ARTISAN®
TECHNOLOGY GROUP

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

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

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

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

Custom engineering
so your equipment works exactly as you specify.

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

Expert team | Trust guarantee | 100% satisfaction

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

Find the *Parker / Compumotor PDFX15* at our website: *Click HERE*

Parker
Automation

CE

PDFX Series
Mini Stepping Drive
User Guide



Part No: 1600.256.02 July, 2000 (For software revision 1.7 onwards)

IMPORTANT INFORMATION FOR USERS

Installation and Operation of Motion Control Equipment

It is important that motion control equipment is installed and operated in such a way that all applicable safety requirements are met. It is your responsibility as an installer to ensure that you identify the relevant safety standards and comply with them; failure to do so may result in damage to equipment and personal injury. In particular, you should study the contents of this user guide carefully before installing or operating the equipment.

The installation, set-up, test and maintenance procedures given in this User Guide should only be carried out by competent personnel trained in the installation of electronic equipment. Such personnel should be aware of the potential electrical and mechanical hazards associated with mains-powered motion control equipment - please see the safety warning below. The individual or group having overall responsibility for this equipment must ensure that operators are adequately trained.

Under no circumstances will the suppliers of the equipment be liable for any incidental, consequential or special damages of any kind whatsoever, including but not limited to lost profits arising from or in any way connected with the use of the equipment or this user guide.



SAFETY WARNING

High-performance motion control equipment is capable of producing rapid movement and very high forces. Unexpected motion may occur especially during the development of controller programs. ***KEEP WELL CLEAR*** of any machinery driven by stepper or servo motors. Never touch any part of the equipment while it is in operation.

This product is sold as a motion control component to be installed in a complete system using good engineering practice. Care must be taken to ensure that the product is installed and used in a safe manner according to local safety laws and regulations. In particular, the product must be enclosed such that no part is accessible while power may be applied.

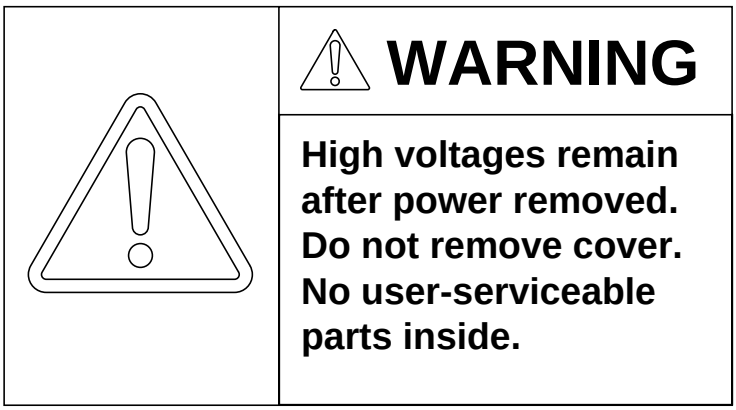
If the equipment is used in any manner that does not conform to the instructions given in this user guide, then the protection provided by the equipment may be impaired.

The information in this user guide, including any apparatus, methods, techniques, and concepts described herein, are the proprietary property of Parker Electromechanical Division or its licensors, and may not be copied, disclosed, or used for any purpose not expressly authorised by the owner thereof.

Since Parker Electromechanical constantly strives to improve all of its products, we reserve the right to modify equipment and user guides without prior notice. No part of this user guide may be reproduced in any form without the prior consent of Parker Electromechanical Division.

© Electromechanical Division of Parker Hannifin plc, 2000
– All Rights Reserved –

The following Warning label is fitted to the drive:



Symbols used on the PDFX series of drives have the following meanings:



Refer to the accompanying documentation

Contact Addresses

For engineering assistance in Europe:
Parker Hannifin plc
Electromechanical
Division - Digiplan
21 Balena Close
Poole, Dorset
England, BH17 7DX
Tel: +44(0)1202-699000
Fax: +44(0)1202-695750
E-mail: tech.help@parker-emd.com

For engineering assistance in Germany
Parker Hannifin GmbH
Electromechanical
Division - Hauser
P. O. Box: 77607-1720
Robert-Bosch-Str. 22
D-77656 Offenburg, Germany
Tel: (0781) 509-0
Fax: (0781) 509-176

For engineering assistance in the U.S.:
Parker Hannifin Corporation
Compumotor Division
5500 Business Park Drive, Suite D
Rohnert Park
CA 94928
USA
Tel: (800) 358-9070
Fax: (707) 584-3793
FaxBack System: (800) 936-6939
BBS: (707) 584-4059
E-mail: tech_help@cmotor.com

CE

Product Type: **PDFX13 & PDFX15**

The above product is in compliance with the requirements of directives

- **73/23/EEC** **Low Voltage Directive**
- **93/68/EEC** **CE Marking Directive**
- **89/336/EEC** **Electromagnetic Compatibility Directive**

Provided the installation requirements described in this user guide are met, and there are no special requirements of the installation and operating environment so that the application may be considered typical, the PDFX drive series installation will conform to the protection requirements of Council Directive 89/336/EEC as amended by Directive 92/31/EEC on the approximation of the laws of the Member States relating to Electromagnetic Compatibility when operated and maintained as intended.

In assessing the overall compliance of an installation consideration must also be given to the effects of mains harmonics and flicker when interfacing the total supply system to the public low voltage supply system.

Contents

Section 1. Introduction....1
Section 2. Getting Started....3
Section 3. Installation....19
Section 4. Hardware Reference....37
Section 5. Control of the PDFX....43
Section 6. Command Reference....91
Section 7. Maintenance and Troubleshooting....141
Index....147

User Guide Change Summary

This user guide, version 1600.256.02, is the second version of the PDFX Series Mini Stepper Drive User Guide.

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

Major changes in revision 02 include:

Analogue input speed control
FRATE command added
Changes to MODE command
Change in the value of fuses FS2 & FS3
Software release 1.7 updates
EASI-tools release 1.2 updates
Motor wiring identification tables added

Section 1. Introduction

Product Description

Packaged mini stepper PDFX drives, offer a cost effective solution to many single axis applications requiring indexer control. Designed to be used with existing general purpose stepper motors, the drives can be used Direct-On-Line and are fully LVD compliant and EMC compliant when used with an external line filter. Two models are available, the higher power version having a power dump option:

PDFX13 providing a motor voltage of 75V at 2.5A peak*
PDFX15 providing a motor voltage of 75V at 5.0A peak*
PDFX15-D same as PDFX15 with power dump

*Note: default settings are 50% of peak continuous output current

An integrated indexer offers precise control of moves and sequences using a mix of parameter driven and event driven commands. A 24V DC I/O system provides high noise immunity, with 10 configurable inputs and 8 configurable NPN (sourcing) outputs. Control is established using RS232, RS485 or via the Fieldbus.

Software design enables application programs to be written with a minimal amount of user code. Pre-defining move profiles greatly reduces the amount of repetitive code that needs to be generated and the REG command allows registration moves to be accomplished in just a few lines. Event driven code also simplifies code generation by removing the need for long conditional testing sequences.

Product Features

Protection Circuits
- Motor short circuits
- Overvoltage
- 24V reverse supply protection
- Controller outputs short circuit protected

Function Indicators
- Drive Status (HV)
- Indexer Status (Status)
- Fieldbus Status (FB)

Adjustments
(Bit switch)
- Echo ON echo OFF
- Baud rate 9600/19200
- Address selection 1 to 63

Outputs and Inputs
- 10 digital inputs
- 8 logic user outputs
- 2 analogue inputs
- Primary encoder input
- Auxiliary encoder output

Other Features
- Optional power dump

Specified as PDFX15-D

Controls and Indicators

LEDs

Three tri-coloured LEDs are used for monitoring the overall state of the drive. From left to right the LEDs are:

LED legend	Green	Orange	Red
HV (High Voltage)	HV OK	HV OK, Zero phase	Fault
Status of Indexer	Ready & energised	In Standby, but healthy	Fault
Fieldbus	Comms. OK	Reserved*	Failure

Table 1-1. Front Panel LED Display Information

* Refer to the documentation describing the fieldbus being used.

Bit Switch

The front panel 8-way bit switch is used for selecting communication options and the address of the drive.

Drive Configuration

Configuration data such as drive resolution, standby current reduction and the peak continuous output current are held as named system variables which need to be setup, in software, before the drive is energised.

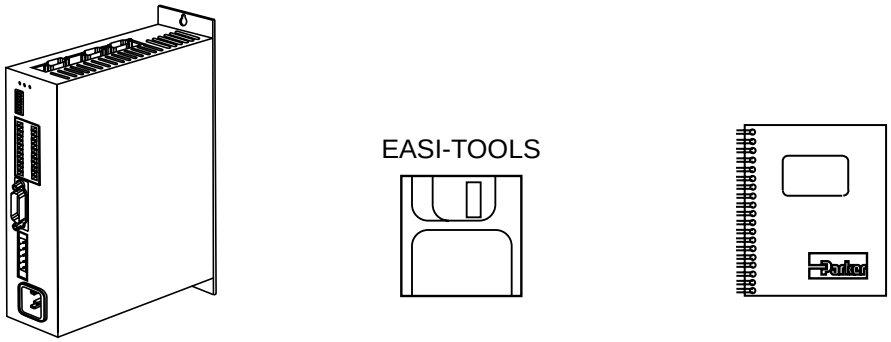
Default Settings

Upon delivery, the default settings are:

Drive resolution (MR) **4000 step/rev**
Standby current reduction (MS) **50% (minimum)**
Peak continuous output current (MC) **50% (minimum)**

Section 2. Getting Started

What You Should Have



DRIVE **SOFTWARE** **USER GUIDE**

Figure 2-1. Items Supplied

Upon receipt, you should inspect your PDFX Series Drive system for obvious damage to the drive or its container. **Report any damage** as soon as possible to your supplier.

Drive

PDFX13, 75V 2.5A peak ministepper drive
PDFX15, 75V 5.0A peak ministepper drive
PDFX15-D, PDFX15 with the power dump option fitted

Software

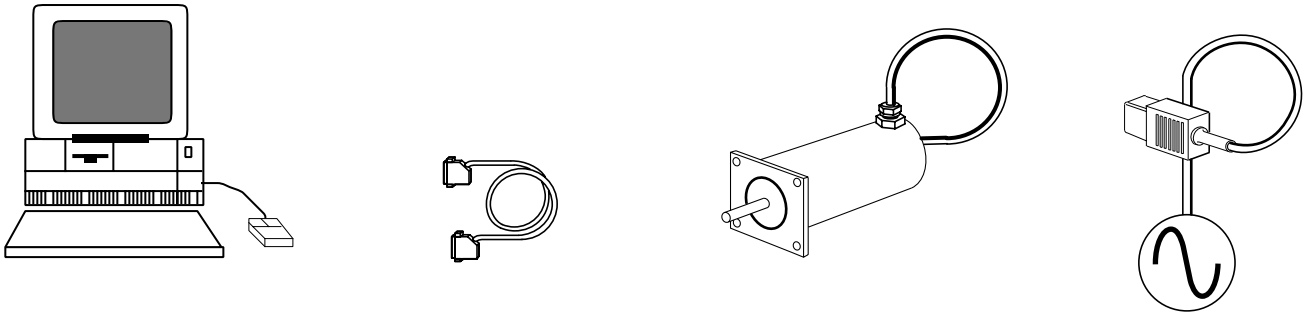
The PDFX uses EASI-TOOLS software supplied on one 3.5 inch diskettes, which runs in a Windows™ environment.

User Guide

The PDFX Series Mini Stepping Drive **User Guide** is supplied in A5 spiral bound format. The document has a part number 1600.256.XX, found at the bottom of the first page. The XX denotes an issue number, the first being 01.

Getting it Working

If you are unfamiliar with this drive, it may be worth performing a pre-installation test to help you become accustomed to its operation and to give you confidence that the drive is working correctly. In addition to the drive, software and user guide, you will also need the following:



PC | **SERIAL LINK LEAD** | **STEPPER MOTOR & CABLE** | **AC SUPPLY**

Figure 2-2. You Will Also Need

Computer Requirements

To be able to run EASI-TOOLS software, necessary for the control and programming of the PDFX, you will require an IBM™ compatible PC running Windows 95/98™ or NT4. A Windows 3.1/3.11™ version is available upon request.

The PC needs to be specified to run Windows 95™ with at least 16MB of RAM, a VGA monitor and a Windows™ compatible mouse. The installed program size is approximately 800K.

Serial Link Lead

If the computer being used is fitted with 9-way D-type COM connectors, you will need a **2-wire plus ground** 'RS232 style' lead which has the **Rx and Tx wires crossed over**. Wiring details are given in the Installation section.

Note: These serial connections do not follow standard RS232 pinouts. Do not attempt to use a null modem cable.

If a 25-way communications port is being used refer to the Installation section for details of plug wiring.

Stepper Motor

The PDFX drive is designed to be used with **general purpose stepper motors up to a frame size of 34**. Provided the motor's minimum inductance is not below 0.7mH and it will not be damaged by the current supplied by the drive, any series or parallel connected motor may be used. The recommended motor inductance is 0.8 to 10mH.

Note: the drive is supplied with the motor current set to a minimum (50% of peak output current), therefore using the PDFX13 the minimum motor current is 1.25A and using the PDFX15 the minimum motor current is 2.5A. Check that this level of current will not harm your motor.

AC Supply

The drive can be powered from a **single phase AC supply** with a voltage of **110V to 240V** RMS ±10%.

Pre-installation Test

Before making any connections to the drive, look at the front panel bit switch settings, for RS232 communications these should be set as follows:

Echo	ON
Baud rate	9600 (OFF)
Address	1

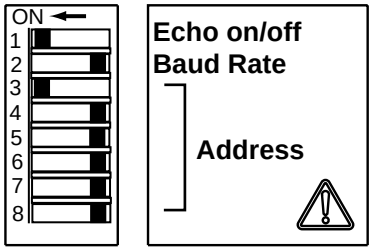


Figure 2-3. Test Settings

Test Connections

The drive is wired as shown in Figure 2-4. Take care to securely fix the motor in position using a supporting 'V clamp' or by firmly attaching the motor to the top of the test bench. A safety earth connection should also be made to the motor's outer casing.

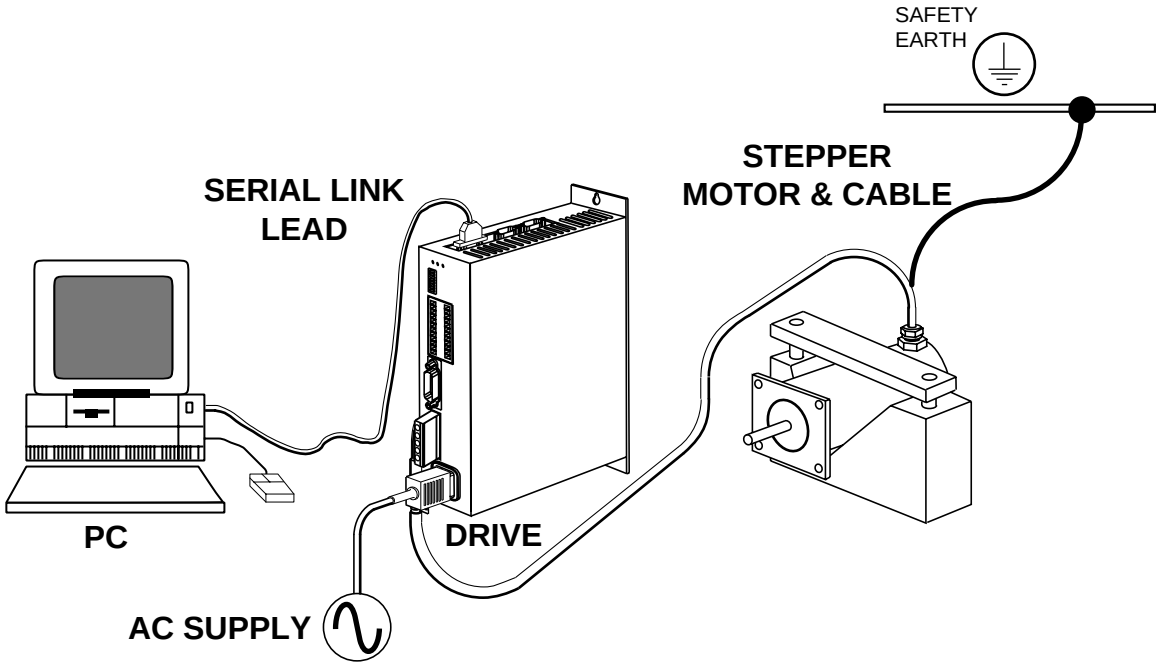


Figure 2-4. Test Connections

Motor Connections

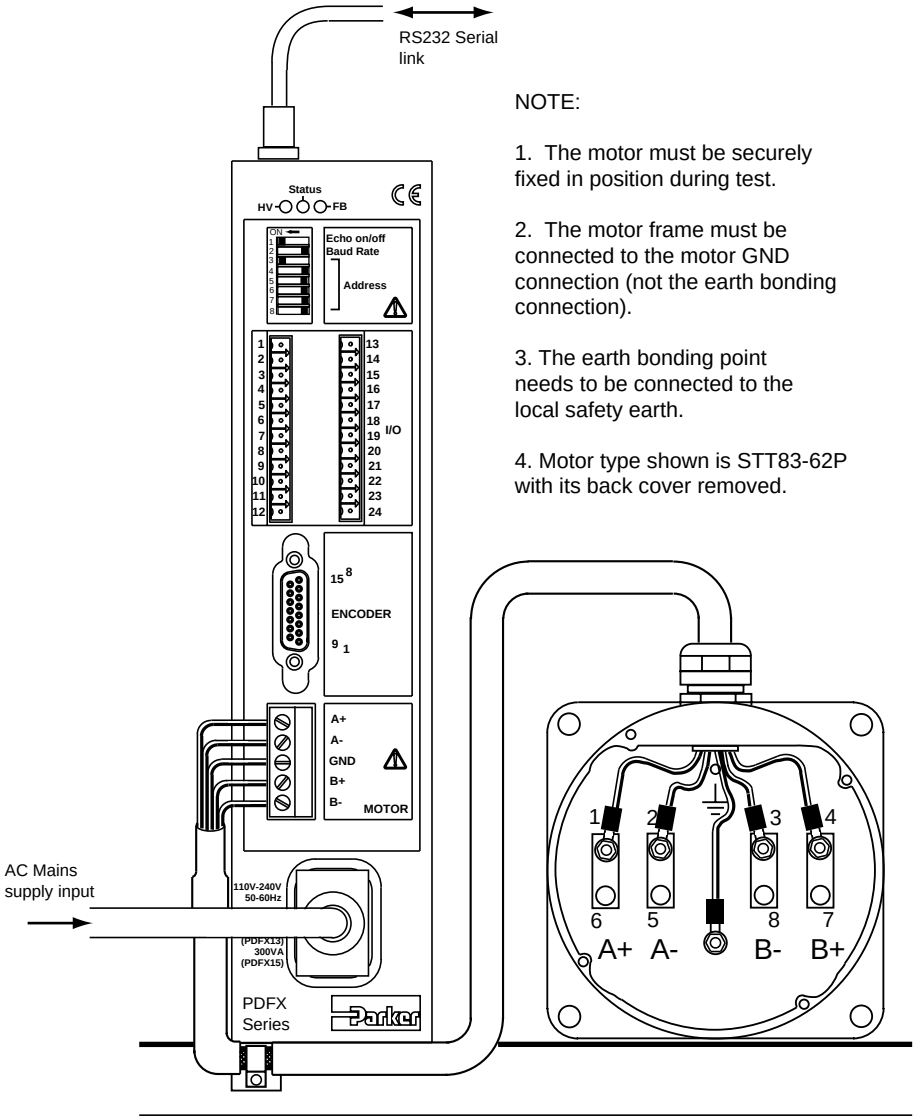


Figure 2-5. Motor Connection Details

Assuming the drive is still configured with factory settings, you can perform the following pre-installation test.
Once all test connections have been made and checked, apply power to the PC and drive. The drive will power up in a non active state with the HV LED emitting a green light. Drive operation is entirely controlled via serial communications using RS232 or RS485. No motor moves can be made until serial communications have been established and commands are sent to the drive.

Establish Communications

Before attempting communication with the drive the supplied software needs to be installed on to the PC's internal hard disk drive. Installed space is less than 1MB.

Once software installation is complete, commands can be downloaded from the PC to the drive to confirm the operation of the PDFX. If EASI-TOOLS has already been loaded you may skip the following Installation and Operation sections up to ***Confirming Drive Operation***.

Software Installation

Before attempting to install the EASI-TOOLS software supplied with your PDFX check that your PC meets the requirements previously defined under ***Computer Requirements***. EASI-TOOLS software is supplied on one 3.5 inch diskette and installs in the usual manner common to Window™ applications.

Installation Procedure

This procedure takes you quickly through the steps necessary to install EASI-TOOLs on your PC. The entire installation process takes less than 10 minutes. Before starting the installation, terminate all applications currently running.

A step by step installation of EASI-TOOLS software follows:

1. Place the EASI-TOOLS installation disk in drive A.

2. Click on the Windows™ START BUTTON and select menu option RUN. In the command line type: **a:\setup** and press return.

Note: If you do not have the menu option RUN, navigate to drive A: and double click the **setup.exe** file.

3. The screen will display the EASI-TOOLS program banner and will prepare an installation setup program.

4. The banner screen is automatically replaced by a Welcome dialog box advising you of the need to exit any programs currently running. To abandon setup in order to exit other programs, select CANCEL. This in turn displays an Exit Setup dialog box giving you the options of Exit Setup, which returns you to Windows™ or Resume which takes you back to the Welcome box.

5. Selecting NEXT> displays a ‘Choose Destination Location’ dialog/selection box which provides the option of installing EASI-TOOLS in the directory of your choice. The default directory is **c:\Parker\EASITool**, other buttons are described within the dialog box, see Figure 2-6.

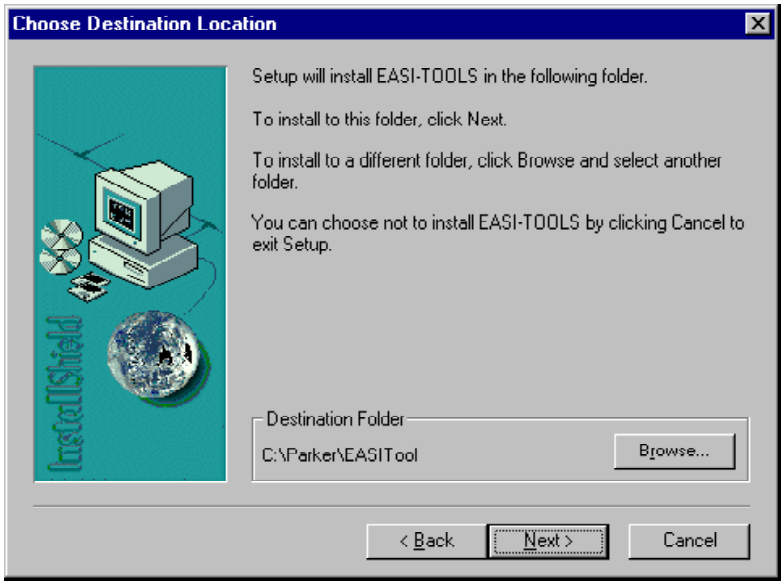


Figure 2-6. Choosing Where to Install EASI-TOOLS

6. Once you have selected a destination for EASI-TOOLS or have decided to use the default directory, select NEXT to begin file transfer.

7. Once EASI-TOOLS has been loaded, the screen will display a message dialog box, stating 'Setup is complete. You may run the installed program by double-clicking on the program icon.' When you click the OK button, the window shown in Figure 2-7 will appear. Note: EASI-TOOLS may also be run from the Start menu.

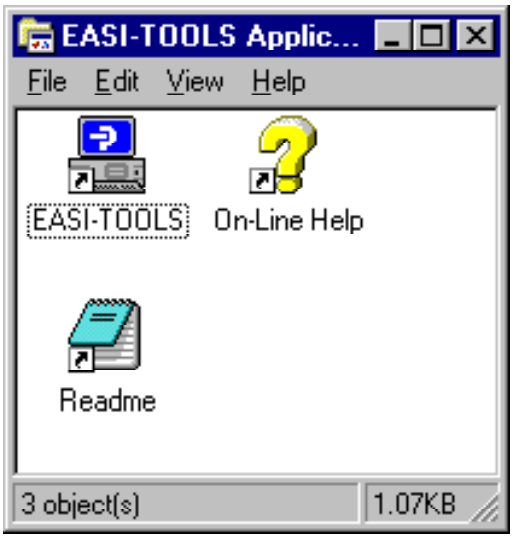


Figure 2-7. EASI-TOOLS Application Window

Uninstalling EASI-TOOLS

To uninstall EASI-TOOLS software, use Windows™ uninstall software facilities available within the Control Panel. All components are removed.

Software Operation

Once installed, EASI-TOOLS can be started from the start menu or by double clicking its application icon. At startup EASI-TOOLS shows the version number being used.

The main application window is displayed, entitled 'Parker Hannifin EMD - EASI TOOLS', and seven pull-down menus become available:

File, Edit, Search, Terminal, Utilities, Windows, Help

The majority of options available within each menu are familiar to Window™ users and will not be fully described here, but options available within Terminal and Utilities are specific to drive control and will be fully described.

Menu Overview

File



Filing Operations

Creates a new editor file, or .prg program file

Opens an existing editor file or program

Save an editor file

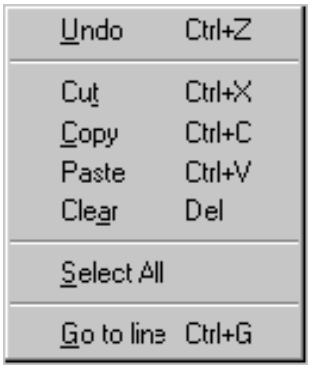
Save an editor file specifing the file name

Print the editor file or contents of terminal buffer

Close current active window

Exit EASI-TOOLS

Edit



Editing Operations

Undo a previous edit (1 level of undo only)

Remove highlighted text to clipboard
Copy highlighted text to clipboard
Paste contents of clipboard to current cursor location
Delete highlighted text

Highlight all text in active editor file window

Go to a particular line within a file*

Search



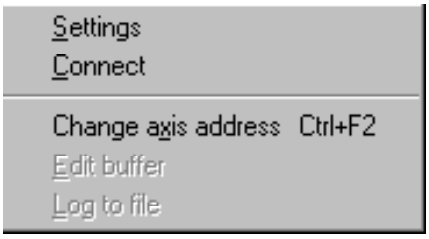
Editor search & replace operations

Find specified text (top down from cursor position)

Repeat search again

Find and replace text (top down from cursor position)

Terminal



Terminal on-line operations

Configure the serial communications

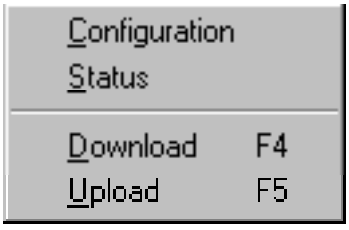
Open/close the terminal (after testing the connection)

Change address on multi axis systems

Create an editor file window or terminal buffer

Open/close file logging terminal buffer actions

Utilities



Specific tools

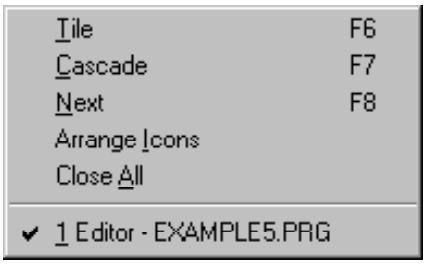
Setup drive for operation

Report of user drive faults and status

Transmit current active editor contents to drive

Upload drive program(s) to new editor file window

Windows



Window controls

Share program desktop space between open windows

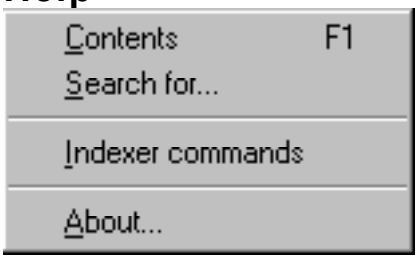
Cascade all open windows on program desktop

Select/activate the next window

Arrange all minimised windows on program desktop

Close all active windows on program desktop

Help



Program help facilities

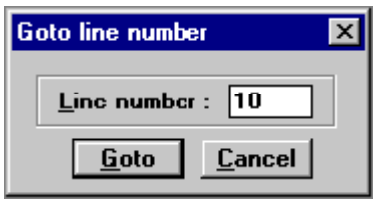
Open help file at the main contents (start)

Prompt for topic string and search help file

Open help for individual EASI commands

EASI-TOOLS version number and copyright

* Selecting ‘Go to line’ from the edit menu will generate the following dialog box, which allows the required line number to be entered. This is useful for locating errors when loading a program.



Communicating with a Drive

This sub-section assumes a correctly wired communications link has been established between the drive and PC and the required address bit switch settings have been made.

Note: Communications can use RS232 or RS485 ports. If you wish to use an RS485 port you will need to install the necessary hardware and software on your PC.

WARNING

To avoid causing damage to a PC serial port the drive must be earthed before making any serial connections.

Configure the serial communications

From the Terminal menu choose Settings to display the following Communications port configuration dialog box.

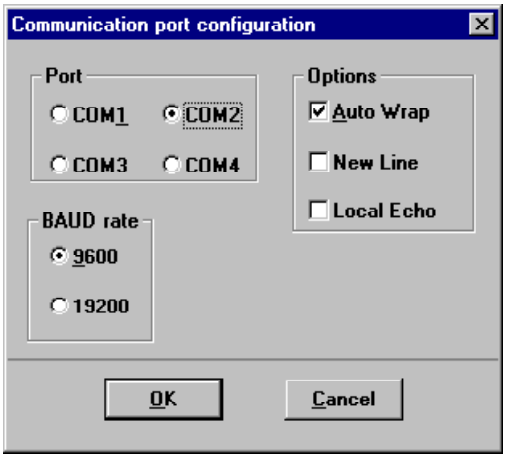


Figure 2-8. Comms Port Configuration

The default settings used are:

Port	COM2
BAUD rate	9600
Options	Auto wrap

The option New Line only needs to be selected if drives other than the PDFX are being used.

Once the required configuration has been selected, click OK. Then, again from the Terminal menu select Connect to start communications. Every time Connect is issued the communications link is tested to establish it is working correctly and the message box 'Testing communications integrity' is flashed on the screen, followed by 'Now on-line to indexer' if the Connect is successfully made. If the link fails refer to the Trouleshooting Section. Note: The baud rate selected must agree with the drive's hardware selected value.

You are now ready to start creating and editing program (.prg) files to control the operation of a connected drive. A number of example files are included within the EASI-TOOLS installation package to give you a start with drive programming.

Note: If you are using a PC with a serial mouse avoid IRQ conflicts between serial ports by selecting a serial port with a different IRQ to that used by your mouse. Comm ports 1 and 3 share the same IRQ as do ports 2 and 4. Usually, if your mouse is connected to COM1, connect the drive to COM2.

Change Axis Address

The Change axis address menu option allows you to change axes in a multi axis system. Selecting the menu option displays the dialog box shown in Figure 2-9.



Figure 2-9. Change Axis Address Tool

Drive Configuration

Assuming successful communications have been made, you can now begin to configure your drive using the Configuration dialog box selected from the Configuration option of the Utilities menu. This box is arranged as a series of tabs giving you access to six configuration options, as shown in Figure 2-10.

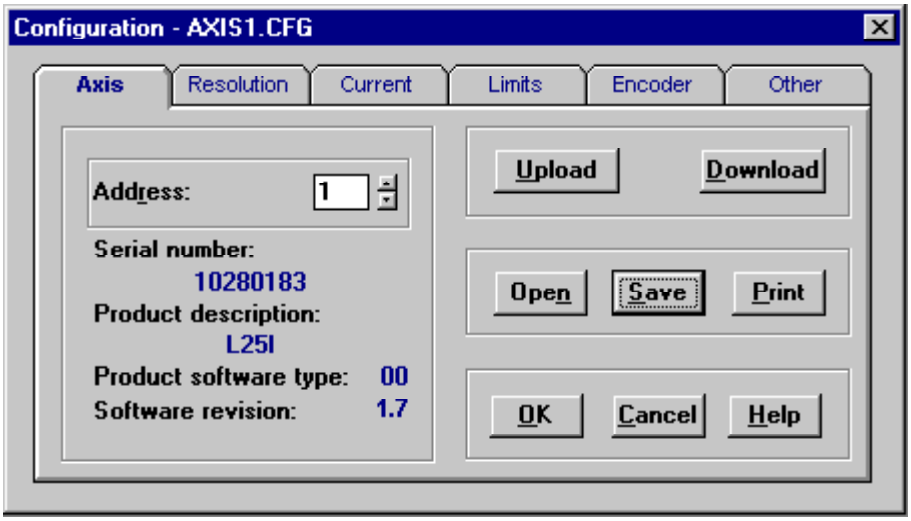


Figure 2-10. Drive Configuration Settings

The right hand side of this box is common to all tab options and will only be described once under Axis Configuration.

Axis Configuration

This window allows you to address a particular drive having the axis address (set within the drive) displayed in the increment/decrement Address box. When different axes are addressed, the software revision and serial number of the drive being addressed will be displayed.

Download Button

When all of the configuration information has been set for every tab position the Download button is used to transfer configuration information to the drive on the presently addressed axis.

Upload Button

Activating this button will upload configuration information from the currently addressed drive to the EASI-TOOL screens, allowing a particular axis configuration to be examined.

Configuration Files

The three buttons Open, Save and Print are all associated with drive configuration files, that allow the configuration of a particular drive to be saved to disk as a .cfg file.

Open
Allows you to search for a particular configuration file that may be opened, edited, downloaded or saved.

Save
Allows the current configuration information to be saved to file in a directory of your choice.

Print
Allows the current configuration file to be printed. The printout lists axis details, motor setup, end of travel limits and other parameters.

OK

Is selected when you have finished making changes to the configuration.

Cancel

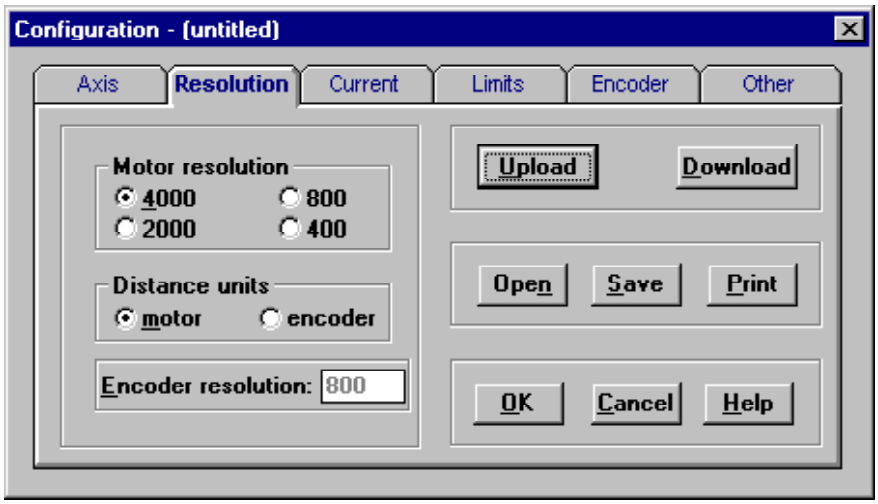
Is used to de-select the configuration dialog box.

