This is the Revision H version of the Servo4 module. The status of this project is finished.

Servo4 Module (Revision H)

Table of Contents

This document is also available as a PDF document.

1. Introduction

The Servo4 module allows for the control of up to 4 hobby grade servos. It can be configured in the following ways:

Pure Servo Mode
: In pure servo mode, it is expected that up to 4 unmodified servos are attached to the board. The four servos can be independently controlled. There is current feedback on all four servos.

Differential Steering Mode
: In differential steering mode, the module can control up to 4 servos. The first two servos are expected to be servos that have modified been for continuous rotation. The second two servos are regular unmodified servos. The first two servos have current feedback and the second two servos do not. There are two trim pots that are used to set the no rotation condition for the first two servos.

MSSC Compatibility Mode
: MSSC stands for Mini−Serial Servro Controller. The MSSC board is quit popular. In MSSC mode, up to 4 servos can be controlled per board. In MSSC compatibility mode, the servos are directly controlled from a host computer via an RS−232 serial port. Up to 16 boards can be chained together to provide control for up to 64 servos. The two trim pots are used to set the "address" of the board.

As you can see, this board is quite flexible. Please see the section on Hardware Configuration to see how the jumpers for each configuration.

2. Hardware Configuration

Up to four RC servos are connected to connectors N2 (servo 0) through N5 (servo3). Each connector has the following pin definitions:

Pin	Location	Description
1	Left	Servo control signal (varies between 0 and 5 volts)
2	Center	5 Volts

3	Right	Ground (0 Volts)

On many servos, the black wire is the ground wire. You will have to check you servo documentation to be absolutely sure though.

The connection to the controlling module occurs via N1 in the upper left corner. Alternatively, in MSSC mode, the connection is via N12.

Power for the servos comes from N6, the blue two terminal connector in the upper right corner. Connect a power source of 6−12 volts to connector N6, where the upper terminal is the positive terminal ('+') and the the lower terminal is negative ('−'). The on board regulator, will regulate the voltage down to 5 volts for the servos.

The hardware configuration for each mode is summarized in the table below:

Mode	**Jumpers**				**Trim Pots**	
	N7	**N8**	**N10**	**N11**	**R4**	**R5**
Pure Servo Mode	Right (2−3)	Right (2−3)	Right (2−3)	Off	Unused	Unused
Differential Steering Mode	Left (1−2)	Left (1−2)	Right (1−2)	Off	Servo 0 Stop	Servo 1 Stop
Differential Steering Calibration Mode	Left (1−2)	Left (1−2)	Left (1−2)	Off	Servo 0 Stop	Servo 1 Stop
MSSC Compatibility Mode	Left (1−2)	Left (1−2)	Right (1−2)	On	Address A2−A3	Address A4−A5

In differential steering calibration mode, N11 is jumpered to the left and it causes yellow LED D1 to light. It causes both servos 0 and 1 to be enabled. The value of trim pot R5 to be sent to servo 0 and trim pot R6 to be sent to servo 1. The purpose of calibration mode is to allow you to adjust the two modified servos that are connected to servo 0 and servo 1 and adjust them until they stop rotating. This frees the programmer from having to experiment to find the `position' number for each servo that corresponds to each servo being motionless. The values of the stop value are read out using the Read Current Draw command for servo 2 and 3.

In MSSC (Mini Serial Servo Controller) mode, all power comes in from the power source connected to N6 (the blue 2−terminal block.) No cable is connected to N1. In order to talk to the Servo4 module, a cable is constructed from a two pin male header and a female 9−pin DB9 connector. The connections are summerized in the table below:

From	**To**	**Description**
DB9 Pin 3	2−pin Header Pin 1	Host Transmitted Data
DB9 Pin 5	2−pin Header Pin 2	Ground (0 Volts)
DB9 Pin 7	DB9 Pin 8	RTS to CTS
DB9 Pin 4	DB9 Pin 1 and DB9 Pin 6	DTR to DSR and DCD

In MSSC (Mini Serial Servo Controller) mode, the servo board controls 4 servos at a time. An address is 6−bits long and is represented as a decimal number between 0 and 63 inclusive as shown below:

aa bb ss

where

aa
: is the two high order address bits and is set by trim pot R5.

bb
: is the two middle order address bits and is set by trim pot R6.

ss
: is the servo specifier and selects between servo 0 through servo 3.

The trim pots are set according to the following table:

Value	Position
00	7:00 (full counter clockwise)
01	10:00
10	2:00
11	5:00 (full clockwise)

Thus, to set the MSSC address to servo bank 0−3, both trim pots are turned full counter clockwise. Similarly, to set the MSSC address to servo bank 60−63, both trim pots are turned full clockwise.

