

## Section 4. HARDWARE REFERENCE

### PDHX15-E Drive Specifications

Parameter	Value
<b>Amplifiers</b>	
Type	MOSFET Chopper
Motor resolution	400, 1000, 2000 and 4000 steps/rev (user-selectable)
Short circuit protection	Phase-to-phase, across phases and phase to ground
Peak output current	5A/phase, switch reduceable
Standby current reduction	To 80% or 50% of programmed peak value after 100ms (switch-selected)
Nominal chopping frequency	20kHz
<b>AC Power input</b>	
Drive supply voltage	95V to 264V AC (absolute limits)
Supply frequency range	47 to 63Hz
Power factor	Better than 0.9 over input voltage range and output power range
Maximum input power	300VA
Input current:	3A rms max
Recommended supply protection	3A MCB type 'C' characteristics
<b>Output current range</b>	2.5A - 5A (350mA steps)
Standby reduction	50% or 80%
<b>Environmental</b>	
Drive dimensions	Height 292mm, Width 75mm , Depth 224mm
Weight	2.9Kg
Operating temperature range	0° - 40°C (32° - 104°F), or 50°C (122°F)if no user access to case
Ingress protection	IP20
Max. power dissipation of drive unit	
PDHX15E	30 Watts
PDHX15E-D	45 Watts
<b>Motors</b>	
Type	2-Phase hybrid or permanent magnet (normally 1.8°)
Number of leads	4, 6, or 8 (5 lead not suitable)
Minimum Motor Inductance	1mH
Optimum Inductance range	1mH-10mH
Power/Motor connection	refer to <b>Installation</b>

## Controller Specification

Parameter	Value
Command Input	RS232C (3-wire) or RS485 (2-wire, single-ended or 4-wire, differential)
RS232 Type	3-wire (Tx, Rx, Gnd). Minimum voltage swing = $\pm 3V$
Parameters	9600 baud, 8 data bits, 1 stop bit, no parity
Connector	8-way mini DIN or 9-way D-type
Configuration	Up to 32 interfaces can be controlled from a single RS232C port. Device address set up by DIL switch
RS485/422 Type	2-wire (single ended) or 4-wire (differential)
Parameters	9600 baud, 8 data bits, 1 stop bit, no parity
Connector	9-way D-type
Configuration	Up to 32 interfaces can be controlled from a single RS485/422 port. Device address set up by DIL switch
<b>Operating Ranges</b>	
Position	$\pm 1$ to 268,435,455 steps
Velocity	0.0001 to 200 revs/sec
Acceleration	0.06 to 999,999 revs/sec <sup>2</sup>
Maximum Encoder Frequency	100 kHz (lines/sec before multiplication)

<b>Resolution Ranges</b>	
Feedback encoder	1 to 32,767 counts/rev
User-programmed	1 to 32,767 steps/rev
Co-ordinate System	Incremental or absolute
Operating Modes	Preset, preset with speed change, continuous, registration, electronic gearbox following, preset following
<b>Motion Program Storage</b>	
Memory Type	Battery-backed RAM
Memory Capacity	8000 characters total
Number of Programs	64
Program Length	Variable up to memory limit
Program Selection	a) Via RS232C/RS485 b) Automatic execution at power up c) Binary address on sequence select inputs d) RP240 e) Thumbwheel (TM8) f) PLC
Opto-isolated I/P's	Home, Limits, Aux-in, Stop, 10 user-definable inputs (also used for program selection)
Optically-isolated O/P's	6 programmable
Type of output	NPN or PNP (software selectable)

**Table 4-1. Controller Specification**

## Fuses

PDHXE drives are fitted with fuses which limit circuit damage in the event of a fault occurring, they are not user replaceable. If the drive fails to operate correctly or you suspect a fuse has blown return the drive for repair. See ***Returning The System*** in the ***Maintenance and Troubleshooting*** section. Warranty is void if the case is opened.