Note: when you select either OK or Cancel and changes have been made to the configuration, you are prompted with a message asking if you wish to save the changes to disk.

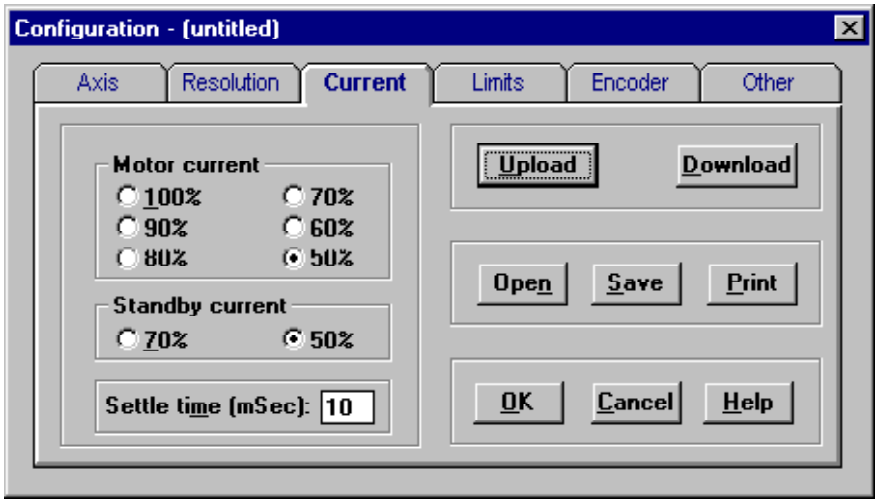
Other Configuration Options

The following configuration screens illustrate the range of configuration options possible within EASI-TOOLS.

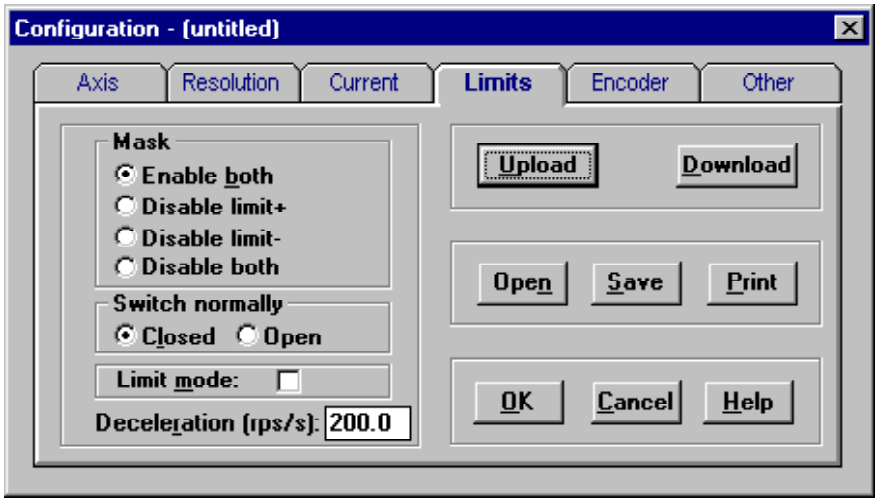
Resolution Configuration



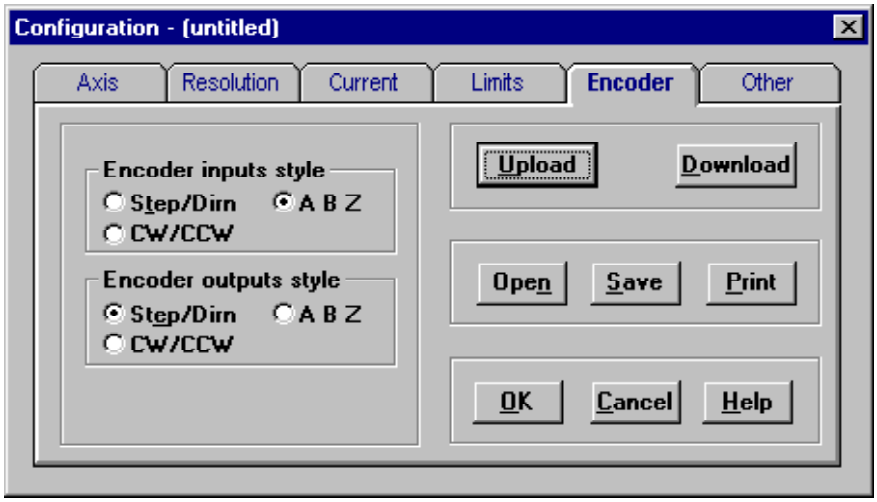
Current Configuration



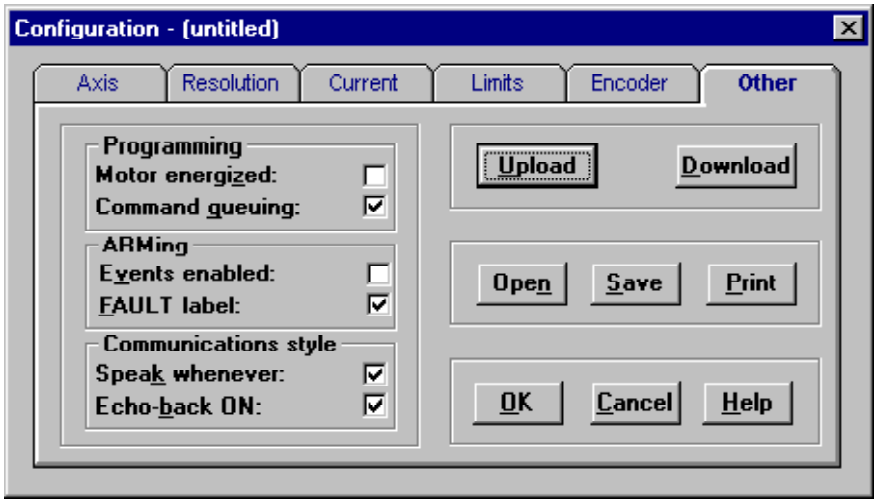
Limits Configuration



Encoder Configuration



Other Configuration



Status

The Utilities menu option Status provides a convenient method of examining the double word status bits. The tool gives access to the status of User Faults, Status bits and Drive Faults using a series of tabs, as shown in Figure 2-11.

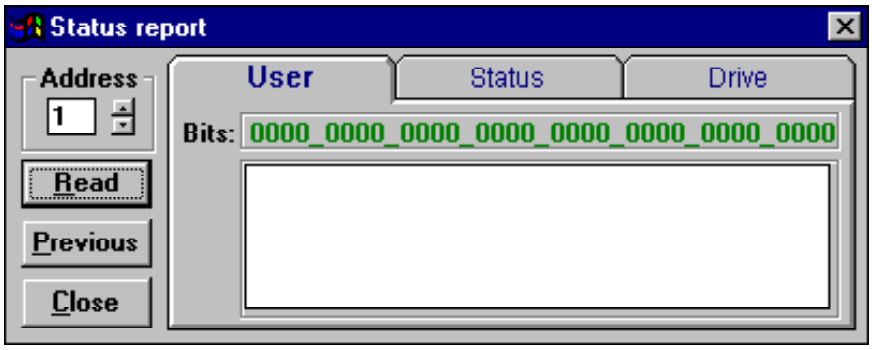


Figure 2-11. User Fault Status Reporting

The Status Report can be permanently displayed during program development or testing to monitor the operation of the drive. The double word status bits are decoded and displayed as text messages within the Status Report window, as shown in Figure 2-12. This eliminates manual decoding errors and gives an immediate update of the drive's status.

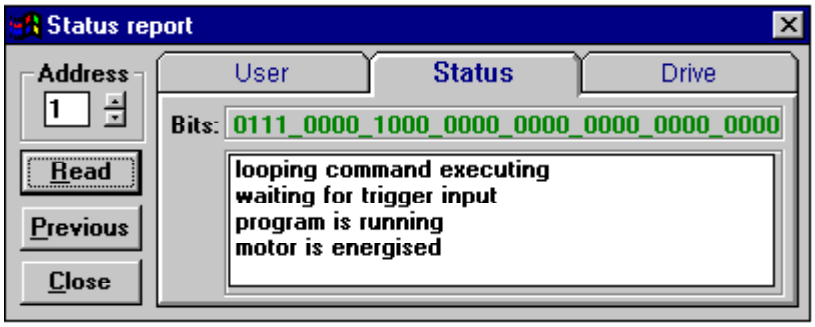


Figure 2-12. Status Reporting

Read

The Read button is used to update all of the Status Reports and is a useful aid when debugging an application.

Previous

Selecting Previous allows the previous status to be re-displayed - useful for comparing the results of programming actions. The Previous reading is only stored to a depth of one, that is, you cannot trace the history of status bits by continually selecting the button.

Close

Selecting Close will exit the Status Report window.

Downloading and Uploading Programs

A drive program that exists within an active edit window can be downloaded to the drive by selecting Download from the Utilities menu. The usual communication checks will be made and if all is OK the program will be downloaded to the drive's internal memory. A shortcut download is provided by function key F4.

A program may also be uploaded from a drive, a useful facility if a drive needs to be swapped between axes. To upload a program select Upload from the Utilities menu. An upload dialog box will be displayed, allowing you to specify the name and address of the program to be uploaded. A shortcut upload is provided by function key F5.

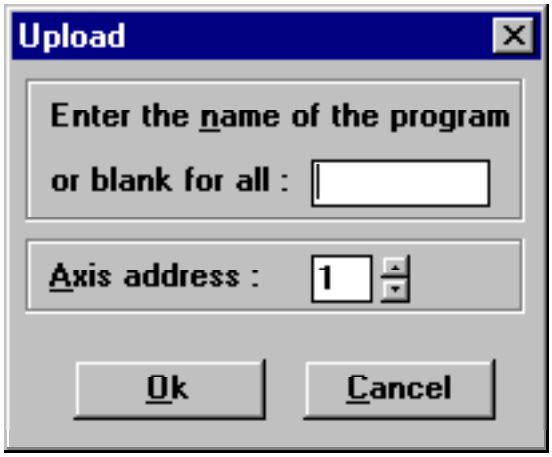


Figure 2-13. Upload Dialog Box

If you receive an error message during program upload refer to the Troubleshooting Section.

Help

EASI-TOOLS has extensive on-line help facilities which allows you to search for help on a particular topic either within the main contents or by entering a topic string. All the commands listed within this user guide are available on-line by selecting Indexer Commands from the Help menu.

Confirming Drive Operation

With the drive and motor wired as described at the beginning of this Section and the serial connection made to a PC running EASI-TOOLS software, the operation of a drive may be confirmed by creating and downloading the following code:

```
1OFF            ; turn OFF before performing RFS
1RFS            ; return to factory settings
.
1START:         ; define start label code
1ON             ; enable the drive
1LIMITS(3,0,0)  ; disable limits
1D4000          ; set distance to 4000 steps (1 rev)
1V1             ; set velocity to 1rev/s
1A10            ; set acceleration to 10rev/s2
1G              ; start motion
1END            ; end definition of start block
1GOTO(START)    ; execute start code block
```

WARNING

The motor must be clamped in a secure position before testing the drive.

Upon execution of this code, the motor should perform 1 revolution and stop. The successful operation of this code confirms the drive is working correctly. If this does not happen refer to the ***Troubleshooting*** Section.

Section 3. Installation

Installation Safety Requirements

PDFX Series drives are designed to meet the requirements of the EU Low Voltage Directive when installed according to the instructions given in this User Guide. However, use of these drives does not guarantee system compliance, it is your responsibility (as a system builder) to verify overall system compliance with LVD. The drive must be installed in an enclosure to protect it from atmospheric contaminants such as oil, moisture, dirt and dust. No operator access should be allowed to the drive while it has AC power applied. Metal equipment cabinets are ideally suited for housing the equipment since they can provide operator protection, EMC screening, and can be fitted with interlocks arranged to remove all AC power when the cabinet door is open.

Power Dump Version

The PDFX15-D is designed for driving high-inertia loads where the application demands rapid deceleration. The power dump resistor built into the PDFX15-D drive cannot be retro fitted. For details of when a dump resistor is required see the Maintenance and Troubleshooting section.

Environment

The drive requires a mains supply of Installation Category II and can be used in an atmosphere of Pollution Degree 2, which means only dry, non conductive pollution is acceptable. An operating temperature of 0° to 50°C Max. can be tolerated and humidity limits are 5-95% non-condensing. The storage temperature limits are -20 to 70°C.

Note: Installation category (also called Overvoltage Category) specifies the level of mains voltage surges which the equipment will be subjected to. The category depends upon the location of the equipment, and on any external surge protection provided. Equipment in an industrial environment, directly connected to major feeders/short branch circuits, is subjected to Installation Category III. If this is the case, a reduction to Installation Category II is required. This can be achieved by use of an isolating transformer with an earthed screen between primary and secondary, or by fitting varistors from live to neutral and from neutral to earth. Varistors for a 240V AC input should be rated to absorb at least 100 Joules.

Drive Cooling

The drive system is cooled by a combination of natural convection and forced air. To aid convection it should be installed vertically in an area where there is at least a 100mm air gap above and below the package and 50mm either side. Avoid mounting heat producing equipment directly below a drive.

Precautions

During installation, take the normal precautions against damage caused by electrostatic discharges. Earthed wrist straps should always be worn.
All user I/O connections should be screened.

A switch or circuit breaker must be included in the installation, which must be clearly marked as the disconnecting device and should be within easy reach of the machine operator.

Drive Dimensions

PDFX drives are supplied as packaged units designed for direct panel mounting. Figure 3-1 shows the overall dimensions of the drive and the distance required between mounting holes, in mm.

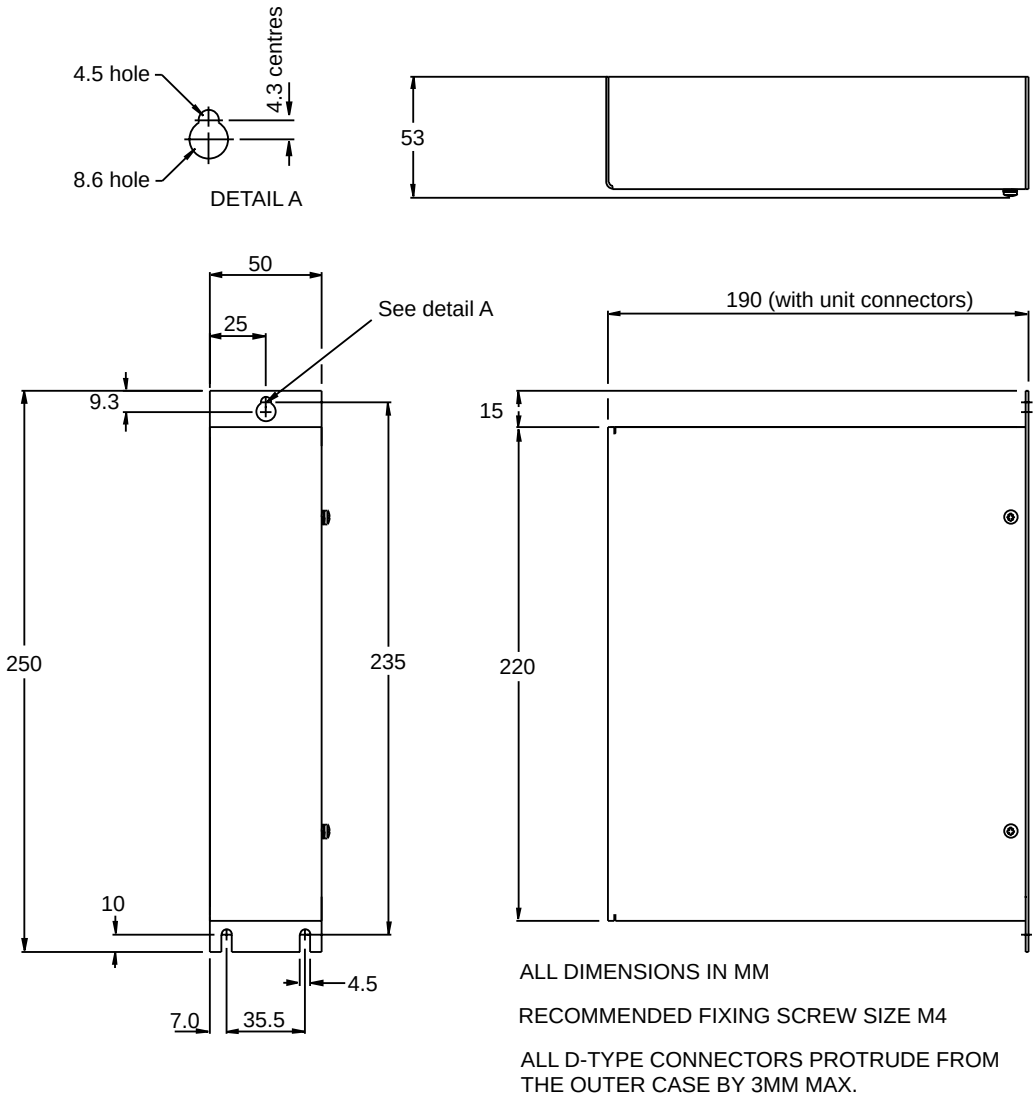


Figure 3-1. PDFX Mounting Dimensions

Connector Position and Layout

Figure 3-2 gives the identification and pin-out of all drive connectors.

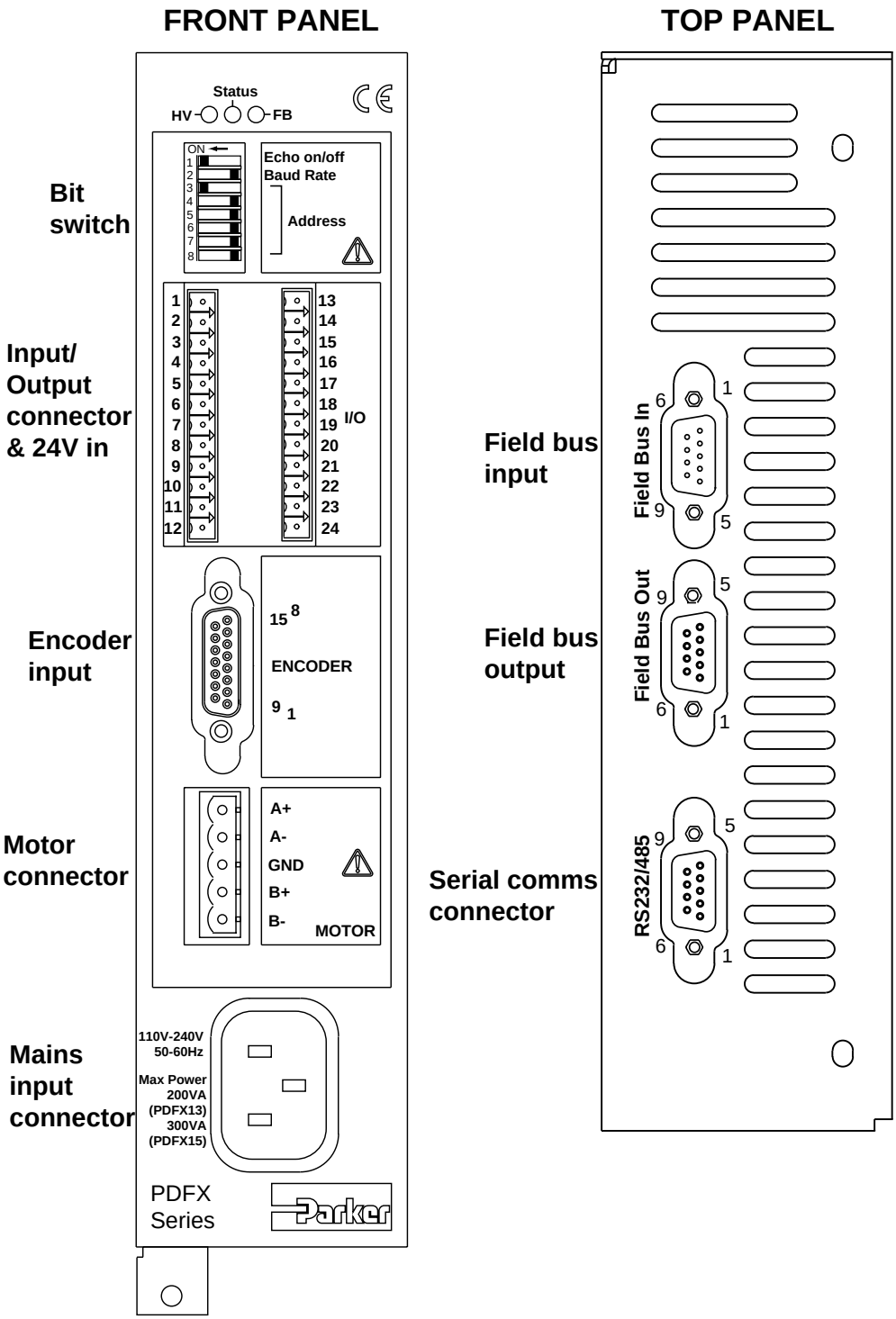


Figure 3-2. Bit Switch and Connector Layout

Cabinet Installation

To produce an EMC and LVD compliant installation drives must be mounted within a steel equipment cabinet. This form of enclosure also offers the benefits of operator protection and reduces the contamination of the equipment from industrial processes.

A steel equipment cabinet will screen radiated emissions provided all panels are bonded to a central earth point. Separate earth circuits are commonly used within equipment cabinets to minimise the interaction between independent circuits. A circuit switching large currents and sharing a common earth return with another low level signal circuit could conduct electrical noise into the low level circuit, thereby possibly interfering with its operation. For this reason so called 'dirty earth' and 'clean earth' circuits may be formed within the same cabinet, but all such circuits will eventually need to be returned to the cabinet's main star earth point.

Individual drives are designed to be mounted on a metal earth plane to which the EMC filter's earth connection is also attached. The earth plane will have its own individual star point earth which should be hard wired (using an insulated copper conductor) back to the cabinet's 'clean earth' connection point.

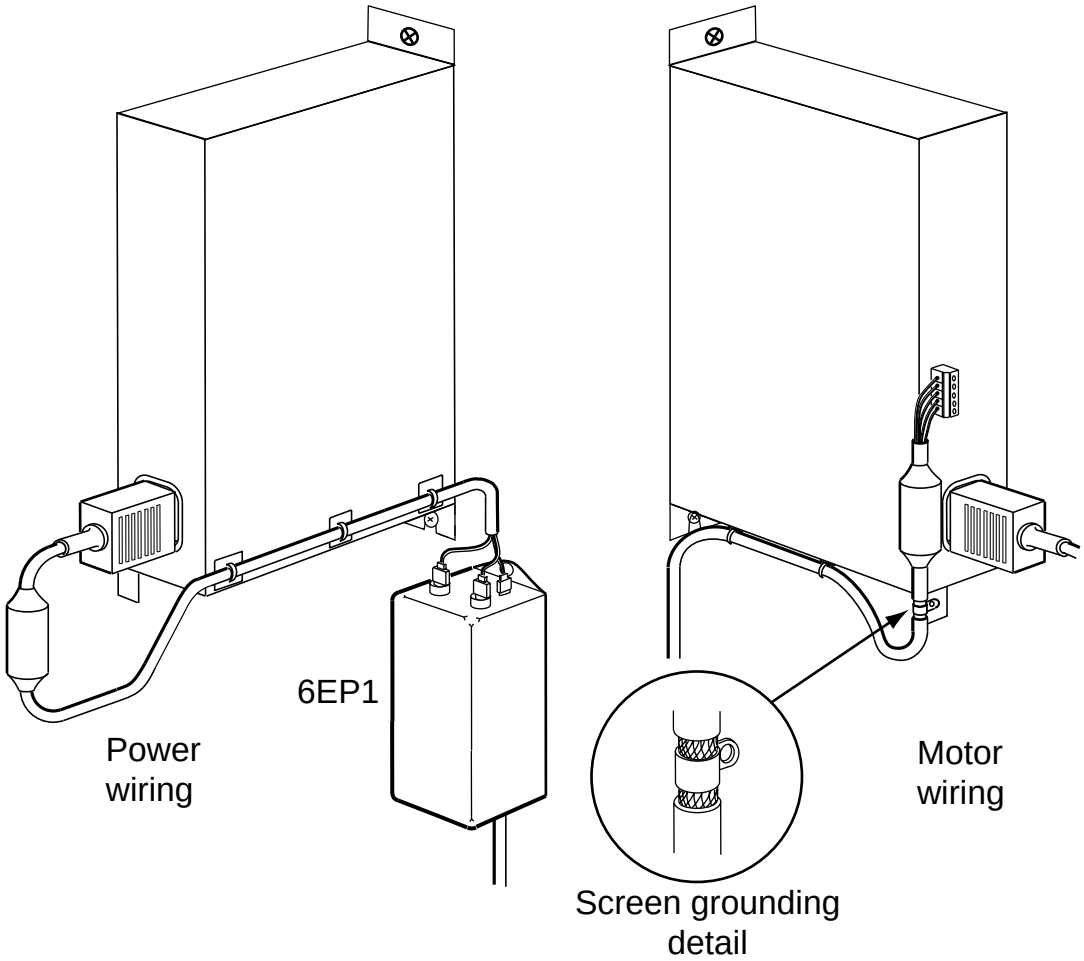


Figure 3-3. Cabinet Mounting

Power Connections

Power is supplied to the PDFX via its front panel IEC mains connector. Direct On Line (DOL) operation is possible from a single phase mains supply of 110 to 240V. Input current requirements of the drive variants are:

PDFX 13	2.5A peak
PDFX 15	5A peak

Use approved mains cable, with a minimum wire size of 0.75m^2.

Fuse Requirements

Each drive contains two input mains fuses, a motor fuse and a dump fuse, if fitted with an optional dump circuit. Both the mains input live and neutral wires are fused with identical value fuses. Fuse values are:

PDFX 13 mains input (FS2 & 3)	3.15A TL HB (5 x 20mm)
PDFX 15 mains input (FS2 & 3)	3.15A TL HB (5 x 20mm)
Motor HV fuse (FS1)	5A QA HB (5 x 20mm)
Dump fuse (if fitted)	500mA TL LB (5 x 20mm)

External +24V DC Supply

The PDFX can be used without an external +24V power supply, but does require one if I/O circuits are being used or where you wish to preserve indexer data in the event of loss of mains power to the drive. The indexer power facility is useful during the commissioning of cabinet mounted drives where AC supplies are automatically disconnected whenever a cabinet door is opened.

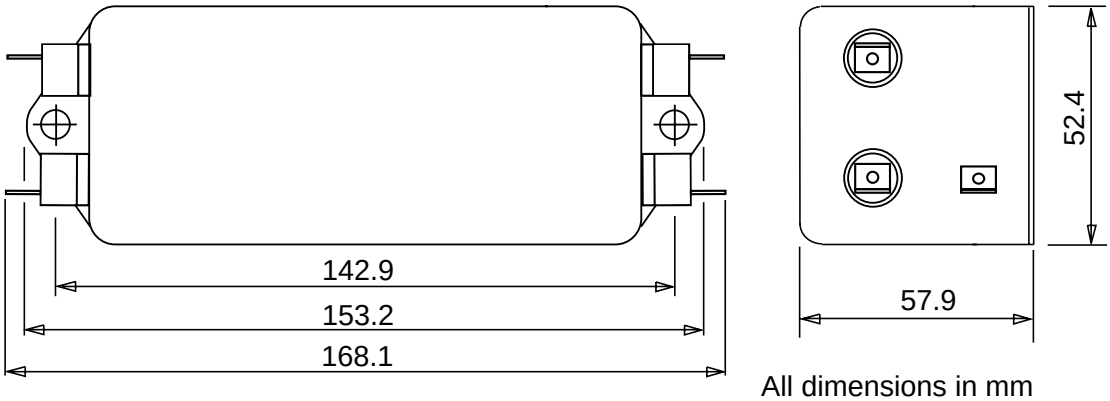
If you have a simple single drive application and do not wish to provide a +24V rail, but do require 24V for a particular sensor, consider using a drive output permanently held ‘high’ to act as a current limited source of +24V.

EMC Considerations

For EMC compliant installation a drive must be installed within a cabinet on a conductive back panel and have a suitable EMC filter fitted to the AC supply input. A suitable filter is the CORCOM 6EP1 or its equivalent. The filter must be fitted close to the drive and share the same common cabinet earth plane. Both the drive and filter must make good electrical contact with the earth plane. On the drive, the mounting side of the bottom bracket is left unpainted to allow a direct electrical contact to be made.

Care should also be taken to avoid cross coupling of signals between filtered and non filtered supply wiring. For example, do not run filtered and non filtered supply cables alongside one another, keep wiring direct and avoid wiring loops when distributing supplies between multiple units.

Filter Dimensions



All dimensions in mm
mounting holes 4.8mm diameter
terminals 6.35mm Fastons

Figure 3-4. 6EP1 EMC Filter Dimensions

Motor Selection

Usually optimum performance will be obtained when the current rating of the motor is between 1 and 1.5 times the drive rating, but remember the current rating of a motor is its DC rating, whilst drives are rated at peak output. To match a particular motor's current rating, multiply the drive's peak output current by 0.7. Drives can be derated to accommodate motors with lower current ratings (using system variable MC), however the high speed torque will be reduced.

Do not use a drive setting which gives an output current greater than the motor rating.

With 4 lead motors the bipolar rating is quoted and this should match the criteria stated above.

With 8 lead motors the bipolar rating of the motor, which is normally quoted, refers to a parallel winding connection. With the windings connected in series the current rating of the motor connection will be 50% that of the bipolar rating, and the motor will give improved low-speed torque, but reduced high-speed torque.

The PDFX 13 & PDFX 15 will drive motors having an inductance as low as 0.5mH, but the recommended motor inductance range is between 0.8mH and 10mH.

The performance of the PDFX 13 & PDFX 15 is optimised for the following motor types, listed in Table 3-1.

Motor Type	Motor Rated Current in Amps*	Motor Inductance in mH	PDFX 15	PDFX 13
STT57-51P	3.0	2.0	✔	✔
STT57-83P	6.6	0.8	✔	
STT57-102P	6.5	1.0	✔	
STT83-62P	4.5	1.5	✔	
STT83-93P	7.0	2.0	✔	
STT83-135P	10.0	1.6	✔	
STT57-51M	1.4	9.5		✔
STT57-83M	2.7	5.7		✔
STT57-102M	4.1	2.5	✔	
STT83-62M	2.4	5.2		✔
STT83-93M	5.6	3.2	✔	
STT83-135M	5.4	5.6	✔	

*(parallel connection)

Table 3-1. Optimum Motor Types

Motor Voltage Ratings

Motors with a withstand voltage rating from phase to earth of 1000V AC should be used. An insulated withstand rating of 500V AC is acceptable if an isolating transformer with earth screen is used to power the system, and the secondary is earthed, as specified.

Large Motors

The largest recommended motor size is a 34-frame 3-stack. Please contact Parker EMD if you wish to use a larger frame motor.

Motor Cables

The recommended wire size for PDFX motor cables, of length less than 20m, is 1mm^2. For motor cable lengths greater than 20m (up to a maximum of 50m) use a wire size of 2.5mm^2. Use a cable containing five conductors plus the braided screen (such as Lapp 34805), the green wire being used to provide an earth return to the drive. Termination at the motor must be made using a 360° bond to the motor body, and this may be achieved by using a suitable clamp. Many stepper motors are designed to accommodate an appropriate terminal gland which can be used for this purpose.

Motor Connections

In the majority of applications the drive will be used with an eight lead motor with the windings connected in parallel or series, as shown in Figure 3-5. Motor connections will need to be determined from the motors data sheet. These are normally identified by wire colour or terminal markings, depending upon the make of the motor.

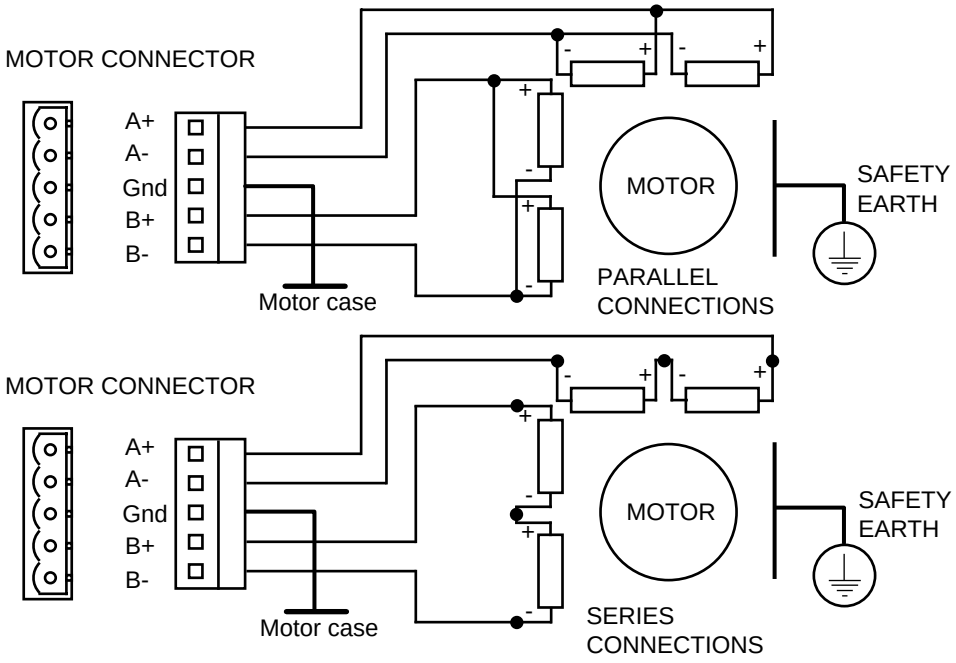


Figure 3-5. 8 Lead Motor Connection Options

Motor Safety Earth/Ground Connection

The case of an installed motor must be independently bonded to a local safety earth point. The safety earth lead should be at least 2.5mm^2 in area.

WARNING - High Temperature

The motor case temperature may exceed 70°C and should be guarded from operator contact.

Motor Default Settings

Motor current is set by system variable MC, the default value being 50% of peak drive current. The resolution is set by MR and has a default value of 4000 steps/rev, whilst the standby current is set by MS and has a default setting of 50% of peak drive current. All default values are listed under the RFS command.

Motor Safety Earth/Ground Connection

The case of an installed motor must be independently bonded to a local safety earth point. The safety earth lead should be at least 2.5mm^2 in area.

