
JOG

Jog Mode Enable

Type	Jog	Product	Rev
Syntax	<!><@><a>JOG	6K	5.0
Units	n/a		
Range	b = 0 (disable), 1 (enable), or X (don't change)		
Default	0		
Response	JOG: *JOG0000_0000 1JOG: *1JOG0		
See Also	DJOG, JOGA, JOGAA, JOGAD, JOGADA, JOGVH, JOGVL, INFNC, LIMFNC		

This command enables jog mode on the appropriate axis. Once jog mode has been enabled, the jog inputs can be used to produce motion on the specific axis. The inputs that will be used as jog inputs are determined by the `INFNC` or `LIMFNC` command. Once the jog inputs have been enabled, they will remain enabled, and able to jog at any time while the motor is *in position*. Or in other words, as long as the motor is not moving the jog inputs will be active.

After processing the `JOG1` command, command processing does not stop and wait for the jog mode to be disabled (`JOG0`). Instead, the jog inputs are enabled and command processing continues with the first command after the `JOG1` command.

<p style="text-align: center;">WARNING</p>

<p style="text-align: center;">If a jog input is active when jog mode is enabled, motion will occur.</p>
--

To disable jog mode, issue the `JOG0` command (to the appropriate axis) at any point in the program.

NOTE: If you are using an RP240 operator panel, you can enable the RP240 Jog Mode with the `DJOG1` command and use the RP240's arrow keys to jog individual axes. To disable the RP240 Jog Mode, use the `!DJOG0` command or press the RP240's **MENU RECALL** button.

Example:

```
1INFNC1-L      ; Input #1 on I/O brick 1 defined as jog velocity select input
1INFNC2-1J     ; Input #2 on I/O brick 1 defined as jog positive-direction
               ; input for axis #1
1INFNC3-1K     ; Input #3 on I/O brick 1 defined as jog negative-direction
               ; input for axis #1
1INFNC4-2J     ; Input #4 on I/O brick 1 defined as jog positive-direction
               ; input for axis #2
1INFNC5-2K     ; Input #5 on I/O brick 1 defined as jog negative-direction
               ; input for axis #2
JOGA100,100    ; Jog acceleration set to 100 units/sec/sec on both axes
JOGAD200,200   ; Jog deceleration set to 200 units/sec/sec on both axes
JOGVH10,8     ; The velocity when the jog velocity select input is high is
               ; 10 units/sec on axis #1 and 8 units/sec on axis 2
JOGVL1,.8     ; The velocity when the jog velocity select input is low is
               ; 1 units/sec on axis #1 and 0.8 units/sec on axis 2
JOG1100       ; Enable jog mode on axes 1 and 2. When an input occurs on
               ; input 2, input 3, input 4, or input 5, the motor will move at
               ; the appropriate jog velocity until the input is released
WAIT(1IN.6=b1) ; Wait for input #6 on I/O brick 1 to become active.
               ; Input #6 is being used as a signal to disable jog mode.
JOG0000       ; Disable jog mode on all axes
```

JOGA Jog Acceleration

Type	Jog	Product	Rev
Syntax	<!><@><a>JOGA<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = units/sec/sec		
Range	0.00001 - 39,999,998 (depending on the scaling factor)		
Default	10.0000		
Response	JOGA: *JOGA10.0000,10.0000,10.0000,10.0000 ... 1JOGA: *1JOGA10.0000		
See Also	DJOG, JOG, JOGAA, JOGAD, JOGADA, JOGVH, JOGVL, INFNC, LIMFNC, SCALE, SCLA		

The Jog Acceleration (JOGA) command specifies the acceleration to be used upon receiving a jog input.

UNITS OF MEASURE and SCALING: refer to page 16.

The jog acceleration remains set until you change it with a subsequent jog acceleration command. Accelerations outside the valid range are flagged as an error, with a message *INVALID DATA-FIELD x, where x is the field number. When an invalid acceleration is entered the previous acceleration value is retained.

If the jog deceleration (JOGAD) command has not been entered, the jog acceleration (JOGA) command will also set the jog deceleration rate. Once the jog deceleration (JOGAD) command has been entered, the jog acceleration (JOGA) command no longer affects jog deceleration.

Example: Refer to the jog mode enable (JOG) command example.

JOGAA Jogging Average Acceleration

Type	Jog; Motion (S-Curve)	Product	Rev
Syntax	<!><@><a>JOGAA<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = units/sec/sec		
Range	0.00001 - 39,999,998 (depending on the scaling factor)		
Default	10.00 (trapezoidal profiling is default, where JOGAA tracks JOGA)		
Response	JOGAA: *JOGAA10.0000,10.0000,10.0000,10.0000 ... 1JOGAA: *1JOGAA10.0000		
See Also	A, ADA, JOG, JOGA, JOGAD, JOGADA, SCALE, SCLA		

The Jogging Average Acceleration (JOGAA) command allows you to specify the average acceleration for an S-curve jogging profile. S-curve profiling provides smoother motion control by reducing the rate of change in acceleration and deceleration; this accel/decel rate of change is known as *jerk*. Refer to page 13 for details on S-curve profiling.

Scaling (SCLA) affects JOGAA the same as it does for JOGA. Refer to page 16 for details on scaling.

Example:

JOGA10,10,10,10 ; Sets the maximum jogging acceleration of all axes
JOGAA5,5,7.5,10 ; Sets the average jogging acceleration of all axes

JOGAD Jog Deceleration

Type	Jog	Product	Rev
Syntax	<!><@><a>JOGAD<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = units/sec/sec		
Range	0.00001 - 39,999,998 (depending on the scaling factor)		
Default	10.0000 (JOGAD tracks JOGA)		
Response	JOGAD: *JOGAD10.0000,10.0000,10.0000,10.0000 ... 1JOGAD: *1JOGAD10.0000		
See Also	DJOG, JOG, JOGA, JOGAA, JOGADA, JOGVH, JOGVL, INFNC, LIMFNC, SCALE, SCLA		

The Jog Deceleration (JOGAD) command specifies the deceleration to be used when a jog input is released.

UNITS OF MEASURE and SCALING: refer to page 16.

The jog deceleration remains set until you change it with a subsequent jog deceleration command. Decelerations outside the valid range are flagged as an error, with a message *INVALID DATA-FIELD x, where x is the field number. When an invalid deceleration is entered the previous deceleration value is retained.

If the jog deceleration (JOGAD) command has not been entered, the jog acceleration (JOGA) command will also set the jog deceleration rate. Once the jog deceleration (JOGAD) command has been entered, the jog acceleration (JOGA) command no longer affects jog deceleration. If JOGAD is set to zero (JOGADØ), then the jog deceleration will once again track whatever the JOGA command is set to.

Example: Refer to the jog mode enable (JOG) command example.

JOGADA Jogging Average Deceleration		Product	Rev
Type	Jog; Motion (S-Curve)		
Syntax	<!><@><a>JOGADA<r>,<r>,<r>,<r>,<r>,<r>,<r>,<r>	6K	5.0
Units	r = units/sec/sec		
Range	0.00001 - 39,999,998 (depending on the scaling factor)		
Default	10.00 (JOGADA tracks JOGAA)		
Response	JOGADA: *JOGADA10.0000,10.0000,10.0000,10.0000 ... 1JOGADA: *1JOGADA10.0000		
See Also	A, AD, JOG, JOGA, JOGAA, JOGAD, SCALE, SCLA		

The Jogging Average Deceleration (JOGADA) command allows you to specify the average deceleration for an S-curve jogging profile. S-curve profiling provides smoother motion control by reducing the rate of change in acceleration and deceleration; this accel/decel rate of change is known as *jerk*. Refer to page 13 for details on S-curve profiling.

Scaling (SCLA) affects JOGADA the same as it does for JOGAD. Refer to page 16 for details on scaling.

Example:

```
JOGAD10,10,10,10 ; Sets the maximum jog deceleration of all four axes
JOGADA5,5,7.5,10 ; Sets the average jog deceleration of all four axes
```

JOGVH Jog Velocity High		Product	Rev
Type	Jog		
Syntax	<!><@><a>JOGVH<r>,<r>,<r>,<r>,<r>,<r>,<r>,<r>	6K	5.0
Units	r = units/sec (scalable with SCLV)		
Range	Stepper Axes: 0.00000-2,048,000 (max. depends on SCLV & PULSE) Servo Axes: 0.00000-6,500,000 (max. depends on SCLV)		
Default	10.0000		
Response	JOGVH: *JOGVH10.0000,10.0000,10.0000,10.0000 ... 1JOGVH: *1JOGVH10.0000		
See Also	DJOG, JOG, JOGA, JOGAA, JOGAD, JOGADA, JOGVL, INFNC, LIMFNC, PULSE, SCALE, SCLV		

The Jog Velocity High (JOGVH) command specifies the velocity to be used upon receiving a jog input with the jog velocity select input active (ON).

The jog high velocity remains set until you change it with a subsequent jog high velocity command. Velocities outside the valid range are flagged as an error, with a message *INVALID DATA-FIELD x, where x is the field number. When an invalid velocity is entered the previous velocity value is retained.

UNITS OF MEASURE and SCALING: refer to page 16.

Example: Refer to the jog mode enable (JOG) command example.

JOGVL Jog Velocity Low

Type	Jog	Product	Rev
Syntax	<!><@><a>JOGVL<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = units/sec (scalable with SCLV)		
Range	Stepper Axes: 0.00000-2,048,000 (max. depends on SCLV & PULSE) Servo Axes: 0.00000-6,500,000 (max. depends on SCLV)		
Default	0.5000		
Response	JOGVL: *JOGVL0.50000,0.50000,0.50000,0.50000 ... 1JOGVL: *1JOGVL0.50000		
See Also	DJOG, JOG, JOGA, JOGAA, JOGAD, JOGADA, JOGVH, INFNC, LIMFNC, PULSE, SCALE, SCLV		

The Jog Velocity Low (JOGVL) command specifies the velocity to be used upon receiving a jog input with the jog velocity select input low, or *OFF*. The velocity remains set until you change it with a subsequent jog velocity low command. Velocities outside the valid range are flagged as an error, with a message *INVALID DATA-FIELD x, where x is the field number. When an invalid velocity is entered the previous velocity value is retained.

