

S Stop Motion

Type	Motion	Product	Rev
Syntax	<!>S	6K	5.0
Units	n/a		
Range	b = 0 (do not stop) or 1 (stop)		
Default	1		
Response	>!S: No response, instead motion is stopped on all axes.		
See Also	C, COMEXC, COMEXS, GO, K		

The Stop Motion (S) command instructs the motor to stop motion on the specified axes. If the Stop (S) command is used without any arguments, motion will be stopped on all axes. The Stop command will bring the specified axes to rest using the last deceleration value (AD) entered.

NOTE

Since all commands are buffered, the next command does not begin until the previous command has finished. This is important because if you place a Stop (S) command after a Go (GO) command in a program, the Stop command will have no effect. For the Stop command to have an effect within a program, continuous command processing mode (COMEXC) must be enabled. If the Stop (S) command is to be used external to the program, the immediate command identifier (!) must be used.

If COMEXS is set to zero, command processing will be terminated when any stop command is issued, or a stop input is activated. If COMEXS is set to 1 or 2, a stop command issued for a specific axis will only stop motion on that axis and will not clear the command buffer. If COMEXS is set to 2, a stop command or input will stop motion and clear the command buffer.

If motion is to be paused and later resumed, the stop command must be used without any arguments (S or !S), and the continue execution on stop (COMEXS) command must be enabled. The continue (!C) command can then be used to resume motion.

Example:

```
GO1111 ; Initiate motion on all axes
!S1100 ; Stop motion on axes 1 and 2 (must use "!S" to stop motion in progress)
```

SCALE Enable/Disable Scale Factors

Type	Scaling	Product	Rev
Syntax	<!>SCALE	6K	5.0
Units	n/a		
Range	b = 0 (disable) or 1 (enable)		
Default	0		
Response	SCALE: *SCALE0		
See Also	DRES, ERES, SCLA, SCLD, SCLMAS, SCLV, SFB, TSTAT		

Scaling allows you to program acceleration, deceleration, velocity, and position values in units of measure that are appropriate for your application. The SCALE command is used to enable or disable scaling (SCALE1 to enable, SCALE0 to disable). When scaling is enabled (SCALE1), all entered data is multiplied by the appropriate scale factor:

Type of Motion	Accel/Decel Scaling	Velocity Scaling	Distance Scaling
Standard Point-to-Point Motion	SCLA	SCLV	SCLD
Contouring, Linear Interpolation	SCLD	SCLD	SCLD
Following	SCLA	SCLV	SCLD for follower distances SCLMAS for master distances

NOTE: Contouring uses only the SCLD value to scale all motion parameters; SCLA & SCLV are not applicable.

When Should I Define Scaling Factors?

Scaling calculations are performed when a program is defined or downloaded. Consequently, you must enable scaling (SCALE1) and define the scaling factors (SCLD, SCLA, SCLV, SCLMAS) *prior* to defining (DEF), uploading (TPROG), or running (RUN or PRUN) the program.

RECOMMENDATION: Place the scaling commands at the beginning of your program file, *before* the location of any defined programs. This ensures that the motion parameters in subsequent programs in your program file are scaled correctly. When you use Motion Planner’s Setup Generator wizard, the scaling commands are automatically placed in the appropriate location in your program file.

ALTERNATIVE: Scaling factors could be defined via a terminal emulator *just before* defining or downloading a program. Because scaling command values are saved in battery-backed RAM (remembered until you issue a RESET command), all subsequent program definitions and downloads will be scaled correctly.

RESTRICTIONS: Scaling commands are not allowed in a program. If there are scaling commands in a program, the controller will report an error message (“COMMAND NOT ALLOWED IN PROGRAM”) when the program is downloaded. If you intend to upload a program with scaled motion parameters, be sure to use Motion Planner. Motion Planner automatically uploads the scaling parameters and places them at the beginning of the program file containing the uploaded program from the controller. This assures correct scaling when the program file is later downloaded.

Servo Products

Scaling can be used with encoder or analog input feedback sources. When the scaling commands (SCLA, SCLD, etc.) are executed, they are specific only to the current feedback source selected with the last SFB command.

If your application requires switching between feedback sources for the same axis, then for each feedback source, you must select the feedback source with the appropriate SFB command and issue the scaling factors specific to operating with that feedback source.

For example, if you have two axes and will be switching between encoder and ANI feedback, you should include code similar to the following in your setup program:

```

SFB1,1      ; Select encoder feedback (subsequent scaling
            ; parameters are specific to encoder feedback)
SCLA4000,4000 ; Program accel/decel in revs/sec/sec
SCLV4000,4000 ; Program velocity in revs/sec
SCLD4000,4000 ; Program distances in revs
SFB2,2      ; Select ANI feedback (subsequent scaling
            ; parameters are specific to ANI feedback)
SCLA205,205  ; Program accel/decel in volts/sec/sec
SCLV205,205  ; Program velocity in volts/sec
SCLD205,205  ; Program distances in volts
    
```

Units of Measure without Scaling (Scaling is disabled (SCALE0) as the factory default condition):

- Stepper axes: All distance values entered are in commanded counts (sometimes referred to as *motor steps*), and all acceleration, deceleration and velocity values entered are internally multiplied by the DRES command value.

- Servo axes:

Motion Attribute	Units of Measure (per feedback source)	
	Encoder	Analog Input
Accel/Decel	Revs/sec/sec *	volts/sec/sec
Velocity	Revs/sec *	volts/sec
Distance	Counts **	Counts **

* All accel/decel & velocity values are internally multiplied by the ERES command value.

** Distance is measured in the counts received from the feedback device.

Contouring & Linear Interpolated Motion: Path acceleration, velocity, and distance are based on the resolution (DRES for steppers, ERES for servos) of axis 1. If multi-tasking is used, path motion units are based on the resolution of the first (lowest number) axis associated with the task (TSKAX).

SCALING EXAMPLES: Refer to page 16.

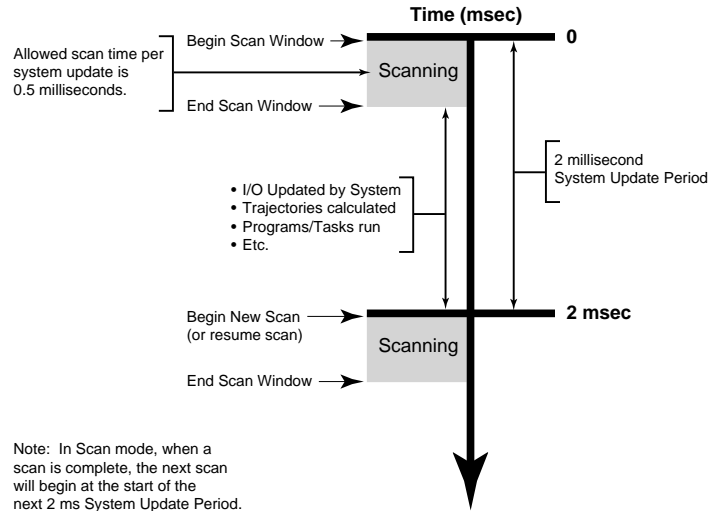
SCANP Scan Compiled PLCP Program

Type	PLC Scan Program	Product	Rev
Syntax	<!>SCANP<t>	6K	5.0
Units	t = text (name of the PLCP program, or CLR)		
Range	t = PLCPi, where i is the number of the desired PLCP program, or t = CLR (to clear or stop the scan function)		
Default	n/a		
Response	n/a		
See Also	PLCP, PCOMP, PRUN, PUCOMP, TSCAN		

Use the SCANP command to initiate scanning a specific compiled PLCP program (PLCPi). For example, SCANP PLCP3 initiates scanning the program defined as PLCP3 (defined with DEF PLCP3) and compiled (PCOMP PLCP3).

The PLCP program is scanned once per 2 ms system update period. Each scan pass is allotted a 0.5 ms window in the 2 ms system update period in which to complete the scan (refer to the diagram on the right). If the scan takes more than 0.5 ms, the scan will pause and continue where it stopped during the next 2-ms system update period. Conversely, if the scan takes less than 0.5 ms, the remaining processing time is used for normal processing.

To check how much time (in 2 ms increments) the last scan took to complete, issue the TSCAN command. For example, if the last PLCP program



took 3 system updates (2 ms each) to scan, then TSCAN would report *TSCAN6, indicating that it took 6 ms to complete the scan.

Launching programs external to the scan: Using the EXE command or the PEXE command, a scan program can launch another program in a specified task. EXE launches a standard, non-compiled program; PEXE launches a compiled program.

Stopping the scan: The scan program can be stopped in either of two ways: using the !K command, or clearing the scan program by issuing a SCANP CLR command.

Timing the PLCP program outputs: It is not possible to control where the PLCP program will pause if the scan takes more than the allowed time. This means that there can be a time lag of several update periods before the outputs, analog outputs, and/or variables affected by the PLCP program are updated. The order in which the scan takes place should be considered when creating PLCP programs to minimize the effects of such a lag. One way to avoid the lag is to create a binary variable as a temporary holding place for the desired output states. The last commands before the END statement of the PLCP program can set the outputs according to the final status of the variable, such that all output states are written at the same time, just as the scan completes. This method is demonstrated in the example program below.

