

Solar Powered Lead Acid Battery Charger

Final Report
Spring Semester 2007

By
Andrew Giddings
Scott Johnson
Jake Shelton
Alex Wolke
Steve Searcy
Doug Antioco

Prepared to partially fulfill the requirements for
EE402

Department of Electrical and Computer Engineering
Colorado State University
Fort Collins, Colorado 80521

Report Approved: _____
Project Advisor

Senior Design Coordinator

ABSTRACT

Our problem is the ever rising costs of non-renewable energy from natural resources and the resulting energy crisis facing our planet. Solar energy is not a revolutionary technology by any means and has long been criticized for its inefficiency and other shortcomings. Modern advances by companies such as DayStar Technologies have made great gains towards reducing the power conversion inefficiencies and have succeeded in creating a solar cell foil that is close to 20% efficient as opposed to the traditional 10% and have greatly reduced costs with CIGS technology [1]. Our product addresses the other implementation difficulties related to solar energy, primarily, “What do we do when the sun goes away every night”. We ought to store the energy right? And how do we propose to do that? Well with batteries, of course! But its not that simple, the battery currents and voltages need to be regulated to provide optimum power transfer to the client because efficiency reduces cost and high cost is the last major road block in making solar a feasible energy alternative. The heart of our product then is microprocessor control of the stored energy.

The approach taken by our team of engineers was essentially an embedded systems approach. We needed to interface the microprocessor which we programmed, with some current/voltage limiting hardware, a buck regulator, opto-isolators, and various other necessary components to effectively read the currents coming off the panels and entering the batteries, and those currents leaving the batteries to power some desired application.

It's true our project as described is not breaking any real ground, however, most products meant to regulate a battery voltage are completely analog and offer very little control to the user. They are also not digitally self-optimizing which is a possibility for us. As far as future improvements and evolutionary capability we are offering an extra buck regulator component and a few pins for a future expansion solar array as technologies in this sector improve to generate even more power. We suspect we can be competitively priced, if not cheaper than our closest competitors who don't offer expansion options and digital flexibility.

TABLE OF CONTENTS

Title	1
Abstract	2
Table of Contents	3
List of Figures and Tables	4
I. Introduction	5
II. Summary of Previous Work	6
III. Design	7
A. Hardware	7
B. Power train	19
C. Software	22
IV. Fabrication	27
V. Budget	29
VI. Conclusions and Future Work	30
References	32
Bibliography	32
Acknowledgements	33
Appendix A	34

LIST OF FIGURES

Figure 1	Old Schematic	7
Figure 2	Original Voltage Meter	8
Figure 3	New Voltage Meter	9
Figure 4	Original Current Meter	9
Figure 5	New Current Meter Circuit	10
Figure 6	Fan and Temperature Sensor Circuit	11
Figure 7	Boost Voltage Circuit	11
Figure 8	New Pin Layout	12
Figure 9	RS232 with Isolation-Couplers	13
Figure 10	Buck Regulators	14
Figure 11	New Microprocessor (Pic18F2620)	14
Figure 12	Switches and Flash Circuit	15
Figure 13	Battery Current Meter	15
Figure 14	Voltage Sources	16
Figure 15	New Schematic	17
Figure 16	Simplified Buck Regulator	20
Figure 17	Typical Connection of the IR2110 regulator chip	21
Figure 18	Schematic diagram of the IR2110 regulator chip	21
Figure 19	MPLAB ICD 2 interface connector	23
Figure 20	4-bit Initialization	24
Figure 21	Chassis with major components	27
Figure 22	Clear Plastic Pannel Cover	28

LIST OF TABLES

Table 1	Character Bit Stream Table	25
Table 2	Budget Costs	29

I. INTRODUCTION

Harnessing solar energy has been the holy grail of renewable energy research for some time now. Photovoltaic solar cells convert light energy to high voltage and low current which can be manipulated to provide power to our modern electrical devices and homes. With the ever rising costs of fossil fuels, the need for an efficient and affordable solar energy system has never been greater. This is why our market of interest focuses on the residential community to help offset the cost of utilities while striving to be environmentally conscientious. One group, Energy[R]evolution claims, “The sunlight which reaches the earth’s surface is enough to provide 2,850 times as much energy as we can currently use. On a global average, each square meter of land is exposed to enough sunlight to produce 1,700 kWh of power every year.”[1] These are encouraging statistics when coupled with advances in our field, including the conversion efficiency of the solar cells, with companies like DayStar Technologies and ENREL developing a solar foil which is nearly 20% efficient up from 10% in the previous generation [2]. However, the energy from this foil would be useless if it could not be stored in some manner for use at another time when the sun may not be as readily available (i.e. batteries). The field of solar energy engineering requires products like ours to maximize this power storage and monitor the power levels.

Our project is essentially all of the technology that must be in place in order to interface the power-generating solar cells to the power-storing batteries. In order to accomplish this we must choose a microprocessor, generate its necessary control signals, read in the voltages and currents coming off both the batteries and panels, and use a buck regulator to manipulate the voltages and currents according to an optimization scheme regulated by the microprocessor.

With the guidance of our industry expert, Jack Gilmore, we plan to demonstrate this controller, at scaled down operating levels, at Engineering Days in the spring of 2007. We intend to have this project demonstrated outside with bona fide solar panels and not simulated ones. Although this is what we plan to accomplish for the end of the year the overall goal for this assignment is to implement this technology into a household environment.

The following is a mid-project review of what we have accomplished so far, the problems we have encountered, the possible solutions and our plans for next semester. In chapter II we'll start with a recap of the previous prototype; its successes, and ultimate failures. Then in chapter III we will elaborate on the technical aspects of the hardware, power train and software of our design. We'll discuss hardware design in detail, explaining what we added from the previous year's effort and what we removed, the options available with the extra pins on our microprocessor, isolation of different sections of the circuit to provide protection to the more sensitive components and explanation of the circuit functions. Next, we'll discuss the hardware involved in the buck regulator circuit in more detail, as well as discuss our decision to utilize the rectifier chips in place of some other components. Next, we'll focus on the software side of our project, what microprocessor we chose and why we choose it, writing to the LCD display and the challenges we encountered in successfully compiling and downloading to the microprocessor. Chapter IV will be a brief discussion on the budget of our project and finally in chapter V we will talk about the future work that needs to be done to achieve our expected goals.

II. REVIEW OF PREVIOUS WORK

What follows is an excerpt from an e-mail from Jack Gilmore at Advanced Energy who spear-headed this project previously and helped develop the original prototype:

“The design started off several years ago with just a relay driven voltage regulation type of charger like in a car. As soon as the first maximum power point charger appeared on the market, I realized that the charger could be improved by adding a buck regulator and regulating the voltage on the panels to the optimum for the panels (16.5V). An improvement of tracking the maximum power point would take an additional investment of a microprocessor control to vary the duty cycle of the buck regulator. This was tried two years ago. The failure of the

With our prototype we believe we have corrected the memory insufficiencies and lack of stack depth by changing our microprocessor to the more powerful PIC18F2620. Also by splitting the bulk project into three smaller sections: hardware, power train, and software each consisting of two engineers we hope to share the workload evenly, and thus be able to place greater effort on the software, to bring this project to fruition.

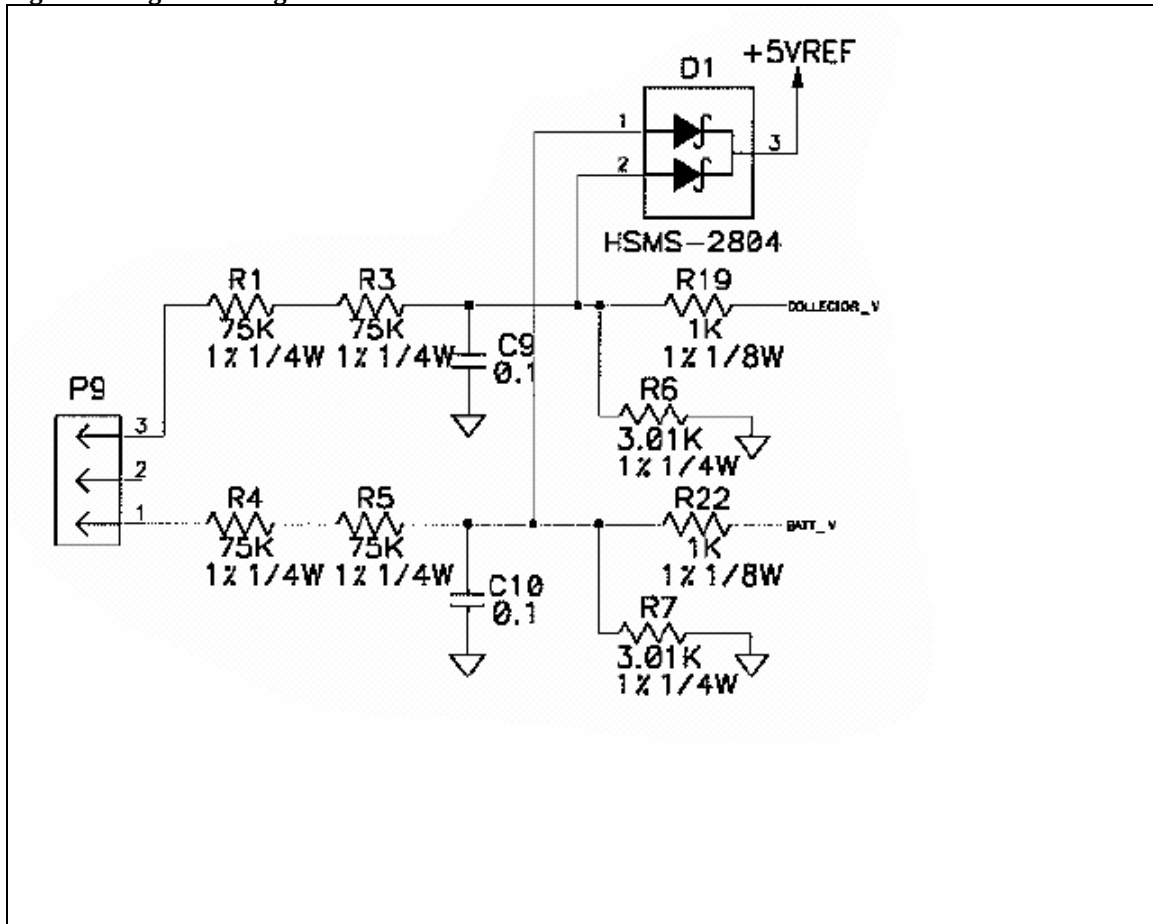
A. Hardware

Figure 1 Old Schematic



The very first change we made was to put in a better microprocessor. The details of which will be covered in the Software design section. It then became desired of us that we should add the capability to the original circuit to be able to work with an additional power source, such as an additional set of solar panels. The first step in doing this was to add in a second voltage meter to detect the additional voltage from the second power source. Here is the original voltage meter circuit:

