Software User Manual

IDEA™ Drive

Version 4.X Firmware

Continued on next page>

< Continued from previous page

AMETEK ADVANCED MOTION SOLUTIONS ©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Revision History

Date	Description
January 2010	Initial version
February 2010	Save debug output Goto if updated for outputs Interrupt and encoder configuration update
January 2011	Added detail to the behavior of interrupts Added detail to the IO wiring diagram Stand alone IO wiring diagram RS-485 wiring Communications features
May 2011	Added RS-485 Pin descriptions Added minimum time between resets
July 2011	Removed model specific information
August 2011	Added information about command strings
September 2011	Updated for 2.0 firmware and software changes Added introduction and part number information
December 2011	Clarified Goto if explanation Updated
March 2013	Clarified part number entry
January 2015	Revised manual for PBL4850E IDEA™ Drive
July 2017	Revised manual for PBL7090E IDEA™ Drive

Introduction / Scope

This manual is intended to provide information on using the IDEA™ Drive from AMETEK. For information pertaining to installation and wiring of a specific IDEA™ Drive model, reference the appropriate hardware manual, available at **www.HaydonKerkPittman.com**

IDEA™ Drive and Software Basics

Thank you for choosing the IDEA™ Drive from AMETEK. The AMETEK Brushless IDEA™ Drive and associated software are a complete package for the easy control of brushless motors within linear and rotary systems. This solution provides advanced features for both immediate execution, as well as user defined motion programs in a user-friendly way. Basic motion commands are implemented through intuitively named buttons, and each button and input field is further clarified through tooltips.

The IDEA™ Drive contains a sophisticated operating system for real time control, a motion profile generator, an advanced sinusoidal servo motor controller, and flexible I/O and communication options, all residing in one integrated package. The IDEA™ Drive permits true distributed and autonomous control of BLDC motors, without the need for external automation devices such as PLC's.

The drive contains an operating system which supports powerful real-time control features such as nested pre-emptive vectored interrupts from multiple sources, real time high speed polling of inputs, and autonomous execution of sophisticated pr -programmed motion profiles.

The flexible motion profile generator supports both trapezoidal and sinusoidal "s" curve velocity motion profiles, with independent settings for acceleration rate, deceleration rate, maximum velocity, as well as separate current limit settings for each segment of the move profile. Failsafe limit interrupts provide protection against abnormal operating conditions or equipment failure at any time during the motion profile execution.

The state of the art sinusoidal vector servo motor control provides smooth, quiet, accurate, and responsive motion. Precise control of motor current provides reliable and efficient operation of the motor.

Startup

Installing the Application (this is a one-time activity): The IDEA™ Drive software can be downloaded from www.HaydonKerkPittman.com Please perform the following instructions: (**NOTE:** This is to be performed without the actuator being attached via the USB cable).

- Download the IDEA graphic user interface software from the Drives section of website. Double click the executable file to start the installation.
- "Welcome to the IDEA Software setup wizard" will appear on screen.
- "Select installation folder" will appear on screen. You may change the location of the installation if you wish. Click Next.
- "Confirm installation" will appear on screen. Click Next.
- "Installing IDEA Software" will appear on screen. This may take a few moments.
- "Installation complete" will appear on screen. A black function window will also appear behind the "Installation complete" dialogue box. This is normal. Do not close this window, it will close automatically. After a few moments, the "Installation complete" dialogue box will show a "close" button. Click Close.
- Installation is now complete and the IDEA™ icon resides on your desktop and in your start menu under All Programs → Haydon Kerk.

Safety Margin Algorithm

The software uses the motors' characteristics to set safe operating limits during operation. Current and speed limitations are implemented to operate the motor at its continuous operation range. These limitations can be viewed in the menu item View → Motor Characteristics.

Motor manufacturers may vary in the methods of testing for and providing motor specifications, therefore the software algorithm may not meet published motor ratings. Additionally, some user may want to overdrive motors in low duty cycle application or when additional motor cooling

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

methods are implemented. Rather than completely limiting our users, the software instead provides a warning when the limits are exceeded. **NOTE:** It is up to the user to verify current limitation against the motor specifications from the manufacture and ensure the motor is being used under safe operating conditions.

Drive Initialization Default State

The drive defaults to the following state upon drive power up or issuing a "Reset" command:

1. Input simulation is turned off.
2. All outputs are set low.
3. The position counter is set to zero.
4. The hold current is set to zero.
5. All interrupts are disabled
6. Motor parameters previously saved to drive are loaded.
7. If a startup program is selected, that program is now begun.

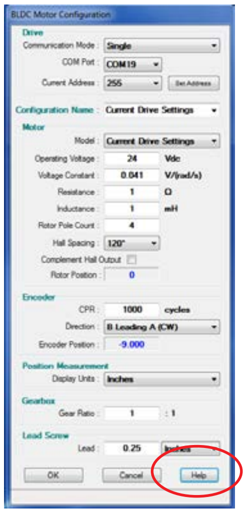
When a program is started, the following occurs:

1. All outputs are set low.
2. The position counter is set to zero.
3. All interrupts are disabled.

Getting Started:

- Apply power to the drive.
- Launch program by double-clicking the IDEA™ icon on your desktop. This brings you to the initial BLDC Motor Configuration pop up window.

  Helpful Hint:
  Click Help for Motor Set Up Guide.



BLDC Motor Configuration

Drive
Communication Mode : Single
COM Port : COM19
Current Address : 255
Configuration Name : Current Drive Settings
Motor
Model : Current Drive Settings
Operating Voltage : 24 Vdc
Voltage Constant : 0.041 V/(rad/s)
Resistance : 1 Ω
Inductance : 1 mH
Rotor Pole Count : 4
Hall Spacing : 120°
Complement Hall Output ☐
Rotor Position : 0
Encoder
CPR : 1000 cycles
Direction : B Leading A (CW)
Encoder Position : -9.000
Position Measurement
Display Units : Inches
Gearbox
Gear Ratio : 1 : 1
Lead Screw
Lead : 0.25 Inches

Drive

Select the Single Communication mode.

Configuration Name

Enter a convenient name to select this system configuration for future reference. Example "My Test Setup".

Motor

Reference Motor Setup Guide for common Pittman BLDC motor configurations by clicking the Help button located at lower right corner of *BLDC Motor Configuration* pop up menu.

Enter the motor characteristics:

a. From the drop-down menu select the motor model
b. Enter the voltage of power supply used to power the drive
c. Motor voltage constant in units of volts/radians/second
d. Motor resistance per phase in units of ohms
e. Motor inductance per phase in units of millihenries
f. Motor magnetic pole counts: 4,6,8, & 10 pole motors are supported
g. Hall spacing 120 or 60°.
h. Complement hall output defaults unchecked, depending on the motors this will need to be checked.
i. Click "OK" to exit screen

Motor configuration parameters will be permanently stored in the drive memory. The software will save up to 5 user motor configurations, and loads the last configuration previously entered.

Position Measurement Unit Selection

- Select between measurement units of inches or millimeters when working with a motor which is part of a linear system.
  1. Enter the lead of the screw of the linear system and select the native units of the screw; in the lead screw section.
- Select between measurement units of Revolutions per Second or Revolutions per Minute when working with a motor which is part of a rotary system.
  NOTE: The lead input parameters will be eliminated from the screen when selecting Revolutions per Second or Revolutions per Minute.

Encoder

Enter the encoder cycles per revolution (CPR) and direction: A leading B or B leading A., which is explained in more detail in the encoder section, page 9.

Gear box

Select the gear ratio if there is not gear box it will automatically be set to a1 to 1 ratio. Click OK.

The "Realtime" display now appears. This mode allows the program to execute immediate actions and motion commands such as extend, retract, go at speed, etc. "Realtime" display is only active if the drive is

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

actively communicating with the PC host. The GUI will default to display "Program" mode if the drive is not actively communicating with the host PC . If the drive is not actively communicating with the host PC, ensure the drive is powered and the USB communication cable is attached between the drive and host PC. Removing and plugging the USB cable into the host PC may also help establish active communications with the drive.

GUI Interface Displays Overview

The IDEA™ Drive GUI provides two distinct application screens. The "Program" screen is used to create and download motion programs to the drive. The "Realtime" screen is used to debug application programs downloaded to the drive, as well as to immediately and directly execute individual motion commands.

Realtime Mode

Below is a screenshot of the User interface in Realtime mode. This mode is only available when a drive is connected and actively communicating with the host PC.

In Realtime mode, each command executed from the window section labeled "Motion" elicits an immediate action from the drive and motor.

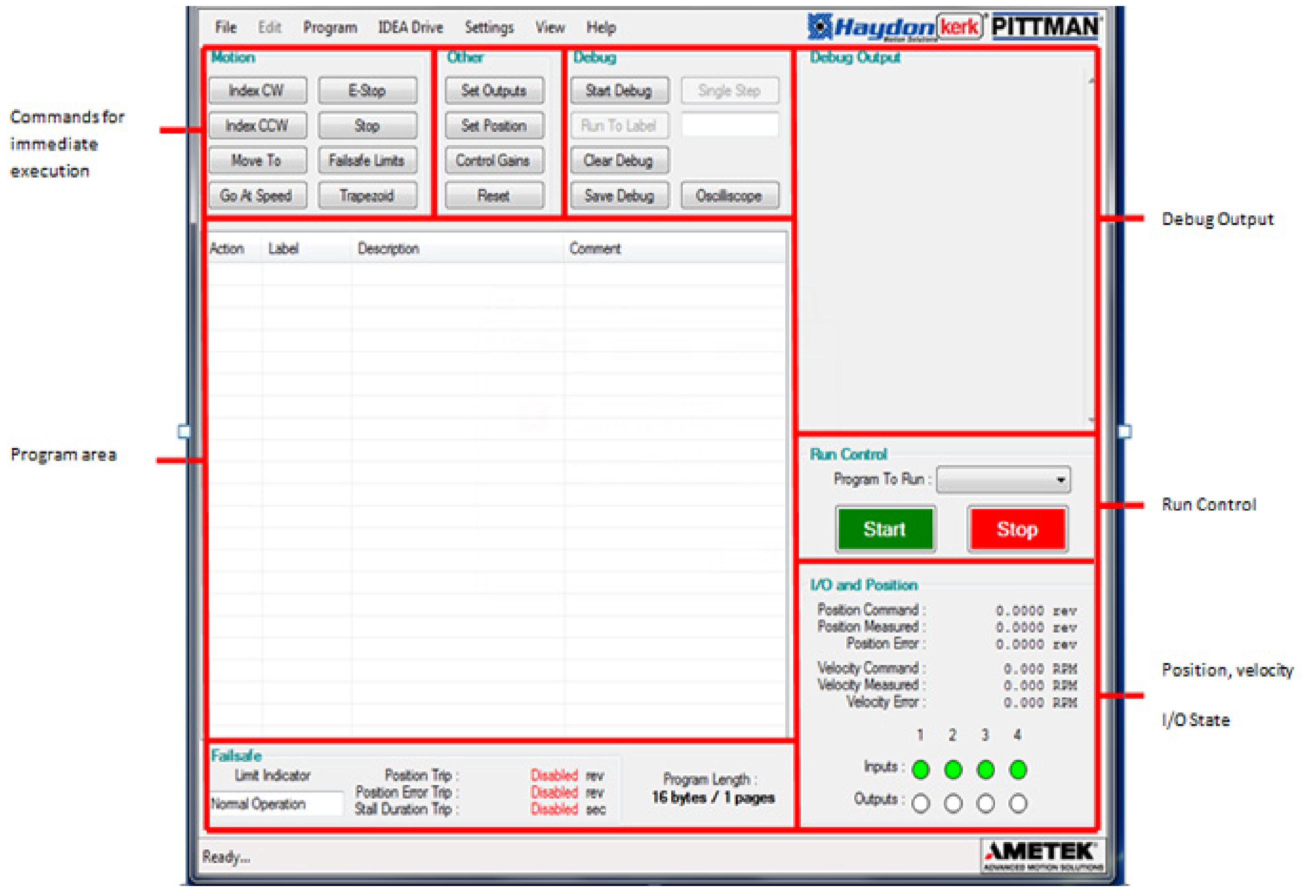
The program area can display any program currently residing on the drive. These programs can be run or aborted using the "Run Control" section.

The I/O and position section provides constant feedback from the drive pertaining to the current state of the drive. This area can also be used to control the state of the outputs, as well as the inputs if in simulation mode.

The IDEA software "Debug Mode" provides a powerful debugging feature to help troubleshoot motion programs that reside on the drive in drive memory.

The debugger consists of two distinct modes of operation:

- "Single-stepping": This mode permits execution of the program one program line at a time, halting program execution after each program line executes. Each step of program line execution is recorded and displayed in a log file for future reference.



Position Command :	0.0000 rev
Position Measured :	0.0000 rev
Position Error :	0.0000 rev
Velocity Command :	0.000 RPM
Velocity Measured :	0.000 RPM
Velocity Error :	0.000 RPM

1 2 3 4
Inputs : ● ● ● ●
Outputs : ○ ○ ○ ○

Limit Indicator: Normal Operation
Position Trip : Disabled rev
Position Error Trip : Disabled rev
Stall Duration Trip : Disabled sec
Program Length : 16 bytes / 1 pages

Screenshot of the User interface in Realtime Mode.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

- “Run To Label”: this mode acts as a break point, providing the ability to quickly run to a specified portion of the program and halt program execution.

Both modes can be used sequentially to assist in debugging a motion program. For example, “Run To Label” can be used to halt program operation at a specific line of code of interest. “Single Stepping” mode can then be used to step thru the code from this point of interest one program line at a time.

Program Mode

This mode is used to develop sophisticated motion profile application programs.

