# BACHELOR'S PAPER

Term paper submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Engineering at the University of Applied Sciences Technikum Wien - Degree Program Bachelor Electronics

# Low Cost Energy Production

By: Tobias Weidhofer
Student Number: el13b001

Supervisor: Dipl.-Ing. Christoph Veigl

Vienna, 24. März 2017

FH University of Applied Sciences
TECHNIKUM WIEN

# Declaration

“As author and creator of this work to hand, I confirm with my signature knowledge of the relevant copyright regulations governed by higher education acts (for example see §§21, 46 and 57 UrhG (Austrian copyright law) as amended as well as §11 of the Statute on Studies Act Provisions / Examination Regulations of the UAS Technikum Wien).

In particular I declare that I have made use of third-party content correctly, regardless what form it may have, and I am aware of any consequences I may face on the part of the degree program director if there should be evidence of missing autonomy and independence or evidence of any intent to fraudulently achieve a pass mark for this work (see §11 para. 1 Statute on Studies Act Provisions / Examination Regulations of the UAS Technikum Wien).

I further declare that up to this date I have not published the work to hand nor have I presented it to another examination board in the same or similar form. I affirm that the version submitted matches the version in the upload tool.“

Vienna, 24.03.2017

Place, Date

Signature

# Kurzfassung

Im Lauf des letzten Jahrzehntes wandelten sich elektronische Geräte vom Luxusgut zum Alltagsgegenstand. Nahezu jeder Haushalt in Österreich verfügt über mindestens ein Smartphone, einen Laptop, ein Tablet oder Ähnliches. Ein Grund für diese Entwicklung ist, dass elektronische Geräte immer günstiger produziert und verkauft werden können und dass nach dem Kauf des Produkts außer den Stromkosten keine weiteren finanziellen Belastungen entstehen.
Strom ist der gemeinsame Nenner aller elektronischen Geräte und kommt bequem und sauber ins Haus geliefert. Dass es in vielen Teilen der Welt eine derartige Versorgung nicht gibt ist für Menschen aus Industriestaaten unvorstellbar.
Menschen in Entwicklungsländern haben zwar oft Zugang zu ausrangierten elektronischen Geräte die mit Geschick wieder in einen lauffähigen Zustand versetzt werden können, es fehlt jedoch an der Stromversorgung um die Geräte zu versorgen.
Diese Arbeit greift diese Problematik auf und ist der Versuch kostengünstige Methoden zu entwickeln kleine Mengen an Strom zu erzeugen. Die Lösung die hier präsentiert wird eignet sich dazu Strom durch Muskelkraft auf einem Fahrrad zu erzeugen. Dafür wurde ein Prototyp in Form einer Platine designt, gefertigt und programmiert. Die Umwandlung wird mit Hilfe eines Buck-Boost Konverters und eines Atmega 328p realisiert. Die Datenweitergabe ist drahtlos mittels eines Esp8266 implementiert.

**Schlagwörter:** Atmega328p, Buck-Boost Konverter, ESP8266, Kostengünstige Stromerzeugung

# Abstract

For a number of years now, electronics have been becoming cheaper by the hour and things like smart phones, laptops and tablets have become attainable for almost everyone in the first world. But all of these items have one thing in common: electricity. Something we take for granted every day but in a lot of developing countries this is not the case. So this poses the following problem: people in Third World countries may have the resources and capabilities to get their hands on electronics to be able to learn things like programming, computer sciences or other technical abilities but the areas they live in may lack the infrastructure of electricity thus making it possible to get their hands on these tools but not actually use them. This thesis addresses that problem and looks for cheap ways to produce small amounts of energy in order to power these low cost and low power but still highly useful devices.

**Keywords:** Atmega328p, Buck-Boost converter, ESP8266, Cost-efficient energy production

# Acknowledgements

My Supervisor Dipl.-Ing. Christoph Veigl for not giving up on me, Katrin Vetter for moral support, my girlfriend Julia Werbick for being there for me and my parents without whom none of this would be possible.

# Contents

# 1 Introdution

## 1.1 Scope of Work / Task Definition

This project was designed to create a converter that could take an input from a generator and convert it to a different voltage. In this case the generator was powered by a bicycle and the generator its self had between 0 and 24V. The output voltage was to be 13,5V in order to charge a battery and power a rectifier simultaneously.The rectifier could then be used to power devices with a voltage of 230V AC. 13,5V was chosen as an output voltage as it is possible to safely charge lead acid batteries over long periods of time without the need of a dedicated battery charging module. The power requirements for the device was 100W resulting in a maximum of 4 A on the input and 7,4 A on the output.
The generator was expected to produce a maximum of 24V although only at peak times.
The battery is a lead acid battery with a nominal voltage of 12V.
Furthermore, the requirement was to be able to measure different metrics like current, voltage and/or power and to output these in a meaningful way either through display, a series of LEDs or, as was chosen, to make the data available over Wi-Fi. The converter and everything attached to it would have to be able to deal with strong voltage fluctuations from the generator which was being powered by a bicycle and the harsh voltage spikes that this would generate.

## 1.2 Related Work / State of the Art

Projects like these have been tried and tested multiple times. But these projects have a one thing in common, they are expensive. For example A-frames for bicycles are purchasable but at a high cost (520EUR)[1] but building one by hand only costs a fraction of this, as this is only constrained by material costs. There are other forms of energy generation as well. Solar, water- or wind- turbines to name a few, but as stated before these solutions are all very expensive.
As usually voltages have to be increased or decreased from the source to the load DC/DC converters also play an important role in energy production. These are comparatively cheap, for example an 80W buck-boost-converter can cost as little as 13EUR[2] but often these solutions are not well documented and hard to reuse for another purpose if the occasion should arise.
Solutions for autonomous energy production do exist but few are cheap, reliable and maybe most importantly self maintainable.

[1] https://www.electricpedals.com/shop-pricing
[2] http://www.ebay.com/itm/LTC3780-DC-5-32V-to-1V-30V-10A-Automatic-Step-Up-Down-Regulator-Charging-Module-/221953001309?hash=item33ad6e075d:g:ObMAAOSwys5WWQQ5

# 2 Hardware

## 2.1 Requirements

The following chapter briefly describes the requirements and possibilities that exist to implement these and describes the pros and cons of each possible component.

### 2.1.1 Microcontrollers

One of the most important components of the prototype was the microconroller. The selection process was governed by following criteria:

- A built-in hardware Pulse Width Modulation (PWM) module with a minimum of 30 KHz was required to pulse the buck-boost-converter without the switching currents exceeding 20A.
- A minimum of five Analog Digital Converter (ADC) connections to evaluate the ammeters and voltage dividers.
- A connection to allow the two microcontrollers to communicate, preferably Universal Asynchronous Receive Transmit (UART) as no clock is required and the setup is simple.
- A low power draw to minimize the energy wasted
- Low cost and high availability to make it accessible to a wider audience.
- Easy usability and programming that the functionality can easily be changed.
- A small footprint to reduce the size of the board.

#### Possible Microcontrollers

There were a great amount of microcontrollers to choose from but since the chosen Integrated Development Enviorment (IDE) was the Arduino IDE [1] the choice of microcontroller was quickly condensed. Due to the factors mentioned in section 2.1.1 the selection was further reduced and the microconrollers that were left are listed below. It noteworthy though, that this is only a selection as the actual number of possible microcontrollers are much higher.

- Arduino Due[2]:
  The Arduino Due was the first choice of board as it has a 32bit architecture and a clock speed of 84Mhz but would have driven the price of the board up and using the Atmel SAM3X8E as a stand-alone chip would not have been feasible to solder by hand.
- Arduino Uno[3]:
  The Arduino Uno would have been cheaper than the Due but was still very big in comparison to the chip it was using, the Atmega328p, and the cost of the board in comparison to the chip alone was very high as well

- Atmega 328p[4]:
  As previously stated, the Atmega328p is the chip that is used within the Arduino Uno. So with a few components it was possible to have the same processing power and connections as with the Uno, but using much less real estate on the Printed Circuit Board (PCB). The Atmega328p was also by far the most cost-efficient solution.

### 2.1.2 Wi-Fi Chip

As the final PCB was to have Wi-Fi to implement communication with the user, a chip was required that could be used as an access point. The criteria for this Chip was not as high as for the main microcontroller. The most important criteria was price, ease of use, a possibility to broadcast Wi-Fi signals, availability and a port to communicate over (preferably UART). The best choice to fill this criteria was the ESP8266[5].

