Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Energy and Process Engineering

Martin August Egerdahl

PV-to-Heat: Control and Monitoring Methods

Master's thesis in Energy and Environmental Engineering
Supervisor: Ole Jørgen Nydal
January 2024

NTNU
Norwegian University of
Science and Technology

Martin August Egerdahl

PV-to-Heat: Control and Monitoring Methods

Master's thesis in Energy and Environmental Engineering
Supervisor: Ole Jørgen Nydal
January 2024

Norwegian University of Science and Technology
Faculty of Engineering
Department of Energy and Process Engineering

NTNU
Norwegian University of
Science and Technology

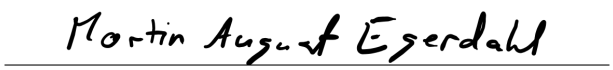
Preface

This master thesis is written as part of the collaboration between NTNU and several universities from countries in Africa, through NORPART. The thesis is written as the final work of Energy and Environmental studies at NTNU during the autumn of 2023. Also, it is a continuation on the project work concluded in the spring of 2023.

Firstly, I would like to thank my supervisor Ole Jørgen Nydal for guidance on this master thesis and my project work. His guidance to my questions and the challenges faced throughout the master thesis work were pivotal for my success. Furthermore, the support and encouragement given to strive further has been a great learning experience, which will be invaluable to have. Also, a thanks to the co-supervisor from my project work Mulu Bayray Kahsay, for being available for questions regarding the Arduino and his interest in my work. I must in addition, give a big thank you to Marius Østnor Døllner, Stein Kristian Skånøy and Paul Svendsen for their help in the experimental work at the EPT laboratory of NTNU Gløshaugen.

During the project work, a stay in Tanzania was an opportunity given by NTNU for me and four other students. The journey has been an inspiration throughout both the project work and this master thesis. The professors and students at the University of Dar es Salaam and Arusha Technical College that were welcoming us in their country with their great hospitality will be very memorable and I am thankful for the experience.

Trondheim, 17th of January, 2023



Martin August Egerdahl

Abstract

Many African countries has an increasing population and economy. However, energy production is regularly not sufficient, leading to blackouts in the cities and rural areas do not have access to electricity. Less than 10 % of the population in Tanzania has access to clean cooking, and fuels such as charcoal and kerosene are used instead, which leads to household air pollution. Developing off-grid power systems to increase the populations access to electricity is an important step in Africa's future. To work towards that prosperous future, this thesis aims to research how three solar charge controllers are affected by different resistances of heating elements to which they are connected.

The three controllers of the experimental work are an MPPT buck converter unit created by the former NTNU student Odin Hoff Gardå, and two models from the South African manufacturer Geyserwise. The two MPPT controllers from Geyserwise differ in target load voltage with the first being 72 V and the latter a 48 V controller. For the Odin and 72V controller, a four in series PV configuration of Solartek PVP26030 were used, while the 48V utilized a $2\,x\,2$ configuration. A junction box was used between the controllers and a pair of heating elements to ease the changing of resistances. Each heating rod of the Backer IU313 heating elements had a specific resistance, thus they could be series or parallel circuited to obtain different resistances.

The Geyserwise 72V had a peak power output of 856 W, at a solar irradiance of 446 W/m^2 with the 3.28 Ω resistance. Although 4.37 Ω was promising, it stagnates at higher solar irradiances, which the higher resistances did as well. Similar performance was displayed by the 48V controller, with a peak power output of 819 W, at an irradiance of 312 W/m^2 and with a 2.62 Ω resistance. However, the controllers seem to be inefficient at lower solar irradiances. The Odin controller had a peak power of 837 W, at an irradiance of 391 W/m^2 and with the 13.1 Ω resistance. Although, further testing with multimeters on the output of the controller showed a large disparity in power levels. Therefore, it is an uncertainty that the controller is displaying correct values when using a four in series PV configuration, compared to the six in series it is capable of. An aspect to note on the Odin controller was that lower resistances had higher peak temperatures and faster rates of change, which is due the controller not being able to deliver the energy and it being transferred to heat instead.

In addition to performing tests of the three controllers, the aim of the work is to further develop an Arduino based data logger for web-based access. The Arduino is a stack of an Arduino Uno R3, data logging shield, solar charger shield and an LCD shield. It is designed to receive temperature data from four thermocouples, but a voltage divider circuit and shunt has been added for voltage and current measurements. The desire is to obtain this information remotely, without the need of accessing the SD card on the unit. To do so, a TinySine SIM7600 shield has been added to the system in hope that it can connect to the cellular network and transmit data.

The transmission of this data was to be done through the MQTT protocol. The protocol uses an MQTT broker to connect clients in a publish-subscribe model, where the Arduino is a client publishing data, and the user is a client subscribing to the data through a computer or mobile application such as MQTT Explorer. Unfortunately, the Arduino IDE program has been unsuccessful in initializing the shield and connect it to the network. The first attempt failed, and later research showed that the use of AT commands was more likely to succeed. However, this too has been unable to achieve the desire of a web-based Arduino and would need further development to succeed.

Sammendrag

Mange afrikanske land har en økende befolkning og økonomi. Imidlertid er energiproduksjonen regelmessig ikke tilstrekkelig, noe som fører til strømbrudd i byene og distriktene har ikke tilgang til strøm. Mindre enn 10 % av befolkningen i Tanzania har tilgang til ren matlaging, og drivstoff som trekull og parafin brukes i stedet, noe som fører til husholdningenes luftforurensning. Å utvikle off-grid systemer for å øke befolkningens tilgang til elektrisitet er et viktig skritt i Afrikas fremtid. For å jobbe mot den velstående fremtiden, har denne oppgaven som mål å undersøke hvordan tre kontrollere for solcellepaneler påvirkes av forskjellige motstander til varmeelementer som de er koblet til.

De tre kontrollerne for det eksperimentelle arbeidet er en MPPT buck-omformer laget av den tidligere NTNU-studenten Odin Hoff Gardå og to modeller fra den sørafrikanske produsenten Geyserwise. De to MPPT-kontrollerne fra Geyserwise er forskjellige i lastspenning, hvor den første er 72 V og den siste en 48 V-kontroller. For Odin- og 72V-kontrolleren ble en fire i serie PV-konfigurasjon av Solartek PVP26030 brukt, mens 48V benyttet en $2\,x\,2$-konfigurasjon. En koblingsboks ble brukt mellom kontrollerne og et par varmeelementer for å lett kunne endre motstanden. Hver varmestav i Backer IU313 varmeelementene hadde en spesifikk motstand, og derfor kunne de serie- eller parallellkobles for å oppnå forskjellige motstander.

Geyserwise 72V hadde en toppeffekt på 856 W, ved solinnstrålingen 446 W/m^2 og med motstanden 3.28 Ω. Selv om 4.37 Ω var lovende, stagnerer den ved høyere solinnstråling, noe de større motstandene også gjorde. Lignende ytelse ble vist av 48V-kontrolleren, med en toppeffekt på 819 W, ved solinnstrålingen 312 W/m^2 og med en motstand på 2.62 Ω. Kontrollerne ser imidlertid ut til å være ineffektive ved lavere solinnstråling. Odin-kontrolleren hadde en toppeffekt på 837 W, ved solinnstrålingen 391 W/m^2 og med en motstand lik 13.1 Ω. Derimot viste ytterligere testing med multimetre på utgangen til kontrolleren en stor forskjell i ytelsesverdier. Derfor er det en usikkerhet at kontrolleren viser riktige verdier ved bruk av en fire i serie PV-konfigurasjon, sammenlignet med de seks i serie den er i stand til. Et aspekt å merke seg på Odin-kontrolleren var at lavere motstand hadde høyere topptemperaturer og raskere endringshastigheter, noe som skyldes at kontrolleren ikke var i stand til å levere energien og at den ble overført til varme i stedet.

I tillegg til å utføre tester av de tre kontrollerne, er målet med arbeidet å videreutvikle en Arduino-basert datalogger for nettbasert tilgang. Arduinoen er en stabel bestående av en Arduino Uno R3, dataloggingsskjold, solcelleladerskjold og et LCD-skjold. Den er designet for å motta temperaturdata fra fire termoelementer, men en spenningsdeler og shunt er lagt til for spennings- og strømmålinger. Ønsket er å få tak i denne informasjonen eksternt, uten behov for tilgang til SD-kortet på enheten. For å gjøre det har et TinySine SIM7600 skjold blitt lagt til systemet i håp om at det kan koble seg til mobilnettverket og overføre data.

Overføringen av disse dataene skulle gjøres med MQTT protokollen. Protokollen bruker en MQTT megler for å koble klienter i en publiserings-abonner modell, der Arduinoen er en klient som publiserer data, og brukeren er en klient som abonnerer på dataene gjennom en datamaskin eller mobilapplikasjon slik som MQTT Explorer. Dessverre har Arduino IDE programmet ikke lyktes med å initialisere skjoldet og koble det til nettverket. Det første forsøket mislyktes og senere undersøkelser viste at bruken av AT-kommandoer var mer sannsynlig å lykkes. Men også dette har ikke vært i stand til å oppnå ønsket om en nettbasert Arduino og vil trenge videreutvikling for å lykkes.

Contents

List of Figures

List of Tables

List of Abbreviations

AC	Alternating Current
AM	Air Mass
APN	Access Point Name
BDLS	Bi-Directional Level Shifter
CAD	Computer-Aided Design
DC	Direct Current
DHI/DIF	Diffuse Horizontal Irradiance/Irradiation
DNI	Direct Normal Irradiance/Irradiation
EPT	Department of Energy and Process Engineering
GHI	Global Horizontal Irradiance/Irradiation
GND	Ground
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
GTI	Global Tilted Irradiance/Irradiation
IC	Incremental Conductance
IDE	Integrated Development Environment
IoT	Internet of Things
IV	Current-Voltage
LCD	Liquid Crystal Display
LiPo	Lithium Polymer
MPP	Maximum Power Point
MPPT	Maximum Power Point Tracking
MQTT	Message Queuing Telemetry Transport
NOK	Norwegian Kroner
NTNU	Norwegian University of Science and Technology
NVE	The Norwegian Water Resources and Energy Directorate
P&O	Perturb and Observe
PTC	Positive Temperature Coefficient
PV	Photovoltaic
PWM	Pulse Width Modulation
RTC	Real-time Clock
SCL	System Clock
SDA	System Data line
SD card	Secure Digital card

SIM	Subscriber Identity Module
SPI	Serial Peripheral Interface Bus
TCR	Temperature Coefficient of Resistance
VAT	Value Added Tax
VATL	Varmetekniske laboratorier
VCC	Voltage Common Collector
VDC	Voltage Divider Circuit
WHO	World Health Organization

List of Symbols and Units

A	Ampere
A_{PV}	Effective PV cell area in square meters
I_{SC}	Short-circuit Current in amperes
I_{MPP}	Current at maximum power point in amperes
m	Meters
P_{max}	Peak power in watts
P_{MPP}	Power at maximum power point in watts
R	Resistance in ohms
R_{tot}	Total resistance in ohms
V	Volts
V_{in}	Voltage in
V_{MPP}	Voltage at maximum power point in volts
V_{OC}	Open-circuit Voltage
V_{out}	Voltage out
W	Watt
W/m^2	Watts per square meter
Wh	Watt hours
Wh/m^2	Watt hours per square meter
W_p	Watt peak
η	Efficiency (dimensionless)
ϕ	Average solar irradiance in watts per square meter
Ω	Ohm

1 Introduction – Energy Transition in Africa

Many of the African countries are growing both in population and in economy. This is at least true for Tanzania, which has an annual population growth of 3 % [1]. Their economy is also on the rise with numbers from 2022 stating a GDP growth between 4 and 5 %. This rise in economy is largely due to the country's tourism, mining and electricity industry [2]. Although this means the country is prospering, rural areas and a large part of the population go without electricity. The power grid is unstable, causing random blackouts in the cities. Tanzania has per now a low percentage of energy coming from renewable sources at 0.7 % when one excludes hydropower. The majority of energy comes from natural gas, with hydropower and backup generators running on petroleum providing the remainder. Less than 10 % have access to clean cooking and solutions for sufficient energy production are of vital importance. [3, 4]

To implement a green, sustainable energy transition, the participation of developing countries is crucial. Introducing new energy solutions that can be utilized in such countries where a population is in growth, and in need of more energy to further develop, is essential. These new energy solutions might need more development, but serves as a tool to lift a large part of the human population out of poverty and into a life where essential commodities are covered. It is an important work that well-developed countries should seek to aid.

Energy for cooking is one of the major concerns in developing countries. Reports from the World Health Organization (WHO) [5] surmises that 3.2 million humans die from household air pollution exposure due to incomplete combustion of the fuels utilized for cooking. Fuels such as wood, coal, charcoal and kerosene are not only polluting the household air, but also contribute to the rise of CO_2 in the atmosphere. The utilization of such fuels causes a large impact on the continents deforestation, which is at twice the world average rate. Charcoal for instance is a relatively cheap and easily available resource for the population, and as the demand increases, so does the deforestation. For the instance of Tanzania, only 1.6 % of the population in rural areas have access to clean cooking solutions. This value increases to 6.8 % when one accounts for the entirety of Sub-Saharan Africa. Off-grid systems based on renewable energy solutions have an immense potential in these areas. These solutions will also help to suppress deforestation, and eventually put an end to the charcoal production. [5–8]

As the power grid in rural Africa is suffering from insufficient infrastructure, off-grid systems should constitute a part of the solution for energy supply. Renewable energy sources, such as solar and wind, play an especially important role in this solution. These energy sources, however, have limitations, in that they are intermittent. The time varying in energy production calls for solutions of storing excess energy for later use. Heat storage can be a solution for off-grid systems in addition to electrical batteries, such that the energy can be available for cooking at a desired time.

1.1 Thesis Background

This master thesis is a continuation of the preliminary project work: *Testing of controller for PV2Heat* [9] which was fulfilled during the spring of 2023. In the project description of that work, an Arduino Data Logger was to be made along with a housing, which was drawn in ArchiCAD and 3D printed at NTNU Gløshaugen. Furthermore, the project was to include testing of the available controllers at the Department of Energy and Process Engineering (EPT) laboratory. Due to time restraints the work focused more on the completion of the data logger and so, the testing of controllers has been reassigned to this master thesis.

This thesis is a part of an ongoing collaboration between NTNU and several universities in Africa, which has been developing off-grid sustainable energy solutions. Prior projects have been developing systems using solar and wind energy, often combined with heat storage. However, these prior projects have not performed extensive testing on the PV controllers which were used. The testing has therefore not taken into account the benefits of using the best optimized resistance of heating elements for the controller. Utilizing heating elements with resistances that are optimized will lead to better performance, as it allows the controller to work at its optimum. The desire is therefore, to conduct a thorough testing of available controllers directly from PV to heating elements. This will ensure that one has the knowledge of what resistance performs best for the controllers at certain sun conditions.

The Arduino data logger developed during the project work, is meant as a standalone unit which can take in data from four thermocouples. That data is stored on an SD card and current measurements can be shown on the systems display. The project work included the manufacturing of a housing for the system, which was drawn in a CAD software and then 3D printed in three parts. However, the historical data can only be extracted from the SD card, and so it has to be removed from the system for file transfer.

1.2 Objectives

The objective of the work is to support the collaboration projects with African universities on the development of heat collection and storage solutions for cooking. The main tasks to be evaluated are:

1. Establish a review of available direct controllers for PV-to-Heat both on MPPT and constant voltage controllers
2. Test the available controllers using the PV panels and suitable heating elements at EPT
3. Evaluate other controllers than those available at EPT
4. Prepare and test an extended version of an Arduino based data logger with web-based access
5. Reporting

The enumerated list of problem statements from the thesis description, starts with an establishment of the available controllers at the EPT laboratory of NTNU Gløshaugen, Trondheim. In conjunction with that, the evaluation of similar controllers is an investigation into alternative solutions for sustainable energy management between solar power and heat storage. The testing of the available controllers at the EPT laboratory will be performed as a part of an experimental work, along with the further development of the Arduino system for a web-based solution.

1.3 Citation

As this master thesis is a continuation of a completed project work, some sections are transferable as they regard equal or associated topics. Therefore, the following sections are either copied, slightly modified or modified from the preliminary project work: *Testing of controllers for PV2Heat* by Egerdahl, Martin August. [9]

Table 1.1: Sections not modified, slightly modified or modified from the preliminary project work. [9]

Section	**Modification**
1	Modified
2.7	Slightly modified
2.4	Modified

1.4 Structure of the Report

The following is an overview of the structure of the report.

Chapter 2 – Technical Background: The necessary theory for the understanding of the thesis is presented in this chapter. It is the foundation of what is presented and discussed in the results and discussion sections of the thesis. Additionally, there are sections of the chapter that are transferred from the project work with additions and/or improvements.

Chapter 3 – Literature Review: A concise description regarding the previous work by students with a similar topic, in addition to a review of relevant literature.

Chapter 4 – Review of Controllers: A chapter regarding the available controllers and an investigation into other controllers than those available for the project.

Chapter 5 – Experimental Conditions: The section describes components and setup related to the experimental work. It presents the location of testing, specifications of the PV panels and heating elements that are used during the experimental work. Also, the experimental setup is presented, along with an introduction to the Arduino system developed during the project work and its further development.

Chapter 6 – Experimental Work: Methodology of the experimental work is explained in the section, along with presentations of the results from testing of the controllers and specific discussion on these. In addition, results regarding the work on a web-based Arduino system is presented and discussed.

Chapter 7 – Conclusion: The section includes important findings from the results and discussion to conclude upon the work of the thesis.

Chapter 8 – Further Work: The further work is proposed regarding both PV-to-Heat and Arduino based data logging and monitoring.

2 Technical Background

Following in this chapter is the background knowledge on solar power, controllers and Arduino, which are relevant for the experimental work and the project in its entirety.

2.1 Solar Power

Solar power is one of the renewable energy solutions which has had a significant increase in installed capacity the last decennium. For 2023 the cumulative share of power capacity is at 14.7 %, soon to surpass that of hydropower at 15.7 %. By 2027 the International Energy Agency expects this solar percentage to rise past coal and become the dominating energy solution worldwide. This comes partially as a result of further development of photovoltaic (PV) panels and subsequently, lower prices. Also, due to the modular nature of PV power systems, ranging from small off-grid solutions, e.g., a cabin or caravan, to residential systems and large utility-scale power plants. [10]

An important definition of PV power is that of Watt peak (W_p). W_p is the unit of power for PV panels at standard test conditions. The three conditions are: 25° C cell temperature, Air Mass 1.5 (AM1.5) and a solar irradiance of 1 000 W/m^2. AM1.5 is a measurement of the distance, energy from the sunlight has to travel through the atmosphere and is dependent on the angle of incidence. AM1 is defined by the sun at zenith, while AM1.5 is 48.2° from zenith. AM0, also known as the solar constant, is the solar irradiance at the edge of the atmosphere, which is equal to 1 367 W/m^2. W_p facilitates a correct comparison of PV panels from different manufacturers. [11, 12]

PV panels efficiency is affected by temperature changes. For a temperature increase the open-circuit voltage of the PV panel decreases by a factor of 2.3 mV per degree. On short-circuit current the increase in temperature will lead to an increase of 6 µA per degree. A lower temperature will therefore lead to a larger increase in V_{OC}, than the effects of temperature on I_{SC}, making PV panels more efficient at lower temperatures. [13]

2.2 Solar Radiation

Radiation from the sun can be assessed in two distinct measures, that of solar irradiation and irradiance. Both important concepts to distinguish. Solar irradiation is the radiation from the sun at a specific area over a period of time, and is measured in watt-hours per square meter Wh/m^2. Solar irradiance however is that solar radiation at a specific area for one moment in time, and is measured in watts per square meter W/m^2. [14, 15]

The solar radiation can be divided into several methods of measurements. This divide is mainly split into:

- GHI – Global Horizontal Irradiation/Irradiance
- GTI – Global Tilted Irradiation/Irradiance
- DNI – Direct Normal Irradiation/Irradiance
- DHI/DIF – Diffuse Horizontal Irradiation/Irradiance

GHI is a measurement of solar irradiation or irradiance where all aspects of the solar radiation are taken into account, i.e., direct, diffuse and reflected (albedo) radiation. This measurement is done on a horizontal plane and from every direction, as in a hemispherical field of view. GTI is similar to GHI, except that the horizontal plane is exchanged with a tilted plane. This tilted plane is then often related to the plane of which PV panels are directed, e.g., a PV panel at 45° elevation, will have a device at the same angle, measuring global irradiation/irradiance. DNI is the direct component of GHI, meaning it measures the solar radiation perpendicular to the angle of incident of solar rays. It excludes the diffuse and reflected radiation, which instead compose the value of DHI. DHI is sometimes referred to as DIF. Figure 2.1 displays the difference in solar rays towards a PV panel, along with GHI. [15–17]

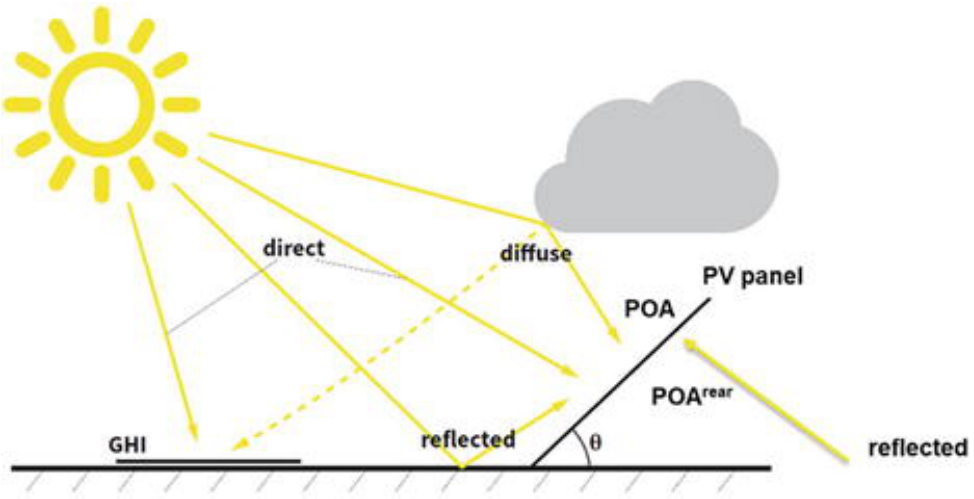


Figure 2.1: Direct, diffuse and reflected solar radiation towards a PV panel and the illustration of GHI. [18]

Calculations for the average solar irradiance can be derived from the equation 2.1, measured in W/m^2. In the equation, P_{max} represents the peak power of a PV panel in watts, while η is the PV panels efficiency and A_{PV} is the effective PV cell area in m^2. [11]

$$\Phi_{in} = \frac{P_{max}}{\eta \cdot A_{PV}} \tag{2.1}$$

2.2.1 Solar Radiation in Norway

Figure 2.2 depicts the solar irradiation of Norway for a day in January to the left, and July to the right. The figure depicts Norway as having a peak irradiation in the summer month of about 5.5 kWh/m^2 per day. For the winter month this has dropped to approximately 350 Wh/m^2. Both peaks are in the southern regions of the country. For Trondheim, the amount of solar irradiation for one year generally fluctuates around 800 kWh/m^2. [11] As per 02.08.2023 the installed capacity of PV power in Trondheim is approximately 6.5

MW_p, with an anticipated yearly production of 4 GWh. Comparing to the entirety of Norway's PV capacity of 473 MW_p from its 24 619 installations, yielding an expected annual power production of 359 GWh. [19]

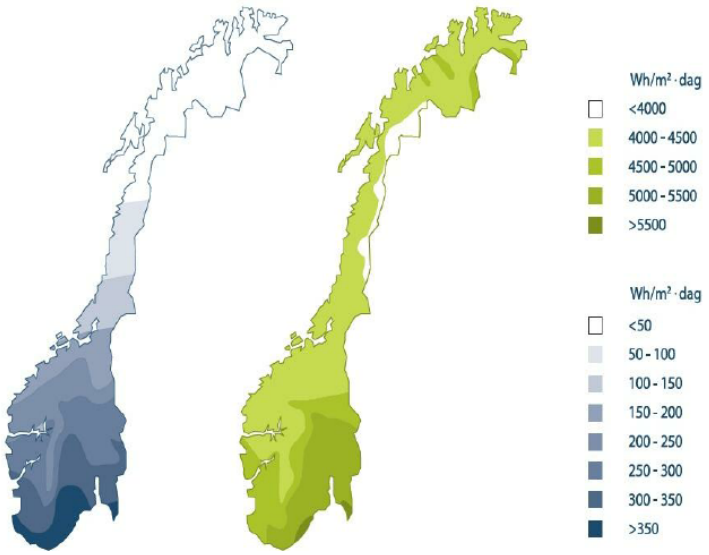


Figure 2.2: Solar irradiation of Norway for a day in January and July, to the left and right respectively. [20]

Although Norway has a relatively low amount of solar irradiation compared to Africa, the low temperature is beneficial and leads to more efficient PV panels. Excluding region and efficiency, other factors involved in the performance of a PV panel are the tilt angle, shading, orientation and the configuration of the PV system. Estimates from NVE state that a general PV system in Norway can produce between 650 and 1 000 kWh/kW_p per year. [21, 22]

2.2.2 Solar Radiation in Africa

As the continent of Africa is split by the equator, sun conditions are strong and the potential for the use of solar powered energy production is high. Figure 2.3 depicts the GHI on both a daily and yearly basis for the entire continent, extracted from a time period of 26 years. At its highest points, the daily GHI is approximately 6.5 kWh/m^2 and yearly this value increases to 2 400 kWh/m^2. However, this data of solar irradiation does not take into account the difference in solar irradiation from season to season, as the figure 2.2 for Norway does. For Tanzania specifically the average solar global horizontal irradiation ranges from 4.74 to 6.53 kWh/m^2, depending on the local region of the country. Average value of the GHI is 5.66 kWh/m^2, while the specific PV power output of the area has an average of 4.51 kWh/kW_p per day. [23, 24]

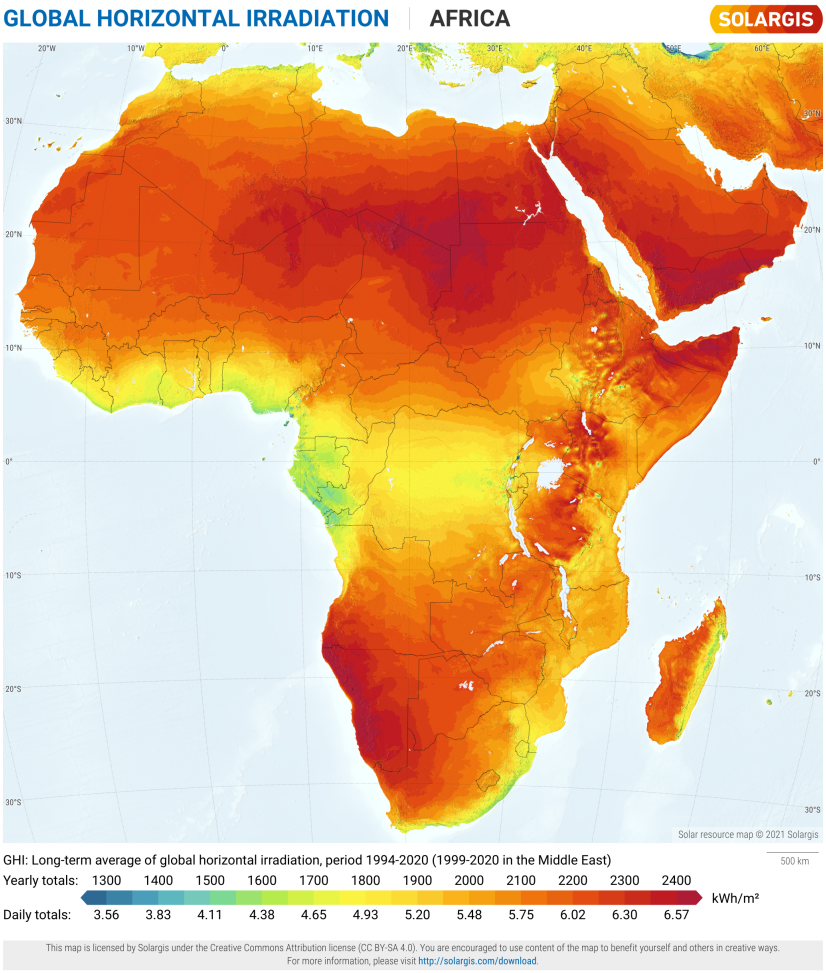


Figure 2.3: Solar irradiation of the African continent, depicting GHI on a daily and yearly basis. [23]

2.2.3 Solar Radiation Measurement Methods

Measurements of the solar radiation is usually performed by one of two measuring instruments, the pyranometer or pyrheliometer. The pyranometer is the more common of the two as it is used for measuring GHI. It has a hemispherical design, meaning it measures both direct and diffuse irradiance with a 180° field of view. Usually, the instrument is mounted on a horizontal plane. However, it can also be used for measuring GTI and in such cases is generally mounted at the same angle as a PV system. The pyrheliometer differs from a pyranometer in that it is directional. Typically mounted on a rotating platform, which follows the sun during the day. The pyrheliometer therefore, is generally used for measuring direct normal irradiance/irradiation. Figure 2.4 depicts a schematic of the pyranometer and pyrheliometer in reference to solar radiation. [15, 25]

Both the pyranometer and pyrheliometer exploits the function of a thermopile, which converts heat to an electric current. This electric current can be measured and when calibrated, be adopted to a value of solar irradiance in W/m^2. To be accurate these thermopiles must have a high absorptivity and a low albedo. For the pyranometer, the hemispherical glass protects the thermopile against convection. For both instruments the

glass is a filter of solar radiation, only letting through specific, desirable wavelengths. [15, 25]

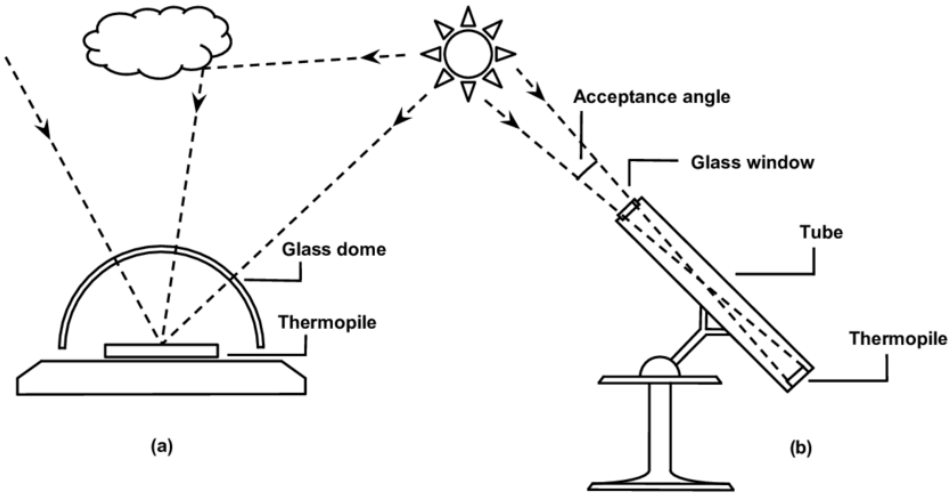


Figure 2.4: Schematic of a pyranometer and pyrheliometer in relation to solar radiation. [15]

2.3 PV Systems

PV systems are divided into two classes, grid-connected and off-grid systems. Grid connected PV systems are typically larger systems for households, industrial/commercial buildings or solar farms. For households and commercial buildings, they produce larger amounts of energy than might be necessary at one time, and it is more profitable to sell energy to the national grid, than store it for oneself. A grid-connected system could also include other power sources in combination with PV, such as wind and battery storage. Although, battery storage is a somewhat costly option. [26]

Off-grid systems are standalone systems of power generation and local consumption. Generally smaller in size than grid-connected, the off-grid systems are usually implemented in rural areas or more remote locations, such as cabins, which is common in Norway. Figure 2.5 is a schematic of an off-grid system, specifically with battery storage and AC load. The first step of the system is PV energy into a solar charge controller, which will be expanded upon in the following chapter. The controller has a DC connection with a battery for storage. In this instance the energy is used in an AC circuit by implementing an inverter, but an off-grid system with a DC circuit or AC & DC circuit is possible as well. However, a battery is not a necessity for an off-grid system, as long as all the energy can be used when it is produced in DC applications. An example of this is to use heating elements in place of the battery or as an addition to it. [26]

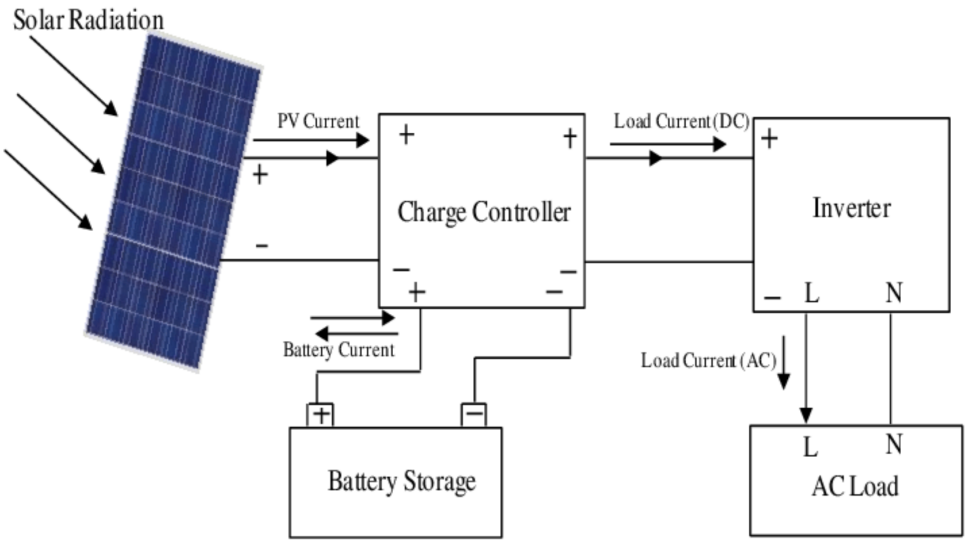


Figure 2.5: Schematic of a general off-grid PV system. [27]

2.4 Solar Charge Controller

A solar charge controller is a unit between the energy produced by a PV panel and how that energy is to be applied for charging and control of a battery, DC or AC load. The function of these controllers is to regulate the current and voltage from a PV panel to optimize power delivery. This regulation is meant to keep the power at the Maximum Power Point (MPP) when sun conditions vary. The PV panels have a current and voltage curve, often denoted as IV-curves, which is seen as the blue lines in figure 2.7. The red lines in the figures are those of the power curves. Controllers used with batteries in off-grid systems, will control the power to the battery and prevent overcharging and depletion. [28, 29]

The figures 2.7a & 2.7b depicts two of the most common types of controllers that are used, the Maximum Power Point Tracking controller (MPPT) and a Pulse Width Modulation (PWM) controller. PWM, as seen in figure 2.7b, is a digital signal with a rectangular waveform. The modulation is done by regulating the duty cycle, meaning the time the signal is at full amplitude or at zero. A duty cycle of 75 % translates to a signal which is at full amplitude for 75 % of one period. Typically, an MPPT controller is better for colder climates, and the two types of controllers will have more equal performances in tropical and subtropical climates. [29–31]

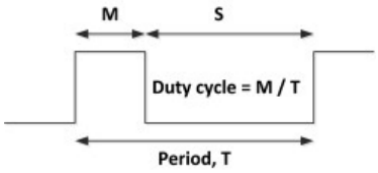
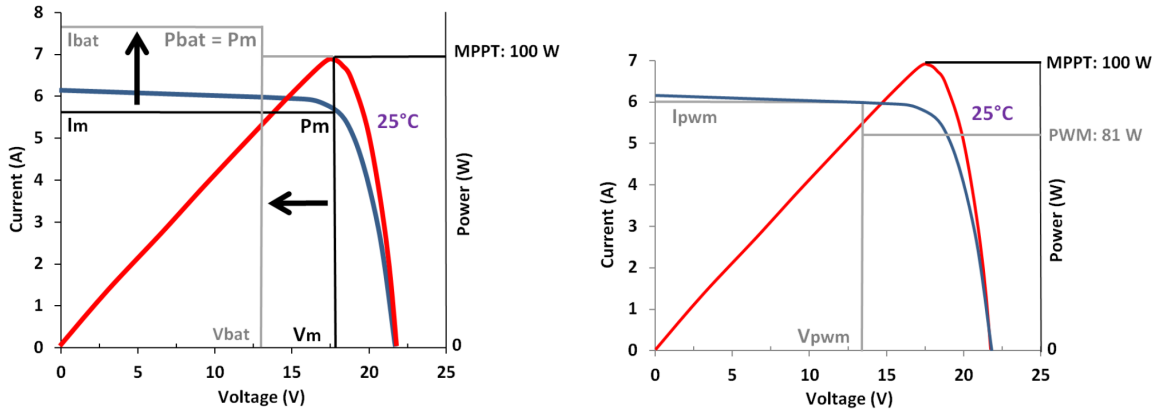


Figure 2.6: Nature of a Pulse Width Modulation signal. [32]

The MPPT controller is a newer and more sophisticated technology compared to the PWM controller, although it too is a PWM signal on the output. It is essential for optimizing power output from an array of PV panels. An MPPT controller is a DC-DC transformer which will operate by reducing the input voltage such that it is operating at the MPP and when this is done, it increases the current from the input to output. Therefore, the controller does not reduce the power produced to that delivered, when disregarding small transformation process losses. An MPPT controller can have efficiency ratings as high as 98 % [33]. While PWM controllers are matching the PV panels voltage to the battery voltage on the output side for it to operate properly, there is not a necessity for an MPPT controller to adjust for this. An MPPT controller is able to have a high input voltage from a PV array to the controller, and then for it to transform that energy to a low voltage battery, meaning they are disconnected from each other and not a factor to consider. The controller benefits therefore from a high input voltage and will perform well in low and high cell temperature environments, but also at lower irradiance. [29, 30]



(a) MPPT.

