Building and Monitoring a Solar-Powered Web Server

Student Paper

Author(s):
Peter, Steven

Publication date:
2023-04

Permanent link:
https://doi.org/10.3929/ethz-b-000613264

ETH zürich

Networked Systems
ETH Zürich — seit 2015

Building and Monitoring a Solar-Powered Web Server

Semester Thesis

Author: Steven Peter

Tutor: Dr. Romain Jacob

Supervisor: Prof. Dr. Laurent Vanbever

March 2023 to April 2023

Abstract

In this thesis we focus on building a solar-powered web server. We present existing websites which are fully or partially solar powered, introduce some background about battery state of charge estimation and how to determine the right solar panel and battery size. Reusing components from older projects, we host a static website on an exemplary setup, which is solely solar powered.

Contents

List of Acronyms

EFF Electronic Frontier Foundation.

NSG Networked Systems Group.

SBC single-board computer.

SoC State of Charge.

Chapter 1

Introduction

1.1 Motivation

The motivation for this thesis is to make first steps into the field of solar-powered network infrastructure. The incentive for solar-powered network devices is to develop a solution for developing countries, where there is either no existing electricity grid or it is not reliable enough. Furthermore, the energy usage of current network infrastructure should be analyzed and reduced.

To gain experience in this field, being inspired by existing examples of solar-powered websites, we wanted the reproduce such a setup by building our own solar-powered web server. The goal of this thesis was to find the challenges in deploying solar-powered devices and enable the possibility to apply the learning from this project to other devices as well.

1.2 Task and Goals

The main objectives of this project were:

- Setting up a solar-powered web server
- Configuring the network to make a sample website externally accessible
- Setting up a battery voltage monitor

Chapter 2

Related Work and Background

This thesis was inspired by existing solar-powered websites. In Section 2.1, we present two such websites and their web server design and highlight the differences to our own setup. Monitoring the battery in such a system is an important part, being a complex but well-understood problem in literature. Section 2.2 summarizes the most important aspects for this project.

2.1 Related Work

2.1.1 Low-tech Magazine

The main inspiration for this thesis came from the Low-tech Magazine, an online magazine based in Barcelona (Spain), which hosts a copy of their website[1] on a solar-powered web server since 2018. Their focus was set on lowering the energy consumption, by using low-power hardware and minimalistic web design, which opposes the current trend of powerful web servers and ever-increasing website file sizes. Their web server is fully solar powered, meaning that their web server will be off-line during longer periods of bad weather [1].

An major decision was to use a static website, which requires a lot less power compared to a database-driven website. Another crucial choice was to reduce the number of images and graphics on their website, and use dithering to compress the remaining images to much smaller file sizes. An example for this can be seen on their About site (Figure 2.1).

Their current setup consists of an Olimex A20 computer to host the website, a 30 W solar panel and a 168 Wh lead-acid battery. Their web server draws only about half the power compared to our web server (1 W to 2.5 W compared to roughly 4 W, see Chapter 4) [2]. Their solar panel has the same rated output power as ours, and the battery they used has less than half the nominal capacity of the battery used in this thesis, which has 396 Wh.

2.1.2 Louwrentius

Another website that is (partially) solar powered is the blog from Louwrentius[2], based in the Netherlands. The author initially used a Raspberry Pi 3B+ as a web server, using a 756 Wh lead-acid battery and a 150 W solar panel to power it. The setup itself is not ideal, because it is in a west-facing apartment complex, which only receives direct sunlight during spring and summer, late in the afternoon. The author explicitly mentions that this setup is neither economically nor ecologically reasonable, due to the low direct radiation to the solar panel. It is a disproportionately large setup for the device and was meant solely to be a hobbyist project [3].

In the meantime, the author changed the setup to a Raspberry Pi 4B, two 370 W solar panels, for a total of 740 W peak output, and a 2760 Wh lead-acid battery. How big one of

[1] `https://solar.lowtechmagazine.com/`
[2] `https://louwrentius.com/`

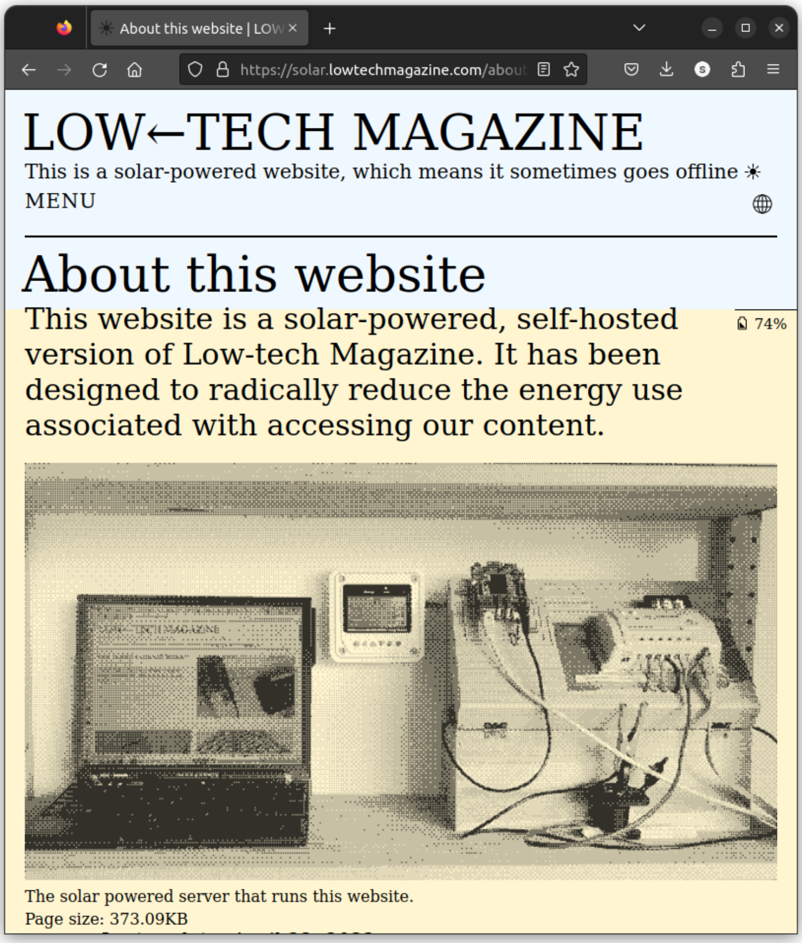


Figure 2.1: About this website - Low-Tech Magazine. The background color indicates the current SoC, the image used is dithered but still reasonably recognizable, overall resulting in a page size of only 373 kB.

those solar panels is, can be seen in Figure 2.2. This setup does not only power the web server, but the author also uses its excess power to charge other electronic devices such as an iPad. In contrast to the website of the Low-tech Magazine and our own setup, this blog will not turn off when the battery does not have enough charge. Instead, it switches to grid electricity [4]. The Raspberry Pi 4B has a similar power draw to our web server with around 3.5 W, while the solar panel and battery are both substantially bigger than what we used in this thesis.

