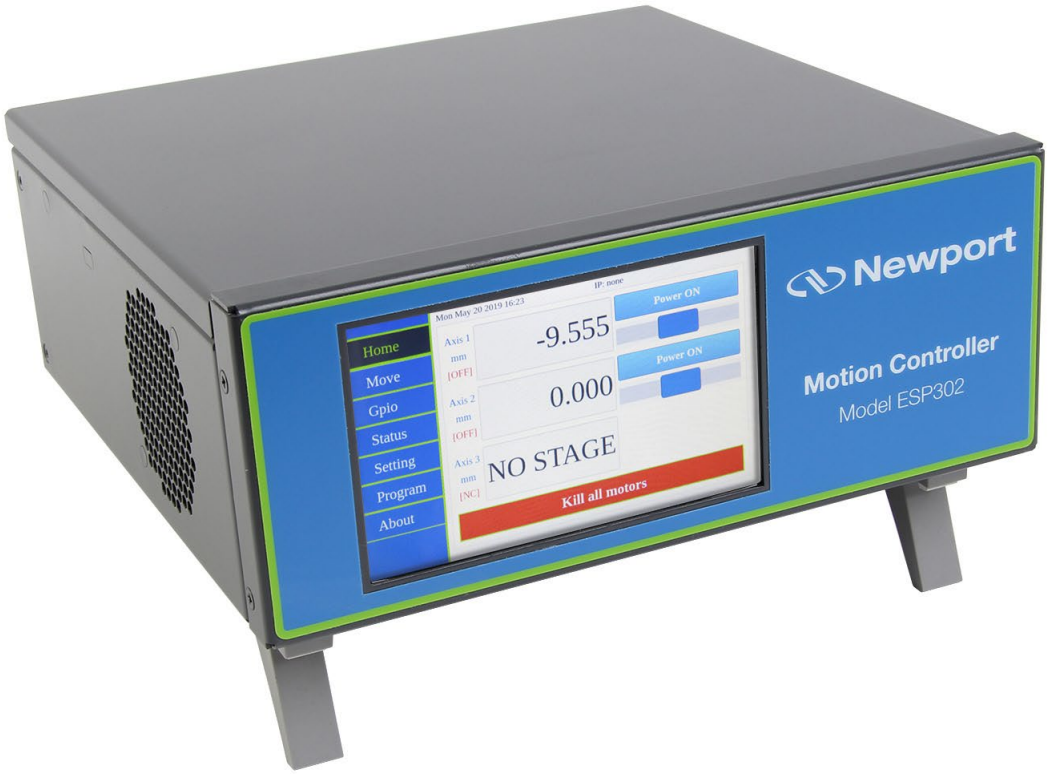
ESP302

Friendly Motion Controller/Driver



ESP COMPATIBLE

Newport™

Programmer's Manual

Table of Contents

Friendly Motion Controller/Driver ESP 302 Controller

1.0 Introduction

1.1 Scope of the Manual

This manual provides descriptions and operating procedures for the integrated 3 axis ESP302 Controller/Driver (ESP = Enhanced System Performance).

It describes the set of commands to be used in REMOTE mode (through one of the COMM., HOST or REMOTE ports) for motion, parameters, status, errors, I/Os, etc.

2.0 Remote Mode

2.1 Programming Modes

The ESP302 is a command driven system. In general, commands are a series of two letter ASCII characters preceded by an axis number and followed by parameters specific to the command. To communicate with the ESP302 controller, a host terminal has to transfer ASCII character commands according to the respective appropriate communication protocol (see section 2.2: "Remote Interfaces" for IEEE488, RS232 or USB interfaces).

As mentioned in the ESP302 User Interface Manual, tThe ESP302 distinguishes between two different programming modes:

COMMAND MODE

In this mode, the ESP302 controller provides a command input buffer enabling the host terminal (e.g., PC) to download a series of commands and then proceed to other tasks while the ESP302 controller processes the commands.

As command characters arrive from the host terminal, they are placed into the command buffer. When a carriage-return (ASCII 13 decimal) terminator is received, the command is interpreted. If the command is valid and its parameter is within the specified range, it will be executed. If the command contains an error, it will not be executed and a corresponding error message will be stored in the error buffer.

NOTE

The ESP302 power up state is command mode.

An example of a typical command sequence is shown below:

Example 1:

1PA + 30	move axis 1 to absolute position 30 units
1WS	wait for axis 1 to stop
2PR-10	move axis 2 to relative position 10 units

Assuming that axis 1 and 2 are configured, ***Example 1*** instructs the ESP controller to move axis 1 to absolute position +30 units, wait for it to stop, and then move axis 2 motor to relative –10 units.

Note that a command prefix identifies the axis or group that should execute a command. Commands received without an axis prefix generate an error. If a command is referenced to a non-existing axis, an error is also generated. See section 2.4 for further details on the command syntax.

Also note that it is necessary to explicitly instruct the ESP controller with the WS (Wait for Stop) command to wait for axis 1 motion to stop. This is necessary because the ESP controller executes commands continuously as long as there are commands in the buffer unless a command is fetched from the buffer that instructs the controller to wait. Executing a move does not automatically suspend command execution until the move is complete. If the **WS** command were not issued in ***Example 1,*** the controller would start the second move immediately after the first move begins and simultaneously move axis 1 and axis 2.

NOTE

Unless instructed otherwise, the ESP controller executes commands in the order received without waiting for completion of previous commands.

Remember that commands must be terminated with a carriage-return (ASCII 13 decimal). Until a terminator is received, characters are simply kept in contiguous buffer space without evaluation.

Example 2:

1PA+30; 1WS; 2PR-10

Example #1 and ***Example #2*** perform the same operations. In ***Example #2*** however, semicolons are used in place of carriage-returns as command delimiters, keeping the ESP302 controller from interpreting any commands on that line until the carriage-return terminator is received at the very end of the string.

PROGRAM EXECUTION MODE

The ESP302 controller also implements an internal program execution mode that enables the user to store up to 100 programs in a 64kB non-volatile memory.

Even while executing stored programs, the ESP302 controller maintains open communication channels so that the host terminal can continue to direct the ESP302 to report any desired status, and even execute other motion commands.

Let's illustrate program execution mode using the previous example:

Example 3:

EP	|	*Invoke program entry mode.*
1PA+30	|	*Enter program.*
1WS		
2PR-10		
QP	|	*Exit program entry mode.*
1EX	|	*Execute compiled program #1.*

As shown above, the sequence of commands has to be downloaded into the ESP302 controller program memory without inadvertently executing them. To facilitate this, the system provides the EP (Enter Program) command; characters received thereafter are redirected to program memory. Command syntax and parameters are not evaluated (even after the carriage-return). Instead, they are treated as a series of characters to be stored in contiguous memory.

2.2 Remote Interfaces

In this manual, *Remote Interface* refers to both communication interfaces that the controller can use to communicate with a computer or a terminal via commands in ASCII format. It is not called a *Computer Interface* since any device capable of sending ASCII characters can be interfaced with the controller.

The remote interface should not be confused with the General Purpose Input/Output (digital I/Os, a.k.a. GPIO).

2.2.1 RS-232C Interface

HARDWARE CONFIGURATION

The serial (RS-232C) communication interface on the ESP controller is accessed through the COMM. 15 pin Sub-D connector located on the rear panel.

ESP302 Start-Up manual shows the pin out of the COMM. connector that may be used to interface to a computer.

COMMUNICATION PROTOCOL

The RS-232C interface must be properly configured on both devices communicating. A correct setting is one that matches **all** parameters (baud rate, number of data bits, number of stop bits, parity type and handshake type) for both devices.

The ESP302's RS-232C configuration is fixed at **8 data bits, no parity, and 1 stop bit.**

To prevent buffer overflow when data is transferred to the ESP302 controller input buffer, a CTS/RTS hardware handshake protocol is implemented. The host terminal can control transmission of characters from the ESP302 by enabling the Request To Send (RTS) signal once the controller's Clear To Send (CTS) signal is ready. Before sending any further characters, the ESP will wait for a CTS from the host.

As soon as its command buffer is full, the controller de-asserts CTS. Then, as memory becomes available because the controller reads and executes commands in its buffer, it re-asserts the CTS signal to the host terminal.

2.2.2 TCP/IP Ethernet Interface

HARDWARE CONFIGURATION

The Ethernet (TCP/IP) communication interface on the ESP controller is accessed through one of the HOST and REMOTE connectors located on the rear panel.

The provided Ethernet straight-through standard cable can be used to connect the controller either directly to a computer or through a network.

The REMOTE port IP address is fixed (*192.168.254.254*) and the HOST port IP address (*factory default: 192.168.0.254*) can be changed through the front panel display or with the web interface (see **ESP302 Start-Up Manual**).

COMMUNICATION PROTOCOL

ESP302 controller supports 10/100/1000 Mbps Ethernet networking.

The port 5001 can be used to send serial commands through a Telnet connection.

The port 5002 is used by the .NET library (see OpenInstrument API in Command interface manual)

2.3 Software Utilities

In order to communicate with the controller, the user must have a terminal or a computer capable of communicating through RS-232C or Ethernet. One approach is to use a computer with communications software that can emulate a terminal. Windows XP provides an RS232 terminal emulation program named Hyper Terminal (HyperTrm.Exe) located in Accessories. HyperTrm allows the user to send ASCII commands to the motion controller. The user can even download text files with stored programs. Additionally, it can be used to download controller firmware for future upgrades.

2.4 Command Syntax

As mentioned previously, the ESP302 controller utilizes an ASCII command set and also outputs system status in ASCII format. Commands may be either upper or lower case characters.

The diagram below illustrates the ESP302 controller command syntax. As indicated in this diagram, a valid command consists of three main fields. The first field consists of a numerical parameter "**xx**", the second field consists of a two letter ASCII mnemonic, and the third field consists of numerical parameter "**nn**". The command is finally terminated by a carriage return. For example, 3PA10.0 is a valid command.

If a command does not require parameter "**xx**" and/or parameter "**nn**", that field may be skipped by leaving a blank character (space). For example, BO1, 3WS, and AB are all valid commands.

If a command requires multiple parameters in the third field, all these parameters must be comma delimited. For example, 1HN1,2 is a valid command.

In a similar fashion, multiple commands can be issued on a single command line by separating the commands by a semi-colon (;). For example, 3MO; 3PA10.0; 3WS; 3MF is a valid command line.

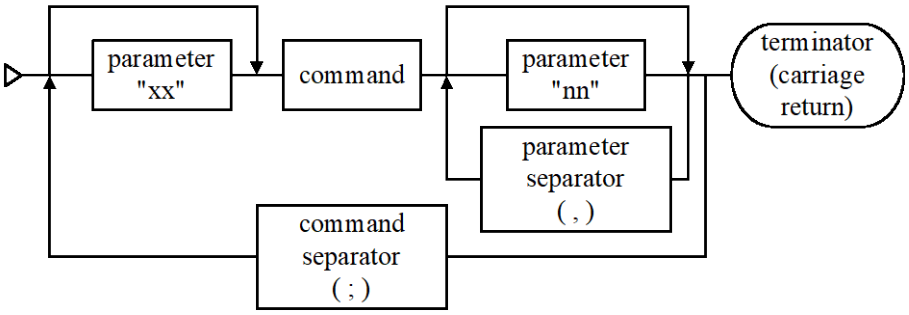


Figure 1: Command Syntax Diagram.

NOTE

A controller command (or a sequence of commands) has to be terminated with a carriage return character. However, responses from the controller are always terminated by a carriage return/line feed combination. This setting may not be changed.

2.4.1 Summary of Command Syntax

Command Format

The general format of a command is a two character mnemonic (**AA**). Both upper and lower case are accepted. Depending on the command, it could also have optional or required preceding (**xx**) and/or following (**nn**) parameters.

Blank Spaces

Blank spaces are allowed and ignored between parameters and commands. For the clarity of the program and memory saving considerations, use blank spaces with restraint. The following two commands are equivalent.

2 PA 1000

2PA1000

but the first example is very confusing and uses more than twice the memory.

Command Line

Commands are executed line by line. A line can consist of one or a number of commands. The controller will interpret the commands in the order they are received and execute them sequentially. This means that commands issued on the same line are executed significantly closer to each other than if they would be issued on separate lines. The maximum number of characters allowed on a command line is 80.

Separator

Commands issued on the same line must be separated by semicolons (**;**).

Multiple parameters issued for the same command are separated by commas (**,**).

Terminator

Each command line must end with a line terminator, i.e., carriage return.

2.5 Command Summary

The controller understands many commands. The following tables list all of them, sorted first by category and then alphabetically. The tables also show the operating modes in which each command can be used. The acronyms used in the tables have the following meaning:

IMM	IMMediate mode	Controller is idle and the commands will be executed immediately.
PGM	ProGraM mode	Controller does not execute but stores all commands as part of a program. EP activates this mode and **QP** exits it.
MIP	Motion In Progress	Controller executes command on the specified axis while in motion.

2.5.1 Command List by Category

General Mode Selection

Cmd.	Description	IMM	PGM	MIP	Page
DO	Set DAC Offset	♦	♦	♦	44
FP	Set Position Display Resolution	♦	♦	♦	52
LC	Lock/Unlock Touchscreen	♦	♦	–	88
MF	Motor OFF	♦	♦	♦	91
MK	Motor Kill	♦	♦	♦	92
MO	Motor ON	♦	♦	♦	93
QD	Update Motor Driver Settings	♦	♦	–	105
RS	Reset the Controller	♦	–	♦	115
TJ	Set Trajectory Mode	♦	♦	–	131
YZ	Set controller command terminator and echo	♦	–	–	152
ZA	Set Amplifier Configuration	♦	♦	–	154
ZB	Set Feedback Configuration	♦	♦	–	155
ZE	Set E-Stop Configuration	♦	♦	–	156
ZF	Set Following Error Configuration	♦	♦	–	157
ZH	Set Hardware Limit Configuration	♦	♦	–	159
ZS	Set Software Limit Configuration	♦	♦	–	161
ZU	Get ESP System Configuration	♦	–	♦	163
ZZ	Set System Configuration	♦	♦	–	164

Table 1: General Mode Selection.

Status Functions

Cmd.	Description	IMM	PGM	MIP	Page
DP	Read Desired Position	♦	♦	♦	45
DV	Read Desired Velocity	♦	–	♦	46
ID	Read Stage Model and Serial Number	♦	–	♦	77
MD	Read Motion Done Status	♦	–	♦	90
PH	Get Hardware Status	♦	–	♦	103
TB	Read Error Message	♦	–	♦	129
TE	Read Error Code	♦	–	♦	130
TP	Read Actual Position	♦	–	♦	132
TS	Get Controller Status	♦	–	♦	133
TV	Get Actual Velocity	♦	–	♦	136
TX	Read Controller Activity	♦	–	♦	137
VE	Read Controller Firmware Version	♦	–	♦	144

Table 2: Status Functions.

Motion & Position Control

Cmd.	Description	IMM	PGM	MIP	Page
AB	Abort Motion	♦	–	♦	17
DH	Define Home	♦	♦	–	42
MT	Move to Hardware Travel Limit	♦	♦	–	94
MV	Move Indefinitely	♦	♦	♦	95
MZ	Move to Nearest Index	♦	♦	–	96
OR	Search for Home	♦	♦	–	100
PA	Move to Absolute Position	♦	♦	♦	102
PR	Move to Relative Position	♦	♦	♦	104
ST	Stop Motion	♦	♦	♦	127

Table 3: Motion & Position Control.

Motion Device Parameters

Cmd.	Description	IMM	PGM	MIP	Page
FE	Set Maximum Following Error Threshold	♦	♦	♦	51
FR	Set Encoder Full Step Resolution	♦	♦	♦	53
QG	Set Gear Constant	♦	♦	–	106
QI	Get Maximum Motor Current	♦	♦	–	107
QM	Get Motor Type	♦	♦	–	108
QR	Get Motor Torque Reduction	♦	♦	♦	110
QS	Set Microstep Factor	♦	♦	–	111
QT	Set Tachometer Gain	♦	♦	–	112
QV	Set Average Motor Voltage	♦	♦	–	113
SL	Set Left Travel Limit	♦	♦	♦	122
SN	Set Axis Displacement Units	♦	♦	–	124
SR	Set Right Travel Limit	♦	♦	–	125
SU	Set Encoder Resolution	♦	♦	–	128

Table 4: Motion Device Parameters.

Programming

Cmd.	Description	IMM	PGM	MIP	Page
AP	Abort Program	♦	♦	♦	22
EO	Automatic Execution on Power ON	♦	–	♦	47
EP	Enter Program Mode	♦	–	–	48
EX	Execute a Program	♦	♦	–	49
LP	List Program	♦	–	♦	89
QP	Quit Program Mode	♦	–	–	109
SM	Save Settings to Non-Volatile Memory	♦	♦	–	123
XM	Read Available Program Memory	♦	–	♦	150
XX	Erase Program	♦	–	♦	151

Table 5: Programming.

Trajectory Definition

Cmd.	Description	IMM	PGM	MIP	Page
AC	Set Acceleration/deceleration	♦	♦	♦	18
AE	Get E-Stop Deceleration	♦	♦	♦	19
AG	Set Deceleration	♦	♦	♦	21
AU	Get Maximum Acceleration and Deceleration	♦	♦	♦	23
BA	Set Backlash Compensation	♦	♦	♦	24
CO	Set Linear Compensation	♦	♦	♦	34
JH	Set Jog High Speed	♦	♦	♦	78
JK	Set Jerk	♦	♦	♦	79
JW	Set Jog Low Speed	♦	♦	♦	81
OH	Set Home Search High Speed	♦	♦	♦	97
OL	Set Home Search Low Speed	♦	♦	♦	98
OM	Set Home Search Mode	♦	♦	♦	99
SH	Set Home Preset Position	♦	♦	♦	119
VA	Set Velocity	♦	♦	♦	142
VB	Set Base Velocity for Step Motors	♦	♦	♦	143
VU	Get Maximum Velocity	♦	♦	♦	146

Table 6: Trajectory Definition.

Flow Control & Sequencing

Cmd.	Description	IMM	PGM	MIP	Page
DL	Define Label	–	♦	–	43
JL	Jump to Label	–	♦	♦	80
RQ	Generate Service Request (SRQ)	♦	♦	♦	114
SA	Set Device Address	♦	♦	♦	116
WP	Wait for Position	♦	♦	♦	147
WS	Wait for Motion Stop	♦	♦	♦	148
WT	Wait	♦	♦	♦	149

Table 7: Flow Control & Sequencing.

I/O Functions

Cmd.	Description	IMM	PGM	MIP	Page
BG	Assign DIO Bits to Execute Stored Programs	♦	–	♦	25
BK	Assign DIO Bits to Inhibit Motion	♦	♦	♦	26
BL	Enable DIO Bits to Inhibit Motion	♦	♦	♦	27
BM	Assign DIO Bits to Notify Motion Status	♦	♦	♦	28
BN	Enable DIO Bits to Notify Motion Status	♦	♦	♦	29
BO	Set DIO Port Direction	♦	♦	♦	30
BP	Assign DIO Bits for Jog Mode	♦	♦	♦	31
BQ	Enable/Disable DIO Jog Mode	♦	♦	♦	32
DC	Setup Data Acquisition	♦	–	♦	36
DD	Get Data Acquisition Done Status	♦	–	♦	38
DE	Enable/Disable Data Acquisition	♦	–	♦	39
DF	Get Data Acquisition Sample Count	♦	–	♦	40
DG	Get Acquisition Data	♦	–	♦	41
SB	Set/Get DIO Port GPIO Bit Status	♦	♦	♦	117
UH	Wait for DIO Bit High	–	♦	–	140
UL	Wait for DIO Bit Low	–	♦	–	141

Table 8: I/O Functions.

Group Functions

Cmd.	Description	IMM	PGM	MIP	Page
HA	Set Group Acceleration/deceleration	♦	♦	♦	56
HB	Read Current Number of Trajectory Elements	♦	–	♦	57
HC	Move Group Along an Arc	♦	♦	♦	58
HD	Set Group Deceleration	♦	♦	♦	61
HE	Set Group E-Stop Deceleration	♦	♦	♦	62
HF	Group OFF	♦	♦	♦	63
HJ	Set Group Jerk	♦	♦	♦	64
HL	Move Group Along a Line	♦	♦	♦	65
HN	Create New Group	♦	♦	–	67
HO	Group ON	♦	♦	♦	69
HP	Read Group Position	♦	–	♦	70
HQ	Wait For Group Command Buffer Level	♦	♦	♦	71
HS	Stop Group Motion	♦	♦	♦	72
HV	Set Group Velocity	♦	♦	♦	73
HW	Wait For Group Motion Stop	♦	♦	♦	74
HX	Delete Group	♦	♦	♦	75
HZ	Read Group Size	♦	♦	♦	76

Table 9: Group Functions.

Digital Filters

Cmd.	Description	IMM	PGM	MIP	Page
AF	Set Acceleration Feed-Forward Gain	♦	♦	♦	20
CL	Set Closed Loop Update Interval	♦	♦	♦	33
DB	Set Position Deadband	♦	♦	♦	35
FV	Set Velocity Filter Frequency	♦	♦	♦	54
KD	Set Derivative Gain	♦	♦	♦	82
KF	Set Corrector Derivative Cutt Off Frequency	♦	♦	♦	83
KI	Set Integral Gain	♦	♦	♦	84
KP	Set Proportional Gain	♦	♦	♦	85
KS	Set Saturation Level of Integral Factor	♦	♦	♦	86
KT	Set Integration Time	♦	♦	♦	87
UF	Update Servo Filter	♦	♦	♦	139
VF	Set Velocity Feed-Forward Gain	♦	♦	♦	145

Table 10: Digital Filters.

Master-Slave Mode Definition

Cmd.	Description	IMM	PGM	MIP	Page
GR	Set Master-Slave Reduction Ratio	♦	♦	♦	55
SI	Set Master-Slave Jog Velocity Update Interval	♦	♦	♦	120
SK	Set Master-Slave Jog Velocity Scaling Coefficients	♦	♦	♦	121
SS	Define Master-Slave Relationship	♦	♦	–	126

Table 11: Master-Slave Mode Definition

2.5.2 Command List – Alphabetical

Cmd.	Description	IMM	PGM	MIP	Page
AB	Abort Motion	♦	–	♦	17
AC	Set Acceleration/deceleration	♦	♦	♦	18
AE	Get E-Stop Deceleration	♦	♦	♦	19
AF	Set Acceleration Feed-Forward Gain	♦	♦	♦	20
AG	Set Deceleration	♦	♦	♦	21
AP	Abort Program	♦	♦	♦	22
AU	Get Maximum Acceleration and Deceleration	♦	♦	♦	23
BA	Set Backlash Compensation	♦	♦	♦	24
BG	Assign DIO Bits to Execute Stored Programs	♦	–	♦	25
BK	Assign DIO Bits to Inhibit Motion	♦	♦	♦	26
BL	Enable DIO Bits to Inhibit Motion	♦	♦	♦	27
BM	Assign DIO Bits to Notify Motion Status	♦	♦	♦	28
BN	Enable DIO Bits to Notify Motion Status	♦	♦	♦	29
BO	Set DIO Port Direction	♦	♦	♦	30
BP	Assign DIO Bits for Jog Mode	♦	♦	♦	31
BQ	Enable DIO Bits for Jog Mode	♦	♦	♦	32
CL	Set Closed Loop Update Interval	♦	♦	♦	33
CO	Set Linear Compensation	♦	♦	♦	34
DB	Set Position Deadband	♦	♦	♦	35
DC	Setup Data Acquisition	♦	–	♦	36
DD	Get Data Acquisition Done Status	♦	–	♦	38
DE	Enable/Disable Data Acquisition	♦	–	♦	39
DF	Get Data Acquisition Sample Count	♦	–	♦	40
DG	Get Acquisition Data	♦	–	♦	41
DH	Define Home	♦	♦	–	42
DL	Define Label	–	♦	–	43
DO	Set DAC Offset	♦	♦	♦	44
DP	Read Desired Position	♦	♦	♦	45
DV	Read Desired Velocity	♦	–	♦	46
EO	Automatic Execution On Power ON	♦	–	♦	47
EP	Enter Program Mode	♦	–	–	48
EX	Execute a Program	♦	♦	–	49
FE	Set Maximum Following Error Threshold	♦	♦	♦	51
FP	Set Position Display Resolution	♦	♦	♦	52
FR	Set Encoder Full Step Resolution	♦	♦	♦	53
FV	Set Current Velocity Filter Frequency	♦	♦	♦	54
GR	Set Master-Slave Reduction Ratio	♦	♦	♦	55
HA	Set Group Acceleration/deceleration	♦	♦	♦	56
HB	Read Current Number of Trajectory Elements	♦	–	♦	57
HC	Move Group Along an Arc	♦	♦	♦	58
HD	Set Group Deceleration	♦	♦	♦	61
HE	Get Group E-Stop Deceleration	♦	♦	♦	62
HF	Group OFF	♦	♦	♦	63
HJ	Set Group Jerk	♦	♦	♦	64
HL	Move Group Along a Line	♦	♦	♦	65
HN	Create New Group	♦	♦	–	67
HO	Group ON	♦	♦	♦	69