## Section 5. BASIC MOTION CONTROL CONCEPTS

---

### Chapter Objectives

This chapter is preliminary to the programming section and describes some of the basic concepts of motion control systems in general and the PDHX-E in particular.

### Motion Profiles

In any motion control application the most important requirement is precise shaft rotation, whether it be with respect to position, time or velocity. The type of motion profile needed will depend upon the motion control requirement. The following sections describe the basic types of motion profiles.

#### Preset Moves

A preset move referred to in this manual is a move of a specified distance (in user steps). Preset moves allow the user to position in relation to the motor's previous stopped position (incremental moves) or in relation to a defined zero reference position (absolute moves). Preset moves are selected by putting the positioner into incremental mode using the MPI command and absolute moves are made using the MPA command.

#### Incremental Preset Moves

If the positioner is in the incremental mode (MPI command), a preset move will move the shaft of the motor the specified distance (in user steps) from its starting position. For example, to move the motor shaft 1.5 revolutions, a preset move with a distance of +6000 steps would be specified, assuming a 4000 step per rev encoder resolution setup. Every time this move is executed, the motor shaft will move 1.5 revolutions positive from its current position. The direction of the move can be specified at the same time as the distance by using the optional sign (D+6000 or D-6000), or it can be defined separately with the H command (H+ or H-).

#### Absolute Preset Moves

A preset move in absolute mode (MPA command) will move the shaft of the motor the specified distance (in user steps) from the absolute zero position. The absolute position can be set to zero with the PZ or SP commands, for instance at the end of a GO HOME move (GH command). The absolute zero position is initially the power-up position, and will remain that way until changed with a PZ command. Any preset move performed while in the absolute mode will position the motor shaft the defined distance (in user steps) from the absolute zero position. For example, with the positioner at the absolute zero position a move with a distance of +4000 will cause the motor shaft to turn 1 revolution in the positive direction. If a move with

the same defined distance is executed immediately after this move, the motor shaft will not turn, since it is already +4000 steps from the absolute zero position.

The direction of an absolute preset move will depend upon the shaft position at the beginning of the move and the position that it is being commanded to move to. For example, if the motor shaft is at absolute position +12,800, and position commanded is +5000, the motor shaft will move in the negative direction a distance of 7800 steps to absolute position +5000.

The positioner saves the mode that it was in at power down and powers up again in the same mode. Issuing the MPA command will set the mode to absolute. Issuing the MPI command will switch the mode from absolute to incremental. The positioner retains the absolute position referenced, even while in incremental mode. However, the counter does have an upper limit just slightly more than 268,400,000 counts.

### **Continuous Moves**

Continuous moves (MC command) cause the motor to accelerate and attain the specified velocity and then continue to move. To change velocity while the motor is moving use the V command. Issuing a stop (the S command) will cause the motor to decelerate to a stop at the last defined acceleration rate. The distance parameter is not used, although it is saved in case the mode is changed back to preset.

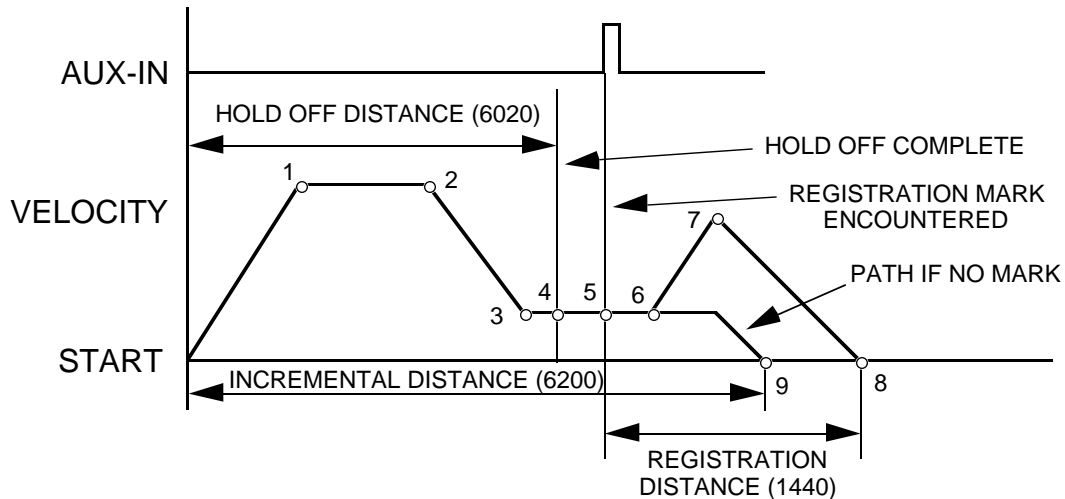
This mode is useful for applications which require constant spinning of the load, when the motor must stop after a period of time has elapsed rather than after a fixed distance, or when the motor must be synchronised to external events such as trigger input signals.