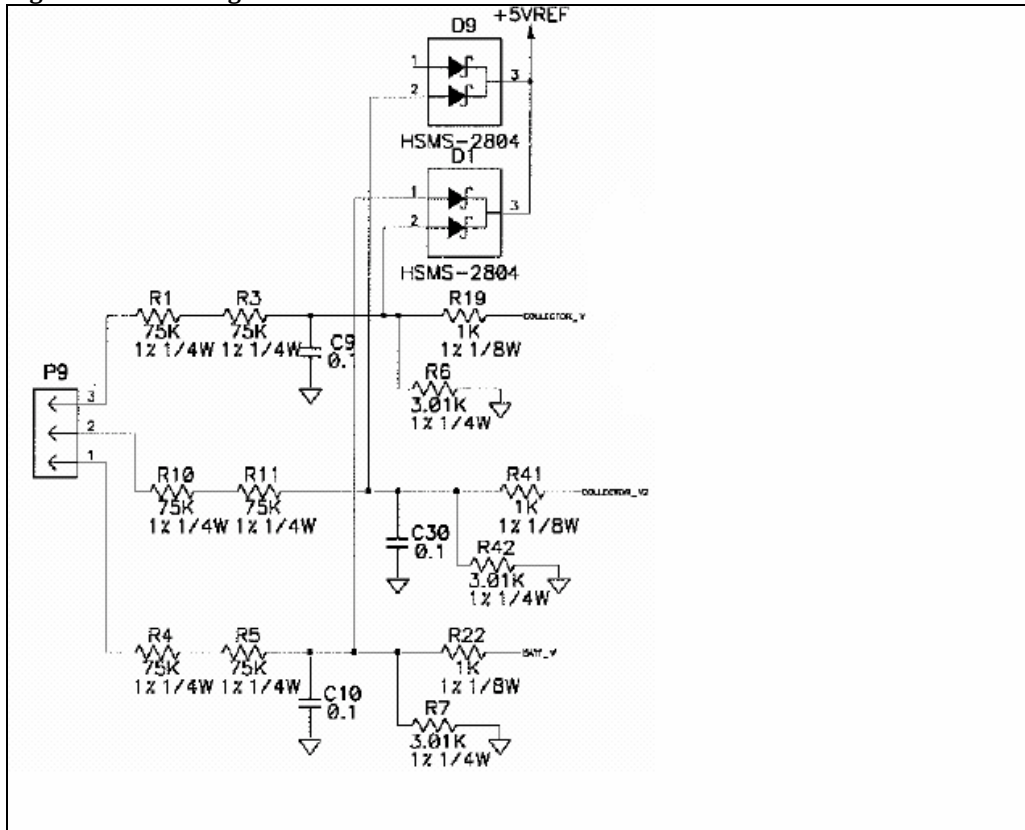
Figure 2 Original Voltage Meter



To do this we used the extra external pin on P9 and added in a duplicate voltage detecting circuit line in with the two existing ones. However as you can see D1 only allows for two zener diode connections from the meters to +5VREF. Since we knew we already had multiple two zener components, we decide to add in an additional two zener, HSMS-2804, part to accommodate the third meter.

Here is the new set of voltage sensing circuits:

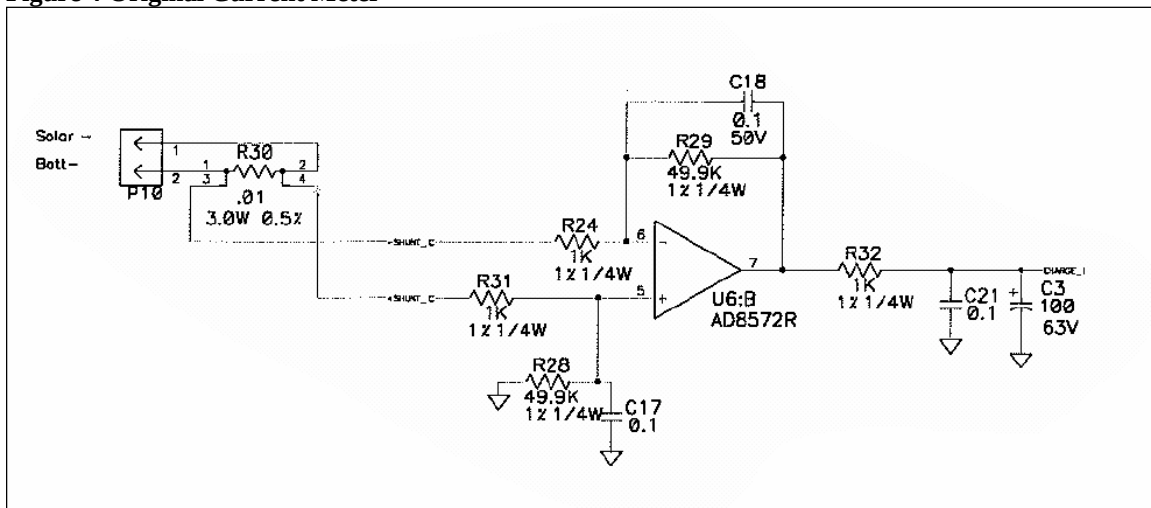
Figure 3 New Voltage Meter



The next part to add was an additional current meter for an additional power source.

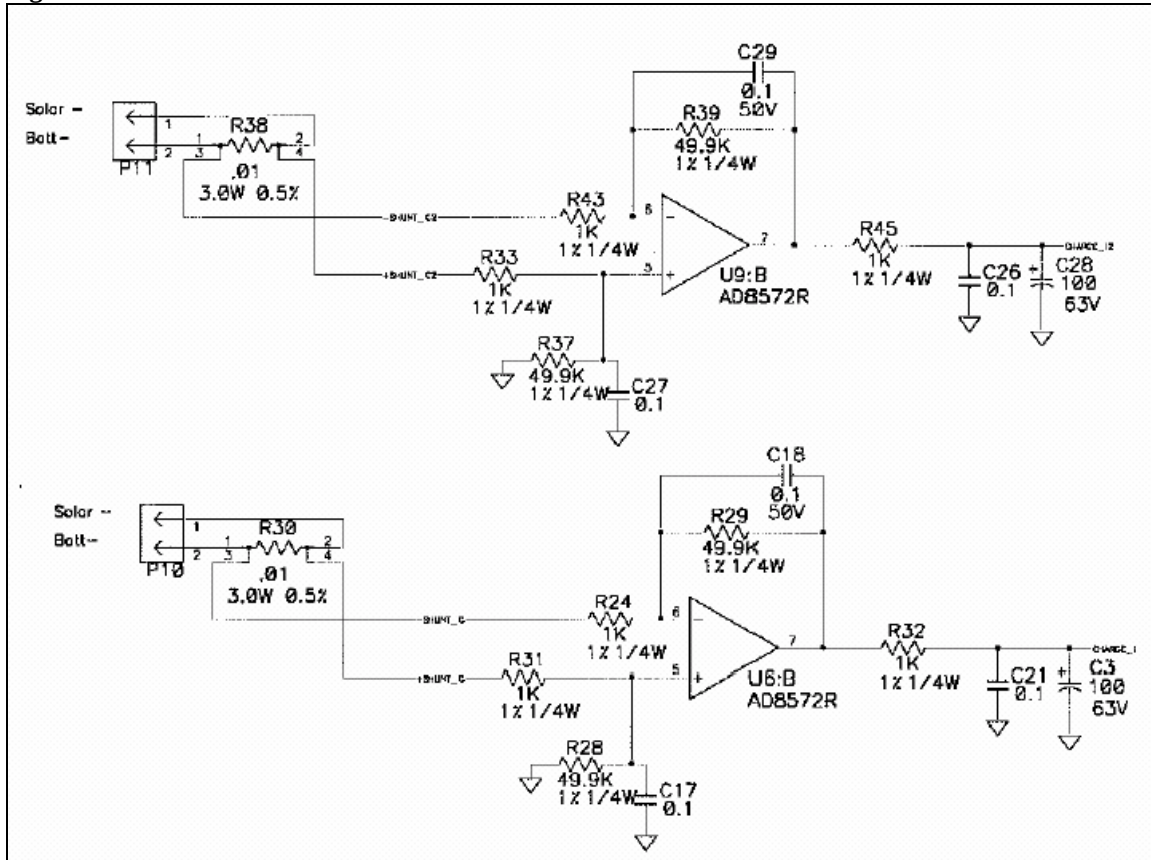
Here is the original Current meter:

Figure 4 Original Current Meter



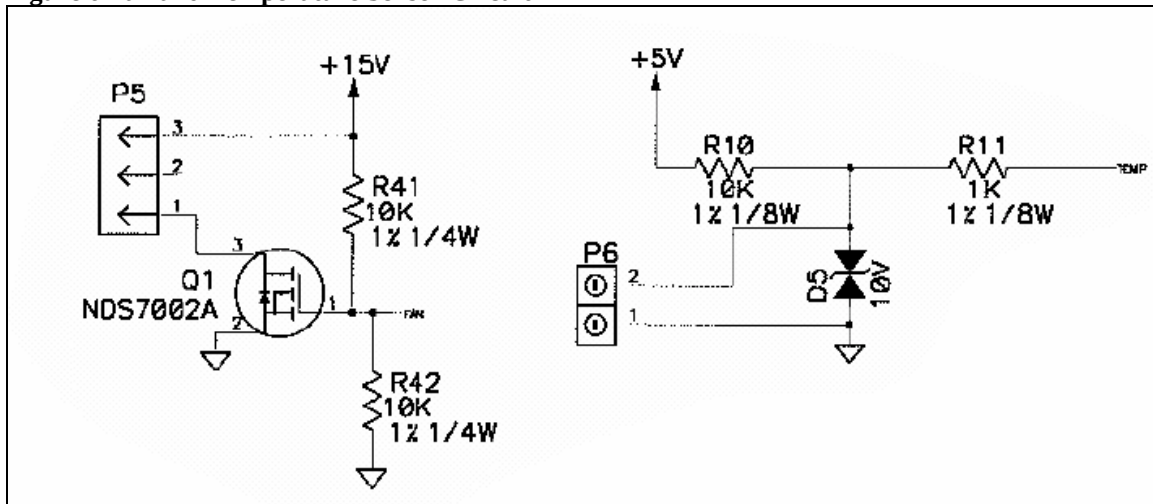
To implement the second current meter we simply had to add in an additional external pin set, and a duplicate meter circuit to the schematic. Here is the new current meter circuit:

Figure 5 New Current Meter Circuit



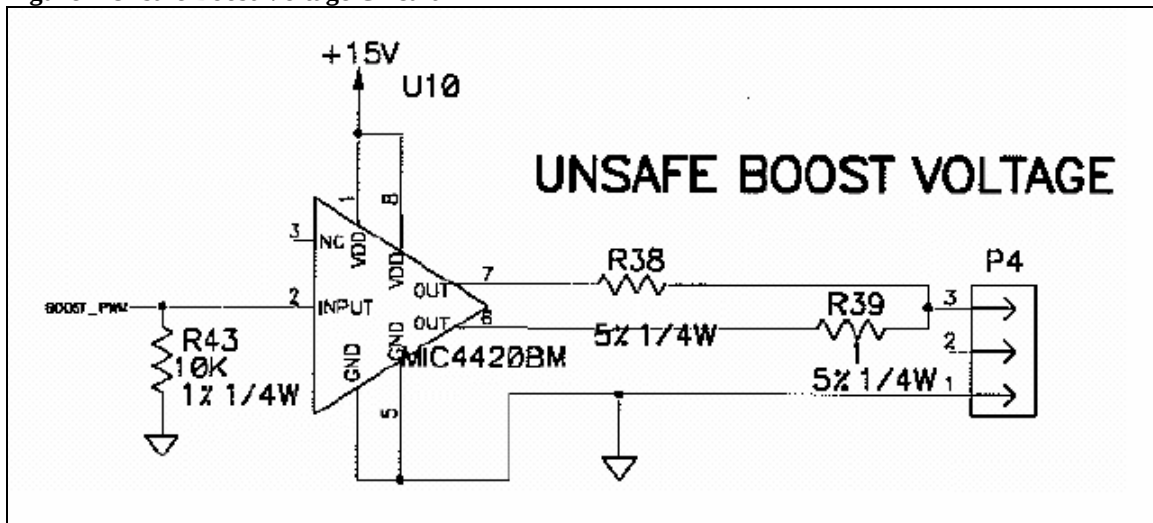
The next implementation step was to add in an additional buck regulator, which will be explained in the Power Train Design Section. The last step to allow for an additional power source was to take the voltage, current, and pulse width modulation (Buck Regulator) pins from the additional parts and connect them to the new processor. However we did not have enough empty pins to start with, so we first had to see if we could remove any unnecessary circuit components. The first part we decided to remove was the external cooling fan. With correct implementation of the circuit via software we would not need a fan to cool it. This actually gave us 2 pins to use, since the fan also had a temperature sensor output to let it know when to turn on. Here is the fan and temperature sensor circuit:

Figure 6 Fan and Temperature Sensor Circuit



The second part we removed was the unsafe boost voltage circuit. We felt that this component was unnecessary, as we could use the power train to effectively control the voltage. Here is the unsafe boost voltage circuit:

Figure 7 Unsafe Boost Voltage Circuit



Now that we had three free pins to use on the processor, we just relabeled the pins, putting the duplicates next to the originals like this:

Figure 8 New Pin Layout

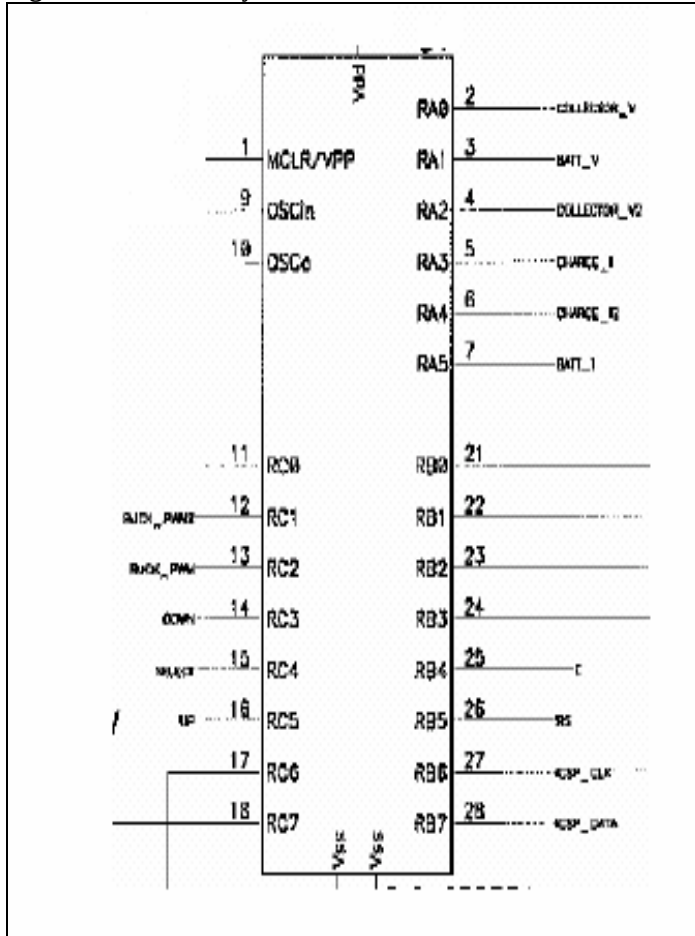
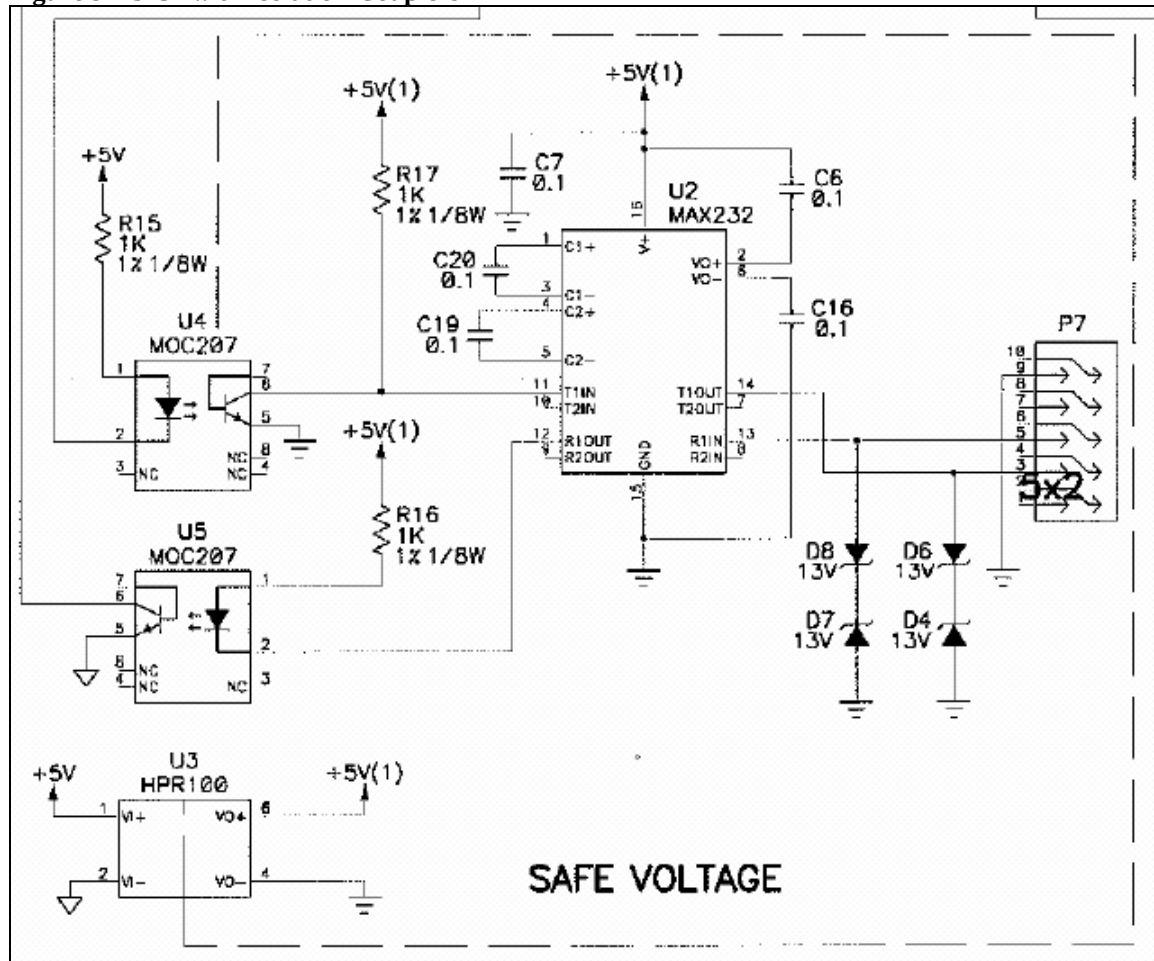


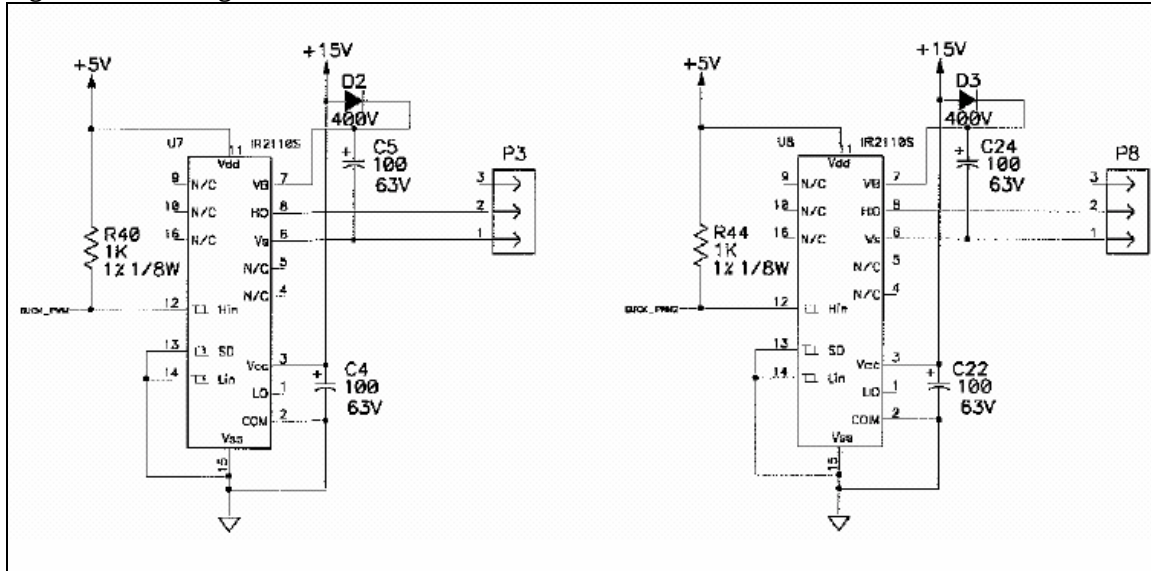
Figure 9 RS232 with Isolation-Couplers



In order to communicate with our processor we will need some hardware safety measures to accompany the components needed to connect the RS232 port for an outside computer in which to monitor the status of operation. For this situation iso-couplers were used so we can transfer signals from the processor to the computer and back while keeping both circuits electronically isolated. A separate, isolated power source is connected to MAX232 chip which is also needed because the voltage standards used in the processor differ from that of the computer, for example transistor to transistor logic coming off the processor uses a logic 1 specify a line voltage of +5 volts and a logic 0 is a line voltage of 0 volts, whereas in the RS232 the logic 1 refers to a voltage of +5 to +15 volts and logic 0 refers to -5 to -15 volts. Applying the MAX232 bridges these differences with the assistance of a few external 10 microfarad capacitors. Another chip called the MAX233