3. Programming

The Servo4 module can independently control up to 4 servos. Each servo has 1) an enable bit and 2) a current position. The position is represented as an 8−bit number. Some experimentation may be needed to determine how the 8−bit numbers correspond to actual servo positions. All servos are initialized to have the enable flags *off*.

In MSSC (Mini Serial Servo Controller) mode, commands are of the form:

nn,ppp[cr]

where

nn
: is a decimal number between 0 and 63, inclusive, that specifies a servo to position.

ppp
: is a decimal number between 0 and 255, inclusive, that specifies the position that the servo is to go to.

[cr]
: is a carriage return character (e.g. decimal ASCII 13) that causes the command to take effect.

An example of a few MSSC commands are shown below, where there is an implicit carriage return after every line:

```
0,128
1,0
255,63,255
```

Note that the last command consists of three comma separated numbers; only the last two numbers are used. Lastly, the receipt of the carriage return automatically enables the servos.

The Servo4 commands are summarized in the table below:

Command	Send/ Receive	Byte Value								Discussion
		7	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
MSSC Command Execute (Carriage Return)	Send	0	0	0	0	1	1	0	1	Set position using previously sent MSSC servo number and position. Enable all four servos.
MSSC Next Number (',')	Send	0	0	1	0	1	1	0	0	Start next MSSC number.
MSSC Decimal Digit ('0'−'9')	Send	0	0	1	1	*d*	*d*	*d*	*d*	Next decimal digit of MSSC number, where *dddd*=0000 corresponds to '0' and *dddd*=1001 is a '9'.
MSSC Ignore	Send	0	0	*x*	*x*	*x*	*x*	*x*	*x*	If 00*xx xxxx* does not match one of the previous MSSC commands, it is simply ignored.
Set High	Send	0	1	*h*	*h*	*h*	*h*	*s*	*s*	Set high order 4 bits of servo *ss* to *hhhh* and set the remaining 4 low order bits to zero.
Set Low	Send	1	0	*l*	*l*	*l*	*l*	*s*	*s*	Set the low order 4 bits of servo *ss* position to *llll*.
Set Enable and Position	Send	1	1	0	0	0	*e*	*s*	*s*	Select servo *ss* and set its position to *pppppppp* and enable flag to *e*.
	Send	*p*	*p*	*p*	*p*	*p*	*p*	*p*	*p*	
Set Enable Flag Only	Send	1	1	0	0	1	*e*	*s*	*s*	Select servo *ss* and set its enable flag to *e*.
Read Position	Send	1	1	0	1	0	0	*s*	*s*	Return the current position *pppppppp* for servo *ss*.
	Receive	*p*	*p*	*p*	*p*	*p*	*p*	*p*	*p*	
Read Enable	Send	1	1	0	1	0	1	*s*	*s*	Return the enable bit *e* for servo *ss*.
	Receive	0	0	0	0	0	0	0	*e*	
Read Enables	Send	1	1	0	1	1	0	0	0	Return the enable flags *eeee* for all four servos.
	Receive	0	0	0	0	*e*	*e*	*e*	*e*	
Set Enables	Send	1	1	0	1	1	0	0	1	Set enable flags for all four servos to *eeee*.
	Send	0	0	0	0	*e*	*e*	*e*	*e*	
Read Current Draw	Send	1	1	0	1	1	1	*s*	*s*	Return the *aaaaaaaa* current draw for servo *ss*.
	Receive	*a*	*a*	*a*	*a*	*a*	*a*	*a*	*a*	
Shared Commands	Send	1	1	1	1	1	*c*	*c*	*c*	Execute shared command *ccc*.

