Midterm Report

Devin Hernandez   
*Electrical Engineering  
The University of Texas at Arlington*Arlington, Texas  
devin.hernandez2@mavs.uta.eduJose Guzman  
*Electrical Engineering*  
*The University of Texas at Arlington*  
Arlington, Texas  
jxg3557@mavs.uta.eduJonathan Dolormente  
*Electrical Engineering  
The University of Texas at Arlington*Arlington, Texas  
Jonathan.dolormente@mavs.uta.eduNguyen Thai Binh Duong  
*Electrical Engineering  
The University of Texas at Arlington*Arlington, Texas  
bxd3589@mavs.uta.edu

*Abstract*— This report will outline important details concerning the progress made and goals achieved for our PV-Battery Integrated Energy Source project. We will first cover the introduction to our project, our procedures for the project, and the results we gather from performing certain tasks, followed by our results and discussions of results.

Keywords—Arduino, battery, photovoltaic, MOSFET, MPPT

# Introduction

In this project, we have been tasked with creating a PV-Battery Integrated Energy Source circuit starting from simulation and proceeding to a final product fabricated from physical components soldered to a printed circuit board (PCB). The procedure, results, and discussion will be in numerical order from two to four. We have not yet reached the final phase of the project, as our PCB design and soldering phases have yet to be completed. As of this report, we are currently still testing our prototypes and designs.

A crucial principle employed in this lab is Maximum Power Point Tracking (MPPT), predominantly utilized in photovoltaic (PV) integrated systems to optimize power efficiency. The power output of a PV array is influenced by various factors, including temperature, cloud cover, and irradiance. Hence, the implementation of MPPT is imperative to ensure the harnessing of maximum power regardless of these varying conditions. Figure 1 illustrates the method through which MPPT is achieved.

A diagram of a solar panel

Description automatically generated

Figure . Typical I-V curve for a solar panel

It's essential to highlight that this representation pertains to a specific scenario and may not be universally applicable. The process involves regulating the voltage, which in turn affects the current emanating from the PV array. The maximum power point is identified by maximizing the product of voltage and current. Consequently, the optimal combination of voltage and current values is determined and adjusted to ensure peak efficiency.

MOSFETs play a significant role in allowing voltage to be stepped up or down in the buck-boost converters used. These devices are sent a PWM signal by the Arduino that tells it exactly how long to turn on and off. This will either allow the voltage to be stepped up or down in the case of buck-boost converters (in which we are using two). For buck-boost converters, the circuit will either be in buck mode when the PWM signal is below 50 percent duty cycle or be in boost mode when above 50 percent duty cycle. This will allow for the manipulation of voltage and therefore current, in turn allowing for a phone charger and battery to be charged.

# Procedure

We started the project by doing research and simulation, as well as assigning roles to our four group members and setting up a basic schedule for our relevant tasks. We were given a large set of literature to use for the project, each containing different topics relating to the project, and other literature that contained information on components that could be used. We were also given some MultiSim files that would be useful for the project.

A diagram of electrical wiring

Description automatically generated

Figure 2. A circuit diagram of the gate driver that we are using. This was provided to us for simulation purposes.

After quickly reviewing our supplied materials, we decided that our group roles would be as follows:

* Jose: The Project Manager of the group; responsible for developing circuit topology, aiding in prototyping and implementation, additional research, and schedule adhesion.
* Devin: The Designer of the group; responsible for maintaining the simulation documents, programming, spearheading the creation of the PCB, and double-checking construction quality.
* Jonathan and Binh: In charge of Prototyping and Implementation. Tasked with the construction of the breadboard prototype and soldering components to the PCB.

At this point in the project, Devin made a list of parts to order for the project so that Jonathan and Binh could put together a prototype, represented in the table named “Parts List 1” appended at the end of the report. We have been given a budget of $230 by the department to order the parts that we need to complete the project. The parts that were ordered with the first list were put toward the Bidirectional DC-DC converter circuit and the making of its dedicated gate driver. The leftover parts were kept as extras. As the prototype was being built with the parts that were ordered, Jose and Devin worked on programming the Arduino to produce a PWM output required by the two MOSFETs in the Bidirectional DC-DC circuit to function properly.

## Breadboarding

First, before any prototype can be created, a circuit topology must be laid out. To begin this process, research must be done to determine the best and most efficient way to connect all our major components. Article [1] gave insight into how our PV array and DC-DC converter should be connected to the rest of our components. The layout for that portion of the circuit can be seen in Figure 3.

Diagram of a solar panel system

Description automatically generated

Figure 3. A general layout for an MPPT-controlled buck-boost converter.

It should be noted that in Figure 3, MPPT occurs before the DC-DC converter. This means the best voltage and current values will be input into the DC-DC converter. Since it is now determined where the load will be located, the Bidirectional DC-DC converter and battery can be connected in parallel with it. These DC-DC converters will need to have their separate gate drivers, which are then controlled by the Arduino. After a layout for the circuit was found, the final topology was created. It can be seen in Figure 4.

A diagram of a circuit board

Description automatically generated

Figure 4. The proposed circuit topology for this project.