HP	Read Group Position	♦	–	♦	70
HQ	Wait For Group Command Buffer Level	♦	♦	♦	71
HS	Stop Group Motion	♦	♦	♦	72
HV	Set Group Velocity	♦	♦	♦	73
HW	Wait for Group Motion Stop	♦	♦	♦	74
HX	Delete Group	♦	♦	♦	75
HZ	Read Group Size	♦	♦	♦	76
ID	Read Stage Model And Serial Number	♦	–	♦	77
JH	Set Jog High Speed	♦	♦	♦	78
JK	Set Jerk	♦	♦	♦	79
JL	Jump to Label	–	♦	♦	80
JW	Set Jog Low Speed	♦	♦	♦	81
KD	Set Derivative Gain	♦	♦	♦	82
KF	Set Corrector Derivative Cutt Off Frequency	♦	♦	♦	83
KI	Set Integral Gain	♦	♦	♦	84
KP	Set Proportional Gain	♦	♦	♦	85
KS	Set Saturation Level of Integral Factor	♦	♦	♦	86
KT	Set Integration Time	♦	♦	♦	87
LC	Lock/Unlock Touchscreen	♦	♦	–	88
LP	List Program	♦	–	♦	89
MD	Read Motion Done Status	♦	–	♦	90
MF	Motor OFF	♦	♦	♦	91
MK	Motor Kill	♦	♦	♦	92
MO	Motor ON	♦	♦	♦	93
MT	Move to Hardware Travel Limit	♦	♦	–	94
MV	Move Indefinitely	♦	♦	♦	95
MZ	Move to Nearest Index	♦	♦	–	96
OH	Set Home Search High Speed	♦	♦	♦	97
OL	Set Home Search Low Speed	♦	♦	♦	98
OM	Set Home Search Mode	♦	♦	♦	99
OR	Search for Home	♦	♦	–	100
PA	Move to Absolute Position	♦	♦	♦	102
PH	Get Hardware Status	♦	–	♦	103
PR	Move to Relative Position	♦	♦	♦	104
QD	Update Motor Driver Settings	♦	♦	–	105
QG	Set Gear Constant	♦	♦	–	106
QI	Get Maximum Motor Current	♦	♦	–	107
QM	Get Motor Type	♦	♦	–	108
QP	Quit Program Mode	♦	–	–	109
QR	Get Motor Torque Reduction	♦	♦	♦	110
QS	Set Microstep Factor	♦	♦	–	111
QT	Set Tachometer Gain	♦	♦	–	112
QV	Set Average Motor Voltage	♦	♦	–	113
RQ	Generate Service Request (SRQ)	♦	♦	♦	114
RS	Reset the Controller	♦	–	♦	115
SA	Set Device Address	♦	♦	♦	116
SB	Set/Get DIO Port GPIO Bit Status	♦	♦	♦	117
SH	Set Home Preset Position	♦	♦	♦	119
SI	Set Master-Slave Jog Velocity Update Interval	♦	♦	♦	120

SK	Set Master-Slave Jog Velocity Scaling Coefficients	♦	♦	♦	121
SL	Set Left Travel Limit	♦	♦	♦	122
SM	Save Settings to Non-Volatile Memory	♦	♦	–	123
SN	Set Axis Displacement Units	♦	♦	–	124
SR	Set Right Travel Limit	♦	♦	–	125
SS	Define Master-Slave Relationship	♦	♦	–	126
ST	Stop Motion	♦	♦	♦	127
SU	Set Encoder Resolution	♦	♦	–	128
TB	Read Error Message	♦	–	♦	129
TE	Read Error Code	♦	–	♦	130
TJ	Set Trajectory Mode	♦	♦	–	131
TP	Read Actual Position	♦	–	♦	132
TS	Get Controller Status	♦	–	♦	133
TV	Get Actual Velocity	♦	–	♦	136
TX	Read Controller Activity	♦	–	♦	137
UF	Update Servo Filter	♦	♦	♦	139
UH	Wait for DIO Bit High	–	♦	–	140
UL	Wait for DIO Bit Low	–	♦	–	141
VA	Set Velocity	♦	♦	♦	142
VB	Set Base Velocity for Step Motors	♦	♦	♦	143
VE	Read Controller Firmware Version	♦	–	♦	144
VF	Set Velocity Feed-Forward Gain	♦	♦	♦	145
VU	Get Maximum Velocity	♦	♦	♦	146
WP	Wait for Position	♦	♦	♦	147
WS	Wait for Motion Stop	♦	♦	♦	148
WT	Wait	♦	♦	♦	149
XM	Read Available Program Memory	♦	–	♦	150
XX	Erase Program	♦	–	♦	151
YZ	Set controller command terminator and echo	♦	–	–	152
ZA	Set Amplifier I/O Configuration	♦	♦	–	154
ZB	Set Feedback Configuration	♦	♦	–	155
ZE	Set E-Stop Configuration	♦	♦	–	156
ZF	Set Following Error Configuration	♦	♦	–	157
ZH	Set Hardware Limit Configuration	♦	♦	–	159
ZS	Set Software Limit Configuration	♦	♦	–	161
ZU	Get ESP System Configuration	♦	–	♦	163
ZZ	Set System Configuration	♦	♦	–	164

Table 12: Command List – Alphabetical.

2.6 Description of Commands

The extensive ESP302 controller command set exists to facilitate application development for wide range of application and needs. However, most simple positioning can be done with just a few commands:

VA – Set velocity.

AC – Set acceleration.

AG – Set deceleration.

PR – Position relative.

PA – Position absolute.

TP – Tell position.

WS – Wait for stop.

NOTE

Most of the commands take an axis number as a parameter (xx). For such commands, the valid range of axis number is from 1 to Max. Axes, where Max. Axes is dependent on the configuration of the ESP302 motion controller.

Commands related to coordinated motion and contouring (group commands) take a group number as a parameter. For such commands, the valid range of group number is from 1 to Max. Groups, where Max. Groups is one-half the Max. Axes.

AA — (Command mnemonic) (Brief definition) (Motor type) *

	IMM	PGM	MIP
Usage	♦	–	♦

(diamonds mark which mode the command can be used in **)

Syntax xx**AA**nn (Generic syntax format).

Parameters

Description **xx** [int] — (Description of parameter).

nn [float] — (Description of parameter).

(parameter could be **int**eger number, **float**ing point number, **char**acter or **str**ing).

Range **xx** — (**Minimum value** to **maximum value**).

nn — (**Minimum value** to **maximum value**).

Units **xx** — (Units description).

nn — (Units description).

Defaults **xx** Missing: (Default or error if parameter is missing).

Out of range: (Default or error if parameter is out of range).

nn Missing: (Default or error if parameter is missing).

Out of range: (Default or error if parameter is out of range).

Description (Detailed description of the command).

NOTE

(Notes, reminders and things to consider when using the command, if any).

Returns **(Type,** format and description of the return the command is generating, if any).

Errors (Error Code) — (Description of errors the command could generate if misused).

Rel. Commands (Command) — (Brief definition of related commands).

Example

(**Command Discussed**) | *(Description).*

(Other Command) | *(Description).*

(*Controller Return*) | *(Description).*

***(Motor type)** – If the command is specific for a motor type (DC or stepping) it will be labeled here, otherwise this field is blank.

** The mode mnemonics has the following meanings:

IMMediate mode – Controller is in idle mode and the commands are executed immediately.

Pro**G**ra**M** mode – Controller does not execute but stores all commands as part of a program.

Motion **I**n **P**rogress – Controller is executing a motion on all or the specified axis.

AB — Abort Motion

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **AB**

Parameters None.

Description This command should be used as an emergency stop. On reception of this command, the controller invokes emergency stop event processing for each axis as configured by **ZE** (e-stop event configuration) command.

By default axes are configured to turn motor power OFF, however, individual axes can be configured to stop using emergency deceleration rate set by **AE** command and maintain motor power.

It should be used only as an immediate command, not in a program.

NOTE

This command affects all axes, however the action taken is determined by each individual's axis ZE command configuration.

Returns None.

Rel. Commands

ST	—	Stop motion.
AE	—	E-stop deceleration.
ZE	—	E-stop deceleration.
MF	—	Motor OFF.
MO	—	Motor ON.

Example

AB	|	Used as an immediate command to stop all motion.

AC — Set Acceleration/deceleration

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**AC**nn or xx**AC**?

Parameters

Description

xx [int] — Axis number.

nn [float] — Acceleration value.

Range

xx — **1** to **Max. Axes**.

nn — **0** to **the maximum programmed value in AU command**, or **?** to read current setting.

Units

xx — None.

nn — Predefined units/second^2.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x11, MAXIMUM ACCELERATION EXCEEDED.

Description

This command is used to set the acceleration and deceleration value for an axis. Its execution is immediate, meaning that the acceleration is changed when the command is processed, even while a motion is in progress.

It can be used as an immediate command or inside a program. If the requested axis is a member of a group, the commanded acceleration becomes effective only after the axis is removed from the group. Refer to Advanced Capabilities section in the ESP302 Features Manual for a detailed description of grouping and related commands.

Avoid changing the acceleration during the acceleration or deceleration periods.

For better predictable results, change acceleration only when the axis is not moving or when it is moving with a constant speed.

Returns

If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

VA — Set velocity.

PA — Execute an absolute motion.

PR — Execute a relative motion.

AU? — Get maximum acceleration and deceleration.

Example

2AU? | *Read maximum allowed acceleration/deceleration of axis #2*

10 | *Controller returns a value of 10 units/s^2.*

2AC9 | *Set acceleration to 9 units/s^2.*

AE — Get E-Stop Deceleration

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**AE**?

Parameters

Description **xx** [int] — Axis number.

Range **xx** — **1** to **Max. Axes**.

Units **xx** — None.

Defaults **xx** Missing: Error 37, AXIS NUMBER MISSING.
Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description This command is used to get the e-stop deceleration value for an axis.

E-stop deceleration is invoked upon a local e-stop condition (e.g. Inhibit) has occurred, if configured to do so, or if the **AB** (abort motion) command is processed.

NOTE

E-stop deceleration value is read-only and is 10 times the normal acceleration/deceleration.

Returns This command reports the current setting.

Rel. Commands

VA	—	Set velocity.
PA	—	Execute an absolute motion.
PR	—	Execute a relative motion.
AU?	—	Get maximum acceleration and deceleration.
AC	—	Set acceleration.

Example

2AE?	|	*Read e-stop deceleration of axis #2.*
100	|	*Controller returns a value of 100 units/s^2.*

AF — Set Acceleration Feed-Forward Gain

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**AF**nn or xx**AF**?

Parameters

Description

xx [int] — Axis number.

nn [float] — Acceleration feed-forward gain factor.

Range

xx — **1** to **Max. Axes**.

nn- — **0** to **Max_Double**, or **?** to read current setting.

Units

xx — None.

nn — None.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description

This command sets the acceleration feed-forward gain factor **AF**. It is active for any DC servo based motion device.

See the "Feed-Forward Loops" section in the ESP302 Features Manual to understand the basic principles of feed-forward.

NOTE

The command can be sent at any time but it has no effect until the UF (update filter) is received.

This command is volatile, to change the parameter permanently change KFeedForwardAcceleration in the configuration file.

Returns

If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

KI — Set integral gain factor.

KD — Set derivative gain factor.

KP — Set proportional gain factor.

KS — Set saturation gain factor.

VF — Set velocity feed-forward gain.

UF — Update filter.

Example

3VF1.5 | *Set acceleration feed-forward gain factor for axis #3 to 1.5.*

3AF? | *report present axis-3 acceleration feedforward setting.*

0.9 | *Controller returns a value of 0.9*

3AF0.8 | *Set acceleration feed-forward gain factor for axis #3 to 0.8.*

3UF | *Update PID filter; only now the AF command takes effect.*

AG — Set Deceleration

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**AG**nn or xx**AG**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Acceleration value.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **the maximum programmed value in AU command**, or **?** to read current setting.

Units

xx	—	None.
nn	—	Predefined units/second^2.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x11, MAXIMUM ACCELERATION EXCEEDED.

Description Obsolete command, but kept for backward compatibility.

This command is equivalent to AC (acceleration = deceleration)

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

AC	—	Set acceleration/deceleration.

AP — Abort Program

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **AP**

xxAP

APnn

Parameters

Description

xx [int]	—	Program number.
nn [string]	—	Program task name

Range

xx	—	**1** to **2000**
nn	—.	**1** to **20** characters

Units

xx	—	None.
nn	—	None.

Description This command is used to interrupt a motion program in execution. It will not stop a motion in progress. It will only stop the program after the current command line finished executing.

It can be used as an immediate command or inside a program.

Inside a program it is useful in conjunction with program flow control commands. It could, for instance, terminate a program on the occurrence of a certain external event, monitored by an I/O bit.

If AP is sent without program number or task name, all running programs will be aborted.

Returns None.

Rel. Commands

EX	—	Execute a program.

Example

3EX	|	*Execute program #3.*
…		
…		
…		
3AP	|	*Abort program 3*
AP"P3"	|	*Abort program with task name "P3"*
AP	|	*Stop all program execution.*

AU — Get Maximum Acceleration and Deceleration

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**AU**?

Parameters

Description **xx** [int] — Axis number.

Range **xx** — **1** to **Max. Axes**.

Units **xx** — None.

Defaults **xx** Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE

Description This command is used to get the maximum acceleration and deceleration value for an axis. This parameter is read-only and can be modified only through the MaximumAcceleration parameter in configuration file.

Returns This command reports the current setting.

Rel. Commands

VA	—	Set velocity.
PA	—	Execute an absolute motion.
PR	—	Execute a relative motion.
AC	—	Set acceleration/deceleration.

Example

AU?	|	*Read maximum allowed acceleration/deceleration of axis #2.*
10	|	*Controller returns a value of 10 units/s^2*

BA — Set Backlash Compensation

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**BA**nn or xx**BA**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	backlash compensation value.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **distance equivalent to 10000 encoder counts**.

Units

xx	—	None.
nn	—	User units.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error 7, PARAMETER OUT OF RANGE.

Description This command initiates a backlash compensation algorithm when motion direction is reversed. The controller keeps track of the motion sequence and for each direction change it adds the specified **nn** correction. Setting **nn** to zero disables the backlash compensation.

NOTE

The command is affective only after a home search (OR) or define home (DH) is performed on the specified axis.

This command is volatile, to change the parameter permanently change Backlash in the configuration file.

Returns If "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands None.

Example

1BA0.0012	|	*Set backlash compensation value for axis #1 to 0.0012 units.*
1BA?	|	*Query backlash compensation value for axis #1.*
0.0012	|	*Controller returns a value of 0.0012 units.*
1OR	|	*Perform home search on axis #1.*
1PA10	|	*Move axis #1 to absolute 10 units.*
1PA0	|	*Move axis #1 to absolute 0 units.*

BG — Assign DIO Bits to Execute Stored Programs

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax xx**BG**nn or xx**BG**?

xxBGfilename,taskname

Parameters

Description

xx [int] — bit number used to trigger stored program execution.

nn [char] — number of stored program to be executed.

Range

xx — **0** to **15**.

nn — **0** to **2000** or **?** to read current setting.

filename — 1 to 250 characters

taskname — 1 to 20 characters

Units None.

Defaults

xx Missing: Error 7, PARAMETER OUT OF RANGE.

Out of range: Error 7, PARAMETER OUT OF RANGE.

Description

This command is used to assign DIO bits for initiating the execution of a desired stored program. Execution of the stored program begins when the specified DIO bit changes its state from HIGH to LOW logic level.

The trigger is disabled when program number is set to 0.

The program command has two syntaxes:

- Legacy syntaxe : xxBGnn

  This syntax is the same than ESP301 controller and allows launching a program by its number (created with EP command). The task name of a program launched with this syntax will be Pnn with nn the number of the program.

- Filename syntax : BGnn,"filename","taskname"

This syntax allows to launche any program file from /Admin/Public/Progs/ folder and to attribute a name to the program task (to kill it with AP command)

NOTE

Each DIO bit has a pulled-up resistor to +5 V. Therefore, all bits will be at HIGH logic level if not connected to external circuit and configured as input.

ESP302 commands are converted to upcase except for characters in quotation marks; it is advisable to put the filename and taskname in quotation marks.

Returns If the "?" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

BO — Set DIO port direction.

EP — Enter program mode.

EX — Execute stored program.

AP — Abort stored program execution.

Example

BO00H | *Set DIO ports A and B to input.*

0BG1 | *Start execution of a s*tored program 1 when DIO bit #0 changes state *from HIGH to LOW.*

BK — Assign DIO Bits to Inhibit Motion

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**BK**nn$_1$, nn$_2$ or xx**BK**?

Parameters

Description

xx [int]	—	Axis number.
nn$_1$ [int]	—	Bit number for inhibiting motion.
nn$_2$ [int]	—	Bit level when axis motion is inhibited.

Range

xx	—	**1** to **Max. Axes**.
nn$_1$	—	**0** to **15**.
nn$_2$	—	**0** = LOW and **1** = HIGH or **?** to read current setting.

Units None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn$_1$	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.
nn$_2$	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description This command is used to assign DIO bits for inhibiting the motion of a selected axis. If the selected axis is already in motion, and DIO bit is asserted, e-stop is executed per E-stop configuration (Refer **ZE** command for further details). If the axis is not moving, any new move commands are refused as long as the DIO bit is asserted. In either case, "DIGITAL I/O INTERLOCK DETECTED" error is generated.

NOTE

The direction of the DIO port (A, B) the desired bit belongs to, should be set to "input" in order for the DIO bit to be read accurately. Refer BO command for further details.

Returns If the "?" sign takes the place of **nn** value, this command reports the current assignment.

Rel. Commands

BL	—	Enable DIO bits to inhibit motion.
BO	—	Set DIO port direction.
BM	—	Assign DIO bits to notify motion status.

Example

BO00H	|	*Set DIO ports A, B to input.*
2BK1, 1	|	*Use DIO bit #1 to inhibit motion of axis #2. This DIO bit should be HIGH when axis #2 motion is inhibited.*
2BL1	|	*Enable inhibition of motion using DIO bits for axis #2.*
2BK?	|	*Query the DIO bit assignment for axis #2.*
1, 1	|	*The controller responds with the assigned values.*

⚠ Since this function is software dependant, it cannot be used for safety purpose. In this case, use the Inhibit port on controller rear panel.

BL — Enable DIO Bits to Inhibit Motion

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**BL**nn or xx**BL**?

Parameters

Description

xx [int] — Axis number.

nn [int] — Disable or enable.

Range

xx — **1** to **Max. Axes**.

nn — **0** = disable, and **1** = enable or **?** to read current setting.

Units None.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description This command is used to disable or enable motion inhibition of requested axes through DIO bits.

Returns If the "?" sign takes the place of **nn** value, this command reports the current status.

Rel. Commands

BK — Assign DIO bits to inhibit motion.

BO — Set DIO port direction.

BM — Assign DIO bits to notify motion status.

BN — Enable DIO bits to notify motion status.

Example

BO00H	*Set DIO ports A and B to input.*
2BK1, 1	*Use DIO bit #1 to inhibit motion of axis #2. This DIO bit should be HIGH when axis #2 motion is inhibited.*
2BL1	*Enable inhibition of motion using DIO bits for axis #2.*
2BK?	*Query the DIO bit assignment for axis #2.*
1, 1	*The controller responds with the assigned values.*
2BL?	*Query the status of inhibiting motion for axis #2 through DIO.*
1	*The controller responds with 1 indicating feature is enabled.*

BM — Assign DIO Bits to Notify Motion Status

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**BM**nn$_1$, nn$_2$ or xx**BM**?

Parameters

Description

xx [int]	—	Axis number.
nn$_1$ [int]	—	Bit number for notifying motion status.
nn$_2$ [int]	—	Bit level when axis is not moving.

Range

xx	—	**1** to **Max. Axes**.
nn$_1$	—	**0** to **15**.
nn$_2$	—	**0** = LOW and **1** = HIGH or **?** to read current setting.

Units None

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn$_1$	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.
nn$_2$	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description This command is used to assign DIO bits for notifying the motion status – moving or not moving – of a selected axis. When the selected axis is not moving, the DIO bit state changes to the level specified with this command (refer parameter nn2).

NOTE

The direction of the DIO port (A, B) the desired bit belongs to, should be set to "output" in order for the DIO bit to be set accurately. Refer BO command for further details.

NOTE

If a motion feature, such as origin search, involves a sequence of moves, the motion status will be set to not moving only after the entire sequence of moves has completed.

Returns If the "?" sign takes the place of **nn** value, this command reports the current assignment.

Rel. Commands

BN	—	Enable DIO bits to notify motion status.
BO	—	Set DIO port direction.

Example

BO02H	|	*Set DIO port GPIO1 to input and port GPIO2 to output.*
2BM9, 1	|	*Use DIO bit #9 to indicate motion status of axis #2. This DIO bit should be HIGH when axis #2 is not moving.*
2BN1	|	*Enable notification of motion using DIO bits for axis #2.*
2BM?	|	*Query the DIO bit assignment for axis #2.*
9, 1	|	*The controller responds with the assigned values.*

BN — Enable DIO Bits to Notify Motion Status

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**BN**nn or xx**BN**?

Parameters

Description

xx [int] — Axis number.

nn [int] — Disable or enable.

Range

xx — **1** to **Max. Axes**.

nn — **0** = disable, and **1** = enable or ? to read current setting.

Units None.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description This command is used to disable or enable notification of requested axis' motion status through DIO bits.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current status.

Rel. Commands

BM — Assign DIO bits to notify motion status.

BO — Set DIO port direction.

BK — Assign DIO bits to inhibit motion.

BL — Enable DIO bits to inhibit motion.

Example

BO02H	*Set DIO port GPIO1 to input and port GPIO2 to output.*
2BM9, 1	*Use DIO bit #9 to indicate motion status of axis #2. This DIO bit should be HIGH when axis #2 is not moving.*
2BN1	*Enable notification of motion using DIO bits for axis #2.*
2BM?	*Query the DIO bit assignment for axis #2.*
9, 1	*The controller responds with the assigned values.*
2BN?	*Query the status of notifying motion status of axis #2 through DIO bits.*
1	*The controller responds with 1 indicating feature is enabled.*

BO — Set DIO Port Direction

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **BO**nn or **BO**?

Parameters

Description **nn** [int] — Direction.

Range **nn** — **0** to **03H** (hexadecimal with leading zero(0)), or **?** to read current setting.

Units **nn** — None.

Defaults **nn** Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error 7, PARAMETER OUT OF RANGE.

Description This command is used to set digital I/O (DIO) port GPIO1 and GPIO2 direction where bit-0 corresponds to port GPIO1and bit-1 to port GPIO2. If any bit is set to zero(0) then its corresponding port will become an input only. If any bit is set to one(1) then its corresponding port will become an output only.

A DIO within a port configured as an input can only report its present HIGH or LOW logic level. Whereas a DIO bit within a port configured as an output can set(1) or clear(0) the corresponding DIO hardware to HIGH or LOW logic level.

Reading the status of a port configured as output returns its present output status.

NOTE

All direction bits are automatically zeroed, or cleared, after a system reset. Therefore all DIO ports turn to input by default.

NOTE

Each DIO bit has a pulled-up resistor to +5 V. Therefore, all bits will be at HIGH logic level if not connected to external circuit and configured as input.

Bit#	Definition	Meaning for	
		Bit LOW	bit HIGH
0	port GPIO1 (DIO bit-0 through bit-7) direction	INPUT	OUTPUT
1	port GPIO2 (DIO bit-8 through bit-15) direction	INPUT	OUTPUT

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting in hexadecimal notation.

Rel. Commands SB — set/clear DIO bits.

Example

BO?	|	*Read DIO port direction configuration.*
0H	|	*Controller returns a value of 0H (all ports are input).*
BO01H	|	*Configure DIO port GPIO1 as output.*
SB0FFH	|	*Set all port GPIO1 DIO output HIGH.*

BP — Assign DIO Bits for Jog Mode

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**BP**nn$_1$,nn$_{2[}$,nn$_3$] or xx**BP**?

Parameters

Description

xx [int]	—	Axis number.
nn$_1$ [int]	—	Bit number for jogging in negative direction.
nn$_2$ [int]	—	Bit number for jogging in positive direction.
nn$_3$ [int]	—	Bit number for control of jogging speed (optional).

Range

xx	—	**1** to **Max. Axes**.
nn$_i$	—	**0** to **15**.

Units

xx	—	None.
nn$_i$	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn$_i$	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description This command is used to assign DIO bits for jogging axes in either negative or positive directions (active low).

The third parameter control the jogging speed according to JH and JW parameters. When jogging speed bit is low the JH speed is selected, when it is high the JW speed is selected.

Returns If "?" sign is issued along with command, the controller returns the DIO bits used for jogging in negative and positive directions respectively.

Rel. Commands

BQ	—	Enable usage of DIO bits for jogging axes.

Example

1BP3, 4	|	*Set DIO bit #3 to jog axis #1 in negative direction and DIO bit #4 to jog axis #1 in positive direction.*
1BP?	|	*Query the DIO bits assigned for jogging*
3,4	|	*Controller returns the bit assignment.*
1BQ1	|	*Enable axis #1 jogging through DIO bits.*

BQ — Enable DIO Bits for Jog Mode

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**BQ**nn or **BQ**?

Parameters

Description

xx [int] — Axis number.

nn [int] — Disable or enable**.**

Range

xx — **1** to **Max. Axes**.

nn — **0** = disable, and **1** = enable**.**

Units

xx — None.

nn — None.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description This command is used to disable or enable jogging of a requested axis through DIO bits.

Returns If "**?**" sign is issued along with command, the controller returns the status of jog through DIO bits.

Rel. Commands BP — Assign *DIO bits for jog mode.*

Example

1BP3,4 | *Set DIO bit #3 to jog axis #1 in negative direction and DIO bit #4 to jog axis #1 in positive direction.*

1BP? | *Query the DIO bits assigned for jogging.*

3,4 | *Controller returns the bit assignment.*

1BQ1 | *Enable axis #1 jogging through DIO bits.*

CL — Set Closed Loop Update Interval

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**CL**nn or xx**CL**?

Parameters

Description

xx [int]	—	Axis number.
nn [int]	—	Closed loop update interval.

Range

xx	—	**0** to **Max. Axes**.
nn	—	**0** to **60000**.

Units

xx	—	None.
nn	—	Milliseconds.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.

Description Obsolete command, but kept for backward compatibility.
This command has no effect.

Returns If "**?**" sign takes the place of **nn** value, this command reports the ISR corrector period.

Rel. Commands

ZB	—	Set feedback configuration.
DB	—	Set position deadband value.

Example

ZB	|	*Set feedback configuration.*
DB	|	*Set position deadband value.*

CO — Set Linear Compensation

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**CO**nn or xx**CO**?

Parameters

Description

xx [int] — Axis number.

nn [float] — Linear compensation value.

