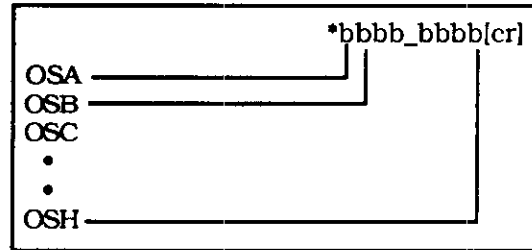


OS

NAME: Function Setup Report
SYNTAX: a**OS**
DEFAULT: None
RANGE: b = 0, 1
VALID:
RESPONSE: a**OS** *bbbb_bbbb[cr]
TYPE: Status
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: Reports the status of **OS** command (**OSA-OSH**) settings.



SEE ALSO: **OSB, OSC, OSD, OSE, OSG, OSH**

OSB

NAME: Back up to Home Switch
SYNTAX: <a>**OSB**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: **OSB0** Do not back up to home switch
OSB1 Back up to home switch

If you do not enable this function, the JSI will consider the motor at Home if the home input is active at the end of deceleration after encountering the active edge of the Home region. If you enable this function, the JSI will decelerate the motor to a stop after encountering the active edge of the Home region, and then move the motor in the opposite direction of the initial Go Home move with the GHF velocity until the active edge of the Home region is encountered. The JSI will then consider the motor at Home. This occurs regardless of whether or not the home input is active at the end of the deceleration of the initial Go Home move.

SEE ALSO: **GH, OSC, OSD, OSG, OSH, OS**

OSC

NAME: Define Active State of Home Switch
SYNTAX: <a>**OSCb**
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: **OSC0** Active state of home input is a low signal level
OSC1 Active state of home input is a high signal level

This command sets the active state of the home input. It enables you to use either a normally closed or a normally open switch for homing. **OSC0** requires a low voltage signal to activate the home limit input.

OSC1 requires a high-voltage signal to activate the home limit input.

SEE ALSO: **GH, OS**
EXAMPLES: **Command** Description
OSC1 Sets home input's active state to a high state

OSD

NAME: Enable Encoder Z Channel Input
SYNTAX: <a>**OSDb**
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: **OSD0** Disable Z channel function during home
OSD1 Recognizes Z channel after encountering the home

The Z channel is used (in conjunction with a load activated switch connected to home limit) to determine the home position. The switch determines the home region. The Z channel determines the exact position in that region that will be used as the home reference.

SEE ALSO: **OSB, OSC, OSE, OSF, GH, GHA, GHF, GHV**
EXAMPLES: **Command** Description
OSD0 Disable Z channel function

OSE

NAME: Jog Enable
SYNTAX: <a>**OSE**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The jogging functions are configured with the **IN** command, and are enabled with the **OSE** command.

OSE0 Jog Disable
OSE1 Jog Enable

SEE ALSO: **IN, JVL, JA, INL**

OSG

NAME: Final homing direction
SYNTAX: <a>**OSG**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: **OSG0** Sets the final Home approach direction to be clockwise
OSG1 Sets the final home approach direction to be counterclockwise

This enables you to approach home from any direction yet stop at home repeatedly in the same edge of the switch

SEE ALSO: **OSH**

OSH

NAME: Reference Edge of Home Switch
SYNTAX: <a>**OSH**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: **OSH0** Selects the clockwise side of the Home signal as the "edge" on which the final approach will stop
OSH1 Selects the counterclockwise side of the home signal as the "edge" on which the final approach will stop

The clockwise edge of the Home switch is defined as the first switch transition seen by the JSI when traveling off of the CW limit in the CCW direction. If $n = 1$, the counterclockwise edge of the Home switch will be referenced as the Home position. The counterclockwise edge of the Home switch is defined as the first switch transition seen by the JSI when traveling off of the CCW limit in the CW direction.

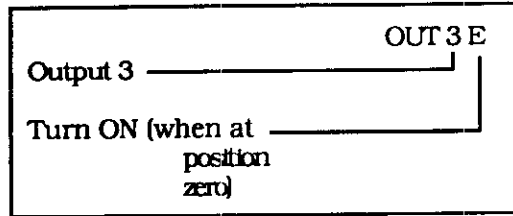
SEE ALSO: **OSG**

OUT

NAME: Output Functions
SYNTAX: <a>**OUT**n
DEFAULT: x = A
RANGE: n = 1-8, x = A-n
VALID:
RESPONSE: a**OUT**n See Example
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command sets the functions for each of the eight outputs. All the outputs can be configured for different functions. The factory setting for all eight outputs are programmable output bits. The following table lists the functions available for each output bit.

A	<i>Programmable Output</i> - Used with O and IO commands
B	<i>Moving/Not Moving</i> - On when motion is occurring
C	<i>Sequence in Progress</i> - On when a sequence is being executed.
D	<i>At Limits</i> - On when the motor has reached either a software or hardware limit
E	<i>At Position Zero</i> - On when the absolute position counter is zero
F	<i>Fault</i> - On when a fault has occurred. See RSE command for possible fault conditions
G	<i>Deadband</i> - On when position error exceeds the deadband set by CDB command
H	<i>Amp Off</i> - Indicates when you turn off the drive using OFF or ST1 command
I	<i>Reserved</i>
J	<i>Strobe</i> - Turns on to load bytes of parallel data. Indicates to the interface that the JSI is ready for data.
K	<i>Command Error</i> - On when RS-232C or BCD input has bad data
L	<i>Position Error Fault</i> - On when following error set by CPE command is exceeded. This situation also creates an error 20
M	<i>Reserved</i>
N	<i>CW Software Limit Reached</i> - CW Software limit set with SL command has activated
O	<i>Reserved</i>
P	<i>CCW Software limit reached</i> - CCW Software limit set with SL command has activated
Q	<i>Reserved</i>
R	<i>CW Hardware limit activated</i>
S	<i>CCW hardware limit activated</i>
T	<i>Output based on position</i> - This output goes on when a condition specified by OUTP command exists.
Z	<i>No Function</i>
OTHERS:	<i>No Function</i>