### 2.1.3 Type of Solution

There were multiple ways to implement a DC/DC converter. Each had their advantages and disadvantages as can be seen in the following list:

- Pre-bought PCB:
  The simplest version of a buck-boost-converter would have been a pre-bought chip. The disadvantage would have been that reading out of values would not have been possible. The reason for this lies in the fixed layout of the pre-bought PCB.
- Buck-boost chip:
  Devising a circuit using a buck-boost chip would have been possible but this would have added more chips to the already full board. A buck-boost chip would also have been more restrictive in terms of customization
- Self designed:
  Using a hardware PWM from a Microcontroller (uC) to control the buck-boost-converter and use the same uC to collect data simultaneously offered the best solution, though at the same time the most complex. This was also the final type of solution used.

### 2.1.4 Disregarded Approaches

In the process of creating this project there were a multitude of approaches that were disregarded in favor of superior, more efficient or cheaper solutions. The first idea was to implement 2 different footprints to accommodate either the six pin ESP8266-01, the first ESP8266 model, or the ESP8266-12/-12E/-12F but this was disregarded as the latter is becoming easier to purchase and can be soldered directly to a PCB so there was no need for a second footprint.
The second idea was to use the Arduino Due as the main processor. As described above in section 2.1.1 the Arduino Due is a very powerful board, but the functionality exceeded the necessary requirements and so with was disregarded.
Another approach was to use through-hole components. These components are easier to solder by hand, but once a higher volume of boards are produced Surface Mounted Devices (SMDs) are faster to produce and have the added benefit of using less space thus requiring less PCB real estate.
The final and most substantial disregarded approach was the buck-converter. A buck-converter is useful for reducing voltages but, as in this project, the input voltage could be either above or below

the output voltage. This converter was quickly replaced by a buck-boost-converter, which will be covered later in this paper.

## 2.2 Final PCB

### 2.2.1 Overview

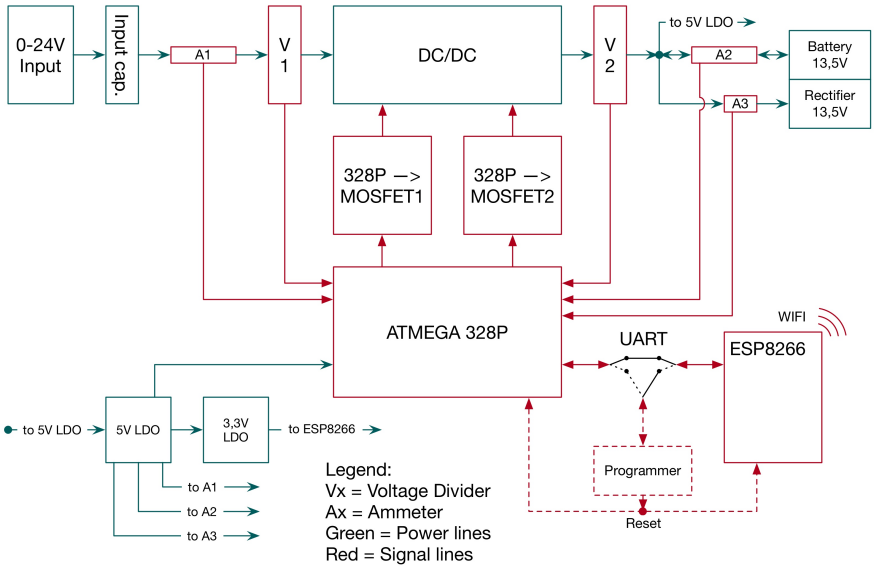


Figure 2.1: Block Diagram

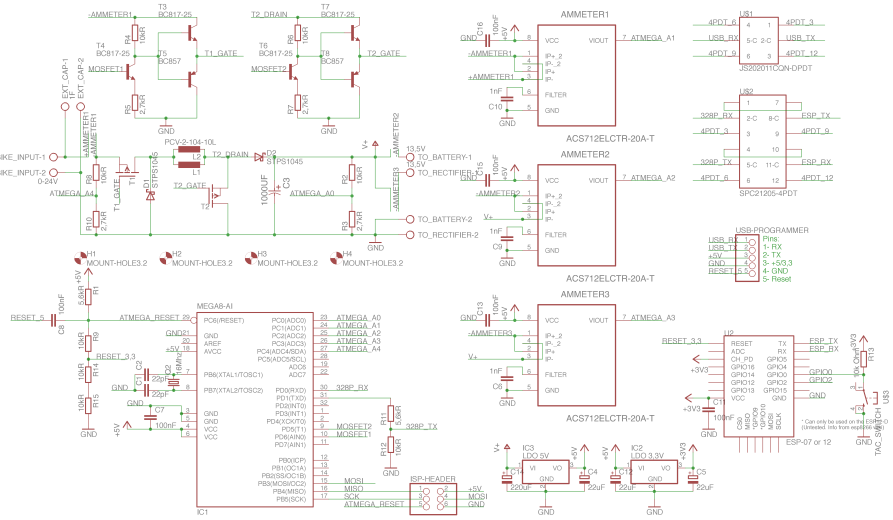

Figure 2.2: Schematic

Figure 2.1 depicts the most relevant blocks of the schematic (figure 2.2) for easier overview.

### 2.2.2 Buck-boost-converter

The first part that had to be calculated was the buck-converter as this was the keystone of the project. Each component had to be calculated in such a way as to be able to withstand the maximum 24 V and more critically the maximum 10 A running through the system.

$$100W/13,5V = 7,4A \tag{2.1}$$

As can be seen in equation 2.1 the maximum allowed current would have been 7,4A but as a buffer it was necessary to calculate all buck-boost relevant components with 20A due to the higher switching current that can be seen in equation 2.4.

All calculations for the buck-boost-converter were achieved using a data sheet from the company TI that published a paper describing the procedure[6]. All the data and formulas from this paper were taken and condensed into an Excel spreadsheet for easier manipulation and calculation which can be found in the Appendix.

The first part was to choose the type of converter. The options were:

- a boost-converter
- a buck-converter
- a buck-boost converter

As mentioned in section 2.1.4, after first considering a buck-converter because of its lower switching current (equation 2.2)

$$\begin{aligned} MaxSwitchingCurrent &= \frac{((VinMax - Vout) * MaxDutyCycle)}{SwitchingFrequency * InductorValue} + IoutMax \\ 8,4A &= \frac{((24V - 13,5V) * 0,45)}{62500Hz * 42\mu H} + 7,5A \end{aligned} \tag{2.2}$$

the final decision was to choose a buck-boost-converter as can be seen in figure 2.3

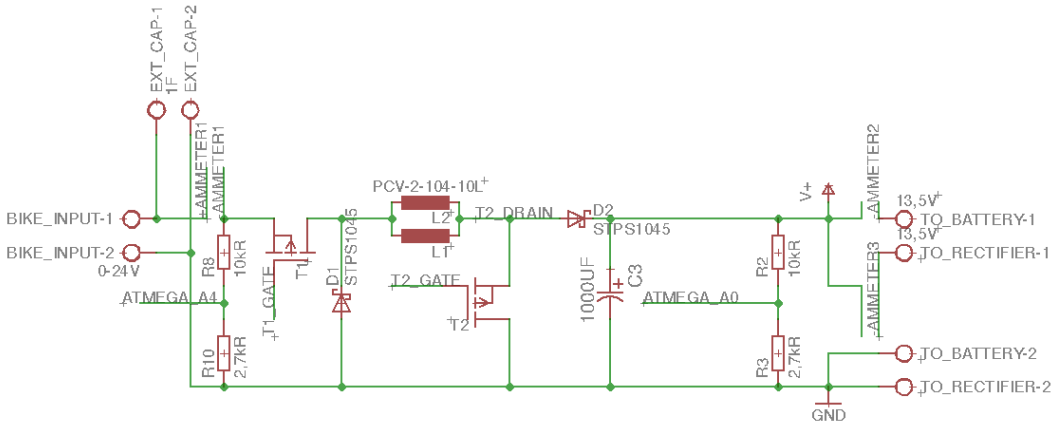


Figure 2.3: Buck Boost schematics

All three converter types operate in a similar fashion. They take the input voltage and break it down into small energy packets which are then forwarded to the output at different rates also known as

the duty cycle. The duty cycle is a percentage of how much a component is turned on or off within a period. The altering of the duty cycle determines how much the output voltage increases or decreases depending on the type of converter used. A buck-converter is used to decrease voltage, a boost converter to increase voltage and a buck-boost-converter to increase and decrease voltage. A buck-boost-converter can only be used as either a buck- or boost-converter at a time and thus requires switching between these two modes.
As dictated by the law of energy conservation energy cannot be created or destroyed. It would be false to think that these converters contradict this law. This implies that as voltage sinks the amperage must increase and vice versa. Thus the power is always equal, once the power loss of the converter itself has been taken into consideration. As stated before, this conversion between voltage and amperage is done by small energy packets. These are made by triggering the Metal–Oxide–Semiconductor Field-Effect Transistor (MOSFET) which directs energy through the inductors and begins to induce a magnetic field. Simultaneously the capacitor is charged. This effect is only sustainable while the inductors are not fully saturated and the capacitor is never completely charged. If this were to occur the full input voltage would be transferred to the output so with negating the aim of the converter.
Normally there are specific microchips that handle the PWM regulation of such converters. For this project this was done using a microcontroller for a multitude of reasons as mentioned in section 2.1.4