(b) PWM.

Figure 2.7: IV- and power curves of Maximum Power Point Tacking and Pulse Width Modulation controllers. [29]

A traditional PWM controller, when connected to a battery will perform in such a way that the PV voltage on the input of the controller will be pulled down to the lower voltage of a battery on the output connection. As it is an older and simpler technology the PWM controller are generally cheaper in price than that of the MPPT. However, the typical conversion efficiency of a PWM controller is lower, at approximately 75 – 80 % [33]. The most beneficial environments for a PWM controller is in small-scale systems where the cell temperature of the PV panel reaches the range of $45 - 75^{\circ}C$. In these conditions the boost effect of the MPPT controller reduces, which minimizes the performance difference between them. However, the PWM controller will perform worse with lower irradiance, partial shading and PV arrays that are not optimized against the battery voltage. [29, 30]

MPPT has several possible control algorithms to boost performance. The most common MPPT technique is that of the Perturb and Observe (P&O) method. Figure 2.8 is a

schematic of the classic perturb and observe algorithm. It is a simple method of tracking the MPP, where it regulates the input voltage or current depending on if the output power has increased or decreased compared to the last observation point, which is done at regular time intervals. The advantages of this method are simplicity and good performance. However, low solar irradiance, swift changes in the solar irradiance and oscillations about the maximum power point are unfavorable for the performance of the P&O method. [34]

Other notable types of MPPT techniques are the Incremental Conductance (IC) algorithm and Fuzzy Logic control. IC is an improvement of P&O, where the benefits include: determination of when MPP is reached, better tracking during swift changes in solar irradiance and less oscillations about the MPP. Both P&O and IC methods of MPPT are suitable for PV systems in residential areas, as they perform well when tracking and adjusting the MPP. The simulations performed by Babaa et al. [34], which are the basis of these conclusions, used a DC-DC boost converter as a controller between the PV panels and the load. A boost converter is a device which will increase an input voltage and output a stable current of power. Other converters include buck and buck-boost. A buck converter is similar in design to a boost converter, however, instead of increasing the input voltage, a buck converter will lower it to the output. A buck-boost converter is a combination of both and therefore has both abilities, either increase or lower the input voltage of the device to the stable output voltage. [34, 35]

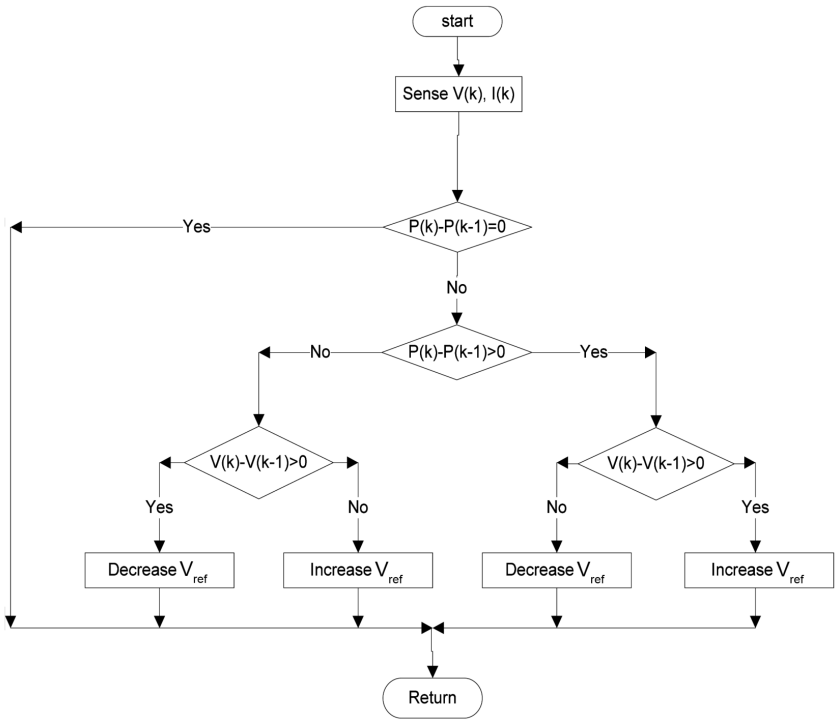


Figure 2.8: A schematic of the general perturb and observe method. [34]

2.5 Heating Elements

Heating elements are electrical devices that convert electricity to heat, by sending a current through a resistive alloy wire which heats the element. An important aspect of heating elements is the Temperature Coefficient of Resistance, TCR. The coefficient is defined by the change in resistance of a material from an ambient temperature. [36, 37]

Tubular heating elements, such as those produced by Backer, often use a nickel-chrome alloy for its resistance wire. The alloy, also known as nichrome, has a TCR of 0.0004 $/^{\circ}C$. However, the coefficient is not linear in relation to the temperature, meaning that they can vary depending on the start and end temperatures. The nichrome is often insulated with a magnesium oxide powder before it is enclosed in the tubular element. Such tubular heating elements can be formed in various ways, be used for multiple heat mediums and are able to reach temperatures as high as $800^{\circ}C$. [36–38]

Another type of heating elements that is interesting, are the self-regulating PTC elements. Positive Temperature Coefficient elements, PTC, are ceramic materials, which often use barium titanate composites. Unlike more linear pure ohmic resistance heating elements, the PTC elements are highly non-linear with regard to temperature change. As seen in figure 2.9, the PTC element have such a characteristic that the resistance increases with temperature in a manner that the current can scarcely flow through the resistor after the element has reached its design temperature. Compared to the tubular heating elements, this means that PTC elements are not able to have a runaway temperature increase above what the elements are rated for, which makes them safer in use. However, PTC elements can be made with different maximum temperatures. [37–39]

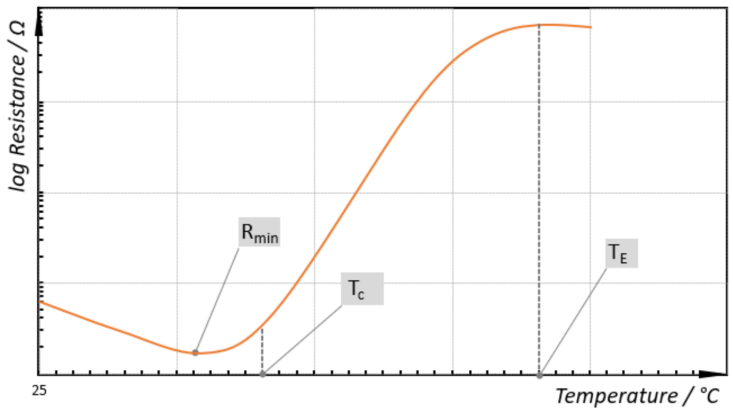


Figure 2.9: Graph of resistance change with temperature for PTC elements. [39]

2.6 Electrical Components

To calculate when multiple resistances are connected with each other, the equations below are necessary. Equation 2.2 are for series connection of resistances, while equation 2.3 are for parallel connections. However, equation 2.2 can be used in equation 2.3 if the desired circuit has a series of resistances in parallel with another series of resistances.

$$R_{tot} = \left(\sum_{i=1}^{N} R_i \right) \tag{2.2}$$

$$R_{tot} = \left(\sum_{i=1}^{N} \frac{1}{R_i} \right)^{-1} \tag{2.3}$$

A **Voltage Divider Circuit** is an electric circuit which is used to produce a lower output voltage than the input. To achieve this, the circuit uses two resistors as shown in figure 2.10. The parameters of this circuit are V_{in}, R_1, R_2 and V_{out}. Their relation is as in equation 2.4. [40]

$$V_{out} = V_{in} * \frac{R_2}{R_1 + R_2} \tag{2.4}$$

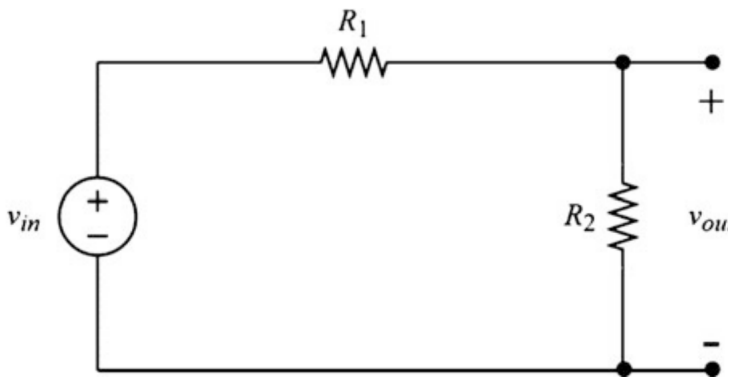


Figure 2.10: Schematic of a voltage divider circuit. [40]

A **Shunt** is a current sensor which usually has a low resistance resistor to run the external circuit through, to enable measurements of the current using Ohms law. However, it can also be done by the Hall effect, where the current moving through the sensor is creating a magnetic field that the integrated circuit of the sensor can translate to a proportional voltage. Figure 2.11 is a schematic of how a shunt resistor is set up to measure current. [41]

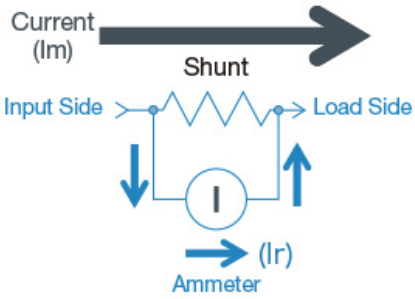


Figure 2.11: A basic shunt resistor schematic. [42]

A **Thermocouple** is a temperature sensor, which consists of two metal wires of different material. When the wire is touching an object that is increasing/decreasing the temperature, a potential difference occurs that is proportional to the temperature difference at the tip of the wire and the reference of the measuring device. K-type are the most prevalent form of thermocouples as they are cheap and have a wide temperature range. [43]

2.7 Introduction to Arduino

Arduino is a hardware manufacturer of microcontrollers and software for programming those controllers. It is based on an open source platform and aims to aid a user experience that is both easy to use and accessible. The possibilities are almost limitless with the Arduino systems and the way one can produce a microcontroller to aid one's purposes. To use the Arduino there is usually a main board, which has some form of communication possibilities with a laptop, typically a USB port. Besides the main board there are other purpose-built boards known as *shields* that can have a multitude of different functions. These shields are generally stackable above the main board so they can communicate to each other through stacking pins. This stack of Arduino shields can also be connected to parts outside of the stack to serve more purposes. The Arduino microcontrollers and the entire network that encompass it provides the access to opportunities for everyone from amateurs, teachers and students to professionals. Arduino have numerous advantages as a microcontroller system, but it is best expressed by themselves: "Inexpensive; Cross-platform; Simple, clear programming environment; Open source and extensible software; Open source and extensible hardware". [44–46]

Following is a list where relevant terminology has been expanded on:

> **I2C** is an abbreviation for *inter-integrated circuit*. It is an interface bus for serial communication. The advantage of this bus is that it only requires two wires in addition to power and ground. There is a master and slave protocol which keeps the interface coordinated. The two extra wires that are needed are for system data (SDA) and system clock (SCL). SDA receives and sends data between the master and slave, while SCL transfer the clock signal. [31, 47]

RTC is an abbreviation for *Real-Time Clock*. It is a unit that will keep track of time in seconds, minutes, hours, days and years. There are adjustments made for leap years and months of less than 31 days. It often run on a separate power source to what it is maintaining time for, and uses the I2C bus for communication. [31]

SPI is an abbreviation for *Serial Peripheral Interface Bus*. Much like the I2C it is a communication interface with a master and slave protocol. Where it differs from the I2C is how it is addressing the slave device, as this uses a chip-select line while I2C have exclusive addresses. [31, 48]

UART is an abbreviation for Universal Asynchronous Receiver/Transmitter, and it is a form of communication for a microprocessor, much like I2C and SPI. Where it differs is that UART is asynchronous, and therefore there is no need for a clock signal between the two devices that are communicating. For communication with UART, three connections or wires between the devices are needed. TX out of device 1 is for sending data to RX of device 2. RX on device 1 is for receiving data from TX on device 2. The third connection is a common ground for the two devices. What is important when using no clock signal is that the baud rate of the two devices are set to be the same. [31, 49]

2.8 MQTT

Message Queuing Telemetry Transport (MQTT) is a protocol similar to HTTP. It is a messaging protocol that is used for the Internet of Things (IoT). HTTP has good potential for devices that send copious amounts of data. It is sometimes used for IoT devices, but it is not ideal. HTTP uses a pull only method of transmitting data, meaning that a temperature sensor can send data to a server when it wants. But, if the device wishes to pull from a server with HTTP, it must continuously ask for updates, which is using large amounts of data. [50–52]

Due to the problems of HTTP, MQTT is a better protocol for IoT devices. It is lightweight and efficient. It uses lower amounts of data to connect to a server, also known as an MQTT broker, which leads to lower battery usage. The lower battery usage is important for IoT devices that will often be wireless, small units. Unlike HTTPs pull only method, MQTT is always connected to the server, and it uses a Publisher/Subscriber architecture, which is bidirectional communication, compared to the unidirectional HTTP. Therefore, MQTT is a more suited protocol for use in IoT systems, and it is often recognized as the standard for messaging in IoT. [50–52]

Figure 2.12 is a schematic representation of how MQTT clients converse with the broker. Both the temperature sensor and computer/mobile device are clients, and the broker is a means of transporting messages. One client publishes messages to the broker, while other clients can subscribe to messages. An MQTT client will subscribe to the broker using a topic, such as “temp” in the figure, to receive data from the publishing client. [53]

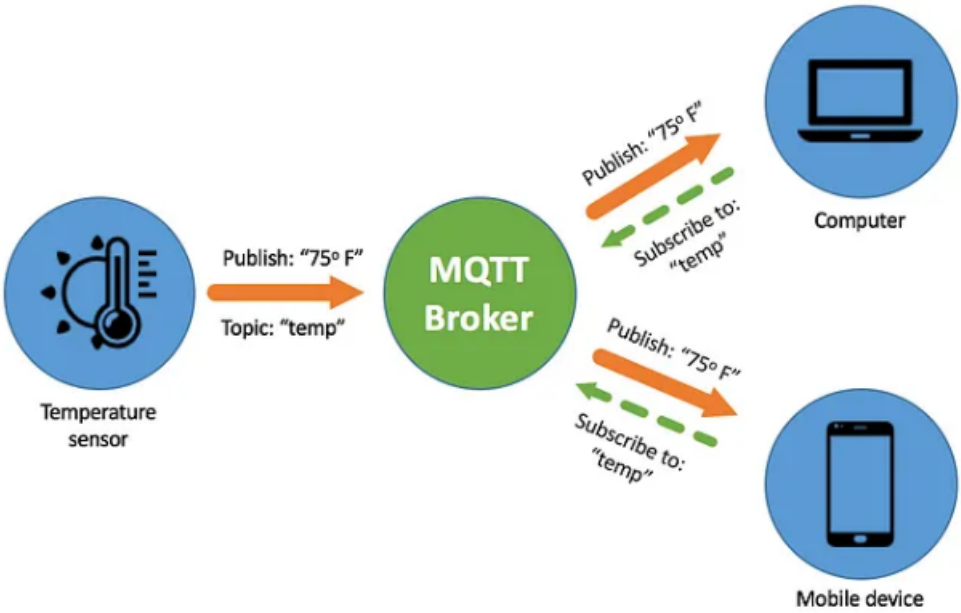


Figure 2.12: Relation between the clients and broker of the MQTT protocol. [52]

3 Review of Controllers

As a basis on solar charge controllers, a review of the available units for the experimental work are described in the following section, along with alternatives from other suppliers.

3.1 Available Controllers for the Experimental Work

There are three solar charge controllers that are available at the EPT laboratory of the NTNU campus Gløshaugen, in Trondheim. Two of which are from the South African manufacturer Geyserwise. The manufacturer started its business in 2004 on developing controllers for the solar water industry in the country. The difference between the two models is the load voltage, which is 72 and 48 volts. Therefore, the models will be referred to as 72V and 48V. Externally, the two models are equal and can be seen in figure 3.1. The Geyserwise controllers are MPPT controllers which utilize a buck converter to step down the input voltage and boost the current so that it regulates to a maximum power point. [54]

Figure 3.1: The Geyserwise ECO MPPT solar charge controller. 72V & 48V model are in equal housings.

The exact model is labeled as a Geyserwise ECO MPPT, and there are particular specifications which are important for the experimental work. Firstly, the controllers have a range of input voltage which should not be exceeded, to ensure no damage is done to them. This range is between 46 and 138 open-circuit voltage. The peak efficiency of the controllers is at 98 % and the maximum output current is at 25 amperes. More information on the Geyserwise controllers is in appendix B.

The third of the available solar charge controllers is a unit developed by Odin Hoff Gardå in February of 2020. It is also an MPPT controller with a buck converter. The controller will be referred to as the Odin controller throughout this thesis. It uses a standard perturb and observe algorithm for its maximum power point tracking. The specific perturb and observe algorithm, along with other information on the controller such as a system diagram, is added in appendix C. The controller is equipped with six inputs for thermocouples, and it is capable of logging data from these temperature sensors. As an additional function the controller is equipped with a potentiometer, so that it can be adjusted manually or run in the automatic MPPT mode.

Furthermore, a LabVIEW program is established along with the controller to access the information it provides. The controller measures current and voltage from the lowest PV panel of the stack used in the experimental work and presents this in the LabVIEW program. As it is only measuring one panel, the value is multiplied to approximate the total voltage from six panels in the PV array.

The controller has a set of maximum ratings which are important to acknowledge. For a single panel, the maximum voltage input is at 40 V, and with six panels in series it is possible to tolerate voltages up towards 230 V. Unlike the Geyserwise controllers with their 25 A maximum load current, the Odin controller can only sustain 10 A. An important aspect to note with this controller, is that the temperature in the heat sink should be monitored and not exceed $40 - 50^{\circ} C$.

Figure 3.2: The solar charge controller developed by Odin Hoff Gardå in 2020.

3.2 Alternative Controllers

There are several options for solar charge controllers, from multiple manufacturers and with different models within those manufacturers. Two well-known manufacturers of solar charge controllers are Victron and Morningstar. Victron produce mainly MPPT controllers meant for battery connection, but they are easily available for acquisition. However, if these controllers are compatible with connection to heating elements is somewhat uncertain. The controller can possibly work in a PV system with batteries and heating elements, where a Victron BMV battery monitor can be used to manage the system. The Victron model SmartSolar Charge Controller MPPT 150/35 has similar specifications to the Geyserwise 48V. The battery load voltage on this controller is capable of 48 V and a maximum current of 35 A, while the Geyserwise model has a maximum of 25 A. In addition, the two models from Victron and Geyserwise have the same efficiency at 98 %. An interesting aspect of this controller is the built-in Bluetooth connection for wireless data extraction, along with storage of this data up to 46 days. A full specification sheet for this model is in appendix D. The model is recommended for a PV panel configuration of three series and two parallels by the Victron sizing calculator when using 260 W panels. [55–58]

Morningstar is an American company, which has produced solar controllers for nearly 30 years. They manufacturer both MPPT and PWM controllers, with multiple models for professional and consumer applications. Their TriStar model is produced in both control methods. The MPPT version is a solar charge controller with industry leading efficiency of 99 %. It is designed for off-grid PV systems connected to a battery load voltage up to 48 V and a maximum power input of 4 200 W. The PWM version of the controller has three separate modes of control: charge, load and diversion control. The charge control is designed to function as a battery charge controller. In the diversion control method, when the connected batteries are at a 100 %, the excess solar energy can be utilized with heating elements in a hot water system. As it is a PWM controller, it is preferable to use in stable sun conditions where shading is not affecting the performance by any substantial amount. The maximum input voltage for this model is 125 V_{OC} and with a load voltage up to 48 V. [59–62]

Microcare is a South African manufacturer of controllers, which was established in 1990. They have developed the PV Geyser Controller for utilizing the energy from PV panels to heat a hot water system. The controller's efficiency is lower than the MPPT models from Victron and Geyserwise at 96 %. There is no mentioning of it being an MPPT controller, so the assumption is that based on the efficiency being similar, it is an MPPT and not a PWM controller. It has a higher maximum open-circuit voltage input with 275 V_{OC}, than the Geyserwise and Odin controller. Figure 3.3 depicts a schematic of the Microcare Geyser Controller system, utilizing PV power and sending it to a heating element in a hot water tank. The specification sheet for this model is seen in appendix E. [63–65]

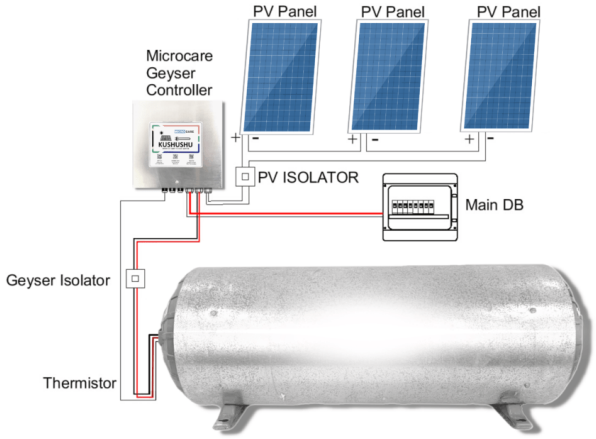


Figure 3.3: Schematic of a system with the Microcare Geyser Controller. [65]

Another South African manufacturer of a solar charge controller is GeyserWorx. They produce a microprocessor-based controller, which utilizes solar energy to heat a hot water system. The GeyserWorx Solar Geyser unit is capable of utilizing three to eight PV panels with an operating voltage of 100 – 280 V_{DC}. It is a more advanced system than that of the Geyserwise and Microcare controller, as it has the possibility of remote control and data extraction, similar to the Victron model. [66–68]

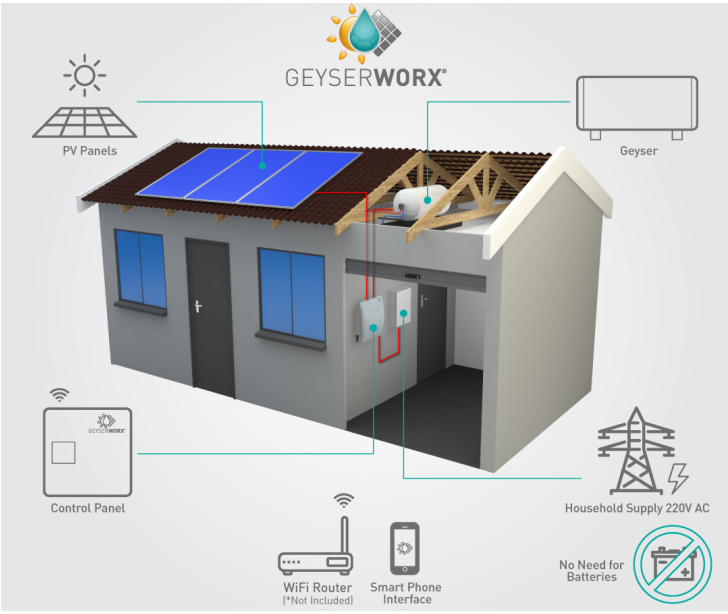


Figure 3.4: Diagram of the system setup with a GeyserWorx Solar Geyser unit. [68]

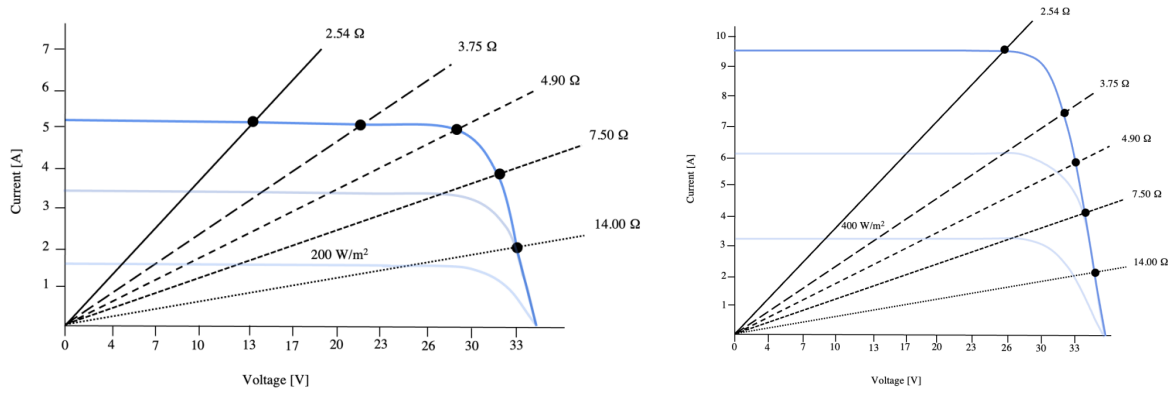
Fronius offers a solution for utilizing PV for water heating with their Fronius Ohmpilot. However, this is in cooperation with their inverters and is meant for larger residential systems, rather than small off-grid systems. [69, 70]

4 Literature Review

During the research on the use of solar charge controllers directly between harnessing solar energy with PV panels and heating elements, a limited amount of information has been found on the subject. In this literature review, influential findings of this research are gathered to acknowledge relevant studies that has been performed by earlier researchers and how that affects this experimental work.

A relevant previous work for the experimental part of this thesis, is that performed by K. Berg & A. A. Vik [37] during their master thesis on "Demonstration of PV Power to High-temperature Heat Storage for Solar Cookers". In this work, several case studies were performed with the same solar charge controllers which are available for the experimental work of this thesis. The focus of the work by Berg & Vik was to test a three-tank system, in which there were installed heating elements. Three different solutions were tested and, in those solutions, cases with the different controllers were used. Of the three solutions, two are interesting to examine, which were a battery system with diversion of excess energy to the heat storage and a direct PV-to-Heat system. Both the Geyserwise 72V and 48V were used, along with the Odin controller and a TriStar PWM controller. A separate case for testing the 72V controller with a PTC heating element was also performed. [37]

In their testing of the Geyserwise 48V controller, a parallel circuit of three PV panels were used, giving it an input voltage of 30.9 V, which is substantially lower than the lowest input voltage requirement of 55 V_{OC} for the controller. Along with this, the input voltage is higher than the limit, but it is commented on this issue, that it is likely not to be an issue due to the current never exceeding 20 A on previous experiments. The controller is connected to two 500 W heating elements with a combined resistance of 2.54 Ω, which results in an operating point quite far from the MPP as seen in the figure 4.1a. However, when the solar irradiance increases, the results are that the operating point is closer to the MPP, as seen in figure 4.1b. [37]



(a) Solar irradiance of 250 W/m^2.

(b) Solar irradiance of 400 W/m^2.

Figure 4.1: IV-curves for the results of testing the Geyserwise 48V controller, performed by Berg & Vik. [37]

Regarding the testing of the Odin controller by Berg & Vik, their PV configuration was six panels in series and the two 500 W heating elements as with the Geyserwise model. For this test, during similar conditions as the Geyserwise at 384 W/m^2, the resistance of the heating elements results in an operating point far from the maximum power point. Figure 4.2 depicts this poor result. [37]

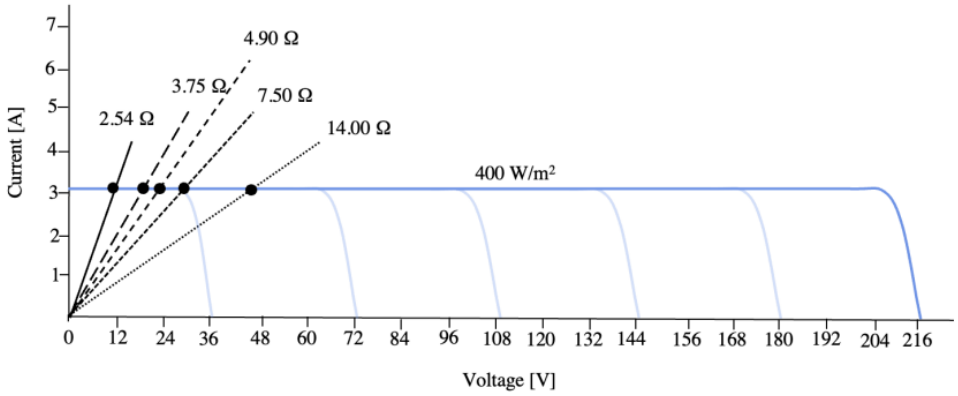


Figure 4.2: IV-curve for the Odin controller at a solar irradiance of 400 W/m^2, performed by Berg & Vik. [37]

The experiences from Berg & Vik in their experiments, is a foundation for which the experiments in this thesis will carry on and expand upon.

5 Experimental Conditions

The experimental work is performed at the Department of Energy and Process Engineering laboratory of NTNU Gløshaugen, Trondheim. As a part of the experimental work regards the testing of controllers between solar power and heating elements, there are PV panels located at the roof of the laboratory that are to be utilized. Along with the PV panels, the measuring device for solar irradiance is mounted on the same roof. The measuring device is a pyranometer, which is horizontally mounted and therefore measuring the Global Horizontal Irradiance. In figure 5.1, an image of the laboratory rooftop with a vertically mounted stack of six PV panels to the left. The pyranometer is mounted on a rod, suspending it a few meters above the rooftop, and is seen in the back right of the figure 5.1.

