IDEA™ Drive

Communications Manual

Haydon Motion Solutions kerk™

www.haydonkerk.com

4-2013

Table of Contents

Revision History

Date	Description
October 2010	Initial release
January 2011	Added “Execute Program” command.
May 2011	Corrected response from Program command
September 2011	Added information about faults Added Read Moving command Updated configure encoder command Alphabetized commands
December 2011	Corrected configure encoder example
April 2013	Corrected program description Corrected table of contents

IDEA Drive Communications Basics

The IDEA drive line of products are commanded through the use of an Ascii based language developed by Haydon Kerk. Each command consists of a character identifying the command, followed by between 0 and 12 parameters separated by commas, and then followed by a carriage return. One difference between this language and those used by competing products is that each motion command encapsulates all parameters needed by the move; there are no parameters to set before a move command is issued. While this makes manual entry of commands into a terminal cumbersome, this is not the intended use of the language. Creation of these commands can be done simply in the software of the controller used to command the drives.

The IDEA drive adheres to a master/slave communications model. The master controller initiates all communications. If information is required from the drive, as in the case of requesting the drive's current position, the controller first sends the command requesting the drive's position, then the drive responds with the requested information, enclosed by several characters to identify the response. The extra characters can then be parsed, and the response read.

For the RS-485 communication option, several drives can be daisy chained together on a single bus. This allows a single controller to send commands to all the drives at once. In this configuration, for each drive to be controlled separately, they must each be given a unique identifier, a number between 0 and 255. This must be done with only one drive attached. The user interface has a function built in to make this process simple. Once each drive on the bus has its own identifier, any command that is sent starting with the '#" character followed by an identifier, followed by the normal command, will be ignored by any drive whose identifier does not match the provided identifier. For example, to send an abort command to the drive whose identifier is 123, the controller would send "#123A" followed by a carriage return. If a command should be executed by all drives at once, the controller would omit the pound and identifier. It is important that the controller never request a response from all the drives at once, as this will cause a data collision when all the drives attempt to respond at once.

One major difference between using this command set to control the drive, and using the IDEA drive user interface is, there are no protections when using the command language. The user interface ensures that based upon the part number entered, no improper values are sent to the drive; with this command set, it is the responsibility of the user to ensure that no damage is done to the drive, motor, or other equipment through the incorrect use of commands.

The parameters for serial communication are as follows:

Bits per Second: 57600

Data bits: 8

Parity: none

Stop Bits: 1

Flow Control: None

Commands

The following describes the commands that make up the IDEA drive communications language, as well as the format for any response required from the drive. When quotation marks are present, the text in between the quotation marks is the important string, and the quotation marks themselves should not be included. When [cr] is shown, it is referring to the Ascii carriage return character, not to be confused with a line feed character. When [parameter] is shown, where parameter is the name of a parameter, it is representing some variable with that name, and the brackets will not be part of the string.

The contexts listed below indicate when each command can be used. Realtime commands can only be executed by direct command to the drive, such as requesting the current position. Program commands can only be a part of a program, and are generally branching or similar commands, such as Goto. Realtime/Program commands can be used anytime, and are generally motion related commands, such as Index. For further explanation of the commands, refer to the IDEA drive users' manual.

Command	Symbol	Context	Arguments	Response
Abort	A	Realtime/Program	none	None
Description	This command causes the drive to immediately stop, and ends the execution and of any programs.			
Arguments	**Argument Description**			**Valid Values or Range**
none				
Example	You want to stop all drive activity.			
Command	"A" followed by a carriage return.			

Command	Symbol	Context	Arguments	Response
Assign Drive Number	y	Realtime	Identifier	None
Description	This command assigns the drive an identifier.			
Arguments	**Argument Description**			**Valid Values or Range**
Identifier	The number that should be associated with the drive.			0 to 255
Example	You want to set the drive's identifier to 136.			
Command	"y136" followed by a carriage return.			

Command	Symbol	Context	Arguments	Response
Check Password	c	Realtime	Password	"`cYES[cr]`c#[cr]" or "`cNO[cr]`c#[cr]"
Description	This command checks to see if a password is the correct password.			
Arguments	**Argument Description**			**Valid Values or Range**
Password	The password in question.			A string, exactly 10 characters long
Example	You want to check if the password is "password ".			
Command	"cpassword " followed by a carriage return.			