For more information on defining a PLC program, refer to the PLCP command description.

Example:

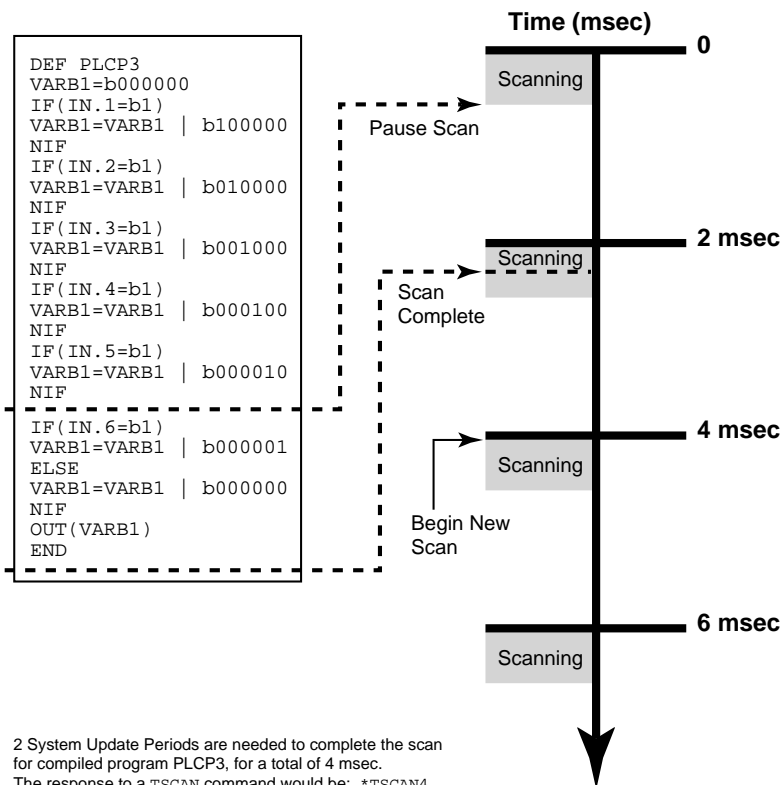
```

DEF PLCP3
; Binary states of outputs 1-6 are represented by VARB1 bit 1-6.
; Outputs 1-6 are set at the end of the program.
VARB1=b000000      ; Initialize binary variable 1
IF(IN.1=b1)        ; If Input 1 is ON, turn Output 1 ON
VARB1=VARB1 | b100000 ; Set binary bit for output 1 only to ON
NIF
IF(IN.2=b1)        ; If Input 2 is ON, turn Output 2 ON
VARB1=VARB1 | b010000 ; Set binary bit for output 2 only to ON
NIF
IF(IN.3=b1)        ; If Input 3 is ON, turn Output 3 ON
VARB1=VARB1 | b001000 ; Set binary bit for output 3 only to ON
NIF
IF(IN.4=b1)        ; If Input 4 is ON, turn Output 4 ON
VARB1=VARB1 | b000100 ; Set binary bit for output 4 only to ON
NIF
IF(IN.5=b1)        ; If Input 5 is ON, turn Output 5 ON
VARB1=VARB1 | b000010 ; Set binary bit for output 5 only to ON
NIF
IF(IN.6=b1)        ; If Input 6 is ON, turn Output 6 ON
VARB1=VARB1 | b000001 ; Set binary bit for output 6 only to ON
NIF
OUT(VARB1)         ; Turn on appropriate outputs
END

PCOMP PLCP3        ; Compile program PLCP3

SCANP PLCP3        ; Run compiled program PLCP3 in Scan mode
; The diagram below illustrates the scan.

```



SCLA Acceleration Scale Factor

Type	Scaling	Product	Rev
Syntax	<!><@><a>SCLA<i>, <i>, <i>, <i>, <i>, <i>, <i>, <i>	6K	5.0
Units	i = counts/unit		
Range	1 - 999,999		
Default	4000 (Servos auto-detect based on SFB: 4000 if encoder, 205 if ANI)		
Response	SCLA: *SCLA4000,4000,4000,4000,4000,4000,4000,4000 !SCLA: *!SCLA4000		
See Also	ANIRNG, FMAXA, SCALE, SCLD, SCLMAS, SCLV, SFB, TSTAT		

When scaling is enabled (SCALE1), all point-to-point acceleration values (A, AA, HOMA, HOMAA, JOGA, JOGAA, JOYA, JOYAA) and deceleration values (AD, ADA, LHAD, LHADA, LSAD, LSADA, HOMAD, HOMADA, JOGAD, JOGADA, JOYAD, JOYADA) are multiplied by the Acceleration Scale Factor (SCLA) command. Since the units are counts/unit, and all the acceleration values are in units/sec/sec, all accelerations will thus be internally represented as counts/sec/sec.

Stepper axes: If scaling is enabled (SCALE1), the entered accel and decel values are internally multiplied by the acceleration scaling factor (SCLA) to convert user units/sec/sec to commanded counts/sec/sec (sometimes referred to as “motor steps”/sec/sec). The entered values are always in reference to commanded counts, regardless of the existence of an encoder.

If scaling is disabled (SCALE0), all accel and decel values are entered in commanded revs/sec/sec; these values are internally multiplied by the drive resolution (DRES) value to obtain accel and decel values in commanded counts/sec/sec for the motion trajectory calculations.

Servo axes: If scaling is enabled (SCALE1), the entered accel and decel values are internally multiplied by the acceleration scaling factor (SCLA) to convert user units/sec/sec to encoder or ANI counts/sec/sec.

If scaling is disabled (SCALE0), all accel and decel values are entered in encoder revs/sec/sec or ANI volts/sec/sec; encoder values are internally multiplied by the encoder resolution (ERES) value to obtain accel and decel values in counts/sec/sec for the motion trajectory calculations.

As the acceleration scaling factor (SCLA) changes, the resolution of the acceleration and deceleration values and the number of positions to the right of the decimal point also change (see table at right). An acceleration value with greater resolution than allowed will be truncated. For example, if scaling is set to SCLA10, the A9.9999 command would be truncated to A9.9.

SCLA (counts/unit)	Decimal Places
1 - 9	0
10 - 99	1
100 - 999	2
1000 - 9999	3
10000 - 99999	4
100000 - 999999	5

The following equations can help you determine the range of acceleration and deceleration values.

Axis Type	Min. Accel or Decel (resolution)	Max. Accel or Decel
Stepper	$\frac{0.001 * DRES}{SCLA}$	$\frac{999.9999 * DRES}{SCLA}$
Servo	Encoder Feedback: $\frac{0.001 * ERES}{SCLA}$	Encoder Feedback: $\frac{999.9999 * ERES}{SCLA}$
	ANI Feedback: * $\frac{0.8205}{SCLA}$	ANI Feedback: * $\frac{20479.9795}{SCLA}$

* This calculation assumes the analog input range (ANIRNG value) is left in its default setting (range is -10V to +10V).

MORE ABOUT SCALING

For additional details on scaling, including scaling examples, refer to page 16.

SCLD Distance Scale Factor

Type	Scaling	Product	Rev
Syntax	<!><@><a>SCLD<i>,<i>,<i>,<i>,<i>,<i>,<i>,<i>	6K	5.0
Units	i = counts/unit		
Range	1 - 999,999		
Default	1		
	(Servos auto-detect based on SFB: 1 if encoder, 205 if ANI)		
Response	SCLD: *SCLD1,1,1,1,1,1,1,1 !SCLD: *!SCLD1		
See Also	[ANI], D, [FB], FOLRN, FSHFD, [PANI], [PC], [PCC], [PCE], [PCMS], [PE], [PER], PSET, [PSHF], [PSLV], REG, REGLD, SCALE, SCLA, SCLV, SCLMAS, SFB, SMPER, TANI, TFB, TPANI, TPC, TPCC, TPCE, TPCMS, TPE, TPER, TPSHF, TPSLV, TSTAT		

If scaling is enabled (SCALE1), all D, PSET, SMPER, and REG command values are internally multiplied by the Distance Scale Factor (SCLD) value. Since the SCLD units are in terms of counts/unit, all distances will thus be internally represented in counts. For instance, if your distance scaling factor is 10000 (SCLD10000) and you enter a distance of 75 (D75), the actual distance moved will be 750,000 (10000 x 75) counts.

This command is useful for allowing the user to specify distances in any unit. For example, if the user had a 25000 step/revolution drive and wanted distance units in terms of revolutions, then SCLD should be set to 25000, and scaling should be enabled (SCALE1).

As the distance scaling factor (SCLD) changes, the resolution of all distance commands and the number of positions to the right of the decimal point also change (see table below). A distance value with greater resolution than allowed will be truncated (e.g., if scaling is set to SCLD25000, the D1.99999 command would be truncated to D1.9999).

