ASHESI

ASHESI UNIVERSITY

SOLAR TRACKING USING REFLECTING MIRRORS

CAPSTONE PROJECT

BSc. Electrical and Electronic Engineering

Abdul-Ghaffar Adam Appiah

2019

ASHESI UNIVERSITY

SOLAR TRACKING USING REFLECTING MIRRORS

CAPSTONE PROJECT

Capstone Project submitted to the Department of Engineering, Ashesi University in partial fulfilment of the requirements for the award of Bachelor of Science degree in Electrical and Electronic Engineering.

Abdul-Ghaffar Adam Appiah

2019

DECLARATION

I hereby declare that this capstone is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature: ………………………………………….

Candidate's Name: ……………………………………………………………….

Date: ……………………………………………….

I hereby declare that preparation and presentation of this capstone were supervised in accordance with the guidelines on supervision of capstone laid down by Ashesi University College. Supervisor's Signature: ………………………………………

Supervisor's Name: ……………………………………………………………….

Date: ………………………………………………

ACKNOWLEDGEMENTS

My sincere appreciation goes to my supervisor Mr. Richard Akparibo who has been resourceful in helping me with my research and guiding me through the entire project. I am also most grateful to the dean of Engineering Dr. Nathan Amanquah for providing me with the initial idea as I thought through cracking the solar tracking issue in Ghana. I will also express my gratitude to Michael Kisi Danquah, a fellow student who helped me with my mechanical design. My final gratitude goes to Nicolas Tali, who ensured that I had all the necessary materials to complete my project successfully.

ABBREVIATIONS

ADC: Analog to Digital Converter

FPGA: Field Programmable Gate Array

FSM: Finite State Machine

MPPT: Maximum Power Point Tracking

PCB: Printed Circuit Board

PV: Photovoltaic

PWM: Pulse with Modulation

VHDL: VHSIC Hardware Description Language

VHSIC: Very High-Speed Integrated Circuit

Abstract

Solar Energy is the fastest growing renewable energy source in the world. The soaring production and the significant technological breakthrough of solar energy in certain countries like China and India has brought about a drastic reduction in the cost of its exploitation. Today, the price of solar PV panels in China has declined by as much as 50%. Despite this immense progress in the solar energy industry, the question of how solar energy has evolved in Ghana and Africa at large, remains unanswered. Solar tracking; a technology that was introduced in 1991 to serve as a proponent of power generation from solar panels, has not yet been implemented anywhere in Ghana and many other countries in Africa. Research shows that Ghana receives sunshine duration of 1,800 hours to 3,000 hours per annum, providing a friendly environment for solar energy exploitation. Due to this, some renewable energy engineers argue that, for a geographical location like Ghana, solar tracking generates an insignificant amount of profit, considering the expense of technology involved in constructing tracking systems. This paper describes the design of "Mtrack"; a low-cost solar tracking technology that uses reflection mirrors to focus sunlight unto the maximum power points of solar panels. This is aimed at reducing the number of expensive motors and sensors used for tracking since this design uses just one motor and a voltage comparator logic but can still focus light unto an entire array of solar panels.

Table of Content

Chapter 1: Introduction

1.1 Background

Solar energy is the energy released from the sun as electromagnetic radiation [1]. This energy is produced from the core of the sun by a process called nuclear fusion, where hydrogen atoms fuse together to form a more complex element called helium [1]. Once this process is completed, electromagnetic radiation is released in the form of heat and light into space. The earth's ozone layer, which protects the earth from external effects from space filters this light allowing a small amount that is less harmful to earth's living organisms to reach the earth [1]. Although this is a very small portion compared to the amount of energy released from the sun, it is enough to supply all earth's energy needs [1]. Solar energy has for a long period of time been harnessed in many ways to benefit living organisms. Some of these include solar thermal systems, where technology has been developed to collect water flowing through a tube and heat it up for domestic use and solar thermal power plants, where steam is generated from solar energy and used to drive turbines for power generation. Also, there are Photovoltaic (PV) applications of solar energy, where electricity is generated from solar panels by photonic emissions. The increase in the application of solar energy is highly beneficial for the earth because this clean energy source can help to reduce all environmentally threatening mechanisms of generating power. This project will mainly focus on PV applications.

The solar PV technology was born when Daryl Chapin, Calvin Fuller, and Gerald Pearson developed the silicon PV cell; the first solar cell, capable of converting the sun's energy into electricity at Bell Labs [2]. With time, satellites were launched into space with solar modules and arrays serving as their primary source of power. Today, the PV technology is applied to provide power for vehicles, industries and homes. The increase in production in the solar energy industry over these years has brought about a significant technological breakthrough in the industry. The technologies introduced helped to increase efficiency of solar panels, optimize power generation among others. Solar tracking is one of these ground-breaking technologies.

Solar tracking which was introduced in 1991 by Array Technologies. This is the process of orienting solar panels towards the sun for optimum power [3]. In a typical solar tracker, there are several motors to move the panels, some light sensors, microcontrollers and other devices to improve sensitivity and selectivity. Solar tracking also depends on the location of installation. For every location, there is a different azimuth angle (the sun's relative direction along the local horizon) and a different zenith angle (the sun's apparent altitude), these two angles bring about the changes in position of the sun throughout the day and throughout the year respectively. A single axis tracker takes into consideration the azimuth angle, and a dual axis tracker takes into consideration both the azimuth angle and the zenith angle [4].

1.2 Problem Definition

As a new technology in the solar energy industry, expenses to construct a solar tracker are still significantly high, hence, although we see this technology in many advanced countries, for a developing country like Ghana and many other countries in Africa, solar tracking may take a while to be introduced here. It is therefore essential for us as Africans to find out ways to reduce the cost of solar tracking to suit our geographical location and beyond.

1.3 Objectives

The objectives of this project work include:

- Coming up with a more efficient way of optimizing solar power.

- Reducing the cost of solar tracking for the average African
- Further exploring the use of digital systems in designing solar energy technology

1.4 Expected Outcomes

The expected outcomes of this project include:

- A fully functioning Field Programmable Gate Array (FPGA) control circuit that moves mirrors to focus light unto solar panels.
- A curve showing that power generated using the control circuit is greater than power generated without it.
- T-test results showing that there is a statistical significance between values generated with the control circuit and values generated without it.

1.5 Motivation of Research

The motivation behind this research is from the fact that there are no solar tracking systems in Ghana and many places in Africa. The reason behind the absence of the tracking systems is that they are highly expensive, and it might not be economically sound to install one in this geographical area. The Mtrack tracking system described in this paper is therefore an appropriate replacement for tracking systems as it uses a single motor and no light sensors. This will reduce the cost of solar tracking for the average Ghanaian.

1.6 Research Methods Used

Research methods used for this project includes:

- Review of existing works
- Experimentation
- Modelling
- Review of safety measures

1.7 Facilities Used

For this project the Ashesi electronics laboratory and the workshop were used. The electronics lab was used for all electronic connections and code testing, whereas the Workshop was used for mechanical construction.

1.8 Project Organization

The paper describes in detail the design of the technology, where existing designs were reviewed with the design objectives clearly stated, followed by design decisions with Pugh matrices and some design iterations. Next the methodology was described clearly expanding on the experimental setup. Results were then explained, and the conclusion was presented.

Chapter 2: Literature Review

2.1 Introduction

Solar trackers have been constantly improved since it's inception in 1991. Many trackers have been built with advanced technology to cut down cost and increase efficiency. All trackers that exist are designed with solar modules or arrays, resting on a set of motors for movement. The motors are usually controlled by microcontrollers as signals are received from a light sensor.

2.2 Solar Tracking Systems

Existing solar trackers come in two forms; the single axis solar tracker and the dual axis solar tracker. Single Axis trackers use two light sensors and only move relative to the sun's azimuth position (position changes throughout the day) [5]. Dual Axis trackers on the other hand use 4 sensors and can move relative to the sun's azimuth position and the zenith position (position changes throughout the season) [6]. In the dual axis solar tracker, the panel rests on a single motor that moves it to track the azimuth position of the sun. Both the panel and the motor also rest on a second motor. The second motor rotates the entire system tracking the zenith position of the sun. Figure 3.0 is an example of a dual axis tracker. Figure 3.1 [6] shows the setup of the first motor and the second motor.

