DEVELOPMENT OF SMART BEEHIVE USING CONVOLUTIONAL NEURAL NETWORK WITH AUTOMATED TREATMENT OF VARROA MITES INFESTATION

A Project Study Presented to the Faculty of

Electronics Engineering Department

College of Engineering

Technological University of the Philippines

In Partial Fulfillment of the Course Requirements for the Degree of

Bachelor of Science in Electronics Engineering

Submitted by:

Cariño, Ian Jasper G.

Elequin, Elaine Joy L.

Gangcuangco, Cherrie Lynne R.

Jayomana, Ellen Grace S.

Laman, Lalaine B.

Adviser:

Engr. Nilo M. Arago

August 2020

APPROVAL SHEET

This project study entitled **"DEVELOPMENT OF SMART BEEHIVE USING CONVOLUTIONAL NEURAL NETWORK WITH AUTOMATED TREATMENT OF VARROA MITES INFESTATION",** has been prepared and submitted by the following proponents:

Cariño, Ian Jasper G.
Elequin, Elaine Joy L.
Gangcuangco, Cherrie Lynne R.
Jayomana, Ellen Grace S.
Laman, Lalaine B.

In partial fulfillment of the requirements for the degree of **Bachelor of Science in Electronics Engineering** is hereby recommended for approval.

ENGR. NILO M. ARAGO
Project Adviser

ENGR. TIMOTHY M. AMADO
Panel Member

08/14/20

ENGR. CHERRY G. PASCION
Panel Member

JOHN PETER M. RAMOS 08/14/2020

ENGR. JOHN PETER M. RAMOS
Panel Member

E-signature of: JAY FEL C. QUIJANO, ECE
Signed: August 11, 2020

ENGR. JAY FEL C. QUIJANO
Panel Member

Accepted and approved in partial fulfillment of the course requirements for the degree of **Bachelor of Science in Electronics Engineering**.

ENGR. EDMON O. FERNANDEZ
Head, ECE Department

ENGR. BENEDICTO N. FORTALEZA
Dean, College of Engineering

ACKNOWLEDGEMENT

This project study became a reality with the kind support and help of many individuals. The proponents would like to extend their sincerest gratitude to all of them.

Foremost, we would like to offer thanks to God Almighty for the wisdom He bestowed upon us, for the strength and courage to work hard, and for paving the road for opportunities in making our project study.

We would also like to express our gratitude to our ever-supportive family who served as an inspiration to us and encouraged us throughout.

To our great adviser, Engr. Nilo M. Arago, for the guidance, constant supervision, and for providing necessary information regarding our project as well as for the support in this endeavor.

To our panel members, Engr. Timothy M. Amado, Engr. Jay Fel C. Quijano, Engr. John Peter M. Ramos, and Engr. Cherry G. Pascion, who approved our study and for giving us a recommendation to further improve and strengthen the purpose of our study.

We would like to give our special gratitude to Mr. Dexter Dandan and his family, for imparting knowledge about beekeeping, for assisting us in making our hive and for his constant support.

Our thanks and appreciation to all the people who encouraged and helped us with all of their abilities.

ABSTRACT

In the Philippine apiculture industry, most beekeepers still practice traditional beekeeping methods to maintain the healthy state of a colony. Since the manual inspection of the beehives is a labor-intensive and intrusive traditional method, the proponents developed and assembled a Smart Beehive with an automated treatment system for varroa mite infestation. Incorporated with the standard Langstroth beehive, the hive measures 17.5 x 27 x 14 inches made of hardwood with an aluminum roof. The technology used load cells to measure the weight of each frame, a DHT22 sensor to monitor temperature and humidity levels inside the beehive, a GSM module to notify the beekeeper on the mite count, and microscope USB cameras placed above the entrance of the beehive that detects varroa mites infestation on bees via image recognition using convolutional neural network.