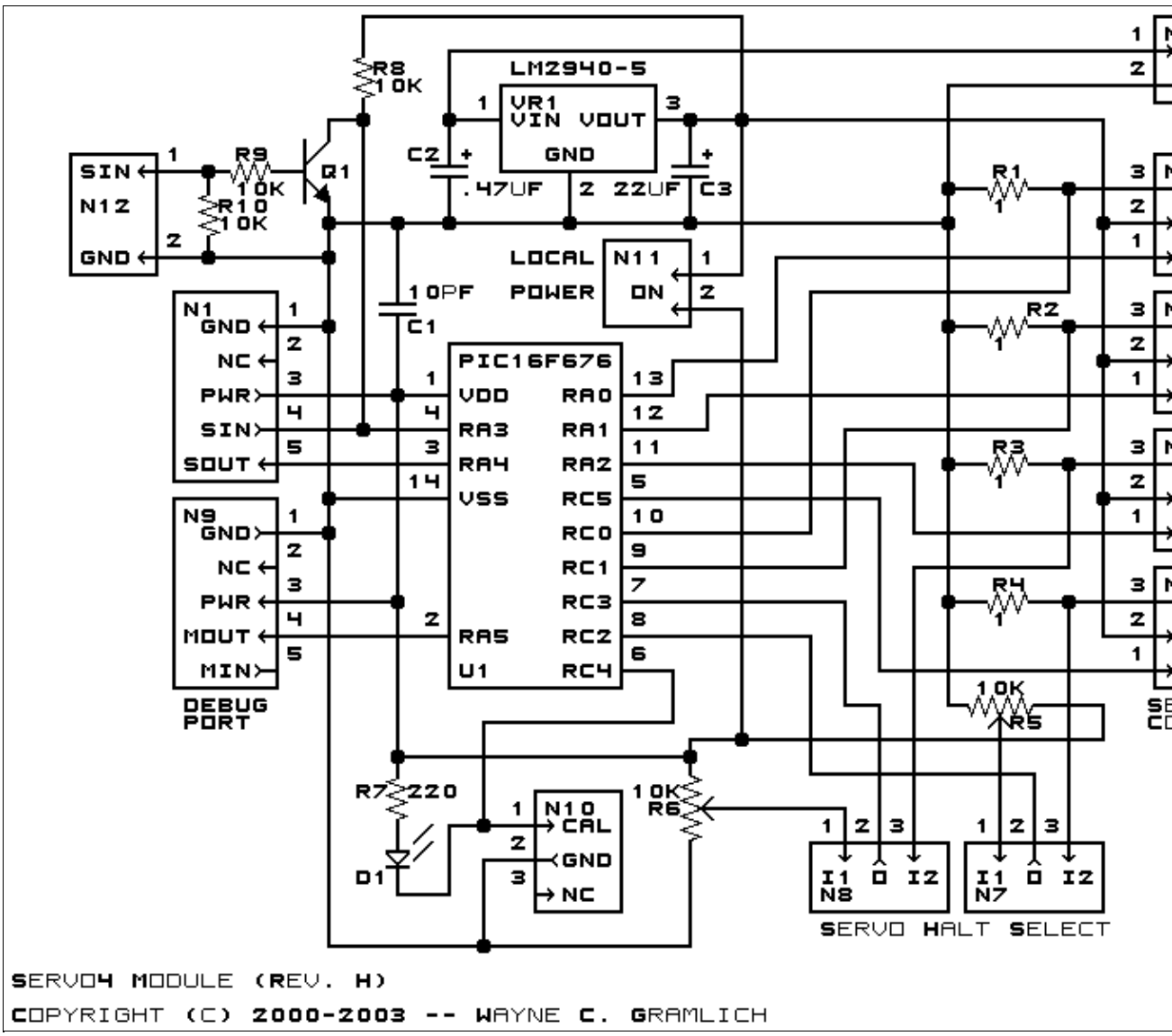
The Servo4 module does *not* know the minimum and maximum extent for each servo. This has to be determined by experimentation.

4. Hardware

The hardware consists of a circuit schematic and a printed circuit board.

4.1 Circuit Schematic

The schematic for the Servo4 module is shown below:



The parts list kept in a separate file −− servo4.ptl.

4.2 Printed Circuit Board

The printed circuit board files are listed below:

servo4_back.png
: The solder side layer is shown below:

servo4_front.png
: The component side layer is shown below:

servo4_artwork.png
: The artwork layer is shown below

servo4.gbl

The RS−274X "Gerber" back (solder side) layer.

servo4.gtl
The RS−274X "Gerber" top (component side) layer.

servo4.gal
The RS−274X "Gerber" artwork layer.

servo4.drl
The "Excellon" NC drill file.

servo4.tol
The "Excellon" tool rack file.

5. Software

The Servo4 software is available as one of:

servo4.ucl
The µCL source file.

servo4.asm
The resulting human readable PIC assembly file.

servo4.lst
The resulting human readable PIC listing file.

servo4.hex
The resulting Intel® Hex file.

6. Issues

The following software issues have came up:

- There is a request for enhancement from William Hubbard for the ability to set "set points" and a command to "return to set point". Reasonable request; it might even fit.
- William Hubbard is requesting the ability to delay servo changes until a single command is sent. Reasonable request; it might be a tight fit.
- Roger Gilbertson requests that the calibration pot be less sensitive. Sounds easy to do.
