Spring 2024

Battery Powered Rice Cooker

Mateo Casalino , '24

Battery Powered Rice Cooker
Mateo Casalino
ENGR090: Senior Design
May 10, 2024

Table of Contents

Abstract

A small low power rice cooker was modified so that its power source came from an external power supply. This allowed the devicet to be remotely operated from a traditional wall socket. A long testing process was carried out to see in-depth information about the current, power, and voltage draw of the rice cooker over an entire cooking and "keep warm" cycle. From this data, the power draw was integrated to find the total energy consumption of the rice cooker and determined what size and type of battery was needed to be purchase. With both the rice cooker and battery acquired, a housing model was drafted on Fusion360 that could hold both the rice cooker and battery together. This was then manufactured using woodshop tools to make the final rice cooker and battery housing prototype.

Introduction

Rice is a staple grain consumed in my home back on the west coast. Peru, like many other countries around the world, relies heavily on rice in their gastronomy [1]. Whether as the focal point of the dish, such as in Peruvian *Tacu-Tacu*, or as an accompaniment in dishes such as *Lomo Saltado*, very subtly, *arroz blanco con ajo* manages to sneak its way into seemingly every plate in Peruvian cuisine.

But like many other students at Swarthmore College, however, I found myself missing home cooked meals. And in particular, just simple Peruvian-style white rice. And while the rice prepared at the new Dining Centre is tasty in its own regard, I found that the preparation and quality varies greatly from day to day, which left me still longing for more. The clear solution was to prepare it myself in my dormitory, but the college clearly states in the student handbook that "Electrical items using excessive wattage (e.g., air conditioning units) are prohibited", such as a rice cooker, and that "The cooking facilities in residence halls (excluding PPR Apartments) are designed only for occasional snack use and not for regular meal preparation" [2]. So here I was, unable to cook rice on the stovetop on the regular and also prohibited from using a rice cooker using campus electricity.

I decided that a fun solution to this problem would be to modify a pre-existing rice cooker and make it battery powered. This would allow me to prepare rice without putting additional strain on the campus power grid, and also to take my rice cooker on the go. A separate carpentered housing was then designed to hold all the components together for ease of use and transportation.

Technical Discussion

Rice cooker power ratings typically vary by their maximum capacity. For example, a simple conventional rice cooker with a six cup capacity, such as a Zojirushi NHS-10 is rated at 120 V / 500 W [3]. However, a similarly sized rice cooker that utilises fuzzy logic and pressurised cooking to cook the rice uses up to 1,240 watts of power [4]. This is far too much power to be used inside non-NPPR dormitory kitchens. Because as previously mentioned, the College's policy on "personal cooking appliances" and "electrical items using excessive wattage" is that they are prohibited. So to make the rice cooker battery powered, I needed to see how much energy the rice cooker uses in the first place and decide on a battery from there.

Before testing the rice cooker, I had to decide on a rice cooker to work with first. I ended up choosing the Macook Mini Rice Cooker 1-1.5 Cups, which was listed on the packaging to take 200 Watts and 120 Volts. The selection process of why I chose this rice cooker is explained more in the design section that follows.

Through the help of an Arduino Uno and an ADE9153A energy metering shield, I was able to measure the current, power, and voltage draw of the rice cooker I decided to work with over the cooking cycle. To connect the power cable to the shield, I cut the hot-wire inside the cable and connected both sides to the spade plugs marked "live in" and "live to load". More specifically, the side closer to the wall connection was attached to "live in" while "live to load" was connected to the side that would go into the rice cooker. Additionally, a quick splice connector was used to connect the neutral wire to the "neutral in" connector. With everything safely attached, I could start collecting data.

To do this, I had to use an external data logging software called "CoolTerm" that allowed me to save data from the serial monitor directly to a .txt file. This software was crucial to the data analysis process because Arduino natively does not natively have a way to save data from the serial monitor or serial plotter to your computer.

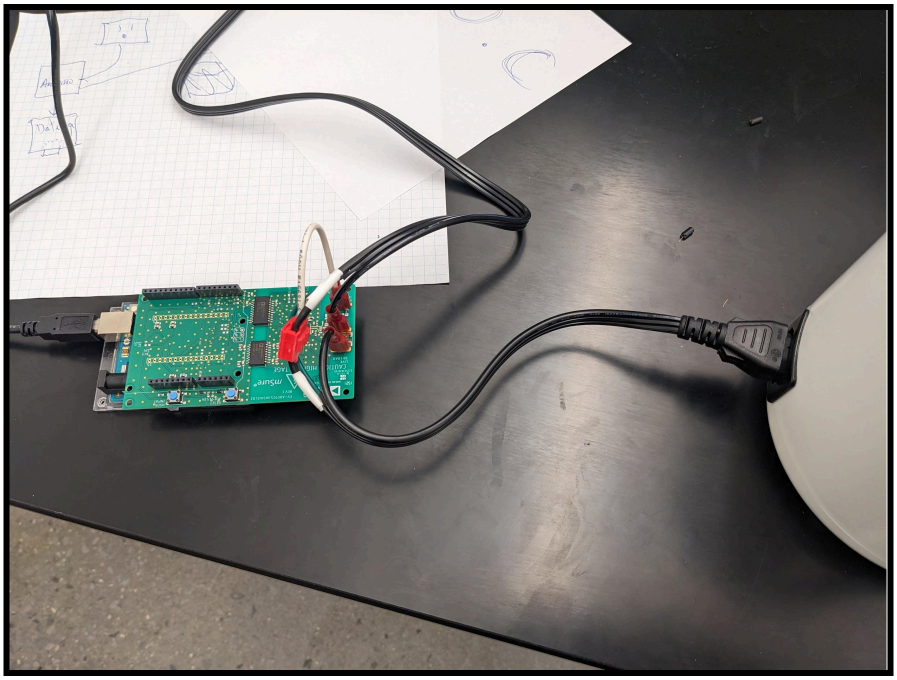
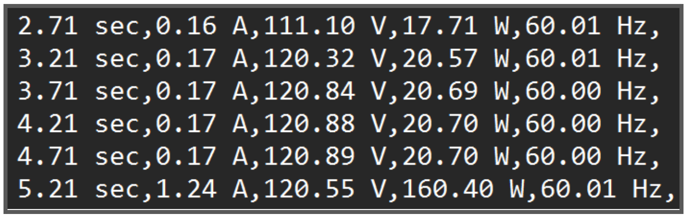