The final topology included in Figure 4 accounts for all major components in the circuit. Figure 4 is the foundation of the project; once this is completed, prototyping and implementation could begin.

Jonathan and Binh worked to create the two gate drivers, one is seen in Figure 5, on the top breadboard. They also worked to create both of the DC-DC converters, both of which are shown in Figure 5 and Figure 6.

A circuit board with wires and cables

Description automatically generated

Figure 5. Two breadboards containing the buck-boost converter, its gate driver, and the Arduino Micro.

A white circuit board with wires and wires

Description automatically generated

Figure 6. The Bidirectional DC-DC converter with its gate driver.

To save space, the Bidirectional DC-DC converter and its gate driver were condensed onto one breadboard, taking up only about half of the breadboard as a whole, as seen in Figure 6.

## Programming

The programming has been especially difficult. Devin worked through four different prototypes of the code, with each prototype using a different implementation of a timer. The prototype attempted to use digitalWrite(); to make a PWM signal directly, though we have not yet tested code to perform the MPPT calculations. This first method did not work because, while it is simple to create two static complementary PWM signals with the delay(); method, we could not find a way to implement adjustable timing to the code as required by the project to function properly. The second version of the code used analogWrite(); to generate the PWM signal, but we found that the map(); method could not generate the signal at the bit rate required, as it performs integer math to make the mapping. The third version of the code takes advantage of the timer registers supplied by the ARM64 processor on the Arduino. We do not truly understand how the timer registers work, but we do know that the code works as it should. We made a fourth version of the code that uses the TimerOne library, but we ran into issues with timing and reverted to version three.

The first two versions of the code had an issue where the period of the compliment signal was not aligned with the period of the main signal. This resulted in what appeared to be a half-period phase shift, which is not ideal, as two MOSFETs cannot be on at the same time. Version four of the code simply did not work the way it was intended. This was more than likely due to a lack of understanding.

## PCB Prototype

The last thing that we started work on was the PCB. We intend to have a fully working PCB design that is ready to be milled out by the time breadboarding and programming are completed.

To make the PCB, Devin started by recreating the electrical schematic in KiCad, a PCB design software similar to the likes of Autodesk EAGLE or NI Ultiboard. Devin also added in some extra components that were not included with the simulation files and made sure that all the devices that were placed had parity with the Multisim counterpart.

A circuit board with many wires

Description automatically generated

Figure 7. One of two gate drivers recreated within the KiCad schematic file.

Once this was done, the parts then had to have their footprints assigned so that KiCad knew what to show when the PCB layout was being created. For the more specialized devices, like the Arduino Micro (Figure 8), their footprint was already assigned.

A screen shot of a computer

Description automatically generated

Figure 8. The footprint of the Arduino Micro. The Arduino Nano and the Arduino Micro have the same footprint.

There is, however, ambiguity for more general components, like resistors, capacitors, and inductors, that need to be measured and given their footprints appropriately. These components don’t have a proper dimensional measurement on their datasheets, simply because they are so generic, so Devin had to measure their dimensions with a set of calipers and make a rough comparison with the provided footprints in KiCad to find which ones could work properly. Once all the required devices were placed into the schematic file and given their footprints, all the devices could then be directly imported into the PCB file and placed. From here, all the parts can then be moved freely around the sheet. Note the white lines that connect all the components within KiCad. These lines make it easier to see what devices connect where and are accurate to the connections made in the schematic.

A computer screen shot of a circuit board

Description automatically generated

Figure 9. The whole circuit placed into the PCB design suite.

The PCB is intended to be a standard two-layer PCB that will have the devices loosely grouped by what they are: gate drivers, DC-DC converters, and the Arduino, along with the sensors and the outside connections for the battery, solar panel, and our load. With that in mind, we can follow a basic rule for laying out traces that helps to prevent errant trace paths: the upper layer traces (in red) will mostly travel vertically, and the lower layer traces (in green) will travel horizontally.

A screenshot of a computer

Description automatically generated

Figure 10. Proposed PCB layout for the Bidirectional DC-DC converter, along with its associated gate driver.

# Results

The first results of importance were the values of our input and output voltages from the PV arrays’ DC-DC converter. The table for the given values can be shown in Table 1.

Table

|  |  |  |  |
| --- | --- | --- | --- |
| Trials | Input Voltage (V) | PWM Signal  (%) | Output Voltage (V) |
| Trial 1 | 5V | 25% | 1.9 |
| Trial 2 | 5V | 50% | 2.6 |
| Trial 3 | 5V | 75% | 2.9 |

Table 1 demonstrates the results of multiple testing trials for the DC-DC converter. It should be mentioned that values were tested at these PWM values because of how these converters operate; to operate under buck mode the duty cycle will need to be < 50%, and to activate boost mode the duty cycle will need to be > 50%. Therefore, the three values 25, 50, and 75 percent will give a clear indication of whether the buck and boost modes are working as desired. It can be seen in these results that the buck mode of the converter is working, but that the boost mode is not. These values are not the values that our group intended to achieve; because of this, a separate set of trials was set up for further testing. These results can be seen in Table 2.