SCLD (counts/unit)	Distance Resolution (units)	Distance Range (units)	Decimal Places
1 - 9	1	0 - ±999,999,999	0
10 - 99	0.1	0.0 - ±99,999,999.9	1
100 - 999	0.01	0.00 - ±9,999,999.99	2
1000 - 9999	0.001	0.000 - ±999,999.999	3
10000 - 99999	0.0001	0.0000 - ±99,999.9999	4
100000 - 999999	0.00001	0.00000 - ±9999.99999	5

FRACTIONAL STEP TRUNCATION

If you are operating in the preset positioning mode (MC0), when the distance scaling factor (SCLD) and the distance value are multiplied, a fraction of one step may possibly be left over. This fraction is truncated when the distance value is used in the move algorithm. This truncation error can accumulate over a period of time, when performing incremental moves continuously in the same direction. To eliminate this truncation problem, set the distance scale factor (SCLD) to 1, or a multiple of 10.

MORE ABOUT SCALING

For additional details on scaling, including scaling examples, refer to page 16.

SCLMAS Master Scale Factor

Type	Following; Scaling	Product	Rev
Syntax	<!><@><a>SCLMAS<i>,<i>,<i>,<i>,<i>,<i>,<i>,<i>	6K	5.0
Units	i = scaling factor		
Range	i = 1 - 999999		
Default	1		
Response	SCLMAS *SCLMAS1,1,1,1,1,1,1,1 1SCLMAS *1SCLMAS1		
See Also	FMCLEN, FMCP, FOLEN, FOLMD, FOLRD, FOLRN, GOWHEN, [PCMS], [PMAS], SCALE, SCLD, TPMAS, TPCMS		

The Master Scale Factor (SCLMAS) command internally multiplies all Following master values by the specified scale factor value. Since the SCLMAS units are in terms of counts/unit, all distances will thus be internally represented in counts. For instance, if your master scaling factor is 10000 (SCLMAS10000) and you enter a master parameter of 75 (e.g., FOLMD75), the internal value will be 750,000 (10000 x 75) counts.

NOTE: The SCLMAS command will not take effect unless scaling is enabled (SCALE1).

This command allows you to specify distances in any unit. For example, if you had a 4000 step/revolution encoder as the master and wanted master units in terms of revolutions, then SCLMAS should be set to 4000.

As the master scaling factor (SCLMAS) changes, the resolution of all master parameter values and the number of positions to the right of the decimal point also change (see table below). A master parameter value with greater resolution than allowed will be truncated (e.g., if scaling is set to SCLD4000, the FOLMD1.9999 command would be truncated to FOLMD1.999).

SCLMAS (counts/unit)	Master Resolution (units)	Master Range (units)	Decimal Places
1 - 9	1	0 - ±999,999,999	0
10 - 99	0.1	0.0 - ±99,999,999.9	1
100 - 999	0.01	0.00 - ±9,999,999.99	2
1000 - 9999	0.001	0.000 - ±999,999.999	3
10000 - 99999	0.0001	0.0000 - ±99,999.9999	4
100000 - 999999	0.00001	0.00000 - ±9999.99999	5

FRACTIONAL STEP TRUNCATION

If you are specifying master distance values (FOLMD), when the master scaling factor (SCLMAS) and the distance value are multiplied, a fraction of one count may possibly be left over. This fraction is truncated when the distance value is used in the move algorithm. This truncation error can accumulate when performing several moves over the specified master distance. To eliminate this truncation problem, set the master scale factor (SCLMAS) to 1, or a multiple of 10.

Example: (refer also to the FOLEN examples, and page 16)

The commands below are a subset of the set-up parameters for an application in which axis 1 is following the encoder input on axis #3 at a 1-to-1 ratio.

```
SCALE1          ; Enable parameter scaling
SCLA25000       ; Set follower acceleration scale factor to 25000 for axis 1
SCLV25000       ; Set follower velocity scale factor to 25000 for axis 1
SCLD25000       ; Set follower distance scale factor to 25000 for axis 1
SCLMAS4000      ; Set master scale factor to 4000 for axis 1
FOLMAS31        ; Axis 1 using encoder input #3 as master
FOLRN1          ; Set follower-to-master Following ratio numerator to 1
                ; (scaled by SCLD)
FOLRD1          ; Set follower-to-master Following ratio denominator to 1.
                ; This sets the ratio to 1:1 (scaled the SCLMAS).
                ; The actual ratio in counts = 25000 to 4000 = 6.25 follower
                ; axis counts per master count.
```

SCLV Velocity Scale Factor

Type	Scaling	Product	Rev
Syntax	<!><@><a>SCLV<i>, <i>, <i>, <i>, <i>, <i>, <i>	6K	5.0
Units	i = counts/unit		
Range	1 - 999,999		
Default	4000 (Servos auto-detect based on SFB: 4000 if encoder, 205 if ANI)		
Response	SCLV: *SCLV4000,4000,4000,4000 ... !SCLV: *!SCLV4000		
See Also	ANIRNG, FMAXV, HOMV, HOMVF, JOGVH, JOGVL, JOYVH, JOYVL, SCALE, SCLA, SCLD, SFB, TSTAT, V		

When scaling is enabled (SCALE1), all velocity values (HOMV, HOMVF, JOGVH, JOGVL, JOYVH, JOYVL, V) are multiplied by the Velocity Scale Factor (SCLV) command. Since the units are counts/unit, all velocities will thus be internally represented in counts/sec.

Steppers: If scaling is enabled (SCALE1), the entered velocity values are internally multiplied by SCLV to convert user units/sec to commanded counts/sec.

If scaling is disabled (SCALE0), all velocity values are entered in commanded revs/sec; these values are internally multiplied by the drive resolution (DRES) value to obtain velocity values in commanded counts/sec for the motion trajectory calculations.

Servos: If scaling is enabled (SCALE1), the entered velocity values are internally multiplied by SCLV to convert user units/sec to encoder or ANI counts/sec.

If scaling is disabled (SCALE0), all velocity values are entered in encoder revs/sec or ANI volts/sec; encoder values are internally multiplied by the encoder resolution (ERES) value to obtain velocity values in counts/sec for the motion trajectory calculations.

As the velocity scaling factor (SCLV) changes, the resolution of the velocity commands and the number of positions to the right of the decimal point also change (see table below). A velocity value with greater resolution than allowed will be truncated. For example, if scaling is set to SCLV10, the V1.9999 command would be truncated to V1.9.

SCLV (steps/unit)	Velocity Resolution (units/sec)	Decimal Places
1 - 9	1	0
10 - 99	0.1	1
100 - 999	0.01	2
1000 - 9999	0.001	3
10000 - 99999	0.0001	4
100000 - 999999	0.00001	5

Use the following equations to determine the maximum velocity range for your product type.

Max. Velocity for Stepper Axes		Max. Velocity for Servo Axes (Servos: determined by feedback source selected for axis #1)	
$\frac{n}{SCLV}$	n = maximum velocity is determined by the PULSE command setting.	Encoder Feedback:	$\frac{6,500,000}{SCLV}$
		ANI Feedback: *	$\frac{1000 * 205}{SCLV}$

* This calculation assumes the analog input range (ANIRNG value) is left in its default setting (range is -10V to +10V).

MORE ABOUT SCALING

For additional details on scaling, including scaling examples, refer to page 16.

[SEG]

Number of Free Segment Buffers

Type	Compiled Motion; Assignment or Comparison	Product	Rev
Syntax	See below	6K	5.0
Units	n/a		
Range	n/a		
Default	n/a		
Response	n/a		
See Also	MEMORY, [SS], TDIR, TMEM, TSEG, TSS		

Use the SEG operator to assign the number of free memory segment buffers in *compiled memory* to a variable (VAR), or to make a comparison against another value. “Compiled memory” is the partition of the 6K controller’s non-volatile memory that stores compiled profiles & PLC programs. Compiled profiles/programs could be a multi-axis *contour* (a series of arcs and lines), an *individual axis profile* (a series of GOBUF commands), a *compound profile* (combination of multi-axis contours and individual axis profiles), or a *PLC program* (for PLC Scan Mode).

System status bit (see TSSF, TSS, and SS) 29 to set when the compiled memory is 75% full, and bit 30 is set if the compiled memory is 100% full.

Syntax: VARn=SEG where “n” is the variable number,
or SEG can be used in an expression such as IF (SEG=1)

SFB

Select Servo Feedback Source

Type	Controller Configuration or Servo	Product	Rev
Syntax	<@><a>SFB<i>, <i>, <i>, <i>, <i>, <i>, <i>	6K	5.0
Units	i = feedback source identifier		
Range	i = 0 (open loop, disable gains), 1 (encoder), or 2 (ANI input)	(applicable to servo axes only)	
Default	1		
Response	SFB *SFB1,1,1,1,1,1,1,1 1SFB *1SFB1		
See Also	[ANI], ANIFB, ANIRNG, AXSDEF, ERES, [FB], OUTPn, [PANI], [PCE], [PE], PSET, SCALE, SCLD, SGAF, SGI, SGILIM, SGP, SGV, SGVF, SMPER, SOFFS, TANI, TFB, TPANI, TPE		

Use the SFB command to select the servo feedback source to be used by each axis. The options are:

Options	Physical Location	Measurement*	Resolution Command
1—Encoder	ENCODER connector only	Encoder counts	ERES (default is 4000 counts/rev)
2—Analog (“ANI”) input **	Analog input SIM on external I/O brick	ADC counts	ANIRNG (default is 205 counts/volt)

* With scaling enabled (SCALE1), encoder and ANI feedback is scaled by the SCLD value.

