

# StepGenerator4 //SG4//

## Manual (V1.01)

# COPTONIX

Luxemburger Str. 31  
D – 13353 Berlin  
Phone: +49 – (0)30 – 61 74 12 48  
Fax: +49 – (0)30 – 61 74 12 47  
[www.coptonix.com](http://www.coptonix.com)  
[support@coptonix.com](mailto:support@coptonix.com)

---

## Features

The StepGenerator4 (SG4) is a cost effective, high performance stepper motor controller with built-in 500MHz 32Bit processor. It controls up to four stepper motor drivers with step and direction signals. The SG4 performs all real time critical tasks such as acceleration, deceleration, linear and circular interpolation and accurate step output based on target positions and velocity settings. Communication between the SG4 and the host (PC, microcontroller, etc.) takes place over asynchronous, full duplex serial port using simple ASCII commands. Furthermore the SG4 contains one I2C-bus controller (I2C-bus master) for communication with external I2C devices such as IO-Expander, AD/DA converter, etc. over the serial port (UART to I2C converter).

- RoHS compliant.
- In-field upgradeable firmware.
- 500 MHz, 32Bit processor.
- 3.3V power supply.
- 4 axes stepper motor controller.
- Outputs Step and direction signals to a stepper driver.
- 8 stop switch inputs (2 per axis).
- User programmable max. velocity, min. velocity, acceleration and deceleration.
- Built-in ramp generator for autonomous positioning and speed control.
- Symmetrical/non-symmetrical trapezoidal acceleration/deceleration driving.
- Inter-Step velocity and position calculations.
- Linear interpolation on all 4 axes XYZU.
- Circular interpolation on 6 planes, XY, XZ, XU, YZ, YU and ZU.
- Large dynamic velocity range from 1 step/s to 100,000 steps/s.
- Acceleration and deceleration range from 1 step/s/s to 100,000 steps/s/s
- Up to +/- 2,147,483,647 steps of motion.
- Read target position, actual position, velocity, acceleration, deceleration, driving state, stop switches and other flags in real-time in the motion process.
- Asynchronous, full duplex serial port
- 16 selectable baud rates from 300bps to 921600 bps.
- Simple ASCII commands over the serial port.
- 4kB serial port FIFO for ASCII commands.
- I2C-bus controller.
- I2C-bus Master transmit and receive.
- I2C 7-bit addressing.
- 3 selectable I2C SCL frequencies: 25 kHz, 50 kHz and 100 kHz.
- 40-pin DIP package PCB (600mil / 19mm x 52mm).



## Pin descriptions

PIN No.	TYPE	NAME AND FUNCTION					
1	I	<b>Ground:</b> 0V reference.					
2	O	<b>CLKU:</b> clock / step signal for U axis					
3	O	<b>CLKZ:</b> clock / step signal for Z axis					
4	O	<b>DIRU:</b> Direction signal for U axis					
5	O	<b>DIRZ:</b> Direction signal for Z axis					
6	O	<b>DIRY:</b> Direction signal for Y axis					
7	O	<b>DIRX:</b> Direction signal for X axis					
8	O	<b>CLKY:</b> clock / step signal for Y axis					
9	O	<b>CLKX:</b> clock / step signal for X axis					
10	I	<b>RXD:</b> Receiver input for the serial port					
11	O	<b>TXD:</b> Transmitter output for the serial port					
12	I	<b>Ground:</b> 0V reference					
13	O	<b>SYS0:</b> leave open, do not connect					
14	O	<b>SYS1:</b> leave open, do not connect					
15	I	<b>BD0</b> BD3–BD0:[3:0] Serial port baud rate input pins.	0x00	300	0x08	14400	
			0x01	600	0x09	19200	
16	I		<b>BD1</b>	0x02	1200	0x0A	38400
				0x03	1800	0x0B	57600
17	I		<b>BD2</b>	0x04	2400	0x0C	115200
				0x05	4800	0x0D	230400
18	I		<b>BD3</b>	0x06	7200	0x0E	460800
				0x07	9600	0x0F	921600
19	O	<b>SYS2:</b> leave open, do not connect					
20	I	<b>3.3V Power Supply:</b> This is the power supply voltage for operation					
21	I	<b>Ground:</b> 0V reference					
22	I	<b>SYS9:</b> leave open, do not connect					
23	I	<b>SYS8:</b> leave open, do not connect					
24	I	<b>SYS7:</b> leave open, do not connect					
25	I	<b>SYS6:</b> leave open, do not connect					
26	O	<b>SYS5:</b> leave open, do not connect					
27	I	<b>SYS4:</b> leave open, do not connect					
28	I	<b>SYS3:</b> leave open, do not connect					
29	I	<b>Ground:</b> 0V reference					
30	I/O	<b>SDA:</b> I2C serial data input/output					
31	I/O	<b>SCL:</b> I2C serial clock input/output					
32	I	<b>SWRU:</b> Right stop switch input for U axis					
33	I	<b>SWRZ:</b> Right stop switch input for Z axis					
34	I	<b>SWRY:</b> Right stop switch input for Y axis					
35	I	<b>SWRX:</b> Right stop switch input for X axis					
36	I	<b>SWLU:</b> Left stop switch input for U axis					
37	I	<b>SWLZ:</b> Left stop switch input for Z axis					
38	I	<b>SWLY:</b> Left stop switch input for Y axis					
39	I	<b>SWLX:</b> Left stop switch input for X axis					
40	I	<b>3.3V Power Supply:</b> This is the power supply voltage for operation					

