 ms and press 'ENTER'.
24. Press the down arrow key once to select line #010 and press the right arrow key 8 times to display 'SET OUTPUT' command and press 'ENTER' command.
25. The set output will display last output selected, which is output 4. Press the right arrow key once to select the next data field and press the up arrow key once to set output 4 LOW and 'ENTER' to continue.
26. Press the down arrow key once to select line #011 and press the left arrow key 10 times to display 'END' command and press the 'ENTER' command. Press 'ENTER' again to confirm the selection.
27. Program completed. Be sure to save the program to EEPROM. Press 'ENTER' and then down arrow key 3 times to select 'SAVE' program. Press 'ENTER' to save

program to EEPROM. The program is shown as follows. (Figure 6.33)

```
 |< PROGRAM #01 >|
000>HOME
002 SET SPEED
003 SET ACCEL. TIME
004 SET DECEL. TIME
005 REPEAT
006 FEED TO LENGTH
007 REPEAT END
008 SET OUTPUT
009 WAIT TIME
010 SET OUTPUT
011 END
```

Figure 6.33 Program example using keypad interface

B. Program an example from PC software

1. Click on the programming button to start programming.
2. Click line 1 in the program box to bring up the command screen, click on the 'Seek Home' command icon from the pop-up screen and choose backward (away from brake) in the seek home window and click 'OK'
3. Click line 2 in the program box, click on the 'Move' icon and enter 70" in the distance text box and 70 in/s, 250 ms, 250 ms in the speed, acceleration & deceleration time text boxes respectively. Choose **absolute** move type and click 'OK'
4. Click line 3 in the program box, click on the 'Repeat' command icon and set loop times by value to 5 and click 'OK'
5. Click line 4 in the program box, click on the 'Move' icon and set **incremental backward** move distance to 10 and click 'OK'
6. Click line 5 in the program box, click on the 'End Repeat' command icon.
7. Click line 6 in the program box, click on the 'Set Output' icon and select output #4 to HIGH and click 'OK'
8. Click line 7 in the program box, click on the 'Wait Time' icon and enter 50ms in the wait time by value text box and click 'OK'
9. Click line 8 in the program box, click on the 'Set Output' icon and select output #4 to LOW and click 'OK'
10. Save the program and download it to controller, using the toolbar buttons. The example program is shown as follows. (Figure 6.34)

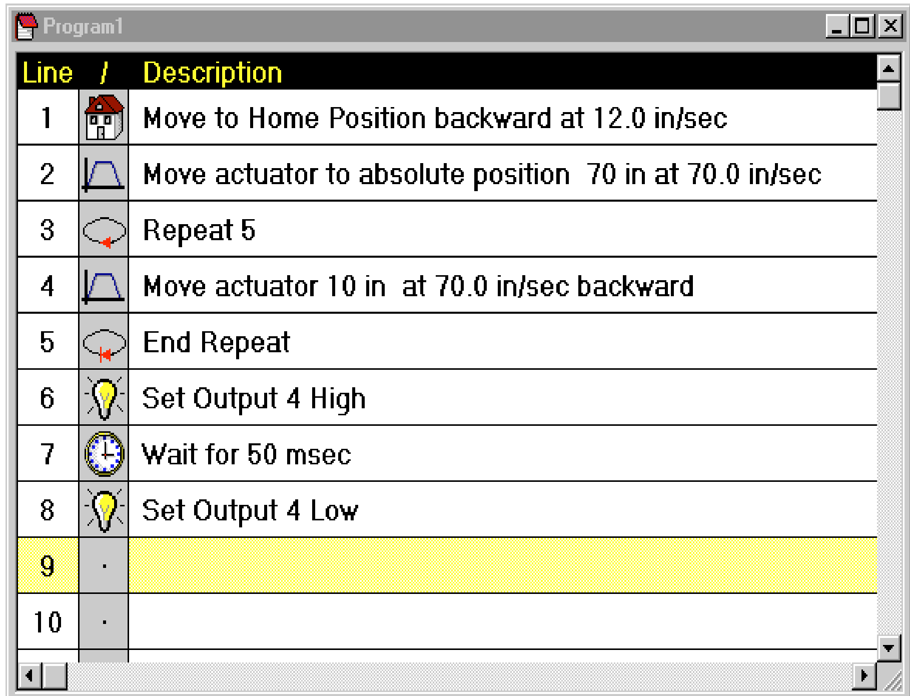


Figure 6.34 Program example using PC software

To run the program click on the run (execute) button on the toolbar.

6.9 Example Applications

APPLICATION 1: PART TRANSFER

Description: A machine builder requires a linear positioning system to transport parts from one station to another using jog pendant.

- Stroke Length: 10 feet
- Guided load about 200 lb.
- Desired speed: 5 in/s (LOW) and 20 in/s (HIGH), user selectable via an input.
- Customer would like to use a 3-position push button for jogging forward, stop, and backward
- Position repeatability < 0.1 in

Tol-O-Matic Solution:

- Use a PAS15SK120 unit with keypad option.
- Use input #7 for program pause as emergency stop.
- Use input #6 for HI/LO speed selection
- Use input #4 & #5 for jogging forward & backward
- Key programming commands: Feed to sensor, variable, goto

Line	/	Description
1		PAS DEMO1
2		Pause for Input 7 Closed
3		Set Servo Gains to KP= 24 KI= 1 KV= 4 KT= 40
4		Move to Home Position backward at 12.0 in/sec
5		Position Repeatibililty 0.099 in
6		Set Variable 8000h to 10
7		If Input 6 Open, Go To Line 9
8		Set Variable 8000h to 20
9		Servo Mode Brake at 25%
10		If Input 4 Open, Go To Line 13
11		Servo Mode Brake at 50%
12		Feed to Sensor Input 4 Open Forward
13		If Input 5 Open, Go To Line 6
14		Servo Mode Brake at 50%
15		Feed to Sensor Input 5 Open Backward
16		Go To Line 6

Figure 6.35 Parts transfer program example

APPLICATION 2: CLAMPING

Description: A packaging company requires a pneumatic positioning vise to approach stacked parts and then provide at least 60 PSI of thrust on stacked items.

- Number of stacked parts varies. Use a proximity sensor to detect presence of stacked parts.
- Need an output signal when compressing process completes
- Vise is normally open at a pre-defined park position before parts are loaded
- Part size ranges from 10-20 in
- Load <= 20 lb
- Position repeatability 0.1 in

Tol-O-Matic Solution:

- Use a PAS10SK24 unit with keypad option.
- Use input #4 to detect stacked parts
- Use thrust mode to contact parts slowly and provide 60 PSI of thrust
- Send output 1 high when done
- Key programming commands: Thrust Mode, feed to sensor, move

Line	/	Description
1		PAS DEMO2
2		Set Servo Gains to KP= 24 KI= 1 KV= 4 KT= 40
3		Move to Home Position backward at 12.0 in/sec
4		Move actuator to absolute position 4 in at 40.0 in
5		Feed to Sensor Input 4 Closed Forward
6		Thrust Mode Brake at 20% Forwards
7		Wait for 2.0 sec
8		Thrust Mode Brake at 0% Forwards
9		Wait for 0.5 sec
10		Set Output 1 High
11		Servo Mode Brake at 25%
12		Go To Line 4

Figure 6.36 Clamping program example

APPLICATION 3: LUMBER CUTTING

Description: A Lumber manufacturer requires a linear positioning system to cut lumbers in different sizes & quantities.

• Need to allow operator to specify lumber cut length and desired number of cuts.

• Need to implement a switch to determine a different cut length or continue current cut size.

• An emergency stop feature is required.

• Prefer control system to continue unfinished job after E-stop.

• Maximum lumber length: 12 feet

• Position repeatability < 0.1 in

Tol-O-Matic Solution:

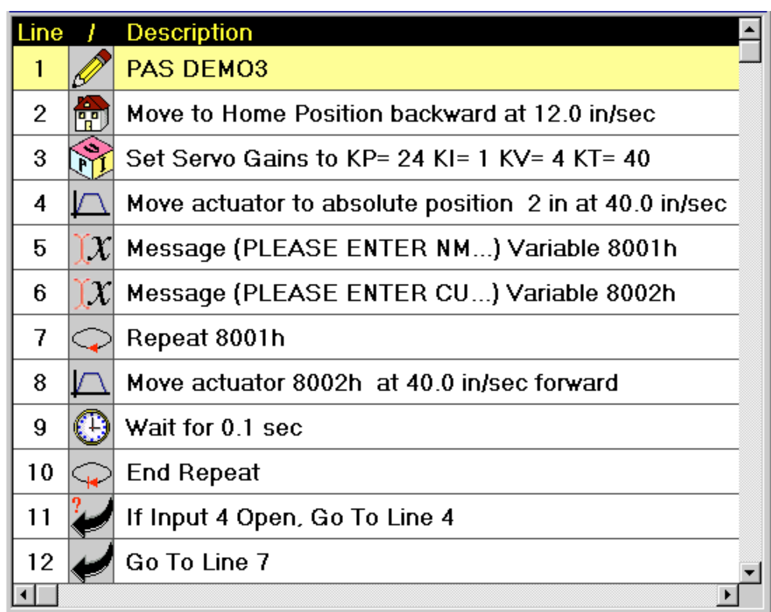
• Use a PAS10SK144 unit with keypad option.

• Use input #4 to determine new cut size and different number of cuts

• Use input #5 for emergency stop

• Use LCD prompt for input to get cut size & number of cuts from operator's entry

• Key programming commands: LCD prompt for input, program pause, repeat loop



Line	/	Description
1		PAS DEMO3
2		Move to Home Position backward at 12.0 in/sec
3		Set Servo Gains to KP= 24 KI= 1 KV= 4 KT= 40
4		Move actuator to absolute position 2 in at 40.0 in/sec
5		Message (PLEASE ENTER NM...) Variable 8001h
6		Message (PLEASE ENTER CU...) Variable 8002h
7		Repeat 8001h
8		Move actuator 8002h at 40.0 in/sec forward
9		Wait for 0.1 sec
10		End Repeat
11		If Input 4 Open, Go To Line 4
12		Go To Line 7

Figure 6.37 Lumber cutting program example

Chapter 7 Tuning

7.1 Data Acquisition

PrecisionAire software has a data acquisition to help properly adjust servo gains, supply air pressure, in-position band, and deceleration/acceleration times for the applications desired speed or positioning performance. It maybe necessary to change or adjust these parameters within a program if loads and speeds change. Note: Air pressure and load changes can affect tuning parameters. See supply air pressure considerations 5.1 prior to tuning.

There are 3 data types that can be selected for data collection (Figure 7.1). The default data collection-sampling rate is set at 16 msec. Click on the "start move & data acquisition" button after specifying absolute move position and speed to start data collection.

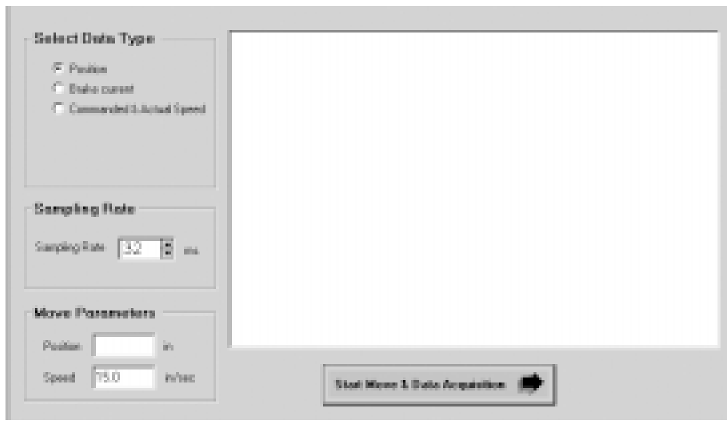


Figure 7.1 Start Move and Data Acquisition

<u>A. Position</u>.

The position data type is used to view over or under shoot of the system. The main factors that affect the positioning of the PrecisionAire system are: supply air pressure, in-position band, deceleration time, and servo gains (Kv and Kt the most critical).

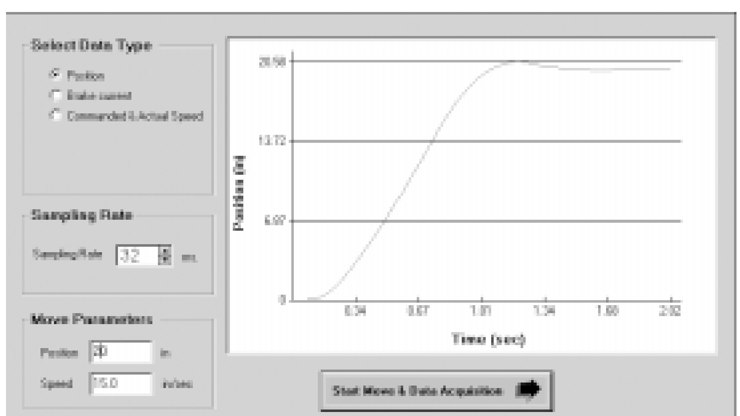


Figure 7.2 Position Data

B. Brake Current.

The brake current is used to monitor the performance of the brake. In high cycle, speed, or load applications were brake temperature could be a factor, the brake current data can be viewed to monitor brake usage.

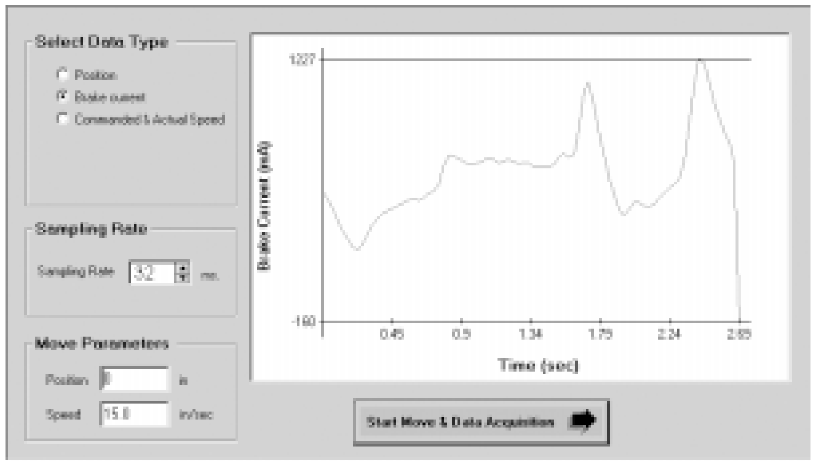


Figure 7.3 Brake Current Data

C. Commanded & Actual Speed.

The command versus actual speed data type is used to monitor the acceleration, velocity, and deceleration at a given air pressure, load, commanded speed, and servo settings (Kv and Kt the most critical). This can be used to determine appropriate supply air pressure for desired speed, the appropriate servo gains for a determined air pressure and load, and the cycle time of a move. It is also helpful in adjustments of the servo settings or deceleration rate for different loads or speeds within the same program.

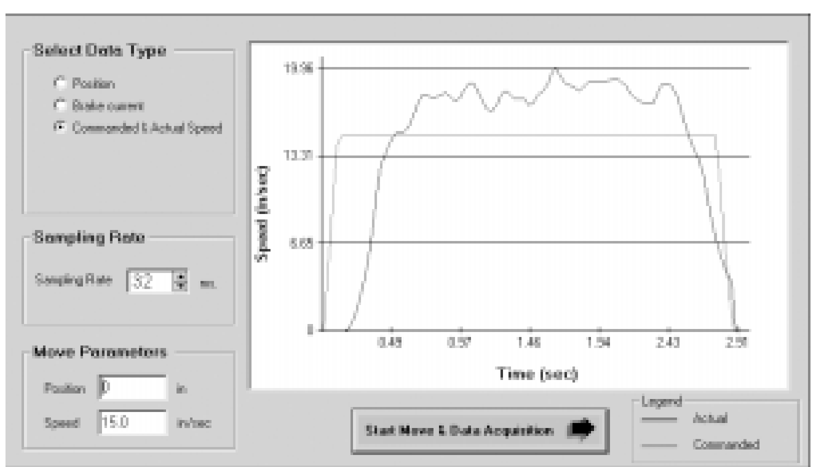


Figure 7.4 Command and Actual Speed Data

7.2 Tuning

To achieve the desired motion profile or positioning required for a move profile, system tuning of the PrecisionAire might be necessary. System tuning can involve changing the supply air pressure, the deceleration/acceleration move rates, the in-position bandwidth, and the servo gains. Tuning within a program could be required for load or speed changes. Tuning may also be required for positioning moves at different locations along the actuator based upon the direction and the proximity of the ends due to the compression characteristics of air.

To optimize tuning, supply air pressure should be set for the desired acceleration or speed of the highest speed move within a program. Typically this air pressure will be no more then 10 PSI above what is required for the desired acceleration or velocity. Operating at higher air pressures can increase the likelihood of overshoot/undershoot or system hesitations. The in-position band should also be set at the application maximum. The lower the in-position band is set the greater likelihood the system will experience overshoot or excessive servoing into position. Once the air pressure and in position band are set, the deceleration rates and servo gains can be used to adjust the tuning to the desired motion profile or system positioning.

There are four servo parameters in the PrecisionAire system that users can manually adjust for better performance. Servo gains can be adjusted in the Setup Parameters or within the program

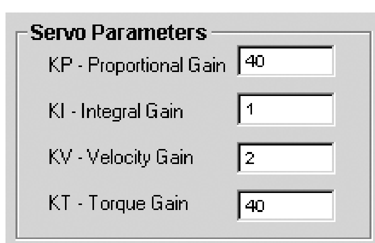


Fig 7.5a Parameters

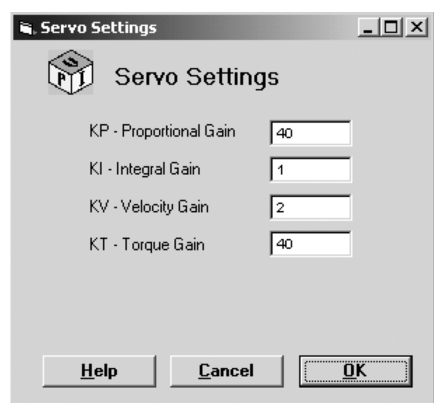


Fig 7.5b Adjustment within program

Servo gains will be used from the Setup Parameter unless changed within a program prior to the move. It may be necessary to use the data acquisition to properly adjust these parameters for the desired system speed or positioning performance. Refer to Chapter

9.1 for a control algorithm block diagram.