2.2 Background

2.2.1 State of Charge

The State of Charge (SoC) of a battery is the remaining energy of a battery relative to its capacity when fully charged. To prevent degradation, lead-acid batteries should not be fully discharged. The SoC is usually scaled accordingly, to only represent the usable capacity. Meaning that 0% is the state at which the battery management system will disconnect the load from the battery.

Determining the SoC for lead-acid batteries can be done with different methods such as coulomb counting, which means to effectively integrate the current flowing in and out of the battery. But this method is costly and not practical. Another method is to determine the

Figure 2.2: Solar panel used by Louwrentius with a rated power of 370 W [4].

specific gravity of the battery, which is the mass of the electrolyte divided by its volume. To measure the specific battery, a Hydrometer is used. Accurate Hydrometers are also quite costly. The method we used determines the SoC by measuring the voltage of the battery. This can be achieved with cheaper hardware, allowing for an easier method to determine the SoC. However, the voltage of the battery is influenced not only by the remaining energy, but also various other parameters such as discharge/charge current, temperature, age, etc. A method to calculate the SoC using the voltage and discharge/charge current is presented in [5]. The SoC is not linearly correlating with the voltage as we can see in Figure 2.3, but those curves during idle and discharge can be approximated by a second order polynomial. However, when charging the battery, the voltage cannot be used as an indication of the current SoC.

There are three phases when charging the battery (see Figure 2.4). During the bulk phase, the current is constant, and the voltage will increase over time. In the absorption phase the voltage is constant and the current is exponentially decrease until the battery is considered fully charged. During the float phase it does only receive a low current to conserve the state.

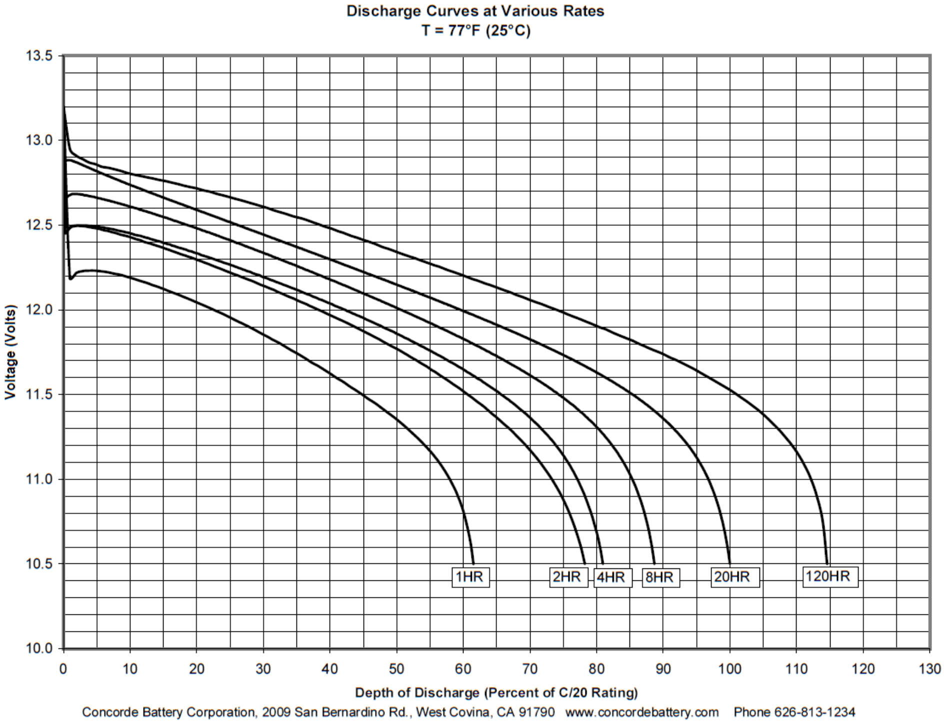


Figure 2.3: Discharge curves at various rates for the lead-acid battery used in this project. Nominal curves from the datasheet [6].

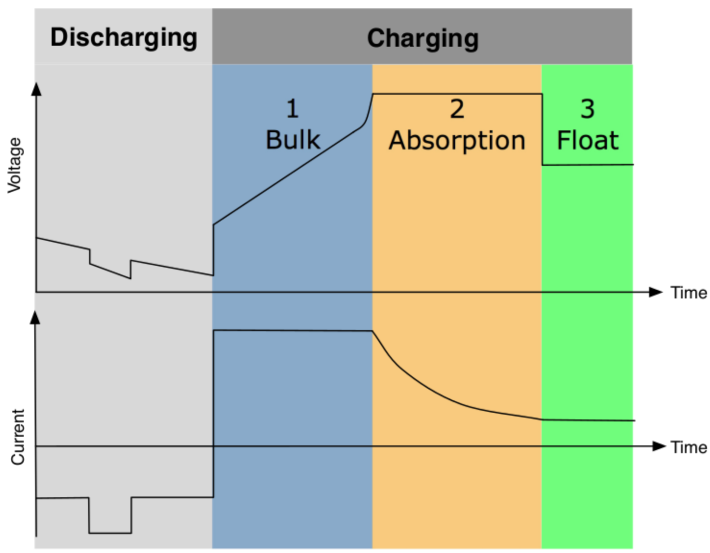
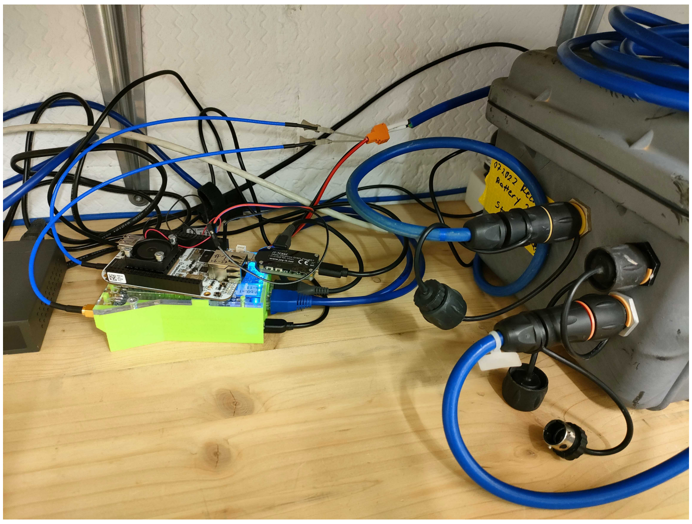