### Inductor

The first component of the buck-boost-converter, that was calculated, was the inductor as this is one of the two key components to determine the operating constraints of the converter. The constraining factor for the system and the inductors was the switching current which was governed by the maximum allowable power through the system. The problem lay not within the current but the combination of the current and switching frequency, which was dictated by the microcontroller, which in this case was 62,5kHz[7]. This resulted in a high power, high inductance inductor which was not easily obtained at a cost-efficient price. The compromise between these factors was to use two inductors parallel, thus halving the inductance value while doubling the allowable current to 20A. These approximate 20A were obtained as can be seen in the equation 2.4 and represents the maximum switching current.

$$\begin{aligned} L &= \frac{Vout * (VinMax - Vout)}{0,3 * SwitchingFrequency * VinMax * IoutMax} \\ 42\mu H &= \frac{13,5V * (24V - 13,5V)}{0,3 * 62500Hz * 24V * 7,5A} \end{aligned} \tag{2.3}$$

$$\begin{aligned} MaxSwitchingCurrent &= \frac{\frac{VinMax * MaxDutyCycle}{SwichingFrequency * InductorValue}}{2} + \frac{MaxOutputCurrent}{1 - MaxDutyCycle} \\ 20A \geq 18,86A &= \frac{\frac{24V * 0,59}{62500Hz * 42\mu H}}{2} + \frac{7,5A}{1 - 0,59} \end{aligned} \tag{2.4}$$

The final inductor is the Coilcraft PCV-2-104-10L has 100$\mu$H which is greater than the needed 42$\mu$H (see equation 2.3) and Irms max of 10.1A[8]. But since the Inductors are in parallel together they still have 50$\mu$H which is still greater than the needed 42$\mu$H.

### Output Capacitor

The second key component was the output capacitor. This component was more readily available as the only criteria was to be able to withstand at least 24 V and have a capacitance equal to that, or higher than the calculated value which can be seen in equation 2.5. The 24V tolerant capacitor was necessary as it is possible for T1 to have the full voltage running through it before the Atmega328 has the possibility to pulse the T1. This is an unlikely scenario, but as it is a possible one has to be considered none the less. As opposed to the voltage the current did not pose a problem since the capacitor is parallel to a plethora of other components and capacitors are self regulating regarding currents. The final capacitor is 50V resistant and has 1000$\mu$F which provides a buffer of 298$\mu$F against the calculated value of 702$\mu$F.

$$\begin{aligned} CoutMin &= \frac{IoutMax * MaxDutyCycle}{SwitchingFrequency * DesiredOutputRipple} \\ 702\mu F &= \frac{7,5A * 0,59}{62500Hz * 0,1V} \end{aligned} \tag{2.5}$$

### Input Capacitor

The input capacitor was already supplied and had a capacity of 2,2 F, which under normal circumstances would have been a very high value to choose. As the energy being produced by the generator highly fluctuates, due to the inconsistent manual input this capacitor was required to make the voltage more consistent.

### Diodes

Diodes, similar to the capacitors mentioned before, were readily available at the currents necessary. One thing to be noted is that these components were not available as SMD since the power dissipation would have been too great for the smaller packages of the SMD. For example, a standard diode in SMD could have anywhere up to 1A[1] but this did not fulfill the requirements for this project.
The final diodes that were chosen were the STPS1045 which have a average forward current of 10A, a maximum reverse voltage of 45V and a surge non-repetitive forward current of 108A[9].

### MOSFETs

To build a buck-boost-converter it was necessary to use P-channel MOSFETs as the gate of these devices can be driven between ground and the input voltage. If a N-channel MOSFET would have been used it would have required external generation of a signal higher than the input voltage. This is why P-channel MOSFETs are more common in buck-boost-converters. These still required external circuitry to be triggered by the microcontroller, as shown in the section2.2.5 but was still simpler than to generate separate signals. As previously stated with other components, the main limiting factor was the current and heat dissipation.
The final MOSFETs used were the Fairchild FQP27P06[10] which provided a source-drain voltage of 60V and a drain current of -27A @ 25°C case temperature.

[1] http://www.farnell.com/datasheets/2067158.pdf

### 2.2.3 Atmega 328p

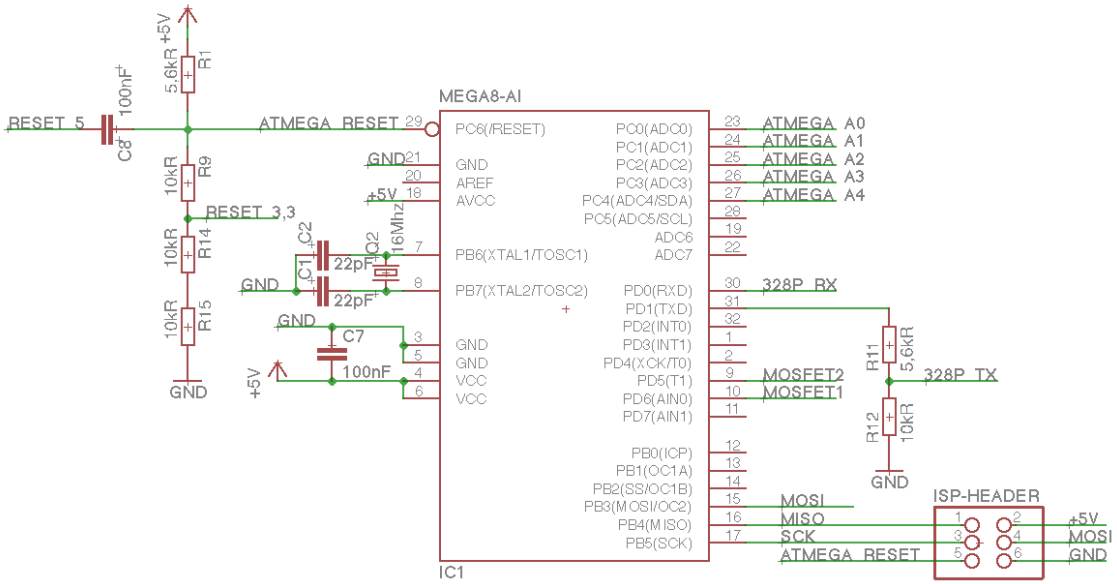


Figure 2.4: Atmega 328p schematic

The final choice that fulfilled all the criteria already mentioned in section 2.1.4 was the ATMEL Atmega 328P[4] microcontroller. To further reduce the power consumption, it would have been possible to supply the chip with 3.3 V instead of 5 V but this would have resulted in a lower clock speed of 8 MHz versus 16 MHz and at the same time lowered the maximum PWM frequency from 62,5kHz down to 31,25kHz. As this microcontroller was already being used to drive the PWM signal, the choice was made to use it as a central hub that would collect and output data to the ESP8266 chip.

### 2.2.4 ESP8266

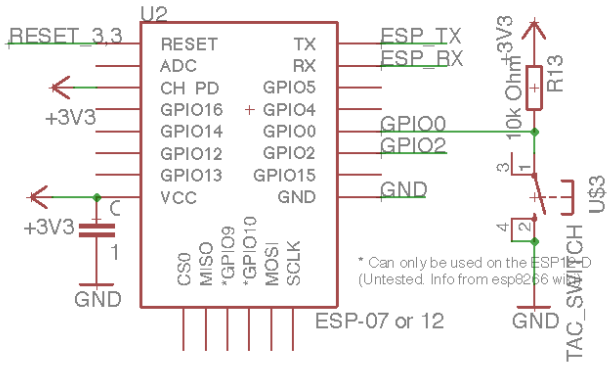


Figure 2.5: Esp8266 schematic

Another thing that had to be considered was how to display the data to the user of the device. This

was done using a Wi-Fi chip as there was a substantial amount of data to display and this was a much more cost-effective solution than to add a physical display. Another cost-effective solution would have been to use an Light Emitting Diode (LED) or a seven-segment display but there was too much data for these types of displays to display all data simultaneously.

The ESP8266 is a Wi-Fi chip and microcontroller combined into one unit with the main purpose of easily implementing Wi-Fi into projects. The chip requires a supply of 3,3V and has a maximum current draw during transmission of 170mA which gives it a maximum power draw of 561mW[5].Figure 2.5 shows the final implementation of the Esp8266 including the switch to set the chips programming mode.

