A special version of the PM300 has been designed to accommodate the particular needs of a servo controller used in a turntable speed control application.

A switch connected to the PM300’s Fast Jog input selects either manual mode where the turntable speed is set on thumbwheel switches, or computer mode where all the standard PM300 commands are available plus some application specific commands. The PM300 is set to axis address 0, therefore no command address prefix is required.

Special Commands:

- **QSCl**: query current scale factor
- **SCAL**: set scale factor
  - range: 4940 to 5461
  - initial value: 5208
  - The maximum boundary of speed in steps/sec is 65535. This therefore sets the maximum speed x scale factor i.e. $5461 \times 1200 = 65532$ steps/sec.
- **RPM**: set speed in r.p.m. x 10
  - range: 100 to 1200
  - i.e. 10.0 to 120.0 r.p.m.
- **RT**: Read value on thumbwheel switch inputs

**Computer Mode**

In computer mode the speed would first be set using the **RPM** command. The motor is then set in constant velocity mode using the CV command. To stop the motor once running a CV-1 command would be sent. The speed may be changed while in CV mode.

The scale factor that converts r.p.m. at the turntable into steps per second at the motor is set using the **SCAL** command. This may be changed while the motor is running to allow calibration.

An extra function added to the ERROR led on the front panel is **At Speed**. This led will illuminate while the motor is accelerating or decelerating to a new speed. The ERROR output is also accessible via the board connector.

**Setting acceleration & deceleration**

The acceleration is set using the **SA** command and the deceleration is set using the **SD** command. Both values are set in steps/second$^2$ at the motor.

**Servo coefficients**

The initial value of the velocity coefficient KV has been set to 80 and the initial value of the proportional coefficient KP has been set to 1500.

**Manual Mode**

In manual mode the PM300 is also operating in CV mode but with the speed set by thumbwheel switches.

The run switch connected to the PM300’s + Jog input sets the motor running. As in computer mode the front panel LED will go out when the motor is at speed. A Direction switch connected to the PM300’s - Jog input may be used to set the direction of rotation before the run switch is activated.

**Thumbwheel switch wiring**

The type of thumbwheel switches used are diode multiplexed BCD. The common connection to each switch is connected to one of the PM300’s Write Ports and the BCD outputs are connected to the PM300’s Read Port inputs.

The thumbwheel switch inputs may be tested using the **RT** command. This returns a numeric value corresponding to the thumbwheel switch settings.

30 March, 1995
PM300 PROGRAMMERS REFERENCE

CONTROL C (ASCII 03)  Hard Stop

Moves, sequences and profiles halted immediately.
Command buffer cleared.
Error signal cleared.
Sets status to Idle.
Auto-execute flag cleared.
Sequences and profiles retained in memory.
Operates on all axes.

ESC (ASCII 27)  Soft Stop

Command buffer cleared.
Motion stopped at the SD rate.
Status returns to Idle.
Auto-execute flag cleared.
Sequences and profiles retained in memory.
Operates on all axes.

AA  Allow Abort

Allow abort mode. If the error between CP and AP exceeds the value of TR (tracking window) the controller latches the error signal, sets the analogue output signal to the amplifier to OV, turns the Error LED on and aborts a move. The PM300 remains aborted until reset by sending RS or powering-down. The controller will in the meantime respond to queries only. If a move command is attempted the controller will respond with an ! TRACKING ABORT error message. This mode is the default if the controller is re-initialised.

In this mode, selected values for the tracking window will have to take into account the normal position error or lag that occurs during rapid acceleration.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial State</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;AA</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td>Enabled</td>
</tr>
</tbody>
</table>

Condition Requirements

None.

Notes:

Value retained on power-up.

Response:

OK  Command has been accepted.

Example:

IAA  Sets axis 1 to abort on a tracking error.
**AB Abort**

The control of the motor may be aborted by sending AB. When aborted, the Error LED will illuminate the Error output will be activated, the servo loop is disabled and the motor shaft will be free to rotate. A user abort may be reset with the RS command. The position encoder is still read while aborted.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial State</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;AB</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None

**Notes:**

The response to a CO command will be **User Abort**.

**Responses:**

- **OK**
  
  Command has been accepted.

**Example:**

1AB Abort axis 1.

---

**AD Toggle address prefix.**

Toggle address prefix. Turns on or off (depending on the previous state) the axis address number attached to the beginning of a reply string. A colon : is added between the address and the reply.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial State</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;AD</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td>Prefix off</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None

**Notes:**

Value retained on power-up.

**Response:**

- **OK**
  
  Command has been accepted.

**Example:**

1AD Toggle address prefix on axis 1.

Reply 1: OK

1AD Toggle address prefix on axis 1.

Reply OK
AE  SET AUTO-EXECUTE SEQUENCE

Set sequence \( n \) to run on power-up (auto-execute) of the controller. This may be used in stand alone systems where there is no permanent host computer or terminal.

**Syntax**

\(<\text{ad}>\text{AE}n\)  

**Units**  
Seq. No.

**Range**  
0 to 7

**Initial State**  
Disabled

**Condition Requirements**

None

**Notes:**

Value retained on power-up.

Cleared by Control-C and ESC.

**Responses:**

- **OK**  
  Command has been accepted.

- **! OUT OF RANGE**  
  Argument is out of valid range.

- **! NO SEQUENCE**  
  Sequence specified has not been defined yet.

**Example:**

1AE5  
Sets auto execute of axis 1 to run sequence 5 on power-up.

AL  ALLOW SOFT LIMITS

Set the soft limit protection enable to ON. Further movement is bounded by the upper and lower soft limits. Soft limits may be turned OFF by the IL command.

**Syntax**

\(<\text{ad}>\text{AL}\)  

**Units**  
N/A

**Range**  
N/A

**Initial State**  
Enabled

**Condition Requirements**

None

**Notes:**

Value retained on power-up.

**Response:**

- **OK**  
  Command has been accepted.

**Example:**

1AL  
Sets the soft limits ON for controller axis 1.
**AP SET ACTUAL POSITION**

Set the actual position value to that given in the argument.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;APnnn</td>
<td>Steps</td>
<td>-2147483647 to 2147483647(±2^32)</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

Idle or Constant velocity

**Notes:**

Value zero on power-up.

**Response:**

OK Command has been accepted.

**Examples:**

- IAP5000 Set the axis 1 Actual Position to 5000.
- or IAP0 Set the axis 1 Actual Position to zero.

---

**AR ALLOW REMOTE (JOG) CONTROLS**

Set the manual JOG control enable to ON. This enables movement by the JOG inputs. The JOG enable may be turned OFF by the IR command.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range to</th>
<th>Initial State</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;AR</td>
<td>N/A</td>
<td>N/A</td>
<td>Enabled</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None

**Notes:**

Value retained on power-up.

**Response:**

OK Command has been accepted.

**Example:**

IAR Sets the jog control enable ON for controller axis 1.