Figure 2.4: Relation between battery terminal voltage (top) and current (bottom) during the charging and discharging processes, showing the three phases of charging [5].

Chapter 3

Implementation

The implementation of a functioning solar-powered web server was the main goal for this thesis. Figure 3.1 shows the full setup with all components used to achieve this. We list the components used in Section 3.1, followed by a description of the main component, the web server, in Section 3.2. After this, we elaborate on considerations for the solar panel and battery size in Section 3.3, present the devices used for data logging in Section 3.4 and lastly talk about the implementation of the SoC estimation in Section 3.5.



(a) Inside: Battery with charge controller in the gray case on the right, web server and data logging devices on the left.

(b) Outside: Solar panel.

Figure 3.1: Full setup used for the solar-powered web server.

3.1 Overview

The main components used to build the solar-powered web server are (see Figure 3.2):

- Web Server: BeagleBone AI
- Solar Panel: Sinosol cleversolar SPR-30 (17.5 V, 30 W)
- Battery: Lifeline GPL-U1T (12 V, 33 Ah)
- Solar Charge Controller: Morningstar SS-6L-12V

We reused those components from past projects to evaluate the feasibility of a solar-powered web server without needing to buy new hardware. This also means that, for example, the battery has already aged and will not provide the nominal capacity that it once had.

The used components are not in production anymore, but the total cost for a similar setup is about CHF 300.- (Web Server: CHF 90.-, Solar Panel: CHF 60.-, Battery: CHF 75.-, Solar Controller: CHF 75.-).

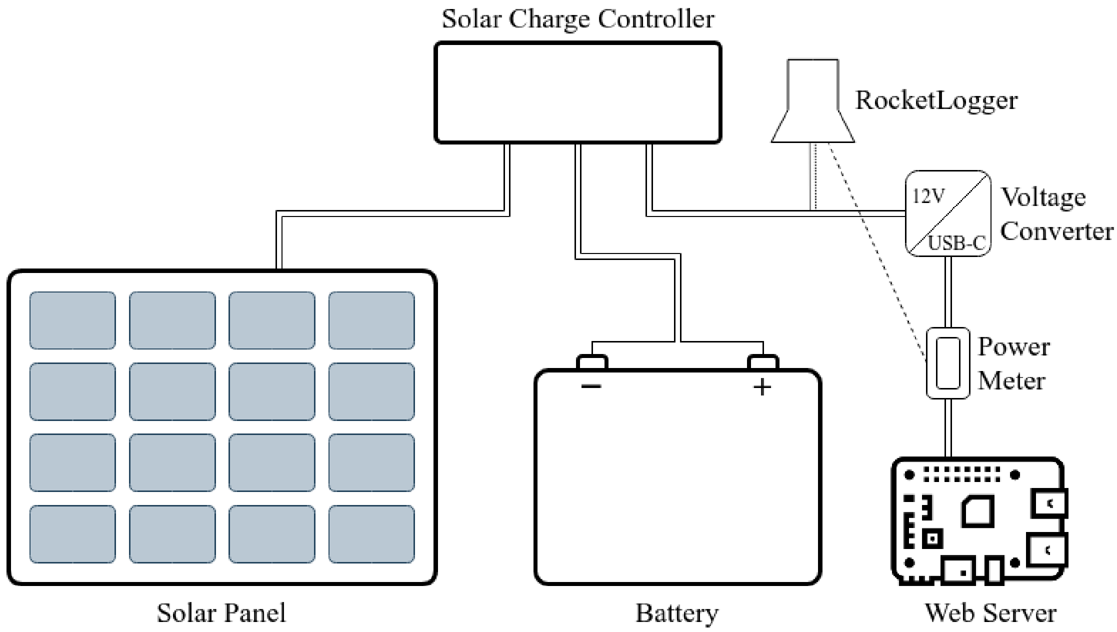


Figure 3.2: Full setup showing the solar panel, the battery, the web server, and the solar charge controller. Also included are the voltage converter, the RocketLogger and the power meter.

The battery voltage varies between 11.5 V and 14 V depending on the SoC. We use the voltage converter *Szwengao WG-1224S0503TC* to convert this variable voltage to 5 V, which is required by the web server. To measure the voltage from the battery during the experiments, we use the *RocketLogger* and to measure the power drawn by the web server, we use the *Joy-IT TC66C*. More on those data logging devices in Section 3.4.

3.2 Web Server Setup - BeagleBone AI

The web server plays a key role in this project. The BeagleBone AI is a single-board computer (SBC) similar to the Raspberry Pi, but with additional machine learning capabilities (see Figure 3.3). It is based on the Texas Instruments AM5729, which is an Arm processor.

For this project we use a version of Debian Buster as operating system, which is specifically adapted to run on this processor. However, since the BeagleBone AI is not in production anymore, new firmware images are not tested on the board anymore and only part of the available images do in fact work. Eventually, we used a minimal console image from April 2020 on the BeagleBone AI, as it is able to run and needs only little configuration to run a web server.