Figure 5.1: The PV panels on the roof of the Thermal Laboratory at Gløshaugen and a pyranometer mounted on a pole to the right in the figure.

Figure 5.2 depicts a direct overhead view of the Gløshaugen campus. The building in the center of the figure is the laboratory in which testing is performed. The building, known as VATL (Varmetekniske laboratorier), is located at the north-northeast corner of the campus.

The PV panels face a direction of approximately 165° south-southeast and are marked by a green circle in figure 5.2, while the pyranometer is marked yellow. The distance between them is approximately 34 meters.

Figure 5.2: Map of Gløshaugen campus with the PV panels marked in green, with the pyranometer for solar irradiance measurements marked by yellow.

There is another measuring station for solar irradiance at NTNU's Gløshaugen campus, which is managed by the Norwegian Meteorological Institute. An exact location on campus is unknown. The measuring device is produced by Kipp Zonen and is a first-class rated pyranometer. The data can be found at: MET Norway SN68173, and is possible to use to audit the data from the VATL rooftop pyranometer. [71]

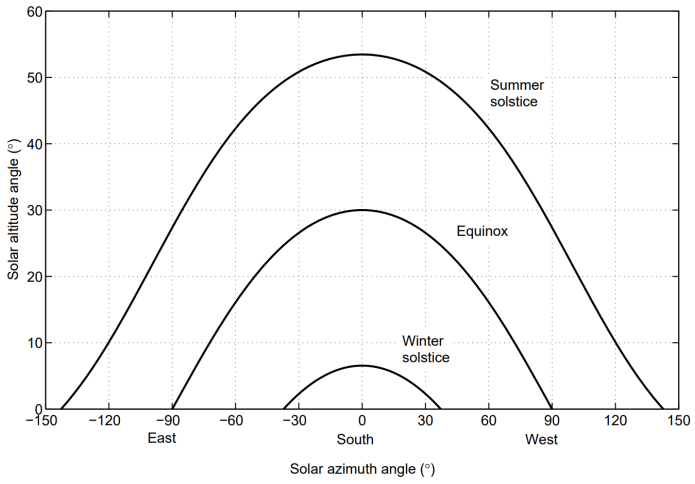


Figure 5.3: Sun path at summer solstice, equinox and winter solstice at a latitude of 60° N. [15]

Figure 5.3 is a schematic of the sun paths for summer solstice, equinox and winter solstice at a latitude of 60° north. The location for the experimental work, Trondheim, is at a latitude of 63.4° N. Therefore, it is comparable to the figure, although they should be slightly lower to represent the sun paths in Trondheim. According to timeanddate.com the angle of peak altitude for the sun in Trondheim varied from approximately 30° to 15°, from start to finish of testing for the controllers. [72]

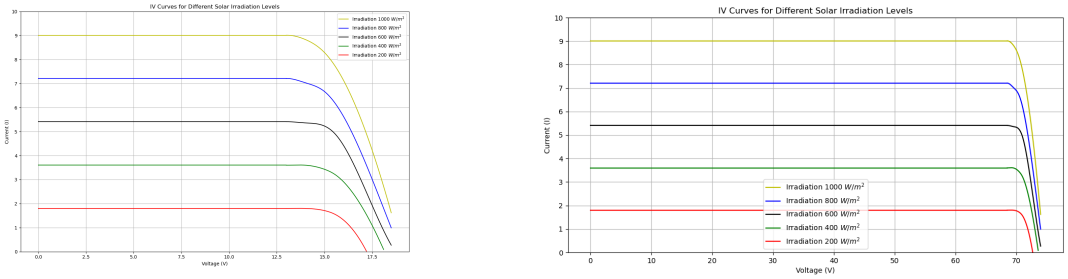
5.1 PV Panels

In table 5.1, the specifications for the PV panels that are being used for the experimental work is listed. The specifications are determined at standard conditions of 1000 W/m^2, AM1.5 and a 25° C cell temperature. The max power output of the PV panel at these conditions is 260 W with a power tolerance of $0 - +5$ W. Each PV panel has a height of 1 650 mm, width of 992 mm and a thickness of 40 mm. This gives them a total surface area of approximately 1.65 m^2. Of the total surface area, 1.46 m^2 is the polycrystalline solar cells. A full catalog of specifications for the Solartek PVP26030 PV panel is presented in appendix A.

Table 5.1: Specification sheet for the Solartek PVP26030 PV panel used in the testing of controllers.

Module Type:	***PVP26030***
$\mathbf{P}_{MPP}$ $[W]$**:**	260
$\mathbf{V}_{MPP}$ $[V]$**:**	30.92
$\mathbf{V}_{OC}$ $[V]$	38.00
$\mathbf{I}_{MPP}$ $[A]$**:**	8.43
$\mathbf{I}_{SC}$ $[A]$**:**	9.01
Conversion efficiency [%]:	15.98

As the datasheet for the PV panels in appendix A, showcases only a minor IV-curve, the IV-curve in figure 5.4 was created using the Python code in appendix F. It is written to be an approximation of the IV-curve in the datasheet; however, this code and plot can be used in review of the results from testing. Figure 5.4b, depicts the IV-curve for the configuration of four Solartek PVP26030 panels in series and is adapted using the Python program in appendix F, that was written for the single PV panel IV-curve in figure 5.4a.



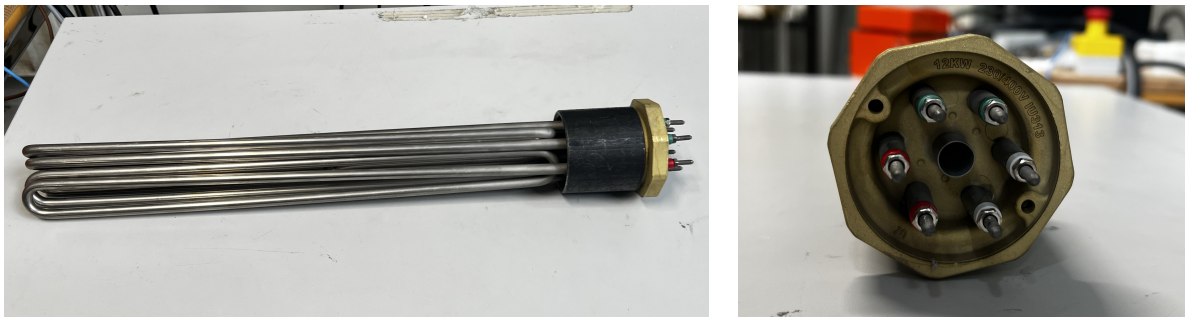
(a) IV-curve for single Solartek PVP26030.

(b) IV-curve for four series connected PV panels.

Figure 5.4: Approximate IV-curve for the Solartek PVP26030 PV panel, presented as a plot using the Python code in appendix F and information from the datasheets. [73, 74]

5.2 Heating Elements

The heating elements (HE) utilized in the experimental work of testing controllers from PV to HE was selected from available elements at the EPT laboratory. The specific model is from the manufacturer Backer, with the model being IU313. A 12 kW heating element with the specification sheet presented in appendix G. [75, 76]



(a) Side profile. (b) Connectors.

Figure 5.5: The Backer IU313 12 kW heating element. Three heating rods with 13.1 Ω resistance each. Specification sheet in appendix G.

The Backer IU313 heating element is made up of three heating rods and is usually meant for a three-phase alternating current with a star connection, as described in the appendix. The heating element has a surface power of 8.0 W/cm^2 and is rated for up to 9 bars of pressure and 120° C, as described in the specification sheet in appendix G. Each of the three heating rods of the element was measured to be at a resistance of 13.1 Ω. When utilizing these heating elements with a direct current, each heating rod can carry a current without the star connection. Parallel and series connections of the heating rods are then made possible and using equation 2.2 & 2.3, the desired resistances can be calculated.

5.3 Experimental Setup

The setup for the experimental work on controllers at the EPT laboratory is as follows and the schematic in figure 5.6 is a representation of it. The concept of the experimental work is using controllers directly between PV panels, harvesting solar energy, and heating elements which can be used to store that energy as heat. For the setup, PV power comes down from the rooftop mounted PV panels and can be connected further from a box, which eventually had an on/off breaker switch to secure safe and straightforward operation. From there, the PV power pass to the controller that is to be tested, where the MPPT function of the controller optimizes the power out of the controller. A junction box is in place after the controller to facilitate the switching between series and parallel connections of the heating elements.

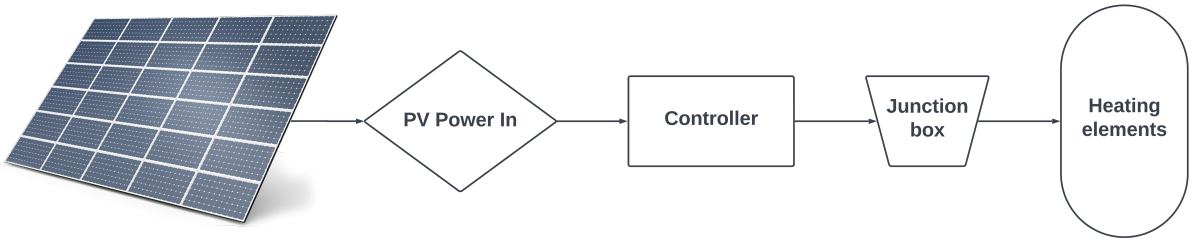


Figure 5.6: Schematic of the experimental setup.

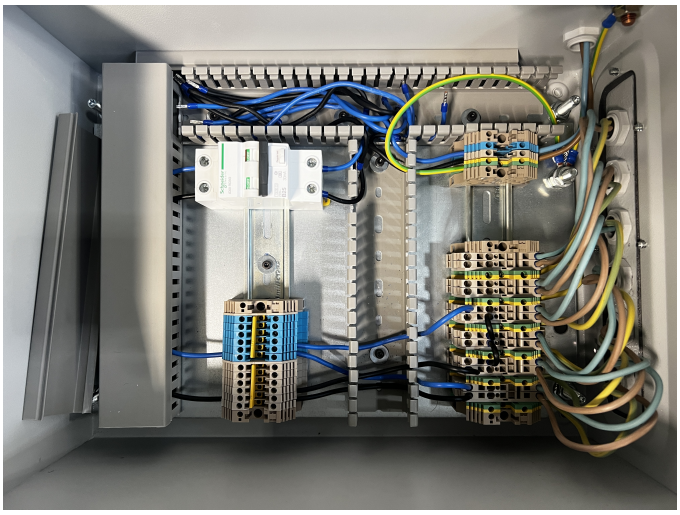
Table 5.2 presents a description of the physical parts of the experimental setup that is depicted in figure 5.7.

Table 5.2: An account of the experimental setup marked in the figure 5.7.

#	**Description**
1	Power in from PV array, with On/Off breaker switch
2	Geyserwise 72V controller
3	Odin controller
4	Junction box
5	Heating elements installed in metal cylinders and filled with water
6	Laptop running a LabVIEW program, to output solar irradiance in W/m^2

Figure 5.7: The experimental setup at the EPT laboratory, with numbering in reference to table 5.2.

Certain parts of the experimental setup, which are depicted in the figure 5.7, will be further expanded upon in the section. These include the junction box, heating elements installation and the two laptop programs used in the testing of the controllers.



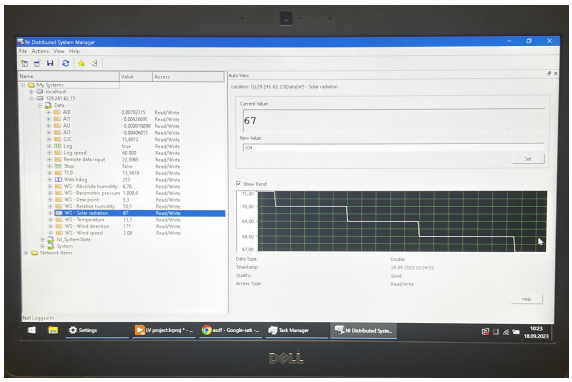
(a) The inside of the junction box.

(b) The heating element installation in metal cylinders.

Figure 5.8: A look at the inside of the junction box and its connection to the rods of the heating elements that are installed in metal cylinders and filled with water.

The junction box is a unit between the controller and the heating elements. Its main purpose is to simplify the process of switching between different resistances, and therefore expediting the testing of the controllers. As seen in figure 5.8a, power comes in from the controller, through a 25 A breaker switch to the blue/grey row in the middle-left bottom of the figure. The blue/grey row dictate the number of parallel connections going to the heating elements. There are in total six heating rods available, and so in the junction box there are connectors for each of these. These are the row of connectors at the bottom right of the figure 5.8a, which are connected to each heating rod through the black cables in figure 5.8b. The figure 5.8a represents a parallel circuit, where one line has a single resistance to it, while the other circuit of the parallel utilizes jumper wires to create a circuit with three resistances.

Figure 5.8b depicts the metal cylinders, which the heating elements are installed in, while the orange boxes are for housing the cables for the connection to the junction box. The metal cylinders are filled with water, rather than a type of oil due to ease of use and cleaning. Also, as the heating elements have an upper limit for temperature at 120° C if the water boils, it is a sign to let the heating elements cool before further testing. Each of the connections in the junction box for the different resistances used to test the controllers are shown in appendix J. Their individual resistances are calculated by using equation 2.2 & 2.3.



(a) LabVIEW program with readouts of the solar irradiance from the rooftop pyranometer, used for the 72V & 48V controller.

(b) LabVIEW program with readouts of solar irradiance, PWM out and temperature of the Odin controller. Power, voltage and current into the controller is also displayed in the program.

Figure 5.9: Laptop programs used in the testing of the 72V, 48V and Odin controller.

For testing of the Geyserwise 72V & 48V controllers, the program in figure 5.9a is used exclusively for readouts of the solar irradiance. This is due to the Geyserwise controllers having a display for readouts of other necessary data. However, for the testing of the Odin controller as seen in figure 5.9b, the program is made in tandem with the controller, as to produce readouts of all necessary data. Both programs have logging capabilities.

5.4 Arduino

The Arduino is a microcontroller meant for data logging of temperatures, along with current and voltage for power measurements. The main stack is composed of an Arduino Uno R3 main board, a SeeedStudio Solar Charger Shield, and an Adafruit Datalogging Shield equipped with SD card, Bi-Directional Level Shifter and Real-time clock. Finally, an Adafruit LCD is added for displaying data. The temperature data is measured by four K-type thermocouples connected to MAX31850 amplifier boards. The amplifier boards are in line on a stripboard, so that only one data line to the data logging shield is necessary. Power and ground lines are also needed from the stripboard to the shield.

A GY712 shunt for current measurements is installed in the system as well as a PV voltage regulator so the solar power can be used to run the system as an off-grid solution. The GY712 shunt is connected to the main Arduino stack by a VCC power line, a data line which goes to the analog input A2 of the data logging shield, and a ground line. The voltage regulator is needed to have the correct voltage of $4.4 - 6\ V$ into the solar input connector of the Solar Charger Shield. The shield is also equipped with a LiPo battery for energy storage and power supply when the solar power is not sufficient.

To enhance the Arduino system, a voltage divider circuit for voltage measurements is to be implemented, along with a 4G shield for data transmission over the internet. This is so that the logged data can be accessed without needing to connect to the Arduino.

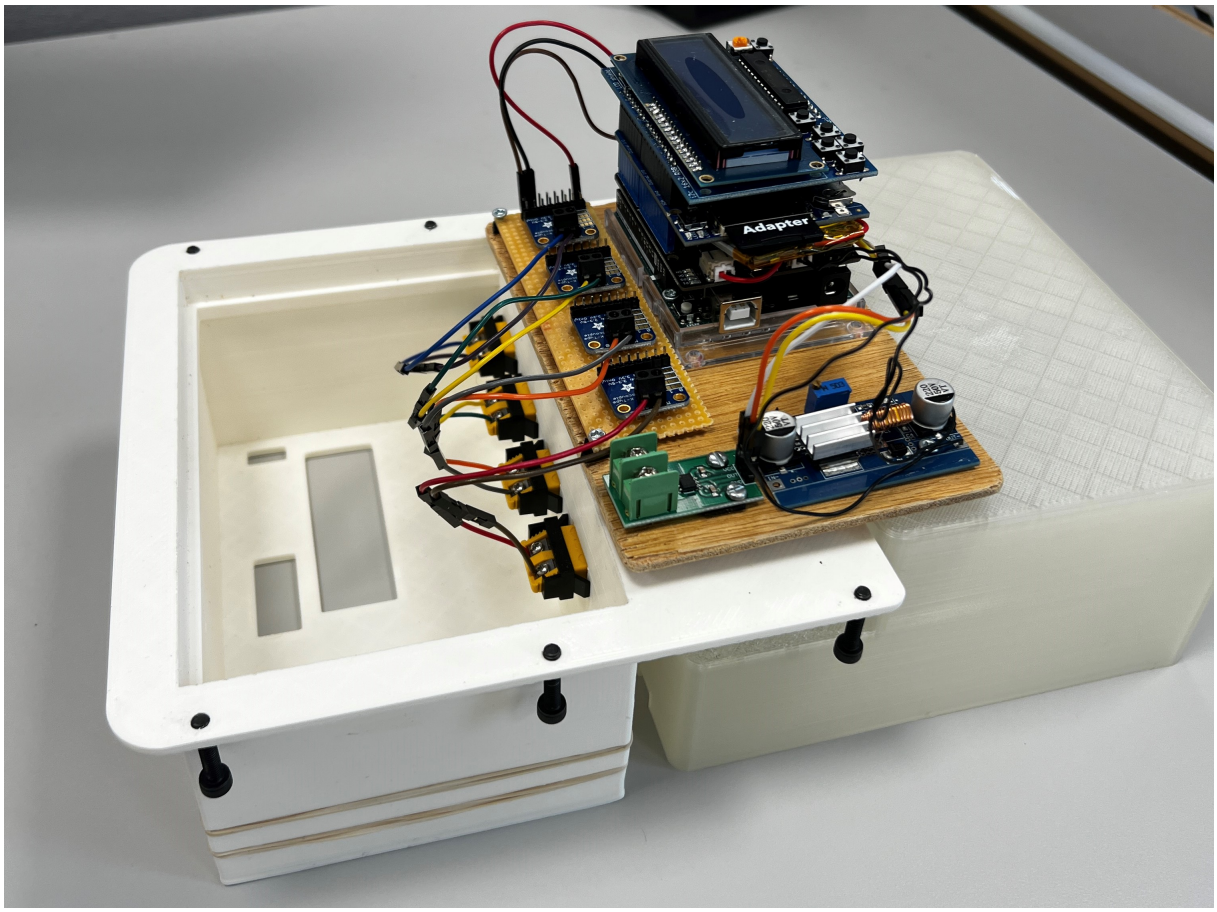


Figure 5.10: The Arduino developed during the project work, with its 3D-printed housing which was designed in ArchiCAD.

The VDC for this application is designed to measure voltage for one PV panel and send a maximum of 5 V to the Arduino. V_{in} should therefore be approximately 40 V. There are several viable solutions of this, based on the available resistors. In this instance the first resistor is a 1.2 $k\Omega$, while the second resistor is 150 Ω. Using equation 2.4, the calculation for the VDC results in $V_{in} = 45\,V$. In figure 5.11 the input voltage is to the left of the figure, while the right is voltage out to the Arduino. The green wire is positive and is connected to the analog input A1 on the data logging shield. If it is desirable to use another analog input, the program Arduino IDE in appendix K has to be adjusted.

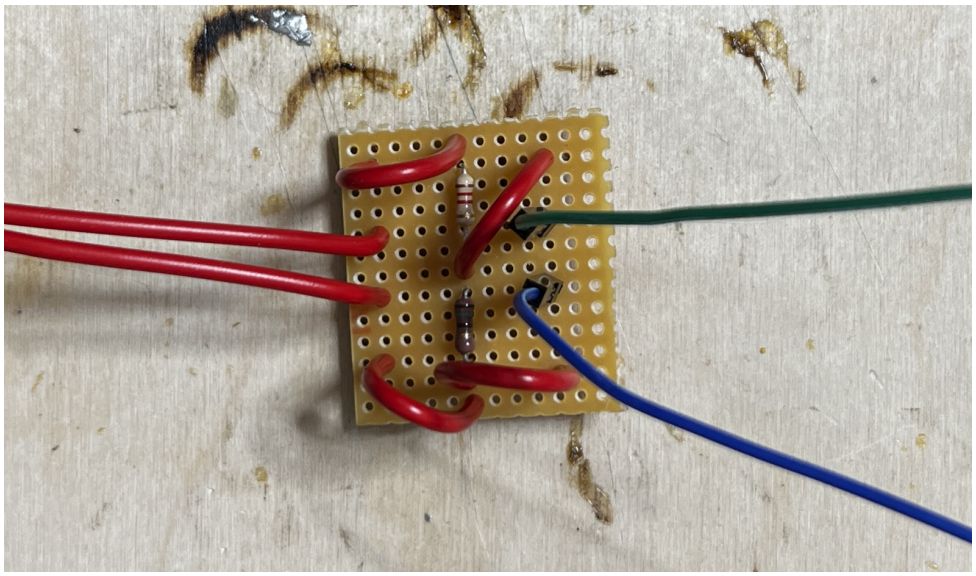


Figure 5.11: Voltage Divider Circuit constructed on a stripboard.

Figure 5.12 is a schematic diagram for the connections needed to operate the shunt and voltage divider circuit. In this instance, the three connectors out of the shunt: VCC, OUT and GND, go to 5V, A2 and Ground on the Adafruit Data Logging Shield of the Arduino. Positive V_{out} of VDC go to A1, while the negative is connected to Ground on the same Arduino shield. The Arduino is designed to be a standalone unit, where it will take power from the PV panel and store the energy in a LiPo battery. To do this, a PV Voltage Regulator is connected to the solar power input of the Solar Charger shield. The PV Voltage Regulator uses a buck converter to step down the voltage to 5 V at the outputs. This is necessary as the solar power input for the Arduino can be variable between 4.4 and 6 V.

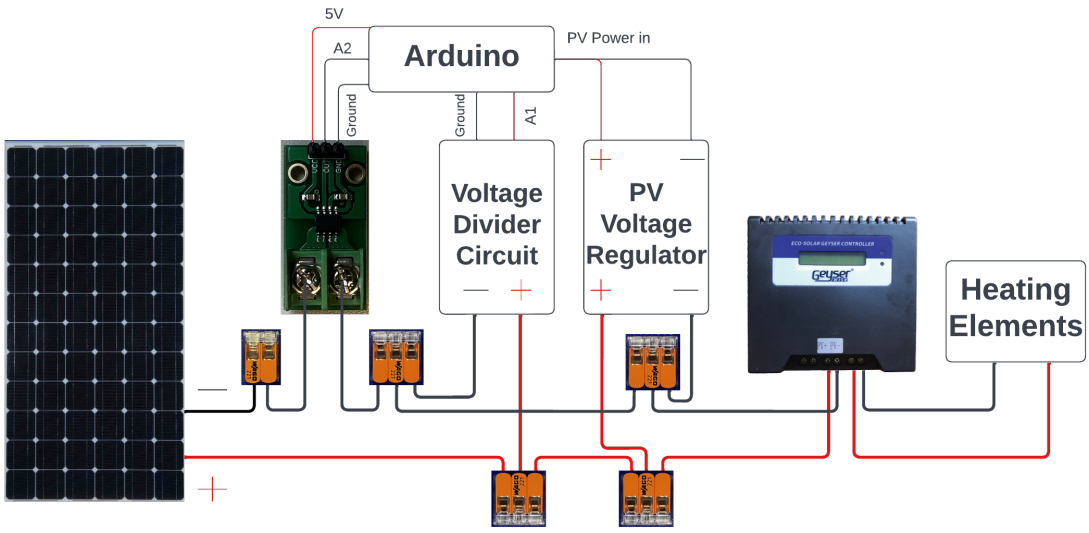


Figure 5.12: Shunt and Voltage Divider Circuit schematic diagram.

To aid the process of connecting the Arduino to the rest of the PV setup when the Arduino system is encapsulated in its housing, the table 5.3 gives a description for each of the connections that emerges from the Arduino housing. There are six connections in total, two for each of the shunt, VDC and PV voltage regulator.

Table 5.3: Description of connecting wires for the Arduino system.

Function	**Description**
Left Shunt	White cable with single black marking
Right Shunt	White cable with no markings
VDC +	Red cable with no markings
VDC -	Red cable with broad single black marking
PV +	Red cable with double black marking
PV -	Red cable with single black marking

5.4.1 Web-based Arduino Implementation

During the evaluation of practical solutions for the implementation of an internet connection to the Arduino, two shields were the main alternatives. A LILYGO ESP32 shield with a SIM7600 module and the TinySine SIM7600 shield. An evaluation of these and other solutions are in appendix L. As the TinySine shield is stackable and with a larger, presumably more powerful antenna, it is the module used for this implementation. The shield is presented in figure 5.13.

Figure 5.13: The TinySine SIM7600 stackable Arduino shield. [77]

From the datasheets of the TinySine SIM7600 shield for Arduino systems in appendix M, one is to assume that the data transfer between the SIM shield and the other shields of the Arduino stack is done through UART communication. This is due to the datasheet specifying lines marked RXD and TXD, which is the same as TX and RX of UART, explained in chapter 2.7. Also, deducted from the datasheet is that the analog A0 pinout is utilized by the status line between the SIM shield and Arduino Uno. Therefore, the current measurements input from the GY712 shunt was transferred from A0 to A2.

5.4.2 MQTT Software

To access the data sent by a web-based Arduino, an MQTT client software is needed if one is to access it via a laptop. One such software, recommended by Ubuntu [78], is MQTT Explorer. Figure 5.14 depicts an example of how the MQTT explorer can be utilized when wanting to log temperatures. There are also multiple applications for both iOS and Android that can be used as a client as well. [79]

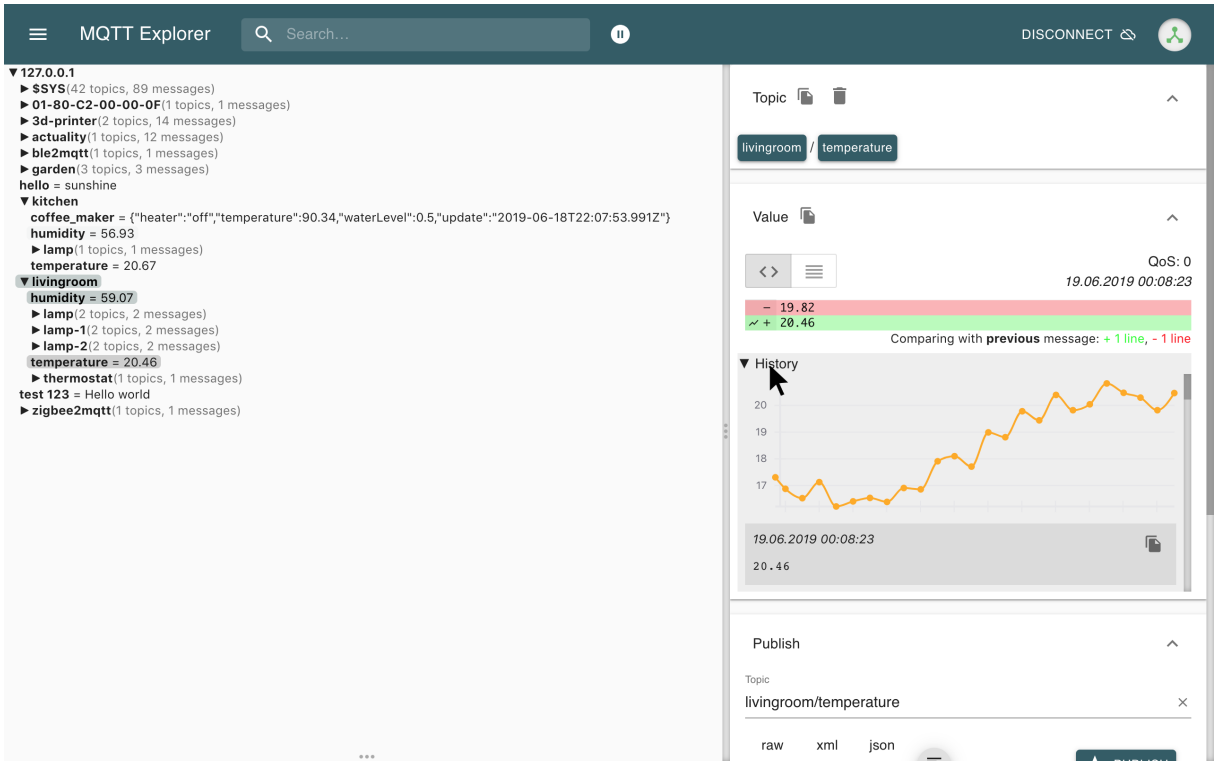


Figure 5.14: An example of temperature logging with MQTT Explorer. [79]

6 Experimental Work

The experimental work consists of two parts. The first regarding testing of the available controllers at the EPT laboratory of NTNU Gløshaugen, Trondheim. The latter being the continued work on the Arduino system and developing a web-based solution for it.

Table 6.1 is an overview of the three available controllers for testing, with which configuration of PV panels they were performed with and at which resistances. Mostly the resistances used were equal, however, during the testing a controller could exhibit characteristics that could prove to be better at other resistances and as such there are a few deviations.

The further development of the Arduino system of the project work to a web-based solution, by implementing a SIM shield with wireless capabilities, is expanded upon after the sections on testing of the controllers.

Table 6.1: The resistances tested for the controllers and their PV configuration.

Controller	72 V	Odin	48 V
PV Configuration	4 Series	4 Series	2 Series x 2 Parallel
Resistances [Ω]	2.62	2.18	2.18
	3.28	2.62	2.62
	4.37	3.28	3.28
	6.55	4.37	4.37
	8.73	6.55	6.55
	9.83	8.73	8.73
	13.1	13.1	13.1
	26.2	26.2	26.2
		39.3	

6.1 Methodology

The testing of the controllers performed at VATL Gløshaugen, aimed to learn their behavior at different sun conditions and resistances. To do so, experiments were conducted exploiting solar energy from PV panels, through the controller to a pair of heating elements which can have different resistances based on how they are circuited. The controllers will regulate that power to the best of their abilities depending on the connected heating rods. The experiments were mainly performed between 10:00 and 14:00 to minimize the performance reduction due to angle of incidence on the PV panel compared to the solar irradiance measurements.

The Geyserwise 72V and 48V controller share the same methodology of testing as they differ only in load voltage. From table 6.1, the setup for the controllers is presented and regarding the Geyserwise models, the 72V was tested with four PV panels in series and a maximum voltage of 152 V_{OC}. This exceeds the input open-circuit voltage of

the controller of 138 V_{OC}. However, as the sun conditions of the season in which the testing was performed were dwindling, and not at a maximum of the potential in the area, the belief was that it would not cause an issue. For the 48V Geyserwise controller, a parallel connection with two PV panels in each parallel was utilized. The data collection for the two controllers were done with solar irradiance from the LabVIEW program seen in figure 5.9a. Figure 6.1 shows the 72V controller display and the information it presents, which were noted when the performance figures had stabilized along with the solar irradiance. Testing for each resistance listed in the table was performed as such. When stable conditions were met and required data collection was done, the resistance could be changed by switching cable connections in the junction box.

Figure 6.1: Readouts of the Geyserwise 72V controller. The display presents PV power in, with power, voltage and current out of the controller.

The testing of the Odin controller differs from that of the two Geyserwise controllers. From the introduction of the chapter, table J gives an explanation on the configuration and resistances of the Odin controller. Similar to the Geyserwise 72V, the Odin controller was tested with a setup of four PV panels in series. This was done to have an equal power input to the Geyserwise 72V controller, such that their performance could be more easily compared. The data collection however, were noted from the designated laptop program for the controller, as seen in figure 5.9b.

After the main testing of the 72V controller, due to some uncertainty regarding the accuracy of the data presented on the Geyserwise display, a pair of multimeters were connected on the output to check the reliability of the display. This was also done for the Odin controller, which is expanded upon in chapter 6.3.

6.2 Results and Discussion – Geyserwise 72V

Testing of the Geyserwise 72V controller was performed as follows from chapter 6.1, with solar energy coming from PV panels to the controller, which forwards it to the junction box and desired resistance. Figure 6.2 is a visual representation of the results from testing the controller with different resistances at a range of solar irradiance. The entire data of the testing is tabulated in appendix N, with the figure depicting the power out of the controller data.

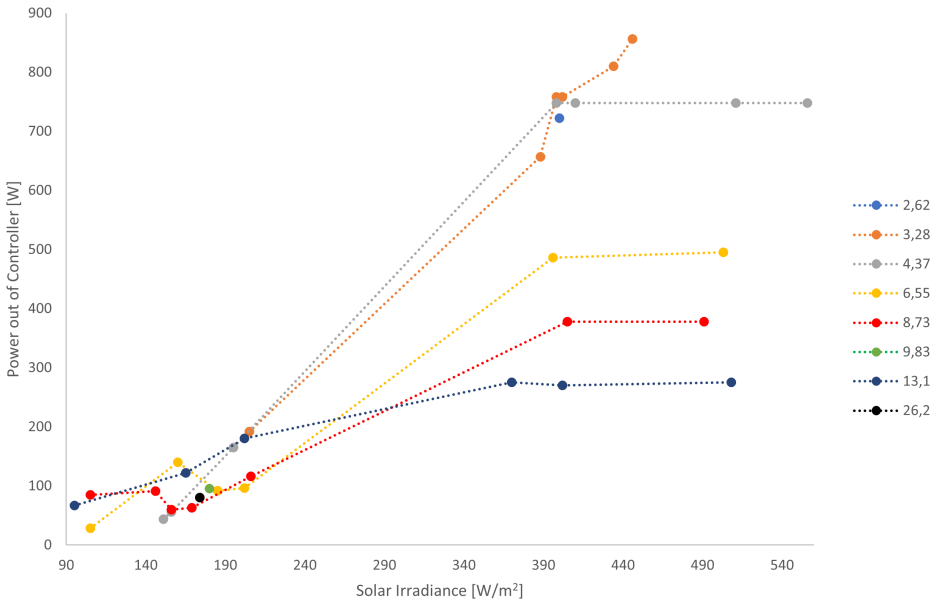


Figure 6.2: Geyserwise 72V diagram, plotted from data in table N.1 in the appendices.