---

18/09/95
### CO Display the Current Operation

Output the current operation that the controller is executing, i.e. its status.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial State</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;CO</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

#### Condition Requirements
None

#### Responses:
- **Constant Velocity**: Constant velocity move is executing.
- **Creep**: Creep steps section of move is executing.
- **Delay**: Time delay counter running.
- **Emergency Stop**: Emergency Stop input active. No moves executing.
- **Execute Cam**: Cam profile move executing.
- **Execute Profile**: Timed profile move executing.
- **Idle**: No moves executing.
- **Index**: Index to datum executing.
- **Jog**: Jog move executing.
- **Motor Stalled**: TH (threshold) value exceeded due to position encoder failure or stalled motor. No moves executing.
- **Move**: Move Absolute or Move Relative executing.
- **Settle**: End of move settle time counter running.
- **Soft Stop**: Decelerating to stop.
- **Synchronising**: Waiting for position parity in absolute gearbox mode.
- **Tracking Abort**: Controller is aborted due to exceeding the tracking window value TR. No moves executing.
- **User Abort**: Controller is aborted due to the use of an AB (user abort) command. No moves executing.

#### Notes:
- Constant velocity move is executing.
- Creep steps section of move is executing.
- Time delay counter running.
- Emergency Stop input active. No moves executing.
- Cam profile move executing.
- Timed profile move executing.
- No moves executing.
- Index to datum executing.
- Jog move executing.
- TH (threshold) value exceeded due to position encoder failure or stalled motor. No moves executing.
- Move Absolute or Move Relative executing.
- End of move settle time counter running.
- Decelerating to stop.
- Waiting for position parity in absolute gearbox mode.
- Controller is aborted due to exceeding the tracking window value TR. No moves executing.
- Controller is aborted due to the use of an AB (user abort) command. No moves executing.

#### Example:
```
1MR10000
ICO
```

**Response:**
Move  Move Absolute or Move Relative executing.

---

18/09/95    PM300CMD DOC 5
### CP SET COMMAND POSITION

Set the command position value to that given in the argument. The command position is the position generated by a move command.

**Syntax**

<ad>CPnnn

**Units**

Steps

**Range**

-2,147,483,647 to 2,147,483,647 (±2^{32})

**Initial Value**

N/A

**Condition Requirements**

Idle or Constant velocity

**Notes:**

Value zero on power-up.

**Response:**

OK

Command has been accepted.

**Examples:**

1CP5000 Set the axis 1 Command Position to 5000.

or 1CP0 Set the axis 1 Command Position to zero.

---

### CR SET CREEP DISTANCE

Set number of creep steps at the end of a move. The motor will decelerate and execute this number of steps at the creep speed.

**Syntax**

<ad>CRnnn

**Units**

Steps

**Range**

0 to 2,147,483,647 (2^{32})

**Initial Value**

0

**Condition Requirements**

Idle or Constant velocity

**Notes:**

Value retained on power-up.

**Response:**

OK

OUT OF RANGE

Command has been accepted.

Argument is out of valid range.

**Examples:**

1CR50 Set the creep distance to 50 steps on axis 1.
**CV**  
**Constant Velocity Move**

A Constant velocity move ramps up at SA acceleration rate, then moves the motor at the set SV speed. The SV command is used to change the speed whilst motion is in progress. The SA and SD rates define the rate at which the change of speed will be made. Constant velocity mode is exited by an ST command, ESC or Control C.

**Syntax**  
<ad>CVn

<table>
<thead>
<tr>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direction</td>
<td>-ve</td>
<td>N/A</td>
</tr>
<tr>
<td>to</td>
<td>+ve</td>
<td></td>
</tr>
</tbody>
</table>

**Condition Requirements**

- **Idle.**

**Notes:**  
During a constant velocity move, SA, SD and SV commands are acted upon immediately.

**Responses:**

- **OK**  
- ! EMERGENCY STOP  
- ! TRACKING ABORT  
- ! USER ABORT  
- ! MOTOR STALLED

- **Command has been accepted.**  
- **The Emergency Stop has been activated.**  
- **Controller has aborted due to a Tracking error.**  
- **Controller is aborted due to a user command.**  
- **Controller is aborted due to stalled motor or encoder loss.**

**Examples:**

- **ICV**  
- **ICV-1**

Start constant velocity move in positive direction on axis 1.

Start constant velocity move in negative direction on axis 1.

---

**DA**  
**Difference Actual Position**

Add value to actual (position encoder's) position.

**Syntax**  
<ad>DAann

<table>
<thead>
<tr>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steps</td>
<td>-2147483648 to 2147483647 (± 2^31)</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

- None.

**Notes:**

- **Command has been accepted.**

**Examples:**

- **IOA**  
- **IDA5000**  
- **IOA**

Get the axis 1 Actual Position.

Difference axis 1 actual position by 5000.

Get the axis 1 Actual Position.

Response:  

- 15000
- 5000
- 20000
**DB Set Deadband**

Set *Deadband* in number of steps either side of command position. The output from the controller will not increase or decrease for position errors less than the value of the *deadband*.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;DBnnn</td>
<td>Steps</td>
<td>0</td>
<td>4000</td>
<td>0</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None.

**Notes:**
Value retained on power-up. The value of *deadband* is set to zero by the TUNE command.

**Responses**

OK

! OUT OF RANGE

Command has been accepted.

Argument is out of valid range.

**Example:**

1DB5 Set deadband on axis 1 to 5 steps.

---

**DC Difference Command Position**

Add value to Command Position.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;DCnnn</td>
<td>Steps</td>
<td>-2147483648</td>
<td>2147483647 (± 2^32)</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None.

**Notes:**

Command has been accepted.

**Examples:**

1OC Get the axis 1 Command Position.

Response: 15000

1DC5000 Difference axis 1 Command position by 5000.

1OC Get the axis 1 Command Position.

Response: 20000

18/09/95
**DE  DELAY**

This command will start a delay timer for the time given in the argument. After completion of this time, the controller will return to the idle mode.

If the next command should not execute until the end of this delay time, and is not a wait for idle command, then the DE command must be followed with a Wait for End (WE) command. This will make the controller wait until it returns to the idle state before executing the next command.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;DEnnn</td>
<td>milliseconds</td>
<td>0</td>
<td>60000</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

Idle and not Constant velocity or Cam.

**Responses:**

- **OK**
  - Command has been accepted.

- **OUT OF RANGE**
  - Argument is out of valid range.

- **TRACKING ABORT**
  - Controller has aborted due to a Tracking error.

- **USER ABORT**
  - Controller is aborted due to a user command.

- **EMERGENCY STOP**
  - The Emergency Stop has been activated.

- **MOTOR STALLED**
  - Controller is aborted due to stalled motor or encoder loss.

- **CONTEXT**
  - Not available in CV, gearbox or Cam modes.

**Examples:**

- 1MR400   Move 400 steps positive.
- IDE2000  Delay for 2 seconds then...
- IMR-400  Move 400 steps negative.
- IWP2220  Turn LED on (write port 1).
- IDE1000  Delay for 1 second.
- IWE      Wait for End of Delay
- IWP2221  Turn LED off (write port 1).
**DP DEFINE PROFILE**

This command will start a Profile definition. The only command that is used during a Profile definition is MR any other commands except for EP will cause a SYNTAX error.