We use NGINX as a web server to host a static website on the BeagleBone AI. The hosted website is a newly developed version of the Networked Systems Group (NSG) website, which transitioned from a dynamic website to a static one. This transition implies that the website can now be hosted on less powerful hardware such as an SBC, since the website is stored as static files and the web server itself does not need to recompile the website for every visitor.

In order to enable HTTPS for the website, the *Certbot*[1] from Electronic Frontier Foundation (EFF) is used to receive a certificate from *Let's Encrypt*[2].

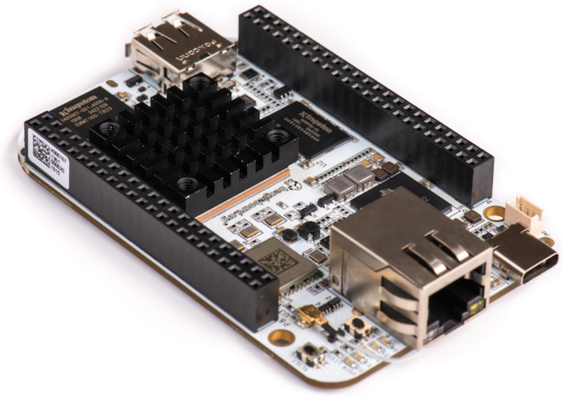


Figure 3.3: BeagleBone AI [7]

3.3 Solar panel and battery size

The web server used in this thesis has a power usage of around 4 W (see Chapter 4). From this we can derive the requirements for the battery capacity and the size of the solar panel.

The battery should only be discharged to a certain percentage to prevent additional stress and damage of the battery. This is around 60% of the nominal capacity [5]. Depending on the used solar controller, the cut-off voltage is already set such that deep discharge is prevented. For an exemplary uptime of 3 days of the web server, without any solar power, we need a battery with a nominal capacity of $72\,\text{h} \cdot 4\,\text{W}/60\,\% = 480\,\text{Wh}$. This is slightly more than the used battery, with a capacity of 396 Wh, though we can expect an uptime of 60 h with our setup. The Low-tech Magazine presents similar battery capacity calculations on their website [8].

The optimal sizing of the solar panel is more complicated because there are numerous factors that influence this, like location, orientation, and inclination. The panel should be able to power at least the web server and the excess power is used to charge the battery for later use when there is less or no solar radiation. The solar panel should be reasonably big to deliver enough power even during winter, when there is less sunlight to harvest. Using data from MeteoSwiss (see Figure 3.4), it is possible to calculate the average global radiation for the location of the solar panel. The weather station used at Zürich Fluntern is about 1 km away from deployment location of our solar setup.

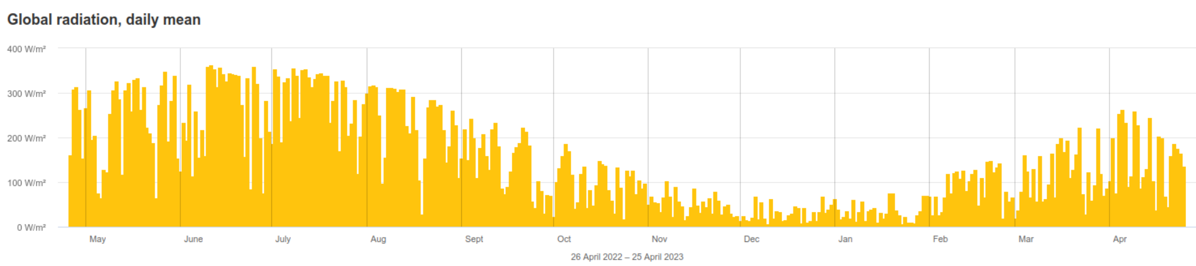


Figure 3.4: Daily mean global radiation for the nearest weather station at Zürich Fluntern. Showing that the average daily radiation is much lower during winter than during summer [9].

[1] `https://certbot.eff.org/`
[2] `https://letsencrypt.org/`

The average global radiation in December and January 2022 was 31 W/m^2. The solar panel ratings are normally given for a radiation of 1000 W/m^2. Meaning that our 30 W rated panel produces 30 W only for this optimal radiation, which is only rarely achieved in Switzerland. The 30 W panel thus produces an average of $30\,\mathrm{W} \cdot \frac{31\,\mathrm{W/m^2}}{1000\,\mathrm{W/m^2}} = 0.93\,\mathrm{W}$ during this time period. To power the web server even during winter, we would need a larger solar panel. For a load of 4 W the solar panel should have a peak power of 130 W. This solar panel will however produce a lot of excess power during the sunnier months of the year. Since this additional energy should not be just wasted, we could use it to power other devices similarly to Louwrentius.

3.4 Data logging

3.4.1 RocketLogger

The RocketLogger[3] is a portable data logger build on top of a BeagleBone that allows for high accuracy voltage and current measurements (see Figure 3.5). It features an intuitive graphical interface and a CLI, which we used to automate the measurement. We power the RocketLogger by the grid and not by the solar panel, to separate it from power usage of the web server, as this is what we are really interested in.

We use the RocketLogger to measure the voltage of the battery. Since the range of voltage possible to measure with the RocketLogger is ±5.5 V, we use a simple voltage divider with a factor of 3 to widen the range to ±16.5 V. We need this wider range, as the nominal voltage of the used battery is 12 V and can rise over 14 V when it is in the absorption state of charging. The battery and charge controller are both in a sealable box, consequently we decided to not mount the RocketLogger to the connection between battery and charge controller, but on the output of the charge controller, which connects to the load. Meaning that we can only measure the voltage as long as the load is not disconnected by the charge controller.

We use Cron as a job-scheduler, to trigger the measurement every 5 minutes. The Cron job will start the measurement using the RocketLogger CLI, to sample the voltage and save it to a separate file in the RLD format, a binary data format. We can then analyze this file using the RocketLogger Python library.