Fig. 1: Arduino and ADE9153A connected to rice cooker

After modifying the example code for the Arduino shield [5], I was able to save the data I needed and upload it to Google sheets. I had the program print out the time, current, voltage, power, and frequency in one line separated by commas. This formatting was important to facilitate using the Text Import Wizard on Excel so that I could migrate the data to Google Sheets



```
2.71 sec,0.16 A,111.10 V,17.71 W,60.01 Hz,
3.21 sec,0.17 A,120.32 V,20.57 W,60.01 Hz,
3.71 sec,0.17 A,120.84 V,20.69 W,60.00 Hz,
4.21 sec,0.17 A,120.88 V,20.70 W,60.00 Hz,
4.71 sec,0.17 A,120.89 V,20.70 W,60.00 Hz,
5.21 sec,1.24 A,120.55 V,160.40 W,60.01 Hz,
```

Fig. 2: 3.5 seconds of data from Arduino

From there, I was able to integrate the power to find the total energy used. Additionally, I was able to plot all the data against time to see how much the variables fluctuate over time.

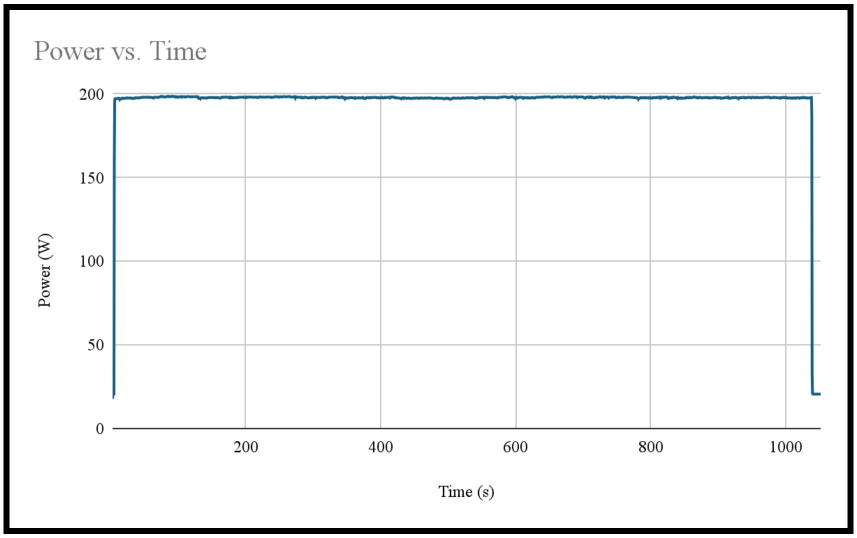


Fig. 3: Power measurement plot of Power (W) vs. Time (s) for 1.5 cups on Macook rice cooker

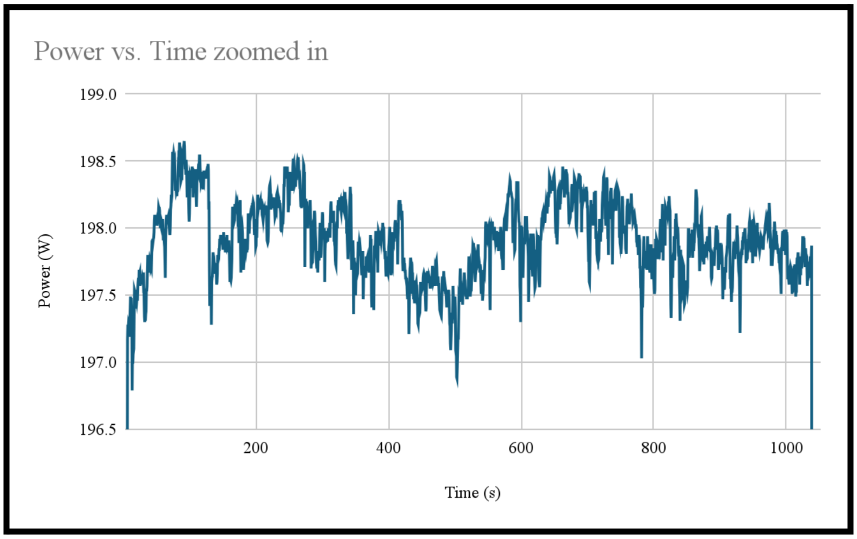


Fig. 4: Zoomed in power measurement plot of Power (W) vs. Time (s) for 1.5 cups on Macook rice cooker

After integrating the power between the start and stop cook time, I found that in total, the rice cooker used about 204439.4885 Joules to cook the rice, or 56.8 Watt-hours (Wh) of energy. This gave me some baseline requirement for my battery that it would need to supply 200W at 120 Volts and have a charge of at least 60 Wh. From these requirements I could search around for a good battery that would fit my needs. I ended up choosing the Powkey Portable Power Station that was rated for 200W of AC power at 120 volts. I chose this battery because it satisfied the power and voltage demands of my rice cooker, but also because it had a 146 watt-hour capacity. This was perfect for my needs because it allowed the rice cooker to be used multiple times on one charge, or to also let the rice cooker stay on the “keep warm” after it is done cooking.