UNITS OF MEASURE and SCALING: refer to page 16.

Example: Refer to the jog mode enable (JOG) command example.

JOY Joystick Mode Enable

Type	Joystick	Product	Rev
Syntax	<!><@><a>JOY	6K	5.0
Units	n/a		
Range	b = 0 (disable), 1 (enable), or X (don't change)		
Default	0		
Response	JOY: *JOY0000_0000 1JOY: *1JOY0		
See Also	ANIRNG, [AS], COMEXC, INFNC, JOYA, JOYAA, JOYAD, JOYADA, JOYAXH, JOYAXL, JOYCDB, JOYCTR, JOYEDB, JOYVH, JOYVL, JOYZ, LIMFNC, TAS, TIN		

The 6K controller supports joystick operation with digital inputs and analog inputs. The digital inputs include the onboard limit inputs and trigger inputs, as well as digital input SIMs on an external I/O brick. The 12-bit analog inputs are available only if you install an analog input SIM on an external I/O brick (default voltage range is -10V to +10V, selectable with ANIRNG).

To Set Up Joystick Operation (refer also to the example code below):

1. Select the required digital inputs and analog inputs required for joystick operation. Connect the joystick as instructed in your controller's *Installation Guide*.
2. Assign the appropriate input functions to the digital inputs used for joystick's operation:
 - Release Input: INFNCi-M for triggers & external inputs, or LIMFNCi-M for limit inputs.
 - Axis Select Input: INFNCi-N for triggers & external inputs, or LIMFNCi-N for limit inputs. NOTE: If you're not using this input, assign the analog inputs to the axes with the JOYAXH command.
 - Velocity Select Input: INFNCi-O for triggers & external inputs, or LIMFNCi-O for limit inputs.
3. (optional) Use the ANIRNG command to select the voltage range for the analog inputs you will use. The default range is -10VDC to +10VDC (other options are 0 to +5V, -5 to +5V, and 0 to +10V).
4. Assign analog inputs to control specific axes, using:
 - JOYAXH: Standard analog input-to-axis assignment.
 - JOYAXL (optional). Analog input-to-axis assignment when the Axis Select Input is low.
5. Define the joystick motion parameters:
 - Max. Velocity when Velocity Select input switch is open/high (JOYVH command). If the Velocity Select input is not used, joystick motion always uses the JOYVH velocity.
 - Max. Velocity when Velocity Select input switch is closed/low (JOYVL command).
 - Accel (JOYA command).

- Accel for s-curve profiling (JOYAA command).
 - Decel (JOYAD command).
 - Decel for s-curve profiling (JOYADA command).
6. Define the usable voltage zone for your joystick:
(make sure you have first assigned the analog inputs – see step 3 above)
- End Deadband (JOYEDB): Defines the voltage offset (from the -10V & +10V endpoints) at which max. velocity occurs. Default is 0.1V, maxing voltage at -9.9V and +9.9V.
 - Center Voltage (JOYCTR or JOYZ): Defines the voltage when the joystick is at rest to be the zero-velocity center. Default JOYCTR setting is 0V.
 - Center Deadband (JOYCDB): Defines the zero-velocity range on either side of the Center Voltage. Default is 0.1V, setting the zero-velocity range at -0.1V to +0.1V.
7. To jog the axes:
- a. In your program, enable Joystick Operation with the JOY command (Joystick Release input must be closed in order to enable joystick mode). When the JOY command enables joystick mode for the affect axes, program execution stops on those axes (assuming the Continuous Command Execution Mode is disabled with the COMEXCØ command).
 - b. Move the load with the joystick.
 - c. When you are finished, open the Joystick Release input to disable joystick mode. This allows program execution to resume with the next statement after the initial JOY command that started the joystick mode.

Programming Example (refer also to the illustration below):

Application Requirements:

This example represents a typical two-axis joystick application in which a high-velocity range is required to move to a region, then a low-velocity range is required for a fine search. After the search is completed it is necessary to record the load positions, then move to the next region. A digital input can be used to indicate that the position should be read. The Joystick Release input is used to exit the joystick mode and continue with the motion program.

Hardware Configuration:

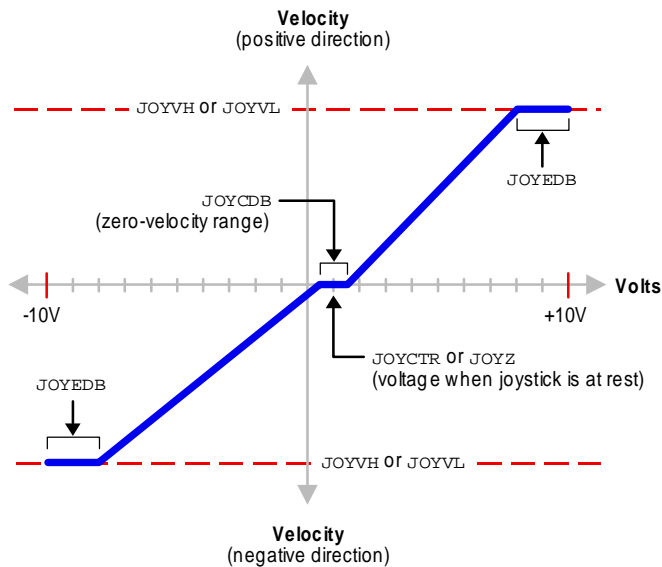
- An analog input SIM is installed in the 3rd slot of I/O brick 1. The eight analog inputs (1-8) are addressed as input numbers 17-24 on the I/O brick. Analog input 17 will control axis 1, and analog input 18 will control axis 2.
- A digital input SIM is installed in the 1st slot of I/O brick 1. The eight digital inputs (1-8) are addressed as input numbers 1-8 on the I/O brick. Digital input 6 will be used for the Joystick Release function, and input 7 will be used for the Joystick Velocity Select input. Input 8 will be used to indicate that the position should be read.

Setup Code (the drawing below shows the usable voltage configuration):

```

1INFNC7-M           ; Assign Joystick Release f(n) to brick 1, input 7
1INFNC8-O           ; Assign Joystick Velocity Select f(n) to brick 1, input 8
JOYAXH1-17,1-18    ; Assign analog input 17 to control axis 1,
                   ; Assign analog input 18 to control axis 2
JOYVH1,1           ; Max. velocity on axes 1 & 2 is 10 units/sec when the
                   ; Velocity Select input switch (1IN.7) is open (high)
JOYVL10,10         ; Max. velocity on axes 1 & 2 is 1 unit/sec when the
                   ; Velocity Select input switch (1IN.7) is closed (low)
JOYA100,100        ; Set joystick accel to 100 units/sec/sec on both axes
JOYAD100,100        ; Set joystick decel to 100 units/sec/sec on both axes
;*** COMMANDS TO SET UP USABLE VOLTAGE: *****
1JOYCTR.17=+1.0    ; Set center voltage for analog input 17 (controls axis 1)
1JOYCTR.18=+1.0    ; and 18 (controls axis 2) to+1.0V. The +1.0V value was
                   ; ascertained by checking the voltage of the both
                   ; inputs (with the 1TANL.17 and 1TAIN.18 commands)
                   ; when the joystick was at rest.
1JOYCDB.17=0.5     ; Set center deadband to compensate for the fact that
1JOYCDB.18=0.5     ; when the joystick is at rest, the voltage received on
                   ; both analog inputs may fluctuate +/- 0.5V on either
                   ; side of the +1.0V center.
1JOYEDB.17=2.0     ; Set end deadband to compensate for the fact that the
1JOYEDB.18=2.0     ; joystick can produce only -8.0V to +8.0V.
;*****
JOY11              ; Enable joystick mode for axes 1 & 2

```



JOYA		Joystick Acceleration	
Type	Joystick	Product	Rev
Syntax	<!><@><a>JOYA<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = units/sec/sec		
Range	0.00120-39,999,998 (depending on the scaling factor)		
Default	10.0000		
Response	JOYA: *JOYA10.0000,10.0000,10.0000,10.0000 ... 1JOYA: *1JOYA10.0000		
See Also	JOY, JOYAA, JOYAD, JOYADA, JOYAXH, JOYAXL, JOYCDB, JOYCTR, JOYEDB, JOYVH, JOYVL, JOYZ, SCALE, SCLA		

The Joystick Acceleration (JOYA) command specifies the acceleration to be used during joystick mode.

UNITS OF MEASURE and SCALING: refer to page 16.

The joystick acceleration remains set until you change it with a subsequent joystick acceleration command. Accelerations outside the valid range are flagged as an error, with a message *INVALID DATA-FIELD x, where x is the field number. When an invalid acceleration is entered the previous acceleration value is retained.

If the joystick deceleration (JOYAD) command has not been entered, the joystick acceleration (JOYA) command will also set the joystick deceleration rate. Once the joystick deceleration (JOYAD) command has been entered, the joystick acceleration (JOYA) command no longer affects joystick deceleration.

Example: Refer to the joystick mode enable (JOY) command example.