There are multiple low-cost versions of ready to solder boards with this Wi-Fi chip, extra storage and on-board antennas already included. The ESP8266 chips themselves are difficult to come by on their own and would've needed copious amounts of time to implement. As stated in section 2.1.4 there are multiple versions of these boards whose costs fluctuate over time.
A drawback of using the ESP8266 is that the user has to own a device that can connect to the chip and have a display of its own. In this day and age the chances of the user owning such a device are high. Another drawback is the power consumption of the ESP8266, which is higher than on a different type of display, but the power usage is minimal in comparison to the output of the generator. One of the advantages, is the fact that the unit becomes more physically robust as there is no display connected externally to the housing and the remote display can be viewed from any place at any angle within the broadcasting range of the chip. This is a huge advantage since the board and housing is permanently connected between the generator and the output. Another advantage lies in the fact that Graphical User Interfaces (GUIs) can be implemented or data saved to a cloud service and accessed remotely.
For this project a HyperText Markup Language (HTML) page was constructed and broadcast to the client. Within the HTML code a command is implemented which reloads the website every second, thus forcing the client to request the latest data. The output of the data is simple (as can be seen in figure 2.6) but has potential to be expanded using more HTML or javascript in the future.

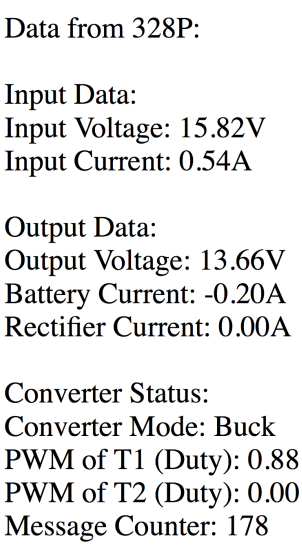


Figure 2.6: GUI screenshot

### 2.2.5 MOSFET-circuit

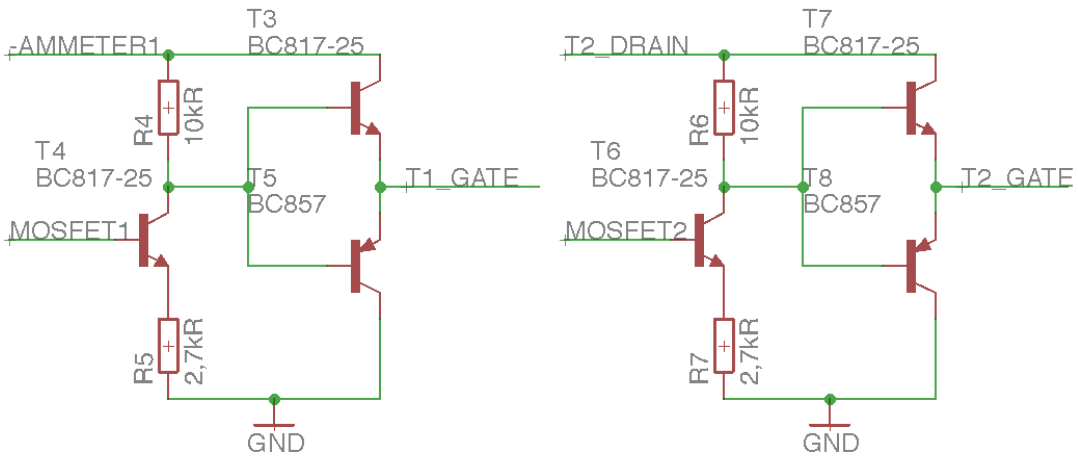


Figure 2.7: MOSFET-circuit schematic

Since it is difficult to drive a P-channel MOSFET using a microcontroller or any low voltage to the gate, it was necessary to devise a circuit that can take the low voltage and low amperage signal line from the microcontroller and driving the P channel MOSFET. This was done using three bipolar transistors and two resistors as can be seen in figure 2.7. Since the MOSFET is a P-channel the logic from the microcontroller is inverted.

This section focuses on circuitry used to power the MOSFETs using a microcontroller. MOSFETs in general act as electronic switches, and each type turns on or off under different circumstances. P-channel MOSFETs, as used in this project, connect source to drain when a low voltage is applied to the gate and disconnect drain and source when a high voltage is applied to the gate. It is important to realize that high and low voltage is always relative to the drain and source voltages. This means that as the input to the buck-boost-converter fluctuates so must the gate of the MOSFET. To turn the MOSFET on the gate-source voltage (Ugs) must be negative [10] and to turn it off again it must exceed this voltage. The problem herein lies in the fact that the microcontroller, which is applying voltage to the gate can only operate within its respective supply voltage, in this case 0 – 5V but the source voltage can range anywhere from 0-24V.
The solution to this problem was to use a series of bipolar transistors and resistors. Only one of the two transistors closest to the gate of the MOSFET is active at once. If both transistors were active at the same time this would result in a short circuit. The base-emitter of both NPN and PNP transistors must be able to withstand 24 V and -24 V, respectively. The resistors within the circuit limit the current flowing to the base of the transistors and ensure that the potential gets pulled to low when the transistor switches between collector and emitter.
The identical circuitry was implemented twice, once for each MOSFET.

### 2.2.6 Measuring Tools

There were multiple values that had to be measured within the circuitry and then delivered to the microcontroller for evaluation. These values include:

- current of the input
- current of each output
- voltage of the input
- voltage of the output

With these metrics, it is possible to calculate a multitude of other values within the microcontroller. For example, the input power, the output power, the overall efficiency of the buck-boost-converter and, most importantly, calculation of the duty cycle and whether to activate the buck or boost mode.

### 2.2.7 Measuring Voltage

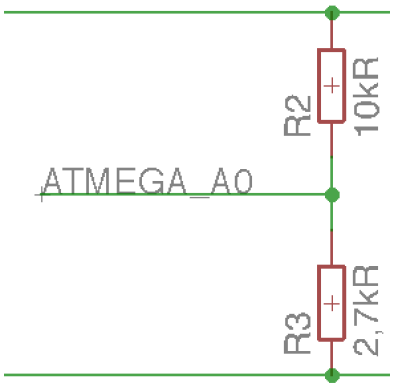


Figure 2.8: Voltage divider schematic

For the microcontroller to regulate the voltage correctly it was necessary to measure the outgoing voltage. This was realized using the simplest way possible by using a voltage divider (seen in figure 2.8) which consists of two resistors that break the voltage down linearly and bring it to a level the microcontroller can read and interpret. The size of the Resistors were calculated as are shown in equation 2.6.

$$R1/(R1+R2)*Vin = VdropR1 = 2,7k\Omega/(2,7k\Omega+10k\Omega)*24V = 5,1V \tag{2.6}$$

This equation shows that at 24V 5,1V will be output to the microcontroller. 5,1V is slightly higher than Vcc (5V) of the 238P but is still well within specification as can be seen in figure 2.9.

The current and voltage were measured using different methods as discussed earlier. The voltage that was measured by a voltage divider needed the ability to map 0-24 V to 0-5 V. Furthermore, the current flowing through the voltage divider had to be high enough to deliver reliable results and

### 30.2 DC Characteristics

**Table 30-1. Common DC characteristics $T_A$ = -40°C to 105°C, $V_{CC}$ = 1.8V to 5.5V (unless otherwise noted)**

| Symbol | Parameter | Condition | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|---|
| $V_{IL}$ | Input Low Voltage, except XTAL1 and RESET pin | $V_{CC}$ = 1.8V - 2.4V<br>$V_{CC}$ = 2.4V - 5.5V | | -0.5<br>-0.5 | | $0.2V_{CC}$(1)<br>$0.3V_{CC}$(1) | V |
| $V_{IH}$ | Input High Voltage, except XTAL1 and RESET pins | $V_{CC}$ = 1.8V - 2.4V<br>$V_{CC}$ = 2.4V - 5.5V | | $0.7V_{CC}$(2)<br>$0.6V_{CC}$(2) | | $V_{CC}$ + 0.5<br>$V_{CC}$ + 0.5 | V |
| $V_{IL1}$ | Input Low Voltage, XTAL1 pin | $V_{CC}$ = 1.8V - 5.5V | | -0.5 | | $0.1V_{CC}$(1) | V |
| $V_{IH1}$ | Input High Voltage, XTAL1 pin | $V_{CC}$ = 1.8V - 2.4V<br>$V_{CC}$ = 2.4V - 5.5V | | $0.8V_{CC}$(2)<br>$0.7V_{CC}$(2) | | $V_{CC}$ + 0.5<br>$V_{CC}$ + 0.5 | V |
| $V_{IL2}$ | Input Low Voltage, RESET pin | $V_{CC}$ = 1.8V - 5.5V | | -0.5 | | $0.1V_{CC}$(1) | V |
| $V_{IH2}$ | Input High Voltage, RESET pin | $V_{CC}$ = 1.8V - 5.5V | | $0.9V_{CC}$(2) | | $V_{CC}$ + 0.5 | V |
| $V_{IL3}$ | Input Low Voltage, RESET pin as I/O | $V_{CC}$ = 1.8V - 2.4V<br>$V_{CC}$ = 2.4V - 5.5V | | -0.5<br>-0.5 | | $0.2V_{CC}$(1)<br>$0.3V_{CC}$(1) | V |
| $V_{IH3}$ | Input High Voltage, RESET pin as I/O | $V_{CC}$ = 1.8V - 2.4V<br>$V_{CC}$ = 2.4V - 5.5V | | $0.7V_{CC}$(2)<br>$0.6V_{CC}$(2) | | $V_{CC}$ + 0.5<br>$V_{CC}$ + 0.5 | V |
| $V_{OL}$ | Output Low Voltage(4) except RESET pin | $I_{OL}$ = 20mA, $V_{CC}$ = 5V | $T_A$=85°C | | | 0.9 | |
| | | | $T_A$=105°C | | | 1.0 | |
| | | $I_{OL}$ = 10mA, $V_{CC}$ = 3V | $T_A$=85°C | | | 0.6 | |
| | | | $T_A$=105°C | | | 0.7 | V |