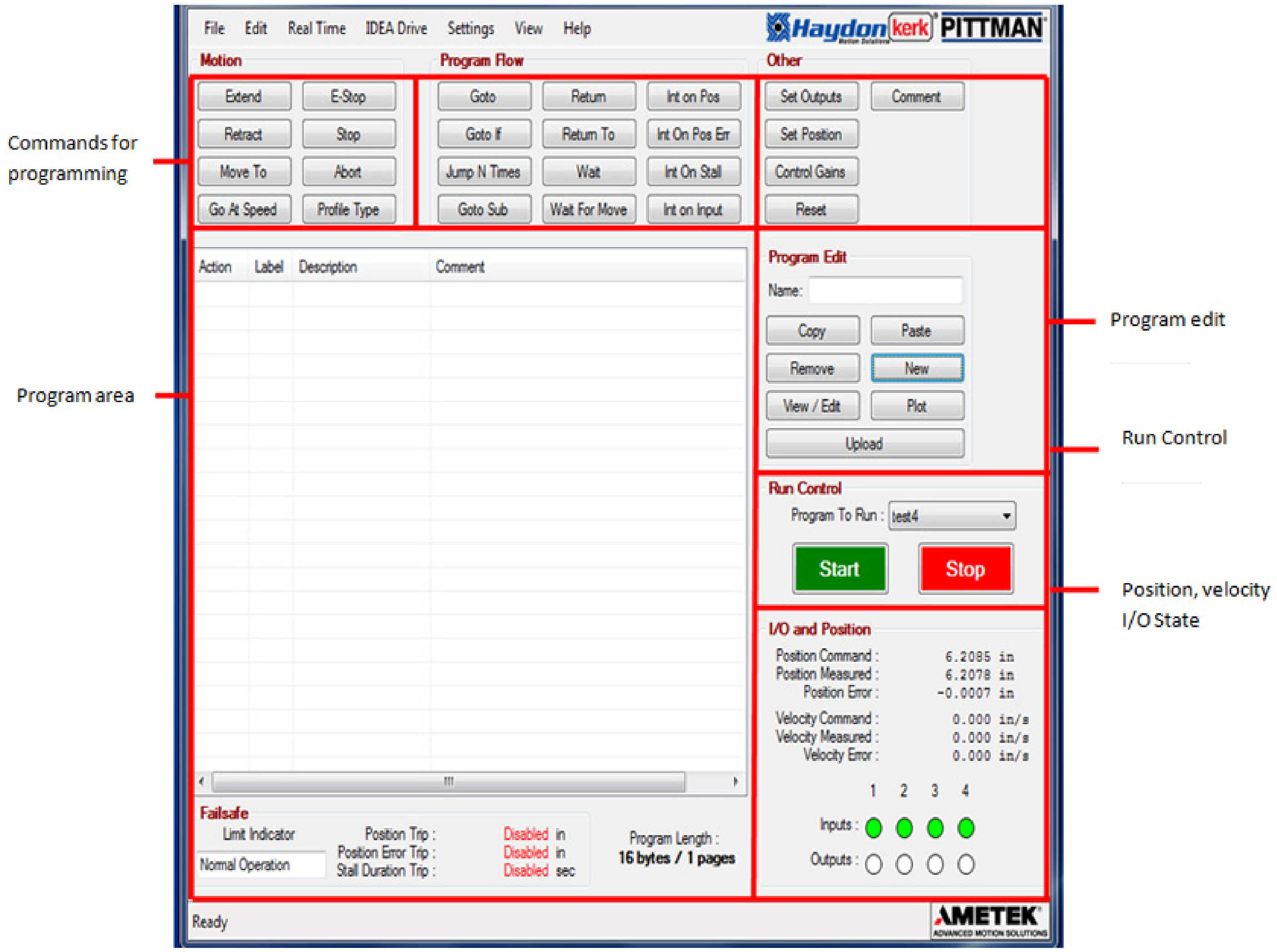
Below is a screenshot of the user interface in Program Mode. This mode is always available regardless if a drive is actively communicating with a host PC.

The I/O and position section provides continuous feedback from the drive pertaining to the current state of the drive. This area can also be used to control the state of the outputs, as well as the inputs if in simulation mode.

The “Run Control” area can be used to execute or stop any program on the drive.

The “Program Edit” area contains buttons used to edit the program as well as gather information about individual commands. The Upload button is also in this area.

The “Program Area” contains the command that make up the program currently being viewed or edited. Commands in the program are executed sequentially, starting at line 1, and moving onto line 2 and so on, unless a branching command is used, which would send execution instead to a specified label. The different commands available are covered in depth on the following page.



Program To Run : test4

Position Command :	6.2085 in
Position Measured :	6.2078 in
Position Error :	-0.0007 in
Velocity Command :	0.000 in/s
Velocity Measured :	0.000 in/s
Velocity Error :	0.000 in/s

	1	2	3	4
Inputs :	●	●	●	●
Outputs :	○	○	○	○

Limit Indicator: Normal Operation

Position Trip :	Disabled	in
Position Error Trip :	Disabled	in
Stall Duration Trip :	Disabled	sec

Program Length : 16 bytes / 1 pages

Ready

Screenshot of the User interface in Program Mode.

The size of the current program is displayed in both bytes and pages in the "Program Length" area. Each page is 1024 bytes long, and the number of pages used is always rounded up.

The IDEA™ Drive Menu Items

- **File**: This menu gives access to functions for manipulation of program files and the User interface itself.
    1. **New**: Clears the current program so a new program can be written.
    2. **Open**: Opens a previously saved program file.
    3. **Save**: Saves the current program to a file.
    4. **Encryption**: Encrypts file.
    5. **Add File:** Adds a previously saved program file to the end of the current program.
    6. **Recover Autosave**: Opens the most recently autosaved file.
    7. **Print**: Prints the current program.
    8. **Exit**: Closes the user interface.
- **Edit**: This menu gives access to functions for manipulating the current program.
    1. **Undo**: Restores the program to what it was before the last action.
    2. **Redo**: Restores the program to what it was before an undo.
    3. **Cut**: Removes a selected line or lines from the program and copies them to be pasted later.
    4. **Copy**: Copies a selected line or lines to be pasted later.
    5. **Paste**: Inserts previously copied lines into a program.
    6. **Select All**: Selects all lines of the program.
- **Real Time/Program**: Used to move between Real time mode and program mode.
- **IDEA™ Drive**: This menu allows access to various drive functions.
    1. **Table of Contents**: Gives a listing of the programs on the drive and their page locations. Also provides a graphical representation of the used space in the drive.
    2. **Startup Program**: Used to choose what program, if any, should begin execution when the drive starts up.
    3. **Simulate Inputs:** This item toggles the drive between using the true input status, or the simulated status through the user interface.
    4. **Adjust Control Loop Gain:** Used to adjust the sensitivity of servo control loop gains.
    5. **Set/Change Password:** Used to configure the password of the drive.
    6. **Restore Factory Defaults:** Removes password protection from the drive, removes all programs on the drive, and restores configuration parameters to default values.
    7. **Firmware Version**: Used to find the firmware version of the drive.
    8. **Update Firmware**: Used to reprogram the drive with the firmware version that was most recent when the user interface was installed.
- **Settings:** This menu gives access to the motor parameters and drive information.
    1. **Drive Type**: select the drive type.
    2. **Configuration**: Motor parameters.
    3. **Show Command String**: Can be selected or unselected. the string will be sent to the drive and it will appear near the bottom of each command.
- **View**: Shows the motor characteristics.
- **Help**: Allows access to information about the user interface and drives.
    1. **About**: Displays a brief description of Haydon Kerk and its products.
    2. **User's Manual**: Displays this manual.
    3. **Communications Manual:** Opens the communications manual.
    4. **Hardware Manuals**: Manuals for individual products.

Features and Concepts

Auto filling of Inputs

The software continually saves user inputs and automatically populates fields whenever possible. This is used to aid the user and speed up the programming process. Initially, on the motor parameter entry screen, the software will save the motor configuration to the windows user settings. The last 5 entered configurations will be saved and the latest configuration will be loaded automatically the next time the software is opened. Additionally, command windows are initially prefilled with recommended values for safe operating conditions of the motor. After a user updates a command, their parameters are saved and subsequently preloaded the next time the command is used.

Tooltips

Tooltips provide the user with additional information about the controls within the software. A tooltip window will be displayed temporarily when the mouse is hovered over controls but not clicked. Tooltips will provide

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

parameter limits for commands when the mouse is hovered over input boxes and functional insight for commands when hovered over buttons.

Unit Conversion

When starting the program, motor parameters and measurement units must be selected. From this information, the number of encoder counts for each command is calculated. This allows the user to write programs in their preferred units rather than having to calculate the number of counts to perform a move. As with any conversion, some rounding error may be present. A lead value is necessary when working with linear units of inches or mm.

View Command String

The IDEA™ Drive can be used without the GUI, by streaming serial commands to it directly from your own programs or devices. To help in the development of these applications, a feature has been created that will show you the string that will be sent for a given command.

To access this feature, select settings → show command string. Once this is checked, when you are editing parameters for commands, the string that would be sent to the drive will appear near the bottom of the window.

For some commands, such as "Goto" and "Goto Sub", there will be values that cannot be displayed, because the location of the program on the drive is not yet known. These values will appear as [Address] and will need to be calculated based upon where in the final program the destination will reside, as well as the start page of the program.

Communications Modes

The Idea Drive supports targeted streaming of motion commands via a RS-485 field bus network. For details on available commands and message formats, reference the *IDEA™ Drive Communications Manual.*

A selection of four distinct communications modes are available for communication over a field bus network. You can change from one of these modes to any other through the "Settings" menu item.

- Offline: In this mode drives are not actively communicating with the host PC, so only the "Program" screen is available. This mode is provided for convenience to permit developing application programs without a drive.

- Single: In this mode, only 1 drive should be communicating with the host PC. This mode is typically used for developing and debugging motion programs which reside on a single drive. It is also used to assign an individual drive a unique field bus address identifier. Both "Realtime" and "Program" screens are available when operating in this communications mode.

- **Addressed Mode**: This mode supports fieldbus connection of multiple drives to the host PC. Each IDEA™ Drive must be assigned its own unique address identifier, which enables them to be a part of a targeted multi-drop fieldbus network. The address range is 0 to 255. Drive addresses attached to the field bus must be unique. Drive addresses must be assigned prior to connecting them to the field bus to avoid address conflicts. Commands can be targeted to an individual drive by referencing the drives assigned address identifier. Reference the *IDEA™ Drive Communications Manual* for detailed information. A drive assigned a unique address identifier can also receive broadcasted commands.

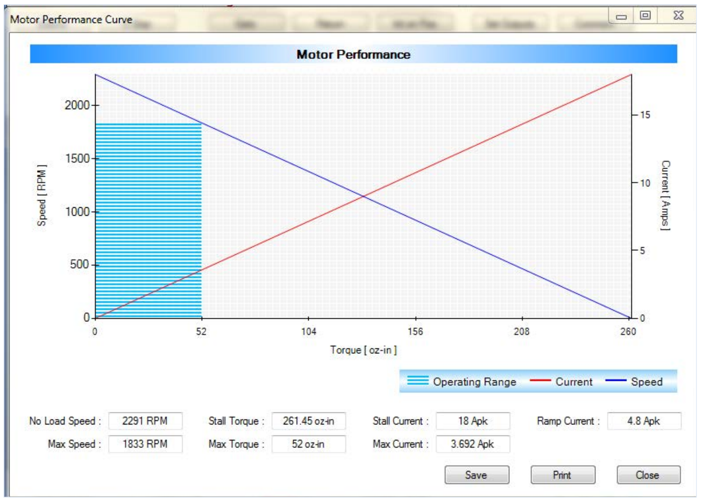
- **Broadcast Mode:** In this mode all drives which are connected to the fieldbus act on the command simultaneously. Only commands which do not generate a response from the drives are permitted to be transmitted across the field bus when operating in broadcast mode. Drive status and feedback information is not available in broadcast mode.

Below is a chart summarizing communications modes and available operating modes.

Modes		Communications			
		Offline	Single	Broadcast	Address
Operating	Program	X	X	-	-
	Realtime	-	X	X	X
	Debug	-	X	-	-

Motor Characteristics

A motor performance curve can be viewed by selecting the Motor Characteristics function under the view menu. The graph is generated by the software using the motor specifications entered by the user at startup. A motor performance curve depicts the limitations of the motor by plotting the generated motor torque versus the speed and current ranges of the motor.

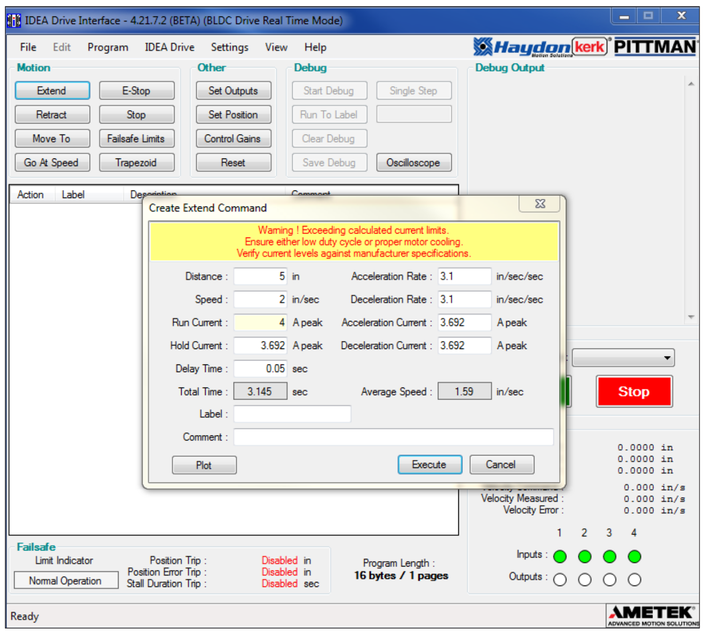


No Load Speed : 2291 RPM
Max Speed : 1833 RPM
Stall Torque : 261.45 oz-in
Max Torque : 52 oz-in
Stall Current : 18 Apk
Max Current : 3.692 Apk
Ramp Current : 4.8 Apk

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

In addition, this graph will also display the motor limitations set by the software. Software limitations are designed with the intention to allow the user to drive the motor at its peak efficiency. Generally, a brushless motors' peak efficiency point falls between 15- 30% of the rated torque and slopes down beyond that. This is also within the area where a motor can operate continuously and is ideal for maximum motor life.

Users can exceed the limitations set by the software. (Recommended for users who are experienced with using brushless motors.) When exceeding limitations, adequate cooling of the motor or a low duty cycle is necessary to prevent damage to the motor. Users will be notified when exceeding limits by a warning message.



Warning ! Exceeding calculated current limits.
Ensure either low duty cycle or proper motor cooling.
Verify current levels against manufacturer specifications.

Distance :	5	in	Acceleration Rate :	3.1	in/sec/sec
Speed :	2	in/sec	Deceleration Rate :	3.1	in/sec/sec
Run Current :	4	A peak	Acceleration Current :	3.692	A peak
Hold Current :	3.692	A peak	Deceleration Current :	3.692	A peak
Delay Time :	0.05	sec			
Total Time :	3.145	sec	Average Speed :	1.59	in/sec
Label :					
Comment :					

```
                    0.0000 in
                    0.0000 in
                    0.0000 in
                     0.000 in/s
Velocity Measured :  0.000 in/s
Velocity Error :     0.000 in/s
```

Failsafe

Limit Indicator	Position Trip :	Disabled	in
Normal Operation	Position Error Trip :	Disabled	in
	Stall Duration Trip :	Disabled	sec

Program Length :
16 bytes / 1 pages

	1	2	3	4
Inputs :	●	●	●	●
Outputs :	○	○	○	○

Maximum Speed

The maximum speed of the drive is limited to the pole count of the motor.

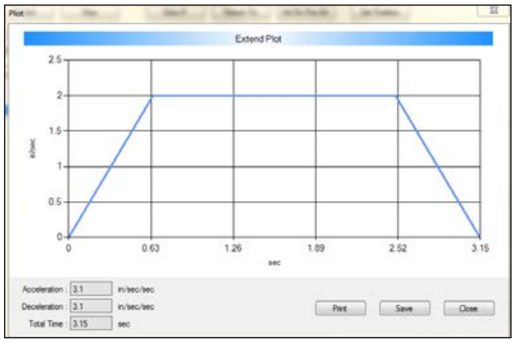
4-pole:	10,000 RPM
6-pole:	6,600 RPM
8-pole:	5,000 RPM
10-pole:	4, 000 RPM

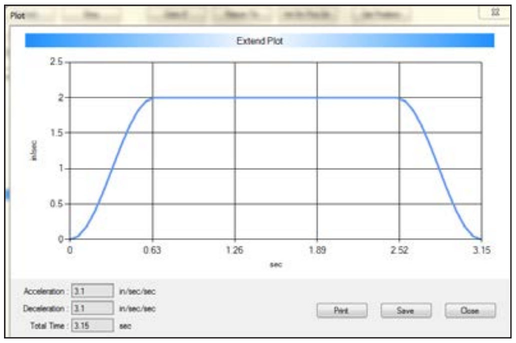
Exceeding these RPM values will result in rapid degradation of motor performance due to poor sinusoidal waveform fidelity.