Motor Wiring Identification Tables

The following tables supply serial and parallel connection information for a range of different motor types.

MAKE	TYPE	A+	A-	B-	B+	NOTES
Parker EMD STT/STL	8-lead	Red	Yel	Pink	Blk	Link Blue & Violet, link White & Grey
	T.Box	1	2	3	4	Link 5 & 6, link 7 & 8
Parker OEM Series	4-lead	-	-	-	-	Internally wired in parallel
	8-lead	Red	Black	Green	White	Link Blue & Yellow Link Orange & Brown
Parker QM & S Motors	8-lead	Red	Black	White	Green	Link Yel & Blue Link Org & Brown
Parker QM & S Motors (except 106-205)	T.Box	1	3	4	5	Link 2 & 6 Link 7 & 8
Parker QM & 106-205 RS & TS	T.Box	1	3	7	8	Link 5 & 6 Link 2 & 4
Pacific Scientific	6-lead	Black	Orange	Red	Yellow	White/Blk/Org,
	8-lead	Black	Orange	Red	Yellow	Link Wh/Blk & Wh/Org Link Wh/Red & Wh/Yel
	T.Box	1	3	2	4	Link 5 & 6, Link 7 & 8
Astrosyn, Rapidsyn, Slo-syn	6-lead	Red	Red/Wh	Grn	Grn/Wh	White & Black N/C
	T.Box (X6)	1	3	4	5	2 & 6 N/C
Slo-syn	8-lead	Red	Red/Wh	Grn	Grn/Wh	Link Black & White Link Org & Blk/Wh
	T.Box (X8)	1	3	5	4	Link 2 & 6, Link 7 & 8
Stebon	8-lead	Red	Yel	Pink	Blk	Link Blue & Violet, link White & Grey
	T.Box	1	2	3	4	Link 5 & 6, link 7 & 8
M.A.E	6-lead	Grn/Wh	Grn	Red	Red/Wh	White & Black N/C
	8-lead	Black	Orange	Red	Yellow	Link Wh/Blk & Wh/Org Link Wh/Red & Wh/Yel
	T.Box	6	5	8	7	Link 1 & 3, Link 2 & 4
Zebotronics	T.Box	1	4	5	8	Link 2 & 3, Link 6 & 7
Oriental	6-lead	Black	Green	Red	Blue	Yellow & White N/C
Sonceboz	8-lead	Green	Grn/Wh	Red	Red/Wh	Link Org & Blk/Wh Link Black & White
Japan Servo	6-lead	Red	Blue	Green	Yellow	2 X White N/C
Escap	8-lead	Brown	Org/Wh	Red	Yel/Wh	Link Brn/Wh & Org Link Red/Wh & Yellow
Bodine	8-lead	Brown	Orange	Yellow	Red	Link Wh/Brn & Wh/Org Link Wh/Yel & Wh/Red
	T.Box	1	3	4	2	Link 5 & 7, Link 6 & 8

Table 3-2. Motor Connection Data - Windings in Series

MAKE	TYPE	A+	A-	B-	B+	NOTES
Parker EMD STT/STL	8-lead T.Box	Red & Blue 1 & 6	Yel & Violet 2 & 5	Wh & Pink 3 & 8	Blk & Grey 4 & 7	
Parker OEM Series	4-lead 8-lead	Red Red & Blue	Black Black & Yellow	Green Green & Org	White White & Brown	
Parker QM & S Motors	8-lead	Red & Blue	Black & Yellow	White & Brn	Green & Orange	
Parker QM & S Motors (except 106-205)	T.Box	1 & 2	3 & 6	4 & 7	5 & 8	
Parker QM & 106-205 RS & TS	T.Box	1 & 5	3 & 6	2 & 7	4 & 8	
Pacific Scientific	6-lead 8-lead T.Box	Black Black & Wh/Or 1 & 5	Wh/Blk/ Orange Orange & Wh/Blk 3 & 6	Red Red/ Wh/Yel 2 & 7	Wh/Red/ Yellow Yellow & Wh/Red 4 & 8	Org & Yellow N/C
Astrosyn, Rapidsyn, Slo-syn	6-lead T.Box (X6)	Red 1	Black 6	Green 4	White 2	Red/Wh & Grn/Wh N/C 3 & 5 N/C
Slo-syn	8-lead T.Box (X8)	Red & White 1 & 2	Blk & Red/Wh 3 & 6	Grn & Blk/Wh 4 & 7	Org & Grn/Wh 5 & 8	
Stebon	8-lead T.Box	Red & Blue 1 & 6	Yel & Violet 2 & 5	Wh & Pink 3 & 8	Blk & Grey 4 & 7	
M.A.E	6-lead 8-lead T.Box	Grn/Wh Black & Wh/Or 3 & 6	White Or & Wh/Blk 1 & 5	Red Red & Wh/Yel 4 & 8	Black Yel & Wh/Red 2 & 7	Grn & Red N/C
Zebotronics	T.Box	1 & 2	3 & 4	5 & 6	7 & 8	
Oriental	6-lead	Black	Yellow	Red	White	Grn & Blue N/C
Sonceboz	8-lead	Grn & Blk/Wh	Or & Grn/Wh	Red & White	Blk & Red/Wh	
Japan Servo	6-lead	Red	White*	Green	White*	
Escap	8-lead	Brn & Orange	Brn/Wh & Org/Wh	Red & Yellow	Red/Wh & Yel/Wh	
Bodine	8-lead T.Box	Brn & Wh/Or 1 & 7	Wh/Brn & Or 3 & 5	Yel & Wh/Red 4 & 6	Wh/Yel & Red 2 & 8	

*use correct white for each phase
For 6-lead motors, connections shown are for one half-winding.
N/C means No Connection

Table 3-3. Motor Connection Data - Windings in Parallel

User I/O Pin Description

The front panel 24-way Phoenix connector is used for the majority of the user I/O connections. This connector also provides the external +24V supply connection point. Two mating screw terminal plugs are supplied with the drive to allow user connections to be made.

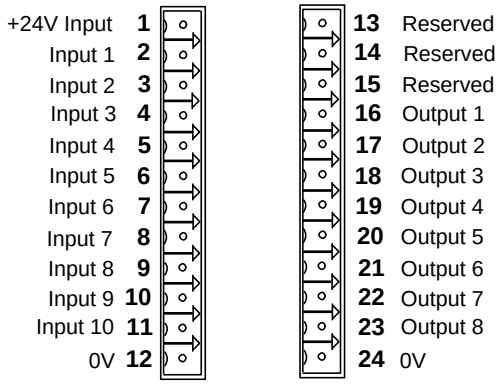


Figure 3-6. User I/O Pin Connections

Digital input and output circuit configurations are described in ***Section 5 Control of the PDFX***. User inputs 7 to 10 (inclusive) can be used as general purpose user inputs or have a dedicated function. These inputs become dedicated for a particular use when their associated command is turned ON. The functions associated with each input are:

Registration	Input 7
Home switch	Input 8
Negative limit	Input 9
Positive limit	Input 10

For example, assuming axis 3 has already been sent a configure homing command, homing could be turned ON by issuing a **3HOME1** command. In this case, axis 3 input 8 becomes dedicated for use with the home switch on axis 3 and cannot be used for any other purpose until the homing command is turned OFF and the input is re-defined (very unlikely if homing is being used). Inputs 9 and 10 automatically default to limit inputs (input 9 becomes LIM- and input 10 becomes LIM+), but they can be re-configured as general purpose user inputs by disabling both positive and negative limits in the LIMITS command.

Primary Encoder Input Connector

The front panel 15-way D-type connector allows an external feedback encoder to be fitted to the drive. An external encoder must be used if position maintenance or stall detection facilities are required by the application.

High speed differential inputs are used for each encoder channel which require screened wiring. The cable screen needs to be connected to the connector shells at either end of the cable, assuming a connector is used at the encoder end.

Pins 1(=GND) and 8(=5V) provide a source of power for the external encoder (5V at 150mA).

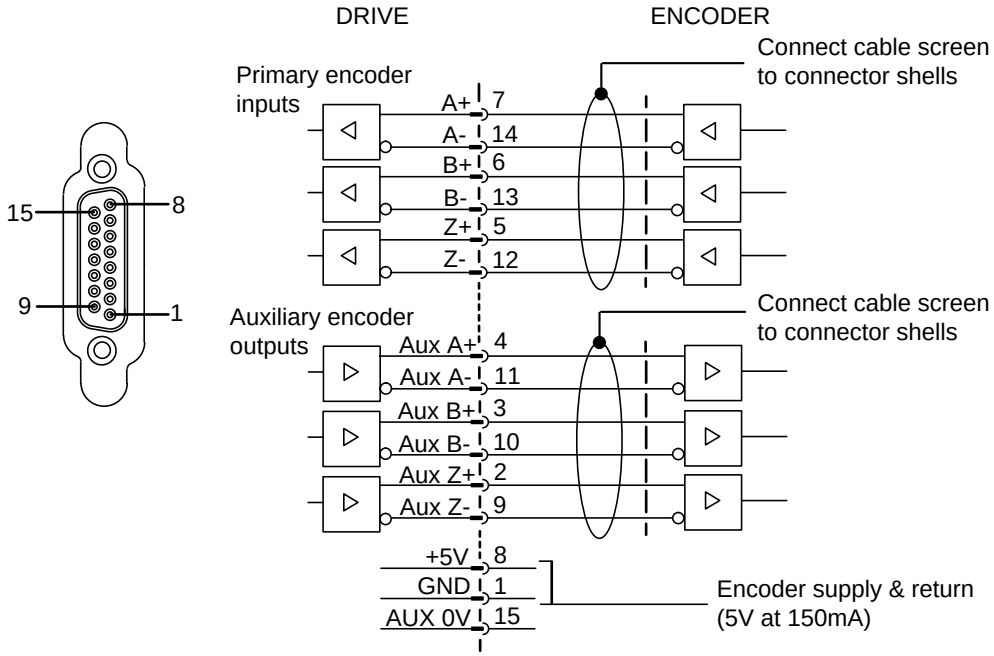


Figure 3-7. Encoder Connection Details

Encoder Outputs

Encoder outputs can be used to supply a step-direction or step-up/step-down signal for use by another drive. The type of output is controlled by system parameter EO as defined in Table 3-4.

Encoder signal	EO=0	EO=1	EO=2
AUX A+	STEP+	CW+	A+
AUX A-	STEP-	CW-	A-
AUX B+	DIR+	CCW+	B+
AUX B-	DIR-	CCW-	B-
AUX Z+			Z+
AUX Z-			Z-

Table 3-4. Encoder Output Configuration

Encoder Inputs

Encoder inputs are controlled by system parameter EI as defined in Table 3-5.

Encoder signal	EI=0	EI=1	EI=2
A+	STEP+	CW+	A+
A-	STEP-	CW-	A-
B+	DIR+	CCW+	B+
B-	DIR-	CCW-	B-
Z+			Z+
Z-			Z-

Table 3-5. Encoder Input Configuration

Serial Communications Connector

Mounted on top of the drive, the 9-way D-type female socket serial communications connector provides both RS232 and RS485 serial options. The pin out and signal names are shown in Figure 3-8. For a complete description of serial port connection options see ***Section 5 Control of the PDFX***.

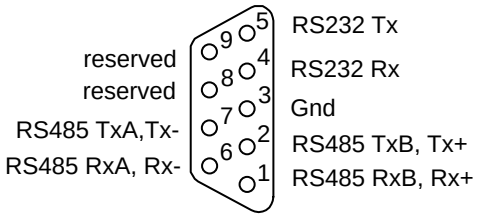


Figure 3-8. RS232/485 Connector

All serial communication connections should be made using screened twisted pairs. The cable screen being returned to the connector shell at both ends of the cable.

RS232/485 Connector

Pin	RS232/RS485
1	RS485 RXB, Rx+
2	RS485 TxB, Tx+
3	RS232 GND
4	RS232 Rx
5	RS232 Tx
6	RS485 RxA, Rx-
7	RS485 TxA, Tx-
8	Reserved
9	Reserved

Table 3-6. RS232/RS485 Connections

RS232/485 Address Selection

Drive hardware address selection is made using the front panel bit switch. The switch, shown in Figure 3-9, has six switch positions dedicated to address selection (bits 3 to 8). An address of '1' is selected by setting bit switch 3 ON and 4 to 8 OFF. By setting patterns of bit switch positions any hardware address from 1 to 63 can be selected, as shown in Table 3-7. Note: An address can be set remotely - see the # command.

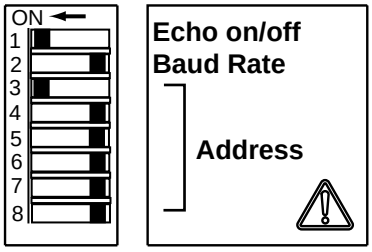


Figure 3-9. Address Selection Bit Switch

Address	Bit Switch Number					
	3	4	5	6	7	8
01	ON	OFF	OFF	OFF	OFF	OFF
02	OFF	ON	OFF	OFF	OFF	OFF
03	ON	ON	OFF	OFF	OFF	OFF
04	OFF	OFF	ON	OFF	OFF	OFF
05	ON	OFF	ON	OFF	OFF	OFF
06	OFF	ON	ON	OFF	OFF	OFF
07	ON	ON	ON	OFF	OFF	OFF
08	OFF	OFF	OFF	ON	OFF	OFF
09	ON	OFF	OFF	ON	OFF	OFF

10	OFF	ON	OFF	ON	OFF	OFF
11	ON	ON	OFF	ON	OFF	OFF
12	OFF	OFF	ON	ON	OFF	OFF
13	ON	OFF	ON	ON	OFF	OFF
14	OFF	ON	ON	ON	OFF	OFF
15	ON	ON	ON	ON	OFF	OFF
16	OFF	OFF	OFF	OFF	ON	OFF

Following a binary pattern to address 63.

61	ON	OFF	ON	ON	ON	ON
62	OFF	ON	ON	ON	ON	ON
63	ON	ON	ON	ON	ON	ON

Table 3-7. Hardware Address Selection Bit Switch Settings

Echo on/off

Selecting echo (bit switch in the ON position) echoes back characters received over the serial link. You will need to set echo ON for RS232 and echo OFF for RS485.

Baud Rate

The baud rate bit switch selects the serial communications speed. In the ON position 19200 baud rate is selected and in the OFF position 9600 baud rate is selected. Both rates can be used for RS485 and RS232 communications.

Software Address Selection

An RS232 address above 63 can be set by software, using the # command. The address following the # command is saved in the drive's non volatile memory following an SV command. To use this saved address upon subsequent resets of the drive the bit selection switches 3 to 8 need to be set all OFF. To return the drive to hardware (bit switch addressing) execute the **<a>#0** command for that particular drive or **0#0** command to set all addresses of all drives by hardware.

Software addressing is useful where a multi-axis system needs to be set-up, especially where the number of axes are greater than the highest hardware address. By issuing a #1 command the first indexer will send a #n+1 command along the daisy chain (assuming echo mode is set) resulting in each drive being given an address which increments around the daisy chain. Consequently, the first drive will be assigned address 1, the second address 2 and so on, until the final drive will echo back (final drive number +1). In this manner drives will be assigned addresses that increment in numeric value around the RS232 daisy chain connection. Software addressing is only accepted if the echo bit switch is set to ON.

To set addresses starting from a particular address, use the command #n, where n is the address you wish to start from. For example, to increment the daisy chain addresses from 6 onwards, use the command #6. If you wish to give a particular axis a new address, you can use the command <a>#n, where <a> represents the old address and 'n' is the new address.

When addresses are incremented using n+1, any previous, non-sequential, address pattern will be re-ordered as a sequential pattern. For example, if a system is composed of drives with addresses 3, 5, 7 (in that order) is sent the command #20, the addresses will be changed to 20, 21, 22. The maximum address that can be sent using incremental addressing is [255 - the number of axes being used].

Software address selection can only be used with RS232, not RS485.

Field Bus Interface (9-way D-type)

In addition to the RS232/485 serial port provided by the main controller, an optional Fieldbus module can be fitted within the PDFX to make available CANbus. :

Access to a bus module is made via the connectors on the top of the drive. The use of two connectors allows drives to be linked in a daisy chain manner, where multi-axis systems are being used. The address used by the bus interface is taken from the axis to which it is attached. Note: If an axis address is changed the power will need to be recycled for the bus interface card to read the new address.

Input Field Bus Connections

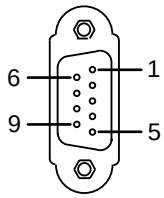


Figure 3-10. Field Bus Input Connector

Pin	CAN Bus
1	CAN Power
2	CAN-L
3	GND2
4	N.C.
5	N.C.
6	CANGND
7	CAN-H
8	N.C.
9	N.C.

Table 3-8. Fieldbus Inputs

Output Field Bus Connections

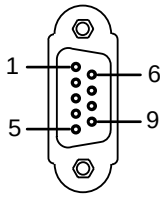


Figure 3-11. Field Bus Output Connector

Pin	CAN Bus
1	CAN Power
2	CAN-L
3	GND
4	N.C.
5	+5V
6	GND2
7	CAN-H
8	N.C.
9	N.C.

Table 3-9. Fieldbus Outputs

Section 4. Hardware Reference

Drive Specification

Functional Specification

Parameter	Value
Amplifier	
Type	MOSFET chopper
Motor resolution	400, 800, 2000, 4000 steps/rev
Maximum stepping rate	200kHz at 4000 steps/rev
Nominal chopping frequency	20kHz
Protection Short circuit Motor overcurrent Over & under voltage Logic supply fault Over temperature External 24V reverse supply	 Phase-to-phase, across phases and phase to ground
Peak output current	2.5A peak +/- 10% (PDFX 13) 5.0A peak +/- 10% (PDFX 15)
Output current adjustment	50% to 100% of peak current, software selectable in 10% increments by system variable MC
Standby current reduction	50% or 70% selected by system variable MS
Standby reduction time	30mS from last step pulse
Motor bus voltage	75V DC +/-10%
Optional external logic supply input	24V DC +/-10%
External logic supply current	500mA plus total current sourced by programmable outputs
Optional power dump (PDFX15 only)	17W continuous rating, 170W peak
Motor inductance range	0.8 to 10mH

Table 4-1. Functional Specification

Indexer Specification

Parameter	Value
Command Interface	
Indexer	
Position range	-2,147,483,647 to +2,147,483,647 steps
Velocity range	0.01 to 50.00 revs/s
Acceleration range	0.1 to 1024.0 rev/s/s
Deceleration range	0.1 to 1024.0 rev/s/s
Positioning modes	Incremental, absolute, registration, continuous run, bidirectional voltage, unidirectional voltage
Communication	
Data format	8 data bits, 1 start bit, 1 stop bit, no parity, optional echoback, Xon/Xoff supported
Baud rate	9600 or 19200 selectable by bit switch
Address setting range	1 - 63 by bit switch, 1 -255 by software
RS232 connection	2 wire plus ground
RS485 connection	2 wire or 4 wire
Analogue control inputs	
Voltage range	+/-10V differential
Maximum input voltage	+/-20V, either input relative to 0V
Input impedance	200K
Resolution	10 bits (20mV)
Digital Inputs	
User programmable inputs	6 (home & limits can be used as additional user-programmable inputs but with limited functionality)
Dedicated inputs	Home, + limit, - limit, registration
Input levels	Logic high 12 - 30V, logic low 0 - 3.4V
Input impedance	2.3K to 0V
Encoder Inputs	
Signal levels	Differential, TTL compatible; quadrature, step/direction or CW/CCW step (1µS min. pulse width)
Maximum frequency	100kHz pre-quadrature
Encoder power output	+5V DC, 150mA maximum

Table 4-2. Indexer Specification

Parameter	Value
Digital Outputs	
User-programmable outputs	8, PNP current sourcing
Output levels	0 - 0.4V (output OFF with 2k load) 22 - 24V ±10% (output ON)
Output current rating	60mA maximum per output (provided by internal +24V supply)
Indexer clock output	Differential TTL levels, configurable as step/direction, CW/CCW step or quadrature

Table 4-2. Indexer Specification (Continued)

Power Specification

Parameter	Value
AC Power Input	
Drive supply voltage	105V to 240V RMS ±10%
Supply frequency range	47 to 63Hz
Internal power supply rating	PDFX13 - 100W continuous, 200W peak (25% duty cycle) PDFX15 - 150W continuous, 300W peak (25% duty cycle). See Figure 4-2.
Input current	
PDFX 13	2A max. at 110V AC
PDFX 15	3A max. at 110V AC
Recommended supply protection	
PDFX 13 mains input (FS2 & 3)	3.15A TL HB (5 x 20mm)
PDFX 15 mains input (FS2 & 3)	3.15A TL HB (5 x 20mm)
Motor HV fuse (FS1)	5A QA HB (5 x 20mm)
Dump fuse (if fitted)	500mA TL LB (5 x 20mm)

Table 4-3. Power Specification

Environment Specification

Parameters	All drive types
Environment	Pollution degree 2 Installation category II
Operating temperature range	0 - 50°C ambient
Humidity	0 - 95% non-condensing
Cooling	Natural convection with integral fan
Housing	Closed steel
Protection class	IP20
Weight	4.2Kg
Dimensions	See Figure 3-1

Note: If the drive is to be used in an overvoltage category III environment, external varistors must be fitted from live to neutral and neutral to earth, rated to absorb at least 100J.

Table 4-4. Environment Specification

Torque-Speed Curves

The following torque-speed curves indicate the performance achieved using Parker ST series stepper motors. Alternative makes of motor may give similar performance for equivalent frame sizes, but this can only be confirmed by measurement.

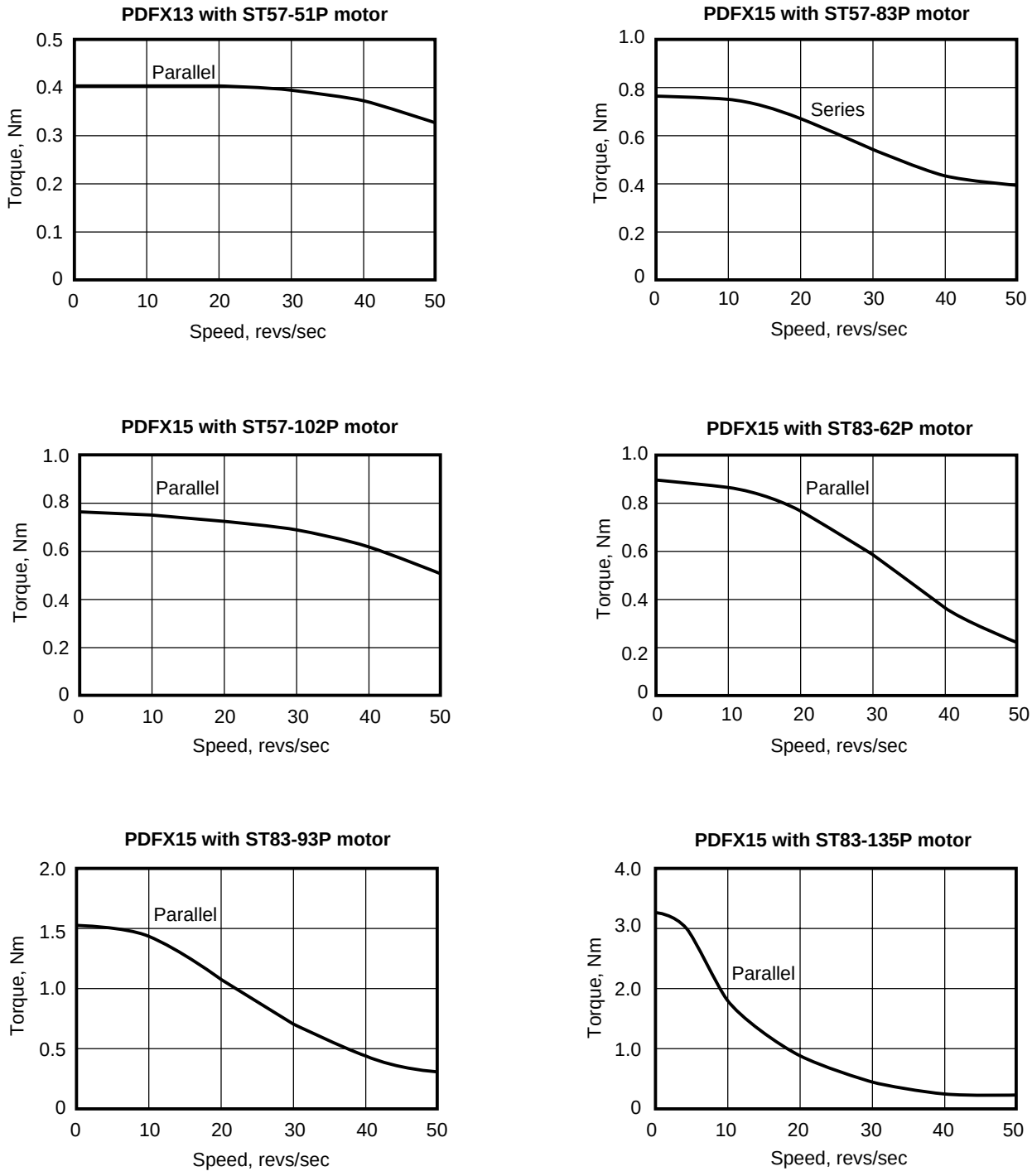


Figure 4-1. Torque Speed Curves

PDFX15 Power Derating Curve

The PDFX15 drive will operate from a supply voltage as low as 95V AC, but will be subject to a de-rating of continuous power as defined by the graph given in Figure 4-2. If your application is being operated at a supply voltage below 120V AC and you experience drive overtemperature faults you may need to decrease the repetition rate of your application to give the drive a greater cool down time between moves.

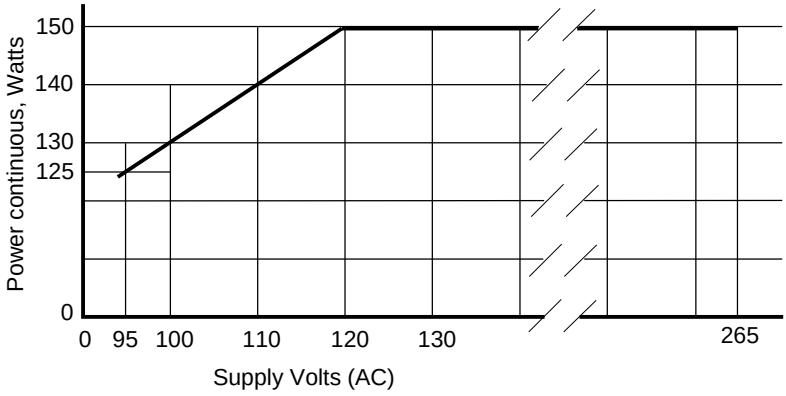


Figure 4-2. PDFX15 De-rating Curve

Section 5. Control of the PDFX

Overview

This section introduces you to the operation of the PDFX indexer, motion control concepts and the way certain software commands are used. Event driven code techniques are explained together with the use of system variables. Fault reporting and the interrogation of system variables are also described within this section.
Features of the drive are introduced in parallel with an explanation of move types and motion profiles. Registration operation, homing concepts, user I/O configuration and position maintenance are also covered, together with stall detection and serial communications.

Controller Operation

The PDFX has an integrated indexer which can be controlled directly by a PC over a serial link, or programmed to respond to code selected by event triggers or user instructions.

Direct Mode

Direct control of the indexer over a serial link can be used for program development/downloading purposes or direct on-line control from an industrial PC or PLC. When used directly the indexer will accept commands prefixed with the drive's address and will action the commands as they are received. In direct mode any controlling application program is stored in a remote location and is only downloaded to the drive when required.

Direct Mode control can be used when first testing a drive:

```
1ON              ;enable the drive
1LIMITS(3,0,0)   ;disable limits
1D4000           ;set distance to 4000 steps
1G               ;execute the move
```

Direct Computer Control

The following information is only required when you need to control the drive directly, using a serial link connection from a PC or PLC.

WARNING - Commands still actioned following a system error

When used in direct mode it is possible for commands to still be actioned following a system error unless suitable programming precautions are taken as described below.

When operating in direct mode, that is supplying the drive with blocks of code over a serial link, there is a possibility of commands still being executed after a system error has occurred. This is caused by the command buffer being immediately cleared following a system error. Once cleared, serial data code blocks still arriving will continue to be executed. To prevent this happening complete blocks of code need to be stored in the drive's buffer before being executed. This will ensure that any system error will clear the

buffer and delete all commands awaiting execution. This can be achieved by prefixing each command block with a delay that allows the complete block to be loaded into the drive's buffer before the code is executed. Typically, 50ms is sufficient to cover most situations. Using a baud rate of 9600 each character takes 1ms to download.

For the purpose of this description a system error is defined as any situation that causes the command buffer to be cleared, typically, hitting a limit or the ocurrence of a drive fault.

For example, the following code, if sent directly, would cause output 1 to go high, then low one second later even though the drive had hit a limit immediately after the 1G command was executed.

```
1G              ;go command
```

drive hits a limit causing a system error

```
1O(1)           ;set output 1 high
1T1             ;delay 1 second
1O(0)           ;set output 1 low
```

To prevent this happening, the following delay could be added.

```
1T0.05          ;allows 50 characters to go into the buffer at 9600 baud
1G              ;go command
```

drive hits a limit, buffer cleared (output command deleted), waiting for next command block

To prevent the time overhead caused by the added delay, PAUSE and CONTINUE could be used, as shown below, although blocks of code would need to be limited to a size of less than 30 characters.

```
1PS             ;pause (only introduces 3ms delay)
1G              ;go command
1O(1)           ;set output 1 high
1T1             ;delay 1 second
1O(0)           ;set output 1 low
1C              ;continue (allows code to enter buffer before running)
```

An alternative method of control, not dependent upon any variable time delay, is to implement a form of handshake control where a small program is downloaded to the drive and is called following every block of code transmitted. The downloaded code could take the form shown below:

```
1GHSAK:         ;go with handshake
1G              ;go
1O(1)           ;set output 1 high
1T1             ;delay 1 second
1O(0)           ;set output 1 low
1END            ;terminate
```

When transmitting command blocks from the PC or PLC at the end of each block include the following line:

```
1GOTO(GHSAK);goto handshake code
```

By monitoring the drive's output the command block download can be confirmed as having taken place. Any error occurring would not allow MOVE1 to execute.

Using this 'code label' form of signalling, a fault routine could be downloaded to the drive, to indicate a system error has occurred. The fault routine could be used to pulse output 2, as shown below:

```
1FAULT:     ;define code label
1O(01)      ;set output 2 high
1T1         ;delay 1 second
1O(00)      ;set output 2 low
1END        ;terminate
```

Using this style of direct computer control, output 1 pulsing high would indicate confirmation of the received code block and output 2 pulsing high would indicate a system error.

Programmed Mode

This mode allows a program stored within the drive to control operations. The program can be written off-line on a PC and then downloaded to the PDFX via a serial link. The application program is stored within the drive and can be automatically invoked at power up.

Code Structure

Program code is written as a series of blocks. Each code block has a unique label at the beginning and is terminated with an END label. The use of labels allows the code to be structured in the form illustrated in Figure 5-1, which shows the block nature together with an example of code.

Declare

Every label used in a program, apart from START, REG, NOREG and FAULT, needs to be declared before it can be used. If a label is declared, but not defined, a runtime error will be signalled when it is called.
Note: START, REG, NOREG and FAULT are all reserved labels.

Labels can only be declared in the command line at the start of a program or within the START code. The choice is between memory efficiency and the retention of declared labels during up-loading/down-loading of programs. By declaring labels in the command line, before any START code, makes the most efficient use of the available memory, but if you then upload the program to a PC and later down load the same program the declarations will have been lost. To retain declared labels you must declare them in the START code, this allows a program to be up-loaded and down-loaded without loss of declared labels, although more memory will be used.

Example of DECLARE being used in the command line:

```
1K                  ;Kill or stop any program currently running
1CLEAR(ALL)         ;Erase all programs
1DECLARE(MAIN)      ;Declare labels
1DECLARE(MOVE1)
1DECLARE(MOVE2)
    .
    .
```

Example of DECLARE being used following the START label:

```
1K                      ;Kill or stop any program currently running
1CLEAR(ALL)             ;Erase all programs
1START:
    1DECLARE(MAIN)      ;Declare labels
    1DECLARE(MOVE1)
    1DECLARE(MOVE2)
        .
1END
```

Labels

Labels consist of up to 5 upper case alphanumeric characters terminated with a colon (:). You should choose a name that is relevant to the operation being performed, or a system label name.
To terminate a code block use ‘aEND’ (no colon).
Upto 14 labels labels can be used, although four of these have already been allocated to START, REG, NOREG and FAULT, leaving ten for general use.

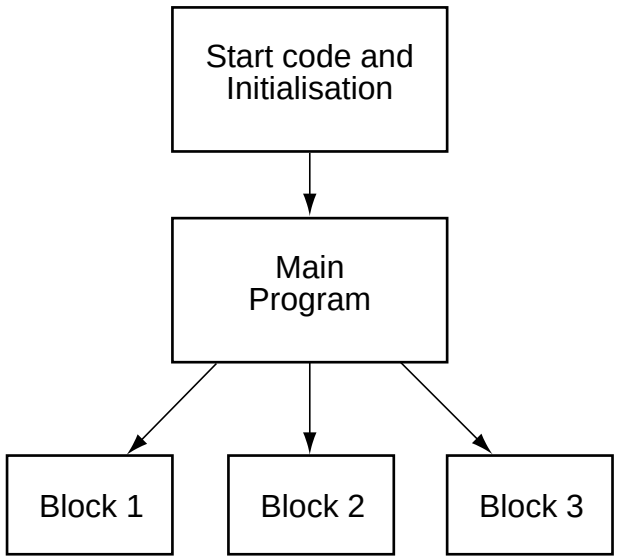
Label Execution

By using the label select command (LSEL), label code blocks can be triggered by a digital pattern appearing on certain user inputs. The command defines the user inputs to be used, the style of code detected (BCD or binary) and the manner in which the code is executed (continuous or re-trigger).

The LSEL command must be armed before input selection of labels will operate.