NOTE: All servo settings are in integer format.

A. Proportional Gain (KP):

The KP gain is the position error gain that will determine how sensitive the controller will respond to the position error. Only change when making moves smaller than 0.5 inches. The controller will respond to position error more effectively when increasing the KP gain. However, setting the KP gain too high will lead the system toward positioning instability. The default KP gain is set to 40 at the factory. For most applications, no adjustment should be necessary.

B. Integral gain (KI):

The KI gain is the position error integral gain. The PrecisionAire controller will accumulate position error while the carrier is approaching the target position and multiply the error by the integral gain. The result is used to determine the required brake current for positioning. The default KI gain is set to 1 at the factory. Increase when making small moves of 0.5 inches or less.

C. Speed gain (KV):

The KV gain is the speed error gain. The KV gain is the controller's response to the commanded velocity. Increase when the actual velocity needs to be as close to the commanded velocity as possible. However, jerky motion will occur if setting the KV gain too high. The default KV gain is set to 3 at the factory.

Note: See air system considerations 5.1 for velocity control.

D. Deceleration current constant (KT):

The KT gain is the brake response to acceleration. The KT gain is used to set the minimum brake current while decelerating the load. It is used to adjust position overshoot or undershoot of a system. The KT gain is used in conjunction with the deceleration rate. Increasing the KT gain or the deceleration rate will reduce overshoot. Decreasing the KT gain or the deceleration rate will reduce undershoot. Some applications will require adjustment in both the KT gain and deceleration rate for individual move profiles. The default KT constant is set to 40 at the factory.

Note: Decreasing air pressure, increasing in-position band, or increasing deceleration time can all reduce overshoot.

E. Tuning Tips:

a. Overshoot –Decrease air pressure, increase in-position band, increase KT, or increase decel time.
b. Undershoot – Decrease air pressure, decrease KT, or decrease deceleration time.
c. Not reaching desired speed -Decrease KV
d. Moving faster than desired speed – Decrease air pressure or increase KV
e. Does not move after issue move command – increase KP
f. Jerky motion – Decrease air pressure or decrease KV
g. Not reaching desired cycle time – Increase in-position band.

7.3 Tuning Examples

A. Setting proper air pressure.

This example uses a PAS15 1.5 inch bore unit, 62 inch stroke actuator, a load weight of 60 lbs., and two 2 position 3 way valves. The desired move is 40 inches at 20 inches/second with an acceleration rate of 0.25 seconds.

- In the Setup window, Set the acceleration rate to 0.25 seconds and the deceleration rate to 0.15 seconds. Also set the Servo parameters to KP 40, KI 1, KV 1, KT 45. (This example uses a large load weight of 60 lbs, for lighter loads the deceleration rate and KT gain can be lowered. The large load weight also affects the acceleration rate achievable.)
- Set the in-position band to a high value. This example uses 0.5 inches.
- Set the air pressure to a low pressure (15-25 PSI).
- Use the Commanded & Actual Speed in the Data Acquisition to see the speed achievable with the current air pressure. See Fig 7.6a, this data file is with 15 PSI air pressure. Notice the speed can be achieved but the acceleration rate is not met.

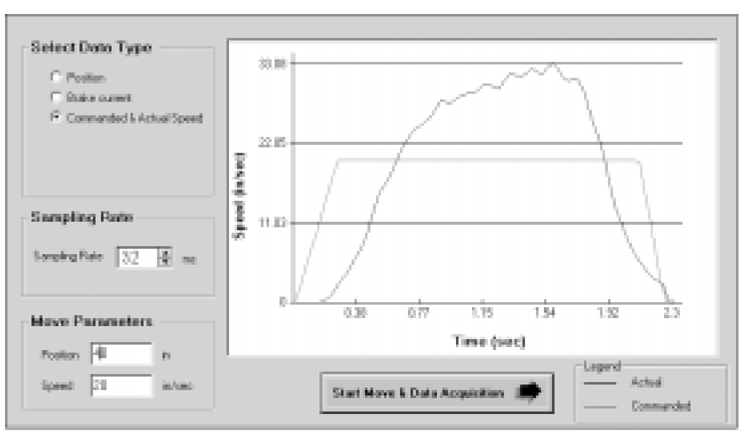


Figure 7.6a

- Increase the air pressure if need be and take another Data Acquisition file. See Fig 7.6b, this data file is with 40 PSI air pressure. Notice the high acceleration rates and speeds, this pressure is too high to optimize the performance of the system.

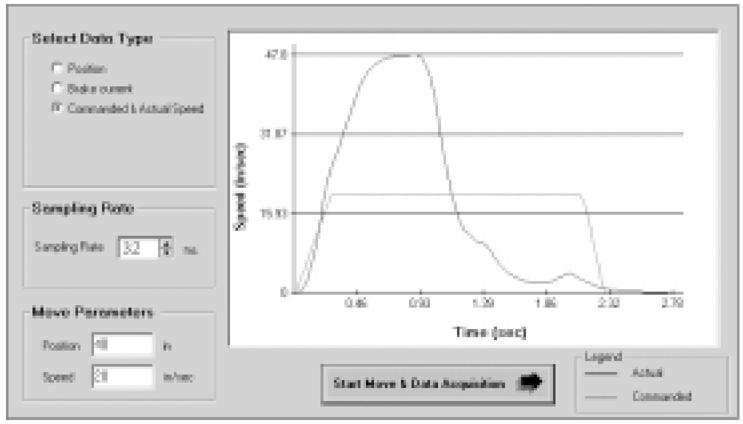


Figure 7.6b

- See figure 7.6c for the proper air pressure (25 PSI) for this example. Notice the speed is higher then desired, but the acceleration rate is met.

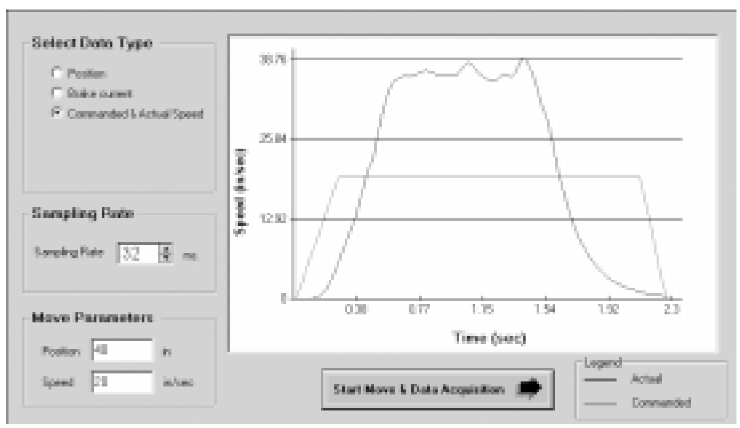


Figure 7.6c

- **Air pressure alternate method:** Remove all power. Manually shift the valves back and forth, adjusting air pressure until speed or force desired is reached. **Warning:** Do not slam carrier into the ends. Damage to actuator will occur.

B. Velocity control

This example uses a PAS15 1.5 inch bore unit, 62 inch stroke actuator, a load weight of 60 lbs., and two 2 position 3 way valves. The desired move is 40 inches at 20 inches/second with an acceleration rate of 0.25 seconds.

- Once the proper air pressure is selected (See setting proper air pressure example), increase the Kv gain and use the Command & Actual Speed in the Data Acquisition to adjust the velocity control. In this example the settings are the same as figure 7.6c, 25 PSI, KP40, KI 1, KV 1, KT 45. See figure 7.6d, KV is adjusted to 4.

Note: PrecisionAire systems are not intended for velocity control. With proper valving, air pressure and tuning it is possible to achieve ± 10% velocity control of a commanded speed. However, due to the characteristics of air, a small change in air flow or pressure can effect the consistency of velocity control. If velocity control is needed a servo system should be used.

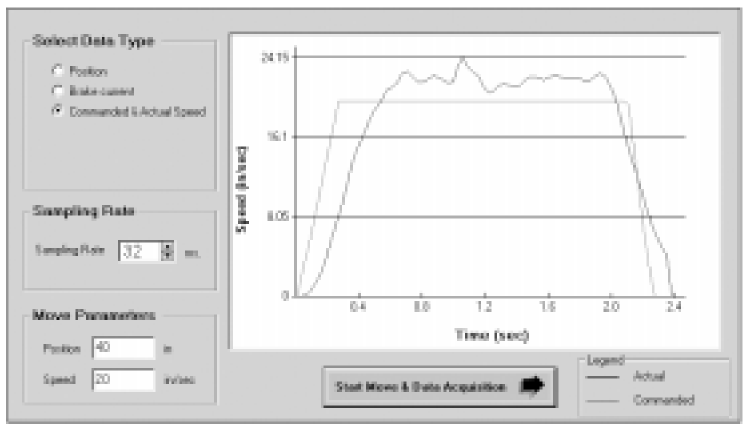


Figure 7.6d

C. Positioning control.

This example uses a PAS15 1.5 inch bore unit, 62 inch stroke actuator, a load weight of 60 lbs., and two 2 position 3 way valves. The desired move is 40 inches at 40 inches/second with an acceleration rate of 0.5 seconds. The positional accuracy is 0.01 inches.

- Once the proper air pressure is selected (See setting proper air pressure example), adjust the deceleration time, KT and KV gains and use the Position data type in the Data Acquisition to adjust the position for over or undershoot.

Note: It may be necessary to adjust these gains once the program is running. Positioning moves at different locations along the actuator based upon the direction of motion and the proximity of the ends can have different servo parameters and deceleration rates due to the compression characteristics of air.

- In this example the air pressure is set at 35 PSI. See Figure 7.7a, the servo gains are set at KP 40, KI 1, KV 2, KT 40. The deceleration rate is set at 0.1 seconds. Notice the actuator overshot 1.81 inches and came back into position.

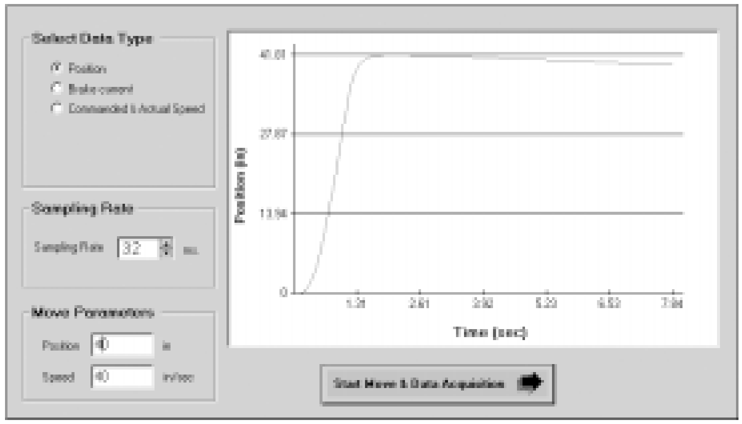


Figure 7.7a

- See figure 7.7b. The servo gains are now adjusted to KP 40, KI 1, KV 3, KT 42. The deceleration rate is also changed to 0.15 seconds. Notice the actuator undershot and moved slowly into position.

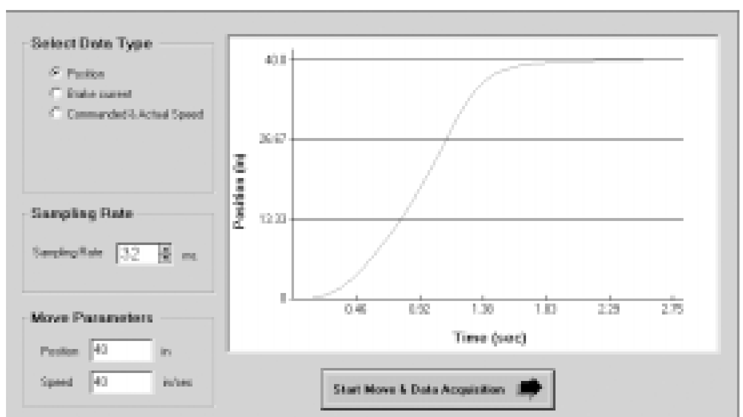


Figure 7.7b

- To decrease cycle time see figure 7.7c. The servo gains and deceleration time are the same as figure 7.7b, however, the in-position band is increased to 0.1 inches.

C. Cycle time control.

Note: Achieving a high positional accuracy and the fastest cycle time is the most difficult application to achieve with the PrecisionAire system. A consistent fast cycle time may not be possible to achieve without increasing the in-position band. Achieving a high positional accuracy can require 1-2 added seconds to servoing into position at the end of a move.

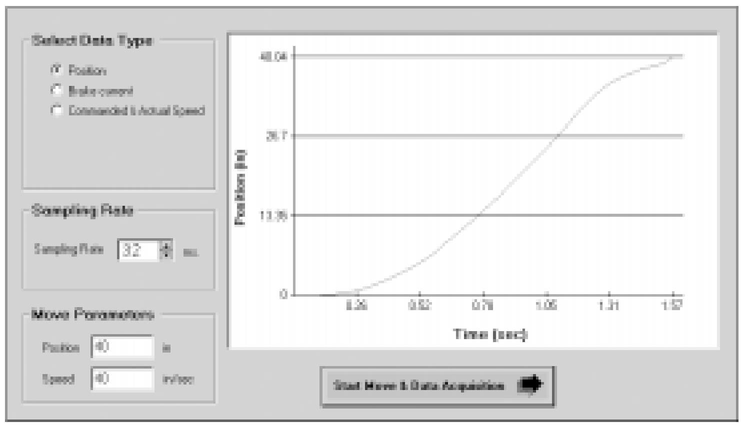


Figure 7.7c

Chapter 8 Troubleshooting

NOTE: A voltmeter is required for system troubleshooting.

8.1 Display & Diagnostics

PrecisionAire software has a display & diagnostics to help with troubleshooting. It displays the I/O status of the controller and also tests the valve(s) and brake. When testing the valves and brake, we recommend users set jog speed slower for safety, and to prevent damage to the system. Jog speeds use the default setup parameters. Display & diagnostics also allows the user to view the fault history and reset the controller.

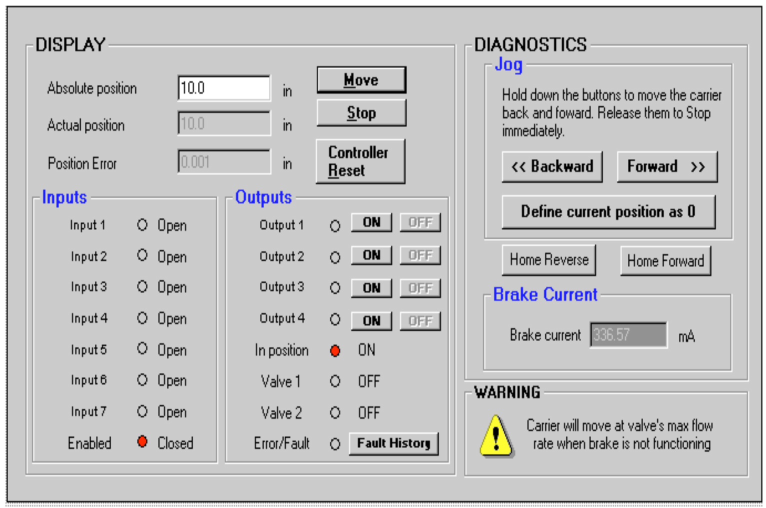


Figure 8.1 Display & Diagnostics Screen

When a fault or error occurs, shut off the air supply, observe the system status, and follow the corresponding troubleshooting path. Do not push or pull on the carrier when the servo loop is activated. Damage or injury may occur if a user does not follow the trouble shooting procedure correctly.

8.2 Fault (Fault LED is ON):

When a fault or error occurs, shut off the air supply, observe the system status, and follow the corresponding troubleshooting path. Do not push or pull on the carrier when the servo loop is activated. Damage or injury may occur if a user does not follow the trouble shooting procedure correctly.

<u>A. Use PC software through RS232:</u>
Go to the 'display & diagnostics' window and click on the fault history button to display the last 16 fault messages. The most

recent fault is stack on top of the list and the oldest message is at the bottom of the list.

B. Use keypad & LCD interface:

Go to 'ERROR' menu under 'DISPLAY' to display the fault history. The controller EEPROM will hold up to 15 most recent fault messages. A tabulated fault message and action are listed below.

Fault codes 01 and 02 are considered as system failures. Power must be recycled to reset the controller. Fault codes 03 thru 0A can be reset through the enable input.