Motion Profiles

The IDEA™ Drive can perform two types of motion profiles; trapezoidal or s-curve velocity profiles. The velocity profile setting can be toggled within the buttons in real-time or program operating mode. In "Realtime" mode the motion profile type selection is saved onto the drives' non-volatile memory for convenience.

A trapezoidal velocity motion profile is a common industry standard motion profile. The s-curve velocity profile may be desired to smooth out movement by eliminating jerk conditions. A jerk condition occurs when an abrupt change of acceleration rate occurs during the move. For a trapezoidal velocity profile, an abrupt change in acceleration occurs at the beginning and the end of a move, as well as during the transition points from constant velocity. S-curve ramping requires higher peak motor current from the drive to achieve the same move duration, but results in less jerk and smoother operation.





Ramping

The IDEA™ Drive provides easy to use acceleration and deceleration ramps. A motion profile will be created by entering a speed along with the acceleration and deceleration rates. The versatility of the IDEA™ Drive allows users to create motion profiles in any of the units chosen; inches, millimeters, or revolutions. This simplifies generating motion profiles for linear and rotary applications alike. The drive automatically calculates a move profile to approximate the desired parameters. For a picture of what the calculated ramp will look like, use the plot function on any "Extend", "Retract", "Move To", or "Go At Speed" command.

Encoder

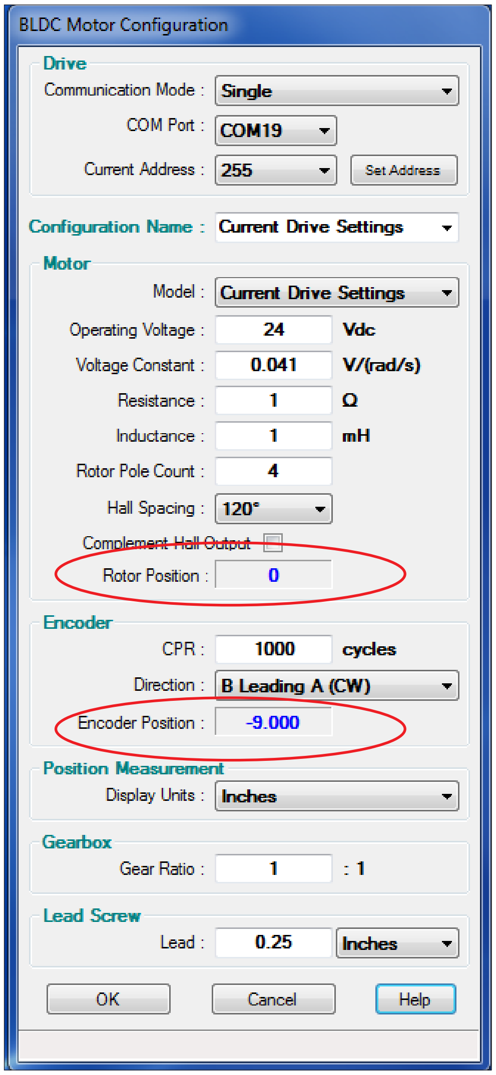
The IDEA™ Drive performs sinusoidal commutation for smooth motor performance, and relies on the encoder for proper operation. Encoders with a higher CPR will allow finer resolutions and tighter precision of positioning. Encoder operation can be verified by disconnecting the motor phase wires from the drive and rotating the shaft by hand clockwise when looking directly at the mounting front face of the motor. Position counter indicator should increment in a positive direction. If encoder position

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

decrements in a negative direction than encoder channels should be swapped. To swap encoder channels in software, navigate to the Settings → Configuration → Encoder direction menu. Select the opposite encoder channel setting <A leads B>or <B leads A> from the value currently displayed.

Hall Effect Sensors

The drive accommodates motors with either 60° or 120° hall sensor cell spacing. Although the drive uses sinusoidal commutation to drive the motor, the hall effect sensors are used for phase initialization; to start motor movement and determine direction. Hall signal operation can be verified by disconnecting the motor phase wires from the drive, and rotating the shaft slowly by hand clockwise when looking directly at the mounting front face of the motor. Navigate to the Settings → Configuration Motor menu and observe the Rotor Position indicator. Numbers should transition sequentially between 0 and 5 when turning the shaft slowly. If numbers transition non-sequentially, or if a 7 appears, then one or more hall effect sensor signals are not operating correctly. The complement hall sensor check box is used to invert the logic of the hall cells. Reference the motor setup guide by motor model for the proper setting.


BLDC Motor Configuration

Drive
Communication Mode : Single
COM Port : COM19
Current Address : 255

Configuration Name : Current Drive Settings

Motor
Model : Current Drive Settings
Operating Voltage : 24 Vdc
Voltage Constant : 0.041 V/(rad/s)
Resistance : 1 Ω
Inductance : 1 mH
Rotor Pole Count : 4
Hall Spacing : 120°
Complement Hall Output ☐
Rotor Position : 0

Encoder
CPR : 1000 cycles
Direction : B Leading A (CW)
Encoder Position : -9.000

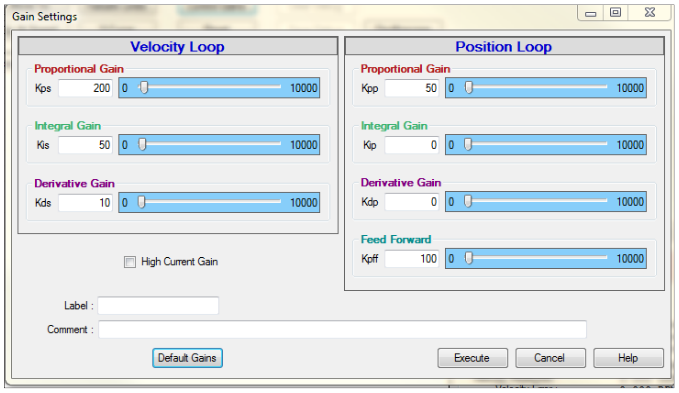
Position Measurement
Display Units : Inches

Gearbox
Gear Ratio : 1 : 1

Lead Screw
Lead : 0.25 Inches

Control Gains Settings

The advanced gain settings window is used to adjust servo control gains to adjust and enhance the dynamic performance and positioning accuracy of the motor.


Gain Settings

Velocity Loop
Proportional Gain — Kps 200 (0–10000)
Integral Gain — Kis 50 (0–10000)
Derivative Gain — Kds 10 (0–10000)
☐ High Current Gain

Position Loop
Proportional Gain — Kpp 50 (0–10000)
Integral Gain — Kip 0 (0–10000)
Derivative Gain — Kdp 0 (0–10000)
Feed Forward — Kpff 100 (0–10000)

Label :
Comment :

This window will allow users to adjust the servo control loops of the brushless IDEA™ Drive. Users can adjust the proportional, integral, and derivative gains of the velocity PID loop and position PID loop. Users should only adjust these gains if they are familiar with their functionality. Reference **www.HaydonKerkPittman.com** for additional information and recommendations for adjusting servo control gain settings.

Saving Programs to the Drive

After a program has been created, the program can be saved onto the drives memory. The program needs to be named, using the Program Name text box. The program name may be up to 10 characters long, and cannot contain the "," character. Any spaces at the end of the program name will not be saved, so "Program1" is the same as "Program1 ". Programs residing on drive memory may not have the same name, so attempting to upload a program with the same name as an existing program will result in a warning, and continuing will cause the existing program to be overwritten.

When the upload button is pressed, the user interface automatically finds the first area on the drive that can fit the program. If there is no place on the drive where the program can fit, the user is responsible for either moving or removing some programs. To move a program to free up larger blocks of free space, simply save the program to the drive again under the same name, and the program will be moved to the first page where it can be fit.

Removing Programs

Programs can be removed from the IDEA™ Drive in one of three ways. Either use the "Display Table of Contents" option under "IDEA™ Drive ", then select the program to be removed in the list, then press "Remove" in the bottom left corner; delete can also be selected with the right click or use the "Delete Program From Drive" option under "IDEA™ Drive ", then select the program to be removed in the drop-down menu and press "OK".

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Table of Contents

The IDEA™ Drive user interface provides a list of all programs on the drive, the pages being used by said programs, and a graphical representation of the contents and free space of the drive. This feature is accessed through the option under IDEA™ Drive → Display Table of Contents menu. This feature can be helpful in freeing up space for additional programs, or seeing which programs could be moved to decrease fragmenting free space.

Startup Program

For fully autonomous operation, the controller will need to start execution once power is applied, as opposed to being started through the user interface. In order to set a program to start on power up, select IDEA™ Drive → Set Startup Program . In the associated window, the current startup program is displayed in bold, and a new startup program can be selected using the drop-down menu of all programs currently on the drive. If it is desired to not have a startup program, "No Startup Program" should be selected.

Saving / Loading / Combining Programs

The IDEA™ user interface allows for programs to be saved to your computer or other drives. To do this, once the program is complete, go to File → Save. This will open a save file dialog box. It is recommended that programs be backed up using the save function, to protect your work in case the program on the drive is overwritten.

To open a saved file, ensure you are in program mode and go to File → Open. This will open an open file dialog box.

The IDEA user interface also provides the ability to combine two or more different programs. With a program open, go to File → Add File. This will open an open file dialog. The file you select will be added to the end of the current program. This can be done multiple times to combine multiple programs.

Autosave

The IDEA™ user interface provides an autosave feature to protect against losing all your work. Once every minute, if the program is non-empty, the user interface automatically saves the file. If the program crashes, or is accidentally erased, the program can be recovered through the File → Recover Autosave. This will restore the most recently autosaved program.

NOTE: If a Recover Autosave is performed with encryption checked and a program line was never written with encryption checked an error will occur stating the autosave.epf could not be found.

NOTE: If you lose a program and wish to recover it through this feature, be sure not to open any other programs first. If this is done, the auto save feature may overwrite the file you wish to recover.

Over Current Protection

The IDEA™ Drive and user interface provide protection against overdriving the current in the motor. When the user interface is opened, the parameters entered for the motor provides the necessary information to calculate the maximum current of the motor. The user interface uses this to prevent the user from entering values that would be damaging to the motor.

Accel / Decel Current Boost

The IDEA™ Drive provides the ability to boost the current for the acceleration ramp and deceleration ramp independently. This may be necessary to provide additional motor torque when implementing sharp ramping rates.

NOTE: It is important for the user to ensure that the current boost feature is not over used. Repeated long ramp without rest can damage the motor if the boosted current is above the rated current of the motor.

Password Protection

The IDEA™ Drive has a password protection feature. When enabled, this feature prevents any program from being read back without the correct password. Passwords can be up to 10 characters in length and cannot contain the "," character. Spaces at the end of the password are ignored, so "Password1" is the same as "Password1 ".

Password protection does not prevent programs from being erased from or written to the drive. This feature is only meant to ensure that programs on password protected drives cannot be copied by a third party.

NOTE: If a drive has been password protected and the password has been lost the drive can be restore from IDEA™ Drive → Restore Factory Defaults. This will remove the password protection, but all programs on the drive will also be lost.

Inputs and Outputs

The IDEA™ Drive has four optically isolated inputs and four optically isolated open-collector outputs. The voltage range for these is 5-24VDC. As the outputs are open-collector, they will need a pull-up resistor tied to the Opto-supply. The outputs can sink up to 200mA each.

NOTE: When an input is not connected to anything, it is logic high. This allows connecting two IDEA™ Drives without the use of pull-up resistors.

NOTE: The inputs can be used in two ways. They can be connected to logic levels that swing between opto ground and opto supply, or they can be attached to a switch connected to opto ground. In the second configuration, when the switch is open, the drive will see this as a logic high, when the switch is closed, and the input is connected to opto ground, the drive will see this as a logic low.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

NOTE: When an input is connected to a mechanical switch or relay, a phenomenon called "bounce" can occur. When the switch contact is almost closed, several electrical arcs can form. If an input is being used as an interrupt, each arc will be a rising and falling edge, causing several false interrupts to trigger. Any input being used as an interrupt source should only be attached to solid state devices or a switch with de-bounce circuitry.

In many cases, it is desirable to test a program that uses the inputs and outputs without connecting the hardware. In the bottom right hand corner of the user interface, there are eight circles, each representing one of the I/O. These show the current state of all the inputs and outputs, with a green filled circle representing logic high, and an empty circle representing logic low. While a program is not running, or while a program is being used in the debug mode, these output circles can be clicked to toggle the state of the outputs

Simulating Inputs

Since the inputs are controlled externally, under normal circumstances, the user does not control them through the user interface. If it is desired to control the inputs through the user interface, the feature can be access via: IDEA™ Drive → simulate Inputs. When this feature is turned on, the actual states of the inputs are ignored and the user interface tells the drive what state to consider the inputs to be. Simulating input can be used during real time and when debugging a program.

NOTE: The simulate inputs feature cannot be turned on or off while a program is running.

Debugger

The IDEA™ user interface has a debugger to help in troubleshooting programs. The debugger is available in the Realtime mode. Once in the Realtime mode, the program to be debugged is selected by the "Program to Run" drop down menu. Once the program is selected, the debug feature is started by pressing the "Start Debug" button. So long as the drive is in debug mode, each program line executed by the drive is displayed in the debug log window. This window can be cleared at any time by using the "Clear Debug" button. The most recently executed command is also highlighted in the program area. Debug mode is turned off by either pressing the red "**Stop**" button, or when the program ends.

There are two ways of advancing through programs in debug mode, single step and running to a label. Each time the "Single Step" button is pressed, the drive executes one command. The single step button remains inactive until the active command completes execution.

The "Run To Label: button simulates a breakpoint. When the "Run To Label" button is pressed, the drive begins to execute the program normally, until the label in the text box to the right of the "Run To Label" button is reached. If this label does not exist in the program, execution will continue until the red "Stop" button is pressed or execution ends on its own.

To aide in readability, as well as documentation, the debug log file can be saved as a text file using the "Save Debug" button.

In the Debug section the oscilloscope option is available. It gives the ability to graph: position error, velocity error, command current, command velocity, command position, measured velocity, measured current, measured position, supply voltage and phase voltage.

Subroutines

A subroutine is a sequence of commands that can be used from anywhere in the program. When a subroutine is called, the address of the next command that was going to be executed is stored on what is called a stack. When the subroutine is exited using the "Return" command, execution of the program resumes at the address that was stored on the stack. A maximum of 10 addresses can be held in the stack. If there are 10 addresses on the stack, and another subroutine is called without a subroutine completing, a "Stack Overflow" error will be asserted and program execution will abort. It is up to the user to ensure that stack overflows do not occur.

Subroutines are exited using one of two commands, "Return" or "Return to". When "Return" is used, program execution resumes at the address stored on the stack. When "Return To" is used, a destination address is specified. Execution now resumes at the label specified, and the stack is emptied. The use of "Return to" exits all subroutines at once.