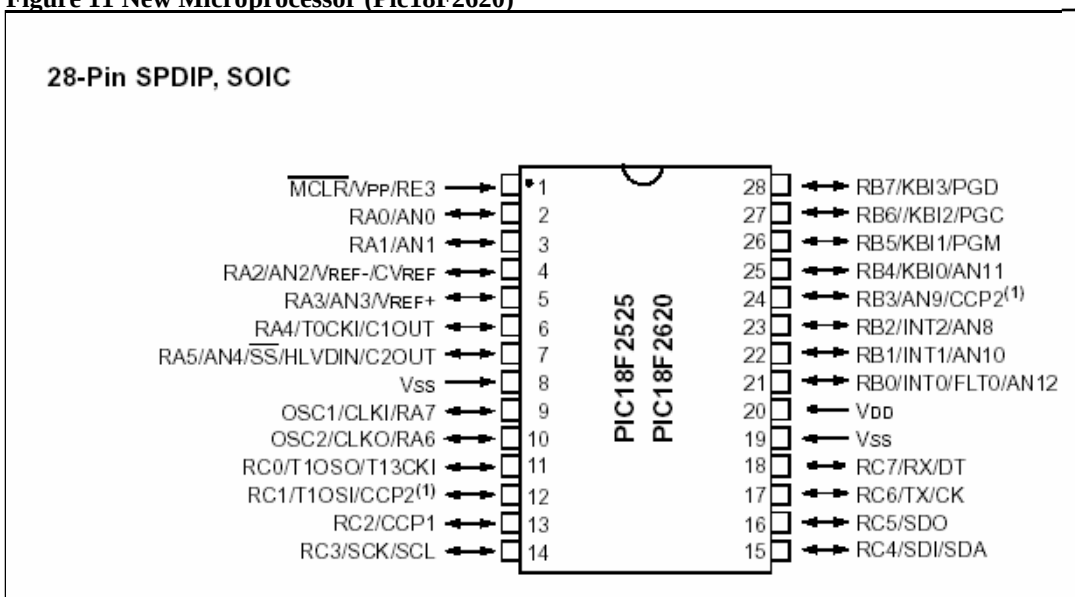
can be used with no external part but the size of the device is larger and there is a higher price tag [3].

Figure 10 Buck Regulators



The buck regulators are the one of the workhorses of our project and the involve converting high voltage low current into low voltage high current for charging the battery. We have been able to replace the previous buck regulator with one that has been implemented on a chip (IR2110SPbf), which is opto-isolated so there is no need for external iso-couplers. More about the buck regulator will be discussed in the power train section of this report.

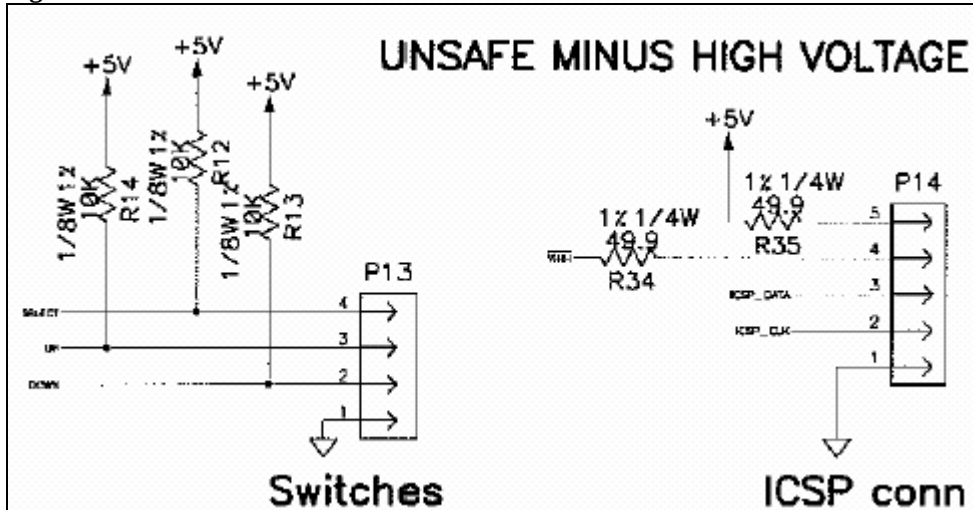
Figure 11 New Microprocessor (Pic18F2620)



[4]

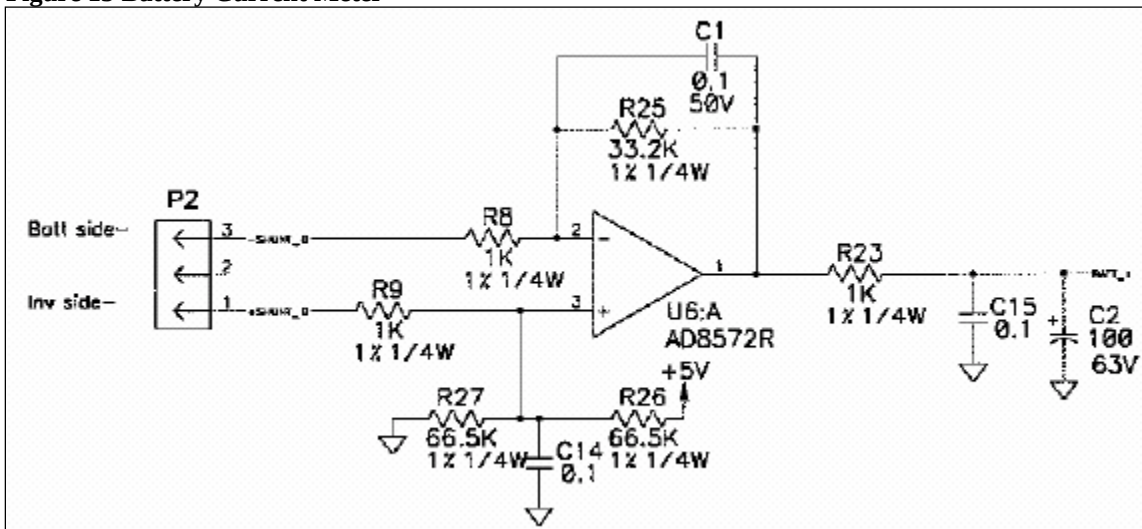
The new microprocessor is more capable of performing the actions needed to make this project become a success. This processor will be discussed more in depth within the software section.

Figure 12 Switches and Flash Circuit



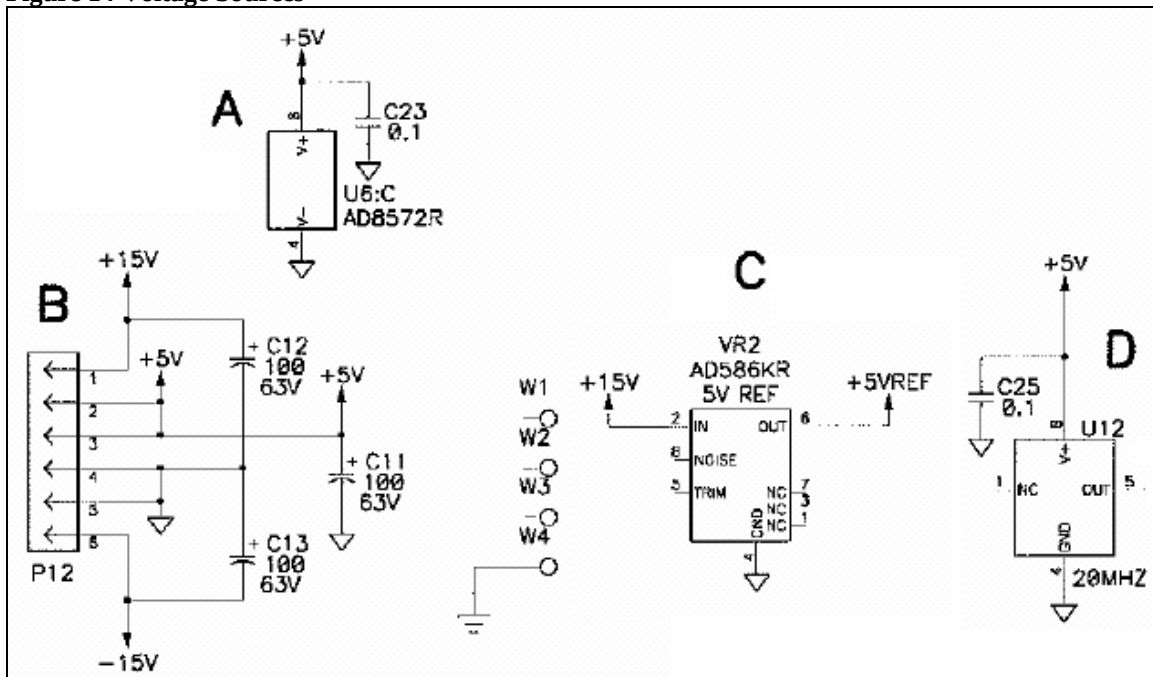
These components have kept from the original design, the switch circuit is for outside control of the device and is comprised of three options, up, down and select for use of monitoring the voltages and currents on the liquid crystal display as well as using a new feature which is to strike the battery with a higher voltage than the charging voltage, but only when it is fully charged, which has been found out to prolong the battery life. This method of charging will also be discussed in the software section as well. The Flash circuit is the component needed for programming the processor.