### Operating Conditions

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
VDD	DC supply voltage	3.00	3.30	3.60	V	
IDD	Supply current		100	125	mA	
Cl	I/O load capacitance			25	pF	
Ta	Ambient operating temperature	0		70	°C	

### DC Characteristics

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
V(IH)	Input high voltage	2.00		3.60	V	A
V(IL)	Input low voltage	-0.30		0.70	V	A
V(OH)	Output high voltage	2.70			V	B,C
V(OL)	Output low voltage			0.60	V	B,C
R(PU)	Pull-up resistance		35K		Ω	D
R(PD)	Pull-down resistance		35K		Ω	D

A All pins except power supply pins.

B All I/Os are 4mA drivers

C Measured with 4mA drivers sourcing 4mA.

D Used to guarantee logic state for an I/O when high impedance. The internal pull-ups/pull-downs should not be used to pull external circuitry.

### Reset Timing

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
T(RST)	Reset pulse width	5			μs	
T(INIT)	Initialization time			150	μs	A

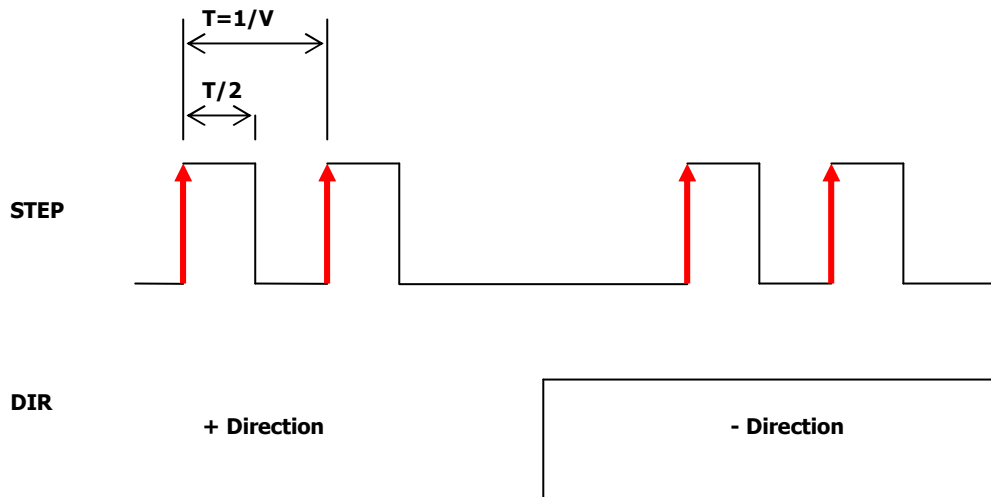
A Shows the time taken to start booting after RST has gone high.

### ESD Stress Voltage

Symbol	Parameter	MIN	TYP	MAX	UNITS	Notes
HBM	Human body model	-2.00		2.00	KV	
MM	I/O load capacitance	-200		200	V	

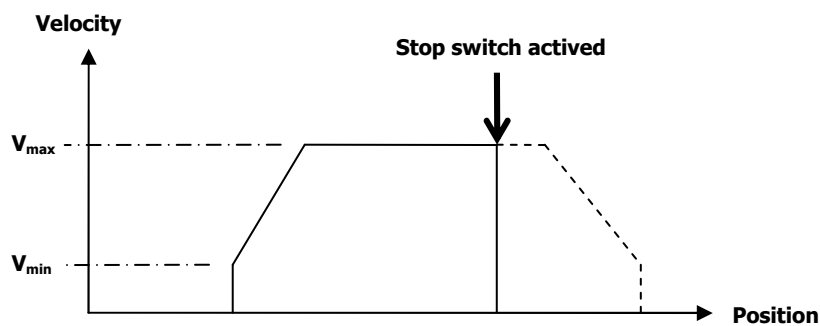
## Step and direction signals

The SG4 supports only one type of pulse output: STEP is for step signal outputting (positive edge), and DIR is for direction signal outputting. The pulse width ( $T/2$ ) depends on the velocity.  
 $T/2_{\min}(V_{\max}) = 5\mu\text{s}$  and  $T/2_{\max}(V_{\min}) = 500\text{ms}$ .



## Stop switch inputs

The SG4 has a total of eight low active stop switch inputs. For each axis, one positive and one negative stop switch input. If a stop switch becomes active while the corresponding motor is moving in its direction, then velocity parameters are ignored and the motor is stopped immediately.



---

## The Serial Port

The SG4 provides one asynchronous, full duplex serial port with configurable baud rate between 300 bps and 921600 bps. The serial port is used for communication with the SG4. A 4kB (4096 Bytes) FIFO is available for buffering positioning commands.

Driving all 4 axes from the actual position to a target position is one command. e.g. if the actual position is  $X_a=500$ ,  $Y_a=1000$ ,  $Z_a=-100$ ,  $U_a=150$  and all four axes should to be moved (linear interpolated) to  $X_t=1750$ ,  $Y_t=5000$ ,  $Z_t=1100$ ,  $U_t=-600$ , then the ascii command would be:

relative: "s50: n100 x1250 y4000 z1200 u-750\r"

OR

absolute: "s51: n100 x1750 y5000 z1100 u-600\r"

The absolute positioning command needs 44 Bytes (including header) space in the FIFO and the relative positioning command needs only 24 Bytes. A circular interpolation command needs 28 Bytes space.