In table I.1 of appendix I, an evaluation is formed of the resistance that is optimal for the 72V controller at the standard conditions described in chapter 2.1. For the four in series PV panels at 1 000 W/m^2 it presents a resistance at the load voltage of 5.0 Ω. The figure 6.2 presents the power out of the controller with different resistances, ranging from 2.62 to 26.2 Ω. At lower solar irradiance, the controller delivers little power and the differences in the resistance used is negligible. However, when the irradiance increases the differences is revealed. Especially for the higher levels of resistance, the power out seem to stagnate and is limited compared to the lower resistances. Even though the optimum is 5 Ω from calculations, the 4.37 Ω resistance of the heating elements stagnates before the 3.28 Ω. The peak power delivered to the heating elements is with the 3.28 Ω resistance, which yields 856 W at a solar irradiance of 446 W/m^2.

After the main testing of the controller, some concern on whether the display of the Geyserwise controller was correct arose. To investigate and solve this concern, a test where a pair of multimeters were circuited at the output of the controller, to measure voltage and current. As the multimeters had a circuit breaker at 10 A, and the controller often boosting the current above this level, the testing was done quickly and not as sufficient as it perhaps should be. However, from the brief testing it seemed as if the display

of the controller was correct on both voltage and current out. No apparent considerable inaccuracies at least, though at low currents the display presented values which jumped by 0.5 A, making it more inaccurate at those lower current levels of up to 5 A approximately.

A possible issue mentioned in chapter 6.1 on the maximum open-circuit voltage from four PV panels exceeding the input specifications of the controller, arose as a problem during the experiments of the 72V controller. At a point during the testing, sun conditions were better than the conventional levels and as such the voltage from the PV panels were too high for the 138 V_{OC} of the controller. When this occurred, the controller was unable to start the process of regulating voltage and current to reach the maximum power point. Further testing should therefore probably be done with either three PV panels in series, or a circuit of two panels in each parallel ($2\,x\,2$) to mitigate this issue. A $2\,x\,2$ circuit for the PV panels would suggest a resistance at load voltage of 5.0 Ω according to appendix I. It did not however, cause any further issues at later testing, as voltage levels did not reach above the controller specifications. It was instead at a stable starting point for the regulating at approximately 136 – 138 V_{OC} and so the controller could perform as intended. This drop in maximum open-circuit voltage from the PV panels of approximately 15 V, could be due to partial shading of the panels. The controller was occasionally able to start its process even when the input voltage was slightly above 138 V_{OC}.

6.3 Results and Discussion – Odin

Testing of the Odin controller was performed as follows from chapter 6.1, with solar energy from PV panels to the controller, which forwards it to the junction box and desired resistance. Figure 6.3 is a visual representation of the results from testing the controller with different resistances at a range of solar irradiance. The entire data of the testing is tabulated in appendix O, with the figure depicting the power out of the controller data.

Figure 6.3 presents power input to the controller from the LabVIEW program readings in table O.1. The measurements range in solar irradiance from 125 W/m^2 to 391 W/m^2. Unlike the Geyserwise 72V controller, none of the different resistances seem to be stagnating, but the trends of their performances are distinct. Both 8.73 & 13.1 Ω perform well and show peak power of 802.7 & 837.1 W respectively.

The table O.1 contain the data noted for the testing of the Odin controller from the LabVIEW program. A point to note with the table is that the program is calibrated for the use of six PV panels in series. However, the measurements presented in the program is taken from a single PV panel and multiplied. As such the values are higher than they should be and have to be adjusted by 2/3 to account for the testing with four PV panels. The table presents these adjusted values for power and voltage on the input of the controller.

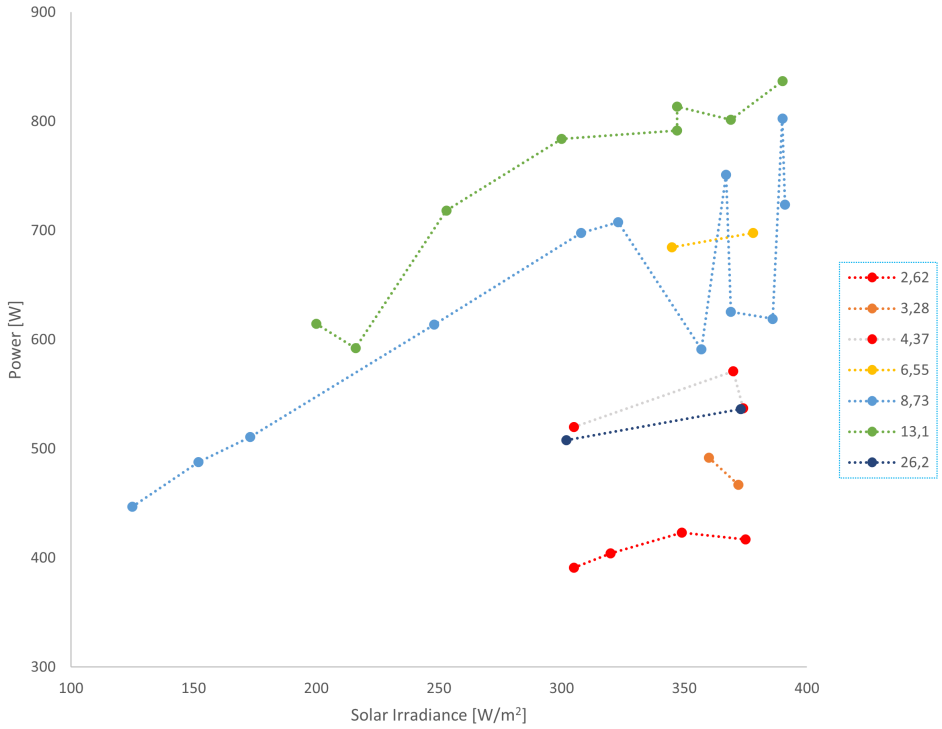


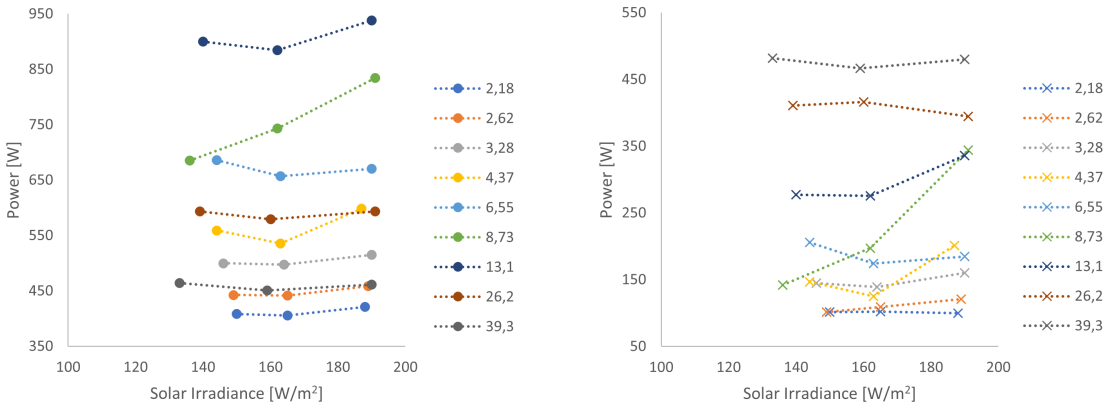
Figure 6.3: Odin graph of the power performance with different resistances.

As with the Geyserwise 72V, the Odin controller was tested with a pair of multimeters at the output, after the main experiments were concluded. Table O.2 in the appendices is the entirety of the data recorded from these tests. As with the first experiments, these values are also adjusted by 2/3 to account for the use of four panels instead of six. While multimeters are not necessarily the way to measure a PWM signal, it was fairly accurate with the 72V controller and so the assumption is that the results are accurate for the Odin controller as well.

Figure 6.4 presents the power on the input as shown in the LabVIEW program, as well as that of the measurements taken from the multimeters for comparison. While the 13.1 Ω resistance appear to show the highest levels of power from that LabVIEW program, the multimeter data indicate that 39.3 Ω is allowing more power through to the heating elements. From the tabulated data in the appendices, while the PWM is at approximately 50 % for 13.1 Ω, it is at 100 % for the 39.3 Ω. While the PWM was at a 100 %, voltage in the LabVIEW program is the same as that measured on the output with the multimeters. While looking at the lower resistance data, it appears that the controller is not boosting the current to the same levels of the Geyserwise controller, while regulating the voltage and current, and as such is losing power.

A cause for concern in these results with multimeters is the lack of change in power when the solar irradiance increases. Only the 8.73 Ω resistance shows any form of linear increase in power within the range of irradiance for the testing. As this is an issue, the credibility of these later experiments is lessened and could be a reason to discount them, at least until further testing could be performed with the correct measuring devices.

From the LabVIEW program, peak power input is with 13.1 Ω resistance. The power value is 938.27 W at 190 W/m^2, which seems to be quite large. However, the multimeter data gives a peak power output of 479.91 W at 190.0 W/m^2 for the 39.3 Ω. This result is somewhat more comparable to the results of the Geyserwise data presented in chapter 6.2. Another issue with this data though, is that it is not comparable to the earlier experiments with the Odin controller as seen in figure 6.3. While figure 6.3 & 6.4a are both taken from the LabVIEW program, the latter shows larger power at lower solar irradiance. An answer to this might be that the later testing had a lower sun path, resulting in more direct irradiance to the vertically mounted PV panels. Also, temperatures were lower which yields better efficiency of the panels.



(a) Power from LabVIEW program.

(b) Power from multimeter measurements.

Figure 6.4: Testing of the Odin controller with a set of multimeters to measure current and voltage output along with measurements from the LabVIEW program of the controller.

An issue described by the manual in appendix C for the Odin controller, is that the temperature can get high, in which case the box of the controller should be opened. This is described to be between 40 & 50° C. While conducting experiments with the controller, temperature measurements was also recorded. The peak temperature registered for each resistance tested is presented in figure 6.5. However, the figure of peak temperatures does not reveal the rate of temperature change for the different tests. For lower resistances, the temperature was increasing at a significantly faster rate than for higher resistances, and would quickly pass 40° C. Higher resistances would be more stable at the mid 30's. The reason for the lower resistances not being higher, is that the experiments are not performed as a lengthy and continuous operation. If an experiment were performed where operation was longer, the lower resistances would probably surpass 50° C and be well above the temperatures of those that the higher resistances would have.

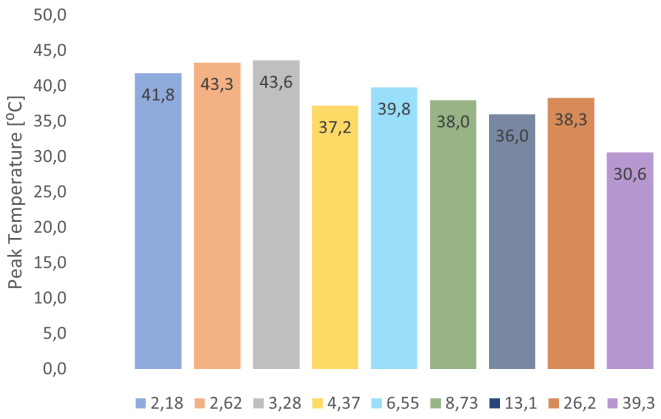


Figure 6.5: Peak temperatures of the Odin controller while testing separate resistances.

6.4 Results and Discussion – Geyserwise 48V

Testing of the Geyserwise 48V controller was performed as follows from chapter 6.1, with solar energy from PV panels to the controller, which forwards it to the junction box and desired resistance. Figure 6.6 is a visual representation of the results from testing the controller with different resistances at a range of solar irradiance. The entire data of the testing is tabulated in appendix P, with the figure depicting the power out of the controller data.

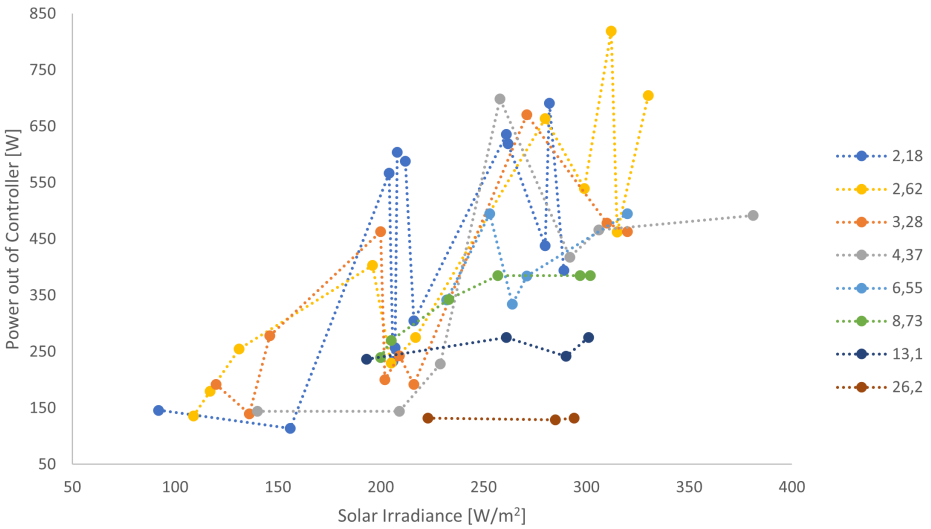
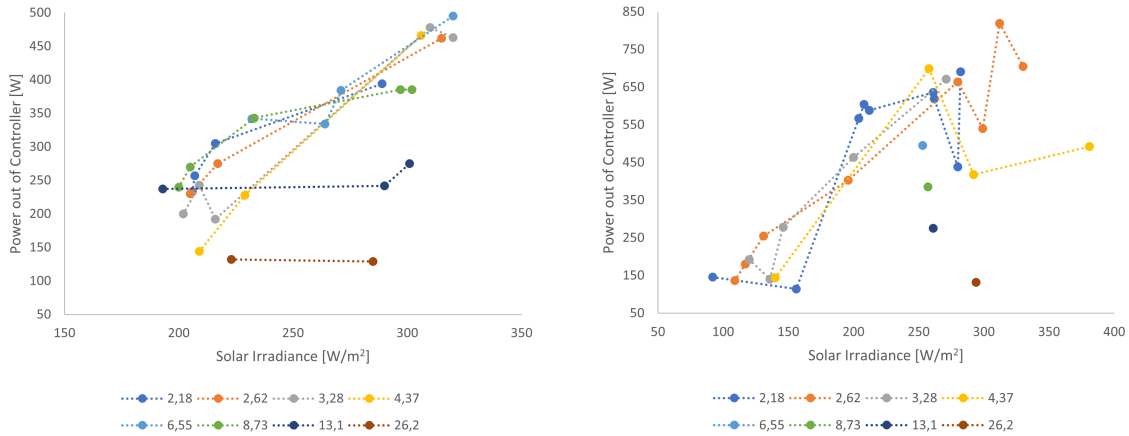


Figure 6.6: Geyserwise 48V controller power output diagram of experiments with various resistances.

In table I.2 of appendix I, an evaluation is formed of the resistance that is optimal for the 48V controller at the standard conditions described in chapter 2.1. For the circuit with two series connected PV panels in each parallel at 1 000 W/m^2, it presents a resistance at the load voltage of 2.2 Ω. The figure 6.6 presents power input at solar irradiance ranging from 92 to 381 W/m^2 for the resistances. Those tested for the 48V controller range from 2.18 to 26.2 Ω. Results from the testing as shown in the figure, are quite varying within each resistance and when comparing them to each other. Lower solar irradiance offers little power output, while the performance at the higher levels of the experiments are better. The peak power achieved for the Geyserwise 48V is 819 W, at an irradiance of 312 W/m^2. This is achieved with the 2.62 Ω resistance. This result correlates well with the target ohms of 2.2 Ω at load voltage from the calculations in the appendix. Unlike the 72V and Odin controller, no test with multimeters were performed with the Geyserwise 48V. It seemed as there were no necessity of undergoing such a test, as it had already been done with the 72V controller to check if the display was correct. As the performance of the 48V is comparable to that of the 72V, it is assumed that the display of this controller is also presenting the data as it should to the display.

The irregularities in the results from the main experiments with the 48V controller are a concern. This behavior is probably due to the separate occasions of testing having quite unstable conditions, rendering the results somewhat uncertain. However, as seen in figure 6.7, if separated into dates that the tests were performed for this controller the results become somewhat clearer.



(a) Day 1.

(b) Day 2.

Figure 6.7: The results of testing the Geyserwise 48V controller separated into days of testing.

Figure 6.7a show a good linear performance for several of the resistances tested with this controller. Only the two highest resistances appear to have stagnating issues. However, there is minor difference in the power output between the resistances below 8.73 Ω. Peak power for this occasion is 495 W at an irradiance of 320 W/m^2. This is achieved with the 6.55 Ω resistance of the heating elements.

Figure 6.7b however, contain more of the irregularities in results from testing, but it has significantly higher power outputs. As the PV panels and the pyranometer are a distance from each other, unstable conditions serve as a factor of uncertainty which can influence the results. Peak power in this scenario is as presented for the main results: 819 W at 312 W/m^2 for the 2.62 Ω heating element circuit. As the performance of the second experiments are 65 % higher at equivalent solar irradiance, it suggests that the first testing might have had factors influencing the results and thereby rendering them negligible.

6.5 Comparison of Controllers

The Geyserwise 72V and Odin controller, were given the same PV configuration to ease the comparison between their performance. The same amount of PV panels and similar levels of solar irradiance should mean comparable results. However, their design difference appears to cause quite different performance results for the resistances that were tested in their experiments. In addition, the PV configuration is not optimal for either the Geyserwise or Odin controller. Geyserwise 72V would probably benefit from a $2\,x\,2$ solution to avoid the high input voltage issues, while the Odin controller is capable of utilizing all six panels in series, to boost input power and thus output power.

One major difference in their performance is that the Geyserwise controller has better results for lower resistances, around its calculated target ohms. Odin performance results show that the lower resistances deliver little power and that a higher resistance circuited heating element is better. The 72V controller has slightly higher peak power, which amounts to an increase of approximately 2.3 %, when compared to the Odin unit from the initial tests without multimeters. This peak power discrepancy is with 3.27 & 13.1 Ω respectively, even though they have equal PV configuration and conditions.

The differences between the Geyserwise 72V and 48V controllers are minor. Initial understanding was that the load voltage for the Geyserwise controllers were different, at 48 & 72 volts. The bar code numbers are not the same between them and they have the same open-circuit input voltage specifications. This disparity in load voltage, produce target ohms from calculations which differ from each other. As a result, the resistance which is optimal for performance changes from 3.27 Ω for the 72V controller to 2.62 Ω for the 48V version. In addition, the controllers present similar voltage levels at the output side, while current measurements have a higher peak on the 48V.

A condition of these comparisons between the three controllers that were included in the experiments, is that the results from testing can vary with higher solar irradiance. At a maximum value during all testing of 556 W/m^2, it is a possibility of different results when irradiance exceeds this amount. For all three controllers, the value of solar irradiance rarely exceeded 400 W/m^2, and the peak amount is only applicable for the Geyserwise 72V controller.

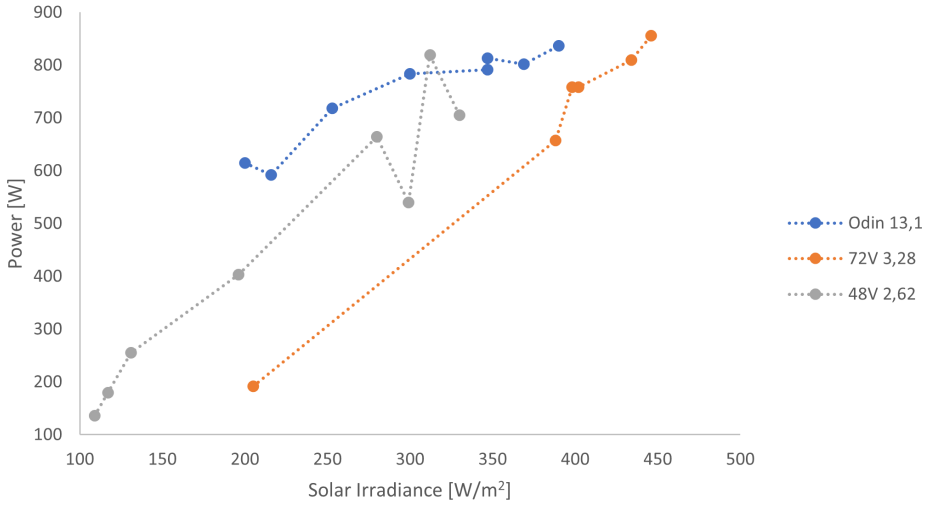


Figure 6.8: Comparison of the resistances with peak power output for each controller.

The figure 6.8, adds the three resistances which had peak power output for each controller together for comparison. From the Geyserwise 72V controller, the peak result was from 3.28 Ω resistance, while 4.37 Ω was close but seemed to stagnate at higher solar irradiance. The 48V controller shows promising results compared to the 72V, providing higher power outputs at lower irradiances. Testing both Geyserwise controllers with a $2\,x\,2$ PV configuration should be done to confirm if the 48V has better performance. Results from the Odin controller is equivalent to the Geyserwise controllers for a 13.1 Ω resistance. However, these results are somewhat uncertain as there is a large disparity between the results presented in the LabVIEW program, and those from the multimeters.

6.6 Sources of Error

Although the experimental work on the controllers was performed at equivalent conditions, some factors affect the certainty of the results and therefore might lead to faulty comparisons or deductions. One of these main factors is the relation of measuring solar irradiance to the power coming from the PV panels. As the pyranometer is at a distance of approximately 34 meters from the PV panels, one or the other can be affected by partial shading, which would result in faulty cataloging of the data. To mitigate this issue, the tests were performed such that generally stable conditions were met before being registered as a data point.

Another factor when regarding the PV panels and pyranometer relationship, is that the performance of the PV panels varies throughout the day, due to the angle of incidence of solar irradiance on the panels changing because of the sun path. However, as the pyranometer is a hemispherical unit which takes in irradiance from every angle on the horizontal plane (GHI), the value it measures in W/m^2 will not change as drastically as the performance of the PV panels are affected by the change in incidence angle. An alternative to this measuring of GHI, a possibility could be the use of a fixed pyrheliometer offering

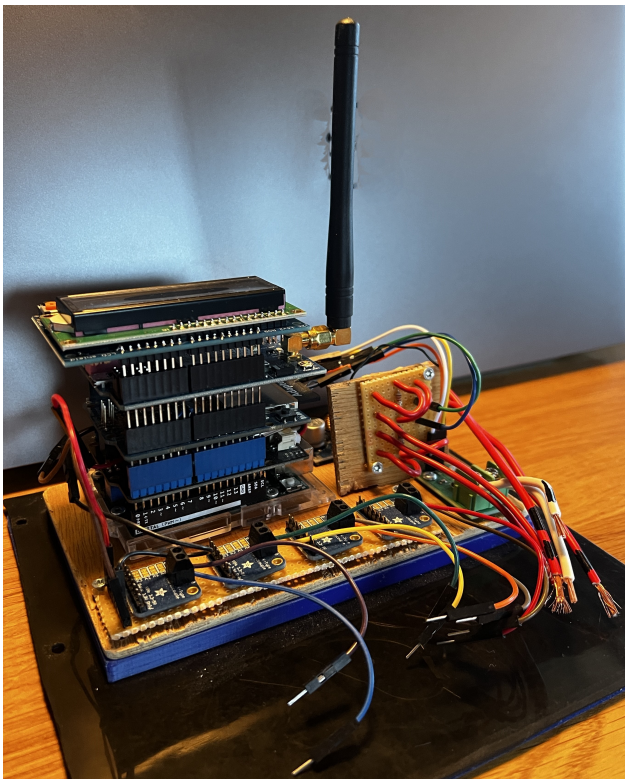
DNI measurements instead. However, this will generally take in a smaller angle of sunlight than what can be used by the PV panels. Another option is to use the pyranometer in a vertical plane as the panels are mounted, and thus measure GTI instead. Neither solution is optimal but an interesting aspect to experiment with. In addition, the mounting of the solar irradiance measuring device should be closer to the PV panels, to limit shading from affecting the results and therefore increasing their certainty.

The Odin controller has showed some results that are ambiguous, which leads to some uncertainty on whether it is operating as intended. The probable error with this controller is that it is set up for a six panel series PV configuration in the program it is running. As such, the results when using four panels instead will not be correct and in this instance, the results were adjusted by 2/3 to correct the results. However, there may be other factors in the program which this correction does not accommodate for, which would lead to the results being inaccurate and flawed.

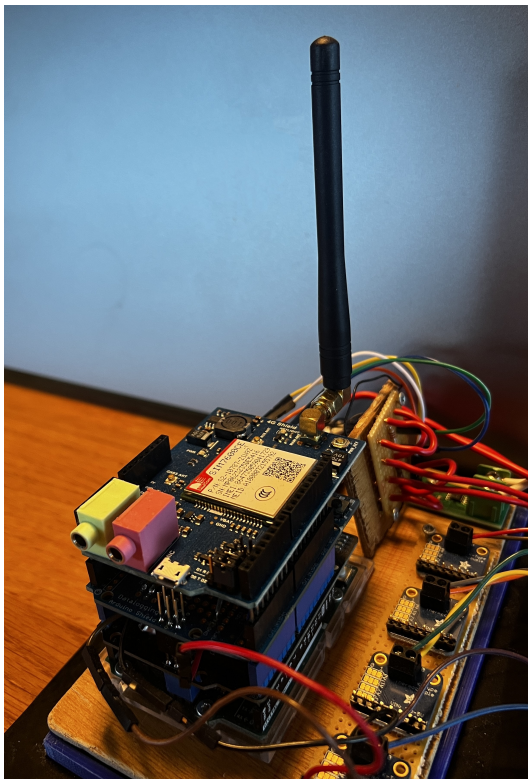
As in all experimental testing, human error can be a potential source of error, especially in instances where manual measurements are performed. This human error can lead to deviations in results that negatively impact the credibility and certainty of the work. All experimental work seeks to minimize or completely eradicate this human error factor; however, it is often unattainable to accomplish and will generally affect the work in one way or another.

6.7 Results and Discussion – Web-based Arduino

To create a web-based solution for the Arduino system that was developed during the project work, a SIM shield has been researched, bought and installed in the Arduino main stack of shields. Along with a new LCD shield and the TinySine SIM7600 module, the full stack of the system is seen in the figure 6.9a. Quite a bit of height is added by this new SIM shield, which could be reduced by removing the microphone and audio jack of the TinySine module. The microphone and audio jack are not necessary for this project and are seen as the green and pink units on the SIM shield in figure 6.9b. To fit with this new height in the housing for the system, which was developed during the project work, it is necessary to file down the ridge on which the wooden base plate is seated on. In addition, the antenna is an issue with the limited space within the housing and either a larger housing would need to be created to accommodate it, replacing it with a wire antenna or creating an opening for it in the existing housing.



(a) Both SIM7600 and LCD shield on the Arduino stack.



(b) Only the SIM7600 shield on the Arduino stack.

Figure 6.9: The Arduino with the TinySine SIM7600 and new LCD shield.

With the SIM shield implemented in the Arduino stack and a SIM card with a data plan provider obtained, the process of adapting the original Arduino IDE program from the project work seen in appendix K, into one that will operate with the new addition can begin. To achieve the function of the TinySine SIM7600 unit as a solution for a web-based Arduino system, there has been several renditions and attempts at a working program. One of which used a set of codes to connect the SIM shield with the MQTT broker. This

initial program, presented in appendix Q, did not function as was desired and further research suggested that the use of AT commands were the correct way of achieving the connection between the SIM shield and the MQTT broker.

The attempt of using AT commands with the original Arduino IDE program is presented in its entirety in appendix R. This undertaking of connecting the SIM shield with the MQTT Explorer has unfortunately, also resulted in a failure of the desired function. However, each section of the program is presented in the following figures, so that it is a possibility of further developing the program such that it operates as intended.

Figure 6.10 presents the additions to the definitions of the original program. Added to the initial set of libraries needed to run the system, *PubSubClient* is necessary when wanting to use MQTT. *SoftwareSerial* defines the RX and TX connections of the UART communication that the SIM shield utilizes. On the Arduino Uno R3 these pinouts are Digital 0 and Digital 1, respectively. The MQTT broker details relate to the settings made in MQTT Explorer and the data plan provider's APN. Further clarification on the MQTT broker details follows with figure 6.13. The APN used in this instance is *telenor.m2m*, which is a Machine-to-Machine subscription.

```
#include <PubSubClient.h>

SoftwareSerial sim7600(0, 1); // Software Serial pins for SIM7600 (RX, TX)

//MQTT Broker Details
const char* apn = "telenor.m2m"; // APN from cellular provider
const char* mqtt_server = "mqtt.mosquitto.org";
const char* mqtt_username = " ";
const char* mqtt_password = " ";
const char* mqtt_topic = "ArduinoTest/Martin/Data";
```

Figure 6.10: Introductory definitions for the TinySine SIM7600 and MQTT broker in the Arduino IDE program with AT commands.

In figure 5.3, the AT commands that are used in the last version of the Arduino IDE program are presented. It is set in the setup section of the program, as it tries to initialize the SIM shield and connect it to the MQTT broker. One item to note is that in the settings of the broker, a username and password can be used but it is not in this instance. Therefore, they are blank in the code. The APN details are necessary in this part of the program as well. In addition, the port number used in the settings of MQTT Explorer is required.

```
void setup(){ // This only runs one time after startup.

    sim7600.begin(9600);

  delay(2000);

  Serial.println("Initializing SIM7600...");

  // Check if the module is responsive
  if (!sendATCommand("AT")) {
    Serial.println("Error communicating with SIM7600");
    while (1);
  }

  // Set APN for your cellular network provider
  if (!sendATCommand("AT+CGDCONT=1,\"IP\",\"telenor.m2m\"")) {
    Serial.println("Error setting APN");
    while (1);
  }

  // Connect to GPRS
  if (!sendATCommand("AT+CGATT=1")) {
    Serial.println("Error connecting to GPRS");
    while (1);
  }

  // Start TCP connection with MQTT broker
  if (!sendATCommand("AT+CSOCKSETPN=1,\"telenor.m2m\"")) {
    Serial.println("Error setting PDP context");
    while (1);
  }

  if (!sendATCommand("AT+CSOCKAUTH=0,1,\" \",\"\"")) {  // Username and password
    Serial.println("Error setting MQTT username and password");
    while (1);
  }

  // MQTT Broker Host and Port
  if (!sendATCommand("AT+CSOCKCONN=0,\"TCP\",\"test.mosquitto.org\",1883")) {
    Serial.println("Error connecting to MQTT broker");
    while (1);
  }

  Serial.println("Connected to MQTT broker");

  delay(1000);
}
```

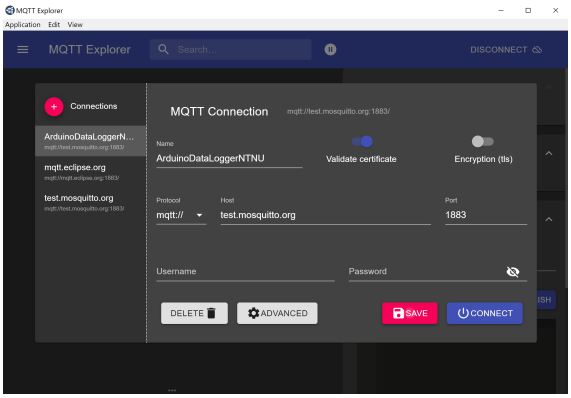
Figure 6.11: The section of Arduino IDE program with AT commands to connect the TinySine SIM7600 to the MQTT broker.

The code in figure 6.12 is an attempt to publish the recorded data from the temperature sensors, the current measurements from the shunt and voltage measurements from the voltage divider circuit. However, since the initializing of the SIM shield and its connection to MQTT Explorer was not achieved, this section of the program has not been able to be assessed and further developed. As such, it is unknown if the code is sufficient to publish the data to the broker or if there are additional steps required. Ideally, the program should read the data from the SD card with a set time interval, e.g., 60 minutes, to reduce the amount of energy that is used by the addition of the SIM7600 module. This is due to the limited energy in the LiPo battery. To do so would demand further code, however it has not been prioritized as establishing the connection between the Arduino and the MQTT broker took precedence.

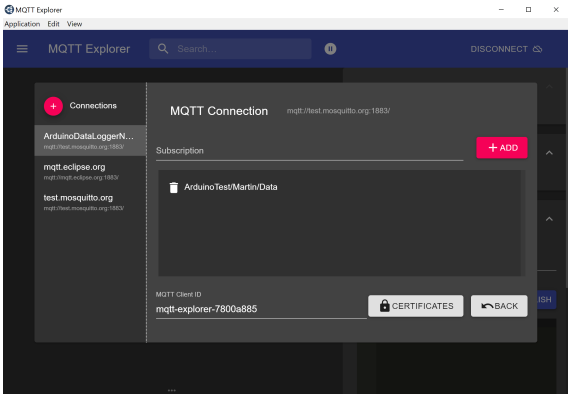
```
publishDataToMQTT(dateString);
delay(60000); //  Sixty second delay, to not flood the SD card with readings.

void publishDataToMQTT(String dataLine) {
  if (mqtt.connected()) {
    mqtt.publish(ArduinoTest/Martin/Data, dataLine.c_str());
  }
}
```

Figure 6.12: The publish section in the void loop of the Arduino IDE program.