Figure 13 Battery Current Meter



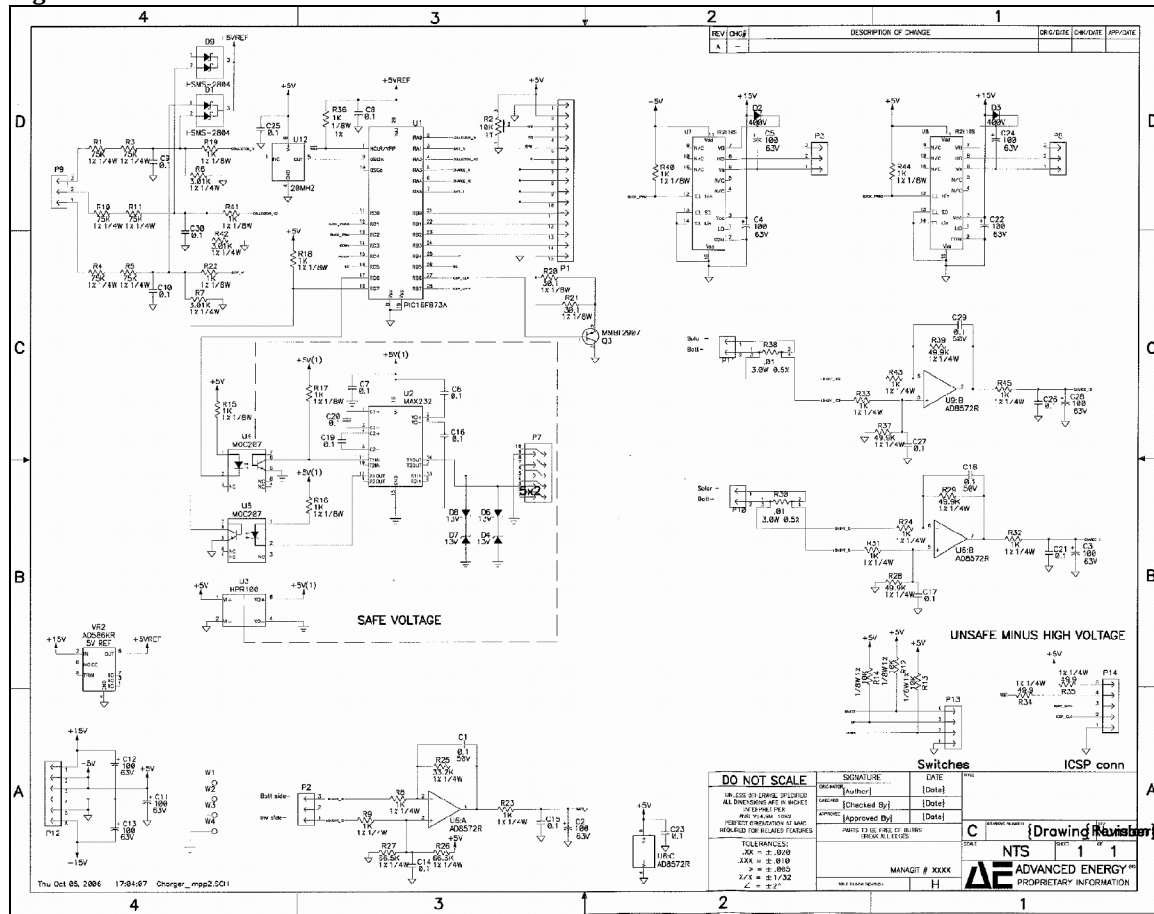
Like in the original design, we have a circuit to measure the value of current coming from the battery. The remaining figures are other smaller but essential parts of our device, the component A is a voltage source that will be used for each of the operational amplifiers that are being used to measure the currents coming off either the solar panels or the battery. Component B is the voltage sources to be used within the layout. Component C is a high precision 5-volt reference used to power the processor, the circuit with components listed W1 through W4 are the grounds to be used for the layout. Component D is the clock that will be used for the pulse width modulator.

Figure 14 Voltage Sources



Here is the final schematic we will be using for the remainder of the project:

Figure 15 New Schematic



Now that the design was complete at the end of the first semester, we will now cover the process of building and debugging our circuit. We will include both our successes and our failures in building the circuit. At the start we had to take our circuit, which was already in the necessary CAD program covered in the original budget section, and make sure everything was acceptable before printing the circuit. To do this we mainly had to make sure the software had decided to run all the traces with correct spacing for higher voltages. Once this was correct we proceeded to actually print the circuit. The blank boards used in this prototyping machine are a piece of dielectric with solid copper on each side. To print the board we place the blank in the machine (both of which are covered in the original budget), and set the etching bits to match the necessary

spacing for our traces. Once it was set up we simply loaded our design into its memory and pushed start. To create the board it simply used the etching bits to clear away the unused part of the copper. Since this board is for prototyping, it is not the fastest of board making machines, as it took nearly four hours to create our board. Once the board was complete we checked it for errors, and found only one where part of the connection pad for a capacitor had some copper missing. Jack Gilmore discussed that this could be fixed just by realigning the capacitor, so we did not need to recreate the board. With a completed board we were now ready to start populating the board with parts. To do this we went through all of AE's part bins to compile our list of necessary parts. With the parts gathered we were ready to start soldering. However neither of the hardware group members had ever soldered a part before, so the second step in populating the board was to learn how to solder. We learned to solder from Jack by de-soldering and re-soldering some of the motherboard resistors on an old board, which took quite some time, learning the correct amount of solder to be placed, and learning some tips to increase the speed of soldering a part. It would be good to note that overall the parts were fairly inexpensive, under \$10, not including the chip which is covered in the original budget, and the soldering tools were about \$40 total. Once we knew how to solder the parts correctly, we now proceeded to add parts to the board, smallest first. For new solderers, this became a lengthy process, taking about two weeks to get about half the board done. At about this point our board was misplaced. We had been storing the board with Jack Gilmore, and through relocation of offices, our board was lost and never found. So another board was etched and soldered.

The replacement prototype was completed the week before engineering days, hardware and software debugging was necessary. A meeting was set up at AE to debug the circuit and we were able to debug the circuit using current/voltage flow techniques. Our advisor at AE was going to be unavailable for the following week so we obtained the board for the software group as we were fairly sure the hardware was debugged. After realizing that the other group was not getting expected results we had a full meeting to look at the circuit, which turned out to still have software and hardware incompatibilities. The new LCD screen did not work with the fresh circuit, and it also did not work with the old circuit so we used the other LCD which worked with the older circuit and it also did not register on the new circuit either. Therefore we understood that there were still at

least hardware problems involved. We think the LCD problem might be involved with the LCD connection but we were not confident if that was the case or how to fix it. We were at least about to find out that the pulse width modulator was functioning correctly and that we were able to communicate to the processor.

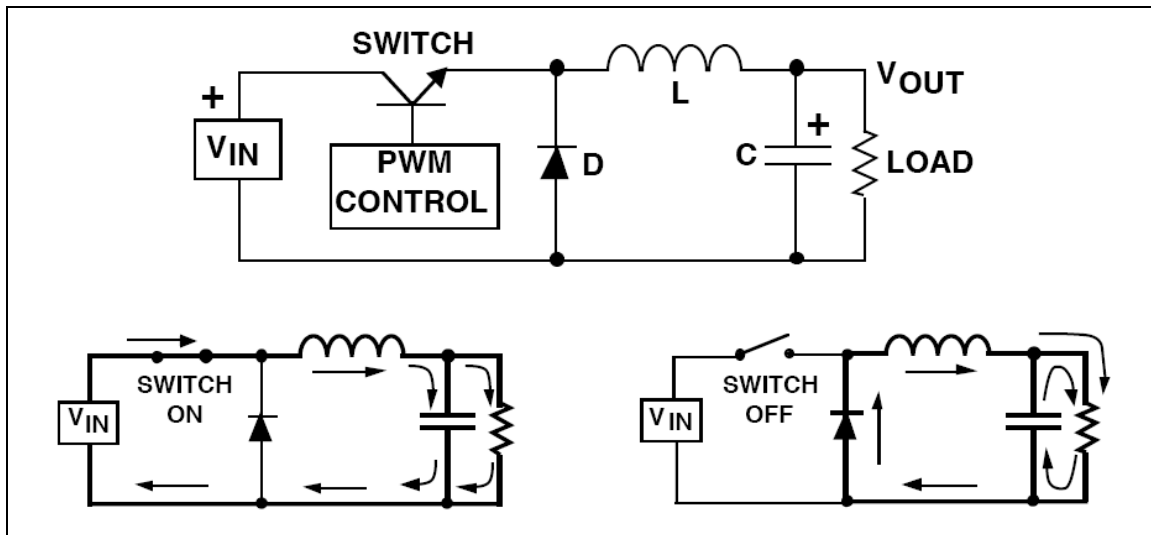
Future work that will be needed will be a more thorough debugging of the circuit and LCD connections, establish the RS232 link and the final assembly.

B. Power Train

The power train portion of our project is arguably one of the least time-consuming components but nevertheless, is most integral to the controller's performance. Here, we have one fundamental goal, to take the relatively high voltage, low current being generated on the panels, and convert it into a high current, low voltage configuration in order to charge the lead acid batteries. The mechanism by which this is accomplished is a device known as a buck regulator.

A buck regulator consists of an inductor, a capacitor, a diode, and a transistor that is used as a switch. The switch is controlled by the PWM (see Appendix A), which is generated by a clock on the microprocessor. When the square pulses of the PWM are high, the transistor is turned on, and the high voltage from the panels is applied to the inductor, generating a current through it (see Figure 16). This current is delivered to the load and charges the capacitor. Then when the square pulse is low, the transistor turns off, and the voltage across the inductor is removed. However, current in an inductor can't change instantaneously so the high voltage reduces to maintain this current. Eventually the negative voltage at the input of the inductor drops below the threshold of the diode, turning it on and discharging the inductor current through the load. During this off time the capacitor also discharges through the load contributing some to the total load current. By varying the duty cycle of the clock we can control how much of the ~ 380 V the load sees, which will be ~ 48 V if 4 series 12 V batteries are to be charged. The capacitor acts as a filter on the 380 volt pulses, minimizing the decay between pulses (inductor current ripple), and generating an approximate DC output voltage [6].