Design

Constraints

The largest constraint on the design of my project was power. I found that standard rice cookers only go as low as 200W, with most hovering between using 300 to 800 Watts according to a study done by the Indian Department of Consumer Affairs [6]. And if I wanted my design to be battery-powered, I needed to go as low-power as possible to minimise the cost of batteries. This constraint simultaneously guided how I selected which rice cooker to base my design around and also how I would choose the batteries for my final design. As previously mentioned, I decided to work with the Macook Mini Rice Cooker because it was only using 200W and satisfied my low power requirements.

Additionally, the Macook Mini Rice Cooker satisfied another constraint that I had which was cooking capacity. I wanted my rice cooker to cook at least 1.5 cups of rice so that it would be useable for a single person to use. A larger cup capacity could have possibly led to a clumsy design that wouldn't have been pleasant to carry, while a smaller cup capacity wouldn't have been practical for food preparation.

In regards to the construction of the housing, I was primarily using medium-density fibreboard (MDF). Due to the structural properties of MDF, it is prone to splitting when screwed into the sides of the material. Because of this, I was constrained to only using glue when attaching all the pieces together.

The last constraint on my project was that all the components had to be within a $400 budget, in compliance with the Swarthmore College Engineering Design Project requirements.

Requirements

The main requirement for my project is that I want my rice cooker to be able to fully cook one serving of rice fully powered by an external battery. One additional requirement that I put in place for my project was to have the rice cooker be able to cook 1.5 cups of rice in under 40 minutes. Typical rice cookers of that size can cook rice in under half an hour, so I want to make sure that making a rice cooker battery-powered won't create more problems than it can solve. The cooking capacity constraint helped satisfy this requirement, since smaller rice cookers tend to cook rice more quickly as they have less water to boil. Another requirement I had was that the housing of the rice cooker would fit in a 1 foot cube. This was so that the end product would not be overly cumbersome for what it was worth. I did not want to create more problems than I would solve.

Standards and codes

One professional code that I wanted to reference during my design process was the National Electrical Code, or NFPA 70. I wanted guidance to make sure that all my wiring for when I was testing the power draw of the rice cooker was up to standard and safe for personal use. I ultimately ended up finding that this code was not very useful for my purposes because the NFPA 70 dealt more with electrical wiring in buildings rather than for at home hobbyist electrical engineers. Instead, I opted to take the same precautions I took during my electrical engineering courses at Swarthmore College that taught me circuit building. I was similarly working with high voltage electronics that I had to be careful when handling them. For example, I would make sure to touch a metal surface for a few seconds to discharge the built up static electricity before interfacing with the circuit board.

Housing design

The initial housing design was made using Fusion 360. This was so I could create a rough draft of what I wanted the physical model to look like when I was building it. The actual dimensions that I plugged into my design on Fusion didn't matter as much as the proportions of the pieces relative to each other. This was because for my purpose, having all the cuts match each other in the mattered more than the actual dimensions itself. This meant not moving the table saw guide between cuts so that all my pieces were the same length precisely. Below is a photo of what I wanted my housing to look like in brown, with the white cylinder representing the rice cooker and the black box signifying the battery.

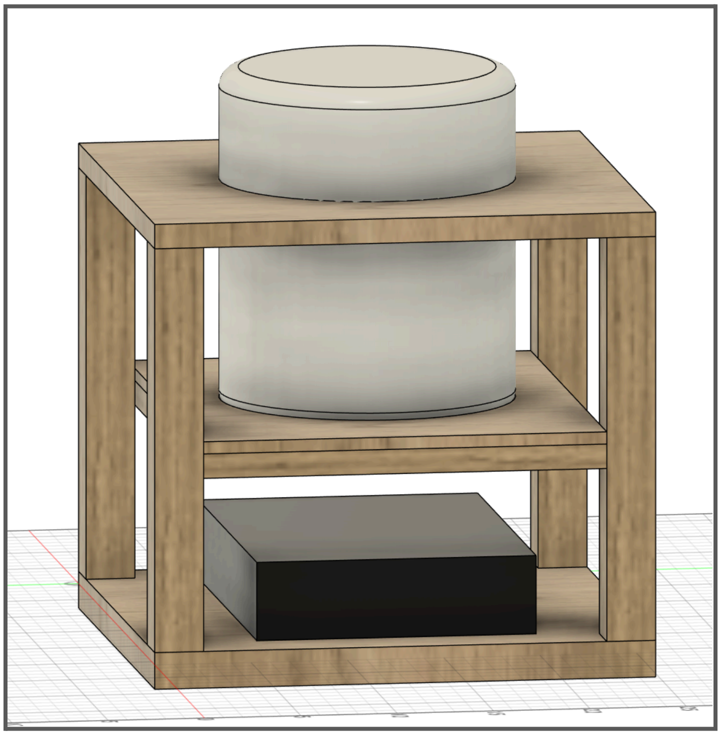


Fig. 5: Fusion 360 model of my housing design prototype

The housing was designed so that both the rice cooker and battery would fit snugly in their shelf levels. I anticipated achieving this effect by making laser-cut profiles of the two components that they could rest in and glueing them to the platforms.