Fault message (code)	Description	Action
ADC conversion fault (01)	DSP ADC voltage reference LOW.	Contact Tol-O-Matic
Brake current fault (02)	Exceed brake max current	1. Check brake connection 2. Measure brake resistance 1.0 in brake - 7.2 to 8 ohm 1.5 in brake - 3.6 to 4 ohm 3. A jumper should be installed on controller board J5 for 1.5 in brake
Position fault (03)	Unable to position	1. Increase encoder monitoring 2. Decrease integral gain Ki 3. Increase brake torque decel gain 4. Increase position window 5. Check brake in-line fuse and coil resistance
Reverse position limit (04)	Exceed reverse position limit	Modify reverse position limit setting
Forward position limit (05)	Exceed forward position limit	Modify forward position limit setting
Maximum speed fault (06)	Exceed maximum speed setting	1. Check commanded speed 2. Increase max speed setting
Speed following error (07)	Exceed speed following error setting	1. Increase speed gain Kv 2. Increase speed error setting
Enable fault (08)	Controller NOT enabled	1. Check enable input wiring 2. See Chapter 5.7 input wiring
Move profile timeout (09)	(Actual move time - commanded move time) >= timeout setting	1. Check and adjust tuning para. KV, KI, or KT 2. Increase profile timeout setting
Current following fault (0A)	Brake current following error	1. Check brake wire connection 2. Check brake coil resistance

Figure 8.2 List of controller fault message, description and action plan

8.3 Power LED is OFF after Power ON:

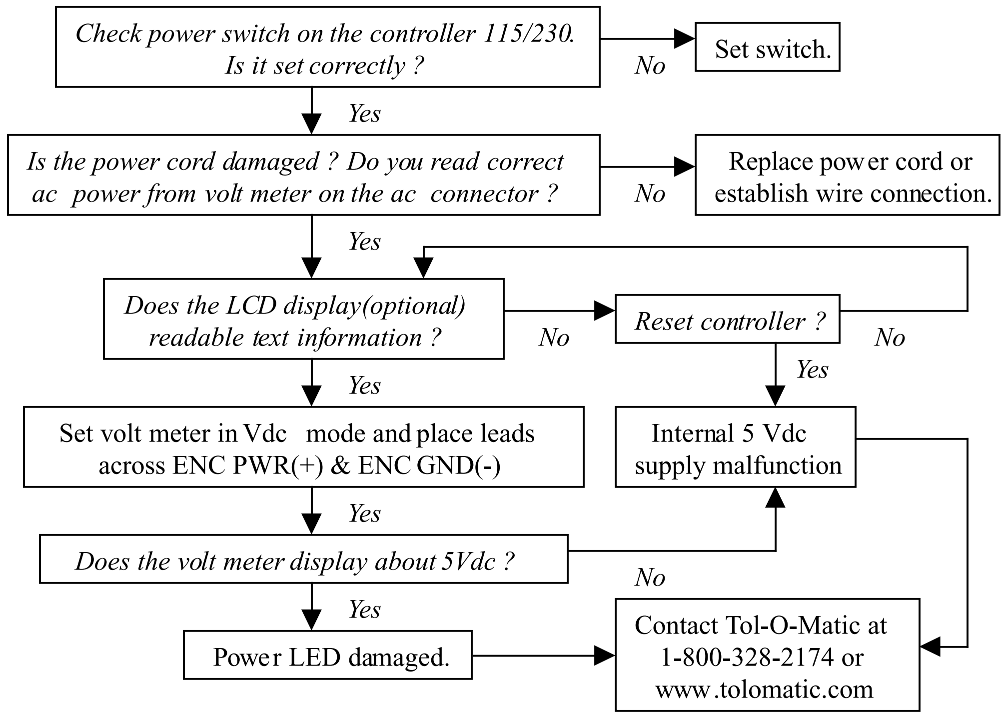


Figure 8.3 Flow chart for trouble shooting with power LED off fault

8.4 No Communication with Controller:

A. Refer to Section 8.1 if LED does not light.
B. Check RS232 connections.
C. Check to insure proper COMM port is selected.
D. Check ONLINE Status.

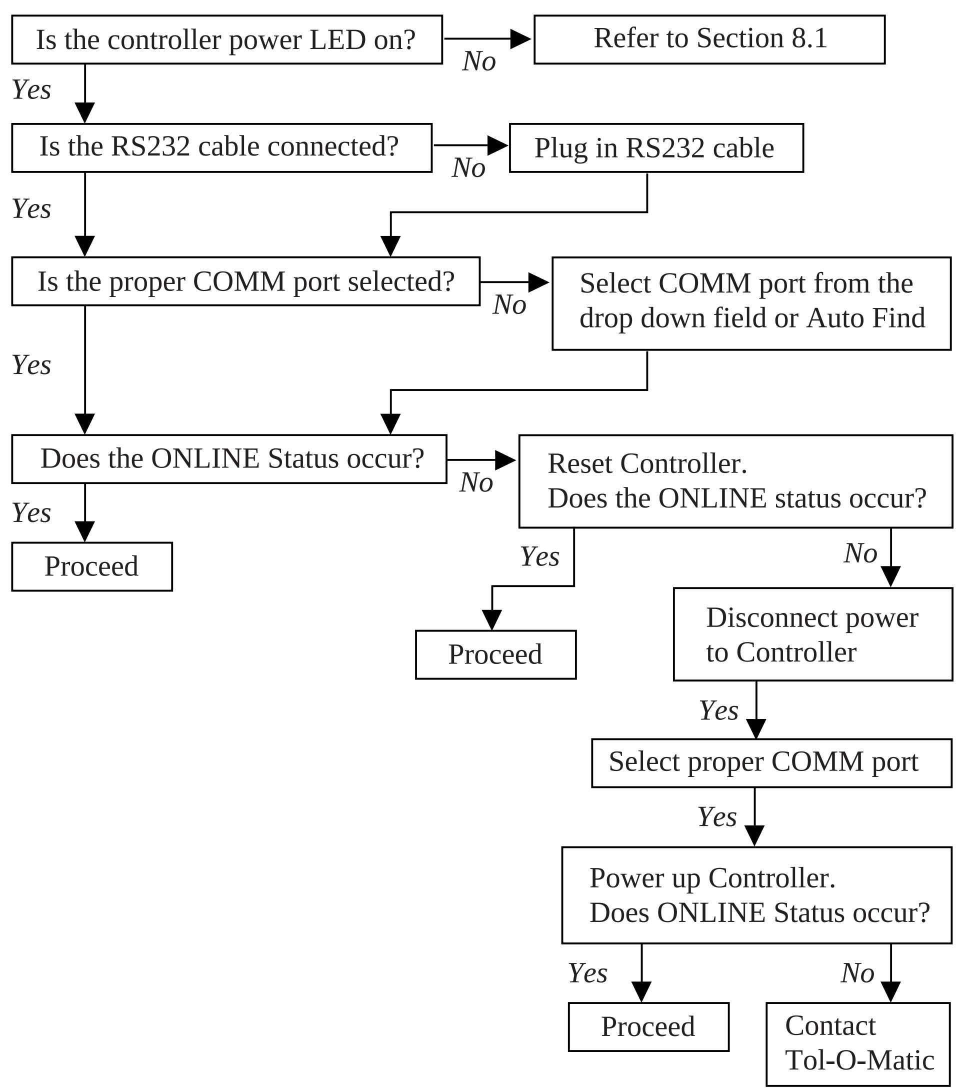


Figure 8.4 Flow chart for checking Controller communications

8.5 No Motion Occurred:

A. Check 'ENABLE' input.
Controller needs to be enabled to perform any motion. Please refer to Chapter 5.5, input connection under hardware setup in this manual for input wiring.

B. Check air pressure and hose connection.

C. Check in-position band setting

D. Check valve solenoid connection.
D.1 Use keypad & LCD interface

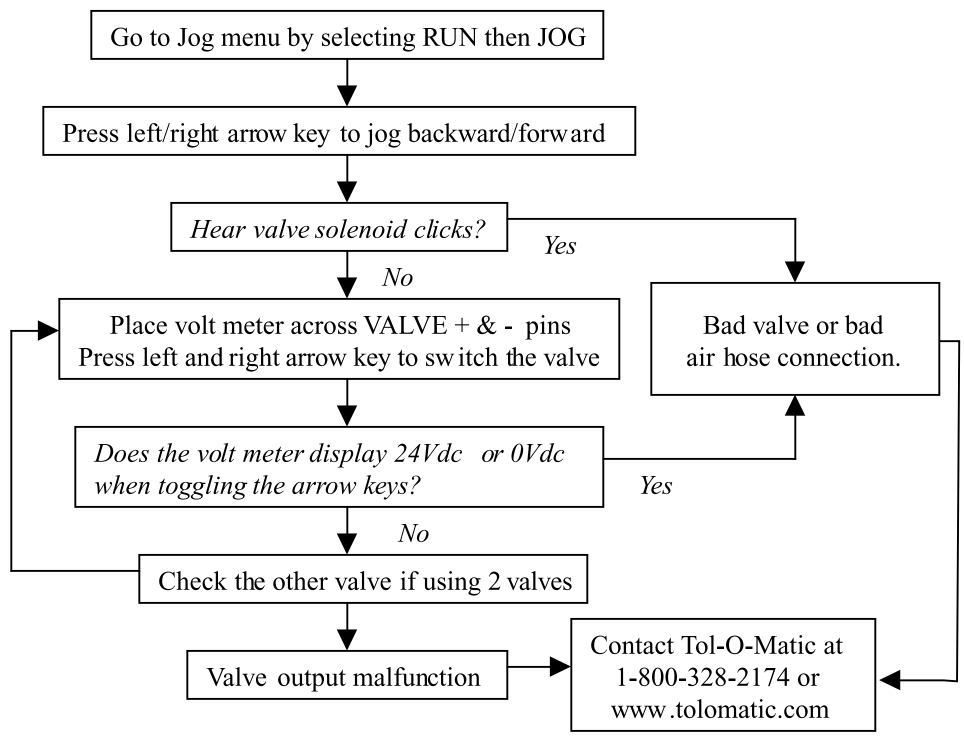


Figure 8.5 Flow chart for checking valve connection using keypad interface

D.2 Use PC software through RS232

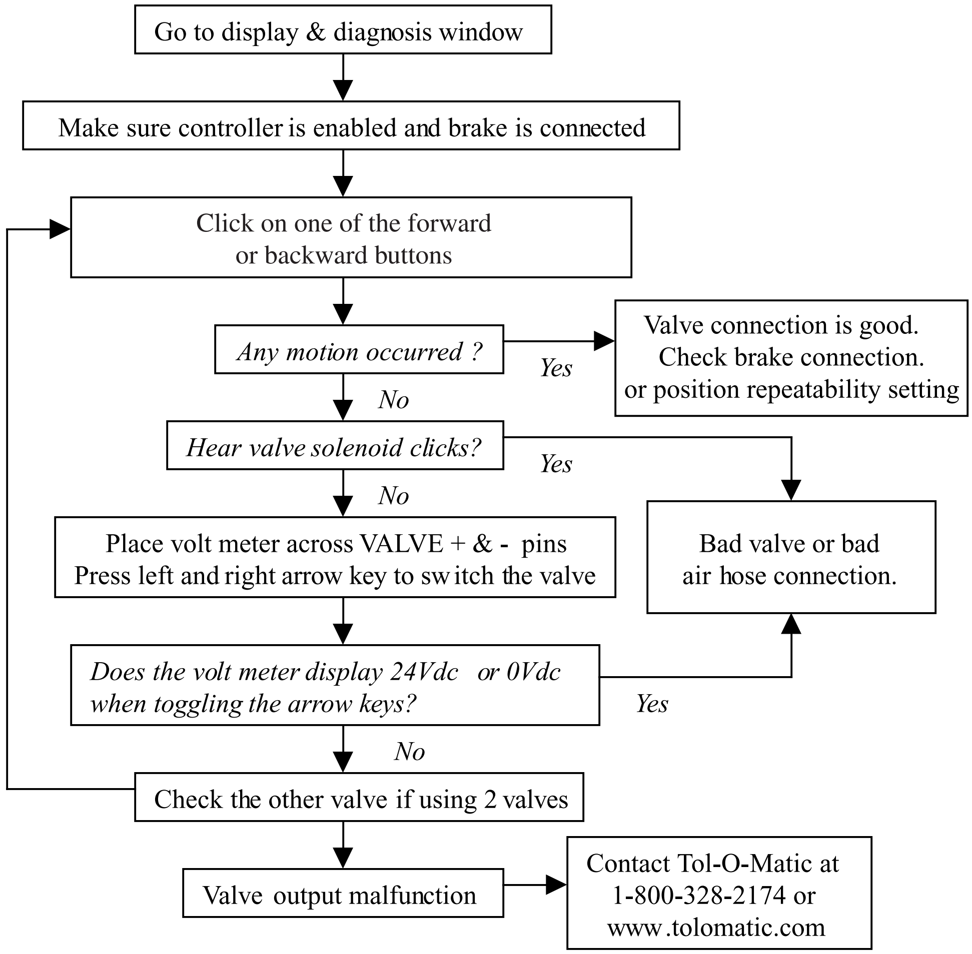


Figure 8.6 Flow chart for checking valve connection using PC software

E. Check belt connection

E.1 Disconnect air to the cylinder

E.2 Disconnect brake

E.3 Manually move the carrier and watch the brake shaft turns. If the brake shaft does not rotate then the timing belt or the belt clamp inside the actuator may be broken, or belt may be excessively loose.

F Check encoder reading

F.1 Use keypad & LCD interface

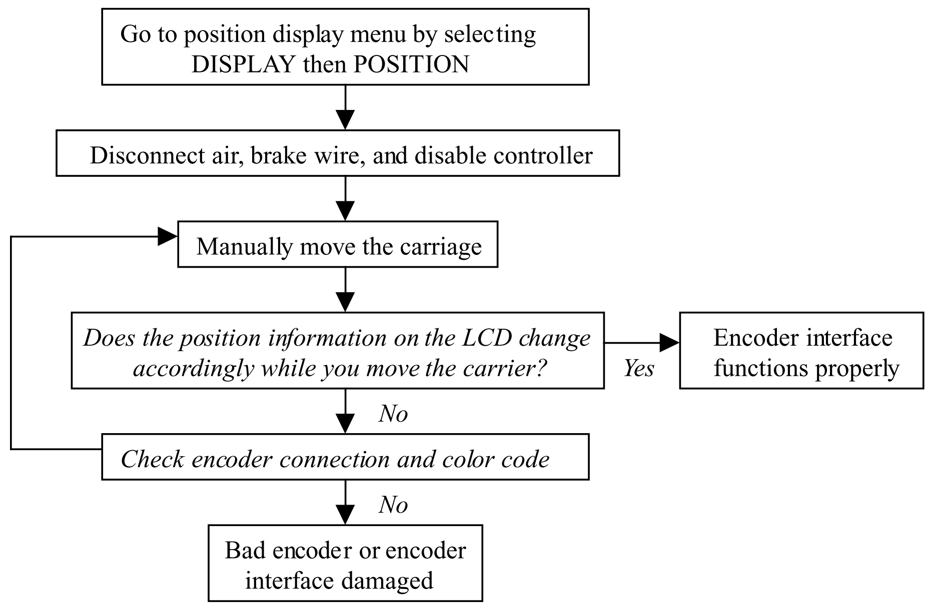


Figure 8.7 Flow chart for checking encoder connection using keypad interface

F.2 Use PC software through RS232

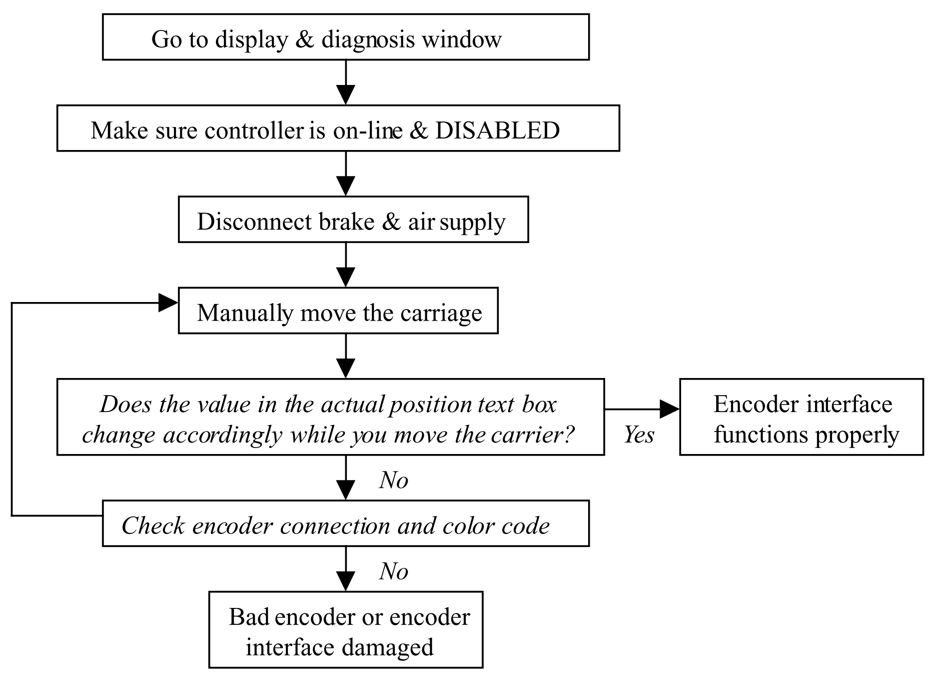


Figure 8.8 Flow chart for checking encoder connection using PC software

8.6 No LCD Display:

A. Check power switch — 230 Vac or 115 Vac
B. Turn contrast pot C.W. up to 25 turns
C. Bad LCD module or connection, contact Tol-O-Matic

8.7 Carriage Runaway:

User can jog carrier but it runs away when executing a program or making a single move. It is very likely that the valve(s) connections are backward. Swap valve forward and backward wires, or swap the airline connections at the ends of the cylinder. The brake could be damaged. Test the brake coil resistance. Resistance should be about 8 ohms for a PAS10 and about 4 ohms for a PAS15. Contact Tol-O-Matic if the brake is damaged.