The commands that follow this DP command will not be executed, but will be stored in the on board non-volatile memory until the End Profile definition (EP) command is received. If a Control-C or ESCAPE command is received or the controller runs out of memory, the Profile definition will cease, the Profile will not be stored and the controller will return to the idle state.

### Syntax

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;DP</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

### Condition Requirements

Idle.

### Notes:

- If a Profile is defined, that fact is shown on the QA page.

### Responses:

- **OK**: Command has been accepted.
- **! RECURSIVE DP**: DP attempted when already defining a Profile.
- **! MEMORY OFLO**: The available memory has overflowed.
- **! DP SYNTAX**: Command is not MR or EP.

### Example:

1. **IDP**: Start Profile definition.
2. **IMR200**: First Profile move.
3. **IMR500**: Next Profile move.
4. **IMR-500**: "
5. **IMR-200**: "
6. **IMR50**: "
7. **IEP**: End of Profile definition.
**DS DEFINE SEQUENCE**

This command will start a sequence definition. There are eight sequences that may be defined and the argument selects which sequence is to be defined (0 to 7).

All valid commands that follow this DS command will not be executed, but will be stored in the on board non-volatile memory until the End Sequence definition (ES) command is received. If a Control-C or ESCAPE command is received or the controller runs out of memory, the sequence definition will cease, the sequence will not be stored and the controller will return to the idle state.

**Syntax**
<ad>DSn  Seq No 0 7

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range to Initial Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSn</td>
<td>Seq No</td>
<td>0 to 7</td>
</tr>
</tbody>
</table>

**Condition Requirements**
Idle

**Notes:**
Auto-execute flag is cleared. The sequences defined are shown on the QA page.

**Responses**
OK
OUT OF RANGE  Argument (sequence number) is out of valid range.
RECURSIVE DS  DS attempted when already defining a sequence.
MEMORY OFLO  The available memory has overflowed.

**Example:**
IDS4  Start definition of sequence 4.
ISV2000  Set slew speed.
IMA8000  First move (absolute).
IMRS000  Next move (relative).
IMR3000  Next move (relative).
ISV20000  Set new slew speed.
IMA0  Next move (return to start position).
IXS4  Execute sequence 4 (loop to start of this sequence).
IES  End of sequence definition.

**EP END PROFILE DEFINITION**

This command will end a Profile definition. The Profile definition must have been started by the Define Profile (DP) command.

**Syntax**
<ad>EP

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range to Initial Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>EP</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**
Define Profile

**Notes:**

**Responses:**
OK
EP WITHOUT DP  EP attempted when NOT already defining a Profile.

**Example:**
IDP  Start Profile definition.
IMR200  First Profile move.
IMR500  Next Profile move.
IEP  End of Profile definition.

18/09/95
ES END SEQUENCE DEFINITION

This command will end a sequence definition. The sequence definition must have been started by the Define Sequence (DS) command. No argument is necessary as the sequence number is specified with the Define Sequence (DS) command.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;ES</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td>N/A</td>
</tr>
</tbody>
</table>

Condition Requirements: Define Sequence

Responses:

- **OK**: Command has been accepted.
- **ES WITHOUT DS**: ES attempted when NOT already defining a sequence.

Example:

1. DS2: Start definition of sequence 2.
2. MR400: First move (relative).
3. MR-400: Next move (relative).
4. XS5: Execute sequence 5 (transfer control to start of this sequence 5).
5. ES: End of sequence definition.
Help pages. The commands HE1 and HE2 return pages showing Digilop commands. These help pages give a concise list of the commands available and their function.

**Syntax**

<table>
<thead>
<tr>
<th>&lt;ad&gt;-HEEn</th>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>page</td>
<td></td>
<td>0 to 2</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None.

Notes:

The command HE0 is the same as HE1. Values of HE greater than 1 show HE2.

**Responses:**

**HE1**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>allow abort</td>
</tr>
<tr>
<td>AD</td>
<td>toggle addressing</td>
</tr>
<tr>
<td>AL</td>
<td>allow limits</td>
</tr>
<tr>
<td>AR</td>
<td>allow jog</td>
</tr>
<tr>
<td>CP&lt;position&gt;</td>
<td>set command position</td>
</tr>
<tr>
<td>CV&lt;direction&gt;</td>
<td>constant velocity</td>
</tr>
<tr>
<td>DB&lt;distance&gt;</td>
<td>set deadband</td>
</tr>
<tr>
<td>DE&lt;time in ms&gt;</td>
<td>delay</td>
</tr>
<tr>
<td>DS&lt;seq no.&gt;..ES</td>
<td>define sequence</td>
</tr>
<tr>
<td>ID</td>
<td>identify</td>
</tr>
<tr>
<td>IA</td>
<td>ignore abort</td>
</tr>
<tr>
<td>IR</td>
<td>inhibit jog</td>
</tr>
<tr>
<td>KF&lt;value&gt;</td>
<td>set feedforward co.</td>
</tr>
<tr>
<td>KS&lt;value&gt;</td>
<td>set sum co.</td>
</tr>
<tr>
<td>LL&lt;position&gt;</td>
<td>set lower soft limit</td>
</tr>
<tr>
<td>MA&lt;position&gt;</td>
<td>move absolute</td>
</tr>
<tr>
<td>OA</td>
<td>output actual pos</td>
</tr>
<tr>
<td>OD</td>
<td>output difference</td>
</tr>
<tr>
<td>QA</td>
<td>query all</td>
</tr>
<tr>
<td>OS</td>
<td>query speeds</td>
</tr>
<tr>
<td>RS</td>
<td>reset from abort</td>
</tr>
<tr>
<td>RSST</td>
<td>reset from stalled</td>
</tr>
<tr>
<td>SD&lt;deceleration&gt;</td>
<td>set deceleration</td>
</tr>
<tr>
<td>SC&lt;speed&gt;</td>
<td>set creep speed</td>
</tr>
<tr>
<td>ST</td>
<td>stop</td>
</tr>
<tr>
<td>TH&lt;value&gt;</td>
<td>set stall threshold</td>
</tr>
<tr>
<td>TUNE</td>
<td>tune coefficients</td>
</tr>
<tr>
<td>UP, US</td>
<td>undefine cam etc.</td>
</tr>
<tr>
<td>WA&lt;bit pattern&gt;</td>
<td>wait for condition</td>
</tr>
<tr>
<td>WE</td>
<td>wait for end of move</td>
</tr>
<tr>
<td>XP&lt;distance&gt;</td>
<td>set window</td>
</tr>
<tr>
<td>XS&lt;time in ms&gt;</td>
<td>execute profile</td>
</tr>
</tbody>
</table>