## The I2C Bus

For communication with external devices an I2C bus controller is available. The I2C bus controller operates only as master at 25, 50 or 100 kHz. The SCL and SDA signals need external pull-up resistors to 3.3V.

In a CNC application one would need additional I/Os e.g. to switch on/off a cooling system, vacuum cleaner and so on. With the I2C bus controller as Master numerous bus participants can be addressed via the serial port such as IO-Expander, PWM, sensors, AD/DA converters, EEPROM, LCDs, 7 segment display etc.

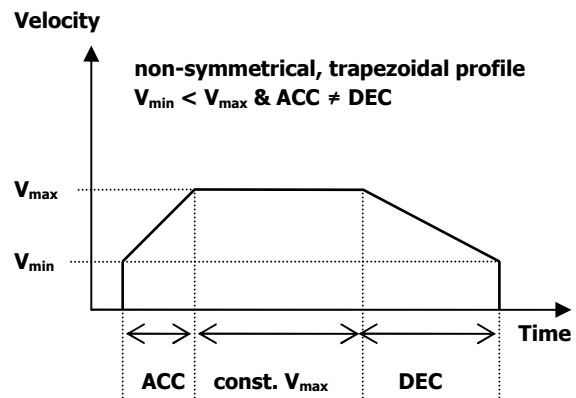
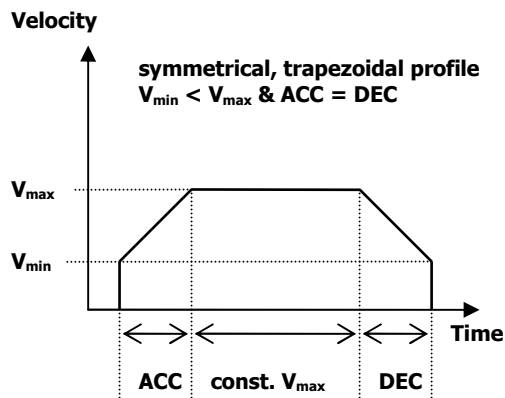
e.g. The ascii command for addressing an IO-Expander with a slave address of 0x40 and setting all it's pins to "1s" would be: "iw:4001FF\r".

## Velocity parameters

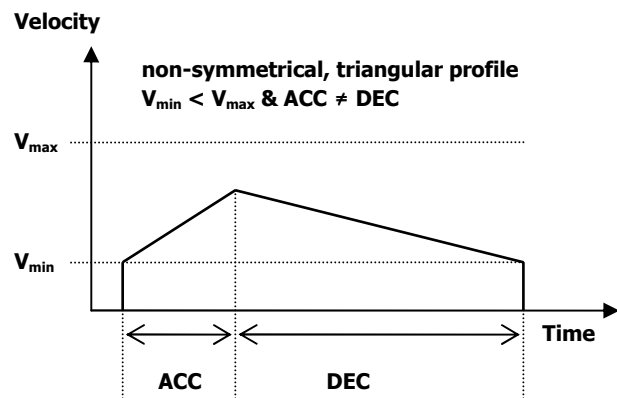
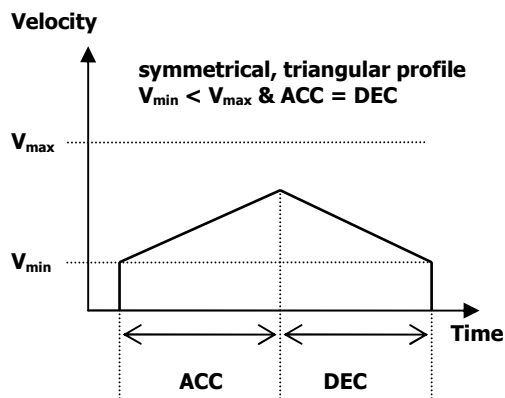
The velocity parameters are:

- minimum velocity  $V_{min}$  [steps/second] ;  $1 \leq V_{min} \leq 100,000$  steps/second
- maximum velocity  $V_{max}$  [steps/second] ;  $1 \leq V_{max} \leq 100,000$  steps/second
- acceleration  $ACC$  [steps/second/second] ;  $1 \leq ACC \leq 100,000$  steps/second/second
- deceleration  $DEC$  [steps/second/second] ;  $1 \leq DEC \leq 100,000$  steps/second/second

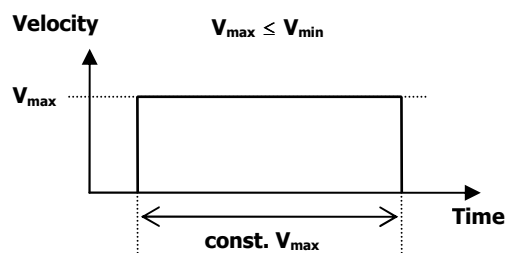
After a positioning command is issued, the SG4 starts driving a motor at minimum (start) velocity  $V_{min}$ , accelerates to the maximum velocity  $V_{max}$  at acceleration rate  $ACC$ , continues driving the motor at maximum velocity  $V_{max}$  to a precalculated position, decelerates at deceleration rate  $DEC$  to the minimum velocity  $V_{min}$  and stops after at target position.