### **Speed Change Mode**

The MQ command allows speed changes to be made during a preset move. The speed changes can be triggered at a set distance using the TRD command or by an input using the TRE command.

### **Registration Moves**

A move may be programmed to end a specified distance after a registration pulse appears at AUX-IN. The mode is selected using the TRR command with the required registration distance in the MQ or MC modes. Its use is easiest to understand in the context of an example registration move:



**Figure 5-1. Example Registration Move**

**Example Program**

<b>MQ</b>	Select MQ mode
<b>D6200</b>	Set move distance to 6200 steps
<b>A100</b>	Set all accelerations to 100 revs/sec <sup>2</sup>
<b>V10</b>	Set velocity to 10 revs/sec
<b>G</b>	Go
<b>TRD4000</b>	At distance = 4000 steps
<b>V1</b>	Change velocity to 1 rev/sec
<b>TRD6020</b>	At distance 6020 look for registration pulse
<b>TRR1440</b>	Travel 1440 steps from registration pulse
<b>V50</b>	At a velocity of 50 revs/sec
<b>TRIP</b>	Set defined Output when in position

**Program Operation**

When executing this program, the axis will first accelerate at 100 steps/rev<sup>2</sup> to 10 revs/sec (1) and when it reaches a distance of 4000 steps (2), its speed will be reduced to 1 rev/sec (3) whilst it is waiting for the registration pulse. Shortly after the pulse is received (6), the axis will accelerate towards 50 revs/sec, but will not reach the speed before starting to decelerate (7) to stop at 1440 steps from where the pulse occurred (8).

If the pulse was not received after 6020 steps (4 - the hold off distance) and before 6200 steps (the incremental distance), the axis will stop at 6200 steps (9). This sets a window of 180 steps during which the pulse is expected.

### Program Criteria

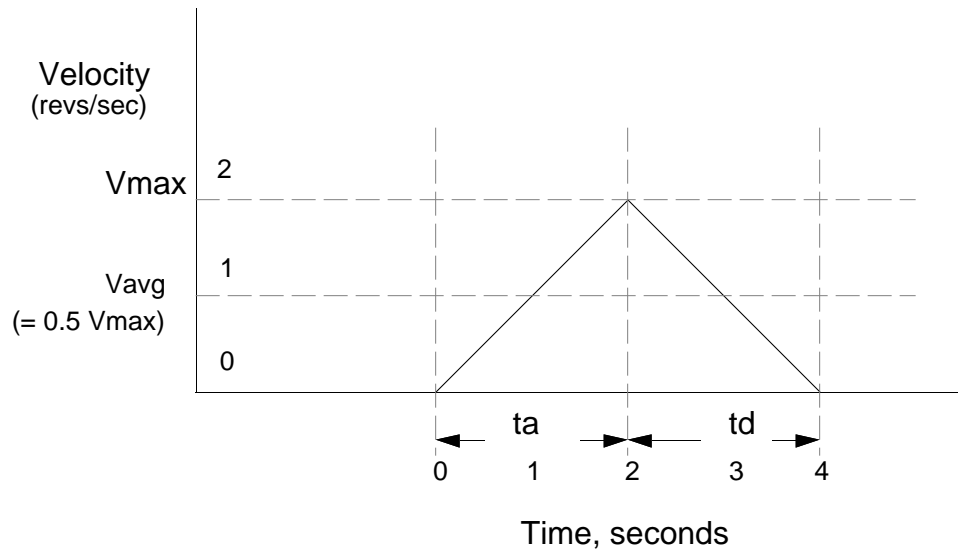
The distance command (D6200)	This command sets the distance the axis will travel if no registration pulse is received and the maximum distance at which the pulse will be recognised.
Trigger distance command (TRD4000)	The distance at which the axis will start to decelerate to the slower speed of 1 rev/sec (V1) in readiness for the registration pulse is set by this command.
Trigger distance command (TRD6020)	This command sets the minimum distance at which the system will recognise the registration pulse.
The TRR command (TRR1440)	This command sets up the registration mode and the registration distance (the distance the axis will travel after the pulse is received)
Registration move velocity (V50)	To save time in travelling the registration distance, the axis then accelerates towards 50 revs/sec (V50), but it does not reach that speed before starting to decelerate to stop at 1440 steps from where the pulse occurred.

