Miami University ECE 448/9 Senior Design Project Report

Power Measurement of a Computing System (Fall 2021/Spring 2022)

By: Owen Hardy ('22), Sam Rutschilling ('22), Jordan Smith ('22)

Dr. Peter Jamieson, Dr. Mark Scott (Advisors)

Table of Contents

Abstract

The original goal of this project was to create a power measurement system that could log the power consumption of various perophirals plugged into a power strip. This would give an idea of how much power each piece of equipment was using at any given time.

Before work on the project began, we changed the overall scope of the project to determine the power consumption on a Raspberry Pi for various Python and Objective-C scripts. These scripts will call several different functions including sending email messages, writing to files, accessing webpages, and performing basic arithmetic operations. This allows us to understand the power consumption of small computing tasks and their cost. The ultimate goal of this is to understand how much power consumption certain automatic background tasks use on computers and other devices to determine if the slight added benefit of these operations is truly worth it.

Introduction

Every single process running on a computer consumes power, no matter how small it may seem. Even adding a single line of code or an image to a document costs power, or adding a small logo in the footnote of an email can cost extra. While this extra power draw may seem miniscule, it can add up over time and with multiple people connected to the same server, the cost multiplies with each user. So, the question must be asked: are these small processes anything to worry about? Is the extra power bump from running these processes significant and should they be kept to a minimum? (lol, this probably needs altered quite a bit)

Project Background

Project Research

Solution Implementation

After we determined that using the Texas Instruments INA219 chip would be an appropriate current and power measurement a method to record the data output by it was needed. The INA 219 utilizes I2C communication so an Arduino Uno was used to communicate between the sensor and a PC used for data collection. On the PC, we created a python script using the Pyserial library to store the data streaming in from the sensor.

We decided to use SQLITE to store the data instead of other formats such as .txt (tag delimted text) or .csv (comma seperated value) for several reasons: SQLITE stores data in columns and tables so it is much less likely to become corrupt if it is incorrectly closed or not closed at all. SQLITE also allows easy storage of the date / time value as well, which allows us to store the exact time of a data point. SQLITE also allows easy searching and calculation of averages of data sets as it allows use of all common SQL (structured query language) querys and commands. This allows calculation of the average idle and operation values using just a single query line in the open source DB Browser for SQLITE. These same queries were also used to calculate the time values for each of these scripts. That data was then used to calculate the Power consumption for each computing event event.