Figure 2.9: Pin characteristics of 328p [4]

simultaneously not drain a connected battery when no power was being generated at the input as can be seen in equation 2.7

$$Vin/(R1 + R2) = A = 12V/(10k\Omega + 2,7k\Omega) = 944\mu A \tag{2.7}$$

## 2.2.8 Measuring Current

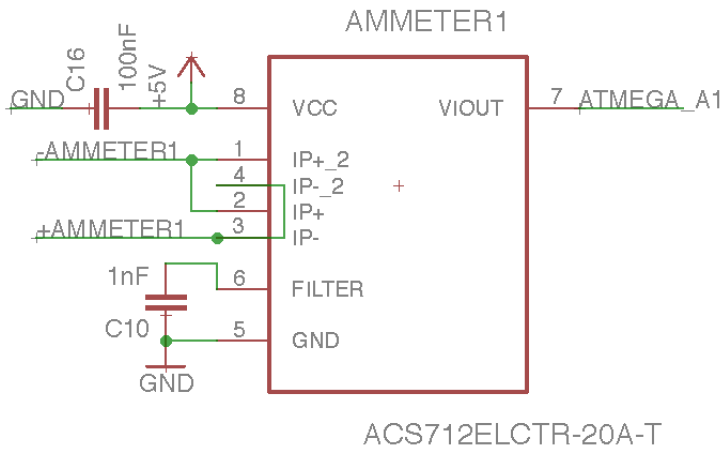


Figure 2.10: Ammeter Schematic

To be able to read the current of the system for each in and output, a form of measuring tool had to be implemented. It would have been possible to do this using a shunt resistor but this would have increased the voltage divider count from 2 to 8, since the voltage would have had to be measured before and after each shunt resistor. Furthermore is was easier to use an Integrated Circuit (IC) as only one ADC line to the microcontroller was used. The final IC that was used was the ACS712 (as seen implemented in figure 2.10) which comes in three versions. Each can measure a maximum current of +/- 5A, 20A, 30A respectively. The 20A version was used as the 5A version would not have the possibility of reading the maximum current of 7,5A and the sensitivity is higher of the 20A (100mV/A) version compared to the 30A (66mV/A) version. The readout voltage ranges from 0,5V to 4V, this is analog to -20A to 20A and leaves 0A at 2,5V [11]. Three of these ammeters were used to monitor the one incoming and two outgoing currents. This is useful to calculate the power, monitor the efficiency of the overall buck-boost-converter and to shut off the system in the event of overheating due to too much current being drawn.

### 2.2.9 Programming of microcontroller

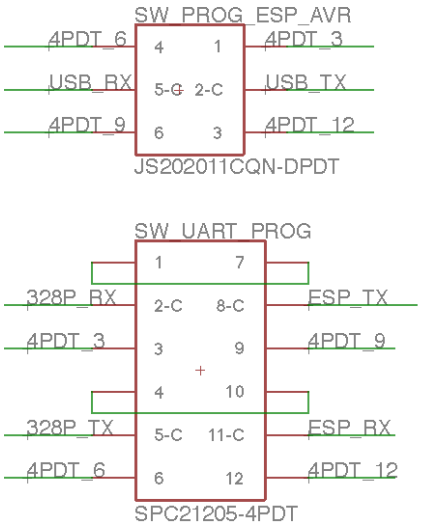


Figure 2.11: Switch Schematic

Since the Wi-Fi chip is also a microcontroller, two microcontrollers were now present on the board which posed the problem that once soldered they would become more difficult to program. The solution to this problem resulted in the usage of two switches. Each microcontroller could be programmed separately, but would still be able to communicate with the other while no programming was necessary. The best implementation proved to be one switch that was dedicated to normal or programming mode. This was achieved using a 4 pole double throw (4PDT) switch that allowed four independent circuits to be toggled using only one stud, similar to using four single switches. The second switch was a double pole double throw (DPDT). Once in programming mode, using the 4PDT switch it was possible to choose the microcontroller to be programmed using the DPDT switch. The implementation was not straight forward as every connection to the RX and TX lines had to be crossed or uncrossed accordingly as can be seen in the figure 2.11. Through this solution, it is now possible to program both chips from a single programmer which is not only convenient but also cost-effective.

### 2.2.10 Power Supply of Digital Components

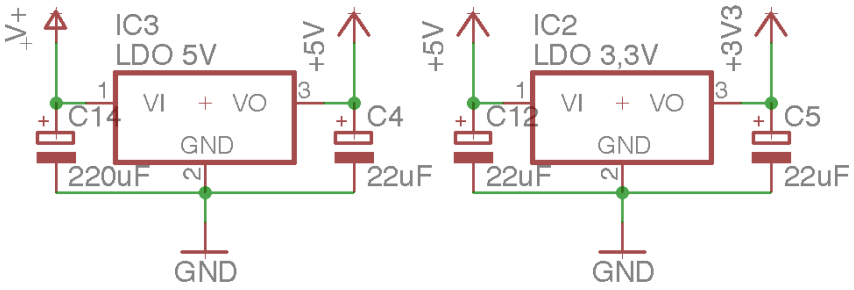


Figure 2.12: LDO Schematic

The microcontrollers and ICs had to be powered using a different source as the voltage coming from the buck-boost-converter was 12V and the digital components all required either 5V or 3,3V. All digital components required 5V except the ESP8266 which required 3,3V. A Low-DropOut (LDO) regulator was used for each of the necessary voltages.
The figure 2.13 shows how the maximum power consumption was calculated. All values were derived either from the datasheets [12] of the component or through direct measurements of the component. The LDOs were then chosen according to the required voltage and maximum power. The final LDOs are the LF33CV and LF50CV which work with an input voltage of maximum 3,3V and 5V, respectively and a maximum output current of 1A which exceeds the necessary amount.

| Power consumption of system | | | | | |
|---|---|---|---|---|---|
| Amount | Component | I [A] | U[V] | P[W] | Total P [W] |
| 4 | ampmeters | 0,013 | 5 | 0,065 | 0,26 |
| 2 | mosfet controls | 0,008 | 5 | 0,04 | 0,08 |
| 1 | esp8266 | 0,17 | 3,3 | 0,561 | 0,561 |
| 1 | atmega 328p | 0,012 | 5 | 0,06 | 0,06 |
| | | | | | |
| Total Amps / Power: | | 0,25 | | | 0,961 |

Figure 2.13: Power Consumption of Components

# 3 Software

The final step to implementing this project was to write the software for each microcontroller that would allow the control of the buck boost converter, the collection of data, the transfer of this data, the communication between the microcontrollers and finally the output of the data to the user.

The first step to realizing the software was to create a protocol that would transfer the necessary data between the two microcontrollers. The communication would be facilitated using UART as this is a robust form of communication. The exchange of information is always facilitated by the Esp8266 as it only requests data when a client does so. This means that the resources of the Atmega 328p are only used to transfer data when really necessary. The protocol is initialized by the ESP8266 by sending a "1" to the Atmega328 which upon receiving answers with the following data:
<IV,IC,OV,BC,RC,M,PWMT1,PWMT2,MC#>

- IV = Input Voltage; The voltage being generated by the generator output in steps between 0-1023
- IC = Input Current; The current flowing from the generator in steps between 0-1023
- OV = Output Voltage; The voltage being generated by the buck-boost-converter in steps between 0-1023
- BC = Battery Current; The current flowing to the Battery connector in steps between 0-1023
- RC = Rectifier Current;The current flowing to the rectifier connector in steps between 0-1023
- M = Mode; What mode the buck-boost-converter is in. 0=off, 1=buck, 2=boost
- PWMT1 = PWM Value of T1; The output to the PWM controlling the gate of T1 in steps from 0-255
- PWMT2 = PWM Value of T2; The output to the PWM controlling the gate of T1 in steps from 0-255
- MC = Message Counter; A counter of how many messages have been sent since boot of uC. This resets at 255 and is mainly for test purposes.