If a "Return" or "Return To" command is used when the stack is empty, a "Stack Underflow" error will be asserted and execution of the program will halt. It is the responsibility of the user to ensure that stack underflows do not occur.

There are two ways in which a subroutine can be called; the simplest is the "Goto Sub" command. A "Goto Sub" command is used to call a subroutine from within the program, usually to complete a sequence of commands which is used repeatedly in the program. The address stored on the stack when a "Goto Sub" command is used is the address immediately after the "Goto Sub".

The second way to call a subroutine is through interrupts. Interrupts can occur at any time, and are explained in the interrupts section, page 29. When a subroutine is called by an interrupt, the address stored in the stack is the address of the command which would have next been executed. If the interrupt is triggered during a "Wait" or "Wait For Move" command, the address of the "Wait" or "Wait For Move" command is stored, and if the subroutine is completed by the "Return" command, the wait is resumed until it's completion. Note: If a "Wait" command is interrupted, the time spent executing the interrupt or interrupts counts toward the "Wait" command's delay time.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Interrupts

General Description:

Nested vectored preemptive interrupts are a very powerful real-time control feature which permits responding to real time events with minimum latency (delay). Preemptive means an interrupt of higher priority will pre-empt an interrupt of lower priority, immediately executing the higher priority task. Vectored means the interrupt automatically vectors (jumps) to the higher priority task defined as a program label. Nested means interrupts of different priority can be pending (active) simultaneously. Interrupts can occur asynchronously at any place or time within the program flow. The *Return* statement signifies where the interrupt servicing task completes (the interrupt is serviced), and the program flow jumps back to the point of program flow when the interrupt occurred and resumes normal program execution.

Interrupts can be generated by a transition on control logic inputs, or from defined events such as reaching a limit position, generating a position error beyond a threshold value, or operating in current limit continuously for a defined duration. All interrupts respond in the same fashion, the only difference being how they are triggered.

When an interrupt is triggered, it branches to a subroutine whose starting point is defined by a label. All interrupts need two parameters, the label of the subroutine to be run when triggered, and the priority of the interrupt. For more information on subroutines, see Subroutines, page 12.

The priority of the interrupt determines the order in which any pending interrupts are serviced. If an interrupt is triggered while a lower priority interrupt is being serviced, the program immediately begins servicing the new interrupt. If an interrupt is triggered while an interrupt or equal or higher priority is being serviced, the new interrupt is put into a queue of pending interrupts and will be serviced when all higher priority interrupts have been serviced and any interrupts of the same priority which were triggered first have been serviced.

Interrupts are asynchronous events, meaning they can happen at any time or place within the normal program flow. Therefore, it is recommended interrupts should be defined and activated at the beginning of the program. Interrupts remain active until they are intentionally disabled.

It is also recommended interrupts jump to dedicated subroutines which reside outside of the normal program flow. This ensures that interrupts do not interfere with normal program flow, and that interrupts are serviced correctly, with a “Return” statement residing at the end of the servicing routine signifying the end of the interrupt.

There are 5 interrupt queues, one for each priority level, each being 10 interrupts long at most. If there are 10 interrupts in one queue, and another interrupt of that priority is triggered, the new interrupt will be ignored and an “Interrupt queue full” error will be asserted.

IMPORTANT: Interrupts require a “Return” statement in the servicing routine to signify the end of the interrupt task. The “Return” statement removes the interrupt from the queue (list) of pending interrupts to permit another interrupt of equal or lesser priority to occur.

NOTE: If an interrupt is reconfigured to trigger a new subroutine when there is an instance of that interrupt in a queue, the interrupt in the queue will still execute, but will execute the new subroutine, not the subroutine that the interrupt was originally configured to jump to. Care should be taken when programming to ensure this does not occur.

Interrupt Sources

Input Triggered: Each input can be configured to trigger an interrupt through the “Int on Input” button. The “INT” radio button should be selected for any input which is to be used as an interrupt source.

The trigger type should also be selected. The types are:

Rising edge: Occurs when the input level goes from low to high.

Falling edge: Occurs when the input level goes from high to low.

Both edges: Occurs anytime the input changes state.

Position Triggered: Using the “Int On Position” command in a program, an interrupt can be set to trigger when the motor reaches a specific position. Note only one interrupt on position can be initialized at once. If it is desired to set an interrupt on multiple positions, the position of the interrupt needs to be redefined.

This interrupt is useful to detect a position limit of travel relative to a defined position origin.

Current Triggered: Using the “Int On Stall” command, an interrupt will be triggered if the motor operates continuously in current limit for a specified duration. Please note the current limit duration timer will reset to zero if at any instant the current limit condition disappears within the specified duration. This interrupt is useful to detect abnormal operating conditions such as abnormal motor loading or stall conditions.

Position Error Triggered: Using the “Int On Position Error ” command in a program, an interrupt can be set to trigger when the motor position absolute error becomes greater than a specified threshold. This interrupt is useful to detect potential run away conditions such as loss of feedback or loss of phase, or to quickly detect a stall or hard stop condition.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Errors

During operation, the user interface may report an error, below is an explanation of each of these errors.

IO Error: An error occurred when attempting to update the IO. This error occurs when communications between the drive and user interface is interrupted. This may happen on occasion when the drive is busy with a task and temporarily does not respond to the user interface. Press “Retry”. If the message continues to appear, check all connections.

Stack Underflow: This error occurs when a running program runs to a “Return” command while not within a subroutine. Check the running program for ways that the program could get to a “Return” without having used a “Goto Sub” or an interrupt.

Stack Overflow: This error occurs when 10 or more subroutines are called without returning. Check the running program to ensure that all subroutines end with either a “Return” or “Return To”, and that you are not nesting too many subroutines.

Driver Overtemp: This error occurs when the internal temperature of the drive exceeds a safe level. Consider moving the drive to a cooler location, reducing the hold and/or run currents, adding additional heat sinking, or adding active cooling.

Encoder Error: This error occurs when the encoder encounters a problem.

Interrupt Queue Full: This error occurs when a running program encounters more than 10 interrupts of the same priority without having serviced any. This could occur due to an interrupt source causing multiple extraneous interrupts, or by one interrupt subroutine not returning, thus preventing execution of any other interrupts. Check your interrupt sources, and ensure that all interrupt subroutines end in a “Return” or “Return To” command.

Loop Overflow: This error occurs when a running program is in more than 10 “Jump N Times” commands at once. Ensure that the running program does not nest more than 10 “Jump N Times” commands.

Drive Current Limit: This error occurs when the drive is given a command with a current setting higher than its capability.

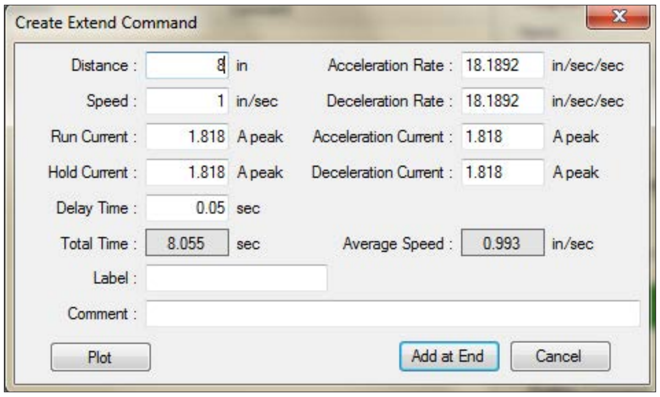
Bad Checksum, Update Aborted: This error is most likely caused by communications being interrupted during a firmware update. Hit “OK” and attempt to update the firmware again. If the problem persists, contact AMETEK Advanced Motion Solutions.

Unknown Error: If you ever receive this error, make a note of what you were doing at the time, and contact AMETEK Advanced Motion Solutions.

Explanation of Commands

Extend / Index CW

The extend command moves the motor a specified distance forwards from its current location. When using a rotary motor, this command rotates the motor clockwise, as viewed from the output shaft.


Create Extend Command

Distance :	8	in	Acceleration Rate :	18.1892	in/sec/sec
Speed :	1	in/sec	Deceleration Rate :	18.1892	in/sec/sec
Run Current :	1.818	A peak	Acceleration Current :	1.818	A peak
Hold Current :	1.818	A peak	Deceleration Current :	1.818	A peak
Delay Time :	0.05	sec			
Total Time :	8.055	sec	Average Speed :	0.993	in/sec
Label :					
Comment :					

Parameters

Distance: This is the distance that the motor will extend or rotate to.

Speed: This is the top speed at which the motor will extend or rotate at.

Run Current: This is the maximum peak current per phase that may be applied to the windings while the motor moves at the top speed.

Hold Current: This is the maximum peak current per phase that may be applied to the windings when the motor is at standstill.

Accel Rate: This is the rate at which the motor will be ramped up from standstill to the run speed.

Decel Rate: This is the rate at which the motor will be ramped down from the run speed to standstill.

Accel Current: During acceleration, the maximum peak current per phase that may be applied to the windings used to bring the motor up to the run speed from a standstill.

Decel Current: During deceleration, the maximum peak current per phase that may be applied to the windings used to stop the motor from the run speed to a standstill.

Delay Time: The dwell time between switching from the decel current to the hold current.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

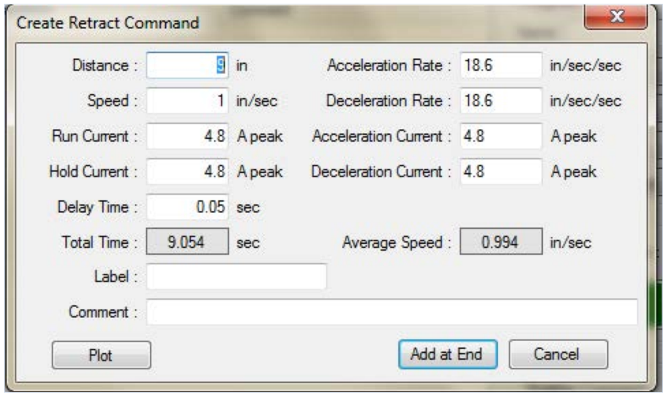
The following program extends the motor in a linear system 1", and waits for the move to complete before executing any commands which may follow:

Action	Label	Description	Comment
0		Extend 1 in	
1		Wait For Move	

This command is used in example 1.

Retract/Index CCW

The retract command moves the motor a specified distance reverse from its current location. When using a rotary motor, this command rotates the motor counterclockwise, as viewed from the output shaft.



Distance :	9	in	Acceleration Rate :	18.6	in/sec/sec
Speed :	1	in/sec	Deceleration Rate :	18.6	in/sec/sec
Run Current :	4.8	A peak	Acceleration Current :	4.8	A peak
Hold Current :	4.8	A peak	Deceleration Current :	4.8	A peak
Delay Time :	0.05	sec			
Total Time :	9.054	sec	Average Speed :	0.994	in/sec
Label :					
Comment :					

Parameters

Distance: This is the distance that the motor will retract or rotate to.

Speed: This is the top speed at which the motor will extend or rotate at.

Run Current: This is the maximum peak current per phase that may be applied to the windings while the motor moves at the top speed.

Hold Current: This is the maximum peak current per phase that may be applied to the windings when the motor is at standstill.

Accel Rate: This is the rate at which the motor will be ramped up from standstill to the run speed.

Dekle Rate: This is the rate at which the motor will be ramped down from the run speed to standstill.

Accel Current: During acceleration, the maximum peak current per phase that may be applied to the windings used to bring the motor up to the run speed from a standstill.

Decel Current: During deceleration, the maximum peak current per phase that may be applied to the windings used to stop the motor from the run speed to a standstill.

Delay Time: The dwell time between switching from the decel current to the hold current.

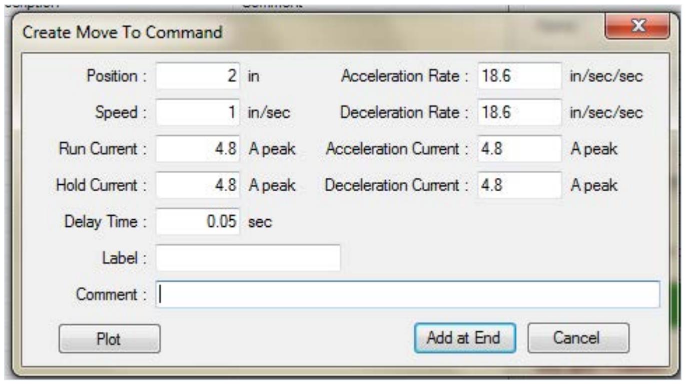
The following program retracts the motor in a linear system 1", and waits for the move to complete before executing any commands which may follow:

Action	Label	Description	Comment
0		Retract 1 in	
1		Wait For Move	

This command is used in examples 1, 2, 3, 4, 5 and 6.

Move To

The move to command moves the motor to a specific location, based upon the encoder.



Position :	2	in	Acceleration Rate :	18.6	in/sec/sec
Speed :	1	in/sec	Deceleration Rate :	18.6	in/sec/sec
Run Current :	4.8	A peak	Acceleration Current :	4.8	A peak
Hold Current :	4.8	A peak	Deceleration Current :	4.8	A peak
Delay Time :	0.05	sec			
Label :					
Comment :					

Parameters

Position: This is the position to which the motor will move to.

Speed: This is the top speed at which the motor will extend or rotate at.

Run Current: This is the maximum peak current per phase that may be applied to the windings while the motor moves at the top speed.

Hold Current: This is the maximum peak current per phase that may be applied to the windings when the motor is at standstill.

Accel Rate: This is the rate at which the motor will be ramped up from standstill to the run speed.

Decel Rate: This is the rate at which the motor will be ramped down from the run speed to standstill.

Accel Current: During acceleration, the maximum peak current per phase that may be applied to the windings used to bring the motor up to the run speed from a standstill.

Decel Current: During deceleration, the maximum peak current per phase that may be applied to the windings used to stop the motor from the run speed to a standstill.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Delay Time: The dwell time between switching from the decel current to the hold current.

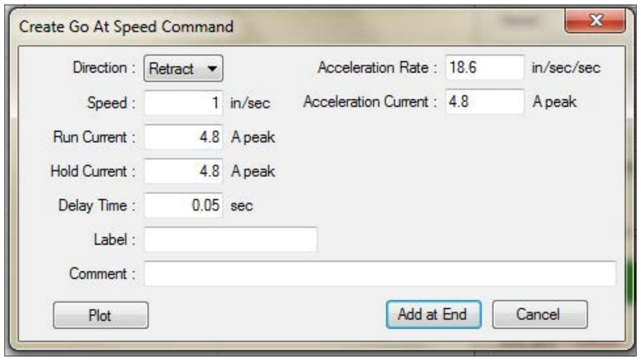
The following program moves the motor to a point 1" forward from where the motor was at the beginning of the program, and waits for the move to complete before executing any commands which may follow:

Action	Label	Description	Comment
0		Move To 1 in	
1		Wait For Move	

This command is used in examples 4, 5, and 6.

Go At Speed

The Go At Speed command moves the motor in a specified direction at a specified speed.


Create Go At Speed Command

Direction : Retract
Speed : 1 in/sec
Run Current : 4.8 A peak
Hold Current : 4.8 A peak
Delay Time : 0.05 sec
Label :
Comment :
Acceleration Rate : 18.6 in/sec/sec
Acceleration Current : 4.8 A peak

Parameters

Direction: This is the direction in which the motor will move.