### Motion Profiles

Velocity, acceleration and distance must be defined before any preset move can be executed. The value of these parameters determines the type of motion profile as either triangular or trapezoidal.

#### Triangular Profile

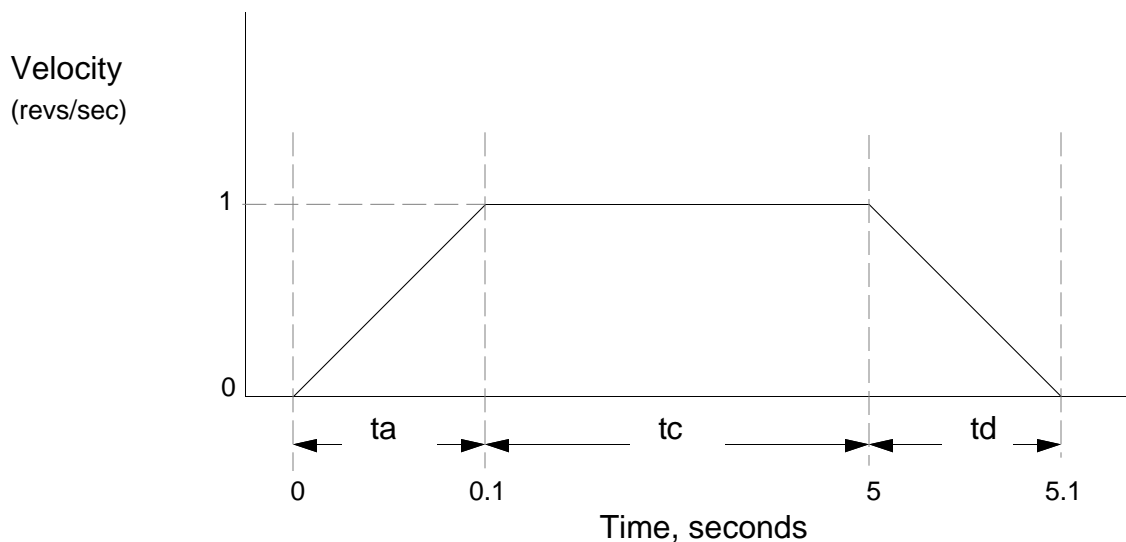
A triangular profile will result when the velocity and acceleration are set such that the designed velocity is not attained before half of the specified distance has been travelled. This results from either a very low acceleration or a very high velocity or both over a relatively short distance. For example, if the acceleration is set to 1 rev/sec/sec, velocity is set to 5 revs/sec and distance is set to 16000 steps (2 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/sec. The motion profile for this move would look like this.