## 3.1 Atmega328p

The code for the Atmega328p was the biggest portion of programming, as this was the central hub for all data and simultaneously controlled the buck-boost-converter. The Arduino IDE[1] was chosen for its ease of use and the possibility to create code very quickly. The explanation of code parts is in the appended file. The adjusting of the PWM for each MOSFET is regulated by measuring the outgoing voltage and. If the output voltage is higher than the target voltage the pwm is reduced by a step, if it is lower the pwm is increased. The choice of mode (buck or boost) is realized by comparing the input voltage to the target voltage. If the input voltage is below the target voltage boost mode is activated and vice versa. During each iteration of the program (each time the pwm is updated) the

program checks the UART input buffer. If any messages have been received and the received value is a "1" all current data is transmitted to the Esp8266.

## 3.2 ESP8266

The code for the ESP8266 required three main components. The communication with the 328p using UART and the handling of access point functionality, communication with the client devices, and building and displaying web-pages to the client using HTML. Again the Arduino IDE[1] was used for its ease of use and availability of example codes. The explanation of code parts is in the appended file. The final GUI can be seen in figure 2.6.

# 4 Results

## 4.1 Final Prototype

The final Prototype can be seen in the image 4.1 and 4.2. This is the board as it is etched, once from each side.

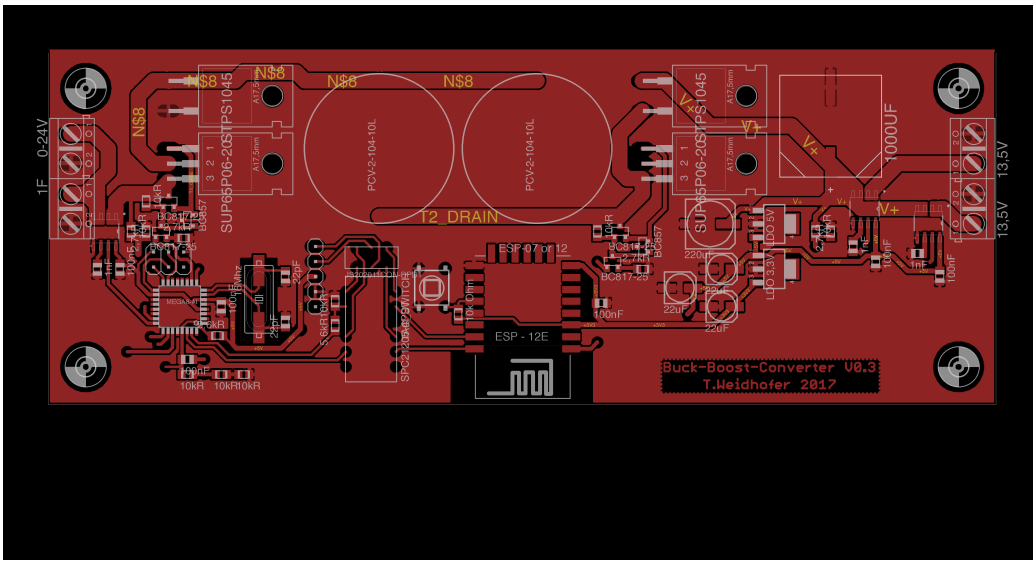


Figure 4.1: Board in top view

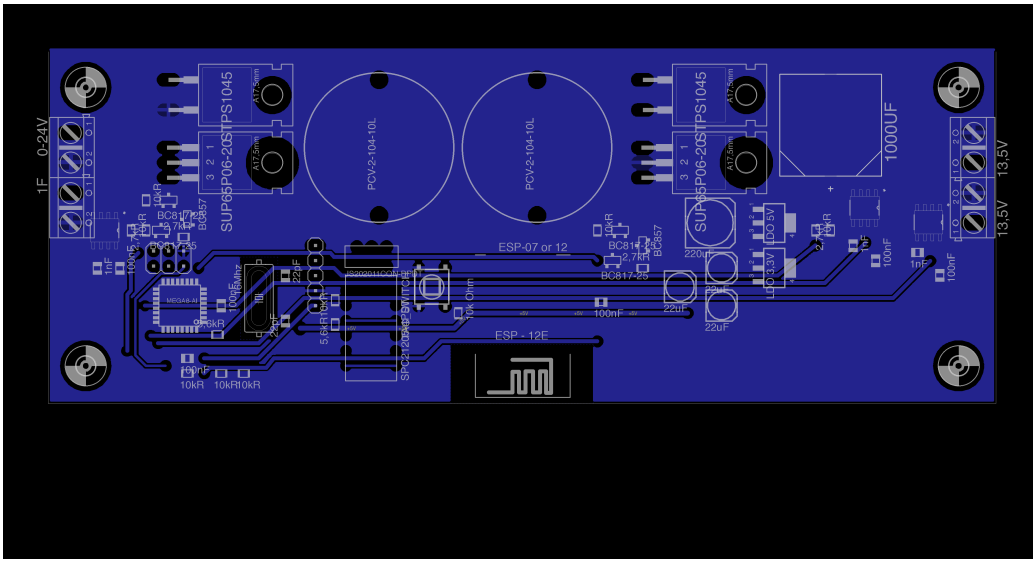


Figure 4.2: Board in bottom view

## 4.2 Measurements

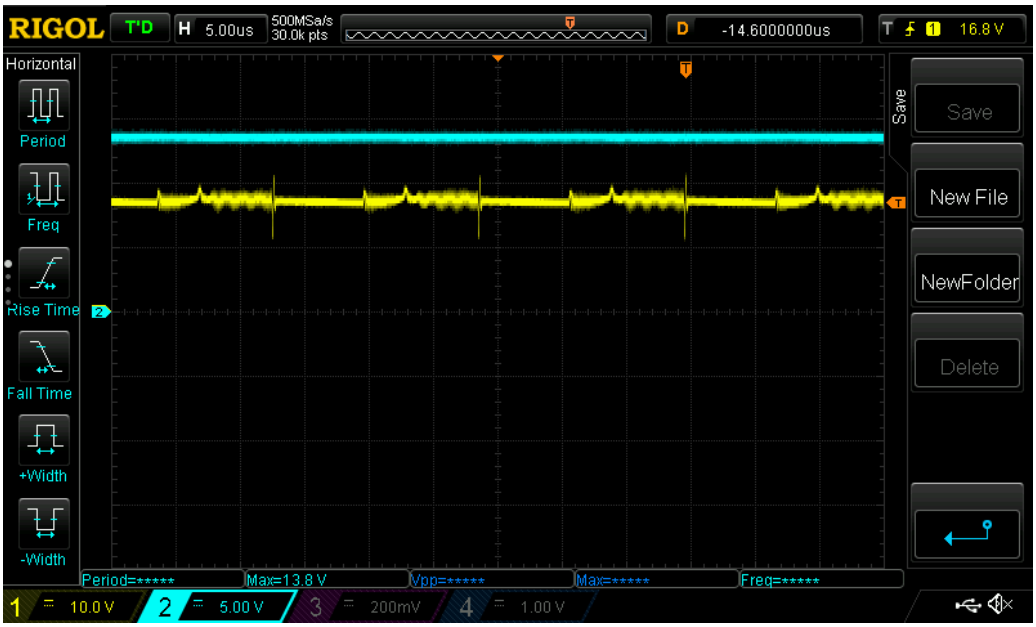


Figure 4.3: Image from oscilloscope with an input voltage of 17V

The measurements seen in figure 4.3 are of the in and output voltages of the buck-boost converter. The output voltage (represented by the blue line) measures approximately 13,5V which was the desired result. The input voltage (represented by the yellow line) was set to 17V which is reflected. The system was correctly running in buck mode as the input voltage was lower than the target voltage.

# 5 Conclusion and Future Prospects

This project has shown that it is possible to create a device that is low cost and can produce enough energy to power devices under 100W and charge batteries. This could find a lot of uses in areas where there is no power grid but enough manpower to generate electricity, possibly for more than one device at a time. The cost of this project can also be kept low as parts (batteries, generators, etc.) can be scavenged from scrapyards or car dumps.
A drawback though of the whole concept is the idea that a bicycle is needed to power this device which, while the PCB itself has a small footprint, the surrounding components including the bicycle do not. This also leads to the next drawback: Cost. As stated earlier it is possible to scavenge components but that does not negate the fact that a PCB is still cost intensive to produce as either CNC machines or etch baths are necessary. Furthermore the availability of components is a factor that may not be forgotten, as areas without power grids are unlikely to have internet availability or telephones where these items could be obtained from.
In conclusion, projects like these have a very specific market, but for the market they are designed for they provide a great benefit.
This project could be refined in the future to implement cloud services or a more sophisticated GUI but at the same time kept simple as to keep costs down and maintain ease of use and implementation.