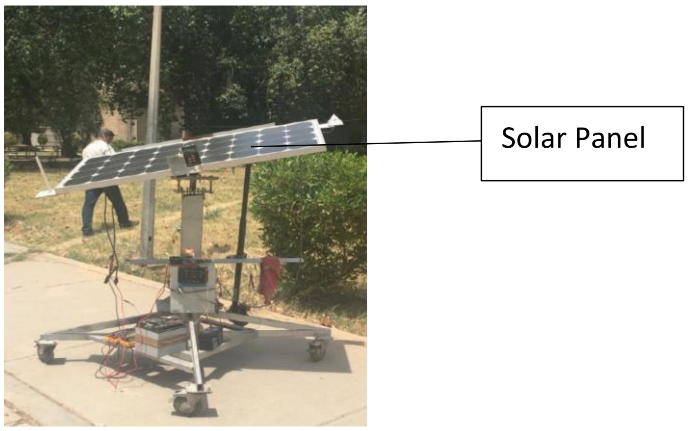

Figure 3.1 A Dual-Axis Solar Tracker.

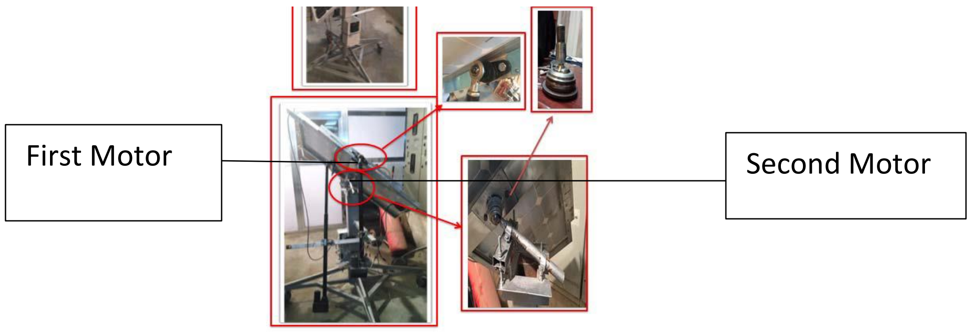

Figure 3.2 Setup of first motor and second motor of dual axis solar tracker.

The difference between this design and that of the single axis tracker is that, the single axis tracker uses one motor per module. Research shows that existing designs for single axis and dual axis trackers are very expensive and in certain regions it is insignificant to use trackers during a certain time of the year. Table 3.1[7] shows data obtained from the cost analysis performed in Worcester, MA on fixed solar systems and solar trackers during the winter (January) and the Summer (July).

Table 3.1 Data obtained from the cost analysis performed on fixed solar systems and trackers

Watt Hours generated per day by 2520W Off grid solar system in Worcester, MA				
Mount type and cost	**System Cost**	**January**	**July**	**Average**
Fixed 2,520W $1,400	$16K	5,739Wh	9,284Wh	7,596Wh
Single Axis tracker $3,500	$18K	6,583Wh	12,322Wh	9,452Wh
Increase	$2K	15% 844Wh	33% 3,038Wh	24% 1,856Wh
Dual Axis Tracker $6,300	$21K	6,921Wh	12,998Wh	9,790Wh
Increase	$5K	20% 1,182Wh	40% 3,714Wh	29% 2,194Wh
Fixed 3,360W $2,000	$17K–19K	7,653Wh	12,380Wh	10,130Wh
Increase	$1K– $3K	33% 1,914Wh	33% 3,096Wh	33% 2,534Wh

The data above was obtained from a solar system, mounted using 9 SolarWorld 280W panels and 12 Outback AGM batteries for 534Ah. [7]. The "increase" section in the table represents increase in cost and output power after the various trackers have been installed and the "fixed 3,360W $2000" section represents data collected after 3 more panels were added to the system costing $2000 ($600 more than the system where 9 panels were used). From the table it can be observed that the average increase in energy for both winter and summer when three extra panels were added was more than the average increase when trackers are used; with the extra panels providing an increase of 2,534Wh, the dual axis tracker, 2,194Wh and the single axis tracker, 1,856Wh. Also, it can be observed that the tracking system costs twice as much as the addition of three extra panels. This shows that the existing designs for solar tracking are highly expensive considering the amount of power increase it provides, presenting itself as an insignificant technology for some African countries like Ghana. This makes the implementation of the Mtrack highly essential as it drastically reduces the cost of solar tracking.

2.2 Review of Related works

As energy demand increases, the capacity of solar farms increases. The increase in the number of solar panels used in every array per installation is gradually increasing the number of motors and sensors needed if solar tracking is to be designed for that installation. This is gradually increasing the cost of solar tracking. Currently, trackers that involve the movement of mirrors to focus light unto solar panels are inexistent, the closest design to the Mtrack system is described in the paper "Photovoltaic Systems with Solar Tracking Mirrors" by Jun Hu and Toshiaki Yachi. This paper describes a design in which mirrors were placed around solar panels to direct sunlight to the optimum power point of the panels but in this case the mirrors are fixed and not controlled by any electric circuit to move as the sun’s position changes. Hence MTRACK is a completely innovative design and will be a way to drastically reduce the cost of solar tracking.

Chapter 3: Design Methodology

3.1 Introduction and Design Objective

As solar trackers are highly expensive, the Mtrack system seeks to reduce the cost of solar tracking. The system will use inexpensive plane mirrors and a motor controlled by a FPGA to direct sunlight unto fixed solar panels to optimize energy. It also uses a voltage comparator which is an inbuilt functionality of the FPGA instead of light sensors that add extra cost to the design. This will reduce the number of motors used in the existing tracking systems, take away the light sensing aspect, and reduce the stress of constantly moving panels which might reduce the efficiency and ultimately reduce the life span of the panels. Also, if a tracking system is to be mounted on an already installed fixed solar system, there will be no need taking off the mounted panels to install the tracker.

3.2 Design Decisions

For this project, two inexpensive 20 x 30 inch, ¼ inch thickness plane mirrors were used. The purpose of the plane mirrors is to reflect sunlight unto the panels. A NEMA 17 bipolar stepper motor was also used to move the mirrors. In order to increase the torque needed to move the mirrors a set of gears with gear ratio 2:1 were 3D printed for the motor. A set of bearings were also placed on the shaft to ensure smooth movement of the mirrors. The system that served as the brain of the entire design was thought through carefully. There were two options that were considered for the tracking; designing a printed circuit board (PCB) with all the necessary electronic components and using an FPGA. The FPGA was selected because in coming up with a prototype and a proof of concept for a tracking mechanism, it was important to consider a method the will be less time consuming and less expensive. The basys 3 was the FPGA used for this project and it has an in-built Analog to Digital Converter (ADC), register, counter and a memory that was suitable for the tracking system. Also, two 20W Solar panels were used. Pugh Matrices have been created in the next sub section to detail out how the devices were chosen.

3.2.1 Pugh Matrix

Table 3.2 Pugh matrix for system's central control

		FPGA (Baseline)	PCB
Criteria			
Requires Additional Components		0	-1
Ultimate cost when used		0	-1

Table 3.3 Pugh matrix for mirror

		20 x 30 inch, ¼ inch thickness, plane mirror (Baseline)	Concave Mirror
Criteria			
Reflectivity		0	-1
Cost		0	0

Table 3.4 Pugh matrix for motor

		NEMA 17 Bipolar stepper motor (Baseline)	Actuator
Criteria			
Ease of control		0	-1
cost		0	-1

Table 3.5 Pugh matrix for Charge Controller

		MPPT Charge Controller (Baseline)	PWM Charge Controller
Criteria			
Availability		0	-1
cost		0	0
Compatibility with Raspberry Pi		0	-1

Matrix key

+1 - better than
0 - equal to
-1 - worse than

3.3 Mechanical Design and Construction

The mechanical design of this project comes in two parts, the part with the fixed solar panels, and the part with the moving mirrors. The images below represent 3D models of the design.

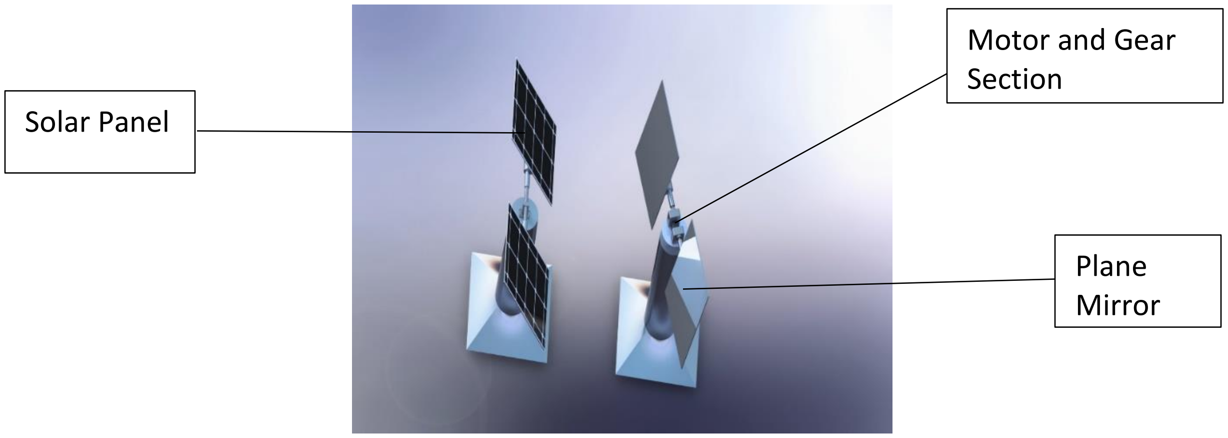

Figure 3.3 Mechanical design of Mtrack solar tracking system

From the image above, the part with the mirror has the stepper motor locked up between two metal blocks. These blocks contain gears that are connected to the main shaft where the mirrors are attached. The gears increase the torque provided by the stepper motor to be able to move the mirrors. The part with the solar panels is connected to a fixed shaft inclined at a certain angle. Each of these parts has its own electrical connection.