JOYAA		Joystick Average Acceleration	
Type	Motion (S-Curve)	Product	Rev
Syntax	<!><@><a>JOYAA<r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = units/sec/sec		
Range	0.00120-39,999,998 (depending on the scaling factor)		
Default	10.00 (trapezoidal profiling is default, where JOYAA tracks JOYA)		
Response	JOYAA: *JOYAA10.0000,10.0000,10.0000,10.0000 ... 1JOYAA: *1JOYAA10.0000		
See Also	AA, AD, JOY, JOYA, JOYAD, JOYADA, SCALE, SCLA		

The Joystick Average Acceleration (JOYAA) command allows you to specify the average acceleration for an S-curve profile. S-curve profiling provides smoother motion control by reducing the rate of change

in acceleration and deceleration; this accel/decel rate of change is known as *jerk*. Refer to page 13 for details on S-curve profiling.

Accelerating Scaling (SCLA) affects JOYAA the same as it does for JOYA. Refer to page 16 for details on scaling.

Example:

```
JOYA10,10,10,10 ; Set the maximum joystick acceleration of all four axes
JOYAA5,5,7.5,10 ; Set the average joystick acceleration of all four axes
```

JOYAD Joystick Deceleration

Type	Joystick	Product	Rev
Syntax	<!><@><a>JOYAD<r>,<r>,<r>,<r>,<r>,<r>,<r>,<r>	6K	5.0
Units	r = units/sec/sec		
Range	0.00120-39,999,998 (depending on the scaling factor)		
Default	10.0000 (JOYAD tracks JOYA)		
Response	JOYAD: *JOYAD10.0000,10.0000,10.0000,10.0000 ... 1JOYAD: *1JOYAD10.0000		
See Also	JOY, JOYA, JOYAA, JOYADA, JOYAXH, JOYAXL, JOYCDB, JOYCTR, JOYEDB, JOYVH, JOYVL, JOYZ, SCALE, SCLA		

The Joystick Deceleration (JOYAD) command specifies the deceleration to be used during the joystick mode.

UNITS OF MEASURE and SCALING: refer to page 16.

The joystick deceleration remains set until you change it with a subsequent joystick deceleration command. Decelerations outside the valid range are flagged as an error, with a message *INVALID DATA-FIELD x, where x is the field number. When an invalid deceleration is entered the previous deceleration value is retained.

If the joystick deceleration (JOYAD) command has not been entered, the joystick acceleration (JOYA) command will also set the joystick deceleration rate. Once the joystick deceleration (JOYAD) command has been entered, the joystick acceleration (JOYA) command no longer affects joystick deceleration. If JOYAD is set to zero (JOYADØ), then the joystick deceleration will once again track whatever the JOYA command is set to.

Example: Refer to the joystick mode enable (JOY) command example.

JOYADA Joystick Average Deceleration

Type	Motion (S-Curve)	Product	Rev
Syntax	<!><@><a>JOYADA<r>,<r>,<r>,<r>,<r>,<r>,<r>,<r>	6K	5.0
Units	r = units/sec/sec		
Range	0.00120-39,999,998 (depending on the scaling factor)		
Default	10.00 (JOYADA tracks JOYAA)		
Response	JOYADA: *JOYADA10.0000,10.0000,10.0000,10.0000 ... 1JOYADA: *1JOYADA10.0000		
See Also	A, AD, JOY, JOYA, JOYAA, JOYAD, SCALE, SCLA		

The Joystick Average Deceleration (JOYADA) command allows you to specify the average deceleration for an S-curve joystick profile. S-curve profiling provides smoother motion control by reducing the rate of change in acceleration and deceleration; this accel/decel rate of change is known as *jerk*. Refer to page 13 for details on S-curve profiling.

Acceleration Scaling (SCLA) affects JOYADA the same as it does for JOYAD. Refer to page 16 for details on scaling.

Example:

```
JOYAD10,10,10,10 ; Sets the maximum joystick deceleration of all four axes
JOYADA5,5,7.5,10 ; Sets the average joystick deceleration of all four axes
```

JOYAXH Joystick Analog Channel High

Type	Joystick	Product	Rev
Syntax	<!><@><a>JOYAXH<B-i>, <B-i>, <B-i>, <B-i>, <B-i>, <B-i>, <B-i>, <B-i>	6K	5.0
Units	B = I/O brick number i = Location of the analog input on I/O brick (see page 6)		
Range	B = 1-8 i = 1-32		
Default	0-0,0-0,0-0,0-0,0-0,0-0,0-0,0-0		
Response	JOYAXH: *JOYAXH1-1,1-2,1-3,1-4,1-5,1-6,1-7,1-8 1JOYAXH: *1JOYAXH1-1		
See Also	ANIRNG, [IN], INFNC, JOY, JOYA, JOYAA, JOYAD, JOYADA, JOYAXL, JOYCDB, JOYCTR, JOYEDB, JOYVH, JOYVL, JOYZ, [LIM], LIMFNC, TIN, TLIM		

The Joystick Analog Channel High (JOYAXH) command specifies the analog input that will control each axis while the Joystick Axis Select input (INFNCi-N or LIMFNCi-N) is open and the corresponding axis is in Joystick Mode. A single analog input can control more than one axis (e.g., JOYAXH1-1,1-1 assigns the analog input at location 1 on I/O brick 1 to control axes 1 and 2). If the Joystick Axis Select input is not used, the JOYAXH command determines which axes are controlled by which analog inputs.

To understand how specific I/O points are addressed on the I/O bricks, refer to page 6.

NOTE: The 12-bit analog inputs are available only if you install an analog input SIM on an external I/O brick. Use the ANIRNG command to select the voltage range for the analog inputs you will use. The default range is -10VDC to +10VDC (other options are 0 to +5V, -5 to +5V, and 0 to +10V).

Example: Refer to the joystick mode enable (JOY) command example.

JOYAXL Joystick Analog Channel Low

Type	Joystick	Product	Rev
Syntax	<!><@><a>JOYAXL<B-i>, <B-i>, <B-i>, <B-i>, <B-i>, <B-i>, <B-i>, <B-i>	6K	5.0
Units	B = I/O brick number i = Location of the analog input on I/O brick (see page 6)		
Range	B = 1-8 i = 1-32		
Default	0-0,0-0,0-0,0-0,0-0,0-0,0-0,0-0		
Response	JOYAXL: *JOYAXL1-1,1-2,1-3,1-4,1-5,1-6,1-7,1-8 1JOYAXL: *1JOYAXL1-1		
See Also	ANIRNG, [IN], INFNC, JOY, JOYA, JOYAA, JOYAD, JOYADA, JOYAXH, JOYCDB, JOYCTR, JOYEDB, JOYVH, JOYVL, JOYZ, [LIM], LIMFNC, TIN, TLIM		

The Joystick Analog Channel Low (JOYAXL) command specifies the analog input that will control each axis while the Joystick Axis Select input (INFNCi-N or LIMFNCi-N) is closed and the corresponding axis is in Joystick Mode. A single analog input can control more than one axis (e.g., JOYAXL1-1,1-1 assigns the analog input at location 1 on I/O brick 1 to control axes 1 and 2). If the Joystick Axis Select input is not used, the JOYAXL command has no effect; instead, the JOYAXH command determines which axes are controlled by which analog inputs.

To understand how to address specific I/O points on the I/O bricks, refer to page 6.

NOTE: The 12-bit analog inputs are available only if you install an analog input SIM on an external I/O brick. Use the ANIRNG command to select the voltage range for the analog inputs you will use. The default range is -10VDC to +10VDC (other options are 0 to +5V, -5 to +5V, and 0 to +10V).

Example: Refer to the joystick mode enable (JOY) command example.

JOYCDB Joystick Center Deadband

Type	Joystick	Product	Rev
Syntax	<!><@><a>JOYCDB<.i><=r>	6K	5.0
Units	i = I/O location for the analog input on brick B (see page 6) r = volts		
Range	i = 1-32 r = -10.00 - +10.00 (depending on ANIRNG setting)		
Default	0.1		
Response	1JOYCDB.1 *1JOYCDB.1=0.1		
See Also	ANIRNG, INFNC, JOY, JOYA, JOYAA, JOYAD, JOYADA, JOYAXH, JOYAXL, JOYCTR, JOYEDB, JOYVH, JOYVL, JOYZ, LIMFNC		

The JOYCDB command defines, for the specified analog input(s), the zero-velocity range on either side of the Center Voltage established with the JOYCTR command or the JOYZ command. The default setting is 0.1V, which sets the zero-velocity range at -0.1V to +0.1V (assuming the default JOYCTR default of 0.0V is used). **NOTE:** Executing the JOYCDB command before the JOYAXH command will cause an error (“INPUT(S) NOT DEFINED AS JOYSTICK INPUT”).

Example: Refer to the joystick mode enable (JOY) command example.

JOYCTR Joystick Center

Type	Joystick	Product	Rev
Syntax	<!><@><a>JOYCTR<.i><=r>	6K	5.0
Units	i = I/O location for the analog input on brick B (see page 6) r = volts		
Range	i = 1-32 r = -10.00 - +10.00 (depending on ANIRNG setting)		
Default	0.00		
Response	1JOYCTR.1 *1JOYCTR.1=0.00		
See Also	ANIRNG, INFNC, JOY, JOYA, JOYAA, JOYAD, JOYADA, JOYAXH, JOYAXL, JOYCDB, JOYEDB, JOYVH, JOYVL, JOYZ, LIMFNC		

The JOYCTR command defines, for the specified analog input(s), the voltage to be considered as the zero-velocity center (usually associated with leaving the joystick in the resting position). Default is 0V. The zero-velocity range about the center is determined by the JOYCDB command. As an alternative to the JOYCTR command, you could use the JOYZ command. **NOTE:** Executing the JOYCTR command before the JOYAXH command will cause an error (“INPUT(S) NOT DEFINED AS JOYSTICK INPUT”).

Example: Refer to the joystick mode enable (JOY) command example.