If number of steps is not big enough to accelerate to the maximum velocity  $V_{max}$ , then a triangular profile is generated.



If  $V_{max}$  is less than or equal to  $V_{min}$ , then the SG4 drives a motor with constant velocity at  $V_{max}$ .



## Linear interpolation

The SG4 supports full linear interpolation on any 1, 2, 3 or 4 axes in the position range of  $\pm 2,147,483,647$  steps. The linear interpolation command needs at least one parameter for one axis, e.g:

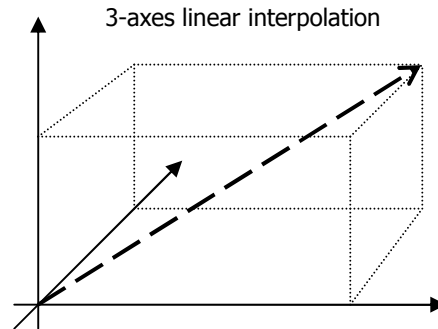
"s51: x1000\r" -> driving X-axis to position 1000

"s51: x1250 y2000\r" -> driving X and Y-axis to position 1250;2000.

"s51: z500\r" -> driving Z-axis to position 500

"s51: y250 u1750\r" -> driving Y and U-axis to position 250;1750.

"s51: x1000 y2000 z3000 u4000\r" -> driving X, Y, Z and U-axis to position 1000;2000;3000;4000.



## Circular interpolation

The SG4 supports circular interpolation on 6 working planes, the XY, XZ, XU, YZ, YU and ZU. A circular interpolation command needs at least the START position ( $X_s, Y_s$ ), the END position ( $X_e, Y_e$ ) and the CENTER position ( $X_c, Y_c$ ). The START position is the actual position. The CENTER position is always given relatively to the starting position of the arc. The parameters for the center position are I and J.

Examples:

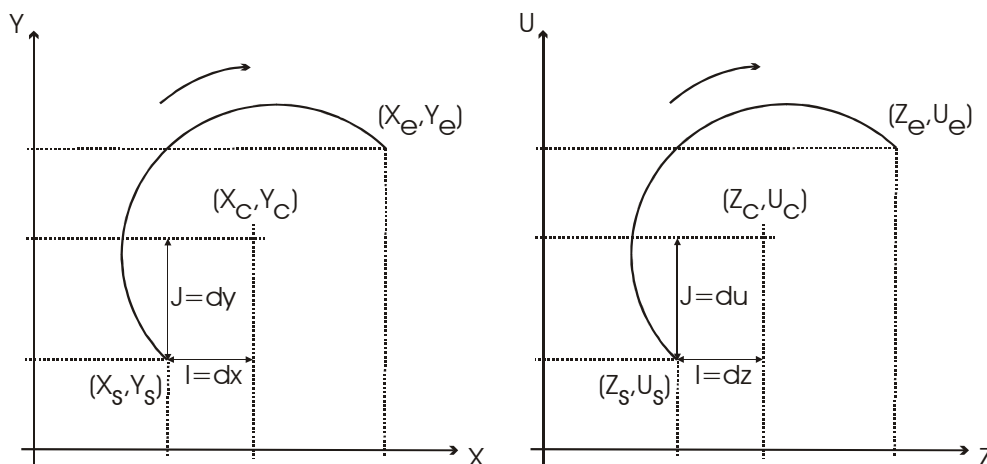
"s52: x1000 y0 I500 J0\r"

Working plane will be automatically set to XY-Plane.

The radius and the center position of the circle will be calculated using the parameters I and J. SG4 drives X and Y-axis clockwise from the actual position (0,0) to the target position (1000,0).

If the target position is equal to the actual position, then the SG4 will drive a full circle:

"s52: z0 u0 i500 j0\r" -> ZU-Plane selected; full circle; radius = 500; center of circle = (500,0).



## ASCII Commands

- All sent/received characters are represented as ASCII data
- Command ID and parameter are separated by a colon ":"
- Command ID and parameter are always terminated with a Carriage Return ( $\backslash r = 13 \text{ DEC} = 0x0D \text{ HEX}$ ).

Example: "s1:2000\r" => set maximum velocity to 2000 steps/second

Command ID: "s1" = set  $V_{\max}$ .

Parameter: "2000" = 2000 steps/second.

Carriage Return: "\r" = 0x0D.

On success the SG4 sends an acknowledge / return value: "s1:;\r"

If the command fails, the SG4 sends one of the following messages:

"fifo\_full\r"

"param\_error\r"

"i2c\_param\_err\r"

"i2c\_scl\_err\r"

"running\r"

"unknown\_cmd\r"

## Set maximum velocity – $V_{\max}$ [steps/second]

sets the upper velocity limit.

Command ID: "s1".

The command bypasses the FIFO, and is executed after the current command is finished.