** Before an analog input can be selected for feedback, it must be configured with the ANIFB command.

NOTE

Parameters for scaling (SCLA, SCLD, etc.), tuning gains (SGI, SGP, etc.), position offset (PSET) and maximum position error (SMPER) are specific to the feedback source currently selected with the last SFB command.

If your application requires switching between feedback sources for the same axis, then for each feedback source, you must issue the SFB command and then enter the scaling, gains, PSET and SMPER commands specific to that feedback source.

The feedback source can be changed only if motion is not in progress. When the feedback source is changed, the new setpoint will be determined by taking the new feedback source's value and adding any existing position error. Changing the source will disable the Output On Position commands (OUTPn).

USING SFBØ

Setting the SFB command value to zero has these effects:

- **WARNING:** The end-of-travel limits are disabled. Make sure that it is safe to operate without end-of-travel limits before using SFBØ.
- Gain values (SGILIM, SGAF, SGI, SGP, etc.) set to zero (open-loop operation).
- SMPER value set to zero (position error is allowed to increase without causing a fault).
- Subsequent attempts to change gain values or SMPER will cause an error message ("NOT ALLOWED IF SFBØ")
- SOFFS set to zero, but allows subsequent servo offset changes to affect motion.
- Disables output-on-position (OUTPA - OUTPH) functions.
- Any subsequent changes to PSET, PSETCLR, SCLD, SCLA, SCLV, and SOFFS are lost when another feedback source is selected.

Recommendation: Use the Disable Drive On Kill more, enabled with the KDRIVE command, so that the controller will shut down the drive if a kill command (e.g., !K) is executed or if a kill input is activated. Keep in mind that shutting down the drive allows the load to freewheel if there is not brake installed.

Example (to be placed outside of a program, because of the scaling parameters):

```
DRIVE0          ; Disable (shutdown) axis #1
SFB1            ; Select encoder feedback for axis #1 (subsequent scaling,
                ; gains, and PSET are specific to encoder feedback operation)
ERES4000       ; Set encoder resolution
SCLA4000       ; Set scaling for programming acceleration in revs/sec/sec
SCLV4000       ; Set scaling for programming velocity in revs/sec
SCLD4000       ; Set scaling for programming distance in revs
SGP5           ; Set proportional feedback gain to 5
SGI1           ; Set integral feedback gain to 1
SGV1           ; Set velocity feedback gain to 1
PSET0          ; Set current position as absolute position zero
1ANIRNG.17=4   ; Select a voltage range of -10V to +10V for the 1st analog
                ; input channel in SIM slot 3 (I/O location 17) of I/O brick 1.
                ; This means the counting resolution will be 205 counts/volt.
ANIFB1-17      ; Select the 1st analog input channel in SIM slot 3
                ; (I/O location 17) of I/O brick 1 to be used as
                ; feedback for axis 1. ((required before the SFB command))
SFB2           ; Select ANI feedback for axis #1 (subsequent scaling, gains,
                ; and PSET are specific to ANI feedback operation)
SCLA205        ; Set scaling for programming acceleration in volts/sec/sec
SCLV205        ; Set scaling for programming velocity in volts/sec
SCLD205        ; Set scaling for programming distance in volts
SGP1           ; Set proportional feedback gain to 1
SGI0           ; Set integral feedback gain to zero
SGV.5          ; Set velocity feedback gain to 0.5
PSET0          ; Set current position as absolute position zero
SFB1           ; Select encoder feedback for axis #1
```

SGAF Acceleration Feedforward Gain

Type	Servo	Product	Rev
Syntax	<!><@><a>SGAF<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = microvolts/step/sec/sec		
Range	0.00000000 - 2800000.00000000		(applicable to servo axes only)
Default	0		
Response	SGAF: *SGAF0,0,0,0,0,0,0,0 1SGAF: *1SGAF0		
See Also	SFB, SGENB, SGI, SGP, SGSET, SGV, SGVF, TGAIN, TSGSET		

Use the Acceleration Feedforward Gain (SGAF) command to set the gain for the acceleration feedforward term in the servo control algorithm. Introducing acceleration feedforward control improves *position tracking performance* when the system is commanded to accelerate or decelerate.

The SGAF value is multiplied by the *commanded acceleration* (calculated by the 6K controller's DSP move profile routine) to produce the control signal.

Acceleration feedforward control can improve the performance of contouring and linear interpolation applications, as well as reduce the time required to reach the commanded velocity. *However, if your application only requires point-to-point moves, acceleration feedforward control is not necessary (leave the SGAF command setting at zero—default).*

Acceleration feedforward control does not affect the servo system's stability, nor does it have any effect at constant velocity or at steady state.

NOTE

The SGAF command is specific to the current feedback source (selected with the last SFB command). Therefore, if your application requires switching between feedback sources for the same axis, then for each feedback source, you must select the feedback source with the appropriate SFB command and then issue the SGAF command with the gain values specific to the selected feedback source.
--

For more information on servo tuning and how the acceleration feedforward gain affects performance, refer to your product's *Installation Guide* or to the *Servo Tuner User Guide*.

Example:

SGAF0.5555,43.554,0,0 ; Set the acceleration feedforward for axes 1 and 2

SGENB Enable a Servo Gain Set

Type	Servo	Product	Rev
Syntax	<!><@><a>SGENB<i>, <i>, <i>, <i>, <i>, <i>, <i>, <i>	6K	5.0
Units	i = gain set identification number (see SGSET command)		
Range	1 - 5		(applicable to servo axes only)
Default	n/a		
Response	n/a		
See Also	SFB, SGAF, SGAFN, SGI, SGILIM, SGP, SGSET, SGV, SGVF, SOFFS, TGAIN, TSGSET		

This command allows you to enable any combination of the five gain sets to any combination of axes. The gain sets are set with the SGSET command. A gain set can be enabled during motion at any specified point in the profile, or when not in motion. For example, you could use one set of gain parameters for the constant velocity portion of the profile, and when you approach the target position a different set of gains can be enabled.

NOTE

The tuning gains in a given gain set are specific to the feedback source that was in use (selected with the last SFB command) at the time the gains were established with the respective gain commands (SGI, SGP, etc.). Make sure that the gain set you enable is appropriate to the feedback source you are using at the time.
--

For more information on servo tuning, refer to your product's *Installation Guide* or to the Motion Planner help system.

Example:

```
SGP5,5,10,10 ; Sets the gains for the proportional gain
SGI.1,.1,0,0 ; Sets the gains for the integral gain
SGV50,60,0,0 ; Sets the gains for the velocity gain
SGVF5,6,10,11 ; Sets the gains for the velocity feedforward gain
SGAF0,0,0,0 ; Sets the gains for the acceleration feedforward gain
SGSET3 ; Assign SGP, SGI, SGV, SGVF, & SGAF gains to servo gain set 3
SGP75,75,40,40 ; Sets the gains for the proportional gain
SGI5,5,5,7 ; Sets the gains for the integral gain
SGV1,.45,2,2 ; Sets the gains for the velocity gain
SGVF0,8,0,9 ; Sets the gains for the velocity feedforward gain
SGAF18,20,22,24 ; Sets the gains for the acceleration feedforward gain
SGSET1 ; Assign SGP, SGI, SGV, SGAF, & SGVF gains to servo gain set 1
SGENB1,3,3,1 ; Enables gain set 1 gains on axis 1 &4; enables gain set 3 on
; axis 2 & 3
TGAIN ; Displays the current value for all gains. Example response:
; *SGP75,5,10,40
; *SGI5,.1,0,7
; *SGV1,60,0,2
; *SGVF0,6,10,9
; *SGAF18,0,0,24
```

SGI

Integral Feedback Gain

Type	Servo	Product	Rev
Syntax	<!><@><a>SGI<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = millivolts/step * sec		
Range	0.00000000-2,800,000.00000000		(applicable to servo axes only)
Default	0.0		
Response	SGI: *SGI0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0 1SGI: *1SGI0.0		
See Also	SFB, SGAF, SGENB, SGILIM, SGP, SGSET, SGV, SGVF, TGAIN, TSGSET		

Use the Integral Gain (SGI) command to set the gain of the integral term in the control algorithm. The primary function of the integral gain is to reduce or eliminate final position error (e.g., due to friction, gravity, etc.) and improve system accuracy during motion. If a position error exists (commanded position not equal to actual position—see TPER command), this control signal will ramp up until it is high enough to overcome the friction and drive the motor toward its commanded position. *If acceptable position accuracy is achieved with proportional gain (SGP), then the integral gain (SGI) need not be used.*

If the integral gain is set too high relative to the other gains, the system may become oscillatory or unstable. The integral gain can also cause excessive position overshoot and oscillation if an appreciable position error has persisted long enough during the transient period (time taken to reach the position setpoint); this effect can be reduced by using the SGILIM command to limit the integral term windup.