Finally, the settings for MQTT Explorer are presented in figure 6.13. The main settings, which include the protocol, host and port are displayed in figure 6.13a. Figure 6.13b is the display of advanced settings, which are primarily the establishing of topics. The topics are where each set of data can be published, and therefore it is present in the code of figure 6.12 as well. The setting of a username and password can be done in the main settings. However, it is not necessary and was therefore not set in this instance.



(a) Main settings of the MQTT broker with port, host and protocol set.



(b) Advanced settings showing the topic. More topics can be added if it is desired.

Figure 6.13: Settings of the MQTT Explorer, which are necessary for the Arduino IDE program.

Multiple renditions of program in Arduino IDE have been evaluated, but none has performed as desired. There can be several reasons for this, the main being that there is a part of code which is not as it should be. Another cause for problems could be due to a conflict of pinouts between the shields that are used in the Arduino stack. However, the different schematics of the shields show that the LCD uses the pinouts A4 (Analog 4) and A5. The TinySine SIM7600 uses A0, while the VDC and shunt utilize A1 and A2 respectively. The SIM shield also appears to be the lone user of UART, instead of SPI and I2C, and therefore, should not be a cause of issue.

Whether this solution of using an MQTT broker to create a web-based solution for the Arduino system, is somewhat uncertain as the work has not resulted in a viable solution. However, researching the possible systems for solving this issue has led to this, probably being the best way of doing so. Finding projects with a more identical set up of parts has proven difficult and so, the programming has been a result of researching several projects where there might have been similarities and with viable solutions.

7 Conclusion

This master thesis has sections regarding an Arduino system, which is a continuation of previous project work and how this system can be developed into a web-based system. In addition, the thesis seeks to perform experimental testing of three solar charge controllers that are available at the EPT laboratory of NTNU Gløshaugen, Trondheim.

The three solar charge controllers, which were part of the experimental work, were tested under similar conditions to evaluate their performance. The two Geyserwise controllers performed well, except at lower solar irradiances. The 72V unit had peak power outputs above 800 W at solar irradiances above 400 W/m^2. 3.28 & 4.37 Ω were the resistances that performed best with this controller, which correlates well with the calculations on the target at load voltage of 5.0 Ω. Testing of the 48V unit showed similar performance as the 72V; however, with a lower optimal resistance as expected by the calculations for target ohms. Peak power for this controller was at 819 W for the 2.62 Ω resistance. This was achieved at a lower solar irradiance than the 72V model, but they operate with different PV configurations. Therefore, executing experiments with the two Geyserwise controllers with the same $2\,x\,2$ configuration would be interesting.

The Odin controller reached peak power with a resistance of 13.1 Ω. However, there is a large disparity between the peak power from the LabVIEW program for the controller and the multimeters that were used for measuring voltage and current on the output of the unit. Unfortunately, there appears to be an issue with the Odin controller. The problem might be that the algorithm is set up for a PV configuration of six panels in series and when these tests were done with four PV panels instead, it is not enough to adjust the power values by 2/3 as done in this work. Therefore, the results on the Odin controller are uncertain and should be performed with the six panels it is capable of instead. An aspect of the Odin controller to note is that the experiments performed with lower resistance showed higher peak temperatures and a higher rate of increase in temperature. As this temperature reaches the values of 40 – $50^{\circ}C$, it becomes troublesome for the controller and much of the incoming energy is transferred into heat.

Several possible solutions for developing the Arduino further with a web-based system were considered. The TinySine SIM7600 shield was bought as a method of connecting the existing Arduino with the cellular network and thus, publish the logged data so that it is accessible anywhere and anytime. Research showed that a common method for achieving this was via an MQTT broker and the Windows app, MQTT Explorer, was used as the receiving client of Arduino data. An Arduino IDE program for connecting the SIM shield to the cellular network was written, however it was unsuccessful for its intent. Further research suggested the use of AT commands in the program instead, and so the code was altered to accomplish a web connection. This method has unfortunately, also been unsuccessful in initializing the SIM shield and creating the link between the Arduino and the broker. The method of accomplishing a functional web-based Arduino by using a SIM shield with an MQTT broker seems to be correct, through the research completed during this thesis. However, further development of the Arduino IDE program must be done to become an operational unit for data logging with remote access.

8 Further work

The primary issue for further work is the testing of the controllers at a higher solar irradiance than what was achievable during the experimental period of this master work. Testing the controllers at these higher irradiance's of approximately 400+ W/m^2, ideally up towards the levels of the standard conditions of 1 000 W/m^2. Researching if there is a performance difference between the three controllers at this level and if the optimal resistance for peak power changes for each device.

A secondary objective of the further work could be to change the configuration of PV panels to a perhaps more optimal solution. This case would apply to the Geyserwise 72V and Odin controller. For the 72V, a parallel circuit with two panels in each series, as with the 48V would be interesting to experiment with. In addition, it would remove the issue of having too high input voltage and would not change the target ohms at load voltage from the calculations in appendix I of 5 Ω. This PV configuration is described in the manual of Geyserwise controllers in appendix B.

A reason for having four panels in series was to achieve high enough voltage, as the sun conditions were not optimal, and therefore the premise was that it would not be a problem. As the experiments showed, it became a minor issue but would probably cause larger complications when tested at a higher solar irradiance. The second purpose was to have the equal PV configuration for the 72V & Odin controller to aid the basis of comparison between them. However, for the Odin controller to achieve its optimal performance, a PV configuration of six panels in series should be tested.

There may be an issue with the algorithm for the Odin controller when performing experiments with four PV panels, instead of the six panels it is previously set up for. Therefore, looking into this part of the Odin controller would be interesting. Along with this, performing new tests with four panels would certainly be interesting, to provide new results that can be compared with the experiments for the Geyserwise controllers here.

Regarding the further work on the Arduino, one main aspect is the clear contender for this. The continuing of developing the Arduino IDE program such that it functions as intended and will send the data that the Arduino system is logging to the MQTT Explorer. Having the data available anywhere and anytime is a considerable benefit for the data logging system, such that it is necessary to further develop. Researching other options for creating a web-based Arduino solution is another interesting subject. One such option is to instead of using a MQTT broker and client, explore the possibility of using a cloud storage unit. If the raw data can be published to a solution akin to that and such have access to it anywhere, the data can then be extracted and processed as desired.

When or if, the web-based Arduino system is ready to be tested, testing in coalition with a longer running experiment of the controllers, would be interesting. Utilizing a set of thermocouples in the tubes of the heating elements to measure the water temperature, along with the connection for the shunt and voltage divider circuit. Performing reliability testing of both the controllers and Arduino system together over several hours or days,

would give results on how a set resistance behaves under such conditions. Furthermore, providing an answer on the power consumption of the SIM shield and if the system is capable of sustaining enough energy from a PV panel with the limited battery capacity, or if a larger battery might be necessary.

References

[1] The World Bank Group. *Population growth (annual %) - Tanzania | Data*. 2023. URL: https://data.worldbank.org/indicator/SP.POP.GROW?locations=TZ (visited on 15/03/2023).

[2] The World Bank Group. *Overview*. 23rd Sept. 2022. URL: https://www.worldbank.org/en/country/tanzania/overview (visited on 15/03/2023).

[3] The World Bank Group. *Tanzania | Data*. 2023. URL: https://data.worldbank.org/country/tanzania?view=chart (visited on 15/03/2023).

[4] IEA. *Tanzania electricity generation by technology in the Africa Case, 2010-2040 – Charts – Data & Statistics*. 26th Oct. 2022. URL: https://www.iea.org/data-and-statistics/charts/tanzania-electricity-generation-by-technology-in-the-africa-case-2010-2040 (visited on 15/03/2023).

[5] WHO. *Household air pollution*. 28th Nov. 2022. URL: https://www.who.int/news-room/fact-sheets/detail/household-air-pollution-and-health (visited on 27/11/2023).

[6] World Bank Group. *World Bank Open Data*. 2021. URL: https://data.worldbank.org (visited on 27/11/2023).

[7] Martina Igini. *Deforestation in Africa: Causes, Effects, and Solutions*. Earth.Org. 24th Mar. 2022. URL: https://earth.org/deforestation-in-africa/ (visited on 27/11/2023).

[8] Jessica Breitfeller. *Charcoal: An Important Driver of Deforestation in Africa*. 18th Aug. 2015. URL: https://www.forest-trends.org/blog/charcoal-an-important-driver-of-deforestation-in-africa/ (visited on 27/11/2023).

[9] Martin August Egerdahl. *Testing of controllers for PV2Heat*. June 2023. (Visited on 15/11/2023).

[10] Piotr Bojek. *Solar PV*. IEA. 11th July 2023. URL: https://www.iea.org/energy-system/renewables/solar-pv (visited on 15/11/2023).

[11] Håvard Karoliussen. *TFNE1001 Fornybar Energi Grunnkurs - Solenergi - Solceller og Solfangere*. 2018. (Visited on 15/11/2023).

[12] Knut A. Rosvold. *Watt peak*. In: *Store norske leksikon*. 11th Feb. 2023. URL: https://snl.no/watt_peak (visited on 15/11/2023).

[13] Maria Tsoutsouva. *TMT4285 Hydrogen technology, Fuel cells and Solar cells - Lecture 8: PV Module and Systems*. 2022. (Visited on 21/11/2023).

[14] Maria Tsoutsouva. *TMT4285 Hydrogen technology, Fuel cells and Solar cells - Lecture 2: Radiation*. 2022. (Visited on 21/11/2023).

[15] Joakim Widén and Joakim Munkhammar. *Solar Radiation Theory*. Apr. 2019. ISBN: 978-91-506-2760-2. DOI: 10.33063/diva-381852. (Visited on 21/11/2023).

[16] Oriol Planas. *What Is Solar Irradiance? Solar Irradiation*. 7th July 2021. URL: https://solar-energy.technology/what-is-solar-energy/solar-radiation/solar-irradiation (visited on 24/11/2023).

[17] Solargis. *Solar radiation modeling*. (n.d.) URL: https://solargis.com/docs/methodology/solar-radiation-modeling (visited on 24/11/2023).

[18] Marc A.N. Korevaar. "Measuring Solar Irradiance for Photovoltaics". In: *Solar Radiation*. Ed. by Mohammadreza Aghaei. Rijeka: IntechOpen, 2022. Chap. 2. DOI: 10.5772/intechopen.105580. URL: https://doi.org/10.5772/intechopen.105580 (visited on 24/11/2023).

[19] NVE. *Oversikt over solkraft i Norge - NVE*. 2nd Aug. 2023. URL: https://www.nve.no/energi/energisystem/solkraft/oversikt-over-solkraft-i-norge/ (visited on 15/11/2023).

[20] Asbjørn Skaaland et al. *Potential and Challenges for Building Integrated Photovoltaics in the Agder Region*. 1st Jan. 2011. URL: https://www.researchgate.net/publication/268487729_Potential_and_Challenges_for_Building_Integrated_Photovoltaics_in_the_Agder_Region (visited on 15/11/2023).

[21] Norsk Solenergiforening. *Solceller*. (n.d.) URL: https://www.solenergi.no/solstrm (visited on 21/11/2023).

[22] NVE. *Solkraft*. 20th July 2023. URL: https://www.nve.no/energi/energisystem/solkraft/ (visited on 21/11/2023).

[23] Solargis. *Solar resource maps of Africa*. 2021. URL: https://solargis.com/maps-and-gis-data/download/africa (visited on 15/11/2023).

[24] World Bank Group, ESMAP and Solargis. *Global Solar Atlas*. (n.d.) URL: https://globalsolaratlas.info/detail?c=-6.39557,34.892579,6&r=TZA (visited on 27/11/2023).

[25] Maria Tsoutsouva. *TMT4285 Hydrogen technology, Fuel cells and Solar cells - Lecture 3: Angles*. 2022. (Visited on 27/11/2023).

[26] Martin August Egerdahl, Thomas Dybvig Hansen and Johan Shields. "Solcelleanlegg og energilagring på Sveberg Teknosenter". Bachelor thesis. NTNU, May 2021. URL: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2779965 (visited on 18/12/2023).

[27] Osamede Asowata et al. "Optimizing the output power of a stationary PV panel". In: Sept. 2013. URL: https://www.researchgate.net/publication/258148774_Optimizing_the_output_power_of_a_stationary_PV_panel (visited on 17/12/2023).

[28] Morningstar Corporation. *What is a Solar Charge Controller and What Does it Do?* (n.d.) URL: https://www.morningstarcorp.com/faq/what-is-solar-charge-controller/ (visited on 19/12/2023).

[29] Victron Energy. *Which solar charge controller: PWM or MPPT?* 20th Jan. 2020. URL: https://www.victronenergy.com/upload/documents/Technical-Information-Which-solar-charge-controller-PWM-or-MPPT.pdf (visited on 19/12/2023).

[30] Morningstar Corporation. *What are the Different Types of Solar Charge Controllers?* (n.d.) URL: https://www.morningstarcorp.com/faq/what-are-the-different-types-of-solar-charge-controllers/ (visited on 19/12/2023).

[31] Hubert Ward. "Introductory Programs with the 32-bit PIC Microcontroller : A Line-by-Line Code Analysis and Reference Guide for Embedded Programming in

C". eng. In: 1st ed. 2023. Berkeley, CA, 2023. ISBN: 9781484290514. (Visited on 05/10/2023).

[32] Dogan Ibrahim. "Chapter 8 - Advanced PIC32 Projects". In: *Designing Embedded Systems with 32-Bit PIC Microcontrollers and MikroC*. Ed. by Dogan Ibrahim. Oxford: Newnes, 2014, pp. 359–442. ISBN: 978-0-08-097786-7. DOI: https://doi.org/10.1016/B978-0-08-097786-7.00008-7. URL: https://www.sciencedirect.com/science/article/pii/B9780080977867000087 (visited on 21/12/2023).

[33] Zed Song. *PWM vs MPPT Charge Controllers: Which one should you choose?* GridFree®. (n.d.) URL: https://gridfree.store/blogs/how-to-articles/mppt-vs-pwm-charge-controllers-which-one-should-you-choose (visited on 19/12/2023).

[34] Saleh Elkelani Babaa, Matthew Armstrong and Volker Pickert. "Overview of Maximum Power Point Tracking Control Methods for PV Systems". In: *Journal of Power and Energy Engineering* 02.8 (Aug. 2014), pp. 59–72. ISSN: 2327-588X, 2327-5901. DOI: 10.4236/jpee.2014.28006. URL: http://www.scirp.org/journal/doi.aspx?DOI=10.4236/jpee.2014.28006 (visited on 17/12/2023).

[35] RECOM Power. *An Introduction to Buck, Boost, and Buck/Boost Converters*. 30th Aug. 2020. URL: https://recom-power.com/en/rec-n-an-introduction-to-buck,-boost,-and-buck!sboost-converters-131.html?2 (visited on 19/12/2023).

[36] EETech Media. *Temperature Coefficient of Resistance*. (n.d.) URL: https://eepower.com/resistor-guide/resistor-fundamentals/temperature-coefficient-of-resistance/ (visited on 17/12/2023).

[37] Kristina Berg and Andrea Austjord Vik. "Demonstration of PV Power to High-temperature Heat Storage for Solar Cookers". Master thesis. NTNU, June 2022. URL: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/3023806 (visited on 17/12/2023).

[38] Heatrod Elements. *Tubular and Finned Heating Elements*. Backer Group Member, (n.d.) URL: https://www.heatrod.com/heating-elements/tubular--finned-elements (visited on 17/12/2023).

[39] DBK Industrial UK. *PTC Heating Technology*. (n.d.) URL: https://www.dbk-group.co.uk/ptc-heating (visited on 17/12/2023).

[40] Gengsheng Lawrence Zeng and Megan Zeng. *Electric Circuits: A Concise, Conceptual Tutorial*. eng. 1st ed. Cham: Springer International Publishing AG, 2021. ISBN: 9783030605148. DOI: 10.1007/978-3-030-60515-5. (Visited on 28/09/2023).

[41] Allegro MicroSystems. *ACS712: Hall-Effect-Based Linear Current Sensor IC*. 2023. URL: https://www.allegromicro.com/en/products/sense/current-sensor-ics/zero-to-fifty-amp-integrated-conductor-sensor-ics/acs712 (visited on 28/05/2023).

[42] ROHM Semiconductor. *Shunt Resistors*. (n.d.) URL: https://www.rohm.com/electronics-basics/resistors/shunt-resistors (visited on 29/12/2023).

[43] Encyclopædia Britannica Inc. *thermocouple*. eng. 2020. URL: https://academic.eb.com/levels/collegiate/article/thermocouple/72080 (visited on 30/05/2023).

[44] Arduino. *About Arduino*. 15th Sept. 2021. URL: https://www.arduino.cc/en/about (visited on 30/05/2023).

[45] Arduino. *What is Arduino?* 5th Feb. 2018. URL: https://www.arduino.cc/en/Guide/Introduction (visited on 30/05/2023).

[46] Terence O'Neill. "Arduino". eng. In: Cherry Lake Publishing, 2014. ISBN: 1-62431-203-9. (Visited on 30/05/2023).

[47] Warren Gay. "I2C". eng. In: *Beginning STM32*. Berkeley, CA: Apress, 2018, pp. 195–221. ISBN: 1484236238. (Visited on 30/05/2023).

[48] Warren Gay. "SPI Flash". eng. In: *Beginning STM32*. United States: Apress L. P, 2018, pp. 115–145. ISBN: 1484236238. (Visited on 30/05/2023).

[49] Warren Gay. "USART". eng. In: *Beginning STM32*. United States: Apress L. P, 2018, pp. 73–96. ISBN: 1484236238. (Visited on 05/10/2023).

[50] MQTT.org. *MQTT - The Standard for IoT Messaging*. 2022. URL: https://mqtt.org/ (visited on 11/12/2023).

[51] Adafruit. *MQTT, Adafruit IO & You!* (n.d.) URL: https://learn.adafruit.com/mqtt-adafruit-io-and-you/overview (visited on 11/12/2023).

[52] Nitin Sharma. *Getting Started with MQTT — Part 1*. 29th Apr. 2020. URL: https://nitin-sharma.medium.com/getting-started-with-mqtt-part-1-a3c365e3a488 (visited on 10/12/2023).

[53] HiveMQ. *MQTT Essentials - All Core Concepts Explained*. 2023. URL: https://www.hivemq.com/mqtt/ (visited on 11/12/2023).

[54] Geyserwise. *Company History*. (n.d.) URL: https://www.geyserwise.com/about-us/company-history/ (visited on 05/01/2024).

[55] Victron Energy. *SmartSolar Charge Controller MPPT 150/35 & 150/45*. (n.d.) URL: https://www.victronenergy.com/upload/documents/Datasheet-SmartSolar-charge-controller-MPPT-150-35-&-150-45-EN.pdf (visited on 06/01/2024).

[56] Victron Energy. *MPPT Calculator*. (n.d.) URL: https://www.victronenergy.com/mppt-calculator (visited on 06/01/2024).

[57] Victron Energy. *SmartSolar MPPT 150/35 & 150/45*. (n.d.) URL: https://www.victronenergy.com/solar-charge-controllers/smartsolar-150-35 (visited on 06/01/2024).

[58] bluepowershop. *Can I use excess power to heat water off grid? - Victron Community*. 21st Nov. 2018. URL: https://community.victronenergy.com/questions/1407/can-i-use-excess-power-to-heat-water-off-grid.html (visited on 06/01/2024).

[59] Morningstar Corporation. *TriStar – Three Function Solar Controller*. Sept. 2021. URL: https://www.morningstarcorp.com/wp-content/uploads/datasheet-tristar-en.pdf (visited on 06/01/2024).

[60] Morningstar Corporation. *Diversion Controller*. (n.d.) URL: https://www.morningstarcorp.com/glossary/diversion-controller/ (visited on 06/01/2024).

[61] Morningstar Corporation. *History*. (n.d.) URL: https://www.morningstarcorp.com/company/history/ (visited on 06/01/2024).

[62] Morningstar Corporation. *TriStar MPPT – Solar Controller with Maximum Power Point Tracking*. Sept. 2021. URL: https://www.morningstarcorp.com/wp-content/uploads/datasheet-tristar-mppt-en.pdf (visited on 06/01/2024).

[63] Microcare. *Geyser Controller*. (n.d.) URL: https://microcare.co.za/wp-content/uploads/2023/02/Geyser-Controller-datasheet.pdf (visited on 06/01/2024).

[64] Microcare. *About*. 17th Oct. 2022. URL: https://microcare.co.za/about/ (visited on 06/01/2024).

[65] Microcare. *PV Geyser Solution*. 20th Oct. 2022. URL: https://microcare.co.za/pv-geyser-solution/ (visited on 06/01/2024).

[66] GeyserWorx. *Geyserworx® Geyser Solar Power Conversion Unit - White*. (n.d.) URL: https://geyserworx.co.za/store/Geyserworx-Solar-Geyser-Unit-White (visited on 06/01/2024).

[67] GeyserWorx. *About Us*. (n.d.) URL: https://geyserworx.co.za/store/about_us (visited on 06/01/2024).

[68] GeyserWorx. *The Smart way to convert to Solar Power!* (n.d.) URL: https://geyserworx.co.za/ (visited on 06/01/2024).

[69] Fronius International GmbH. *Hot water with photovoltaics*. (n.d.) URL: https://www.fronius.com/en/solar-energy/home-owners/products-and-solutions/heating-with-pv (visited on 06/01/2024).

[70] Fronius International GmbH. *Fronius Ohmpilot*. (n.d.) URL: https://www.fronius.com/en/solar-energy/installers-partners/technical-data/all-products/solutions/fronius-solution-for-heat-generation/fronius-ohmpilot/fronius-ohmpilot (visited on 06/01/2024).

[71] Kristian Skeie and Arild Gustavsen. "Predicting solar radiation using a parametric cloud model". In: *E3S Web of Conferences* 172 (Jan. 2020), p. 11006. DOI: 10.1051/e3sconf/202017211006. (Visited on 12/12/2023).

[72] Steffen Thorsen. *Sun & moon times today, Trondheim, Norway*. 2023. URL: https://www.timeanddate.com/astronomy/norway/trondheim (visited on 05/10/2023).

[73] GETEK AS. *Solartek - PV high-performance modules*. (n.d.) URL: https://getek.no/wp-content/uploads/2017/03/Solartek-2017.pdf (visited on 29/08/2023).

[74] GETEK AS. *Solartek - Solcellepaneler*. Feb. 2016. URL: https://getek.no/wp-content/uploads/2017/03/SOLARTEKstandard_N_25-235W.pdf (visited on 29/08/2023).

[75] Backer AB. *IMMERSION HEATER FOR WATER R50 ELEMENT TUBE IN COPPER, BRASS HEAD*. (n.d.) URL: https://www.backergroup.com/download/18.521b1d3d17e5b499ae7ed9/1642582095101/Immersion%20heater%20for%20water%20R50%20element%20tube%20in%20copper,%20brass%20head.pdf (visited on 04/09/2023).

[76] Norske Backer. *About us*. (n.d.) URL: https://norskebacker.no/en/about-us-eng/ (visited on 04/09/2023).

[77] TinySine. *4G(LTE)/3G/GSM Shield for Arduino with GPS - SIM7600CE*. 2023. URL: https://www.tinyosshop.com/index.php?route=product/product&product_id=1109 (visited on 15/12/2023).

[78] Ubuntu [@ubuntu]. *If you're seeking a comprehensive and easy to use #MQTT client for #Ubuntu then look no further than MQTT Explorer.* Twitter. 7th May 2019. URL: https://twitter.com/ubuntu/status/1125747632910077953 (visited on 15/12/2023).

[79] Thomas Nordquist. *MQTT Explorer*. (n.d.) URL: http://mqtt-explorer.com/ (visited on 15/12/2023).

A Solartek PVP26030 Solar Panel

The Solartek PV high-performance modules. [73]

solartek®

PV high-performance modules

- Guaranteed positive output tolerance -0/+5Wp by single measuring
- Maximum 8000Pa snow load
- Maximum stability through aluminum frame Soft Grip
- High-quality junction box and connector systems
- Made in Europe

PV modules for extreme areas

12 years manufacturer's warranty

25 years linear performance guarantee to 85% output

Modern and fully automated production

GETEK ENERGY

www.getek.no
post@getek.no

solartek®

PVP – Polycrystalline series:

-option: black frame

Type	PVP25030	PVP26030	PVP27030
Nominal output Pmpp	250Wp	260Wp	270Wp
Nominal voltage Umpp	30,70V	30,92V	30,94V
Nominal current Impp	8,18A	8,43A	8,80A
Short circuit current Isc	8,41A	9,01A	9,41A
Open circuit voltage Uoc	37,80V	38,00V	39,26V
Module conversion efficiency	15,37%	15,98%	16,40%

Electrical characteristics (at Standard Test Conditions (STC) of irradiance 1000W / m2 , spectrum AM 25°C)

Design	
Frontside	3,2mm anti-reflective glass
Cells	60polycrystalline high efficiency cells 156x156 mm (6")
Backside	Composite film
Frame	40mm silver anodized aluminium frame

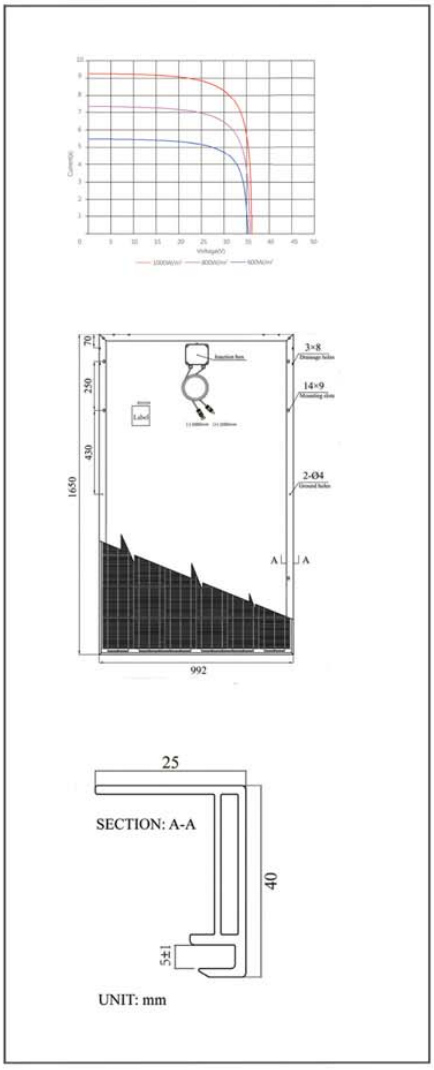
Mechanical data	
LxWxH	1650x992x40 mm
Weight	19,5kg with frame

Power conection	
Socket	Protection classe IP65 (3 bypass diodes
Wire	Approx 110 cm / 4 mm2
Plug-in system	Plug / Socket IP67

Limit values	
System voltage	1000VDC
NOCT*	45°C +/- 2K
Max. load-carring capacity	5400 N/m² tested to 8000 Pa
Reverse current feed IR	16,0A

*NOCT, irradiance 800 W/m2 AM 1,5, wind speed 1 m/sec, Temperature 20°C

Temperature coefficient	
Voltage Uoc	-0,30%K
Current Isc	+0,04%K
Output Pmpp	-0,42%K



solartek®

PVMB – Monocrystalline all black series:

Type	PVMB25030	PVMB26030	PVMB27030	PVMB28030
Nominal output Pmpp	250Wp	260Wp	270Wp	280Wp
Nominal voltage Umpp	29,65V	30,42V	30,94V	31,30V
Nominal current Impp	8,47A	8,60A	8,80A	8,96A
Short circuit current Isc	8,80A	9,06A	9,41A	9,50A
Open circuit voltage Uoc	37,98V	38,30V	39,26V	39,32V
Module conversion efficiency	15,37%	15,98%	16,60%	17,20%

Electrical characteristics (at Standard Test Conditions (STC) of irradiance 1000W / m2 , spectrum AM 25°C)

Design	
Frontside	3,2mm anti-reflective glass
Cells	60monocrystalline high efficiency cells 156x156 mm (6")
Backside	Composite film
Frame	40mm silver anodized aluminium frame

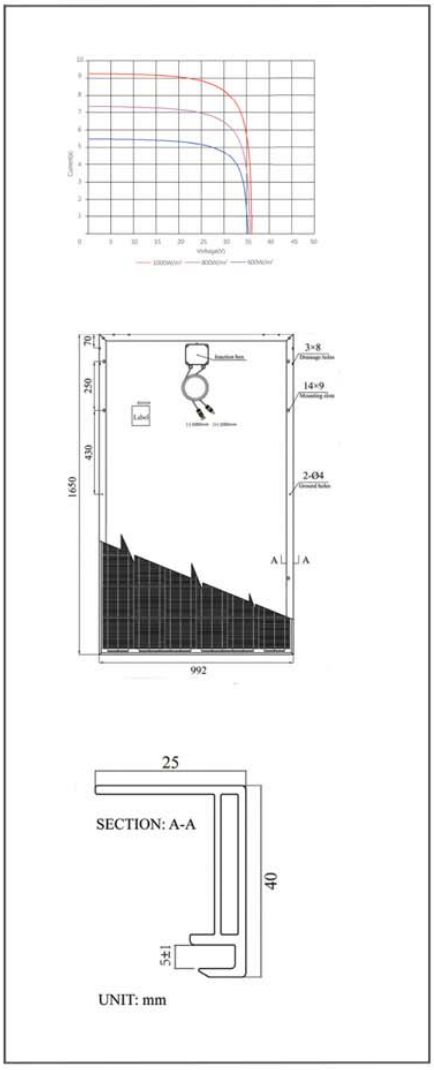
Mechanical data	
LxWxH	1650x992x40 mm
Weight	19,5kg with frame

Power conection	
Socket	Protection classe IP65 (3 bypass diodes
Wire	Approx 110 cm / 4 mm2
Plug-in system	Plug / Socket IP67

Limit values	
System voltage	1000VDC
NOCT*	45°C +/- 2K
Max. load-carring capacity	N/m² tested to 8000 Pa 5400
Reverse current feed IR	16,0A

*NOCT, irradiance 800 W/m2 AM1,5; wind speed 1 m/sec; Temperature 20°C

Temperature coefficient	
Voltage Uoc	-0,30%K
Current Isc	+0,04%K
Output Pmpp	-0,42%K



solartek®

PVMHE -
MonoCrystalline hight efficiency series:

Type	PVMHE28030	PVMHE29030	PVMHE30030
Nominal output Pmpp	280Wp	290Wp	300Wp
Nominal voltage Umpp	31,30V	31,58V	32,10V
Nominal current Impp	8,96A	9,20A	9,41A
Short circuit current Isc	9,50A	9,66A	9,97A
Open circuit voltage Uoc	39,50V	39,60V	39,80V
Module conversion efficiency	17,20%	17,60%	18,35%

Electrical characteristics (at Standard Test Conditions (STC) of irradiance 1000W / m2 , spectrum AM 25°C)

Design	
Frontside	3,2mm anti-reflective glass
Cells	60monocrystalline high efficiency cells 156x156 mm (6")
Backside	Composite film
Frame	40mm silver anodized aluminium frame

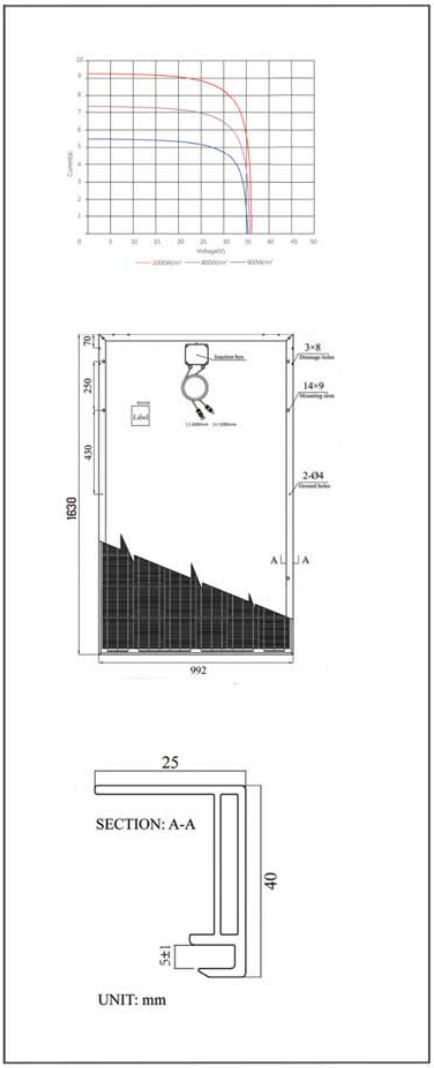
Mechanical data	
LxWxH	1630x992x40 mm
Weight	19,5kg with frame

Power conection	
Socket	Protection classe IP65 (3 bypass diodes
Wire	Approx 110 cm / 4 mm2
Plug-in system	Plug / Socket IP67

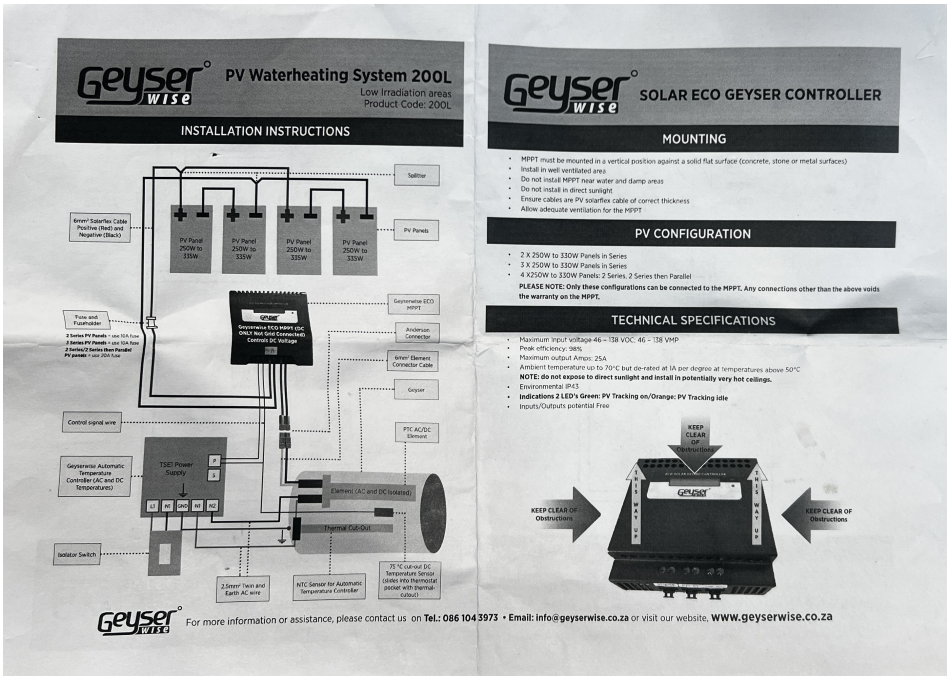
Limit values	
System voltage	1000VDC
NOCT*	42,9°C +/- 2K
Max. load-carring capacity	N/m² tested to 8000 Pa 5400
Reverse current feed IR	16,0A

*NOCT, irradiance 800 W/m2 AM 1,5; wind speed 1 m/sec; Temperature 20°C

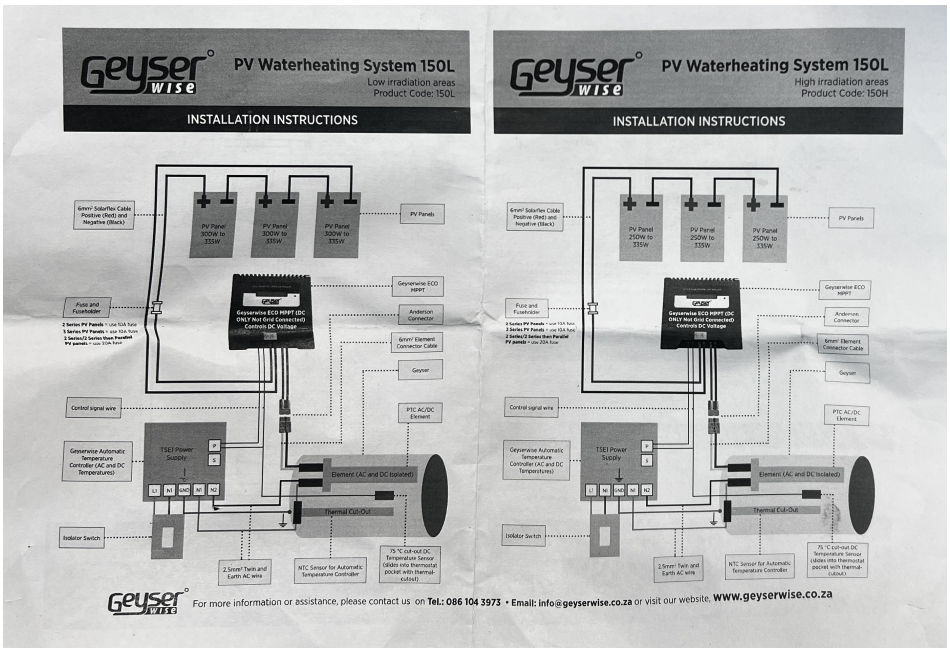
Temperature coefficient	
Voltage Uoc	-0,29%K
Current Isc	+0,05%K
Output Pmpp	-0,42%K



B Geyserwise Controller Manual



(a) Page 1 and 4, on the right and left respectively.