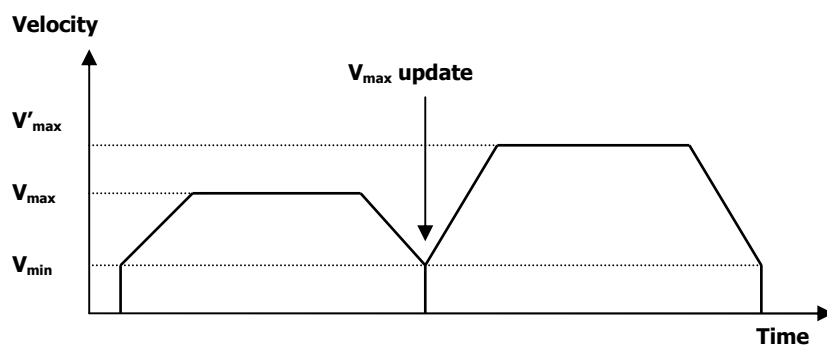
Command ID: "s5".

This command is buffered first into FIFO, and then executed after all previous FIFO buffered commands are successfully finished.

Examples:

Set  $V_{\max}$  to 10000 steps/second:

"s1: 10000\r" -> return value "s1:;\r" **OR** "s5: 10000\r" -> return value "s5:;\r"



---

### **Set minimum velocity – $V_{min}$ [steps/second]**

sets the lower velocity limit.

Command ID: "**s2**".

The command bypasses the FIFO, and is executed after the current command is finished.

Command ID: "**s6**".

This command is buffered first into FIFO, and then executed after all previous FIFO buffered commands are successfully finished.

Examples:

Set  $V_{min}$  to 2000 steps/second:

"s2: 2000\r" -> return value "s2:;\r" **OR** "s6: 2000\r" -> return value "s6:;\r"

### **Set acceleration – ACC [steps/second/second]**

sets the acceleration rate.

Command ID: "**s3**".

The command bypasses the FIFO, and is executed after the current command is finished.

Command ID: "**s7**".

This command is buffered first into FIFO, and then executed after all previous FIFO buffered commands are successfully finished.

Examples:

Set  $ACC$  to 1000 steps/second/second:

"s3: 1000\r" -> return value "s3:;\r" **OR** "s7: 1000\r" -> return value "s7:;\r"

### **Set deceleration – DEC [steps/second/second]**

sets the deceleration rate.

Command ID: "**s4**".

The command bypasses the FIFO, and is executed after the current command is finished.

Command ID: "**s8**".

This command is buffered first into FIFO, and then executed after all previous FIFO buffered commands are successfully finished.

Examples:

Set  $DEC$  to 1000 steps/second/second:

"s4: 1000\r" -> return value "s4:;\r" **OR** "s8: 1000\r" -> return value "s8:;\r"

### Linear interpolation – Incremental positioning [steps]

moves all 4 axes number of steps relative to the actual position.

Command ID: **"s50"**.

This command is buffered first into FIFO, and then executed after all previous FIFO buffered commands are successfully finished. At least a target position of one axis must be specified.

Example – One axis:

Actual position: (1000, 1500, -1000, 500)

Target position: (1500, 1500, -1000, 500)

"s50: x500\r" -> return value "s50:;\r"

This command moves only the x-axis to position 1500.

Example – Two axes:

Actual position: (1000, 1500, -1000, 500)

Target position: (1500, 1750, -1000, 500)

"s50: x500 y250\r" -> return value "s50:;\r"

This command moves only the x-axis to 1500 and y-axis to 1750.

Example – Four axes:

Actual position: (1000, 1500, -1000, 500)

Target position: (1500, 1750, -500, -500)

"s50: x500 y250 z500 u-1000\r" -> return value "s50:;\r"

A linear interpolation command may take one more optional parameter, the User Defined ID Number. e.g. in a CNC application, G-Code may have a line number e.g. **N10** G01 X1500 Y1750 Z-500.

"s50: **n10** x500 y250 z-1000\r" -> return value "s50:;\r"

The User Defined ID Number gives the possibility to determine at any time the currently running command (G-Code line number).

### Linear interpolation – absolute positioning [steps]

moves all 4 axes number of steps relative to Zero position.

Command ID: **"s51"**.

This command is buffered first into FIFO, and then executed after all previous FIFO buffered commands are successfully finished. At least a target position of one axis must be specified.

Example – Four axes:

Actual position: (1000, 1500, -1000, 500)

Target position: (1500, 1750, -500, -500)

"s51: n10 x1500 y1750 z-500 u-500\r" -> return value "s51:;\r"

## Circular interpolation [steps]

The SG4 supports circular interpolation on 6 working planes, the XY, XZ, XU, YZ, YU and ZU. It is possible to drive arcs and full circles clockwise (CW) or counter clockwise (CCW).

Circular interpolation commands are buffered first into FIFO, and then executed after all previous FIFO buffered commands are successfully finished.