Chapter 9 Technical Information

9.1 Control-block Diagram

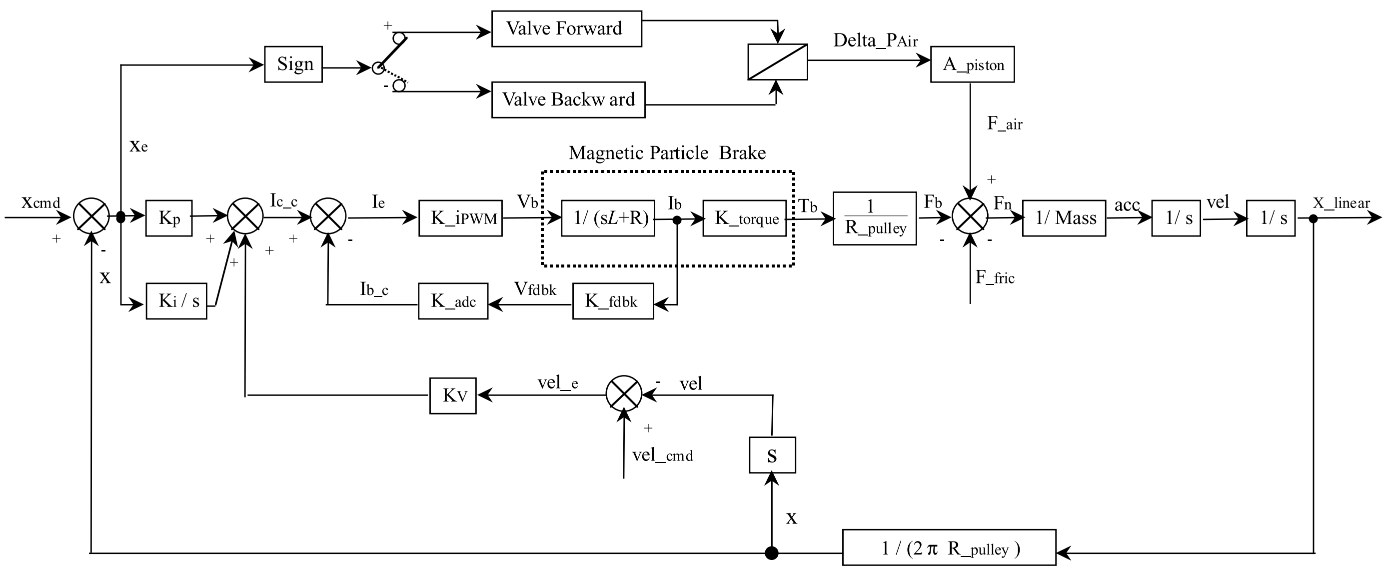
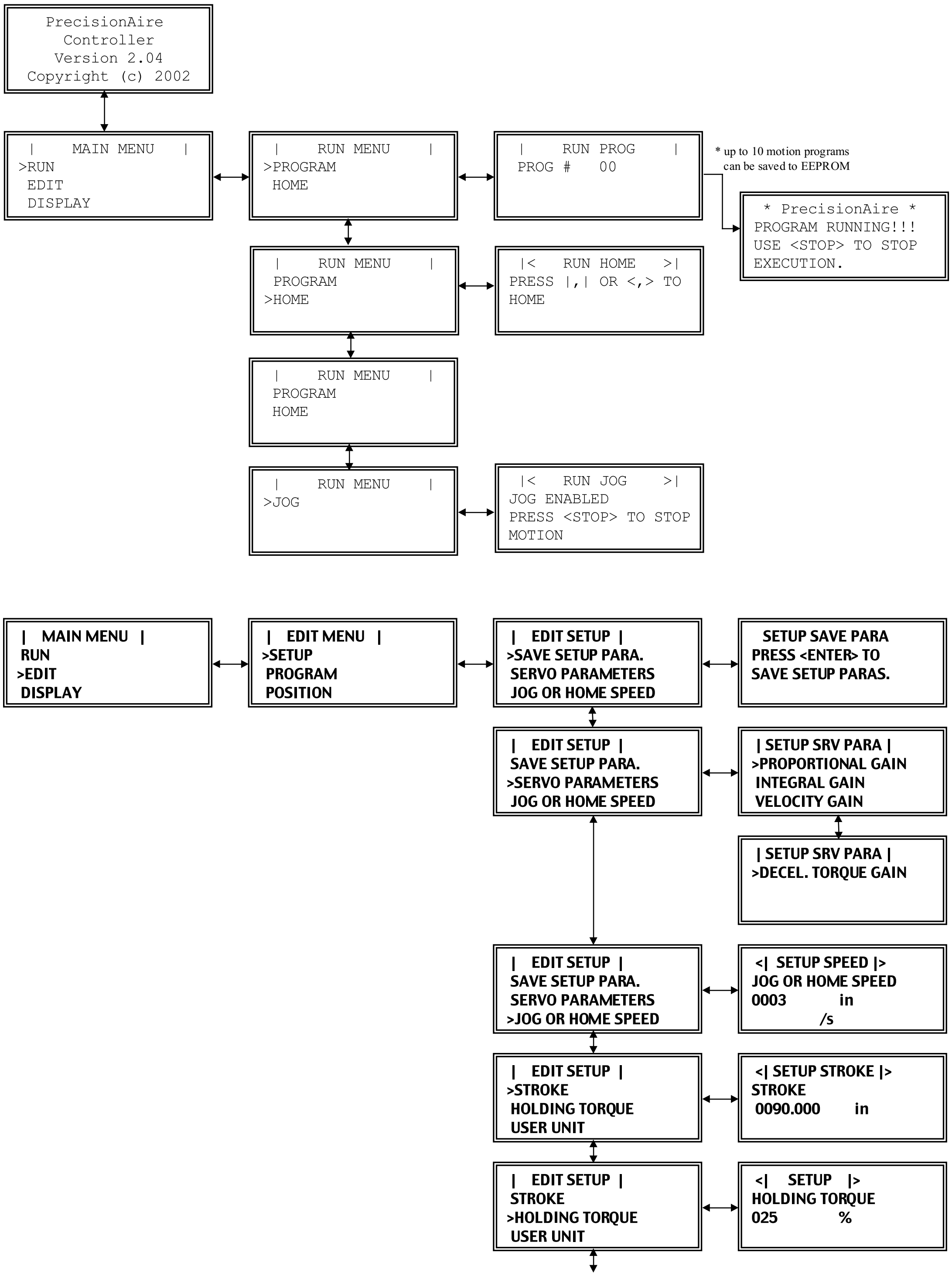


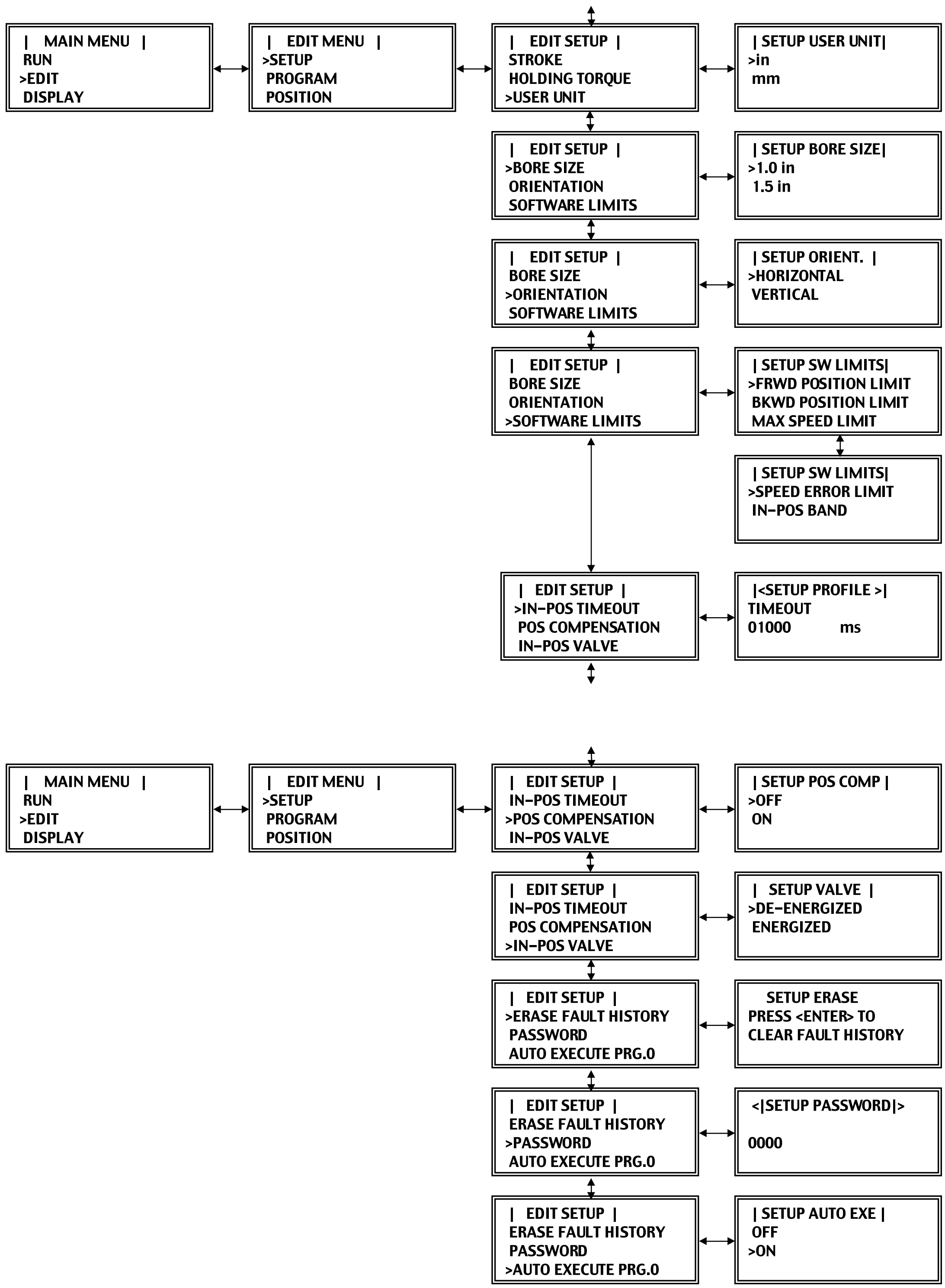
Figure 9.1 Precisionaire control-block diagram

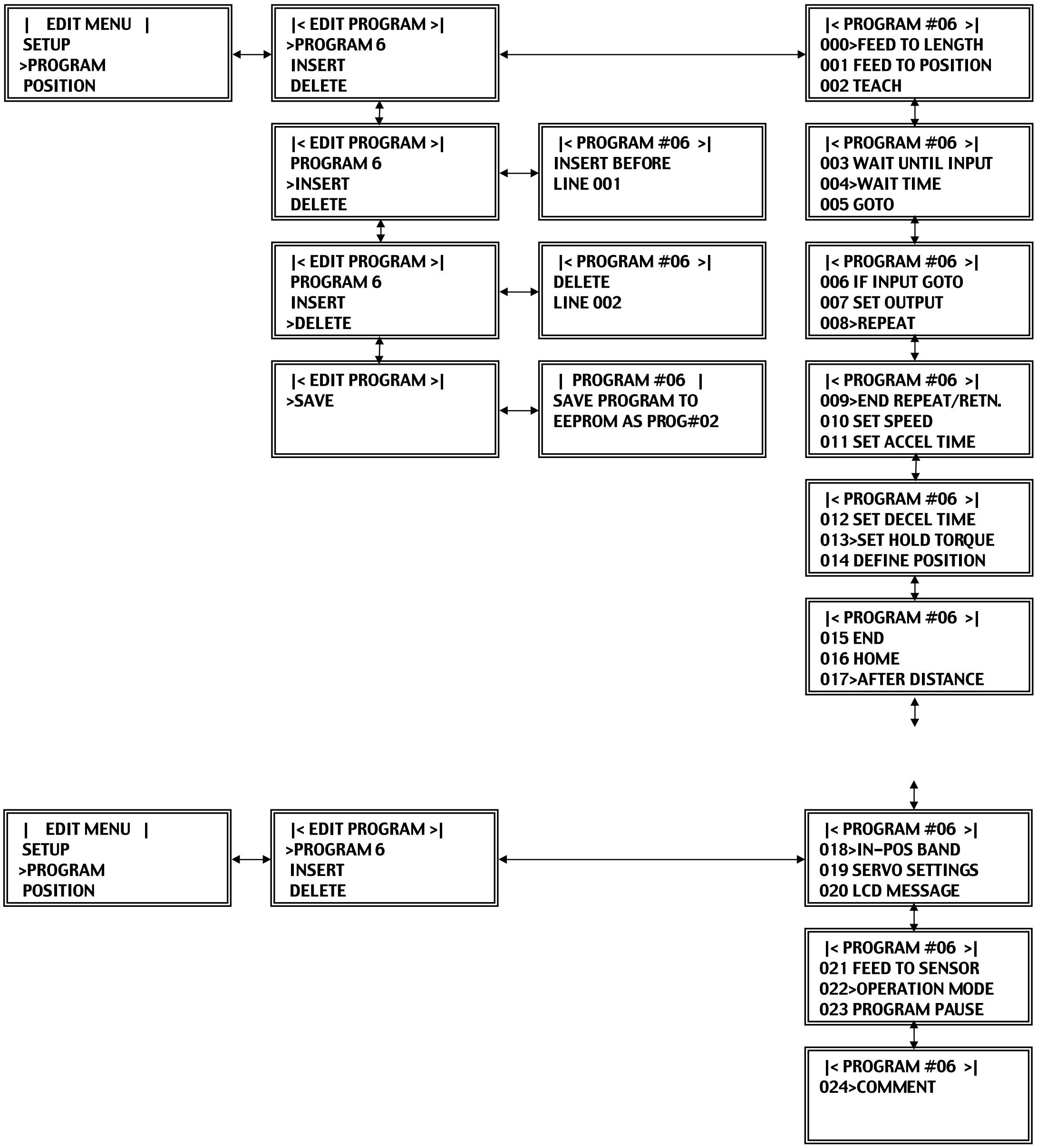
Where,

Xcmd	- Position command in counts	acc	- Carriage acceleration
X	- Position in counts	vel	- Carriage velocity in counts/sec
X_linear	- Linear position in user unit	vel_e	- Velocity error in counts/sec
Xe	- Position error in counts	vel_cmd	- Velocity co mmand in counts/sec
Ic_c	- Current command	A_piston	- Cylinder piston area
Ie	- Current error	R_pulley	- Pulley pitch radius
Ib_c	- Current feedback	K_torque	- Brake torque constant
Vb	- Voltage applied to brake coil	K_fdbk	- Current feedback constant
Vfdbk	- Feedback vo ltage	K_adc	- Analog to digital conversion constant
Ib	- Brake current	K_iPWM	- PWM duty cyc le constant
Tb	- Brake torque	Kv	- Velocity control gain
Fb	- Stopping force from brake	Kp	- Proportional gain
F_air	- Thrust force from air pressure	Ki	- Integral gain
F_fric	- External friction force	L	- Brake coil inductance
Fn	- Net force to drive load	R	- Brake coil resistance
Delta_Pair	- Pressure difference betw een cylinder chambers	s	- Laplace transform operator

9.2 LCD Screens




| MAIN MENU |
RUN
>EDIT
DISPLAY
| EDIT MENU |
>SETUP
PROGRAM
POSITION
| EDIT SETUP |
STROKE
HOLDING TORQUE
>USER UNIT
| SETUP USER UNIT|
>in
mm
| EDIT SETUP |
>BORE SIZE
ORIENTATION
SOFTWARE LIMITS
| SETUP BORE SIZE|
>1.0 in
1.5 in
| EDIT SETUP |
BORE SIZE
>ORIENTATION
SOFTWARE LIMITS
| SETUP ORIENT. |
>HORIZONTAL
VERTICAL
| EDIT SETUP |
BORE SIZE
ORIENTATION
>SOFTWARE LIMITS
| SETUP SW LIMITS|
>FRWD POSITION LIMIT
BKWD POSITION LIMIT
MAX SPEED LIMIT
| SETUP SW LIMITS|
>SPEED ERROR LIMIT
IN-POS BAND
| EDIT SETUP |
>IN-POS TIMEOUT
POS COMPENSATION
IN-POS VALVE
|<SETUP PROFILE >|
TIMEOUT
01000 ms
| MAIN MENU |
RUN
>EDIT
DISPLAY
| EDIT MENU |
>SETUP
PROGRAM
POSITION
| EDIT SETUP |
IN-POS TIMEOUT
>POS COMPENSATION
IN-POS VALVE
| SETUP POS COMP |
>OFF
ON
| EDIT SETUP |
IN-POS TIMEOUT
POS COMPENSATION
>IN-POS VALVE
| SETUP VALVE |
>DE-ENERGIZED
ENERGIZED
| EDIT SETUP |
>ERASE FAULT HISTORY
PASSWORD
AUTO EXECUTE PRG.0
SETUP ERASE
PRESS <ENTER> TO
CLEAR FAULT HISTORY
| EDIT SETUP |
ERASE FAULT HISTORY
>PASSWORD
AUTO EXECUTE PRG.0
<|SETUP PASSWORD|>
0000
| EDIT SETUP |
ERASE FAULT HISTORY
PASSWORD
>AUTO EXECUTE PRG.0
| SETUP AUTO EXE |
OFF
>ON


| EDIT MENU |
SETUP
>PROGRAM
POSITION
|< EDIT PROGRAM >|
>PROGRAM 6
INSERT
DELETE
|< PROGRAM #06 >|
000>FEED TO LENGTH
001 FEED TO POSITION
002 TEACH
|< EDIT PROGRAM >|
PROGRAM 6
>INSERT
DELETE
|< PROGRAM #06 >|
INSERT BEFORE
LINE 001
|< PROGRAM #06 >|
003 WAIT UNTIL INPUT
004>WAIT TIME
005 GOTO
|< EDIT PROGRAM >|
PROGRAM 6
INSERT
>DELETE
|< PROGRAM #06 >|
DELETE
LINE 002
|< PROGRAM #06 >|
006 IF INPUT GOTO
007 SET OUTPUT
008>REPEAT
|< EDIT PROGRAM >|
>SAVE
| PROGRAM #06 |
SAVE PROGRAM TO
EEPROM AS PROG#02
|< PROGRAM #06 >|
009>END REPEAT/RETN.
010 SET SPEED
011 SET ACCEL TIME
|< PROGRAM #06 >|
012 SET DECEL TIME
013>SET HOLD TORQUE
014 DEFINE POSITION
|< PROGRAM #06 >|
015 END
016 HOME
017>AFTER DISTANCE
| EDIT MENU |
SETUP
>PROGRAM
POSITION
|< EDIT PROGRAM >|
>PROGRAM 6
INSERT
DELETE
|< PROGRAM #06 >|
018>IN-POS BAND
019 SERVO SETTINGS
020 LCD MESSAGE
|< PROGRAM #06 >|
021 FEED TO SENSOR
022>OPERATION MODE
023 PROGRAM PAUSE
|< PROGRAM #06 >|
024>COMMENT