NOTE

The SGI command is specific to the current feedback source (selected with the last SFB command). Therefore, if your application requires switching between feedback sources for the same axis, then for each feedback source, you must select the feedback source with the appropriate SFB command and then issue the SGI command with the gain values specific to the selected feedback source.

For more information on servo tuning, refer to your product's *Installation Guide* or to the Motion Planner help system.

Example:

```
SGI15,14.5 ; Set the integral gain for axes 1 and 2
```

SGILIM Integral Windup Limit

Type	Servo	Product	Rev
Syntax	<!><@><a>SGILIM<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = volts		
Range	0-65,535		(applicable to servo axes only)
Default	200		
Response	SGILIM: *SGILIM200,200,200,200,200,200,200,200 1SGILIM: *1SGILIM200		
See Also	SFB, SGENB, SGI, TGAIN, TSGSET		

If integral control (SGI) is used and an appreciable position error has persisted long enough during the transient period (time taken to reach the setpoint), the control signal generated by the integral action can end up too high and saturate to the maximum level of the controller's analog control signal output. This phenomenon is called *integrator windup*.

After windup occurs, it will take a while before the integrator output returns to a level within the limit of the controller's output. Such a delay causes excessive position overshoot and oscillation. Therefore, the integral windup limit (SGILIM) command is provided for you to set the absolute limit of the integral and, in essence, turn off the integral action as soon as it reaches the limit; thus, position overshoot and oscillation can be reduced.

NOTE

The SGILIM command is specific to the current feedback source (selected with the last SFB command). Therefore, if your application requires switching between feedback sources for the same axis, then for each feedback source, you must select the feedback source with the appropriate SFB command and then issue the SGILIM command with the gain values specific to the selected feedback source.

For more information on servo tuning, refer to your product's *Installation Guide* or to the Motion Planner help system.

Example:

```
SGI44,43,55,0 ; Sets the integral gain term  
SGILIM15,15,15,15 ; Sets the integral windup limit on the integral gain term
```

SGP Proportional Feedback Gain

Type	Servo	Product	Rev
Syntax	<!><@><a>SGP<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = millivolts/step		
Range	0.00000000-2,800,000.00000000		(applicable to servo axes only)
Default	0.5		
Response	SGP: *SGP0.5,0.5,0.5,0.5 ... 1SGP: *1SGP0.5		
See Also	SFB, SGAF, SGENB, SGI, SGSET, SGV, SGVF, TGAIN, TSGSET		

This command allows you to set the gain of the proportional term in the servo control algorithm. The output of the proportional term is proportional to the difference between the commanded position and the actual position read from the feedback device. The primary function of the proportional term is to stabilize the system and speed up the response. It can also be used to reduce the steady state position error.

When the proportional gain (SGP) is used alone (i.e., the other gain terms are set to zero), setting this gain too high can cause the system to become oscillatory, underdamped, or even unstable.

NOTE

The SGP command is specific to the current feedback source (selected with the last SFB command). Therefore, if your application requires switching between feedback sources for the same axis, then for each feedback source, you must select the feedback source with the appropriate SFB command and then issue the SGP command with the gain values specific to the selected feedback source.

For more information on servo tuning, refer to your product's *Installation Guide* or to the Motion Planner help system.

Example:

```
SGP10,4.22233,2.22,.0445245 ; Sets the proportional gain of all axes
```

SGSET		Save a Servo Gain Set	
Type	Servo	Product	Rev
Syntax	<!>SGSET<i>	6K	5.0
Units	i = gain set identification number		
Range	1-5		(applicable to servo axes only)
Default	n/a		
Response	n/a		
See Also	SFB, SGAF, SGENB, SGI, SGILIM, SGP, SGV, SGVF, SOFFS, TGAIN, TSGSET		

This command allows you to save the presently assigned gain values (SGP, SGI, SGV, SGAF, and SGVF) as a set of gains. Stand-alone servo controllers save (into battery-backed RAM) the gains and the axes and feedback sources to which they are assigned. Up to 5 sets of gains can be saved. Any gain set can be displayed using the TSGSET command.

Any gain set can be enabled with the SGENB command during motion at any specified point in the profile, or when not in motion. For example, you could use one set of gain parameters for the constant velocity portion of the profile, and when you approach the target position a different set of gains can be enabled.

NOTE

The tuning gains in a given gain set are specific to the feedback source that was in use (selected with the last SFB command) at the time the gains were established with the respective gain commands (SGI, SGP, etc.). If your application requires you to switch between feedback sources for the same axis, make sure that the gain set you enable is appropriate to the feedback source you are using at the time.

For more information on servo tuning, refer to your product's *Installation Guide* or to the Motion Planner help system.

Example:

```
SGP5,5,10,10 ; Sets the gains for the proportional gain
SGI.1,.1,0,0 ; Sets the gains for the integral gain
SGV50,60,0,0 ; Sets the gains for the velocity gain
SGVF5,6,10,11 ; Sets the gains for the velocity feedforward gain
SGAF0,0,0,0 ; Sets the gains for the acceleration feedforward gain
SGSET3 ; Assign SGP, SGI, SGV, SGVF, & SGAF gains to servo gain set 3
SGP75,75,40,40 ; Sets the gains for the proportional gain
SGI5,5,5,7 ; Sets the gains for the integral gain
SGV1,.45,2,2 ; Sets the gains for the velocity gain
SGVF0,8,0,9 ; Sets the gains for the velocity feedforward gain
SGAF18,20,22,24 ; Sets the gains for the acceleration feedforward gain
SGSET1 ; Assign SGP, SGI, SGV, SGAF, & SGVF gains to servo gain set 1
SGENB1,3,3,1 ; Enables gain set 1 gains on axis 1 &4; enables gain set 3 on
; axis 2 & 3
TGAIN ; Displays the current value for all gains. Example response:
; *SGP75,5,10,40
; *SGI5,.1,0,7
; *SGV1,60,0,2
; *SGVF0,6,10,9
; *SGAF18,0,0,24
```

SGV

Velocity Feedback Gain

Type	Servo	Product	Rev
Syntax	<!><@><a>SGV<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = microvolts/step/sec		
Range	0.00000000-2,800,000.00000000		(applicable to servo axes only)
Default	0.0		
Response	SGV: *SGV0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0 1SGV: *1SGV0.0		
See Also	ERES, SFB, SGAF, SGI, SGP, SGVF, TGAIN, TSGSET		

This command allows you to control the velocity feedback gain in the servo algorithm. Using velocity feedback, the controller's output signal is made proportional to the velocity, or rate of change, of the feedback device position. Since it acts on the rate of change of the position, the action of this term is to anticipate position error and correct it before it becomes too large. This increases damping and tends to make the system more stable.

If this term is too large, the response will be slowed to the point that the system is over-damped. This gain can increase position tracking error, which can be countered by the velocity feedforward term (SGVF).

Since the feedback device signal has finite resolution, the velocity accuracy has a limit. Therefore, if the velocity feedback gain (SGV) is too high, the errors due to the finite resolution are magnified and a noisy, or *chattering*, response may be observed.

NOTE

The SGV command is specific to the current feedback source (selected with the last SFB command). Therefore, if your application requires switching between feedback sources for the same axis, then for each feedback source, you must select the feedback source with the appropriate SFB command and then issue the SGV command with the gain values specific to the selected feedback source.

For more information on servo tuning, refer to your product's *Installation Guide* or to the Motion Planner help system.

Example:

SGV100,97,43.334,0 ; Sets the velocity gain term for all the axes

SGVF

Velocity Feedforward Gain

Type	Servo	Product	Rev
Syntax	<!><@><a>SGVF<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = microvolts/step/sec		
Range	0.00000000-2,800,000.00000000		(applicable to servo axes only)
Default	0.0		
Response	SGVF: *SGVF0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0 1SGVF: *1SGVF0.0		
See Also	SFB, SGAF, SGENB, SGI, SGP, SGSET, SGV, TGAIN, TSGSET		

Use the Velocity Feedforward Gain (SGVF) command to set the velocity feedforward gain. Introducing velocity feedforward control improves *position tracking performance* when the system is commanded to move at constant velocity. The tracking error is mainly attributed to friction, torque load, and velocity feedback control (SGV).

The SGVF value is multiplied by the *commanded velocity* (calculated by the 6K controller's DSP move profile routine) to produce the control signal.

Velocity feedforward control can improve the performance of interpolation (linear and circular) application. *However, if your application only requires short, point-to-point moves, velocity feedforward control is not necessary (leave the SGVF command setting at zero—default).*

Because velocity feedforward control is not in the servo feedback loop, it does not affect the servo system's stability, nor does it have any effect at steady state. Therefore, the only limits on how high you can set the velocity feedforward gain (SGVF) are: when it *saturates the control output* (tries to exceed the servo

controller's $\pm 10V$ analog control signal range); or when it causes the actual position to *precede* the commanded position.