(b) Page 2 and 3.

Figure B.1: The Geyserwise controllers manual that is included in the shipping box.

A more in depth manual is found at the following link: GeyserWise PV System installers guide.

C Odin MPPT Load Controller

Relevant excerpts from the manual of the MPPT Load Controller created by Odin Hoff Gardå in February 2020.

MPPT Load Controller

Documentation
Rev. 2
Feb 2020 - Odin Hoff Gardå

Features

- 6 thermocouple inputs
- DC/DC buck converter
- P&O MPPT
- Proportional voltage MPPT
- RMS power measurement
- Datalogging to microSD card and real-time clock.
- Manual load adjustment mode

Maximum ratings

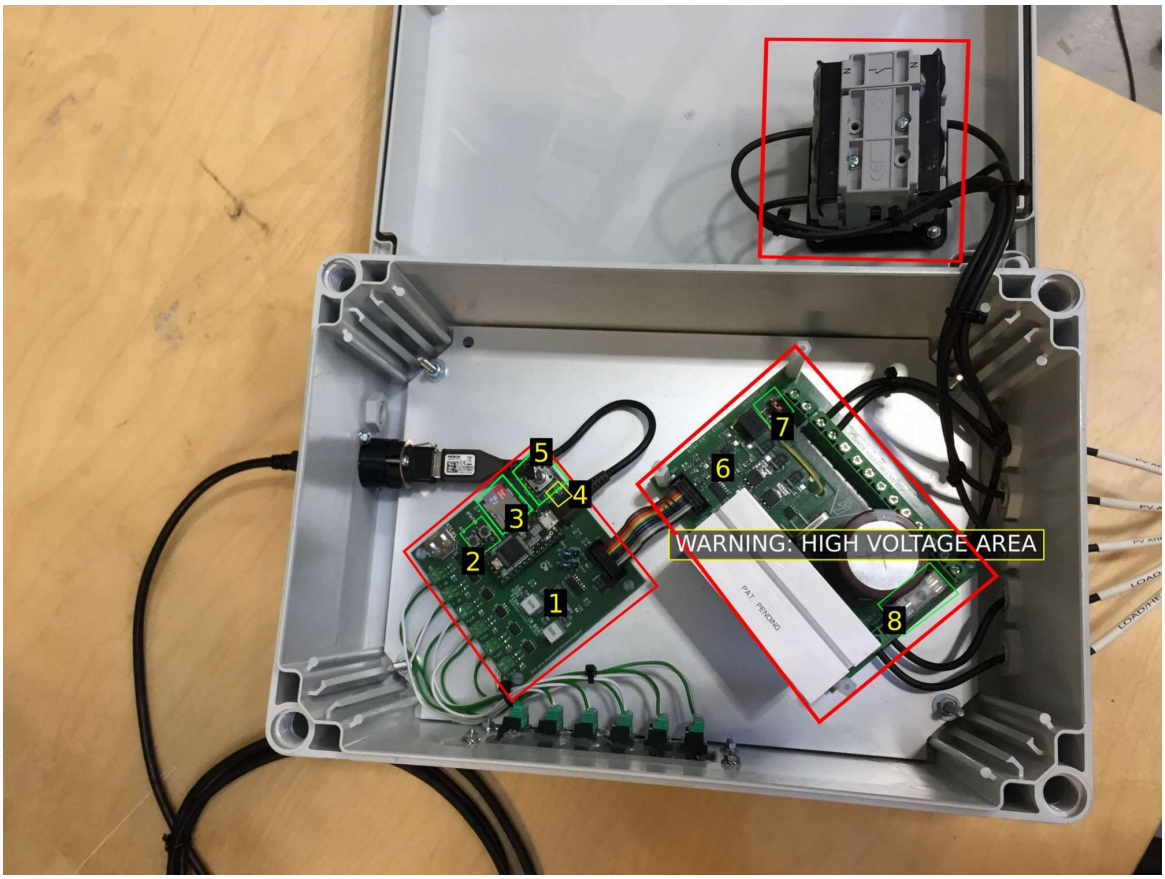
Parameter	Min	Nominal	Max	Unit
Voltage (single panel)	16	38	40	V
Voltage (6 panels in series)	96	220	230	V
Load current	0		10	A
Aux. power voltage	16	24	42	V

WARNING: Parts at high voltage potential are exposed inside the enclosure. Connection should only be done by a qualified person.

WARNING: Always make sure polarity and voltages are correct. Reverse polarity and over-voltage can destroy the system.

1

Overview:



1. Low voltage controller board 2. Push buttons S1 and S2 3. microSD card
4. Status LEDs 5. Potmeter for manual PWM control 6. High voltage power board
7. AUX power jumper 8. Main fuse

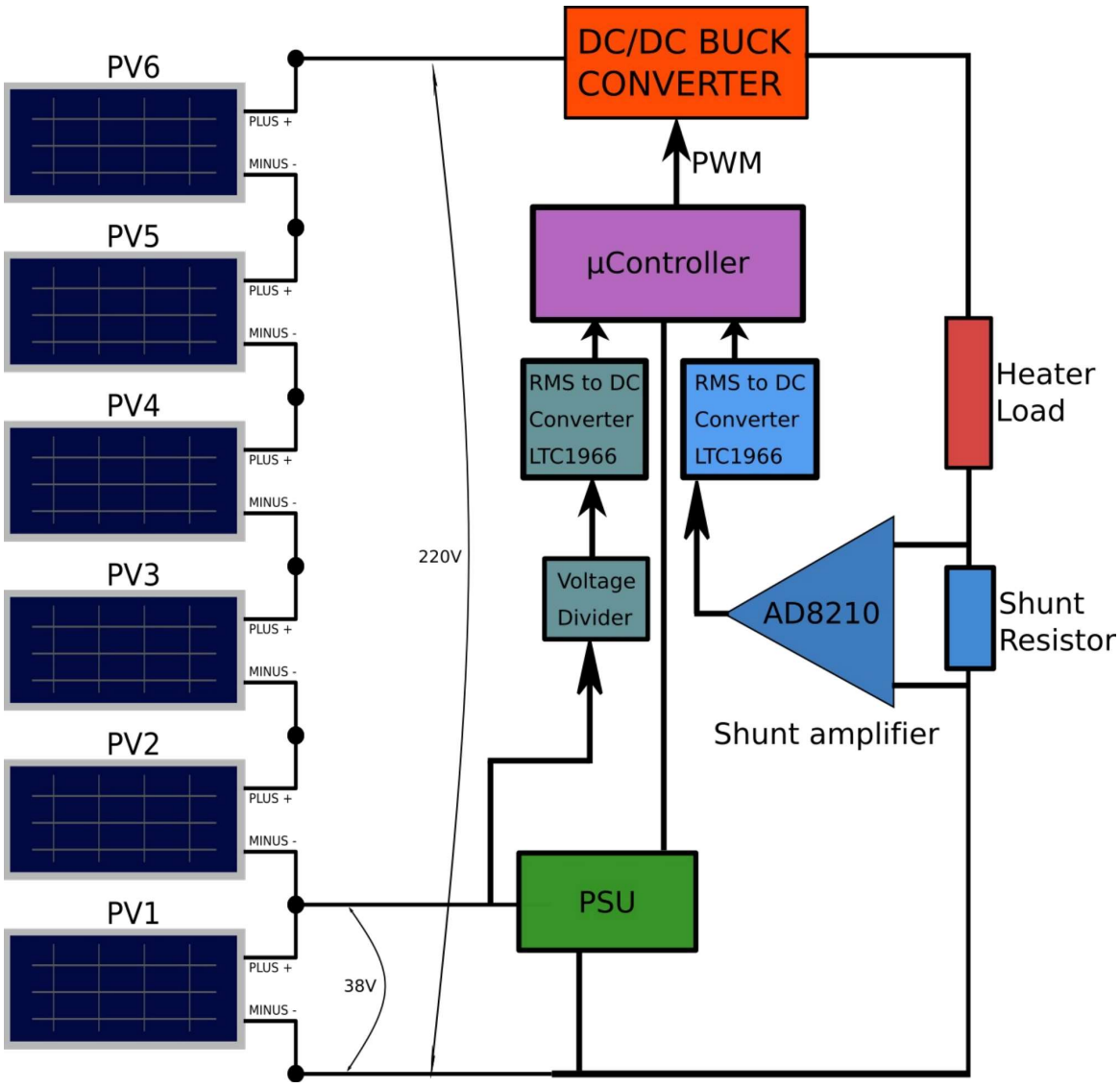
6

You should now be able to read real-time measurements including: thermocouples, voltage, current, power and PWM output to the DC/DC converter. If nothing happens, try turing the power off and on again on the controller box and restart the software on the PC.

Note: If the Heatsink temperature gets above 40-50 degrees C, consider opening the controller box to reduce the temperature.

Note: Data is logged to files on the computer (in the folder LogFiles on desktop).

Simplified system diagram



Auxiliary power connection

External power can be connected to drive the controller board so that datalogging and temperatures are logged even when there is no power from the solar PV array present. The voltage must be between 16V and 42V DC. **Do not reverse the polarity.**

AUX power jumper near the connection terminals must be set from **PV** to **AUX**.

RMS voltage and current measurements

The voltage of the lowest panel (PV1) is measured and multiplied by the number of panels to approximate the total voltage of the PV array. RMS calculations are done in hardware using LTC1966 RMS to DC converters.

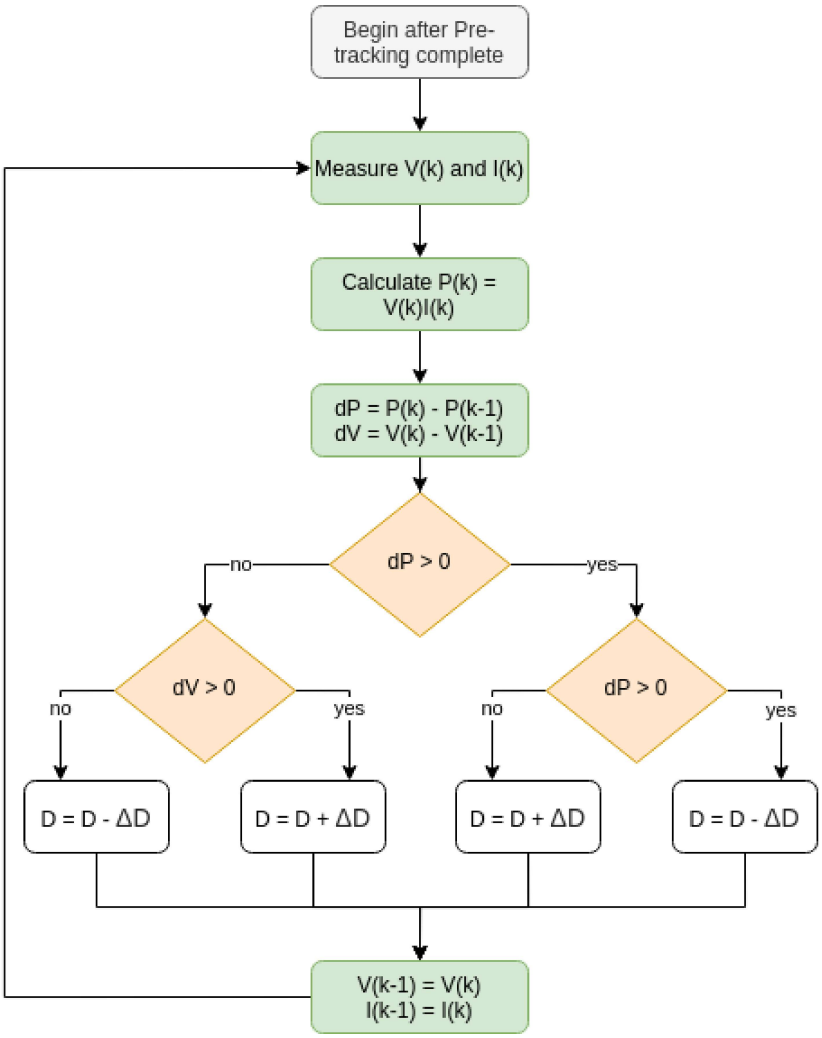
The current is measured using a 0.025 Ohm shunt resistor together with a current shunt amplifier (AD8210). As with voltage, the RMS is computed using either software or with LTC1966.

Maximum Power Point Tracking (MPPT)

The algorithm begins by the following **pre-tracking routine**:

1. Measure the open circuit voltage.
2. Increase the load until voltage has decreased 10% from the open circuit voltage.
3. Pre-Tracking complete, start MPPT

The MPPT algorithm is of the standard «perturb and observe» type (P&O).



V(k) Voltage at time k
I(k) Current at time k
D PWM Duty cycle
ΔD Stepsize

Note: In the actual implementation, the stepsize depends on the system voltage.

D Victron SmartSolar Charge Controller MPPT 150/35 & 150/45

victron energy
BLUE POWER

SmartSolar Charge Controller MPPT 150/35 & 150/45

www.victronenergy.com

SmartSolar Charge Controller MPPT 150/35

Bluetooth Smart built-in
The wireless solution to set-up, monitor, update and synchronise SmartSolar Charge Controllers.

VE.Direct
For a wired data connection to a Color Control GX, other GX products, PC or other devices

Ultrafast Maximum Power Point Tracking (MPPT)
Especially in case of a clouded sky, when light intensity is changing continuously, an ultra-fast MPPT controller will improve energy harvest by up to 30 % compared to PWM charge controllers and by up to 10 % compared to slower MPPT controllers.

Advanced Maximum Power Point Detection in case of partial shading conditions
If partial shading occurs, two or more maximum power points may be present on the power-voltage curve.
Conventional MPPTs tend to lock to a local MPP, which may not be the optimum MPP.
The innovative BlueSolar algorithm will always maximize energy harvest by locking to the optimum MPP.

Outstanding conversion efficiency
No cooling fan. Maximum efficiency exceeds 98 %. Full output current up to 40 °C (104 °F).

Flexible charge algorithm
Fully programmable charge algorithm (see the software page on our website), and eight preprogrammed algorithms, selectable with a rotary switch (see manual for details).

Extensive electronic protection
- Over-temperature protection and power derating when temperature is high.
- PV short circuit and PV reverse polarity protection.
- PV reverse current protection.

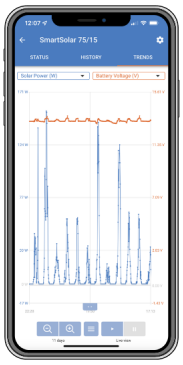
Internal temperature sensor
Compensates absorption and float charge voltage for temperature.

Optional external battery voltage and temperature sensing via Bluetooth
A Smart Battery Sense or a BMV-712 Smart Battery Monitor can be used to communicate battery voltage and temperature to one or more SmartSolar Charge Controllers.

Fully discharged battery recovery function
Will initiate charging even if the battery has been discharged to zero volts.
Will reconnect to a fully discharged Li-ion battery with integrated disconnect function.

Bluetooth sensing Smart Battery Sense

Bluetooth sensing BMV-712 Smart Battery Monitor



SmartSolar Charge Controller	MPPT 150/35	MPPT 150/45
Battery voltage	12 / 24 / 48 V Auto Select (software tool needed to select 36 V)	
Rated charge current	35 A	45 A
Nominal PV power 1a, b)	35 A 12 V: 500 W / 24 V: 1000 W / 36 V: 1500 W / 48 V: 2000 W 45 A 12 V: 650 W / 24 V: 1300 W / 36 V: 1950 W / 48 V: 2600 W	
Max. PV short circuit current 2)	40 A	50 A
Maximum PV open circuit voltage	150 V absolute maximum coldest conditions 145 V start-up and operating maximum	
Maximum efficiency	98 %	
Self-consumption	12 V: 20 mA 24 V: 15 mA 48 V: 10 mA	
Charge voltage 'absorption'	Default setting: 14,4 / 28,8 / 43,2 / 57,6 V (adjustable)	
Charge voltage 'float'	Default setting: 13,8 / 27,6 / 41,4 / 55,2 V (adjustable)	
Charge algorithm	multi-stage adaptive (eight pre-programmed algorithms)	
Temperature compensation	-16 mV / -32 mV / -64 mV / °C	
Protection	PV reverse polarity / output short circuit / over-temperature	
Operating temperature	-30 to +60°C (full rated output up to 40°C)	
Humidity	95 %, non-condensing	
Data communication port	VE.Direct See the data communication white paper on our website	
ENCLOSURE		
Colour	Blue (RAL 5012)	
Power terminals	16 mm² / AWG6	
Protection category	IP43 (electronic components), IP22 (connection area)	
Weight	1,25 kg	
Dimensions (h x w x d)	130 x 186 x 70 mm	
STANDARDS		
Safety	EN/IEC 62109-1, UL 1741, CSA C22.2	
STORED TRENDS		
Data stored	Battery voltage,current and temperature, as well as load output current, PV voltage and PV current.	
Number of days trends data is stored	46	

1a) If more PV power is connected, the controller will limit input power.
1b) The PV voltage must exceed Vbat + 5 V for the controller to start.
Thereafter the minimum PV voltage is Vbat + 1 V.
2) A PV array with a higher short circuit current may damage the controller.

Victron Energy B.V. | De Paal 35 | 1351 JG Almere | The Netherlands
E-mail: sales@victronenergy.com | **www.victronenergy.com**

victron energy
BLUE POWER

E Microcare Geyser Controller

MICRO CARE

Geyser Controller

The Microcare PV Geyser Controller (aka Kushushu) is a locally designed product that uses standard solar panels to power a common geyser element, outperforming the current range of thermal solar geysers in the market. With no plumbing required this locally designed innovation is easy to retrofit to any system making it price competitive at initial outlay and for the lifespan of the system.

It's unique design uses solar panels, not circulating water, thus preventing the problems associated with the old style collectors that suffer from freezing over in winter and boiling over in summer. For any building the new Microcare PV Geyser Controller is the preferred solution as there is only electric wires to be installed compared to the long pipes required for traditional solar geysers. There is also no heat loss as water doesn't need to circulate and therefore no waiting for water to get hot and a reduction in waste.

Features include:

- Reduce your energy spend by 30%
- Fit to an existing geyser with no plumbing required
- Retrofits to any 2kW, 3kW or 4kW element
- No need to replace Element or Thermostat
- Out performs solar thermal heaters & operates in inclement weather
- Efficiency greater than 96% on solar
- Mains override if no solar is present
- Wi-Fi App for set-up & daily operation
- Low heat dissipation
- Accurate digital temperature control

PRODUCT SPECIFICATIONS	
Suitable AC Elements	2kW, 3kW or 4kW
Max Input Solar Panel Voltage	275Voc
Min Input Solar Panel Voltage (Recommended)	2kW – 185Voc, 3 & 4kW – 139Voc
Max Input Solar Panel Power	2000W (2kW)
Min Input Solar Panel Power	900W (0.9kW)
Rated AC Input Grid Amps	20A
Rated AC Input Voltage	230V AC
Dimensions (l x w x d)	26 x 25 x 4cm
Weight	1.4kg
Warranty	24 months

Microcare recommends 4 x 335W Panels to be installed per Element

+27 (0)41 453 5761 sales@microcare.co.za www.microcare.co.za

A user manual for the controller is found at the following link: Microcare PV Geyser Controller User Manual.

F Python code – SOLARTEK PVP26030 IV-curve

Python program written to adapt the IV-curve of the datasheets into one that can be used to compare results.

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.interpolate import interp1d

# Sample IV curve data for different irradiation levels (replace with actual data)

irradiation_levels = [1000, 800, 600, 400, 200]  # in W/m^2
colors = ['y', 'b', 'k', 'g', 'r']


# IV curve data for each irradiation level
iv_curves = [
    {'voltage': np.array([0, 13, 13.05, 14, 15, 16.5, 18.5]),
     'current': np.array([9.01, 9.01, 9.01, 8.83, 8.29, 6.307, 1.62])},
    {'voltage': np.array([0, 13, 13.05, 14, 15, 16.5, 18.5]),
     'current': np.array([7.21, 7.21, 7.21, 7.03, 6.6674, 4.86, 0.991])},
    {'voltage': np.array([0, 13, 13.05, 14, 15, 15.5, 16.5, 18.5]),
     'current': np.array([5.41, 5.41, 5.41, 5.36, 5.23, 4.95, 3.69, 0.27])},
    {'voltage': np.array([0, 13, 13.05, 14, 15, 15.5, 16.5, 18.1]),
     'current': np.array([3.6, 3.6, 3.6, 3.6, 3.43, 3.24, 2.43, 0.09])},
    {'voltage': np.array([0, 13, 13.05, 14, 15, 15.5, 16.5, 17.2]),
     'current': np.array([1.8, 1.8, 1.8, 1.8, 1.71, 1.576, 0.901, 0.02])},
]


# Create a plot
plt.figure(figsize=(8, 6))

for i, iv_curve in enumerate(iv_curves):
    voltage = iv_curve['voltage']
    current = iv_curve['current']
    #plt.plot(voltage, current, linestyle='-', label=f'Irradiation {irradiation_levels[i]} W/m^2', color=colors[i])

 # Find the index of the voltage value where you want to start smoothing (e.g., 13 V)
    start_index = np.where(voltage >= 13)[0][0]

    # Plot the original data points up to the specified voltage
    plt.plot(voltage[:start_index + 1], current[:start_index + 1], linestyle='-', color=colors[i])

    # Perform cubic spline interpolation on data points after the specified voltage
    interp_func = interp1d(voltage[start_index:], current[start_index:], kind='cubic')
    voltage_smooth = np.linspace(min(voltage[start_index:]), max(voltage[start_index:]), 100000)  # Smooth voltage values
    current_smooth = interp_func(voltage_smooth)

    # Plot the smoothed curve after the specified voltage
    plt.plot(voltage_smooth, current_smooth, label=f'Irradiation {irradiation_levels[i]} $W/m^2$', color=colors[i])

plt.title('IV Curves for Different Solar Irradiation Levels')
plt.xlabel('Voltage (V)')
plt.ylabel('Current (I)')
plt.grid(True)
plt.legend()

# Set the y-axis limits to be between 0 and 10 for current
plt.ylim(0, 10)

# Set more points on the y-axis
plt.yticks(np.arange(0, 11, 1))

# Show the plot
plt.show()
```

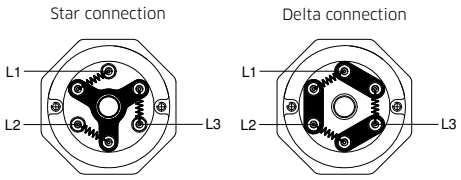
G Backer IU313 Heating Elements

IMMERSION HEATER FOR WATER R50
ELEMENT TUBE IN COPPER, BRASS HEAD

BACKER

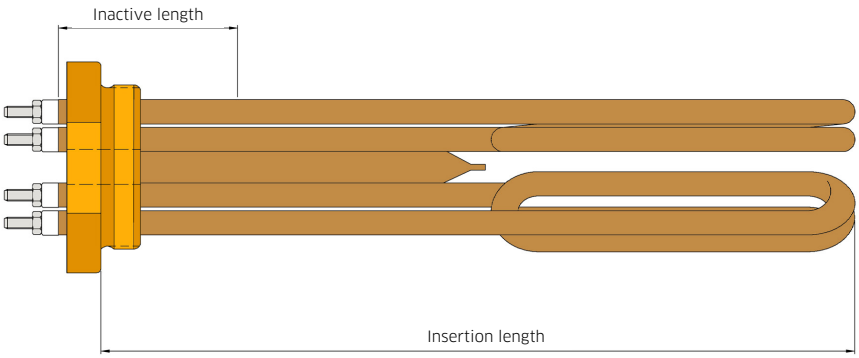
Description

With element tube in copper and brass head R50.
Tubular element in copper 2.0090 (C12200).
Brass head R50 (G 2").
The element is approved for 9 bar, temperature 120°C.
Thermostat tube with inside diameter 10,6 mm for thermostat and/or temperature limiter.



Technical specification

Type	Effect	Voltage	Insertion length	Length thermostat tube	Inactive length	Surface power W/cm²	Connection	Item number
IU31	1500 W	230/400 V	180 mm	187 mm	60 mm	7,2 W/cm²	Y	2520 341 001
IU33	2250 W	230/400 V	230 mm	187 mm	60 mm	7,9 W/cm²	Y	2520 341 002
IU34	3000 W	230/400 V	280 mm	260 mm	60 mm	8,2 W/cm²	Y	2520 341 003
IU36	4500 W	230/400 V	390 mm	260 mm	60 mm	8,3 W/cm²	Y	2520 341 005
IU39	6000 W	230/400 V	390 mm	260 mm	60 mm	8,1 W/cm²	Y	2520 341 007
IU310	7500 W	230/400 V	390 mm	260 mm	60 mm	8,3 W/cm²	Y	2520 341 008
IU311	9000 W	230/400 V	390 mm	260 mm	60 mm	9,9 W/cm²	Y	2520 341 009
IU312	10500 W	230/400 V	860 mm	260 mm	60 mm	8,0 W/cm²	Y	2520 341 010
IU313	12000 W	230/400 V	985 mm	260 mm	60 mm	8,0 W/cm²	Y	2520 341 011
IU314	13500 W	230/400 V	1085 mm	260 mm	60 mm	8,1 W/cm²	Y	2520 341 012
IU315	15000 W	3x400 V	1235 mm	260 mm	60 mm	7,9 W/cm²	D	2520 341 013



Backer-SE-Ver.01

Head Office: Backer AB · SE-282 71 Sösdala · Sweden · Phone. +46 451 661 00 · www.backer.eu

H Python code – SD Card Read and Plot

A Python program written to plot data from the SD card in the Arduino, once the card is inserted in a laptop. The program seeks to read the data from the SD card and plot the data for the current day, last seven days and the last 30 days.

In [ ]:

```
import os
import glob
import matplotlib.pyplot as plt
import datetime

# Step 1: List all .txt files on the SD card
sd_card_path = r'D:\\'  # Drive letter of your SD card
file_pattern = 'DATA*.txt'
file_list = glob.glob(os.path.join(sd_card_path, file_pattern))

# Step 2: Read and extract data with the provided format
timestamps = []
temperature_1 = []
temperature_2 = []
temperature_3 = []
temperature_4 = []

for file_path in file_list:
    with open(file_path, 'r') as file:
        lines = file.readlines()
        for line in lines:
            data = line.strip().split(',')
            if len(data) == 6:
                timestamp_str, temp1_str, temp2_str, temp3_str, temp4_str, _ = data
                timestamps.append(timestamp_str)
                temperature_1.append(float(temp1_str))
                temperature_2.append(float(temp2_str))
                temperature_3.append(float(temp3_str))
                temperature_4.append(float(temp4_str))

# Define a function to filter data for a specific date range
def filter_data(start_date, end_date):
    filtered_timestamps = []
    filtered_temperature_1 = []
    filtered_temperature_2 = []
    filtered_temperature_3 = []
    filtered_temperature_4 = []

    for i in range(len(timestamps)):
        timestamp_date = datetime.datetime.strptime(timestamps[i], '%d/%m/%Y').date()
        if start_date <= timestamp_date <= end_date:
            filtered_timestamps.append(timestamps[i])
            filtered_temperature_1.append(temperature_1[i])
            filtered_temperature_2.append(temperature_2[i])
            filtered_temperature_3.append(temperature_3[i])
            filtered_temperature_4.append(temperature_4[i])

    return filtered_timestamps, filtered_temperature_1, filtered_temperature_2, filtered_temperature_3, filtered_temperature_4

# Get the current date
current_date = datetime.datetime.now().date()

# Define the current day start and end times (00:00 to 23:59)
current_day_start_time = datetime.datetime(current_date.year, current_date.month, current_date.day, 0, 0)
current_day_end_time = datetime.datetime(current_date.year, current_date.month, current_date.day, 23, 59)

# Define the last 7 days start time (00:00 on the first day) and end time (Latest measurement)
last_7_days_start_time = datetime.datetime(current_date.year, current_date.month, current_date.day - 6, 0, 0)
last_7_days_end_time = max(timestamps)  # Use the latest measurement as the end time

# Define the last 30 days start time (00:00 on the first day) and end time (Latest measurement)
last_30_days_start_time = datetime.datetime(current_date.year, current_date.month, current_date.day - 29, 0, 0)
last_30_days_end_time = max(timestamps)  # Use the latest measurement as the end time

# Create a custom function to create plots with custom x-axis labels
def create_custom_time_plot(timestamps, temp_1, temp_2, temp_3, temp_4, title, start_time, end_time):
    plt.figure(figsize=(12, 6))
    plt.plot(timestamps, temp_1, label='Temperature 1')
    plt.plot(timestamps, temp_2, label='Temperature 2')
    plt.plot(timestamps, temp_3, label='Temperature 3')
    plt.plot(timestamps, temp_4, label='Temperature 4')
    plt.xlabel('Time')
    plt.ylabel('Temperature')
    plt.title(title)
    plt.legend()
    plt.grid(True)
    plt.xticks(rotation=45)

    # Customize the x-axis labels based on the specified start and end times
    plt.xlim(start_time, end_time)
    plt.gca().xaxis.set_major_locator(plt.MaxNLocator(10))  # Adjust the number of x-axis ticks
    plt.tight_layout()
    plt.show()


# Filter and plot data for the current day
filtered_timestamps_current, temp_1_current, temp_2_current, temp_3_current, temp_4_current = filter_data(current_day_start_time.date(), current_day_end_time.date())
if filtered_timestamps_current:
    create_custom_time_plot(
        filtered_timestamps_current, temp_1_current, temp_2_current, temp_3_current, temp_4_current,
        'Temperature Measurements for Current Day',
        current_day_start_time, current_day_end_time
    )
else:
    print("No data available for the current day.")