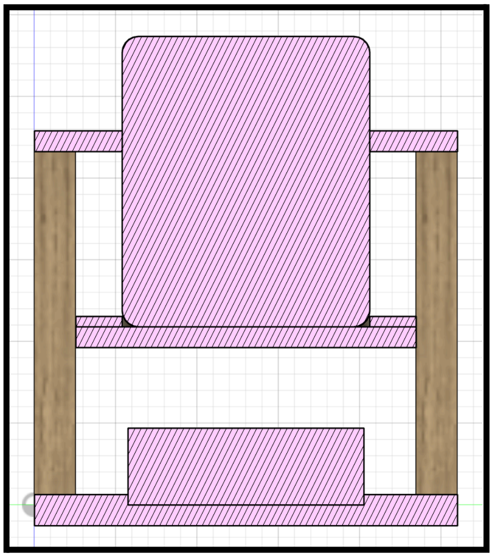


Fig. 6: Cross sectional view of the housing design

Results and Discussion

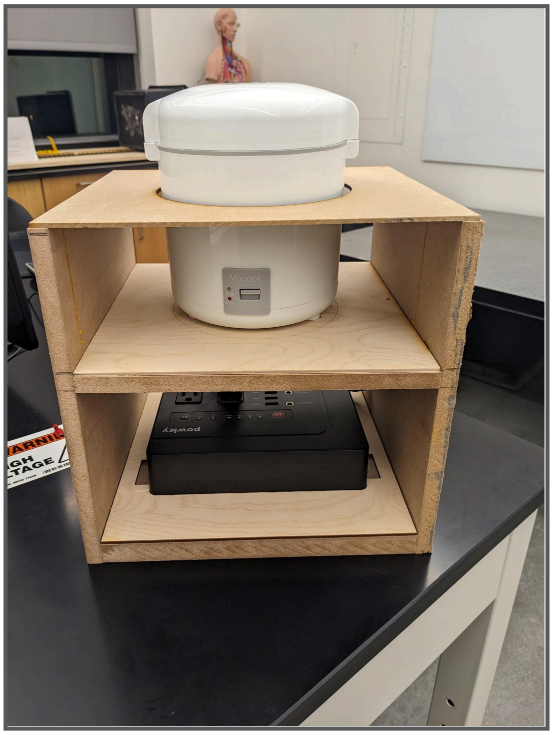


Fig. 7: Battery powered rice cooker inside housing

After constructing the final housing and putting all the components together, I had a final prototype of my design. The final design differs slightly from the original mockup that I had made on Fusion 360. Initially, I had wanted to build the housing using a four-post design that would maximise airflow over the rice cooker and battery. But when I got to thinking about how all the pieces of MDF would fit together, I quickly realised that my original design would be hard to build and not structurally sound, especially if I wanted to pick up the carrier from anywhere that wasn't the bottom base. Instead, I opted for a side panel design that would maximise glueing surface area for the support panels that the components would rest on. Additionally, the side panels made assembly much easier by having more flat surfaces to justify the pieces against while letting the glue dry. So while this design restricted some airflow, it more than made it up in ease of construction and structural stability.

The housing design ended up being relatively compact. Including the space taken up by the rice cooker sticking out the top, the housing's dimensions were 28.8 cm x 26.1 cm x 27.8 cm. Currently, there is just empty space in front and behind both the rice cooker and battery, so some area could have been saved if I cut the flooring to trim the excess off. In terms of weight, the housing weighed only 7 pounds and 4.5 oz, where the rice cooker and battery combined weighed in at 12 pounds and 13.4 oz.

But while the housing is relatively compact and lightweight, I would have liked to have made it more portable. Currently, there isn't really a convenient way to carry the housing that is not just holding it from the bottom. In the future, I would like to screw some side handles by the top of the housing into the side walls so that it can be easier carried from the top. Despite the lack of handles, however, the housing still feels quite structurally sound and adept at holding all the components together. The laser cut profiles of the rice cooker and battery especially do a good job at preventing the pieces from sliding around when carried.

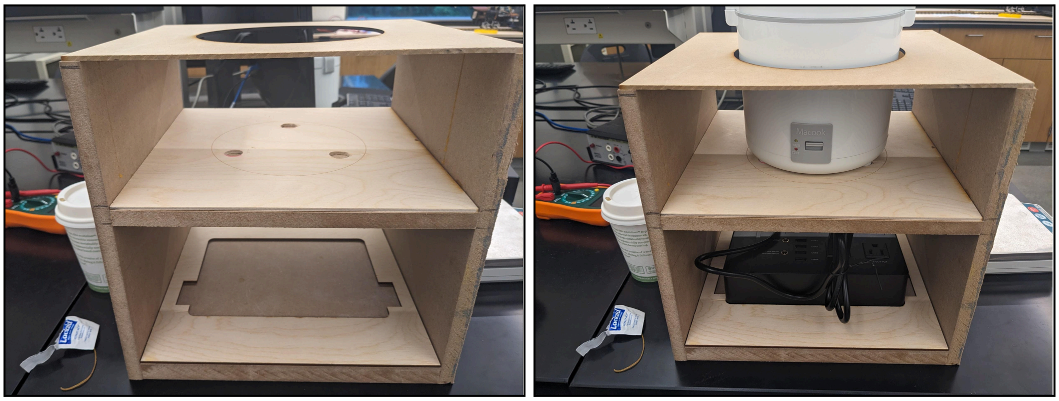


Fig. 8 and 9: Holding frames for rice cooker and battery

When it came to actual performance, I found that the rice cooker operated almost identically when placed in the housing and hooked up to the battery compared to normal wall connection performance. After performing a simple timed test using the same lab setup, I saw that the rice cooker took only 18 seconds longer than the original 17 minute and 15 second time in my first test. This is likely due to the random nature of cooking rice and how the variation of rice density in the rice cooker can alter how quickly the water heats up and evaporates.

Conclusion

In accordance with the requirements I initially put on my design, the overall project can be considered a success since a battery powered rice cooker was ultimately made. The finished design was able to cook rice well under my forty minute threshold, coming in at only 17 minutes and 33 seconds. The rice cooker was also able to cook 1.5 cups exactly, although the quality of the cooked rice suffered when pushed to capacity. Practically speaking in the future, I would likely only cook one cup of rice to obtain my preferred texture. When cooking 1.5 cups, the rice ended up too dry to my personal liking. The final design also ended up fitting in the one foot cube I had originally required my housing to fit inside. But while this did partially help my efforts in trying to make a portable housing design, further work would need to be done to make the housing effectively moveable.

In regards to future work, I would like to strip back the individual casings of the rice cooker and battery so I could more easily integrate them together in a preferably heat moulded case that would make the product more of a "battery powered rice cooker" than a "battery attached rice cooker". Additionally, holding the two components closer together in a tighter housing would also help with the current portability issues.