3.4 Electrical Set Up

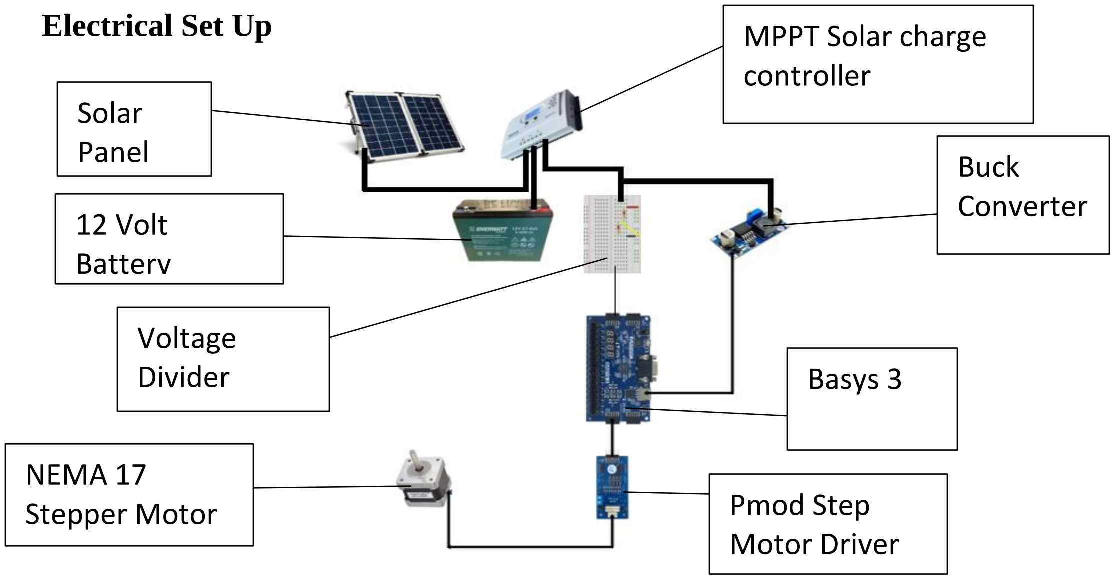

Figure 3.4 Electrical design of Mtrack solar tracking system

The image above represents the electrical connections made to prove the concept of the Mtrack system. As described in chapter 3.3, the system is in two parts, the part with the moving mirrors and the part with the fixed solar panels. In this design, there is a maximum power point tracking (MPPT) charge controller to which a 12V battery and two solar panels are connected. The load output of the charge controller is connected to a basys 3 FPGA board through a voltage divider. The 12V from the charge controller was passed through a voltage divider because the ADC port of the basys 3 board can only take 1V. Power is taped from the connection between the load section of the charge controller and the voltage divider to power the FPGA. This is stepped down using a buck converter because the FPGA can only take 5V. A motor is then connected to the basys 3 board through a pmod step stepper motor driver to receive signals in order to move the mirrors accordingly. The signals are sent from a VHSIC

Hardware Description Language (VHDL) code written using the Xilinx Vivado software for the basys 3 FPGA board. The design using Xilinx Vivado is described in the next sub chapter.

3.5 Software Design

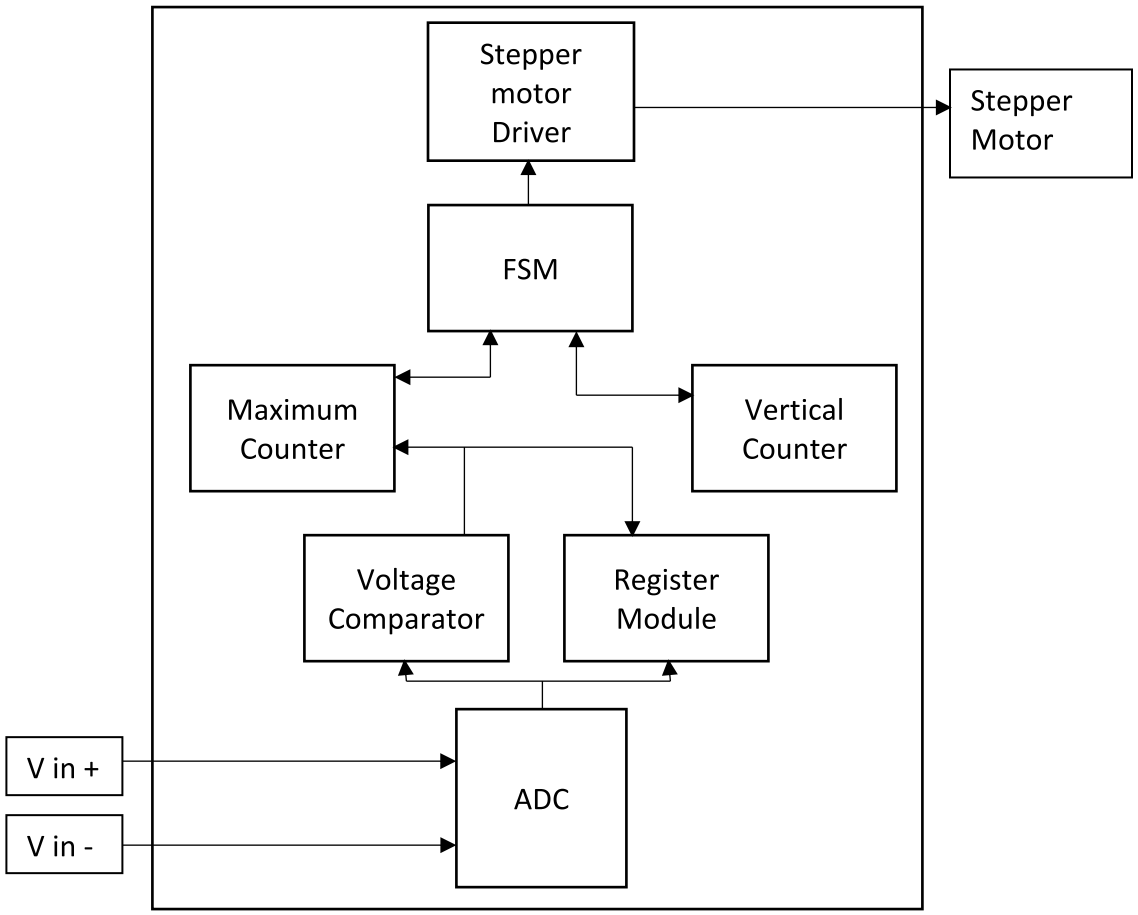

Figure 3.4 VHDL Design of Mtrack System

The basys 3 FPGA board consists of several in built digital electronic components and a code is needed to configure the various components. Languages used to configure these components are known as hardware description languages; VHDL and Verilog are examples of such languages. For this project VHDL was used. Figure 3.4 shows the design of the VHDL code that serves as the brain of the Mtrack system. The code is written using the behavioral and structural modeling technique. For the Mtrack system, components on the basys 3 board that were needed were the ADC, the register module, the voltage comparator, counters and the servo driver. A VHDL code was written for each of these components using behavioral modeling after which they were put together in a structural code. Also, a VHDL code was written for an Finite State Machine (FSM), which serves as the central point of communication between the counters and the stepper motor driver. The VHDL code for the various components including the structural code that puts all components together can be found in the appendix.

3.5.1 Finite State Machine

An FSM is an abstract machine that can move from one state to another in response to certain inputs, the movement through the states is called transition [8]. The FSM was needed in the VHDL design to set up a relationship between the vertical counter, the maximum counter and the stepper motor driver.

3.5.2 Analog to Digital Converter

The ADC is the part of the FPGA that takes analogue signals coming from the solar panel and converts it to digital signals, this is done with a special behavioral VHDL code. The signals need to be changed to digital because the FPGA can only work with digital signals [9]. This voltage is stepped down to 1V before the code is run on the FPGA because the ADC can only take 1V [10]. For the ADC code to function appropriately in vivado, it had to be configured within the vivado IP catalog [11].