Figure 16 Simplified Buck Regulator

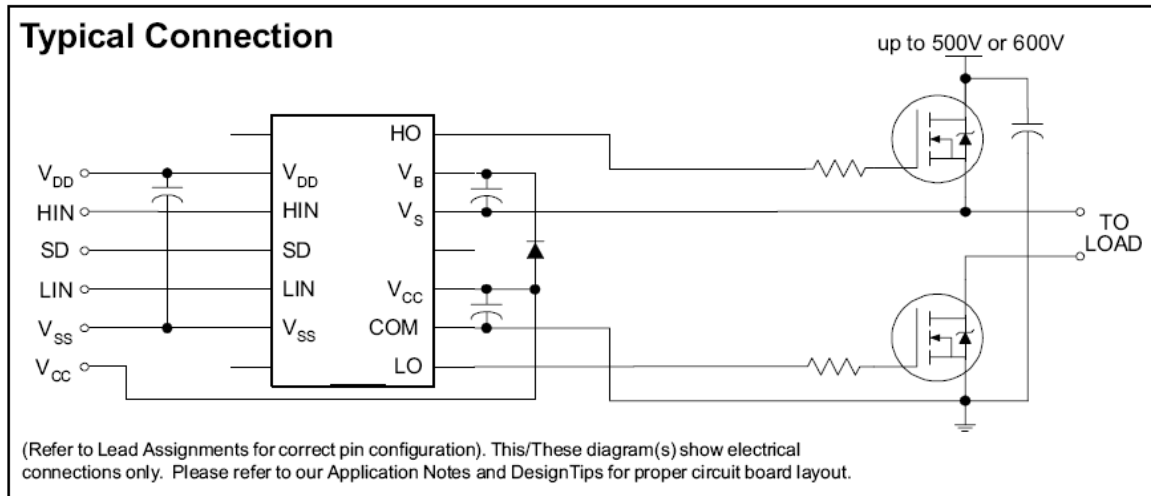


In the previous year's effort (see Figure 1), the MIC4420BM component served as the switching transistor and the LM78L05 acted as the diode/resistor current shunt in the Buck regulator just discussed. In our design, however, we used the APT2X60D60J dual rectifier diode isotop package for the diode and the APT5010JPR isotop package was our transistor switch. Both packages were rated for power systems use at 500V. We also replaced other components that tackled opto-isolation of the dangerous power levels of the buck with an International Rectifier IR2110 chip (see Figures 17-18) [7]. This freed up some space on the board and allowed us to include a second IR2110 for an auxiliary set of expansion panels relatively easily. The inductor, which is located off-board was rated for approximately a hundred amperes in order to support such massive swings in voltage. The capacitor and resistor you see in parallel at the back end of the buck regulator represent the battery.

We also realized the need for shaping capacitors on the front end of our buck regulator where the panels connect to the gate of the switch. We discovered that a large foil capacitor could smooth out our ripple current and withstand the 30A max current, while an array of electrolytic caps (about 60 μF) would smooth out the large swings in voltage (0-380 V) stemming from alternately connecting and disconnecting the panels. Our research showed no single capacitor that would satisfy these two criterium so we connected both types of caps from the panel V_{IN} node to ground in parallel. When it came to assembling this buck regulator circuit we used a large heat sink donated from Advanced Energy to mount the FET and diode on and we cut up rolls of sheet copper in

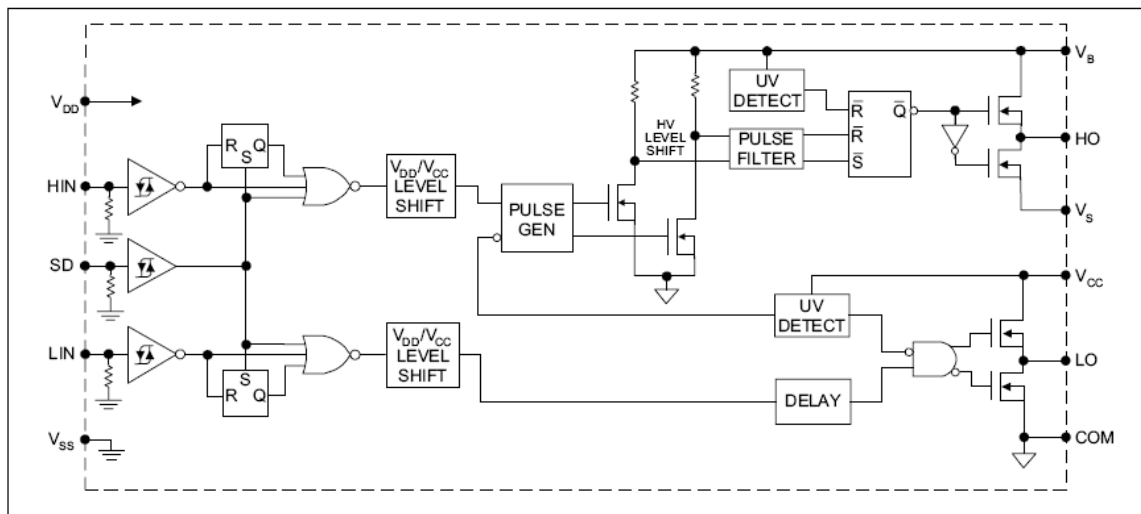
wide strips to make the connections between the caps, FET, and diode because we wanted to minimize stray inductance in these time constant critical switching components. The other connections to the inductor and external components were made with regular larger gauge wire where stray inductance wasn't an issue.

Figure 17 Typical Connection of the IR2110 regulator chip



[7]

Figure 18 Schematic diagram of the IR2110 regulator chip



[7]

C. Software

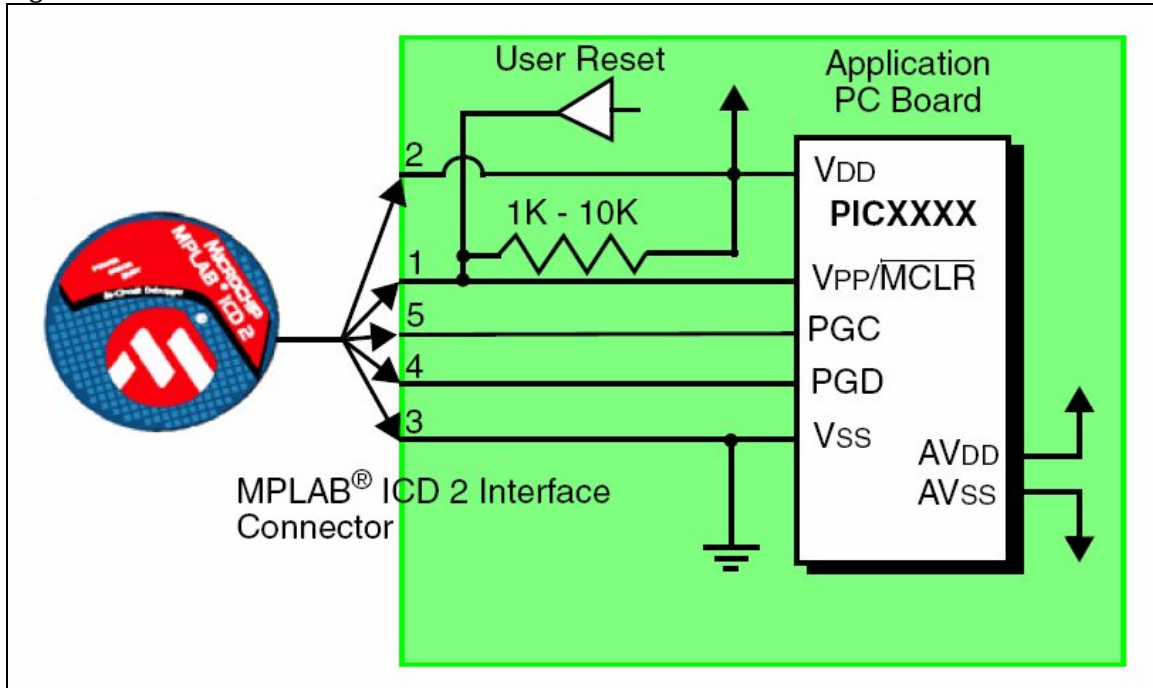
Our first task was to choose the microchip that we were going to use for this project. We chose Microchip microprocessors as they are free to sample, and if sent into production, the microprocessor is cheap and powerful. Our initial design specifications were to have a 10-12 bit A/D converter, several I/O ports, a pulse width modulator, at least 3K Bytes of RAM and 1K of EEPROM. We chose a Microchip PIC18F2620 which is a 28-pin package with 10-bit A/D conversion, C compiler optimized architecture, and 3K of RAM and 1K of EEPROM. We could have used a PIC24 which is a 16-bit microprocessor, but it was a 64 pin package and the project that we were provided with from previous work is also a 28 pin package. The reason we chose not to go with the microprocessor from the previous work was due to the fact that it had insufficient memory and only an 8-bit A/D converter. The previous project required more decimal accuracy so we went with the 10-bit A/D converter.

The software team's job is to interface the circuit and microprocessor with the batteries and solar panels. The main goal is to measure the maximum power distribution in order to distribute power to and from the batteries from the solar panel and to the home or electrical device that will be used. In doing this, there will be an LCD display and menu system in order to change and set battery charging constants, verify voltage and current readouts, and other general calibrations on the system. Our first plan of action was to output text to the LCD display, which is done by the microprocessor sending data streams to the LCD in order for it to display its respective character. This turned out to be a more difficult task than it appeared.

One of the main difficulties we had in starting the software interfacing was acquiring the necessary software to communicate and program the microprocessor. Microchip has a software developers' Integrated Development Environment (IDE) called MPLAB. This IDE is a powerful programming tool that supplies large libraries of tutorials to make programming easier. Microchip also provides a C18 C compiler that we found for educational purposes was unfortunately a trial version which is now limiting our program size and certain compile time optimizations. The retail value of this software is \$495 per license, which is clearly out of our budget range. We will, however continue to use this compiler until it isn't sufficient for our purposes. We also required an

In Circuit Debugger to program and debug our microprocessor. For this we choose the Microchip ICD 2. The MPLAB ICD 2 is a low cost, real-time debugger and programmer which will connect to our PIC MCU. The ICD 2 connects to the computer using a USB port and connects to the microprocessor using a 5 wire connection to hook the clock, the data, the ground and the voltages as seen in **Figure 19**.