Range

xx — **1** to **Max. Axes**.

nn — **-2e+9** to **2e+9**.

Units

xx — None.

nn — None.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error 7, PARAMETER OUT OF RANGE.

Description This command allows users to compensate for linear positioning errors due to stage inaccuracies. Such errors decrease or increase actual motion linearly over the travel range.

The linear compensation value, **nn** is calculated according to the formula given below:

$$nn = \frac{error}{travel}$$

where:

travel = Measured travel range.

Error = Error accumulated over the measured travel range.

NOTE

The command is affective only after a home search (OR) or define home (DH) is performed on the specified axis.

This command is volatile, to change the parameter permanently change LinearEncoderCorrection in the configuration file.

Returns If "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands None.

Example If a stage has a travel range of 100 mm and it accumulates an error of 0.003 mm over the complete travel range,

$$nn = \left(\frac{0.003}{100}\right) = 0.00003$$

1CO0.00003	|	*Set linear compensation value for axis #1 to 0.00003.*
1CO?	|	*Query linear compensation value for axis #1.*
0.00003	|	*Controller returns a value of 0.00003.*
1OR	|	*Perform home search on axis #1.*
1PA10	|	*Move axis #1 to absolute 10 units.*

DB — Set Position Deadband

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**DB**nn or xx**DB**?

Parameters

Description

xx [int]	—	Axis number.
nn [int]	—	deadband value.

Range

xx	—	**0** to **Max. Axes**.
nn	—	**0** to **Max_Long**.

Units

xx	—	None.
nn	—	Encoder counts.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.

Description Obsolete command, but kept for backward compatibility.
This command has no effect.

Returns If "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

ZB	—	Set feedback configuration.
CL	—	Set closed loop update interval.

DC — Setup Data Acquisition

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **DC**nn$_1$,nn$_2$,nn$_3$,nn$_4$,nn$_5$,nn$_6$

Parameters

Description

	nn$_1$ [int]	—	Data acquisition mode.
	nn$_2$ [int]	—	Axis used to trigger data acquisition.
	nn$_3$ [int]	—	Data acquisition parameter 3 – not used.
	nn$_4$ [int]	—	Data acquisition parameter 4.
	nn$_5$ [int]	—	Data acquisition rate.
	nn$_6$ [int]	—	Number of data samples to be acquired.

Range

nn$_1$	—	**0** = Start data acquisition immediately. **1** = Start data acquisition when trigger axis starts motion. **2** = Start data acquisition when trigger axis reaches slow speed.
nn$_2$	—	**1** to **Max. Axes**.
nn$_3$	—	**0**.
nn$_4$	—	**0** to **7**.
nn$_5$	—	**0** to **15000**.
nn$_6$	—	**1** to **1000**.

Units None.

Defaults

nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error 7, PARAMETER OUT OF RANGE.

Description This command is used to setup data acquisition — encoder position in increments — using ESP motion controller.

Parameter nn$_1$: Data acquisition modes 0—2 support different triggers to start gathering.

Parameter nn$_2$: Data acquisition is triggered by the motion of an axis specified through this parameter. Exceptions to this requirement are in the case of data acquisition mode 0. For this case enabling data acquisition is sufficient to start the data acquisition process.

For all other modes, two conditions — enabling of data acquisition and any mode dependent conditions such as trigger axis starting motion or reaching slew speed — must be met in order to start the data acquisition process.

Parameter nn$_3$: Set this value to 0.

Parameter nn4: This parameter is used to identify the encoder position feedback axes to be collected in increments. Please refer to table below.

nn4	Position feedback axes collected
0	none
1	axis 1
2	axis 2
3	axes 1 & 2
4	axis 3
5	axes 1 & 3
6	axes 2 & 4
7	axes 1,2,3

Parameter nn5: The rate at which data is to be acquired is specified through this parameter. The rate specified is in multiples of the servo rate. For example, a value of 1 (or 0) implies data acquisition every servo cycle, a value of 2 implies every other servo cycle, and so on.

Parameter nn6: The number of samples of data to be acquired is specified through this parameter. Data acquisition process is considered to be "done" only after the number of samples specified by this parameter is acquired by the controller. The status of data acquisition process may be found by issuing ASCII command **DD**. Once the data acquisition is done, ASCII command **DG** may be used to collect the data from the controller.

NOTE

The controller responds with a servo cycle tick count along with every data sample collected.

Returns None.

Rel. Commands

DD	—	Get data-acquisition done status.
DE	—	enable/disable data-acquisition
DF	—	Get data-acquisition status – number of samples collected
DG	—	Get data-acquisition data

Example

DC1,2,0,4,1,10	|	*Acquire encoder position for axis 3 as soon as the axis 2 motion starts. Collect 10 samples, one sample / servo cycle.*
DE1	|	*Enable data acquisition.*
3TP	|	*Get current position of axis 3.*
2.000	|	*Axis 3 position = 2.000.*
2PA5	|	*Start motion on axis 2.*
DD	|	*Query data-acquisition done status.*
1	|	*1 = true, 0 = false.*
If true,		
DE0	|	*Disable trace variable data acquisition.*
DF	|	*Get the current data acquisition sample count.*
10	|	*10 points are acquired in memory.*
DG	|	*Get data collected.*
N N N N N N N N N N	|	*Collected data in binary format (use an adapted software to get data).*

DD — Get Data Acquisition Done Status

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **DD**

Parameters None.

Description This command returns the status of a data acquisition request.

Returns **aa**, where:
aa = **1** for True or **0** for False.

Rel. Commands

DC	—	setup data acquisition request.
DG	—	Get acquired data.
DF	—	Data acquisition status, returns #of samples collected.
DE	—	Enable/disable data acquisition.

Example

DC1,2,0,4,1,10	|	*Acquire encoder position for axis 3 as soon as the axis 2 motion starts. Collect 10 samples, one sample / servo cycle.*
DE1	|	*Enable data acquisition.*
3TP	|	*Get current position of axis 3.*
2.000	|	*Axis 3 position = 2.000.*
2PA5	|	*Start motion on axis 2.*
DD	|	*Query data-acquisition done status.*
1	|	*1 = true, 0 = false.*
If true,		
DE0	|	*Disable trace variable data acquisition.*
DF	|	*Get the current data acquisition sample count.*
10	|	*10 points are acquired in memory.*
DG	|	*Get data collected.*
N N N N N N N N N N	|	*Collected data in binary format (use an adapted software to get data).*

DE — Enable/Disable Data Acquisition

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **DE**nn

Parameters **nn**

Description **nn** [int] — True | False.

Range **nn** — **1** for True, **0** for False.

Description This command is used to enable/disable the data acquisition request.

NOTE

This command cannot be issued when:

1. **An axis is being homed (refer ASCII command, OR).**
2. **An axis is being moved to a travel limit (refer ASCII command, MT).**
3. **An axis is being moved to an index (refer ASCII command, MZ).**

Returns None.

Rel. Commands

DC	—	Setup data acquisition request.
DG	—	Get acquired data.
DF	—	Data acquisition status, returns #of samples collected.
DD	—	Data acquisition done status.

Example

DC1,2,0,4,1,10	|	*Acquire encoder position for axis 3 as soon as the axis 2 motion starts. Collect 10 samples, one sample / servo cycle.*
DE1	|	*Enable data acquisition.*
3TP	|	*Get current position of axis 3.*
2.000	|	*Axis 3 position = 2.000.*
2PA5	|	*Start motion on axis 2.*
DD	|	*Query data-acquisition done status.*
1	|	*1 = true, 0 = false.*
If true,		
DE0	|	*Disable trace variable data acquisition.*
DF	|	*Get the current data acquisition sample count.*
10	|	*10 points are acquired in memory.*
DG	|	*Get data collected.*
N N N N N N N N N N	|	*Collected data in binary format (use an adapted software to get data).*

DF — Get Data Acquisition Sample Count

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **DF**

Parameters None.

Description This command returns the number of a data acquisition collected to the point of this request.

Returns **aa**, where:
aa = **Number of samples**.

Rel. Commands

DC	—	Setup data acquisition request.
DG	—	Get acquired data
DD	—	Data acquisition done status
DE	—	Enable/disable data acquisition
TM	—	Set trace mode
TT	—	Read trace data

Example 1

DC1,2,0,4,1,10	|	*Acquire encoder position for axis 3 as soon as the axis 2 motion starts. Collect 10 samples, one sample / servo cycle.*
DE1	|	*Enable data acquisition.*
3TP	|	*Get current position of axis 3.*
2.000	|	*Axis 3 position = 2.000.*
2PA5	|	*Start motion on axis 2.*
DD	|	*Query data-acquisition done status.*
1	|	*1 = true, 0 = false.*
If true,		
DE0	|	*Disable trace variable data acquisition.*
DF	|	*Get the current data acquisition sample count.*
10	|	*10 points are acquired in memory.*
DG	|	*Get data collected.*
N N N N N N N N N N	|	*Collected data in binary format (use an adapted software to get data).*

Example 2

SP0.001	|	*Set sample period to 1ms.*
1TM1000	|	*Acquire trace variable data for axis 1. Collect 1000 samples.*
1PR5	|	*Start a move on axis 1 to start acquisition.*
DD	|	*Query data-acquisition done status.*
1	|	*1 = true, 0 = false.*
If true,		
TT Gathering.dat	|	*Disable trace variable data acquisition.*

DG — Get Acquisition Data

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **DG**

Parameters None.

Description This command is used to retrieve data acquired from a data acquisition request.

Returns This command returns byte wide binary data. Each four bytes (b3b2b1b0) represents one DSP 32 bit word. The number of bytes returned depends on the setup request. (See **DC** command).

NOTE

If collected data are ZERO then the binary response will be NULL. In this case, the DG response cannot be seen from a command line.

Example of returned byte wide binary data after an acquisition of **10 points**:
Returned data: N N N N N N N N N N
Data length = 10 * 4 bytes = 40

```
20 4e 00 00 20 4e 00 00 20 4e 00 00 20 4e 00 00    N.. N.. N.. N..
20 4e 00 00 20 4e 00 00 20 4e 00 00 20 4e 00 00    N.. N.. N.. N..
20 4e 00 00 20 4e 00 00                            N.. N..
```

```
b0 b1 b2 b3  => (hexa)b3b2b1b0
```

```
20 4e 00 00  => (hexa)00004e20 = (decimal)20000
```

Rel. Commands

DC	—	Setup data acquisition request.
DE	—	Enable/disable data acquisition.
DF	—	Data acquisition status, returns # of samples collected.
DD	—	Data acquisition done status.

Example

DC1,2,0,4,1,10	|	*Acquire encoder position for axis 3 as soon as the axis 2 motion starts. Collect 10 samples, one sample / servo cycle.*
DE1	|	*Enable data acquisition.*
3TP	|	*Get current position of axis 3.*
2.000	|	*Axis 3 position = 2.000.*
2PA5	|	*Start motion on axis 2.*
DD	|	*Query data-acquisition done status.*
1	|	*1 = true, 0 = false.*
If true,		
DE0	|	*Disable trace variable data acquisition.*
DF	|	*Get the current data acquisition sample count.*
10	|	*10 points are acquired in memory.*
DG	|	*Get data collected.*
N N N N N N N N N N	|	*Collected data in binary format (use an adapted software to get data).*

DH — Define Home

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**DH**nn or xx**DH**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Position value.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**-2e+9** to **+2e+9**.

Units

xx	—	None.
nn	—	Predefined units.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description This command is used to define the value of the current position. This means that the current position will be preset to the value defined by parameter '**nn**'.

NOTE

Soft limits will be changed automatically to the corresponding values.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the last setting value.

Rel. Commands

OR	—	Execute a home search cycle.

Example

3OR1	|	*Perform a home search on axis #3.*
…		
…		
…		
3DH	|	*Define current position on axis #3 at 0 units*
…		
…		
…		
3DH20.0	|	*Define current position on axis #3 at 20.0 units.*

DL — Define Label

	IMM	PGM	MIP
Usage	–	♦	–

Syntax xx**DL**

Parameters

Description **xx** [int] — Label number.

Range **xx** — **1** to **100**.

Units **xx** — None.

Default **xx** Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description This command defines a label inside a program. In combination with JL (jump to label) command, they offer significant program flow control.

The operation of the DL/JL command pair is similar to commands in other computer languages that allow conditional jumps (or GOTO's) to pre-defined labels in a program.

NOTE

This command does not generate an error when not used inside a program. Since it can not do any harm, it is only ignored.

Returns None.

Rel. Commands JL — Jump to label.

Example

3XX	|	*Clear program 3 from memory, if any.*
3EP	|	*Create program 3.*
1DL	|	*Define label 1.*
…		
…		
…		
1JL 5	|	*Jump to label 1 five(5) times.*
QP	|	*End entering program and quit programming mode.*
3EX	|	*Run stored program number 3.*

DO — Set DAC Offset

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**DO**nn or xx**DO**?

Parameters

Description

xx [int]	—	DAC channel number.
nn [float]	—	DAC offset value.

Range **xx** — **1** to **Max. Axes**.

Units **xx** — None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.

Description Obsolete command, but kept for backward compatibility.
This command does nothing.

Returns If the "**?**" sign takes the place of **nn** value, this command reports 0.

Rel. Commands None.

DP — Read Desired Position

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**DP**?

Parameters

Description **xx** [int] — Axis number.

Range **xx** — **1** to **Max. Axes**.

Units **xx** — None.

Defaults **xx** Missing: Error 37, AXIS NUMBER MISSING.
Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description This command is used to read the desired position. It returns the instantaneous desired position.

The command could be sent at any time but its real use is while a motion is in progress.

Returns **nn**, where:
nn = **Desired position** in pre-defined units.

Rel. Commands
PA — Move to an absolute position.
PR — Move to a relative position.
TP — Read actual position.

Example

3TP?	|	*Read position on axis #3.*
5.32	|	*Controller returns position 5.32 for axis #3.*
3PR2.2	|	Start *a relative motion of 2.2 on axis #3.*
3DP?	|	*Read desired position on axis #3.*
7.52	|	*Controller returns desired position 7.52 for axis #3.*

DV — Read Desired Velocity

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax xx**DV**

Parameters

Description **xx** [int] — Axis number.

Range **xx** — **1** to **Max. Axes**.

Units **xx** — None.

Defaults **xx** Missing: Error 37, AXIS NUMBER MISSING.
Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description This command is used to read the desired velocity of an axis. The command can be sent at any time but its real use is while motion is in progress.

Returns **nn**, where:
nn = **Desired velocity of the axis** in pre-defined units.

Rel. Commands PA — Move to an absolute position.
PR — Move to a relative position.

Example

3TP?	|	*Read position on axis #3.*
5.32	|	*Controller returns position 5.32 units for axis #3*
3PR2.2	|	*Start a relative motion of 2.2 units on axis #3*
3DV	|	*Read desired velocity on axis #3.*
0.2	|	*Controller returns velocity 0.2 units/s for axis #3*
3DP?	|	*Read desired position on axis #3*
7.52	|	*Controller returns desired position 7.52 units for axis #3*

EO — Automatic Execution on Power ON

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax xx**EO**nn

EOnn,filename

EO?

Parameters

Description

xx [int] — Program number.

nn [int] — Number of times of execution.

Range

xx — **1** to **2000**.

nn — **1** to **Max_Long**.

filename — 1 to 250 characters

Units

xx — None.

nn — None.

Defaults None.

Description This command sets the program number that is automatically executed on power on. If **nn** is missing, the **xx** numbered program is executed once.

The program command has two syntaxes:

- Legacy syntaxe : xxEOnn

  This syntax is the same than ESP301 controller and allows launching a program by its number (created with EP command).

- Filename syntax : EOnn,"filename"

  This syntax allows to launche any program file from /Admin/Public/Progs/ folder

The task name of a program launched at boot is BOOTPROG.

NOTE

ESP302 commands are converted to upcase except for characters in quotation marks; it is advisable to put the filename in quotation marks.

Returns If the sign "**?**" takes place of **nn** value, this command reports the file name of the program that is executed on power on and the number of times of execution.

Rel. Commands

QP — Quit programming mode.

EX — Execute stored program.

AP — Abort stored program execution.

XX — Erase program.

Example

3EO	|	*Set program #3 to be executed once on power on.*
EO?	|	*Query the program number executed on power on.*
P3.txt,1	|	*Controller returns program #3 executed once on power on.*
EO	|	*Reset automatic program execution – no program is executed on power on.*

EP — Enter Program Mode

	IMM	PGM	MIP
Usage	♦	–	–

Syntax xx**EP**

Parameters

Description **xx** [int] — Program number.

Range **xx** — **1** to **127**.

Units **xx** — None.

Defaults **xx** Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error 7, PARAMETER OUT OF RANGE.

Description This command sets the controller in programming mode. All the commands following this one will not be executed immediately but stored in memory as part of program number **xx**. To exit program entry mode and return to immediate mode, use QP command.

Programs can be entered in any order. If a program already exists then it must be first deleted using XX command.

NOTE

Programs are automatically stored into in file /Admin/Public/Progs/Pxx.txt when created.

Returns None.

Rel. Commands

QP	—	Quit programming mode.
EX	—	Execute stored program.
AP	—	Abort stored program execution.
XX	—	Erase program.

Example

```
3XX   |  Clear program 3 from memory, if any.
3EP   |  Activate program mode and enter following commands as program 3
…
…
…
QP    |  End entering program and quit programming mode
3EX   |  Run stored program number 3.
```

EX — Execute a Program

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**EX**nn

EXnn,filename,taskname

EX?

Parameters

Description

xx [int] — Program number.

nn [int] — **Number** of times to execute the program.

Range

xx — **1** to **2000**.

nn — **1** to **2147385345.**

filename — 1 to 250 characters

taskname — 1 to 20 characters

Units

xx — None.

nn — None.

Defaults

xx Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error 7, PARAMETER OUT OF RANGE.

nn Missing: 1 assumed.

Out of range: Error 7, PARAMETER OUT OF RANGE.

Description

This command is used to start executing a program. When the command is received the controller executes the program line by line or according to the flow control instructions.

To stop the program execution, use the AP command.

The program command has two syntaxes:

- Legacy syntaxe : xxEXnn

  This syntax is the same than ESP301 controller and allows launching a program by its number (created with EP command). The task name of a program launched with this syntax will be Pnn with nn the number of the program.

- Filename syntax : EXnn,"filename","taskname"

This syntax allows to launche any program file from /Admin/Public/Progs/ folder and to attribute a name to the program task (to kill it with AP command)

NOTE

ESP302 commands are converted to upcase except for characters in quotation marks; it is advisable to put the filename and taskname in quotation marks.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the list of current running tasks separated by a comma.

Rel. Commands

QP — Quit programming mode.

EP — enter program mode.

AP — Abort stored program execution.

XX — Erase program.

Example 3XX | *Clear program 3 from memory, if any.*

3EP	|	*Activate program mode and enter following commands as program 3.*
…		
…		
…		
QP	|	*End entering program and quit programming mode*
3EX	|	*Run stored program number 3.*
EX?	|	*Ask current running programs*
1,P3	|	*Controller returns 1 running program with task name P3*
EX10,"MyProg.txt","MyTask"	|	*Run 10x stored program "MyProg.txt" with task name MyTask*
EX?	|	*Ask current running programs*
2,P3,MyTask	|	*Controller returns 2 running programs with tasks names P3 and MyTask*
3AP	|	*Abort program 3*
AP"MyTask"	|	*Abort program with task name "MyTask"*

FE — Set Maximum Following Error Threshold

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**FE**nn or xx**FE**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Maximum allowed following error.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to (**Max_Long * encoder resolution**), or **?** to read current setting.

Units

xx	—	None.
nn	—	Predefined units.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description This command sets the maximum allowed following error threshold for an axis. This error is defined as the difference between the real position and the theoretical position of a motion device. The real position is the one reported by the position sensing device (encoder, scale, etc.) and the theoretical position is calculated by the controller each servo cycle (100 µs). If, for any axis and any servo cycle, the following error exceeds the preset maximum allowed following error, the controller aborts motion using e-stop deceleration and turns motor power OFF.

NOTE

This command is volatile, to change the parameter permanently change FatalFollowingError in the configuration file.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

ZF	—	Set following error event configuration.

Example

3FE?	|	*Read maximum following error for axis #3.*
0.5	|	*Controller returns for axis #3 following error of 0.5 unit.*
3FE1.0	|	*Set maximum following error for axis #3 to 1 unit.*

FP — Set Position Display Resolution

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**FP**nn or xx**FP**?

Parameters

Description

xx [int]	—	Axis number.
nn [int]	—	Display resolution.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **7**, or **?** to read present setting.

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE

Description This command is used to set the display resolution of position information. For instance, if **nn** = 4, the display will show values as low as 0.0001 units. If **nn** = 7, the display will show values in exponential form. If the user units (refer SN command) are in encoder counts or stepper increments, the position information is displayed in integer form, independent of the value set by this command.

Returns If "**?**" sign takes the place of **nn** value, this command reports current setting.

Rel. Commands None.

Example

1FP?	|	*Read position display resolution for axis #1.*
4	|	*Controller returns a value of 4.*
1TP	|	*Read actual position of axis #1.*
5.0001	|	*Controller returns position value.*
1FP2	|	*Set position display resolution for axis #1 to 2.*
1TP	|	*Read actual position of axis #1.*
5.00	|	*Controller returns position value.*
1FP7	|	*Set position display resolution for axis #1 to 7.*
1TP	|	*Read actual position of axis #1.*
5.000000E+0	|	*Controller returns position value.*

FR — Set Encoder Full-Step Resolution

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**FR**nn or xx**FR**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Encoder full step resolution.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**2e-9** to **2e+9** in user defined units, or **?** to read present setting.

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description Obsolete command, but kept for backward compatibility.

This command is equivalent to SU.

Returns If "**?**" sign takes the place of **nn** value, this command reports current setting.

Rel. Commands

SU	—	Set encoder resolution.

FV — Set Current Velocity Filter Frequency

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**FV**nn or xx**FV**?

Parameters

Description

xx [int] — Axis number.

nn [float] — Filter cut off frequency.

Range

xx — **1** to **Max. Axes**.

nn — **0** to **5000**, or **?** to read current setting.

Units

xx — None.

nn — Hertz.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description This command sets the actual velocity filter cut off frequency used for the velocity returned by TV command.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands TV — Get actual velocity

Example **3FV100** | *Set velocity filter cut off frequency for axis #3 to 100Hz.*

GR — Set Master-Slave Reduction Ratio

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**GR**nn or xx**GR**?

Parameters

Description

xx [int] — Axis number.

nn [float] — Reduction ratio.

Range

xx — **1** to **Max. Axes**.

nn — **±0.000001 to ±1,000,000**.

Units

xx — None.

nn — None.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description This command sets the master-slave reduction ratio for a slave axis. The trajectory of the slave is the desired trajectory or actual position of the master scaled by reduction ratio.

NOTE

Use this command very carefully. The slave axis will have its speed and acceleration in the same ratio as the position. Also, ensure that the ratio used for the slave axis does not cause overflow of this axis' parameters (speed, acceleration), especially with ratios greater than 1.

Returns If "**?**" sign is issued along with command, the controller returns master-slave reduction ratio.

Rel. Commands SS — Define master-slave relationship.

Example

2SS1	|	*Set axis 2 to be the slave of axis 1.*
2SS?	|	*Query the master axis number for axis 2.*
1	|	*Controller returns a value of 1.*
2GR0.5	|	*Set the reduction ratio of axis 2 to 0.5.*
2GR?	|	*Query the reduction ratio of axis 2.*
0.5	|	*Controller returns a value of 0.5.*

HA — Set Group Acceleration/deceleration

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HA**nn or **HA**?

Parameters

Description **nn** [float] — Vector acceleration value.

Range **nn** — **0** to **minimum of the maximum acceleration values of all axes assigned to this group**.

Units **nn** — Predefined units/second^2.

Defaults

nn Missing: Error 7, PARAMETER OUT OF RANGE.
Negative: Error 30, GROUP PARAMETER OUT OF RANGE.
Out of range: Error 32, GROUP MAXIMUM ACCELERATION EXCEEDED.

Description This command is used to set the vectorial acceleration and deceleration value for a group. This value will be used during coordinated motion of axes assigned to the group. It will override any original acceleration values specified for individual axes using AC command. The axes' original values will be restored when the group to which they have been assigned is deleted.

This command takes effect immediately. It can be executed when controller is idling or motion is in progress or inside a program.

NOTE

Avoid changing acceleration during acceleration or deceleration phases of a move. For better predictable results, change acceleration only when all the axes assigned to this group are not in motion.

Returns If "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

AU — Set maximum acceleration and deceleration for an axis.
HN — Create a new group.

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
1AU?	|	*Query maximum acceleration of axis #1.*
50	|	*Controller returns a value of 50 units/second^2.*
2AU?	|	*Query maximum acceleration of axis #2.*
60	|	*Controller returns a value of 60 units/second^2.*
HA50	|	*Set vectorial acceleration of the group to 50 units/second^2.*
HA?	|	*Query vectorial acceleration of the group.*
50	|	*Controller returns a value of 50 units/second^2.*

HB — Read Current Number of Trajectory Elements

	IMM	PGM	MIP
Usage	♦		♦

Syntax **HB**

Parameters None.

Defaults If no group has been created,

controller returns error number 15, GROUP NUMBER NOT ASSIGNED.

Description This command is used to read the current trajectory elements in progress.

Returns This command reports the current trajectory element in progress and the total number of elements in trajectory buffer.

Rel. Commands

HN	—	Create a new group.
HX	—	Delete a group.
HC	—	Move group along an arc
HL	—	Move group along a line.

Example

1HN1,2	|	*Create a new group (#1) with physical axes 1 and 2.*
1HN?	|	*Read axes assigned to group #1.*
1,2	|	*Controller returns the axes assigned to group #1.*
HL5,5;HC0,10,180;HL-10,10;HC-5,5,180	|	*Execute several trajectory elements.*
HB	|	*Read current trajectory elements in progress.*
1,4	|	*Controller returns the current trajectory element in progress (element #1 here) and the number of elements in trajectory buffer (4 elements here).*

HC — Move Group Along an Arc

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HC**nn_1,nn_2,nn_3 or **HC**?