Speed: This is the top speed at which the motor will extend or rotate at.

Run Current: This is the maximum peak current per phase that may be applied to the windings while the motor moves at the top speed.

Hold Current: This is the maximum peak current per phase that may be applied to the windings when the motor is at standstill.

Accel Rate: This is the rate at which the motor will be ramped up from standstill to the run speed.

Accel Current: During acceleration, the maximum peak current per phase that may be applied to the windings used to bring the motor up to the run speed from a standstill.

Decel Current: During deceleration, the maximum peak current per phase that may be applied to the windings used to stop the motor from the run speed to a standstill.

Delay Time: The dwell time between switching from the decel current to the hold current.

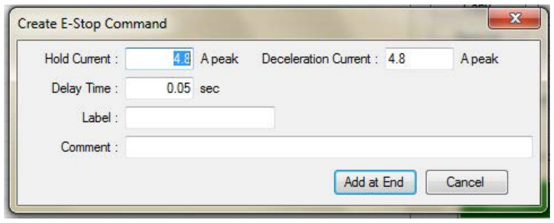
The following program moves the motor for 1 second.

Action	Label	Description	Comment
0		Go At Speed 1 in/sec	
1		Wait 1 sec	

This command is used in example 6.

E-Stop

The E-Stop command brings the motor to an immediate stop. This does not halt the program.


Create E-Stop Command

Hold Current : 4.8 A peak
Delay Time : 0.05 sec
Label :
Comment :
Deceleration Current : 4.8 A peak

Parameters

Hold Current: This is the maximum peak current per phase that may be applied to the windings when the motor is at standstill.

Decel Current: During deceleration, the maximum peak current per phase that may be applied to the windings used to stop the motor from the run speed to a standstill.

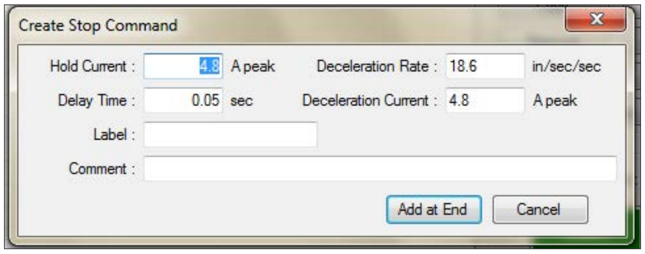
Delay Time: The dwell time between switching from the decel current to the hold current.

The following program starts a move at 1" per second, waits 0.5 seconds, and the stops the actuator.

Action	Label	Description	Comment
0		Go At Speed 1 in/sec	
1		Wait 0.5 sec	
2		E-Stop	

Stop

The Stop command brings the motor to a stop with an optional deceleration ramp. This does not halt program execution.


Create Stop Command

Hold Current : 4.8 A peak
Delay Time : 0.05 sec
Label :
Comment :
Deceleration Rate : 18.6 in/sec/sec
Deceleration Current : 4.8 A peak

Parameters

Hold Current: This is the maximum peak current per phase that may be applied to the windings when the motor is at standstill.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Delay Time: The dwell time between switching from the decel current to the hold current.

Decel Rate: This is the rate at which the motor will be ramped down from the run speed to standstill.

Decel Current: During deceleration, the maximum peak current per phase that may be applied to the windings used to stop the motor from the run speed to a standstill.

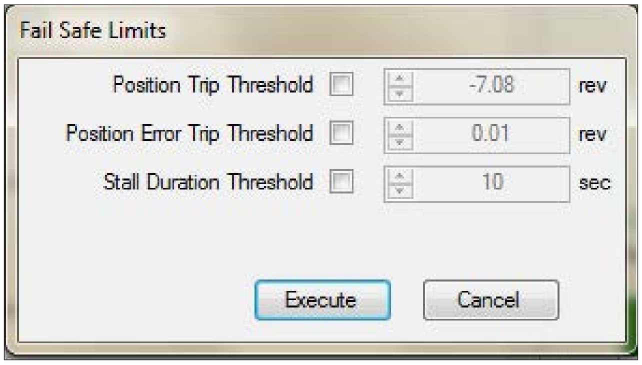
The following program starts a move at 1" per second, waits 0.5 seconds, and stops the actuator.

Action	Label	Description	Comment
0		Go At Speed 1 in/sec	
1		Wait .5 sec	
2		Stop	

This command is used in example 5.

Failsafe Limits

Failsafe limits can be set for position, position trip error and current duration thresholds. **NOTE**: This function is used only in Real Time Mode. When only one command is tested. Position will be relative to the origin.



Fail Safe Limits

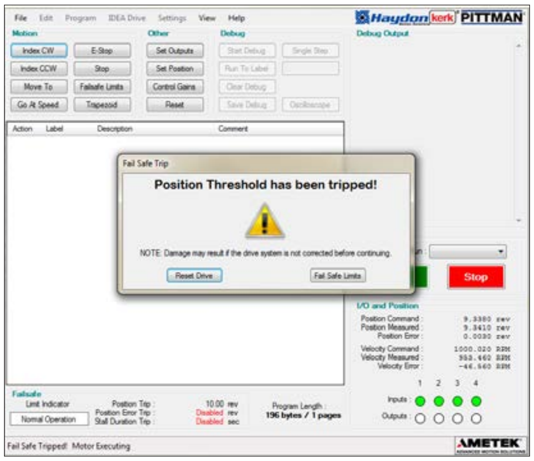
Position Trip Threshold	☐	-7.08	rev
Position Error Trip Threshold	☐	0.01	rev
Stall Duration Threshold	☐	10	sec

Parameters

Position Trip Threshold: Revolution in which limit is set to. It will automatically update.

Position Error Trip Threshold: Position error in which the limit can be set to.

Stall Duration Thresholds: Current duration for limit to be set.



Fail Safe Trip

Position Threshold has been tripped!

NOTE: Damage may result if the drive system is not corrected before continuing.

Position Command : 9.3380 rev
Position Measured : 9.3410 rev
Position Error : 0.0030 rev
Velocity Command : 1000.020 RPM
Velocity Measured : 953.460 RPM
Velocity Error : -46.560 RPM

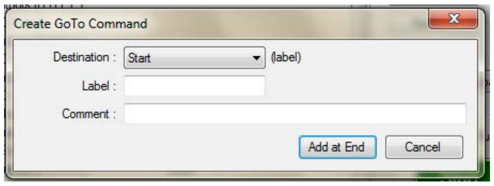
Position Trip : 10.00 rev
Position Error Trip : Disabled rev
Stall Duration Trip : Disabled sec

Program Length : 196 bytes / 1 pages

Fail Safe Tripped! Motor Executing

Goto

The Goto command goes to a specified label.



Create GoTo Command

Destination : Start (label)
Label :
Comment :

Parameters

Destination: This is the label of the command to which the program will go.

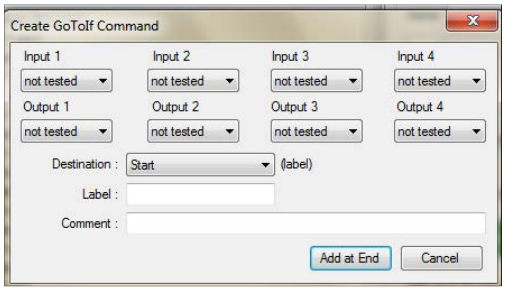
The following program extends .25", waits for the move to **Stop**, then repeats until the program is aborted.

Action	Label	Description	Comment
0	extend	Extend 0.25 in	
1		Wait For Move	
2		Goto extend	

This command is used in examples 1, 5 and 6.

Goto If

The Goto If command goes to a specified label if the current states of the inputs match the specified conditions, and goes to the next line in the program otherwise.



Create GoToIf Command

Input 1	Input 2	Input 3	Input 4
not tested	not tested	not tested	not tested
Output 1	**Output 2**	**Output 3**	**Output 4**
not tested	not tested	not tested	not tested

Destination : Start (label)
Label :
Comment :

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Parameters

Destination: This is the label of the command that should be jumped to.

Inputs/Outputs: Each I/O can be specified as High, Low, or not tested. Any I/O set to "Not Tested" will be ignored; the state of the I/O when the command is executed must match all settings of high or low to go to the specified label, otherwise, the next line of the program is executed.

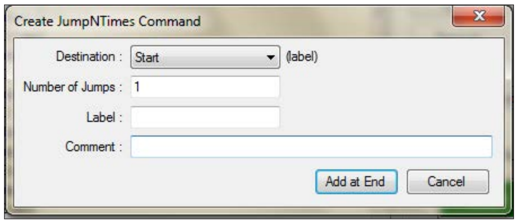
The following program extends .25", waits for the move to stop, then repeats so long as input 1 is high and input 2 is low.

Action	Label	Description	Comment
0	extend	Extend 0.25 in	
1		Wait For Move	
2		Goto If extend	1 set to high, 2 set to low

This command is used in example 5.

Jump N Times

The Jump N Times command goes to a specified label a specified number of times. Once the number of jumps has been completed, execution continues at the next line in the program, if any exist.


Create JumpNTimes Command

Destination : Start (label)

Number of Jumps : 1

Label :

Comment :

Add at End | Cancel

Parameters

Destination: This is the label of the command that should be jumped to.

Number of Jumps: This is how many times the command should jump.

The following program extends .25", waits for the move to Stop, then repeats 3 times.

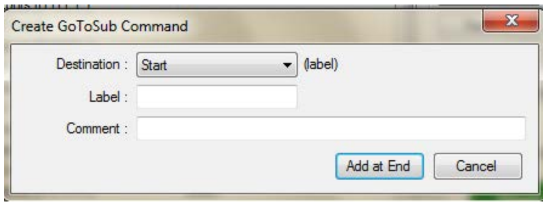
NOTE: Because the command goes to the extend command 3 times from the jump n times command, the extend is performed a total of 4 times.

Action	Label	Description	Comment
0	extend	Extend 0.25 in	
1		Wait For Move	
2		Jump N Times extend	Number of Jumps: 3

This command is used in example 2.

Goto Sub

The Goto Sub command goes to a subroutine starting with the specified label. For further explanation of subroutines, see Subroutines, page 12.


Create GoToSub Command

Destination : Start (label)

Label :

Comment :

Add at End | Cancel

Parameters

Destination: This is the label of the command that is the start of the subroutine.

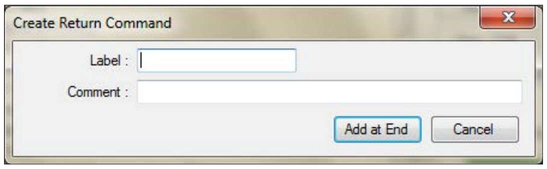
The following program runs a subroutine starting at label extend, which extends the motor 0.25", waits for the move to complete, then returns, then repeats.

Action	Label	Description	Comment
0	start	Goto Sub extend	
1		Goto start	
2			
3	extend	Extend 0.25 in	
4		Wait For Move	
5		Return	

This command is used in example 3.

Return

The return command ends a subroutine and returns execution to the location from which the subroutine was called. For further explanation of subroutines, see Subroutines, page 12.


Create Return Command

Label :

Comment :

Add at End | Cancel

Parameters

This command has no parameters.

The following program runs a subroutine starting at label extend, which extends the motor 0.25", waits for the move to complete, then returns, then repeats.

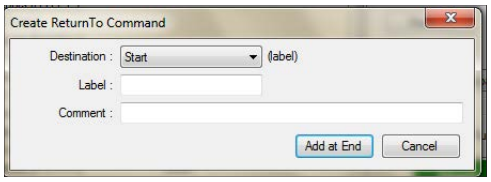
Action	Label	Description	Comment
0	start	Goto Sub extend	
1		Goto start	
2			
3	extend	Extend 0.25 in	
4		Wait For Move	
5		Return	

This command is used in example 3.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Return To

The Return To command ends a subroutine, clears the stack, clears any pending interrupts, clears any "Jump N Times" commands, and goes to a specified label. For further explanation of subroutines, see Subroutines, page 12.



Parameters

Destination: The label of the command that should be executed next.

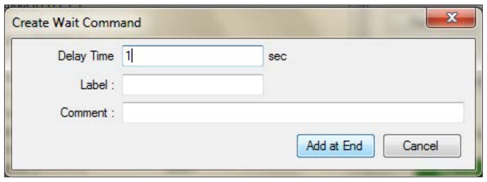
The following program runs a subroutine starting at label extend, which extends the motor 0.25", waits for the move to complete, then exits the subroutine, goes to the abort command, and aborts the program.

Action	Label	Description	Comment
0	start	Goto Sub extend	
1		Goto start	
2			
3	extend	Extend 0.25 in	
4		Wait For Move	
5		Return To abort	
6			
7	abort	Abort	

This command is used in example 6.

Wait

The Wait command delays execution of the next command in the program for a specified time.



Parameters

Delay Time: The amount of time that the program should wait.

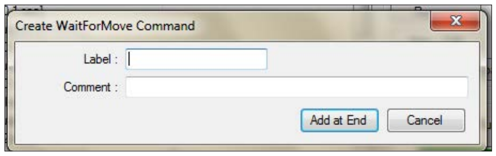
The following program begins moving the actuator at 1" per second, waits 0.5 seconds, then stops the motor.

Action	Label	Description	Comment
0		Go At Speed 1 in/sec	
1		Wait 0.5 sec	
2		Stop	

This command is used in examples 1, 3, and 4.

Wait For Move

The Wait For Move command delays execution of the next command in the program until a move has completed. This command is automatically added after every Extend, Retract, and Move To, but can be removed if necessary.



Parameters

This command has no parameters.

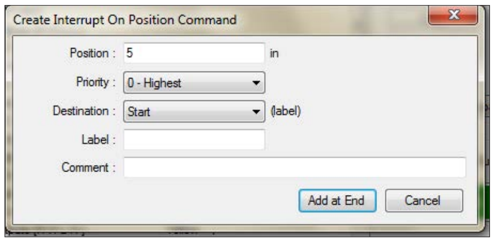
The following program extends the motor 1", and waits for the move to complete before executing any commands which may follow:

Action	Label	Description	Comment
0		Extend 1 in	
1		Wait For Move	

This command is used in examples 1, 2, 3, 4, 5, and 6.

INT On Pos

The INT On Pos command sets an interrupt to be triggered when the motor reaches a specified position. For further explanation of interrupts, see Interrupts, page 13.



Parameters

Position: The position, based upon the position counter of the drive, at which the interrupt should be triggered.

Destination: The label of the subroutine that should be executed when the position is reached.

Priority: The priority of the interrupts.

NOTE: If an interrupt is set for a position, and the position counter is adjusted using the "Set Position" command, the interrupt will still occur at the same point.

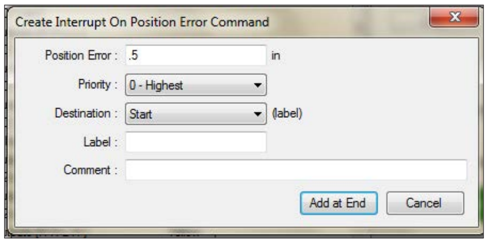
Example: The drive is turned on, and an interrupt is set on position 0.5". The "set Position" command is then used, changing what was the 0"

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

position to the 1" position. The interrupt will now trigger when the drive's position counter reaches 1.5".