Outputs 1 through 8 can be configured as any one of the functions described above. For example, output 3 is set as an at position zero indicator::



SEE ALSO:
EXAMPLE 1:

IN, OUTP, OUTL
The OUT command also reports the status of each output bit.
aOUT1 *1_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)

EXAMPLE 2:

By not specifying the output bit number, JSI reports the status of all the output bits.
aOUT
*1_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*2_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*3_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*4_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*5_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*6_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*7_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)
*8_A_PROGRAMMABLE_OUTPUT_(STATUS_ON)

OUTL

NAME:
SYNTAX:
DEFAULT:
RANGE:
VALID:
RESPONSE:
TYPE:
ATTRIBUTES:
DESCRIPTION:

Set Active Output Level
<a>**OUTLn**
n = 0
n = 0, 1, x = Low or High
aOUTL *n_OUTPUT_SIGNAL_LEVEL_ACTIVE_x[cr]
Setup
Buffered, Device Address Optional, Savable in Sequence
This command configures the voltage level that the JSI considers to be an active output signal.

OUTL0 Sets a low level (0V) as an active signal
OUTL1 Sets a high level (5-30V) as an active signal

Outputs 1 through 8 are open collector outputs. Therefore, you must supply power (5-30VDC) to run the outputs. The output is able to sink up to 300 mAmps of current. You must configure the outputs using the **OUT** command if you wish to use the outputs for anything other than programmable outputs.

SEE ALSO:
EXAMPLE:

OUT	<u>Command</u>	<u>Description</u>
	> OUTL0	Set a low level (0V) active signal
	> OUTL1	Set output 1 as a programmable output
	> 1OUTL *0_OUTPUT_SIGNAL_LEVEL_ACTIVE_LOW[cr]	

OUTP

NAME: Output on Position
SYNTAX: <a>**OUTP**xn,<t>
DEFAULT: x = A, n = 0, t = 5
VALID:
RESPONSE: a**OUTP** *OUTPUT_ON_x_POSITION_n[cr]
 *POSITION_OUTPUT_ON_TIME_t_MILLISECONDS_[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: An output can be configured to activate based on the position with the OUT command. The comparison position is entered as either an incremental or absolute position.

The command **OUTP** is configured as **OUTP**xn,<t> where x is A for Absolute or I for incremental mode, n is distance, and t is the time in milliseconds the output is active (only in incremental mode).

In absolute mode, the output turns on when the current position is greater than the entered distance. The time is ignored in this mode.

In incremental mode, the output turns on each time the distance is traversed and stays activated for the entered time. The sign for the distance will convert to a plus sign as the distance is an absolute value.

SEE ALSO: **OUT**
EXAMPLE 1:

<u>Command</u>	<u>Description</u>
> OUT2T	Set Output 2 to activate based on position
> OUTPA1000	When the absolute position is greater than +1000, the output turns on

EXAMPLE 2:

<u>Command</u>	<u>Description</u>
> OUT2T	Set output 2 to activate based on position
> OUTPI,5000,500	The output will activate for 500 milliseconds every 5000 steps

PR

NAME: Absolute Position Report
SYNTAX: a**PR**
DEFAULT: None
RANGE: n = ±2,147,483,647
VALID:
RESPONSE: a**PR** *±n[cr]
TYPE: Status
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: Absolute position request; response is *snnnnnnnnnn[cr]. Reports motor position with respect to power up position or last time a **SP** command was issued. The units of distance are scaled by the **CMR** command. The absolute position counter can track up to +/- 2³¹ - 1, or 2,147,483,647 steps. If the counter is overrun in the relative position mode (by running the motor continuously for long periods of time, 24 hours at 20 RPS and 5,000 steps per rev), the absolute position will be invalid.

In preset mode, response to this command will be reported after the move is done. In continuous mode, the response to this command will be reported after motor reaches constant velocity.

SEE ALSO:

MPI, MPA, MN, PZ, D

EXAMPLES:

<u>Command</u>	<u>Description</u>
> PZ	Resets the absolute counter to zero
> LDS	Disable both CW & CCW limits
> A10	Set Acceleration to 10 rps ²
> V5	Set velocity to 5 rps
> D5000	Set move distance to 5000 steps
> G	Executes the move (Go)
> LPR	Request absolute position report. Response should be (*+00000005000[cr])

PS

NAME: Pause
 SYNTAX: <a>**PS**
 DEFAULT: None
 RANGE: None
 VALID:
 RESPONSE: None
 TYPE: Programming
 ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
 DESCRIPTION:

This command pauses execution of a command string or sequence following the Pause (**PS**) command until the JSI receives a Continue (**C**) command. This command is useful if you need to enter a complete string of commands before you can execute your other commands.

This command is useful for interactive tests and in synchronizing multiple indexes that have long command strings.

SEE ALSO:

C

EXAMPLES:

<u>Command</u>	<u>Description</u>
> PS	Pauses execution of following commands until the JSI receives the Continue (C) command
> A5	Sets acceleration to 5 rps ²
> V5	Sets velocity to 5 rps
> D25000	Sets move distance to 25,000 steps
> G	Executes the move (Go)
> T2	Delays the next move for 2 sec
> G	Executes the move (Go)
> C	Continues Execution

When the JSI receives the **C** command, the motor moves 25,000 steps twice with a 2 second delay.