NOTE

The `SGVF` command is specific to the current feedback source (selected with the last `SFB` command). Therefore, if your application requires switching between feedback sources on the same axis, then for each feedback source, you must select the feedback source with the appropriate `SFB` command and then issue the `SGVF` command with the gain values specific to the selected feedback source.

For more information on servo tuning, refer to your product's *Installation Guide* or to the Motion Planner help system.

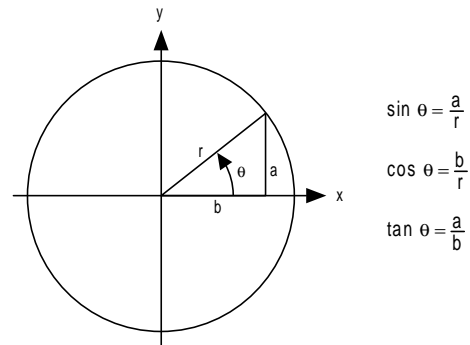
Example:

`SGVF3555,3555,4000,4000 ;` Sets the velocity feedforward for all axes

[SIN ()] Sine

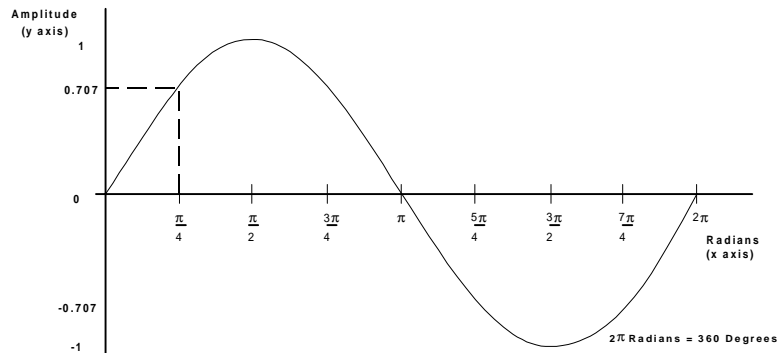
Type	Operator (Trigonometric)	Product	Rev
Syntax	... SIN(r) (See below)	6K	5.0
Units	r = value in radian or degrees based on RADIAN command		
Range	± 17500.0000000 radians		
Default	n/a		
Response	n/a		
See Also	[ATAN], [COS], [PI], RADIAN, [TAN], VAR		

This operator is used to calculate the sine of a number given in radians or degrees (see the `RADIAN` command). If "a" and "b" are coordinates of a point on a circle of radius "r", then the angle of measure "θ" can be defined by the equation: $\sin \theta = \frac{a}{r}$



If a value is given in radians and a conversion is needed to degrees, use the formula: $360^\circ = 2\pi$ radians.

The graph on the right shows the amplitude of **y** on the unit circle for different values of **x**.



Syntax: `VARx=SIN(r)` where "x" is the numeric variable number and "r" is a value provided in either degrees or radians based on the `RADIAN` command. Parentheses () must be placed around the `SIN` operand. The result will be specified to 5 decimal places.

Example:

`RADIAN1`
`VAR1=5 * SIN(PI/4) ;` Set variable 1 equal to 5 times the sine of Pi divided by 4

SINAMP Virtual Master Sine Wave Amplitude

Type	Following	Product	Rev
Syntax	<!><@><a>SINAMP<i>,<i>,<i>,<i>,<i>,<i>,<i>,<i>	6K	5.0
Units	i = amplitude		
Range	0-8192 (max. peak to peak is 16384)		
Default	0		
Response	SINAMP *SINAMP0,0,0,0,0,0,0,0 1SINAMP *1SINAMP0		
See Also	FOLMAS, FVMACC, FVMFRQ, SINANG, SINGO		

Use the SINAMP command to define the amplitude of the internal sine wave when it has been designated as the virtual master. By designating the internal sine wave as a master, the user may produce a sinusoidally oscillating motion, with control of the phase, amplitude, and center of oscillation.

The SINAMP command allows a change in follower amplitude without changing the center of oscillation. It affects the sine wave immediately, without any built in ramp in amplitude. If a gentle change in amplitude is desired, write a user program which repeatedly issues the command with small changes in value until the desired value is reached.

The peak-to-peak amplitude of a virtual master sine wave is twice the value specified with the SINAMP command.

SINANG Virtual Master Sine Wave Angle

Type	Following	Product	Rev
Syntax	<!><@><a>SINANG<r>,<r>,<r>,<r>,<r>,<r>,<r>,<r>	6K	5.0
Units	degrees		
Range	0.0-360.0		
Default	0		
Response	SINAMP *SINAMP0,0,0,0,0,0,0,0 1SINAMP *1SINAMP0		
See Also	FOLMAS, FVMACC, FVMFRQ, SINAMP, SINGO		

The SINANG command is used to define the phase angle when the internal sine wave is designated as the virtual master. By designating the internal sine wave as a master, the user may produce a sinusoidally oscillating motion, with control of the phase, amplitude, and center of oscillation.

There is one sine wave per axis, each using the variable count frequency (FVMFRQ) of that axis to increase or decrease the angle from which the sine is calculated. Each count of the count frequency changes the angle by one-tenth (0.1) of a degree. For example, a FVMFRQ value of 3600 would create an angular frequency of 3600 tenths of degrees per second, or 1 cycle per second. When used as a source for the sine wave, the maximum value for FVMFRQ is 144000. This results in a maximum of 40 Hz angular frequency. Frequencies higher than this are not allowed because they may be subject to aliasing.

SINGO Virtual Master - Initiate Internal Sine Wave

Type	Following	Product	Rev
Syntax	<!><@><a>SINGO	6K	5.0
Units	n/a		
Range	b = 1 (restart sine wave from previous angle & amplitude) or 0 (stop sine wave)		
Default	0		
Response	SINGO *SINGO0000_0000 1SINGO *1SINGO0		
See Also	FOLMAS, FVMACC, FVMFRQ, SINAMP, SINANG		

The SINGO command is used to restart the internal sine wave from zero degrees. By designating the internal sine wave as a master, the user may produce a sinusoidally oscillating motion, with control of the phase, amplitude, and center of oscillation.

The `SINGO` command with a “0” parameter abruptly stops the sine wave, without changing its current magnitude. Using the `SINGO` command with a “1” parameter abruptly starts the sine wave, also without changing its current magnitude. To gently pause the follower output, change the `FVMFRQ` value to zero with a moderate `FVMACC` value; to resume the follower output, restore the original `FVMFRQ` value.

The `SINGO` command with a “1” parameter always starts at the previous angle, which may not be the desired start of oscillation. The `SINANG` command will instantly change the angle and corresponding sine of the angle. This represents an abrupt change in master position. If the follower axis is still following when this occurs, there will be an abrupt change in commanded follower position. To start the follower properly, move the follower to the desired start position first (using `MC0`, `D`, `GO`), then issue `SINANG`, then `MC1`, `GO1`, and finally `SINGO`.

SMPER		Maximum Allowable Position Error	
Type	Servo	Product	Rev
Syntax	<!><@><a>SMPER<r>,<r>,<r>,<r>,<r>,<r>,<r>,<r>	6K	5.0
Units	r = feedback device steps (scalable with <code>SCLD</code>)		
Range	0-200,000,000 (0 = do not monitor position error condition)		(applicable to
Default	4000		servo axes only)
Response	SMPER: *SMPER4000,4000,4000,4000 ... 1SMPER: *1SMPER4000		
See Also	[AS], CMDDIR, ENCPOL, [ER], ERES, ERROR, ERRORP, SCALE, SCLD, SFB, SGILIM, TANI, TAS, TER, TFB, TPC, TPE, TPER		

This command allows you to set the maximum position error allowed before an error condition occurs. The position error, monitored once per system update period, is the difference between the commanded position and the actual position as read by the feedback device selected with the last `SFB` command. When the position error exceeds the value entered by the `SMPER` command, an error condition is latched (see `TAS` or `AS` bit #23) and the 6K controller issues a shutdown to the faulted axis and sets its analog output command to zero volts. To enable the system again, the `DRIVE1` command must be issued to the affected axis, which also sets the commanded position equal to the actual feedback device position (incremental devices will be zeroed).

If the `SMPER` value is set to zero (`SMPER0`), the position error condition is not monitored, allowing the position error to accumulate without causing a fault.

When `SMPER` is set to a non-zero value, the maximum position error acts as the servo system fault monitor; if the system becomes unstable or loses position feedback, the controller detects the resulting position error, shuts down the drive, and sets an error status bit. You can enable `ERROR` command bit #12 to continually check for the position error condition, and when it occurs to branch to a programmed response defined in the `ERRORP` program. You can check the status of this error condition with the `TAS`, `AS`, `TER`, and `ER` commands. You can check the actual position error with the `TPER` and `PER` commands.

If scaling is enabled (`SCALE1`), the `SMPER` value is multiplied by the `SCLD` value.