Structure

The code example of an absolute positioning move shown in Figure 5-1 demonstrates how code may be written that follows the block structure. Start code is first used to initialise the drive:




```
1START:
1DECLARE(MAIN)
1DECLARE(MOVE2)
1LIMITS(3,0,0)
1OFF
1W(MC,50)
1W(MR,4000)
1ON
1GOTO(MAIN)
1END
1MAIN:
1PROFILE2(40,10,-48000,25,2)
1GOSUB(MOVE2)
1END
1MOVE2:
1W(PA,0)
1MA
1USE(2)
1G
1END
```

Figure 5-1. Program Structure

Example:

```
1START:                 ; start label definition
1DECLARE(MAIN)          ; declare labels
1DECLARE(MOVE2)
1LIMITS(3,0,0)          ; configure limits.
1OFF                    ; remove power before changing motor parameters
1W(MC,50)               ; set motor current to 50%.
1W(MR,4000)             ; set motor resolution 4000 steps rev.
1ON
1GOTO(MAIN)             ; goto main program
1END
```

The MAIN part of the program is used to define profiles and to control the order of moves:

```
1MAIN:                          ; main label definition
1PROFILE2(40,10,-48000,25,2)    ; define move parameters
1GOSUB(MOVE2)                   ; jump to label move 2
1END                            ; end of label definition
```

Finally individual moves are called from the main part of the program:

```
1MOVE2:                         ; define program label  move
1W(PA,0)                        ; reset position absolute to zero
1MA                             ; absolute positioning move
1USE(2)                         ; use motion profile 2
1G                              ; execute move
1END                            ; end of program move 2 definition
```

Note: PROFILE2 defined in the main part of the program has the following characteristics:

ACCELERATION 40rps^2 , DECELERATION 10rps^2, DISTANCE 48000 steps at 4000 steps/rev (12 REVS MOVE), NEGATIVE DIRECTION , VELOCITY 25 rps , 2 rps START / STOP SPEED.

In small programs the start code can be combined with the main part of the program. For the experienced X-code users the lower blocks of code, accessed via subroutines, are the equivalent of sequences.

A second example illustrates the code required for an incremental move. Here the START and MAIN code blocks have been combined within the START block:

```
1START:                         ; start label definition
1DECLARE(MOVE1)                 ; declare move1 label
1LIMITS(3,0,0)                  ; configure limits (disable, n/c).
1OFF                            ; remove power before changing motor parameters
1W(MC,50)                       ; set motor current to 50%.
1W(MR,4000)                     ; set motor resolution 4000 steps rev.
1ON
1PROFILE1(80,20,24000,20,0)     ; define move parameters
1GOTO(MOVE1)                    ; transfer to label move 1
1END                            ; end of label definition

1MOVE1:                         ; define program label.
1MI                             ; incremental positioning move
1USE(1)                         ; use motion profile 1
1G                              ; execute move
1END                            ; end of program move 1 definition.
```

Note: [1] **DEVICE ADDRESSING IS REQUIRED FOR EACH COMMAND**

[2] PROFILE1 has the following characteristics:

ACCELERATION 80rps^2 , DECELERATION 20rps^2, DISTANCE 24000 steps (6 REVS MOVE), POSITIVE DIRECTION , VELOCITY 20 rps , 0 START / STOP SPEED.

LOOP Command

The block structure of the code lends itself to performing repetitive operations, using the LOOP command. A particular labelled block of code can be called a specified number of times. Such code could be used for intermediate repetitive processes, such as marking the surface of each work piece at some point in the production cycle.

An example using the LOOP command is given below, again the START and MAIN code blocks have been combined within the START block:

```
1START:                          ; start label definition
1DECLARE(LOAD)                   ; declare label
1LIMITS(3,0,0)                   ; configure limits
1OFF                             ; remove power before changing motor parameters
1W(MC,50)                        ; set motor current to 50%.
1W(MR,4000)                      ; set motor resolution 4000 steps rev.
1ON
1PROFILE3(100,50,4000,35,1)      ; define move parameters
1MI                              ; set mode to incremental
1LOOP(LOAD,6)                    ; repeat the load unload 6 times
1END                             ; end of label definition

1LOAD:                           ; define program label load
1USE(3)                          ; use motion profile 3
1O(XXX0XXXX)                     ; ensure o/p 4 is off
1T1                              ; wait for 1 sec delay
1G                               ; execute move
1O(XXX1XXXX)                     ; turn on o/p 4
1T1                              ; wait for 1 sec delay
1END                             ; end of label definition
```

Reserved System Labels

Certain labels have been pre-defined and are recognised by the controller as containing code used for common operations. If event triggered code is enabled (ARM1), the code entered for these common operations will be automatically run when the event occurs. System labels have the following names:

START: specifies the power on code, run using the ARM1 command
FAULT: specifies the code that is to be run when a fault occurs
REG: specifies the code to be run when a registration mark is detected within the registration window
NOREG: specifies the code to be run when a registration mark is not detected within the registration window

Note: If necessary, these labels can be used for other purposes, but cannot be re-named.

Fault Label

The pre-declared label named FAULT is used to identify a block of code that is executed when a particular problem (fault) has been detected. The code following the FAULT label

needs to change the state of an output, to indicate a fault has occurred and then go on to possibly diagnose the problem. Once the problem has been corrected, the FAULT code will need to detect an external 'reset', by monitoring a designated input and then execute an ON command to clear the FAULT. At the end of the FAULT code a GOTO(START) can be issued to restart the program. **This style of programming will always ensure that once a fault is detected the drive will stop and will not start again until commanded to do so.**

Before the code following a FAULT label can be executed certain conditions must be met, these are:

- FAULT must be defined
- ARM must be set to enable event triggered code
- ARM must be set to enable a FAULT label

This means FAULT label code must be present and the **ARM11** command exists at the beginning of the code.

The conditions under which the FAULT label is called will vary depending upon the fault itself and the condition of various other commands and command parameters. An exact description is presented in Table 4-1. However, in general, a FAULT label will be called given any one of the following conditions:

- An attempt to go home further onto a limit is made and the limit is enabled.
- An attempt to go further onto a limit is made with no fault label currently running, the limit decision is stop on limit and the limit is enabled.
- A limit is hit during motion and the move is not a go home, a fault label is not being run, the limit decision is stop on limit and the limit is enabled.
- The drive has stalled and the stall detect command STALL is enabled and set to stop on stall.
- A drive fault has occurred, but no drive programming is taking place.
- When it is called from a GOTO, GOSUB or LOOP command*.

*Note: in this case a FAULT has not actually occurred, consequently the FAULT label will be called irrespective of the state of the ARM command.

Table 5-1 summarises the conditions necessary for the FAULT label to be called. The FAULT label will **not** be called when any one of the following conditions occur:

- There is an error whilst sending a command
- There is a general run time error with the program
- The program memory area becomes full
- A label is attempted to be run when it does not exist
- The transmit buffer or receive buffer suffer an overflow

Fault Condition	**Command & parameter conditions**								
	FAULT label defined	Main ARM bit	Not GH	Fault ARM bit	Limit is enabled	Not running fault label	Limit decision is stop	Stop on stall set	Not programming the drive
GH onto a limit	Y	Y	N/A	Y	Y	N/A	N/A	N/A	N/A
G onto a limit	Y	Y	N/A	Y	Y	Y	Y	N/A	N/A
Hit limit	Y	Y	Y	Y	Y	Y	Y	N/A	N/A
Stalled	Y	Y	N/A	Y	N/A	N/A	N/A	Y	N/A
Drive fault	Y	Y	N/A	Y	N/A	N/A	N/A	N/A	Y
GOTO	Y	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Y
GOSUB	Y	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Y
LOOP	Y	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Y

Table 5-1. Conditions Required to Call a Fault Label

Example

The following example shows the use of a FAULT label within a program.

```
1ARM11                  ;enable auto-run on power-up & enable fault routine
1SV                     ;save the settings

1START:                 ;start of program
1ARM11                  ;re-enable auto-run & fault in case 'K' command sent
.
<initialisation commands>
.
1O(1XXXXXXX)            ;turn on output 1 - drive OK
.
<main process commands>
.
1END

1FAULT:                 ;fault routine
1O(0XXXXXXX)            ;turn off output 1 - drive fault
.
<diagnostic code - if required>*
.
1TR(IN,=,1XXXXXXX)      ;wait for input 1 to become active (RESET)
1ON                     ;clear fault
1GOTO(START)            ;run from start of program again
1END
```

*Note: An example of diagnostic code is given in the sub-section entitled ***Conditional Code*** later within this section.

Event Driven Code

Event driven code is code that is executed when a particular event occurs, such as a fault, a registration mark being detected or an input changing state. The event has the ability to effectively interrupt the drive's 'normal' operation and, under certain circumstances, can immediately jump to a defined label and start executing the code following that label. Its use allows a much faster reaction time to external signals than can be obtained using conventional polling techniques.

Event driven code is enabled or disabled using the ARM command. The command allows you to control the way the drive responds to events, although certain incidents such as limits, or home will always be actioned regardless of the state of the ARM command. Event driven code is enabled immediately ARM1 is set, consequently it could be used to jump out of a labelled code block.

The system label START: introduces the drive's setup and initialisation code. With ARM enabled the code is automatically executed at system start-up*. Consequently the code needs to be saved with ARM1 set. If you save a program with ARM0 set, the start-up code will not run and the indexer will only respond to serial input commands.

*Unless a drive fault is pending and a fault routine is defined and armed.

Input Events

You can let user inputs trigger events by the use of labels and associated user input numbers. By including the code you wish to action, following a pre-defined input label, will enable your code to be run when the defined user input is activated. For example, to select one of three labels using two user inputs, the code would be:

```
1START:
1CLEAR(ALL)       ;clear memory
1DECLARE(L1)      ;declare label 1
1DECLARE(L2)      ;declare label 2
1DECLARE(L3)      ;declare label 3
1LSEL1(0,2,1)     ;define inputs and code

1A20              ;set acceleration
1V5               ;set velocity
1O(00000000)      ;set all outputs low
1END

1L1:              ;label 1 code
1O(1)             ;set output 1 high
1D1000            ;set distance to 1000 steps
1G                ;move 1000 steps
1T1               ;wait for 1 second
1O(0)             ;set output 1 low
1END
```

```
1L2:          ;label 2 code
1O(01)        ;set output 2 high
1D-2000       ;set distance to -2000 steps
1G            ;move -2000 steps
1T1           ;wait for 1 second
1O(00)        ;set output 2 low
1END

1L3:          ;label 3 code
1O(001)       ;set output 3 high
1D3000        ;set distance to 3000 steps
1G            ;move 3000 steps
1T1           ;wait for 1 second
1O(000)       ;set output 3 low
1END
```

Note: Event triggered code execution occurs when the indexer detects an event for which a label exists.

The occurrence of an event trigger will cause the program code to immediately jump to the pre-defined input label and perform the code following that label. When the END label of this code is reached the program flow will not return to the code it was executing just before the event took place, but will continue to scan the inputs for another trigger event.

When using event triggers you must be aware that altering any basic operating parameters, such as velocity, in the event code will change the value being used in subsequent events. Consequently, you will need to fully define the move required in each event code block. This is readily accomplished by the use of USE commands to select a particular default value.

System Variables

System variables are named variables held within the PDFX, that are used for storing a variety of system values and settings, for example system variable EP holds the value of the motor encoder position in steps. All system variables can be read using the Report System Parameter (R command), but only certain variables can be written to using the Write (W command).

System variable values may be tested using the IF command. This allows conditional branching within the program code, enabling equal to, not equal to, greater than or less than decisions to be made. Wait for trigger (TR command) can also test system variables by delaying code execution until the value of a system variable matches some stored number or string within the program.

Table of System Variables

Table 5-2 lists system variables in alphabetic order together with their read/write status and range of values stored.

Var	Name	R	W	Range/default value
BU	Buffer usage	Y	N	% of program buffer used 0-100
CQ	Command queuing	Y	Y	1= Pauses until move complete (default) 0= continuous execution
DF	Drive Fault status	Y	N	See following section
DU	Distance Units	Y	Y	0= Motor steps or 1= Encoder steps, default=0
EI	Encoder Input	Y	Y	0=step/dir, 1=cw/ccw, 2=quad ABZ, default=2
EO	Encoder signal output	Y	Y	0=step/dir, 1=cw/ccw, 2=quad ABZ, default=0
EP	Encoder Position	Y	Y	-2,147,483,648 to 2,147,483,647 counts
ER	Feedback encoder resolution	Y	Y	1-65,535
EX	Comms. Response Style & echo control	Y	Y	0= speak when spoken to, echo off, RS485,multi axis 1= speak whenever, echo off, RS485, single axis 2= speak when spoken to, echo on, RS232,multi axis 3= speak whenever, echo on, RS232, single axis
IN	Inputs	N	N	Same format as IS command
IP	In Position flag	Y	N	1= In position or 0= not yet in position
IT	In Position Time	Y	Y	1-500mS, default=10mS
MC	Motor current	Y	Y	50, 60, 70, 80, 90, 100 (in percent), default see RFS
MR	Motor resolution	Y	Y	400, 800, 2000, 4000, default see RFS
MS	Motor standby current	Y	Y	50, 70 (in percent) , default see RFS
MV	Moving	Y	N	Flag 1= moving or 0 = not moving
PA	Position Absolute	Y	Y	-2,147,483,648 to 2,147,483,647 steps/counts, default 0
PE	Position Error	Y	N	Encoder count or motor steps, depending on DU
PI	Position Incremental	Y	Y	-2,147,483,648 to 2,147,483,647 steps/counts, default 0
RB	Ready/Busy flag	Y	N	Flag 0= ready or 1= busy
RM	Registration Move	Y	N	Flag 1= reg move in progress, 0 = not doing reg move
RV	ReVision of software	Y	N	x.yy major.minor
SN	Serial number	Y	N	Drive serial number (if available)
ST	Status of indexing	Y	N	See following section
UF	User program Fault status	Y	N	See following section

Table 5-2. List of System Variables

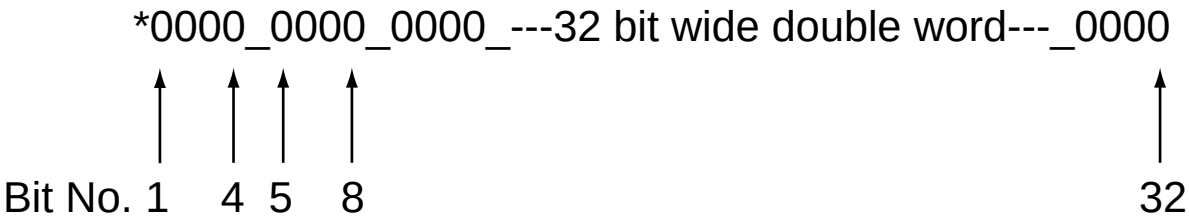
Reporting the Status of Variables

By examining Table 5-2 you can see that most system variables take a numerical value or record a simple ON/OFF state (0 or 1). Certain variables perform a reporting function which provides you with information on the status of the indexer and any drive faults present in the hardware or user program code.

System variables can be read using the R command. For example, **1R(RV)** will report **1.7*, meaning the revision level of software in the drive being accessed is 1.7.

Status Variable Reporting

Variable ST is a 32-bit double word that contains status information.
When read, ST reports a 32-bit double word pattern of the form:



Where a bit is set (displayed as a 1) its bit number can be determined and compared with the bit number value given in Table 5-3 to determine the Status Information being reported. Use the Read command to display the ST word pattern, that is '**aR(ST)'.**

Bit Number	Status Information
1	Command processing paused
2	Looping (command executing)
3	Wait for trigger (input)
4	Running program
5	Going home
6	Waiting for delay timeout
7	Registration in progress
9	Motor energised
11	Event triggered - active until trigger inputs are reset
12	Input in LSEL not matching label
13	-ve limit seen during last move
14	+ve limit seen during last move

Table 5-3. Status Bits Description (continued over the page)

17	Executing a position maintenance move
18	Possible stall
19	Moving (In motion)
20	Stationary (In position)
21	No registration signal seen in registration window
22	Cannot stop within the defined registration distance
23 to 32	Reserved

Table 5-3. Status Bits Description (continued)

Status Variable Byte Reporting

A convenient and more compact way of interrogating the status variable is to test it a byte at a time using the **STn** within a read command, where n is used to select the byte to be tested. For example to read or test the first 8 bits (first byte) of the ST variable status word, use ST1.
Since the status word consists of 4 bytes the relevant part of the word can be read using ST1 (bits 1 to 8), ST2 (bits 9 to 16), ST3 (bits 17 to 24) or ST4 (bits 25 to 32).

Fault Status Reporting

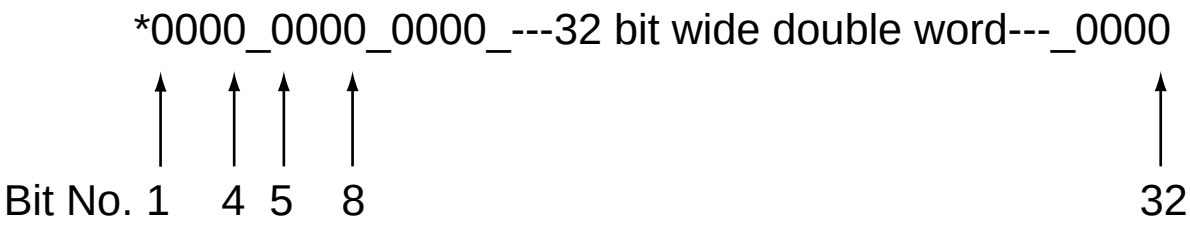
Faults are classified into two groups:

Drive Faults DF (hardware faults present in the drive)
or
User Faults UF (user faults caused by program or external hardware)

Drive Faults

Hardware drive faults will cause the drive output stage to be turned OFF (de-energised). This will cause the Drive LED to turn RED. Once the fault has been corrected the drive may be re-energised using the ON command.

When read, DF reports a 32-bit double word pattern of the form:



Where a bit is set (displayed as a 1) its bit number can be determined and compared with the bit number value given in Table 5-4 to determine the Drive Fault being reported.
Use the Read command to display the DF word pattern, that is '**aR(DF)'.**

Bit Number	DF Information
1	Composite fault
2	Output stage over current
3	Supply rail failure
4	Ambient over temperature
5	Drive over temperature
6	Configuration error
7	Motor high voltage rail failure
8	Output fault
9 to 32	Reserved

Table 5-4. Drive Fault Bit Description

Drive Fault Byte Reporting

In exactly the same way as the status variable, the drive fault status can be reported a byte at a time, using **DFn** within a read command.

User Faults

User faults can be caused by programming errors, such as issuing a GO command when the drive is de-energised or other testing or commissioning actions. They are reported in a 32-bit word format the same as Drive Faults.

Performing a read UF command will report the current state of any User Faults listed in Table 5-5.

Bit Number	UF Information
1	Value is out of range
2	Incorrect command syntax
3	Last label already in use
4	Label of this name not defined
5	Missing Z pulse when homing
6	Homing failed - no signal detected
7	Home signal too narrow
8	Drive de-energised
9	Cannot relate END statement to a label
10	Program memory buffer full
11	No more motion profiles available
12	No more sequence labels available

Table 5-5. User Fault Bit Description (continued over the page)

13	End of travel limit hit
14	Still moving
15	Deceleration error
16	Transmit buffer overflow*
17	User program nesting overflow
18	Reserved
19	Drive not ready
21	Boost duty cycle error (not used on PDFX)
22	Save error
23 to 32	Reserved

Table 5-5. User Fault Bit Description (continued)

*sends an ASCII 'bell' character to indicate a buffer overflow condition.

User Fault Byte Reporting

In exactly the same way as the status variable, the user fault status can be reported a byte at a time, using **UFn** within a read command. For example to read or test the first 8 bits (first byte) of the UF variable status word, use UF1. Since the status word consists of 4 bytes the relevant part of the word can be read using UF1 (bits 1 to 8), UF2 (bits 9 to 16), UF3 (bits 17 to 24) or UF4 (bits 25 to 32).

Resetting User Fault Bits

The User Fault variable (UF) is cleared to all zeroes once it has been read, but it is not cleared when it is tested within an IF or TR (wait for trigger) command.

Byte Testing

Remember, the code can be used to test a particular byte of the user fault word. For example:

```
1IF(UF2,<>,10X10X10)    ; if contents of UF2 does not equal 10X10X10 execute
                        ; the next line of code, otherwise skip the next line
1A500                   ; acceleration and deceleration changed to 500rps2 if
                        ; previous test was true
1R(UF2)                 ; read the value of byte 2 of the user fault status word
*01010101               ; contents of byte 2 (which is now cleared by the read)
```

Note: When the UF is tested it is not cleared to all zeroes, but it is by the read.

This example uses a conditional test to compare UF2 with 10X10X10. The use of conditional tests within IF and TR commands is described in the sub-section entitled ***Conditional Code***.

Reporting System Information During Code Development

Whilst developing a program using EASI-Tools, it is likely that certain blocks of code when downloaded to the drive will return an **E* error code. To analyse the cause of the error you can make use of the following system variables:

RB (Ready/Busy flag)
DF (Drive Fault)
ST (Status of indexing)
UF (User Faults)

Which, when read, will report back the cause of the error. For example, selecting status report 'User' following a **E* may report back 'Label of this name not defined'.

If a command does not execute when you expect it to, read the status of the RB system variable. It will be busy (=1) if any of the following are active: motor moving, paused, looping, waiting for a trigger, running a program, going home, executing a time delay or performing a registration move. If RB is not busy, read the drive status followed by user faults, finally check the E command to make sure communications are enabled.

Within EASI-Tools a system variable can be read using the status report window or using the report command directly from the edit window (For example **3R(ST)**). Using this style of report an immediate response will be returned which will not be saved within the program code. If you wish to save the response, use the single byte version of the report command, that is **3R(ST1)**, **3R(ST2)**, **3R(ST3)** or **3R(ST4)** depending upon which byte of the variable you wish to capture.

If the indexer is waiting on a trigger command, you can still send an interrogation command such as **1R(RB)**, **1R(DF1)**, **1R(ST1)**, **1R(UF1)**, **1IS**, **1O**, **1A** .....and a report will be returned. However, if a buffered command is sent, such as **G** or **1A10**, then all future interrogation commands are buffered, apart from **1R(RB)**, **1R(DF)**, **1R(ST)** and **1R(UF)**. Throughout this reporting process, such commands as **1K**, **1S**, **1PS**, **1C**, **1E1** and **1E0** can still be executed.

Conditional Code

The flow of a motion control program will depend upon the position of the motor in combination with the value of particular inputs and commands. System variables are used to continuously monitor the state of a drive's indexer and are able to report such things as 'status of indexing', 'position error' or 'moving'/'not moving' as listed in Table 5-1. Certain system variables are capable of being tested by the TR (wait for trigger) or IF (test condition) commands. This allows the value of a system variable to be tested in the following ways:

=	Equals
<>	Does not equal
>	Greater than
<	Less than

The TR command pauses program execution until the required trigger condition is met, while the IF command tests the value of a system variable and executes the next line of code if it is true, otherwise it skips the next line of code. Use of these commands allows synchronisation with external events and program branching.

System variables which may be used in conjunction with the IF command are listed in Table 5-6. Where the variable can also be used with the TR command a 'Y' appears in the TR column.

Variable	Name	>	<	=	<>	TR	Format
DF	Drive fault status	N	N	Y	Y	Y	binary
EP	Encoder position	Y	Y	Y*	Y	Y	decimal
IN	Inputs	N	N	Y	Y	Y	binary
IP	In position flag	N	N	Y	Y	Y	bit
MV	Moving	N	N	Y	Y	Y	bit
PA	Position absolute	Y	Y	Y*	Y	Y	decimal
PE	Position error	Y	Y	Y*	Y	Y	decimal
PI	Position incremental	Y	Y	Y*	Y	Y	decimal
RM	Registration move	N	N	Y	Y	N	bit
ST	Status of indexing	N	N	Y	Y	Y	binary
UF	User program fault status	N	N	Y	Y	N	binary

* Not recommended during motion

Table 5-6. System Variables that can be used for Conditional Control

Conditional Code Example

The following code is a good example of how the conditional IF statement can be used for fault diagnosis within the FAULT label.

```
1FAULT:                          ;define check label
      1IF(UF2,=,XXXXXX1X)        ;deceleration error
          1"Decel_Err"
      1IF(DF1,<>,00000000)       ;warning of a drive fault
          1"Drive_Flt"
      1IF(ST1,=,XXXXX1XX)        ;waiting for a delay timeout
          1"Delay_timeout"
      1IF(ST2,=,1XXXXXXX)        ;motor is energised
          1"Motor_On"
      1T1                        ;wait 1 second
1END                             ;end of definition
```

Command Queuing

Command queuing (system variable CQ) in mode incremental is normally enabled, this means commands are buffered, each command waiting for the previous command to complete before the next one is issued. In certain circumstances this sequential operation needs to be disabled, for example if you need to generate a trigger pulse part way through a move. Normally, the move would complete before the trigger command is executed, but by disabling command queuing the trigger command becomes immediate and will operate when the required trigger conditions are met.
For example, the following code would allow output 1 to signal PA is greater than 10000 before finishing the move.

```
1MAIN:              ;define label
1CQ0                ;enable continuous execution of commands
1G                  ;go
1TR(PA,>,10000)     ;trigger when position absolute becomes greater than 10000
1O(1)               ;output 1
1END
```

Motion Control Using the PDFX Command Set

Move Types

Mechanical movement results from the rotary motion of a motor shaft. By controlling the velocity, acceleration, distance and direction of the motor, different move profiles can be created for particular applications. Move types can be preset, meaning a move is made in a controlled way over a specified distance, or continuous where only acceleration, velocity and direction are defined, distance being ignored. Various move types can be selected using the MODE command.

Preset Moves

Preset moves allow you to position a target or work-piece in relation to the motor's previous stopped position (incremental moves) or in relation to a defined zero reference position (absolute moves).

Absolute Preset Moves

An absolute preset move will move the shaft of the motor a specified distance from the absolute zero position (MODE A).

Incremental Preset Moves

When the MODE command is used to select indexed move with incremental positioning (Mode I), the motor shaft can be moved a specified distance from its starting position in either a clockwise (CW) or counter clockwise (CCW) direction.

Note: a positive direction is defined as one resulting in clockwise (CW) rotation of the motor shaft when viewed from the flange.

Continuous Moves

This mode is useful for applications which require constant spinning of the load, when the motor must stop after a period of time has elapsed rather than after a fixed distance, or when the motor must be synchronised to external events such as trigger input signals (MODE C).

Voltage Controlled Moves

A feature of the PDFX drive is an input which allows an external analogue signal to control the velocity of motion when executing continuous moves. This can be in one direction only, called a unidirectional voltage controlled move (MU), or in both directions, which is called a bi-directional voltage controlled move (MB). For MB to work in both directions a deadband must be specified, for example **VS0.1**, otherwise rotation will be clockwise only.

The range of control possible is set by the value of the present velocity, for example if the MODE was set to unidirectional and the present velocity was 30 revs per second a variable voltage could control the speed of the motor up to 30 rps. The lower end of the velocity

control range is zero, but in a practical system the velocity will be brought to a halt as soon as the edge of the dead band is encountered, which is set by the start/stop command velocity parameter VS. The programmed motor speed for MU or MB modes is given by :

$$\text{Programmed velocity} = \text{commanded velocity} \times \frac{\text{input volts}}{10}$$

Where: input volts is the voltage difference between inputs AV+ and AV-
commanded velocity is the velocity entered using the V command

Acceleration and deceleration rates are limited by the values set using the A, AA and AD commands.

If the programmed velocity falls below the start/stop velocity VS, the motor will stop. This allows a well defined dead-band to be set up in which no motor movement takes place despite changes in input voltage, as shown in Figure 5-2.

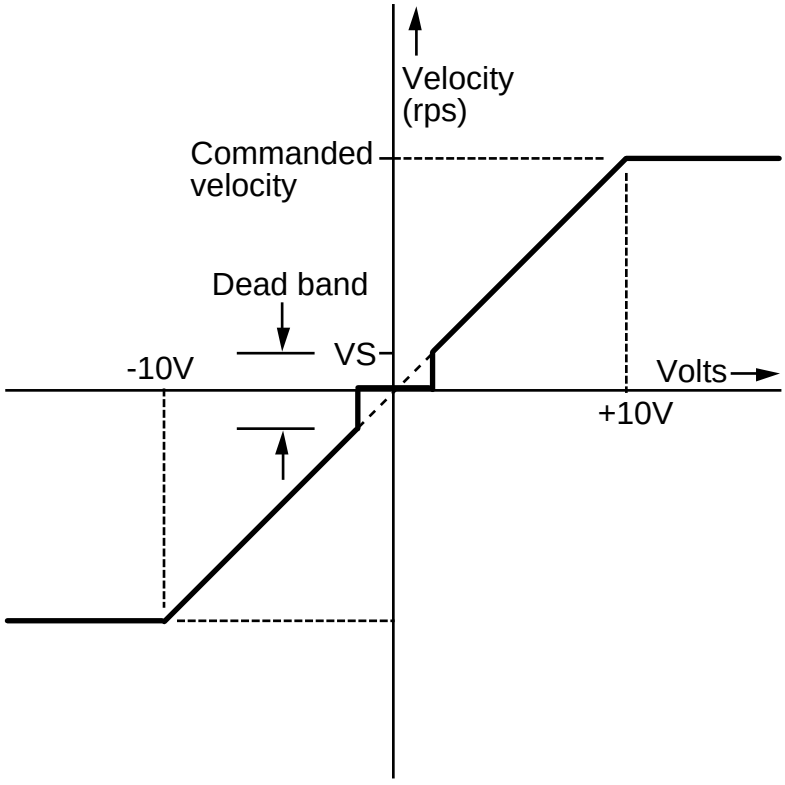


Figure 5-2. Bi-directional Voltage Control Mode

It is recommended a dead-band characteristic is used to limit the effects of electrical noise and offsets, and to ensure a clean motor stop. A good starting point for determining the value of VS is to set it to approximately 5% of the value of the commanded velocity V.
In bi-directional mode (MB) the motor rotates in a clockwise (CW) direction with a positive voltage applied and in a counter clockwise (CCW) direction when a negative voltage is applied. See motor direction box.

In unidirectional mode (MU) only the top half of the characteristic is followed. The motor responds only to positive voltages and rotates in the CW direction only. When a negative voltage is applied the motor remains stationary.

Motor Direction & Positive Motion

A positive direction command usually produces clockwise (CW) rotation of the motor shaft when viewed from the shaft end*.
However, when limit switches are used it is important that the positive direction produces motion towards the positive limit switch (see sub-section on HOMING). If this is not the case, interchange the motor connections to A+ and A- to reverse motor direction. If the motor is fitted with an encoder, one phase of the encoder output will also need to be reversed to match the motor direction.

* In practice this depends on internal motor wiring which varies between motor manufacturers.

The analogue input can be used to manually vary the speed of a motor via a potentiometer or it may be used to automatically match the speed of a motor to an external analogue control signal.

Program Example

```
; The following example demonstrates how to use the analogue input as a jog facility.
; It is assumed that a potentiometer is connected to the analogue inputs so that +/-10V may
;be supplied
; Input 1 is wired to a switch which selects between normal running and jog mode
; When the normal code ends the input will be tested by the main routine and either re-
;called or the jog routine
; will be called instead.
; The jog routine uses input 2 as a RUN/STOP input.

1K
1OFF
1CLEAR(ALL)
1DECLARE(ANJOG)
1DECLARE(MAIN)
1DECLARE(NORM)

1ANJOG:
1O(1)          ; set jog mode active output
1V5            ; full scale analogue i/p equates to 25 rps
1A1            ; slowly accelerate*
1VS0.2         ; set a small deadband so that there is a position on the jog
               pot where the motor can stop
1MB            ; set-up a bi-directional move
1ON
1W(CQ,0)       ; turn off command queuing
```

```
1TR(IN,=,X1)          ; wait for jog enable switch
1G
1TR(IN,=,X0)          ; wait for jog disable switch
1S
1W(CQ,1)
1O(0)                 ; clear jog mode active output
1END

1MAIN:
1IF(IN,=,1X)          ; jog request
1GOSUB(ANJOG)         ;
1IF(IN,=,0X)          ; normal running
1GOSUB(NORM)          ;
1END

1START:                              ; example start script
1OFF                                 ; make sure drive de-energised
1O(00000000)                         ; turn off all outputs
1LIMITS(0,1,0)                       ; set up limits
1PROFILE1(100,100,20000,25,0)        ; set up a move profile
1MI                                  ; mode incremental
1LOOP(MAIN,0)                        ; loop main forever
1END

1NORM:
1O(X1)                               ; set normal mode active output
                                     ; normal program goes here
1O(X0)                               ; clear normal mode active output
1END
```

*Note: Setting too high an acceleration rate could cause the motor to stall if the speed control potentiometer was subjected to a rapid change of position, that is a ‘sharp twist’.

Motion Profiles

In any motion control application the most important requirement is precise, controlled shaft rotation, whether it be with respect to position, time or velocity. This pattern of movement is called a Motion Profile. Generally, such a profile can be represented graphically in the form of a diagram of time or distance moved plotted against velocity. For example, the triangular shaped profile shown in Figure 5-3 would be obtained if you programmed either a very low acceleration or a very high velocity or both over a relatively short distance.

Triangular Profile

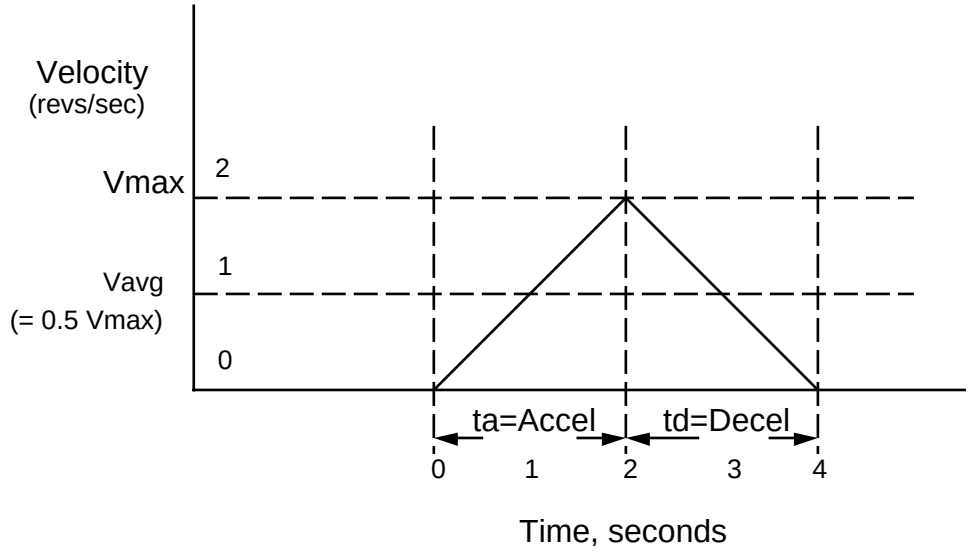


Figure 5-3. Triangular Profile

Setting the acceleration to 1 rev/sec^2 with the velocity set to 5 revs/sec over a distance of 16000 steps (4 revs), a triangular motion profile will result. This is because by the time the motor shaft has reached a velocity of 2 revs/sec, it will also have travelled half of the defined distance due to the acceleration setting of 1 rev/sec^2.