JOYEDB Joystick End Deadband

Type	Joystick	Product	Rev
Syntax	<!><@><a>JOYEDB<.i><=r>	6K	5.0
Units	i = I/O location for the analog input on brick B (see page 6) r = volts		
Range	i = 1-32 r = -10.00 - +10.00 (depending on ANIRNG setting)		
Default	0.1		
Response	1JOYEDB.1 *1JOYCTR.1=0.1		
See Also	ANIRNG, INFNC, JOY, JOYA, JOYAA, JOYAD, JOYADA, JOYAXH, JOYAXL, JOYCDB, JOYCTR, JOYVH, JOYVL, JOYZ, LIMFNC		

The JOYEDB command defines, for the specified analog input(s), the voltage offset (from the -10V & +10V endpoints) at which maximum velocity occurs. This command is useful if your joystick does not reach either limit of the voltage range (-10.00V to +10.00V). The default setting is 0.1V, creating a maximum voltage range of -9.9V to +9.9V. **NOTE:** Executing the JOYEDB command before the JOYAXH command will cause an error (“INPUT(S) NOT DEFINED AS JOYSTICK INPUT”).

Example: Refer to the joystick mode enable (JOY) command example.

JOYVH Joystick Velocity — Velocity Select Input High

Type	Joystick	Product	Rev
Syntax	<!><@><a>JOYVH<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = units/sec		
Range	Stepper Axes: 0.00000-2,048,000 (max. depends on SCLV & PULSE) Servo Axes: 0.00000-6,500,000 (max. depends on SCLV)		
Default	0.5000		
Response	JOYVH: *JOYVH0.5000,0.5000,0.5000,0.5000 ... 1JOYVH: *1JOYVH0.5000		
See Also	[IN], INFNC, JOY, JOYA, JOYAA, JOYAD, JOYADA, JOYAXH, JOYAXL, JOYCDB, JOYCTR, JOYEDB, JOYVL, JOYZ, [LIM], LIMFNC, PULSE. SCALE, SCLV, TIN, TLIM		

The Joystick Velocity High (JOYVH) command specifies the maximum velocity that can be obtained at full deflection during joystick mode, with the Joystick Velocity Select input open (high). The Joystick Velocity Select input function is defined with the INFNCi-O command or the LIMFNCi-O command. If the Velocity Select input is not used, joystick motion always uses the JOYVH velocity.

NOTE: The data fields (<r>, <r>, <r>, <r>...) represent the axes, **not the analog inputs**.

The joystick velocity must be entered prior to entering joystick mode (JOY). The joystick velocity high remains set until you change it with a subsequent JOYVH command. Velocities outside the valid range are flagged as an error, with a message *INVALID DATA-FIELD x, where x is the field number. When an invalid velocity is entered the previous velocity value is retained.

UNITS OF MEASURE and SCALING: refer to page 16.
--

Example: Refer to the joystick mode enable (JOY) command example.

JOYVL Joystick Velocity — Velocity Select Input Low

Type	Joystick	Product	Rev
Syntax	<!><@><a>JOYVL<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = units/sec		
Range	Stepper Axes: 0.00000-2,048,000 (max. depends on SCLV & PULSE) Servo Axes: 0.00000-6,500,000 (max. depends on SCLV)		
Default	0.2000		
Response	JOYVL: *JOYVL0.2000,0.2000,0.2000,0.2000 ... 1JOYVL: *1JOYVL0.2000		
See Also	[IN], INFNC, JOY, JOYA, JOYAA, JOYAD, JOYADA, JOYAXH, JOYAXL, JOYCDB, JOYCTR, JOYEDB, JOYVH, JOYZ, [LIM], LIMFNC, PULSE. SCALE, SCLV, TIN, TLIM		

The Joystick Velocity Low (JOYVL) command specifies the maximum velocity that can be obtained at full deflection during joystick mode, with the Joystick Velocity Select input closed (low). The Joystick Velocity Select input function is defined with the INFNCi-O command or the LIMFNCi-O command. If the Velocity Select input is not used, joystick motion always uses the JOYVH velocity.

NOTE: The data fields (<r>, <r>, <r>, <r>...) represent the axes, **not the analog channels**.

The joystick velocity must be entered prior to entering joystick mode (JOY). The joystick velocity low remains set until you change it with a subsequent JOYVL command. Velocities outside the valid range are flagged as an error, with a message *INVALID DATA-FIELD x, where x is the field number. When an invalid velocity is entered the previous velocity value is retained.

UNITS OF MEASURE and SCALING: refer to page 16.
--

Example: Refer to the joystick mode enable (JOY) command example.

JOYZ

Joystick Zero

Type	Joystick	Product	Rev
Syntax	<!><@>JOYZ<.i><=b> (multiple inputs per brick may be configured at one time)	6K	5.0
Units	B = I/O brick number i = Location of the analog input on I/O brick (see page 6) b = enable bit		
Range	B = 1-8 i = 1-32 b = 0 (don't zero), 1 (zero), or X (don't change)		
Default	n/a		
Response	n/a		
See Also	ANIRNG, JOY, JOYA, JOYAA, JOYAD, JOYADA, JOYAXH, JOYAXL, JOYCDB, JOYCTR, JOYEDB, JOYVH, JOYVL		

The Joystick Zero (JOYZ) command defines the voltage when the joystick is at rest to be the zero-velocity center. Simply leave the joystick in its resting position and issue a JOYZ command to define the current voltage of the respective analog inputs as the zero-velocity center. The zero-velocity range about the center is determined by the JOYCDB command.

The JOYZ command is an alternative to using the JOYCTR command.

Example:

```
1INFNC7-M           ; Assign Joystick Release f(n) to brick 1, input 7
1INFNC8-0           ; Assign Joystick Velocity Select f(n) to brick 1, input 8
JOYAXH1-17,1-18    ; Assign analog input 17 to control axis 1,
                   ; Assign analog input 18 to control axis 2
JOYVH10,10         ; Max. velocity on axes 1 & 2 is 10 units/sec when the
                   ; Velocity Select input (1IN.7) is high (sinking current)
JOYVL1,1           ; Max. velocity on axes 1 & 2 is 1 unit/sec when the
                   ; Velocity Select input (1IN.7) is low (not sinking current)
JOYA100,100        ; Set joystick accel to 100 units/sec/sec on both axes
JOYAD100,100        ; Set joystick decel to 100 units/sec/sec on both axes
;**** COMMANDS TO SET UP USABLE VOLTAGE: ****
1JOYZ.17=1         ; These command are executed while the joystick is at rest. They
1JOYZ.18=1         ; set the current voltage on analog input 17 (controls axis 1)
                   ; and input 18 (controls axis 2) as the zero-velocity center.
1JOYCDB.17=0.5     ; Set center deadband to compensate for the fact that
1JOYCDB.18=0.5     ; when the joystick is at rest, the voltage received on
                   ; both analog inputs may fluctuate +/- 0.5V on either
                   ; side of the zero-velocity center established with JOYZ.
1JOYEDB.17=2.0     ; Set end deadband to compensate for the fact that the
1JOYEDB.18=2.0     ; joystick can produce only -8.0V to +8.0V.
;*****
JOY11              ; Enable joystick mode for axes 1 & 2
```

JUMP

Jump to a Program or Label (and do not return)

Type	Program or Subroutine Definition or Program Flow Control	Product	Rev
Syntax	<!>JUMP<t>	6K	5.0
Units	t = text (name of program/label)		
Range	Text name of 6 characters or less		
Default	n/a		
Response	n/a		
See Also	\$, DEF, DEL, END, GOSUB, GOTO, IF, L, LN, NIF, NWHILE, REPEAT, RUN, UNTIL, WHILE		

The JUMP command branches to the corresponding program name or label when executed. A program or label name consists of 6 or fewer alpha-numeric characters.

All nested IFs, WHILEs, and REPEATs, loops, and subroutines are cleared; thus, the program or label that the JUMP initiates will **not** return control to the line after the JUMP, when the program completes operation. Instead, the program will end.

If an invalid program or label name is entered, the JUMP will be ignored, and processing will continue with the line after the JUMP.

Example

```
; *****
; * In this example, the program place is executed and calls the pick *
; * subroutine. The pick subroutine then initiates motion on axes 1 & 2 *
; * (GO1100) and jumps to the program called load to initiate motion on *
; * axis 3 (GO001). Then, because the JUMP command cleared the pick *
; * subroutine, program execution is terminated instead of returning to *
; * the place program. *
; *****
DEF pick          ; Begin definition of subroutine named pick
GO1100           ; Initiate motion on axes 1 and 2
JUMP load        ; Jump to the program named load
END              ; End subroutine definition
DEF load         ; Begin definition of program named load
GO001           ; Initiate motion on axis 3
END             ; End program definition
DEF place       ; Begin definition of subroutine named place
GOSUB pick      ; Gosub to subroutine named pick
GO1000         ; Initiate motion on axis 1
END            ; End subroutine definition
RUN place      ; Execute program named place
```

K Kill Motion

Type	Motion	Product	Rev
Syntax	<!><@>K	6K	5.0
Units	n/a		
Range	b = 0 (don't kill), 1 (kill), or X (don't change)		
Default	n/a		
Response	!K No response, instead motion is killed on all axes		
See Also	DRFLVL, FOLK, GO, <CTRL>K, KDRIVE, LHAD, LHADA, S, SCANP, TAS		

The Kill Motion (K) command instructs the motor to stop motion on the specified axes. If the Kill (K) command is used without any arguments (K or !K), motion will be stopped on all axes, and program execution will be terminated. When the Kill (K) command is used with ones in the command fields (e.g., KØ11Ø), motion will be stopped on the axes specified with ones (1), and program execution will continue with the next command. The Kill command will be used most frequently with the immediate command delimiter in front of the command (!K). By using the immediate Kill (!K) command, motion will be stopped at the time the command is received.

Motion is stopped at the rate set with the LHADA and LHAD commands. If you want the drive to be disabled upon executing a K or !K command, enable the *Disable Drive on Kill* mode with the KDRIVE command.