Tests were conducted to compare the significant difference of using a weighing scale and load sensor, thermometer, hygrometer, and temperature and humidity sensor. A paired t-test was used to determine the reliability of the sensor and based on the p-values obtained, the sensors are accurate and can be used to replace the digital tools. The mites were detected with 99% accuracy. All the data acquired, and the time of updates are all stored in the database. For remote monitoring, the mobile application BeeSmart is used to access the data gathered and to control the automated treatment. In conclusion, smart beehive reduces the need for manual inspection and monitoring and increases efficiency in beekeeping.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

THE PROBLEM AND ITS BACKGROUND

1.1 Introduction

As the world's population grows rapidly, the need for diverse, balanced, and quality food increases to ensure the well-being of humankind. Bees are known for their significant role in pollination and production of honey and other products. Since the Philippines is a tropical country that is rich with biodiversity and natural resources, it is a suitable place for beekeeping. Beekeeping is a growing industry that requires low investments. Also, the market potential of the products produced in beekeeping is high, locally, and internationally. Moreover, according to Rita dela Cruz of the Bureau of Agricultural Research, with the appropriate action and methods, it is seen to deal with food security and supply profitable opportunities for the Philippines. Efficient beehive maintenance is a huge factor for successful beekeeping.

Several challenges pose a threat to maintaining a good and healthy beehive. These challenges include monitoring and inspection of the beehive and pest infestation. According to the Philippine Council for Agriculture, Aquatic and Natural Resources Research and Development of the Department of Science and Technology (DOST-PCAARRD), hive management and inspection should be done weekly or thrice every two weeks. In the Philippines, inspection is commonly done manually, which is both intrusive to

the colony and time and labor-intensive to the beekeepers. Moreover, manual inspection helps determine whether there is a presence or absence of pests.

One of the most serious pests of bees is the Varroa mites which are typically seen on European honeybees or Apis Mellifera – the commonly used species in beekeeping due to its abundant honey production. The Varroa mites attach itself to the bee and weaken the bee by sucking its blood which results in the deformed wing. A significant mite infestation will lead to the death of the colony.

1.2 Background of the Study

Beekeeping is a demanding occupation that requires a lot of time and effort. Nowadays, bees are declining at a worrying pace. The task of a beekeeper is to take care of the bees by observing and monitoring their health and conducting beehive inspection to prevent pest infestation. However, beehive inspection disturbs the colony and decreases the labor productivity of beekeepers.

According to Gross, et al. (2020), throughout the last 50 years, Varroa mite infestation on Apis Mellifera has raised great concern among beekeepers internationally. Detecting early infestation of varroa mites can have a big impact. Some conventional methods of detecting mite infestation can affect the quality of the honey produced; some sacrifice a certain number of bees to save the whole colony. Existing treatments on the market are quite

expensive and inaccessible. Also, mites are immunized from these treatments after several uses.

Innovation on beekeeping is constantly improving. Previous foreign studies have proven a smart monitoring system for beehives. However, due to its high cost, Philippine beekeepers haven't adapted to these technological advances and still use the traditional manual way of monitoring hives and collecting honey. Furthermore, existing studies can be enhanced to cope up with the demands of the apiculture industry in the country.

1.3 Statement of the Problem

Previous studies have proven various ideas, schemes, and designs to deal with beekeeping, with each approach having its advantages and disadvantages. The maintenance of the hive is significant to the production of honey. The traditional way of monitoring and detecting pests on the beehive is still observed by most beekeepers. However, such a method takes a lot of time, effort, and disturbance to the colony. Data gathered from the traditional beekeeping methods are recorded in journals that can often be lost. Moreover, these data are mostly inaccessible and inaccurate.

An existing study recommends the higher accuracy of image recognition of the Varroa mites infesting on Apis Mellifera. In addition to this, bees enter the hive in various manners, making it difficult for the camera to determine whether a mite is present. Moreover, the number and size of the entrances of the hives is significant in the protection of the colony against the

invasion of robbing honey bees, cold temperatures, and productivity of the colony.

1.4 Objectives of the Study

1.4.1 General Objective

This study aims to develop a smart beehive using a Convolutional Neural Network with the automated treatment of varroa mites infestation.

1.4.2 Specific Objectives

1. To construct a smart beehive with multiple entrances, having a Sensor Network System which will acquire the thermal reading, atmospheric moisture, and weighing system of a honey frame using Raspberry Pi.
2. To develop an image recognition system using Convolutional Neural Network which will detect the early infestation of Varroa mites on the species Apis Mellifera within the beehive.
3. To develop a notification system using the GSM module and enhance the mobile application that will aid the beekeepers in monitoring the condition of the beehive.
4. To design an automated system for the treatment of the beehive infested with varroa mites.
5. To test and evaluate the efficiency of the system.

1.5 Significance of the Study

The study aims to ease the monitoring of the hive and the detection of Varroa mites. As a result, the good condition of the hives and better health of the bees will be achieved. The system will also inform the beekeeper if a honey frame is ready for harvesting if there are a significant temperature and humidity change inside the beehive and if there is a mite infestation within the hive. Thus, the production rate will increase, the cost for replacement of an infested colony will decrease, and the amount of time and labor allotted for the inspection of the beehives will be reduced.