**Figure 5-2. Triangular Profile**

#### Trapezoidal Profile

A trapezoidal move profile results when the defined velocity 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/sec, velocity is set to 1 rev/sec, and distance is specified as 20000 steps (5 revs), the resulting motion profile would look like this:



ta = Accelerate  
tc = Constant velocity  
td = Decelerate

**Figure 5-3. Trapezoidal Profile**

### **User Profiles**

The user may define a move profile using the MC command to establish the continuous mode. Velocity can be programmed on the fly by the V command.

The positioner also has a mode MQ, which is like the preset move mode in that the move distance is pre-defined, but it is possible to change speed in the middle of the move as required based on a distance, input or time delay trigger.

### **Encoder Following**

The commands SIM and CCS may be used to allow the motor of one axis to follow the encoder of another axis or an externally-generated step and direction signal. When the control module is to be used in following mode, the input from the encoder to be followed should be connected to the External Encoder connector pins 1 - 9 (as shown in Figure 2-2). These inputs can be configured using the CCS command as an encoder input (x1, x2 or x4).

The SIM command may be used to select:

- a) Normal indexer operation (SIM0), used for reverting to normal operation by overriding previous SIM commands
- b) Encoder following with indexer motion commands inoperative (SIM1).
- c) Encoder following with indexer commands operative (SIM 2), allowing the superimposing of indexer moves.
- d) Software scaled encoder following (SIM3).
- e) Software scaled encoder following with direction reversal (SIM4).
- f) Preset following index mode (SIM 5).

For SIM1 and SIM2 operation the motor output rate follows the encoder input, using hardware scaling, at a ratio of 1 or less.

### **Software Scaling**

This is the scaling of the encoder input when using the SIM3 or SIM4 commands to achieve following at a ratio greater or less than 1. Unlike hardware scaling, exact following ratios can be achieved by controlling both the numerator and denominator parts of the fraction used to set the scaling ratio, thus ratios such as 3:1 can be obtained.

The scaling ratio is set using the CMR command value divided by the CUR command value to give :

$$\text{Motor Output Rate} = \text{input rate} \times \frac{\text{CMR}}{\text{CUR}}$$

Both CMR values and CUR values are individually limited to a range of 1 to 32,767, and when combined as a fraction are further limited, by software scaling to having both numerator and denominator in the range 1 to 255. This results in a maximum division ratio of 1/255, or at the other extreme a maximum multiplication ratio of 255.

Software scaling does not allow the superimposing of indexer moves.

When using SIM3 or SIM 4 operation for short moves it is possible to predict the number of steps the motor will take using the formula :

Number of motor steps to move =

$$\text{INT}\left(\frac{\text{CMR} \times \text{no. of pulses received} + \text{prev. remainder}}{\text{CUR}}\right)$$

where INT means "take the integer part of", with the value rounded towards zero whether positive or negative. The controller can repeatedly apply this formula to establish an iterative calculation. In situations where short moves have been programmed the ratio of CMR/CUR may only approximate the number of steps the motor is required to move, but since the remainder is carried forward no steps are lost. This allows the CMR/CUR value to better approximate the following ratio in subsequent moves.

In summary, short moves may only approximate the defined following ratio, but no positioning accuracy is lost in later moves.

#### **Buffered Clock Mode**

SIM3 or SIM4 work in a buffered clock mode, which allows the controller to buffer an unprofiled following-input pulse stream until the output velocity equals the following input velocity. The controller accelerates the output velocity to match the input velocity using an exponential acceleration profile, the time constant of which is set using the CAG command. The default time constant value of CAG is 1.00ms to accept already profiled follower inputs.

The input pulses are buffered (or stored) at the input resolution, the actual number being stored at any particular instant being termed the following error. This can lead to input pulses being lost above a maximum speed, due to excessive following error.

## 44 PDHX-E SERIES DRIVE USER GUIDE

---

The maximum number of input pulses that can be buffered is

+/-32767 which requires CAG to be less than  $\frac{32767}{\text{input pulse rate}}$  to prevent overflow.