Command	Symbol	Context	Arguments	Response
Comment	C	Program	Comment	None
Description	This command creates a comment in the program.			
Arguments	**Argument Description**			**Valid Values or Range**
Comment	A string, must be exactly 10 characters long.			
Example	You want to add a comment that says "Extend 1in".			
Command	"CExtend 1in" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Configure Encoder	z	Realtime/Program	DeadBand, StallHunts, Destination, Priority	None
Description	This command configures the encoder.			
Arguments	**Argument Description**			**Valid Values or Range**
DeadBand	The number of 1/64th steps away from the desired location where the drive will begin to correct.			1 to 65535, or 0 to disable
Stall Hunts	The number of attempts at a given move the drive will make.			0 to 255
Destination	The address of the subroutine that should be run after all stall hunts are exhausted, if desired.			0 to 86012, multiples of four only. Must be the address of a valid command.
Priority	The priority of the interrupt for when the stall hunts are exhausted.			0 to 4, 10 to disable
Encoder Resolution	The resolution of the encoder being used in pulses per channel per revolution.			Motor resolution to 10000
Motor Resolution	The resolution of the motor being used, in full steps per revolution.			20 to 400
Example	You have a 1000 line encoder, a 1.8º motor, and you want the drive to correct for position errors greater than 1 full step, retry moves twice, and do not want to trigger an interrupt after the second failure.			
Command	"z64,2,0,10,1000,200" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Configure Input Interrupts	i	Program	Input1 config, input2 config, input3 config, input4 config, intput1 destination, input2 destination, input3 destination, input4 destination, input1 priority, input2 priority, input3 priority, input4 priority	None
Description	This command is used to configure the interrupt settings for in inputs.			
Arguments	**Argument Description**			**Valid Values or Range**
Config	What kind of interrupt the input should be. 1 for Falling edge, 2 for rising edge, 3 for both edges, 0 for disabled.			0,1,2,3
Destination	The address of the subroutine that should handle the interrupt.			0 to 87036, multiples of four only.
Priority	The priority of the interrupt; lower numbered priorities are handled first.			0 to 4
Example	You want to set a rising edge interrupt on input 2, whose destination is address 512 and priority is 1, and all other input interrupts disabled.			
Command	"i0,2,0,0,0,512,0,0,4,1,4,4" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**

E-Stop	E	RealTime/Program	Decel Current, Hold Current, Delay Time	None
Description	This command stops the motor without decelerating.			
Arguments	**Argument Description**			**Valid Values or Range**
Decel Current	The rms current, in milliamps, used to stop the motor.			0 to 5005, dependant on Drive
Hold Current	The rms current, in milliamps, for after the motor has stopped.			0 to 3850, dependant on Drive
Delay	The time, in milliamps, between the last step of a move and when the current is set to the hold current.			50 to 300
Example	You wish to immediately stop the motor with a decel current of 2.0 Arms, and waiting .05 seconds between the last step and changing to a hold current of 0.5 Arms.			
Command	"E2000,500,50" followed by a carriage return			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Execute Program	m	Realtime	Program name	None
Description	This command begins the execution of a program without changing the state of the outputs or motor.			
Arguments	**Argument Description**			**Valid Values or Range**
Program Name	The name of the program to run.			A string, exactly 10 characters long
Example	You want to run a program named "program 1 ", without returning to the default state.			
Command	"mprogram 1 " followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**