Motion Command Screens

```
|< PROG#06 L000 >|
FEED TO LENGTH,DIST=
+0034.000      in
```

```
|< PROG#06 L001 >|
FEED TO POSITION
+0012.000      in
```

```
|< PROG#06 L002 >|
TEACH, POS=
+0089.000(stored) in
+0012.500(current)
```

```
|< PROG#06 L003 >|
WAIT UNTIL INPUT 007
GOES HIGH
```

```
|< PROG#06 L004 >|
WAIT TIME
0100          ms
```

```
|< PROG#06 L005 >|
GOTO LINE
255
```

```
|< PROG#06 L006 >|
IF INPUT 005 HIGH
GOTO LINE 123
```

```
|< PROG#06 L007 >|
SET OUTPUT 04
TO LOW
```

```
|< PROG#06 L008 >|
REPEAT
123          TIMES
```

```
  PROG#06 L009
END REPEAT
```

```
|< PROG#06 L010 >|
SET SPEED
0080          in
              /s
```

```
|< PROG#06 L011 >|
SET ACCEL TIME
0080          ms
```

```
|< PROG#06 L012 >|
SET DECEL TIME
0100          ms
```

```
|< PROG#06 L013 >|
SET HOLDING TORQUE
025            %
```

```
|< PROG#06 L014 >|
DEFINE POSITION
+0023.000      in
```

```
  PROG#06 L015
END OF PROGRAM
GOTO [SAVE PROG] TO
SAVE THE PROGRAM
```

```
| PROG#06 L016 |
HOME
0  0-AWAY FROM BRAKE
   1-TOWARD BRAKE
```

```
|< PROG#06 L017 >|
SET OUTPUT 00
TO HIGH AFTRE DIST.
+0012.500      in
```

```
|< PROG#06 L018 >|
IN-POSITION BAND
±0000.000      in
```

```
|< PROG#06 L019 >|
 KP  KI  KV  KT
024 002 004 032
```

```
|< PROG#06 L020 >|
LCD MESSAGE
PLEASE EDIT FROM PC
```

```
|< PROG#06 L021 >|
FEED FWD TIL INP 001
GOES  LOW
```

```
|< PROG#06 L022 >|
HOLDING TORQUE
025            %
THRUST FWD
```

```
|< PROG#06 L023 >|
PAUSE WHEN INPUT 001
GOES HIGH
```

Motion Command Screens (CON'T)

```
|< PROG#06 L024 >|
COMMENT
PLEASE EDIT FROM PC
```

```
|< PROG#09 L000 >|
FEED TO LENGTH,DIST=
VARI.#00
```

```
|< PROG#09 L001 >|
FEED TO POSITION
VARI.#11
```

```
|< PROG#09 L002 >|
WAIT TIME
VARI.#22
```

```
|< PROG#09 L003 >|
REPEAT
VARI.#33
```

```
|< PROG#09 L004 >|
SET SPEED
VARI.#44
              /s
```

```
|< PROG#09 L005 >|
SET ACCEL TIME
VARI.#55
```

```
|< PROG#09 L006 >|
SET DECEL TIME
VARI.#63
```

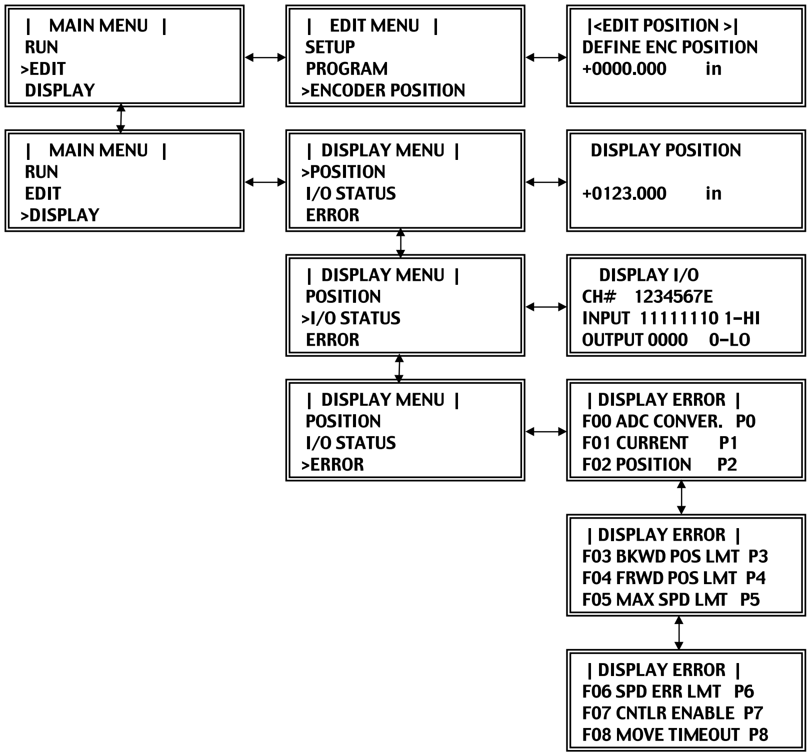
```
|< PROG#09 L007 >|
IF INPUT 121 &_LO
GOTO PROG 007
```

```
|< PROG#09 L007 >|
IF INPUT 061 &_HI
CALL PROG 006
```

```
|< PROG#06 L005 >|
GOTO PROG
5
```

```
|< PROG#06 L005 >|
CALL PROG
4
```

```
  PROG#06 L009
SUBROUTINE RETURN
```


| MAIN MENU |
RUN
>EDIT
DISPLAY
| EDIT MENU |
SETUP
PROGRAM
>ENCODER POSITION
|<EDIT POSITION >|
DEFINE ENC POSITION
+0000.000 in
| MAIN MENU |
RUN
EDIT
>DISPLAY
| DISPLAY MENU |
>POSITION
I/O STATUS
ERROR
DISPLAY POSITION
+0123.000 in
| DISPLAY MENU |
POSITION
>I/O STATUS
ERROR
DISPLAY I/O
CH# 1234567E
INPUT 11111110 1-HI
OUTPUT 0000 0-LO
| DISPLAY MENU |
POSITION
I/O STATUS
>ERROR
| DISPLAY ERROR |
F00 ADC CONVER. P0
F01 CURRENT P1
F02 POSITION P2
| DISPLAY ERROR |
F03 BKWD POS LMT P3
F04 FRWD POS LMT P4
F05 MAX SPD LMT P5
| DISPLAY ERROR |
F06 SPD ERR LMT P6
F07 CNTLR ENABLE P7
F08 MOVE TIMEOUT P8

A: Warranty Information

The following product warranty and return goods information summarizes the product warranty and return policy of Tol-O-Matic. A copy of the formal Return Goods and Field Service Policy is available upon request.

Defective Equipment

If the user is unable to correct a problem, and the product is defective, the unit may be returned to any distributor of Tol-O-Matic products for repair or replacement.

There are no field serviceable parts in the controller/drive. If the controller/drive fails, the unit should be returned to the factory for repair or replacement. To save unnecessary work repair charges, please verify that the controller/drive unit is defective before returning it for repair.

PrecisionAire controllers/drives are warranted against defects in material and assembly. Limitations to warranty coverage are detailed in Return Goods and Field Service Policy. Products that have been modified by the customer, physically mishandled, or otherwise abused through incorrect wiring, inappropriate settings, and so on, are exempt from the warranty plan.

Return Procedure

To ensure accurate processing and prompt return of any Tol-O-Matic product, the following procedure must be followed:

1. Call the nearest distributor of Tol-O-Matic product to obtain a Return Material Authorization (RMA) number. Do not return the controller/drive or any other equipment without a valid RMA number. Returns lacking a valid RMA number will not be accepted and will be returned to the sender.
2. Pack the control/drive in the original shipping carton. Tol-O-Matic is not responsible or liable for damage resulting from improper packing or shipment. Repaired units are shipped via UPS Ground delivery. If another method of shipping is desired, please indicated this when requesting the RMA number and include this information with the return unit.

Product Support

PrecisionAire product support is available over the phone. When you call, you should have the hardware and software manuals at hand. Be prepared to give the following information.

A. The version number of the hardware and software products.
B. The type of hardware that you are using.
C. The fault message and the exact wording of any message that appears on your screen.
D. How you have tried to solve the problem.

Distributor & Representative Network

Tol-O-Matic has a wide network of distributors that are trained to support our products. If you encounter problems, call the distributor or representative where you purchased the product before contacting the factory.

B: Binary Code

Decimal	Binary	Decimal	Binary	Decimal	Binary	Decimal	Binary
0	0000000	32	0100000	64	1000000	96	1100000
1	0000001	33	0100001	65	1000001	97	1100001
2	0000010	34	0100010	66	1000010	98	1100010
3	0000011	35	0100011	67	1000011	99	1100011
4	0000100	36	0100100	68	1000100	100	1100100
5	0000101	37	0100101	69	1000101	101	1100101
6	0000110	38	0100110	70	1000110	102	1100110
7	0000111	39	0100111	71	1000111	103	1100111
8	0001000	40	0101000	72	1001000	104	1101000
9	0001001	41	0101001	73	1001001	105	1101001
10	0001010	42	0101010	74	1001010	106	1101010
11	0001011	43	0101011	75	1001011	107	1101011
12	0001100	44	0101100	76	1001100	108	1101100
13	0001101	45	0101101	77	1001101	109	1101101
14	0001110	46	0101110	78	1001110	110	1101110
15	0001111	47	0101111	79	1001111	111	1101111
16	0010000	48	0110000	80	1010000	112	1110000
17	0010001	49	0110001	81	1010001	113	1110001
18	0010010	50	0110010	82	1010010	114	1110010
19	0010011	51	0110011	83	1010011	115	1110011
20	0010100	52	0110100	84	1010100	116	1110100
21	0010101	53	0110101	85	1010101	117	1110101
22	0010110	54	0110110	86	1010110	118	1110110
23	0010111	55	0110111	87	1010111	119	1110111
24	0011000	56	0111000	88	1011000	120	1111000
25	0011001	57	0111001	89	1011001	121	1111001
26	0011010	58	0111010	90	1011010	122	1111010
27	0011011	59	0111011	91	1011011	123	1111011
28	0011100	60	0111100	92	1011100	124	1111100
29	0011101	61	0111101	93	1011101	125	1111101
30	0011110	62	0111110	94	1011110	126	1111110
31	0011111	63	0111111	95	1011111	127	1111111

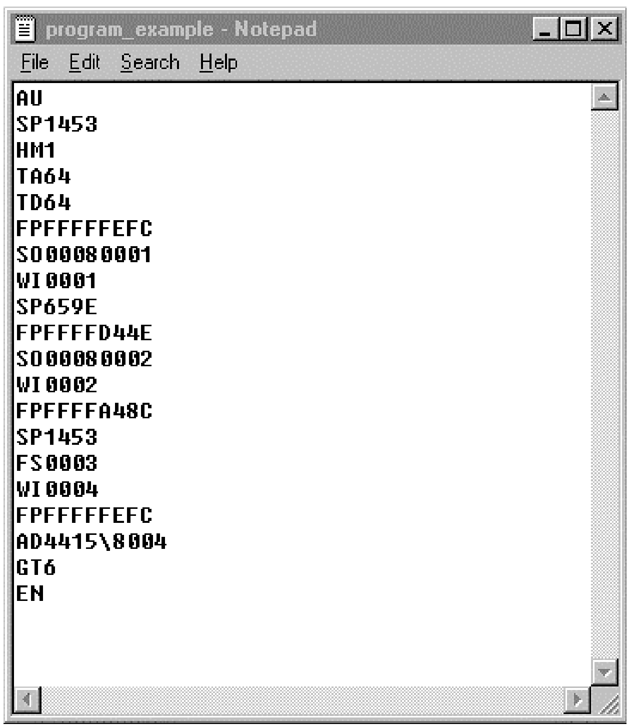
C: Technical Reference – Communication Protocol

What is a Programming Command?

Programming commands are used to write a motion program. It is different from a terminal command that the user can enter on a terminal. A program is typically created in a plain text format and saved in user's PC. After being created in a text format, a program can then be downloaded to controller and saved in controller EEPROM. After downloaded to the RAM of PrecisionAire controller, program commands can be sequentially executed line by line for an application.

Example using Microsoft® Notepad:

Open the notepad application and create a program in text format by including programming commands as follows. Please refer to the programming commands in this reference section for details on all available commands. A program needs to be loaded to RAM in order for execution. Execution always starts from first line of program

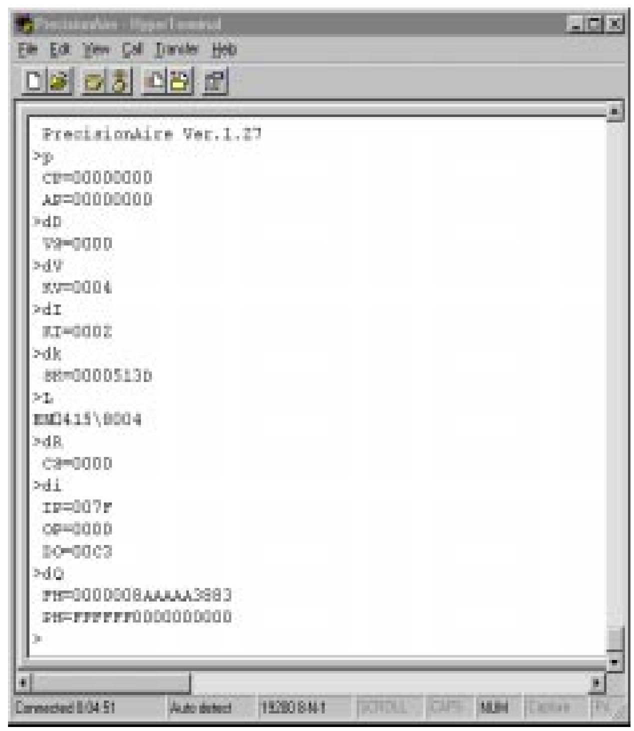


What is a Terminal Command?

Terminal commands are used to set parameters, request information, execute or terminate a program. Parameters such as actuator stroke length, servo settings, bore size, or holding torque can be configured in terminal mode. Program created in text file can be downloaded and saved in EEPROM in terminal mode. Please refer to the terminal commands in this reference section for detail descriptions about all available commands.

Example using HyperTerminal in Windows:

Open a hyperterminal application with baud rate 19200, 8 data bits, none parity, one stop bit and none flow control. Check "Echo typed characters locally" under ASCII setup and select COM port for serial connection to PAS controller. Command prompt (>) will respond after hitting ENTER key when communication is established.



To download a program:

1. Type 'L' and hit 'ENTER' key to switch to download mode.
2. Click on the "Transfer" from file menu bar and choose "Send Text File …".
3. Open the program file created in notepad.
4. Use 'Ctrl+C' to finish program download.

Example to make a move in hyperterminal

When controller is powered up initially. Please make sure that the servo loop is activated by send terminal command 's'. Then you can use 'P' command to specify absolute position then use 'g' command to initiate a move.

The example above shows to make an absolute move to 2000H encoder counts at acceleration and deceleration time of 080H (128 ms) and speed of 7000H counts/sec. When a go command 'g' is issued, controller will move carrier to commanded position.

AD: After Distance

Syntax:
AD [Data 1] [\ Data 2]

Description:
The after distance command is used to trigger output after a distance specified in Data 1. The after distance command has to be appended after feed to position (FP) or feed to length (FL) command.

Arguments:
Data 1: 32-bit distance in encoder counts in hexadecimal format,
Or 16-bit address of a variable
Data 2: Output control, after constant distance
100x: Set single output x HIGH (x=1 to 4)
000x: Set single output x LOW (x=1 to 4)
800x: Set multiple output as BYTE (x=0 to Fh)
A00x: Set multiple output HIGH (x=0 to Fh)
C00x: Set multiple output LOW (x=0 to Fh)

Or,
Output control, after variable distance
180x: Set single output x HIGH (x=1 to 4)
080x: Set single output x LOW (x=1 to 4)
880x: Set multiple output as BYTE (x=0 to Fh)
A80x: Set multiple output HIGH (x=0 to Fh)
C80x: Set multiple output LOW (x=0 to Fh)

Type:
Program command

Related Command:
FP – Feed to Position
FL – Feed to Length

Example:

Instruction	Interpretation
AD1000\1001	; Turn output 1 HIGH after 4096 ; displacement counts
AD2000\0002	; Turn output 2 LOW after 8192 ; displacement counts
AD1000\800A	; Set output 2 & 4 HIGH and 1 & 3 LOW ; after 4096 displacement counts
AD1000\A00A	; Set output 2 & 4 LOW with the other ; output remain unchanged after 4096 ; displacement counts
AD1000\C00A	; Set output 2 & 4 HIGH with the other ; output remain unchanged after 4096 ; displacement counts
SC0050\8000	; Assign constant 0050h to variable #0
SC1000\8001	; Assign constant 1000h to variable #1
FL1800	; Feed forward 1800h(6144) encoder counts
AD8000\1801	; Turn output 1 HIGH after distance stored ; in variable #0
AD8001\0801	; Turn output 1 LOW after distance stored in ; variable #1

DP: Define Encoder Position

Syntax:
DP [Data 1]

Description:
The set define position command will redefine the current encoder position to the value specified in Data1.

Arguments:
Data 1: 32-bit encoder in hexadecimal format.

Type:
Program command

Related Command:
FP – Feed to Position
FL – Feed to Length
HM – Carriage Home

Example:

Instruction	Interpretation
DP1000	; Define current position as 4096 counts
DPFFFFF000	; Define current position as –4096 counts

EN:End of Program

Syntax:
EN

Description:
The program end command specify the end of program.

Arguments:
None

Type:
Program command

Related Command:
AU – Auto Execution

Example:

Instruction	Interpretation
EN	; End of program

FI: If Input Goto

Syntax:
FI [Data 1] [Data 2] [\Data3]

Description:
The if input goto command will jump to a command line or program specified in Data3 if the input condition is TRUE.