# Filter and plot data for the last 7 days
filtered_timestamps_7, temp_1_7, temp_2_7, temp_3_7, temp_4_7 = filter_data(last_7_days_start_time.date(), last_7_days_end_time.date())
if filtered_timestamps_7:
    create_custom_time_plot(
        filtered_timestamps_7, temp_1_7, temp_2_7, temp_3_7, temp_4_7,
        'Temperature Measurements for Last 7 Days',
        last_7_days_start_time, last_7_days_end_time
    )
else:
    print("No data available for the last 7 days.")

# Filter and plot data for the last 30 days
filtered_timestamps_30, temp_1_30, temp_2_30, temp_3_30, temp_4_30 = filter_data(last_30_days_start_time.date(), last_30_days_end_time.date())
if filtered_timestamps_30:
    create_custom_time_plot(
        filtered_timestamps_30, temp_1_30, temp_2_30, temp_3_30, temp_4_30,
        'Temperature Measurements for Last 30 Days',
        last_30_days_start_time, last_30_days_end_time
    )
else:
    print("No data available for the last 30 days.")
```

I Evaluation of Ohms at Load Voltage

Table I.1: Evaluation of Ohms as load voltage for the Geyserwise 72V.

#PV panels	1	2	3	4	5	6
Pmax [W]	260	520	780	1040	1300	1560
Voc [V]	38	76	114	152	190	228
Isc [A]	9,01	18,02	27,03	36,04	45,05	54,06
Vmpp [V]	30,92	61,84	92,76	123,68	154,6	185,52
Impp [A]	8,43	16,86	25,29	33,72	42,15	50,58

The element rating should be according to the mppt voltage at max, the controller says 72 V max load voltage
Load voltage ***72***

PV Configuration	Voltage [V]	Current [A]	Power [W]	Ohm at load voltage	Target Ohms [Ω]
1 serial	30,92	8,43	260,66	19,888	19,9
2 serial	61,84	8,43	521,31	9,944	9,9
3 serial	92,76	8,43	781,97	6,629	6,6
4 serial	123,68	8,43	1042,62	4,972	5,0
5 serial	154,60	8,43	1303,28	3,978	4,0
6 serial	185,52	8,43	1560,00	3,323	3,3
2 x 2	61,84	16,86	1042,62	4,972	5,0
2 x 3	92,76	16,86	1563,93	3,315	3,3

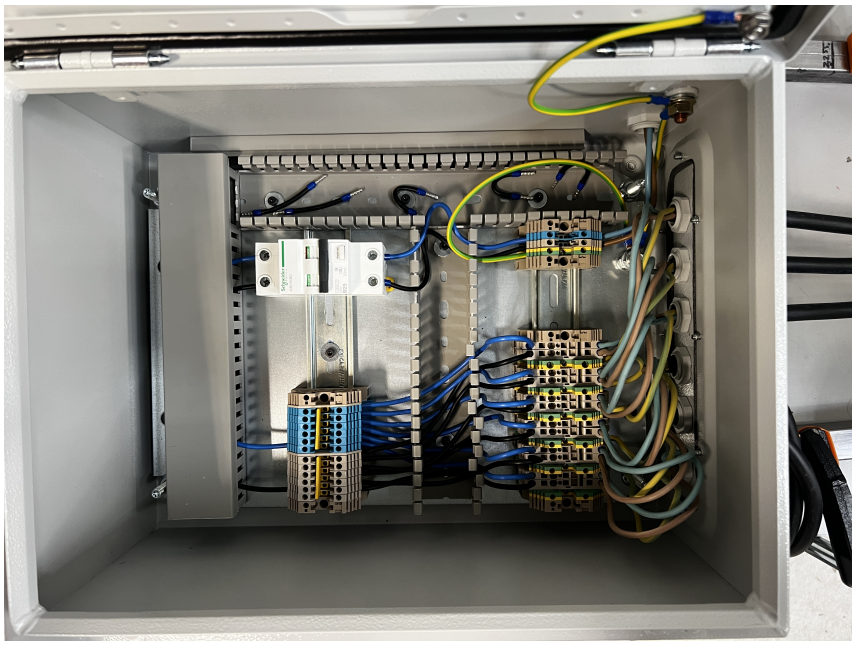
Table I.2: Evaluation of Ohms as load voltage for the Geyserwise 48V.

#PV Panels	1	2	3	4	5	6
Pmax [W]	260	520	780	1040	1300	1560
Voc [V]	38	76	114	152	190	228
Isc [A]	9,01	18,02	27,03	36,04	45,05	54,06
Vmpp [V]	30,92	61,84	92,76	123,68	154,6	185,52
Impp [A]	8,43	16,86	25,29	33,72	42,15	50,58

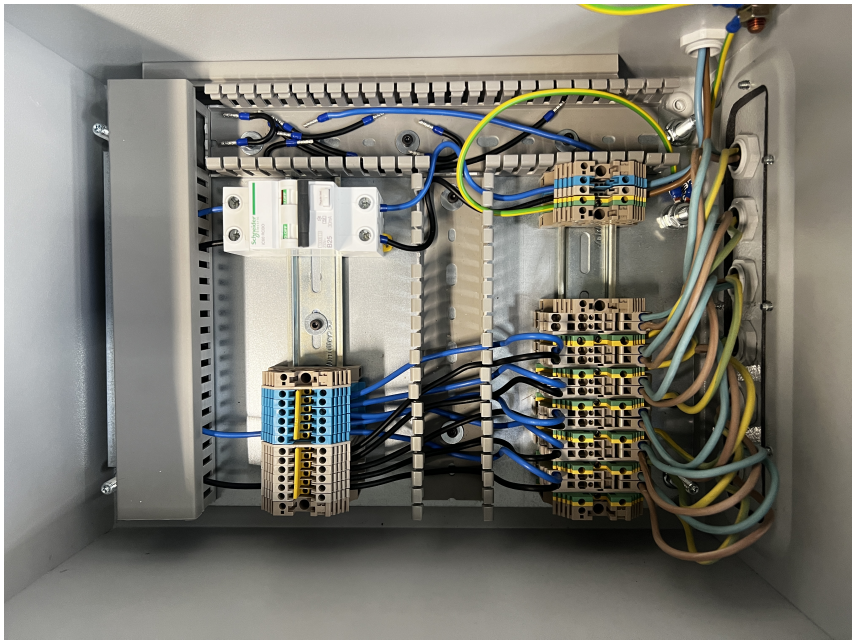
The element rating should be according to the mppt voltage at max, the controller says 72 V max load voltage
Load voltage **48**

PV Configuration	Voltage [V]	Current [A]	Power [W]	Ohm at load voltage	Target Ohms [Ω]
1 serial	30,92	8,43	260,66	8,839	8,8
2 serial	61,84	8,43	521,31	4,420	4,4
3 serial	92,76	8,43	781,97	2,946	2,9
4xserial	123,68	8,43	1042,62	2,210	2,2
5 serial	154,6	8,43	1303,28	1,768	1,8
6 serial	185,52	8,43	1560,00	1,477	1,5
2 x 2	61,84	16,86	1042,62	2,210	2,2
2 x 3	92,76	16,86	1563,93	1,473	1,5

J Resistances Circuited in Junction Box



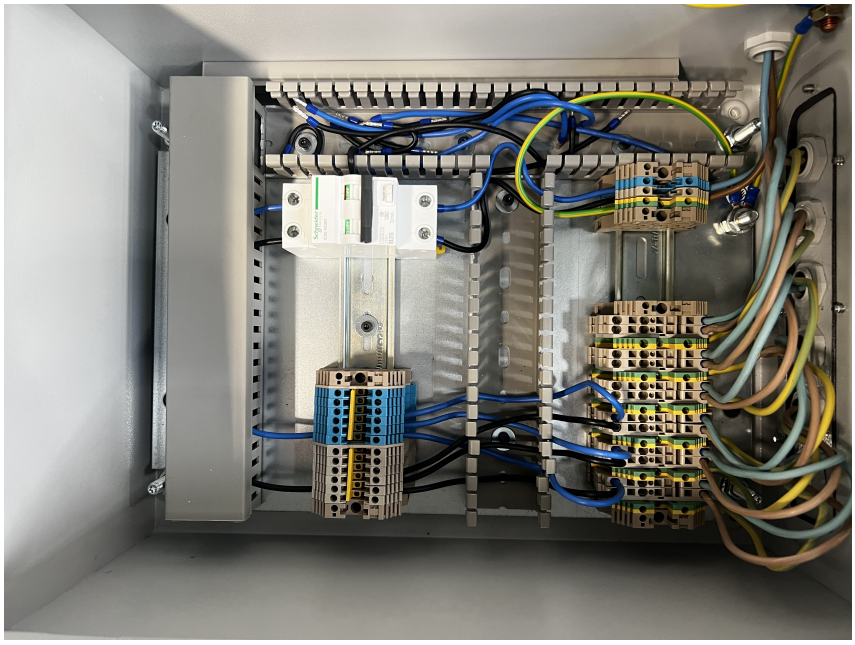
(a) Six parallel: 2.18 Ω



(b) Five parallel: 2.62 Ω

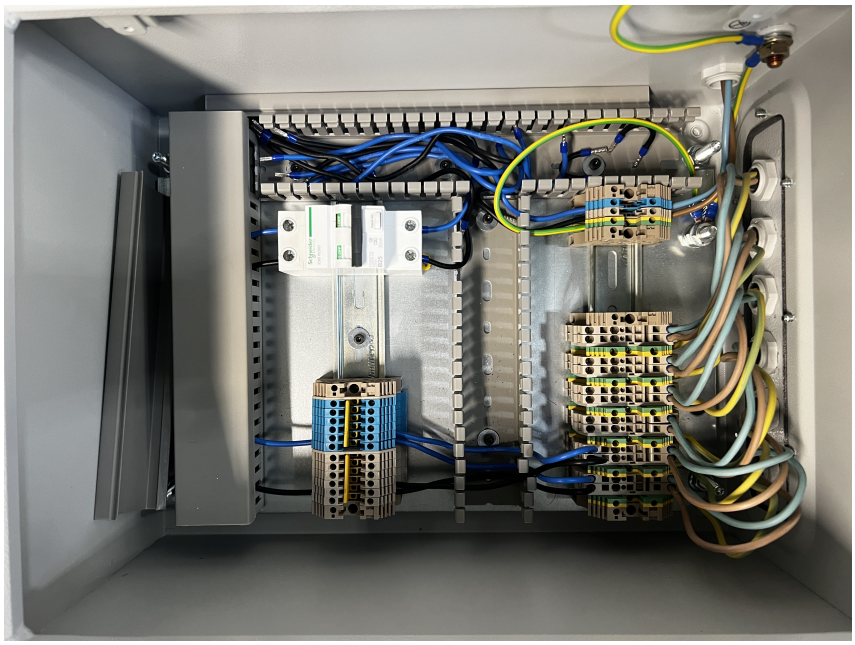
Figure J.1: Inside of the junction box to depict the circuits of the different resistances.

(c) Four parallel: 3.28 Ω

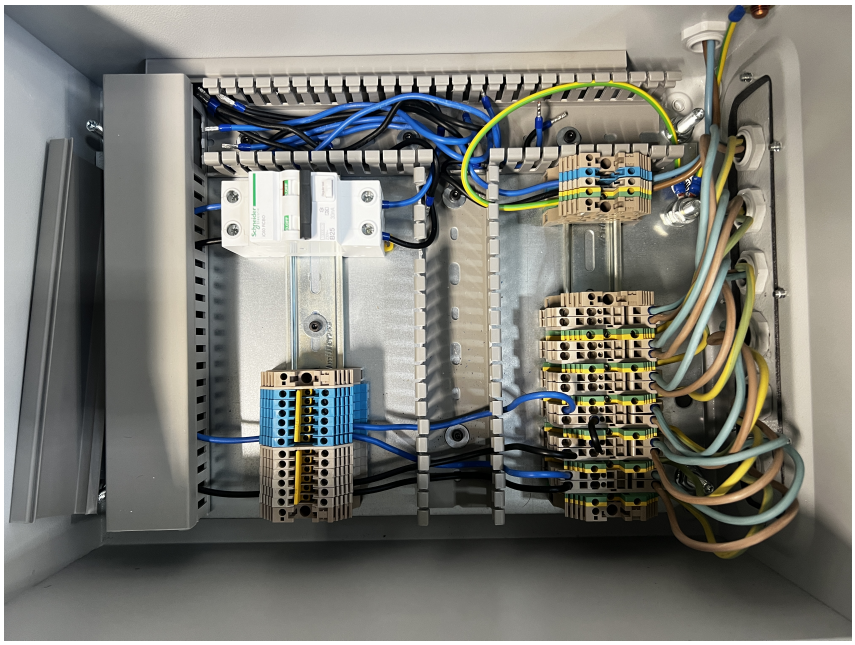


(d) Three parallel: 4.37 Ω

Figure J.1: Inside of the junction box to depict the circuits of the different resistances.

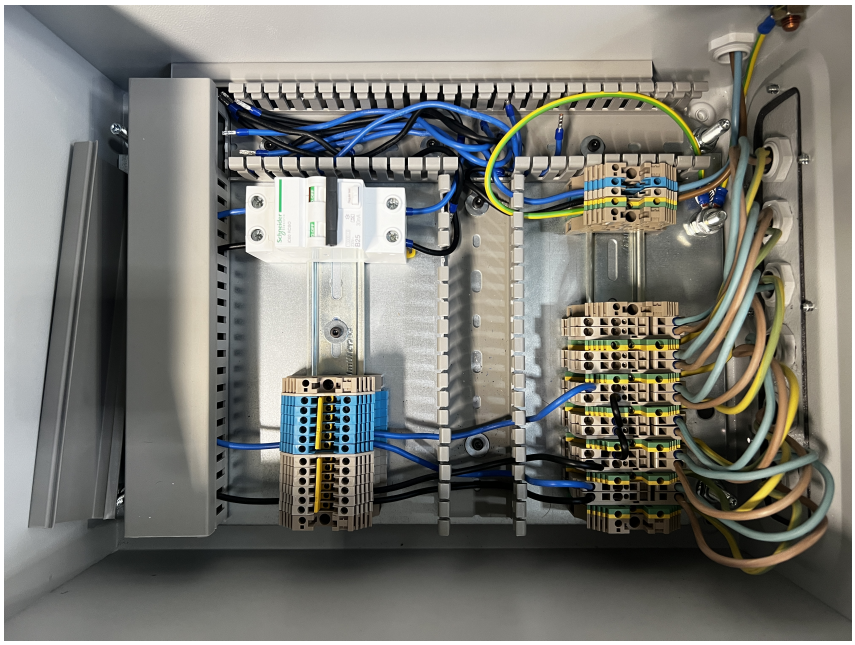


(e) Two parallel: 6.55 Ω



(f) Two parallel, with one parallel having two resistances in series: 8.73 Ω

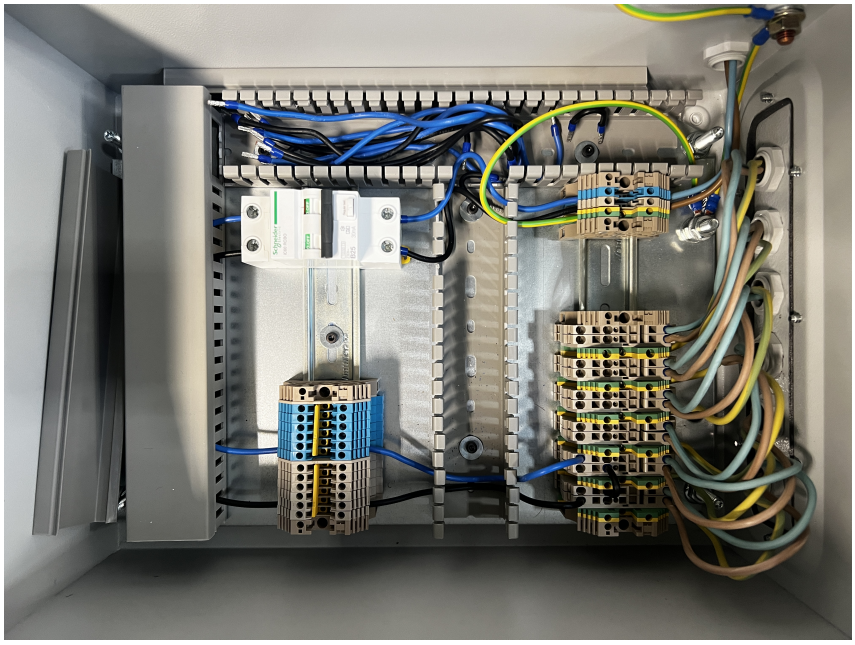
Figure J.1: Inside of the junction box to depict the circuits of the different resistances.



(g) Two parallel, with one parallel having three resistances in series: 9.83 Ω

(h) Two parallel, with each parallel having two resistances in series: 13.1 Ω

Figure J.1: Inside of the junction box to depict the circuits of the different resistances.



(i) Two series: 26.2 Ω

(j) Three series: 39.3 Ω

Figure J.1: Inside of the junction box to depict the circuits of the different resistances.

K Arduino IDE

Initial Arduino IDE program provided by the supervisor Ole Jørgen Nydal for data logging with the Arduino. Additional program integrated for the shunt and voltage divider circuit measurements, provided by the co-supervisor of the project work Mulu Bayray Kahsay.

```
// Temperature, Voltage and Current Logger: TVCLoggerSD_LCD_V2    Date: September 5, 2022
//   SD card file saved in CVS format every minute: date,hh,mm,T1,T2,T3,T4,V,I
//
// Temperature Lopgger: TempLoggerSD_Serial_V1    Date: July 5, 2022
// This is a temperature logger. Uses four Max38 and four k type thermocouple with a level shifter .
// Adafruit data logger shield is used for logging into an SD card and RTC for date and time.
// An LCD is connected to display the date, time and the four temperature values.
//
// Wiring Note:
// The temeprature values through the level shifter use one wire connection to pin 2 of the Arduino.
// The data logger shield is connected to Arduino pin 13, 12 and 11 for the SPI and pin 10 for the data
output to SD.
// The LCD can not share any of the above pins. Hence use the LCD (rs,e,db4,db5,db6,db7) to
(8,9,5,4,3,7) on Arduino pins.

// This serial monitor version is for testing the setup of the logger. Use the LCD version for actual
experiment setup.

//Libraries included

#include <DallasTemperature.h>
#include <OneWire.h>
#include <Wire.h>
#include <SPI.h>
#include <SD.h>
#include "RTClib.h"

//Definitions, where the various pins are connected, allocating memory to arrays and variables.
#define ONE_WIRE_BUS 2  // Data is connected to pin 2
#define SD_CS 10    // pin to write to data logger shield pin 10
#define TEMPERATURE_PRECISION 9
RTC_PCF8523 rtc;
char timeChar[100];
char dateChar[50];
char temperatureChar[10];
float temperature = 0;
float previousTemperature = 0;
String dateString;
int minuteNow=0;
int minutePrevious=0;

OneWire oneWire(ONE_WIRE_BUS); // Setup a oneWire instance to communicate with any OneWire devices (not
just Maxim/Dallas temperature ICs)
DeviceAddress insideThermometer; // arrays to hold device addresses
DallasTemperature sensors(&oneWire);
File myFile; //Creating empty file for the SD functions

# define ANALOG_IN_VOL A1 // defining ADC pin to A1 for Voltage
# define ANALOG_IN_CUR A0 // defining ADC pin to A0 for Current
```

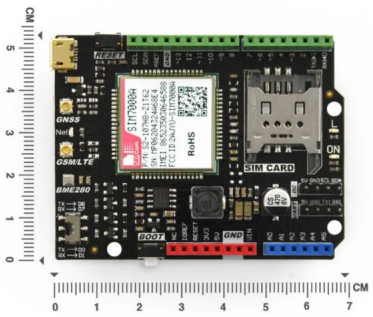
```
// Voltage divider parameters
float adc_v=0.0;
float in_v=0.0;
float R1=4750.0;
float R2=1000.0;
float ref_v=5.0;
float sensitivity=0.066;
int adc_value=0;


void setup(){
// This only runs one time after startup.

  pinMode(SD_CS, OUTPUT);
  Serial.begin(9600);
  Wire.begin();
  rtc.begin();
  sensors.begin();
  delay(1000); //1 second delay, to ensure that all processes has time to begin.

// This is only if battery was off to set the date and time again. Uncomment to adjust.
// Check your PC date and time is correctly set. It adjusts to the date and time of the PC.
rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));


  Serial.print("Initializing SD card...");

  // see if the card is present and can be initialized:
  if (!SD.begin(SD_CS)) {
    Serial.println("Card failed, or not present");
    // don't do anything more:
    while (1);
  }
  Serial.println("card initialized.");
// SD is successfully initialized.

// Find Thermometer adresses
  if (!sensors.getAddress(insideThermometer, 0))
  if (!sensors.getAddress(insideThermometer, 1))
  if (!sensors.getAddress(insideThermometer, 2))
  if (!sensors.getAddress(insideThermometer, 3))

  sensors.setResolution(insideThermometer, TEMPERATURE_PRECISION);  // set the resolution as defined.
}

void loop()
{
    // This runs continously in a loop.
```

```
// Opening file with the name data.txt
myFile=SD.open("DATA.txt",FILE_WRITE);
//Will create file if it doesn't exist
if(!myFile){
  Serial.println("Error opening data.txt file."); // Error message if opening fails.
  while(1); // Hold if opening fails.

}
 //Request temperature update from the sensors.
  sensors.requestTemperatures();

 //Get the current date-time
  DateTime now = rtc.now();
  minuteNow = now.minute();
  if (minuteNow!=minutePrevious){ //If the time has changed, update the string
    dateString = String(now.day())+"/"+String(now.month());
    dateString= dateString+"/"+ String(now.year());
    minutePrevious = minuteNow;
    String hours = String(now.hour());
    if(now.minute()<10) //If the minute counter is less than 10, add a 0 in front.
    {
      hours = hours+":0"+String(now.minute());
    }else
    {
      hours = hours+":"+String(now.minute());
    }

//Get latest date and time values.
    hours.toCharArray(timeChar,100);
    dateString.toCharArray(dateChar,100);
  }

// Saving date and time to SD card
    myFile.print(dateChar);
    myFile.print(",");
    myFile.print(timeChar);
    myFile.print(",");
// Display date and time to serial monitor
    Serial.print(dateChar);
    Serial.print(",");
    Serial.print(timeChar);
    Serial.print(",");

 for (uint8_t i = 0; i < 4; i++){
// Temperature values to SD card
//    myFile.print("  T");
//    myFile.print(i+1);
//    myFile.print(":");
    myFile.print(sensors.getTempCByIndex(i));
```

```
    myFile.print(",");
// Temperature values for display to serial monitor
//    Serial.print("  T");
//    Serial.print(i+1);
//    Serial.print(":");
    Serial.print(sensors.getTempCByIndex(i));
    Serial.print(",");
    }
    myFile.println(" ");
    Serial.println(" ");
myFile.close();  //Close the file.

delay(60000); //  Sixty second delay, to not flood the SD card with readings. Data will be saved every
sixty seconds.