Parameters

Description

nn_1 [float] — First coordinate of arc center.

nn_2 [float] — Second coordinate of arc center.

nn_3 [float] — Arc sweep angle.

Range

nn_1, nn_2 — Any position within the travel limits.

nn_3 — Any angle.

Units

nn_1, nn_2 — Predefined units.

nn_3 — Degrees.

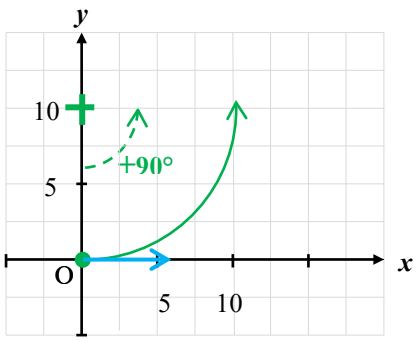
Defaults

nn_i Missing: Error 29, GROUP PARAMETER MISSING.

Description

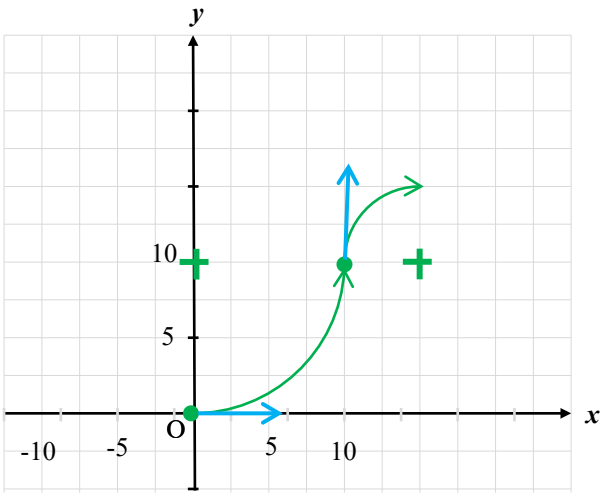
This command initiates motion of a group along an arc. It causes all axes assigned to the group to move with predefined vectorial (tangential) velocity, acceleration and deceleration along an arc. The group target position is determined based on the position of axes at the beginning of move, center of arc and sweep angle.

Example: **HC0,10,90**



If this command is received while a group move is in progress, the new command gets enqueued into a "via point" buffer. Please refer to Advanced Capabilities section in the ESP302 Features Manual for a detailed description of via point buffer implementation. The enqueued commands get executed on a FIFO basis when the move already in progress has reached its destination. The group does not come to a stop at the end of last move. Instead, there will be a smooth transition to the new move command, just as if it were one compound move (combination of multiple moves).

The starting angle, to accept an arc element, must be inferior to 1.15 °.



NOTE

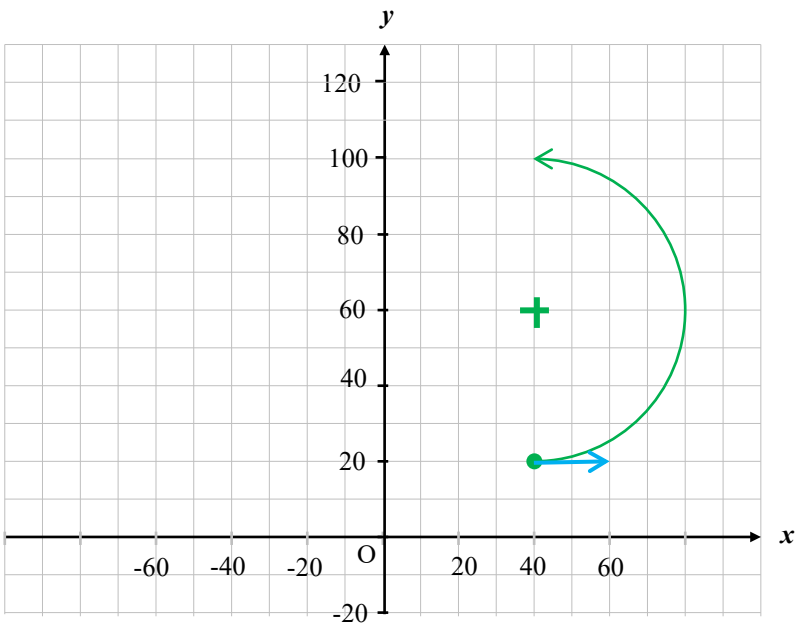
The transition from last move to new move will be smooth if tangential velocity at the end of last move is the same as that at the beginning of new move.

Returns If "**?**" sign takes the place of **nn** values, this command reports the commanded center position of arc and sweep angle.

Rel. Commands

HN	—	Create a new group.
HV	—	Set vectorial velocity for a group.
HA	—	Set vectorial acceleration and deceleration for a group.
HO	—	Enable a group.
HF	—	Disable a group.
HL	—	Move a group of axes to desired position along a line.

Example	HN1,2	|	*Create a new group with physical axes 1 and 2.*
	HV10	|	*Set vectorial velocity of group to 10 units/second.*
	HA50	|	*Set vectorial acceleration of group to 50 units/second^2.*
	HO	|	*Enable group.*
	HP?	|	*Query current group position.*
	40,20	|	*Controller returns axis #1 = 40 units and axis #2 = 20 units.*
	HC40,60,180	|	*Set axis #1 arc center = 40 units.* *Set axis #2 arc center = 60 units.* *Set sweep angle of arc = 180 degrees.*
	HC?	|	*Query target position of the commanded move.*
	40, 60, 180	|	*Controller returns axis #1 arc center = 40 units, axis #2 arc center = 60 units and arc sweep angle = 180 degrees.*
	HP?	|	*Query current group position.*
	40,100	|	*Controller returns axis #1 = 40 units and axis #2 = 100 units.*



HD — Set Group Deceleration

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HD**nn or **HD**?

Parameters

Description **nn** [float] — Vector deceleration value.

Range **nn** — **0** to **minimum of the maximum deceleration values of all axes assigned to this group**.

Units **nn** — Predefined units/second^2.

Defaults **nn** Missing: Error 7, PARAMETER OUT OF RANGE.
Negative: Error 30, GROUP PARAMETER OUT OF RANGE.
Out of range: Error 33, GROUP MAXIMUM DECELERATION EXCEEDED.

Description Obsolete command, but kept for backward compatibility.
This command is equivalent to HA (acceleration = deceleration).

Returns If "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands
AU — Set maximum acceleration and deceleration for an axis.
HN — Create a new group.
HA — Set vectorial acceleration and deceleration for a group.

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
1AU?	|	*Query maximum deceleration of axis #1.*
50	|	*Controller returns a value of 50 units/second^2.*
2AU?	|	*Query maximum deceleration of axis #2.*
60	|	*Controller returns a value of 60 units/second^2.*
HD50	|	*Set vectorial deceleration of group to 50 units/second^2.*
HD?	|	*Query vectorial deceleration of group.*
50	|	*Controller returns a value of 50 units/second^2.*

HE — Get Group E-Stop Deceleration

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HE**?

Parameters None

Description This command is used to get the vectorial e-stop deceleration value for a group. This value will be used during coordinated motion of axes assigned to the group.

E-stop deceleration is invoked upon a local e-stop condition (e.g., Inhibit) has occurred, if configured to do so, or if the AB (abort motion) command is processed.

NOTE

E-stop deceleration value is read-only and is 10 times the normal vector acceleration/deceleration.

Returns This command reports the current setting.

Rel. Commands

HN	—	Create a new group.
HV	—	Set vectorial velocity for a group.
HA	—	Set vectorial acceleration for a group.

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
HE?	|	*Query vectorial e-stop deceleration of group.*
100	|	*Controller returns a value of 100 units/second^2.*

HF — Group OFF

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HF** or **HF**?

Parameters None

Description This command turns power OFF of all axes assigned to a group. Refer to **MF** command to turn the power OFF of individual axes. The group power is assumed to be OFF if power to anyone of the axes in the group is OFF.

Returns If "**?**" sign is issued along with command, the controller returns:

1	—	Group power is ON
0	—	Group power is OFF.

Rel. Commands

HN	—	Create a new group.
HO	—	Turn group power ON.

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
HO	|	T*urn group power ON.*
HF?	|	*Query group power status.*
1	|	*Controller returns a value of 1.*
HF	|	Turn *group power OFF.*
HF?	|	*Query group power status.*
0	|	*Controller returns a value of 0.*

HJ — Set Group Jerk

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HJ**nn or **HJ**?

Parameters

Description **nn** [float] — Vector jerk time value.

Range **nn** — **0** to **2e9**.

Units **nn** — Seconds.

Description Obsolete command, but kept for backward compatibility.

This command has no effect.

Vectorial jerk time = vectorial velocity / vectorial acceleration / 2.

Returns If "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

HN	—	Create a new group.
HV	—	Set vectorial velocity for a group.
HA	—	Set vectorial acceleration and deceleration for a group.
HK	—	Set vectorial e-stop jerk for a group.

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
HJ?	|	*Query vectorial deceleration of group.*
0.05	|	*Controller returns a value of 0.05 second.*

HL — Move Group Along a Line

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HL**nn_1, nn_2 or **HL**?

Parameters

Description

nn_1 [float] — Target position of first axis in the group.

nn_2 [float] — Target position of second axis in the group.

Range **nn_i** — Any position within the travel limits.

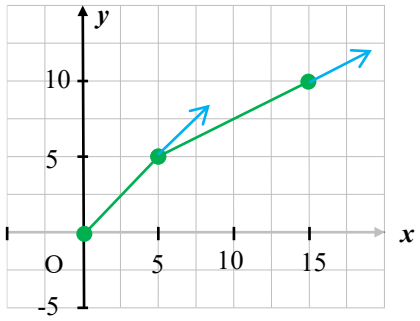
Units **nn_i** — Predefined units.

Defaults **nn_i** Missing: Error 29, GROUP PARAMETER MISSING.

Description This command initiates motion of a group along a line. It causes both axes assigned to the group to move with predefined vectorial (tangential) velocity, acceleration and deceleration along a line.

If this command is received while a group move is in progress, the new command gets enqueued into a "via point" buffer. Please refer to Advanced Capabilities section in the ESP302 Features Manual for a detailed description of via point buffer implementation. The enqueued commands get executed on a FIFO basis when the move already in progress has reached its destination. The group does not come to a stop at the end of last move. Instead, there will be a smooth transition to the new move command, just as if it were one compound move (combination of multiple moves). The allowed angle discontinuity to accept to chain two lines is set to 45°.

Example: HL5,5;HL15,10

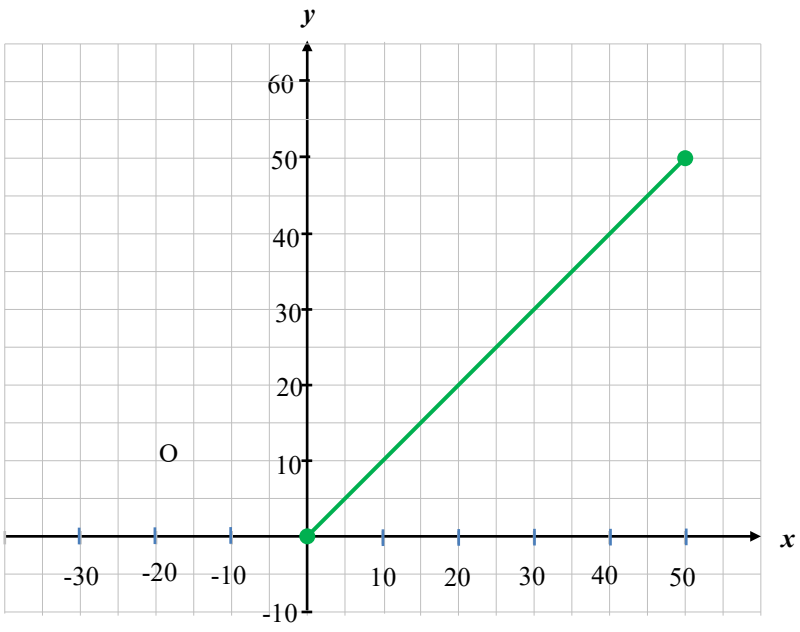


NOTE

The transition from last move to new move will be smooth if tangential velocity at the end of last move is the same as that at the beginning of new move.

Returns If "**?**" sign takes the place of **nn** values, this command reports the target positions of axes assigned to the group.

Rel. Commands	HN	—	Create a new group.
	HV	—	Set vectorial velocity for a group.
	HA	—	Set vectorial acceleration and deceleration for a group.
	HO	—	Enable a group.
	HF	—	Disable a group.
	HC	—	Move a group of axes to desired position along an arc.
Example	HN1,2	|	*Create a new group with physical axes 1 and 2.*
	HV10	|	*Set vectorial velocity of group to 10 units/second.*
	HA50	|	*Set vectorial acceleration of group to 50 units/second^2.*
	HO	|	*Enable group.*
	HP?	|	*Query current group position.*
	0,0	|	*Controller returns axis #1 = 0 units and axis #2 = 0 units.*
	HL50, 50	|	*Move axis #1 to a target position = 50 units.* *Move axis #2 to a target position = 50 units.*
	HL?	|	*Query target position of the commanded move.*
	50,50	|	*Controller returns axis #1 = 50 units and axis #2 = 50 units.*



HN — Create New Group

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax **HN**nn_1, nn_2 or **HN**?

Parameters

Description

nn_1 [int]	—	Physical axis number to be assigned as first axis in this group.
nn_2 [int]	—	physical axis number to be assigned as second axis in this group.

Range	**nn_i**	— **1** to **Max. Axes**.
Units	**nn_i**	— None.
Defaults	**nn_i** Missing:	Error 29, GROUP PARAMETER MISSING.
	Out of range:	Error 17, GROUP AXIS OUT OF RANGE.
	Already assigned:	Error 16, GROUP NUMBER ALREADY ASSIGNED.
	Duplicated:	Error 19, GROUP AXIS DUPLICATED.

Description This command is used to create a new group of two axes. A few rules are in place to facilitate easy management of groups.

- A group has to be created with two axes assigned to it before any command related to groups can be issued. The controller returns error 15, GROUP NUMBER NOT ASSIGNED, if, for instance, one tries to set group velocity before creating a group.
- A group has to be deleted (refer **HX** command) before axes assigned to the group can be changed. The controller returns error 16, GROUP NUMBER ALREADY ASSIGNED, if one attempts to change axes assigned to a group already created. Please see the following table for correct method to change axes assigned to a group:

Correct Method	Incorrect Method
HN1,2	HN1,2
HX	HN2,3
HN2,3	

- An axis cannot be assigned more than once in a group. The controller returns error 19, GROUP AXIS DUPLICATED, if one attempts to assign an axis more than once to a group.
- The order in which axes are assigned to a group is very important. This is because it specifies the frame of reference in which coordinated motion of axes takes place. For instance, the command **HN1,2** assigns axis numbers 1 and 2 to the group, where axis #1 is equivalent to X-axis and axis #2 is equivalent to Y-axis in a traditional cartesian coordinate system. Reversing the ordering of axes (viz. **HN2,1**) reverses the axis assignment.

Returns If "**?**" sign takes the place of **nn** values, this command reports the axes assigned to the group in the order of their assignment.

Rel. Commands

HV	—	Set vectorial velocity for a group.
HA	—	Set vectorial acceleration and deceleration for a group.
HO	—	Enable a group.
HF	—	Disable a group.
HC	—	Move a group of axes to desired position along an arc.
HL	—	Move a group of axes to desired position along a line.

Example	**HN1,2**	|	*Create a new group with physical axes 1 and 2.*
	HN?	|	*Query axis assigned to the group.*
	1,2	|	*Controller returns the axes assigned to the group.*
	HN2,3	|	Try c*reating a new group with physical axes 2 and 3.*
	HN?	|	*Query axis assigned to the group.*
	1,2	|	*Controller returns the axes assigned to the group.*
	TB?	|	*Read error message.*
0, 450322, GROUP NUMBER ALREADY ASSIGNED		|	*Controller returns error message.*
	HX	|	Delete *group.*
	HN2,3	|	*Create a new group with physical axes 2 and 3.*
	HN?	|	*Query axis assigned to the group.*
	2,3	|	*Controller returns the axes assigned to the group.*

HO — Group ON

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HO** or **HO**?

Parameters None

Description This command turns power ON of all axes assigned to a group. Refer **MO** command to turn the power ON of individual axes. The group power is assumed to be ON if power to all axes in the group is ON.

Returns If "**?**" sign is issued along with command, the controller returns:

1	—	Group power is ON.
0	—	Group power is OFF.

Rel. Commands

HN	—	Create a new group.
HF	—	Turn group power OFF

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
HO	|	Turn *group power ON.*
HO?	|	*Query group power status.*
1	|	*Controller returns a value of 1.*
HF	|	Turn *group power OFF.*
HO?	|	*Query group power status.*
0	|	*Controller returns a value of 0.*

HP — Read Group Position

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **HP**

Parameters None

Description This command is used to read the instantaneous real position of all axes assigned to a group.

Returns **nn_1, nn_2** where:
nn_1 = **Actual position of *1st* axis** in the group.
nn_2 = **Actual position of *2nd* axis** in the group.

Rel. Commands

HN	—	Create a new group.
HC	—	Move a group of axes to desired position along an arc.
HL	—	Move a group of axes to desired position along a line.

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
HP	|	*Read position of the group.*
10,50	|	*Controller returns axis #1 = 10 units, axis #2 = 50 units.*

HQ — Wait for Group Command Buffer Level

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HQ**nn or **HQ**?

Parameters

Description **nn** [float] — Level in group via point buffer.

Range **nn** — **1** to **100** (default for maximum targets in via point buffer).

Units **nn** — None.

Defaults **nn** Missing: Error 29, GROUP PARAMETER MISSING.

Description This command stops enqueuing new commands into the via point buffer until the buffer level equals **nn**. As commands in the buffer get executed on a FIFO basis and the buffer level equals **nn**, commands issued subsequent to this one get executed.

Returns If "**?**" sign takes the place of **nn** value, the controller returns the room available in via point buffer for more commands.

Rel. Commands

HN	—	Create a new group.
HL	—	Move group to target position along a line.
HC	—	Move group to target position along an arc.

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
HV10	|	*Set vectorial velocity of group to 10 units/second.*
HA50	|	*Set vectorial acceleration of group to 50 units/second^2.*
HO	|	*Enable group.*
HL10,10	|	*Move group to target pos. 10,10 (ax. #1 = 10, #2 = 10 units).*
HL20,20	|	*Move group to target pos. 20,20 (ax. #1 = 20, #2 = 20 units). This command gets enqueued in the via point buffer if it was received prior completion of the previous move command.*
HL50,50	|	*Move group to target pos. 50,50 (ax. #1 = 50, #2 = 50 units).*
HQ10	|	*Wait until the via point buffer level equals 10 commands.*
HC40,60,180	|	*Move group along an arc with center of arc at (40,60) units, by a sweep angle of 180 deg. from current position.*

HS — Stop Group Motion

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HS** or **HS**?

Parameters None

Description This command stops the motion of all axes assigned to a group using vector deceleration set using **HD** command.

Returns If "**?**" sign is supplied along with the command, the controller returns:

1	—	Group motion is stopped
0	—	Group motion is in progress.

Rel. Commands

HN	—	Create a new group.
HC	—	Move a group of axes to desired position along an arc.
HL	—	Move a group of axes to desired position along a line.

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
HV10	|	*Set vectorial velocity of group to 10 units/second.*
HA50	|	*Set vectorial acceleration of group to 50 units/second^2.*
HO	|	*Enable group.*
HP?	|	*Query current group position.*
0,0	|	*Controller returns axis #1 = 0 units and axis #2 = 0 units.*
HL50, 50	|	*Move axis #1 to a target position = 50 units.* *Move axis #2 to a target position = 50 units.*
HS?	|	*Query if motion of group is stopped.*
0	|	*Controller returns 0, meaning group is in motion.*
HS	|	Stop *motion of group.*
HS?	|	*Query if motion of group is stopped.*
1	|	*Controller returns 1, meaning group #1 motion has stopped.*
HP?	|	*Query current group position.*
27,26	|	*Controller returns axis #1 = 27 units and axis #2 = 26 units.*

HV — Set Group Velocity

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HV**nn or **HV**?

Parameters

Description **nn** [float] — Vector velocity value.

Range **nn** — **0** to **minimum of the maximum velocity values of all axes assigned to this group**.

Units **nn** — Predefined units/second.

Defaults

nn	Missing:	Error 7, PARAMETER OUT OF RANGE.
	negative:	Error 30, GROUP PARAMETER OUT OF RANGE.
	Out of range:	Error 31, GROUP MAXIMUM VELOCITY EXCEEDED.

Description This command is used to set the vectorial velocity value for a group. This value will be used during coordinated motion of axes assigned to the group. It will override any original acceleration values specified for individual axes using **VA** command. The axes' original values will be restored when the group to which they have been assigned is deleted.

This command takes effect immediately. It can be executed when controller is idling or motion is in progress or inside a program.

NOTE

Avoid changing velocity during acceleration or deceleration phases of a move. For better predictable results, change velocity only when all the axes assigned to this group are not in motion.

Returns If "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

VU	—	Set maximum velocity for an axis.
HN	—	Create a new group.

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
1VU?	|	*Query maximum velocity of axis #1.*
10	|	*Controller returns a value of 10 units/second.*
2VU?	|	*Query maximum velocity of axis #2.*
15	|	*Controller returns a value of 15 units/second.*
HV10	|	*Set vectorial velocity of group to 10 units/second.*
HV?	|	*Query vectorial velocity of group.*
10	|	*Controller returns a value of 10 units/second.*

HW — Wait for Group Motion Stop

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HW**nn

Parameters

Description	**nn** [float]	—	Delay after group motion is complete.
Range	**nn**	—	**0** to **60000**.
Units	**nn**	—	Milliseconds.
Defaults	**nn** Missing:		Error 7, PARAMETER OUT OF RANGE.
	negative:		Error 30, GROUP PARAMETER OUT OF RANGE.
	Out of range:		Error 26, MAXIMUM WAIT DURATION EXCEEDED.

Description This command stops execution of any commands subsequent to it until the one prior to it has been completed. For instance, if a command preceding it is a group move command such as **HL** or **HC**, it stops execution of any commands following it until the group has reached target position. If **nn** is not equal to zero, the controller waits an additional **nn** milliseconds after the group motion is complete before executing any further commands.

Returns None.

Rel. Commands	HN	—	Create a new group.
	HL	—	Move group to target position along a line.

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
HV10	|	*Set vectorial velocity of group to 10 units/second.*
HA50	|	*Set vectorial acceleration of group to 50 units/second^2.*
HO	|	*Enable group.*
HL50, 50; **HW500**; HL60,70	|	*Move group to a target position = 50, 50 units (axis #1 = 50 units and axis #2 = 50 units), wait for the group to reach target position, wait an additional 500 ms, and then move group to a target position = 60, 70 units (axis #1 = 60 units and axis #2 = 70 units).*

HX — Delete Group

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HX**

Parameters None.

Description This command deletes the group and makes available any axes that were assigned to it for future assignments.

Returns None.

Rel. Commands HN — Create a new group.

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
HN?	|	*Query axes assigned to group.*
1,2	|	*Controller returns the axes assigned to group.*
HX	|	*Delete group.*
HN?	|	*Query axis assigned to group.*
TB?	|	*Read error message.*
0, 475322, GROUP NUMBER NOT ASSIGNED	|	*Controller returns error message.*

HZ — Read Group Size

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **HZ**

Parameters None.

Description This command is used to read the number of axes assigned to a group.

Returns This command reports the current setting.

Rel. Commands

HN	—	Create a new group.
HX	—	Delete a group.

Example

HN1,2	|	*Create a new group with physical axes 1 and 2.*
HN?	|	*Read axes assigned to group.*
1,2	|	*Controller returns the axes assigned to group.*
HZ	|	*Read size of group.*
2	|	*Controller returns 2.*
HX	|	*Delete group.*
HZ	|	*Read size of group.*
TB?	|	*Read error message.*
0, 475322, GROUP NUMBER NOT ASSIGNED	|	*Controller returns error message.*

ID — Read Stage Model and Serial Number

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax xx**ID**?

Parameters

Description **xx** [int] — Axis number.

Range **xx** — **1** to **Max. Axes**.

Units **xx** — None.

Defaults **xx** Missing: Error 37, AXIS NUMBER MISSING.
Out of range: Error 9, AXIS NUMBER OUT OF RANGE.
Timeout: Error 2, RS-232 COMMUNICATION TIME-OUT.

Description This command is used to read Newport ESP compatible positioner (stage) model and serial number.

NOTE

An important information needed when asking for help with the motion control system or when reporting a problem is the stage model and serial number. Use this command to determine the positioner model and serial number.

Returns **nn$_1$,nn$_2$,nn$_3$**

where: **nn$_1$** = model number.
nn$_2$ = serial number.
nn$_3$ = configuration section from stage database file.

Rel. Commands None.

Example **1ID?** | *Read axis-1 positioner model and serial number.*
UTS50PP,SNB189401,UTS@UTS50PP@XPS-DRV11 | *Controller returns stage model, serial number and configuration section from stage database file.*

JH — Set Jog High Speed

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**JH**nn or xx**JH**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	High speed value.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **JogMaximumVelocity**, or **?** to read present setting.

Units

xx	—	None.
nn	—	Preset units/second.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x10, MAXIMUM VELOCITY EXCEEDED.

Description This command is used to set the high speed for jogging an axis. Its execution is immediate, meaning that the value is changed when the command is processed, including when motion is in progress. It can be used as an immediate command or inside a program.

NOTE

This command is volatile, to change the parameter permanently change JogMaximumVelocity in the configuration file.

Returns If "**?**" sign takes the place of **nn** value, this command reports current setting.

Rel. Commands

JW	—	Set jog low speed.
VU	—	Set maximum velocity.