Go At Speed	Q	RealTime/Program	Speed, Start Speed, End Speed, Accel, Decel, Run Current, Hold Current, Accel Current, Decel Current, Delay Time, Step Mode	None
Description	This command moves the motor to a position, with the given parameters.			
Arguments	**Argument Description**			**Valid Values or Range**
Run Speed	The number of steps per second the motor should move at the top speed, in the given step mode.			0 or -50 to -75000 or 50 to 75000
Start Speed	The number of steps per second the motor should move when starting the move, in the given step mode.			0 or 50 to 75000 Must be less than Run Speed
End Speed	The number of steps per second the motor should move when ending the move, in the given step mode.			0 or 50 to 75000 Must be less than Run Speed
Accel Rate	Rate at which the speed should rise from the Start Speed to the Run Speed.			0, or 500 to 16777215
Decel Rate	Rate at which the speed should fall from the Run Speed to the Final Speed.			0, or 500 to 16777215
Run Current	The rms current, in milliamps for the move.			0 to 3850, dependant on Drive
Hold Current	The rms current, in milliamps, for after the move has completed.			0 to 3850, dependant on Drive
Accel Current	The rms current, in milliamps, for the acceleration portion of the move.			0 to 5005, dependant on Drive
Decel Current	The rms current, in milliamps, for the deceleration portion of the move.			0 to 5005, dependant on Drive
Delay	The time, in milliseconds, between the last step of a move and when the current is set to the hold current.			50 to 300
Step Mode	Defines the step size, where 1 is a full step, 2 is a half step, and so on.			1,2,4,8,16,32,64.
Example	Desired move backwards, in 1/8th step mode, at a speed of 3200 1/8th steps per second, starting at 1200 1/8th steps per second, accelerating at a rate of 40000 1/8th steps per second per second, decelerating at a rate of 100000 1/8th steps per second per second to an end speed of 2000 1/8th steps per second, with a run current of 1.6 Arms, accel current of 1.9 Arms, decel current of 2.0 Arms, and waiting .05 seconds between the last step and changing to a hold current of 0.5 Arms.			
Command	"Q-3200,1200,2000,40000,100000,1600,500,1900,2000,50,8" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Goto	G	Program	Destination	None
Description	This command causes the program to continue execution at the specified address.			
Arguments	**Argument Description**			**Valid Values or Range**

Destination	The address of the command that should be run	0 to 86012, multiples of four only. Must be the address of a valid command.
Example	You want to continue execution at address 1024.	
Command	"G1024" followed by a carriage return.	

Command	**Symbol**	**Context**	**Arguments**	**Response**
Goto If	L	Program	Destination, Condition	None
Description	This command causes the program to continue execution at the specified address if the condition is met.			
Arguments	**Argument Description**			**Valid Values or Range**
Destination	The address of the command that should be run.			0 to 86012, multiples of four only. Must be the address of a valid command.
Condition	2 bytes indicating which I/O are tested, and the test values for each. The least significant byte corresponds to the inputs, and the most significant byte corresponds to the outputs. For each byte, the least significant nibble represents the condition being tested, a 1 meaning a high input or output, and a 0 representing a low input or output. The more significant nibble decides which of those conditions are to be tested, with a 1 representing an input or output should be tested. The least significant bit corresponds to input1, the next to input 2, and so on.			0 to 65535
Example	You want to continue execution at address 1024 if input 2 is high.			

Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Total
0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	34

Command	"L1024, 34" followed by a carriage return.

Command	**Symbol**	**Context**	**Arguments**	**Response**
Goto Sub	S	Program	Destination	None
Description	This command causes the program to execute the subroutine at the given destination.			
Arguments	**Argument Description**			**Valid Values or Range**
Destination	The address of the subroutine that should be run.			0 to 86012, multiples of four only. Must be the address of a valid command.
Example	You want to run a subroutine at address 1024.			
Command	"S1024" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Index	I	RealTime/Program	Distance, Speed, Start Speed, End Speed, Accel, Decel, Run Current, Hold Current, Accel Current, Decel Current, Delay Time, Step Mode	None