**HE2**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MA&lt;position&gt;</td>
<td>move absolute</td>
</tr>
<tr>
<td>OA</td>
<td>output actual pos</td>
</tr>
<tr>
<td>OD</td>
<td>output difference</td>
</tr>
<tr>
<td>QA</td>
<td>query all</td>
</tr>
<tr>
<td>OS</td>
<td>query speeds</td>
</tr>
<tr>
<td>RS</td>
<td>reset from abort</td>
</tr>
<tr>
<td>RSST</td>
<td>reset from stalled</td>
</tr>
<tr>
<td>SD&lt;deceleration&gt;</td>
<td>set deceleration</td>
</tr>
<tr>
<td>SC&lt;speed&gt;</td>
<td>set creep speed</td>
</tr>
<tr>
<td>ST</td>
<td>stop</td>
</tr>
<tr>
<td>TH&lt;value&gt;</td>
<td>set stall threshold</td>
</tr>
<tr>
<td>TUNE</td>
<td>tune coefficients</td>
</tr>
<tr>
<td>UP, US</td>
<td>undefine cam etc.</td>
</tr>
<tr>
<td>WA&lt;bit pattern&gt;</td>
<td>wait for condition</td>
</tr>
<tr>
<td>WE</td>
<td>wait for end of move</td>
</tr>
<tr>
<td>XP&lt;distance&gt;</td>
<td>set window</td>
</tr>
<tr>
<td>XS&lt;time in ms&gt;</td>
<td>execute profile</td>
</tr>
</tbody>
</table>

**Example:**

1HE1  Show the first help page of the controller of axis 1.
IGNORE ABORT

Ignore Tracking abort. If the error between the command position and actual position exceeds the TR Tracking window value, the controller does not abort but continues to control the motor in the normal way. The Error signal and LED are still activated for the duration of the error. If the size of the tracking window is altered when in this mode, the error message 'TRACKING DISABLED' will be returned. The setting of the Tracking Abort, either Enabled or Disabled is shown in the QA page.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad=IA</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td>Enabled</td>
</tr>
</tbody>
</table>

Condition Requirements
None.

Notes:
Value retained on power-up.

Response:
OK
Command has been accepted

Example:
IA Tracking errors on axis 1 will not cause an abort.

IDENTIFY

This command is used to give the type of controller and its internal software revision.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;ID</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td>N/A</td>
</tr>
</tbody>
</table>

Condition Requirements
None.

Notes:

Response:
Mclennan Servo Supplies Ltd. PM300 V6.12

Example:
IAD Toggle address prefix on or off.
ID Identify controller of axis 1. If this was a PM300 and the address prefix was previously off, it would respond:
1:Mclennan Servo Supplies Ltd. PM300 V6.12
### IGNORE SOFT LIMITS

Set the soft limit protection enable to OFF. Further movement is NOT bounded by the upper and lower soft limits. Soft limits may be turned ON by the AL (allow limits) command. Hard limits will still be active and cannot be disabled.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;IL</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td>Enabled.</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None.

**Notes:** Value retained on power-up.

**Response:**

OK

Command has been accepted.

**Example:**

IL

Sets the soft limits OFF for controller axis 1.

---

### INITIALISE

This command will set all the programmable parameters back to their initial values, clear sequences and profiles. This is used to re-initialise all the non-volatile memory values to 'safe' values - e.g. if the controller was to be used in a new application.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;IN</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None.

**Notes:**

None.

**Response:**

OK

Command has been accepted.

**Example:**

IN

Set all parameters on axis I back to their initial values.

---

### INHIBIT REMOTE (JOGS) CONTROLS

Disable movement by the JOG inputs. The JOG inputs may be re-enable by the AR (allow remote) command.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;IR</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td>Enabled.</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None

**Notes:** Value retained on power-up.

**Response:**

OK

Command has been accepted.

**Example:**

IR

Disable the Jog control inputs for controller on axis I.
This command is used to find a datum point of a mechanism. Index to datum ramps the motor up at SA rate, then moves at the set SV rate until a *slow down* (datum approach) signal is received on read port 1. It will then ramp down at SA rate to SC creep speed until receipt of a *stop* (datum stop) signal on Read Port 2. User may then define this position as required.

**Important.** The respective SV and SA rates and index sensor positions should be set such that the creep speed is reached before the final stop signal is received. For accurate location of the datum position the creep speed SC should be set at below 1000 steps per second. After an index operation the positions are not reset.

If no Datum Approach switch is used then the search is performed only at the creep speed.

The IX-1 command will perform the same in the negative direction.

Soft limits are not used during an Index mark search.

**Syntax**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;IXnnn</td>
<td>N/A</td>
<td>-ve</td>
<td>+ve</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

Idle.

**Responses**

<table>
<thead>
<tr>
<th>Command</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
<td>Command has been accepted.</td>
</tr>
<tr>
<td>HARD LIMIT</td>
<td>Move attempted when already on hard limit.</td>
</tr>
<tr>
<td>TRACKING ABORT</td>
<td>Controller has aborted due to a Tracking error.</td>
</tr>
<tr>
<td>USER ABORT</td>
<td>Controller is aborted due to a user command.</td>
</tr>
<tr>
<td>EMERGENCY STOP</td>
<td>The Emergency Stop has been activated.</td>
</tr>
<tr>
<td>MOTOR STALLED</td>
<td>Controller is aborted due to stalled motor or encoder loss.</td>
</tr>
</tbody>
</table>

**Examples:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lIX</td>
<td>Search for datum point of axis 1 in positive direction.</td>
</tr>
<tr>
<td>lIX-1</td>
<td>Search for datum point of axis 1 in negative direction.</td>
</tr>
</tbody>
</table>

---

**KF**

Set velocity feedforward servo coefficient. This compensates for the position offset caused by the velocity lag introduced by KV. For positioning moves KF is normally set at zero, but for Profiles and Cam moves where the actual position should not lag behind the command position, KF should be set equal to KV.

It is not usually necessary for complete compensation of the velocity lag as this adversely affects the settling time of the system.

**Syntax**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;KFnnn</td>
<td>Number</td>
<td>0</td>
<td>32767</td>
<td>0</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None. Value retained on power-up.

**Responses**

<table>
<thead>
<tr>
<th>Command</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
<td>Command has been accepted.</td>
</tr>
<tr>
<td>OUT OF RANGE</td>
<td>Argument is out of valid range.</td>
</tr>
</tbody>
</table>

**Examples:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lKF500</td>
<td>Set velocity feedforward on axis 1 to 500.</td>
</tr>
</tbody>
</table>
KP Set Proportional Gain Coefficient

Set proportional gain servo coefficient. The stiffness and accuracy of the servo loop are controlled by the magnitude of the proportional gain.

Syntax
<ad>KPnnn Number 0 to 32767 10

Condition Requirements
None

Notes:
Value retained on power-up.

Responses
OK Command has been accepted.
1 OUT OF RANGE Argument is out of valid range.

Examples:
1KP100 Set the proportional gain on axis 1 to 100.

KS Set Sum Gain Coefficient

The Sum servo coefficient is the sum of the integral and proportional components of the servo control loop. The accuracy of the servo loop depends on having a non-zero value of KS at the expense of transient response.