This project did still do a good job at being a proof of concept prototype of what battery powered cooking appliances could look like. The design of a rice cooker is really intelligent because they are able to automatically stop cooking once all the liquid inside has absorbed or evaporated. This means a battery powered rice cooker could also become a battery powered vegetable steamer or even a cake baker. So even if my project doesn't move beyond cooking rice, it was still a good proof of concept that the built-in "keep warm" rice cooker function can be powered remotely, which could provide legitimate practical use for transporting food or catering companies. Currently, caterers mainly rely on insulated pans to keep food warm for transportation to the destination [7]. Innovations in the use of battery technology could allow caterers to keep food within a safe temperature range for longer periods of time, showing how a fully battery powered rice cooker has implications beyond the scope of this project.

Acknowledgements

I would like to take the space here to acknowledge all the people that helped me with carrying out my project. First I would like to acknowledge my project advisor, Professor Lynne Molter. Her help was instrumental in helping me plan my project and keeping it on track. J. Johnson and Joshua Jordan should also both be thanked for allowing me to use their respective shops to manufacture my housing design in the later stages of the project. Also Ed Jaoudi should be thanked for all the help he gave with the electronics side of my project and with helping me find the correct products I needed. Cassy Burnett should also be thanked for helping me order all the parts for my project in a quick and timely manner. And lastly, I'd like to thank all my friends and family who stuck by me while I was working on this project for the past months.

Bibliography

[1] Carolina Rice, “A Guide To Peruvian Food With Recipes,” *Carolina® Rice*, Nov. 03, 2021. https://carolinarice.com/cooking/peruvian-food-quick-guide/ (accessed May 10, 2024).

[2] Swarthmore College, “Student Handbook,” *www.swarthmore.edu*, 2023. https://www.swarthmore.edu/student-handbook (accessed May 10, 2024).

[3] Zojirushi, “Rice Cooker / Steamer NHS-06/10/18,” *Zojirushi Online Store*. https://shop.zojirushi.com/products/nhs?variant=42422576939230 (accessed May 10, 2024).

[4] Zojirushi, “Pressure Induction Heating Rice Cooker & Warmer NP-NWC10/18,” *Zojirushi Online Store*. https://shop.zojirushi.com/products/npnwc (accessed May 10, 2024).

[5] analogdevicesinc, “ADE9153AAPI_Test.ino,” *GitHub*, Jan. 08, 2018. https://github.com/analogdevicesinc/arduino/blob/master/Arduino%20Uno%20R3/examples/ADE9153A_examples/ADE9153AAPI_Test/ADE9153AAPI_Test.ino (accessed May 10, 2024).

[6] Department of Consumer Affairs, “Electric Rice Cookers: One kitchen chore made easy,” Dec. 2013. Accessed: May 10, 2024. [Online]. Available: https://consumeraffairs.nic.in/sites/default/files/file-uploads/ctocpas/ElectricRiceCooker_13.pdf

[7] Twin Oaks Caterers, “What are the tips for keeping the catered food warm and not overcooking it?,” *twinoakscaterers.com*, Sep. 29, 2023. https://twinoakscaterers.com/keeping-catered-food-warm/ (accessed May 10, 2024).

Appendix

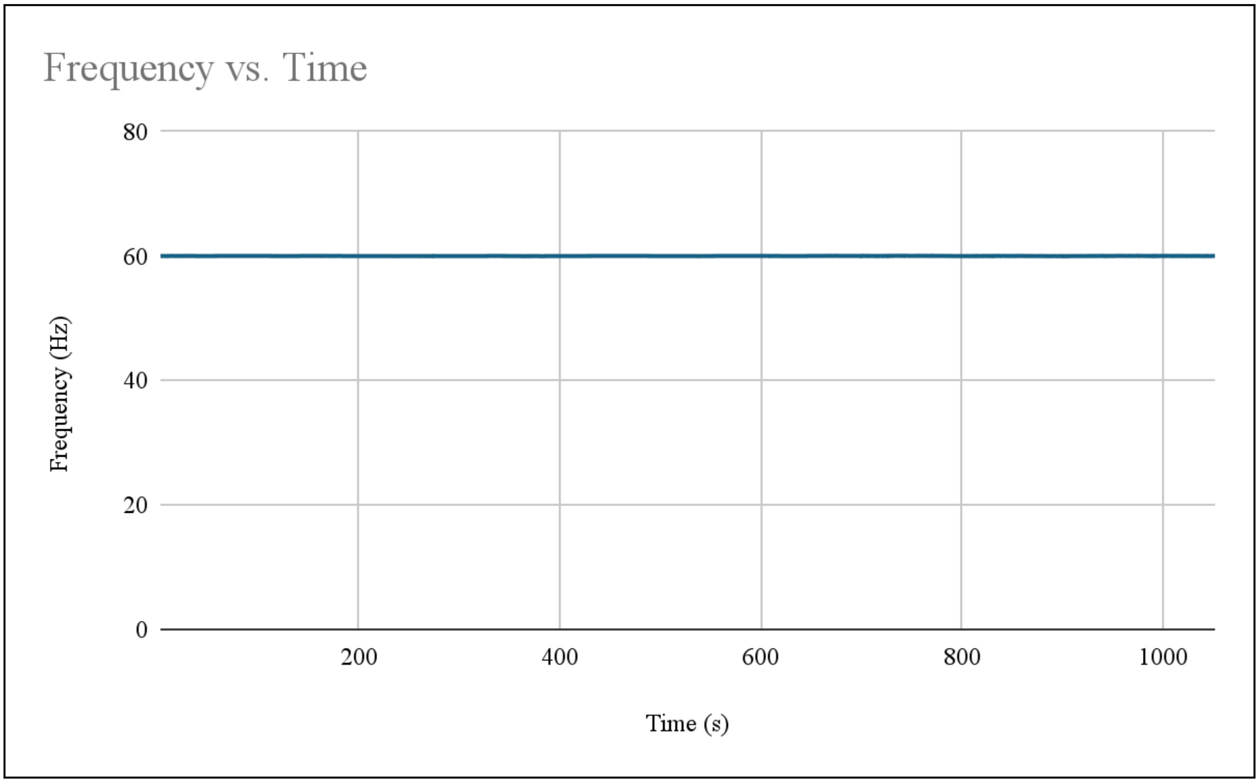
Arduino code:

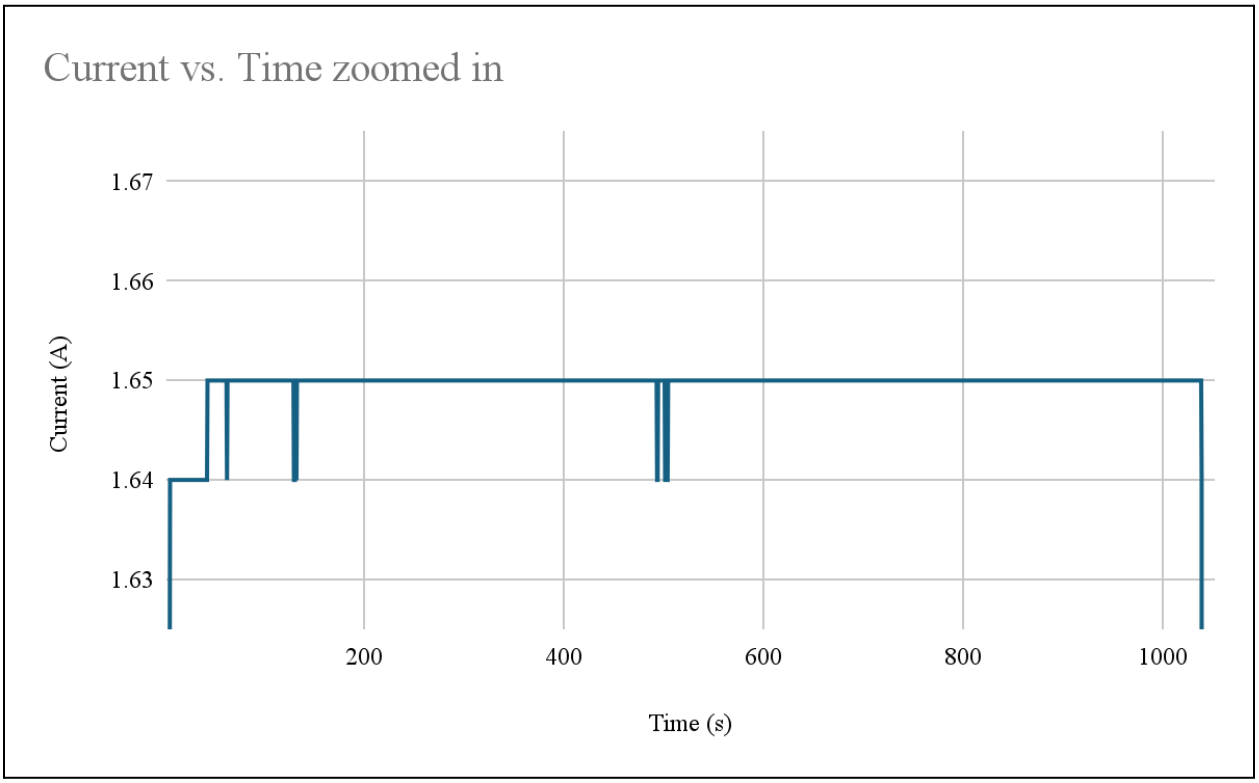
```
#include <ADE9153A.h>
#include <ADE9153AAPI.h>

/*
 * Test Code for the ADE9153AAPI
 *
 * Designed specifically to work with the EV_ADE9153ASHIELDZ board and
 * ADE9153AAPI library
 * ---- http://www.analog.com/ADE9153A
 *
 * Created by David Lath for Analog Devices Inc., January 8, 2018
 *
 * Copyright (c) 2018, Analog Devices, Inc.  All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted (subject to the limitations in the disclaimer
 * below) provided that the following conditions are met:
 *
 * * Redistributions of source code must retain the above copyright notice,
 * this list of conditions and the following disclaimer.
 *
 * * Redistributions in binary form must reproduce the above copyright
 * notice, this list of conditions and the following disclaimer in the
 * documentation and/or other materials provided with the distribution.
 *
 * * Neither the name of Analog Devices, Inc. nor the names of its
 * contributors may be used to endorse or promote products derived from this
 * software without specific prior written permission.
 *
 * NO EXPRESS OR IMPLIED LICENSES TO ANY PARTY'S PATENT RIGHTS ARE GRANTED
 * BY THIS LICENSE.  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND
 * CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING,
 * BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
 * FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 * OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED
 * TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
 * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
 * NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
 * SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */
#define ARM_MATH_CM0PLUS

#include <SPI.h>
#include  <ADE9153A.h>
#include <ADE9153AAPI.h>

/* Basic initializations */
#define SPI_SPEED 1000000     //SPI Speed
#define CS_PIN 8              //8-->Arduino Zero. 15-->ESP8266
#define ADE9153A_RESET_PIN 4  //On-board Reset Pin
#define USER_INPUT 5          //On-board User Input Button Pin
#define LED 6                 //On-board LED pin
ADE9153AClass ade9153A;
```