Description	This command moves the motor forward or backwards a defined number of steps, with the given parameters.	
Arguments	**Argument Description**	**Valid Values or Range**
Distance	The positive or negative number of 1/64th steps the motor should move.	-18446744073709551616 to 18446744073709551615
Run Speed	The number of steps per second the motor should move at the top speed, in the given step mode.	0 or 50 to 75000
Start Speed	The number of steps per second the motor should move when starting the move, in the given step mode.	0 or 50 to 75000 Must be less than Run Speed
End Speed	The number of steps per second the motor should move when ending the move, in the given step mode.	0 or 50 to 75000 Must be less than Run Speed
Accel Rate	Rate at which the speed should rise from the Start Speed to the Run Speed.	0, or 500 to 16777215
Decel Rate	Rate at which the speed should fall from the Run Speed to the Final Speed.	0, or 500 to 16777215
Run Current	The rms current, in milliamps for the move.	0 to 3850, dependant on Drive
Hold Current	The rms current, in milliamps, for after the move has completed.	0 to 3850, dependant on Drive
Accel Current	The rms current, in milliamps, for the acceleration portion of the move.	0 to 5005, dependant on Drive
Decel Current	The rms current, in milliamps, for the deceleration portion of the move.	0 to 5005, dependant on Drive
Delay	The time, in milliseconds, between the last step of a move and when the current is set to the hold current.	50 to 300
Step Mode	Defines the step size, where 1 is a full step, 2 is a half step, and so on.	1,2,4,8,16,32,64.
Example	Desired move is backwards 9600 1/64th steps, in 1/8th step mode, at a speed of 3200 1/8th steps per second, starting at 1200 1/8th steps per second, accelerating at a rate of 40000 1/8th steps per second per second, decelerating at a rate of 100000 1/8th steps per second per second to an end speed of 2000 1/8th steps per second, with a run current of 1.6 Arms, accel current of 1.9 Arms, decel current of 2.0 Arms, and waiting .05 seconds between the last step and changing to a hold current of 0.5 Arms.	
Command	"I-9600,3200,1200,2000,40000,100000,1600,500,1900,2000,50,8" followed by a carriage return.	

Command	Symbol	Context	Arguments	Response
Interrupt on Position	T	Program	Position, Destination, Priority	None
Description	This command sets an interrupt to occur at a given position.			
Arguments	**Argument Description**			**Valid Values or Range**
Position	The position where the interrupt should be triggered.			-18446744073709551616 to 18446744073709551615
Destination	The address of the subroutine to be run when the interrupt is triggered.			0 to 86012, multiples of four only. Must be the address of a valid command.
Priority	The priority of the interrupt; lower values are a higher priority.			0 to 4, 10 to disable
Example	You want to set a trip point at position 0, that runs a subroutine at address 1024, and has the highest priority.			
Command	"T0,1024,0" followed by a carriage return			

Command	Symbol	Context	Arguments	Response
Jump N Times	J	Program	Destination, Jumps	None
Description	This command causes the program to continue execution at the specified address a specified number of times.			
Arguments	**Argument Description**			**Valid Values or Range**
Destination	The address of the command that should be run.			0 to 86012, multiples of four only. Must be the address of a valid command.
Jumps	The number of times execution should branch to the destination address.			0 to 65535
Example	You want to continue execution at address 1024, and do so 3 times.			
Command	"J1024, 3" followed by a carriage return.			

Command	Symbol	Context	Arguments	Response
Label	B	Program	Label name	None
Description	This command creates a label in the program.			
Arguments	**Argument Description**			**Valid Values or Range**
Label Name	A string, must be exactly 10 characters long.			
Example	You want to add a label called "Start".			
Command	"BStart " followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Move To Position	M	RealTime/Program	Position, Speed, Start Speed, End Speed, Accel, Decel, Run Current, Hold Current, Accel Current, Decel Current, Delay Time, Step Mode	None
Description	This command moves the motor to a position, with the given parameters.			
Arguments	**Argument Description**			**Valid Values or Range**
Position	The positive or negative position, based on 1/64th steps, the motor should move to.			-18446744073709551616 to 18446744073709551615
Run Speed	The number of steps per second the motor should move at the top speed, in the given step mode.			0 or 50 to 75000
Start Speed	The number of steps per second the motor should move when starting the move, in the given step mode.			0 or 50 to 75000 Must be less than Run Speed
End Speed	The number of steps per second the motor should move when ending the move, in the given step mode.			0 or 50 to 75000 Must be less than Run Speed
Accel Rate	Rate at which the speed should rise from the Start Speed to the Run Speed.			0, or 500 to 16777215
Decel Rate	Rate at which the speed should fall from the Run Speed to the Final Speed.			0, or 500 to 16777215
Run Current	The rms current, in milliamps for the move.			0 to 3850, dependant on Drive
Hold Current	The rms current, in milliamps, for after the move has completed.			0 to 3850, dependant on Drive
Accel Current	The rms current, in milliamps, for the acceleration portion of the move.			0 to 5005, dependant on Drive
Decel Current	The rms current, in milliamps, for the deceleration portion of the move.			0 to 5005, dependant on Drive
Delay	The time, in milliseconds, between the last step of a move and when the current is set to the hold current.			50 to 300
Step Mode	Defines the step size, where 1 is a full step, 2 is a half step, and so on.			1,2,4,8,16,32,64.
Example	Desired move is to position 0, in 1/8th step mode, at a speed of 3200 1/8th steps per second, starting at 1200 1/8th steps per second, accelerating at a rate of 40000 1/8th steps per second per second, decelerating at a rate of 100000 1/8th steps per second per second to an end speed of 2000 1/8th steps per second, with a run current of 1.6 Arms, accel current of 1.9 Arms, decel current of 2.0 Arms, and waiting .05 seconds between the last step and changing to a hold current of 0.5 Arms.			
Command	"M0,3200,1200,2000,40000,100000,1600,500,1900,2000,50,8" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
No-op	w	Program	none	None
Description	This command is used to insert an extra line in a program.			
Arguments	**Argument Description**			**Valid Values or Range**
none				
Example	This command would be used in a custom user interface.			
Command	"w" followed by a carriage return.			
Command	**Symbol**	**Context**	**Arguments**	**Response**