CAUTION: In the KDRIVE mode, a K or !K command immediately shuts down the drive, allowing the load to *free wheel* to a stop.

If the axis is involved in a PLC Scan (initiated with SCANP), a K command will clear the scan.

Example:

```
A2,2,25000,25000 ; Set acceleration to 2, 2, 25000, and 25000 units/sec/sec
; for axes 1, 2, 3 and 4
AD2,2,25000,25000 ; Set deceleration to 2, 2, 25000, and 25000 units/sec/sec
; for axes 1, 2, 3 and 4
V1,1,1,2        ; Set velocity to 1, 1, 1, and 2 units/sec for axes 1, 2, 3
; and 4, respectively
@D10           ; Set distance on all axes to 10 units
@GO1          ; Initiate motion on all axes -- motion begins.
; After a short period the Kill command is sent.
!K            ; Kill motion on all axes (steppers stop instantaneously,
; servos stop at the LHADA/LHAD decel)
```

<CTRL>K Kill Motion

Type	Motion	Product	Rev
Syntax	<CTRL>K	6K	5.0
Units	n/a		
Range	n/a		
Default	n/a		
Response	<CTRL>K: No response, instead motion is killed on all axes		
See Also	GO, K, KDRIVE, LHAD, LHADA, S		

The Kill Motion (<ctrl>K) command instructs the controller to stop motion on all axes, and terminate program execution. In essence, the <ctrl>K command is an immediate kill (!K) command.

Motion is stopped at the rate set with the LHADA and LHAD commands. If the *Disable Dive on Kill* mode is enabled with the KDRIVE command, a <ctrl>K command immediately shuts down the drive, allowing the load to *free wheel* to a stop.

Example:

```
A2,2,25000,25000 ; Sets acceleration to 2, 2, 25000, and 25000 units/sec/sec
                  ; for axes 1, 2, 3 and 4
AD2,2,25000,25000 ; Sets deceleration to 2, 2, 25000, and 25000 units/sec/sec
                  ; for axes 1, 2, 3 and 4
V1,1,1,2         ; Sets velocity to 1, 1, 1, and 2 units/sec for axes 1, 2, 3
                  ; & 4, respectively
@D10             ; Set distance on all axes to 10 units
@GO1             ; Initiate motion on all axes -- motion begins.
                  ; After a short period the Kill command is sent.
<CTRL>K         ; Kill motion on all axes (steppers stop instantaneously,
                  ; servos stop at the LHADA/LHAD deceleration)
```

KDRIVE Disable Drive on Kill

Type	Controller Configuration	Product	Rev
Syntax	<!><@><a>KDRIVE	6K	5.0
Units	b = enable bit		
Range	0 (disable), 1 (enable), or X (don't change)	(applicable to servo axes only)	
Default	0		
Response	KDRIVE: *KDRIVE0000_0000 LKDRIVE: *KDRIVE0		
See Also	DRIVE, INFNC, K, <ctrl>K, LIMFNC		

If you enable the Disable Drive on Kill function (KDRIVE1), then when a kill command (K, !K, or <ctrl>K) is processed or a kill input (INFNCi-C or LIMFNCi-C) is activated, the drive will be disabled immediately; this cuts all control to the motor and allows the load to freewheel to a stop (although steppers have some detent torque).

When the drive is disabled (shutdown/de-energized):

- Stepper Axis: Shutdown+ sources current and Shutdown- sinks current.
- Servo Axis: SHTNO relay output is disconnected from COM, and the SHTNC relay output is connected to COM.

To re-enable the drive, issue the DRIVE1 command to the affect axis or axes.

If you leave the KDRIVE command in its default state (∅, disabled), the kill function behaves in its normal manner, leaving the drive enabled.

Example:

```
KDRIVE11        ; Set axes 1 & 2 to de-energize the drive during a kill
K               ; Kill is performed and drives are de-energized
```

L

Loop

Type	Loops; Program Flow Control	Product	Rev
Syntax	<!>L<i>	6K	5.0
Units	i = number of times to loop		
Range	0-999,999,999		
Default	0		
Response	L: No response; instead, this has the same function as L0		
See Also	LN, LX, PLN, PLOOP		

When you combine the Loop (L) command with the end of loop (LN) command, all of the commands between L and LN will be repeated the number of times indicated by n. If <i> = \emptyset , or if no argument is specified, all the commands between L and LN will be repeated indefinitely. The loop can be stopped by issuing a Terminate Loop (!LX) command, an immediate Kill (!K) command, or an immediate Halt (!HALT) command.

The loop can be paused by issuing an immediate Pause (!PS) command or a Stop (!S) command with COMEXS enabled. The loop can then be resumed with the immediate Continue (!C) command. You may nest loops up to 16 levels deep.

NOTE: Be careful about performing a GOTO between the L and LN commands. Branching to a different location within the same program will cause the next loop encountered to be nested within the previous loop, unless an LN command has already been encountered.

Example:

```
L5           ; Repeat the commands between L and LN five times
GO1110      ; Start motion on axes 1, 2, and 3, axis 4 will remain motionless
LN          ; End loop
```

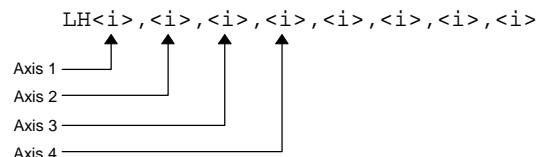
LH

Hardware End-of-Travel Limit — Enable Checking

Type	Limit (End-of-Travel)	Product	Rev
Syntax	<!><@><a>LH<i>,<i>,<i>,<i>,<i>,<i>,<i>,<i>	6K	5.0
Units	n/a		
Range	i = 0 (disable both), 1 (disable positive-direction), 2 (disable negative-direction), or 3 (enable both)		
Default	3		
Response	LH: *LH3,3,3,3,3,3,3,3 lLH: *lLH3		
See Also	[AS], [ER], ERROR, INFNC, INLVL, LHAD, LHADA, [LIM], LIMEN, LIMFNC, LIMLVL, LS, LSAD, LSADA, LSNEG, LSPOS, TAS, TER, TLIM, TSTAT		

Use the LH command to enable or disable the inputs defined as end-of-travel limit inputs. This pertains to onboard limit inputs defined with the LIMFNCi-aR and LIMFNCi-aS commands (this is the factory default configuration for limits), as well as to onboard triggers and external digital inputs defined with the INFNCi-aR and INFNCi-aS commands.

Command Syntax:



With limits disabled, motion will not be restricted. When a specific limit is enabled (positive- or negative-direction), and the limit wiring for the enabled limit is a physical open circuit, motion will be restricted (assuming LHLVL \emptyset or INLVL \emptyset). The LHLVL controls the active level for onboard limit inputs, and the INLVL command controls the active level for onboard triggers and external digital inputs.

Disable negative-direction limit; Disable positive-direction limit:	i = 0
Enable negative-direction limit; Disable positive-direction limit:	i = 1
Disable negative-direction limit; Enable positive-direction limit:	i = 2
Enable negative-direction limit; Enable positive-direction limit:	i = 3

If an “end-of-travel limit” input is redefined with a different function (i.e., not LIMFNCi-R, LIMFNCi-S, INFNCi-R or INFNCi-S), it is no longer controlled by the LH command. If the input is a limit (on the “LIMITS/HOME” connector), use the LIMEN command; if the input is a trigger or external digital input, use the INEN command.

NOTE

If a hard limit is encountered while limits are enabled, motion must occur in the opposite direction after correcting the limit condition (resetting the switch); then you can make a move in the original direction. If limits are disabled, you are free to make a move in either direction.

Example:

```
LH3,3           ; Enable limits on axes 1 and 2
LHAD100,100    ; Set hard limit decel to 100 units/sec/sec on axes 1 and 2
LIMLVL00x00    ; Active low hard limits for axes 1 & 2
A10,12        ; Set acceleration to 10 and 12 units/sec/sec for axes 1 and 2
V1,1          ; Set velocity to 1 unit/sec for axes 1 and 2
D100000,1000  ; Set distance to 100000 and 1000 units for axes 1 and 2
GO11XX        ; Initiate motion on axes 1 and 2
```

LHAD Hard Limit Deceleration

Type	Limit (End-of-Travel)	Product	Rev
Syntax	<!><@><a>LHAD<r>,<r>,<r>,<r>,<r>,<r>,<r>,<r>	6K	5.0
Units	r = units/sec/sec		
Range	0.00001-39,999,998 (depending on the scaling factor)		
Default	100.0000		
Response	LHAD: *LHAD100.0000,100.0000,100.0000,100.0000 ... !LHAD: *!LHAD100.0000		
See Also	DRES, DRFLVL, INFNC, K, LH, LHADA, [LIM], LIMFNC, LIMLVL, LS, LSAD, LSADA, LSNEG, LSPOS, SCALE, SCLA		

The Hard Limit Deceleration (LHAD) command determines the value at which to decelerate after an end-of-travel limit has been hit. This applies to the on-board dedicated limits, as well as to any inputs configured as end-of-travel limits (INFNCi-R or INFNCi-S).

UNITS OF MEASURE and SCALING: refer to page 16.

When a drive fault, a Kill command (K, !K, or ^K), or a Kill input (INFNCi-C or LIMFNCi-C) occurs, motion is stopped at the rate set with the LHAD and LHADA commands. If the *Disable Drive on Kill* mode is enabled (KDRIVE1), the drive is immediately shut down upon a Kill command or input and allows the motor/load to *freewheel* to a stop without a controlled deceleration.

The hard limit deceleration remains set until you change it with a subsequent hard limit deceleration command. Decelerations outside the valid range are flagged as an error, with a message *INVALID DATA-FIELD x, where x is the field number. When an invalid deceleration is entered the previous deceleration value is retained.