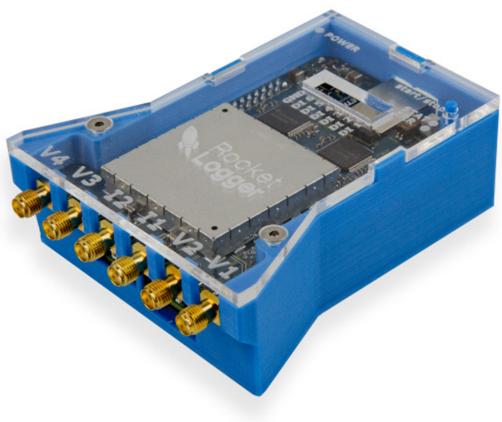


Figure 3.5: RocketLogger [10]

[3] https://rocketlogger.ethz.ch/

3.4.2 Joy-IT TC66C

The Joy-IT TC66C is a rebranded USB-C power meter, which is produced by the Chinese Company *Ruideng* (see Figure 3.6). Officially there is only an application to interface the device with Windows or Android. It allows for quick measurements to get an impression of the power usage of any USB-C device, however the accuracy of the device is not evaluated in this thesis. The TC66C can be either powered directly with the plugged device or with an additional Micro-USB connection, which is used in this thesis, to access the data readings and electrically separate the power meter from the energy consumption from the web server.

An unofficial Python library is available[4], that allows for any device to interface the power meter. The measurements can then be started by running the Python script, enabling automated measurements. All data samples are then stored in a CSV file. For the following experiments, the TC66C is mounted in line with the USB-C which powers the web server, and connected additionally via Micro-USB to the RocketLogger. On the RocketLogger another Cron job is used to start the measurements and save the data.

The full setup including the voltage converter and also the data logging devices can be seen in Figure 3.2.

Figure 3.6: Joy-IT TC66C [11]

3.5 SoC Estimation

To track the voltage measurements of the RocketLogger, the data needs to be converted. This is done by the BeagleBone AI using the RocketLogger Python library. The data is fetched and then converted from RLD to a JSON file. The data of the measurement is averaged over the measurement period, and the factor of 3 of the voltage divider is used to calculate the current voltage of the battery.

The voltage information can then be used to estimate the SoC. The state machine proposed in [5] (see Figure 3.7) is used to track the current state of the battery (Discharge, Bulk, Absorption, Float).

For the different states, different SoC estimations could be applied. To simplify the calculation, it is assumed that the load draws only negligible power from the battery. Figure 2.3 is used to fit a second order polynomial to the curve, to convert the voltage to the SoC. This could be further improved, by accurately tracking the current flowing out of and into the battery. Especially during Bulk and Absorption charging, where the voltage of the battery will not represent the actual SoC.

The voltage and SoC history is then plotted and displayed on a dedicated website also hosted on the BeagleBone AI (see Figure 3.8).

[4] `https://github.com/TheHWcave/TC66`

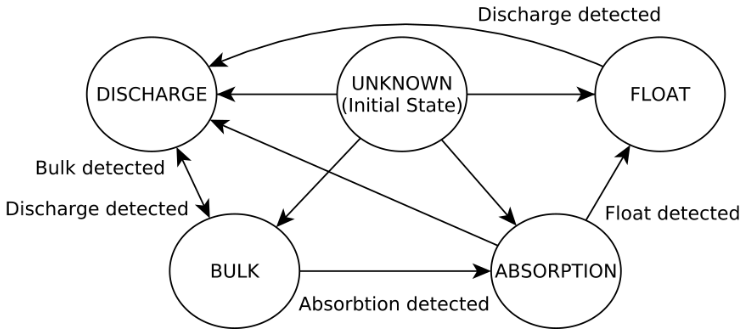


Figure 3.7: State Machine for charge/discharge tracking [5]

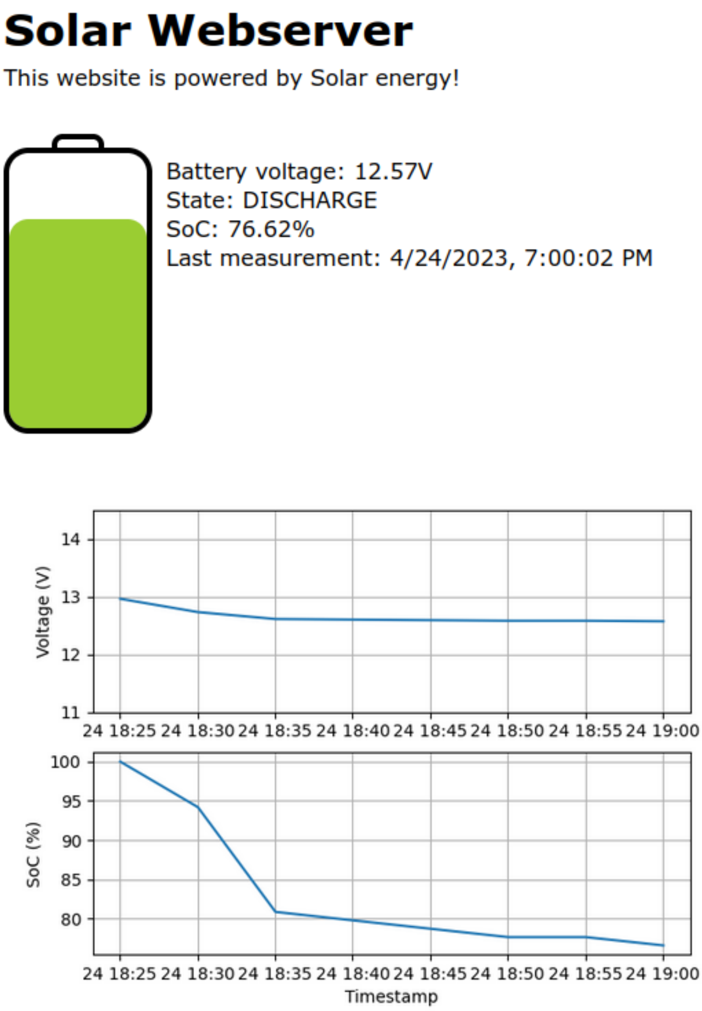


Figure 3.8: Website to display current SoC and plot of the historical data

Chapter 4

Evaluation

To evaluate the overall performance of the solar-powered web server multiple experiments were conducted. Firstly, the energy consumption to serve different file sizes was measured. Secondly, the whole setup was tested live for a week, run solely by solar power. Lastly, the actual battery capacity was analyzed without recharging it, simulating bad weather conditions.