Arguments:

Data 1:	16-bit control data to select one of the following input or jump types.
	Goto a line:
	0 – Goto line when single input LOW
	1 – Goto line when single input HIGH
	8 – Goto line when all 7 inputs match status specified in Data 2
	A – Goto line when multiple inputs all LOW
	C – Goto line when multiple inputs all HIGH
	Goto a program:
	1000 – Goto program when single input LOW
	1001 – Goto program when single input HIGH
	1008 – Goto program when all 7 inputs match status specified in Data 2
	100A – Goto program when multiple inputs all LOW
	100C – Goto program when multiple inputs all HIGH
	Call a program:
	3000 – Call program when single input LOW
	3001 – Call program when single input HIGH
	3008 – Call program when all 7 inputs match status specified in Data 2
	300A – Call program when multiple inputs all LOW
	300C – Call program when multiple inputs all HIGH
Data 2:	Specify one of 7 inputs or input binary combination in hexadecimal format Single input - Data 2 = 1 to 7 Multiple inputs – Data 2 = 00h to 7Fh
Data 3:	Specify line number or program number in hexadecimal format Line - Data 3 = 00h to FFh Program – Data 3 = 0 to 9

Type:
Program command

Related Command:
WT – Wait Time
SO – Set Output
WI – Wait For Input

Example:

Instruction	Interpretation
FI0007\0032	; If input 7 is LOW, goto LINE #50
FI00010007\0032	; If input 7 goes HIGH, goto LINE #50
FI00080037\0032	; If input 4 & 7 are LOW and input ; 1,2,3,5 & 6 are HIGH, goto LINE #50
FI000A0037\0032	; If input 1,2,3,5 & 6 are LOW, ; goto LINE #50
FI000C0037\0032	; If input 1,2,3,5 & 6 are HIGH, ; goto LINE #50
FI10000007\0004	; If input 7 is LOW, goto PROG #4
FI10010007\0004	; If input 7 goes HIGH, goto PROG #4
FI10080037\0004	; If input 4 & 7 are LOW and input ; 1,2,3,5 & 6 are HIGH, goto PROG #4
FI100A0037\0004	; If input 1,2,3,5 & 6 are LOW, ; goto PROG #4
FI100C0037\0004	; If input 1,2,3,5 & 6 are HIGH, ; goto PROG #4
FI30000007\0004	; If input 7 is LOW, call PROG #4
FI30010007\0004	; If input 7 goes HIGH, call PROG #4
FI30080037\0004	; If input 4 & 7 are LOW and input ; 1,2,3,5 & 6 are HIGH, call PROG #4
FI300A0037\0004	; If input 1,2,3,5 & 6 are LOW, ; call PROG #4
FI300C0037\0004	; If input 1,2,3,5 & 6 are HIGH, ; call PROG #4

FL: Feed to Length / Cut to Length

Syntax:
FL [Data 1] [\ Vari D]

Description:
The feed to length command moves carriage by a relative distance in encoder quadrature counts. Incremental distance can be actually specified (Data 1) or referred to a double word variable (Vari D). For "Feed to Length" move, incremental distance is added to last commanded position. For "Cut to Length" move, incremental distance is added to current encoder position.

Arguments:

Data 1:	32-bit encoder increments in hexadecimal format
Vari D:	Double word variable address, Vari D = 80xx, where xx = 00h to 3Fh
Note:	Bit 11 of Vari D is used for cut-to-length control flag Vari D = 0800 or 88xx for incremental move (cut to length) based on current encoder position.

Type:
Program command

FL: Feed to Length / Cut to Length (cont.)

Related Command:

FP – Feed to Position
DP – Define Encoder Position
MG – Display Message on LCD & Read Data
FS – Feed to Sensor
OM – Operation Mode

Example:

Instruction	Interpretation
FL1000	; Feed to length 1000h(4096) encoder counts
FL02008000	; Feed to length 2008000h encoder counts
FL\8001	; Make incremental move based on the value ; stored in double word variable #1
FLFFFFF000	; Feed to length -4096 encoder counts
CP = 00004000	; Commanded position at 4000h
AP = 00004002	; Current encoder position at 4002h
FL1000\0800	; Cut to length 1000h to target pos 5002h
FL\883F	; Cut to length specified in variable #63

FP: Feed to Position

Syntax:

FP [Data 1] [\ Vari D]

Description:

The feed to position command moves carriage to an absolute position reference to home (0) position in encoder quadrature counts. Absolute position can be actually specified (Data 1) or referred to a double word variable (Vari D).

Arguments:

Data 1:	32-bit encoder position in hexadecimal format
Vari D:	Double word variable address, Vari D = 80xx, where xx = 00h to 3Fh

Type:

Program command

Related Command:

FL – Feed to Length
DP – Define Encoder Position
MG – Display Message on LCD & Read Data
FS – Feed to Sensor
OM – Set Operation Mode

Example:

Instruction	Interpretation
FP1000	; Feed to position 1000h encoder counts
FP02008000	; Feed to position 2008000h encoder counts
FP\800D	; Move to absolute position stored in double ; word variable #13
FPFFFFF000	; Feed to position -4096 encoder counts

FS: Feed to Sensor

Syntax:

FS [Data 1] [Data 2]

Description:

The feed to sensor command combines jog and wait for inputs functions to move carrier forward or backward (specified in Data 1) until a specified input status is reached. When an input is detected, PAS controller will set brake current to 100%, supply pressurized air on both sides of the cylinder and then drop brake current down to specified in-position holding torque.

Arguments:

Data 1:	16-bit data to select one of the following input type and move direction. 0000h – Move backward until specified input LOW 0001h – Move backward until specified input HIGH 1000h – Move forward until specified input LOW 1001h – Move forward until specified input HIGH
Data 2:	Specify one of 7 inputs or input binary combination in hexadecimal format Single input - Data 2 = 1 to 7 Multiple inputs – Data 2 = 00h to 7Fh

Type:

Program & terminal command

Related Command:

FL – Feed to Length
FP – Feed to Position
OM – Set Operation Mode

Example:

Instruction	Interpretation
FS0001	; Keep moving carriage backward until ; input 1 goes LOW
FS10010007	; Move carriage forward until ; input 7 goes HIGH
FS10080037	; Feed forward until input 4 & 7 LOW ; and input 1,2,3,5 & 6 HIGH
FS000A0037	; Feed backward until input 1,2,3,5 & 6 ; LOW and ignore the other inputs
FS100C0037	; Feed forward until input 1,2,3,5 & 6 HIGH ; and ignore the other inputs

FS: Feed to Sensor (cont.)

Note:

1. Position repeatability of feed to sensor move is determined on move speed, load, sensor and valve response time, valve Cv, supplied air pressure and servo settings.

2. Maximum move speed is limited to activated input duration that can be sensed by input. PAS requires a minimum input duration of 10 ms.

3. For higher speed application, higher in-position holding torque will reduce carrier in-position settling time.

GT: Go To

Syntax:
GT [Data 1]
GT [Vari 1] [Data 2] [\ Vari 2]

Description:
The goto command will jump program execution to a different line or different program specified in Data 1. It can also be used to initiate the jump based on a logical comparison between Vari 1 & 2.

Arguments:

Data 1: Line or program number to specify jump location in hexadecimal format.
Line – Data 1 = 0h to 0FFh
Program – Data 1 = 1000h to 1009h

Data 2: Logic control & line or program number to specify jump location in hexadecimal format.

BIT #	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0

Bit F~E: Reserved
Bit D: Goto (0) or program call (1) control bit
Bit C: Jump to line (0) or program (1) control bit
Bit B: Reserved
Bit A~8: Logic control bits.
000 – no logic expression
001 – greater than (>)
010 – greater than or equal (>=)
011 – equal (=)
100 – Less than (<)
101 – Less than or equal (<=)
110 – Not equal (<>)
Bit 7~0: Line number or program number.

Vari #: Variable address, Vari # = 80xx, where xx = 00h to 3Fh

Type:
Program command

Related Command:
FI – If Input Go to

Example:

Instruction	Interpretation
GT0064	; Go to program line #100 (64h)
GT8000 0264\8001	; Go to program line #100 if value stored in ; variable #0 is greater than or equal ; to that in variable #1
GT1005	; Go to program #5
GT8002 1606\8003	; Go to program #6 if value stored in ; variable #2 is not equal to that in ; variable #3
GT3005	; Call program #5
GT8002 3106\8003	; Call program #6 if value stored in ; variable #2 is greater than that in ; variable #3

HM: Carriage Home

Syntax:
HM [Data 1]

Description:
The home command will move carriage toward the end of stroke and define the position as zero position.

Arguments:
Data 1 = 0, home away from the brake
Data 1 = 1, home toward the brake

Type:
Program command

Related Command:
DP – Define Position

Example:

Instruction	Interpretation
HM0	; Home carriage away from brake
HM1	; Home carriage toward brake

MG: Display Message on LCD & Read Data

Syntax:
MG [Str] [\ Vari]

Description:
The display message command will display text message specified in Str and store data input from keypad to a variable Vari.

Arguments:

Str:	Text string up to 60 characters per command line. Backlash '\' is reserved for controller internal use.
Vari:	Variable address Vari = 80xx for float format variable with user's unit & sign conversion, where xx = 00h to 3Fh Vari = A0xx for integer format variable without user's unit & sign conversion, where xx = 00h to 3Fh Vari = 0 for no data entry, display string only.

Note: Maximum MG and NO string buffer is 1024 characters per program. MG command shares same text buffer with NO (no operation, comment) command.

Type:
Program command

Related Command:
FL – Feed to Length
FP – Feed to Position
WT – Wait For Time
RP – Repeat Loop
SP – Set Speed
TA – Set Acceleration Time
TD – Set Deceleration Time
NO – Comment, No Operation

Example:

Instruction	Interpretation
MGPlease Enter Speed (in/sec)\8001	; Display string in LCD. ; Convert speed data entered ; by user from keypad & store; it to double word vari. #1
MGPlease Enter Wait Time\A002	; Display string in LCD and ; store wait time to ; variable #2 (integer format)
MGMachine door is open!\0	; Display string in LCD with ; no variable data entry
MG\0	; Clear LCD screen
MGPlease Enter Loops\A003	; Display string in LCD and ; store # of loops to ; variable #3

NO: Comment, No Operation

Syntax: NO

Description:
The comment command allow users to comment each program up to 1024 character.

Note: Maximum NO and MG string buffer is 1024 characters per program. NO command shares same text memory with MG command.

Arguments:
None

Type:
Program command

Related Command:
MG – Display Message on LCD & Read Data

Example:

Instruction	**Interpretation**
NOThis is a comment\0	; Program comment, ; comment string has to be; ended with "\0"

OM: Set Operation Mode

Syntax:
OM [Data 1] [Data 2] \ [Data 3]

Description:
The set operation mode command allows user to configure PAS controller in regular servo mode or in thrust mode. Data 1 specifies servo or thrust mode, Data 2 specifies holding torque on brake, and Data 3 specifies move direction if in thrust mode.

Arguments:

Data 1:	16-bit data to select servo or thrust mode. 0000h – Configure controller in regular servo mode 0001h – Configure controller in thrust mode
Data 2:	16-bit data to specify current applied to brake. Data 2 = 00h to 06Ch
Data 3:	16-bit data to specify thrust mode move direction. 0 – Move carrier backward, away from brake. 1 – Move carrier forward, toward brake.

Type:
Program command

Related Command:
FL – Feed to Length

OM: Set Operation Mode (cont.)

FP – Feed to Position
FS – Feed to Sensor

Example:

Instruction	Interpretation
OM00010010\0001	; Thrust mode, Move carrier forward ; with brake holding torque of 010h
OM001B	; Set controller to servo mode with brake ; holding torque of 01Bh

Note: Terminal command resume (Z) will not resume motion or program in thrust mode. Program needs to be re-executed after stop (x) or pause (z) in thrust mode.

PI: Pause Program on Input

Syntax:
PI [Data 1] [Data 2]

Description:
The pause program on inputs command will pause program execution and stop motion immediately when single or multiple input status is reached. Data 2 is selected for up to 7 inputs in binary combination.
Note: Program pause will automatically be disabled when jump to a different program. Use PI0 to disable pause on inputs.

Arguments:

Data 1:	16-bit data to select one of the following input type. 0 - Pause when single input goes LOW 1 – Pause when single input goes HIGH 8 – Pause when all inputs match status specified in Data 2 A – Pause when multiple inputs are LOW C – Pause when multiple inputs are HIGH
Data 2:	Specify one of 7 inputs or input binary combination in hexadecimal format Single input - Data 2 = 1 to 7 Multiple inputs – Data 2 = 00h to 7Fh

Type:
Program command

Related Command:
WI – Wait For Input

Example:

Instruction	Interpretation
PI0	; Disable pause on inputs
PI0001	; Pause when input 1 goes LOW
PI00010007	; Pause when input 7 goes HIGH
PI00080037	; Pause when input 4 & 7 are LOW ; and input 1,2,3,5 & 6 are HIGH
PI000A0037	; Pause when input 1,2,3,5 & 6 are LOW ; and ignore the other inputs
PI000C0037	; Pause when input 1,2,3,5 & 6 are HIGH ; and ignore the other inputs

PR: Set Position Repeatability

Syntax:
PR [Data 1]

Description:
The set position repeatability command will set the position repeatability to the value specified in Data 1. Minimum of Data 1 is 5 encoder counts.

Arguments:

Data 1:	16-bit data to specify position repeatability in encoder counts in hexadecimal format. Data 1 = 05h to FFFFh

Type:
Program command
Related Command:
SS – Set Servo Settings

Example:

Instruction	**Interpretation**
PR0010	; Set position repeatability to 16 counts
PR0005	; Set position repeatability to 5 counts

RE: Repeat End / Return

Syntax:
RE [Data 1]

Description:
The repeat end command specify the end of a repeat loop and will branch program execution to the beginning of the loop (RP) if the remaining loop number greater than zero.

Arguments:
Data 1 = 0, end of repeat loop
Data 1 = 0054, return of program call

RE: Repeat End / Return (cont.)

Type:
Program command

Related Command:
RP – Repeat

Example:

Instruction	Interpretation
RE	; End of repeat loop
RE54	; Return of program call

RP: Repeat Loop

Syntax:
RP [Data 1] [\ Vari S]

Description:
The repeat command specify number of repeat loops. Controller will execute program between repeat (RP) and repeat end (RE) for number of loops that specified in Data 1 or stored in a variable Vari S.

Arguments:

Data 1:	16-bit data to specify number of repeat loops in hexadecimal format.
Vari S:	Variable address, Vari S = 80xx, where xx = 00h to 3Fh

Type:
Program command

Related Command:
RE – Repeat End
MG – Display Message on LCD & Read Data

Example:

Instruction	Interpretation
RP0064	; Repeat 100 times
RP\8005	; Repeat number of loops specified in ; variable #5

SC: Set Constant to Variable

Syntax:
SC [Data] [\ Vari]

Description:
The set constant to variable command will assign a constant value (Data) to a variable (Vari).

Arguments:

Data:	16-bit or 32-bit constant value in hexadecimal format.
Vari:	Variable address, Vari = 80xx for variable, where xx = 00h to 3Fh

Type:
Program command

Related Command:
GT – Go To
MG – Display Message on LCD & Read Data
SV – Store Internal Value to Variable

Example:

Instruction	Interpretation
SC0100\8001	; Assign 0100h to variable #1
SC00402000\8000	; Assign 00402000h to variable #0

SL: Configure Software Limits

Syntax:
SL [Flag]

Description:
The configure software limits command is use to enable or disable position and speed limits in a program.

Arguments:
Flag = 0, Disable software limits check
Flag = 1, Enable software limits check.

Type:
Program command

Related Command:
None

Example:

Instruction	Interpretation
SL1	; Enable software limits
SL0	; Disable software limits

SP: Set Speed

Syntax:
SP [Data 1] [\ Vari D]

Description:
The set speed command will set move speed in counts per second in Data 1. It can also take speed data from a double word variable Vari D.

Arguments:

Data 1:	32-bit data to specify speed in counts/sec in hexadecimal format.
Vari D:	Double word variable address, Vari D = 80xx, where xx = 00h to 3Fh

Type:
Program command

Related Command:
TA – Set Acceleration time
TD – Set Deceleration Time
MG – Display Message on LCD & Read Data

Example:

Instruction	Interpretation
SP00004000	; Set speed 16384 counts/sec, or 4 encoder ; revolutions per second.
SP\8003	; Set speed using data stored in double word ; variable #3

SS: Set Servo Settings

Syntax:
SS [Data 1] [Data 2] [\ Data 3]

Description:
The set servo settings command will set the four servo parameters (KP, KI, KV, KT) in the program.

Arguments:

Data 1: 16-bit servo settings in hexadecimal format

BIT #	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0

Bit F~8: Setting for KP (0 to 80h)
Bit 7~0: Setting for KI (0 to 20h).

Data 2: 16-bit servo settings in hexadecimal format

BIT #	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0

Bit F~8: Setting for KV (0 to 10h)
Bit 7~0: Setting for KT (0 to 6Ch).

Data 3: Setting mask control

BIT #	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0

Bit F~D: Reserved
Bit C: Mask bit for KP, 1-write KP in Data 1 to controller, 0 – do not write KP to controller
Bit B~9: Reserved
Bit 8: Mask bit for KI, 1-write KI in Data 1 to controller, 0 – do not write KI to controller
Bit 7~5: Reserved
Bit 4: Mask bit for KV, 1-write KV in Data 1 to controller, 0 – do not write KV to controller
Bit 3~1: Reserved
Bit 0: Mask bit for KT, 1-write KT in Data 1 to controller, 0 – do not write KT to controller

Type:
Program command

Related Command:
HT – Set In-Position Holding Torque
PR – Set Position Repeatability

Example:

Instruction	Interpretation
SS18030440\1111	; Set KP=24, KI=3, KV=4, KT=64
SS18010544\0101	; Set KI=1, KT=68, but dot write KP & KV

SV: Store Internal Value to Variable

Syntax:
SV [Data] [\ Vari]

Description:
The store internal value to variable command will save value of a internal data (e.g. position or speed) to a variable (Vari).