3.5.3 Register Module

The register module is designed to store the current maximum voltage. It also sends the voltage to a separate model called the voltage comparator that compares the maximum voltage in the register to the voltages coming from the ADC [12].

3.5.4 Voltage Comparator

The voltage comparator is made in such a way that, it takes two voltage values and sends the greater one to the register to be stored. The two voltages compared are taken from the ADC and the register. Once the system is started, and the motor is moving, the first voltage that comes from the ADC is stored in the register [10]. The subsequent voltages are passed through the voltage comparator where the comparator compares the voltage initially stored in the register and the voltages coming from the ADC. If the comparator identifies a voltage higher than the one originally stored in the register, it replaces the voltage in the register with the new higher voltage [10].

3.5.5 Counters

There were two counters created for the project. The Vertical Counter and the Maximum Counter. The vertical counter starts counting once the Mtrack system is started. It has a specific value it counts to. Once it reaches this value it sends a signal to the FSM to activate the next counter. The counter activated every time the vertical counter reaches its set value is the maximum counter [13].

The maximum counter starts incrementing once the systems starts. During the incrementing process, if a new maximum voltage is found in the register a signal is sent to the maximum counter to reset it. This erases the maximum counter's previous count and causes it to start incrementing again [13]. This happens anytime there is a new maximum voltage until the vertical sweep is completed. Once the vertical counter reaches it's set value, the FSM sends a signal to the maximum counter to start decrementing. Once the decrementing starts a signal is sent from the FSM to the stepper motor driver to move the motors to the position where the last maximum voltage was recorded [13]. By the time the maximum counter is done with its count, the motor will have moved the mirror to a position where the sun is reflected well enough to produce the maximum voltage from the solar panels. This is how the mirror focuses light unto the solar panels.

3.5.6 Stepper Motor Driver

The stepper motor driver module basically moves the motor that rotates the frame of the mirror. It takes a signal from the FSM and send a Pulse-Width Modulation (PWM) signal to the motor. This signal determines the speed and the direction of the motor [14]. The VHDL architecture for the stepper motor driver was designed using the structural model, which means there were two behavioral models created, imported and connected in a way that will provide a desired output for the stepper motor driver. These two behavioral models are the "pulse-with-modulation-control" and the "clock-divider"

In the VHDL architecture for the stepper motor driver, two signals are put into a certain logic to create a value for the pulse-with-modulation-control. The pulse-with-modulation-control then creates a PWM signal in accordance to the desired speed and direction of the motor [14]. For the stepper motor used for this project, a wave width of 1.5ms stops it from moving, anything above that causes it to move anticlockwise and anything below it causes it to move clockwise [14]. Since a low speed was needed for rotation, a clockwise wave width of 1.48ms was used and an anticlockwise wave width of 1.52ms was used in the VHDL code. The clock-divider module was used to divide the clock speed which has a default frequency of 100Mhz for the basis 3 board into smaller bits to be used by the motor driver. The entire VHDL code can be found in appendix A of this paper.

Chapter 4: Results and Discussion

4.1 Results

This section consists of the results obtained after the code was run and how the code worked with the entire system put together. It also captures some experiments conducted to validate the reliability of the Mtrack system.

The VHDL code successfully complied in the vivado software. The elaborated design displayed after the code was run is shown in figure 4.0.

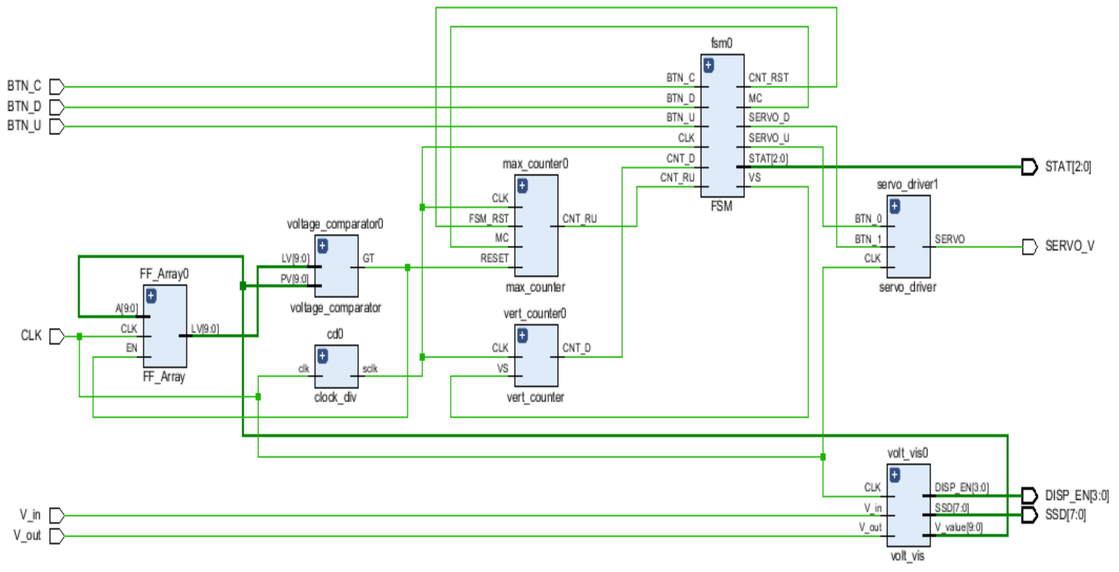

Figure 4.0: Elaborated design of Mtrack VHDL architecture

The code after a successful compilation was loaded unto the basys 3 board. The code successfully controlled the motor to move the mirrors as expected. The entire system was then placed in the sun for values to be recorded. The load voltage, current and power values were recorded, with two different set ups; the fixed solar panels without the mirror tracking component and a set up with the mirror tracking component. The set ups were mounted on two different days each; a day when the sun was significantly high and a day with a significantly low temperature. For both set ups and for each day, readings were taken from 7:30 am to 5:00 pm with a 30 minutes interval. Current values recorded was the current supplied by the solar panel which is equal to the current through the basys 3 board (peaking 1.14A), the voltage values recorded were voltage readings from the solar panel (peaking at 17. 5V), hence the power recorded was the power supplied by the solar panels (peaking at 20W). Table 4.0 shows the data recorded after the experiment. Figure 4.1 shows the setup of the system without the mirror tracking component, figure 4.2 shows the setup of the system with the mirror tracking component, figure 4.3 shows the gear and motor section and figure 4.4 shows the control system