### **Preset Following Index Mode**

The following mode SIM5 selects indexing at a speed determined by the external input. In this mode the controller sets the motor velocity to a speed in rps determined as a percentage of the following input speed in rps. The percentage following factor is set by the FOL command which can be varied between 0.0 and 5000.0%.

When using this mode the acceleration is fixed to whatever is defined by the A command, and the V command value has no effect since the FOL command percentage value will now control the velocity.

The velocity is dependent on the user resolution (CUR value) and the motor resolution (CMR value). CUR and CMR set the encoder / motor resolution ratio so that the input shaft speed will match the output shaft speed with FOL set to 100%.

The FOL value can be entered from a variable, for example:

1FOL(VAR1)

**Pulses must be present at the input when the G command is executed.**

## **Position Maintenance**

By attaching an encoder to the load, Position Maintenance can be used to correct position errors between motor and encoder at the end of a move. The controller will detect the difference between the number of steps the motor was commanded to move and the number of steps reported by the encoder. This position error is used, at the end of a move, to drive the motor in a direction to give the correct encoder reading.

To make use of Position Maintenance an encoder will need to be fitted at the load end of the system and connected to the controller. If the encoder were connected directly to the stepper motor shaft the stepper motor would need to have a step resolution at least 4X that of the encoder. For example, using the PDHX-E ministepping drive set to 4000 steps/rev will require the use of a 250 step/rev quadrature encoder, which after processing will generate 1000 steps/rev (with CCS0 i.e. normal x4 decode). In a practical application the motor may drive the load via a gearbox or the encoder may sense the load in a manner that alters the number of

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.

Position Maintenance is enabled using the FSC command and can be operated in Encoder Step mode or Motor Step mode using the FSB1 command. Note: If motor steps are chosen the position resolution will still be determined by the encoder.

### **Position Maintenance Using The PDHX-E**

The PDHX-E can use Position Maintenance to correct for positional errors at the end of a move. If a 4,000-step move is made, the motor will attempt to go 4,000 steps (or 1 rev). At the end of the move the controller will read the encoder to check it reads 1,000 steps or one revolution. If not, it will correct the motor's final position by commanding the motor to move until the encoder reads 1,000 steps. In practice, mechanical alignment errors and lost motion within the system will not allow precise matching of motor and encoder, consequently an error band (Dead Band Window) is defined using the DW command. With DW10 the motor may have an error of 10 mini steps before Position Maintenance will attempt to correct the error. If the number is set too small, the motor may oscillate about its correct position.

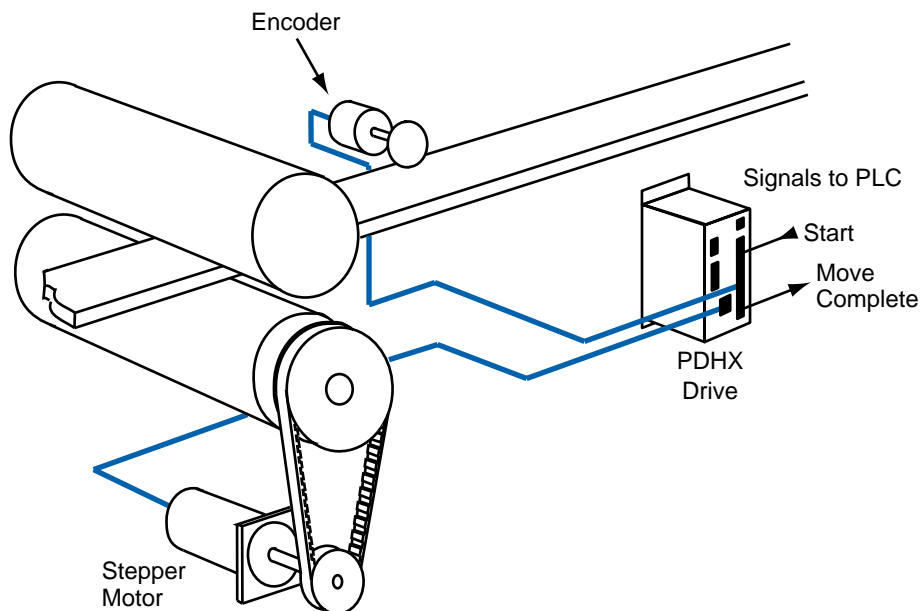
If a Dead Band Window has been set and Position Maintenance is used to correct a position error the motor will only be moved by an amount that just re-positions the motor back within the Dead Band Window. Greater re-positioning accuracy will only be achieved by making the Dead Band Window smaller.