Arguments:

Data:	16-bit or 32-bit data reference in hexadecimal format.
Vari:	Variable address, Vari = 80xx for double word data, Vari = 90xx for single word data, Where xx = 00h to 3Fh

Type:
Program command

Related Command:
GT – Go To
MG – Display Message on LCD & Read Data
SC – Set Constant to Variable

Example:

Instruction	Interpretation
SV0000\8001	; Store encoder position, ref. address 00, to ; double word variable #1
SV0006\803F	; Store max. speed, ref. address 06, to ; double word variable #63
SV0007\901F	; Store move time, ref. address 080, to ; double word variable #31

Note:

INTERNAL DATA	ADDRESS REFERENCE	DATA TYPE
Encoder Position	0000	32-bit
Commanded Position	0002	32-bit
Commanded Speed	0004	32-bit
Maximum Speed	0006	32-bit
Move Time	0080	16-bit

TA: Set Acceleration Time

Syntax:
TA [Data 1] [\ Vari S]

Description:
The set acceleration time command will set time in millisecond for acceleration. It can also take the time specified in a variable Vari S.

Arguments:
Data 1: 16-bit data to specify acceleration time in milliseconds in hexadecimal format.
Vari S: Variable address,
Vari S = 80xx, where xx = 00h to 3Fh

Type:
Program & terminal command

Related Command:
SP – Set Speed
TD – Set Deceleration Time
MG – Display Message on LCD & Read Data

Example:

Instruction	Interpretation
TA0064	; Set accel time 100 ms
TA\800B	; Set accel time specified in ; variable #11

TD: Set Deceleration Time

Syntax:
TD [Data 1] [\ Vari S]

Description:
The set deceleration time command will set time in millisecond for acceleration. It can also take the time specified in a variable Vari S.

Arguments:
Data 1: 16-bit data to specify acceleration time in milliseconds in hexadecimal format.
Vari S: Variable address,
Vari S = 80xx, where xx = 00h to 3Fh

Type:
Program & terminal command

Related Command:
SP – Set Speed
TA – Set Acceleration Time
MG – Display Message on LCD & Read Data

Example:

Instruction	Interpretation
TD0100	; Set decel time 256 ms
TD\800F	; Set decel time specified in ; variable #15

WI: Wait For Input

Syntax:
WI [Data 1] [Data 2]

Description:
The wait for input command will pause program execution until single or multiple input status is reached. It can wait for up to 7 inputs in binary combination.

Arguments:
Data 1: 16-bit data to select one of the following input type.
0 - Wait for single input LOW
1 – Wait for single input HIGH
8 – Wait for all 7 inputs match status specified in Data 2A – Wait for multiple inputs all LOW
C – Wait for multiple inputs all HIGH
Data 2: Specify one of 7 inputs or input binary combination in hexadecimal format
Single input - Data 2 = 1 to 7
Multiple inputs – Data 2 = 00h to 7Fh

WI: Wait For Input (cont.)

Type:
Program command

Related Command:
WT – Wait Time
SO – Set Output
FI – If Input Go to
PI – Program Pause on Input

Example:

Instruction	**Interpretation**
WI0001	; Wait for input 1 goes LOW
WI00010007	; Wait for input 7 goes HIGH
WI00080037	; Wait for input 4 & 7 LOW
	; and input 1,2,3,5 & 6 HIGH
WI000A0037	; Wait for input 1,2,3,5 & 6 LOW
	; and ignore the other inputs
WI000C0037	; Wait for input 1,2,3,5 & 6 HIGH
	; and ignore the other inputs

WT: Wait Time

Syntax:
WI [Data 1] [\ Vari S]

Description:
The wait time command will halt program execution for a period of time specified in Data 1. It can also wait for a time specified in variable Vari S.

Arguments:

Data 1:	16-bit data to specify wait time in milliseconds in hexadecimal format.
Vari S:	Variable address,
	Vari S = 80xx, where xx = 00h to 3Fh

Type:
Program command

Related Command:
WI – Wait For Input
FI – If Input Go to
MG – Display Message on LCD & Read Data

Example:

Instruction	Interpretation
WT0064	; Wait for 100 ms
WI\800A	; Wait for time specified in
	; variable #10

a: Set Reverse Position Limit

Syntax:
a [HEX_32]

Arguments:
HEX_32 is a 32-bit signed reverse position limit in hexadecimal format.

Related Command:
dm – Display software limits
A – Set forward position limit

Example:

Instruction	Interpretation
aFFFFF000	; Set reverse position limit to –4096 counts

A: Set Forward Position Limit

Syntax:
A [HEX_32]

Arguments:
HEX_32 is a 32-bit signed forward position limit in hexadecimal format.

Related Command:
dm – Display software limits
a – Set reverse position limit

Example:

Instruction	Interpretation
A00002000	; Set forward position limit to 8192 counts

bV: Set backward valve ON

Syntax:
bV

Arguments:
None

Related Command:
fV – Set forward valve ON
H – Set in-position holding torque
FS – Feed to sensor

Example:

Instruction	Interpretation
x	; Stop servo loop
H0010	; Set brake current to 010h
bV	; Set backward valve ON, carrier moves
	; away from brake

B: Configure position compensation & program auto execution

Syntax:
B [Flag]

Arguments:
Flag = 0, set auto execution OFF, position compensation OFF
Flag = 1, set auto execution OFF, position compensation ON
Flag = 8, set auto execution ON, position compensation OFF
Flag = 9, set auto execution ON, position compensation ON

Related Command:
dB – Display position compensation & auto execution flag

Example:

Instruction	Interpretation
B1	; Set pos compensation ON, auto exe. OFF
B9	; Set pos compensation ON, auto exe. ON

c: Display keypad password

Syntax:
c

Arguments:
None

Related Command:
CO – Set and overwrite keypad interface password

Example:

Instruction	Interpretation
c	; Display keypad password in HEX
	Controller return:
CO=04D2	; Password is 04D2 hex or 1234 decimal

CO: Set and Overwrite Keypad Interface Password

Syntax:
CO [HEX_16]

Arguments:
CO command will set and overwrite existing keypad password. HEX_16 is the password code from hexadecimal 0h to 270Fh (i.e. 0 to 9999 decimal) for keypad interface.

Related Command:
None

Example:

Instruction	**Interpretation**
CO04D2	; Set and overwrite keypad password to
	; 1234 decimal

Cc: Clear Position Compensation Table

Syntax:
Cc

Arguments:
None

Related Command:
Cl – Load position compensation table to RAM
Cs– Save position compensation table to EEPROM

Example:

Instruction	**Interpretation**
Cc	; Clear position compensation table in RAM

Cl: Load Position Compensation Table to RAM

Syntax:
Cl

Arguments:
None.
Cl will use current program pointer to load position compensation table from EEPROM to RAM.

Related Command:
Cc – Clear position compensation table
Cs– Save position compensation table to EEPROM

Example:

Instruction	**Interpretation**
dp	; Display program number
Controller return:	
PN=0001	; Program number 1
Cl	; Load program #1 position compensation
	; table to RAM

Cs: Save Position Compensation Table to EEPROM

Syntax:
Cs

Arguments:
None.
Cs will use current program pointer to save compensation table to EEPROM.

Related Command:
Cc – Clear position compensation table
Cl – Load position compensation table to RAM

Example:

Instruction	**Interpretation**
dp	; Display program number
Controller return:	
PN=0009	; Program number 9
Cs	; Save position compensation table to
	; EEPROM program #9

db: Display actuator bore size

Syntax:
db

Arguments:
None

Related Command:
o – Set actuator bore size

Example:

Instruction	**Interpretation**
db	; Display bore size
Controller return:	
BO=0000	; Bore size 1-in (0)

dB: Display position compensation & auto execution flag

Syntax:
dB

Arguments:
None

Related Command:
B – Configure internal position compensation

Example:

Instruction	**Interpretation**
dB	; Display position compensation
Controller return:	
CF=000	; Position compensation flag ON (1)
CF=0008	; Program auto execution flag ON (1)

dc: Display current command

Syntax:
dc

Arguments:
None

Related Command:
df – Display current feedback

Example:

Instruction	Interpretation
dc	; Display current command
Controller return:	
CI=001B	; Current input (PWM command) = 1Bh

dD: Display in-position valve status

Syntax:
dD

Arguments:
None

Related Command:
D – Set in-position valve output

Example:

Instruction	Interpretation
dD	; Display in-position valve status
Controller return:	
VS=0000	;Valve status = both de-energized

dE: Display encoder monitoring

Syntax:
dE

Arguments:
None

Related Command:
i - Set encoder monitoring

Example:

Instruction	**Interpretation**
dE	; Display encoder monitoring
Controller return:	;

de: Display position error

Syntax:
de

Arguments:
None

Related Command:
dw – Set position repeatability

Example:

Instruction	**Interpretation**
de	; Display position error
Controller return:	
PE=00000002	; Position error = 2 encoder counts

df: Display current feedback

Syntax:
df

Arguments:
None

Related Command:
dc – Display current command

Example:

Instruction	**Interpretation**
df	; Display current feedback
Controller return:	
AF=00FC	; Analog feedback = FCh counts

dF: Display fault

Syntax:
dF

Arguments:
None

Related Command:
dQ – Display controller fault history

Example:

Instruction	Interpretation
dF	; Display fault
Controller return:	
No flt	; No fault

dh: Display in-position holding torque

Syntax:
dh

Arguments:
None

Related Command:
H – Set in-position holding torque

Example:

Instruction	Interpretation
dh	; Display in-position holding torque
Controller return:	
HT=0019	; Holding torque = 19h

di: Display input/output status

Syntax:
di

Arguments:
None

Related Command:
n – Set output

Example:

Instruction	Interpretation
di	; Display I/O status
Controller return:	
IP=007F	; Input byte = 7Fh
OP=0000	; Output byte = 0h
DO=00C3	; Internal dedicated output = C3h

Note:

IP – Input byte:

BIT #	7	6	5	4	3	2	1	0
BIT	EN	IN7	IN6	IN5	IN4	IN3	IN2	IN1

OP – Output byte:

BIT #	7	6	5	4	3	2	1	0
BIT	—	—	FAULT	IN_Pos	OUT4	OUT3	OUT2	OUT1

DO – Internal dedicated output byte:

BIT #	7	6	5	4	3	2	1	0
BIT	KEY Y	KEY X	VAL 2	VAL 1	—	E	R/W	RS

BIT 7: Keypad row X enable
BIT 6: Keypad column Y enable
BIT 5: Valve 2 (backward) activated
BIT 4: Valve 1 (forward) activated
BIT 3: Reserved
BIT 2: LCD Enable
BIT 1: LCD Read/write
BIT 0: LCD register select

dI: Display integral gain KI

Syntax:
dI

Arguments:
None

Related Command:
I – Set integral gain

Example:

Instruction	Interpretation
dI	; Display integral gain
Controller return:	
KI=0001	; Integral gain KI = 1

dk: Display actuator stroke length

Syntax:
dk

Arguments:
None

Related Command:
k – Set actuator stroke length

Example:

Instruction	**Interpretation**
dk	; Display actuator stroke length
Controller return:	
SK=00003069	; Stroke = 3096h counts

dl: Display load/weight

Syntax:
dl

Arguments:
None

Related Command:
N – Set load/weight

Example:

Instruction	**Interpretation**
dl	; Display load/weight
Controller return:	
WT=0007	; Load/weight = 7

dL: Display executing program & line number

Syntax:
dL

Arguments:
None

Related Command:
dp – Display program number

Example:

Instruction	**Interpretation**
dL	; Display program line number
Controller return:	
EL=4002	; Executing program 4, line #2

dm: Display software limits

Syntax:
dm

Arguments:
None.

Related Command:
a – Set reverse position limit
A – Set forward position limit
v – Set maximum speed limit
w – Set speed following error limit
L0 – Disable software limit check
L1 – Enable software limit check

Example:

Instruction	**Interpretation**
dm	; Display software limits
Controller return:	
RL=FFFFF000	; Reverse position limit = -4096 counts
FL=00004000	; Forward position limit = 16384 counts
SL=00010000	; Maximum speed limit = 65536 counts/sec
SW=1C00	; Speed following error = 7168 counts/sec
LE=0000	;Software limits disabled

dM: Display selected variable name & data

Syntax:
dM

Arguments:
None

Related Command:
m – Select variable
M – Set variable name or data

Example:

Instruction	**Interpretation**
dM	; Display variable
Controller return:	
V#=003F	; Variable #63 (3Fh)
Vname64=1234	; Variable value is 1234h

do: Display overshoot

Syntax:
do

Arguments:
None

Example:

Instruction	Interpretation
do	; Display overshoot
Controller return:	
OS=0001	; Overshoot = 1 count

dO: Display actuator orientation

Syntax:
dO

Arguments:
None

Related Command:
O – Set actuator orientation

Example:

Instruction	Interpretation
dO	; Display actuator orientation
Controller return:	
OR=0000	; Orientation = 0 (Horizontal)

dP: Display proportional gain KP

Syntax:
dP

Arguments:
None

Related Command:
K – Set proportional gain KP

Example

Instruction	Interpretation
dP	; Display KP gain
Controller return:	
KP=0018	; KP gain = 018h

dp: Display program running status

Syntax:
dp

Arguments:
None

Related Command:
dL – Display executing program & line number

Example:

Instruction	Interpretation
dp	; Display program running flag
Controller return:	
PE=1111	; Program running
	Or
PE=0000	; Program NOT running

dq: Display data collection sampling rate

Syntax:
dq

Arguments:
None

Related Command:
S – Set data collection sampling rate
R – Set data collection type
dr –Display data collection type
Example:

Instruction	Interpretation
dq	; Display sampling rate
Controller return:	
DQ=0010	; Data sampling rate = 16 ms

dQ: Display controller fault history

Syntax:
dQ

Arguments:
None

Related Command:
dF – Display fault
eQ – Clear fault history

Example:

Instruction	Interpretation
dQ	; Display controller fault history
Controller return:	
FH=0023333357322534;	Fault history, least significant digit (4) is ; the most recent fault.
PH=FF02302654120001	; Latest bwd pos fault (4) occurs in prog#1

Fault Message (Code)	Description	Action
ADC conversion fault (01)	DSP ADC voltage reference LOW.	Contact Tol-O-Matic
Brake current fault (02)	Exceed brake max current	1. Check brake connection 2. Measure brake resistance 1.0 in brake - 8 ohm 1.5 in brake - 4 ohm 3. A jumper should be installed on controller board J5 for 1.5 in brake
Position fault (03)	Unable to position	1. Decrease integral gain Ki 2. Increase brake torque decel gain 3. Increase position window 4. Check brake in-line fuse and coil resistance
Reverse position limit (04)	Exceed reverse position limit	Modify reverse position limit setting
Forward position limit (05)	Exceed forward position limit	Modify forward position limit setting
Maximum speed fault (06)	Exceed maximum speed setting	1. Check commanded speed 2. Increase max speed setting
Speed following error (07)	Exceed speed following error setting	1. Increase speed gain Kv 2. Increase speed error setting
Enable fault (08)	Controller NOT enabled	1. Check enable input wiring 2. See Chapter 5.7 input wiring
Move profile timeout (09)	(Actual move time - commanded move time) >= timeout setting	1. Check and adjust tuning para. KV, KI, or KT 2. Increase profile timeout setting
Current following fault (0A)	Brake current following error	1. Check brake wire connection 2. Check brake coil resistance

dr: Display data collection type

Syntax:
dr

Arguments:
None

Related Command:
S – Set data collection sampling rate
R – Set data collection type
dq – Display data collection sampling rate

Example:

Instruction	Interpretation
dr	; Display data collection type
Controller return:	
DT=0000	; Data type = 0 (position)

dR: Display program execution trace status

Syntax:
dR

Return Data:

BIT #	F-B	A	9	8	7	6	5	4	3	2	1	0
Desc	Rsvd	LCD	Fault	Trace	Feed Sensor	Home	Jog	Tune	Prog Run	In-Pos	Move	Servo

Bit 0: 1-Servo loop running
Bit 1: 1-Move flag ON
Bit 2: 1-Carrier in position
Bit 3: 1-Program running
Bit 4: 1-Tuning
Bit 5: 1-Jogging
Bit 6: 1-Homing
Bit 7: 1-Feed to sensor/input
Bit 8: 1-Trace mode ON
Bit 9: 1- Fault
Bit A: 1- LCD prompt message
Bit F~B: Reserved

Example:

Instruction	Interpretation
dR	; Display controller status
Controller return:	
CS=0101	; Servo loop and program trace are ON

ds: Display commanded speed

Syntax:
ds

Arguments:
None

Related Command:
V – Set commanded speed

Example:

Instruction	Interpretation
ds	; Display commanded speed
Controller return:	
CS=00004000	; Commanded speed = 16384 counts/sec

dS: Display actual maximum speed

Syntax:
dS

Arguments:
None

Related Command:
V – Set commanded speed
dv – Display actual speed

Example:

Instruction	Interpretation
dS	; Display maximum carrier speed
Controller return:	
MS=00004800	; Actual max. speed = 18432 counts/sec

dt: Display deceleration torque constant KT

Syntax:
dt

Arguments:
None

Related Command:
G – Set deceleration torque constant KT

Example:

Instruction	Interpretation
dt	; Display KT
Controller return:	
BT=0033	; Brake torque = 033h

dU: Display user unit

Syntax:
dU

Arguments:
None

Related Command:
U – Set user unit

Example:

Instruction	Interpretation
dU	; Display user unit
Controller return:	
UU=0000	; User unit = 0 (inch)

dv: Display actual speed

Syntax:
dv

Arguments:
None

Related Command:
V – Set commanded speed
dS – Display actual maximum speed

Example:

Instruction	Interpretation
dv	; Display actual speed
Controller return:	
SP=00004000	; Actual speed = 16384 counts/sec

dV: Display velocity gain KV

Syntax:
dV

Arguments:
None

Related Command:
Y – Set velocity gain KV

Example:

Instruction	Interpretation
dV	; Display velocity gain KV
Controller return:	
KV=0005	; KV gain = 5

dw: Display position repeatability

Syntax:
dw

Arguments:
None

Related Command:
de – Display position error

Example:

Instruction	**Interpretation**
dw	; Display position repeatability
Controller return:	
SE=0006	; Position repeatability = 6 counts

dx: Display program pause by inputs status

Syntax:
dx

Arguments:
None

Related Command:
z – Halt program execution
Z – Resume program execution

Example:

Instruction	Interpretation
dx	; Display program pause by inputs status
Controller return:	
PI = 10000004	; Program pause by terminal command ; Program will pause when input 4 is LOW or,
PI = 200A0003	; Program pause by inputs 1 & 2 ; Pause when both input 1 & 2 are LOW

Note:

Bit 6~0:	Input assignment, 01h~07h for single input, 00h~07Fh for multiple inputs	
Bit 15~7:	Reserved	
Bit 19~16:	Input type,	0000b – single input LOW 0001b – single input HIGH 1000b – multiple inputs BYTE 1100b – multiple inputs &_HI 1010b – multiple inputs &_LO
Bit 27~20:	Reserved	
Bit 31~28:	Pause status,	0000b – not paused 0001b – pause by terminal command, 'z' or 'zx' 0010b – paused by inputs 0011b – paused by inputs & terminal command

dz: Display firmware revision information

Syntax:
dz

Arguments:
None

Related Command:
None

Example:

Instruction	**Interpretation**
dz	; Display firmware revision message
Controller return:	
PrecisionAire Ver.1.26	

D: Set in-position valve output

Syntax:
D [Flag]

Arguments:
Flag = 0, set both valve coils de-energized when in-position
Flag = 1, set both valve coils energized when in-position

Related Command:
dD – Display in-position valve status

Example:

Instruction	**Interpretation**
D1	; Set both valves energized when in-position

e: Jog carriage

Syntax:
e [Flag]

Arguments:
Flag = 0, jog backward (away from brake)
Flag = 1, jog forward (toward brake)

Related Command:
x – Stop motion

Example:

Instruction	Interpretation
s	; Activate servo loop
e1	; Jog carriage forward
x	; Stop jogging

eQ: Clear fault history

Syntax:
eQ

Arguments:
None

Related Command:
dQ – Display fault history

Example:

Instruction	Interpretation
eQ	; Clear fault history
dQ	; Display fault history
Controller return:	
FH=0000000000000000	; Fault code history
PH=FFFFFFFFFFFFFFFF	; Program # history

E: Define encoder position

Syntax:
E [HEX_32]

Arguments:
HEX_32 is a 32-bit signed encoder position data in hexadecimal format.