Figure 19 MPLAB ICD 2 interface connector



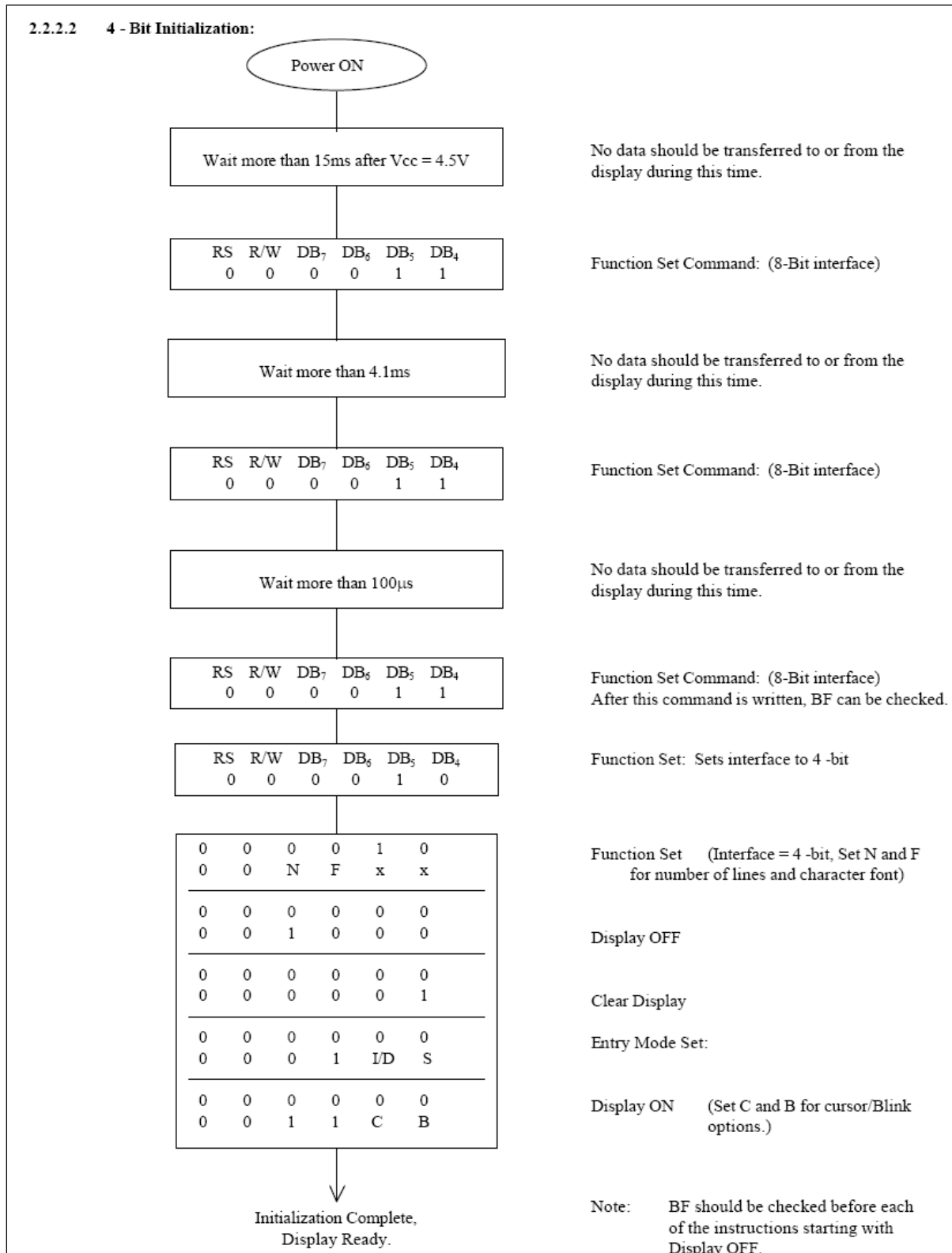
Once the ICD 2 was in place on the prototype board, we tested to see if we could program the microcontroller. The results of the testing were that the ICD2 debugger would not connect to our microcontroller when we were using any version of our IDE newer than v7.00. Once this problem was solved, we soon realized that our C18 compiler would not compile with any IDE versions older than 7.22. The solution to this problem with the advice from the people at microchip was to take out a few resistors on our board that were needed in the v7.00 but not needed in v7.5.

We then wrote a program to display text on our LCD. At first, we simply initialized the appropriate registers and tried a print("Hello World"). For our testing phase, we used a simulated UART which displayed the correct print statement. However, our microcontroller is hooked up to our LCD using 4 I/O pins, and not a serial

connection. The ports RB0 – RB3 are used to communicate to the LCD display. We initialized these I/O ports using our C18 Manual which came with the compiler, and tried following the steps in **Figure 20** to initialize the LCD in order to transfer the letters to it.

Figure 20 4-bit Initialization

Figure 20 4 Bit Initialization



This initialization will need to be a subroutine so that every time the program needs to write to the LCD, this subroutine will be summoned. In order to program not just one letter, but strings of letters, a character array method will have to be created. The font table in **Table 1** is the data streams that need to be sent to the LCD in order for it to display its respective character [7].

Table 1 Character Bit Stream Table

High order bit Low order bit	0000	0010	0011	0100	0101	0110	0111	1010	1011	1100	1101	1110	1111
X X X X0000	CG RAM (1)		0	@	P	`	P		—	9	≡	α	p
X X X X0001	(2)	!	1	A	Q	a	q	=	7	≠	4	ã	q
X X X X0010	(3)	"	2	B	R	b	r	「	イ	ツ	×	β	θ
X X X X0011	(4)	#	3	C	S	c	s	」	ウ	〒	€	ε	∞
X X X X0100	(5)	\$	4	D	T	d	t	、	エ	ト	¥	μ	Ω
X X X X0101	(6)	%	5	E	U	e	u	・	オ	ナ	1	ç	û
X X X X0110	(7)	&	6	F	V	f	v	ヲ	カ	ニ	ヨ	ρ	Σ
X X X X0111	(8)	'	7	G	W	g	w	7	≠	7	う	g	π
X X X X1000	(1)	(	8	H	X	h	x	ィ	ウ	ネ	リ	ƒ	̄
X X X X1001	(2)	)	9	I	Y	i	y	ウ	ツ	ル	ニ	ˆ	y
X X X X1010	(3)	*	:	J	Z	j	z	エ	コ	ン	レ	j	≠
X X X X1011	(4)	+	;	K	[	k	(	オ	サ	ヒ	ロ	×	π
X X X X1100	(5)	,	<	L	¥	l	l	¥	シ	フ	ワ	φ	π
	(6)	—	=	M	I	m	)	ユ	ズ	△	ン	±	÷
	(7)	。	>	N	^	n	→	ヨ	セ	ホ	△	ñ	
	(8)	/	?	0	_	o	←	ッ	リ	マ	°	ö	■

Note: Character of high order bit 1110 and 1111 may be inadequate.

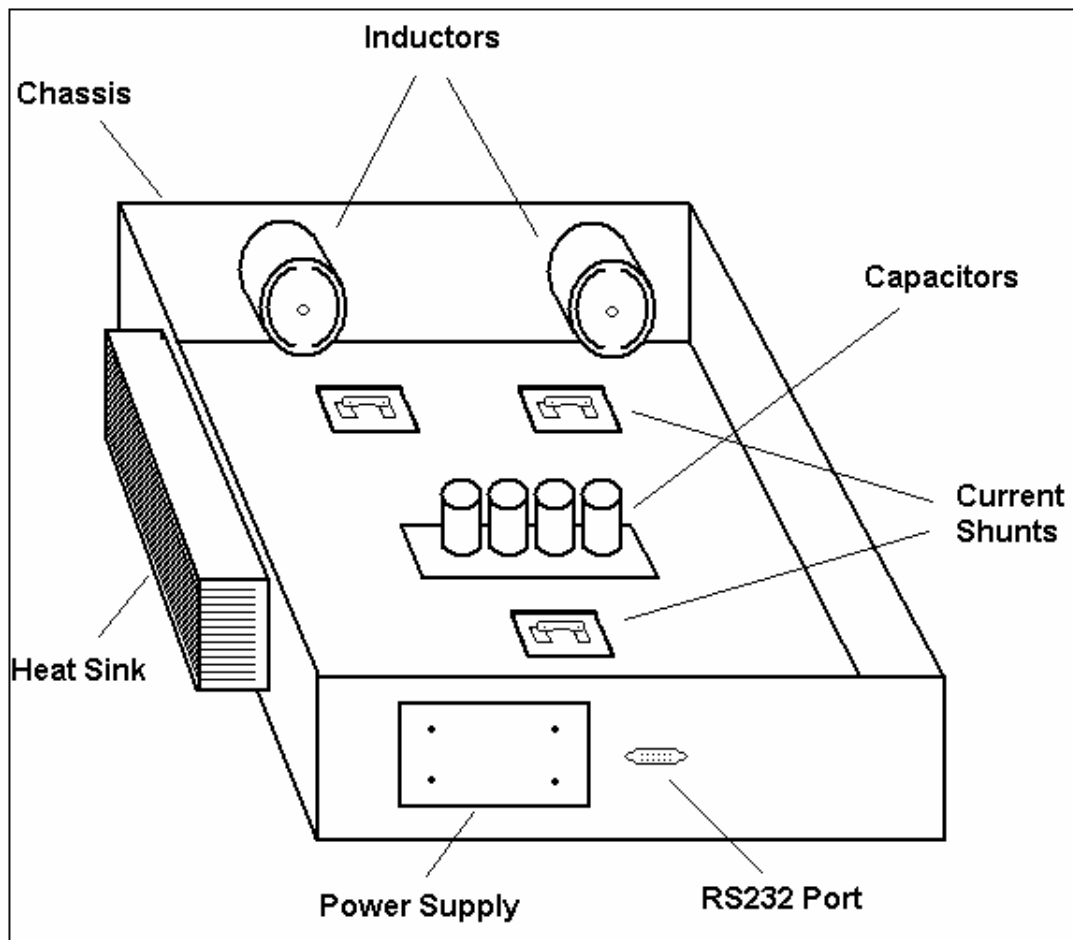
Once we successfully wrote a character or string to the display, we implemented a menu system. This menu system will be navigated by 3 buttons that are attached to our microcontroller I/O RC3 – RC5. Two buttons will navigate up and down and the third button will be a select. This will allow for browsing and calibration of the duty cycle to the pulse width modulator.

After we implemented the LCD, we began to work with the pulse width modulator on the microcontroller in order to change the power from the buck regulator. Pin RC1 is going to be hooked to the buck regulator which will change the pwm to produce the appropriate voltage that has to go to the battery. Lead Acid batteries charge in 3 stages so in order to do this correctly with a microcontroller, we will need to monitor both time using the clock and voltages using our op amps. Stage 1 takes about 5 hours and the battery is gradually charged to 70%. Stage 2 takes about 5 hours and the charge current is gradually reduced as the cell is being saturated. Stage 3 is the float charge which compensates for the self-discharge. In order to implement this, we will use a method with a case statement in order to judge what state the battery is in currently. Stage 1 will simply use constant current to charge. Stage 2 will use constant voltage with around 2.4V to charge. Stage 3 will use a float charge or constant voltage of 2.25V. Lead acid batteries are also temperature sensitive so a 2.25V charge will need around 25° C. A temperature gauge will not be implemented so the owner of the battery charger will need to take precaution as to not overheat or freeze the batteries. The stages of charging will be implemented using global variables as for later calibration if needed using the LCD menu system.