# Bibliography


[1] arduino.cc. (2017, 02). [Online]. Available: https://www.arduino.cc/en/Main/Software

[2] (2017, 02). [Online]. Available: https://www.arduino.cc/en/Main/arduinoBoardDue

[3] (2017, 02). [Online]. Available: https://www.arduino.cc/en/main/arduinoBoardUno

[4] atmel.com. (2017, 02). [Online]. Available: http://www.atmel.com/images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Complete.pdf

[5] (2017, 02). [Online]. Available: https://cdn-shop.adafruit.com/product-files/2471/0A-ESP8266__Datasheet__EN_v4.3.pdf

[6] [Online]. Available: http://www.ti.com/lit/an/slva535a/slva535a.pdf

[7] [Online]. Available: http://playground.arduino.cc/Code/PwmFrequency

[8] [Online]. Available: http://www.mouser.com/ds/2/597/pcv-463466.pdf

[9] [Online]. Available: http://www.st.com/content/ccc/resource/technical/document/datasheet/a0/25/53/8d/0a/15/4e/6f/CD00000817.pdf/files/CD00000817.pdf/jcr:content/translations/en.CD00000817.pdf

[10] [Online]. Available: https://www.sparkfun.com/datasheets/Components/General/FQP27P06.pdf

[11] [Online]. Available: https://www.sparkfun.com/datasheets/BreakoutBoards/0712.pdf

[12] [Online]. Available: http://www.st.com/content/ccc/resource/technical/document/datasheet/c4/0e/7e/2a/be/bc/4c/bd/CD00000546.pdf/files/CD00000546.pdf/jcr:content/translations/en.CD00000546.pdf

[13] O. Bishop, *Understanding Electrical and Electronics Maths*. Newnes, 1993.

[14] B. Grimm, G. Häberle, H. Häberle, W. Philipp, W. Schleer, B. Schiemann, D. Schnell, and D. Schmid, *Fachkunde Industrieelektronik und Informationstechnik*, 8th ed. Europa Lehrmittel, 2003.

[15] K. Beuth, *Bauelemente Elektronik 2*, 19th ed. Vogel Fachbuch, 2010.

# List of Figures

# List of Abbreviations

**ADC** Analog Digital Converter

**GUI** Graphical User Interface

**HTML** HyperText Markup Language

**IC** Integrated Circuit

**LDO** Low-DropOut

**LED** Light Emitting Diode

**MOSFET** Metal–Oxide–Semiconductor Field-Effect Transistor

**PCB** Printed Circuit Board

**PWM** Pulse Width Modulation

**SMD** Surface Mounted Device

**UART** Universal Asynchronous Receive Transmit

**uC** Microcontroller

# 6 Appendix

## 6.1 Atmega328p code

```
//parts of this code are taken from: https://learn.sparkfun.com/tutorials/esp8266-
#include <ESP8266WiFi.h>

//wifi password
const char WiFiAPPSK[] = "testing";

WiFiServer server(80);

void setup()
{
  setupWiFi();
  server.begin();
  Serial.begin(115200);
}

void loop()
{
  // Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {
    return;
  }

  // Read the first line of the request
  String req = client.readStringUntil('\r');
  client.flush();

  // use 'val' to see if the correct request has been made,
  // if so send a request to recieve data from the Atmegea328P
  int val = -1;
  if (req.indexOf("/read") != -1)
  {
    val = 1;
    Serial.write('1');
  }

  client.flush();

  // Star to build the response. Start with the common header:
  String s = "HTTP/1.1 200 OK\r\n";
```

```
s += "Content-Type: text/html\r\n";
s += "Refresh: 1\r\n";
s += "\r\n";
s += "<!DOCTYPE HTML>\r\n<html>\r\n";

// if data has been transmitted start to read it and output it
// to the string s in an already user friendly format.
if (val == 1)
{
  if (Serial.available() > 0)
  {
    s += "Data from 328P: ";
    s += "<br><br>Input Data:";
    s += "<br>Input Voltage: ";
    s += String(float(Serial.readStringUntil(',').toInt())*0.0234375);
    s += "V";
    s += "<br>Input Current:  ";
    s += String(float((Serial.readStringUntil(',').toInt())-512)*-0.04878);
    s += "A";
    s += "<br><br>Output Data:";
    s += "<br>Output Voltage: ";
    s += String(float(Serial.readStringUntil(',').toInt())*0.0234375);
    s += "V";
    s += "<br>Battery Current:  ";
    s += String(float((Serial.readStringUntil(',').toInt())-512)*-0.04878);
    s += "A";
    s += "<br>Rectifier Current: ";
    s += String(float((Serial.readStringUntil(',').toInt())-512)*-0.04878);
    s += "A";
    s += "<br><br>Converter Status:";
    s += "<br>Converter Mode: ";
    int converterMode = Serial.readStringUntil(',').toInt();
    if(converterMode == 1)
    {s += "Buck ";}
    else if(converterMode == 2)
    {s += "Boost ";}
    else
    {s += "Off or error";}
    s += "<br>PWM of T1 (Duty): ";
    s += String(float(Serial.readStringUntil(',').toInt())*0.003921);
    s += "<br>PWM of T2 (Duty): ";
    s += String(float(Serial.readStringUntil(',').toInt())*0.003921);
    s += "<br>Message Counter: ";
    s += String(Serial.readStringUntil('#'));
  }

  //if no data has come back, eihter the Atmega 328p is busy or the switches are
  else
  {
    s += String("no data, check Uart connections");
  }
```

```
  }
  else
  {
    s += "Invalid Request.<br> Try /read.";
  }
  s += "</html>\n";

  // Send the response to the client
  client.print(s);
  delay(1);

  // The client will be disconnected
  // when the function returns and 'client' object is detroyed
}

//WIFI Setup as per the url at the top of the code
void setupWiFi()
{
  WiFi.mode(WIFI_AP);

  // Do a little work to get a unique-ish name. Append the
  // last two bytes of the MAC (HEX'd) to "Thing-":
  uint8_t mac[WL_MAC_ADDR_LENGTH];
  WiFi.softAPmacAddress(mac);
  String macID = String(mac[WL_MAC_ADDR_LENGTH - 2], HEX) +
                 String(mac[WL_MAC_ADDR_LENGTH - 1], HEX);
  macID.toUpperCase();
  String AP_NameString = "ESP8266 Thing " + macID;

  char AP_NameChar[AP_NameString.length() + 1];
  memset(AP_NameChar, 0, AP_NameString.length() + 1);

  for (int i=0; i<AP_NameString.length(); i++)
    AP_NameChar[i] = AP_NameString.charAt(i);

  WiFi.softAP(AP_NameChar, WiFiAPPSK);
}
```