Example

2VU?	|	*Read maximum velocity allowed axis #2.*
10	|	*Controller returns a value of 10.0 units/second for axis #2.*
2JH7.5	|	*Set jog high speed to 7.5 units/second for axis #2.*
2JH?	|	*Read jog high speed value for axis #2.*
7.5	|	*Controller returns a value of 7.5 units/second for axis #2.*

JK — Set Jerk

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**JK**nn or xx**JK**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Jerk value.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **2e9**.

Units

xx	—	None.
nn	—	Seconds. or **?** to read current setting.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error xx15, MAXIMUM JERK EXCEEDED.

Description Obsolete command, but kept for backward compatibility.

This command has not effect.

Returns If "?" sign takes the place of **nn** value, this command reports the jerk time value (= current velocity / current acceleration / 2)

.

Rel. Commands

AC	—	Set acceleration.	VA	—	Set velocity.

Example

2JK?	|	*Read jerk time value of axis #2.*
0.05	|	*Controller returns a jerk time value of 0.05 seconds.*

JL — Jump to Label

	IMM	PGM	MIP
Usage	–	♦	♦

Syntax xx**JL**nn

Parameters

Description

xx [int]	—	Label number.
nn [int]	—	Loop count.

Range

xx	—	**1** to **100**.
nn	—	**1** to **65535**.

Units

xx	—	None.
nn	—	None.

Default

xx	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.
nn	Missing:	Assume infinite.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description This command changes the flow of the program execution by jumping to a predefined label **xx**. This is a flow control command that alters the normal sequential flow of a program. It must be used in conjunction with the DL command which defines a label.

Parameter **nn** determines the number of times to repeat the jump before allowing the program flow to go ahead.

Returns None.

Rel. Commands

DL	—	Define label.

Example

3XX	|	*Clear program 3 from memory, if any.*
3EP	|	*Create program 3*
1DL	|	*Define label 1.*
…		
…		
…		
1JL5	|	*Jump to label 1 five(5) times.*
QP	|	*End entering program and quit programming mode.*
3EX	|	*Run stored program number 3.*

JW — Set Jog Low Speed

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**JW**nn or xx**JW**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Low speed value.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **JogMaximumVelocity**, or **?** to read present setting.

Units

xx	—	None.
nn	—	Preset units/second.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x10, MAXIMUM VELOCITY EXCEEDED.

Description This command is used to set the low speed for jogging an axis. Its execution is immediate, meaning that the value is changed when the command is processed, including when motion is in progress. It can be used as an immediate command or inside a program.

NOTE

This command is volatile, to change the parameter permanently change JogMaximumVelocity in the configuration file.

At startup, JW= JogMaximumVelocity/10

Returns If "**?**" sign takes the place of **nn** value, this command reports current setting.

Rel. Commands

JH	—	Set jog high speed.
VU	—	Set maximum velocity.

Example

2VU?	|	*Read maximum velocity allowed axis #2.*
10	|	*Controller returns a value of 10.0 units/second for axis #2*
2JW2.5	|	*Set jog low speed to 2.5 units/second for axis #2.*
2JW?	|	*Read jog low speed value for axis #2.*
2.5	|	*Controller returns a value of 2.5 units/second for axis #2.*

KD — Set Derivative Gain

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**KD**nn or xx**KD**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Derivative gain factor Kd.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **Max_Double**, or **?** to read current setting.

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description This command sets the derivative gain factor Kd of the PID closed loop. It is active for any DC servo based motion device that has been selected to operate in closed loop.

The command can be sent at any time but it has no effect until the **UF** (update filter) is received.

See the "Servo Tuning" section in the ESP302 Features Manual for a detailed description of how to adjust the PID filter parameters.

NOTE

This command is volatile, to change the parameter permanently change KD in the configuration file.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

KI	—	Set integral gain factor.
KP	—	Set proportional gain factor.
KS	—	Set saturation gain factor.
UF	—	Update filter.

Example

3KD0.01	|	*Set derivative gain factor for axis #3 to 0.01.*
…		
…		
…		
3UF	|	*Update PID filter; only now the KD command takes effect.*

KF — Set Corrector Derivative Cutt Off Frequency

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**KF**nn or xx**KF**?

Parameters

Description

xx [int] — Axis number.

nn [float] — Derivative cut off frequency.

Range

xx — **1** to **Max. Axes**.

nn — **0** to **5000**, or **?** to read current setting.

Units

xx — None.

nn — Hertz.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description

This command sets the derivative filter cut off frequency of the PID closed loop. It is active for any DC servo based motion device that has been selected to operate in closed loop. A value equal to zero disables the filter.

The command can be sent at any time but it has no effect until the **UF** (update filter) is received.

See the "Servo Tuning" section in the ESP302 Features Manual for a detailed description of how to adjust the PID filter parameters.

NOTE

This command is volatile, to change the parameter permanently change DerivativeFilterCutOffFrequency in the configuration file.

Returns

If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

KD — Set derivative gain factor.

KI — Set integral gain factor.

KP — Set proportional gain factor.

KS — Set saturation gain factor.

UF — Update filter.

Example

3KF4000 | *Set derivative filter cut off frequency for axis #3 to 4KHz.*

…

…

…

3UF | *Update PID filter; only now the KF command takes effect.*

KI — Set Integral Gain

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**KI**nn or xx**KI**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	integral gain factor.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **Max_Double**, or **?** to read current setting**.**

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description This command sets the integral gain factor Ki of the PID closed loop. It is active for any DC servo based motion device that has been selected to operate in closed loop.

The command can be sent at any time but it has no effect until the **UF** (update filter) is received.

See the "Servo Tuning" section in the ESP302 Features Manual for a detailed description of how to adjust the PID filter parameters.

NOTE

This command is volatile, to change the parameter permanently change KI in the configuration file.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

KD	—	Set integral gain factor.
KP	—	Set proportional gain factor.
KS	—	Set saturation gain factor.
UF	—	Update filter.

Example

3KI0.01	|	*Set integral gain factor for axis #3 to 0.01.*
…		
…		
…		
3UF	|	*Update PID filter; only now the KI command takes effect.*

KP — Set Proportional Gain

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**KP**nn or xx**KP**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Proportional gain factor Kp.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **Max_Double**, or **?** to read current setting**.**

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description

This command sets the proportional gain factor Kp of the PID closed loop. It is active for any DC servo based motion device that has been selected to operate in closed loop.

The command can be sent at any time but it has no effect until the **UF** (update filter) is received.

See the "Servo Tuning" section in the ESP302 Features Manual for a detailed description of how to adjust the PID filter parameters.

NOTE

This command is volatile, to change the parameter permanently change KP in the configuration file.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

KI	—	Set integral gain factor.
KD	—	Set proportional gain factor.
KS	—	Set saturation gain factor.
UF	—	Update filter.

Example

3KP0.01	|	*Set proportional gain factor for axis #3 to 0.01.*
…		
…		
…		
3UF	|	*Update PID filter; only now the KP command takes effect.*

KS — Set Saturation Level of Integral Factor

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**KS**nn or xx**KS**?

Parameters

Description

xx [int] — Axis number.

nn [float] — Saturation level of integrator.

Range

xx — **1** to **Max. Axes**.

nn — **0** to **1**, or **?** to read current setting**.**

Units

xx — None.

nn — None (coefficient; 1=100%).

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description

This command sets the saturation level of the integral factor of the PID closed loop and is useful for preventing integral wind-up. It is active for any DC servo based motion device that has been selected to operate in closed loop.

The command can be sent at any time but it has no effect until the **UF** (update filter) is received.

See the "Servo Tuning" section in the ESP302 Features Manual for a detailed description of how to adjust the PID filter parameters.

NOTE

This command is volatile, to change the parameter permanently change KS in the configuration file.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

KI — Set integral gain factor.

KP — Set proportional gain factor.

KD — Set derivative gain factor.

UF — Update filter.

Example

3KS0.01 | *Set saturation level for axis #3 to 0.01.*

…

…

…

3UF | *Update PID filter; only now the KS command takes effect.*

KT — Set Integration Time

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**KT**nn or xx**KT**?

Parameters

Description

xx [int] — Axis number.

nn [float] — Integration time of integrator.

Range

xx — **1** to **Max. Axes**.

nn — **0** to **Max_Double**, or **?** to read current setting**.**

Units

xx — None.

nn — Seconds.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description

This command sets the integration time of the PID closed loop integrator and is useful for preventing integral wind-up. It is active for any DC servo based motion device that has been selected to operate in closed loop.

The command can be sent at any time but it has no effect until the **UF** (update filter) is received.

See the "Servo Tuning" section in the ESP302 Features Manual for a detailed description of how to adjust the PID filter parameters.

NOTE

This command is volatile, to change the parameter permanently change IntegrationTime in the configuration file.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

KI — Set integral gain factor.

KP — Set proportional gain factor.

KD — Set derivative gain factor.

UF — Update filter.

Example

2KT3 | *Set integration time for axis #2 to 3 seconds.*

…

…

…

3UF | *Update PID filter; only now the KT command takes effect.*

LC — Lock/Unlock Touchscreen

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax **LC**nn or **LC**?

Parameters

Description **nn** [int] — Lock option.

Range **nn** — **0** – **2** or **?** to read current setting.

Units **nn** — None.

Defaults **nn** Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description This command is used to lock/unlock the touchscreen of the ESP302. The parameter value means:

0 = Unlock the touchscreen.

1 = Lock all buttons but "Motor ON/OFF".

2 = Lock all buttons.

Returns If the "?" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

Example

LC1	|	*Lock the touchscreen except "Motor ON/OFF".*
LC?	|	*Get lock status.*
1	|	*Returns current setting.*

LP — List Program

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax xx**LP**

Parameters

Description **xx** [int] — Program number.

Range **xx** — **1** to **127**.

Units **xx** — None.

Defaults **xx** Missing: Error 38, COMMAND PARAMETER MISSING.
Out of range: Error 7, PARAMETER OUT OF RANGE.

Description This command reads a specified program from non-volatile memory.

NOTE

The program list always terminates with the word "END".

Returns Program listing.

Rel. Commands EP — Enter program mode.

Example

3LP	|	*List program number 3.*
3MO	|	*Enable axis 3 motor power.*
1DL	|	*Define return label 1.*
3PR+10	|	*Move axis 3 relative +10 units.*
3WS500	|	*Wait 500 ms after axis 3 stops.*
3PR-10	|	*Move axis 3 relative -10 units.*
3WS500	|	*Wait 500 ms after axis 3 stops.*
1JL5	|	*Jump to label 1 location 5 times.*
END	|	*End of program list.*

MD — Read Motion Done Status

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax xx**MD**?

Parameters

Description	**xx** [int]	—	Axis number.
Range	**xx**	—	**1** to **Max. Axes**.
Units	**xx**	—	None.
Defaults	**xx** Missing:		Error 37, AXIS NUMBER MISSING.
	Out of range:		Error 9, AXIS NUMBER OUT OF RANGE.

Description This command is used to read the motion status for the specified axis **xx**. The **MD** command can be used to monitor Homing, absolute, and relative displacement move completion status.

Returns	**nn**	—	**0** or **1**, where: 0 = **Motion <u>NOT</u> done** (FALSE). 1 = **Motion done** (TRUE).
Rel. Commands	PA	—	Move to an absolute position.
	PR	—	Move to a relative position.
	OR	—	Move to home position.

Example	**3MD?**	*Read axis #3 move done status.*
	1	*Controller returns status 1 (motion done) for axis #3.*
	3PR2.2	*Start a relative motion of 2.2 on axis #3.*
	3MD?	*Read axis #3 move done status.*
	0	*Controller returns status 0 (motion not done) for axis #3.*

MF — Motor OFF

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**MF** or xx**MF**?

Parameters

Description **xx** [int] — Axis number.

Range **xx** — **1** to **Max. Axes**.

Units **xx** — None.

Defaults **xx** Missing: Error 37, AXIS NUMBER MISSING.
Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description This command turns power OFF of the specified motor (axis).
If the MF command is sent with no axis parameter, all axes are powered OFF.

Returns If "**?**" sign is issued along with command, the controller returns:
1: Motor power is ON.
0: Motor power is OFF.

Rel. Commands
AB — Abort motion.
ST — Stop motion.
MO — Turn motor power ON.

Example

2MF	*Turn axis #2 motor power OFF.*
2MF?	*Query axis #2 motor power status.*
0	*Controller returns a value of 0.*
2MO	Turn *axis #2 motor power ON.*
2MF?	*Query axis #2 motor power status.*
1	*Controller returns a value of 1.*

MK — Motor Kill

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**MK** or xx**MK**?

Parameters

Description **xx** [int] — Axis number.

Range **xx** — **1** to **Max. Axes**.

Units **xx** — None.

Defaults **xx** Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description This command turns power OFF of the specified motor (axis), and cancel the position origin done with OR command.
If the MK command is sent with no axis parameter, all axes are killed.

Returns If "**?**" sign is issued along with command, the controller returns:

1: Motor power is ON.

0: Motor power is OFF.

Rel. Commands

AB	—	Abort motion.
ST	—	Stop motion.
MO	—	Turn motor power ON.
MF	—	Turn motor power OFF.

Example

2MO	|	Turn *axis #2 motor power ON.*
2OR	|	Execute Axis 2 Home search.
2TS?	|	*Query axis #2 status.*
B@	|	*Controller returns Origine Done & Motor ON*
2MK	|	*Kill axis #2.*
2TS?	|	*Query axis #2 status.*
P@	|	*Controller returns Origine not Done & Motor OFF*

MO — Motor ON

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**MO** or xx**MO**?

Parameters

Description **xx** [int] — Axis number.

Range **xx** — To **Max. Axes**.

Units **xx** — None.

Defaults **xx** Missing: Error 37, AXIS NUMBER MISSING.
Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description This command turns power ON of the specified motor (axis).
If the MO command is sent with no axis parameter, all axes are powered ON.

CAUTION

If the motor power was turned off by the controller detecting a fault condition, before turning the power back on, make sure that the cause of the fault was corrected.

Returns If "**?**" sign is issued along with command, the controller returns:
1: Motor power is ON.
0: Motor power is OFF.

Rel. Commands AB — Abort motion.
ST — Stop motion.
MF — Turn motor power OFF.

Example

MO	|	*Turn axis #2 motor power ON.*
2MO?	|	*Query axis #2 motor power status.*
1	|	*Controller returns a value of 1.*
2MF	|	*Turn axis #2 motor power OFF.*
2MO?	|	*Query axis #2 motor power status.*
0	|	*Controller returns a value of 0.*

MT — Move to Hardware Travel Limit

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**MT**nn or xx**MT**?

Parameters

Description

xx [int]	—	Axis number.
nn [char]	—	Direction of motion.

Range

xx	—	**1** to **Max. Axes**.
nn	—	+ for positive direction or – for negative direction.

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Positive direction.

Description This command is used to move an axis to its limit (positive or negative). It uses the home search speed during travel to hardware limit.

NOTE

Software travel limits have to be disabled with ZS command to allow a move to hardware limits.

Returns If "**?**" sign takes the place of **nn** value, this command reports **1** if motion is done, or **0** if motion is in progress.

Rel. Commands

OR	—	Home location search.
OH	—	Set home search speed.
ZS	—	Set software limits configuration.

Example

3MT+	|	*Move axis #3 to positive travel limit.*
3MT?	|	*Query motion status.*
0	|	*Controller returns 0 indicating motion is in progress.*

MV — Move Indefinitely

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**MV**nn or xx**MV**?

Parameters

Description

xx [int] — Axis number.

nn [char] — Direction of motion.

Range

xx — **1** to **Max. Axes**.

nn — + for positive direction or – for negative direction.

Units

xx — None.

nn — None.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE

nn Missing: Positive direction.

If a limit is reached:

Out of range: Error x04, POSITIVE HARDWARE LIMIT DETECTED.

Out of range: Error x05, NEGATIVE HARDWARE LIMIT DETECTED.

Out of range: Error x06, POSITIVE SOFTWARE LIMIT DETECTED.

Out of range: Error x07, NEGATIVE SOFTWARE LIMIT DETECTED.

Description This command initiates infinite motion. When received, the selected axis **xx** will move indefinitely, with the predefined acceleration and velocity, in the direction specified by **nn**. If the requested axis is member of a group, this command does not initiate the desired motion. Instead, error x31, "COMMAND NOT ALLOWED DUE TO GROUP ASSIGNMENT" is generated. Refer HL and HC commands to move along a line or an arc.

NOTE

Although the command is accepted while a motion is in progress, care should be taken not to reverse direction of motion.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the motion done status.

Rel. Commands

PA — Move to absolute position.

PR — Move to relative position.

ST — Stop motion.

MD — Move done status.

Example

3MV+	*Move axis #3 indefinitely in positive direction.*
3MV?	*Query status of move.*
0	*Controller returns 0 meaning, motion is in progress.*
3ST	*Stop axis #3 motion.*
3MV–	*Move axis #3 indefinitely in negative direction.*

MZ — Move to Nearest Index

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**MZ**nn or xx**MZ**?

Parameters

Description

xx [int]	—	Axis number.
nn [char]	—	Direction of motion.

Range

xx	—	**1** to **Max. Axes**.
nn	—	+ for positive direction or – for negative direction.

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Positive direction.

Description This command is used to move an axis to its nearest index (in positive or negative direction). It uses the home search speed during travel to nearest index.

Returns If "**?**" sign takes the place of **nn** value, this command reports **1** if motion is done, or **0** if motion is in progress.

Rel. Commands

OR	—	Home location search.
OH	—	Set home search speed.

Example

3MZ+	|	*Move axis #3 to nearest index in positive direction.*
3MZ?	|	*Query motion status.*
0	|	*Controller returns 0 indicating motion is in progress.*

OH — Set Home Search High Speed

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**OH**nn or xx**OH**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	high speed value.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **maximum value allowed by VU command**, or **?** to read present setting.

Units

xx	—	None.
nn	—	Preset units/second.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x10, MAXIMUM VELOCITY EXCEEDED.
		Error x24, SPEED OUT OF RANGE.

Description This command sets the high speed used to search for home location for an axis. Its execution is immediate, meaning that the value is changed when the command is processed, including when motion is in progress. It can be used as an immediate command or inside a program.

NOTE

This command is volatile, to change the parameter permanently change HomeSearchMaximumVelocity in the configuration file.

Returns If "**?**" sign takes the place of **nn** value, this command reports current setting.

Rel. Commands

OR	—	Search for home.
OL	—	Set home search low speed.

Example

3OH10	|	*Set home search high speed of axis #3 to 10 units/s.*
3OH?	|	*Query home search high speed of axis #3.*
10	|	*Controller returns a value of 10.0 units/second.*

OL — Set Home Search Low Speed

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**OL**nn or xx**OL**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	low speed value.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **maximum value allowed by OH command**, or ? to read present setting.

Units

xx	—	None.
nn	—	Preset units/second.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x10, MAXIMUM VELOCITY EXCEEDED.
		Error x24, SPEED OUT OF RANGE.

Description This command sets the low speed used to search for home location for an axis. Its execution is immediate, meaning that the value is changed when the command is processed, including when motion is in progress. It can be used as an immediate command or inside a program.

NOTE

This command is volatile, to change the parameter permanently change HomeSearchMaximumVelocity in the configuration file.

At startup, OL= HomeSearchMaximumVelocity / 2.

Returns If "**?**" sign takes the place of **nn** value, this command reports current setting.

Rel. Commands

OR	—	Search for home.
OH	—	Set home search high speed.
OL	—	Set home search low speed.

Example

3OL2	|	*Set home search low speed of axis #3 to 2 units/s.*
3OL?	|	*Query home search low speed of axis #3.*
2	|	*Controller returns a value of 2 units/second.*

OM — Set Home Search Mode

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**OM**nn

Parameters

Description

xx [int]	—	Axis number.
nn [int]	—	Home search mode.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **6.**

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description

This command selects the home search type without invoking the home search sequence (see the description of **OR** command for more information on home search). The seven home search types are +0 Position Count, Home Switch and Index Signals, Home Switch Signal, Positive Limit Signal, Negative Limit Signal, Positive Limit and Index Signals and Negative Limit and Index Signals.

If **nn** = 0 and the front panel HOME search push button is pressed, the axes will search for zero position count. If **nn** = 1 and the front panel HOME search push button is pressed, the axis will search for combined Home and Index signal transitions. The controller responds similarly for other values of **nn**.

The **nn** parameter is overwritten by the **OR** command parameter.

NOTE

This command is volatile, to change the parameter permanently change HomeSearchSequenceType in the configuration file.

Returns If "**?**" sign takes the place of **nn** value, this command reports current setting.

Rel. Commands

OR	—	Search for home.

Example

3OM1	|	*Set axis #3 home search mode to 1.*
3OR	|	*Start home search on axis #3 using mode 1.*

OR — Search for Home

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**OR**nn

Parameters

Description

xx [int] — Axis number.

nn [int] — Home mode**.**

Range

xx — **0** to **Max. Axes**.

nn — **0** to **6** where:

0 = **Find +0 Position Count**.

1 = **Find Home and Index Signals**.

2 = **Find Home Signal**.

3 = **Find Positive Limit Signal**.

4 = **Find Negative Limit Signal**.

5 = **Find Positive Limit and Index Signals**.

6 = **Find Negative Limit and Index Signals**.

Units

xx — None.

nn — None.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Out of range: Error x01, PARAMETER OUT OF RANGE.

Description

This command executes a Home search routine on the axis specified by **xx**. If **xx** = 0, a home search routine is initiated sequentially on all installed axes. If **nn** is missing, the axis will search for home using the mode specified using **OM** command. If **nn** = 0, the axis will search for zero position count. If **nn** = 1, the axis will search for combined Home and Index signal transitions. If **nn** = 2, the axis will search for Home signal transition only. If **nn** = 3, the axis will search for positive limit signal transition. If **nn** = 4, the axis will search for negative limit signal transition. If **nn** = 5, the axis will search for positive limit and index signal transition. If **nn** = 6, the axis will search for negative limit and index signal transition.

At the end of a home search routine, the position of axes is reset to the value specified using SH command.

The home search motion status can be monitored with the Motion Done (MD) status command. If a fault condition such as E-stop occurs while home search is in progress or if this command is issued to an axis before enabling it, the controller returns error x20, "HOMING ABORTED".

For a detailed description of the home search routine see the Home Search chapter in the Motion Control Tutorial section in the ESP302 Features Manual.

NOTE

This command should be executed once every time the controller power is turned ON or the controller performs a complete system reset. There is no need to issue this command in any other case since the controller always keeps track of position, even when the motor power is OFF.

Returns None.

Rel. Commands	DH	—	Define home.
	OH	—	Set home search speed.
	OM	—	Set home search mode.
	MD	—	Read motion done status.
	SH	—	Set home preset position.
Example	3MO	|	*Turn axis #3 motor power ON.*
	3SH0	|	*Set axis #3 home position to 0 units.*
	3OR1	|	*Perform a home search on axis #3.*
	3MD?	|	*Query axis #3 motion status.*
	1	|	*Controller returns a value of 1, when motion is done.*
	3TP	|	*Query axis #3 position.*
	0	|	*Controller returns a value of 0 units.*

PA — Move to Absolute Position

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**PA**nn or xx**PA**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Absolute position destination.

Range

xx	—	**1** to **Max. Axes**.
nn	—	Any position within the travel limits and within ±**Max_Long * encoder resolution**.

Units

xx	—	None.
nn	—	Defined motion units.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x04, POSITIVE HARDWARE LIMIT EXCEEDED.
	Out of range:	Error x05, NEGATIVE HARDWARE LIMIT EXCEEDED.
	Out of range:	Error x06, POSITIVE SOFTWARE LIMIT EXCEEDED.
	Out of range:	Error x07, NEGATIVE SOFTWARE LIMIT EXCEEDED.

Description This command initiates an absolute motion. When received, the selected axis **xx** will move, with the predefined acceleration and velocity, to the absolute position specified by **nn**. If the requested axis is member of a group, this command does not initiate the desired motion. Instead, error x31, " COMMAND NOT ALLOWED DUE TO GROUP ASSIGNMENT" is generated. Refer HL and HC commands to move along a line or an arc.

NOTE

Even though the command is accepted while a motion is in progress, care should be taken not to reverse direction of motion. When this command is received, the controller verifies if it will produce a change of direction.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current position; the same as TP?

Rel. Commands

AC	—	Set acceleration/deceleration.
PR	—	Move to relative position.
ST	—	Stop motion.
MD	—	Move done status.
VA	—	Set velocity.

Example

3VA8	|	*Set velocity of axis #2 to 8 units/s.*
3PA12.34	|	*Move axis #2 to absolute position 12.34.*

PH — Get Hardware Status

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **PH**

Parameters None.

Description This command is used to get general hardware status for all axes. This routine allows user to observe the various digital input signals as they appear to the controller.

Hardware Status Register #1

BIT#	DEFINITION	Meaning for	
		BIT LOW	BIT HIGH
0	axis 1 +hardware travel limit	NO	YES
1	axis 2 +hardware travel limit	NO	YES
2	axis 3 +hardware travel limit	NO	YES
3			
4			
5			
6			
7			
8	axis 1 -hardware travel limit	NO	YES
9	axis 2 -hardware travel limit	NO	YES
10	axis 3 -hardware travel limit	NO	YES
11			
12			
13			
14			
15			
16	axis 1 motor fault	NO	YES
17	axis 2 motor fault	NO	YES
18	axis 3 motor fault	NO	YES

Hardware Status Register #2

BIT#	DEFINITION	Meaning for	
		BIT LOW	BIT HIGH
0	axis 1 home signal	NO	YES
1	axis 2 home signal	NO	YES
2	axis 3 home signal	NO	YES
3			
4			
5			
6			
7			
8	axis 1 index signal	NO	YES
9	axis 2 index signal	NO	YES
10	axis 3 index signal	NO	YES

Returns This command reports the current status in hexadecimal notation.

Rel. Commands ZU — Get ESP system configuration.

ZZ — Get system configuration.

