

30 Servo Controller User Manual

This document outlines the usage of the SC30 servo controller available from www.scratch-technology.co.uk

Features

- Low cost design
- 30 independent outputs
- 0.7 degree resolution
- Individual position control
- Individual enable / disable
- Low power consumption
- Programmable baud rate
- Scripting function (coming soon)

Outline

The SC30 produces 30 independently controllable pulse outputs that specify to the connected servos the position they must move to.

The device requires a suitable DC voltage supply, a Host PC and some 5 volt RC type servo motors.

Typical servos have an angular range of ~180 degrees. The SC30 has a resolution of 255 steps over this range, giving an angular resolution of ~0.7 degrees / step.

Positions are specified to the SC30 from the host computer – they may be in the form of a raw position (0-255) or as a percentage (0-100).

Each servo control pulse may be independently enabled or disabled, this will cause the servo(s) in question to maintain their commanded position actively or cease to do so.

The SC30 servo output pulses are driven directly from an 18 series PIC device from Microchip. Data sheets are available from www.microchip.com Care must be taken to never source or sink more than 20mA on any of the servo control signal outputs.

Communications

The device communicates with a PC host via a standard COM port (RS232), a 9 pin D type cable is required.

The device may be controlled manually by a user through a terminal emulation program, for example, 'hyperterminal' on Windows or 'cutecom' on Linux.

Default configuration of the device is as follows:

|

Parameter	Default Value	Allowed Values
Baud Rate	115200	115200 (1) 57600 (2) 19200 (3) 9600 (4)
Flow Control	None	None
Parity	None	None
Data Bits	8	8
Start Bit	1	1
Stop Bits	1	1

in short, the default configuration is 115200bps, 8N1. Only the baud rate may be changed – this is achieved using the BAUD command as explained below.

Commands

Move

Move individual servos, groups of servos or all servos to the position specified.

The general format of this command is:

M<number>[#]<position>[#/%]<enter>

- M is the move command
- <number> specifies the selected servo(s) and is:
 - values delimited with commas ',' e.g. 21,1,0,15
 - ranges of servos e.g. 3-12
 - a '*' on it's own for all servos
- # '#' delimits the servo number section
- <position> is either as a percentage (0-100) or as an absolute value(0-255)
- [#/%] either of the '%' or # symbols

Note: The * servo specifier is invalid unless used alone, no servo numbers or ranges can be specified with it.

Examples of this command:

Command	Result
M*#255#<enter>	Move all servos to position 255
M*#100%<enter>	Same as above
M3,1#50%<enter>	Move servos 3 and 1 to 50% position
M25#10#<enter>	Move servo 25 to position 10
M1,2,12-15#10%<enter>	Move servos 1,2,12,13,14 and15 to the 10% position

Enable

Enable individual servos, groups of servos or all servos.

The general format of this command is:

E<number>#<enter>

E*#<enter> Enable all servos

<number> works in the same way as specified with the 'Move' command, comma separated lists and ranges are valid.

Disable

The disable command has identical syntax to the enable command, the result is that the selected servo or all servos are disabled.

D<number>#<enter>

D*#<enter> Disable all servos

Baud

The baud command allows user selection of the device baud rate. The chosen baud rate is stored in EEPROM and is remembered after a power cycle.

B<number>#

<number> is a single digit from 1 to 4 that corresponds to a baud rate as follows:

- <number> = 1 -> BAUD = 115200
- <number> = 2 -> BAUD = 57600
- <number> = 3 -> BAUD = 19200
- <number> = 4 -> BAUD = 9600

all communications are 8N1.

Invalid Commands

Commands are monitored as they are entered. If an invalid command is being typed a ? is returned to the PC and the previously entered parts of the command are lost. The user must re-enter the command.