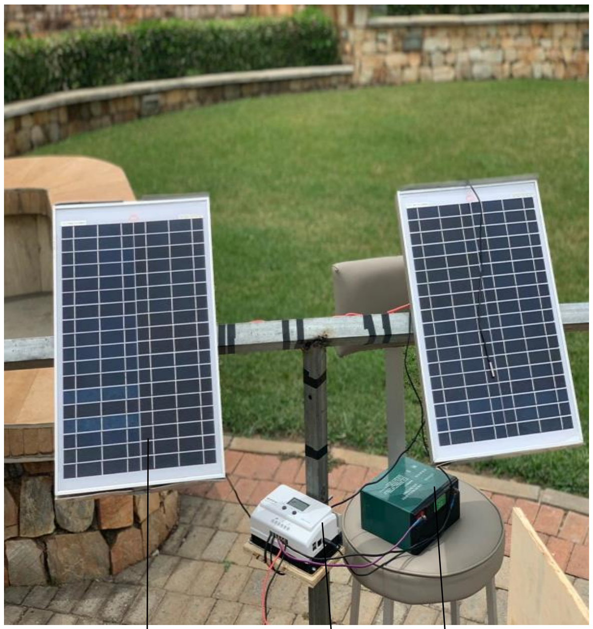

Figure 4.1: Setup without tracking component

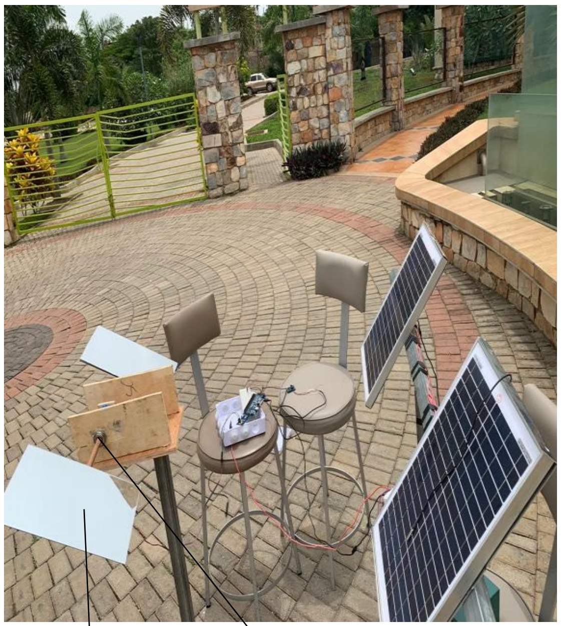

Figure 4.2: Setup with tracking component

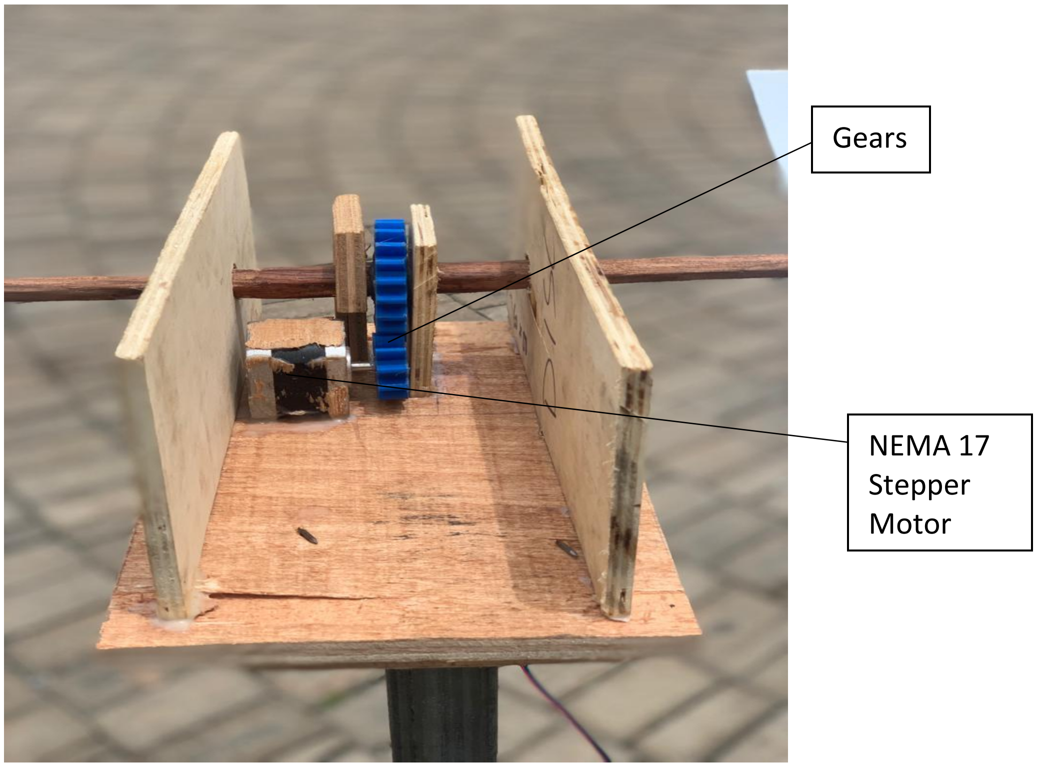

Figure: 4.3 Gear and motor section

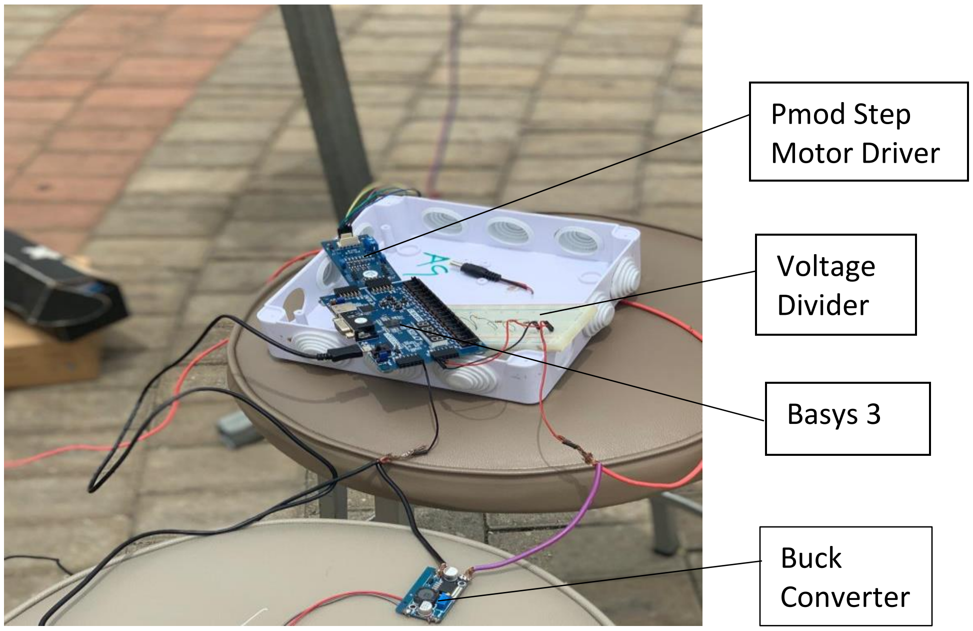

Figure 4.4: Control system

Table 4.0: Results obtained after testing Mtrack system

	High Temperature Days						Low Temperature Days					
	Day 1			Day 2			Day 1			Day 2		
	Without Tracking			With Tracking			Without Tracking			With Tracking		
Parameter / Time	I/A	V/V	P/W	I/A	V/V	P/W	I/A	V/V	P/W	I/A	V/V	P/W
7:30 am	0.6	11.1	6.7	0.6	10.9	6.5	0.4	8.3	3.3	0.5	8.9	4.5
8:00 am	0.7	11.3	7.9	0.8	12.8	10.2	0.4	8.1	3.2	0.4	8.3	3.3
8:30 am	0.6	10.9	6.5	0.7	11.6	8.1	0.5	8.7	4.4	0.4	8.4	3.7
9:00 am	0.7	11.2	7.8	0.7	11.1	7.8	0.4	8.4	3.4	0.4	8.2	3.3
9:30 am	0.8	13.3	10.6	0.9	15.2	13.7	0.5	8.5	4.3	0.8	13.9	11.1
10:00 am	0.8	14.7	11.8	0.9	15.9	14.3	0.6	11.0	6.6	0.9	15.7	14.1
10:30 am	0.9	15.1	13.6	0.9	15.0	13.5	0.8	13.2	10.6	0.9	16.1	14.5
11:00 am	0.9	15.0	13.5	1.0	17.0	17.0	0.9	15.4	13.9	0.9	16.3	14.7
11:30 am	0.9	14.9	13.4	1.0	17.0	17.0	0.8	13.9	11.1	1.0	16.6	16.6
12:00 pm	1.0	16.9	16.9	1.0	17.5	17.5	0.9	15.9	14.3	0.9	16.4	14.76
12:30 pm	1.0	17.0	17.0	1.0	17.5	17.5	0.9	16.1	14.5	0.9	16.5	14.85
1:00 pm	1.0	16.7	16.7	1.0	17.3	17.3	0.9	16.0	14.4	1.0	17.1	17.1
1:30pm	0.9	16.5	14.9	1.0	17.4	17.4	0.9	16.3	14.7	0.9	16.2	14.6
2:00pm	1.0	17.1	17.1	1.0	16.5	16.5	0.8	14.1	11.3	0.8	14.2	11.4
2:30pm	1.0	17.1	17.1	1.0	16.7	16.7	0.8	14.3	11.4	0.8	14.8	11.8
3:00pm	1.0	16.6	16.6	0.9	15.9	14.3	0.9	15.1	13.6	0.9	15.0	13.5
3:30pm	0.8	14.9	11.9	0.9	16.0	14.4	0.8	12.7	10.2	0.8	12.5	10.0

4:00pm	0.7	11.3	7.9	0.9	15.4	13.9	0.6	10.1	6.1	0.8	13.3	10.6
4:30pm	0.8	12.8	10.2	0.9	15.5	14.0	0.6	10.3	6.2	0.7	11.2	7.8
5:00pm	0.6	11.0	6.6	0.8	13.3	10.7	0.4	8.4	3.4	0.6	11.0	6.6

From the data collected, a comparison was made between the values recorded when the tracking system was implemented, and values recorded without the system. Since we hope to prove that there is more power generated with the tracking system, than without it, a T-test was performed to check the reliability of the Mtrack system. For the t-test, the power values of the set ups, with and without tracking were taken as the two independent groups to be tested. Two t-tests were performed, one for high temperature days and another for low temperature days. Table 4.1 and 4.2 show the t-test results for high temperature days and low temperature days respectively. Figure 4.2 and 4.3 are graphical representations for the values collected during high temperature days and low temperature days.