PZ

NAME: Set Absolute Counter to Zero
SYNTAX: <a>PZ
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command sets the absolute position counter to zero.

SEE ALSO: **MN, MP1, MPA, PR, D**

<u>Command</u>	<u>Description</u>
> MPA	Make all preset moves with respect to absolute zero position
> A10	Set Acceleration to 10 rps ²
> V5	Set Velocity to 5 rps
> D2500	Set move distance to 2500 steps
> G	Executes the move (Go)
> 1PR	Report Absolute Position (Response = *+2500)
> PZ	Zero the absolute counter
> 1PR	Report absolute position (Response = *0)

Q

NAME: Quote
SYNTAX: a"x
DEFAULT: None
RANGE: X = Any ASCII character or number (in the range of 33–126 decimal), with a maximum string length of 18 characters.
VALID:
RESPONSE: **X** = The ASCII range of characters accepted by the command
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Required, Savable in Sequence
DESCRIPTION: Any characters entered after the quotation marks(") will be transmitted, exactly as they were entered, over the RS-232C link. A space entered by the space bar indicates the end of the command. A space is always sent after the last character in the string. This command is used during buffered moves or sequences, or to command other Compumotor devices to move. The ASCII range of characters accepted by the command is 33 - 126 (decimal).

SEE ALSO:
EXAMPLE 1:

<u>Command</u>	<u>Description</u>
> MN	Set to mode normal (Preset Moves)
> A10	Set acceleration to 10 rps ²
> V5	Set velocity to 5 rps
> D12500	Set distance to 12,500 steps
> G	Executes the move (Go)
> 1"MOVE_DONE	After motor finished the move, the Compumotor JSI will send the message MOVE_DONE out from the RS-232C port.

EXAMPLE 2:

<u>Command</u>	<u>Description</u>
> MN	Set to mode normal (Preset Moves)
> A10	Set acceleration to 10 rps ²
> V5	Set velocity to 5 rps
> D12500	Set distance to 12,500 steps
> G	Executes the move (Go)
> 1"2XR1	Once the move is done, Run Sequence 1 is commanded on a unit with device address 2.

90

NAME: Exit Velocity Profiling Mode
SYNTAX: <a>**90**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The **90** command exits the velocity profiling mode. Motion will stop when **90** is issued.

SEE ALSO: **91, RM**
EXAMPLES: See **91** command

91

NAME: Enter Velocity Profiling Mode
SYNTAX: <a>**91**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The **91** command enters the JSI in velocity profiling mode. Subsequent **RM** commands will cause an immediate change in motion velocity. Use **90** to exit this mode.

SEE ALSO: **90, RM**
EXAMPLES: Command

<u>Command</u>	<u>Description</u>
> 91	Enter Velocity Profiling mode
> RM0100	Go to RM velocity of (100 hex) RM rps
> RM0500	Go to RM velocity of (500 hex) RM rps
> RM1000	Go to RM velocity of (1000 hex) RM rps
> RM0500	Go to RM velocity of (500 hex) RM rps
> RM0100	Go to RM velocity of (100 hex) RM rps
> 90	Exit velocity profiling mode

Motion will stop when **90** command is entered.

R

NAME: Report JSI Status
SYNTAX: a**R**
DEFAULT: None
RANGE: x=R, B, S, C
VALID:
RESPONSE: aR *x[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: The Request JSI Status (**R**) command can be used to indicate the general status of the JSI. Possible responses are:

Response

<u>Character</u>	<u>Definition</u>
* R	Ready
* S	Ready, Attention Needed
* B	Busy
* C	Busy, Attention Needed

The following conditions will cause a response indicating that the JSI is busy:

- * Performing a preset move
- * Performing a continuous move
- * A time delay is in progress. (**T** command)
- * In **RM** mode
- * Paused
- * Waiting on a Trigger
- * In Jog mode
- * Going Home
- * In Power-on sequence mode
- * Running a sequence
- * Executing a loop

The following conditions will cause a response indicating than an error exists.

- * A feedback error condition exists.
- * Go home failed
- * Limit has been encountered
- * Sequence execution was unsuccessful

When the response indicates that attention is required, more details on the error condition are available by using the **RA** or **RB** commands.

It is not recommended that this command be used in tight polling loops which could result in microprocessor over load. Inserting time delays can alleviate this problem.

This command is not intended to be used to determine if a move is complete. Rather it should be used after the move is complete to determine if there might be errors or faults.

Use a buffered status request command or a programmable output to indicate move completion.

SEE ALSO:
EXAMPLE:

RA, RB
Command
 > **IR**

Description
 (Response) = *R (JSI ready to accept a command and no error conditions require attention.)

RA

NAME: Limit Switch Status Report
 SYNTAX: aRA
 DEFAULT:
 RANGE: x = @, A, B, C, D, E, F, G, H, I, J, K, L, M, N, O
 VALID:
 RESPONSE: aRA *x[cr]
 TYPE: Status
 ATTRIBUTES: Immediate, Device Address Required, Never Saved
 DESCRIPTION: The Limit Switch Status Report (RA) command responds with the status of the end of travel limits during the last move as well as the present condition. This is done by responding with one of 16 characters representing the conditions listed below.