Table

|  |  |  |  |
| --- | --- | --- | --- |
| Trials | Input Voltage (V) | PWM Signal  (%) | Output Voltage (V) |
| Trial 1 | 12V | 25% | 4.14 |
| Trial 2 | 12V | 50% | 6.13 |
| Trial 3 | 12V | 75% | 7.13 |

Table 1 provides additional results when the input voltage is changed. The results shown from both Tables 1 and 2 depict a working buck converter instead of a buck-boost converter. Subsequent analysis revealed that the root cause of this discrepancy was the suboptimal frequency of the PWM signal generated by the Arduino, which was measured at a mere 490Hz—significantly lower than the target frequency of approximately 20kHz necessary for optimal operation. This discrepancy necessitates additional experimentation to achieve the desired frequency and converter functionality.

A screen shot of a device

Description automatically generated

Figure 11. PWM Signal at 490Hz

A screen with a graph on it

Description automatically generated

Figure 12. PWM Signal at 31kHz

Figure 11 shows a renewed PWM signal with the desired frequency we need for our circuit. Further results, including the load voltage with the renewed PWM signal and different inductor values, have not yet been tested. In addition, the bi-directional DC-DC has also not been tested. Therefore, there are no additional results to be displayed until further testing can be conducted with these new variables.

# Discussion of Results

As of current, we have conducted 6 separate trials with our circuit. For the first 3 trials, the circuit received 5V with the only varying parameter being the pulse width signal. The first trial begins with a 25% signal and is increased by 25% for each following trial. The resulting output voltage is recorded in the above table. The remaining 3 trials were tested in the same fashion, except the input was raised to 12V. We see from the table that each increase in pulse width increases the output voltage, effectively showcasing a working buck converter. As noted previously in “Results”, all six trials were tested under the old Arduino code which would output a PWM signal at 490Hz instead of 31kHz. Although we have done some testing with the updated frequency, we still do not have clear results that give us an indication of a better-functioning circuit. There will still need to be more testing of both the updated PWM signal and varying inductor values. The results from these specific tests are presumed to be accomplished by the end of this week.

It is notable to mention that the values from our DC-to-DC converter are not the values we want. The DC-DC converter that our group aimed to create was a buck-boost converter, however after discussing our results, what we had achieved was a functional buck converter. Although this was not what we had intended, our team agreed that a buck converter could still be used to accomplish the goal we had set. The power input from the PV array is expected to reach 100W under ideal conditions; this is more than enough to supply both our load and battery with power. Because of this, a buck converter can be placed in conjunction with our PV array and always buck our output to 5V, therefore charging the phone (load).

# Conclusions

In conclusion, our group has made substantial progress on the PV-Battery Integrated Energy Source project. Thus far, we have assigned roles, ordered necessary components, developed Arduino code to produce the required PWM signals, and begun prototyping the bidirectional DC-DC converter circuit. While debugging the Arduino code presented some challenges, we now have a working version that properly generates the PWM signals to control the converter's MOSFETs. Our initial testing of the DC-DC converter verifies that it can operate in both buck and boost modes as intended. However, the output power levels do not yet match the target specifications. Our next steps will be to adjust the inductor and capacitor values in the converter circuit to achieve the desired voltage and current outputs. We will also finalize and test the overall system integration, including the PV array, battery storage, and load. Pending successful results from the breadboard prototype, we will then design the PCB and solder the components to create the final integrated device. With the majority of tasks completed in these initial phases, we are on track to meet the project timeline. Through continued teamwork, troubleshooting, and iterative improvements, we are confident the product will meet all requirements. This project has already provided useful hands-on experience in integrating electrical components, programming microcontrollers, circuit simulation, and prototyping techniques.

# References

[1] Dileep. G and S. N. Singh, “Selection of non-isolated DC-DC converters for solar photovoltaic system,” *Renewable and Sustainable Energy Reviews*, vol. 76, pp. 1230–1247, 2017. doi:10.1016/j.rser.2017.03.130

# Appendix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parts List 1** | | | | |
| Part Name | Part # | Quantity | Unit Price | Total Price |
| MOSFET (10pcs) | LRF540 | 1 | $7.99 | $7.99 |
| Diode | 863-IN4002G | 10 | $0.19 | $1.90 |
| Inductor 100mH | 580-19R107C | 10 | $1.47 | $14.70 |
| Capacitor 47 μF | 80-ESW476M035AE3AA | 10 | $0.27 | $2.70 |
| Capacitor 470 μF | 80-ESK477M035AH2EA | 10 | $0.50 | $5.00 |
| Voltage Regulator 5V | 511-L7805CV | 5 | $0.69 | $3.45 |
| USB 2.0 Port | 530-SS-52100-001 | 5 | $0.56 | $2.80 |
| Gate Driver | 942-IR2110PBF | 2 | $2.92 | $5.84 |
| Arduino Micro | A000053 | 1 | $23.95 | $23.95 |
| Breadboard (10pcs) | pcb5 | 1 | $19.99 | $19.99 |
|  |  |  | Gross Cost | $88.32 |
|  |  |  | Net Cost | $95.61 |
|  |  |  | Remaining Budget | $134.39 |