NOTE

The `SMPER` command is specific to the current feedback source (selected with the last `SFB` command). Therefore, if your application requires switching between feedback sources on the same axis, then for each feedback source, you must select the feedback source with the appropriate `SFB` command and then issue the `SMPER` command with the gain values specific to the selected feedback source.

Example:
`ERES4000,4000,4000,4000 ; Set encoder resolution for all axes to 4000 counts/rev`
`SMPER4000,4000,4000,4000 ; Set maximum allowable position error to 1 rev for`
`; all 4 axes. If the position error exceeds 4000 counts`
`; (1 rev) a fault condition will occur.`

SOFFS Servo Control Signal Offset

Type	Servo	Product	Rev
Syntax	<!><@><a>SOFFS<r>, <r>, <r>, <r>, <r>, <r>, <r>, <r>	6K	5.0
Units	r = volts		
Range	-10.000 to 10.000 (resolution is 0.001 volts)		
Default	0		
Response	SOFFS: *SOFFS0,0,0,0,0,0,0,0 1SOFFS: *1SOFFS0		
See Also	[DAC], DACLIM, SGENB, SGSET, TDAC, TGAIN, TSGSET		

This command allows you to set an offset voltage to the commanded analog control signal output (commanded analog output + SOFFS value = offset analog output). With this command, you can set an offset voltage to the drive system so that the motor will be stationary in an open-loop configuration. *This is the same effect as the balance input on most analog servo drives.*

CAUTION

If there is little or no load attached, the SOFFS offset may cause an acceleration to a high speed.

Typically, this offset will be set to zero. This offers a method for setting the analog output command to a known voltage. By setting the SGP, SGI, SGV, SGAF, & SGVF gains to zero, the analog output will reflect this offset value and the system becomes an open-loop configuration.

Use the TDAC command to check the voltage being commanded at the 6K controller's analog output (voltage displayed includes any offset in effect). An axis configured as a stepper can use the SOFFS command to set the DAC output voltage.

Example:

SOFFS0,0,1,2 ; Sets the offset voltage on all axes

[SQRT()] Square Root

Type	Operator (Mathematical)	Product	Rev
Syntax	See below	6K	5.0
Units	n/a		
Range	n/a		
Default	n/a		
Response	n/a		
See Also	[=], [+], [-], [*], [/], VAR		

This operator takes the square root of a value. The result, if multiplied by itself, will *approximately equal* the original value (the difference is attributed to round-off error). The resulting value has 3 decimal places.

Syntax: VARn=SQRT(expression) where “n” is the variable number, and the expression can be a number or a mathematical expression. The SQRT of a negative number is not allowed. Parentheses (()) must be placed around the SQRT operand.

Example:

VAR1=SQRT(25) ; Set variable 1 equal to the square root of 25 (result = 5)

[SS] System Status

Type	Assignment or Comparison	Product	Rev
Syntax	See below	6K	5.0
Units	n/a		
Range	n/a		
Default	n/a		
Response	n/a		
See Also	IF, TCMDER, TRGFN, [TRIG], TTRIG, TSS, TSSF, TSTAT, VARB		

Use the SS operator to assign the system status bits to a binary variable (VARB), or to make a comparison against a binary or hexadecimal value. To make a comparison against a binary value, the letter b (b or B)

must be placed in front of the value. The binary value itself must only contain ones, zeros, or Xs (1, 0, X, x). To make a comparison against a hexadecimal value, the letter h (h or H) must be placed in front of the value. The hexadecimal value itself must only contain the letters A through F, or the numbers 0 through 9.

Syntax: VARBn=<i%>SS where “n” is the binary variable number, or SS can be used in an expression such as IF(SS=b1101), or IF(SS=h7F). **NOTE:** If you are using multi-tasking, be aware that each task has its own system status register. If you wish to check the system status of an external task (a task other than the task that is executing the SS operator), then you must prefix the SS operator to address the targeted task (e.g., 2%SS for the system status of Task 2).

The function of each system status bit is shown below.

BIT (Left to Right)	Function (1 = yes, 0 = no)	BIT (Left to Right)	Function (1 = yes, 0 = no)
1	System Ready	17	Loading Thumbwheel Data (TW)
2	Reserved	18	External Program Select Mode (INSELP)
3	Executing a Program	19	Dwell in Progress (T command)
4	Immediate Command (set if last command was immediate)	20	Waiting for RP240 Data—DREAD or DREADF
5	In ASCII Mode	21	RP240 Connected — current PORT setting only
6	In Echo Mode — current PORT setting only	22	Non-volatile Memory Error
7	Defining a Program	23	Servo data gathering transmission in progress (servo axes only)
8	In Trace Mode	24	Reserved
9	In Step Mode	25	RESERVED
10	In Translation Mode	26	RESERVED
11	Command Error Occurred (bit is cleared when TCMDEP is issued)	26	RESERVED
12	Break Point Active (BP)	28	RESERVED
13	Pause Active	29	Compiled memory is 75% full
14	Wait Active (WAIT)	30	Compiled memory is 100% full
15	Monitoring On Condition (ONCOND)	31 *	Compile operation failed (PCOMP) **
16	Waiting for Data (READ)	32	Reserved

- * Bit #31: failed PCOMP compile is cleared on power up, RESET, or after successful compile. Possible causes include:
- Errors in profile design (e.g., change direction while at non-zero velocity; distance & velocity equate to < 1 count per system update; preset move profile ends in non-zero velocity)
 - Profile will cause a Following error (see TFSF, TFS, or FS command descriptions)
 - Out of memory (see SS bit #30)
 - Axis already in motion at the time of the PCOMP command
 - Loop programming errors (e.g., no matching PLOOP or PLN; more than 4 embedded PLOOP/END loops)
 - PLCP program contains invalid commands.

If it is desired to assign only one bit of the system status value to a binary variable, instead of all 32, the bit select (.) operator can be used. For example, VARB1=SS.12 assigns system status bit 12 to binary variable 1: *VARB1=XXXX_XXXX_XXX0_XXXX_XXXX_XXXX_XXXX_XXXX.

Example:

```

VARB1=SS ; System status assigned to binary variable 1
IF(SS=b111011x11) ; If the system status contains 1s in bit locations 1, 2, 3,
; 5, 6, 8, & 9, and a 0 in bit location 4, do the IF
; statement
IF(SS=h7F00) ; If the system status contains 1s in bit locations 1, 2, 3,
; 5, 6, 7, & 8, and 0s in every other bit location, do the IF
; statement
NIF ; End of second IF statement
NIF ; End of first IF statement

```

STARTP Start-Up Program

Type	Subroutines	Product	Rev
Syntax	<!>STARTP<t>	6K	5.0
Units	t = text (name of program)		
Range	Text name of 6 characters or less		
Default	n/a		
Response	STARTP: *STARTP MAIN		
See Also	DEF, RESET, SCALE		

The Start-Up Program (STARTP) command specifies the name of the program that will automatically when the 6K product is powered up or reset with the RESET command. If the program that is identified as the STARTP program is deleted with the DEL command, the STARTP is automatically cleared. If you wish to prevent the STARTP program from being executed, without having to delete the assigned program, issue the STARTP CLR command.

Example:

```
STARTP WakeUp ; Set program WakeUp as the program that will start to run
               ; after power is cycled or the 6K product is reset
STARTP CLR    ; Clears the program WakeUp from its assignment as the
               ; start-up program
DEL WakeUp    ; Deletes the program WakeUp and clears the STARTP command
               ; (no power-up program will be executed)
```

STEP Single Step Mode Enable

Type	Program Debug Tool	Product	Rev
Syntax	<!>STEP	6K	5.0
Units	n/a		
Range	b = 0 (disable), 1 (enable) or X (don't care)		
Default	0		
Response	STEP: *STEP0		
See Also	[#], BP, [SS], TRACE, TRACEP, TRANS, TSS		

The Single Step Mode Enable (STEP) command enables single command step mode. Single step mode is used for stepping through a defined (DEF) program. To execute single step mode:

1. Define a program (DEF)
2. Enable single step mode (STEP1)
3. Run the program (RUN)
4. Use the immediate pound (!#) to step through the program

Each step (!#) command will initiate the next command to be processed.