Response Character	Last Move Terminated By		Limit Switch Active	
	CW Limit	CCW Limit	CCW Limit	CW Limit
*@	NO	NO	NO	NO
*A	YES	NO	NO	NO
*B	NO	YES	NO	NO
*C	YES	YES	NO	NO
*D	NO	NO	YES	NO
*E	YES	NO	YES	NO
*F	NO	YES	YES	NO
*G	YES	YES	YES	NO
*H	NO	NO	NO	YES
*I	YES	NO	NO	YES
*J	NO	YES	NO	YES
*K	YES	YES	NO	YES
*L	NO	NO	YES	YES
*M	YES	NO	YES	YES
*N	NO	YES	YES	YES
*O	YES	YES	YES	YES

The RA command is useful when the motor will not move in either or both directions. The report back will indicate whether or not the last move was terminated by one or both end of travel limits.

RB

NAME: Loop, Pause, Shutdown, Trigger Status Report
 SYNTAX: aRB
 DEFAULT: None
 RANGE: x = @, A, B, C, D, E, F, G, H, I, J, K, L, M, N, O
 VALID:
 RESPONSE: aRB *x[cr]
 TYPE: Status
 ATTRIBUTES: Immediate, Device Address Required, Never Saved
 DESCRIPTION: Responds with the form *x[cr] where x = "@" - "O"

This command reports the status of four functions within the JSI; Loop in Progress, Pause in Progress, Trigger Wait Active and Shutdown active.

Response Character	Loop Active	Pause Active	Shutdown Active	Trigger Active
*@	NO	NO	NO	NO
*A	YES	NO	NO	NO
*B	NO	YES	NO	NO
*C	YES	YES	NO	NO
*D	NO	NO	YES	NO
*E	YES	NO	YES	NO
*F	NO	YES	YES	NO
*G	YES	YES	YES	NO
*H	NO	NO	NO	YES
*I	YES	NO	NO	YES
*J	NO	YES	NO	YES
*K	YES	YES	NO	YES
*L	NO	NO	YES	YES
*M	YES	NO	YES	YES
*N	NO	YES	YES	YES
*O	YES	YES	YES	YES

SEE ALSO:

R, RA, TR, PS, L, ST

EXAMPLES:

CommandDescription

> 1RB

After using 1RB, response came back as *A. This means that the JSI is currently executing a loop .

RFS

NAME: Return Drive Parameters to Factory Setting

SYNTAX: <a>RFS

DEFAULT: None

RANGE: None

VALID:

RESPONSE: None

TYPE: Setup

ATTRIBUTES: Immediate, Device Address Optional, Never Saved

DESCRIPTION: Return the drive to factory setting. The Save (SV) command must be used to retain the values in nonvolatile memory. The following settings are changed:

Proportional Gain (PG) set to 10
 Proportional Maximum (CPM) set to 500
 Integral Gain (CIG) set to 0
 Integral Maximum (CIM) set to 1000
 Differential Gain (CDG) set to 0
 Differential Maximum (CDM) set to 100
 Dead Band (CDB) set to 0
 Motion Resolution (CMR) set to 4000
 Maximum Position Error (CPE) set to 4000
 Integrator Sum Limit (CIL) set to 65,535

SEE ALSO:

RIFS, SV

EXAMPLES:

None

RG

NAME: Go Home Status Report
SYNTAX: a**RG**
DEFAULT: None
RANGE: x = @,A
VALID:
RESPONSE: a**RG** *x[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: Go Home Status request; responds with either *@ or *A indicating success or failure of last go home attempt.

<u>Response</u>	<u>Go Home Successful</u>
*@	NO
*A	YES

SEE ALSO: **GH, R, RA, RB**

RIFS

NAME: Return JSI to Factory Settings
SYNTAX: <a>**RIFS**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: Execution of this command will cause the following JSI parameters to return to factory default settings.

Configure Feed Back (**CFB**) set to 1 (incremental encoder)
 4000 (Feedback resolution)
 Software Limits (**SL**) set to 0, 0
 Jog Velocity High (**JVH**) set to 10 rps
 Jog Velocity Low (**JVL**) set to 1 rps
 Jog Acceleration (**JA**) set to 99 rps²
 Limit Acceleration (**LA**) set to 900 rps²
 Go Home Velocity (**GHV**) set to 1 rps
 Go Home Final (**GHF**) set to 0.1 rps
 Go Home Acceleration (**GHA**) set to 99 rps²
 Velocity (**V**) set to 1 rps
 Acceleration (**A**) set to 10 rps²
 Output Peak (**COP**) set to 100
 SS flags set to 0
 OS flags set to 0
 Configurable Inputs (**IN**) set to trigger inputs (A)
 Configurable Outputs (**OUT**) set to programmable outputs (A)
 All Sequences (1 - 100) are erased.

The command out signal must be disabled (**OFF** or **ST0**) to execute the **RIFS** command.

SEE ALSO: **RFS**
EXAMPLES: None

RM

NAME: Rate Multiplier in Velocity Streaming Mode
SYNTAX: <A>RMh
DEFAULT: RM00000
RANGE: h = 00000 to FFFFF
VALID:
RESPONSE: None
TYPE: Motion
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: RM command followed by 5 hexadecimal digits represent a velocity

The velocity change is essentially instantaneous; there is no acceleration/deceleration ramp between velocities. A limit switch-closure will stop movement while in velocity profiling mode, but does not cause the JSI to exit velocity streaming mode. **RM** (profiling) mode is unidirectional. The direction will be the last activated direction either from an actual move or from a **D** or **H** command. Bidirectional moves using this mode can be made by returning to velocity zero, switching off **RM** mode, changing the direction, and re-enabling **RM** mode. This extra overhead should be acceptable given the need to attend to going to velocity zero when changing directions in real situations.

