Building, Testing, and Calibrating the SPS

Gustav Pettersson January 2018

# Building

1. The required components to build the SPS PCB are given in the “Components\_SPS\_V2-5” excel file and the PCB design plans are in the “SPS\_V2-5” Eagle files.
2. Install the standoffs.
3. The components are surface mount, but can be hand soldered without issue. First install all small components and packages, then the large ones.
4. The display and teensy are both mounted by placing .1” headers between the board and component (using only male headers and soldering directly is fine).
5. The TIP-122 should be first mounted to the heatsink and then placed as one component. Solder down the heatsink legs first, then the TIP-122.
6. Attach fan, noting that the fan is powered directly from the input voltage and must be selected accordingly. If less than half of maximum current is used, the fan may be omitted.

# Testing

1. Connect a micro USB cable to the Teensy, select the Teensy COM port and board in the Arduino IDE (follow instructions on [www.pjrc.com/teensy/teensyduino.html](http://www.pjrc.com/teensy/teensyduino.html) for details)
2. Upload the code “SPSTeensy” to the board through the Arduino IDE.
3. The display should come on, displaying the current and voltage (about 2.5V, 0A) at output.
4. Connect the TTL-232RG UART-USB adapter between the PC and SPS. Select the newly appearing COM port in the Arduino IDE.
5. Test that communication (over UART adapter, 115200 baud) works by sending “W” over the serial port and receiving “SPS”. (Use Tools > Serial Monitor)
6. Set a unique board ID by sending “S#” (where # is 1-255), retrieve by sending “I”. (This is stored permanently in the EEPROM.)
7. Mark the SPS with the board ID and input voltage.
8. Disconnect the Teensy USB cable.
9. Connect a 12V lab supply with a 100mA current limit. The SPS should turn on and draw around 50 mA. (Fan disconnected.)
10. Connect the SPS to a DC load set to 5V. Increase current limit on lab supply to 500 mA. Open the programme “SendDACvalue”, input the COM port used before (when setting the ID) and run. The programme should say “Connected to device with hardware ID #”. Connect fan.
11. (Recommended: Connect oscilloscope to transistor base to check for oscillations during the following steps.)
12. Enter 500 and send. The current should read about 260mA. Vary voltage 0-10V.
13. Increase current limit to 1A, send 1000. Should give around 530mA. Vary voltage 0-10V.
14. Increase current limit to 2A, send 2000. Should give around 1060mA. Vary voltage 0-10V.
15. Send 3000. Should give around 1600mA. Vary voltage 0-10V.
16. Increase current limit to 2.2A, send 3800. Should give around 2020mA. Vary voltage 0-10V.
17. Send 0 to disable the SPS. The testing is complete. (Use Ctrl-C to exit the programme.)

# Calibrating

## Voltage

1. **Connect a 24V fan. Use a lab supply with 24V input and 500mA current limit.**
2. Connect to DC load, set the load to constant voltage, 5V.
3. Use “SendDACvalue” to set the DAC to 500 (around 260mA).
4. Ctrl-C out, run “ReadADCvalues”, the terminal should start printing the voltage and current shown on the display, as well as “DAC: 500” and “ADC: xxx”.
5. Use the DC load to set the following voltages one by one: .1 .25 .5 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21. Write down the ADC value printed in Matlab for each.
6. Use Ctrl-C to exit the script and then run “ReadADCvaluesSTOP”.
7. Run “SendDACvalue” and send 0 to disable the SPS.

## Current

1. **Connect a 12V fan. Use a lab supply with 12V input and 2.5A current limit.**
2. Connect to DC load, set the load to constant voltage, 5V.
3. Open “SendDACvalue” and set the DAC to the following values one by one: 10 25 50 100 200 400 600 800 1000 1200 1400 1600 1800 2000 2200 2400 2600 2800 3000 3200 3400 3600 3800 4000. Write down the measured current on the DC load for each.
4. Send 0 to disable the SPS.

## Calculating

1. Input the values from the voltage and current calibration in the script “CalculateCalibrationPolynomials”. Run the script.
2. Check that the plots show straight lines and no outliers.
3. A “Calibration string” should be generated similar to: “1876.1 -3.0523 0.0053984 0.026077”
4. Open the Matlab variable “SPSCalibrationData” and add a new line. The first entry is the hardware id # programmed into the SPS and the following four is the calibration string:
   1. load(‘SPSCalibrationData.mat’)
   2. SPSCalibrationData = [SPSCalibrationData; # 1876.1 -3.0523 0.0053984 0.026077]
   3. Check that the new line was added correctly.
   4. save(‘SPSCalibrationData.mat’)