Related Command:
P – Display position information

Example:

Instruction	Interpretation
x	; Stop motion
E00001000	; Define current position as 4096 counts

fV: Set forward valve ON

Syntax:
fV

Arguments:
None

Related Command:
bV – Set backward valve ON
H – Set in-position holding torque
FS – Feed to sensor

Example:

Instruction	Interpretation
x	; Stop servo loop
H0010	; Set brake current to 010h
fV	; Set forward valve ON, carrier moves
	; toward brake

FS: Feed to Sensor

Syntax:
FS [Data 1] [Data 2]

Description:
The feed to sensor command combines jog and wait for inputs functions to move carrier forward or backward (specified in Data 1) until a specified input status is reached. When an input is detected, PAS controller will set brake current fully, supply pressurized air on both sides of the cylinder and then drop brake current down to specified in-position holding torque.

Arguments:

Data 1:	16-bit data to select one of the following input type and move direction. 0000h – Move backward until specified input LOW 0001h – Move backward until specified input HIGH 1000h – Move forward until specified input LOW 1001h – Move forward until specified input HIGH

Data 2:	Specify one of 7 inputs or input binary combination in hexadecimal format Single input - Data 2 = 1 to 7 Multiple inputs – Data 2 = 00h to 7Fh

Related Command:
fV – Set forward valve ON
bV – Set backward valve ON

Example:

Instruction	Interpretation
FS0001	; Keep moving carriage backward until ; input 1 goes LOW
FS10010007	; Move carriage forward until ; input 7 goes HIGH
FS10080037	; Feed forward until input 4 & 7 LOW ; and input 1,2,3,5 & 6 HIGH
FS000A0037	; Feed backward until input 1,2,3,5 & 6 ; LOW and ignore the other inputs
FS100C0037	; Feed forward until input 1,2,3,5 & 6 HIGH ; and ignore the other inputs

Note:
Position repeatability for feed to sensor move may depend on move speed, load, sensor and valve response time, supplied air pressure and servo settings.
Maximum move speed is limited to activated input duration that can be sensed by input. PAS requires a minimum input duration of 10 ms.

g: Go, begin move

Syntax:
g

Arguments:
None

Related Command:
x – Stop motion

Example:

Instruction	Interpretation
s	; Activate servo loop
P00001000	; Set commanded position 4096 counts
g	; Move to commanded position

G: Set deceleration torque constant KT

Syntax:
G [HEX_16]

Arguments:
HEX_16 is a 16-bit unsigned deceleration torque constant in hexadecimal format.

Related Command:
dt – Display deceleration torque constant KT

Example:

Instruction	**Interpretation**
G0040	; Set KT gain to 64 = 040h

h: Carriage home

Syntax:
h [Flag]

Arguments:
Flag = 0, home backward (away from brake)
Flag = 1, home forward (toward brake)

Related Command:
x – Stop motion

Example:

Instruction	Interpretation
s	; Activate servo loop
h0	; Home backward

H: Set in-position holding torque

Syntax:
H [HEX_16]

Arguments:
HEX_16 is a 16-bit unsigned in-position holding torque in hexadecimal format.

Related Command:
dh – Display in-position holding torque

Example:

Instruction	**Interpretation**
H001B	; Set in-position holding torque to 01Bh

I: Set integral gain KI

Syntax:
I [HEX_16]

Arguments:
HEX_16 is a 16-bit unsigned integral gain in hexadecimal format.

Related Command:
dI – Display integral gain KI

Example:

Instruction	Interpretation
I0001	; Set integral gain KI to 01h

i: Set encoder monitoring

Syntax:
i

Arguments:
None

Related Command:
dE - Display encoder monitoring

Example:

Instruction	Interpretation
	;

j: Load program from EEPROM to RAM

Syntax:
j [NUM]

Arguments:
NUM is a 4-bit unsigned program number in hexadecimal format. NUM = 0 to 9.

Related Command:
J – Save program to EEPROM   l – List program

Example:

Instruction	Interpretation
j4	; Load program #4 to RAM
l	; List program

J:Save program to EEPROM

Syntax:
J [NUM]

Arguments:
NUM is a 4-bit unsigned program number in hexadecimal format. NUM = 0 to 9.

Related Command:
j – Load program from EEPROM to RAM

Example:

Instruction	Interpretation
J4	; Save program to EEPROM program #4

k: Set actuator stroke length

Syntax:
k [HEX_31]

Arguments:
HEX_31 is a 31-bit unsigned stroke data in encoder counts in hexadecimal format.

Related Command:
dk – Display actuator stroke length

Example:

Instruction	Interpretation
k00008000	; Set stroke length to 08000h

K: Set proportional gain KP

Syntax:
K [HEX_16]

Arguments:
HEX_16 is a 16-bit unsigned proportional gain in hexadecimal format.

Related Command:
dP – Display proportional gain KP

Example:

Instruction	Interpretation
K0018	; Set KP=018h

l: List program

Syntax:
l

Arguments:
None

Related Command:
j – load program from EEPROM to RAM

Example:

Instruction	Interpretation
l	; List program
Controller return:	
HM0	; Home backward
FP2000	; Feed to position 02000h
EN	; End of program

L: Download program / Configure software limit

Syntax:
L [Flag]

Arguments:
None – Download program to RAM
Flag = 0, disable software limits
Flag = 1, enable software limits

Related Command:
Control+C (^C) – End program download
dm – Display software limits

Example:

Instruction	Interpretation
L	; Begin load program to RAM
HM0	; Home backward
FP2000	; Feed to position 02000h
EN	; End of program
^C	; End program download
L1	; Enable software limits
dm	; Display software limits
Controller return:	
RL=FFFFF000	; Reverse position limit = -4096 counts
FL=00004000	; Forward position limit = 16384 counts
SL=00010000	; Maximum speed limit = 65536 counts/sec
SW=1C00	; Speed following error = 7168 counts/sec
LE=0001	; Software limits enabled

m: Select variable

Syntax:
m [HEX_16]

Arguments:
HEX_16 is a 16-bit unsigned variable number in hexadecimal format. Use 8000-803F for variable #0 to #63.

Related Command:
dM – Display selected variable data
M – Set variable name or data

Example:

Instruction	Interpretation
m8000	; Select variable #0

M: Set variable name or data

Syntax:
M [HEX], M*, Mx [TEXT]

Arguments:
HEX is a 16 or 32-bit unsigned data in hexadecimal format.
TEXT is a maximum 8-character string for variable name.

Related Command:
dM – Display selected variable name & data
m – Select variable
yV – Save variables to EEPROM

Example:

Instruction	Interpretation
m803F	; Select variable #63
M1000 1234	; Set 10001234h to selected variable
MxPOS64	; Name double variable #63 as “POS64”
M*	; Clear all variables to 0

n: Set output

Syntax:
n [HEX_16]

Arguments:
HEX_16 is the binary combination of 4 output channels as illustrated below. HEX_16 = 00h to 0Fh

Related Command:
di – Display input/output status

Example:

Instruction	Interpretation
n000A	; Set output 1 & 3 LOW and 2 & 4 HIGH
	; 0Ah = 01010b

N: Set load/weight

Syntax:
N [HEX_16]

Arguments:
HEX_16 is a 16-bit unsigned load/weight in hexadecimal format.

Related Command:
dl – Display load/weight

Example:

Instruction	Interpretation
N7	; Set load/weight to 7

o: Set actuator bore size

Syntax:
o [Flag]

Arguments:
Flag = 0, 1-in bore
Flag = 1, 1.5-in bore

Related Command:
db – Display actuator bore size

Example:

Instruction	Interpretation
o0	; Set actuator bore 1-in

O: Set actuator orientation

Syntax:
O [Flag]

Arguments:
Flag = 0, horizontal
Flag = 1, vertical

Related Command:
dO – Display actuator orientation

Example:

Instruction	**Interpretation**
O1	; Actuator mounted vertically

p: Display position information

Syntax:
p

Arguments:
None

Related Command:
P – Set commanded position

Example:

Instruction	**Interpretation**
p	; Display position information
Controller return:	
CP=00001000	; Commanded position = 4096 counts
AP=00001001	; Actual position = 4097 count

P: Set commanded position

Syntax:
P [HEX_32]

Arguments:
HEX_32 is a 32-bit signed position data in encoder counts in hexadecimal format.

Related Command:
p – Display position information

Example:

Instruction	**Interpretation**
P00001000	; Set commanded position 01000h

q: Quit motion program

Syntax:
q

Arguments:
None

Related Command:
r – Run motion program
x – Stop motion

Example:

Instruction	**Interpretation**
q	; Quit motion program

Q: Controller soft reset

Syntax:
Q

Arguments:
None

Related Command:
None

Example:

Instruction	**Interpretation**
Q	; Controller reset

Note: Please allow two seconds for internal controller reset. Sending serial command during controller reset may cause serial port over-run error. If serial communication is not established, please recycle power or press reset button to reset controller.

r: Run motion program

Syntax:
r

Arguments:
None

Related Command:
q – Quit motion program
x – Stop motion

Example:

Instruction	Interpretation
j1	; Load program #1 to RAM
s	; Activate servo loop
r	; Run program #1

R: Set data collection type

Syntax:
R [Flag]

Arguments:
Flag = 0, no data collection
Flag = 1, position data
Flag = 2, actual speed data
Flag = 3, commanded speed
Flag = 4, current feedback data
Flag = F, command & actual speed data

Related Command:
dr – Display data collection type

Example:

Instruction	Interpretation
R1	; Collect position information

s: Activate servo loop

Syntax:
s

Arguments:
None

Related Command:
X – Stop servo loop

Example:

Instruction	Interpretation
s	; Activate servo loop
P1000	; Set commanded position 01000h
g	; Move to commanded position

S: Set data collection sampling rate

Syntax:
S [Rate_flag]

Arguments:
Rate_flag = 3 to 9.
Sampling rate = 2^Rate - flag (ms)

Related Command:
dq – Display data collection sampling rate

Example:

Instruction	Interpretation
R5	; Set sampling rate 2^5=32 ms

t: Display motion profile time

Syntax:
t

Arguments:
None

Related Command:
TA – Set acceleration time
TD – Set deceleration time
TO – Set profile timeout

Example:

Instruction	Interpretation
t	; Display motion profile time
Controller return:	
TA=0080	; Acceleration time = 128 ms
TC=0100	; Constant speed time = 256 ms
TD=0064	; Deceleration time = 100 ms
TO=0400	; Profile time out = 1024 ms

TA: Set acceleration time

Syntax:
TA [HEX_16]

Arguments:
HEX_16 is a 16-bit unsigned acceleration time in millisecond in hexadecimal format.

Related Command:
t – Display motion profile time

Example:

Instruction	Interpretation
TA0080	; Set acceleration time 080h = 128 ms

TD: Set deceleration time

Syntax:
TD [HEX_16]

Arguments:
HEX_16 is a 16-bit unsigned deceleration in millisecond in hexadecimal format.

Related Command:
t – Display motion profile time

Example:

Instruction	Interpretation
TD0064	; Set acceleration time 064h = 100ms

TO: Set profile timeout

Syntax:
TO [HEX_16]

Arguments:
HEX_16 is a 16-bit unsigned profile timeout in millisecond in hexadecimal format.

Related Command:
t – Display motion profile time

Example:

Instruction	Interpretation
TO0400	; Set profile timeout 400h = 1024 ms

TR: Set program execution trace mode

Syntax:
TR [Flag]

Arguments:
Flag = 0, program execution trace OFF
Flag = 1, program execution trace ON

Related Command:
dR – Display program execution trace status

Example:

Instruction	Interpretation
TR1	; Set trace mode ON

u: Upload collected data

Syntax:
u [Flag]

Arguments:
Flag = 0, 32-bit data (position, speed, commanded speed)
Flag = 1, 16-bit data (current feedback)

Related Command:
R – Set data collection type
S – Set data collection sampling rate
UT – Upload tuning speed data

Example:

Instruction	Interpretation
u1	; upload 16-bit data

uT: Upload tuning speed data

Syntax:
uT

Arguments:
None

Related Command:
u – Upload collected data

Example:

Instruction	Interpretation
uT	; upload 32-bit tuning speed data

U: Set user unit

Syntax:
U [Flag]

Arguments:
Flag = 0, inch
Flag = 1, mm

Related Command:
du – Display user unit

Example:

Instruction	Interpretation
U1	; Set user unit to mm

v: Set maximum speed limit

Syntax:
v [HEX_32]

Arguments:
HEX_32 is the maximum speed in counts per second in hexadecimal format.

Related Command:
dm – Display software limits

Example:

Instruction	Interpretation
v00010000	; Set max. speed limit to 10000h counts/sec

V: Set commanded speed

Syntax:
V [HEX_32]

Arguments:
HEX_32 is the commanded speed in counts per second in hexadecimal format.

Related Command:
ds – Display actual speed

Example:

Instruction	Interpretation
V00008000	; Set commanded speed to 8000h counts/sec

w: Set speed following error limit

Syntax:
w [HEX_16]

Arguments:
HEX_16 is the speed following error in counts per second in hexadecimal format.

Related Command:
dm – Display software limits

Example:

Instruction	Interpretation
w1c00	; Set speed following error limit to 1c00h ; counts/sec

W: Set position repeatability

Syntax:
W [HEX_16]

Arguments:
HEX_16 is the position repeatability in counts in hexadecimal format.

Related Command:
dw – Display position repeatability

Example:

Instruction	Interpretation
W0005	; set position repeatability to +/- 5 counts

x: Stop motion

Syntax:
x

Arguments:
None

Related Command:
e – Jog carriage
g – Go, begin move
h – Carriage home
r – Run motion program

Example:

Instruction	**Interpretation**
x	; Stop motion

X: Stop servo loop

Syntax:
X

Arguments:
None

Related Command:
s – Activate servo loop

Example:

Instruction	Interpretation
X	; Stop servo loop

y: Save controller settings

Syntax:
y

Arguments:
None.

Related Command:
None.

Example:

Instruction	**Interpretation**
y	; Save controller settings to EEPROM

yV: Save variables to EEPROM

Syntax:
yV

Arguments:
None.

Related Command:
dM – Display selected variable data
m – Select variable
M – Set variable name or data

Example:

Instruction	**Interpretation**
yV	; Save variable data and names to EEPROM

Y: Set velocity gain KV

Syntax:
Y [HEX_16]

Arguments:
HEX_16 is the velocity gain in hexadecimal format.

Related Command:
dV – Display velocity gain KV

Example:

Instruction	**Interpretation**
Y0005	; Set velocity gain KV to 5

z: Halt program execution after finishing current command

Syntax:
z

Arguments:
None.

Related Command:
Z – Resume program execution

Example:

Instruction	**Interpretation**
z	; Halt program execution after finishing current command

zx: Halt program execution and stop motion immediately

Syntax:
zx

Arguments:
None.

Related Command:
Z – Resume program execution

Example:

Instruction	Interpretation
zx	; Halt program execution and stop move
	; right away
Z	; Resume program and continue unfinished
	; move

Z: Resume program execution

Syntax:
Z

Arguments:
None.

Related Command:
z – Halt program execution

Example:

Instruction	Interpretation
Z	; Resume program execution

^C: End Program Download

Syntax:
Control + C

Arguments:
None.

Related Command:
L– Download program to RAM

Example:

Instruction	Interpretation
L	; Download program begins
	; Send text motion program here
^C	; End program downloadzzz

TOL-O-MATIC, INC.
3800 County Road 116
Hamel, MN 55340
763.478.8000 Telephone
800.328.2174 Toll Free
763.478.8707 Fax
http://www.tolomatic.com

PrecisionAire™