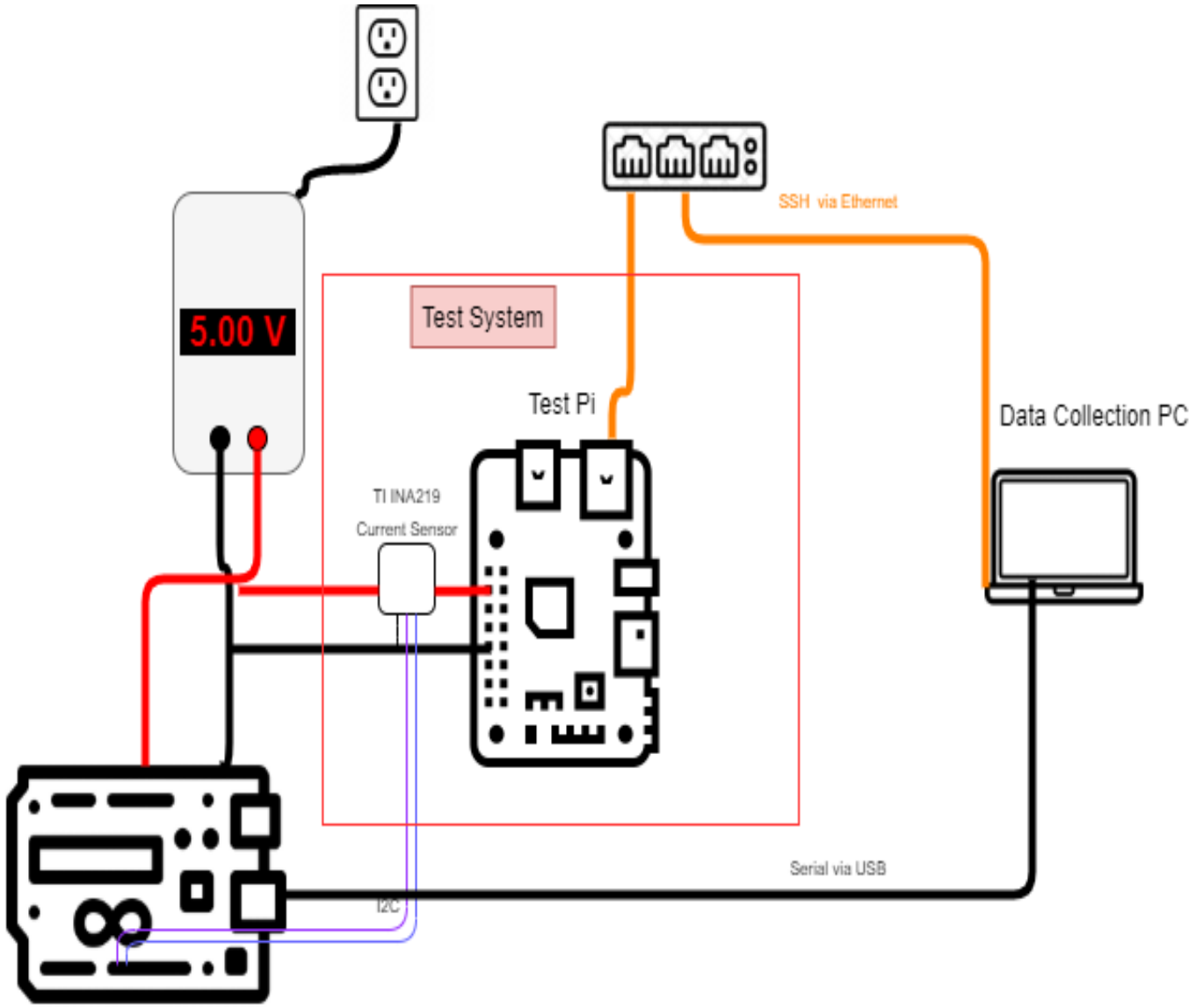

Figure 1: Diagram of the complete computing system. The measured system is the Raspberry Pi 3B+ at the center of the layout. The arduino is used for data collection, and the router is used for communicating with the raspberry pi as well as planning for future use.

Most of the data collected was collected with a timing of 10ms, which means for every second there are 100 data points, which means that there are several thousand data points for most of the trials. This frequency of data collection is high enough to create enough data so that we can be confident our results are a true representation of the systems power consumption.

In the recorded data for each of the trials some jumps and increases can be seen in the data, especially when viewed visually as below. These are background tasks operating on the OS level and should not cause an issue as they are present in both the average baseline power consumption and the average operation power consumption.