Syntax
<ad>KSnnn Number 0 to 32767 0

Condition Requirements
None

Notes:
Value retained on power-up.

Responses
OK Command has been accepted.
1 OUT OF RANGE Argument is out of valid range.

Examples:
1KS50 Set the Sum gain on axis 1 to 50.
KV  Set Velocity Feedback Coefficient

The value of this coefficient defines the magnitude of the velocity feedback signal derived from the position encoder. This coefficient influences the transient response by producing a damping effect. It affects the system by reducing overshoot and enhancing stability, but too high a value can create a buzzy system, and ultimately an unstable system.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range to Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;KVnnn</td>
<td>Number</td>
<td>0 to 32767 0</td>
</tr>
</tbody>
</table>

**Condition Requirements**
None.

**Notes:**
Value retained on power-up.

**Responses**

- OK
- ! OUT OF RANGE

**Examples:**

1KV500  Set the Velocity feedback on axis 1 to 500.

LL  SET LOWER SOFT LIMIT POSITION

This command will set the Lower Soft Limit Position to the value given in the argument. Subsequent moves by the Move Absolute (MA) or Move Relative (MR) and manual Jog moves will not be allowed below this Lower Limit if the Soft Limits are enabled.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range to Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;LLnnn</td>
<td>Steps</td>
<td>-2147483648 to 2147483647 (-2(^{32}))</td>
</tr>
</tbody>
</table>

**Condition Requirements**
Idle or Constant velocity

**Notes:**
Value retained on power-up.

**Responses**

- OK
- ! LIMITS DISABLED
- ! LIMIT CONFLICT

**Example:**

ILL-4000  Set the axis 1 Lower Soft Limit Position to -4000.
### MA  MOVE TO ABSOLUTE POSITION

This command will move the motor to the position given in the argument. This position is relative to the Command Position of zero.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;&lt;ad&gt;MAnnn</td>
<td>Steps</td>
<td>-2147483647</td>
<td>2147483647 (±2^32)</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

*Idle*

**Responses**

- **OK**
  - Command has been accepted.
- **EMERGENCY STOP**
  - The Emergency Stop has been activated.
- **TRACKING ABORT**
  - Controller has aborted due to a Tracking error.
- **USER ABORT**
  - Controller is aborted due to a user command.
- **MOTOR STALLED**
  - Controller is aborted due to stalled motor or encoder loss.
- **HARD LIMIT**
  - Move attempted when already on a hard limit.
- **SOFT LIMIT**
  - Move attempted beyond a soft limit.

**Example:**

If axis 1 has a current Command Position of 5000 then the command:

```
IMA4000
```

Will move 1000 steps in the negative direction to arrive at a Command position of 4000.

### MR  MOVE TO RELATIVE POSITION

This command will move the motor to the position given in the argument relative to the current Command Position.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;&lt;ad&gt;MRnnn</td>
<td>Steps</td>
<td>-2147483647</td>
<td>2147483647 (±2^32)</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

*Idle*

**Responses**

- **OK**
  - Command has been accepted.
- **EMERGENCY STOP**
  - The Emergency Stop has been activated.
- **TRACKING ABORT**
  - Controller has aborted due to a Tracking error.
- **USER ABORT**
  - Controller is aborted due to a user command.
- **MOTOR STALLED**
  - Controller is aborted due to stalled motor or encoder loss.
- **HARD LIMIT**
  - Move attempted when already on a hard limit.
- **SOFT LIMIT**
  - Move attempted beyond a soft limit.

**Example:**

If axis 1 has a current Command Position of 5000 then the command:

```
IMR4000
```

Will move 4000 steps in the positive direction to arrive at a Command position of 9000.
### OA OUTPUT ACTUAL POSITION

This command will give the current encoder read Actual Position. This position is derived from the incoming position encoder pulses.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;OA</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None

**Response:**

The response is a string of numeric characters.

**Example:**

If the controller of axis 1 currently has an Actual Position of 20501 then the command: 
1OA will respond:  AP=20501

### OC OUTPUT COMMAND POSITION

This command will give the current Command Position.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;OC</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None.

**Response:**

The response is a string of numeric characters.

**Example:**

If the controller of axis 1 currently has a Command Position of 45280 then the command: 
1OC will respond:  CP=45280

---

18/09/95 PM300CMD DOC 20
OD OUTPUT DIFFERENCE BETWEEN COMMAND AND ACTUAL POSITIONS

This command will give the difference between the current Command Position and the current encoder read Actual Position. Numerically it is the Command Position (CP) - Actual Position (AP).

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;OD</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Condition Requirements
None.

Response:
The response is a string of numeric characters.

Example:
If the controller of axis 1 currently has a Current position of 1000 and an Actual Position of 1050 then the command:
1OD
will respond: DP= -50

OS OUTPUT STATUS STRING

This command will return an eight bit string that indicates the status of the controller in a format that is more easily interpreted by a host computer.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;OS</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Condition Requirements
None.

Responses:
The response is a string of four numeric characters. The characters are either '0' for not active or '1' for active. The 1: part would only appear if the AD address toggle had been set to prefix replies with the axis address of the replying controller.

1:00000000
- Emergency Stop: 1 = active.
- Motor Stalled: 1 = active.
- Tracking Abort: 1 = active.
- User Abort: 1 = active.
- Controller Idle, i.e., awaiting next command: 1 = idle.
- Not Error: 1 = not stopped, not stalled nor aborted.
- +ve Hard Limit: 1 = activated.
- -ve Hard Limit: 1 = activated.

Example:
If the PM300 on axis 1 currently is idle, not stopped, stalled, aborted nor on either hard limit, then the command:
1OS
will respond: 00110000
Query All. Returns all of the current settings and modes of the controller along with the current positions in a single page format.

Syntax
<ad>QA

Units
N/A

Range
N/A

to
N/A

Initial Value
N/A

Condition Requirements
None.

Response:
The response is alpha-numeric strings of characters. Each line gives the parameter name and its value. See example for the format.

Example:
IQA

Will generate a response of the form:

```
Mclennan Servo Supplies Ltd PM300 V6.12
Address: 1
Status: Idle
KP=2132
KS=2304
Slew Speed = 200000
Deceleration = 10000
Acceleration = 10000
Creep Speed = 100
Jog Speed = 100
Settling = 10
Window = 4
Tracking Abort: Enabled
Soft Limits: Enabled
Lower Limit = -2147483648
Lower hard limit: Off
Gbox Num = 1
Command Pos = 98789
Pos Error = 0
Autoexec: Sequence #6
Sequences: 0,1,2,3,4,5,6
Cam Defined
Memory Usage 95%
Read Port: 1111
```

18/09/95 PM300CMD.DOC 22
### QK Query K Coefficients

Query servo loop coefficients. Returns the current settings of the KP, KS, KV, and KF coefficients.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;QK</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None

**Response:**

The response is an alpha-numeric string of characters showing the parameter name and its value. See example for the format.