IV. Fabrication

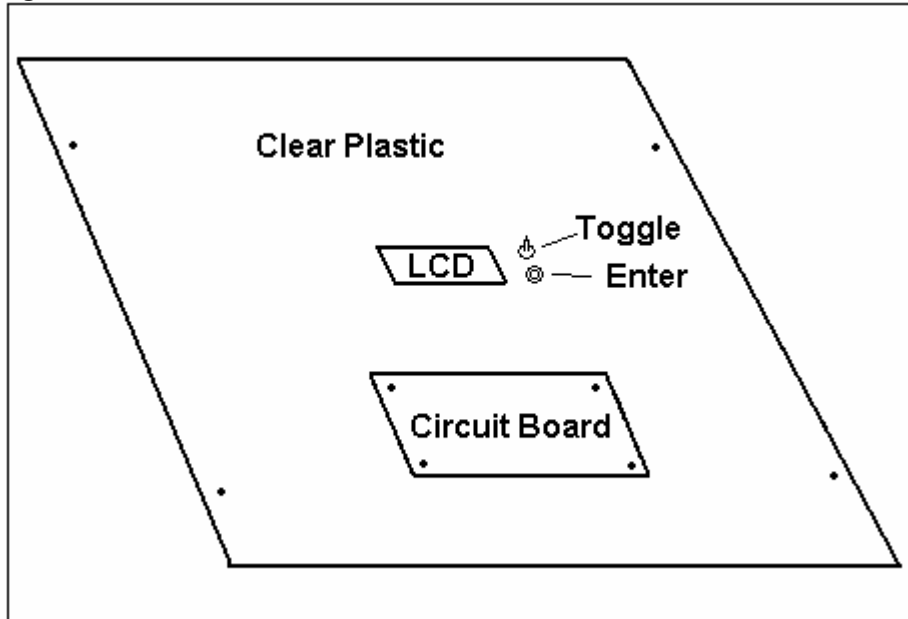
After the power train group had finished their design of the power system, they focused their attention on building the prototype. The specification for this portion of the project again came from our industry advisor Jack Gilmore. He told us that he wanted all the components contained within a single chassis with inputs for the solar panels and outputs for the battery array and the circuit panel. After this, he allowed us to make decisions about where each component should go. We knew that the wires that connected the diode and switch (attached to the back of the heat sink) to the capacitors should be as short as possible to reduce stray inductance. So we tried to make the capacitors as central to all the components and as close to the diode and switch as possible. We needed two inductors for each solar panel arrays (only have one right now, but may have two in the future). We needed three shunts for the batteries and for the two solar arrays. We also needed a RS232 port for communication from the computer to the microchip (used only for troubleshooting). **Figure 21** shows are initial design.

Figure 21 Chassis with major components



We also decided that we would like to have the circuit board attached a clear plastic cover panel. The LCD and the toggle and enter buttons would be attached to the panel. This would allow for safe viewing of all the components at E-Days. If this were to be placed in a home, we would make the cover panel opaque for aesthetic reasons. **Figure 22** shows our design for the clear plastic cover panel.

Figure 22 Clear Plastic Pannel Cover



After we designed both the box and the cover panel, we began the fabrication. All of the components for this portion of the project were donated by Advanced Energy. When we received the chassis, we began drilling holes and attaching components according to our design. By E-Days, we had all the components attached and everything wired up.

V. BUDGET

The budget for this project is based on fifty dollars per student per semester. This makes a total of \$600.00 for this project. The following is a cost breakdown of our expenditures thus far and our planned expenditures for the following semester.

Table 2 Budget Costs

Count	Description	Cost	Total Cost	Charge to Project
	Circuit board components			
	Microchip PIC18F2620 28 pin			Sample
	All components		\$ 40.65	Donated
	Components of the Charger			
1	Display	\$ 11.61	\$ 11.61	Donated
1	Heatsink	\$ 15.00	\$ 15.00	Donated
1	Circuit board	\$ 14.00	\$ 14.00	Donated
1	40W power supply	\$ 26.50	\$ 26.50	Donated
3	Key switches	\$ 1.01	\$ 3.03	Donated
1	Inductor pot core	\$ 43.00	\$ 43.00	Donated
1	Power Diode 1kV 30A	\$ 9.00	\$ 9.00	Donated
1	FET 500V 77A	\$ 23.94	\$ 23.94	Donated
1	50 A shunt.	\$ 9.91	\$ 9.91	Donated
1	Chassis	\$ 19.62	\$ 19.62	Donated
	Total cost of charger	\$ 216.26	\$ 216.26	Donated
4	Unisolar US-64 solar panels	\$ 279.00	\$1,116.00	Donated
1	Lead acid battery	\$ 50.00	\$ 50.00	Donated
1	Inverter (300 W sine wave)	\$ 130.00	\$ 130.00	Donated
	In circuit Debugger			
1	ICD2 debug module	\$ 169.49	\$ 169.49	\$ 169.49
	Shareware C18 C compiler	\$ 495.00	\$ 495.00	Trial offer
	Equipment used to make the board			
	pcad 1 seat license	\$12,000.00	\$12,000.00	Use of donated
	PCB router	\$30,000.00	\$30,000.00	Use of donated
	Miscellaneous parts and tools	\$ 35.00	\$ 35.00	\$ 35.00
	Miscellaneous administrative costs	\$ 6.00	\$ 6.00	\$ 6.00
	Future Cost to the project			
4	Aluminum extrusion 1x1 x8ft	\$ 11.24	\$ 44.96	\$ 44.96
	Total charged to project	\$ 210.49		
	Left in project:	\$ 389.51		

VI. CONCLUSIONS AND FUTURE WORK

In conclusion, the solar powered lead acid battery charger project has proven to be more work than we could handle in two semesters. Although we made great gains this semester in the hardware components, board fabrication, and software interfacing, we didn't actually get all of the code in place necessary to make the measurements, calculate the required feedback duty cycle, and allow the unit to self-optimize its power curves unattended.

For next semester we are recommending that two software students very familiar with C embedded systems PIC programming finish up the software necessary for this project's ultimate success. Our problem was in not having the sufficient tools or experience necessary with these kinds of devices to be able to isolate the debugging issues of the software to one localized arena. For example if we received an error during execution we weren't sure if it was a Microchip MPLab software run-time error, whether we had failed to initialize a register on the PIC properly for program download and execution, or whether the biasing components on the PIC processor pins had a ground fault or other hardware related problem. This ambiguity led to a lot of headaches and probably the failure to meet our goals.

As we turn the project over to fresh hands, the current team has finished all of the components necessary for the hardware and power train to function properly. All of the wires, chips, R's, L's, C's have been chosen and connected and we assume will function properly. We also have successfully interfaced with the computer through the ICD2 in-circuit debugger module to the chip and successfully flashed/executed a menu system and duty cycle (PWM) selector program. The next step would require reading in the four measurements: battery voltage, battery current, panel voltage, and panel current. Once these value have been A/D converted we can manipulate theses hexadecimal representations by coding C equivalent subroutines for $P=IV$, and $V_{out}=duty_cycle \cdot V_{in}$, etc. equations. Then the PIC would need to constantly monitor these measurements and make corrective decisions autonomously. We also expect next year's students to have modes of charger operation that address the stages of battery charging including a CC mode, CV mode, and "boiling mode" which evenly disperses the charges periodically to increase battery life and performance. Eventually we would even like to see the software

in place for the addition of a second array of solar panels (from a different manufacturer) and the second buck regulator as well as variable battery array size to make things more adaptable to a customer's particular equipment.

We feel this project has taught us the value of planning ahead, the value of a finding (and using) an industry mentor, and the difficulties unique to multidisciplinary embedded system designs (utilizes power, software, hardware, optics, active, passive, large signal, small signal, digital control, filters, and system design). We have learned that the most influence you can have on your project is at the very beginning, so a strong understanding of your design goals and a pathway to success are vital to project completion. Although we had met over the summer and established a general understanding of how the unit should work, we did not get a technical understanding of the project until it was much too late to meet most of our goals we had outlined. Also, scope of an engineering project is a very important boundary to get nailed down. Along the way we added the second panel array components and capability before we had even prototyped the first array which was a distraction. On the topic of prototyping, we discovered (too late) that a breadboard implementation of the PIC microprocessor and associated electronics would have facilitated easier debugging and should be extremely valuable to a software team next time. To sum up, the progress we made on the project this year, though considerable, was ultimately hampered by scope broadening, time scheduling, and insufficient tools/knowledge for debugging a PCB (recommending breadboard). Hopefully, next semester's students will benefit from our extensive documentation and complete the software in time for E-days '08.

References

- [1] “Energy [R]evolution: energy technologies: solar power (photovoltaics),” [Online Document], [2007 April 8], Available at HTTP: <http://www.energyblueprint.info/30.0.html>
- [2] “DayStar Technologies: Products: Technology: Copper Indium Gallium diSelenide (CIGS),” [Online Document], [2006 December 2], Available at HTTP: <http://www.daystartech.com/technology.cfm>
- [3] J. Gilmore, “RE: Final Report” [Online Document], [2006 December 1], E-mail to Andrew Giddings [Andrew@engr.colostate.edu]
- [4] K. Ross, “Project: RS-232 to TTL cable,” *Encoder: The Newsletter of the Seattle Robotics Society* [Online]. Available at HTTP: <http://www.seattlerobotics.org/encoder/aug97/cable.html> [Accessed Dec. 2nd, 2006]
- [5] “Microchip PIC18F2620 Data Sheet,” [Online Document], [2006 December 1], Available at HTTP: <http://ww1.microchip.com/downloads/en/DeviceDoc/39626b.pdf>
- [6] “Switching Regulators,” [Online Document] 34, [2006 November 29], Available at HTTP: <http://www.national.com/appinfo/power/files/f5.pdf>
- [7] “International Rectifier IR2110 Data Sheet,” [Online Document], [2006 December 1], Available at HTTP: http://www.alliedelec.com/Images/Products/Datasheets/BM/INTERNATIONAL_RECTIFIER/International-Rectifier_Actives-and-Passives_2737836.pdf
- [8] “Dot Matrix Character LCD Module User’s Manual,” [Online Document], [2006 December 2], Available at HTTP: http://www.optrex.com/pdf/Dmcman_full.pdf

Bibliography

- “Solar Cell,” in *Wikipedia: The Free Encyclopedia*, 2006 [Online], Available Wikipedia Online: http://en.wikipedia.org/wiki/Solar_cell#Solar_cell_efficiency_factors
- W. Kester and B. Erisman, “Section 3: Switching Regulators,” [Online Document], 3.10, [2006 December 2], Available HTTP: http://www.analog.com/UploadedFiles/Associated_Docs/84245885Power_sect3.PDF

Acknowledgements

Special thanks go out to Jack Gilmore and Advanced Energy for their significant contributions to our senior design project; from design considerations to component donations. We would also like to express our gratitude to Microchip for their PIC microcontroller sample.

Appendix A

- A/D** – Analog to Digital
- CIGS** – Copper Indium Gallium diSelenide, fabrication process for solar foil
- EEPROM** – Electrically Erasable Programmable Read Only Memory
- HPVEE** – Hewlett Packard Visual Engineering Environment
- I/O** – Input and Output
- ICD2** – In Circuit Debugger V.2.00
- IDE** – Integrated Development Environment
- IR2110** – International Rectifier chip for buck regulator
- LCD** – Liquid Crystal Display
- MPLAB** – Integrated Development Environment for interfacing PIC microcontrollers
with bit streams created from assembly or c projects
- PIC** – Peripheral Interface Controller
- PWM** – Pulse Width Modulator
- RS232** – Radio-Electronics-Television Manufacturers' Association Standard serial
communication port