This commands takes 4 parameters:

- the target position of the first axis (X, Y, Z or U)
- the target position of the second axis (X, Y, Z or U)
- the center position relative to the first axis or zero position ( I )
- the center position relative to the second axis or zero position ( J )

Precedence: X -> Y -> Z -> U  
XY -> XZ -> XU -> YZ -> YU -> ZU

Circular interpolation - Command IDs:

Absolute target position, incremental center, CW:	"s52"
Absolute target position, incremental center, CCW:	"s53"
Absolute target position, absolute center, CW:	"s54"
Absolute target position, absolute center, CCW:	"s55"
Incremental target position, incremental center, CW:	"s56"
Incremental target position, incremental center, CCW:	"s57"
Incremental target position, absolute center, CW:	"s58"
Incremental target position, absolute center, CCW:	"s59"

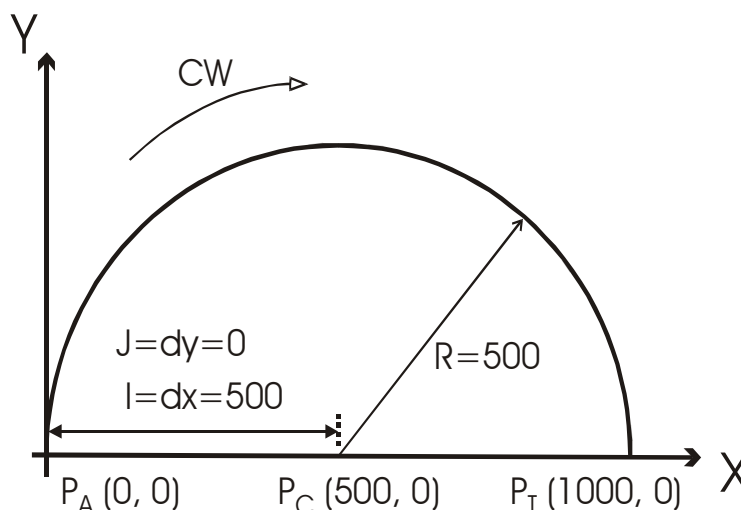
Example 1:

Actual position: (0, 0, 0, 0)

"s52: x1000 y0 i500 j0\r"

- Working plane is set automatically to XY-Plane.
- Radius is calculated using the parameters I, J and the actual position (0, 0, 0, 0).
- Center of circle is calculated using the parameters I, J and the actual position.
- Summary:

Interpolation:	Circular
Direction:	Clockwise
Working plane:	XY
Actual position:	X=0, Y=0
Target position:	X=1000, Y=0
Center position:	X=500, Y=0
Radius of circle:	R=500



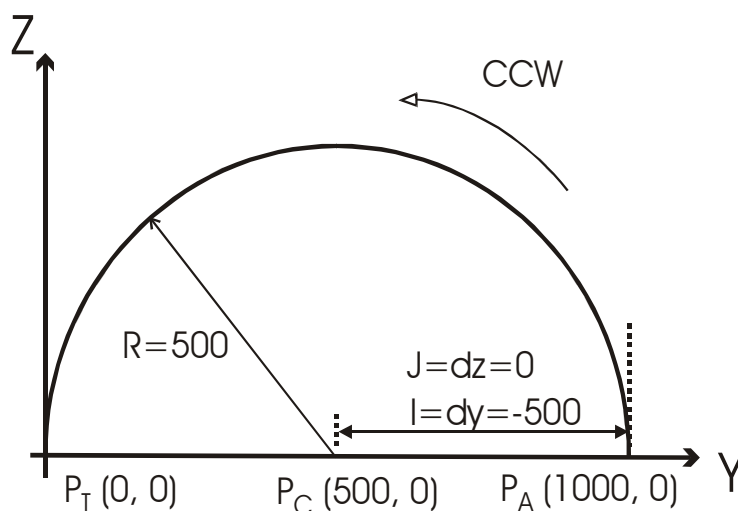
Example 2:

Actual position: (0, 1000, 0, 0)

"s53: y0 z0 i-500 j0\r"

- Working plane is set automatically to YZ-Plane.
- Radius is calculated using the parameters I, J and the actual position (0, 1000, 0, 0).
- Center of circle is calculated using the parameters I, J and the actual position.
- Summary:

Interpolation:	Circular
Direction:	Counter Clockwise
Working plane:	YZ
Actual position:	Y=1000, Z=0
Target position:	Y=0, Z=0
Center position:	Y=500, Z=0
Radius of circle:	R=500



### Delay [microsecond]

A delay command forces the system to wait some time before performing the next command. The time interval for delay is in the range from 1 $\mu$ s to 20s (in 1 $\mu$ s steps).

This command is buffered first into FIFO, and then executed after all previous FIFO buffered commands are successfully finished.

Command ID: "s40"

Example 1:

"s40:5000\r" -> return value "s40:;\r"

Example 2:

"s51:x0 y0 z500 u0\r"	//go to home position (0, 0, 0, 500, 0)
"s40:1000\r"	//wait 1ms
"s51:z-100\r"	//go to milling depth (0, 0, -100, 0)
"s40:1000\r"	//wait 1ms
"s51:x1000 y1000\r"	//go to position (1000, 1000, -100, 0) linear interpolation
"s40:5000\r"	//wait 5ms
"s53:x0 y0 i-500 j-500\r"	//go to position (0, 0, -100, 0); circular interpolation CCW
	//Center X=500, Y=500; Radius = 500.

### Set Radius Accuracy [steps]

With this command, the required accuracy of the circle radius is parameterized. When using circular interpolation "s52" and "s53", the difference between radii length

R1(actual position ; center position) and R2(target position ; center position)

must be equal to or less than this "Radius Accuracy"-parameter. If the difference between the radii length is greater than the "Radius Accuracy", an error is generated, error flag bit is set and FIFO is disabled. In order to resume normal operation, the FIFO must be cleared using the command "ResetFIFO"

Command ID: "s60".