**Example:**

- **IN** Set to initial values.
- **IKP2909** Set proportional gain to 2909.
- **IKV357** Set velocity feedback to 357.
- **IKS3258** Set Sum coefficient to 3258.
- **IQK** Will generate a response of the form:
  
  \[ KP=2909, KS=3258, KV=357, KF=0 \]

### QS Query Speeds

Query the current settings for the speeds and accelerations. Returns the current settings of SV, SC, SA and SD.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;QS</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None

**Response:**

The response is an alpha-numeric string of characters showing the parameter name and its value. See example for the format.

**Example:**

- **ISC1000** Set creep speed to 1000 steps/sec.
- **ISV16200** Set slew speed to 16200 steps/sec.
- **ISA100000** Set deceleration to 100,000 steps/sec\(^2\).
- **ISD100000** Set deceleration to 100,000 steps/sec\(^2\).
- **IQK** Will generate a response of the form:
  
  \[ SV=16200, SC=1000, SA=100000, SD=100000 \]
### RP READ INPUT PORT

This command will examine the read port inputs and return their current state as a four digit numeric string of either 0 or 1 characters. The string starts with read port 4. A 1 indicates that the input is low (0V or open-circuit) and a 0 indicates that the input is high (+24V).

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;RP</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None.

**Notes:**

If an RP command is executed with the read ports open circuit, a reply of 1111 will be returned.

**Responses**

A four digit numeric string.

**Example:**

If the following states are present on the inputs:

<table>
<thead>
<tr>
<th>PORT</th>
<th>STATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Low</td>
</tr>
<tr>
<td>3</td>
<td>Low</td>
</tr>
<tr>
<td>2</td>
<td>Low</td>
</tr>
<tr>
<td>1</td>
<td>High</td>
</tr>
</tbody>
</table>

then the command 1RP will reply: 1110

### RS RESET FROM ABORT

This command will reset the tracking abort or user abort conditions and re-enable the servo control loop. It will also set the Command position to be equal to the Actual position.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;RS</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None.

**Notes:**

- OK: Command has been accepted.
- ! EMERGENCY STOP: The Emergency Stop has been activated.
- ! MOTOR STALLED: Controller is aborted due to stalled motor or encoder loss.
- ! NOT ABORTED: The unit has not aborted.

**Responses**

- OK
- ! EMERGENCY STOP
- ! MOTOR STALLED
- ! NOT ABORTED

**Example:**

IRS Reset abort on axis 1 controller.
**RSES**  
RESET FROM EMERGENCY STOP

This command will, if the emergency stop input is not active, resets the Emergency Stop condition and re-enables the servo control loop. It will also set the Command position to be equal to the Actual position.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;RSES</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**  
None.

**Response**
- OK  
  Command has been accepted.
- 1 NOT STOPPED  
  Stop Input has not been activated.

**Example:**
1RSES  
Reset stopped condition on axis 1 controller.

---

**RSST**  
RESET FROM MOTOR STALLED

This command will reset the Motor Stalled condition and re-enable the servo control loop. It will also set the Command position to be equal to the Actual position.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;RSST</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**  
None.

**Response**
- OK  
  Command has been accepted.
- 1 NOT STALLED  
  Motor Stalled threshold has not been exceeded.

**Example:**
1RSST  
Reset motor stalled condition on axis 1 controller.

18/09/95  
PM300CMD.DOC  
25
### SA  SET ACCELERATION

Set the acceleration rate for changes of velocity for all following moves. This may also be used during a constant velocity move.

**Syntax**

<ad>SAnnn

**Units**

Steps/sec²

**Range**

1 to 20000000

**Initial Value**

10000

**Condition Requirements**

Idle or Constant velocity

**Notes:**

Value retained on power-up.

**Responses:**

OK

1 OUT OF RANGE

Command has been accepted.

Argument is out of valid range.

**Example:**

ISA10000  Sets acceleration of axis 1 controller to 10000 Steps/sec².

---

### SC  SET CREEP SPEED

Set the creep speed for all following moves. This is the speed that at which moves with a non-zero creep distance will stop.

It is also the speed that slow datum search will be moved at (IX command).

**Syntax**

<ad>SCnnn

**Units**

Steps/sec

**Range**

1 to 400000

**Initial Value**

100

**Condition Requirements**

Idle or Constant velocity

**Notes:**

Value retained on power-up.

**Responses:**

OK

1 OUT OF RANGE

Command has been accepted.

Argument is out of valid range.

**Example:**

ISC700  Sets creep speed of axis 1 controller to 700 Steps/sec.

---
SD SET DECELERATION

Set the deceleration rate for changes of velocity for all following moves. This may also be used during a constant velocity move.

Syntax
<ad>SDnnn

Units
Steps/sec\(^2\)

Range
1

to
20000000

Initial Value
10000

Condition Requirements
Idle or Constant velocity

Notes:
Value retained on power-up.

Responses:
OK
! OUT OF RANGE

Command has been accepted.
Argument is out of valid range.

Example:
ISD10000 Sets acceleration of axis 1 controller to 100000 Steps/sec\(^2\).

SE SET SETTLING TIME

Set the settling time for all following moves. This time elapses at the end of each move to allow the motor to settle. The end of a move is defined by the OD (position difference) value being less than the WI (end of move window) value for the SE (settling) time.

Syntax
<ad>SEnnn

Units
milliseconds

Range
0

to
20000

Initial Value
10

Condition Requirements
Idle or Constant velocity

Notes:
Value retained on power-up. While the settling time is elapsing the CO command will give a reply of Settle.

Responses
OK
! OUT OF RANGE

Command has been accepted.
Argument is out of valid range.

Example:
ISE1000 Sets settling time of axis 1 controller to 1 second.
Skip next command if true. The controller will *skip over* (ignore) the following command if the read ports correspond to the bit pattern specified. This command will examine the read port inputs and compare them with the specified bit pattern argument. If the inputs are equal to the specified bit pattern, then the controller will skip over, i.e. not execute the next command. If no commands are in the command buffer or in a sequence the *next* command will be the next received command. If the *skip* condition is not met, then the next command will be executed as normal. If the next command is skipped, the controller will give the response *SKIPPED* instead of *OK* or any other response for that command.

The bit pattern is specified as a four digit binary number of either 0, 1 or 2 characters starting with read port 4, through to 1. A 0 defines that the input must be high (+24V), a 1 defines that the input must be low (0V) and a 2 defines that the input is not relevant or *don't care*. If less than four digits are specified in the argument, then the preceding ones are assumed as high (0).

This command may be used to introduce a conditional response to some machine functions, and may be used to create *smart sequences*.

**Syntax**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;SNbbbb</td>
<td>Bit pattern</td>
<td>4 digits of 0, 1 or 2</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None.

**Responses**

- **OK**
  - Command has been accepted.

- **! SN SYNTAX**
  - Invalid argument i.e. bit specified was not 0, 1 or 2 OR the number of bits was greater than 4.

**Example:**

1DS3 Define the start of the sequence 3.
1WA2221 Wait here until read port 1 goes low.
1SN2212 Skip next command if read port 2 is low, state of ports 1, 3 & 4 not important.
1FX Search for datum (only executed if port 2 was high, above).
1MR1000 Move motor 1000 steps.
1XS Loop back to start of sequence.
1ES End definition of sequence.