Interrupt (INT) On Pos Error

The Interrupt On Position Error command allows an interrupt to be triggered when the position error is exceeded it can be set as shown below



Create Interrupt On Position Error Command

Position Error : .5 in

Priority : 0 - Highest

Destination : Start (label)

Label :

Comment :

Parameters

Position: The position error, based upon the position counter of the drive, at which the interrupt should be triggered.

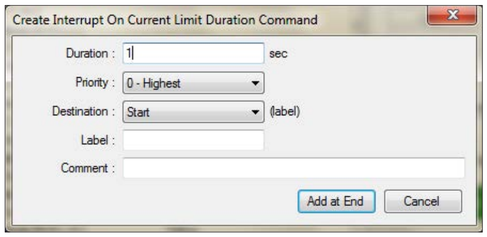
Destination: The label of the subroutine that should be executed when the position is reached.

Priority: The priority of the interrupts.

```
Action   Label      Description                      Comment
0                   Int On Position Error [1 rev]
1                   Index CCW [1000 rev]
2                   Wait For Move
3                   Wait [60 sec]
4        START      Set Position [0 rev]
5                   Wait [3 sec]
6                   Index CW [100 rev]
```

Interrupt (INT) On Stall

The Interrupt On Stall command allows an interrupt to be triggered when the current limit is excited. The duration of the current limit is set as shown below:



Create Interrupt On Current Limit Duration Command

Duration : 1 sec

Priority : 0 - Highest

Destination : Start (label)

Label :

Comment :

Parameters

Duration: The amount of time the interrupt will be triggered for.

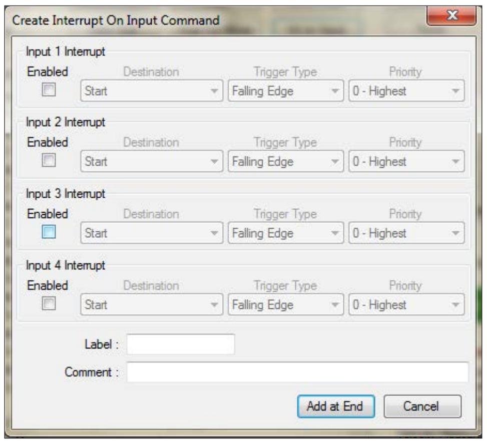
Destination: This is the label for the subroutine associated with the interrupt.

Priority: The priority of the interrupts.

```
Action   Label      Description                          Comment
0                   Int On Current Limit Duration [1 sec]
1                   Index CCW [1000 rev]
2                   Wait For Move
3                   Wait [60 sec]
4        START      Set Position [0 rev]
5                   Wait [3 sec]
```

Interrupt (INT) on Input

The Interrupt on Input command allows an interrupt to be triggered when any of the inputs are changed. For further explanation of interrupts, see Interrupts, page 13.



Create Interrupt On Input Command

	Enabled	Destination	Trigger Type	Priority
Input 1 Interrupt	☐	Start	Falling Edge	0 - Highest
Input 2 Interrupt	☐	Start	Falling Edge	0 - Highest
Input 3 Interrupt	☐	Start	Falling Edge	0 - Highest
Input 4 Interrupt	☐	Start	Falling Edge	0 - Highest

Label :

Comment :

Parameters

Each input has the same parameters.

Disabled/Enabled: Select "Enabled" if the input should cause an interrupt, select "Disabled" otherwise.

Destination: This is the label for the subroutine associated with the interrupt.

Trigger Type: If "Falling Edge" is selected, the interrupt will be triggered when the input goes from a logic high to a logic low. If "Rising Edge" is selected, the interrupt will trigger when the input goes from logic low to logic high. If "Both Edges" is selected, the interrupt will trigger any time the state of the input changes.

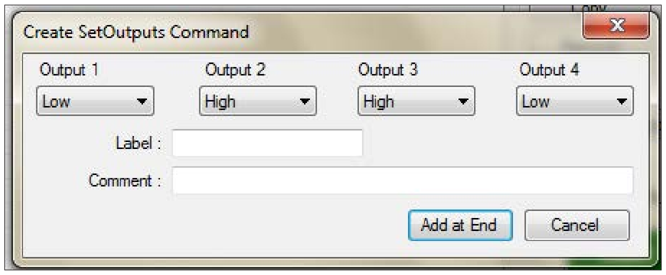
©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

The following program sets an interrupt for the rising edge of input 1, and then loops continuously. When input one changes to logic high, output 1 is set to high.

Action	Label	Description	Comment
0		Int on Input INT, GP, GP, GP	
1	goto	Goto goto	
2	set Output	Set Outputs H X X X	
3		Return	

Set Outputs

The Set Outputs command sets the logic state of the outputs.



Create SetOutputs Command

Output 1: Low
Output 2: High
Output 3: High
Output 4: Low
Label :
Comment :

Parameters

Outputs: Each output is individually set as either high, which will bring the outputs to the opto-supply voltage, low, which brings the output to the opto-ground voltage, or no change, which will leave the output in its current state.

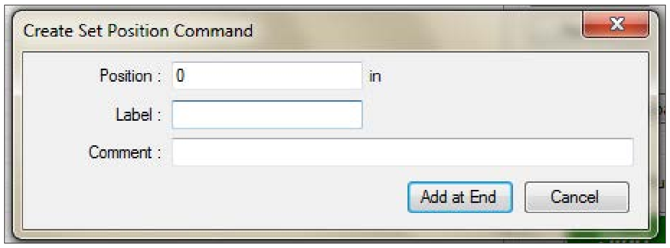
The following program sets output 1 high, waits 0.5 seconds, and then sets output 1 low. Outputs 2-3-4 are not tested.

Action	Label	Description	Comment
0		Set Outputs [H X X X]	
1		Wait [2 sec]	
2		Set Outputs [L X X L]	
3		Wait [.5 sec]	

This command is used in example 3.

Set Position

The Set Position command sets the position counter of the drive to the specified value.



Create Set Position Command

Position : 0 in
Label :
Comment :

Parameters

Position: The value to which the current position of the drive should be set.

The following program retracts the motor 1”, sets the position to 0”, and then moves to the 0.5” position, which causes a 0.5” extend.

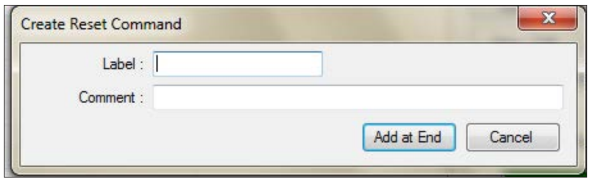
Action	Label	Description	Comment
0		Retract 1 in	
1		Wait For Move	
2		Set Position 0 in	
3		Move To 0.5 in	
4		Wait For Move	

This command is used in examples 4, 5, and 6.

Control Gains*:* reference page 10.

Reset

The Reset command restarts the drive, like turning the drive off and on. If used in a program, this command will not be executed less than 1/10th of a second after the drive has seen a reset, to prevent an uninterruptible cycle of resets.



Create Reset Command

Label :
Comment :

Parameters

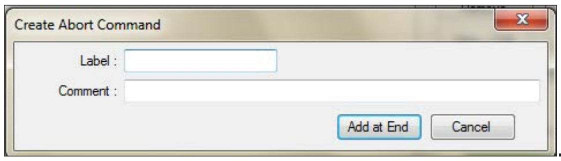
This command has no parameters.

The following program resets the drive.

Action	Label	Description	Comment
0		Reset	Resets the device

Abort

The Abort command immediately stops any moves without deceleration, applies the last specified holding current and halts execution of any running program.



Create Abort Command

Label :
Comment :

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Parameters

This command has no parameters.

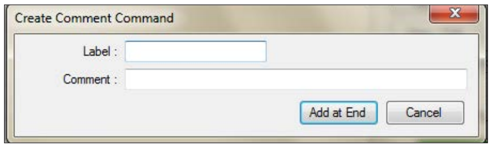
The following program aborts the program.

Action	Label	Description	Comment
0		Abort	Ends the program

This command is used in examples 3 and 6.

Comment

The Comment command performs no action. This command is used to add extra information into a program for better documentation. Each comment line affords an additional 30 characters of comments. This command can also be used to add breaks between sections of code, making it more readable.



Parameters

This command has no parameters.

The following program performs no actions.

Action	Label	Description	Comment
0			This program performs no
1			function, but does help to
2			demonstrate the usefulness
3			of the comment command
4			
5			By adding in a comment command
6			without a comment, I have made
7			a line break for readability.

Programming Examples

Because of the variety of motors that this product can be used with, examples for every product, or a general example for all products cannot be achieved. Speeds, distances and other parameters will need to be changed for your actuator. The following examples are used with a motor that is part of a linear system the measurement units will be Inches.

Example One

Extend the motor within a linear application 0.4 inches, wait one second, retract the motor 0.2 inches, wait one second, extend the motor 0.6 inches, wait 1 second, retract the motor 0.8 inches, wait 1 second, and repeat from the first extend indefinitely. Let the linear speed for each move be 1 inch per second.

We first want to extend 0.4 inches.

- Click the “Extend” button.
- Input the distance as 0.4 inches.
- Input the speed as 1 inch per second.
- All other parameters can be left as the defaults.
- Enter “Start” as the label.
- You can add a comment in the comment line if you wish.
- Click the “Add at End” button to place the command in program.

NOTE: You will notice that when the extend command populates in the program field on the screen, that it is followed by a second command that states “Wait For Move.” This is also true when using the Retract and Move To commands. This is a command to allow the move to finish before execution of the next command.

We now want to wait 1 second before moving again.

- Click the “Wait” button.
- Input the delay time as 1 second. No label or comment is necessary.
- Click “Add at End” to place this command into the program.

We now want to retract 0.2 inches.

- Click the “Retract” button.
- Input the distance as 0.2 inches
- Input the speed as 1 inch per second
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the “Add at End” button to place the command in program.

We now want to wait another second.

- Click the “Wait” button.
- The delay time will be 1 second, as previously entered. No label or comment is necessary.
- Click “Add at End” to place this command into the program.

We now want to extend 0.6 inches.

- Click the “Extend” button. As before, many items are already populated. This time the distance and speed are also populated.
- Input the distance as 0.6 inches
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the “Add at End” button to place the command in program.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

We now want to wait one second.

- Click the "Wait" button.
- The delay time will be 1 second as previously entered. No label or comment is necessary.
- Click "Add at End" to place this command into the program.

We now want to make the final move of retracting 0.8 inches.

- Click the "Retract" button.
- Input the distance as 0.8 inches
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Add at End" button to place the command in program.

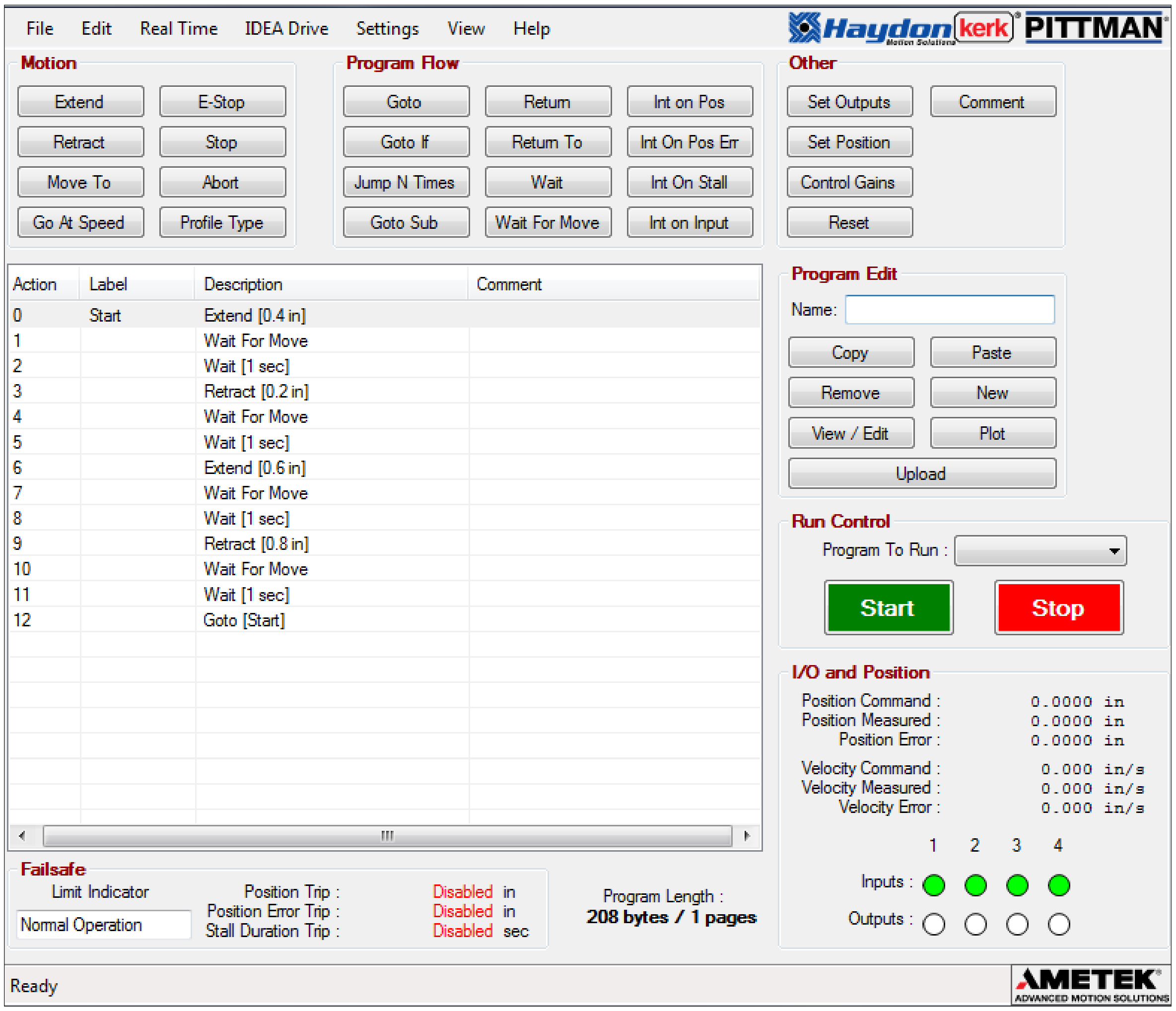
We now want to wait one second.

- Click the "Wait" button.
- The delay time will be 1 second as previously entered. No label or comment is necessary.
- Click "Add at End" to place this command into the program.

We now want to repeat.

- Click the "Goto" button.
- Select "Start" as the destination
- Click "Add at End" to place this command into the program.

The completed program looks as follows:



Action	Label	Description	Comment
0	Start	Extend [0.4 in]	
1		Wait For Move	
2		Wait [1 sec]	
3		Retract [0.2 in]	
4		Wait For Move	
5		Wait [1 sec]	
6		Extend [0.6 in]	
7		Wait For Move	
8		Wait [1 sec]	
9		Retract [0.8 in]	
10		Wait For Move	
11		Wait [1 sec]	
12		Goto [Start]	

```
Position Command :   0.0000 in
Position Measured :  0.0000 in
Position Error :     0.0000 in
Velocity Command :   0.000 in/s
Velocity Measured :  0.000 in/s
Velocity Error :     0.000 in/s
           1  2  3  4
Inputs :   ●  ●  ●  ●
Outputs :  ○  ○  ○  ○
```

Limit Indicator: Normal Operation

Position Trip : Disabled in
Position Error Trip : Disabled in
Stall Duration Trip : Disabled sec

Program Length : **208 bytes / 1 pages**

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Example Two

This example will extend the motor within a linear application .5", retract the motor .5", then repeat those two moves 4 more times. Once the repetitions are complete, the motor will extend 1". Let the linear speed for each move be 1 inch per second.