Table 4.1: T-test results for high temperature days

	Without Tracking		With Tracking
Mean (x)	12.235		13.915
Variance (S^2)	15.91397368		12.30871053
Observations (n)	20		20
T value		2.683364084	
Critical value		2.093024054	
P value		0.001578474	
Alpha value		0.05	

Table 4.2: T-test results for low temperature days

	Without Tracking		With Tracking
Mean (x)	9.045		10.9405
Variance (S^2)	19.93102632		20.94259447
Observations (n)	20		20
T value		2.378904919	
Critical value		2.093024054	
P value		0.003151238	
Alpha value		0.05	

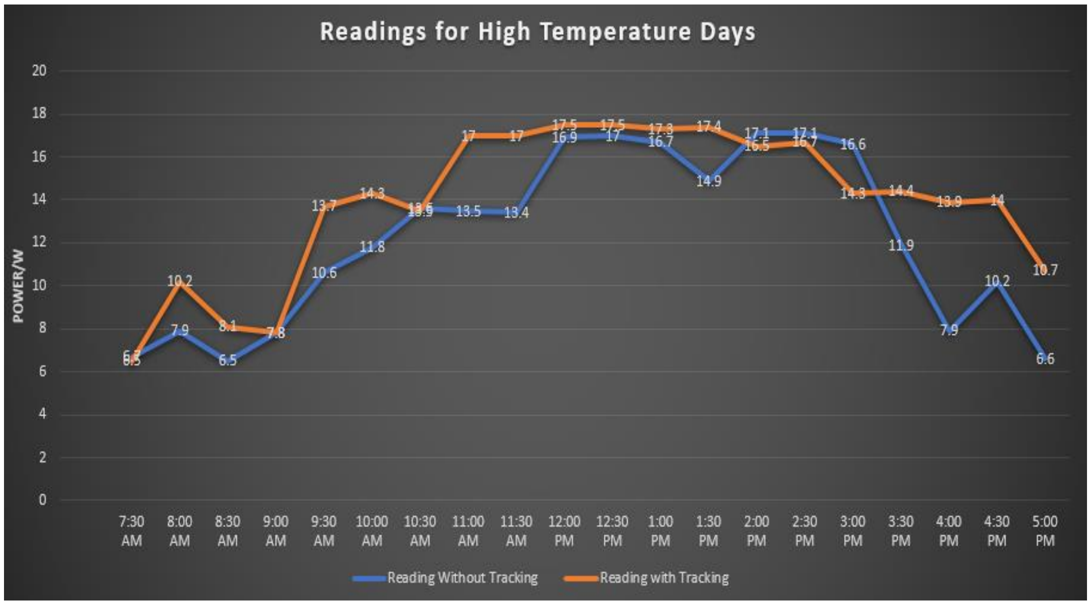

Figure 4.2: Graphical representation of values collected during high temperature days.

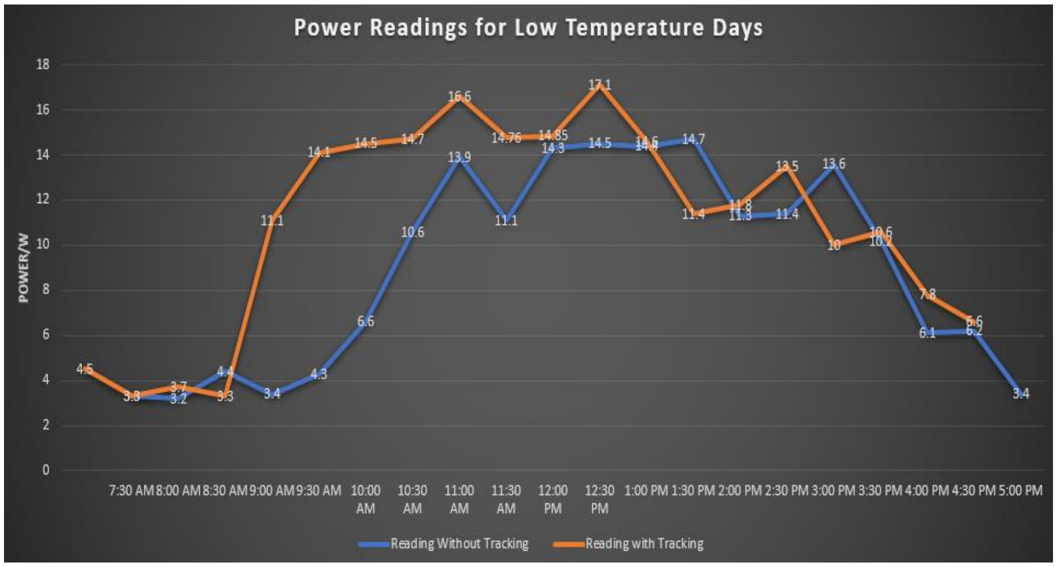

Figure 4.3: Graphical representation of values collected during low temperature days.

4.2 Discussion

From figure 4.2 and 4.3, the orange curve represents values collected with the tracking system whereas the blue curve represents the values collected without the tracking system. Although the graphs show that, the tracking system improves power generation for both high and low temperature days, there were certain times during the low temperature days where power values collected without tracking were greater than power values recorded with the tracking system. The T-tests performed helped to prove that the tracking system was generally a better option and these results obtained wouldn't be obtained with a random set up. A t-test comes with a null hypothesis, which is an initial declaration. For this project, the null hypothesis decided on was that, there is no difference between the two data sets (values with tracking and values without tracking). Which means that, by rejecting the null hypothesis it means there is a significant difference between the two groups (which we hope to prove) and vice versa. A statistical significance means the null hypothesis is rejected and vice versa. After every T-test, a t-value is obtained, the t-value shows the variation between the two sets of data and the variation within the individual data points collected. The higher the t-value, the more spread out the values are and vice versa. For every t-value, there is a critical value below which the t-value cannot be accepted statistically or there is no statistical significance. A p-value is also obtained from a T-test. The p-value shows the probability of obtaining the same results given a random occurrence. A low p-value means that there is a low probability of obtaining the same results and a high p-value means that there is a high probability of obtaining the same results given a random occurrence. For every p-value, there is an alpha value above which the p-value will be too large to ensure statistical significance. This means if the p-value is greater than the alpha value, there will be no statistical significance and vice versa. From table 4.1, the t-value obtained after the T-test performed for the high temperature days was 2.683364084, this value was greater than the critical value which is 2.093024054. Also, the p-value obtained was 0.001578474, smaller than the alpha value which is 0.05. Based on these values it can be concluded that there is a statistical significance between the two groups hence the null hypothesis can be rejected. The graph in figure 4.2 show that values obtained from the tracking system were better and the T-test validated this observation. The same conclusion can be drawn for the low temperature days referring to table 4.2 with a T-value of 2.378904919, a critical value of 2.093024054, a p-value of 0.003151238, and an alpha value of 0.05.

Chapter 5: Conclusion, Limitation and Future Work

5.1 Conclusion

In conclusion, this paper has introduced a major issue with the installation of tracking systems in Ghana. This issue primarily has to do with the cost of financing the systems. The paper points out in the introduction that the number of motors and sensors used for tracking per installation is a leading contributor to its high expense. The paper then introduces the Mtrack design - a design that seeks to solve the issue -. The Mtrack system is described as one with a pair of reflecting mirrors that reflect sunlight unto the panels for maximum power. The Mtrack system is solving the problem of excess motors and sensors in the sense that it does not need more than a single motor to reflect light unto an entire array of panels. It also uses a voltage comparator logic with the help of a basys 3 FPGA board which eliminated the idea of using light sensors for tracking. The paper describes some experiments done after fabricating the Mtrack system which produces curves evident that the Mtrack system was a better tracking system. It also provides T-test results showing a statistical significance between groups of values recorded with the tracking system and without the tracking system. Despite a few flaws and limitations, the Mtrack system has been proven through this paper to be a reliable and a cost-effective mechanism for tracking sunlight for maximum power production.

5.2 Limitation

In the quest to prove that the Mtrack system was a better tracking option, there was the need to compare results obtained after testing the system with already existing designs, Unfortunately, there was no tracking system designed with the kind of solar panels used for the Mtrack design. Also, irradiance recorded for the location within which tracking systems existed were extremely different from that of Ghana where the Mtrack system was tested. This made it difficult to conduct a t-test to compare the results obtained from the Mtrack system to already existing tracking mechanisms. This limited me to only proving that panels with the Mtrack system produced more power than panels without it.

5.3 Future Works

This section discusses the future additions to the Mtrack system once it moves from the prototype stage.