## 6.2 ESP8266 code

```
#include <SoftwareSerial.h>
//Global definitions
SoftwareSerial ESPserial(2, 3); // RX | TX
int targetOutputVoltage = 582;
int hysteresisInSteps = 5;
int t1GatePin = 6;
int t2GatePin = 5;
int outputVoltagePin = A0;
int inputVoltagePin = A4;
```

```
int inputCurrentPin = A1;
int batteryCurrentPin = A2;
int rectifierCurrentPin = A3;

int outputVoltage = 0;
int inputVoltage = 0;
int inputCurrent = 0;
int batteryCurrent = 0;
int rectifierCurrent = 0;

int mode = 2; // 0 = off, 1 = buck, 2 = boost //buck= t1 pulsing, t2 off //boost= t
t1 off, t2 off

int pwmValueT1 = 1;
int pwmValueT2 = 0;

int count1 = 0;

int incomingByte = 0;

void setup()
{
  pinMode(t2GatePin, OUTPUT);
  digitalWrite(t2GatePin, LOW);
  pinMode(t1GatePin, OUTPUT);
  digitalWrite(t1GatePin, LOW);

  // Analog pins would not have to be declared
  pinMode(outputVoltagePin, INPUT);
  pinMode(inputVoltagePin , INPUT);
  pinMode(inputCurrentPin, INPUT);
  pinMode(batteryCurrentPin, INPUT);
  pinMode(rectifierCurrentPin, INPUT);

  TCCR0B = TCCR0B & 0b11111000 | 0x01;

  Serial.begin(115200);      // communication speed with the host
}


void loop()
{
    //This section checks if there is data in the rx buffer.
    //If there is and it equals "1" the saved data is transfered to the ESP8266
    incomingByte = Serial.read();
    if(incomingByte == '1')
    {
      String serialSend;
      serialSend += String(inputVoltage);
      serialSend += String(',');
      serialSend += String(inputCurrent);
```

```
      serialSend += String(',');
      serialSend += String(outputVoltage);
      serialSend += String(',');
      serialSend += String(batteryCurrent);
      serialSend += String(',');
      serialSend += String(rectifierCurrent);
      serialSend += String(',');
      serialSend += String(mode);
      serialSend += String(',');
      serialSend += String(pwmValueT1);
      serialSend += String(',');
      serialSend += String(pwmValueT2);
      serialSend += String(',');
      serialSend += String(count1);
      serialSend += String('#');
      Serial.print(serialSend);
      count1++;
    }
    //Here data is read out from all the ADC pins and saved to be used later
  outputVoltage    = analogRead(outputVoltagePin);
  inputVoltage     = analogRead(inputVoltagePin);
  inputCurrent     = analogRead(inputCurrentPin);
  batteryCurrent   = analogRead(batteryCurrentPin);
  rectifierCurrent = analogRead(rectifierCurrentPin);

  choosemode();// see function
  pwmSafety(); // see function
  //The current pwm values are transfered to the ADCs which in turn power the mosf
  analogWrite(t2GatePin, pwmValueT2);
  analogWrite(t1GatePin, pwmValueT1);
  delay(5);
}


//One of the two main modes that makes sure that T2 is off and changes the pwmValue
void buckmode(void)
{
  digitalWrite(pwmValueT2, LOW);
  pwmValueT2 = 0;
  if (outputVoltage < (targetOutputVoltage - hysteresisInSteps))
  {
    pwmValueT1 += 1;

  }
  else if (outputVoltage > (targetOutputVoltage + hysteresisInSteps))
  {
    pwmValueT1 -= 1;
  }
  else
  {
    ;
```

```
  }
}
//The other main mode that makes sure that T1 is on and changes the pwmValue of T2
void boostmode(void)
{
  digitalWrite(pwmValueT1, HIGH);
  pwmValueT1 = 255;
  if (outputVoltage <= (targetOutputVoltage - hysteresisInSteps))
  {
    pwmValueT2 -= 1;

  }
  else if (outputVoltage >= (targetOutputVoltage + hysteresisInSteps))
  {
    pwmValueT2 += 1;
  }
  else
  {
    ;
  }
}
//Off mode is called if an emergency stop is necessary.
void offmode(void)
{
  pwmValueT1 = 0;
  pwmValueT2 = 0;

}

//This function sets the mode that is to be executed, depending on the relationship
void choosemode(void)
{
  if (inputVoltage <= targetOutputVoltage) // set boost mode
    {
      mode = 2;
      boostmode();
    }

  else if(inputVoltage > targetOutputVoltage) // set buck mode
    {
      mode = 1;
      buckmode();
    }
  else if(true) //set off mode
    {
      mode = 0;
      offmode();
    }
  else
    {
      ;
```

```
    }

}

//as the pwm values can go below 0 or above 255 this function is called to stop er
void pwmSafety(void)
{
  if (pwmValueT1 > 255)
  {
    pwmValueT1 = 255;
  }

  if (pwmValueT1 < 0)
  {
    pwmValueT1 = 0;
  }

  /*if(mode == 2)
  {
    if (pwmValueT2 > 255  )  //and voltage is higher than 9V  // || pwmValueT2 <=
    {
      pwmValueT2 = 128;
    }
    if (pwmValueT2 < -1 && outputVoltage>300);
    {
      pwmValueT2 = 0;
    }
  }*/

  //else
  {
    if (pwmValueT2 > 255)
    {
      pwmValueT2 = 255;
    }

    if (pwmValueT2 < 0)
    {
      pwmValueT2 = 0;
    }
  }
}
```

## 6.3 Bill of materials

| Qty | Value | Device | Package | Parts | per unit cost | total cost | cost with mwst | supplier | ord nr. | Description |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | PINHD-1X5 | 1X05 | USB-PROGRAMMER | 0,128 | 0,128 | 0,1536 | farnell | 1462952 | PIN HEADER |
| 1 | | PINHD-2X3 | 2X03 | ISP-HEADER | 0,16 | 0,16 | 0,16 | conrad | 1390118 - 62 | PIN HEADER |
| 1 | 1000UF | EEV-FK1H102M-CASE-J16 | CAPAE1700X1700N | C3 | 1,85 | 1,85 | 2,22 | farnell | 2630479 | Aluminum Electrolytic Capacitors,Surface Mount Type |
| 6 | 100nF | C-EUC0805 | C0805 | C7, C8, C11, C13, C15, C16 | 0,0156 | 0,0936 | 0,11232 | farnell | 1759265 | CAPACITOR, European symbol |
| 9 | 10kR | R-EU_M0805 | M0805 | R2, R4, R6, R8, R9, R12, R13, R14, R15 | 0,0058 | 0,0522 | 0,06264 | farnell | 2447553 | RESISTOR, European symbol |
| 1 | 16Mhz | CRYSTALHC49UP | HC49UP | Q2 | 0,192 | 0,192 | 0,2304 | farnell | 2395958 | CRYSTAL |
| 3 | 1nF | C-EUC0805 | C0805 | C6, C9, C10 | 0,0127 | 0,0381 | 0,04572 | farnell | 9406344 | CAPACITOR, European symbol |
| 4 | 2,7kR | R-EU_M0805 | M0805 | R3, R5, R7, R10 | 0,0058 | 0,0232 | 0,02784 | farnell | ------ | RESISTOR, European symbol |
| 1 | 220uF | CPOL-EU153CLV-0807 | 153CLV-0807 | C14 | 0,298 | 0,298 | 0,3576 | farnell | 2611386 | POLARIZED CAPACITOR, European symbol |
| 2 | 22pF | C-EUC0805 | C0805 | C1, C2 | 0,0186 | 0,0372 | 0,04464 | farnell | | CAPACITOR, European symbol |
| 3 | 22uF | CPOL-EU153CLV-0505 | 153CLV-0505 | C4, C5, C12 | 0,0931 | 0,2793 | 0,33516 | farnell | 2611384 | POLARIZED CAPACITOR, European symbol |
| 2 | 5,6kR | R-EU_M0805 | M0805 | R1, R11 | 0,0058 | 0,0116 | 0,01392 | farnell | ------ | RESISTOR, European symbol |
| 3 | ACS712ELCTR-20A-T | ACS712ELCTR-20A-T | SOIC127P600X175-8N | AMMETER1, AMMETER2, AMMETER3 | 4,31 | 12,93 | 15,516 | farnell | 1329624 | Linear Current Sensor |
| 4 | BC817-25 | BC817-25-NPN-SOT23-BEC | SOT23-BEC | T3, T4, T6, T7 | 0,0101 | 0,0404 | 0,04848 | farnell | 1081224 | NPN Transistror |
| 2 | BC857 | BC857ALT1-PNP-SOT23-BEC | SOT23-BEC | T5, T8 | 0,0264 | 0,0528 | 0,06336 | farnell | 2464030 | PNP Transistror |
| 1 | ESP-07 or 12 | ESP12ESMD | ESP12E-SMD | U2 | 3,798 | 3,798 | 3,798 | amazon | B01CCZY2SG | ESP12E Module |
| 1 | JS202011CQN-DPDT | JS202011CQN-DPDT | 2X3_SWITCH | U$1 | 0,394 | 0,394 | 0,4728 | farnell | 2320018 | |
| 1 | LDO 3,3V | MCP1703DB | SOT223 | IC2 | 2,17 | 2,17 | 2,604 | farnell | 9778314 | 250 mA, 16V, Low Quiescent Current LDO Regulator |
| 1 | LDO 5V | MCP1703DB | SOT223 | IC3 | 1,78 | 1,78 | 2,136 | farnell | 1469076 | 250 mA, 16V, Low Quiescent Current LDO Regulator |
| 1 | Atmega328P | Atmega328P | TQFP32-08 | IC1 | 2,79 | 2,79 | 3,348 | farnell | 1715486 | MICROCONTROLLER |
| 2 | PCV-2-104-10L | PCV-2-104-10L | TJ5-U2 | L1, L2 | 5,24 | 10,48 | 12,576 | farnell | 2457690 | |
| 1 | SPC21205-4PDT | SPC21205-4PDT | 2X6_SWITCH | U$2 | 1,78 | 1,78 | 2,136 | farnell | 2543088 | |
| 2 | STPS1045 | STPS1045 | TO220AC | D1, D2 | 0,774 | 1,548 | 1,8576 | farnell | 9803351 | DIODE |
| 2 | FQP27P06 | FQP27P06 | TO220 | T1, T2 | 1,13 | 2,26 | 2,712 | farnell | 9846530 | P-Channel Enhancement MOSFET -60V; 60A; 0,020Ohm |
| 1 | TAC_SWITCH | TAC_SWITCH | TL1105SP | U$3 | 0,24 | 0,24 | 0,24 | conrad | 701749 - 62 | |
| | | | | | | | | | | |
| 1 | PCB 70uM | | | | 6,59 | 6,59 | 6,59 | conrad | 529737 - 62 | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | Total: | 57,86 | | | |

Figure 6.1: Bill of materials