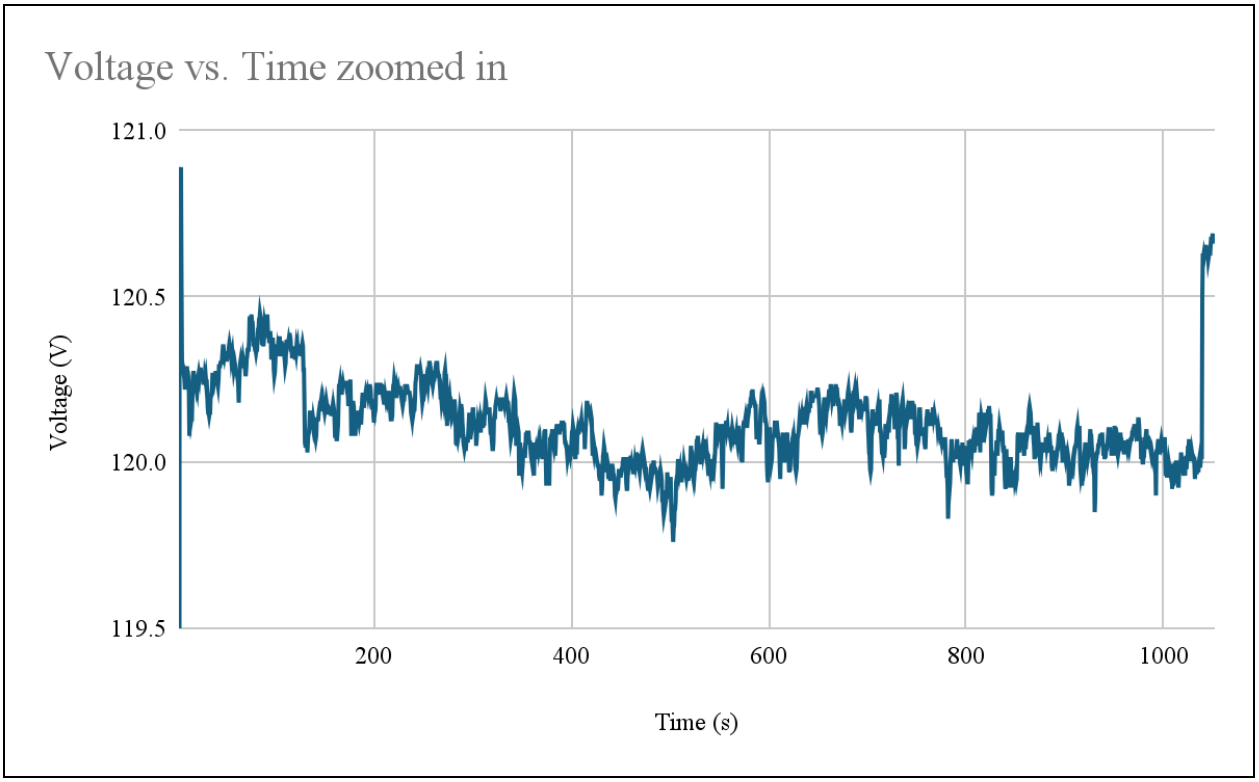
```
struct EnergyRegs energyVals;  //Energy register values are read and stored in
EnergyRegs structure
struct PowerRegs powerVals;    //Metrology data can be accessed from these structures
struct RMSRegs rmsVals;
struct PQRegs pqVals;
struct AcalRegs acalVals;
struct Temperature tempVal;

void readandwrite(void);
void resetADE9153A(void);

int ledState = LOW;
int inputState = LOW;
unsigned long lastReport = 0;
const long reportInterval = 500;
const long blinkInterval = 500;

void setup() {
  /* Pin and serial monitor setup */
  pinMode(LED, OUTPUT);
  pinMode(USER_INPUT, INPUT);
  pinMode(ADE9153A_RESET_PIN, OUTPUT);
  digitalWrite(ADE9153A_RESET_PIN, HIGH);
  Serial.begin(115200);

  resetADE9153A();            //Reset ADE9153A for clean startup
  delay(1000);
  /*SPI initialization and test*/
  bool commscheck = ade9153A.SPI_Init(SPI_SPEED,CS_PIN); //Initialize SPI
  if (!commscheck) {
    Serial.println("ADE9153A Shield not detected. Plug in Shield and reset the
Arduino");
    while (!commscheck) {     //Hold until arduino is reset
      delay(1000);
    }
  }

  ade9153A.SetupADE9153A(); //Setup ADE9153A according to ADE9153AAPI.h
  /* Read and Print Specific Register using ADE9153A SPI Library */
  Serial.println(String(ade9153A.SPI_Read_32(REG_VERSION_PRODUCT), HEX)); // Version
of IC
  ade9153A.SPI_Write_32(REG_AIGAIN, -268435456); //AIGAIN to -1 to account for IAP-IAN
swap
  delay(500);
}

void loop() {
  /* Main loop */
  /* Returns metrology to the serial monitor and waits for USER_INPUT button press to
run autocal */
  unsigned long currentReport = millis();

  if ((currentReport - lastReport) >= reportInterval){
    lastReport = currentReport;
    readandwrite();
  }

  inputState = digitalRead(USER_INPUT);

  if (inputState == LOW) {
    Serial.println("Autocalibrating Current Channel");
    ade9153A.StartAcal_AINormal();
    runLength(20);
    ade9153A.StopAcal();
    Serial.println("Autocalibrating Voltage Channel");
    ade9153A.StartAcal_AV();
    runLength(40);
    ade9153A.StopAcal();
    delay(100);

    ade9153A.ReadAcalRegs(&acalVals);
    Serial.print("AICC: ");
    Serial.println(acalVals.AICC);
    Serial.print("AICERT: ");
    Serial.println(acalVals.AcalAICERTReg);
    Serial.print("AVCC: ");
    Serial.println(acalVals.AVCC);
    Serial.print("AVCERT: ");
    Serial.println(acalVals.AcalAVCERTReg);
    long Igain = (-(acalVals.AICC / 838.190) - 1) * 134217728;
    long Vgain = ((acalVals.AVCC / 13411.05) - 1) * 134217728;
    ade9153A.SPI_Write_32(REG_AIGAIN, Igain);
    ade9153A.SPI_Write_32(REG_AVGAIN, Vgain);

    Serial.println("Autocalibration Complete");
    delay(2000);
  }
}

void readandwrite()
{
 /* Read and Print WATT Register using ADE9153A Read Library */
  ade9153A.ReadPowerRegs(&powerVals);    //Template to read Power registers from
ADE9000 and store data in Arduino MCU
  ade9153A.ReadRMSRegs(&rmsVals);
  ade9153A.ReadPQRegs(&pqVals);
  ade9153A.ReadTemperature(&tempVal);

//  Serial.print("RMS Current:\t");
  Serial.print(millis()/1000.0);
  Serial.print(" sec,");

  Serial.print(rmsVals.CurrentRMSValue/1000);
  Serial.print(" A,");

//  Serial.print("RMS Voltage:\t");
  Serial.print(rmsVals.VoltageRMSValue/1000);
  Serial.print(" V,");

//  Serial.print("Active Power:\t");
  Serial.print(powerVals.ActivePowerValue/1000);
  Serial.print(" W,");

//  Serial.print("Reactive Power:\t");
//  Serial.print(powerVals.FundReactivePowerValue/1000);
//  Serial.println(" VAR");

//  Serial.print("Apparent Power:\t");
//  Serial.print(powerVals.ApparentPowerValue/1000);
//  Serial.println(" VA");

//  Serial.print("Power Factor:\t");
//  Serial.print(pqVals.PowerFactorValue);
//  Serial.print(" PF,");

//  Serial.print("Frequency:\t");
```