As the mirror sweeps across the panel, it reads certain values, and then settles at the maximum value. The maximum value is the highest value among the values collected but may not necessarily be the maximum power point of the panel. This introduced a flaw in the Mtrack design because there were fewer values recorded after each sweep. The fewer values resulted from the size of mirrors used for the system. Hence some of the values recorded as the maximum power points of the panels were not accurate. It can be inferred from this explanation that, the higher the number of values collected, the higher the probability of settling at the exact maximum power point of the panels after every sweep and the larger the mirrors the higher the number of values recorded and the higher the probability. Hence in the future, larger mirrors will be used which will require a heavier motor to produce a higher toque to be able to move it. Also, since this is a simulation, a basys 3 board was used. In future a PCB will be designed specifically for this system with a customized casing for the purpose of commercialization.

References

[1] Need.org, 2019. [Online]. Available: https://www.need.org/files/curriculum/infobook/solari.pdf. [Accessed: 10- Jan- 2019].

[2] S. Hoang, Repository.wellesley.edu, 2019. [Online]. Available: https://repository.wellesley.edu/cgi/viewcontent.cgi?article=1019&context=library_awards. [Accessed: 10- Jan- 2019].

[3] Theseus.fi, 2019. [Online]. Available: https://www.theseus.fi/bitstream/handle/10024/111666/Nguyen_Nam.pdf?sequence=1. [Accessed: 10- Jan- 2019].

[4] A. Singh, and N. Aphiratsakun, “AU solar tracking system”, IEEE, p.4, 2015.

[5] F. Mustapha, S. Shakir and F. Mustafa, "Simple design and implementation of solar tracking system two axis with four sensors for Baghdad city", IEEE, p. 5, 2018.

[6] A. Beaudet, "Solar Tracking Tips: To Track Or Not To Track?", *Solar Power News & DIY Solar Tips*, 2019. [Online]. Available: https://www.altestore.com/blog/2015/06/solar-tracking-tips-to-track-or-not-to-track/#.XDTJhVX7TIU. [Accessed: 02- Mar- 2019].

[7] 2015. [Online]. Available: https://www.altestore.com/blog/2015/06/solar-tracking-tips-to-track-or-not-to-track/#.XDe50lX7TIV. [Accessed: 10- Jan- 2019].

[8] Ben-Ari M., Mondada F, “Finite State Machines” 2018, In: Elements of Robotics, Springer, Cham.

[9] "Basys 3 Reference [Reference.Digilentinc]", Reference.digilentinc.com, 2019. [Online]. Available: https://reference.digilentinc.com/basys3:refmanual. [Accessed: 21- Apr- 2019].

[10] "Basys3 XADC Demo [Reference.Digilentinc]", Reference.digilentinc.com, 2019. [Online]. Available: https://reference.digilentinc.com/basys3:xadcdemo. [Accessed: 21- Apr- 2019].

[11] Xilinx.com, 2019. [Online]. Available: https://www.xilinx.com/support/documentation/ip_documentation/xadc_wiz/v3_3/pg091-xadc-wiz.pdf. [Accessed: 21- Apr- 2019].

[12] T. Radtke and S. Fritzsche, "Simulation of n-qubit quantum systems. I. Quantum registers and quantum gates", Computer Physics Communications, vol. 173, no. 1-2, pp. 91-113, 2005. Available: 10.1016/j.cpc.2005.07.006.

[13] Uobabylon.edu.iq, 2019. [Online]. Available: http://www.uobabylon.edu.iq/eprints/paper_1_7204_163.pdf. [Accessed: 21- Apr- 2019].

[14]"Stepping Motor Control (with VHDL) - Logic - eewiki", Digikey.com, 2019. [Online]. Available: https://www.digikey.com/eewiki/pages/viewpage.action?pageId=4096117. [Accessed: 21- Apr- 2019].

[15] "Vivado 2015 4 Basys3 22 Mar 16 alternate", YouTube, 2019. [Online]. Available: https://www.youtube.com/watch?v=Lt_2Gxa_wTA. [Accessed: 21- Apr- 2019].

[16] "AR# 64764: Vivado Logic Analyzer - Warning: [Labtools 27-3123] The debug hub core was not detected at User Scan Chain 1 or 3", Xilinx.com, 2019. [Online]. Available: https://www.xilinx.com/support/answers/64764.html. [Accessed: 21- Apr- 2019].

[17] Forums.xilinx.com, 2019. [Online]. Available: https://forums.xilinx.com/t5/General-Technical-Discussion/warning-debug-core-was-not-detected/td-p/744291. [Accessed: 21- Apr- 2019].

[18] M. Soares dos Santos and J. Ferreira, "Novel intelligent real-time position tracking system using FPGA and fuzzy logic", ISA Transactions, vol. 53, no. 2, pp. 402-414, 2014. Available: 10.1016/j.isatra.2013.09.003.

[19] "AUTOMATIC SOLAR TRACKING ROBOTIC SYSTEM BASED ON SOLAR INTENSITY", International Journal of Recent Trends in Engineering and Research, vol. 3, no. 3, pp. 179-181, 2017. Available: 10.23883/ijrter.conf.20170331.035.hglea.

Appendix

VHDL code

Top Module code

```
----------------------------------------------------------------------------------
-- Company: ASHESI UNIVERSITY
-- Engineer: ABDUL-GHAFFAR ADAM
--
-- Create Date: 02/10/2019 12:34:58 PM
-- Design Name:
-- Module Name: MTRACK - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity MTRACK is
   Port ( BTN_UP, BTN_DOWN, BTN_CONTROL: in STD_LOGIC;
          CLK : in STD_LOGIC;
          Volt_in : in STD_LOGIC;
          Volt_out : in STD_LOGIC;
          DISPLAY_EN : out STD_LOGIC_VECTOR (3 downto 0);
          SS_DISP : out STD_LOGIC_VECTOR (7 downto 0);
          STEPPER_V : out STD_LOGIC;
          STATE: out STD_LOGIC_VECTOR (2 downto 0));
end MTRACK;

architecture Behavioral of MTRACK is
```

```
component Finite_State_Machine is
   Port (BTN_UP, BTN_DOWN, BTN_CONTROL : in STD_LOGIC;
        Control_Return, downward : in STD_LOGIC;
        CLK : in STD_LOGIC;
        Signal_V, Signal_MC : out STD_LOGIC;
        STEPPER_U, STEPPER_D : out STD_LOGIC;
        STATE: out STD_LOGIC_VECTOR (2 downto 0);
        Control_Reset: out STD_LOGIC);
end component;
component STEPPER_DRIVER is
   Port ( CLK : in STD_LOGIC;
        BTN_0 : in STD_LOGIC;
        BTN_1 : in STD_LOGIC;
        STEPPER : out STD_LOGIC);
end component;

component voltage_visualization is
   Port ( CLK : in STD_LOGIC;
        Volt_in : in STD_LOGIC;
        Volt_out : in STD_LOGIC;
        Volt_value : out STD_LOGIC_VECTOR (9 downto 0);
        DISPLAY_EN : out STD_LOGIC_VECTOR (3 downto 0);
        SS_DISP : out STD_LOGIC_VECTOR (7 downto 0));
end component;

component voltage_comp is
   Port ( VP : in STD_LOGIC_VECTOR (9 downto 0);
        V2 : in STD_LOGIC_VECTOR (9 downto 0);
        TG : out STD_LOGIC);
end component;

component count_maximum is
   Port ( CLK : in STD_LOGIC;
        Reset_FSM: in STD_LOGIC;
        RESET : in STD_LOGIC;
        Signal_MC : in STD_LOGIC;
        Control_Return : out STD_LOGIC);
end component;

component Count_Vertical is
   Port ( CLK : in STD_LOGIC;
        Signal_V : in STD_LOGIC;
        downward: out STD_LOGIC);
```

```
end component;

component Register is
  Port (CLK : in STD_LOGIC;
        EN : in STD_LOGIC;
        V1: in STD_LOGIC_VECTOR(9 downto 0);
        V2: out STD_LOGIC_VECTOR(9 downto 0):= "0000000000" );
end component;
end Behavioral;
```

Finite State Machine code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Finite_State_Machine is
   Port (BTN_UP, BTN_DOWN, BTN_CONTROL: in STD_LOGIC;
         Control_Return, downward: in STD_LOGIC;
         CLK: in STD_LOGIC;
         Signal_V, Signal_MC: out STD_LOGIC;
         STEPPER_U, STEPPER_D: out STD_LOGIC;
         STATE: out STD_LOGIC_VECTOR (2 downto 0);
         Control_Reset: out STD_LOGIC);
end Finite_State_Machine;

architecture Behavioral of Finite_State_Machine is
   type state is (man, vert_sweep, vert_max);
   signal Preset_State: state;
   signal Next_State: state;
begin

sync_proc: process(CLK, Next_State)
begin
   if rising_edge(CLK) then
      Present_State <= Next_State;
   end if;
end process sync_proc;

comb_proc: process(Present_State, BTN_UP, BTN_DOWN, BTN_CONTROL, Control_Return,
downward)
begin
   STEPPER_P <= '0';STEPPER_D <= '0';
   case Present_State is
```

