These are my notes on the hardware and schematic for my Arduino Peak Power Tracker Solar Charger project. These notes are based on my schematic ArduinoSolar.pdf on my website www.timnolan.com.

This project is based on the open source hardware platform of the Arduino Duemilanove. I got mine at Sparkfun (www.sparkfun.com) but there are other places you can get them (www.makershed.com). The Arduino webpage that includes links to the Duemilanove schematic is at http://arduino.cc/en/Main/ArduinoBoardDuemilanove.

The Arduino Duemilanove takes care of the microprocessor part of the project. I built the charger part of the project on a ProtoShield Development Kit that I also got at Sparkfun. This is a small prototype board that has connectors to mate directly to the Arduino processor board and bring all the signals up to the prototype space. I was able to fit the complete charger circuit into this prototype space. If you take a look at the photos on my website you can see how the components are placed on the ProtoShield. I've also included the parts list for this project with the Digikey and Sparkfun parts numbers in the file ArduinoSolarPartsList.txt on my website www.timnolan.com.

The charger circuit is basically a DC/DC converter (in a buck configuration) controlled by the software in the microprocessor. The software figures out the voltage of solar panels where the Peak or Maximum Power is produced and controls the DC/DC converter to match the solar panel voltage to the battery voltage. The processor controls the DC/DC converter by generating a PWM signal that switches the MOSFETs at a 50kHz frequency. The transfer ratio of voltage in vs. voltage out is based on the duty cycle of the PWM signal.

Looking at the schematic ArduinoSolar.pdf the connector (J1) for the Solar Panel input is in the upper left corner. There is no protection for polarity reversal so make sure that the solar panels are connected correctly. The solar panel input voltage is connected to the Vin or Raw input to the Arduino board which goes to a voltage regulator to generate 5V to run the processor. The 5v also comes back to the ProtoShield board. The solar panel ground input is connected to the ground of the ProtoShield and Arduino processor board. The solar panel input voltage is divided down by R4 and R5 and sent to the Analog_0 input of the Arduino to be read by the processor.

The solar input current is read by R1 the .005 ohm current sense resistor that generates a very small voltage when the current flows. That voltage is picked up and amplified by IC1 the MAX4173H which is a high side current sense amplifier. The MAX4173H has an amplification of factor of 100. So if 5A is flowing through R1 it generates 5A x .005 ohms = 0.025V x 100 = 2.5V to be read by the processor. The MAX4173H only comes in surface mount SOIC8 package so I used a little conversion board from Sparkfun. I soldered the MAX4173H to the board and then used the 8 pin dip socket to plug it into the ProtoShield board.

C2 is the input filter capacitor that smooths out the input current pulses. Q1 is the blocking MOSFET that keeps the battery power from flowing back into the solar panels at night. Normally this is done by a diode in the power path but a since all diodes have a voltage drop a MOSFET is much more efficient. Notice that Q1 is turned around so the intrinsic MOSFET diode does not conduct. Q1 turns on when Q2 is on from voltage through D2. R3 drains the voltage off the gate of Q1 so it turns off when Q2 turns off.

Q2 is the main switching MOSFET for the buck converter and Q3 is the synchronous switching MOSFET. The MOSFET are driven by IC2 which is an IR2104 MOSFET driver. The IR2104 takes the PWM signal (Digital_9) from the processor input on pin 2 and uses it to drive the switching MOSFETs. The IR2104 can also be shut down with the control signal (low on Digital_8) from the processor on pin 3. Since Q2 is an NFET it needs a gate drive voltage that is 10V higher than the source voltage which is the solar input. So the IR2104 uses a charge pump circuit made of D3 and C4 to boost the gate drive voltage to turn

on the high side MOSFET. This charge pump circuit only works when the MOSFETs are switching. The software keeps track of the PWM duty cycle and never allows 100% or always on. It caps the PWM duty cycle at 99.9% to keep the charge pump working.

D1 is an ultra fast diode that will start conducting current before Q3 turns on. It is supposed to make the converter more efficient but it may not be necessary. L1 is the main inductor that smooths the switching current and along with C3 it smooths the output voltage. Since the DC/DC converter is switching at 50kHz the 33uH value of the inductor should be sufficient. C7 and R10 are the snubber network used to cut down on the ringing of the inductor voltage.

To measure the battery voltage R6 and R7 make up the voltage divider which feeds the signal to Analog_2 to be read by the processor. J3 is the battery connector, there is no reverse polarity protection so make sure the battery is connected correctly. I did not have space to fit a fuse so make sure you put a fuse in your battery wiring harness. Batteries hold a lot of energy and you can start a fire if something goes wrong and you do not have a fuse. Also you might want to put a diode in battery wiring harness during development. If you look at the schematic you can see that if Q3 ever stays on for any length of time that it is a dead short across the battery. This would not happen in normal operation but if you have a software bug or you stop the system to load new software it might end up with Q3 on. A diode would stop this from happening but you would lose efficency in the diode voltage drop. Once you have the software worked out this diode is no longer necessary. I've burned up my fair share of MOSFETs before I figured this out.

I've included the two LEDs and switch that are on the ProtoShield board in my schematic. I use the one LED JC2 as a heartbeat indicator in my system.