Example: Refer to the hard limit enable (LH) command example.

LHADA Hard Limit Average Deceleration

Type	Motion (S-Curve)	Product	Rev
Syntax	<!><@><a>LHADA<r>,<r>,<r>,<r>,<r>,<r>,<r>,<r>	6K	5.0
Units	r = units/sec/sec		
Range	0.00001-39,999,998 (depending on the scaling factor)		
Default	100.000 (default is a constant deceleration ramp, where LHADA tracks LHAD)		
Response	LHADA: *LHADA100.0000,100.000,100.000,100.000 ... !LHADA: *!LHADA100.0000		
See Also	AD, ADA, INFNC, K, LHAD, LIMFNC, LIMLVL, SCALE, SCLA		

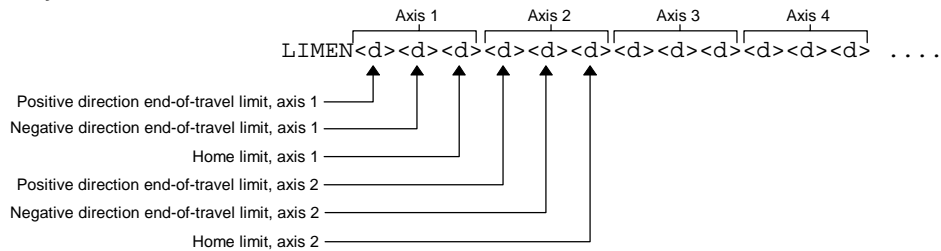
The Hard Limit Average Deceleration (LHADA) command allows you to specify the average deceleration for an S-curve deceleration profile when a limit is hit. S-curve profiling provides smoother motion control by

LIMEN Limit Input Enable

Type	Inputs; Program Debug Tool	Product	Rev
Syntax	<!>LIMEN<d><d><d> (one <d> for each limit input)	6K	5.0
Units	n/a		
Range	d = 0 (disable, leave off), 1 (disable, leave on), E (enable), or X (don't change)		
Default	E		
Response	LIMEN: *LIMENEEE_EEE_EEE_EEE_EEE_EEE_EEE LIMEN.3 *E		
See Also	HOMLVL, INEN, LH, [LIM], LIMFNC, LIMLVL, TIO, TLIM		

The LIMEN command allows you to simulate the activation of specific limit inputs (without actually wiring the inputs to the controller) by disabling them and setting them to a specific level (ON or OFF). This is useful for testing and debugging your program (see program example below). LIMEN may only be used for onboard limit inputs (found on the "LIMITS/HOME" connector), not for triggers or external digital inputs. The default state is enabled (E), requiring external wiring to exercise their respective LIMFNC functions.

Command Syntax:



The LH command is required to enable checking the state of the end-of-travel limits (i.e., LIMFNci-R, or LIMFNci-S); for example, LH1 is required to detect the occurrence of the hardware limit activation, as reported with axis status bits 15 and 16 (see TASF, TAS, AS). The default LH condition is enabled (LH1).

Input bit assignments for the LIMEN command vary by product, because of the number of limit inputs available. The input bit patterns for onboard and external I/O bricks are explained on page 6 of this document.

Example:

```

DEL tsting          ; Delete program called "tsting"
DEF tsting          ; Begin definition of program called "tsting"
LIMFNc10-E         ; Define hardware end-of-travel limit #10 (normally defined
                  ; as the positive direction end-of-travel limit for axis 4)
                  ; as a "pause/resume" input.
COMEXR1           ; Activating the pause/resume input will pause command and
                  ; motion execution, de-activating the pause/resume input
                  ; will resume command and motion execution.
MC111             ; Set axes 1-3 to continuous motion profiling mode
A15,15,15         ; Set acceleration on axes 1-3 to 15 units/sec/sec
AD5,5,5           ; Set deceleration on axes 1-3 to 5 units/sec/sec
V4,4,4            ; Set velocity on axes 1-3 to 4 revs/sec
GO111             ; Initiate continuous motion on axes 1-3
END               ; End definition of program called "tsting"
; While this program is running and motion is in progress, you can send
; immediate LIMEN commands to simulate the function of the "pause/resume"
; input as follows:
; 1. Start the program by sending the RUN TSTING command to the controller.
;    Axes 1-3 will start moving, all using the same continuous motion profile.
; 2. Send the !LIMEN.10=1 command to the controller. This disables the
;    "pause/resume" input but simulates its activation. Motion and
;    program execution will pause.
; 3. Send the !LIMEN.10=0 command to the controller. This disables the
;    "pause/resume" input but simulates its de-activation. Motion and
;    program execution will resume.
; 4. Send the !LIMEN.10=E command to the controller. This re-enables the
;    "pause/resume" input for normal operation with an external switch
;    or sensor.
; 5. To stop this experiment, send the !K command to the controller.
;    This "kills" program execution and motion on all three axes.
  
```

LIMFNC Input Function for Limit Inputs

Type	Inputs; Limits (end of travel); Homing	Product	Rev
Syntax	<!>LIMFNC<i>-<<a>c>	6K	5.0
Units	i = input # on the "LIMITS/HOME" connector (see page 6); a = axis # (or program # for function P); c = function identifier letter		
Range	i = 1-24 (product dependent); a = 1-8 (product dependent); c = A-T		
Default	A		
Response	LIMFNC: *LIMFNC1-A NO FUNCTION - STATUS OFF (repeated for all onboard limit inputs)		
See Also	COMEXR, COMEXS, [ER], ERROR, INDEB, INFNC, INPLC, INSELP, INSTW, INTHW, JOY, JOYAXH, JOYAXL, JOYVH, JOYVL, K, KDRIVE, LH, [LIM], LIMEN, LIMFNC, LIMLVL, PSET, [SS], TER, TIN, TRGFN, TRGLOT, [TRIG], TSS, TSTAT, TRIG		

The Limit Input Function (LIMFNC) command defines the function of each individual limit input found on the "LIMITS/HOME" connector(s). The factory default configuration is that each dedicated hardware end-of-travel and home limit is assigned to its respective LIMFNC function. That is, axis 1 positive limit is assigned to LIMFNC1-1R, axis 1 negative limit is assigned to LIMFNC2-1S, axis 1 home limit is assigned to LIMFNC3-1T, etc. A limit of 32 limit inputs may be assigned LIMFNC functions; this excludes functions A ("general-purpose") and R, S, and T (end-of-travel and home limit input functions).

Input debounce. By default, the limit inputs are not debounced. However, when a limit input is assigned a function other than its respective LIMFNC function, it is debounced with the Input Debounce Time (INDEB) command setting for I/O brick zero (default is 4 ms). The INDEB debounce is the period of time that the input must be held in a certain state before the controller recognizes it. This directly affects the rate at which the inputs can change state and be recognized. If a limit is once again returned to its respective LIMFNC function, the debounce is removed.

Input bit assignments vary by product. The number of limits inputs and axes available depends on your product (each axis has two end-of-travel limits and one home limit) — see page 6 for details.

Input scan rate: The limit inputs are scanned once per *system update* (2 milliseconds).

Enabling & disabling inputs. Limit inputs assigned an end-of-travel input function (functions R or S described below) are enabled/disabled with the LH command — the default is enabled. Limit input functions may be overridden with the LIMEN command — the default is enabled (no override).

Multitasking. If the LIMFNC command does not include the task identifier (%) prefix, the function affects the task that executes the LIMFNC command. The functions that may be directed to a task with % are: C, D (without an axis specified), E, F, and P (e.g., 2%LIMFNC3-F assigns limit input 3 as a user fault input for task 2). Multiple tasks may share the same input, but the input may only be assigned one function.

Identifier Function Description

- A **No special function** (general-purpose input). Status can be used with the LIM assignment/comparison operator.
- B **BCD Program Select.** BCD input assignment to programs, lowest numbered input is least significant bit (LSB). BCD values for inputs are as follows:

	BCD Value
Least Significant Bit Value	1
.	2
.	4
.	8
.	10
.	20
.	40
.	80
Most Significant Bit Value	100

Note: If fewer inputs than shown above are defined to be Program Select Inputs, then the highest input number defined as a Program Select Input is the most significant bit.

An input defined as a BCD Program Select Input will not function until the INSELP command has been enabled.