```
when manual =>
   STATE <= "001";
   if (BTN_CONTROL = '1') then
      Next_State <= vertical_sweep;
      Control_Reset <= '0';
      Signal_V <= '1';
      Signal_MC <= '0';
   else
      if (BTN_UP = '1') then
         STEPPER_U <= '1'; STEPPER_D <= '0';
      else
         STEPPER_U <= '0';
      end if;
      if (BTN_DOWN = '1') then
         STEPPER_D <= '1'; STEPPER_U <= '0';
      else
         STEPPER_D <= '0';
      end if;
      Next_State <= manual;
      Control_Reset <= '1';
      Signal_V <= '0';
      Signal_MC <= '0';
   end if;
when vertical_sweep =>
   STATE <= "010";
   Control_Reset <= '0';
   if (downward = '1') then
      STEPPER_D <= '1'; STEPPER_U <= '0';
      Next_State <= vertical_sweep;
      Signal_V <= '1';
      Signal_MC <= '0';
   else
      STEPPER_D <= '0'; STEPPER_U <= '0';
      Next_State <= vertical_maximum;
      Signal_V <= '0';
      Signal_MC <= '1';
   end if;
when vertical_maximum =>
   STATE <= "100";
   Control_Reset <= '0';
   if (Control_Return = '1') then
      STEPPER_U <= '1'; STEPPER_D <= '0';
      Next_State <= vert_max;
      Signal_V <= '0';
```

```
              Signal_MC <= '1';
            else
              STEPPER_U <= '0'; STEPPER_D <= '0';
              Next_State <= man;
              Signal_V <= '0';
              Signal_MC <= '0';
            end if;
        when others =>
            Next_State <= manual; STEPPER_U <= '0'; STEPPER_D <= '0';
            Control_Reset <= '1'; Signal_V <= '0'; Signal_MC <= '0';
            STATE <= "000";
     end case;
end process comb_proc;
end Behavioral;
```

Voltage Visualization code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

library UNISIM;
use UNISIM.VComponents.all;

entity voltage_visualization is
   Port ( CLK : in STD_LOGIC;
          Volt_in : in STD_LOGIC;
          Volt_out : in STD_LOGIC;
          Volt_value : out STD_LOGIC_VECTOR (9 downto 0);
          DISPLAY_EN : out STD_LOGIC_VECTOR (3 downto 0);
          SS_DISP : out STD_LOGIC_VECTOR (7 downto 0));
end voltage_visualization;

architecture Behavioral of voltage_visualization is

component adc is
   Port ( Volt_in : in STD_LOGIC;
          Volt_out : in STD_LOGIC;
          clk : in STD_LOGIC;
          d_rd : out STD_LOGIC;
          d_out : out STD_LOGIC_VECTOR (15 downto 0));
end component;

component sseg_decoder is
```

```
   Port ( ALU_Value : in std_logic_vector(9 downto 0);
                      _sign_ : in std_logic;
                    val : in std_logic;
        CLK : in std_logic;
        DISPLAY_EN : out std_logic_vector(3 downto 0);
        SEG: out std_logic_vector(7 downto 0));
end component;

signal valid, sign_ : STD_LOGIC;
signal valu : STD_LOGIC_VECTOR (15 downto 0);

begin

valid__ <= '1';
sign_ <= '0';
V_value <= value(15 downto 6);

adc0 : adc
   port map( Volt_in => Volt_in,
          Volt_out => Volt_out,
          clk => CLK,
          d_rd => open,
          d_out => value);

sseg0 : sseg_dec
   port map( ALU_VAL => value(15 downto 6),
          _sign_ => sign_,
          VAL => valid__,
          CLK => CLK,
          DISPLAY_EN => DISPLAY_EN,
          SEG => SS_DISP);

end Behavioral;
```

Voltage Comparator code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity voltage_comp is
   Port ( VP: in STD_LOGIC_VECTOR (9 downto 0);
        V2 : in STD_LOGIC_VECTOR (9 downto 0);
        TG: out STD_LOGIC);
```

```
end voltage_comp;

architecture Behavioral of voltage_comp is

begin
comp : process(VP, V2)
begin
   if VP(9 downto 4) > V2(9 downto 4) then
      TG <= '1';
   else TG <= '0';
   end if;
end process comp;

end Behavioral;
```

Maximum Counter code

```
library IEEE;
use IEEE.STD_LOGIC_unsigned.ALL;
use IEEE.STD_LOGIC_1164.ALL;

entity count_ maximum is
   Port (CLK: in STD_LOGIC;
         Reset_FSM: in STD_LOGIC;
         RESET: in STD_LOGIC;
         Signal_MC: in STD_LOGIC;
         Control_Return: out STD_LOGIC);
end count_ maximum;

architecture Behavioral of count_ maximum is

begin

clock_count: process(CLK, Reset_FSM, RESET, Signal_MC)
variable current_count: STD_LOGIC_VECTOR(12 downto 0):= "0000000000000";
begin
      if RESET= '1' or Reset_FSM = '1' then
         current_count := "0000000000000";
         Control_Return <= '0';
      elsif Rising_Edge(CLK) then
         if Signal_MC = '0' then
            current_count := current_count + 1;
            Control_Return <= '0';
```

```
        elsif Signal_MC = '1' then
            current_count := current_count - 1;
            if current_count = "000000000000" then
                Control_Return <= '0';
            else
                Control_Return <= '1';
            end if;
        end if;
    end if;
end process count_clock;

end Behavioral;
```

Vertical Counter code

```
library IEEE;
use IEEE.STD_LOGIC_unsigned.ALL;
use IEEE.STD_LOGIC_1164.ALL;

entity count_vertical is
    Port (CLK : in STD_LOGIC;
           Signal_V : in STD_LOGIC;
           downward: out STD_LOGIC);
end count_vertical;

architecture Behavioral of count_vertical is

begin

clock_count: process(CLK, Signal_V)
variable current_count : STD_LOGIC_VECTOR(11 downto 0):= "000000000000";
begin
    if Rising_Edge(CLK) then
        if Signal_V = '1' then
            current_count := current_count + 1;
            if current_count = "111111111111" then
                downward <= '0';
            else
                downward<= '1';
            end if;
        elsif Signal_V = '0' then
            current_count := "000000000000";
```

```
          downward<= '0';
       end if;
    end if;
end process clock_count;

end Behavioral;
```

Register code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Register is
  Port (CLK : in STD_LOGIC;
        EN : in STD_LOGIC;
        V1: in STD_LOGIC_VECTOR(9 downto 0);
        V2 : out STD_LOGIC_VECTOR(9 downto 0):= "0000000000" );
end Register;

architecture Behavioral of Register is
signal maintain: STD_LOGIC_VECTOR (9 downto 0) := "0000000000";
begin
comp: process(CLK, V1, V2)
    begin
       if rising_edge(CLK) then
          if EN <= '1' then
             V2 <= V1;
             maintain <= V1;
          else
             LV <= maintain;
          end if;
       end if;
    end process comp;
end Behavioral;
```

Stepper Motor code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
```

```
entity STEPPER_DRIVER is
    Port (CLK : in STD_LOGIC;
           BTN_0 : in STD_LOGIC;
           BTN_1 : in STD_LOGIC;
           STEPPER : out STD_LOGIC);
end STEPPER_DRIVER;

architecture Behavioral of STEPPER_DRIVER is

component pwmodulation_control is
    Port (
          CLK : in STD_LOGIC;
          DIRECTION : in STD_LOGIC_VECTOR (1 downto 0);
          EN : in STD_LOGIC;
          STEPPER : out STD_LOGIC);
end component;

component clk_divider2 is
    Port (
           clk : in std_logic;
           sclk : out std_logic);
end component;

signal direction: STD_LOGIC_VECTOR (1 downto 0);
signal inter_clk: STD_LOGIC;

begin

process(BUTTON_0, BUTTON_1)
begin
    if BUTTON_0 = '0' and BUTTON_1 = '0' then
        direction <= "00";
    elsif BUTTON_0 = '1' and BUTTON_1 = '0' then
        direction <= "01";
    elsif BUTTON_0 = '0' and BUTTON_1 = '1' then
        direction <= "10";
    else
        direction <= "00";
    end if;
end process;

pwm_control_0 : pwm_control
    port map(
            CLK => inter_clk,
            DIRECTION => direction,
            EN => '1',
            STEPPER => STEPPER);

clk_div2_0 : clk_div2
    port map(
            clk => CLK,
            sclk => inter_clk);

end Behavioral;
```