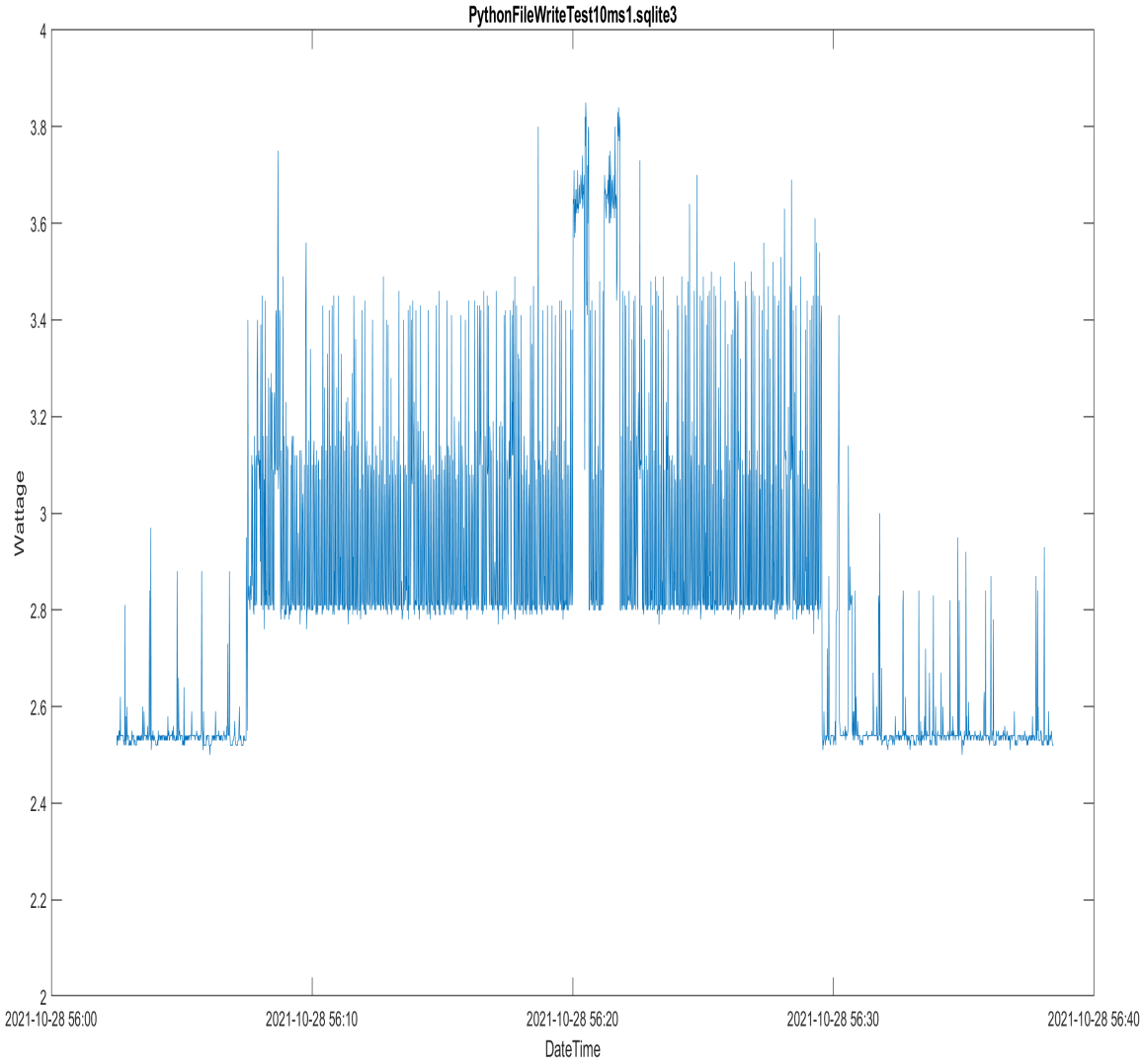
PythonFileWriteTest10ms1.sqlite3
Wattage
DateTime
4
3.8
3.6
3.4
3.2
3
2.8
2.6
2.4
2.2
2
2021-10-28 56:00
2021-10-28 56:10
2021-10-28 56:20
2021-10-28 56:30
2021-10-28 56:40

Figure 2: Shows a visual plot of the data from one of the trials, specifically a file write test using Python.

Data Findings & Interpretation

- Python Addition/Subtraction/Multiply/Divide print
- Python Addition/Subtraction/Multiply/Divide no print
- Python Write to file
- Objective-C Write to file
- Python LAN ping print
- Python LAN ping no print

Operation	Average Increase in Power (w)	Power Usage / Computing Event (W/s)	Power Usage / Computing Event (kwh)
Python LAN Ping & Print	0.25	0.00286	7.94445E-10
Python LAN Ping No Print	0.19	0.001967564	5.46546E-10
Python Addition & Print	0.904	3.47814E-05	9.66151E-12
Python Subtraction & Print	0.896	2.00675E-06	5.57432E-13
Python Multiplication by Constant 2 & Print	0.145	0.000472495	1.31249E-10
Python Division & Print	0.189	0.000765223	1.42579E-10
Python Addition No Print	0.447	1.15415E-06	3.20599E-13
Python Subtraction No Print	0.643	1.05705E-06	2.93624E-13
Python Multiplication by Constant 2 No Print	0.195	0.00048287	1.34131E-10
Python Division No Print	0.124	0.000278534	9.02944E-11
Python Write To File	0.42	0.1.1844E-06	3.29E-13
Objective-C Write to File	0.414	7.27639E-08	2.02122E-14

Future Project Goals

So far, the majority of what we have done is setting up the problem and basic testing to ensure our setup works as we'd expect. In the future, we plan to move onto the next step for the project, that is, the meat of the project. We will be setting up an email server and a script to send emails. Using these, we will be testing the different kinds of content that can be sent via email and how they affect power consumption.

After gathering data from the email servers, we will be looking into the Google Search Bar. Or, more specifically, the auto fill feature of the search bar. The auto fill feature of the Google Search Bar is a process that gives the user options to fill in what they've already typed. For instance, if the user typed "How to" into the search bar, Google might suggest that the query they're searching for is to "how to commit murder", or "how to hide a body", or "how to remove the evidence". Our goal in this instance is to determine how much power having that auto fill feature on at all times will take up over not having it on at all.

Conclusion