To generate the hexadecimal for any particular speed:

$$dddd = (\text{feedback resolution}(\text{steps/rv}) * 256 * (\text{servo sampling time}))$$

Where the feedback resolution is from the **CFB** command and the servo sampling time is either 0.000512 (**CFB1**) or 0.001024 (**CFB2**). Then convert dddd to hex

SEE ALSO:
EXAMPLE:

g1, g0	<u>Description</u>
Command	
> CFB1,4000	Set to feedback resolution of 4000 and servo sampling time to 0.0000512
> g1	Enter Velocity streaming mode
> RM00418	Accelerate to 2 rps
> RM00830	Accelerate to 4 rps
> T1	Run at 4 rps for 1 sec
> RM00418	Decelerate to 2 rps
> RM00000	Decelerate to 0 rps
> g0	Exit velocity streaming mode

Notes on Velocity Profiling Mode:

Situations requiring non-linear accelerations may use the **g0**, **g1** and **RM** commands. **g1** is used to enter the velocity profiling mode, and **g0** is used to exit. While in this mode the **RM** command is used to generate velocity values that are immediately implemented while the motor is moving. This means that the **RM** command must be sent to the JSI at the time the change in velocity is required. This creates a stair-step effect in velocity change. By implementing a large number of very small instantaneous velocity changes, a smooth, non-linear acceleration ramp can be achieved.

RS

NAME: Report Status of Sequence Execution
SYNTAX: a**RS**
DEFAULT:
RANGE: x = @, A, B, C, D
VALID:
RESPONSE: a**RS** *x[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: Responds in the form *x[cr] where X = "@" - "D".
 Bit 0: Sequence started; Bit 1: Sequence Ended

Response Character	Sequence Started	Sequence Ended	Bad Loop
*@	NO	NO	NO
*A	YES	NO	NO
*B	NO	YES	NO
*C	YES	YES	NO
*D	N/A	N/A	YES

Whenever a sequence is started, the sequence start bit is set and the sequence end bit is cleared (this only occurs if the sequence is valid and is actually run). Whenever a sequence is ended, the start bit is cleared and the end bit is set. Any abrupt move termination (e.g., limit activation), or a **K** or **S** command clears both bits.

***D** is reported when there is an unbalanced number of loops and loop terminators inside a sequence. Starting a loop in one sequence and terminating it in another sequence is not allowed. Nested loops require complete closure before execution will begin.

Sequence started is true when: An **XR**, **XRP** or a power-up successfully starts a sequence.

It is false when: A **STOP** or a **KILL** command is received, or limits are hit.

Sequence Ended is true when: An **XT** is encountered, when a **STOP** or **KILL** is executed, or when End-of-Travel limit is encountered.

Sequence ended is false when: A sequence is successfully started.

SEE ALSO:

XR, **XRP**

RSE

NAME: Report Servo Errors
SYNTAX: **aRSE**
DEFAULT: None
RANGE: See Below
VALID:
RESPONSE: See Below
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: You can find out what error condition exists in the JSI using this command. If you see the two digit LED flashing a code, you should start troubleshooting the unit using the **RSE** command. The error number and its description are reported. The possible error messages are:

16	Amplifier off
20	Following error exceeded
23	Enable plug not inserted
30	CRC error in EEPROM
41	CW Hardware Limit
42	CCW Hardware Limit
43	CW Software Limit
44	CCW Software Limit
60	Commanded Off
66	User Fault

POSSIBLE RESPONSES:

<u>Command</u>	<u>Response</u>
aRSE	*NO_ERRORS[cr]
aRSE	*COMMAND_OUT_DISABLED_BY_x[cr] x = 16,20, 23, 30, 60, 66
aRSE	*JSI_DISABLED_BY_x[cr] x = 41, 42, 43, 44

SEE ALSO: None

RV

NAME: Revision Level Report
SYNTAX: **aRV**
DEFAULT:
RANGE: n = 0 - 9, x = A - Z
VALID:
RESPONSE: **aRV *92-nnnnnn-nn<xn>**[cr]
TYPE: Status
ATTRIBUTES: Immediate, Device Address Required, Never Saved
DESCRIPTION: The Revision (**RV**) command responds with the software part number and its revision level. The response is in the form shown below:

***92-nnnnnn-nn<xn>**[cr]
 part number revision level

The part number identifies which product the software is written for, as well as any special features that the software may include. The revision level identifies when the software was written. You may want to record this information in your own records for future use. This type of information is useful when you consult Parker Compumotor's Applications Department.

SEE ALSO:

EXAMPLES:

<u>Command</u>	<u>Response</u>
1RV	Response = *92-007830-01Y1

 The product is identified by 92-007830. The revision level is identified by 01Y1.

S

NAME: Stop
SYNTAX: <a>**S**
DEFAULT: None
RANGE: None
VALID:
RESPONSE: None
TYPE:
ATTRIBUTES: Immediate, Device Address Optional, Never Saved
DESCRIPTION: This command decelerates motion to a stop using the last defined Acceleration (**A**) command. This command normally clears any remaining commands in the command buffer, unless prevented from doing so by the Clear/Save The Command Buffer On Stop (**SSH1**) command. When the **SSH1** command is present, the **S** command stops only the current move. The JSI executes the next command in the buffer. The Stop (**S**) command does not stop motion in Velocity Streaming or Rate Multiplier (**RM**) mode. If you are in the **RM** mode, issue an Exit Velocity Profiling Mode (**QØ**) command to stop the motion.

If the **SSL** command is enabled, issuing a Continue (**C**) command after a **S** command will resume execution by completing the interrupted move and continuing sequence or buffer execution.