Example:

```
DEF tester    ; Begin definition of program named tester
V1,1,1,1     ; Set velocity to 1 unit/sec on all axes
A10,10,10,10 ; Set acceleration to 10 units/sec/sec on all axes
               ; (Note: This command will not be executed until a !# sign
               ; is received.)
D1,2,3,4     ; Set distance to 1 unit on axis 1, 2 units on axis 2,
               ; 3 units on axis 3, and 4 units on axis 4
GO1101      ; Initiate motion on axes 1, 2, and 4
OUT11X1     ; Turn on onboard outputs 1, 2, and 4, leave 3 unchanged
END         ; End program definition
STEP1      ; Enable single step mode
RUN tester  ; Execute program named tester
; *****
; * At this point no action will occur because single step mode *
; * has been enabled. Here's how to execute commands: *
; * !#2 (Execute 1st 2 commands in the program: V1,1,1,1 & A10,10,10,10) *
; * !# (Execute 1 command: command to be executed is D1,2,3,4) *
; * !#1 (Execute 1 command: command to be executed is GO110) *
; * !#2 (Execute 2 commands: commands to be executed are OUT11X1 & END) *
; *****
```

STRGTD Target Distance Zone

Type	Servo	Product	Rev
Syntax	<!><@><a>STRGTD<r>,<r>,<r>,<r>,<r>,<r>,<r>,<r>	6K	5.0
Units	r = distance units (scalable with SCLD)		
Range	0-999,999,999.99999		(applicable only to servo axes)
Default	50		
Response	STRGTD: *STRGTD50,50,50,50 ... 1STRGTD: *1STRGTD50		
See Also	[AS], SCLD, STRGTE, STRGTT, STRGTV, TAS, TSTLT		

This command sets the target distance zone used in the Target Zone Settling Mode. The target distance zone is a range of positions around the desired endpoint that the load must be within before motion is considered complete. If scaling is enabled (SCALE1), the STRGTD value is multiplied by the distance scale factor (SCLD).

When using the Target Zone Mode, the load's actual position and actual velocity must be within the *target zone* (that is, within the distance zone defined by STRGTD and within the velocity zone defined by STRGTV) before motion can be determined complete. Axis status bit #24 (see TASF, TAS, or AS) indicates when the axis is within the zone specified with STRGTD and STRGTV; this bit is usable even if the Target Zone Mode is not enabled (STRGTE0).

If the load does not settle into the target zone before the timeout period set by STRGTT, the controller detects an error (see TASF, TAS, or AS bit #25). If this error occurs, you can prevent subsequent command and/or move execution by enabling the ERROR command to continually check for this error condition, and when it occurs to branch to a programmed response defined in the ERRORP program. (Refer to the ERRORP command description for an example of using an error program.)

*** For a more information on target zone operation, refer to the Programmer's Guide.

Example:

```
STRGTD5,5,5,5      ; Sets the distance target zone to +/-5 units
STRGTV.01,.01,.01 ; Sets the velocity target zone to <= 0.01 units/sec
STRGTT10,10,10,10 ; Sets the timeout period to 10 milliseconds on all axes
STRGTE1111        ; Enables the target zone criterion for all axes
;
; Given these target zone commands, a move with a distance of 8,000 units
; (@D8000) must end up between position 7,995 and 8,005 and settle down
; to <=0.01 units/sec within 10 ms after the commanded profile is complete.
```

STRGTE Enable Target Zone Settling Mode

Type	Servo	Product	Rev
Syntax	<!><@><a>STRGTE	6K	5.0
Units	n/a		
Range	b = 0 (disable), 1 (enable), or X (don't care)		(applicable only to servo axes)
Default	0		
Response	STRGTE: *STRGTE0000_0000 1STRGTE: *1STRGTE0		
See Also	COMEXC, STRGTD, STRGTT, STRGTV, TSTLT		

This command enables or disables the Target Zone Settling Mode. When using the target zone settling criterion, the load's actual position and actual velocity must be within the *target zone* (that is, within the position band defined by STRGTD and within the velocity band defined by STRGTV) before motion can be determined complete.

If the load does not settle into the target zone before the timeout period set by STRGTT, the controller detects an error (see TAS or AS bit #25). If this error occurs, you can prevent subsequent command and/or move execution by enabling the ERROR command to continually check for this error condition, and when it occurs to branch to a programmed response defined in the ERRORP program.

*** For a more information on target zone operation, refer to the Programmer's Guide.

Example:

```

STRGTD5,5,5,5          ; Sets the distance target zone to +/-5 units
STRGTV.01,.01,.01,.01 ; Sets the velocity target zone to <= 0.01 units/sec
STRGTT10,10,10,10     ; Sets the timeout period to 10 milliseconds on all axes
STRGTE1111           ; Enables the target zone criterion for all axes
;
; Given these target zone commands, a move with a distance of 8,000 units
; (@D8000) must end up between position 7,995 and 8,005 and settle down
; to <=0.01 units/sec within 10 ms after the commanded profile is complete.

```

STRGTT Target Settling Timeout Period

Type	Servo	Product	Rev
Syntax	<!><@><a>STRGTT<i>,<i>,<i>,<i>,<i>,<i>,<i>,<i>	6K	5.0
Units	r = milliseconds		
Range	0-5000		(applicable only to
Default	1000		servo axes)
Response	STRGTT: *STRGTT1000,1000,1000,1000 ... 1STRGTT: *1STRGTT1000		
See Also	[AS], [ER], ERROR, ERRORP, STRGTD, STRGTE, STRGTV, TAS, TER, TSTLT		

This command sets the maximum time allowed for the load to settle within the defined target zone before an error occurs.

This command is useful only if the Target Zone Settling Mode is enabled with the STRGTE command. When using the Target Zone Settling Mode, the load's actual position and actual velocity must be within the *target zone* (that is, within the position band defined by STRGTD and within the velocity zone defined by STRGTV) before motion can be determined complete. If the load does not settle into the target zone before the timeout period set by STRGTT, the servo controller detects an error (see TAS or AS bit #25).

If this error occurs, you can prevent subsequent command and/or move execution by enabling the ERROR command to continually check for this error condition, and when it occurs to branch to a programmed response defined in the ERRORP program. (Refer to the ERRORP command description for an example of using an error program.) You can check the status of the error condition with the TER and ER commands.

*** For a more information on target zone operation, refer to the Programmer's Guide.

Example (see STRGTE):

STRGTV Target Velocity Zone

Type	Servo	Product	Rev
Syntax	<!><@><a>STRGTV<r>,<r>,<r>,<r>,<r>,<r>,<r>,<r>	6K	5.0
Units	r = units/sec (scalable by SCLV)		
Range	0.00000-1,600,000.00000		(applicable only to
Default	1.00000		servo axes)
Response	STRGTV: *STRGTV1.0000,1.0000,1.0000,1.0000 ... 1STRGTV: *1STRGTV1.0000		
See Also	[AS], SCLV, STRGTD, STRGTE, STRGTT, TAS, TSTLT		

This command sets the target velocity zone for use in the Target Zone Settling Mode. The target velocity zone is a velocity range that the load must be within before motion is considered complete. If scaling (SCALE) is enabled, the STRGTV value is multiplied by the velocity scale factor (SCLV).

When using the Target Zone Mode, the load's actual position and actual velocity must be within the *target zone* (that is, within the distance zone defined by STRGTD and less than or equal to the velocity defined by STRGTV) before motion can be determined complete. Axis status bit #24 (see TASF, TAS, or AS) indicates when the axis is within the zone specified with STRGTD and STRGTV; this bit is usable even if the Target Zone Mode is not enabled (STRGTE0).

If the load does not settle into the target zone before the timeout period set by STRGTT, the servo controller detects an error (see TAS or AS bit #25). If this error occurs, you can prevent subsequent command and/or move execution by enabling the ERROR command to continually check for this error condition, and when it occurs to branch to a programmed response defined in the ERRORP program. (Refer to the ERRORP command description for an example of using an error program.)

*** For a more information on target zone operation, refer to the Programmer's Guide.

Example:

```
STRGTD5,5,5,5           ; Sets the distance target zone to +/-5 units
STRGTV.01,.01,.01,.01 ; Sets the velocity target zone to <= 0.01 units/sec
STRGTT10,10,10,10      ; Sets the timeout period to 10 milliseconds on all axes
STRGTE1111            ; Enables the target zone criterion for all axes
;
; Given these target zone commands, a move with a distance of 8,000 units
; (@D8000) must end up between position 7,995 and 8,005 and settle down
; to <=0.01 units/sec within 10 ms after the commanded profile is complete.
```

[SWAP] Task Swap Assignment

Type	Assignment or Comparison	Product	Rev
Syntax	See Below	6K	5.0
Units	n/a		
Range	n/a		
Default	n/a		
Response	n/a		
See Also	%, [SS], TTASK, TSWAP, TSKTRN, TSKAX, TSS		

The Task Swap Assignment command (SWAP) allows a binary bit pattern indicating the tasks that are currently active to be assigned to a binary variable, or evaluated in a conditional statement such as IF or WAIT. This is useful for ascertaining which tasks have any activity. To ascertain exactly what activity a specific task has at a given time, use the system status (SS or TSS).

SWAP's binary 10-bit pattern represents tasks 1-10, from left to right. A "1" indicates that the task is active, and a "0" indicates that the task is inactive. The "Task Supervisor", represented by task Ø, is always active and is therefore not included in the SWAP and TSWAP status.

Syntax: VARBn=SWAP where "n" is the binary variable number, or SWAP can be used in an expression such as IF(SWAP=b1001000000) or IF(SWAP.3=b1) or IF(SWAP=h7F0).

To check the status of only one task, you may use the bit select (.) operator. For example, VARB1=SWAP.2 assigns the binary state of Task2 to binary variable 1; or WAIT(SWAP.2=b1) establishes a wait condition that evaluates true when Task2 becomes active.