4.1 Energy consumption for different website file sizes

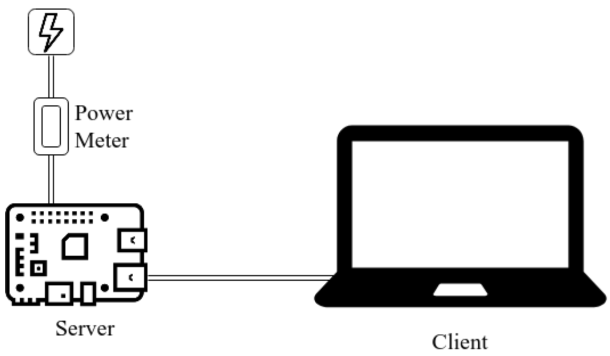


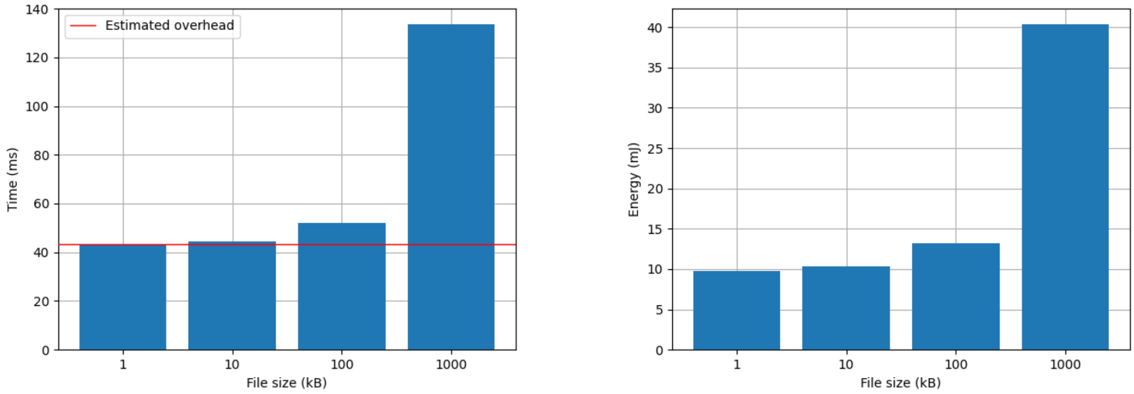
Figure 4.1: Setup for the evaluation of the energy consumption for different file sizes

Figure 4.1 shows the setup used for this evaluation. The BeagleBone AI is powered by the grid, and the TC66C is used to measure the power draw of the BeagleBone AI. The client is directly connected to the web server via Ethernet, to eliminate possible fluctuation in transmission delay in an intermediate network. The web server hosts various sample files ranging in size from 1 kB to 100 MB.

The initial test revealed the average time for a single request from the client to the server to be answered, the results are listed in Table 4.1 and shown for the four smallest files in Figure 4.2a. From this we can derive that every request has an overhead of about 43.4 ms regardless of the file size and the data transfer lasts approximately 0.09 ms/kB.

Resulting from this, we chose the number of requests sent by the client for the energy evaluation. The number of requests was chosen such that each test is completed in under a minute.

The power draw in the idle state measured over a period of 1 min averages to 3.72 W. This average is used as a baseline to compare the additional energy usage serving the different files. The results can be seen in Figure 4.2b and Table 4.2. Again, for a three smallest files, the results are almost identical. Afterwards, the additional energy consumed per request scales linearly, if we subtract a baseline of 10 mJ per request.



(a) Average time per request   (b) Average energy per request

Figure 4.2: Time and energy for serving the four smaller files. The results clearly show that when subtracting the estimated overhead from the time and energy measurements, the remaining value scales linearly with the file size.

File size	Time per request
1 kB	43.5 ms
10 kB	44.5 ms
100 kB	52.2 ms
1 MB	133.5 ms
10 MB	926.2 ms
100 MB	9.085 s

Table 4.1: Average time per request to serve different file sizes

File size	Requests	Additional energy	Energy per request
1 kB	1000	9.8 J	9.8 mJ
10 kB	1000	10.4 J	10.4 mJ
100 kB	1000	13.2 J	13.2 mJ
1 MB	300	12.1 J	40.3 mJ
10 MB	60	17.9 J	298.8 mJ
100 MB	6	16.4 J	2.730 J

Table 4.2: Average energy consumed to serve different file sizes

To do a comparison of how long the battery lasts without recharging, with the web server being idle versus how long it lasts, when serving a certain amount of requests, the landing page of the new NSG can be taken as an example, which weighs around 5 MB. For this we use the following equation:

$$E_{tot} = N * E_{req} + t * P_{idle} \tag{4.1}$$

where E_{total}: total energy consumed, N: number of requests, E_{req}: energy per request, t: time until battery is discharged (load is disconnected) and P_{idle}: idle power.

The battery we used in this project has a nominal capacity of 396 Wh. Meaning that the effective capacity is around $396\,\text{Wh} \cdot 60\,\% = 238\,\text{Wh}$, before the solar charge controller disconnects the web server from the battery. For the average idle power draw of 3.72 W this means that the web server should last approximately 64 h. Derived from the energy measurements, we can assume that serving a 5 MB website will consume roughly 160 mJ of additional energy. Serving 1'000'000 requests for the landing page will thus require 160 kJ or 44 Wh of energy. This reduces the uptime of the web server to 52 h.

4.2 Live deployment for a week

For the next experiment we deployed the full setup and run the web server with solar power for a week. Our main goal here was to test, if the chosen setup works at all and what its limitations are. The setup can be seen in Figure 3.1. The web server was mainly running idle with occasional manual fetching of the battery status website (Figure 3.8) to see if the web server is still active.