We first want to extend 0.5".

- Click the "Extend" button.
- Input the distance as 0.5 inches.
- Input the speed as 1 inch per second.
- All other parameters can be left as the defaults.
- Enter "Start" as the label.
- You can add a comment in the comment line if you wish.
- Click the "Add at End" button to place the command in program.

NOTE: You will notice that when the extend command populates in the program field on the screen, that it is followed by a second command that states "Wait For Move". This is also true when using the Retract and Move To commands. This is a command to allow the move to finish before execution of the next command.

We now want to retract 0.5".

- Click the "Retract" button.
- Input the distance as 0.5 inches.
- Input the speed as 1 inch per second.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Add at End" button to place the command in program.

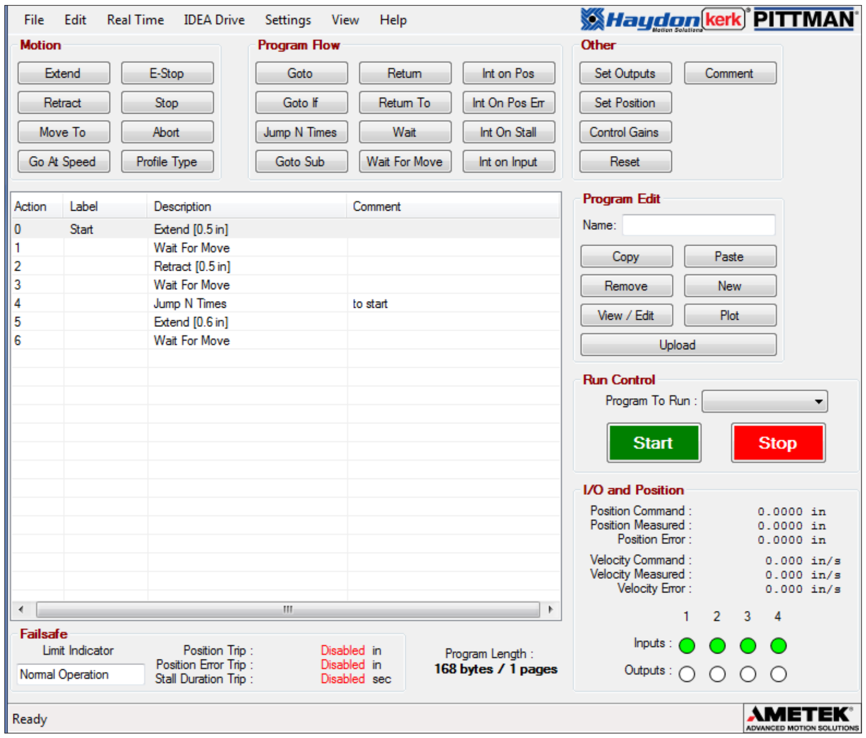
We now want to repeat the first two moves four times.

- Click the "Jump N Times" button.
- Select "Start" as the destination.
- Enter the number of jumps as 4.
- Click "Add at End" to place this command into the program.

We now want to extend 1".

- Click the "Extend" button.
- Input the distance as 1".
- Input the speed as 1 inch per second.
- All other parameters can be left as the defaults.
- No Label is required
- You can add a comment in the comment line if you wish.
- Click the "Add at End" button to place the command in program.

The completed program looks as follows:



Action	Label	Description	Comment
0	Start	Extend [0.5 in]	
1		Wait For Move	
2		Retract [0.5 in]	
3		Wait For Move	
4		Jump N Times	to start
5		Extend [0.6 in]	
6		Wait For Move	

Name:

Program To Run :

```
Position Command :    0.0000 in
Position Measured :   0.0000 in
Position Error :      0.0000 in
Velocity Command :    0.000 in/s
Velocity Measured :   0.000 in/s
Velocity Error :      0.000 in/s
```

	1	2	3	4
Inputs :	●	●	●	●
Outputs :	○	○	○	○

Limit Indicator: Normal Operation

Position Trip : Disabled in
Position Error Trip : Disabled in
Stall Duration Trip : Disabled sec

Program Length : **168 bytes / 1 pages**

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Example Three

This example will extend the motor within a linear application .5", turn all four outputs high, wait .5 seconds, turn all outputs low, retract the motor .5", turn all four outputs high, wait .5 seconds, turn all outputs low, then end the program. We will accomplish this by using a subroutine.

We first want to extend 0.5".

- Click the "Extend" button.
- Input the distance as 0.5 inches.
- Input the speed as 1 inch per second.
- All other parameters can be left as the defaults.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Add at End" button to place the command in program.

NOTE: You will notice that when the extend command populates in the program field on the screen, that it is followed by a second command that states "Wait For Move". This is also true when using the Retract and Move To commands. This is a command to allow the move to finish before execution of the next command.

We need a subroutine.

- Click the "Set Outputs" button.
- Select "High" for each of the four outputs.
- Enter "Toggle" as the label.
- You can add a comment in the comment line if you wish.
- Click the "Add at End" button to place the command in program

We now want to use a subroutine to toggle the outputs on and off.

- Highlight the "Set Outputs" command and then click the "Goto Sub" button.
- Enter the destination as "Toggle".
- No label is required.
- You can add a comment in the comment line if you wish.
- Click the "Insert Before" button to place the command in program.

We now want to retract 0.5".

- Highlight "Goto Sub" command and then click the "Retract" button.
- Input the distance as 0.5 inches.
- Input the speed as 1 inch per second.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Insert After" button to place the command in program.

We now want to use a subroutine to toggle the outputs on and off.

- Highlight the last command entered line #4 and then click the "Goto Sub" button.
- Select the destination "Toggle".
- No label is required.
- You can add a comment in the comment line if you wish.
- Click the "Insert After" button to place the command in program.

We now want to stop the program.

- Highlight the last command enter (line #5) and then click the "Abort" button.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Insert After " button to place the command in program.

Now we need to wait 0.1 seconds.

- Highlight the set Outputs command and then click the "Wait" button.
- Enter 0.5 seconds as the delay time.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Insert After " button to place the command in program.

Now we need to set the outputs low.

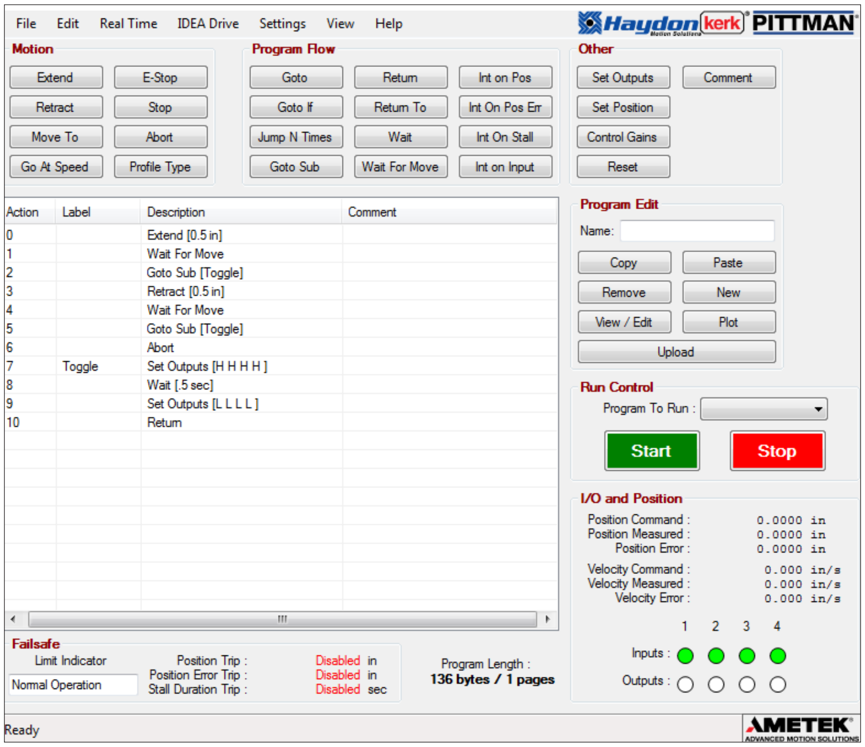
- Click the "Set Outputs" button.
- Select "Low" for each of the four outputs.
- No label is required.
- You can add a comment in the comment line if you wish.
- Click the "Add at End" button to place the command in program.

Now we need to return to the main body of the program. To go back to where the subroutine was called from, we use a "Return" command.

- Click the "Return" button.
- No label is required.
- You can add a comment in the comment line if you wish.
- Click the "Add at End" button to place the command in program.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

The completed program looks as follows:



Action	Label	Description	Comment
0		Extend [0.5 in]	
1		Wait For Move	
2		Goto Sub [Toggle]	
3		Retract [0.5 in]	
4		Wait For Move	
5		Goto Sub [Toggle]	
6		Abort	
7	Toggle	Set Outputs [H H H H]	
8		Wait [.5 sec]	
9		Set Outputs [L L L L]	
10		Return	

Name:

Program To Run :

Position Command :	0.0000	in
Position Measured :	0.0000	in
Position Error :	0.0000	in
Velocity Command :	0.000	in/s
Velocity Measured :	0.000	in/s
Velocity Error :	0.000	in/s

	1	2	3	4
Inputs :	●	●	●	●
Outputs :	○	○	○	○

Limit Indicator: Normal Operation

Position Trip :	Disabled	in
Position Error Trip :	Disabled	in
Stall Duration Trip :	Disabled	sec

Program Length : **136 bytes / 1 pages**

Ready

Example Four

This example is different in that the motor within a linear application does not need to start in the fully retracted position. We will perform a homing routine, to find the fully retracted position, and then move to 0.5” from that point, wait 1 second, and then return to the fully retracted position.

We first want to retract the full stroke of the actuator.

- Click the “Retract” button.
- Input the distance as 1 inch.
- Input the speed as 0.5 inch per second.
- All other parameters can be left as the defaults.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the “Add at End” button to place the command in program.

NOTE: You will notice that when the retract command populates in the program field on the screen, that it is followed by a second command that states “Wait For Move“. This is also true when using the Extend and Move To commands. This is a command to allow the move to finish before execution of the next command.

We are now at the fully retracted position; to keep track of this we will use the “Set Position” command.

- Click the “Set Position” button.
- Enter a position of 0”.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click “Add at End” to place this command into the program.

We now want to move to the 0.5” position.

- Click the “Move To” button.
- Enter a position of 0.5”.
- Enter a speed of 1” per second.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the “Add At End” button to place the command in program.

We now want to wait 1 second.

- Click the “Wait” button.
- Enter a delay time of 1 second.
- No Label is required.

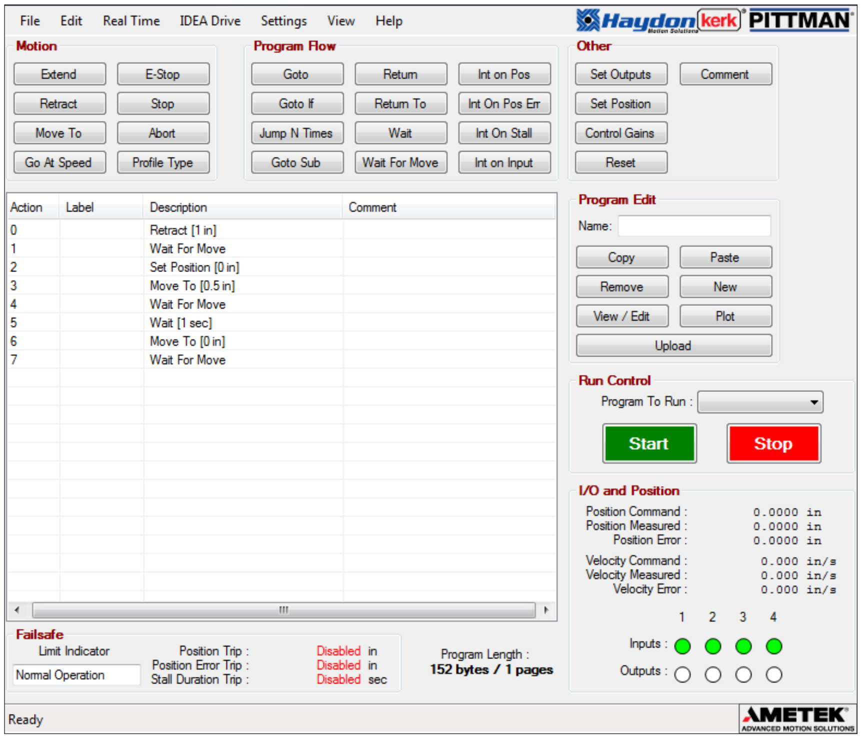
©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

- You can add a comment in the comment line if you wish.
- Click "Add at End" to place this command into the program.

We now want to move back to the 0" position.

- Click the "Move To" button.
- Enter a position of 0".
- All other parameters are populated.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Add at End" button to place the command in program.

The completed program looks as follows:



Action	Label	Description	Comment
0		Retract [1 in]	
1		Wait For Move	
2		Set Position [0 in]	
3		Move To [0.5 in]	
4		Wait For Move	
5		Wait [1 sec]	
6		Move To [0 in]	
7		Wait For Move	

```
Position Command :   0.0000 in
Position Measured :  0.0000 in
Position Error :     0.0000 in
Velocity Command :   0.000 in/s
Velocity Measured :  0.000 in/s
Velocity Error :     0.000 in/s
```

Limit Indicator: Normal Operation
Position Trip : Disabled in
Position Error Trip : Disabled in
Stall Duration Trip : Disabled sec

Program Length : 152 bytes / 1 pages

Inputs : 1 ● 2 ● 3 ● 4 ●
Outputs : 1 ○ 2 ○ 3 ○ 4 ○

Example Five

n this example, we will first find the fully retracted position of the motor within a linear application, then the motor will move to the 0" position if input 1 is high and input 2 is low, move to the 1" position if input 1 is low and input 2 is high, or stop if inputs 1 and 2 are both high, or both low.

We first want to retract the full stroke of the actuator.

- Click the "Retract" button.
- Input the distance as 1".
- Input the speed as 0.5 inch per second.
- All other parameters can be left as the defaults.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Add at End" button to place the command in program.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

NOTE: You will notice that when the retract command populates in the program field on the screen, that it is followed by a second command that states "Wait For Move". This is also true when using the Extend and Move To commands. This is a command to allow the move to finish before execution of the next command.

We are now at the fully retracted position; to keep track of this we will use the "Set Position" command.

- Click the "Set Position" button.
- Enter a position of 0".
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click "Add at End" to place this command into the program.

We now need the commands that will be used to move the motor. We will start with the "Retract" move.

- Click the "Move To" button.
- Enter a position of 0".
- Input the speed as 0.5 inch per second.
- Enter "Retract" as the label.
- You can add a comment in the comment line if you wish.
- Click "Add at End" to place this command into the program.

We will next add the "Extend" move.