Trapezoidal Profile

A trapezoidal move profile results when the defined velocity, you have programmed, is attained before the motor shaft has moved half of the specified distance. This is due to a defined velocity that is low, a defined acceleration that is high, a move distance that is long, or a combination of all three. For example, if the acceleration is set to 10 revs/ sec^2, velocity is set to 1 rev/sec, and distance is specified as 20000 steps (5 revs), the resulting motion profile would look like this:

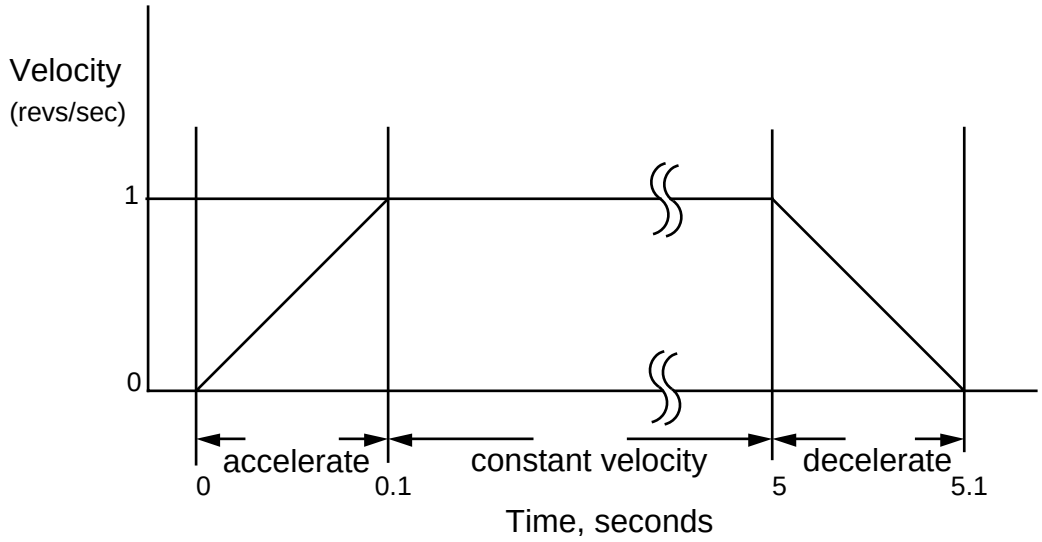


Figure 5-4. Trapezoidal Profile

Point to Point Positioning

Point to point positioning is often required in pick-and-place applications where a series of x,y co-ordinate moves need to be performed quickly and to high accuracy. This type of application can greatly benefit from the PDFX's instantaneous start/stop speed. By starting a stepper motor at a pre-defined velocity rather than letting it 'ramp' up to speed can save a considerable amount of move time within repetitive point to point positioning moves. In a similar manner, time may also be saved by stopping the motor immediately rather than allowing the speed to 'ramp' down to zero.

The trapezoidal profile often used for point to point positioning can be modified to include instantaneous start/stop, resulting in the profile shown in Figure 5-5.

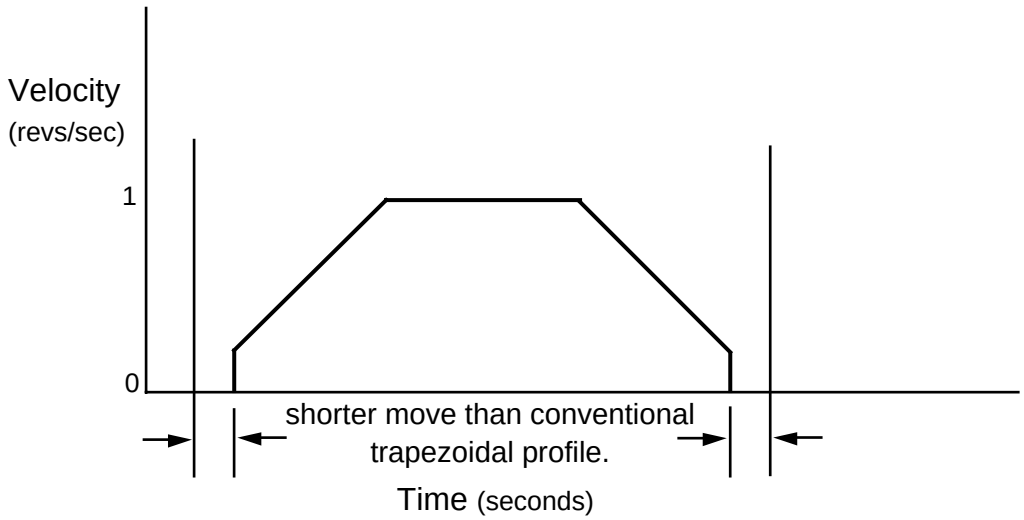


Figure 5-5. Modified Trapezoidal Profile

Registration

One of the major uses of registration is for packaging and labelling applications where a registration mark or label edge is used to sense the orientation of an object. Once detected a registration move can be triggered, which is a separate independent move that, for example, may position a jar for a labelling operation. The registration move itself often needs to be performed quickly (faster than the current move, to prevent queuing in serial batch processes), Figure 5-6 illustrates a typical registration move. The hold off distance prevents a spurious input incorrectly triggering a registration move. The fast status input, which detects a registration mark or edge signal can only trigger a registration move after hold off and within the registration window.

Note: A registration move is always performed in mode incremental, even if the drive is configured for mode absolute.

A successful detection of registration will immediately cause a registration move to be actioned, which is defined by the move profile number held in the registration command. At the end of the registration move motion halts and the code is continued following the registration command. If no registration mark is detected the original move is completed and the code will jump to the NOREG: label.

Care must be taken to make sure the registration move can be performed in the time available, taking into account the velocity of the initial move and the characteristics of the required registration move. If insufficient time is allowed, the registration move may appear to be performed, but the NOREG label is executed.

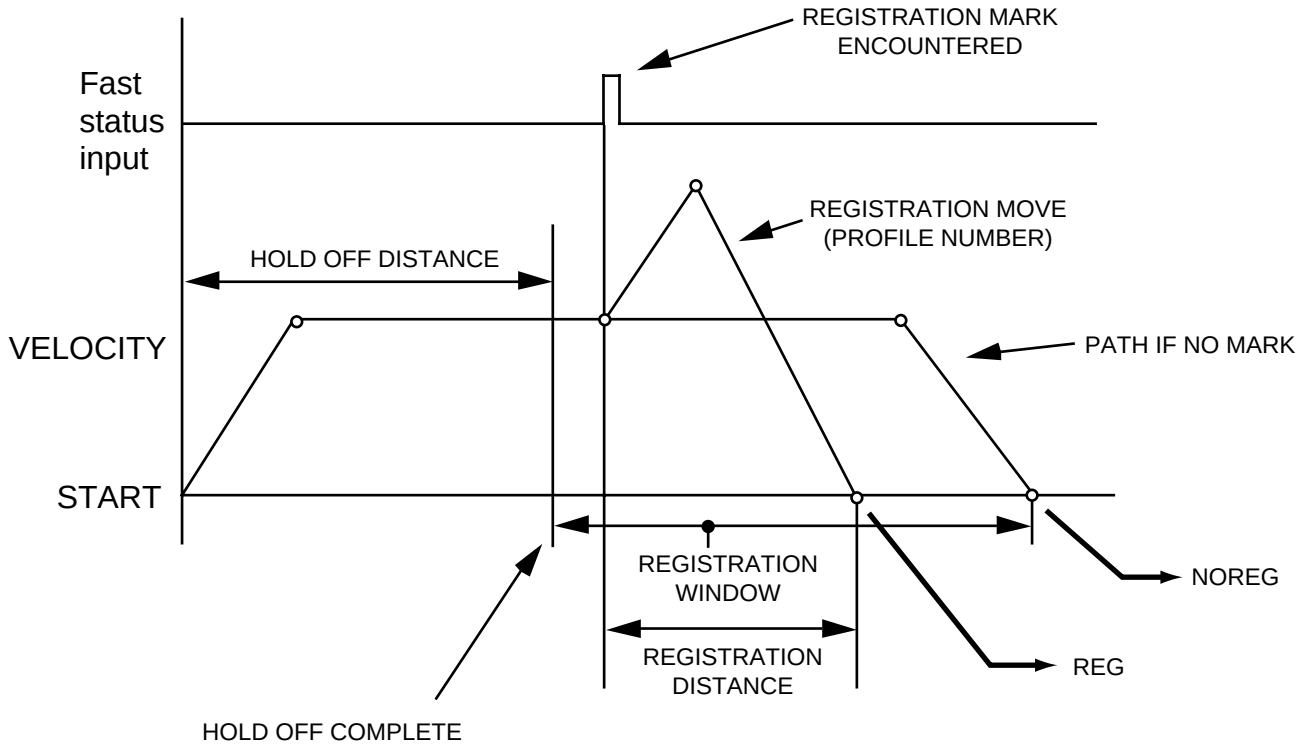


Figure 5-6. Registration Move Profile

A successful registration will cause the code, following the registration move, to jump to the REG label, from which normal program operation can continue.

Before you can perform a registration move, the following code elements must be in place:

1. Enable the registration function.
2. Completely specify the registration move required, in terms of distance, velocity, acceleration and deceleration.

Once a registration move has been defined, registration can be enabled/disabled using **aREG1** (to turn it ON) or **aREG0** (to turn it OFF), where 'a' defines the axis address.

When registration is enabled, any valid input edge will activate the registration move (whilst moving), however once activated any subsequent edge will have no effect. Consequently once the registration signal has been accepted for the current move all other registration signals will be ignored until a new move has been started.

An example of registration code is given below:

```
1START:                    ;start label definition
1PROFILE4(10,10,40000,5,1) ;define move parameters
1PROFILE5(20,20,10000,10,5) ;define move parameters
1ARM1                      ;arm the registration move
1REG1(1,5,5000)            ;define registration move parameters

1USE(4)                    ;use motion profile 4
1G                         ;execute move
1END                       ;end label definition

1REG:                      ;on reg mark valid turn on o/p 8 (batch counter)
1O(XXXXXXX1)
1T0.5                      ;wait for 500ms delay
1O(XXXXXXX0)               ;turn off o/p 8
1END                       ;end of label definition

1NOREG:                    ;if reg mark not valid/seen
1O(XXXXXX1X)               ;turn on o/p 7
1T0.25                     ;wait for 250ms delay
1O(XXXXXX0X)               ;turn off o/p 7
1END                       ;end of label definition
```

Homing

The term 'homing' refers to an automatic return to a mechanical reference position which is usually performed when the system is first powered up. All subsequent moves will then be relative to this reference position. The home position is usually determined by an optical or proximity switch, though a mechanical switch can also be used. When very high accuracy

is required, the home position can be referenced to the index track of an encoder which eliminates errors due to switch repeatability.

Another feature of homing on the PDFX is the ability to seek the nearest drive zero phase position, which also can improve the homing repeatability.

Definition Of Terms

To aid the description of homing operations the following terms are defined:

Positive motion - is motion towards the positive limit
Home switch positive edge - is the edge of the home switch on the positive limit side
Home switch negative edge - is the edge of the home switch on the negative limit side
Home switch operating range - is the distance moved whilst the switch is operated
Zero phase - a defined current state in the motor which results in one of 50 repeatable shaft positions

Four of these terms are illustrated in Figure 5-7.

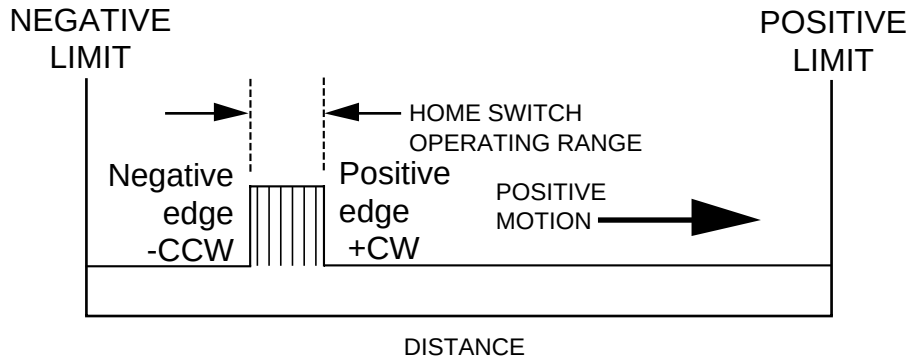


Figure 5-7. Home Switch Operation

Switch Considerations

Any type of switch will have an operating range that may correspond with a significant rotation of the motor shaft, depending upon the gear ratio between motor and load. Consequently, just detecting the home switch voltage level will not give a well defined home position. To improve the accuracy it is possible to stop on either the positive or negative edge of the home switch. To improve repeatability still further, the edge may be detected in combination with either the zero phase of the motor or the index pulse of an encoder (if fitted).

Switches generally exhibit a hysteresis characteristic when operated from opposite directions, therefore homing moves always make the final approach to the home switch from the same direction.

Homing Configuration Command

The command HOME is used to define the absolute position of the indexer, that is system variable PA is set to zero at the home position. The command allows you to define the

mechanical edge of the home switch at which you wish home to be. The command also allows you a choice of home switch type, that is normally closed or normally open, however if you change the switch type this does not change the edge you are homing to. Remember the positive edge is the mechanical edge of the home switch closest to the positive limit.

Other features of the HOME configuration command allow adjustment of the search speed and direction, the acceleration or deceleration rate to be used and mode selection. When setting the deceleration rate you must ensure sufficient distance is left between the home switch and any limit to make sure motion is brought to a halt after the home switch is detected and before a limit is reached. If not, the system will be brought to an immediate halt as soon as the limit is detected.

Mode Selection

By choosing mode 0 the indexer will detect the defined edge and will then decelerate to rest within the home switch operating range. Mode 1 will cause motion to stop at the mechanical edge of your choice (positive or negative). Mode 2 will behave the same as mode 1, but the indexer will also take account of the nearest drive zero phase position and will stop at that phase position. Finally mode 3 will use an encoder index pulse (if available) in combination with the requested mechanical edge detection and will stop at the nearest index pulse.

Mode 0 operation simply returns the motor to its home position at some point between the negative edge and positive edge of the home switch. Apart from knowing which edge of the switch was used the exact position within the home switch range is undefined. A more precise home position can be obtained by using mode 1, which allows the home position to be defined as either the positive or negative edge of the home switch. Note, although mode 1 fixes the home position at one of two edges the precise position is still subject to the repeatability of the home switch itself. Practical applications will exhibit variations in switch performance and consequently the home position will still be subject to variation by a small number of motor steps. To reduce this switch operating point variation modes 2 and 3 can be used.

If you wish to use Mode 2 your home switch must reliably and consistently operate in less than the distance separating two phase zero pulses, otherwise the home position will be inconsistent. For a 200 step/rev motor phase zero will occur 50 times per rev so the gearing between the motor and load must be arranged to allow unambiguous detection of the same phase zero occuring after the required home edge is detected. To aid the setup of an application using zero phase home detection, the HV LED will illuminate with an amber coloured light at every phase zero position of the motor.

If an encoder is fitted to the motor, Mode 3 operation offers the optimum way of consistently locating the home position. As only one index pulse occurs per rev of the motor, the mechanics of the application can be arranged to position the occurrence of the index pulse well away from either edge of the home switch. This means inconsistent home switch operation cannot influence the repeatability of the home position.

Go Home Command

The go home command (GH) is used to return to the reference home position. Issuing a GH command will cause motion in a direction defined by the HOME configuration command. Figure 5-8 shows the path taken if motion was started between the positive edge of the home switch and the positive limit (positive side of home). The dotted line represents positive movement and the solid line negative, although once past the positive edge of the home switch both merge to follow one common path. Positive movement results in motion towards the positive limit, once the limit is hit motion is reversed* and finally heads for the home switch. Negative motion will immediately head for the home switch.

*Note: Limit inputs must be armed to allow a move to bounce off a limit.

Assuming home is the positive edge of the home switch, as soon as the edge is detected motion is decelerated to a stop, direction of travel is reversed and a distance is calculated to move just outside the positive edge of the home switch. This new move is performed in a positive direction. Again motion is stopped and the direction of travel is reversed and a negative approach is made at a fixed velocity of 0.1 rps (start/stop range of motor) as soon as the positive edge is again detected the motor is stopped.

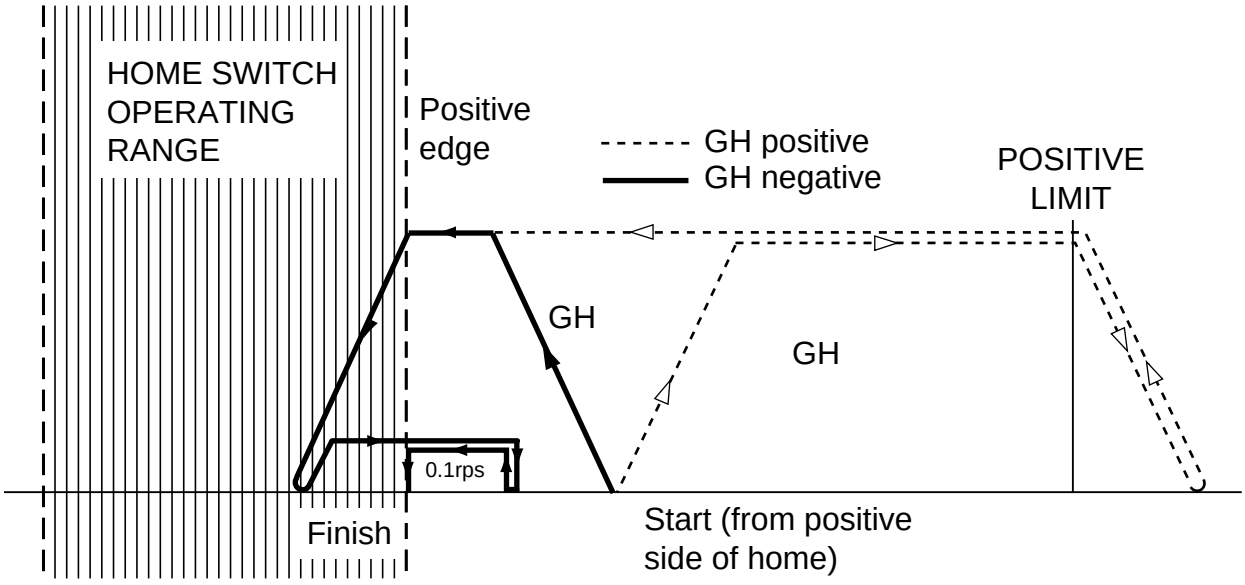


Figure 5-8. Go Home to Positive Edge

If the negative edge of the home switch is selected in the homing configuration command a similar motion path would be followed, but finishing on the other side of the home switch, as shown in Figure 5-9.

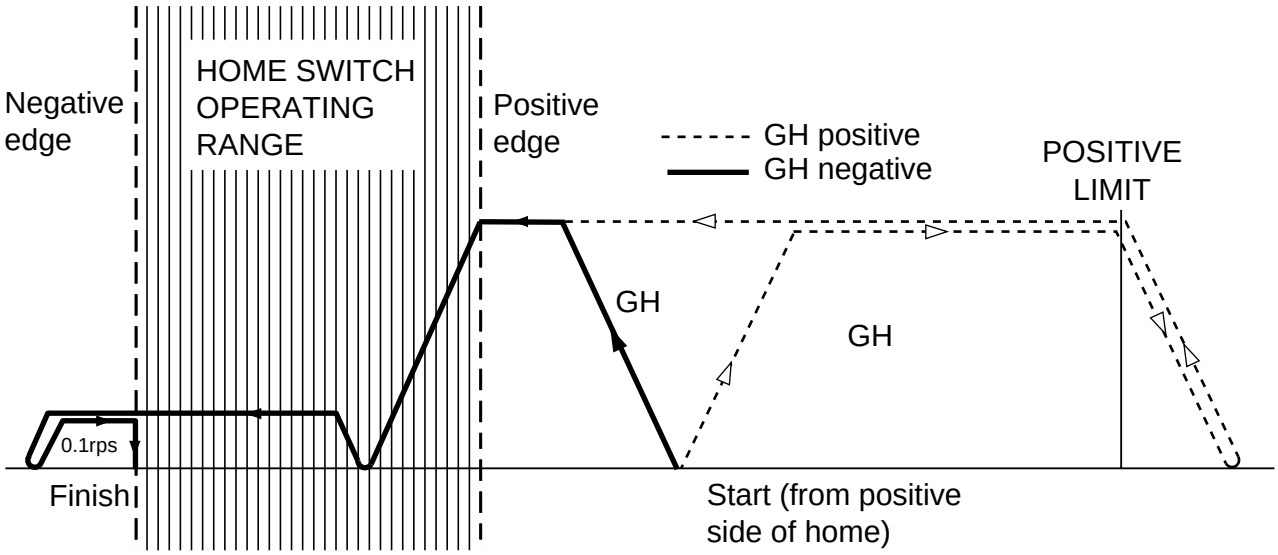


Figure 5-9. Go Home to Negative Edge

Motion starting on the negative side of the home switch will behave in a similar way, the only difference being the direction of travel. If the drive was started up already within the boundaries of the home switch and a go home command was given for a particular edge the motion would follow the path shown in Figure 5-10, depending upon which edge was requested. In this situation the home position is known so the indexer knows in which direction to travel to seek the appropriate edge. In Figure 5-10 acceleration and deceleration are set to the same value.

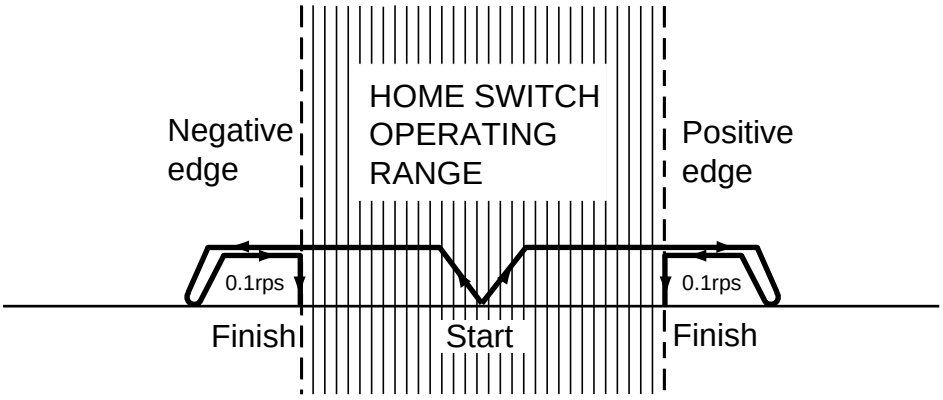


Figure 5-10. Go Home Starting from Home

Note: If the home configuration command is set to mode 0 and the home switch is already in its active range, no movement will take place.

Final Direction of Travel

Note: that no matter where motion starts from, that is from positive side of home switch, in the home switch region or from the negative side of the home switch, or in which direction it goes from its starting point (positive or negative), its final direction of travel towards a nominated home switch edge is always the same. Direction of travel towards the positive

edge of the home switch is always negative and the direction towards the negative edge of the home switch is always positive. This minimises variations in the home switch operating point between separate homing moves.

Example of Homing (Datum) Routine

```
1START:                 ; start label definition
1DECLARE(MOVE3)         ; declare label
1LIMITS(3,0,0)          ; configure limits (disable, n/c).
1OFF                    ; remove motor power before changing motor parameters
1W(MC,50)               ; set motor current to 50%.
1W(MR,4000)             ; set motor resolution 4000 steps rev.
1ON
1HOME1(+,1,-15,100,1)   ; configure the home parameters
1GOTO(MOVE3)            ; transfer to label move 3
1END                    ; end of label definition

1MOVE3:                 ; define program label move 3
1O(0)                   ; turn off o/p 1
1GH                     ; execute the go home move
1O(1)                   ; turn on o/p 1 after go home complete
1A100                   ; set acceleration to 100rps2
1V25                    ; set velocity to 25 rps
1D4000                  ; distance to 1 motor rev
1G                      ; execute move
1END                    ; end of program move 3 definition
```

Limit Switches

The drive has two limit inputs, the positive limit input and the negative limit input. When wiring the limit switches it is essential to check that a positive direction command produces motion towards the positive limit switch. If this is not the case, interchange the motor connections to A+ and A- to reverse the motor direction. If the motor is fitted with an encoder, one phase of the encoder output will also need to be reversed to match the motor direction.

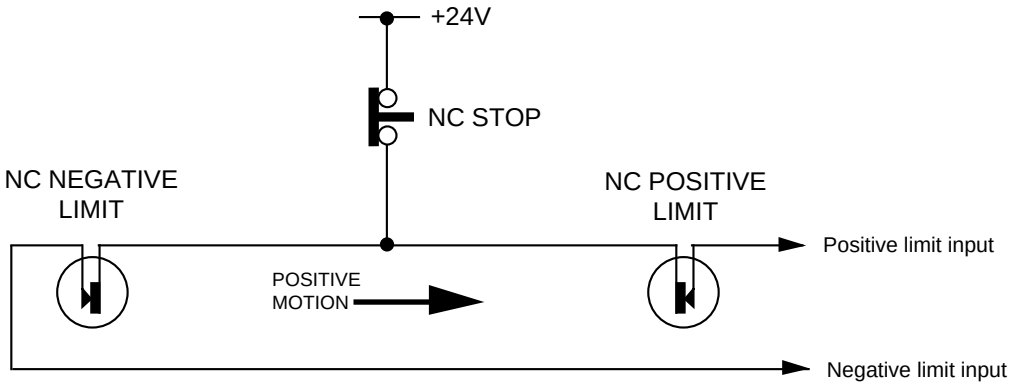


Figure 5-11. Limit and Stop Switch Configuration

Stop Input

A Stop function can be implemented based upon the scheme shown in Figure 5-11. Whilst motion is in progress, momentarily pressing the STOP button will take both limit inputs 'low' which will stop the motor at the limit deceleration rate. To use this form of hardware stop, limits will need to be enabled in the LIMITS command with the type field set to '1' (default).

Pressing the STOP button will activate both limits at once, this will cause motion to halt and no more movement will take place until both limits have been cleared. The status will report the first limit seen, but if both limit switches operated within the same millisecond period the positive limit will be reported as being active.

Following a hardware stop being actioned, the drive should be sent a Z command to reset its indexer (aZ).

User I/O Configuration

The PDFX has ten logic inputs, one of which is dedicated for high speed use, and one analogue differential input for speed control. Eight logic outputs are available for signalling and logic control. All user inputs and outputs operate with high voltage logic levels to maximise noise immunity.

Analogue Input

A differential analogue input can be used as a bipolar ±10V input or as a unipolar 0 to 10V input. Figure 5-12 shows the input circuit.

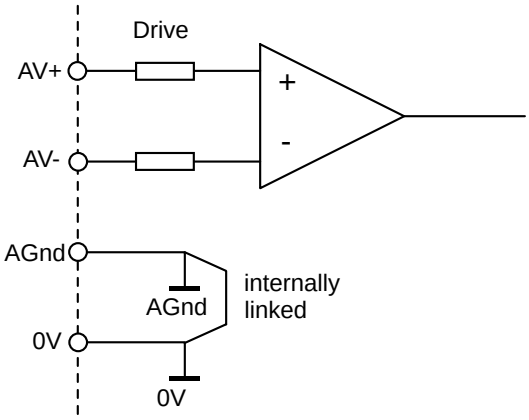


Figure 5-12. Analogue Input Circuit

When you make connections to the analogue input always use both AV+ and AV-, for example Figure 5-13 shows unipolar and bipolar connections of a potentiometer to the analogue input. Note how the AV- connection is wired in both cases.

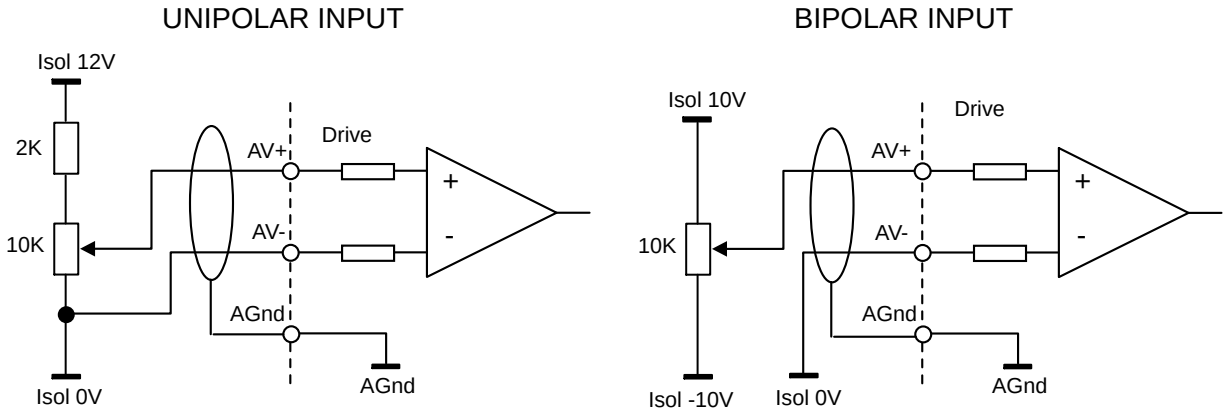


Figure 5-13. Unipolar and Bipolar Connections

If a single +24V is the only supply available, a bipolar connection can be arranged as shown in Figure 5-14. Use of this circuit arrangement will produce a slightly non-linear response.

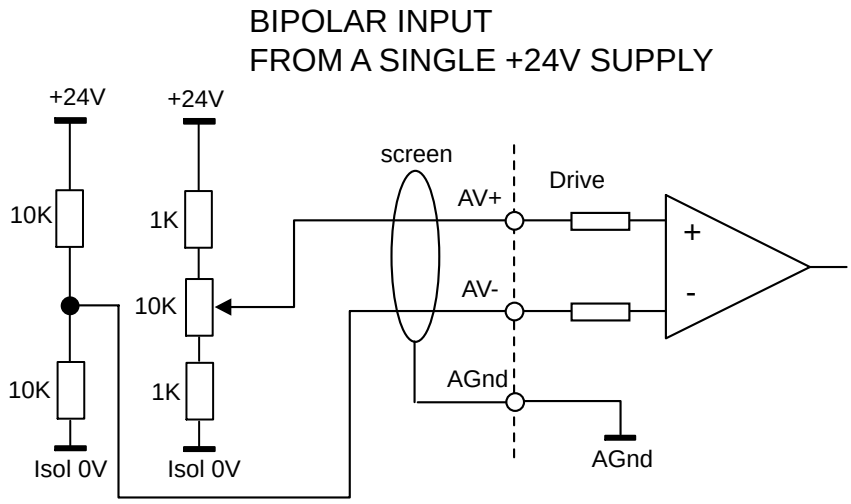


Figure 5-14. Bipolar Connections Using a Single 24V Supply

When making a remote analogue connection to the drive it is recommended that you use a screened twisted pair cable and arrange the connections as shown in Figure 5-15.

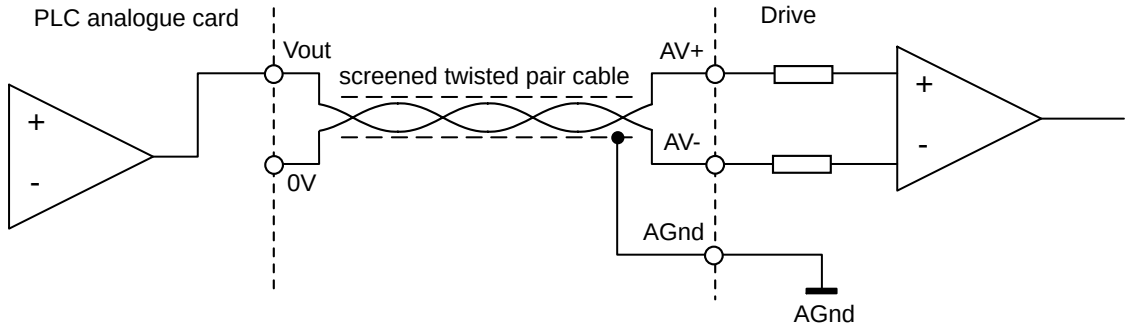


Figure 5-15. Remote Analogue Connection

User Logic Inputs

Ten high level logic inputs are provided for use with 24V PNP style control signals. Each individual input circuit has the circuit arrangement shown in Figure 5-16.

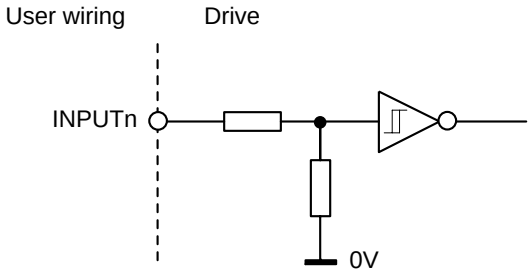


Figure 5-16. Input Logic Circuit

This style of input can be used with PLC outputs or simple switch contacts as shown in Figure 5-17.

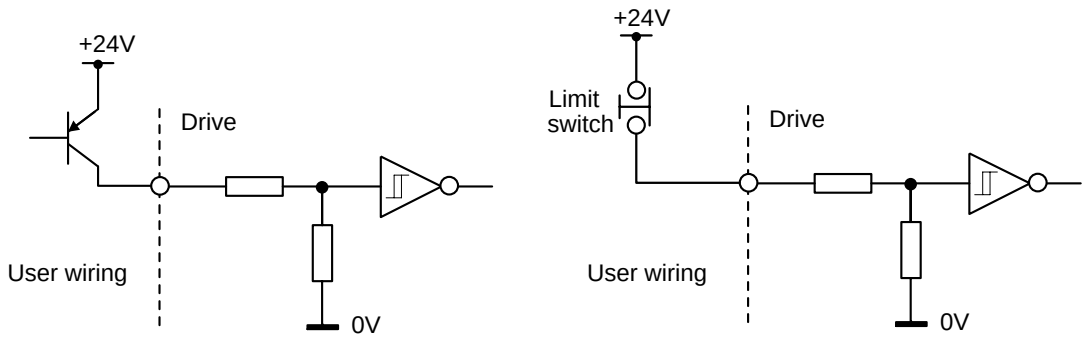


Figure 5-17. Input Driven from PNP Output or Limit Switch

User Outputs

Eight NPN 24V user outputs are available, having a circuit arrangement shown in Figure 5-18.

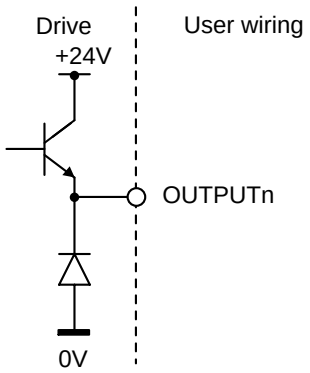


Figure 5-18. Output Logic Circuit

Outputs of this type can be used to drive relays directly, as shown in Figure 5-19. Each output is rated at a current of 60mA, but this cannot be drawn by all eight outputs simultaneously. A current limit of 400mA applies to the total current drawn by all eight outputs operating together. A prolonged use of these outputs at current levels greater than 400mA in total, will cause the outputs to turn OFF (drop to 0V), a drive fault sequence will be run, if defined and programmed.