1.6 Scope and Limitation

This study focuses on real-time beehive monitoring and detection of early infestation of Varroa mites on Apis Mellifera. Apis Mellifera is the type of bee in the system since these species are commonly used in commercial beekeeping and are usually infested by Varroa mites.

Weight sensors are used to monitor the weight of each honey frame, temperature, and humidity sensors are installed inside the hive and four entrances with USB microscope cameras each for capturing the image of bees entering the beehive and transmitting it to the Raspberry Pi to be processed and uploaded to the database. The real-time status of the beehive is accessible through the BeeSmart mobile application. Also, basic treatment for the infested beehive is implemented. Other parameters that the researchers might

encounter during the conduction of this study is subjected to future development.

CHAPTER 2
REVIEW OF RELATED LITERATURE AND STUDIES

This chapter presents the related studies useful in the development of the idea in conceptualizing the project.

2.1 Related Studies on Sensor Network System

2.1.1 Implementation of a multi-node temperature measurement system for bee colonies online monitoring.

This study by Aleksejs Zacepins and Jurijs Meitalovs focuses on the development of a multi-node temperature measurement system for bee colonies. The temperature measurement system is an online monitoring system that consists of two major parts. First is the remote measurement nodes for real-time bee colony temperature measurement. The second major part is the webserver measurement summarising and demonstrating to the end-users. This monitoring system helps beekeepers obtain actual and timely data and information. Also, this system is a useful beekeeping method to improve beehive management efficiently. (Zacepins, and Meitalovs, 2014)

2.1.2 Development of a daily activities monitoring system for honeybee activities.

This study develops a honeybee behavior monitoring system. The system monitors the in and out daily activities of bees. It automatically records environmental parameters inside and near the beehive. The information and

data collected by the system will help the researchers understand the connection between bees and their environment. (Chen et al., 2015)

2.1.3 Beekeeping in the future - Smart apiary management

This study presents the author's vision for the implementation of precise beekeeping and smart apiary concept. It monitors different parameters of the bee colony like temperature, humidity, gas content, sound vibration, etc. Wired and wireless data transmission are monitored here. Usually, apiaries are located in urban areas that is why alternative energy is needed for powering all the devices. Solar panels mounted on the hive is the most relevant alternative power supply. Automatic beehive heating and cooling system implementation into practice is considered. Cooling maintains the stable temperature in the hive and heating helps save the colony's physiological resources. (Ahrendt et al., 2016)

2.1.4 Low-Cost Platform for Monitoring Honey Production and Bees Health

This study presents a low-cost platform for monitoring honey production and the health of bees in a colony. Monitoring includes the weight, temperatures inside and outside the beehive, the humidity, and the CO_2 concentration of the beehive. The system can estimate the honey production and the health of the bees. Also, pests can be detected such as Aethina tumida beetle, and Varroa destructor mite. This system is meant to be accessible to a

a large number of beekeepers. The data is sent via SMS to the beekeeper at his request or if there is damage to the colony. (Adochiei et al., 2018)

2.1.5 Design and Development of a smart weighing scale for beehive monitoring.

This study describes the development of a wireless platform weighing scale, as a part of the smart beehive. A single point impact cell was integrated into the design of the scales. The final weighing system was linked via a high precision analog to digital converter (ADC) to an off-the-shelf processing platform enabled with a low power Zigbee radio, to allow for data transfer to the base station. The productivity of the colony, its health, and condition can exactly be affected by its changes in weight. One of the key metrics of the strength of the beehive is the weight of the colony. (Fitzgerald et al., 2015)

2.2 Related studies on image recognition using Convolutional Neural Network

2.2.1 Recognition of Pollen-bearing Bees from Video using Convolutional Neural Network

This study is about the recognition of pollen-bearing honey bees from videos of the entrance of the hive. Several approaches are considered for the computer vision task. This includes baseline classifiers, shallow Convolutional Neural Network, and deeper networks from literature. The video capture system is designed to observe the bees that are entering the hive. Annotated datasets were released for public access. This contains several images of pollen and

non-pollen bearing bees. In conclusion, simple CNN architecture performed better than pre-trained CNN models. (Rodriguez et al., 2018)