Example:

"s54:5\r" -> return value "s54:;\r"

e.g. if the resolution of a milling machine is 0.002 mm/step, then the radius accuracy is 0.01mm. If the difference between radii length is greater than 0.01mm, an error is generated.

### Set actual position [steps]

This command initializes the actual position. e.g. after Homing it is useful to set the actual positions of all axes to zero.

Command ID: "s61".

Example 1:

Actual position: (0, 0, 0, 0)

```
"s51: x1000 y1000 z1000 u1000\r" //go to position (1000, 1000, 1000, 1000)
"s55: x0 y0 z0 u0\r" //do not drive axes, only set position to (0, 0, 0, 0)
```

```
"s55: x0\r"
"s55: x0 y0\r"
"s55: y0\r"
"s55: x0 z0\r"
"s55: y0 z0\r"
"s55: y0 u0\r"
```

Example 2:

Set actual position (0, 0, 0, 0) to (1000, 500, 100, -150)

```
"s55:x1000 y500 z100 u-150\r"
```

## HardStop

A HardStop command forces all axes to stop immediately. Velocity parameters are ignored. Once HardStop command is issued, then the FIFO will be disabled. Thus all FIFO pending commands will not be executed. Call "Continue" command to enable the FIFO and to resume execution of pending commands.

Command ID: "d"

Example 1:

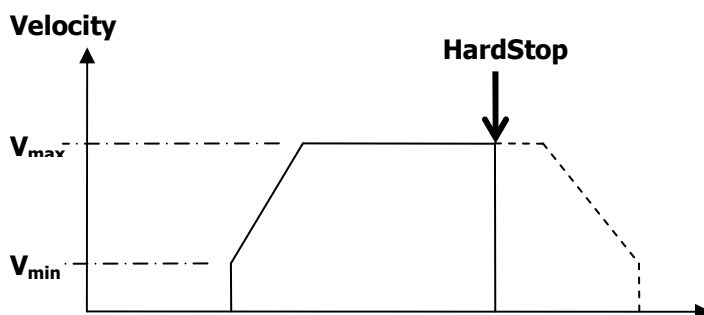
"d\r" -> return value "d;\r"

Example 2:

"s51:x0 y0 z500 u0\r" // go to home position (0, 0, 500, 0)

"s51:x1000\r" //go to position (1000, 0, 500, 0)

"d\r" //Hardstop -> stop immediately



## SoftStop

A SoftStop command starts deceleration to  $V_{min}$  and then stops all axes. Once SoftStop command is issued, then the FIFO will be disabled. Thus all FIFO pending commands will not be executed. Call "Continue" command to enable the FIFO and to resume execution of pending commands.

Command ID: "t"

Example 1:

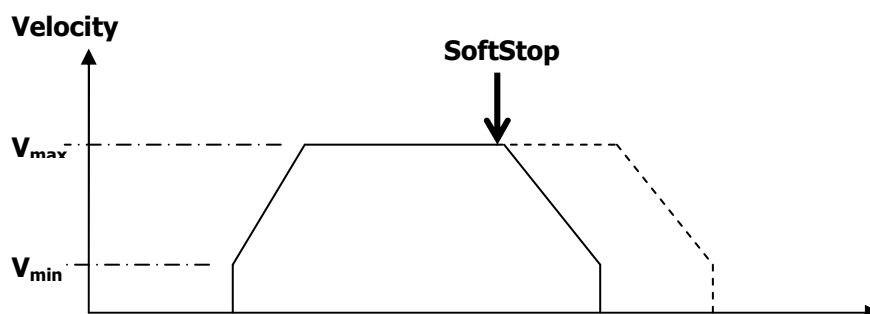
"t\r" -> return value "t;\r"

Example 2:

"s51:x0 y0 z500 u0\r" //go to home position (0, 0, 500, 0)

"s51:x1000\r" //go to position (1000, 0, 500, 0)

"t\r" //SoftStop -> start deceleration to  $V_{min}$  and stop



### **ResetFIFO**

This command clears all pending commands in the FIFO and enables FIFO again.

Command ID: "r"

Example 1:

"r\r" -> return value "r;\r"

### **Continue**

After a "HardStop", "SoftStop" or a Stop due to external events (Stop switch) the FIFO is disabled and execution of pending commands is stopped. The "Continue" command enables the FIFO and resumes normal operation.

Command ID: "c"

Example 1:

"c\r" -> return value "c;\r"

### **Reset Interrupt Flag**

After a Stop switch becomes active the corresponding flag bit is set. Interrupt flag bits must be cleared by software.

Command ID: "f"

Example 1:

"f\r" -> return value "f;\r"

---

**Get maximum velocity –  $V_{max}$  [steps/second]**

This command returns the maximum velocity  $V_{max}$ .

Command ID: **"g1"**

Example:

"g1\r" -> return value "g1:1000;\r" //  $V_{max} = 1000$  steps/second

**Get minimum velocity –  $V_{min}$  [steps/second]**

This command returns the minimum velocity  $V_{min}$ .

Command ID: **"g2"**

Example:

"g2\r" -> return value "g2:500;\r" //  $V_{min} = 500$  steps/second

**Get actual velocity –  $V_{act}$  [steps/second]**

This command returns the actual velocity  $V_{act}$ . In the motion process  $V_{act}$  is greater than zero.