The sequence shown is a repeated incremental indexing motion, but allows an operator to search for datum only when required, usually just after power-up. If the AE (Auto execute) flag is set, the system could be operated without the presence of a host computer.
ST STOP

This command will stop any current move, decelerate the motor speed down at the SD rate, then stop and return to idle mode.

This command is buffered and is only responded to when it reached in the command queue. Care must therefore be taken that there are no commands that hold up the queue between the move command and the ST command.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;ST</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**
None

**Notes:**
Will exit constant velocity mode or gearbox mode.

**Responses**
- **OK**
  - Command has been accepted.
- **EMERGENCY STOP**
  - The Emergency Stop has been activated.
- **MOTOR STALLED**
  - Controller is aborted due to stalled motor or encoder loss.

**Example:**
- **ICV**
  - Will start axis 1 moving in constant velocity mode.
- **1ST**
  - This will then stop the current move of axis 1.

SV SET VELOCITY

Set the Slew (maximum) velocity for all following moves. This may also be used during a constant velocity move.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;SVnnn</td>
<td>Steps/sec</td>
<td>1</td>
<td>400000</td>
<td>100</td>
</tr>
</tbody>
</table>

**Condition Requirements**
Idle or Constant velocity

**Notes:**
Value retained on power-up.

**Responses**
- **OK**
  - Command has been accepted.
- **OUT OF RANGE**
  - Argument is out of valid range.

**Example:**
- **ISV5000**
  - Sets slew speed of axis 1 controller to 5000 Steps/sec.
TH SET THRESHOLD

This command will set the motor stalled threshold. Failure of an encoder is indistinguishable from a stalled motor, and messages from the PM300 refer to motor stalled rather than encoder failure.

A stalled motor (or encoder failure) is detected by looking for changes in the position encoder signals (or equivalently the changes in observed motor position). If the motor does not move, and the voltage output value from the PM300 exceeds the value set by the TH command for a time of 256ms, then the PM300 will set its output to zero and set a Motor Stalled condition.

The servo system will have coulomb friction and the voltage required to overcome this friction, varies from system to system, so the value of TH must be large enough not to nuisance trigger but small enough to detect any failure.

If a stalled motor condition occurs, the error signal and front panel LED are both activated, and movement is stopped. Subsequent moves will not function but will return the response ! MOTOR STALLED until reset by either a Reset Stall (RSST) command or by powering off.

The response to a CO command is Motor Stalled.

Syntax
<ad>THnnn

Units
Steps

Range
0 to 2047

Initial Value
200

Condition Requirements
None.

Notes:
Value retained on power-up.

Responses:
OK
! OUT OF RANGE

Command has been accepted.
Argument is out of valid range.

Example:
ITH400 Set the Threshold before motor stalled condition for axis 1 to 400.
This command will set the tracking window. The *Tracking window* is the allowable difference between the *Command Position* and the *Actual Position*. When the motor is stationary this is the allowable static error. During a move, a changing command position is generated. The *Tracking Window* operates on the difference between the *actual position* and this moving command position. The servo system will have a *following error*, so the value of TR must be large enough not to nuisance trigger but small enough to detect any failure.

If the *tracking window* is exceeded the *Error output signal* and front panel LED are activated and (if abort is enabled) the controller aborts.

The abort function may be inhibited by using the 1A (ignore abort) command, or enabled using the AA (allow abort) command.

Subsequent moves will not function but will return the response 1 TRACKING ABORT until reset by either a Reset (RS) command or by powering off.

**Syntax**

```
<ad> TRnnn
```

**Units**

Steps

**Range**

0 to 2,147,483,647 ($2^{31}$)

**Initial Value**

4000

**Condition Requirements**

None.

**Notes:**

Value retained on power-up.

**Responses:**

- **OK**
  - Command has been accepted
- **OUT OF RANGE**
  - Argument is out of valid range
- **TRACKING DISABLED**
  - Warning that tracking abort is inhibited. Value will be accepted.

**Example:**

1TR400  
Set the Tracking Window for axis 1 to 400 steps.
An approximate set of servo coefficients can usually be derived by invoking the **TUNE** command. The controller will exercise the motor over a small displacement for a few seconds and obtain a set of values for the $K$ coefficients that should be stable and provide a reasonable disturbance rejection.

The tuning algorithm may fail if there is excessive backlash, if the low frequency loop gain is either very small or very large or the feedback encoder phasing is wrong. Further optimisation of system response will be required in almost all cases to achieve the desired performance.

The **TUNE** command only affects $KP$, $KV$, $KS$ and $KV$ therefore its use in a double encoder system is inappropriate and may produce a **TUNE FAILURE** error.

### Syntax

```
<ad>TUNE
```

<table>
<thead>
<tr>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Condition Requirements</th>
<th>Notes:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Idle</td>
<td>Value retained on power-up.</td>
</tr>
<tr>
<td></td>
<td>Deadband value set to zero.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Responses:</th>
</tr>
</thead>
<tbody>
<tr>
<td>n, m</td>
</tr>
<tr>
<td>! TUNE FAILURE</td>
</tr>
<tr>
<td>! EMERGENCY STOP</td>
</tr>
<tr>
<td>! TRACKING ABORT</td>
</tr>
<tr>
<td>! MOTOR STALLED</td>
</tr>
<tr>
<td>! USER ABORT</td>
</tr>
<tr>
<td>! HARD LIMIT</td>
</tr>
</tbody>
</table>

**Example:**

```
1 TUNE
```

Tune coefficients on axis 1 controller.

### SET UPPER SOFT LIMIT POSITION

This command will set the Upper Soft Limit Position to the value given in the argument. Subsequent moves by the Move Absolute (MA) or Move Relative (MR) and manual Jog moves will not be allowed above this Upper Limit if the Soft Limits are enabled.

```
<ad>ULnnn
```

<table>
<thead>
<tr>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steps</td>
<td>-2147483647 to 2147483647 ($\pm 2^{32}$)</td>
<td>2147483647 ($2^{32}$)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Condition Requirements</th>
<th>Notes:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Idle or Constant velocity</td>
<td>Value retained on power-up.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
</tr>
<tr>
<td>! LIMITS DISABLED</td>
</tr>
<tr>
<td>! LIMIT CONFLICT</td>
</tr>
</tbody>
</table>

**Example:**

```
1UL8000
```

Set the axis 1 Upper Soft Limit Position to 8000.
This command will undefine or cancel a Profile definition. This will then free the memory used by the Profile for use in other sequences or profile definitions.

Syntax

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;UP</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>

Condition Requirements

Idle.

Responses:

OK

Command has been accepted.

Example:

IUP  Delete Profile from axis 1 controller.

This command will undefine or cancel a sequence definition. This will then free the memory used by the sequence for use in other sequences or profile definitions.

Syntax

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;USn</td>
<td>Seq. No</td>
<td>0</td>
<td>7</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Condition Requirements

Idle.