### **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, set by the Maintenance Velocity (MV) command.

### **Use of Position Maintenance**

Used with a ministepping drive, Position Maintenance is generally used to correct for load positioning errors at the end of a move. A typical application would be the correction of slippage in pinch rollers. A stepper motor is used to drive the pinch rollers and the encoder senses the movement of material passing through the rollers via a drive wheel, as shown in Figure 5-4. If slippage occurs driving the required length of material through the rollers the encoder count will disagree with the motor step count, therefore Position Maintenance can be used to give the motor more steps until the correct encoder count is achieved.



**Figure 5-4. Position Maintenance Used to Correct for Slippage**

## Stall Detect

Once an encoder is fitted to the motor, Stall Detection can be used. A stall occurs when the error between the commanded position and the actual position, determined by the encoder, exceeds the value set in the maximum allowable position error (CPE) command. The value of CPE will be in motor steps and should not be set less than 40 steps in 4000 step/rev mode. This is to allow for rotor lead/lag during acceleration and deceleration. If the motor becomes desynchronised, the minimum resulting error will be 80 steps in 4000 step/rev mode.

## Stop-on-stall

You can enable the Stop-on-stall function with the FSD1 command. The move will terminate, without any delay, as soon as the stall is detected. The function can be used in Motor Step or Encoder Step mode.

## Output-on-stall

A drive output can be defined to act as an Output-on-stall by assigning it the letter L. When a stall condition occurs the output will be activated, but it will not latch the stall condition. By selecting an output as an output-on-stall you are not causing the motor to stop on a stall. The motor will not stop on a stall unless you enable it with an FSD1 command.

### **Interaction of Stall Detect and Position Maintenance**

If both Stall Detect and Position Maintenance are enabled and the motor is moved away from its expected position, a stall will occur or position maintenance will attempt to bring the motor back to position. The decision as to what happens is made as follows:

If  $DPE$  (position error)  $>$   $CPE$  (maximum allowable position error) a stall condition exists, if stop-on-stall is enabled the motor will stop.

If  $DPE$  (position error)  $>$   $DW$  (deadband window), position maintenance will occur.

A value of  $DPE$  less than  $DW$  will leave the motor in its present position, a value of  $DPE$  between  $DW$  and  $CPE$  will be corrected using position maintenance and a value of  $DPE$  the same as or greater than  $CPE$  will cause a stall condition.

### **Program Storage**

The program memory is battery backed up RAM with memory retention of 10,000 hours. The RAM has 8K characters available for sequence storage and a write protect facility is provided. Programs are stored in variable length buffers and the total length cannot exceed 8K.

#### **Write Protection**

A write protection facility is incorporated for the protection of stored sequences and parameters stored by the SV command. If switch 8 (on top of the drive) is ON then the memory is not write protected. An attempted SV command when the switch is OFF will result in the following error message being displayed:

#15 BACKUP RAM IS WRITE PROTECTED

### **Motion Program Selection**

The system may be set up so that no sequence is executed at power-up. In this case sequence execution would be initiated from the controller over the RS232C.

Alternatively, a single sequence to be automatically executed at power-up can be programmed. After that sequence is executed, control can pass to another sequence or to the RS232C interface.

Sequence selection may also be initiated via thumbwheel, RP240, and PLC inputs. Up to 64 user defined sequences can be selected for execution in this way.