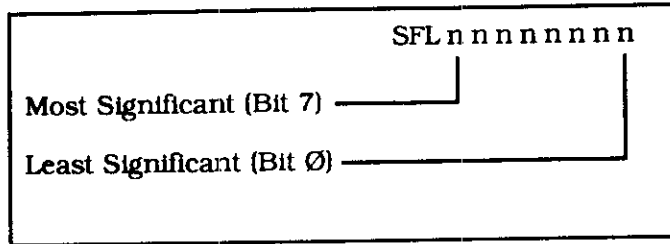
SEE ALSO: **K, SSH, QØ, A, SSL, C**

<u>Command</u>	<u>Description</u>
> MC	Sets move in continuous mode
> A1	Sets acceleration to 1 rps ²
> V1Ø	Sets velocity to 10 rps
> G	Executes the move (Go)
> A5	Sets Acceleration to 5 rps ²
> S	Stops (motor decelerates) to 0 rps at a rate of 5 rps ²

The **S** command is not buffered since it is an immediate command. As soon as the JSI receives the **S** command, it stops motion.

SFL

NAME: Set User Flag
SYNTAX: <a>**SFL**nnnnnnnn
DEFAULT: n = 0
RANGE: n = 0, 1 or X
VALID:
RESPONSE: <a>**SFL** *nnnn_nnnn[cr]
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional
DESCRIPTION: This command sets a condition of 8 different bits. You can define the state of bits 1 through 8 using this command to be either a 0, 1, or X. A 0 clears the corresponding bit, as 1 sets the corresponding bit, and an x retains the bit's present value. Not all the bits need to be defined during an **SFL** command; **SFL11** sets bits 6 and 7 while leaving the remaining bits unaltered.



Once you set bits 1 through 8, you can use the **IFFL** (If the flags match) command to do an **IF_THEN** operation.

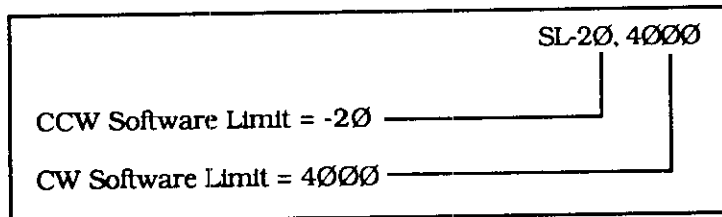
SEE ALSO:
EXAMPLE:

IFFL, NIF	<u>Description</u>
Command	
> SFL1010	Set bits 5 and 7, and clear bits 6 and 4
> IFFL1010	If user flag bits 5 and 7 are set, then issue the following commands
> A10	Set acceleration to 10 rps ²
> V5	Set velocity to 5 rps
> D25000	Set distance to 25,000 steps
> G	Executes the move (Go)
> NIF	End of IFFL

If **IFFL** pattern matches the **SFL** pattern, the motor will move 25,000 steps.

SL

NAME: Software Limits
SYNTAX: <a>**SL**n, x
DEFAULT: n = 0, x = 0
RANGE: n, x = ±2,147,483,647
VALID:
RESPONSE: a**SL** ***CCW_SOFTWARE_LIMIT_n**[cr] ***CW_SOFTWARE_LIMIT_x**[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command defines the counterclockwise (CCW) and Clockwise (CW) software end of travel limits respectively. Once you define the CCW and CW software limits, the motor will not be able to exceed those positions, unless the software limits are disabled by the **SLD3** command. You may use the **OUT** command to configure 1 or more outputs to signal if a software end of travel limit has been encountered. When the motor reaches the software limit, it will come to an immediate stop, using the acceleration specified.



SEE ALSO:
EXAMPLE:

OUT, SLD, LA	<u>Description</u>
Command	
> SL-10000,26944	Set CCW software limit to -1000 steps. Set CW software limit to 26944 steps.
> SLD0	Enable both CW and CCW software limits
> PZ	Set absolute position to zero
> A10	Set acceleration to 10 rps ²
> V5	Set velocity to 5 rps
> D40000	Set distance to 40,000 steps
> G	Executes the move (Go)

Motor will start moving and starts decelerating when it reaches 26944 steps.

SLD

NAME: Software Limit Disable
SYNTAX: <a>**SLD**n
DEFAULT: n = 3
RANGE: n = 0 - 3
VALID:
RESPONSE: a**SLD** *0_CW_AND_CCW_SOFTWARE_TRAVEL_LIMITS_ENABLED[cr] or
 *CCW_SOFTWARE_TRAVEL_LIMIT_ENABLED[cr] or
 *2_CW_SOFTWARE_TRAVEL_LIMIT_ENABLED[cr] or
 *3_NO_SOFTWARE_TRAVEL_LIMITS_ENABLED[cr]

TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command enables or disables the software end of travel limits defined by the **SL** command

- SLD0** Enables both CW and CCW Software limits. motion will not be allowed to go past the software limits
SLD1 Disables CW Software limit. Can travel past CW Software limit
SLD2 Disables CCW Software Limit. Can travel past CCW Software Limit.
SLD3 Disable both CW and CCW Software limits. Motor will ignore both CW and CCW software limit.

This command is very similar to the Hardware Limit Disable (**LD**) command, except it uses software limits.

You can use the **OUT** command to configure an output to indicate that the software limit has been reached. However, if you disable the limits, the output will not be activated even if the motion goes past the software limit.

SEE ALSO:
EXAMPLE:

SL, OUT	<u>Description</u>
> SL0,0	Set CCW software limit to 0 and CW software limit to 0
> PZ	Set absolute position to zero
> A10	Set acceleration to 10 rps ²
> V5	Set velocity to 5 rps
> D25000	Set distance to 25,000 steps
> SLD3	Disable both CW and CCW software limits
> G	Executes the move (Go)

If **SLD0** was entered in place of **SLD3**, motion would not have occurred, because the motor is sitting on the software limit.