Identifier	Function Description
C	Kill. Kills motion on all axes and halts all command processing (refer to <code>K</code> and <code>KDRIVE</code> command descriptions for further details on the <i>kill</i> function). This is an edge detection function and is not intended to inhibit motion. To inhibit motion, use the Pause/Resume function (<code>LIMFNCi-E</code>). When enabled with the <code>ERROR</code> command, bit #6 of the <code>TER</code> and <code>ER</code> commands will report the kill status.
<a>D	Stop. Stops motion. Axis number is optional; if no axis number is specified, motion is stopped on all axes. If <code>COMEXS</code> is set to zero (<code>COMEXS0</code>), program execution will be terminated. If <code>COMEXS</code> is set to 1 (<code>COMEXS1</code>), command processing will continue. With <code>COMEXS</code> set to 2 (<code>COMEXS2</code>), program execution is terminated, but the <code>INSELP</code> value is retained. Motion deceleration during the stop is controlled by the <code>AD</code> & <code>ADA</code> commands. If error bit #8 is enabled (e.g., <code>ERROR.8-1</code>), activating a Stop input will set the error bit and cause a branch to the <code>ERRORP</code> program.
E	Pause/Continue. If <code>COMEXR</code> is disabled (<code>COMEXR0</code>), then only command execution pauses, not motion. With <code>COMEXR</code> enabled (<code>COMEXR1</code>), both command and motion execution are paused. After motion stops, you can release the input or issue a continue (<code>!C</code>) command to resume command processing (and motion of in <code>COMEXR1</code> mode).
F	User Fault. Refer to the <code>ERROR</code> command. If error bit #7 is enabled (e.g., <code>ERROR.7-1</code>), activating a User Fault input will set the error bit and cause a branch to the <code>ERRORP</code> program. CAUTION: Activating the user fault input sends an <code>!K</code> command to the controller, "killing" motion on all axes (refer to the <code>K</code> command description for ramifications).
G,H	Reserved
I	Alarm Event - Will cause the 6K controller to set an Alarm Event in the Communications Server over the Ethernet interface. You must first enable the Alarm checking bit for this input-driven alarm (<code>INTHW.23-1</code>). For details on using alarms, refer to the <i>6K Series Programmer's Guide</i> .
aJ	JOG positive-direction - Will jog the axis specified in a positive-direction. The <code>JOG</code> command must be enabled for this function to work. Axis number required.
aK	JOG negative-direction. Will jog the axis specified in a negative-direction. The <code>JOG</code> command must be enabled for this function to work. Axis number required.
aL	JOG Speed Select. Selects the high or low velocity range while jogging. If the input is active, the high jog velocity range will be selected. Axis number is optional. If no axis number is designated, it defaults to all axes.
M	Joystick Release. Signals the controller to end joystick operation and resume program execution with the next statement in your program. When the input is open (high), the joystick mode is disabled (joystick mode can be enabled only if the input is closed, and only with the <code>JOY</code> command). When the input is closed (low), joystick mode can be enabled with the <code>JOY</code> command. The process of using Joystick mode is: <ol style="list-style-type: none"> 1. Assign the "Joystick Release" input function to a programmable input. 2. At the appropriate place in the program, enable joystick control of motion (with the <code>JOY</code> command). (Joystick mode cannot be enabled unless the "Joystick Release" input is closed.) When the <code>JOY</code> command enables joystick mode for the affect axes, program execution stops on those axes (assuming the Continuous Command Execution Mode is disabled with the <code>COMEXC0</code> command). 3. Use the joystick to move the axes as required. 4. When you are finished using the joystick, open the "Joystick Release" input to disable the joystick mode. This allows program execution to resume with the next statement after the initial <code>JOY</code> command that started the joystick mode.
N	Joystick Axis Select. Allows you to control two pairs of axes with one joystick. Use the <code>JOYAXH</code> and <code>JOYAXL</code> commands to assign analog inputs to control specific axes. Opening the Axis Select input (input is high) selects the <code>JOYAXH</code> configuration. Closing the Axis Select input (input is low) selects the <code>JOYAXL</code> configuration. NOTE: When this input is not connected, the <code>JOYAXH</code> configuration is always in effect.

Continued from previous page

- **Joystick Velocity Select.** Allows you to select the velocity for joystick motion. The JOYVH and JOYVL commands establish two joystick velocities. Opening the Velocity Select input (input is high) selects the JOYVH configuration. Closing the Velocity Select input (input is low) selects the JOYVL configuration. The JOYVL velocity could be used to quickly move to a location, the JOYVH velocity could be used for low-speed accurate positioning. NOTE: When this input is not connected, joystick motion always uses the JOYVH velocity setting.

- iP **Program Select.** One to one correspondence for input vs. program number. The program number comes from the TDIR command. The number specified before the program name is the number to specify within this input definition. For example, in the LIMFNC1-3P command, 3 is the program number. An input defined as a Program Select Input will not function until the INSELP command has been enabled.

- Q **Program Security.** Issuing the LIMFNCi-Q command enables the *Program Security* feature and assigns the *Program Access* function to the specified programmable input.

The program security feature denies you access to the DEF, DEL, ERASE, MEMORY, LIMFNC, and INFNC commands until you activate the program access input. Being denied access to these commands effectively restricts altering the user memory allocation. If you try to use these commands when program security is active (program access input is not activated), you will receive the error message *ACCESS DENIED. *The LIMFNCi-Q command is not saved in battery-backed RAM, so you may want to put it in the start-up program (STARTP).*

For example, once you issue the LIMFNC10-Q command, the positive end-of-travel limit for axis 4 is assigned the program access function and access to the DEF, DEL, ERASE, MEMORY, LIMFNC, and INFNC commands will be denied until you activate the input.

To regain access to these commands without the use of the program access input, you must issue the LIMEN command to disable the program security input, make the required user memory changes, and then issue the LIMEN command to re-enable the input. For example, if limit input 3 is assigned as the Program Security input, use LIMEN. 3=1 to disable the input and leave it activated, make the necessary user memory changes, and then use LIMEN. 3=E to re-enable the input.

- aR **End-of-Travel Limit, Positive Direction.** This is the factory default function for each dedicated hardware positive-direction end-of-travel limit input found in the "LIMITS" connector(s). If a trigger input or a digital input on an external I/O brick is assigned this function (e.g. 2INFNC1-1R), then change the respective limit input's function to something else (e.g., change LIMFNC1-1R to LIMFNC1-A). When an input is assigned this function, it is not debounced.

- aS **End-of-Travel Limit, Negative Direction.** This is the factory default function for each dedicated hardware negative-direction end-of-travel limit input found in the "LIMITS/HOME" connector(s). If a trigger input or a digital input on an external I/O brick is assigned this function (e.g. 2INFNC2-1S), then change the respective limit input's function to something else (e.g., change LIMFNC2-1S to LIMFNC2-A). When an input is assigned this function, it is not debounced.

- aT **Home Limit.** This is the factory default function for each dedicated hardware home limit input found in the "LIMITS/HOME" connector(s). If a trigger input or a digital input on an external I/O brick is assigned this function (e.g. 2INFNC3-1T), then change the respective limit input's function to something else (e.g., change LIMFNC3-1T to LIMFNC3-A). When an input is assigned this function, it is not debounced.

Example:

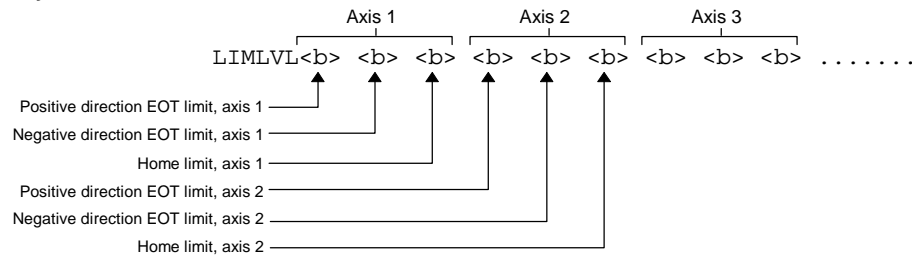
```
LIMFNC10-3D      ; Redefine the positive EOT input for axis 4 (limit input #10)
                  ; to be a stop input for axis 3
```

LIMLVL Hardware Limit Input Active Level

Type	Limit (End-of-Travel and Homing)	Product	Rev
Syntax	<!>LIMLVL ... (see drawing below)	6Kn	5.0
Units	n/a		
Range	b = 0 (active low: requires n.c. EOT switch & n.o. home switch), 1 (active high: requires n.o. EOT switch & n.c. home switch), or X (don't care)		
Default	0		
Response	LIMLVL: *LIMLVL000_000_000_000_000_000_000_000		
See Also	[AS], LH, LIMEN, [LIM], LIMFNC, HOM, TAS, TLIM		

Use the LIMLVL command to define the active state of all dedicated hardware end-of-travel and home limits (found on the “LIMITS/HOME” connectors). The default state is active low.

Command Syntax:



Active Level Setting	Required Switch Type *	State	LIM/TLIM Report
Active low (LIMLVL0) <i>This is the default setting.</i>	End-of-travel limit: N.C. Home limit: N.O.	Grounded — sinking current (device driving the input is on)	1 (active)
		Not Grounded — not sinking current (device driving the input is off)	0 (inactive)
Active high (LIMLVL1)	End-of-travel limit: N.O. Home limit: N.C.	Grounded — sinking current (device driving the input is on)	0 (inactive)
		Not Grounded — not sinking current (device driving the input is off)	1 (active)

* Compumotor recommends that all end-of-travel limit switches be normally-closed, because with normally-closed limit switches the limit function (i.e., inhibit motion) is considered active when the switch contact is open or if the wiring to the switch is broken.

Axis Status (AS, TAS, and TASF) bits 15 and 16 indicate when a hardware end-of-travel limit has been activated (i.e., invoking the “inhibit motion” function).

Wiring instructions and specifications for the limit inputs are provided in your 6K product’s *Installation Guide*.

LN		End of Loop	
Type	Loops or Program Flow Control	Product	Rev
Syntax	<!>LN	6K	5.0
Units	n/a		
Range	n/a		
Default	n/a		
Response	No response; used in conjunction with the L command		
See Also	L, LX		

The End of Loop (LN) command marks the end of a loop. You must use this command in conjunction with the Loop (L) command. All buffered commands that you enter between the L and LN commands are executed as many times as the number that you enter following the L command. You may nest loops up to 16 levels deep. **NOTE:** Be careful about performing a GOTO between the L and LN commands. Branching to a different location within the same program will cause the next loop encountered to be nested within the previous loop, unless an LN command has already been encountered.

Example:

```
L5          ; Repeat the commands between L and LN five times
GO1110     ; Start motion on axes 1, 2, and 3, axis 4 will remain motionless
LN         ; End loop
```

LOCK		Lock Resource to Task	
Type	Multi-tasking	Product	Rev
Syntax	<!>LOCK<i,i>	6K	5.0
Units	1st i = resource number		
	2nd i = 1 (lock the resource) or 0 (unlock the resource)		
Range	1st i = 1 (COM1 port), 2 (COM2 port), or 3 (task swapping)		
	2nd i = 1 (lock the resource) or 0 (unlock the resource)		
Default	0 (= not locked)		
Response	LOCK (see example below)		
See Also	[,], DRPCHK, E, PORT, TSKTRN		

Use the LOCK command to make a resource available only to the specified task. The LOCK-able resources are:

- COM1 — the “RS-232” communication port or the “ETHERNET” communication port
- COM2 — the “RS-232/485” communication port
- Task Swapping — When task swapping is locked to a specific task, statements in all other tasks will not be executed until the task swapping is again unlocked.