The results of the experiment can be seen in Figure 4.3. The top graph shows the measured voltage of the battery. The dashed lines show important voltage levels, where $V_{absorption}$ is the voltage the solar charge controller uses during the absorption phase of charging. V_{float} is the voltage at which the battery will be held in the last phase of charging. V_{max} is the voltage, which the battery will show when it is fully charged, and not being actively charged/discharged, according to the datasheet. If the voltage rises over this threshold the SoC is considered to be 100%. And V_{cutoff} is the voltage at which the web server will be disconnected from the battery. Thus whenever the voltage drops below this voltage level, the web server will be offline, translating to a SoC of 0%.

The SoC should not be considered to be accurate when the battery is charging, as explained in Section 3.5. However from a qualitative view, the SoC is still quite interesting to analyze. The web server lasted for almost two nights and a day, before shutting off. Though in the meantime, the battery was slightly charged during the day. On the 17th and 18th of April, the weather was very sunny, as can be seen by looking at the global radiation on those days. The battery appears to be fully charged after each of those days, but still the battery capacity did not suffice to power the web server during the whole night. A likely reason for this is that the battery is too old and has already degraded too much to be used in this or any other application. During the days, where there was less global radiation such as the 16th, 19th and 20th of April, the web server was powered only infrequently during the day, and almost instantly turned off at sunset.

Another noticeable trend is the inconsistency of the solar charge controller. Once the battery falls below the cutoff voltage, the load should be disconnected until the battery has recovered enough, to again power the load. However, as can be seen in Figure 4.3 the load is frequently disconnected and reconnected, resulting in the web server regularly rebooting. This again could indicate that the battery is not suitable anymore, and its voltage will almost instantly drop a lot whenever the load is reconnected, enough to trigger the solar charge controller. Another possibility is that the difference between cutoff voltage and the voltage to reconnect the load is set too small or some other technical issue with the solar charge controller.

Figure 4.3: Results for the weekly experiment showing voltage, estimated SoC and the hourly average global radiation for the nearby weather station in Zürich Fluntern [12]

4.3 Battery discharge test

The last experiment was conducted, to evaluate the actual capacity the battery is still capable to deliver. For this, the battery was plugged into a battery charger powered by grid electricity, first being fully charged and then left in the floating state over the course of multiple hours. The battery was then connected to the setup as before, but without connecting the solar panel. This allowed to study the energy capacity of the battery alone, instead of combining it with the energy gained from the solar panel.

The web server ran from 18:30 until 02:21 on the next day, resulting in an effective uptime of 7 h 51 min. Figure 4.4 shows the output voltage of the solar charge controller. Similar to before the SoC does not correlate linearly with time, even though the power draw of the web server stays approximately constant. The estimation of the SoC could thus be improved, especially for this use case, as we know the expected power draw of the BeagleBone AI when idle.

Figure 4.5 shows the voltage and current measured with the Joy-IT TC66C, from which the power is calculated. The results show, that the power during startup peak at 6 W and settle down to an average of 4 W while idle. Integrated over time, we get the total consumed energy, which is 113.5 kJ or 31.53 Wh during the period from 18:30 and 02:21. After this period, the web server is powered only for a few minutes before being disconnected again. Compared to the nominal capacity of 396 Wh this is a difference of factor of 10. Since the solar charge controller does not allow to deep discharge the battery, the effective capacity of a new battery should be considered reduced by a factor of 2. Either, the battery was not fully charged, which needs to be tested by redoing the experiment, or the battery is truly heavily degraded. Another component that needs further investigation is the solar charge controller, because the load should not be disconnected and reconnected as frequently as observed.

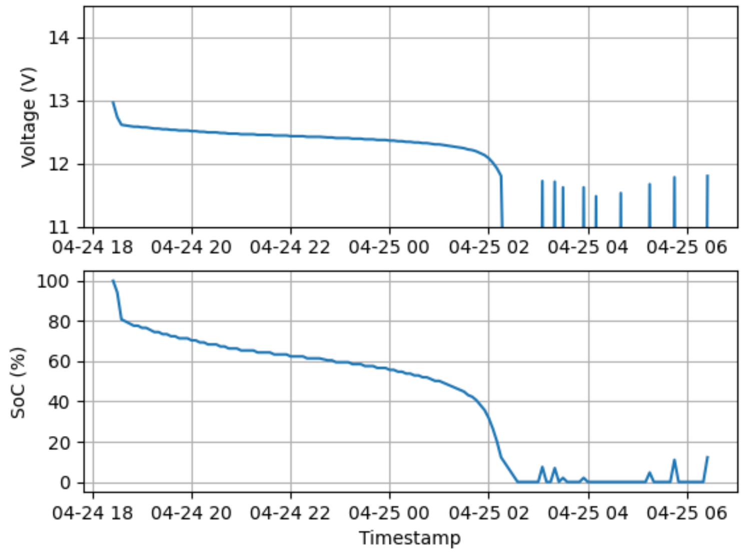


Figure 4.4: Output voltage of the solar charge controller during the battery discharge test. An initial drop shows that the voltage is highly reactive to current draw of the web server. For the following period, the battery voltage decreases with an almost linear curve, before rapidly dropping below the cutoff voltage.