Command ID: **"g3"**

Example 1:

"g3\r" -> return value "g3:5000;\r" // The actual velocity is 5000 steps/second

Example 2:

"g3\r" -> return value "g3:0;\r" // The actual velocity is ZERO. All axes are stopped.

**Get acceleration –  $ACC$  [steps/second/second]**

This command returns the acceleration rate  $ACC$ .

Command ID: **"g4"**

Example:

"g4\r" -> return value "g4:1000;\r" //  $ACC = 1000$  steps/second/second

**Get deceleration –  $DEC$  [steps/second/second]**

This command returns the acceleration rate  $DEC$ .

Command ID: **"g5"**

Example:

"g5\r" -> return value "g5:1000;\r" //  $DEC = 1000$  steps/second/second

**Get actual position [steps]**

This command returns the actual position of all axes.

Command ID: **"g6"**

Example:

"g6\r" -> return value "g6:1000;1500;2000;-3000;\r"

The actual position of 4 axes is: X=1000; Y=1500; Z=2000; U=-3000

**Get target position [steps]**

This command returns the target position of all axes.

Command ID: **"g7"**

Example:

"g7\r" -> return value "g7:10000;5000;2500;1250;\r"

The target position of axes is: X=10000; Y=5000; Z=2500; U=1250

### Get status [steps]

This command returns the current state of the eight stop switches, interrupt flag bits, running state, error flag bit, FIFO enabled bit and delay flag bit.

Command ID: **"g8"**

Example:

"g8\r" -> return value "g8:000010FF;\r"

The return value is 32bit hexadecimal string:

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 LSB	Left Stop Switches If Bit is set to "1", then the corresponding input is HIGH (not active), if "0" then the corresponding input is LOW (active). Bit 4: X-axis Bit 5: Y-axis Bit 6: Z-axis Bit 7: U-axis				Right Stop Switches If Bit is set to "1", then the corresponding input is HIGH (not active), if "0" then the corresponding input is LOW (active). Bit 0: X-axis Bit 1: Y-axis Bit 2: Z-axis Bit 3: U-axis			
1	0	DELAY	ERROR	FIFO	Running State: 0x00: Stopped 0x01: Acceleration phase 0x02: Running with max. velocity 0x03: Deceleration phase 0x04: Running with min. velocity 0x05: SoftStop 0x06: HardStop 0x07: Stopping			
2	NOT USED IN CURRENT VERSION (0x00)							
3 MSB	Interrupt Flag Bits (Left Stop Switches) If Bit is set to "1", then the corresponding Stop Switch was or still active, if "0" then the corresponding Stop Switch did not become active yet. These Bits must be cleared by software using the "Reset Interrupt Flag" command.  Bit 4: X-axis Bit 5: Y-axis Bit 6: Z-axis Bit 7: U-axis				Interrupt Flag Bits (Right Stop Switches) If Bit is set to "1", then the corresponding Stop Switch was or still active, if "0" then the corresponding Stop Switch did not become active yet. These Bits must be cleared by software using the "Reset Interrupt Flag" command.  Bit 0: X-axis Bit 1: Y-axis Bit 2: Z-axis Bit 3: U-axis			

---

### Get ID Number

This command returns the User Defined ID Number **N**.

Command ID: **"g9"**

Example:

"g9\r" -> return value "g9:110;\r"

The returned ID number is 110.

### Get actual parameters

This command returns the actual position of all axes, the actual velocity, the actual ID number and the status.

Command ID: **"g10"**

Example:

"g10\r" -> return value "g10:000014FF;110;1542;3501;-562;0;4000;\r"

Status = 0x000014FF

ID Number = 110

X actual position = 1542

Y actual position = 3501

Z actual position = -562

U actual position = 0

Actual velocity = 4000 steps/second

### Get Radius Accuracy

This command returns the radius accuracy (in steps).

Command ID: **"g11"**

Example:

"g11\r" -> return value "g11:5;\r"

The returned radius accuracy value is 5.

## Write I2C

Writes data to an I2C device.

Command ID: **"iw"**

This command takes at least 3 parameters:

- Slave address
- Number of bytes to write
- Data to write

All parameters are written as hexadecimal string.

Example:

Write two bytes 0xAA and 0xFF to slave address 0x40

**"iw:4002AAFF\r"** -> return value: on success **"iw:02;\r"** and on failure **"iw:00;\r"**

Command ID: **"iw"**

Slave address: **0x40**

Number of bytes: **0x02**

Data: **0xAA, 0xFF**

## Read I2C

Reads data from an I2C device.

Command ID: **"ir"**

This command takes two parameters:

- Slave address
- Number of bytes to read

All parameters are written as hexadecimal string.

Example:

Read two bytes from slave address **0x40**

**"ir:4002\r"** -> return value: on success **"ir:02C2A4;\r"** and on failure **"ir:00\r"**

Command ID: **"ir"**

Slave address: **0x40**

Number of bytes: **0x02**

Data: **0xC2, 0xA4**

## Set I2C Frequency

Set the I2C SCL frequency.

Valid frequencies: 25 kHz, 50 kHz or 100 kHz

Command ID: **"if"**

Example:

**"if:100\r"** -> return value **"if;\r"**

This command would set the I2C frequency to 100 kHz.

**Dimensions (mm)**

40-pin DIP package PCB