// Voltage data from Voltage Divider
adc_value=analogRead(ANALOG_IN_VOL);
adc_v=adc_value*ref_v/1023.0;
in_v=adc_v/(R2/(R1+R2));
// Write volatge value to SD
myFile.print(" V:");
myFile.print(in_v,1);  //in_v in one decimal place
myFile.print(" V");
// Display voltage value to LCD
lcd.setCursor(0,3);
lcd.print("V=");
lcd.print(in_v,1);  //in_v in one decimal place
lcd.print(" V ");
// Current data from the GY sensor
unsigned int x=0;
float GYValue=0.0, Samples=0.0, AvgGY=0.0,GYValueF=0.0;
for (int x=0;x<100;x++){  // To get 100 samples
  GYValue=analogRead(ANALOG_IN_CUR); // Reading from pin A0
  Samples=Samples+GYValue; // Adding Samples
  delay (3);              // Let ADC settle for 3 ms
}
AvgGY=Samples/100.0;     // Taking average of the samples
GYValueF=(AvgGY*(5.0/1023.0)-2.5)/sensitivity;
// Write current value to SD
myFile.print(" I:");
myFile.print(GYValueF,1);
myFile.print(" A");
// Display current value to LCD
lcd.setCursor(11,3);
lcd.print("I=");
lcd.print(GYValueF,1);
lcd.print(" A");
}
```

L Evaluation of Web-based Arduino Solutions

Sim implementation to Arduino, for web-based data extraction:
1.
DFRobot Arduino NB-IoT Expansion Shield,
Price: 645,59 NOK (inc. VAT)
SIM7000 chip, GNSS, GSM, LTE
Supports SIM card and NB-IoT card
(NOTE: Please ensure the SIM frequency range being used is included in the frequency range it supports)
Control Via AT Commands
Input Voltage (recommended): 7 ~ 12V
Input Voltage (limits): 6 ~ 20V
Operating voltage: 5V
Example codes to http available



Input voltage on this shield is a problem for the Arduino system as the power supply from PV is put through a voltage regulator to send 5V to the Arduino main stack. No power supply between 7-12 V

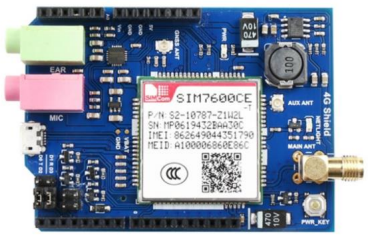
Requirements: Hardware:

DFR0216 DFRduino UNO R3 - Arduino Compatible x1 Vin = 7-12V
DFR0505 SIM7000 Arduino NB-IoT/LTE/GPRS Expansion Shield x1
USB Wire x1
7V~12V DC Power Supply x1
7V~12V DC Power Supply x1

Link: https://no.rs-online.com/web/p/shields-for-arduino/2473226?cm_mmc=NO-PLA-DS3A-_-google-_-CSS_NO_EN_Pmax_Test-_--_-2473226&matchtype=&gclid=Cj0KCQjw9fqnBhDSARIsAHlcQYQYZOiLep9bJ6k54LeoURF0fC3c4YgK-_CGs-CcsW-WqyGpsBPShPgaAtRwEALw_wcB&gclsrc=aw.ds
& https://docs.rs-online.com/ef9a/A700000008916367.pdf

Telia NB-IoT: https://business.teliacompany.com/internet-of-things/iot-connectivity/LPWA-IoT?gclid=CjwKCAjwu4WoBhBkEiwAojNdXi_Qn8bWzDsPiLdlV3rBQmsZCYt1vXEZ4tcNUczbbhVlb3uBiZ_muhoCtpIQAvD_BwE
& https://shop.business.teliacompany.com/s/iot-data-pool?language=en_US

Additional info: https://wiki.dfrobot.com/SIM7600CE-T_4G(LTE)_Shield_V1.0_SKU_TEL0124
& https://wiki.dfrobot.com/SIM7600G_H_LTE_Shield_V1.0_SKU_TEL0124
……
2.
Tinysine SIM7600CE
Price: $65
SIM7600CE chip
4G/GPRS specifications:
Dual-band 900MHz EGSM, 1800MHz DCS-connect onto any global GSM network with any 4G SIM.
Make and receive voice calls using a headset with a microphone
Send and receive SMS messages
Send and receive GPRS data (TCP / IP, HTTP, etc.)



Not much information, no notes on specifications regarding operating voltage or input voltage. Some example code available, but not necessarily relevant example code
Link:
https://www.tinyosshop.com/index.php?route=product/product&filter_name=sim7600&filter_description=true&filter_sub_category=true&product_id=1109
Schematic: https://www.tinyosshop.com/datasheet/4GShieldV1r1sch.pdf

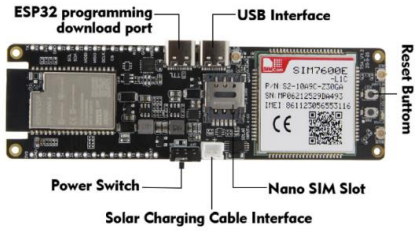
……

3. Bee Data Logger
Price: $29.99
Data logging shield without internet connection, mostly here due to the YouTube tutorial of connecting the shield to an MQTT broker. Not necessary when the Arduino stack already has a data logging shield.

Link: https://www.lectronz.com/products/esp32-data-logger
YouTube tutorial: https://www.youtube.com/watch?v=vbtSqaVAqRw

……

4. LILYGO® TTGO T-SIM7600E-L1C
Price: Approx. $40+shipping
SIM7600E chip (Europe, Middle East, Africa, South Korea, Thailand)
Example codes available, MQTT included
18650 battery connection as power backup
Solar charging interface, solar input: 4,4-6V
Interfaces: i2c, spi, uart, etc
No info on operating voltage but has 3.3V through holes.



Possibly the best solution but might only be possible to send data when PV power is available (dependant on battery). Power can be taken from the voltage regulator of the Arduino system which supplies 5V. Separate from the stack.

Link: https://tinyurl.com/566ua7jk
MQTT example: https://github.com/Xinyuan-LilyGO/T-SIM7600X/blob/master/examples/MqttClient/MqttClient.ino

……

5. TinySine SIM7600CE-T 4G(LTE) Shield

Price: $65+shipping

SIM7600CE chip

Stackable shield, large antenna included

Easiest solution when the module is stackable. Comes with a large antenna which is probably more powerful than those included with the other solutions. Communication over UART. Microphone and speaker ports included. No soldering required.

Link:https://www.tinyosshop.com/index.php?route=product/product&product_id=1109

……

Adafruit guide to MQTT (adafruit.io): https://learn.adafruit.com/mqtt-in-circuitpython/connecting-to-the-adafruit-io-mqtt-broker
More info: https://learn.adafruit.com/mqtt-adafruit-io-and-you/getting-started-on-adafruit-io

MQTT seem better for this application than HTTP:
Chances are you're familiar with HTTP - its used for every website. HTTP is stateless, so you have to have a connection per data transfer - one connection every time you want to write data, one connection for reading. HTTP is great for huge amounts of data such as used for websites, and it can be used for IoT connections. But it's not lightweight and its not terribly fast.
Another problem with HTTP is that it's pull only - your toaster can only send data to the server whenever it wants (e.g. "Toast is done!"). If it wants to pull data from the server, it has to constantly connect and ask ("Any updates to the Toast darkness level?" "What about now?" "Anything now?") which is really data and time consuming. Pull updates are either slow (check only every few minutes) or data/power intensive (check constantly)

M TinySine SIM7600 Schematics

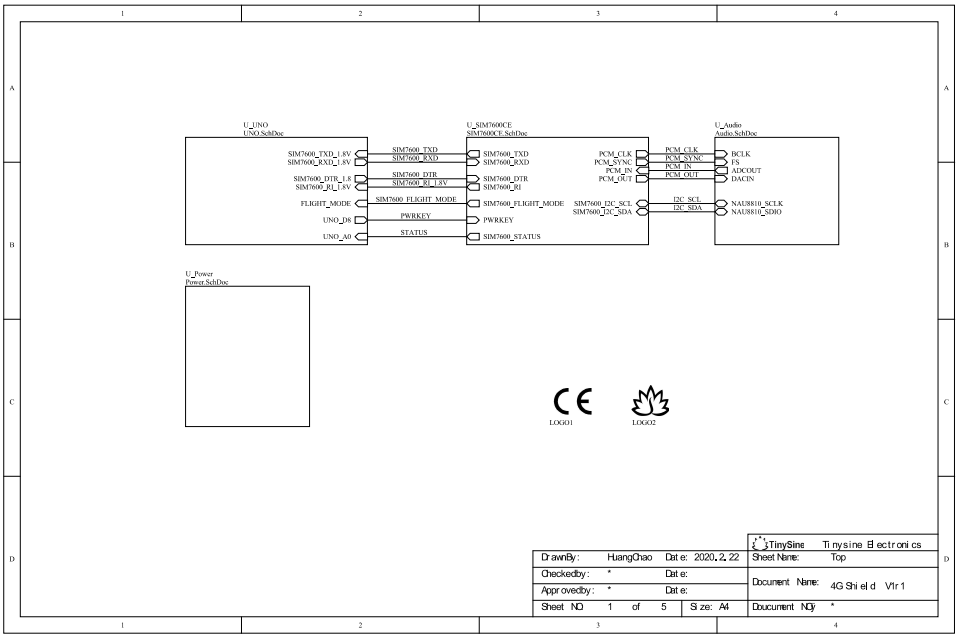


Figure M.1: Pinouts between an Arduino Uno and the TinySine SIM7600 shield.

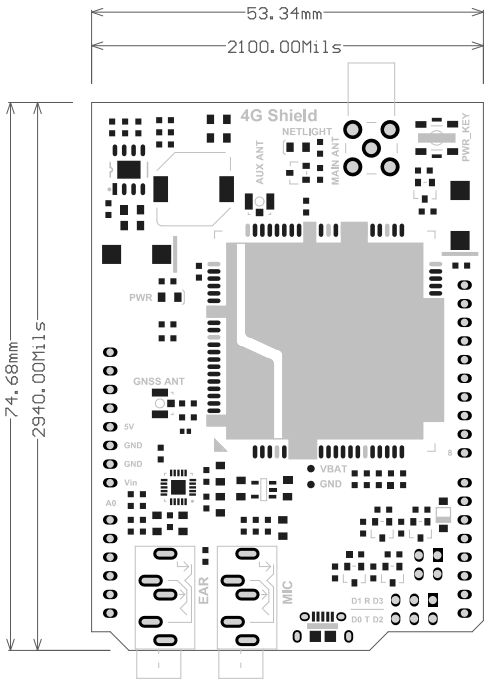


Figure M.2: Topology of the TinySine SIM7600 shield

Full datasheet of the TinySine SIM7600 shield can be found at: TinySine SIM7600 Datasheet.

N Geyserwise 72V – Tabulated Data

Table N.1: 72V controller experimental data.

Resistance: [Ω]	**Solar Irradiance:** [W/m^2]	**PV$_{in}$:** [V]	**P$_{out}$:** [W]	**U$_{out}$:** [V]	**I$_{out}$:** [A]
2,62	400	111	722	42,8	17,3
3,28	205	88	192	24,6	8,1
	388	110	657	45,6	14,7
	398	108	758	48,1	15,8
	402	105	758	48,8	15,8
	434	109	810	50,7	16,3
	446	115	856	51	16,8
4,37	151	86,0	44,0	13,1	3,5
	156	104,0	56,0	14,8	4,0
	195	110,0	165,0	25,0	6,6
	398	112,0	748,0	55,0	13,7
	410	120,0	748,0	55,0	13,7
	511	121,0	748,0	55,0	13,7
	556	121,0	748,0	55,1	13,7
6,55	105	76	28	12,7	2,5
	160	116,0	140,0	28,3	5,1
	185	124,0	92,0	24,0	4,0
	202	121,0	96,0	24,1	4,0
	396	120,0	486,0	54,5	9,1
	503	129,0	495,0	54,8	9,1
8,73	105	114,0	85,0	25,8	3,5
	146	95,0	91,0	27,6	3,5
	156	93,0	60,0	20,5	3,0
	169	125,0	63,0	21,2	3,0
	206	113,0	116,0	29,2	4,0
	405	133,0	378,0	54,7	7,1
	491	132,0	378,0	54,9	7,1
9,83	180	116	95	28,5	3,5
13,1	95	95,0	67,0	28,4	2,5
	165	120,0	122,0	36,7	3,5
	202	117,0	180,0	45,7	4,0
	370	135,0	275,0	55,2	5,1
	402	135,0	270,0	54,8	5,1
	508	134,0	275,0	55,0	5,1
26,2	174	99,0	80,0	40,1	2,0

O Odin – Tabulated Data

Table O.1: Odin controller experimental data.

		Readings from LabVIEW					Adjusted by:	2/3
Resistance:	**Solar Irradiance:**	**PWM:**	**P_{in}:**	**T:**	**U_{in}:**	**I_{in}:**	**$P_{in,ad}$:**	**$U_{in,ad}$:**
$[\Omega]$	$[W/m^2]$	$[\%]$	$[W]$	$[°C]$	$[V]$	$[A]$	$[W]$	$[V]$
2,62	305	14,3	586,5	43,1	165,3	3,5	391,0	110,2
	320	15,5	606,2	41,8	161,9	3,7	404,1	107,9
	349	18,9	634,9	38,5	152,3	4,2	423,3	101,5
	375	15,9	625,5	43,3	165,2	3,8	417,0	110,1
3,28	360	22,9	737,6	36,7	158,4	4,7	491,7	105,6
	372	20,4	700,7	40	160,4	4,4	467,1	106,9
4,37	305	25,4	780,1	34,3	158,0	4,9	520,1	105,3
	370	29,6	856,4	37,1	159,2	5,4	570,9	106,1
	374	23,1	805,7	37,2	172,7	4,7	537,1	115,1
6,55	345	38,4	1026,8	31,9	165,8	6,2	684,5	110,5
	378	40,1	1046,5	37,9	164,4	6,4	697,7	109,6
8,73	125	17,5	670,3	31,3	167,5	4	446,9	111,7
	152	18,0	731,8	31,5	180,3	4,1	487,9	120,2
	173	18,8	766,4	31,7	188,2	4,1	510,9	125,5
	248	25,3	920,5	32,2	188,3	4,9	613,7	125,5
	308	31,4	1046,6	32,5	190,3	5,5	697,7	126,9
	323	34,4	1061,5	28,4	182,3	5,8	707,7	121,5
	357	21,2	886,5	33,8	198,9	4,5	591,0	132,6
	367	39,9	1126,4	32,6	178,0	6,3	750,9	118,7
	369	23,6	938,3	34,0	197,6	4,8	625,5	131,7
	386	22,9	928,6	34,7	198,2	4,7	619,1	132,1
	390	45,1	1204,1	37,4	178,0	6,8	802,7	118,7
	391	33,3	1085,7	35,5	189,8	5,7	723,8	126,5
13,1	200	33,8	922,0	35,8	190,2	5,7	614,7	126,8
	216	34,9	888,3	35,7	179,3	5,1	592,2	119,5
	253	35,8	1077,2	35,8	190,2	5,7	718,1	126,8
	300	43,1	1175,8	30,8	188,9	6,2	783,9	125,9
	347	43,0	1187,3	35,6	188,9	6,4	791,5	125,9
	347	47,3	1220,2	35,1	187,3	6,5	813,5	124,9
	369	45,1	1202,4	35,4	188,9	6,4	801,6	125,9
	390	46,2	1255,6	36,0	191,4	6,6	837,1	127,6
26,2	302	74,8	761,7	30,8	190,1	4,0	507,8	126,7
	373	81,6	804,5	38,3	190,8	4,2	536,3	127,2

Table O.2: Odin controller data for tests with multimeter.

	Readings from LabVIEW						Adjusted by: 2/3		Multimeter		
Resistance: $[W]$	**Solar Irradiance:** $[W/m^2]$	**PWM:** [%]	$\mathbf{P}_{in}$**:** $[W]$	**T:** $[°C]$	$\mathbf{U}_{in}$**:** $[V]$	$\mathbf{I}_{in}$**:** $[A]$	$\mathbf{P}_{in,\,ad}$**:** $[W]$	$\mathbf{U}_{in,\,ad}$**:** [V]	$\mathbf{I}_{out}$**:** $[A]$	$\mathbf{U}_{out}$**:** $[V]$	$\mathbf{P}_{out}$**:** $[W]$
2,18	150,0	15,2	612,8	34,4	165,5	3,7	408,53	110,33	6,498	15,65	101,69
	165,0	15,3	608,6	36,6	164,1	3,7	405,73	109,40	6,392	15,99	102,21
	188,0	13,7	631,5	41,8	181,8	3,5	421,00	121,20	6,412	15,52	99,51
2,62	149,0	14,8	663,9	31,8	181,8	3,6	442,60	121,20	5,981	16,98	101,56
	165,0	16,2	662,4	34,4	179,0	3,8	441,60	119,33	6,200	17,57	108,93
	189,0	16,5	688,1	43,1	178,7	3,9	458,73	119,13	6,522	18,55	120,98
3,28	146,0	21,2	749,8	28,5	166,8	4,5	499,87	111,20	6,445	22,46	144,75
	164,0	20,6	746,3	33,2	170,4	4,4	497,53	113,60	6,308	22,00	138,78
	190,0	22,3	772,5	43,6	169,6	4,5	515,00	113,07	6,770	23,63	159,98
4,37	144,0	21,6	838,8	26,9	186,9	4,5	559,20	124,60	5,671	25,91	146,94
	163,0	19,5	803,6	31,4	190,5	4,2	535,73	127,00	5,217	23,92	124,79
	187,0	26,4	897,4	31,9	178,7	5,0	598,27	119,13	6,630	30,37	201,35
6,55	144,0	30,4	1029,3	24,5	188,2	5,5	686,20	125,47	5,530	37,23	205,88
	163,0	27,5	985,4	27,6	191,5	5,1	656,93	127,67	5,091	34,17	173,96
	190,0	27,9	1005,4	39,8	194,7	5,2	670,27	129,80	5,237	35,23	184,50
8,73	136,0	26,9	1028,1	21,9	203,1	5,1	685,40	135,40	4,008	35,36	141,72
	162,0	33,0	1115,0	26,5	194,6	5,7	743,33	129,73	4,719	41,82	197,35
	191,0	46,3	1251,8	38,0	182,0	6,9	834,53	121,33	6,230	55,27	344,33
13,1	140,0	46,1	1349,5	23,3	198,5	6,8	899,67	132,33	4,581	60,50	277,15
	162,0	46,6	1326,7	25,4	195,9	6,8	884,47	130,60	4,561	60,38	275,39
	190,0	51,4	1407,4	35,7	197,0	7,1	938,27	131,33	5,037	66,70	335,97
26,2	139,0	76,7	890,7	22,4	203,4	4,4	593,80	135,60	3,965	103,60	410,77
	160,0	78,4	868,9	25,1	199,8	4,3	579,27	133,20	3,997	104,20	416,49
	191,0	74,4	889,7	33,7	205,6	4,3	593,13	137,07	3,882	101,60	394,41
39,3	133,0	100,0	696,5	20,6	206,0	3,4	464,33	137,33	3,508	137,30	481,65
	159,0	100,0	676,3	24,9	203,1	3,3	450,87	135,40	3,454	135,10	466,64
	190,0	100,0	691,6	30,6	205,7	3,4	461,07	137,13	3,503	137,00	479,91

P Geyserwise 48V – Tabulated Data

Table P.1: 48V controller experimental data.

Resistance: [Ω]	**Date**	**Solar Irradiance:** $[W/m^2]$	**PV** $_{in}$**:** $[V]$	**P**$_{out}$**:** $[W]$	**U**$_{out}$**:** $[V]$	**I**$_{out}$**:** $[A]$
2,18	02.10.2023	92	56	146	17,7	8,6
	02.10.2023	156	57	114	15,4	7,6
	02.10.2023	204	63	567	35,1	16,3
	29.09.2023	206	52	233	22,1	10,7
	29.09.2023	207	58	257	23,4	11,2
	02.10.2023	208	60	604	36,1	16,8
	02.10.2023	212	55	588	35,4	16,8
	29.09.2023	216	53	305	25,6	12,2
	02.10.2023	261	54	636	37,2	17,3
	02.10.2023	262	51	619	36,4	17,3
	02.10.2023	280	58	438	30,9	14,7
	02.10.2023	282	56	691	38,5	18,3
	29.09.2023	289	58	394	29,3	13,7
2,62	02.10.2023	109	39	136	18,1	7,6
	02.10.2023	117	56	180	21,2	8,6
	02.10.2023	131	56	255	25,1	10,2
	02.10.2023	196	54	403	32,3	12,7
	29.09.2023	205	60	230	24,2	9,6
	29.09.2023	217	46	275	26	10,7
	02.10.2023	280	60	664	41,1	16,3
	02.10.2023	299	68	540	37,3	14,7
	02.10.2023	312	55	819	45,7	18,3
	29.09.2023	315	55	462	34,1	13,7
	02.10.2023	330	44	705	42	16,8
3,28	02.10.2023	120	57	192	24,9	7,6
	02.10.2023	136	51	140	20,9	7,1
	02.10.2023	146	49	278	29,2	9,6
	02.10.2023	200	59	463	38,1	12,2
	29.09.2023	202	56	200	25,4	8,1
	29.09.2023	209	58	243	27,6	9,1
	29.09.2023	216	56	192	24,4	8,1
	02.10.2023	271	55	671	46,3	14,7
	29.09.2023	310	56	478	38,5	12,7
	29.09.2023	320	56	463	37,9	12,2
4,37	02.10.2023	140	57,0	144,0	23,9	6,1

Resistance: [Ω]	**Date**	**Solar Irradiance:** [W/m^2]	**PV** $_{in}$**:** [V]	**P**$_{out}$**:** [W]	**U**$_{out}$**:** [V]	**I**$_{out}$**:** [A]
	29.09.2023	209	57	144	243	6,1
	29.09.2023	229	56,0	228,0	30,0	7,6
	02.10.2023	258	56,0	699,0	53,2	13,2
	02.10.2023	292	56,0	418,0	41,7	10,2
	29.09.2023	306	60,0	466,0	44,0	10,7
	02.10.2023	381	67,0	492,0	44,9	11,2
6,55	29.09.2023	232	56,0	342,0	45,8	7,6
	02.10.2023	253	65,0	495,0	55,0	9,1
	29.09.2023	264	60,0	334,0	44,5	7,6
	29.09.2023	271	58	384	48,3	8,1
	29.09.2023	320	61	495	55	9,1
8,73	29.09.2023	200	51,0	240,0	43,3	5,6
	29.09.2023	205	58,0	270,0	55,1	6,1
	29.09.2023	233	59,0	343,0	52,0	6,6
	02.10.2023	257	67,0	385,0	55,1	7,1
	29.09.2023	297	65	385	55	7,1
	29.09.2023	302	65,0	385,0	55,1	7,1
13,1	29.09.2023	193	57	237	54,5	4,5
	02.10.2023	261	69	275	55,1	5,1
	29.09.2023	290	67	242	55	4,5
	29.09.2023	301	67	275	55,2	5,1
26,2	29.09.2023	223	69	132	55	2,5
	29.09.2023	285	69	129	54,4	2,5
	02.10.2023	294	70	132	55	3,5

Q Initial Arduino IDE program for a web-based solution

The first rendition of a completed program in Arduino IDE for the implementation of a TinySine SIM7600 shield.

```
// Temperature, Voltage and Current Logger: TVCLoggerSD_LCD_V2    Date:
September 5, 2022
//   SD card file saved in CVS format every minute: date,hh,mm,T1,T2,T3,T4,V,I
//
// Temperature Lopgger: TempLoggerSD_Serial_V1    Date: July 5, 2022
// This is a temperature logger. Uses four Max38 and four k type thermocouple
with a level shifter .
// Adafruit data logger shield is used for logging into an SD card and RTC for
date and time.
// An LCD is connected to display the date, time and the four temperature
values.
//
// Wiring Note:
// The temeprature values through the level shifter use one wire connection to
pin 2 of the Arduino.
// The data logger shield is connected to Arduino pin 13, 12 and 11 for the
SPI and pin 10 for the data output to SD.
// The LCD can not share any of the above pins. Hence use the LCD
(rs,e,db4,db5,db6,db7) to (8,9,5,4,3,7) on Arduino pins.

// This serial monitor version is for testing the setup of the logger. Use the
LCD version for actual experiment setup.

//Libraries included

#include <DallasTemperature.h>
#include <OneWire.h>
#include <Wire.h>
#include <SPI.h>
#include <SD.h>
#include "RTClib.h"
#include <TinyGsmClient.h>
#include <PubSubClient.h>

//Definitions, where the various pins are connected, allocating memory to
arrays and variables.
#define ONE_WIRE_BUS 2  // Data is connected to pin 2
#define SD_CS 10    // pin to write to data logger shield pin
10
#define TEMPERATURE_PRECISION 9
RTC_PCF8523 rtc;
char timeChar[100];
char dateChar[50];
char temperatureChar[10];
float temperature = 0;
float previousTemperature = 0;
String dateString;
int minuteNow=0;
int minutePrevious=0;
```

```
OneWire oneWire(ONE_WIRE_BUS); // Setup a oneWire instance to communicate with
any OneWire devices (not just Maxim/Dallas temperature ICs)
DeviceAddress insideThermometer; // arrays to hold device addresses
DallasTemperature sensors(&oneWire);
File myFile; //Creating empty file for the SD functions

// TinySine SIM7600 settings
TinyGsm modem(Serial1);
TinyGsmClient client(modem);
PubSubClient mqtt(client);

const char* apn = "telenor.m2m"; // Contact your cellular provider for the
correct APN
const char* mqtt_server = "mqtt.mosquitto.org";
const char* mqtt_username = " ";
const char* mqtt_password = " ";
const char* mqtt_topic = "ArduinoTest/Martin/Data";

//Shunt and VDC

# define ANALOG_IN_VOL A1 // defining ADC pin to A1 for Voltage
# define ANALOG_IN_CUR A2 // defining ADC pin to A2 for Current
// Voltage divider parameters
float adc_v=0.0;
float in_v=0.0;
float R1=1200.0;
float R2=150.0;
float ref_v=5.0;
float sensitivity=0.066;
int adc_value=0;


void setup(){
// This only runs one time after startup.

  pinMode(SD_CS, OUTPUT);
  Serial.begin(9600);
  Wire.begin();
  rtc.begin();
  sensors.begin();
  delay(1000); //1 second delay, to ensure that all processes has time to
begin.

  // Initialize SIM7600
  Serial1.begin(115200);
  modem.restart();
  if (!modem.gprsConnect(apn)) {
    Serial.println("Failed to connect to GSM network");
```

```
    while (1);
  }

  // Initialize MQTT
  mqtt.setServer(mqtt_server, 1883);
  mqtt.setCallback(mqttCallback);

// This is only if battery was off to set the date and time again. Uncomment
to adjust.
// Check your PC date and time is correctly set. It adjusts to the date and
time of the PC.
rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));


  Serial.print("Initializing SD card...");

  // see if the card is present and can be initialized:
  if (!SD.begin(SD_CS)) {
    Serial.println("Card failed, or not present");
    // don't do anything more:
    while (1);
  }
  Serial.println("card initialized.");
// SD is successfully initialized.

// Find Thermometer adresses
  if (!sensors.getAddress(insideThermometer, 0))
  if (!sensors.getAddress(insideThermometer, 1))
  if (!sensors.getAddress(insideThermometer, 2))
  if (!sensors.getAddress(insideThermometer, 3))

  sensors.setResolution(insideThermometer, TEMPERATURE_PRECISION);  // set the
resolution as defined.
}


void loop()
{
    // This runs continously in a loop.

// Opening file with the name data.txt
myFile=SD.open("DATA.txt",FILE_WRITE);
//Will create file if it doesn't exist
if(!myFile){
  Serial.println("Error opening data.txt file."); // Error message if opening
fails.
  while(1); // Hold if opening fails.

}
```

```
 //Request temperature update from the sensors.
  sensors.requestTemperatures();

 //Get the current date-time
  DateTime now = rtc.now();
  minuteNow = now.minute();
  if (minuteNow!=minutePrevious){ //If the time has changed, update the string
    dateString = String(now.day())+"/"+String(now.month());
    dateString= dateString+"/"+ String(now.year());
    minutePrevious = minuteNow;
    String hours = String(now.hour());
    if(now.minute()<10) //If the minute counter is less than 10, add a 0 in
front.
    {
      hours = hours+":0"+String(now.minute());
    }else
    {
      hours = hours+":"+String(now.minute());
    }

//Get latest date and time values.
    hours.toCharArray(timeChar,100);
    dateString.toCharArray(dateChar,100);
  }

// Saving date and time to SD card
    myFile.print(dateChar);
    myFile.print(",");
    myFile.print(timeChar);
    myFile.print(",");
// Display date and time to serial monitor
    Serial.print(dateChar);
    Serial.print(",");
    Serial.print(timeChar);
    Serial.print(",");

 for (uint8_t i = 0; i < 4; i++){
// Temperature values to SD card
//    myFile.print("  T");
//    myFile.print(i+1);
//    myFile.print(":");
    myFile.print(sensors.getTempCByIndex(i));
    myFile.print(",");
// Temperature values for display to serial monitor
//    Serial.print("  T");
//    Serial.print(i+1);
//    Serial.print(":");
```

```
    Serial.print(sensors.getTempCByIndex(i));
    Serial.print(",");
    }
    myFile.println(" ");
    Serial.println(" ");
myFile.close();  //Close the file.

publishDataToMQTT(dateString);
delay(60000); //  Sixty second delay, to not flood the SD card with readings.
Data will be saved every sixty seconds.


// Voltage data from Voltage Divider
//adc_value=analogRead(ANALOG_IN_VOL);
//adc_v=adc_value*ref_v/1023.0;
//in_v=adc_v/(R2/(R1+R2));
// Write volatge value to SD
//myFile.print(" V:");
//myFile.print(in_v,1);  //in_v in one decimal place
//myFile.print(" V");
// Display voltage value to LCD
//lcd.setCursor(0,3);
//lcd.print("V=");
//lcd.print(in_v,1);  //in_v in one decimal place
//lcd.print(" V ");
// Current data from the GY sensor
//unsigned int x=0;
//float GYValue=0.0, Samples=0.0, AvgGY=0.0,GYValueF=0.0;
//for (int x=0;x<100;x++){  // To get 100 samples
//  GYValue=analogRead(ANALOG_IN_CUR); // Reading from pin A0
//  Samples=Samples+GYValue; // Adding Samples
//  delay (3);              // Let ADC settle for 3 ms
//}
//AvgGY=Samples/100.0;     // Taking average of the samples
//GYValueF=(AvgGY*(5.0/1023.0)-2.5)/sensitivity;
// Write current value to SD
//myFile.print(" I:");
//myFile.print(GYValueF,1);
//myFile.print(" A");
// Display current value to LCD
//lcd.setCursor(11,3);
//lcd.print("I=");
//lcd.print(GYValueF,1);
//lcd.print(" A");

}

void publishDataToMQTT(String dataLine) {
  if (mqtt.connected()) {
```

```
    mqtt.publish(ArduinoTest/Martin/Data, dataLine.c_str());
  }
}

//void mqttCallback(char* topic, byte* payload, unsigned int length) {
  // Handle MQTT messages if needed
//}
```

R Arduino IDE program for a web-based solution using AT commands

Final attempt at an Arduino IDE program using AT commands for the implementation of a web-based Arduino.

```
// Temperature, Voltage and Current Logger: TVCLoggerSD_LCD_V2    Date:
September 5, 2022
//   SD card file saved in CVS format every minute: date,hh,mm,T1,T2,T3,T4,V,I
//
// Temperature Lopgger: TempLoggerSD_Serial_V1    Date: July 5, 2022
// This is a temperature logger. Uses four Max38 and four k type thermocouple
with a level shifter .
// Adafruit data logger shield is used for logging into an SD card and RTC for
date and time.
// An LCD is connected to display the date, time and the four temperature
values.
//
// Wiring Note:
// The temeprature values through the level shifter use one wire connection to
pin 2 of the Arduino.
// The data logger shield is connected to Arduino pin 13, 12 and 11 for the
SPI and pin 10 for the data output to SD.
// The LCD can not share any of the above pins. Hence use the LCD
(rs,e,db4,db5,db6,db7) to (8,9,5,4,3,7) on Arduino pins.

// This serial monitor version is for testing the setup of the logger. Use the
LCD version for actual experiment setup.

//Libraries included

#include <DallasTemperature.h>
#include <OneWire.h>
#include <Wire.h>
#include <SPI.h>
#include <SD.h>
#include "RTClib.h"
#include <PubSubClient.h>

SoftwareSerial sim7600(0, 1); // Software Serial pins for SIM7600 (RX, TX)

//Definitions, where the various pins are connected, allocating memory to
arrays and variables.
#define ONE_WIRE_BUS 2  // Data is connected to pin 2
#define SD_CS 10    // pin to write to data logger shield pin
10
#define TEMPERATURE_PRECISION 9
RTC_PCF8523 rtc;
char timeChar[100];
char dateChar[50];
char temperatureChar[10];
float temperature = 0;
float previousTemperature = 0;
String dateString;
int minuteNow=0;
```

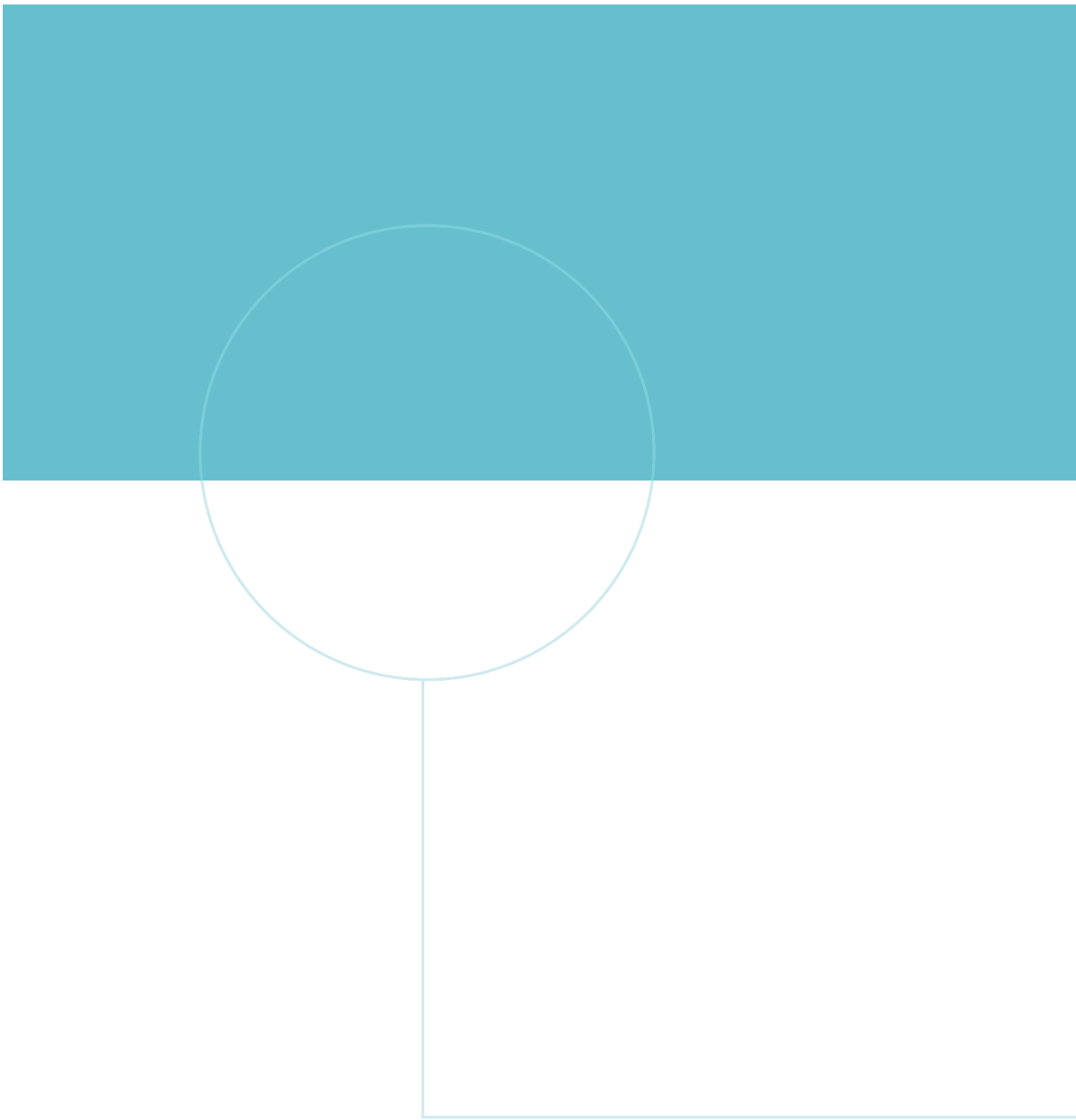
```
int minutePrevious=0;

OneWire oneWire(ONE_WIRE_BUS); // Setup a oneWire instance to communicate with
any OneWire devices (not just Maxim/Dallas temperature ICs)
DeviceAddress insideThermometer; // arrays to hold device addresses
DallasTemperature sensors(&oneWire);
File myFile; //Creating empty file for the SD functions


//MQTT Broker Details
const char* apn = "telenor.m2m"; // Contact your cellular provider for the
correct APN
const char* mqtt_server = "mqtt.mosquitto.org";
const char* mqtt_username = " ";
const char* mqtt_password = " ";
const char* mqtt_topic = "ArduinoTest/Martin/Data";

//Shunt and VDC

# define ANALOG_IN_VOL A1 // defining ADC pin to A1 for Voltage
# define ANALOG_IN_CUR A2 // defining ADC pin to A2 for Current
// Voltage divider parameters
float adc_v=0.0;
float in_v=0.0;
float R1=1200.0;
float R2=150.0;
float ref_v=5.0;
float sensitivity=0.066;
int adc_value=0;


void setup(){
// This only runs one time after startup.

  pinMode(SD_CS, OUTPUT);
  Serial.begin(9600);
  sim7600.begin(9600);
  Wire.begin();
  rtc.begin();
  sensors.begin();
  delay(1000); //1 second delay, to ensure that all processes has time to
begin.



// This is only if battery was off to set the date and time again. Uncomment
to adjust.
// Check your PC date and time is correctly set. It adjusts to the date and
time of the PC.
```

```
rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));


  Serial.print("Initializing SD card...");

  // see if the card is present and can be initialized:
  if (!SD.begin(SD_CS)) {
    Serial.println("Card failed, or not present");
    // don't do anything more:
    while (1);
  }
  Serial.println("card initialized.");
// SD is successfully initialized.

// Find Thermometer adresses
  if (!sensors.getAddress(insideThermometer, 0))
  if (!sensors.getAddress(insideThermometer, 1))
  if (!sensors.getAddress(insideThermometer, 2))
  if (!sensors.getAddress(insideThermometer, 3))

  sensors.setResolution(insideThermometer, TEMPERATURE_PRECISION);  // set the
resolution as defined.

  delay(2000);

  Serial.println("Initializing SIM7600...");

  // Check if the module is responsive
  if (!sendATCommand("AT")) {
    Serial.println("Error communicating with SIM7600");
    while (1);
  }

  // Set APN for your cellular network provider
  if (!sendATCommand("AT+CGDCONT=1,\"IP\",\"telenor.m2m\"")) {
    Serial.println("Error setting APN");
    while (1);
  }

  // Connect to GPRS
  if (!sendATCommand("AT+CGATT=1")) {
    Serial.println("Error connecting to GPRS");
    while (1);
  }

  // Start TCP connection with MQTT broker
  if (!sendATCommand("AT+CSOCKSETPN=1,\"telenor.m2m\"")) {
    Serial.println("Error setting PDP context");
    while (1);
```

```
  }

  if (!sendATCommand("AT+CSOCKAUTH=0,1,\" \",\"\"")) {  // Username and
password
    Serial.println("Error setting MQTT username and password");
    while (1);
  }

  // MQTT Broker Host and Port
  if (!sendATCommand("AT+CSOCKCONN=0,\"TCP\",\"test.mosquitto.org\",1883")) {
    Serial.println("Error connecting to MQTT broker");
    while (1);
  }

  Serial.println("Connected to MQTT broker");

  delay(1000);
}


void loop()
{
    // This runs continously in a loop.

// Opening file with the name data.txt
myFile=SD.open("DATA.txt",FILE_WRITE);
//Will create file if it doesn't exist
if(!myFile){
  Serial.println("Error opening data.txt file."); // Error message if opening
fails.
  while(1); // Hold if opening fails.

}


 //Request temperature update from the sensors.
  sensors.requestTemperatures();

 //Get the current date-time
  DateTime now = rtc.now();
  minuteNow = now.minute();
  if (minuteNow!=minutePrevious){ //If the time has changed, update the string
    dateString = String(now.day())+"/"+String(now.month());
    dateString= dateString+"/"+ String(now.year());
    minutePrevious = minuteNow;
    String hours = String(now.hour());
    if(now.minute()<10) //If the minute counter is less than 10, add a 0 in
front.
    {
```

```
      hours = hours+":0"+String(now.minute());
    }else
    {
      hours = hours+":"+String(now.minute());
    }

//Get latest date and time values.
    hours.toCharArray(timeChar,100);
    dateString.toCharArray(dateChar,100);
  }

// Saving date and time to SD card
    myFile.print(dateChar);
    myFile.print(",");
    myFile.print(timeChar);
    myFile.print(",");
// Display date and time to serial monitor
    Serial.print(dateChar);
    Serial.print(",");
    Serial.print(timeChar);
    Serial.print(",");

 for (uint8_t i = 0; i < 4; i++){
// Temperature values to SD card
//    myFile.print("  T");
//    myFile.print(i+1);
//    myFile.print(":");
    myFile.print(sensors.getTempCByIndex(i));
    myFile.print(",");
// Temperature values for display to serial monitor
//    Serial.print("  T");
//    Serial.print(i+1);
//    Serial.print(":");
    Serial.print(sensors.getTempCByIndex(i));
    Serial.print(",");
    }
    myFile.println(" ");
    Serial.println(" ");
myFile.close();  //Close the file.

publishDataToMQTT(dateString);
delay(60000); //  Sixty second delay, to not flood the SD card with readings.
Data will be saved every sixty seconds.


// Voltage data from Voltage Divider
//adc_value=analogRead(ANALOG_IN_VOL);
//adc_v=adc_value*ref_v/1023.0;
//in_v=adc_v/(R2/(R1+R2));
```

```
// Write volatge value to SD
//myFile.print(" V:");
//myFile.print(in_v,1);  //in_v in one decimal place
//myFile.print(" V");
// Display voltage value to LCD
//lcd.setCursor(0,3);
//lcd.print("V=");
//lcd.print(in_v,1);  //in_v in one decimal place
//lcd.print(" V ");
// Current data from the GY sensor
//unsigned int x=0;
//float GYValue=0.0, Samples=0.0, AvgGY=0.0,GYValueF=0.0;
//for (int x=0;x<100;x++){  // To get 100 samples
//  GYValue=analogRead(ANALOG_IN_CUR); // Reading from pin A0
//  Samples=Samples+GYValue; // Adding Samples
//  delay (3);              // Let ADC settle for 3 ms
//}
//AvgGY=Samples/100.0;     // Taking average of the samples
//GYValueF=(AvgGY*(5.0/1023.0)-2.5)/sensitivity;
// Write current value to SD
//myFile.print(" I:");
//myFile.print(GYValueF,1);
//myFile.print(" A");
// Display current value to LCD
//lcd.setCursor(11,3);
//lcd.print("I=");
//lcd.print(GYValueF,1);
//lcd.print(" A");

}

void publishDataToMQTT(String dataLine) {
  if (mqtt.connected()) {
    mqtt.publish(ArduinoTest/Martin/Data, dataLine.c_str());
  }
}

//void mqttCallback(char* topic, byte* payload, unsigned int length) {
  // Handle MQTT messages if needed
//}
```



NTNU

Norwegian University of
Science and Technology