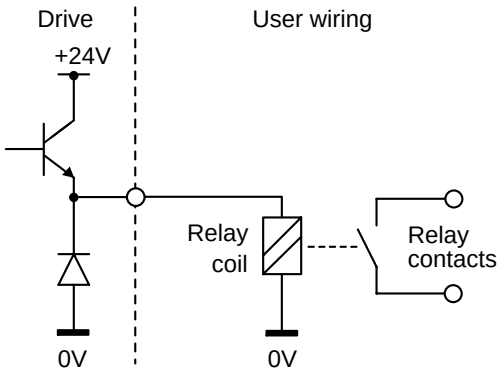


Figure 5-19. Output Logic Circuit

Note: the drive output circuit includes a protection diode to suppress relay coil back e.m.f effects. It is important that that the 0V of the coil is referenced back to the I/O 0V connector.

Fanout Capability

Each output has a fanout capability of 4. Figure 5-20 shows one Paragon user output driving four user inputs, internal input and output decoupling capacitor values are also shown.

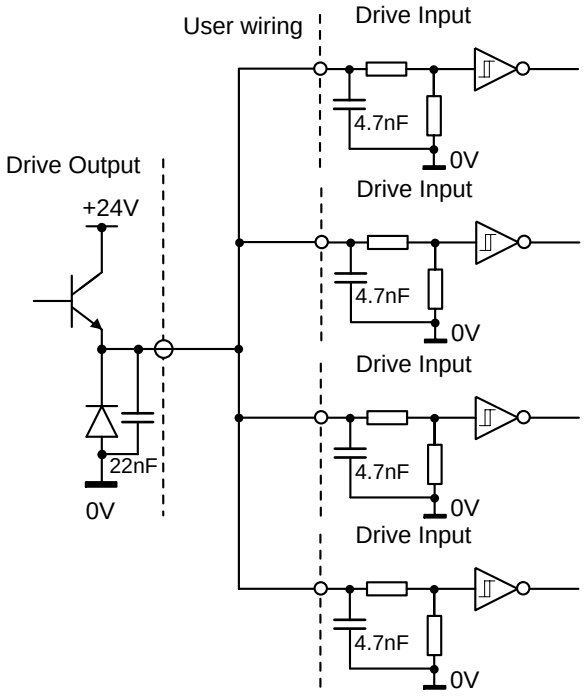


Figure 5-20. Maximum Output Fanout of Four

Simple Interface Considerations

In a situation where there is a requirement for a single low current +24V supply, such as providing power for a proximity home switch, consider using a permanently enabled output to provide the necessary supply voltage. In this situation make sure the current demand does not exceed the upper limit of 60mA.

Using Closed Loop Operation

Closed loop refers to the operation of a stepper motor/drive where the position of the stepper motor shaft is measured and compared with the position demanded by the input signal to the drive. This is normally arranged using an encoder attached to the motor's shaft and electrically connected to the stepper drive's encoder input, as shown in Figure 5-21.

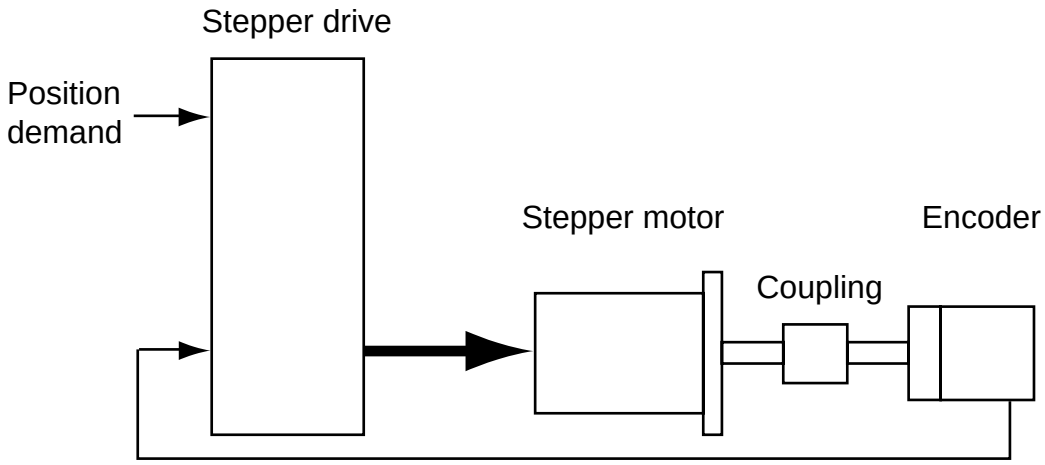


Figure 5-21. Closed Loop Operation

Closed loop operation is normally used in applications where a motor stall must be detected (stall detect) or where a known position of the motor shaft must be maintained within known limits (position maintenance).

Encoder Setup

To operate in closed-loop mode check that the encoder is correctly connected to the drive and attached to the motor shaft. For a correctly connected system, entering a positive distance value should cause the motor shaft to rotate in a CW (Clock Wise) direction when viewed from the shaft end and should cause the encoder count to increase in a positive direction. The encoder can be checked by entering a positive distance value (D) and noting the direction travelled by the motor shaft. Then de-energise the motor (using the OFF command) and read the current encoder position using the 1R(EP) command. Now, rotate the encoder shaft in the same positive direction by about half a turn. Again read the encoder position, which should be greater than the first reading, indicating that the encoder count is increasing for positive motion. If the second count is less than the first, cross over the A- and A+ signals on the encoder connector, and repeat the test until an increasing count is obtained.

To enable a stepper motor to position its shaft at the required encoder position, it must have a step position resolution at least four times that of the encoder being used. This 4:1 ratio (or greater) is necessary to account for any slight misalignment and coupling compliance between the motor and encoder. The drive supports four motor resolutions of 400, 800, 2000 and 4000 motor steps per revolution. This means, to create a stable system with a drive setting of 4000 steps per revolution, you will need to use an encoder which has a post quadrature resolution of 1000 counts per revolution, or less.

Note: Post quadrature resolution is a hardware technique for increasing encoder resolution by a factor of 4, consequently a 1000 counts per revolution encoder will have an actual line count of 250.

Position Maintenance

Position Maintenance is a method of correcting position errors by adding or subtracting motor steps once a move has been completed. To be able to make use of Position Maintenance a drive system needs to be fitted with a load or motor mounted encoder. The drive's controller will detect the difference between the number of steps the motor was commanded to move and the number of steps actually being reported by the encoder. This resultant position error is used, at the end of a move, to further command the motor in a direction to give the correct target encoder position, as shown in Figure 5-22.

Position maintenance can be used in mode incremental and mode absolute. It must not be used in conjunction with registration or homing.

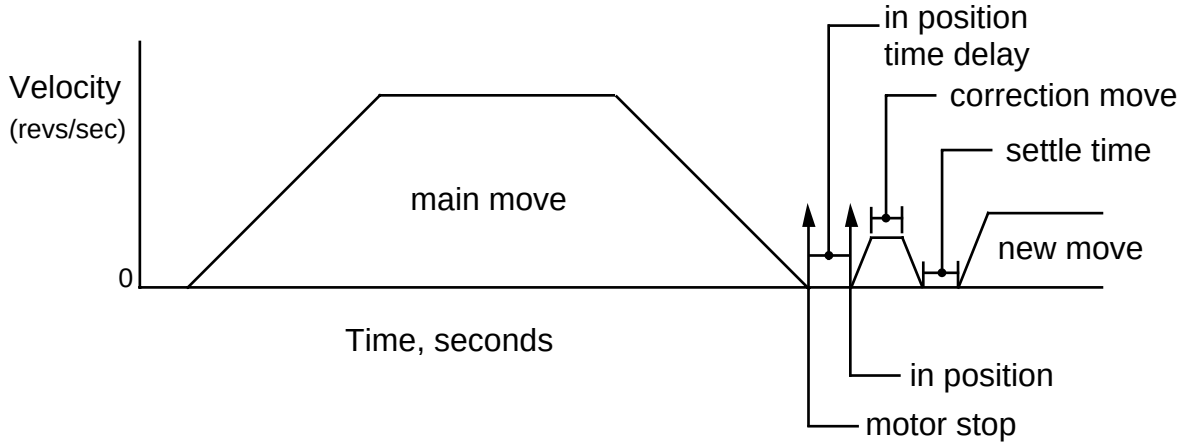


Figure 5-22. Position Maintenance Move Profile

Selecting Motor and Encoder Resolutions

To make use of Position Maintenance distances must be measured in encoder steps (DU=1) and the motor is recommended as having a step resolution of at least four times that of the encoder being used. This ratio allows precise encoder positioning by the adjustment of motor steps. In a practical application the motor may drive the load via a gearbox or the encoder may sense the load position in a manner that alters the number of encoder steps being generated. The aim is to achieve at least 4 motor steps for every encoder edge, thereby maintaining the minimum X4 step resolution of the motor over the encoder.

Dead Band Range

With Position Maintenance enabled, if you command the motor to move one revolution, at the end of the move you would expect the encoder count to read four times the encoder resolution. In practice, mechanical alignment errors and lost motion within the system will usually result in a small offset existing between commanded motor steps and the encoder reading. To take account of this offset an error band is defined, known as the Dead Band

Range. It has a range of 0 to 32767 encoder counts and a default value of 0. The number of counts entered must be positive, but the range will check the number of counts on both positive and negative sides of the target position. Position maintenance will have deemed to be successful if the final correction move positions the motor within the dead band range.

The correcting move sequence will be repeated whenever the sensed position is outside of the dead band and Position Maintenance is enabled and no other motion has been commanded. The maximum possible error that can exist without corrective action will be determined by the dead band. If the dead band were to be configured too narrow and the settle time to be negligible then it is possible to have oscillation around the target position. If the dead band were too wide then it is possible to have an inherent error carried forward into the next move. Position Maintenance will keep track of the error and compensate for the error at the end of the next move when used in mode incremental. Mode absolute, by nature of the mode, will always be less than the dead band distance away from the target position.

Output

An optional output can be used to signal when position maintenance is enabled and the motor is in position. In position is defined as not moving whilst positioned within the dead band range.

Settle Time

When Position Maintenance is enabled all moves will track actual position against commanded position. Position Maintenance allows the in-position signal to be held off for a settle-time, the value of which can be programmed in the command parameters.

Speed Of Correction

Once a move has been completed and the controller decides position maintenance correction is required, it will move the motor at a fixed speed of 1rps.

Example of Position Maintenance

The following code illustrates how position maintenance is implemented and the use of system variables used to set and check motor and encoder resolution. The drive must be set to encoder mode of operation for position maintenance to work. The example is based upon a motor resolution of 4000 steps per rev and a 200 line encoder giving 800 counts per rev.

```
1DECLARE(TRIAL)
1TRIAL:
1OFF
1W(MR,4000)          ; set motor resolution
1W(ER,800)           ; set encoder resolution
1ON
1R(EI)               ; check encoder is set to quadrature operation
1W(DU,1)             ; set encoder mode of operation
1POSMAIN0(20,3,0)    ; set-up position maintenance, dead band of 20 encoder
                       steps,ouput 3 to be used, no programmed settle time
```

```
1D4000          ; program distance, equivalent to 5 revs in encoder mode
1V5             ; set velocity to 5 rev/s
1A10            ; set acceleration to 10 rev/s/s
1POSMAIN1       ; enable position maintenance
1ARM1           ; arm command
1POSMAIN        ; check status of command
1W(PA,0)        ; set absolute position to zero
1G              ; start the move, motor turns 5 revs
1R(PA)          ; read position
1END
```

Following the G command the system will attempt to correct any final position error at the end of the move.

Stall Detection

Stall detection can only be used if an encoder is fitted to the motor. A stall is reported when the error between the commanded position and the actual position, determined by the encoder, exceeds the value set in the error window of the STALL command.

Stall Detection Set-up

Using closed-loop operation for stall detection, you have a choice of setting distances in motor counts or encoder counts (preferred option), depending on the state of the DU command. Once set however, changing DU will not automatically correct the error window units to take account of the Distance Units new setting. A stall condition is signalled when the number of steps fed to the motor does not match the number of steps received by the encoder. During a move the indexer constantly monitors any build-up of stall error, and once the difference exceeds a programmed error window, a stall condition is reported. Note, the stall error count is reset following an **ON**, **STALL**, **GH** or **G** command.

The error window size needs to be large enough to detect a single desynchronisation of the motor which is the equivalent of one rotor tooth or 4 full steps (7.2 degrees). Allowing for the normal lag and lead occurring during acceleration and deceleration, of up to 3.6 degrees, an overall error window of 5 degrees should be set - 14 steps with a 250-line encoder.

Stop on Stall

When STALL is enabled (on/off parameter set to a 1), motion is stopped if the error between the commanded position and the actual position exceeds the error window value. If a fault label is defined for this condition a fault will be reported and can be identified by reading the status bits. If stop on stall is not enabled the motor will attempt to finish the move and any programmed output will be turned ON.

Output

Any of the drive's outputs 1 to 8 can be turned ON when a stall condition is detected. This command option allows a stall to be signalled externally by lighting a lamp or LED.

Serial Communications

The PDFX has a single serial communications channel, but two electrical interfaces offering RS232 or RS485 transmit and receive operation. Consequently only one form of communication can be used at any particular time and the interface must be wired to suit the interface used.

Multiple drives can be connected using an RS232 daisy chain or RS485 4-wire or 2-wire multi-drop system. Drive hardware addressing is described in the Configuration section.

RS232 Communications

An RS232 interface is made available at pins 4 and 5 of the D-type 9-way RS232/485 connector. Where only one drive is being controlled from a PC the connections shown in Figure 5-23 may be used. Commands sent to a single drive must be prefixed with the address of that drive, set-up on the drive's bit switch (see Installation Section for bit switch settings).

Note: PC 9-way D-type serial connections are often 'brought out' as male plugs, in which case the pin positions will be the reverse of those shown. That is pin 1 becomes 5 and 5 becomes 1.

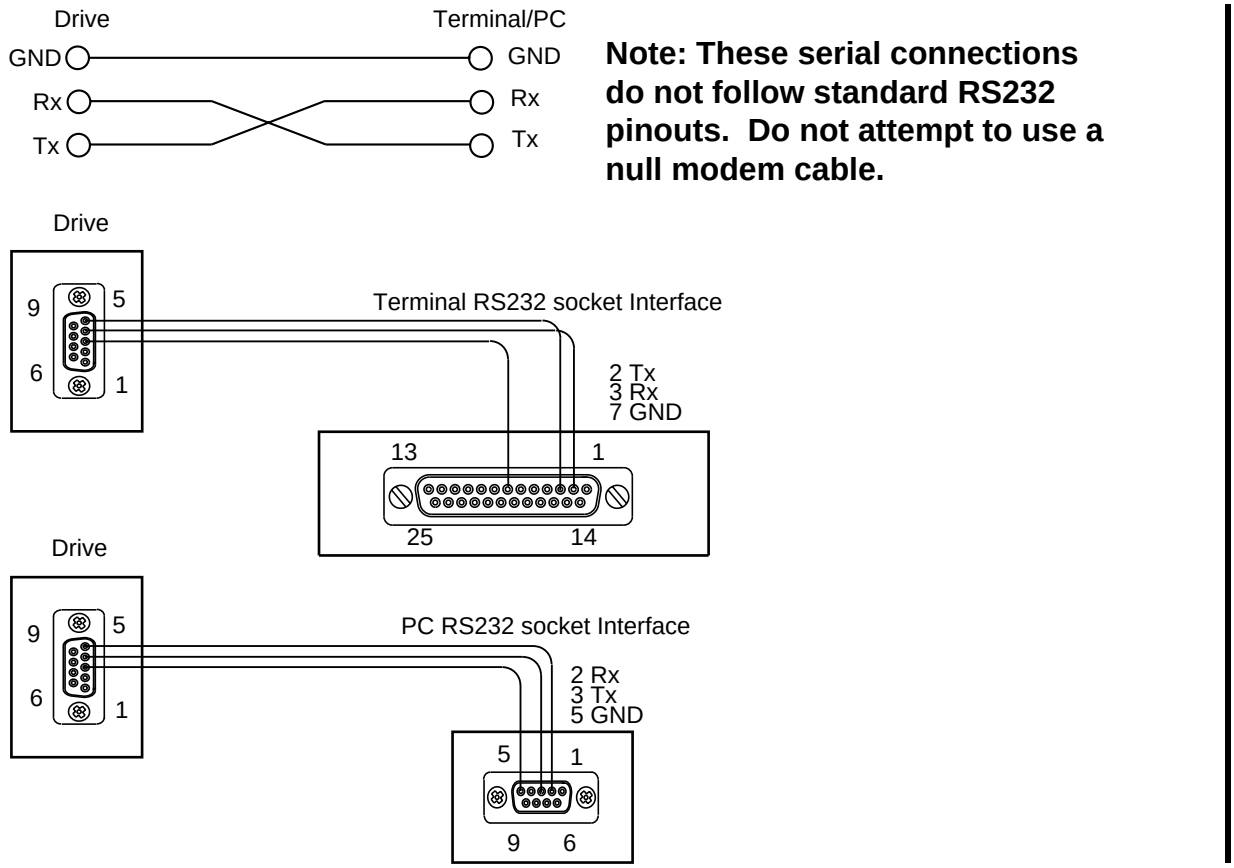


Figure 5-23. Simple RS232 to Terminal/PC Connection

When an application requires more than one drive, you will need to assign a unique address to each drive being used. PDFX drives have an address range of 1 to 63, set using the front panel bit switch.

RS232 Daisy Chain Connection

Multiple drives can be controlled using an RS232 daisy chain connection arranged as shown in Figure 5-24. Daisy chain RS232 connections require echo back to be turned ON and any terminal set to full-duplex with local echo turned OFF.

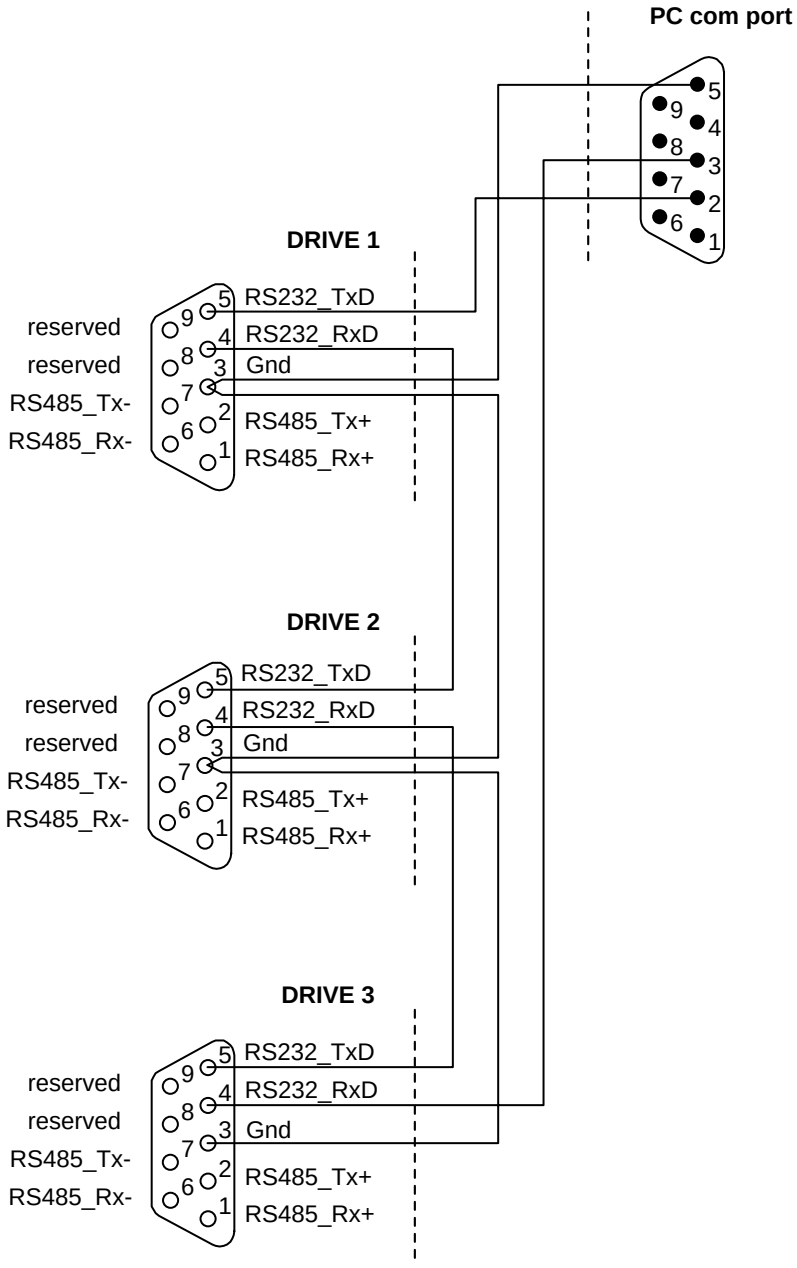


Figure 5-24. RS232 Daisy Chain Connection

RS485 Communications

RS485 uses a 2-wire differential interface offering greater noise immunity than RS232, much greater path lengths and improved transmission speed. Using RS485, the transmitting device can be tri-stated allowing the sharing of common transmit and receive lines, thus forming a 2-wire RS485 bus.

RS485 connections can be made in one of two ways:

4-wire simplex connections
2-wire half duplex (party line mode)

4-Wire Simplex Connections

This simplex mode allows for the differential transfer of data across a 4-wire interface in a way similar to the use of RS232. The advantage of this mode is the length of data line (1200m max. compared to RS232's 15m max.) and its higher resistance to pick-up and noise. Using this connection method multiple receivers can be connected on the same line allowing multi drop systems to be implemented.

2-Wire Half Duplex

This solution leads to more efficient cabling than the 4-wire system, but care has to be taken to avoid data collisions. Like the 4-wire system, the 2-wire system also gives multi-drop capability.

RS485 Multi-drop Connection

RS485 multi-drop connection of drives provides a 2-wire differential interface, allowing up to 32 drives to share a common transmit and receive path. The theoretical circuit layout of a multi-drop connection is shown in Figure 5-25. For a practical implementation of this form of connection see Figure 5-27.

When using a RS485 multi-drop bit switch 1 should be set to the OFF position to disable echo-back.

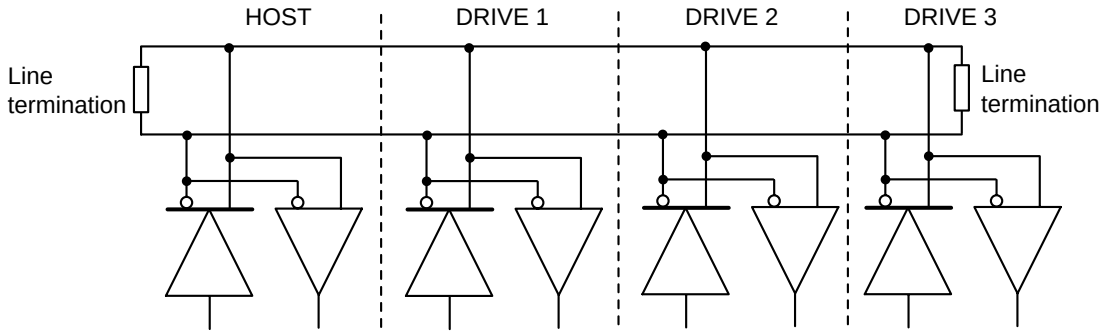


Figure 5-25. RS485 Muti-drop Connection

RS485 Transmission Line Considerations

Irrespective of the type of connection made, be it 4-wire simplex or 2-wire half-duplex, short RS485 connections can be considered as lumped parameter loads and do not need to be treated as transmission lines. As line lengths increase beyond ten metres, transmission line effects will become noticeable and an un-terminated line will generate reflections which will be subjected to greater delays as the line length increases. At low data rates (9600 baud) these reflections will not degrade communications and can be ignored.

When operating at 19200 baud, line termination is recommended at line lengths greater than 100m.

Wire Recommendations

RS485 connections will need to be made using screened twisted pair data transmission cable. The cable should have a characteristic impedance of 100 Ω and a capacitance value of 50pF to 60pF per metre.

Connector Recommendations

Screened 9-way D-type connectors should be used, allowing a double cable entry to implement the connection scheme shown in Figure 5-25. Where resistive terminations are required to be made, re-wireable D-type connectors, designed to accept line terminating resistors, should be used.

Terminations

When the line requires termination, (line lengths greater than 100m) 100Ω ±1% quarter watt metal film resistors may be used. A convenient way of mounting these resistors is to incorporate them into a re-wireable D-type connector designed to accept termination resistors.

RS485 Naming Conventions

The signalling sense of voltages appearing across the line is:

MARK or OFF - When the A terminal of the generator is negative with respect to the B terminal for a binary 1

SPACE or ON - When the A terminal of the generator is positive with respect to the B terminal for a binary 0

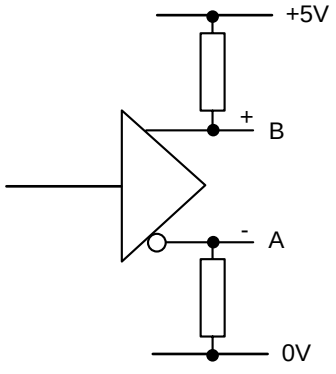


Figure 5-26. RS485 Naming Conventions

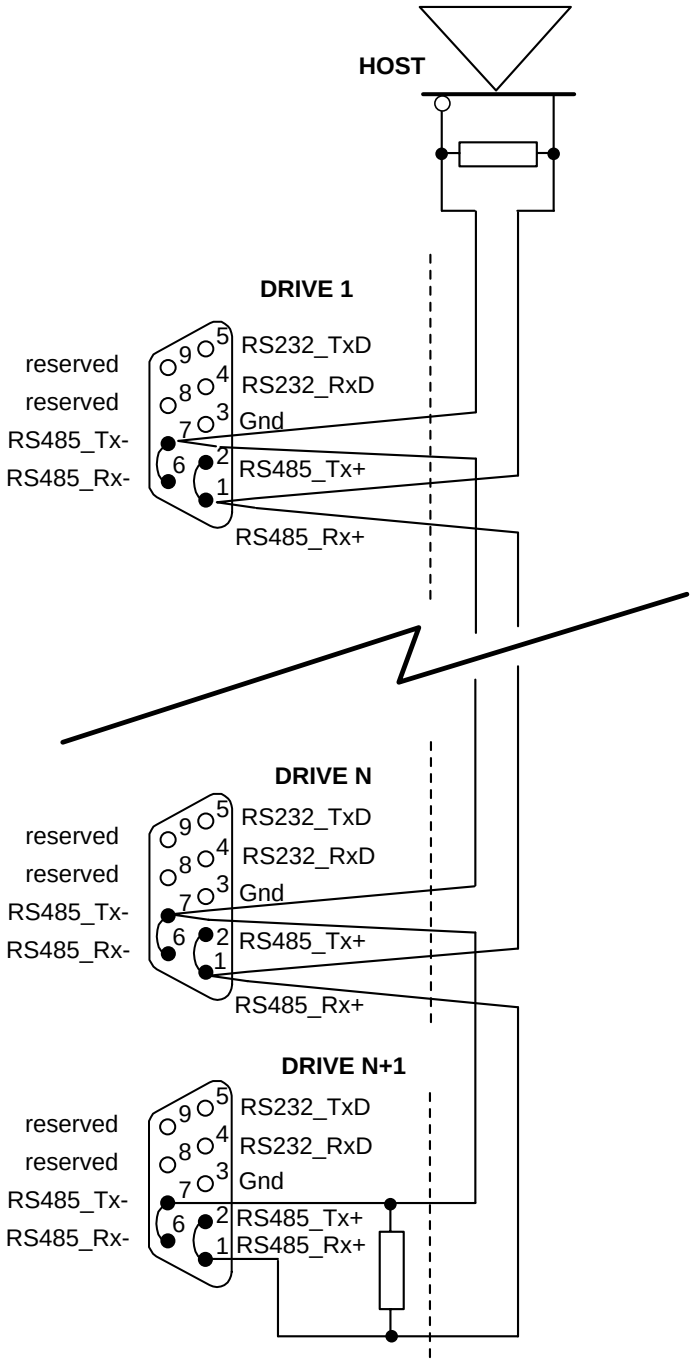


Figure 5-27. RS485 Muti-drop Connector Wiring

Fail Safe Biasing

When you have more than one RS485 transmitter connected to the same pair of wires (2-wire bus), and no transmission is taking place (idle state), all of the transmitters will be in high impedance mode. This will allow the bus to 'float' leaving the connected receivers susceptible to any electrical noise present on the bus. To reduce the effect of this noise pull-up and pull-down resistors can be added to the inverting and non-inverting inputs of the last receiver on the bus, as shown in Figure 5-28. Again a re-wireable D-type connector

designed to accept termination resistors will need to be used. Bias resistors can be 1200Ω ±1% quarter watt metal film types.

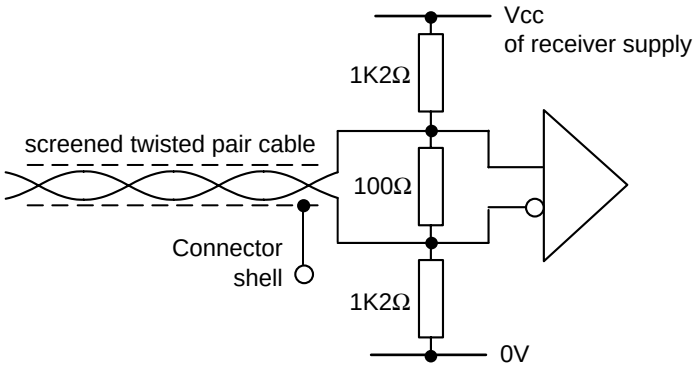


Figure 5-28 RS485 Fail Safe Biasing

Section 6. Command Reference

Command Description

Each command has been given a simple 1 to 7 character name, usually an abbreviation of its full descriptive title. Commands are listed in alphabetic order with any non alphabetic symbols appearing last.

Each individual description will include a one line header giving the abbreviated name followed by its full descriptive name. The following lines give the command's syntax, units of measurement, range of values, any default value and a reference to other related commands. Where commands contain a list of parameters a simple layout is used with only the syntax of the command being displayed.

Every command requires an address. Where several drives need to respond to a common set of global commands, prefix each command with the address 0. To prevent spurious feedback any report or read command using address 0 will be ignored. Note that any command missing an address prefix will also be ignored.

Where commands (such as TR, R, WR and IF) include a system variable it is treated as a command parameter. System variables are used to store internal drive values and settings, they can be read, tested and some may be written to, but they are all dedicated for a particular use by the system and cannot be used for storing user data within a program.

Command Syntax

Generally, a command's syntax consists of an address 'a' followed by the abbreviated command name either followed by parenthesis containing the command's parameters or simply the range parameter 'n'. Within the parenthesis form of command, each parameter is separated by a comma and any optional parameter is shown in italics.

Commands not requiring any parameter string have a syntax shown in Figure 6-1.

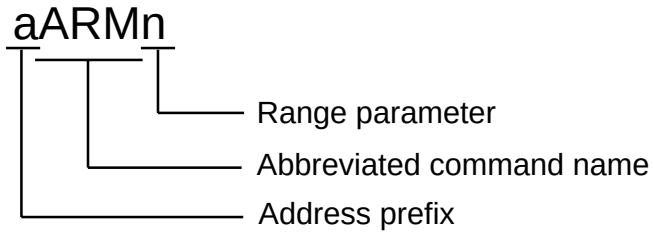


Figure 6-1. Simple Command Syntax

Commands which include a parameter string can be simple one parameter commands such as GOSUB or CLEAR where the single parameter is a label, or multi parameter commands containing a string of parameter values. Both styles of parameter commands are shown in Figure 6-2.

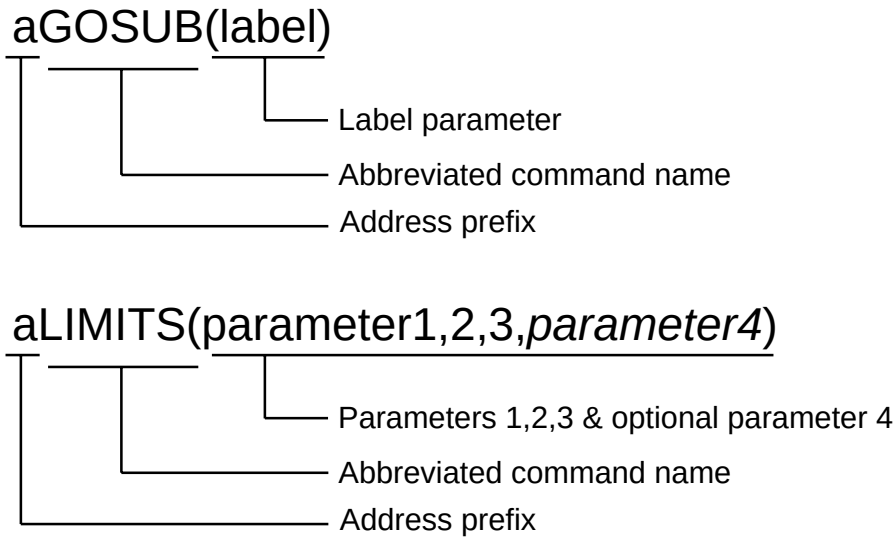


Figure 6-2. Parameter Commands

Attention

[1] Commands can only be terminated with a carriage return. A space is not valid.

[2] A command must not contain any space characters (apart from Quote " ").

[3] All commands are device specific, that is, they always need an address.

[4] For reasons of clarity, program examples appear as if being downloaded via EASI-TOOLS, that is they contain comments and line feeds etc.

Command Properties

Each command has a particular set of properties which govern the way the command can be used.

Commands can have the following properties:

Immediate only
Immediate or buffered
Can be used in labelled block
Can't be used in labelled block
Saved by SV
Not saved by SV

Immediate Only

Immediate only commands are:

C, KILL, S, R(RB), R(UF), R(DF) and R(ST)

These commands are acted upon by the indexer as soon as they are received.

Immediate or Buffered

Immediate or buffered commands are immediate unless command execution is being delayed or command queuing is enabled whilst moving, in which case the command is buffered. When command execution is being delayed, that is awaiting the results of a trigger command or waiting for a pause or time delay to finish, interrogation commands can be sent and get executed immediately, therefore such commands as 1IS, 1R(ST), 1R(UF), 1A etc. are correctly reported. However, if an action type of command is sent (such as G or 1A10) the action command just sent and any other interrogation commands get buffered and will not report back until the trigger, time delay or pause is finished. In this situation 'Immediate only' commands can be executed.

Example

```
1TR(IN,=,X0X0111111)  ;input trigger false
1R(MC)                ;read motor current
*50
1IS                   ;read input status
*0111100000
1A50                  ;send action command
1R(MR)                ;report commands are delayed until the trigger
                      ;command is complete
1IS
```

Can be used in labelled block

Means the command can be saved within a labelled program block. In this case the command will only be executed when that labelled block has been requested. Commands used within the definition of a labelled block are only saved when SV is sent.