Example **PH** | *Read hardware status.*

18000404H, 4H | *Controller returns the status of the two hardware regsisters.*

PR — Move to Relative Position

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**PR**nn

Parameters

Description

xx [int]		—	Axis number.
nn [float]		—	Relative motion increment.

Range

xx		—	**1** to **Max. Axes**.
nn		—	Any value that will not cause exceeding the software limits and within **Max_Long * encoder resolution**.

Units

xx		—	None.
nn		—	Defined motion units.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x04, POSITIVE HARDWARE LIMIT EXCEEDED.
	Out of range:	Error x05, NEGATIVE HARDWARE LIMIT EXCEEDED.
	Out of range:	Error x06, POSITIVE SOFTWARE LIMIT EXCEEDED.
	Out of range:	Error x07, NEGATIVE SOFTWARE LIMIT EXCEEDED.

Description

This command initiates a relative motion. When received, the selected axis **xx** will move, with the predefined acceleration and velocity, to relative position **nn** units away from the current position. If the requested axis is member of a group, this command does not initiate the desired motion. Instead, error x31, "COMMAND NOT ALLOWED DUE TO GROUP ASSIGNMENT" is generated. Refer **HL** and **HC** commands to move along a line or an arc.

NOTE

Even though the command is accepted while a motion is in progress, care should be taken not to reverse direction of motion.

NOTE

Successive relative moves not multiple of encoder resolution can lead to cumulative error due to position rounding.

Returns None.

Rel. Commands

AC	—	Set acceleration/deceleration.
PA	—	Move to absolute position.
MD	—	Move done status.
ST	—	Stop motion.
VA	—	Set velocity.

Example

3VA8	|	*Set velocity of axis #3 to 8 units/s.*
3PR2.34	|	*Move axis #3 2.34 units away from the current position.*

QD — Update Motor Driver Settings

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**QD**

Parameters

Description	**xx** [int]	—	Axis number.
Range	**xx**	—	**1** to **Max. Axes**.
Units	**xx**	—	None.

Description Obsolete command, but kept for backward compatibility.

This command has no effect.

Returns None.

QG — Set Gear Constant

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**QG**nn or xx**QG**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	gear constant.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **2e9**, or **?** to read present setting.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.

Description Obsolete command, but kept for backward compatibility.

This command has no effect.

Returns If the "**?**" sign takes the place of **nn** value, this command reports 0.

QI — Get Maximum Motor Current

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**QI**?

Parameters

Description **xx** [int] — Axis number.

Range **xx** — **1** to **Max. Axes**.

Units **xx** — None.

Description This command is read-only and is used to get the maximum motor current output for axis xx.

This parameter can be changed in the configuration file:

- PeakCurrentPerPhase for stepper motors
- CurrentLimit for others

Returns This command reports the current setting nn in Amperes.

Rel. Commands QM — Get motor type.

Example

2QI?	|	*Read maximum motor current setting of axis #2.*
1.6	|	*Controller returns a value of 1.6 Amp. for axis #2.*

QM — Get Motor Type

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**QM**?

Parameters

Description **xx** [int] — Axis number.

Range **xx** — 1 to Max. Axes.

Units **xx** — None.

Defaults **xx** Missing: Error 37, AXIS NUMBER MISSING.
Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description This command is read-only and returns the motor type for the axis xx.

Returns **aa**, **0** to **2** where:
0 = motor type undefined (default)
1 = DC motor
2 = stepper motor

Rel. Commands QI — Set maximum motor current.

Example

2QM?	|	*Read motor type of axis #2.*
0	|	*Controller returns a value of 0 (motor undefined) for axis #2.*

QP — Quit Program Mode

	IMM	PGM	MIP
Usage	♦	–	–

Syntax **QP**

Parameters None.

Description This command quits the controller from programming mode. All the commands following this one will be executed immediately.

Returns None.

Rel. Commands

EX	—	Execute stored program.
AP	—	Abort stored program execution.
XX	—	Erase program.

Example

3XX	|	*Clear program 3 from memory, if any.*
3EP	|	*Activate program mode and enter following commands as program 3.*
…		
…		
…		
QP	|	*End entering program and quit programming mode.*
3EX	|	*Run stored program number 3.*

QR — Get Motor Torque Reduction

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**QR**?

Parameters

Description **xx** [int] — Axis number.

Range **xx** — **1** to **Max. Axes**.

Units **xx** — None.

Defaults **xx** Missing: Error 37, AXIS NUMBER MISSING.
Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description This command is read-only and returns the motor's current (i.e., torque) reduction percentage **nn$_2$** after motion has stopped and the time **nn$_1$** has expired. The purpose of this command is to help reduce the motor heating typically generated by stepper motors.

The current reduction percentage can be changed in the configuration file with StandbyPeakCurrentPerPhase:

nn2 = 100 * StandbyPeakCurrentPerPhase / ScalingCurrent

nn1 is a constant equal to 5000ms.

NOTE

For motors other than stepper this command returns 0,0.

Returns **nn$_1$**, **nn$_2$** where:

nn$_1$ = delay period (milliseconds)

nn$_2$ = motor current reduction percentage (%)

Rel. Commands QM — Get motor type.

QI — Get maximum motor current.

Example **2QR?** | *Query motor #2 torque reduction settings.*

5000,50 | *Controller returns 5000 ms and 50%.*

QS — Set Microstep Factor

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**QS**nn or xx**QS**?

Parameters

Description

xx [int]	—	Axis number.
nn [int]	—	Microstep value.

Range

xx	—	**1** to **Max. Axes**.

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.

Description Obsolete command, but kept for backward compatibility.

This command has no effect.

Returns If the "**?**" sign takes the place of **nn** value, this command reports 0.

Rel. Commands

QD	—	Update driver.
QI	—	Set maximum motor current.

Example

2QS?	|	*Read microstep factor of axis #2.*
0	|	*Controller returns a value of 0 for axis #2.*

QT — Set Tachometer Gain

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**QT**nn or xx**QT**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Tachometer gain.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **20**, or **?** to read present setting.

Units

xx	—	None.
nn	—	Volts/Krpm.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.

Description Obsolete command, but kept for backward compatibility.

This command has no effect.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

QD	—	Update driver.
QI	—	Set motor maximum current.

Example

2QT?	|	*Read tachometer gain setting of axis #2.*
0	|	*Controller returns a value of 0 V/Krpm for axis #2.*

QV — Set Average Motor Voltage

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**QV**nn or xx**QV**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Motor voltage.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **maximum driver rating**, or **?** to read present setting.

Units

xx	—	None.
nn	—	Volts.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.

Description Obsolete command, but kept for backward compatibility.

This command has no effect.

Returns If the "**?**" sign takes the place of **nn** value, this command reports 48.

Rel. Commands

QD	—	Update driver.
QI	—	Set maximum motor current.

Example

2QV?	|	*Read average motor voltage setting of axis #2.*
48.0	|	*Controller returns a value of 48Volts for axis #2.*

RQ — Generate Service Request (SRQ)

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **RQ**nn

Parameters

Description	**nn** [int]	—	Interrupt number.
Range	**nn**	—	**0** to **31**.
Units	**nn**	—	None.
Defaults	**nn**	Missing:	0.
		Out of range:	Error 7, PARAMETER OUT OF RANGE.

Description This command generates an interrupt service request to the host computer. The parameter **nn** is used to identify the **RQ** command which generated the interrupt. Upon receiving the interrupt, the host computer interrupt service routine should perform an IEEE 488 serial poll. If the interrupt was as a result of the **RQ** command, then bit 6 of the response is 1 and the lower five bits equal the parameter **nn**.

This command can be used to notify the host computer of the progress or flow of command execution in the motion controller.

Returns None.

Rel. Commands	SA	—	Set device address.

Example

2PR200;2WS;1PR100;1WS;**RQ3**	|	*Generate interrupt when RQ command is encountered and set bit 0 and 1.*

RS — Reset the Controller

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **RS**

Parameters None.

Description This command is used to perform a hardware reset of the controller. It performs the following preliminary tasks before resetting the controller:

1) Stop all the axes that are in motion. The deceleration value specified using the command AG is used to stop the axes.
2) Wait for 500 ms to allow the axes to settle.
3) Disable all the axes by turning the power OFF.
4) Reboot the system.

This process can take anywhere up to 20 seconds depending upon the controller configuration.

Returns None.

Rel. Commands None.

Example **RS** | *Reset the controller.*

SA — Set Device Address

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax SAnn or SA?

Parameters

Description **nn** [int] — address number.

Range **nn** — **1** to **30**.

Units **nn** — None.

Defaults **nn** Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error 7, PARAMETER OUT OF RANGE.

Description This command is used to set and report the device (i.e., ESP controller) address for use with IEEE-488 communications.

The address change takes affect immediately after the command is processed.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands None.

Example

SA3	|	*Set device address to 3.*
SA?	|	*Read present device address setting.*
3	|	*Controller returns device address #3.*

SB — Set/Get DIO Port GPIO Bit Status

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **SB**nn or **SB**?

Parameters

Description **nn** [int] — DIO value.

Range **nn** — **0** to **0FFFFH** (hexadecimal), or **?** to read current setting.

Units **nn** — None.

Defaults **nn** Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error 7, PARAMETER OUT OF RANGE.

Description This command is used to either set all digital I/O (DIO) port GPIO1 and GPIO2 logic level or read its present status. Bits 0-7 correspond to port GPIO1, and bits 8-15 to port GPIO2. Each 8-bit port can be set as either input or output with the **BO** command.

A DIO within a port configured as an input can only report its present HIGH or LOW logic level. Whereas a DIO bit within a port configured as an output can set(1) or clear(0) the corresponding DIO hardware to HIGH or LOW logic level.

Reading the status of a port configured as output returns its present output status.

NOTE

All direction bits are automatically zeroed, or cleared, after a system reset. Therefore all DIO ports turn to input by default.

NOTE

Each DIO bit has a pulled-up resistor to +5 V. Therefore, all bits will be at HIGH logic level if not connected to external circuit and configured as input.

BIT#	DEFINITION	0	1
0	GPIO1.DIO[0]	LOW	HIGH
1	GPIO1.DIO[1]	LOW	HIGH
2	GPIO1.DIO[2]	LOW	HIGH
3	GPIO1.DIO[3]	LOW	HIGH
4	GPIO1.DIO[4]	LOW	HIGH
5	GPIO1.DIO[5]	LOW	HIGH
6	GPIO1.DIO[6]	LOW	HIGH
7	GPIO1.DIO[7]	LOW	HIGH
8	GPIO2.DIO[0]	LOW	HIGH
9	GPIO2.DIO[1]	LOW	HIGH
10	GPIO2.DIO[2]	LOW	HIGH
11	GPIO2.DIO[3]	LOW	HIGH
12	GPIO2.DIO[4]	LOW	HIGH
13	GPIO2.DIO[5]	LOW	HIGH
14	GPIO2.DIO[6]	LOW	HIGH
15	GPIO2.DIO[7]	LOW	HIGH

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting in hexadecimal notation.

Rel. Commands BO — Set DIO port direction.

Example	BO?	|	*Read DIO port direction configuration.*
	0H	|	*Controller returns a value of 0H (all ports are input).*
	BO1H	|	Configure *DIO port GPIO1 as output.*
	SB0FFH	|	*Set all port GPIO1 DIO output HIGH.*

SH — Set Home Preset Position

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**SH**nn or xx**SH**?

Parameters

Description

xx [int] — Axis number.

nn [float] — Home preset position**.**

Range

xx — **1** to **Max. Axes**.

nn — Any position within the travel limits**.**

Units

xx — None.

nn — Defined motion units.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description This command defines the value that is loaded in the position counter when home is found. The default value for all motion devices is 0. This means that unless a new value is defined using this command, the home position will be set to 0 when a home search is initiated using the **OR** command or from the front panel (if available).

NOTE

The change takes effect only when a subsequent home search routine is performed. To make the change permanent, change the HomePreset parameter in the configuration file.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands DH — Define home.

Example

3MO	|	*Turn axis #3 motor power ON.*
3SH75.0	|	*Set axis #3 home position to 75.0 units.*
3OR1	|	*Perform a home search on axis #3.*
3MD?	|	*Query axis #3 motion status.*
1	|	*Controller returns a value of 1, when motion is done.*
3TP	|	*Query axis #3 position.*
75.0	|	*Controller returns a value of 75.0 units.*

SI — Set Master-Slave Jog Velocity Update Interval

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **SI**nn or **SI**?

Parameters

Description **nn** [int] — Jog velocity update interval.

Range **nn** — **1** to **1000**.

Units **nn** — Milliseconds.

Defaults **nn** Missing: Error 38, COMMAND PARAMETER MISSING.
Out of range: Error 7, PARAMETER OUT OF RANGE.

Description Obsolete command, but kept for backward compatibility.
This command has no effect.
Slave axis will jog with his master according to GR reduction ratio.

Returns If "**?**" sign is issued along with command, the controller returns slave axis.
jog velocity update interval.

Rel. Commands SS — Define master-slave relationship.
GR — Set master-slave reduction ratio.

SK — Set Master-Slave Jog Velocity Scaling Coefficients

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **SK**nn_1, nn_2 or **SK**?

Parameters

Description **nn**$_i$ [float] — Jog velocity scaling coefficients.

Range **nn**$_i$ — None.

Units **nn**$_i$ — None.

Description Obsolete command, but kept for backward compatibility.
This command has no effect.
Slave axis will jog with his master according to GR reduction ratio.

Returns If "**?**" sign is issued along with command, the controller returns slave axis.
jog velocity scaling coefficients.

Rel. Commands SS — Define master-slave relationship.
GR — set master-slave reduction ratio.

SL — Set Left Travel Limit

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**SL**nn or xx**SL**?

Parameters

Description

xx [int] — Axis number.

nn [float] — left (negative) software limit.

Range

xx — **1** to **Max. Axes**.

nn — **-Max_Long * encoder resolution** to **0**.

Units

xx — None.

nn — Predefined motion units.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description

This command defines the value for the negative (left) software travel limit. It should be used to restrict travel in the negative direction to protect the motion device or its load. For instance, if traveling full range, a stage could push its load into an obstacle. To prevent this, the user can reduce the allowed travel by changing the software travel limit.

Since a motion device must be allowed to find its home position, the home switch and/or sensor must be inside the travel limits. This means that both positive and negative travel limits cannot be set on the same side of the home position. A more obvious restriction is that the negative limit cannot be greater than the positive limit. If any of these restrictions is not respected, the controller will return PARAMETER OUT OF RANGE.

NOTE

If the command is issued for an axis in motion, the new limit should not be set inside the current travel.

NOTE

Be careful when using this command. The controller does not know the real hardware limits of the motion device. Always set the software limits inside the hardware limits (limit switches). In normal operation, a motion device should never hit a limit switch.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

OR — Search for home.

SR — Set right travel limit.

Example **1SL41.4** | *Set negative travel limit of axis #1 to 41.4 units.*

SM — Save Settings to Non-Volatile Memory

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax **SM**

Parameters None.

Description Obsolete command, but kept for backward compatibility.

This command has no effect.

To change the settings permanently use the configuration file.

NOTE

User programs created with EP command are automatically saved to non-volatile memory.

Returns None.

Rel. Commands None.

SN — Set Axis Displacement Units

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**SN**nn or xx**SN**?

Parameters

Description

xx [int]	—	Axis number.
nn [int]	—	Displacement units.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **11** where: 0 = Encoder count, 1 = Motor step, 2 = Millimeter, 3 = Micrometer, 4 = Inches, 5 = Milli-inches, 6 = Micro-inches, 7 = Degree, 8 = Gradian, 9 = Radian, 10 = Milliradian, 11 = Microradian, or **?** to read present setting.

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description This command is used to set the displacement units for the for axis **xx**.

Position is automatically converted according to new unit.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

SU	—	Set encoder resolution.

Example

2SN	|	*Read displacement unit setting of axis #2.*
2	|	*Controller returns a value 2 (millimeter) for axis #2.*
2SN0	|	*Set displacement unit to 0 (encoder count) for axis #2.*

SR — Set Right Travel Limit

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**SR**nn or xx**SR**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Right (positive) software limit.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **Max_Long * encoder resolution**.

Units

xx	—	None.
nn	—	Defined motion units.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description

This command defines the value for the positive (right) software travel limit. It should be used to restrict travel in the positive direction to protect the motion device or its load. For instance, if traveling full range, a stage could push its load into an obstacle. To prevent this, the user can reduce the allowed travel by changing the software travel limit.

Since a motion device must be allowed to find its home position, the home switch and/or sensor must be inside the travel limits. This means that both positive and negative travel limits cannot be set on the same side of the home position. A more obvious restriction is that the negative limit cannot be greater than the positive limit. If any of these restrictions is not respected, the controller will return PARAMETER OUT OF RANGE

NOTE

If the command is issued for an axis in motion, the new limit should not be set inside the current travel.

NOTE

Be careful when using this command. The controller does not know the real hardware limits of the motion device. Always set the software limits inside the hardware limits (limit switches). In normal operation, a motion device should never hit a limit switch.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

OR	—	Search for home.
SL	—	Set left travel limit.

Example **1SR41.4** | *Set positive travel limit of axis #1 to 41.4 units.*

SS — Define Master-Slave Relationship

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**SS**nn or xx**SS**?

Parameters

Description

xx [int]	—	Axis number to be defined as a slave.
nn [int]	—	Axis number to be defined as a master.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**1** to **Max. Axes**.

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE

Description This command defines master-slave relationship between any two axes. A few rules are in place for ease of use.

- An axis cannot be assigned as its own slave if it is already in a trajectory mode that is specific to master-slaving.
- A slave axis cannot be moved individually using PA or PR commands if its trajectory mode is specific to master-slaving.

This command gets executed immediately, and can also be called from within a program.

To disable the master-slave relationship, set the master axe to 0.

Returns If "**?**" sign is issued along with command, the controller returns master axis number.

Rel. Commands

GR	—	Set master-slave reduction ratio

Example

2SS1	|	*Set axis 2 to be the slave of axis 1.*
2SS?	|	*Query the master axis number for axis 2.*
1	|	*Controller returns a value of 1.*
2GR1.0	|	*Set the reduction ratio of axis 2 to 1.0.*
1MO	|	*Turn axis 1 motor power ON.*
2MO	|	*Turn axis 2 motor power ON.*
1PA10	|	*Move axis 1 to absolute 10 units.*
2PA20	|	*Move axis 2 to absolute 20 units.*
TB	|	*Read error messages.*
232, 242000, AXIS-2 INVALID TRAJECTORY MODE FOR MOVING	|	*Controller returns appropriate error message.*

ST — Stop Motion

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**ST**

Parameters

Description	**xx** [int]	—	Axis number.
Range	**xx**	—	**1** to **Max. Axes**.
Units	**xx**	—	None.
Defaults	**xx** out of range:		Error 9, AXIS NUMBER OUT OF RANGE.

Description This command stops a motion in progress using deceleration rate programmed with **AC** (set deceleration/deceleration) command on the specified axes. If the **ST** command is sent with no axis parameter, all axes are stopped.

Returns None.

Rel. Commands	AB	—	Abort motion.
	AC	—	Set acceleration/deceleration.
	MF	—	Motor power off.

Example	2PA40	|	*Move axis #2 to absolute position 40.*
	2ST	|	*Stop motion on axis #2.*

SU — Set Encoder Resolution

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**SU**nn or xx**SU**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Encoder resolution.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**2e-9** to **2e+9** in user defined units, or ? to read present setting.

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description This command is used to set the encoder resolution for axis **xx**.

NOTE

The encoder resolution can only be changed when encoder feedback is enabled. See ZF command.

Returns If "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

SU	—	Set encoder resolution.
QD	—	Update driver.
ZF	—	set feedback and following error configuration.

Example

2SU?	|	*Read encoder resolution setting of axis #2.*
0.0001	|	*Controller returns a value of 0.0001 units for axis #2.*
2SU0.0005	|	*Set encoder resolution to 0.0005 units for axis #2.*

TB — Read Error Message

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **TB**nn or **TB**?

Parameters

Description **nn** [int] — Error code.

Range **nn** — **0** to **Max. error code**.

Units **nn** — None.

Defaults **nn** Missing: Interpreted as "?"'.

Out of range:Error description not available.

Description This command is used to retreive the error code, timestamp, and the associated message of the error code **nn**.

The error code is one numerical value up to three(3) digits long (see Appendix for complete listing). In general, non-axis specific errors numbers range from 1-99. Axis-**1** specific errors range from **1**00-**1**99, Axis-**2** errors range from **2**00-**2**99 and so on.

The timestamp is in terms of servo cycle (100 µs) ticks accumulated since the last System Reset, incrementing at the servo interrupt interval (100us default).

The message is a description of the error associated with it.

All arguments are separated by commas.

If "**?**" sign takes the place of **nn** value or if **nn** is missing, this command reports the description of the older error in the error buffer.

NOTE

Errors are maintained in a FIFO buffer ten(10) elements deep. When an error is read using TB or TE, the controller returns the first error and the error buffer is cleared by one(1) element. This means that an error can be read only once, with either command.

Returns **aa**, **bb**, **cc**, where:
aa = **Error code** (See Appendix for complete listing).
bb = **Timestamp**.
cc = **Error message**..

Rel. Commands TE — Read error code.

Example

	TB	*Read error message.*
0, 451322, NO ERROR DETECTED		*Controller returns no error.*
	8PA12.3	*Move axis #8 to position 12.3.*
	TB?	*Read error message.*
9, 451339, AXIS NUMBER OUT OF RANGE		*Controller returns error code, timestamp, and description.*

TE — Read Error Code

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **TE**nn or**TE**?

Parameters

Description **nn** [int] — Query type.

Range **nn** — **1** to **2** where:
1 = get the oldest known error code without removing it,
2 = get the number of errors currently in the FIFO,
or **?** to read oldest known error code.

Units **nn** — None.

Defaults **nn** Missing: Interpreted as "?".
Out of range: Interpreted as "?".
Timeout: Error 2, RS-232 COMMUNICATION TIME-OUT.

Description This command is used to read the error code.

The error code is one numerical value up to three digits long (see Appendix for complete listing).

In general, non-axis specific errors numbers range from 1-99. Axis-**1** specific errors range from **1**00-**1**99, Axis-**2** errors range from **2**00-**2**99 and so on.

NOTE

Errors are maintained in a FIFO buffer ten(10) elements deep. When an error is read using TB or TE, the controller returns the first error and the error buffer is cleared by one(1) element. This means that an error can be read only once, with either command.

Returns **aa**, where:
aa = **Error code number** or **number of errors in the buffer**.
See Appendix for complete listing.

Rel. Commands TB — Read error message.

Example

TE?	|	*Read error message.*
0	|	*Controller returns no error.*
8PA12.3	|	*Move axis #8 to position 12.3.*
TE?	|	*Read error message.*
9	|	*Controller returns error code 9 meaning incorrect axis number.*

TJ — Set Trajectory Mode

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**TJ**nn or xx**TJ**?

Parameters

Description

xx [int] — Axis number.

nn [int] — Trajectory mode.

Range

xx — **1** to **Max. Axes**.

nn — **1** to **6**, where:
2 = s-curve mode,

Units **xx** — None.

nn — None.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description Obsolete command, but kept for backward compatibility.

This command has no effect. The trajectory mode if fixed to *s-curve.*

Returns If the "**?**" sign takes the place of **nn** value, this command reports 2.

Rel. Commands

SS — Set master-slave relationship.

GR — Set master/slave gear ratio.

Example

1TJ? | *Report current trajectory mode setting on axis #1.*

2 | *Controller returns trajectory mode 2 (s-curve) for axis #1.*

TP — Read Actual Position

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax xx**TP**

Parameters

Description **xx** [int] — Axis number.

Range **xx** — **1** to **Max. Axes**.
None = all axes

Units **xx** — None.

Defaults **xx** Missing: Returns position of each axis.
Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description This command is used to read the actual position. It returns the instantaneous real position of the specified axis.

Returns **nn**, or $\mathbf{nn_i}$, … $\mathbf{nn_i}$ where:
nn = **Actual position** of requested axis in pre-defined units
$\mathbf{nn_i}$ = **Actual position** of ith axis in pre-defined units

Rel. Commands

PA	—	Move to an absolute position.
PR	—	Move to a relative position.
DP	—	Read instantaneous desired position.

Example

3TP	|	*Read real position on axis #3.*
5.322	|	*Controller returns real position 5.322 for axis #3.*
TP	|	*Read real position on all axes.*
0,1.452 ,5.322	|	*Controller returns real position of each axis:* *0 for axis #1.* *1.452 for axis #2* *5.322 for axis #3*

TS — Read Controller Status

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **TS** or xx**TS** or xx**TS1**

Parameters

Description **xx** [int] — Axis number.

Range **xx** — **1** to **Max. Axes**.
None = controller status

Units **xx** — None.

Defaults **xx** Missing: Returns controller status.
Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description This command is used to read the controller axes status byte or an axis/driver status byte.

TS: Controller axes status

xxTS : Axis xx status

xxTS1 : Axis xx driver status

The bytes returned are in the form of an ASCII character. The value of each bit in the status byte can be deduced after converting the ASCII character into a binary value. Each bit of the status byte represents a particular controller parameter, as described in the following table.

NOTE

Please refer to the Appendix for a complete ASCII to binary conversion table.