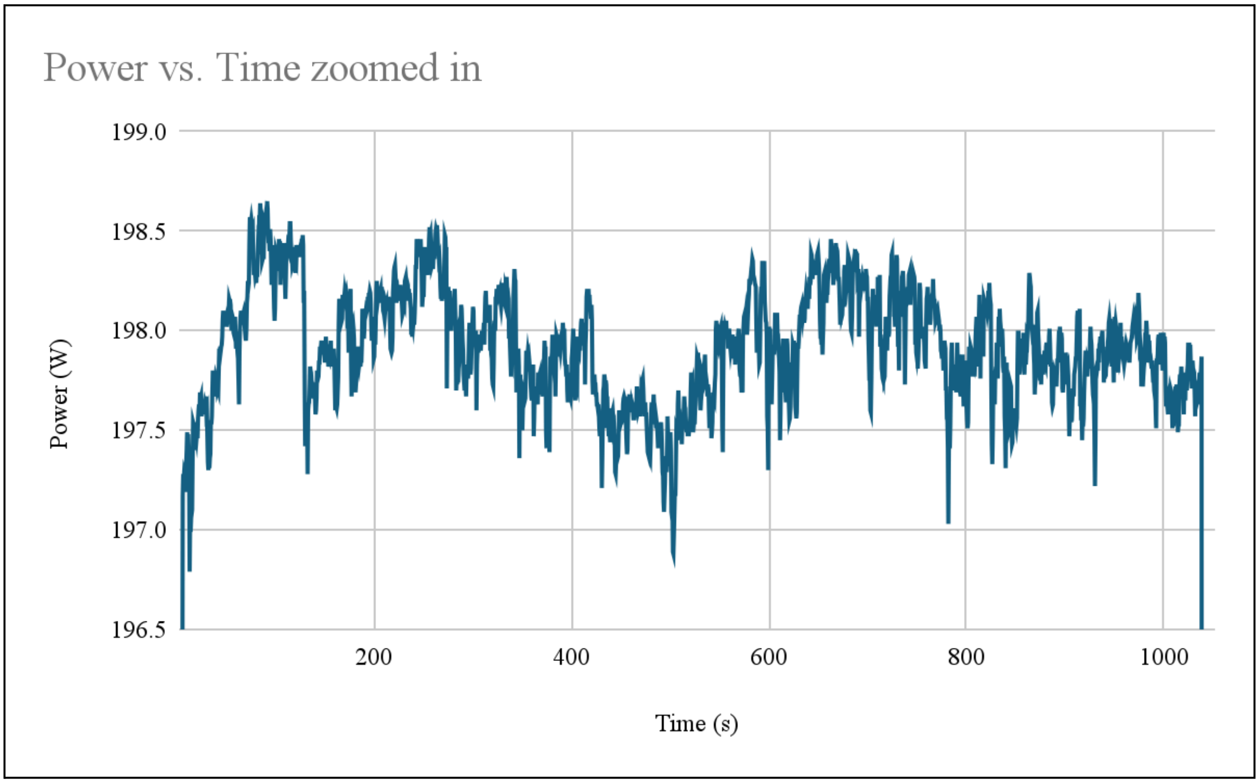
```
  Serial.print(pqVals.FrequencyValue);
  Serial.println(" Hz,");

//  Serial.print("Temperature:\t");
//  Serial.print(tempVal.TemperatureVal);
//  Serial.println(" degC");


//  Serial.println("");
//  Serial.println("");
}

void resetADE9153A(void)
{
 digitalWrite(ADE9153A_RESET_PIN, LOW);
 delay(100);
 digitalWrite(ADE9153A_RESET_PIN, HIGH);
 delay(1000);
 Serial.println("Reset Done");
}

void runLength(long seconds)
{
  unsigned long startTime = millis();

  while (millis() - startTime < (seconds*1000)){
    digitalWrite(LED, HIGH);
    delay(blinkInterval);
    digitalWrite(LED, LOW);
    delay(blinkInterval);
  }
}
```

Energy test plots for 1.5 cups of sushi rice:






Voltage vs. Time zoomed in
121.0
120.5
120.0
119.5
Voltage (V)
200
400
600
800
1000
Time (s)


Power vs. Time zoomed in
199.0
198.5
198.0
197.5
197.0
196.5
Power (W)
200
400
600
800
1000
Time (s)