Program	P	Realtime	(Program Name, Start Location, Length) or none	None or "`P[Program size][CR]`P#[CR]"
Description	This command starts and ends the process of writing a program.			
Arguments	**Argument Description**			**Valid Values or Range**
Program Name	The name for the program, if it is the same as a program already on the drive, the old program will be removed.			A string; must be exactly 10 characters.
Start Location	The page number where the program should begin. If the program overlaps with any other program, the old program will be deleted. Each page has 1024 bytes of space.			1 to 85
Length	The number of pages the program will take up.			1 to 85
Example	You want to write a program name program 1, on the first page of memory.			
Command	"Pprogram 1 , 1,1" followed by a carriage return. Then followed by the commands that make up the program, each separated by a carriage return, followed by "P" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Read Current Position	I	Realtime	None	"`I[value][cr]`I#[cr]" where value represents the motor position.
Description	This command requests the position of the motor either theoretical, or actual if an encoder is enabled.			
Arguments	**Argument Description**			**Valid Values or Range**
None				
Example	You want to check the position of the drive.			
Command	"I" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Read Drive Number	k	Realtime	None	"`k[value][cr]`k#[cr]" where [value] is a number.
Description	This command requests drive identifier.			
Arguments	**Argument Description**			**Valid Values or Range**
None				
Example	You want to read the drive's identifier.			
Command	"k" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**

Read Encoder Settings	b	Realtime	None	"`b[deadband],[stallhunts], [encoder_CPR][cr]`b#[cr]"
Description	This command requests the encoder configuration of the drive.			
Arguments	**Argument Description**			**Valid Values or Range**
None				
Example	You want to check the encoder settings on the drive.			
Command	"b" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Read Executing	r	Realtime	None	"`rYES[cr]`r#[cr]" or "`rNO[cr]`r#[cr]"
Description	This command requests whether the drive is actively running a program.			
Arguments	**Argument Description**			**Valid Values or Range**
None				
Example	You want to check if the drive is executing a program.			
Command	"r" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Read Faults	f	Realtime	None	"`f[value][cr]`f#[cr]" where value represents the errors present. Each bit represents a specific error, as defined below.
Description	This command requests the error status of the drive.			
Arguments	**Argument Description**			**Valid Values or Range**
None				
Example	You want to check the error status of the drive.			
Command	"f" followed by a carriage return.			

Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Over Speed	Bad Checksum	Current Limit	Loop Overflow	Int Queue Full	Encoder Error	Temperature	Stack Overflow	Stack Underflow

Command	**Symbol**	**Context**	**Arguments**	**Response**
Read Firmware Version	v	Realtime	None	"`v[value][cr]`v#[cr]" where [value] is a number.
Description	This command requests the firmware version of the drive.			
Arguments	**Argument Description**			**Valid Values or Range**
None				
Example	You want to check the firmware version on the drive.			
Command	"v" followed by a carriage return.			

	Symbol	**Context**	**Arguments**	**Response**

Command				
Read IO	:	Realtime	none	"`:[value][CR]`:#[CR]", Where [value] is a number between 0 and 255, formed from 1 byte, with ones being highs, zeros being lows, the most significant bit corresponding to output4, and the least significant bit corresponding to input1.
Description	This command requests the status of the inputs and outputs.			
Arguments	**Argument Description**			**Valid Values or Range**
none				
Example	Want to know the status of the input and outputs. For this example, outputs 1 and 2 will be high, and inputs 2, 3, and 4 will be high, all others will be low.			
Command	":" followed by a carriage return.			

