

Using the TTT Serial Servo Controller Chip

Pin descriptions () indicates marking on the chip label.

- 1 (+) +3V to +5.5V operating voltage, 5V is best setting.
- 2 (S/L) Tied high through a 4.7K resistor puts chip in short pulse (90 degree) mode
Tied low through a 4.7K resistor puts chip in long pulse (180 degree) mode
The modes are set on power-up and won't change until the power is cycled.
- 3 (3) This is the fourth servo (of servos 0-3)
- 4 (I) This is the 2400 BAUD serial input, may need to pull high with a 4.7K resistor.
- 5 (2) This is the third servo (of servos 0-3)
- 6 (1) This is the second servo (of servos 0-3)
- 7 (0) This is the first servo (of servos 0-3)
- 8 (G) This is the ground pin for the chip

When installing the TTT SSC into your project there are a few electrical guidelines that should be followed.

1. Install a .1uf ceramic cap near the chip, this eliminates most noise problems.
2. If you are using the same voltage line to power your servos as you are to power the chip then you should also install a 10uf to 33uf electrolytic cap near the PIC to avoid "brownouts" that the servos may cause.
3. Pull the serial input line high (I pin) with a 4.7K or 10K resistor to make sure that no random noise is interpreted as a serial start bit if you are getting glitching.
4. Install a jumper on the S/L pin such that if the jumper is not installed then you have a logic high there (pulled high with a 10K resistor) and if the jumper is there you pull to ground. Alternatively you can just tie this pin high or low if you never change your mind.

To communicate to the TTT SSC use a normal RS232 connection at 2400 BAUD. The RS232 is NON-inverted, or TRUE RS232. It requires a single byte whose format is: [SSPPPPPP]. Where SS is the servo target, 0-3 and PPPPPP is the position desired, 0-63. This is the binary format. The servos will be numbered as follows. Just add the position value to the servo value below for a command.

Servo 0	Add 0
Servo 1	Add 64
Servo 2	Add 128
Servo 3	Add 192

In other words, if you want servo 2 (third servo) to be set to its midpoint, the data you send to the chip would be 32 + 128 or 160.

The TTT SSC will check the incoming data for framing errors and reject most bad data.

Sending a 0 to servo 0 (just plain 0 sent) will disable servos 0 and 1, servos 2 and 3 will continue to follow commands. You can still write to servos 0 and 1, they just won't respond. This is to give a reliable "stop" command to robots that use hacked servos as drive motors. When you send a 0 to any other servo but 0 (send 64 for example) it will re-enable servos 0 and 1 and they will take up with whatever the last command sent to them was. 0 is therefor not a valid servo position.

RC Hobby servos work using a pulse that is from 1ms to 2ms with 1.5ms being the center position. Some servos will function from 0ms to 2ms, these latter servos are using what we call "long pulse" decoding. The TTT SSC can be set for either short (1-2ms) timing for long (0-2ms) timing by tying pin 2 high or low, respectively.

On reset or power-up, the TTT SSC will default to the center position for short pulse servos (1.5ms) and will remain there until that servo is assigned another position. The PIC12C508 will change timing characteristics at different voltages in this configuration so it is best if it gets at least 5V for proper timing of servos. Erratic behavior will occur if it gets a lower voltage supplied to the (+) pin.

If lots of positioning information is being sent over a short period of time (unusual circumstance) the servos may exhibit some "jitter". This is normal and unavoidable for a chip that only runs at 4MHZ. With normal operation no jitter should be observed.

This is a Parallax Basic Stamp I Example of how to use the TTT SSC chip.

```
symbol n=b0
symbol servo = b1
symbol posit = b2
symbol sout = b3

'LED flash to show we are alive, LED on P1
for n = 1 to 5
  low 1
  pause 50
  high 1
  pause 50
next

'Test servos 0 and 1 with some steps out and back
for servo = 0 to 1
  for posit = 22 to 62 step 10
    sout = servo * 64 + posit
    serout 0,2400,(sout)
    pause 500
  next posit
next servo

for servo = 1 to 0 step -1
  for posit = 62 to 22 step -5
    sout = servo * 64 + posit
    serout 0,2400,(sout)
    pause 250
  next posit
next servo

done:
goto done

end
```

Dennis Clark
TTT Enterprises
June, 1999