Can't be used in labelled block

Means the command cannot be saved within a labelled program block.

Saved by SV

The current programmed value of a command may be saved by sending the SV command. The saved value will become the default value on power-up or following a Z command.

Not Saved by SV

The current programmed value of a command may not be saved by sending the SV command.

A Acceleration/Deceleration

Syntax	Units	Range of 'n'	Default	See also
aAn	Revs/sec^2	0.1 to 1023.9	10	AA AD

Description This command will set both the acceleration and deceleration rates of the motor to the same value. Values set for the **AA** and **AD** commands, if previously set, will be over written.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

To set the acceleration and deceleration rates of axis 1 to 120 rps^2, type **1A120**
To determine the acceleration of axis 1, type..... **1A**
The response is **120.0 120.0*
Overrange value..... **1A1050**
Will be reported as ..... **E* (meaning error)

Note In a positioning move the deceleration time is limited to 65 seconds. If a very low value for deceleration is used together with a high peak velocity such that the deceleration time is greater than 65 seconds, then an error will be reported when **G** is issued.

AA Acceleration

Syntax	Units	Range of 'n'	Default	See also
aAAn	Revs/sec^2	0.1 to 1023.9	10	A AD

Description The **AA** command will set or report the programmed linear acceleration rate of the motor. The acceleration value assigned to the **AA** command will be over written by any following **A** command.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

To set the acceleration and deceleration rates of axis 1 to 120 rps^2, type **1AA120**
To determine the acceleration of axis 1, type..... **1AA**
The response is **120.0*
Overrange value..... **1A1024**
Will be reported as ..... **E* (meaning error)

AD Deceleration

Syntax	Units	Range of 'n'	Default	See also
aADn	Revs/sec^2	0.1 to 1023.9	10	A AA

Description The **AD** command will set or report the programmed linear deceleration rate of the motor. The deceleration value assigned to the **AD** command will be over written by any following **A** command.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

To set the deceleration rate of axis 4 to 320 rps^2, type... **4AD320**
To report the current deceleration rate of axis 4, type..... **4AD**
The response is..... **320*
Overrange value..... **1A1040**
Will be reported as..... **E* (meaning error)

Note In a positioning move the deceleration time is limited to 65 seconds. If a very low value for deceleration is used together with a high peak velocity such that the deceleration time is greater than 65 seconds, then an error will be reported when **G** is issued.

ARM Enable event triggered code

Syntax	Units	Range of 'n & m'	Default for nm	See also
aARMnm	-	0 or 1	01	HOME LSEL POSMAIN REG STALL

The **ARM** command allows you to enable (arm) or disable (disarm) the following commands: **HOME, LSEL, POSMAIN, REG** & **STALL**

n=1 : arm
n=0 : disarm (default condition)

The second parameter 'm' is a fault switch that enables/disables the FAULT label from being run or not.

m=1 : fault label is enabled (default condition)
m=0 : fault label is disabled

All individually armed commands can be globally disarmed using only one command and can then be re-armed where required.

At power on, when saved and armed, the indexer will execute the code following the START: label (if defined).

The fault label parameter (fault switch), when enabled, will call the FAULT label when any one of the following conditions occur:

1. When driving further onto a limit, whilst the limit mode is set as stop on limit and the fault switch is enabled.*

2. When hitting a limit during a move, whilst the limit mode is set as stop on limit and the fault switch is enabled.*

3. Having a hardware drive fault with the fault switch enabled.

*Note: If, within the LIMIT command, the mode is set to '1' (stop when a limit is hit but continue the program) motion will be stopped at the programmed limit deceleration, no FAULT label will be called and the program will continue in a normal manner.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example The code following the START label will be run at power up:

```
1START:                          ;start label
1ON
1LIMITS(3,0,0)                   ;disable limits
1PROFILE1(100,100,4000,25,0)     ;define profile 1
1USE(1)                          ;use profile 1
1G                               ;execute profile 1
1END

1ARM1                            ;arm the start label
1SV                              ;save the code
```

Note If you save the indexer with **ARM0**, then the start-up sequence will fail to run, and the indexer will wait for serial commands.

Requesting an ARM status will report the state of all the commands that may be armed or disarmed, for example:

```
aARM
*ARM1
*HOME1
*LSEL1
*POSMAIN0
*REG0
*STALL0
*FAULT0
```

C Continue

Syntax	Units	Range of 'n'	Default	See also
aC	-	-	-	PS

Description The **C** (continue) command causes a user program to resume execution following a pause command. The command has no effect upon motion.

Properties Immediate only, can't be used in labelled block, not saved by SV

Example

```
1PS          ;pause commands
1A100        ;acceleration 100rps2
1V20         ;velocity 20rps
1G           ;go
1"TEST"      ;add TEST comment
1C           ;continue
TEST         ;message TEST is displayed
```

CLEAR Clear user code

Syntax	Units	Range of 'n'	Default	See also
aCLEAR(label)	-	-	-	-

Description The **CLEAR** command deletes user program instructions from the **label** specified until the END statement associated with that label. If a subroutine has been cleared, but its associated **GOSUB** command still exists, at run time the code will halt, motion will stop and *E will be reported.

Specifying the ALL keyword as the label will delete all user programs within the drive addressed.

Properties Immediate or buffered, can't be used in labelled block, not saved by SV

Example

```
0CLEAR(ALL)      ; Clear memory of anything defined so far (all drives)
5CLEAR(START)    ; deletes the user code within the START label, but
                 ; the declaration START is not cleared
```

Note Once a label has been declared it cannot be cleared. You can only clear declarations by using a **CLEAR(ALL)**.

Declare Declare

Syntax	Units	Range of 'n'	Default	See also
aDeclare(label)	-	-	-	See below

Description All labels, apart from START, REG, NOREG & FAULT need to be declared at the beginning of the program using a **DECLARE** command. You can declare up to 10 labels as well as the four system labels.

Labels can only be declared in the command line or inside the START label. If you wish to upload your program all declarations must be made within the START label.

If a label has been declared, but not defined, a run time error will be signalled when it is called by a **GOTO**, **GOSUB** or **LOOP** command.

When a label has been declared and defined, clearing it will only get rid of the definition, the declaration will still remain. Declarations can only be cleared using a CLEAR(ALL).

Typing **aDeclare** by itself will list the percentage of memory used by each label type.

Properties Immediate or buffered, can be used in labelled block, (but only within the START label), saved by SV.

Example

```
1DECLARE(CUT2)      ;declare label CUT2
1DECLARE
      *START 0.8%
      *REG 0.0%
      *NOREG 0.0%
      *FAULT 0.0%
      *CUT2 0.0%
```

D Distance

Syntax	Units	Range of 'n'	Default	See also
aDn	Steps	-2,147,483,648 to 2,147,483,647	4000	M

Description The **D** command will set or report the programmed move distance. The value programmed is only used for preset moves and is ignored in MC (Move Continuous). Depending upon the state of system variable DU, distance units can be specified or measured in motor steps (default) or encoder steps.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example To set the move distance of axis 2 to 15000 steps type.. **2D15000**
To report the current programmed move distance of axis 2, type..... **2D**
The controller responds with..... **15000*

If a value entered is out of range *E will be reported and the current value will not be altered.

Distance reports the current direction as influenced by the H command in MI (Mode Incremental) only. For example:

```
MI          ;mode incremental
1D4000      ;set distance to 4000 steps
1D          ;report distance
*4000       ;value reported
1H-         ;change direction
1D          ;report distance
*-4000      ;value reported
```

E Enable/Disable Communications

Syntax	Units	Range of 'n'	Default	See also
aEn	-	0 or 1	1	-

Description The **E** command disables or re-enables serial communications. E1 turns command communications ON, whilst E0 turns it OFF. In E0 mode the only command seen by the controller is E1.

Properties Immediate or buffered, can't be used in labelled block, saved by SV

Example An example of when to use this command is if global commands are being sent but one axis needs to ignore them whilst all others action them.

To enable axis 6 to accept commands over the RS232 serial link, type **6E1**

To disables communications using the RS232 serial link, type **6E0**

EXIT Exit from loop

Syntax	Units	Range of 'n'	Default	See also
aEXIT	-	-	-	Loop

Description The **EXIT** command is used to terminate a loop function instantly.

Properties Immediate or buffered, can be used in labelled block

Example

```
4START:                  ; signifies this is the power on sequence
4DECLARE(GRIP)
4PROFILE2(100,100,4000,25,0)  ; define profile 2
4LOOP(GRIP,0)            ; repeat the grip/ungrip code forever
4O(XX1XXXXX)             ; set output 3
4T5                      ; If we exit the loop then we pulse output
4O(XX0XXXXX)             ; 3 to indicate gripper tension out of tolerance
4END                     ;

4GRIP:
4USE(2)                  ; Use motion parameters from profile 2
4G                       ; do the move
4O(XXX1XXXX)             ; trigger the measurement
4H                       ; change to ungrip/grip
4T1                      ; delay for 1 sec
4IF(IN,=,XXX1XXXX)       ; gripper tension out of tolerance
4EXIT                    ; abort loop
4IF(IN,=,XXX0XXXX)       ; else continue the loop
4G
4END
```

Note In the example shown above, once the EXIT command is encountered the cycle in progress will be completed, then the code will be returned to the line immediately following the LOOP command. That is, output 3 is pulsed for 5 seconds to indicate the gripper tension is out of tolerance.

FRATE Feed Rate Override

Syntax	Units	Range of 'n'	Default	See also
aFRATEn	-	0 or 1	0	-

Description Feed Rate Override, the **FRATE** command, is used together with the analogue input to scale the peak velocity of the drive (V). *The purpose of the command is to allow the speed of the process being performed by the drive to be controlled by a single external analogue signal.*

The resolution of the control is 1% (0.1V) of the analogue input voltage range (10V = 512 counts = 100%). If the analogue voltage drops below a level equivalent to 1% of the target peak velocity (the velocity being requested by the analogue input voltage), the value used will be taken as 1% of the user set peak velocity (velocity set by the V command). This will be applied to values as low as 0.01rps, which is the minimum velocity allowed.

FRATE can only be used in modes continuous, absolute and incremental. A user fault is reported (drive not ready) if use in any other mode is attempted. The velocity capping of a move in modes MA or MI will take priority over FRATE. Note: voltage controlled modes B and U cannot be used because the same analogue input is required for external speed control.

Properties Immediate or buffered, can be used in labelled block

Example

```
1ON:          ; energise the drive
1V30          ; set the user peak velocity as 30rps
1MC           ; set mode continuous
1FRATE1       ; enable feed rate override
G             ; go
```

At this stage an analogue voltage (say 1.85V) is introduced to the differential analogue input. The actual velocity used is given by:

$$\frac{\text{rounded down (analogue input X 10) X user set peak velocity}}{100}$$

$$\frac{(1.85 \times 10) \times 30}{100} = \frac{(1.8 \times 10) \times 30}{100} = 5.40\text{rps}$$

Note In the example shown above, because of the rounding down, an input voltage of 1.89V would also give a speed of 5.40rps.

FRATE only works in unipolar mode (0 to +10V), any voltage with a negative polarity will be regarded as 0% of full scale value.

G Go

Syntax	Units	Range of 'n'	Default	See also
aG	-	-	-	PS S K VS

Description Issuing a **G** command starts motion using the parameters specified by the **V**, **AA**, **AD**, **D** and **VS** commands or via the **PROFILE** and **USE** commands. The mode of motion must have been previously set as this determines which parameters are used and which are ignored. For example, mode continuous will ignore the distance parameter.

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example

```
1PROFILE3(150,200,1500,25,1)  ;define profile 3
1USE(3)                       ;use profile 3
1G                            ;perform profile 3
```

Note If no motion occurs after **G** is issued, the cause can be determined by using the R(UF) command. Refer to the section on system parameters for more information.

GH Go Home

Syntax	Units	Range of 'n'	Default	See also
aGH	-	-	-	HOME S K

Description The go home command instructs the indexer to search for the home position as defined by the home input switch. For this command to function correctly, the home function must have been used to define the homing parameters.

For more information see the sub-section on homing in ***Section 5. Control of the PDFX***.

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example

```
1START:                 ;START label definition
1HOME1(+,1,-15,100,2)   ;define home
1GH                     ;go to datum position
1END
```

Note If no motion occurs after **GH** is issued, the cause can be determined by using the R(UF) command to report faults. Refer to the section on system parameters for more information.
Whilst going home, registration (if armed) will be disarmed and on successful completion of the **GH** routine the registration armed state will be restored.

GOSUB GO to SUBroutine

Syntax	Units	Range of 'n'	Default	See also
aGOSUB(label)	-	-	-	LOOP

Description The **GOSUB** command continues user program execution from the **label** specified and once the END statement is reached (in the called code), program execution returns to the calling routine.
GOSUBs can be nested to a maximum of 10 times, although the number of nestings will be decreased if used in combination with a **LOOP** command.

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example

```
1DECLARE(MOVE1)
1DECLARE(MOVE2)
1START:                                 ; code run after power on
1PROFILE1(360,360,400000,20,0.5)        ; define some move profiles
1PROFILE2(360,360,400000,45,0.5)
1GOSUB(MOVE1)                           ; go do move 1 and come back
1GOSUB(MOVE2)                           ; go do move 2 and come back
1O(11XXXXXX)                            ; set outputs 1 and 2
1END

1MOVE1:
1USE(1)                                 ; use the move profile 1
1G
1O(XXX1XXXX)                            ; turn output 4 on
1T0.1                                   ; wait for 100mS
1O(XXX0XXXX)                            ; turn output 4 off
1END

1MOVE2:
1USE(2)                                 ; use the move profile 2
1G
1T1                                     ; pause for settle time
1TR(IP,=,1)
1END
```

Note If you exceed the number of nesting levels the program will halt and return a **E*. R(UF) will return a 'Program nesting overflow' message.

If a **GOTO** command is used, the number of nesting levels is set to zero.

GOTO GO TO routine

Syntax	Units	Range of 'n'	Default	See also
aGOTO(label)	-	-	-	GOSUB

Description The **GOTO** command continues user program execution from the **label** specified.
Program execution does not return to the original place in the program (use **GOSUB** if the called function is required to return).

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example

```
2START:                                  ; code run after power on
2DECLARE(MOVE1)                          ; declare move 1
2DECLARE(MOVE2)                          ; declare move 2
2PROFILE1(360,360,400000,20,0.5)         ; define some move profiles
2PROFILE2(360,360,400000,45,0.5)
2GOTO(MOVE1)                             ; perform move 1
2END

2MOVE1:
2USE(1)                                  ; use the move profile 1
2G
2GOTO(MOVE2)
2END

2MOVE2:
2USE(2)                                  ; use the move profile 2
2G
2GOTO(MOVE1)
2END
```

Note If a **GOTO** command is used in conjunction with a **GOSUB** or **LOOP** command the number of nesting levels is set to zero.
The example shown above will give endless motion, only a FAULT condition or an immediate KILL or STOP command via comms. would stop this program.

H Change direction

Syntax	Units	Range of 'n'	Default	See also
aHn	-	+ - or blank	+	LOOP

Description The **H** command changes the direction of motion. Specifying H+ sets the direction to clockwise, H- counter clockwise, and H alone reverses the current direction.

This command has no effect in Mode Absolute and is ignored.

In Mode Continuous, only the **H** command can set the direction of motion as the **D** command is ignored for continuous moves.

If **H** is entered whilst the motor is moving, the direction will not change until the motor comes to a stop and another **G** command is given.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

See the example in the LOOP command description as well.

```
3DECLARE(MAIN)          ; declare main code
3OFF                    ; turn off to change motor parameters
3W(MR,4000)             ; motor resolution to 4000 steps
3ON
3MI                     ; mode incremental
3A20                    ; accel and decel to 20
3V15                    ; top speed of 15 rps
3D-8000                 ; 2 revs ccw
3G                      ; move
3H                      ; go cw next time
3G                      ; 2 revs cw
3START:
3PROFILE1(360,360,40000,20,0.5)  ; define profile 1
3MI                     ; mode incremental
3GOTO(MAIN)
3END

3MAIN:
3USE(1)          ; use profile 1 parameters
3G               ; do the move (CW)
3H               ; change direction (CCW)
3G               ; go back
3END             ; end of user program
```

Note

CAUTION

The USE command or the D command will re-define the move direction each time it is called, apart from when in mode continuous*. To set up a loop to go CW (clockwise) then CCW (counter clockwise), make sure the D or USE command is outside of the LOOP otherwise the direction will be the same each time around the loop.

*In mode continuous, moves are always made with the direction being decided by the H command.

HOME Configure Homing

Syntax **aHOMEon/off(reference_edge,home_type,direction_&_velocity, acceleration/deceleration,mode)**

Description The Configure Homing command is used to setup homing prior to the use of the **GH** (Go Home) command. Before homing can be used the home input must be turned **ON** (ON:1, OFF:0), the default setting is OFF.

The **reference edge** parameter is used to select the required edge of the home switch regarded as home (+ for positive edge or - for negative edge). Default is negative edge. See Figure - **Home Switch Operation**.

The **home_type** parameter is used to select the type of switch to be used for homing, the choice is:

Home switch normally closed 1

Home switch normally open 0 (default)

Velocity and direction determines the velocity at which homing is performed and the direction in which home is initially searched for (the default settings are + & 1rps). Whilst **acceleration/deceleration** sets the acceleration and deceleration rates used (default of 10r/s/s).

The **mode** parameter determines what happens when the specified edge of the home switch is encountered:

- mode 0 The indexer positions the motor in the active window of the switch (default setting).
- mode 1 The motor is positioned to the required edge of the switch + or -.
- mode 2 The indexer is set to seek the nearest drive zero phase position, to improve homing repeatability.
- mode 3 If an encoder with a Z channel is used then the indexer will seek the Z position after detecting the specified home switch edge.

When the home position is reached, the absolute position of the indexer is set to 0. (system variable PA). The incremental position reports the distance moved to reach the home position. (system variable PI).

Typing **aHOME** on its own will return the current parameter values for the nominated axis. For example:

The command **3HOME** will return:

** AR0 E- TP0 V+1.00 A10.0 M0*

Meaning the command is armed, reference edge is negative, home type

is 1, velocity is 10 rps positive, acceleration is 10 rev/s^2 and mode is 0.

Once configure homing has been setup, it can be applied (turned ON), or armed using the simplified form of command:

aHOME1

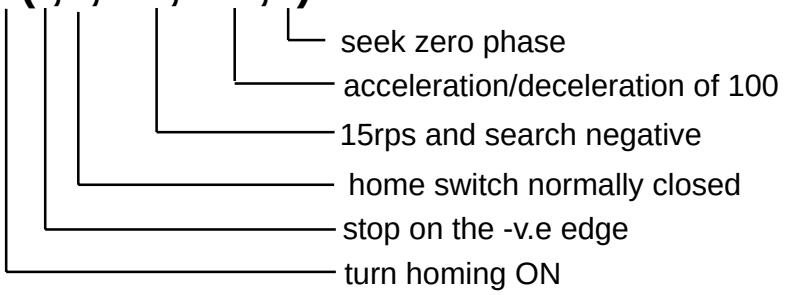
Or turned OFF using:

aHOME0

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example On axis 3 search for home in the negative direction at a velocity of 15 rps and acceleration/deceleration of 100 rps^2. The motor is to stop on the negative edge of home and then seek zero phase.

3HOME1(-,1,-15,100,2)



The go home final velocity of 0.1 rps is used to complete the last part of the move.

Note Also see **GH** command.

IF Test condition

Syntax **aIF(system_variable,relation,value)**

Description The **IF** command compares the specified **system variable** with the specified **value** using the specified **relation**. If the condition is met, the next line of code is executed otherwise it is skipped.

Refer to the table of system variables in ***Control of the PDFX***.

Valid relations for the comparison are:

=	Equals
<>	Does not equal
>	Greater than
<	Less than

Properties Immediate or buffered, can be used in labelled block, not saved by SV
Note: input can be accepted from the keyboard/terminal.

Example

```
2IF(PA,=,450)     ; if absolute indexer position = 450 steps on axis 2
2O(11000000)      ; set outputs 1 and 2, otherwise
2IF(PA,>,500)     ; if absolute indexer position > 500 steps
2O(00000001)      ; set output 8
```

Note If you wish to use the **IF** command during motion, command queuing (system variable CQ) must be set for continuous execution (CQ=0).

IS Input Status

Syntax	Units	Range of 'n'	Default	See also
aIS	-	-	-	-

Description The **IS** command reports the status of the configurable user inputs. The response is ten (0 or 1) digits corresponding to the following input bits:

Front Panel Pin Number	Bit	Function
2	1	User input 1
3	2	User input 2
4	3	User input 3
5	4	User input 4
6	5	User input 5
7	6	User input 6
8	7	Registration/ input 7
9	8	Home/ input 8
10	9	LIM- / input 9
11	10	LIM+ / input 10

0 represents a low condition on the input
1 represents a high condition on the input

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example To check the input status of axis 1, type **1IS**
The response is **0100000100*

User inputs 2 and home input 8 are high. All other inputs are low.

K Kill

Syntax	Units	Range of 'n'	Default	See also
aK	-	-	-	S PS

Description The **KILL** command is used to immediately stop motion. Once issued, it will cut the pulse train to the motor.

Carefully consider the use of this command in applications where a load with a large inertia may be required to stop quickly. By commanding K the motor could stall and lose torque. For this reason, a load with large inertia should be stopped mechanically to avoid overshoot of limit switches. A dump option may be required to protect the drive from over voltage trips. This command is used as an immediate stop and does not attempt to control deceleration.

For a controlled stop use the **S** (stop) command.

The **KILL** command cannot be used in a sequence its use is primarily for emergency situations.

K will also terminate any program execution and disarm the event triggered code. To re-arm the event triggered code after a kill command is issued, use the **ARM1** command.

Properties Immediate only

Example

```
1G      ;set drive in motion
1K      ;stop everything
```

Note The **K** command does require a device address or 0K to kill all axis. It will stop a time delay (T command) and will abort a program.

This command does not replace the requirement for an additional hardware device to cut power to the motor in the event of an emergency situation arising.

LIMITS Configure limit inputs

Syntax **aLIMITS(mask,type,mode,*LD*)**

Description The **LIMITS** command allows the user to define whether the LIM+/- inputs are being used as limit inputs or user inputs via the **LIM_MASK**.

The **mask** field takes the following values:

0	Enable limits (default setting)
1	Disable limit +
2	Disable limit -
3	Disable limit + & -

The +ve limit switch is the switch that is reached when the motor reaches the end of travel for a move with +ve distance/velocity.

The -ve limit switch is the switch that is reached when the motor reaches the end of travel for a move with -ve distance/velocity.

type field takes the following values:

1	Limits normally closed (default setting)
0	Limits normally open

mode field takes the following values:

0	Stop when a limit is hit and abort the program, then go to a predefined, and armed fault sequence, otherwise stop (default setting).
1	Stop when a limit is hit and continue the program. In certain applications this allows the limit switch to define a home position.

The optional ***LD*** parameter sets the required deceleration rate after hitting a limit, the default deceleration is 200 rps^2 : See deceleration command **AD** for range settings.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

```
3LIMITS(0,1,0,100)   ; +ve limit enabled, normally closed switch
                     ; -ve limit enabled, normally closed switch
                     ; user inputs 9 and 10 unavailable for use
                     : deceleration 100rps2
3LIMITS(1,1,0,100)   ; +ve limit disabled
                     ; user input 10 available for use
                     ; -ve limit enabled, normally closed switch
                     ; user input 9 unavailable for use
                     : deceleration 100rps2
```

To report the current configuration of the limits, type **3LIMITS**
The response, using the above example **LM1 TP1 M0 LD100.0*

See also **IS AD**

Notes The default **LIMIT** type value is 1, that is normally closed.

LIST List user program

Syntax **aLIST(*label*)**

Description The **LIST** command is used to view a user subroutine program in memory specified by the optional **label** parameter.
If the label specified is ALL, the entire user program is listed.
This command has no effect when the source/destination is fieldbus.

Properties Immediate or buffered, can't be used in labelled block, not saved by SV

Example Providing a START subroutine has been entered, typing 1LIST(START) or 1LIST(ALL) would produce the following :

1START:
1PROFILE1(360,360,400000,20,0.5)
1PROFILE2(360,360,400000,45,0.5)
1GOTO(MOVE1)
1END
1MOVE1:
1USE(1)
1G
1H
1END

Note Typing **1LIST(MOVE1)** will only produce the code following label MOVE1 as far as END. That is :

1MOVE1:
1USE(1)
1G
1H
1END

LOOP Repeat user code

Syntax **aLOOP(label,cycles)**

Description The **LOOP** command repeatedly calls a **labelled** block of code a number of times specified by the **cycles** parameter. Note: If the number of cycles is set to 0 the loop will continue indefinitely.

Nesting of loops up to 5 levels is permitted.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example Run the grip code for a mechanical elasticity tester 6 times, and delay for 1 second between each grip cycle to allow for a sensor to measure deflection.

```
2START:                 ; signifies this is the power on sequence
2DECLARE(GRIP)
2PROFILE2(150,200,4800,45,1)
2USE(2)                 ; Use motion parameters from profile 2
2LOOP(GRIP,6)           ; repeat the grip/ungrip code 6 times
2END                    ;

2GRIP:
2G                      ; do the move on axis 2
2O(XXX1XXXX)            ; signal grip cycle
2H                      ; change to ungrip/grip
2G                      ; do the move again
2T1                     ; delay for one second
2O(XXX0XXXX)            ; signal end grip cycle
2END
```

Note Also see the **EXIT**, **KILL** and **GOSUB** command.

If you use a **GOTO** command within a **LOOP**, it will stop program execution of the loop and the number of nesting levels will be set to zero.

If you exceed the number of nesting levels the program will halt and return a **E*. R(UF) will return a 'Program nesting overflow' message.

LSEL Label Select

Syntax **aLSELon/off(code,inputs,execution)**

Description The label select command allows the code following a label having the name Lnn (where nn is the detected input code) to be performed when a certain user input pattern is detected on a defined set of inputs. The code can be continuously repeated or may be re-triggered, depending upon the command's **execution** parameter setting.

For **LSEL** to work, the indexer must be globally armed (ARM1).

Properties Immediate or buffered, can be used in labelled block, saved by SV

Parameter	Range
On	1
Off	0 (default setting)
Code	0 BCD 1 binary (default setting)
Inputs	1 to 5
Execution	0 continuously repeated (default setting) 1 re-triggered

Example The main code configures the label select command to detect a binary code on 5 inputs (all high gives decimal 31), and if detected, to continuously run the code at label L31.

Use binary mode for PLC control and BCD for control via a thumwheel.

```
1START:
1DECLARE(L31)
1LSEL1(1,5,0)
1ARM1
.
1L31:
1A10
1V10
1G
1END
```

To check the current mode, type **1LSEL**

The response will be **AR1 B/D1 IN5 C/R0*

Note If the inputs remain high, the code following label 31 will run continuously. If the inactive execution mode was selected, all inputs would need to go to 00000 and then 11111 before running the code at label 31.
Inputs must be present for a time exceeding the scan time, see **SN**.

See also **ARM** and **SN**

M Mode

Syntax	Units	Range of 'n'	Default	See also
aMn	-	see below	-	-

Description The mode command sets up the mode of operation of the indexer.

The values of n are:

A - indexed move with absolute positioning
B - bidirectional voltage controlled move
C - continuous move
I - indexed move with incremental positioning
U - unidirectional voltage controlled move

Note: When in voltage controlled mode (uni- or bi-directional), the current velocity setting is equivalent to the maximum input voltage and the deadband in rps is taken from the start stop velocity parameter VS. The acceleration and deceleration values are taken from AA and AD respectively as for any other move.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example The code below sets up unidirectional voltage controlled velocity mode on axis 3, and uses acceleration and deceleration values of 200 rps^2. The maximum velocity of 25 rps is achieved when the analogue voltage is 10V.

```
3PROFILE1(200,200,0,25,1)
3USE(1)
3MU
3G
```

To check the current mode, type..... **3M**
The response will be..... **MU*

O Output

Syntax	Units	pattern	Default	See also
aO(pattern)	-	see below	00000000	IS

Description The **O** command applies the specified binary **pattern** to the user outputs.

Pattern takes the bit values 0, 1, X, where 0 is output off, 1 is output on and X represents an unchanged state.
Pattern is 8 bits in length in the order of outputs 1 to 8
Trailing 'X's are not required.

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example

```
2O(11001100)      ; sets outputs 1,2,5,6 ON all others OFF

2O(X0XXXXX1)      ; leaves outputs 1,3,4,5,6,7 as they were and
                  ; turns output 2 OFF, output 8 ON.

2O(1010)          ;turns ON outputs 1 & 3 and turns OFF 2 & 4. All
                  ;others outputs remain unchanged.
```

OFF Shutdown motor power

Syntax	Units	Range of 'n'	Default	See also
aOFF	-	-	OFF	ON

Description Issuing an **OFF** command de-energises the drive to shutdown the motor power. The indexer responds to move commands that are issued after an **OFF** with *E. If you check the fault variable UF, you will see the 'Drive disabled' bit set to indicate that the drive was de-energised when a move was attempted.
OFF reduces motor heating and allows manual positioning of the load, assuming the system mechanics allow this and it is safe to do so.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

```
1OFF    ;shut down motor power on axis 1
```

ON Turn ON motor power

Syntax	Units	Range of 'n'	Default	See also
aON	-	-	-	OFF

Description Issuing an **ON** command energises the drive.

The command allows execution of moves provided the motor is not on a limit.
Configuration system variables, such as MR, MC, MS are loaded into the drive's indexer when the **ON** command is issued. The current state of the drive fault registers will be cleared.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

```
1OFF          ;drive must be OFF for configuration
1W(MC,50)     ;configure motor current
1W(MR,400)    ;configure motor resolution
1ON           ;turn drive ON again
```

POSMAIN Position maintenance

Syntax **aPOSMAINon/off(dead_band_range,*output*,*settletime*)**

Description The **POSMAIN** command enables position maintenance, when in encoder distance mode (DU=1). The **on/off** parameter takes the values of 1 for ON and 0 for OFF. With position maintenance enabled, if the motor stops within the **dead band range** for the optionally specified settle time then the indexer will regard the move as having been completed. If the motor stops outside of the dead band range position maintenance will be performed to aim for the original target.

The optional parameter ***output*** specifies a user output that will be turned ON when the move is within the dead band window.
The optional parameter ***settletime*** specifies how long in milliseconds that the indexer will wait after motion has ceased, before checking the feedback encoder.

For **POSMAIN** to work, the indexer must be globally armed (ARM1).

Parameter ranges are:

Parameter	Range	Units
ON	1	
OFF	0 (default)	
Deadband Range	0 to 32767 (default=0)	Encoder counts
Output	1 to 8	
Settle time	0 to 65535	milliseconds

Note: dead band range is measured in encoder count units as a +/- band, consequently a dead band of 10 will be equivalent to ±10 or 20 units wide.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

```
1POSMAIN1(10,3)   ; Enable position maintenance. Allow
                  ; a 10 step dead band window and set o/p 3
                  ; when within the desired range
1POSMAIN1(20)     ; Enable position maintenance.  Allow a
                  ; 20 step dead band window.  Output not set.
```

Note Position maintenance is performed at the end of a move to correct any overall position error. You must select encoder mode of operation (DU=1) to use POSMAIN. See also **ARM**.

PROFILE Define move profile

Syntax **aPROFILEnumber(AA,AD,D,V,VS)**

Description The **PROFILE** command sets up a table of move profiles in the indexer memory. These profiles can be recalled at any time by the **USE** command.

The **PROFILE** command parameters are:

Acceleration	**AA**
Deceleration	**AD**
Distance	**D**
Velocity	**V**
VStartStop	**VS**

Ranges for the **AA**, **AD**, **D**, **V** and **VS** commands are as stated for each individual command.
The range of PROFILE number is 0 to 8, but PROFILE0 cannot be defined.

Use PROFILE0 to read the current profile settings. The format of the returned message will be:

*0AA___AD___D___V___VS___

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example Profile 1 is to represent a move of 1500 steps on axis 3 at a velocity of 25 rps and acceleration/deceleration of 200 rps^2 start stop speed of 1rps

3PROFILE1(200,200,1500,25,1)

Profile 2 is to represent a move of 4800 steps on axis 4 at a velocity of 45 rps, acceleration of 150 rps^2 deceleration of 200 rps^2, start stop speed of 1rps

3PROFILE2(150,200,4800,45,1)

The following move profiles will now be available in memory on axis 3:

Profile number	1	2
Acceleration	200	150
Deceleration	200	200
Distance	1500	4800
Velocity	25	45
Start/Stop velocity	1	1

The move parameters specified by Profile 2 may be recalled with the statement..... **4USE(2)**

Note A profile command will overwrite any individually programmed values of acceleration, deceleration, distance and velocity once the **USE** command is issued.

PS Pause

Syntax	Units	Range of 'n'	Default	See also
aPS	-	-	-	C

Description The **PS** (pause) command causes immediate command execution to cease until a continue (**C**) command is issued. The command is useful as a debug aid when testing trial code blocks.
The **PS** command cannot be used whilst running a program.

Properties Immediate or buffered, can't be used in labelled block, not saved by SV

Example

```
0PS       ;global pause
1D4000    ;setup axis 1
1V5       ;  .
1A50      ;  .
2D8000    ;setup axis 2
2V10      ;  .
2A100     ;  .
0G        ;global GO
0C        ;global continue
```

Note If too many commands are used after a pause, the command buffer will become full and will not respond to a continue command.

Typically, pause should only be used with around ten commands.

R Report system parameter

Syntax	Units	Range of 'n'	Default	See also
aR(system_variable)	**-**	**-**	**-**	**W**

Description The **R** command allows the user to read the specified **system variable**.

Refer to the table of system parameters for more information.

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Note: R(RB) is immediate only

Example **2W(MR,4000)** ;set variable MR (drive resolution) to be 4000
2R(MR) ;report the current value of variable MR
The response will be..... **4000*

REG Registration move

Syntax **aREGon/off(edge,*profile_number*,*hold_off_distance*,*registration window*)**

Description The **REG** command, once turned ON, defines a registration move. The move is actioned once motion has been started by issuing a **G** command. After a number of steps, determined by the optional **hold off distance**, the indexer will begin to search for a valid registration signal. If the specified **edge** is seen on the dedicated registration input a move is made using the optional **profile number** setting.

Execution of the user program then GOSUBs to the code immediately following the REG label.

If no mark is seen by the end of the move or the end of the optional ***registration window***, the user program GOSUBs to subroutine at the NOREG: label.

For REG to work, the indexer must be globally armed (ARM1). See **ARM**.

The ability to jump to different code segments dependent on detecting or not detecting the registration mark is a useful code feature. Confirmed registration can be used to turn ON or pulse an output, whilst lack of registration can be used to trigger an error routine.