Controller axes status (TS)

Bit #	Function	Meaning for	
		Bit LOW	Bit HIGH
0	Axis #1 in motion	Stationary	In motion
1	Axis #2 in motion	Stationary	In motion
2	Axis #3 in motion	Stationary	In motion
3	Reserved	Default	—
4	Motor power of at least one axis	OFF	ON
5	Reserved	Default	—
6	Reserved	—	Default
7	Reserved	Default	—

Axis status (xxTS)

	Bit #	Function	Meaning for Bit LOW	Meaning for Bit HIGH
1st byte	0	Axis is connected	YES	NO
	1	Motor state	OFF	ON
	2	Axis is in motion	NO	YES
	3	Reserved	Default	—
	4	Origine done	YES	NO
	5	Reserved	Default	—
	6	Reserved	—	Default
	7	Reserved	Default	—
2nd byte	0	Following error	NO	YES
	1	Motor fault	NO	YES
	2	EOR- is reached	NO	YES
	3	EOR+ is reached	NO	YES
	4	ZM is reached	NO	YES
	5	Reserved	Default	—
	6	Reserved	—	Default
	7	Reserved	Default	—

Driver status (xxTS1)

	Bit #	Function	Meaning for Bit LOW	Meaning for Bit HIGH
1st byte	0	Short circuit	NO	YES
	1	Fuse broken or Low supply voltage	NO	YES
	2	Thermistance (motor or driver)	NO	YES
	3	Parameters error	NO	YES
	4	RMS current limit	NO	YES
	5	Reserved	Default	—
	6	Reserved	—	Default
	7	Reserved	Default	—
2nd byte	0	Reserved (SubD pin 10)	—	Default
	1	Reserved (SubD pin 11)	—	Default
	2	Reserved (SubD pin 12)	—	Default
	3	Current Limit	NO	YES
	4	Reserved	Default	—
	5	Reserved	Default	—
	6	Reserved	—	Default
	7	Reserved	Default	—

Returns ASCII character representing the status byte.

Rel. Commands TX — Read controller activity.

Example	**TS**	|	*Read controller axes status.*
	S	|	*Controller returns character S indicating axes #1 and #2 are in motion, and motor power of at least one axis is ON.*
	3TS	|	*Read axis #3 status.*
	@B	|	*Controller returns character @ and B indicating axes #3 is connected, motor OFF, not in motion, origine done, and with motor fault.*
	3TS1	|	*Read axis #3 driver status.*
	PG	|	*Controller returns character P and G indicating axes #3 has a current limit error.*

TV — Get Actual Velocity

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax xx**TV**

Parameters

Description	**xx** [int]	—	Axis number.
Range	**xx**	—	**1** to **Max. Axes**.
Units	**xx**	—	None.
Defaults	**xx** Missing:		Error 37, AXIS NUMBER MISSING.
	Out of range:		Error 9, AXIS NUMBER OUT OF RANGE.

Description This command is used to read the actual velocity of an axis. The command can be sent at any time but its real use is while motion is in progress.

Returns **nn**, where:
nn = **Actual velocity of the axis** in pre-defined units.

Rel. Commands	PA	—	Move to an absolute position.
	PR	—	Move to a relative position.
	FV	—	Set Current Velocity Filter Frequency

Example

3TP?	|	*Read position on axis #3.*
5.32	|	*Controller returns position 5.32 units for axis #3.*
3PR2.2	|	*Start a relative motion of 2.2 units on axis #3.*
3DV	|	*Read desired velocity on axis #3.*
0.2	|	*Controller returns velocity 0.2 units/s for axis #3.*
3TV	|	*Read actual velocity on axis #3.*
0.205	|	*Controller returns velocity 0.205 units/s for axis #3.*
3DP?	|	*Read desired position on axis #3.*
7.52	|	*Controller returns desired position 7.52 units for axis #3.*

TX — Read Controller Activity

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **TX** or **TX1**

Parameters None.

Description This command is used to read the controller activity or status register.

TX: Controller activity

TX1 : Controller status

The bytes returned are in the form of an ASCII character. The value of each bit in the status byte can be deduced after converting the ASCII character into a binary value. Each bit of the status byte represents a particular parameter, as described in the following table.

NOTE

Please refer to the Appendix for a complete ASCII to binary conversion table.

Controller Activity (TX)

Bit #	Function	Meaning for	
		Bit LOW	Bit HIGH
0	At least one program is executing	NO	YES
1	Wait command is executing	NO	YES
2	Manual jog mode is active	NO	YES
3	Local mode is inactive	Default	—
4	At least one trajectory is executing	NO	YES
5	Reserved	Default	—
6	Reserved	—	Default
7	Reserved	Default	—

Controller status (TX1)

	Bit #	Function	Meaning for Bit LOW	Meaning for Bit HIGH
1st byte	0	Controller boot	OK	Failed
	1	Axis #1 boot configuration	OK	Failed
	2	Axis #2 boot configuration	OK	Failed
	3	Axis #3 boot configuration	OK	Failed
	4	Reserved	Default	—
	5	Reserved	Default	—
	6	Reserved	—	Default
	7	Reserved	Default	—
2nd byte	0	Hardware Inhibit	Detected	Not detected
	1	Drivers 48V power supply	Detected	Not detected
	2	Axis #1 Motor ON	NO	YES
	3	Axis #2 Motor ON	NO	YES
	4	Axis #3 Motor ON	NO	YES
	5	Reserved	Default	—
	6	Reserved	—	Default
	7	Reserved	Default	—
3rd byte	0	Drivers commands underrun	NO	YES
	1	Reserved	Default	—
	2	Reserved	Default	—
	3	Reserved	Default	—
	4	Reserved	Default	—
	5	Reserved	Default	—
	6	Reserved	—	Default
	7	Reserved	Default	—

Returns ASCII character representing the status byte.

Rel. Commands TS — Read controller status.

Example

TX | *Read controller activity.*

P | *Controller returns character P indicating at least one trajectory is executing*

TX1 | *Read controller status.*

DP@ | *Controller returns character D, P and @ indicating that axis #2 boot configuration failed and axis #3 motor is ON.*

UF — Update Servo Filter

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**UF**

Parameters

Description **xx** [int] — Axis number.

Range **xx** — **0** to **Max. Axes**.

Units **xx** — None.

Defaults **xx** Missing: No error, is interpreted like 0.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description This command is used to make active the latest entered PID parameters. Any new value for Kp, Ki, Kd, Ks, Kt and maximum following error are not being used in the PID loop calculation until **UF** command is received. This assures that the parameters are loaded simultaneously, without any transitional glitches in the loop.

If the axis specifier **xx** is missing or set to 0, the controller updates the filters for all axes. If **xx** is a number between 1 and 3, the controller updates only the filter for the specified axis.

Returns None.

Rel. Commands

FE	—	Set maximum following error.
KD	—	Set derivative gain factor.
KI	—	Set integral gain factor.
KP	—	Set proportional gain factor.

Example

3KP0.05	|	*Set proportional gain factor of axis #3 to 0.05.*
3KD0.07	|	*Set derivative gain factor of axis #3 to 0.07.*
3UF	|	*Update servo loop of axis #3 with the new parameters.*

UH — Wait for DIO Bit High

	IMM	PGM	MIP
Usage	–	♦	–

Syntax xx**UH**

Parameters

Description **xx** [int] — DIO bit number.

Range **xx** — **0** to **15**.

Units **xx** — None.

Defaults **xx** Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error 7, PARAMETER OUT OF RANGE.

Description This command causes a program to wait until a selected I/O input bit becomes high. It is level, not edge sensitive. This means that at the time of evaluation, if the specified I/O bit **xx** is high already, the program will continue to execute subsequent commands.

NOTE

All DIO bits are pulled high on the board. Therefore, a missing signal will cause the wait to complete and subsequent commands will continue to be executed.

Returns None.

Rel. Commands UL — Wait for DIO bit low.

Example

1EP	|	*Enter stored program #1.*
1MO	|	*Turn axis #1 motor power ON.*
1MV+	|	*Move axis #1 indefinitely in positive direction.*
13UH	|	*Wait for DIO bit #13 to go HIGH before executing any subsequent commands.*
1ST	|	*Stop axis #1.*
WT500	|	*Wait for 500 ms.*
1MV-	|	*Move axis #1 indefinitely in negative direction.*
QP	|	*Quit program mode.*

UL — Wait for DIO Bit Low

	IMM	PGM	MIP
Usage	–	♦	–

Syntax xx**UL**

Parameters

Description **xx** [int] — DIO bit number.

Range **xx** — **0** to **15**.

Units **xx** — None.

Defaults **xx** Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error 7, PARAMETER OUT OF RANGE.

Description This command causes a program to wait until a selected I/O input bit becomes low. It is level, not edge sensitive. This means that at the time of evaluation, if the specified I/O bit **xx** is low already, the program will continue to execute subsequent commands.

Returns None.

Rel. Commands UH — Wait for DIO bit high.

Example

```
1EP   | Enter stored program #1.
1MO   | Turn axis #1 motor power ON.
1MV+  | Move axis #1 indefinitely in positive direction.
13UL  | Wait for DIO bit #13 to go LOW before executing any subsequent commands.
1ST   | Stop axis #1.
WT500 | Wait for 500 ms.
1MV-  | Move axis #1 indefinitely in negative direction.
QP    | Quit program mode.
```

VA — Set Velocity

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**VA**nn or xx**VA**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Velocity value.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **MaximumVelocity**, or **?** to read current setting.

Units

xx	—	None.
nn	—	Preset units/second.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x10, MAXIMUM VELOCITY EXCEEDED.

Description This command is used to set the velocity value for an axis. Its execution is immediate, meaning that the velocity is changed when the command is processed, even while a motion is in progress.

It can be used as an immediate command or inside a program. If the requested axis is member of a group, the commanded velocity becomes effective only after the axis is removed from the group. Refer to Advanced Capabilities section in the ESP302 Features Manual for a detailed description of grouping and related commands.

Avoid changing the velocity during the acceleration or deceleration periods. For better predictable results, change velocity only when the axis is not moving or when it is moving with a constant speed.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

AC	—	Set acceleration.
VU	—	Get maximum velocity.
PA	—	Execute an absolute motion.
PR	—	Execute a relative motion.

Example

2VA?	|	*Read desired velocity of axis #2.*
10	|	*Controller returns a velocity value of 10 units/s.*
2PA15	|	*Move to absolute position 15.*
WT500	|	*Wait for 500 ms.*
2VA4	|	*Set axis #2 velocity to 4 units/s.*
2VA?	|	*Read velocity of axis #2.*
4	|	*Controller returns a velocity value of 4 units/s.*

VB — Set Base Velocity for Step Motors

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**VB**nn or xx**VB**?

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	Base velocity value.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **MaximumVelocity**, or **?** to read current setting.

Units

xx	—	None.
nn	—	Preset units/second.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.

Description Obsolete command, but kept for backward compatibility.

This command has no effect.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

AC	—	Set acceleration/deceleration.
VA	—	Set velocity.
VU	—	Get maximum velocity.
PA	—	Execute an absolute motion.
PR	—	Execute a relative motion.

Example

2VB?	|	*Read desired base velocity of axis #2.*
5	|	*Controller returns a velocity value of 5 units/s.*

VE — Read Controller Firmware Version

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **VE**nn or **VE**?

Parameters

Description **nn** [int] — Firmware part.

Range **nn** — **0** to **4**,
Missing = 0 = "**?**"

Units **nn** — None.

Defaults **nn** Out of range: Error x01, PARAMETER OUT OF RANGE.
Timeout: Error 2, RS-232 COMMUNICATION TIME-OUT.

Description This command is used to read the controller type and version.

The ESP302 is separated into different parts with different versions.

NOTE

Important information needed when asking for technical support for the motion control system or when reporting a problem is the controller version. Use this command to determine the controller type and in particular, the firmware version.

Returns

VE0 or **VE? or VE**	: ESP302 Snapshot version
VE1	: ESP302 MotionKernel version
VE2	: ESP302 Host version
VE3	: ESP302 FrontPanel version
VE4	: ESP302 Web version

Rel. Commands None.

Example

	VE?	*Read controller Snapshot version.*
ESP302 Snapshot Version N15000		*Controller returns model ESP302 Snapshot Version N15000.*
	VE1	*Read controller MotionKernel version.*
ESP302 MotionKernel Version 1.0.0		*Controller returns model ESP302 MotionKernel Version 1.0.0.*
	VE2	*Read controller Host version.*
ESP302 Host Version 1.0.2		*Controller returns model ESP302 Host Version 1.0.2.*
	VE3	*Read controller FrontPanel version.*
ESP302 FrontPanel Version 1.2.0		*Controller returns model ESP302 FrontPanel Version 1.2.0.*
	VE3	*Read controller Web version.*
ESP302 FrontPanel Web 2.0.1		*Controller returns model ESP302 Web Version 2.0.1.*

VF — Set Velocity Feed-Forward Gain

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**VF**nn or xx**VF**?

Parameters

Description

xx [int] — Axis number.

nn [float] — velocity feed-forward gain factor Vf.

Range

xx — **1** to **Max. Axes**.

nn — **0** to **Max_Double**, or **?** to read current setting.

Units

xx — None.

nn — None.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description This command sets the velocity feed-forward gain factor **VF**. It is active for any DC servo based motion device.

See the "Feed-Forward Loops" section in the ESP302 Features Manual to understand the basic principals of feed-forward.

NOTE

The command can be sent at any time but it has no effect until the UF (update filter) is received.

NOTE

This command is volatile, to change the parameter permanently change KFeedForwardVelocity in the configuration file.

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands

KI — Set integral gain factor.

KS — Set saturation gain factor.

KD — Set derivative gain factor.

KP — Set proportional gain factor.

AF — Set acceleration feed-forward gain.

UF — Update filter.

Example

3AF0.8	*Set acceleration feed-forward gain factor for axis #3 to 0.8.*
3VF?	*report present axis-3 velocity feedforward setting.*
1.4	*Controller returns a value of 1.4.*
3VF1.5	*Set acceleration feed-forward gain factor for axis #3 to 1.5.*
3UF	*Update PID filter; only now the VF command takes effect.*

VU — Get Maximum Velocity

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**VU**?

Parameters

Description **xx** [int] — Axis number.

Range **xx** — To **Max. Axes**.

Units **xx** — None.

Defaults **xx** Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

Description This command is used to get the maximum velocity value for an axis. This parameter is read-only and can be modified only through the MaximumVelocity parameter in configuration file.

Returns **nn** where:

. nn = maximum velocity (predefined units/second)

Rel. Commands

VA	—	Set velocity.
PA	—	Execute an absolute motion.
PR	—	Execute a relative motion.
AC	—	Set acceleration/deceleration.

Example

2VU?	|	*Read maximum allowed velocity of axis #2.*
10	|	*Controller returns a value of 10 units/second.*

WP — Wait for Position

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**WP**nn

Parameters

Description

xx [int]	—	Axis number.
nn [float]	—	position value.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**Starting position** to **destination** of axis number **xx**.

Units

xx	—	None.
nn	—	Predefined units.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error 7, PARAMETER OUT OF RANGE.

Description This command stops program execution until a user specified position is reached. The program continues executing any subsequent commands only after axis **xx** has reached position **nn**.

NOTE

Ensure that position nn is within the travel range of axis xx. The controller cannot always detect if a value is outside the travel range of an axis to flag an error, especially while making coordinated motion of multiple axes.

Wait commands are primarily intended for use in internal program execution or in combination with the **RQ** command. If used in command mode, it is important to note that input command processing is suspended until the wait condition has been satisfied.

Returns None.

Rel. Commands

WT	—	wait.
WS	—	wait for motion stop.

Example

2PA-10; 2WS	|	*Move axis #2 to position –10 units and wait for stop.*
2PA10; **2WP0**; 3PA5	|	*Move axis #2 to position 10 units, wait for axis #2 to reach position 0 units and then move axis #3 to position 5 units.*

WS — Wait for Motion Stop

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax xx**WS**nn

Parameters

Description

xx [int] — Axis number.

nn [int] — Delay after motion is complete.

Range

xx — **0** to **Max. Axes**.

nn — **0** to **60000**.

Units

xx — None.

nn — Milliseconds.

Defaults

xx Missing: All axes in motion.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE

nn Missing: Same as 0

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description

This command stops the program execution until a motion is completed. The program is continued only after axis **xx** reaches its destination. If **xx** is not specified, the controller waits for all motion in progress to end. If **nn** is specified different than 0, the controller waits an additional **nn** milliseconds after the motion is complete and then executes the next commands.

NOTE

Wait commands are primarily intended for use in internal program execution or in combination with the RQ command. If used in command mode, it is important to note that input command processing is suspended until the wait condition has been satisfied.

Returns None.

Rel. Commands

WT — Wait.

WP — Wait for position.

Example

2PA10;2**WS500**;3PA5 | *Move axis #2 to position 10 units, wait for axis #2 to reach destination, wait an additional 500ms and then move axis #3 to position 5 units.*

WT — Wait

	IMM	PGM	MIP
Usage	♦	♦	♦

Syntax **WT**nn

Parameters

Description **nn** [int] — Wait time (delay).

Range **nn** — **0** to **60000**.

Units **nn** — Milliseconds.

Defaults **nn** Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

Description This command causes the controller to pause for a specified amount of time. This means that the controller will wait **nn** milliseconds before executing the next command.

NOTE

Even though this command can be executed in immediate mode, its real value is as a flow control instruction inside programs.

Wait commands are primarily intended for use in internal program execution or in combination with the RQ command. If used in command mode, it is important to note that input command processing is suspended until the wait condition has been satisfied.

Returns None.

Rel. Commands WS — Wait for stop.

WP — Wait for position.

Example

2MO;**WT400**;2PA2.3 | *Turn axis motor ON, wait an additional 400 ms and then move axis 2 to position 2.3 units.*

XM — Read Available Memory

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **XM**

Parameters None.

Description This command reports the amount of unused program memory. The controller has 4G bytes of non-volatile memory available for the controller operating system, the firmware, and the user data like programs and gathering files.
This command reports the amount not used.

Returns Available storage space.

Rel. Commands

EP	—	Enter program download mode.
EX	—	Execute a stored program.
LP	—	List stored program.
XX	—	Delete a stored program.

Example

XM	*Read available memory.*
Available storage space = 495177728	*Controller reports available storage space.*

XX — Erase Program

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax xx**XX**

Parameters

Description	**xx** [int]	—	program number.
Range	**xx**	—	**1** to **127**.
Units	**xx**	—	None.
Defaults	**xx** Missing:		Error 38, COMMAND PARAMETER MISSING.
	Out of range:		Error 7, PARAMETER OUT OF RANGE.

Description This command deletes the program **xx** from controller's non-volatile memory.

Returns None.

Rel. Commands	EP	—	Enter program download mode.
	EX	—	Execute a stored program.
	LP	—	List stored program.
	XM	—	Read available memory.

Example

1XX	|	*Delete program #1.*
XM	|	*Read available memory.*
Available storage space = 60228	|	*Controller reports available storage space.*
2XX	|	*Delete program #2.*
XM	|	*Read available memory.*
Available storage space = 61440	|	*Controller reports available storage space.*

YZ — Set controller command terminator and echo

	IMM	PGM	MIP
Usage	♦	–	–

Syntax **YZ**nn or **YZ**?

Parameters

Description **nn** [int] — controller mode for RS232 or USB.

Range **nn** — **00** in CR, No echo, out CR LF (standard controller).
or **01** in CR, echo CR, out CR LF
or **10** in LF, No echo, out CRLF
or **11** in LF, echo LF, out CRLF
or **12** in LF, echo, out, LF
or **?** to get the current configuration

Units **nn** — None.

Defaults **nn** Missing: Error 38, COMMAND PARAMETER MISSING.
Out of range: Error 7, PARAMETER OUT OF RANGE.

Description This command is used to set the command string terminator and command string echo for RS232 and USB communication.

YZ	*In*	*Echo*	*Out*
00	Command + **CR**	No	[Response + **CR LF**]
01	Command + **CR**	Command + **CR**	[Response + **CR LF**]
10	Command + **LF**	No	[Response + **CR LF**]
11	Command + **LF**	Command + **LF**	[Response + **CR LF**]
12	Command + **LF**	Command	[Response] + **LF**

NOTE:

"YZ0" resets the controller's version to the default value "ESP302 version x.x.x".

NOTE:

After this command, you must save this new configuration in the memory with the "SM" command and reboot the controller with the "RS" command.

[T]: Transmit
[R]: Receive

YZ0	YZ1
[T] 1VE + **CR** [R] ESP302 version x.x.x + **CR LF**	[T] 1VE + **CR** [R] 1VE + **CR** [R] ESP302 version x.x.x + **CR LF**
[T] 1VA10 + **CR**	[T] 1VA10 + **CR** [R] 1VA10 + **CR**
[T] 1VA? + **CR** [R] 10 + **CR LF**	[T] 1VA? + **CR** [R] 1VA? + **CR** [R] 10 + **CR LF**

YZ10	YZ11	YZ12
[T] 1VE + **LF** [R] ESP302 version x.x.x + **CR LF**	[T] 1VE + **LF** [R] 1VE + **LF** [R] ESP302 version x.x.x + **CR LF**	[T] 1VE + **LF** [R] 1VE [R] ESP302 version x.x.x + **LF**
[T] 1VA10 + **LF**	[T] 1VA10 + **LF**	[T] 1VA10 + **LF**
[T] 1VA? + **LF** [R] 10 + **CR LF**	[R] 1VA10 + **LF** [T] 1VA? + **LF** [R] 1VA? + **LF** [R] 10 + **CR LF**	[R] 1VA10 + **LF** [T] 1VA? + **LF** [R] 1VA? [R] 10 + **LF**

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting.

Rel. Commands
SM — Save current settings to non-volatile memory.
VE — Read firmware version.

Example

YZ?		
0	|	*Controller returns value (stardard version).*
VE?	|	*Read controller firmware version.*
ESP302 Snapshot version N200xxx	|	*Controller returns firmware version.*
YZ12	|	*Set command terminator as LF and command Echo.*
SM	|	*Save current setting to non-volatile memory.*
RS	|	*Reboot the controller.*
YZ?		
12	|	*Controller returns value.*
VE?	|	*Read controller firmware version.*
VE? ESP302 Snapshot version N200xxx	|	*Controller returns firmware version.*

ZA — Set Amplifier I/O Configuration

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**ZA**nn or xx**ZA**?

Parameters

Description **xx** [int] — Axis number.

nn [int] — amplifier I/O configuration.

Description Obsolete command, but kept for backward compatibility.

This command has no effect.

ZB — Set Feedback Configuration

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**ZB**nn or xx**ZB**?

Parameters

Description	**xx** [int]	—	Axis number.
	nn [int]	—	Feedback configuration.
Range	**xx**	—	**1** to **Max. Axes**.
	nn	—	**0** to **3FFH** (hexadecimal with leading zero(0)), or **?** to read current setting.
Units	**xx**	—	None.
	nn	—	None.

Defaults	**xx**	Missing:	Error 37, AXIS NUMBER MISSING.
		Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
	nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
		Out of range:	Error x01, PARAMETER OUT OF RANGE.

Description Obsolete command, but kept for backward compatibility.

This command is equivalent to ZF command.

Rel. Commands	ZF	—	Set feedback and following error configuration.

ZE — Set E-Stop Configuration

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**ZE**nn or xx**ZE**?

Parameters

Description

xx [int] — Axis number.

nn [int] — E-stop configuration.

Range

xx — **1** to **Max. Axes**.

nn — **0** to **7H**
or **?** to read current setting.

Units

xx — None.

nn — None.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

out of range: Error x17, ESP CRITICAL SETTINGS ARE PROTECTED.

Description This command is used to set the emergency stop (e-stop) configuration , fault checking, and event handling for axis specified with **xx**.

NOTE

If bit-0 or both bits-1 and -2 are set to zero(0) then no action will be taken by the controller.

---

e-stop configuration

Bit #	Function	Meaning for	
		Bit LOW	Bit HIGH
0	E-stop checking	Disabled	Enabled
1	disable motor power on E-stop event	do not disable motor power on E-stop event	disable motor power on E-stop event
2	abort motion on E-stop event	do not abort motion on E-stop event	abort motion on E-stop event

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting in hexadecimal notation.

Rel. Commands

ZF — Set feedback and following error configuration.

ZH — Set hardware limit configuration.

ZS — Set software limit configuration.

ZZ — Set general system configuration.

Example

2ZE? | Read e-stop configuration of axis #2.

03H | Controller returns a value of 3H for axis #2.

2ZE5H | Set e-stop configuration to 5H for axis #2.

ZF — Set Following Error Configuration

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**ZF**nn or xx**ZF**?

Parameters

Description

xx [int] — Axis number.

nn [int] — following error configuration.

Range

xx — **1** to **Max. Axes**.

nn — **0** to **3FFH** (hexadecimal with leading zero(0)), or ? to read current setting.

Units

xx — None.

nn — None.

Defaults

xx Missing: Error 37, AXIS NUMBER MISSING.

Out of range: Error 9, AXIS NUMBER OUT OF RANGE.

nn Missing: Error 38, COMMAND PARAMETER MISSING.

Out of range: Error x01, PARAMETER OUT OF RANGE.

critical setting: Error x17, ESP CRITICAL SETTINGS ARE PROTECTED.

Description This command is used to set the following error configuration , fault checking, and event handling for axis specified with **xx**.

NOTE

If bit-0 or both bits-1 and -2 are set to zero(0) then no action will be taken by the controller.

Feedback configuration

Bit #	Function	Meaning for	
		Bit LOW	Bit HIGH
0	Position feedback checking	Disabled	Enabled
1	disable motor power on Following error event	do not disable motor power on Following error event	disable motor power on Following error event
2	abort motion on Following error event	do not abort motion on Following error event	abort motion on Following error event
3	Reserved	Default	—
4	Reserved	Default	—
5	Reserved	Default	—
6	Reserved	Default	—
7	Reserved	Default	—
8	Closed/Open loop	Open	Closed
9	Encoder feedback for stepper motors	Normal	Internal

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting in hexadecimal notation.

Rel. Commands	ZE	—	Set e-stop configuration.
	ZH	—	Set hardware limit configuration.
	ZS	—	Set software limit configuration.
	ZZ	—	Set general system configuration.
	FE	—	Set following error threshold.
Example	**2ZF?**	|	*Read following error configuration of axis #2.*
	0107H	|	*Controller returns a value of 0107HH for axis #2.*
	2ZF5H	|	*Set following error configuration to 5H for axis #2.*

ZH — Set Hardware Limit Configuration

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**ZH**nn or xx**ZH**?

Parameters

Description

xx [int]	—	Axis number.
nn [int]	—	Hardware limit configuration.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **0FFFFH** (hexadecimal with leading zero(0)), or ? to read current setting.