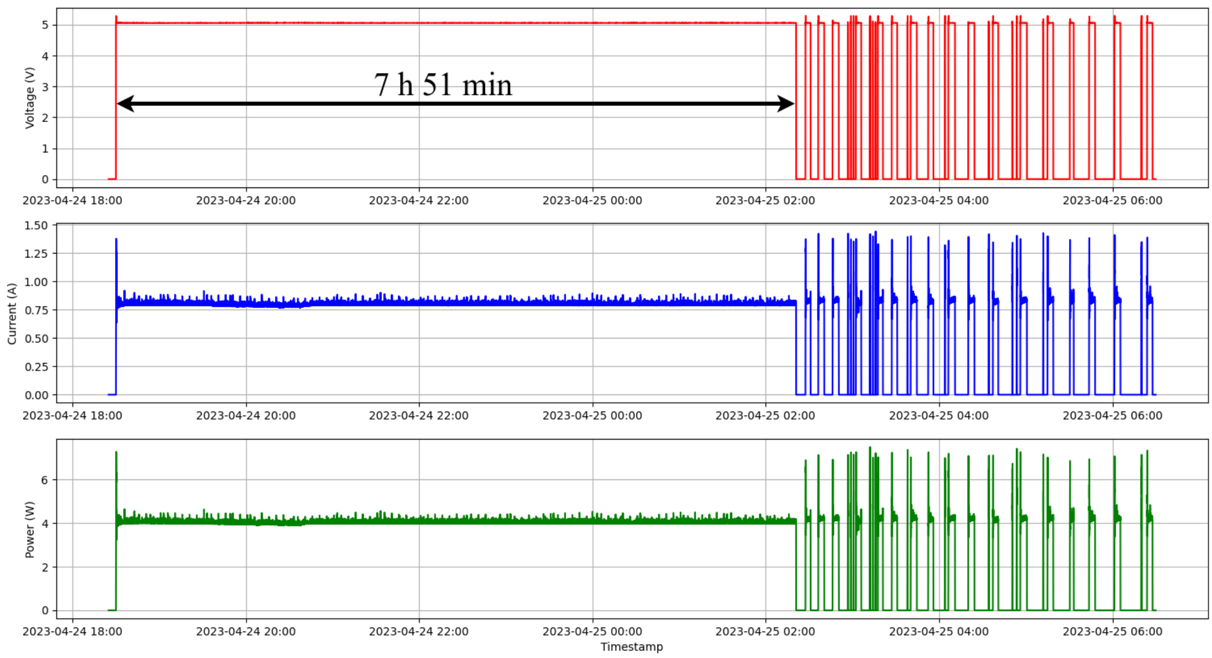


Figure 4.5: Voltage, Current and Power of the web server during the battery discharge test. During the initial period from 18:30 to 02:21 the power stays approximately constant, while during the following period, we can see that the web server is constantly being power cycled.

Chapter 5

Summary and Outlook

This thesis shows a possible setup for a solar-powered web server, combining reused components to a minimal working setup. It analyzes the hurdles of being fully solar powered, especially considering the difficult decision of battery capacity and solar panel size. This thesis evaluates the energy consumption of serving different file sizes, and measures the utilizable capacity of an old battery. Finally, it presents methods for logging the voltage of the battery, estimating the SoC and tracking the power draw of the web server.

For further projects the following points could be addressed:

- The energy consumption of the web server could be reduced, by inspecting the energy usage of different components and turning off unused components (e.g. the Wi-Fi antenna, which is not used but still turned on by default on the BeagleBone AI). Wake-on-LAN could be implemented to only turn on the web server when there is a request to serve, the Ethernet module on the BeagleBone AI theoretically supports this feature, however it is unclear if the firmware of the BeagleBone AI also supports it.
- The selection of the web server hardware could be improved, by comparing the energy consumption of different devices and finding a good trade-off between efficiency and performance.
- Then the battery and solar panel should be chosen to fit the design requirements.
- While it is reasonable to reuse older hardware, it should be evaluated before being deployed. Especially the capacity of the battery should be determined beforehand, to identify if it is still able to fulfil the expectations.
- Data logging could be improved, by using a scalable solution like Grafana or similar tools.
- The battery SoC estimation can be improved by measuring the charge/discharge current of the battery.
- Further network infrastructure devices such as a router could be also be converted to be solar-powered.

Bibliography

[1] lowtechmagazine.com. *How to Build a Low-tech Website?* [Last accessed 25.04.2023]. 2018. URL: https://solar.lowtechmagazine.com/2018/09/how-to-build-a-lowtech-website.html.

[2] lowtechmagazine.com. *About - Which hardware and software do you use?* [Last accessed 25.04.2023]. 2022. URL: https://solar.lowtechmagazine.com/about.html%5C#hardware.

[3] louwrentius.com. *This Blog Is Now Running on Solar Power.* [Last accessed 26.04.2023]. 2020. URL: https://louwrentius.com/this-blog-is-now-running-on-solar-power.html.

[4] louwrentius.com. *I Made My Blog Solar-Powered, Then Things Escalated.* [Last accessed 26.04.2023]. 2023. URL: https://louwrentius.com/i-made-my-blog-solar-powered-then-things-escalated.html.

[5] Bernhard Buchli, Daniel Aschwanden, and Jan Beutel. "Battery State-of-Charge Approximation for Energy Harvesting Embedded Systems". In: vol. 7772. Feb. 2013, pp. 179–196. ISBN: 978-3-642-36671-0. DOI: 10.1007/978-3-642-36672-7_12.

[6] Concorde Battery Corporation. *Technical Manual For Lifeline® Batteries.* 2019.

[7] BeagleBoard.org. *BeagleBoard.org® Launches BeagleBone® AI, Offering a Fast Track to Getting Started with Artificial Intelligence at the Edge.* [Last accessed 26.04.2023]. 2019. URL: https://beagleboard.org/blog/2019-09-19-beaglebone-ai-available.

[8] lowtechmagazine.com. *How Sustainable is a Solar Powered Website?* [Last accessed 25.04.2023]. 2020. URL: https://solar.lowtechmagazine.com/2020/01/how-sustainable-is-a-solar-powered-website.html.

[9] meteoswiss.admin.ch. *Global radiation - Zürich / Fluntern.* [Last accessed 26.04.2023]. 2023. URL: https://www.meteoswiss.admin.ch/services-and-publications/applications/measurement-values-and-measuring-networks.html%5C#param=messwerte-globalstrahlung-10min&lang=en&station=SMA&chart=day.

[10] sennengineering.ch. *The RocketLogger.* [Last accessed 26.04.2023]. URL: https://www.sennengineering.ch/rocketlogger/index.html.

[11] Joy-IT.net. *USB Volt-/Amperemeter.* [Last accessed 26.04.2023]. URL: https://joy-it.net/en/products/JT-TC66C.

[12] meteoswiss.admin.ch. *Global radiation - Zürich / Fluntern.* [Last accessed 22.04.2023]. 2023. URL: https://www.meteoswiss.admin.ch/services-and-publications/applications/measurement-values-and-measuring-networks.html%5C#param=messwerte-globalstrahlung-10min&lang=en&station=SMA&chart=hour.