Responses:

OK

Command has been accepted.

I OUT OF RANGE

Argument (sequence number) is out of valid range.

Example:

IUS6  Delete sequence 6 from axis 1 controller.
**VP **

**VERIFY PROFILE**

This command will list a previously defined Profile.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;VP</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None

**Notes:**

Arguments with a value of zero are not shown.

**Responses:**

The command will either respond with the axis address identifier (if address prefix enabled) followed by each line of the Profile, or an error message:

1 NO PROFILE Profile has not been defined yet

**Example:**

A controller that had previously been programmed with:

1DP Start definition of Profile.
1MR2000 First move.
1MR7000 Next move.
1MR1000 " "
1MR0 " "
I EP End of Profile definition.

The command 1VP would give:

1:
MR 2000
MR 7000
MR 1000
MR
OK
This command will list a previously defined Sequence.

Syntax

<ad>VSn Seq. No. 0 7 N/A

Condition Requirements

None.

Notes:

Arguments with a value of zero are not shown.

Responses

The command will either respond with the axis address identifier (if address prefix enabled) followed by each line of the sequence, or an error message:

OUT OF RANGE  Argument (sequence number) is out of valid range.

NO SEQUENCE  Sequence specified has not been defined yet.

Example:

A controller that had previously been programmed with:

IDS2  Start definition of sequence 2.
IMA2000  First move (absolute).
IMR7000  Next move (relative).
IDE1000  Delay for 1 second.
IMA0  Next move (return to start position).
IXS2  Execute sequence 2 (loop to start of this sequence).
IES  End of sequence definition.

The command 1VS2 would give:

1:
MA 2000
MR 7000
DE 1000
MA
XS 2
OK
This command will examine the read port inputs and compare them with the specified bit pattern argument. It will wait until the inputs are equal to the specified bit pattern before issuing its 'OK' response and moving on to the next command.

The bit pattern is specified as a four digit binary number of either 0, 1 or 2 characters starting with read port 4, through to 1. A 0 defines that the input must be high (+24V), a 1 defines that the input must be low (0V or open-circuit) and a 2 defines that the input is not relevant or 'don't care'. If less that four digits are specified in the argument, then the preceding ones are assumed as low (0).

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;WAabbb</td>
<td>Bit pattern</td>
<td>4 digits of 0, 1 or 2</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Example:**

```plaintext
IWA2210 Will wait until the following condition is on the read input port before continuing:
PORT: 4 3 2 1
STATE: (Ignored) (Ignored) Low High
```

This command will wait for the end of a move or delay. It will wait until any current move or delay has finished and detects the return to the idle state. The 'OK' response will not be issued until the move or delay has been completed. Therefore WE can be used to execute I/O commands after a move is complete.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;WE</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Examples:**

```plaintext
1MR4000 Move 4000 steps positive.
IWE Wait for End of above move.
1WP2220 Turn LED on (write port 1) when move has finished.
1DE1000 Delay for 1 second.
IWE Wait for End of Delay.
1WP2221 Turn LED off (write port 1).
```
This command will set the window for end of move checking. At the end of a move, when the motor comes within the **WI** range of this final target, the SE (settling time) counter counts down. When the settling time reaches zero the controller will either accept the next command or go to the *Idle* condition.

If the motor overshoots the window before the settling time reaches zero, the settling time counter is reset and started again.

**Syntax**

<ad>WInnn

**Units**

Steps

**Range**

0 to 2,147,483,647 (232)

**Initial Value**

4

**Condition Requirements**

None.

**Notes:**

Value retained on power-up.

**Responses**

OK

1 Out of Range

**Example:**

IW12 Set the Window for axis 1 to 2 steps.

---

**WP WRITE TO OUTPUT PORT**

Write to output port. The PM300 controller has four user output ports, known as write ports 1 to 4. This command will set the write port outputs to a state defined by the specified bit pattern argument. The bit pattern is specified as a four digit binary number. The digits will be either 0, 1 or 2 characters starting with write ports 4 through to 1.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;WPbbbb</td>
<td>Bit pattern</td>
<td>4 digits of 0, 1 or 2</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Format:**

Four digit binary string consisting of 0s, 1s or 2s

0 = On +24V (depending on the voltage of Write Port Vsource)

1 = Off 0V or open-circuit

2 = Don’t change

**Condition Requirements**

None.

**Notes:**

Initial state on power-up all = Off

The last write is shown on the QA page.

**Responses:**

OK

1 WP SYNTAX

**Example:**

If a PM300 on axis 1 currently has the following states on its output write ports:

<table>
<thead>
<tr>
<th>PORT</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATE:</td>
<td>on</td>
<td>on</td>
<td>off</td>
<td>off</td>
</tr>
</tbody>
</table>

1WP1200 Will set the outputs to:

<table>
<thead>
<tr>
<th>PORT</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATE:</td>
<td>off</td>
<td>on</td>
<td>on</td>
<td>on</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

18/09/95 PM300CMD DOC
**XP EXECUTE PROFILE**

This command will execute the defined Profile. The move occurs at a rate, defined in milliseconds, for each MR segment to be completed.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;XPnnn</td>
<td>millisecs.</td>
<td>1</td>
<td>65535</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

Idle.

**Responses:**

- **OK**: Command has been accepted.
- **OUT OF RANGE**: Argument is out of valid range.
- **NO PROFILE**: Profile has not been defined yet.
- **EMERGENCY STOP**: The Emergency Stop has been activated.
- **MOTOR STALLED**: Controller is aborted due to stalled motor or encoder loss.
- **TRACKING ABORT**: Controller has aborted due to a Tracking error.
- **USER ABORT**: Controller is aborted due to a user command.

**Example:**

`lXPlOO Axis 1, execute Profile. Each segment takes 100 mS.`

**XS EXECUTE SEQUENCE**

This command will start execution of a sequence. The argument selects which sequence is to be executed (0 to 7). The sequence must have already been defined with a Define Sequence DS command.

If the Execute Sequence (XS) command is encountered during a sequence, it will explicitly transfer control to the beginning of the sequence specified, whether it is the sequence already running or another sequence. It may therefore be used to make a loop type sequence or jump to any other sequence. Please note that it should not be considered as a subroutine. It is like a GOTO rather than a GOSUB.

A sequence execution may be stopped before completion, or if in a continuous loop, by a Control-C or ESCAPE command.

Control-C will stop any movement immediately, exit the sequence and return to idle.

ESCAPE will decelerate any move to a stop, exit the sequence and return to idle.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Units</th>
<th>Range</th>
<th>to</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ad&gt;XSn</td>
<td>Seq. No.</td>
<td>0</td>
<td>7</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Condition Requirements**

None.

**Responses:**

- **OUT OF RANGE**: Argument (sequence number) is out of valid range.
- **NO SEQUENCE**: Sequence specified has not been defined yet.

Other responses may be generated by commands within the sequence. At the completion of the sequence, the response to the last command is sent.

**Example:**

`lXS1 Execute sequence 1`