- Click the "Move to" button.
- Enter a position of 1".
- Enter "Extend" as the label.
- You can add a comment in the comment line if you wish.
- Click "Add at End" to place this command into the program.

We now move based on the input status. Set up the first "Got If" Highlight the set position command and

- Click the "Goto If" button.
- Set Input 1 is "High".
- Set Input 2 as "Low".
- Set Inputs 3 and 4 as "Not Tested".
- All other parameters can be left as the defaults.
- Enter "Retract" as the destination.
- Enter "Test" as the label.
- You can add a comment in the comment line if you wish.
- Click the "Insert After" button to place the command in the program.

Set up the second "Goto If" Highlight the command labeled test then:

- Click the "Goto If" button.
- Select "Extend" as the destination.
- Set Input 1 is "Low".
- Set Input 2 as "High".
- Set Inputs 3 and 4 as "Not Tested".
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Insert After " button to place the command in the program.

If we reach this line, then either both inputs are high, or both inputs are low. So, we want to stop the motor.

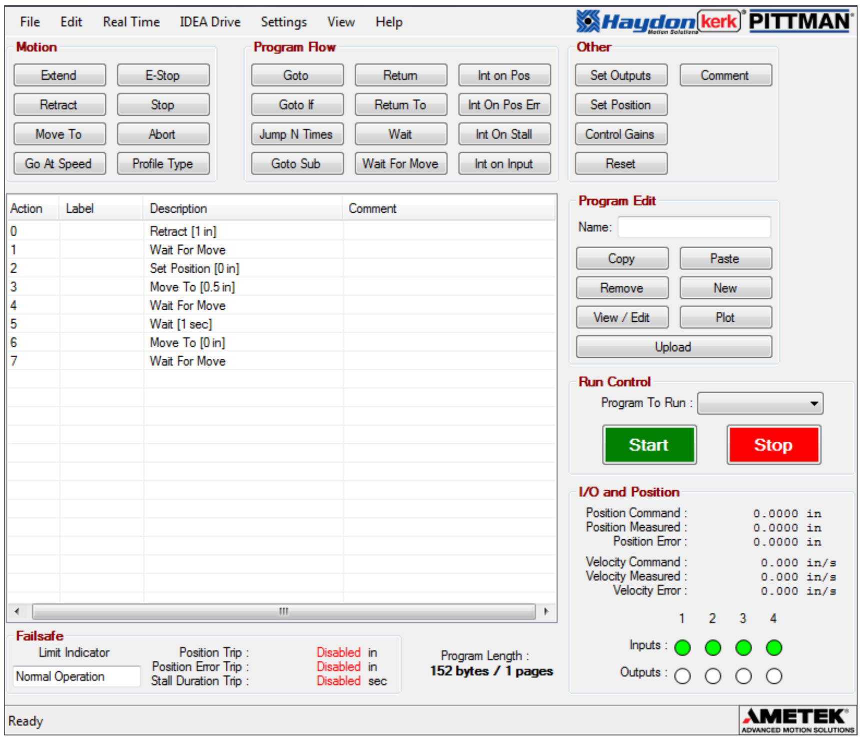
- Highlight "Goto If" extend and then click the "Stop" button.
- All parameters can be left at the defaults.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Insert After " button to place the command in the program.

We now want to check if the input conditions have changed, so we go back to the Goto Ifs.

- Highlight the last line (#9) and then click the "Goto" button.
- Select "Test" as the destination.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Insert After" button to place the command in program.
- Highlight line (#7) and then click the "Goto" button.
- Select "Test" as the destination.
- Click the "Insert After " button to place the command in the program.
- Highlight line (#5) and then click the "Goto" button.
- Select "Test" as the destination.
- Click the "Insert After " button to place the command in the program.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

The completed program looks as follows:



Action	Label	Description	Comment
0		Retract [1 in]	
1		Wait For Move	
2		Set Position [0 in]	
3		Move To [0.5 in]	
4		Wait For Move	
5		Wait [1 sec]	
6		Move To [0 in]	
7		Wait For Move	

```
Position Command :   0.0000 in
Position Measured :  0.0000 in
Position Error :     0.0000 in
Velocity Command :    0.000 in/s
Velocity Measured :   0.000 in/s
Velocity Error :      0.000 in/s

           1   2   3   4
Inputs :   ●   ●   ●   ●
Outputs :  ○   ○   ○   ○
```

```
Limit Indicator      Position Trip :        Disabled  in
Normal Operation     Position Error Trip :  Disabled  in
                     Stall Duration Trip :  Disabled  sec

Program Length :
152 bytes / 1 pages

Ready
```

In this example, we will first find the fully retracted position of the motor within a linear application. The motor will continuously move in the extend direction, until getting 0.9” from the fully retracted position. When the 0.9” position is reached, the motor will retract back to the 0” position, and resume the extend. If at any time during the program, input 1 changes state, the program will abort.

First, we want to retract the full stroke of the actuator.

- Click the “Retract” button.
- Input the distance as 1”.
- Input the speed as 0.5 inch per second.
- All other parameters can be left as the defaults.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the “Add at End” button to place the command in program.

NOTE: You will notice that when the retract command populates in the program field on the screen, that it is followed by a second command that states “Wait For Move.” This is also true when using the Extend and Move To commands. This is a command to allow the move to finish before execution of the next command.

We are now at the fully retracted position; to keep track of this we will use the “Set Position” command.

- Click the “Set Position” button.
- Enter a position of 0”.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click “Add at End” to place this command into the program.

We now want to begin moving.

- Click the “Go at Speed” button.
- Select “Extend” as the direction.
- Enter 1” per second as the speed.
- Enter “Extend” as the label.
- You can add a comment in the comment line if you wish.
- Click the “Add at End” button to place the command in program.

We now want to continuously loop onto the “Go at Speed” command.

- Click the “Goto” button.
- Enter “Extend” as the destination.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Add at End" button to place the command in program.

We now need the interrupt subroutines. We will start with the interrupt subroutine for the interrupt based on position.

- Click the "Move To" button.
- Enter a position of 0".
- Enter a speed of 1" per second.
- Enter "Retract" as the label.
- You can add a comment in the comment line if you wish.
- Click the "Add at End " button to place the command in program.

We now need a subroutine for the interrupt based on the input. Click the "Abort" button.

- Click the "Abort" Button.
- Enter "Abort" as the label.
- You can add a comment in the comment line if you wish.
- Click the "Add at End" button to place the command in program.

To force the actuator to go back to the 0" position when the 0.9" position is reached, we will use an interrupt based on position. Highlight the "Set position command and then:

- Click the "Int On Pos" button.
- Enter a position of 0.9".
- Enter "Retract" as the destination.
- Select "1" as priority.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click "Insert After" to place this command into the program.

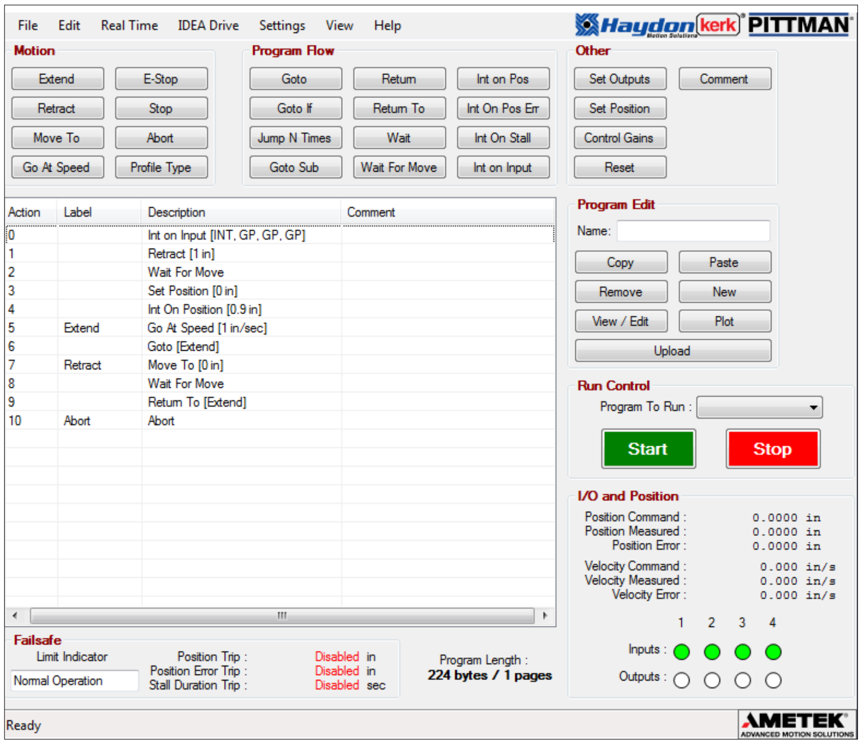
We now need to set up the interrupt triggered by the input. Highlight line 0 and then:

- Click the "Int on Input" Button.
- Select "Enabled" for Input 1 Interrupt
- Enter "Abort" as the destination.
- Select "Both Edges" as the trigger type.
- Select "0-Highest" as the priority.
- Click the "Insert Before" button to place the command in program.

We now need to exit the subroutine.

- Highlight line 8 and then click the "Return To" button.
- Enter "Extend" as the destination.
- No Label is required.
- You can add a comment in the comment line if you wish.
- Click the "Insert after" button to place the command in program.

The completed program looks as follows:



Action	Label	Description	Comment
0		Int on Input [INT, GP, GP, GP]	
1		Retract [1 in]	
2		Wait For Move	
3		Set Position [0 in]	
4		Int On Position [0.9 in]	
5	Extend	Go At Speed [1 in/sec]	
6		Goto [Extend]	
7	Retract	Move To [0 in]	
8		Wait For Move	
9		Return To [Extend]	
10	Abort	Abort	

Name:
Program To Run :

Limit Indicator: Normal Operation
Position Trip : Disabled in
Position Error Trip : Disabled in
Stall Duration Trip : Disabled sec

Program Length :
224 bytes / 1 pages

```
Position Command :   0.0000 in
Position Measured :  0.0000 in
Position Error :     0.0000 in
Velocity Command :   0.000 in/s
Velocity Measured :  0.000 in/s
Velocity Error :     0.000 in/s
          1  2  3  4
Inputs :  ●  ●  ●  ●
Outputs : ○  ○  ○  ○
```

Ready

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Glossary

Abort: Stops movement of the actuator with holding current and ends any running program.

Accel Current: Allow the user to implement up to a 30% increase in peak current per phase during the beginning of the acceleration ramp.

Accel Rate: The acceleration rate to be used with a move.

Clear: Clear the entire program from the program screen

Comment: Allows the user to insert comments within the program

Copy: Allows the user to copy a given program line or lines and insert them elsewhere in the program

Current position Box: Indicates the current position of the motor

Decel Current: Allows the user to implement up to a 30% increase in peak current per phase during the end of the deceleration ramp.

Decel Rate: The deceleration rate to be used with a move.

Delay Time (in reference to a move): The time between when the last step in a move profile is taken, and when the current it set to the hold current.

Delay Time (in reference to a "Wait" command): The amount of time that the wait command should delay execution of the next command.

Destination: The address to which the program should branch.

Distance: How far a move should go.

E-Stop: Abruptly stops the motor without any deceleration

Encoder: A feedback device that converts position data to an electronic signal that the drive keeps track of.

Encryption: programs can be saved as encrypted files.

Extend: Extends the motor within a linear system forward, or clockwise. The user inputs the distance and speed. Items such as run current and hold current are auto populated based on the characteristics of the motor. These values can be over ridden provided the inserted value does not exceed the devices limitations.

Failsafe Limit: Used only in real-time mode. To enable the failsafe limits, check the position trip, position error trip, and/or stall duration trip.

Go at Speed: Extends (CW) or retracts (CCW) the motor at a given speed indefinitely.

Goto: Branching statement for programming. Branches to a destination label.

Goto if: Branching statement for programming. Branches to a destination label if the input conditions are met.

Goto Sub: Branching statement for programming that navigates the program to a subroutine.

Hold Current: The maximum, peak current limit applied to the phases of the motor which the drive may use to hold the load in position when the motor is at a standstill.

I/O Box: Displays the state of each general purpose I/O.

Int on Input: Interrupt on Input. This is an interrupt that occurs when an input changes state.

Int on Pos: Interrupt on Position. This is an interrupt that occurs when the encoder reaches a specific position.

Int on Pos Err: Interrupt when the set position error is surpassed.

Int on Stall: Interrupts when the duration of the current limit is exceeded.

Interrupt: An asynchronous event that causes the execution of a subroutine.

Jump N Times: Allows the program to jump N times to a specified label

Label: A string that identifies a command. Used to branch to the command.

Move To: Moves the motor to a specified position.

Oscilloscope: Motion commands characteristics can be checked it is helpful in adjusting the desired control gains.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654

Paste: Used in conjunction with the copy or cut functions. After one or more commands are selected, another location in the program is then highlighted by the user. The paste button is then pressed and the line or lines are inserted above the highlighted line.

Plot: Any move can be shown as a plot of speed vs time. Highlight the move command of interest then press the plot button.

Position: A location based upon the encoder counter.

Priority: The determining factor in which interrupts are serviced first.

Profile Type: Used to set motion profile settings; trapezoidal or s-curve.

Program Name Box: This is the location on the screen where the name of the program is inputted by the user. The program is stored in the drive under this name.

Program to Run Box: This is a drop-down menu that list the programs stored on the drive. Double clicking on a given name populates the program into the program screen and creates that program as the active program in the drive.

Remove: This is used to remove one or more lines from a program. Highlight the lines to be removed. Then click the remove button.

Reset: This command simulates turning the drive off and on again.

Retract: Retracts the motor within a linear system reverse, or counter clockwise. The user inputs the distance and speed. Items such run current and hold current are auto populated based on the characteristics of the motor. These values can be over ridden provided the inserted value does not exceed the devices limitations.

Return: Used in conjunction with the goto sub command or interrupts. At the end of a subroutine the “return” command returns to the program to the very next line after the Goto sub line command, or the command that was going to be executed before the interrupt was triggered.

Return To: Used in conjunction with the goto sub command or interrupts. At the end of a subroutine the “return to” command returns to the command at a specified label location.

Run Current: The maximum allowable, peak current to be applied to the motor phases during a move.

Set Outputs: Allow the programmer to set general purpose outputs.

Set Position: Used to change the current position. Using this command, the position counter can be adjusted, usually after a homing routine. Then other commands such as “move to” or “interrupt on position” can be used in relationship with this set position.

Speed: The desired top speed for a move.

Start (large green button): Starts running a program.

Stop: Stops the movement of the motor with a specified deceleration and actively holds the resting position.

Stop (large red button): Immediately aborts motor operation and does not hold resting position.

Subroutine: A section of code used that is entered by using an interrupt, or the “Goto Sub” command. Subroutines must end with a “Return” or “Return To” command.

Upload: Allows the program to be uploaded into drive

View / Edit: After highlighting a specific line in a program, the “View / Edit” button can be pressed causing the details for that line to display. These details can then be modified and updated.

Wait: Allows the programmer to put in a time specific time delay.

Wait for move: Delays execution of the next line of the program until the motor has come to a stop.

©AMETEK, Inc. All rights reserved. Not responsible for any typographic errors. Specifications subject to change. Document #33-654 MCM2019_025A_012019