Once registration has been setup, it can be applied (turned ON) using the simplified form of command:

aREG1 or turned OFF using: **aREG0**

Properties Immediate or buffered, can be used in labelled block, saved by SV

Parameter	Range	Units	Comments
On/Off	1 or 0 (default)		1 ON, 0 OFF
Edge	1 or 0 (default)		1 rising, 0 falling
Profile number	1 to 8		Must be user defined
Hold off distance	0 to 2147483647	steps	
Registration window	0 to 2147483647	steps	

If the registration move direction (as defined by the registration move profile) is different to the pre-registration move direction, the pre-registration move direction will be used.

Example

```
START:
2PROFILE1(10,10,40000,5,1)
2PROFILE2(20,20,20000,10,5)
2REG1(1,1,10000)
2USE(2)
2G
END

REG:
2O(XXXXXXX1)     ; Turn output 8 on : increment batch counter
2T0.5            ; Delay execution for 500mS
2O(XXXXXXX0)     ; Turn off output 8
END

NOREG:           ; if we come here we didn't have a valid reg mark
2O(XXXXXX1X)     ; Turn op7 on :push unlabelled product off conveyor

2T0.25           ; Delay execution for 250mS
2O(XXXXXX0X)     ; Turn off output 7
END
```

RFS Return to factory settings

Syntax	Units	Range of 'n'	Default	See also
aRFS	-	-	-	-

Description Issuing an **RFS** command initialises the indexer to factory default settings. The drive must be de-energised (OFF) for **RFS** to be executed.

The default settings are:

All labels cleared, all outputs set to logic low
SN (scan time) set to 50 milliseconds
V=1 rps
VS=0 rps
D=+4000
A & AD=10 rps^2
MI (Mode incremental)
ARM, HOME, LIMITS, LSEL, POSMAIN, REG, STALL all OFF

MC=50	CQ=1 (queuing enabled)
MR=4000	MV=0 (not moving)
MS=50	IP=1 (in position)
EI=2	PI=0
EO=0	PA=0
ER=800	EP=0

Properties Immediate or buffered, can't be used in labelled block, not saved by SV

Example

```
1LIMITS(3,0,0,900)      ;define limits
1RFS                    ;return to factory settings
1LIMITS                 ;report limits
*LM0 TP1 M0 AD 200
```

Note The address bit switch setting is read after performing an **RFS** command.

S Stop

Syntax	Units	Range of 'n'	Default	See also
aS	-	-	-	PS KILL

Description Use the **S** command to bring motion to a controlled stop. The command will use the current value of deceleration as specified by either the immediate **AD** command or the current profile being used.

The command may be used either from the command line or within a label.

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example

```
1G          ;start the move
1S          ;stop the move
```

SN Scan command

Syntax	Units	Range of 'n'	Default	See also
aSNn	milliseconds	1 to 1000	50ms	-

Description The **SN** command allows you to set an input debounce time in milliseconds when using the drive with a plc. This is the time the user input levels must remain unchanging before the new pattern is accepted. For use with a plc the **SN** command should be set to a time less than the on-time of the plc outputs.

Setting SN too short may make the drive vulnerable to electrical pulse noise on the drive's user input lines.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

```
1SN10       ;set scan time to 10ms
1SN         ;report value of scan time
*10
```

STALL Stall detect

Syntax **aSTALLon/off(error_window,stop,*output*)**

Description The **STALL** command is used to enable stall detect. The **on/off** parameter takes the values of 1 for ON and 0 for OFF and has a default value of OFF. If the error between motor and encoder exceeds the **error window**, then the drive will generate a stall fault. If the **stop** parameter is set to 1, then the motor will stop on stall else it will attempt to finish the move.

The optional parameter ***output*** specifies a user output that will be turned ON if the stall condition arises. The output (if specified) will be turned OFF when the next **G** (go), **GH** (go home), any **ON** command or **Z** (reset) command is received.

For **STALL** to work, the indexer must be globally armed (ARM1).

If stall detection is enabled with stop on stall disabled and distance units are being measured in encoder counts, then motion will continue at a slow rate until the required encoder count is detected.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Parameter	Range	Units
on/off	1 ON, 0 OFF (default)	
error window	0 to 16,777,215 (default=0)	motor or encoder steps
stop on stall	1 stop, 0 continue (default=0)	
output	1 to 8	

Example **1STALL1(100,1,6)** ; Stall detect on. Use 100 step error window
; Stop motion if stall detected and turn ON O/P 6

Notes See also **ARM**.

SV Save configuration

Syntax	Units	Range of 'n'	Default	See also
aSV	-	-	-	-

Description When the **SV** command is issued, the current indexer system variables and user programs are stored in non volatile memory.

This command will not operate whilst the motor is being commanded to move or whilst executing a program. There will be a delay of approximately one second before commands following a save configuration are executed. Any data saved, will be restored following the next power on cycle.

Properties Immediate or buffered, can't be used in labelled block, not saved by SV

Example

```
1RFS        ;return drive to factory settings
1A150       ;acceleration set to 150 rps2
1SV         ;save current settings
1Z          ;reset drive
1A          ;report current value of acceleration
*150.0
```

T Time delay

Syntax	Units	Range of 'n'	Default	See also
aTn	seconds	0.05 to 10	none	IF

Description The **T** command pauses program execution for the time specified by the **delay** parameter. Timing resolution is to within 50ms increments. Any time value specified within the range 0.05 to 10 seconds will be rounded down to the nearest 0.05 second increment. Any value programmed outside of this range will generate an error (**E out of range*).

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example

```
4T6         ; delay for 6 seconds
4T0.38      ; delay 0.35 seconds (rounded down)
```

TR Wait for trigger

Syntax **aTR(system_variable,relation,value)**

Description The **TR** command pauses program execution until the trigger condition is met.

The trigger condition is met if the **relation** between **system_variable** and **value** is true.

Valid relations for the comparison are:

=	Equals
<>	Does not equal
>	Greater than
<	Less than

Value can be a number or a string depending on the **system_variable** being tested. Refer to the system variables table for more information.

If event driven code is ARMED, then the trigger command will only stop sequential program execution. It does not stop event triggered code from being executed. To do this use the ARM0 command before a TR command.

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example **3TR(PA,>,2000)** ; wait for position absolute to be >2000 steps

3TR(IN,=,XXX11XXX) ; wait for user inputs 4 and 5 to be high

Notes If you wish to use the **TR** command during motion, command queuing (system variable CQ) must be set for continuous execution (CQ=0).

USE Use

Syntax	Units	Range of 'n'	Default	See also
aUSE(profile)	-	1 to 8	-	PROFILE

Description The **USE** command copies the pre-defined profile command to the current move parameters.

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example

```
1PROFILE1(200,20,1500,25,1)   ;define profile 1
1PROFILE2(150,200,4800,45,1)  ;define profile 2
1USE(2)                       ; use motion profile 2
```

V Velocity

Syntax	Units	Range of 'n'	Default	See also
aVn	rev/s	0.01 to 50.00	1	VS

Description Velocity command **V** sets or reports the programmed velocity of the motor.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

To set the velocity of axis 3 to 25 rps, type **3V25**
To report the current velocity of axis 3, type..... **3V**
The controller responds with..... **25.0*
No units are reported.

Note

[1] The maximum stepping rate is limited to 200,000 steps per second. With a maximum drive resolution of 4000 steps per second the maximum velocity is 50.0 rev/s.

[2] Programmed velocities less than the start-stop velocity VS will result in a constant-velocity move with no acceleration or deceleration.

Note

[3] In a positioning move the deceleration time is limited to 65 seconds. If a very low value for deceleration is used together with a high peak velocity such that the deceleration time is greater than 65 seconds, the drive will report an error **E* when the G command is given.

[4] A programmed value of velocity can be overwritten by a PROFILE command once the USE command has been issued, but subsequent values of velocity can be programmed to override the value in use.

VS Start stop velocity threshold

Syntax	Units	Range of 'n'	Default	See also
aVSn	rev/s	0.01 to 5.00	0	V

Description

The **VS** command sets or reports the velocity threshold value above which the motor starts to accelerate towards its final velocity set by V, or below which the motor will start and stop with no acceleration/deceleration.

Use of the VS command can shorten move times and avoid low speed resonance. VS must be set to a value within the pull-in speed of the motor. Note: Velocity V can be set below this threshold value for moves made at sub-threshold velocities. All moves made at a velocity less than VS will be made without any acceleration or deceleration time.

Properties

Immediate or buffered, can be used in labelled block, saved by SV

Example

To set the start-stop velocity threshold of axis 5 to 0.5 rps, type **5VS0.5**

To report the current start-stop velocity threshold of axis 5, type **5VS**

For a start-stop velocity threshold of 0.5 rps on axis 5, the controller responds with **0.50*
No units are reported.

W Write system variable

Syntax **aW(system_variable,value)**

Description The **W** command allows you to set a specified **system variable** to a particular **value**.
Refer to the table of system variables in ***Control of the PDFX*** for more information.

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example Set system variable MR (drive resolution) to 4000 **2W(MR,4000)**
Report the current value of system variable MR **2R(MR)**
The controller responds with..... **4000*

Note The drive will need to be de-energised before certain system variables such as MR, MC or MS can be written to.

See also R command.

Z Reset

Syntax	Units	Range of 'n'	Default	See also
aZ	-	-	-	-

Description The **Z** command resets the drive's indexer. This is similar to power cycling the indexer. Upon restart, the user program following the START: label will execute only if the ARM command = 1.
Any commands pending before the Z is issued will be terminated and any buffers and user stacks cleared.

Properties Immediate or buffered, can't be used in labelled block, not saved by SV

Example To reset all drives, type **0Z**

Set comms address remotely

Syntax	Units	Range of 'n'	Default	See also
a#n	-	0 to 255	0	-

Description This command (**#**) allows you to set the unit address via software rather than hardware. It will override the hardware address lines, allowing addresses up to 255 to be used. Upon receipt of the command, the indexer will send a #n+1 command along the daisy chain provided the echo mode is set.
To address a specific axis place the current address 'a' before the # symbol.

0#0 will return addressing to hardware (bit switch) mode for all axes, provided a bit pattern is set on the bit switches, otherwise the command is ignored.

Following an SV (save) command, the address will be stored in non-volatile memory and will be used until the indexer is re-addressed. To ensure that hardware addressing is used from power up after having used #n to reassign the unit's address, send 0#0 followed by SV. This stores 0 as the device address which tells the indexer to get its address from the bit switch every time it powers up.

Properties Immediate or buffered, can't be used in labelled block, saved by SV
Note: Software or hardware mode is saved.

Example If a system with axes 1,3,7,2 (in that order) is sent the command **#10**, the axes will become 10, 11, 12, 13.

This command is only accepted if the Echo bit switch is set to the RS232/Echo ON setting. This is because the automatic addressing only applies to an RS232 daisy chain. Such a daisy chain requires echoing to be enabled.

“ “ Quote command

Syntax	Units	Range of ‘n’	Default	See also
a“ “	-	-	-	-

Description Commands can be sent to other drives or messages sent to remote displays using QUOTE. Using the RS232 link, up to 10 ASCII characters can be used to transmit the required command or message in exactly the way in which it was entered. Only ASCII characters between decimal 32 (space character) and decimal 126 (tilde ‘~’ character) inclusive are allowed.

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example

```
1DECLARE(EXAMP)     ;declare label
1EXAMP:             ;label EXAMP
1”TEST”             ;quote TEST
1END                ;end label
1GOTO(EXAMP)        ;goto label EXAMP
TEST                ;output message
```

System Variables

Var	Name	R	W	Range/default value
BU	Buffer usage	Y	N	% of program buffer used 0-100
CQ	Command queuing	Y	Y	1= Pauses until move complete (default) 0= continuous execution
DF	Drive Fault status	Y	N	See following section
DU	Distance Units	Y	Y	0= Motor steps or 1= Encoder steps, default=0
EI	Encoder Input	Y	Y	0=step/dir, 1=cw/ccw, 2=quad ABZ, default=2
EO	Encoder signal output	Y	Y	0=step/dir, 1=cw/ccw, 2=quad ABZ, default=0
EP	Encoder Position	Y	Y	-2,147,483,648 to 2,147,483,647 counts
ER	Feedback encoder resolution	Y	Y	1-65,535
EX	Comms. Response Style & echo control	Y	Y	0= speak when spoken to, echo off, RS485,multi axis 1= speak whenever, echo off, RS485, single axis 2= speak when spoken to, echo on, RS232,multi axis 3= speak whenever, echo on, RS232, single axis
IN	Inputs	N	N	Same format as IS command
IP	In Position flag	Y	N	1= In position or 0= not yet in position
IT	In Position Time	Y	Y	1-500mS, default=10mS
MC	Motor current	Y	Y	50, 60, 70, 80, 90, 100 (in percent), default see RFS
MR	Motor resolution	Y	Y	400, 800, 2000, 4000, default see RFS
MS	Motor standby current	Y	Y	50, 70 (in percent) , default see RFS
MV	Moving	Y	N	Flag 1= moving or 0 = not moving
PA	Position Absolute	Y	Y	-2,147,483,648 to 2,147,483,647 steps/counts, default 0
PE	Position Error	Y	N	Encoder count or motor steps, depending on DU
PI	Position Incremental	Y	Y	-2,147,483,648 to 2,147,483,647 steps/counts, default 0
RB	Ready/Busy flag	Y	N	Flag 0= ready or 1= busy
RM	Registration Move	Y	N	Flag 1= reg move in progress, 0 = not doing reg move
RV	ReVision of software	Y	N	x.yy major.minor
SN	Serial number	Y	N	Drive serial number
ST	Status of indexing	Y	N	See Reporting the Status of Variables
UF	User program Fault status	Y	N	See Reporting the Status of Variables

Drive Faults

Bit Number	DF Information
1	Composite fault
2	Output stage over current
3	Supply rail failure
4	Ambient over temperature
5	Drive over temperature
6	Configuration error
7	Motor high voltage rail failure
8	Output fault
9 to 32	Reserved

Status Bits

Bit Number	Status Information
1	Command processing paused
2	Looping (command executing)
3	Wait for trigger (input)
4	Running program
5	Going home
6	Waiting for delay timeout
7	Registration in progress
9	Motor energised
11	Event triggered - active until trigger inputs are reset
12	Input in LSEL not matching label
13	-ve limit seen during last move
14	+ve limit seen during last move
17	Executing a position maintenance move
18	Possible stall
19	Moving (In motion)
20	Stationary (In position)
21	No registration signal seen in registration window
22	Cannot stop within the defined registration distance
23 to 32	Reserved

User Faults

Bit Number	UF Information
1	Value is out of range
2	Incorrect command syntax
3	Last label already in use
4	Label of this name not defined
5	Missing Z pulse when homing
6	Homing failed - no signal detected
7	Home signal too narrow
8	Drive de-energised
9	Cannot relate END statement to a label
10	Program memory buffer full
11	No more motion profiles available
12	No more sequence labels available
13	End of travel limit hit
14	Still moving
15	Deceleration error
16	Transmit buffer overflow*
17	User program nesting overflow
18	Reserved
19	Drive not ready
21	Boost duty cycle error
22	Save error
23 to 32	Reserved

Command List

Command	Description
A	Acceleration/Deceleration
AA	Acceleration
AD	Deceleration
ARM	Enable event triggered code
C	Continue
CLEAR	Clear user code
DECLARE	Declare
D	Distance
E	Enable/disable communications
EXIT	Exit from loop
FRATE	Feed Rate Override
G	Go
GH	Go home
GOSUB	Go to subroutine
GOTO	Go to routine
H	Change direction
HOME	Configure homing
IF	Test condition
IS	Input status
K	Kill
LIMITS	Configure limit inputs
LIST	List user program
LOOP	Repeat user code
LSEL	Label select
M	Mode
O	Output
OFF	Shutdown motor power
ON	Turn on motor power

Command	Description
POSMAIN	Position maintenance
PROFILE	Define move profile
P S	Pause
R	Report system parameter
REG	Registration move
RFS	Return to factory settings
S	Stop
S N	Scan command
STALL	Stall detect
S V	Save configuration
T	Time delay
TR	Wait for trigger
USE	Use
V	Velocity
V S	Start stop velocity Threshold
W	Write system variable
Z	Reset
#	Set comms address remotely
“ ”	Quote command

Section 7. Maintenance and Troubleshooting

Drive Maintenance

The PDFX Series of drives do not require any routine maintenance, although occasional checking of the following points is recommended.

Motor Inspection

Periodically check the motor to ensure that no bolts or couplings have become loose during operation, and check the motor cable or leads periodically for signs of wear. Do not make very tight bends or pull on the cable during normal operation. Check all cable connectors and the safety earth connection.

Drive Inspection

Check that the drive is clear of loose material and has a free flow of air through the ventilation slots. Check all drive connector jacking screws are firmly tightened and the motor screen connection is secure.

Troubleshooting

IMPORTANT NOTE - ensure that power is turned off before any connections are removed or changed. Wait until the high-voltage (top left) LED on the drive has gone off before unplugging it.

WARNING - Danger of electric shock

High voltages are present within the drive module and on the motor phase outputs. Do not dismantle the drive cover, there are no user-serviceable parts inside.

In the event of any system malfunction, the first thing to check is the status of the LEDs on the front of the drive.

Drive LED indications

If no drive LEDs are illuminated, verify that AC power is on. If it is, but there are still no drive LEDs on, the drive is faulty and needs to be returned for repair.

Power (top left) LED

In normal operation this LED is either green or orange. When **green** it indicates that the power supplies are healthy; **orange** shows both that power is healthy and that the drive is in its zero phase state. The LED should show orange when the drive is first powered up and no moves have been requested.

Drive Status (top centre) LED

In normal operation this LED should be **green**, showing that the drive is healthy and is energised, i.e. not in a shutdown condition.

When this LED is **orange**, the drive is in a shutdown or de-energised state but is otherwise healthy. In this condition there will be no holding torque from the motor and the drive will not respond to move commands. If this is the cause of the problem, verify that the drive has not been shut down by the OFF command, or saved in the shut-down state.

If the centre LED is **red** it indicates that the drive has shut down due to a fault condition. This may be due to any of the following causes:

1. There is a short circuit across two of the motor connections or between one of the motor connections and ground. Disconnect power, remove the motor connector and check the motor connections with a meter. This condition may also arise if a very low inductance motor is used - the preferred range is 0.8mH to 10mH.

2. The power input voltage is either too high or too low. Check that the AC supply is within the limits specified in the Hardware Reference Section.

3. The drive has shut down due to excess temperature rise. Check that the ventilation slots at the top and bottom of the housing are not obstructed and that there is free flow of air through the drive. Make sure that the drive is not mounted directly above a source of heat such as a power supply, and if possible check the ambient temperature in the vicinity of the drive casing. Allow the drive to cool down and re-check. If the motor has ample torque for the application, try reducing the motor current setting (by means of System Variable MC) which will reduce the drive dissipation.

Note: If a PDFX15 is being used at a low supply voltage and this problem occurs, see PDFX15 Power Derating below.

4. The external 24V DC supply has been connected the wrong way round. Check the polarity of the external supply - see the Installation Section.

5. There is an internal fault in the drive and it will need to be returned for repair.

Further information on drive fault analysis will be found later in this section.

Right most LED

This LED has no significance unless the optional Fieldbus module is fitted.

Further checks

If you can safely do so, check that there is holding torque from the motor. However, take care not to touch any part of the system if this could result in personal injury should unexpected motion occur - see the safety warning at the front of this user guide.

No holding torque from the motor

Disconnect power, remove the motor connector and check the motor connections for continuity across each phase. Check that the drive has not been shut down by an OFF command as detailed above.

Motor has holding torque but will not run

The most likely cause is lack of communication between the PC and the drive. This can be checked by attempting a Connect from the Terminal menu of EASI-TOOLS. If the connection fails with the following screen displayed:

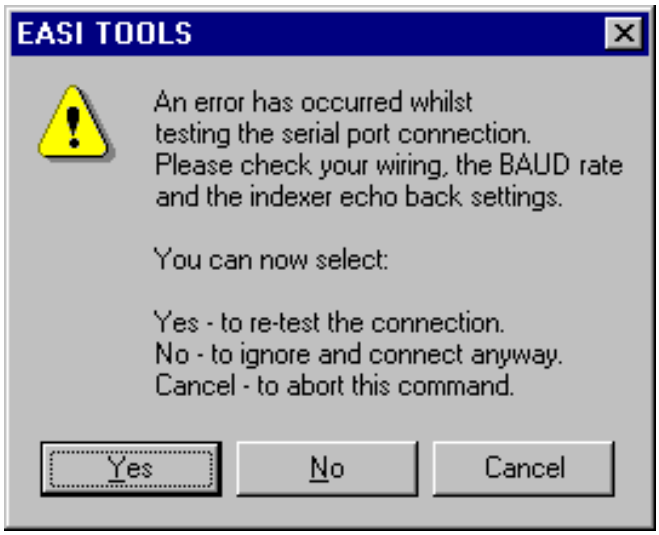


Figure 7-1. Communications Failure Error Message

Check the following:

Bit switch 1 on the front of the drive should be turned ON for RS232 communications and OFF for RS485. Echoback is normally enabled in RS232 mode, but may be disabled by system variable EX.

Bit switch 2 selects a baud rate of 9600 (switch 2 OFF) or 19200 (switch 2 ON). The selected baud rate must match that set up in EASI-Tools.

Ensure the serial port configuration is set correctly in EASI-TOOLS and the correct serial COM port has been selected. If you experience PC IRQ problems, read the note in the software installation section to allow re-allocation of ports.

Check that the drive address setting is correct - see the Installation Section.

If none of the above resolves the lack of communications, an RS232 communications link can be loop tested by removing the communications D-type plug where it connects to the drive and placing a short between lines 4 and 5. In this condition any command sent from the terminal window should be echoed back, confirming the integrity of the overall RS232 link. If this works, but you cannot communicate with the drive or if you have problems with the RS485 communications contact technical support for further advice.

Drive fault analysis

Drive or system problems indicated by the centre drive LED showing red may be analysed for further information as outlined below. The red LED indicates that a fault has caused the drive output stage to be turned OFF (de-energised). Once the fault has been corrected the drive may be re-energised using the ON command (unless the fault is the result of an internal failure).

Drive faults can be identified by performing a read of the system variable DF (Drive Fault) using the R report system parameter command. A list of drive faults is given in Section 5, but is also reproduced in Table 7-1 below:

Use the report command to display the DF word pattern, that is '**aR(DF)**', or use EASI-TOOLS Drive status report to help diagnose drive problems.

Bit Number	DF Information
1	Composite fault
2	Output stage over current
3	Supply rail failure
4	Ambient over temperature
5	Drive over temperature
6	Configuration error
7	Motor high voltage rail failure
8	Output fault
9 to 32	Reserved

Table 7-1. Drive Fault Bit Description

PDFX15 Power Derating

If you are using a PDFX15 at a reduced voltage rating (between 95 and 120V AC) and you experience drive overtemperature faults you may need to decrease the repition rate of your application to give the drive a greater cool down time between moves.

Common Faults Following a Go Command

Often made programming errors can lead to an error being reported following a G command, the most likely are listed below:

[1] LIMIT switch error, the default is limit switch normally closed. To disable LIMITS use a LIMIT mask of 3.

[2] Drive not energised, issue an ON command to turn ON motor power.

[3] In a positioning move the deceleration time is limited to 65 seconds. If a very low value for deceleration is used together with a high peak velocity such that the deceleration time is greater than 65 seconds, the drive will report an error **E* when the G command is given.

Returning the System

Contact the Parker Automation Technology Centre or the machinery manufacturer who supplied the product. Equipment for repair should NOT be returned directly to Digiplan without prior authorisation. Repairs will be carried out by Digiplan but will be processed via your supplier.

Digiplan may at their discretion authorise direct shipment to and from Poole or Rohnert Park, but only by prior arrangement with your supplier. Existing UK and USA customers who purchase equipment directly from Digiplan should contact Poole or Rohnert Park for further information (contact numbers are at the front of this User Guide).

Index

#

set address remotely, 134

A

A acceleration deceleration, 94
AA acceleration, 94
Absolute preset moves, 62
AC supply for test, 5
AD deceleration, 95
Address
  bit switch, 32
  echo bit switch setting, 33
  incrementing, 33
  saving, 33
  selection hardware, 32
  selection table, 33
  setting sequential patterns, 33
  software maximum, 34
  software selection, 33
  software selection limitations, 34
  using the hash command, 34
Adjustments, 1
Analogue control specification, 38
Analogue input, 76
Analogue input circuit, 76
Analogue remote connection, 77
Arm command, 50, 52
ARM event code, 96

B

Baud rate bit switch, 33
Baud rate selection, 11
Bipolar input, 76
Bipolar rating, 24
Bit switch, 2
Bit switch test settings, 5
Bus modules, 34
Bus voltage, 37

C

C continue, 98
Cabinet installation, 22
CAN bus connections, 34
CANbus, 34
Checking drive type, 3
Checking the drive, 4
Chop frequency, 37
Clean earth, 22
CLEAR user code, 98
Clockwise motion, 62
Closed loop applications, 80
Closed loop operation, 80
Code
  structure, 45
Command
  address requirements, 91
  ARM, 52
  label & multi parameter syntax, 92
  presentation, 91
  reference, 91
  simple syntax, 91
Command defaults, 126
Command list, 139
Command properties, 92
Command queuing, 61
Command requirements, 92
Commands sent
  waiting on a trigger, 59
RS485, 86
Communications specification, 38
Computer control
  delay code time, 44
  download time, 44
  pause and continue, 44
  using handshake, 44
Conditional code, 60
Conditional code example, 61
Conditional code table, 60
Configuration
  default values, 2
Configuration button settings, 13
Confirming drive operation, 18
Continuous moves, 62
Cooling requirements, 19
Counts per rev, 81
Current adjustment features, 37

D

D distance, 100
dead band, 63

Declarations
position within the code, 45
Declare, 99
command, 45
examples, 45
Dedicated I/O, 29
Dedicated inputs, 29
Default command and system variable settings, 126
Default directory, 7
Device addressing, 48
DF bit order, 56
DF word, 56
Digital inputs specification, 38
Digital outputs specification, 39
Direct computer control, 43
Direct mode, 43
Disconnecting device, 20
Drive
configuration, 2
cooling, 19
fault reporting, 57, 144
identification, 3
LED, 2
Drive Dimensions, 20
Drive fault
byte reporting, 57
Drive faults, 56
Drive faults list, 137
Drive inspection, 141
Drive status LED faults, 141
Dump version of drive, 19

E

E enable/disable comms, 101
Earth arrangements, 22
EASI-TOOLS, 3
axis configuration, 12
change axis address, 12
communicating with drive, 11
current configuration, 14
downloading, 16
drive configuration, 12
encoder configuration, 15
help, 17
installation directory, 8
limits configuration, 14
menu overview, 9
other configuration, 15
prg files, 11
resolution configuration, 14
running, 8
startup, 9
status reporting, 15, 16
uninstall, 8
uploading, 16
Echo bit switch, 33
Edit menu, 9
EI system variable, 31
Electrostatic precautions, 19
EMC
filter type, 23
installation, 24
EMC filter
dimensions, 24
Encoder
connector, 29
input configuration, 31
output configuration, 30
position maintenance, 29
power requirements, 30
stall detection, 29
supply, 30
Encoder input specification, 38
Encoder resolution, 80
Encoder setup, 80
Environment specification, 40
EO system variable, 30
Event
trigger, 53
Event code example, 52
Event driven code, 52
EXIT from loop, 102
External 24V supply, 23

F

Fanout of outputs, 79
FAULT, 49
Fault analysis, 144
Fault label, 50
call conditions, 50
conditions of execution, 50
example, 51
non call conditions, 50
table, 51
Fault status reporting, 56

Faults following G, 144
Field bus input connector, 34
Field bus interface, 34
Field bus output connector, 35
Fieldbus
  LED, 2
Fieldbus inputs, 34
Fieldbus outputs, 35
File menu, 9
Filter (EMC) installation, 23
Fit kit, 3
Front panel layout, 21
Function indicators, 1
Functional specification, 37
Fuse
  dump, 23
  mains, 23
  motor, 23
  physical size, 23
  type, 23
  values, 23
Fuse ratings, 39

G

Getting started, 3
GH go home, 104
GO, 104
Go home command, 72
Go home while in the home switch, 73
GOSUB go to subroutine, 105
Goto line number box, 10
GOTO routine, 106

H

H change direction, 107
Help menu, 10
Holding torque checks, 142
HOME, 109
  acceleration, 72
  approach speed, 72
  configuration, 71
  definition of terms, 70
  direction of travel, 74
  mode 2, 71
  mode 3, 71
  mode selection, 71
  modes, 71
  switch considerations, 70
Homing
  operations, 70
Humidity, 40

I

I/O pin description, 29
IF command, 60
IF test, 111
Immediate commands, 92
Immediate or buffered commands, 93
Immediate properties, 92
Incremental preset moves, 62
Indexer
  LED, 2
Indexer specification, 38
Input events, 52
Inputs & outputs, 1
Installation, 19
Installation safety requirements, 19
Installation setup program, 7
Installation time for S/W, 7
Interrogation commands, 59
IRQ problems, 11
IS input status, 112

K

KILL, 113

L

Label
  definition, 46
  execution, 46
  naming, 48
  number available, 46
  predefined list, 46
  select command, 46
  specification, 48
  system, 49
Labelled block properties, 93
Large motors, 25
LED indicators, 2
LIMIT changes for stop, 75
Limit inputs, 29
Limit switch wiring, 75
Limit switches, 75
LIMITS, 114
Line count, 81
LIST user program, 115

Logic input circuit, 77
Logic input circuit arrangements, 78
Logic inputs, 77
Logic output circuit, 78
Logic outputs, 78
Loop command, 49
LOOP user code, 116
LSEL command, 46
LSEL example, 52
LVD requirements, 19

M

M indexer mode, 118
Mains voltage input range, 23
Mains wire size, 23
Maintenance, 141
MB, 62
MB mode, 63
MC default value, 2
MODE command, 62
Monitor type, 4
Motion
  profiles, 66
Motor
  cable lengths, 25
  connections, 25, 26
  default settings, 26
  inductance values, 24
  large size, 25
  minimum inductance values, 24
  safety earth connection, 26
  selection, 24
  system variables, 26
  wire size, 25
Motor current check, 5
Motor direction note, 64
Motor inductance check, 4
Motor inspection, 141
Motor test connections, 6
Motors
  4 lead, 24
  6 lead, 24
  8 lead, 24
  optimum types, 25
  voltage rating, 25
Mouse, 4
Move
  types, 62
Moves
  absolute preset, 62
  continuous, 62
  incremental preset, 62
  preset, 62
  voltage controlled, 62
MR default value, 2
MS default value, 2
MU, 62
MU mode, 63

N

NOREG, 49
NOREG label, 68

O

O output, 119
OFF shutdown motor, 119
ON turn on motor, 120
Output current capacity, 78
Output relay drive circuit, 79
Output used to supply 24V, 23
Over temperature faults, 42

P

PA system variable, 71
Parallel connections, 26
PC requirements, 4
PDFX15 derating curve, 42
PDFX15 reduced voltage operation, 144
Current, 23
Point to point positioning, 67
Port configuration, 11
Position Maintenance
  correction velocity, 82
  dead band, 82
  mode limitations, 81
  output, 82
  settle time, 82
POSMAIN position maintenance, 121
Post quadrature resolution, 81
Power connections, 23
Power derating, 42
Power dump, 1, 37
Power dump reference, 1
Power dump version, 19
Power LED faults, 141
Power specification, 39

Pre installation checks, 5
Pre-installation checks, 6
Preset moves, 62
Primary encoder connector, 29
Product
  description, 1
  features, 1
  types, 1
Product features, 1
PROFILE of a move, 122
Program
  examples, 47
Program size, 4
Program structure, 47
Programmed mode, 45
Properties
  immediate, 92
  labelled block, 93
  save, 93
Properties of commands, 92
Protection circuits, 1
Protection class, 40
Protection features, 37
PS pause, 123

Q

Quote command, 135

R

R report system parameter, 124
Reduced supply operation, 42
REG, 49
REG registration move, 124
Registration, 68
  hold off distance, 68
  setup, 69
  window, 68
Registration example, 69
Report commands that can be saved, 59
Returning the system, 145
RFS return to factory settings, 126
RS232
  cable requirements, 31
  communications, 84
  connection details, 84
  daisy chain connections, 85
RS232/485 connector, 31
RS232/RS485 connections, 32
RS485
  2-wire half duplex, 86
  4-wire simplex, 86
  cable requirements, 31
  communication options, 86
  connector recommendations, 87
  fail safe biasing, 89
  fail safe biasing circuit, 89
  line termination, 87
  multi drop, 86
  multi drop circuit, 86
  naming conventions, 87
  signal polarity, 87
  transmission line considerations, 87
  wire recommendations, 87

S

S stop, 127
Save properties, 93
Screened connections, 19
Search menu, 9
Serial cables, 31
Serial comms connector, 31
Serial communications, 84
Serial communications configuration, 11
Serial link 25 way connector, 4
Serial link lead, 4
Series connections, 26
Setup file, 7
Shipping list, 3
Simple interface considerations, 79
SN scan, 127
SN serial number format, 55
Software
  identification, 3
Software file size, 6
Software installation, 7
Software requirements, 4
ST bit order, 55
ST word, 55
Stall
  error window, 83
  output, 83
  stop on, 83
STALL detect, 128
Stall detection, 83
Star point, 22
START, 49

START label, 52
Status Bit description, 55
Status bits list, 137
Status reporting
  immediate, 59
Status variable
  byte reporting, 56
Status variable reporting, 55
Step direction inputs, 31
Step direction outputs, 30
Step rate, 37
Step up step down inputs, 31
Step up step down outputs, 30
Stop input, 75
Supply frequency, 39
Supply volts, 39
SV save configuration, 129
System labels, 49
System variables, 53, 91
  reading, 53
  reporting status, 55
  table of, 53
  testing, 53
  writing, 53
System variables default settings, 126
System variables list, 136

T

T time delay, 129
Temperature
  ambient, 40
  over faults, 42
Temperature rise problems, 142
Terminal menu, 10
Test code, 18
Test connections, 5
Top panel layout, 21
Torque speed curves, 41
TR command, 60
TR wait for trigger, 130
Trapezoidal profile, 66
Trapezoidal profile modified, 67
Triangular profile, 66
Tri-coloured LEDs, 2

U

UF byte, 57
Unipolar input, 76
USE, 131
USE command, 53
User fault
  byte reporting, 58
  clear conditions, 58
  test example, 58
User fault descriptions, 57
User fault reporting, 57
User faults, 56
User faults list, 138
User Guide
  identification, 3
  issue number, 3
User I/O configuration, 76
User I/O connector, 29
User outputs, 78
Utilities menu, 10

V

V velocity, 131
Voltage controlled moves, 62
VS start stop velocity threshold, 132

W

W write system variable, 133
Weight, 40
Welcome box, 7
Windows menu, 10
Windows™, 4
Wire size of motor earth, 26

Z

Z reset, 133

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

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

ARTISAN® TECHNOLOGY GROUP