SN

NAME: Scan Time Delay
SYNTAX: <a>**SN**n
DEFAULT: n = 50
RANGE: n = 1 to 1000
VALID:
RESPONSE: a**SN** *SCAN_TIME=n_MILLISECONDS[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The Scan (**SN**) command allows you to define the debounce time (in milliseconds) for external sequence selection inputs. The debounce time is the amount of time that the sequence inputs must remain constant for a proper reading from a remote controller, such as a programmable logic controller (PLC). If you are using a PLC you should change the debounce time to be greater than the scan time for the PLC to assure that the correct data is present when the data is read.

SEE ALSO: None
EXAMPLES:

Command	<u>Description</u>
> SN150	Sets scan time of sequence select inputs to 150 milliseconds.

SP

NAME: Set Position Absolute
SYNTAX: <a>**SP**n
DEFAULT: n = 0
RANGE: n = ± 1,999,999,999
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command allows you to set the absolute counter (value n). The **SP** command is useful for labeling a position value at a certain point. For example, you can set the zero reference point (home) to some location other than that of the physical hardware home. If you have a cut-off saw, for example, you may not be able to mount the home switch at the cut point. However, by mounting the home switch at a known distance away, and resetting the reference point with the **SP** command, you can make the system function as if the home switch were at the cut point.

Note that the units of the **SP** command are scaled by the **CMR** command. You can get a position report in the same units by using the **PR** command.

SEE ALSO: **MPI, MPA, PR, D, PZ, CMR**
EXAMPLES:

<u>Command</u>	<u>Description</u>
> MN	Sets mode to normal
> A2	Sets acceleration to 2 rps ²
> GH2	Instructs the motor to go home at 2 rps
> PZ	Sets the absolute position counter to zero units
> 1PR	*+000000000000 Reads the absolute position counter
> SP5000	Sets absolute counter to 5,000 units
> 1PR	*+00000005000 Reads the absolute position counter to verify the absolute position

SSG

NAME: Clear/Save The Command Buffer on Limit
SYNTAX: <a>**SSGb**
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: In most cases, it is desirable that upon activating an end of travel limit input, all motion should cease until the problem causing the over-travel is rectified. This will be assured if all commands pending execution in the command buffer are cleared when hitting a limit. This is the case if **SSG0** is specified. If **SSG1** is specified and a limit is activated, the current move is aborted, but the remaining commands in the buffer continue to be executed.

SEE ALSO: **LD, LA, SS**
EXAMPLES:

<u>Command</u>	<u>Description</u>
> SSG1	Save buffer on limit
> A10	Set acceleration to 10 rps ²
> V5	Set velocity to 5 rps
> D25000	Set distance to 25,000 steps
> G	Executes the move (Go)
> O11	Activate programmable outputs 1 and 2

If a limit switch is encountered while executing the move, outputs 1 and 2 will still go on.

SSH

NAME: Clear/Save The Command Buffer on Stop
SYNTAX: <a>**SSHb**
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: a**SSH** *n[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION:

SSH0	=	Clears command buffer
SSH1	=	Saves command buffer

In Normal Operation (**SSH0**) the Stop (**S**) command or a dedicated stop input will cause any commands in the command buffer to be cleared. If you select the Save Command Buffer On Stop (**SSH1**) command, a remote stop input or Stop (**S**) command will only stop execution of a move in progress. It will not stop execution of any commands that remain in the buffer.

SEE ALSO: **LD, SS**
EXAMPLES:

<u>Command</u>	<u>Description</u>
> SSH0	Clears buffer on stop
> A10	Sets acceleration to 10 rps ²
> V5	Sets velocity to 5 rps
> D25000	Sets distance to 25,000 steps
> L50	Loops 50 times
> G	Executes the move (Go)
> T.5	Pauses for 500 msec
> N	Ends Loop
> S	Stops motion

When you issue the **S** command, the JSI will clear the buffer and stop the move.

SSI

NAME: Enable/Disable Interactive Mode
SYNTAX: <a>**SSI**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: When **SSI** is enabled and the device address is set to one, the JSI responds with a ">" when it understands a command and a "?" when it doesn't. Both responses are preceded with a line feed, carriage return sequence. If in Interactive Mode, the JSI will transmit a "*READY" and a ">" when energized or when a Reset (**Z**) command is executed.

If you try to define a loop command, you will not get back a ">" until you finish defining the loop.

SSI command disables the interactive mode. You will not receive ">" or "?" from the JSI.

SEE ALSO: **SS, SSA**

SSJ

NAME: Enable/Disable Continuous Scan Mode
SYNTAX: <a>**SSJ**b
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: **SSJ** enabled: The JSI continuously scans the inputs designated as sequence select inputs by the **IN** command and executes the sequence represented by the BCD number presented on the inputs. If Interrupted Run Mode (**XG**) is active, then all the sequence input lines must go inactive prior to scanning the next sequence. A stop command discontinues continuous sequence scanning.

SSJ disabled: Does not scan the BCD numbers for sequence execution. In this mode, you could execute sequences using the RS-232C interface.

SEE ALSO: **XR, XD, XT, IN, INL, XG, SS**

SSL

NAME: Resume Execution Enable
SYNTAX: <a>SSLb
DEFAULT: b = 0
RANGE: b = 0, 1
VALID:
RESPONSE: None
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command will enable the controller to stop execution of commands, then resume execution using the Continue (C) command.

SSL1 Resume execution of the sequence or commands in the buffer when Continue (C) command is entered
SSL0 Disable resume feature

You can stop a program or move using the Stop (S) or Remote Stop input. If **SSL1** is enabled, the move will resume as soon as you enter the Continue (C) command.