Output4	Output 3	Output 2	Output 1	Input 4	Input 3	Input 2	Input 1	Value
0	0	1	1	1	1	1	0	62

Command	**Symbol**	**Context**	**Arguments**	**Response**
Read Max Current	j	Realtime	None	"`j[value][cr]`j#[cr]" where [value] is a number.
Description	This command requests the maximum current setting of the drive.			
Arguments	**Argument Description**			**Valid Values or Range**
None				
Example	You want to check the maximum current of the drive.			
Command	"j" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Read Moving	o	Realtime	None	"`oYES[cr]`o#[cr]" or "`oNO[cr]`o#[cr]"
Description	This command requests whether the drive is moving.			
Arguments	**Argument Description**			**Valid Values or Range**
None				
Example	You want to check if the drive is moving.			
Command	"o" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Read Program Names	N	Realtime	none	"`N[program1 name],[start page],[end page][CR]`N[program2 name],[start page],[end page][CR]`N#[CR]" More programs would have more entries.
Description	This command requests that all program names and addresses be sent.			
Arguments	**Argument Description**			**Valid Values or Range**
none				
Example	You want to know what programs are residing on the drive.			
Command	"N" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**

Read Startup Program	K	Realtime	none	"`K[program name][CR]`K#[CR]" If there is no startup program, [program name] will be an empty string.
Description	This command requests the name of the startup program.			
Arguments	**Argument Description**			**Valid Values or Range**
none				
Example	Want to know what program is set to run on power up.			
Command	"K" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Recall Program	@	Realtime	Password, Program Name	The commands that make up the program, unless the password was incorrect, in which case there is no response.
Description	This command requests the program be read back.			
Arguments	**Argument Description**			**Valid Values or Range**
Password	The password for the drive			A string; must be exactly 10 characters.
Program Name	The name of the program to be read back.			A string; must be exactly 10 characters.
Example	Want to read back a program named "program 1" from the drive, with no password.			
Command	"@ ,program 1 " followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Remove Password	q	Realtime	Password	None
Description	This command removes a password.			
Arguments	**Argument Description**			**Valid Values or Range**
Password	The current password			A string, exactly 10 characters long
Example	You want to remove the password "password ".			
Command	"qpassword " followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Remove Program	D	Realtime	Program name	None
Description	This command removes a program.			
Arguments	**Argument Description**			**Valid Values or Range**
Program Name	The name of the program to be deleted.			A string, exactly 10 characters long
Example	You want to remove a program named "program 1 " from the drive.			
Command	"Dprogram 1 " followed by a carriage return.			
Command	**Symbol**	**Context**	**Arguments**	**Response**
Restore Factory Defaults	a	Realtime	None	None

Description	This command removes the drive password and deletes all the programs on the drive.	
Arguments	**Argument Description**	**Valid Values or Range**
None		
Example	You want to remove the password on a drive, but forgot that password.	
Command	"a" followed by a carriage return.	

Command	**Symbol**	**Context**	**Arguments**	**Response**
Return	X	Program	none	None
Description	This command returns from a subroutine.			
Arguments	**Argument Description**			**Valid Values or Range**
none				
Example	You want to return from a subroutine to where the subroutine was called from.			
Command	"X" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Return To	V	Program	Destination	None
Description	This command exits a subroutine, branches to a location, and clears all pending interrupts, the return stack and the loop counters.			
Arguments	**Argument Description**			**Valid Values or Range**
Destination	The address to which the program should branch.			0 to 87036, multiples of four only.
Example	You want to exit a subroutine and continue execution somewhere other than where the subroutine was called from, in this case, address 32.			
Command	"V32" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Run Program	Y	Realtime	Program name	None
Description	This command begins the execution of a program, first returning to step 0 and setting all outputs low.			
Arguments	**Argument Description**			**Valid Values or Range**
Program Name	The name of the program to run.			A string, exactly 10 characters long
Example	You want to run a program named "program 1 ", starting from the default state.			
Command	"Yprogram 1 " followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Set Outputs	O	Realtime/Program	Output Value	None
Description	This command sets the state of the outputs.			
Arguments	**Argument Description**			**Valid Values or Range**

Output Value	1 byte indicating which outputs should be set and what they should be set to. The most significant nibble indicates which outputs are being set, and the least significant nibble controls what they are being set to.							0 to 255
Example	You want to set output 3 high, output 2 low, and want to leave outputs 1 and 4 unchanged.							
Bit 8 = 128	Bit 7 = 64	Bit 6 = 32	Bit 5 = 16	Bit 4 = 8	Bit 3 = 4	Bit 2 = 2	Bit 1 = 1	Total
0	1	1	0	0	1	0	0	100
Command	"O100" followed by a carriage return.							