To check the LOCK status of all available resources, enter the LOCK command without field value. Below is an example response:

```
*LOCK1,0 COM PORT 1 - UNLOCKED
*LOCK2,0 COM PORT 2 - UNLOCKED
*LOCK3,0 TASK SWAPPING - UNLOCKED
```

NOTES

- If one task attempts to lock a resource in a different task (e.g., if Task1 attempts to execute the 2%LOCK1,1 command), the controller will respond with an error message (“ALTERNATE TASK NOT ALLOWED”).
- If task “A” attempts to lock a resource that is already locked to task “B”, command processing in task “A” will pause on the LOCK command until task “B” unlocks the resource, at which time task “B” will be able to lock the resource and continue processing.
- A resource may be locked by a task only while that task is executing a program. If program execution is terminated for any reason (e.g., stop, kill, limit, fault, or just reaching the END of a program), all resources locked by that task will become unlocked.

Example:

```

LOCK1,1          ; Ensure exclusive COM1 access for the task executing
                 ; this program
WRITE"travel is" ; First part of output string
WRVAR1          ; Numeric value of travel
WRITE" inches."  ; Finish complete string
LOCK1,0          ; Allow other tasks access to COM1

```

LS**Soft Limit Enable**

Type	Limit (End-of-Travel)	Product	Rev
Syntax	<!><@><a>LS<i>,<i>,<i>,<i>,<i>,<i>,<i>,<i>	6K	5.0
Units	n/a		
Range	i = 0 (disable both), 1 (disable positive-direction), 2 (disable negative-direction) or 3 (enable both)		
Default	0		
Response	LS: *LS0,0,0,0,0,0,0,0 1LS: *1LS0		
See Also	[AS], [ER], LSAD, LSADA, LSNEG, LSPOS, TAS, TER, TSTAT		

The Soft Limit Enable (LS) command determines the status of the programmable soft move distance limits. With soft limits disabled, motion will not be restricted. After a soft limit absolute position has been programmed (LSPOS and LSNEG), and the soft limit is enabled (LS), a move will be restricted upon reaching the programmed soft limit absolute position. The rate at which motion is decelerated to a stop upon reaching a soft limit is determined by the LSAD and LSADA commands.

Disable negative- and positive-direction soft limits	i = 0
Enable negative-direction, disable positive-direction soft limit	i = 1
Enable positive-direction, disable negative-direction soft limit	i = 2
Enable negative- and positive-direction soft limits	i = 3

NOTE: The controller maintains an absolute count, even though you may be programming in the incremental mode (MAØ). The soft limits will also function in incremental mode (MAØ) or continuous mode (MC1). The soft limit position references the commanded position, not the position as measured by the feedback device (e.g., encoder).

NOTE

If a soft limit is encountered while limits are enabled, motion must occur in the opposite direction before a move in the original direction is allowed. You cannot use the PSET command to clear the soft limit condition. If limits are disabled, you are free to make a move in either direction.

Example:

```

LSPOS500000,50000 ; Set soft limit positive-direction absolute positions to be
                 ; 500000 units for axis 1, 50000 units for axis 2
                 ; (Soft limits are always absolute)
LSNEG-500000,-50000 ; Set soft limit negative-direction absolute positions to
                 ; be -500000 units for axis 1, -50000 units for axis 2
                 ; (Soft limits are always absolute)
LS3,3              ; Soft limits are enabled on axes 1 and 2
LSAD100,100        ; Soft limit decel set to 100 units/sec/sec on axes 1 and 2
PSET0,0,0,0        ; Set absolute position on all axes to 0
A10,12             ; Set accel to 10 and 12 units/sec/sec for axes 1 and 2
V1,1               ; Set velocity to 1 unit/sec for axes 1 and 2
D100000,1000       ; Set distance to 100000 and 1000 units for axes 1 and 2
GO11XX            ; Initiate motion on axes 1 and 2

```

LSAD

Soft Limit Deceleration

Type	Limit (End-of-Travel)	Product	Rev
Syntax	<!><@><a>LSAD<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = units/sec/sec		
Range	0.00001-39,999,998 (depending on the scaling factor)		
Default	100.0000		
Response	LSAD: *LSAD100.0000,100.0000,100.0000,100.0000 ... 1LSAD: *1LSAD100.0000		
See Also	DRES, LHAD, LS, LSADA, LSNEG, LSPOS, SCALE, SCLA		

The Soft Limit Deceleration (LSAD) command determines the value at which to decelerate after a programmed soft limit (LSPOS or LSNEG) has been hit.

UNITS OF MEASURE and SCALING: refer to page 16.
--

The soft limit deceleration remains set until you change it with a subsequent soft limit deceleration command. Decelerations outside the valid range are flagged as an error, with a message *INVALID DATA-FIELD x, where x is the field number. When an invalid deceleration is entered the previous deceleration value is retained.

Example: Refer to the soft limit enable (LS) command example.

LSADA

Soft Limit Average Deceleration

Type	Motion (S-Curve)	Product	Rev
Syntax	<!><@><a>LSADA<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = units/sec/sec		
Range	0.00001-39,999,998 (depending on the scaling factor)		
Default	100.0000 (default is a constant deceleration ramp, where LSADA tracks LSAD)		
Response	LSADA: *LSADA100.0000,100.000,100.000,100.000 ... 1LSADA: *1LSADA100.0000		
See Also	AD, ADA, LS, LSAD, SCALE, SCLA		

The Soft Limit Average Deceleration (LSADA) command allows you to specify the average deceleration for an S-curve deceleration profile when a soft limit is hit. S-curve profiling provides smoother motion control by reducing the rate of change in deceleration; this decel rate of change is known as *jerk*. Refer to page 13 for details on S-curve profiling.

Acceleration scaling (SCLA) affects LSADA the same as it does for LSAD. Refer to page 16 for details on scaling.

Example:

```
LSAD10,10,10,10 ; Sets the maximum deceleration of all four axes  
LSADA5,5,7.5,10 ; Sets the average deceleration of all four axes
```

LSNEG

Soft Limit Negative Travel Range

Type	Limit (End-of-Travel)	Product	Rev
Syntax	<!><@><a>LSNEG<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = units of distance		
Range	-999,999,999 - +999,999,999 (scalable)		
Default	+0		
Response	LSNEG: *LSNEG+0,+0,+0,+0,+0,+0,+0,+0 1LSNEG: *1LSNEG+0		
See Also	LS, LSAD, LSADA, LSPOS, PSET, SCALE, SCLD		

The LSNEG command specifies the distance in absolute units where motion will be restricted when traveling in a negative-travel direction. The reference position used to determine absolute position is set to zero upon power-up, and can be reset using the PSET command. **Be sure to set the LSPOS value greater than the LSNEG value.**

The LSNEG value remains set until you change it with a subsequent LSNEG command.

All soft limit values entered are in absolute steps. If scaling is enabled (SCALE1), LSNEG is internally multiplied by the distance scale factor (SCLD). The soft limit position references the commanded position, not the position as measured by a feedback device (e.g., encoder).

Example: Refer to the soft limit enable (LS) command example.

LSPOS

Soft Limit Positive Travel Range

Type	Limit (End-of-Travel)	Product	Rev
Syntax	<!><@><a>LSPOS<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = units of distance		
Range	-999,999,999 - +999,999,999 (scalable by SCLD)		
Default	+0		
Response	LSPOS: *LSPOS+0,+0,+0,+0,+0,+0,+0,+0 1LSPOS: *1LSPOS+0		
See Also	LS, LSAD, LSADA, LSNEG, PSET, SCALE, SCLD		

The LSPOS command specifies the distance in absolute units where motion will be restricted when traveling in a positive-travel direction. The reference position used to determine absolute position is set to zero upon power-up, and can be reset using the PSET command. **Be sure to set the LSPOS value greater than the LSNEG value.**

The LSPOS value remains set until you change it with a subsequent LSPOS command.

All soft limit values entered are in absolute steps. If scaling is enabled (SCALE1), LSPOS is internally multiplied by the distance scale factor (SCLD). The soft limit position references the commanded position, not the position as measured by a feedback device (e.g., encoder).

Example: Refer to the soft limit enable (LS) command example.

LX

Terminate Loop

Type	Loops; Program Flow Control	Product	Rev
Syntax	<!>LX	6K	5.0
Units	n/a		
Range	n/a		
Default	n/a		
Response	n/a		
See Also	L, LN, PLN, PLOOP		

The Terminate Loop (LX) command terminates the current loop in progress. This command does not halt processing of the commands in the loop until the last command in the current loop iteration is executed. At this point, the loop is terminated. If there are nested loops, only the inner most loop is terminated.

This command can be used externally to terminate the loop only if it is preceded by the immediate command specifier (!LX). If the immediate command specifier is not used, the command will have no effect on a loop in progress. An example of where the buffered Terminate Loop command (LX) might be used is provided below.

Example:

```
; *****
; This program will make the move specified by the GO1110 command
; indefinitely until input 2 goes high, at which point, an LX will
; be issued, terminating the loop.
; *****
L0          ; Repeat the commands between L and LN infinitely, or until
           ; a Terminate Loop (LX) command is received
GO1110     ; Start motion on axes 1, 2, and 3,
           ; axis 4 will remain motionless
IF(IN=bX1) ; If onboard trigger input A2 goes high, execute all
           ; statements between IF and NIF
LX         ; Terminate loop
NIF       ; End IF statement
LN        ; End loop
```