SEE ALSO: S, C
EXAMPLE:

<u>Command</u>	<u>Description</u>
> SSL1	Enable resume function
> XE1	Erase sequence 1
> XD1	Define sequence 1
> A1	Set acceleration to 1 rps ²
> V1	Set velocity to 1 rps
> D20000000	Set distance to 200,000 steps
> G	Execute the move (Go)
> T2	Wait 2 seconds
> G	Execute the move (Go)
> XT	End sequence definition
> XR1	Run sequence 1

While the motor is moving, enter a Stop (S) command. The motor will come to a stop. Now enter Continue (C) command, the motor would pick up where it stopped and complete the moves.

ST

NAME: Shutdown
SYNTAX: <a>STn
DEFAULT: n = 0
RANGE: n = 0, 1
VALID:
RESPONSE: None
TYPE: Programming
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: The Shutdown (**ST1**) command disables the JSI command OUT (voltage or current) output. The system ignores move commands that you issue after the **ST1** command.

The **ST0** command enables the command OUT output. Once you enable the command OUT, you can execute moves.

This command allows you to manually position the load. The absolute position counter is set to zero when you enter an **ST0** command.

SEE ALSO: OFF, ON
EXAMPLES:

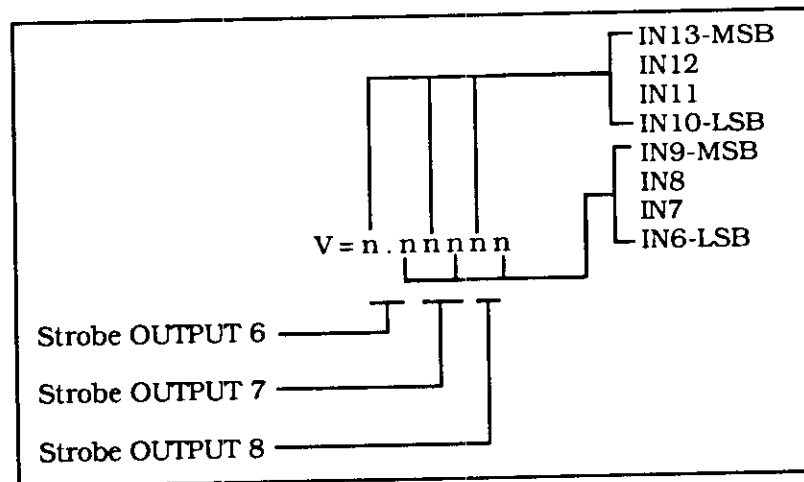
<u>Command</u>	<u>Description</u>
> ST1	Turns off the JSI command OUT output

STR

NAME: Set Strobe Output Delay Time
SYNTAX: <a>STRn
DEFAULT: n = 1000 msec
RANGE: n = 5000
VALID:
RESPONSE: aSTR *STROBE_TIME_n_MILLISECONDS[cr]
TYPE: Setup
ATTRIBUTES: Buffered, Device Address Optional, Savable in Sequence
DESCRIPTION: This command defines the amount of time each strobe output (defined by OUT) stays active. This delay and strobe outputs are used when loading parallel BCD data via remote inputs. The data transferred from the remote inputs are, Velocity (VRD), Distance (DRD), Loop Counts (LRD), and Sequence Numbers (XRD)

The STR command would typically be greater than a PLC scan time (to ensure that the data is present during a read), if used with a PLC or would set to a minimal debounce time if using thumbwheel switches. The strobe output will indicate that the JSI is ready for parallel input.

The data read by the inputs goes from the least significant digit to the most significant digit. The lowest input numbers are the least significant bit of the lower digit. For example, if we specify inputs 6 through 13 as the inputs, the inputs would automatically receive the following values during a VRD operation:



If the data VALID input is used, then when the strobe delay times out, the output strobe will remain active until the data valid input becomes active.

When you strobe the output, the first set of data read by inputs 6 through 13 will become the least significant digits. The next time the output is STROBED, the previously read data will be shifted 2 digits to the left, and the new value read will become the least significant digits. This process continues until you turned on and off the number of strobe outputs you have configured (maximum of 5 strobe outputs available).

If you are using a thumbwheel to set the data, or using a PLC, be sure to have a diode connected backwards between the inputs and the logic ground to prevent current from flowing. This will prevent the JSI Controller from reading false information.

SEE ALSO:
EXAMPLE:

VRD, LRD, DRD, XRD, IN, OUT, Application Design Section

Command	Description
OUT6J	Set output 6 as strobe output
OUT7J	Set output 7 as strobe output
OUT8J	Set output 8 as strobe output
IN6N	Set input 6 as least significant bit data input
IN7N	Set input 7 as 2nd least significant bit data input
IN8N	Set input 8 as 3rd least significant bit data input
IN9N	Set input 9 as 4th least significant bit data input
IN10N	Set input 10 as 5th least significant bit data input
IN11N	Set input 11 as 6th least significant bit data input
IN12N	Set input 12 as 7th least significant bit data input
IN13N	Set input 13 as 8th least significant bit data input
STR500	Set strobe output delay time to 500 mseconds
VRD	Read parallel inputs as velocity data

The example command would do the following as soon as the controller sees the **VRD** command.

- Step 1 JSI would turn on Output 6 for 500 milliseconds
- Step 2 Then it would read the inputs 6 through 13 for the two digits
- Step 3 JSI would then turn off Output 6 and turn on Output 7 for 500 milliseconds
- Step 4 Then it would read two more digits
- Step 5 JSI would then turn off Output 7 and turn on output 8 for 500 milliseconds
- Step 6 Then the JSI would read two more digits.
- Step 7 Output 8 would then turn off