Command	**Symbol**	**Context**	**Arguments**	**Response**
Set Password	p	Realtime	Password	None
Description	This command sets a password, if none exists.			
Arguments	**Argument Description**			**Valid Values or Range**
Password	The desired password.			A string, exactly 10 characters long
Example	You want to set the password as "password ".			
Command	"ppassword " followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Set Position As	Z	Realtime/Program	New Position	None
Description	This command adjusts the position counter.			
Arguments	**Argument Description**			**Valid Values or Range**
New Position	The position, as 1/64th steps, you would like the current position to become.			-18446744073709551616 to 18446744073709551615
Example	After homing, you want to set the current location to 0.			
Command	"Z0" followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Set Startup Program	U	Realtime	Program name	None
Description	This command sets a program as the startup program.			
Arguments	**Argument Description**			**Valid Values or Range**
Program Name	The name of the program to start on power up or reset.			A string, exactly 10 characters long
Example	You want to set a program named "program 1 " as the startup program.			
Command	"Uprogram 1 " followed by a carriage return.			

Command	**Symbol**	**Context**	**Arguments**	**Response**
Software Reset	R	Realtime/Program	none	None
Description	This command causes the drive to restart, acts the same as cycling power.			

Arguments	Argument Description	Valid Values or Range
none		
Example	You want to restart the drive.	
Command	"R" followed by a carriage return.	

Command	Symbol	Context	Arguments	Response
Stop	H	RealTime/Program	End Speed, Decel rate, run current, decel current, hold current, delay time, step mode	None
Description	This command stops the motor using an optional deceleration ramp.			
Arguments	**Argument Description**			**Valid Values or Range**
End Speed	The number of steps per second the motor should move when ending the move, in the given step mode.			0 or 50 to 75000 Must be less than Run Speed
Decel Rate	Rate at which the speed should fall from the current speed to the end speed.			0, or 500 to 16777215
Run Current	The rms current, in milliamps for the deceleration, if too long to use boosted decel current for the entire ramp.			0 to 3850, dependant on Drive
Hold Current	The rms current, in milliamps, for after the move has completed.			0 to 3850, dependant on Drive
Decel Current	The rms current, in milliamps, for the deceleration portion of the move.			0 to 5005, dependant on Drive
Delay	The time, in milliamps, between the last step of a move and when the current is set to the hold current.			50 to 300
Step Mode	Defines the step size, where 1 is a full step, 2 is a half step, and so on.			1,2,4,8,16,32,64.
Example	You wish to stop the motor, in 1/8th step mode, decelerating at a rate of 100000 1/8th steps per second per second to a end speed of 2000 1/8th steps per second, with a run current of 1.6 Arms, decel current of 2.0 Arms, and waiting .05 seconds between the last step and changing to a hold current of 0.5 Arms.			
Command	"H2000,100000,1600,2000,500,50,8" followed by a carriage return			

Command	Symbol	Context	Arguments	Response
Wait For Move	F	Program	none	None

Description	This command causes the program to delay execution of the next command until the motor has stopped moving.	
Arguments	**Argument Description**	**Valid Values or Range**
none		
Example	You have started a move command and do not want the next command to execute until the move has finished.	
Command	"F" followed by a carriage return.	

Command	**Symbol**	**Context**	**Arguments**	**Response**
Wait Time	W	Program	Time	None
Description	This command causes the program to delay execution of the next command for a specified time.			
Arguments	**Argument Description**			**Valid Values or Range**
Time	The amount of time, in milliseconds, that the command should be delayed.			0 to 65535
Example	You have started a move command and do not want the next command to execute for 1 second.			
Command	"W1000" followed by a carriage return.			