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.
	critical setting:	Error x17, ESP CRITICAL SETTINGS ARE PROTECTED.

Description This command is used to set the hardware limit checking, polarity, and event handling for axis specified with **xx**.

NOTE

If bit-0 or both bits-1 and -2 are set to zero(0) then no action will be taken by the controller.

Hardware limit configuration

Bit #	Function	Meaning for Bit LOW	Meaning for Bit HIGH
0	Hardware Limit checking	Disabled	Enabled
1	disable motor power on hardware limit event	do not disable motor power on hardware limit event	disable motor power on hardware limit event
2	abort motion on hardware limit event	do not abort motion on hardware limit event	abort motion on hardware limit event
3	Reserved	Default	—
4	Reserved	Default	—
5	Limits and Home Polarity	—	Default
6	Reserved	Default	—
7	Reserved	Default	—
8	Reserved	Default	—
9	Reserved	Default	—

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting in hexadecimal notation.

Rel. Commands	ZA	—	Set amplifier I/O configuration.
	ZE	—	Set e-stop configuration.
	ZF	—	Set following error configuration.
	ZB	—	Set feedback configuration.
	ZS	—	Set software limit configuration.
	ZZ	—	Set general system configuration.
Example	**2ZH?**	|	*Read hardware limit configuration of axis #2.*
	07H	|	*Controller returns a value of 07H for axis #2.*
	2ZH06H	|	*Set hardware limit configuration to 06H for axis #2.*

ZS — Set Software Limit Configuration

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax xx**ZS**nn or xx**ZS**?

Parameters

Description

xx [int]	—	Axis number.
nn [int]	—	hardware limit configuration.

Range

xx	—	**1** to **Max. Axes**.
nn	—	**0** to **07H** (hexadecimal with leading zero(0)), or **?** to read current setting.

Units

xx	—	None.
nn	—	None.

Defaults

xx	Missing:	Error 37, AXIS NUMBER MISSING.
	Out of range:	Error 9, AXIS NUMBER OUT OF RANGE.
nn	Missing:	Error 38, COMMAND PARAMETER MISSING.
	Out of range:	Error x01, PARAMETER OUT OF RANGE.
	critical setting:	Error x17, ESP CRITICAL SETTINGS ARE PROTECTED.

Description This command is used to set the software limit checking and event handling for axis specified with **xx**.

NOTE

If bit-0 or both bits-1 and -2 are set to zero(0) then no action will be taken by the controller.

Software limit configuration

Bit #	Function	Meaning for Bit LOW	Meaning for Bit HIGH
0	Software limit checking	Disabled	Enabled
1	disable motor power on software limit event	do not disable motor power on software limit event	disable motor power on software limit event
2	abort motion on software limit event	do not abort motion on software limit event	abort motion on software limit event
3	Reserved	Default	—
4	Reserved	Default	—
5	Reserved	Default	—
6	Reserved	Default	—
7	Reserved	Default	—
8	Reserved	Default	—
9	Reserved	Default	—

Returns If the "**?**" sign takes the place of **nn** value, this command reports the current setting in hexadecimal notation.

Rel. Commands	ZA	—	Set amplifier I/O configuration.
	ZE	—	Set e-stop configuration.
	ZF	—	Set following error configuration.
	ZB	—	Set feedback configuration.
	ZH	—	Set hardware limit configuration.
	ZZ	—	Set general system configuration.
	SL	—	Set left limit.
	SR	—	Set right limit.
Example	**2ZS?**	|	*Read software limit configuration of axis #2.*
	07H	|	*Controller returns a value of 7H for axis #2.*
	2ZS5H	|	*Set software limit configuration to 5H for axis #2.*

ZU — Get ESP System Configuration

	IMM	PGM	MIP
Usage	♦	–	♦

Syntax **ZU**

Parameters None.

Description This command is used to get the present ESP system stage/driver configuration. After each system reset or initialization the ESP motion controller detects the presence of Universal drivers and ESP-compatible stages connected.

ESP configuration

Bit#	Value	Definition
0	0	axis-1 universal driver not detected
0	1	axis-1 universal driver detected
1	0	axis-2 universal driver not detected
1	1	axis-2 universal driver detected
2	0	axis-3 universal driver not detected
2	1	axis-3 universal driver detected
3	0	reserved
3	1	reserved
4	0	reserved
4	1	reserved
5	0	reserved
5	1	reserved
6	0	reserved
6	1	reserved
7	0	reserved
7	1	reserved

Returns This command reports the current setting in hexadecimal notation.

Rel. Commands

ZA	—	Set amplifier I/O configuration.
ZB	—	Set feedback configuration.
ZE	—	Set e-stop configuration.
ZF	—	Set following error configuration.
ZH	—	Set hardware limit configuration.
ZS	—	Set software limit configuration.
ZZ	—	Set system configuration.

Example

ZU	|	*Read ESP system configuration.*
03H	|	*Controller returns a value of 03H.*

ZZ — Set System Configuration

	IMM	PGM	MIP
Usage	♦	♦	–

Syntax **ZZ**nn or **ZZ**?

Parameters

Description **nn** [int] — System configuration.

Units **nn** — None.

Description Obsolete command, but kept for backward compatibility.

This command has no effect.

Appendix

3.0 Error Messages

The ESP302 controller has an elaborate command interpreter and system monitor. Every command is analyzed for syntax and correct format after it is received. The result of the analysis is stored in an output buffer in plain English. During moves and while idle, system inputs are monitored and any change is reported to the user via the output buffer. To read the contents of the output buffer, send the command **TB** (tell buffer).

For more compact error messages, use the **TE** command. The ESP302 controller response to this command is a one byte; binary coded error number, e.g., 33.

For the sake of convenience, error messages are divided into two categories – non-axis specific error messages and axis specific error messages. Below is a list of all possible ESP302 controller error messages that are not axis specific:

0 NO ERROR DETECTED

No errors exist in the output buffer.

1 COMMUNICATION TIME-OUT

A communication transfer was initiated and was never completed.

2 Reserved for future use

3 Reserved for future use

4 EMERGENCY STOP ACTIVATED

An emergency stop was executed because the motion controller received a '#' character or "STOP ALL AXES" button was pressed.

5 Reserved for future use

6 COMMAND DOES NOT EXIST

The issued command does not exist. Check the Command Syntax.

7 PARAMETER OUT OF RANGE

The specified parameter is out of range. Refer to the description of issued command for valid parameter range.

8 Reserved for future use

9 AXIS NUMBER OUT OF RANGE

The specified axis number is out of range. Refer to the description of issued command for valid axis number range.

10 Reserved for future use

11 Reserved for future use

12 Reserved for future use

13 GROUP NUMBER MISSING

Group number is not specified. The issued command requires a valid group number. Refer to the description of issued command for valid group number range.

14 GROUP NUMBER OUT OF RANGE

The specified group number is out of range. Refer to the description of issued command for valid group number range.

15 GROUP NUMBER NOT ASSIGNED

No group has been assigned. Refer to the description of HN command to create a new group.

16 GROUP NUMBER ALREADY ASSIGNED

A group has already been assigned. Delete group with HX command before creating a new group with HN command.

17 GROUP AXIS OUT OF RANGE

At least one of the axis numbers specified to be a member of this group is out of range. Refer to the description of HN command for valid range of axis numbers that can be assigned to a group.

18 GROUP AXIS ALREADY ASSIGNED

At least one of the axis numbers specified to be a member of this group is already a member of a different group.

19 GROUP AXIS DUPLICATED

At least one of the axis numbers is specified to be a member of this group more than once.

20 DATA ACQUISITION IS BUSY

Data acquisition is not yet complete.

21 DATA ACQUISITION SETUP ERROR

An error occurred during data acquisition setup. Ensure that data acquisition is disabled and all parameters are within valid range before issuing the command. Refer to the command description for valid range of parameters.

22 DATA ACQUISITION NOT ENABLED

Data acquisition is not yet enabled.

23 Reserved for future use

24 COMMAND SYNTAX ERROR

Syntax error in program compilation

25 Reserved for future use

26 STORED PROGRAM NOT STARTED

An attempt was made to execute a stored program and the program could not be started.

27 COMMAND NOT ALLOWED

The issued command is not valid in the context in which it was issued.

28 FILE SYSTEM OR MEMORY ERROR

The user area reserved for stored programs is full.

29 GROUP PARAMETER MISSING

At least one parameter is missing. Refer to the description of issued command for valid number of parameters.

30 GROUP PARAMETER OUT OF RANGE

The specified group parameter is out of range. Refer to the description of issued command for valid range of parameter.

31 GROUP MAXIMUM VELOCITY EXCEEDED

The specified group velocity exceeds the minimum of the maximum velocities of members of this group. Refer to the description of HV command for more details.

32 GROUP MAXIMUM ACCELERATION EXCEEDED

The specified group acceleration exceeds the minimum of the maximum acceleration of members of this group. Refer to the description of HA command for more details.

33 GROUP MAXIMUM DECELERATION EXCEEDED

The specified group deceleration exceeds the minimum of the maximum decelerations of members of this group. Refer to the description of HD command for more details.

34 GROUP MOVE NOT ALLOWED DURING MOTION

Cannot make a coordinated move when one of the members of the group is being "homed".

35 PROGRAM NOT FOUND

The issued command could not be executed because the stored program requested is not available.

36 PROGRAM HALTED

Program halted due to a command execution error

37 AXIS NUMBER MISSING

Axis number not specified. The issued command requires a valid axis number. Refer to the description of issued command for valid axis number range.

38 COMMAND PARAMETER MISSING

At least one parameter associated with this command is missing. Refer to the description of issued command for valid number of parameters.

39 PROGRAM LABEL NOT FOUND

The issued command could not be executed because the requested label within a stored program is not available.

40 Reserved for future use

41 MAX NUMBER OF LABELS PER PROGRAM EXCEEDED

The number of labels used in the stored program exceeds the allowed value.

42 Reserved for future use

43 TRAJECTORY EXECUTION EXCEED TRAVEL

The trajectory exceed software travel limits. Refer to the description of **SR** or **SL** commands to specify the desired software travel limits.

44 TRAJECTORY NON-EXISTENT

Impossible to execute a trajectory because the trajectory buffer is empty.

45 TRAJECTORY IS TOO LONG

Impossible to add a new trajectory element because the trajectory buffer is full.

46 Reserved for future use

47 UNITS NOT TRANSLATIONAL OR NOT IDENTICAL

The group axis units shall be identical and translational. Refer to the description of **SN** command to specify the desired axis unit.

48 TRAJECTORY ARC RADIUS IS TOO SMALL

Radius of HC trajectory element is too small (<1e-12)

49 TRAJECTORY ARC RADIUS IS TOO LARGE

Radius of HC trajectory element is too large (>1e100)

50 TRAJECTORY LINE TYPE EXPECTED

Error in trajectory parser, a line type was expected.

51 TRAJECTORY LINE DISCONTINUITY IS TOO BIG

The tangeant discontinuity between two trajectories elements is too big (>45°).

52 TRAJECTORY LINE IS IMPOSSBIBLE

The line length is too small (<1e-12).

53 TRAJECTORY ARC TYPE EXPECTED

Error in trajectory parser, an arc type was expected.

54 TRAJECTORY ARC SWEEP ANGLE IS TOO SMALL

The arc angle is too small (<1e-12).

55 Reserved for future use

56 TRAJECTORY ARC CIRCLE IS IMPOSSIBLE

Trajectory arc tangeant from last element exceed 1.15°.

57 Reserved for future use

58 ERROR CALCULATION OVERFLOW

An overflow occurred during a trajectory calcul (division by zero).

Below is a list of all possible error messages that are axis specific. Here, "x" represents the axis number.

x00 MOTOR TYPE NOT DEFINED

A valid motor type was not defined for the requested axis. Refer to the description of **QM** command to define a motor type.

x01 PARAMETER OUT OF RANGE

The specified parameter is out of range. Refer to the description of issued command for valid parameter range.

x02 AMPLIFIER FAULT DETECTED

There was an amplifier fault condition.

x03 FOLLOWING ERROR THRESHOLD EXCEEDED

The real position of specified axis was lagging the desired position by more encoder counts than specified with the **FE** command. Refer to the description of **ZF** command to configure the motion controller tasks upon encountering a following error.

x04 POSITIVE HARDWARE LIMIT DETECTED

The motion controller sensed a high level at its positive travel limit input. Refer to the description of **ZH** command to configure the motion controller tasks upon encountering a hardware limit.

x05 NEGATIVE HARDWARE LIMIT DETECTED

The motion controller sensed a high level at its negative travel limit input. Refer to the description of **ZH** command to configure the motion controller tasks upon encountering a hardware limit.

x06 POSITIVE SOFTWARE LIMIT DETECTED

The motion controller sensed that the axis has reached positive software travel limit. Refer to the description of **SR** command to specify the desired positive software travel limit. Also, refer to the description of **ZS** command to configure the motion controller tasks upon encountering a software limit.

x07 NEGATIVE SOFTWARE LIMIT DETECTED

The motion controller sensed that the axis has reached negative software travel limit. Refer to the description of **SL** command to specify the desired negative software travel limit. Also, refer to the description of **ZS** command to configure the motion controller tasks upon encountering a software limit.

x08 MOTOR / STAGE NOT CONNECTED

The specified axis is not connected to the driver.

x09 FEEDBACK SIGNAL FAULT DETECTED

There was a feedback signal fault condition. Ensure that the encoder feedback is relatively noise free.

x10 MAXIMUM VELOCITY EXCEEDED

The specified axis velocity exceeds maximum velocity allowed for the axis. Refer to the description of **VU** command or set maximum velocity for the axis.

x11 MAXIMUM ACCELERATION EXCEEDED

The specified axis acceleration exceeds maximum acceleration allowed for the axis. Refer to the description of **AU** command to query or set maximum acceleration or deceleration for the axis.

x12 Reserved for future use

x13 MOTOR NOT ENABLED

A command was issued to move an axis that was not powered ON. Refer to the description of **MO** and **MF** commands to turn the power to an axis ON or OFF respectively.

x14 Reserved for future use

x15 MAXIMUM JERK EXCEEDED

The specified axis jerk exceeds maximum jerk allowed for the axis. Refer to the description of **JK** command for valid jerk range.

x16 Reserved for future use

x17 Reserved for future use

x18 Reserved for future use

x19 Reserved for future use

x20 HOMING ABORTED

Axis home search was aborted. This message is obtained when home search was not completed either due to an axis not being enabled or due to the occurrence of a fault condition. Refer to the description of **OR** command for information related to locating the home position of an axis.

x21 Reserved for future use

x22 Reserved for future use

x23 Reserved for future use

x24 Reserved for future use

x25 Reserved for future use

x26 Reserved for future use

x27 Reserved for future use

x28 Reserved for future use

x29 DIGITAL I/O INTERLOCK DETECTED

A motion was requested while general inhibit was asserted.

x30 Reserved for future use

x31 COMMAND NOT ALLOWED DUE TO GROUP ASSIGNMENT

The specified command was not executed because this axis is member of a group. Refer to the description of issued command for further details.

x32 Reserved for future use

x33 Reserved for future use

x34 Reserved for future use

x35 DRIVER I2C READ / WRITE ERROR

An error occurred during communication with the driver board

x36 GROUP TRAJECTORY ERROR

A following error occurred during trajectory execution

x37 STAGE NOT IN CONTROLLER DATABASE

The connected stage cannot be found in the controller stage database

4.0 Binary Conversion Table

Some of the status reporting commands return an ASCII character that must be converted to binary. To aid with the conversion process, the following table converts all character used and some other common ASCII symbols to decimal and binary. To also help in working with the I/O port related commands, the table is extended to a full byte, all 256 values.

Number (decimal)	ASCII Code	Binary Code
0	*Null*	00000000
1	*Soh*	00000001
2	*Stx*	00000010
3	*Etx*	00000011
4	*Eot*	00000100
5	*Enq*	00000101
6	*Ack*	00000110
7	*Bel*	00000111
8	*Bs*	00001000
9	*Tab*	00001001
10	*Lf*	00001010
11	*Vt*	00001011
12	*Ff*	00001100
13	*Cr*	00001101
14	*So*	00001110
15	*Si*	00001111
16	*Dle*	00010000
17	*Dc1*	00010001
18	*Dc2*	00010010
19	*Dc3*	00010011
20	*Dc4*	00010100
21	*Nak*	00010101
22	*Syn*	00010110
23	*Eth*	00010111
24	*Can*	00011000
25	*Em*	00011001
26	*Eof*	00011010
27	*Esc*	00011011
28	*Fs*	00011100
29	*Gs*	00011101
30	*Rs*	00011110
31	*Us*	00011111
32	*Space*	00100000
33	*!*	00100001
34	*"*	00100010
35	*#*	00100011
36	*$*	00100100
37	*%*	00100101
38	*&*	00100110
39	*'*	00100111
40	*(*	00101000
41	*)*	00101001
42	*	00101010
43	+	00101011
44	,	00101100
45	-	00101101
46	.	00101110
47	/	00101111
48	*0*	00110000
49	*1*	00110001
50	*2*	00110010
51	*3*	00110011
52	*4*	00110100
53	*5*	00110101

54	6	00110110
55	7	00110111
56	8	00111000
57	9	00111001
58	:	00111010
59	;	00111011
60	<	00111100
61	=	00111101
62	>	00111110
63	?	00111111
64	@	01000000
65	A	01000001
66	B	01000010
67	C	01000011
68	D	01000100
69	E	01000101
70	F	01000110
71	G	01000111
72	H	01001000
73	I	01001001
74	J	01001010
75	K	01001011
76	L	01001100
77	M	01001101
78	N	01001110
79	O	01001111
80	P	01010000
81	Q	01010001
82	R	01010010
83	S	01010011
84	T	01010100
85	U	01010101
86	V	01010110
87	W	01010111
88	X	01011000
89	Y	01011001
90	Z	01011010
91	[	01011011
92	\	01011100
93	]	01011101
94	^	01011110
95		01011111
96	'	01100000
97	A	01100001
98	B	01100010
99	C	01100011
100	D	01100100
101	E	01100101
102	F	01100110
103	G	01100111
104	H	01101000
105	I	01101001
106	J	01101010
107	K	01101011
108	L	01101100
109	M	01101101
110	N	01101110
111	O	01101111
112	P	01110000
113	Q	01110001
114	R	01110010
115	S	01110011
116	T	01110100
117	U	01110101
118	V	01110110

119	*W*	01110111
120	*X*	01111000
121	*Y*	01111001
122	*Z*	01111010
123	*{*	01111011
124	|	01111100
125	*}*	01111101
126	~	01111110
127		01111111
128		10000000
129		10000001
130		10000010
131		10000011
132		10000100
133		10000101
134		10000110
135		10000111
136		10001000
137		10001001
138		10001010
139		10001011
140		10001100
141		10001101
142		10001110
143		10001111
144		10010000
145		10010001
146		10010010
147		10010011
148		10010100
149		10010101
150		10010110
151		10010111
152		10011000
153		10011001
154		10011010
155		10011011
156		10011100
157		10011101
158		10011110
159		10011111
160		10100000
161		10100001
162		10100010
163		10100011
164		10100100
165		10100101
166		10100110
167		10100111
168		10101000
169		10101001
170		10101010
171		10101011
172		10101100
173		10101101
174		10101110
175		10101111
176		10110000
177		10110001
178		10110010
179		10110011
180		10110100
181		10110101
182		10110110
183		10110111

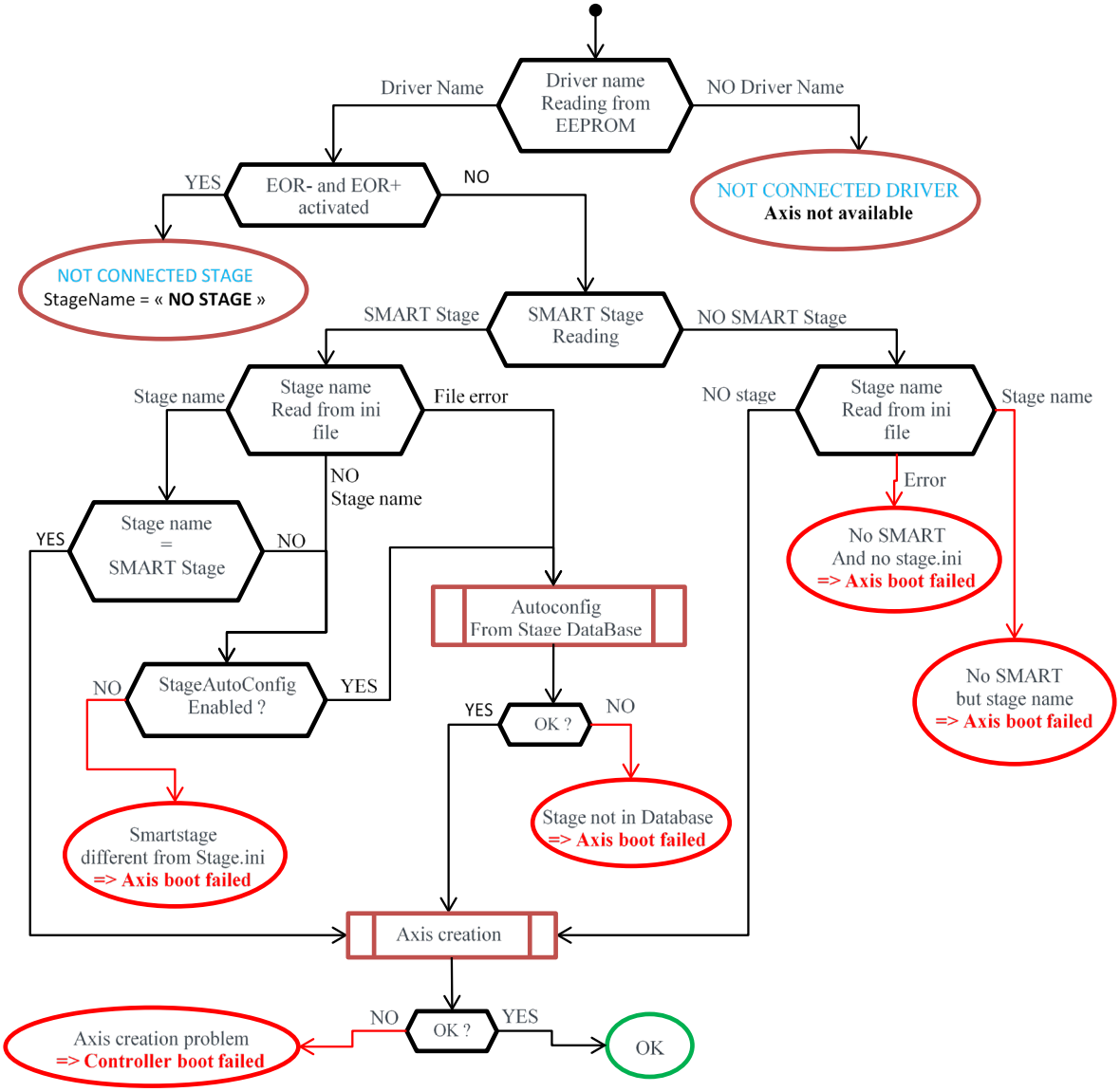
184		10111000
185		10111001
186		10111010
187		10111011
188		10111100
189		10111101
190		10111110
191		10111111
192		11000000
193		11000001
194		11000010
195		11000011
196		11000100
197		11000101
198		11000110
199		11000111
200		11001000
201		11001001
202		11001010
203		11001011
204		11001100
205		11001101
206		11001110
207		11001111
208		11010000
209		11010001
210		11010010
211		11010011
212		11010100
213		11010101
214		11010110
215		11010111
216		11011000
217		11011001
218		11011010
219		11011011
220		11011100
221		11011101
222		11011110
223		11011111
224		11100000
225		11100001
226		11100010
227		11100011
228		11100100
229		11100101
230		11100110
231		11100111
232		11101000
233		11101001
234		11101010
235		11101011
236		11101100
237		11101101
238		11101110
239		11101111
240		11110000
241		11110001
242		11110010
243		11110011
244		11110100
245		11110101
246		11110110
247		11110111
248		11111000

249		11111001
250		11111010
251		11111011
252		11111100
253		11111101
254		11111110
255		11111111

Table 13: Binary Conversion Table (using decimal and ASCII codes).

5.0 ESP Configuration Logic

Each time a stage or stages are disconnected/re-connected, or a system is powered down and then powered back up, the ESP302 controller card verifies the type of stage(s) present and re-configures its own flash memory if necessary (i.e., new stage). The controller card in the ESP302 system configuration, the stage motor and the current type are defined, the controller card will configure the specific axis. Specific ESP logic is shown below.



Service Form

Your Local Representative

Tel.: _____

Fax:_____

Name: _____ Return authorization #:
(Please obtain prior to return of item)

Company:_____

Address:_____ Date:

Country:_____Phone Number:

P.O. Number:_____Fax Number:

Item(s) Being Returned:_____

Model#: _____Serial #:

_____ **Description**:

Reasons of return of goods (please list any specific problems): _____

Visit Newport Online at:

www.newport.com

North America & Asia
Newport Corporation
1791 Deere Ave.
Irvine, CA 92606, USA
Sales
Tel.: +1 (949)-863-3144
e-mail: sales@newport.com
Technical Support
Tel.: +1 (949)-863-3144
e-mail: tech@newport.com
Service, RMAs & Returns
Tel.: +1 (949)-863-3144
e-mail: service@newport.com

Europe
MICRO-CONTROLE Spectra-Physics S.A.S
7 rue des Plantes
45340 Beaune-la-Rolande
France

Sales Europe (EMEAI)
Tel.: +49 (0) 6151-708-0
e-mail: germany@newport.com

Sales France
Tel.: +33 (0)1 60 91 68 68
e-mail: france@newport.com

Sales UK
Tel.: +44 (0)1235 432 710
e-mail: uk@newport.com

Technical Support
e-mail: tech_europe@newport.com

Service & Returns
Tel.: +33 (0)2 38 40 51 55
DST-BEA-RMA-service@newport.com