2.2.2 Basic Algorithms for Bee Hive Monitoring and Laser-based Mite Control

This study implements a beehive monitoring system to monitor parameters in beehives such as temperature, sound, and weight. Different methods were used for mite recognition. Histogram analysis, Hough Transformation and Region Labeling are the methods that were used for image recognition. A laser structure was constructed to control the mites. Information systems are established for gathering information such as temperature in the hive, humidity, rain, weight, and images. This data is important to inform the beekeeper of the health condition of the colonies. (Becker and Szczerbicka, 2016)

2.2.3 Face recognition based on histogram equalization and convolution neural network

This study developed a face recognition method based on histogram equalization and convolutional neural network to improve the recognition rate of face recognition. Principally, the histogram equalization method is utilized to preprocess the image. Subsequently, Google deep learning framework TensorFlow1.3.0 is used to build a convolutional neural network. Its structure is based on the LeNet-5 model and the network is programmed by preprocessed face images. Lastly, the samples are entered into the finalized convolutional

a neural network where the recognition rate is provided. As a result of using the method of histogram equalization and convolution neural network to analyze the face images of the ORL face database, a high recognition rate using the algorithm is obtained. (Lu and Yue, 2018)

2.2.4 Content-based Image Retrieval using Deep Convolutional Neural Network

This paper focuses on a content-based Image Retrieval system using a Deep Convolutional Neural Network. Deep Convolutional Neural Network is widely used by researchers to analyze images for a variety of applications. Experiments are performed on Gray images, RGB color space, YCbCr color space images ensembled into different clusters. A Precision-recall crossover point is used as performance measurement criteria. (Christanti, Hendryli, and Rian, 2019)

2.2.5 Adaptive Wireless Sensor Networks for High-Definition Monitoring in Sustainable Agriculture

The study presents an adaptive solution for the realization of ad hoc wireless sensor networks suitable to transport large quantities of data over narrowband channels. The electromagnetic front-end is accurately designed to maintain radiation efficiency even when inserted in media with random dissipative characteristics. The use of narrowband channels, as well as the possibility to decrease the carrier down to 180 MHz (compatibly with regulatory authorizations), provides robustness to interferences, decreases

signal reactivity, and non-line-of-sight coverage. For all these reasons, the platform is being applied to cultivations, to construct an effective and low-cost instrument, which facilitates sustainable agriculture. (de la Concepcion, Stefanelli, and Trinchero, 2014)

2.3 Related studies on connecting the Raspberry Pi directly to a Cloud and alert monitoring system via GSM module.

2.3.1 Implementation of cloud server for real-time data storage using Raspberry Pi

This study includes a private cloud server that is set up into a Raspberry Pi which can be used as a storage device for real-time signal applications. A Raspberry Pi is a microcontroller in which cloud platforms can be used to obtain cloud computing infrastructures. A cloud is a service that can store a large amount of data and provides access to a restricted user via the Internet. This study also used an Arduino microcontroller in which analog signals can be discretized and transmitted serially to a Raspberry Pi. Hence, the Raspberry Pi can be used as a cloud server. (Nigel, and Princy 2015)

2.3.2 Cloud robotics in the industry using Raspberry Pi

This project is a cloud robot that is used in industrial and manufacturing environments. It works on a ROS platform. Here we use a Raspberry Pi controller to control the various devices attached to it. For testing this implementation, Android phone, camera, DC motor, sensors and a

Raspberry Pi controller has been used. The movement of the robot is provided by DC motors and the direction is controlled from an android environment using the Robot Operating System (ROS). The controller and the receiver end are connected by Wi-Fi. The data input from the gas, temperature, and Infrared sensors is given to the Raspberry Pi controller. A camera is used to provide visual input of the surrounding environment to the robot. The data obtained by the sensors and cameras are processed by the controller and stored in the cloud. (Krishna et al., 2016)

2.3.3 Development of cloud-based light intensity monitoring system using Raspberry Pi

This paper presents a real-time remote Light intensity monitoring system using Raspberry Pi which enables a user to track lighting systems remotely. The main feature of this system is a light intensity being monitored instantaneously and all of the data are stored in a database for future use. This cloud-based monitoring system helps a user to take the necessary actions at the right time. With proper control, a user can achieve the desired output. Real-time remote monitoring systems show an effective solution that minimizes efforts and expenditures to achieve the desired result. And one of these is presented in this project. (Kumar and Jatoth, 2015)

2.3.4 Raspberry Pi: Data Logging IOT device

This study deals with the development of a data logger system. The developed system logs the data on the webserver for remote access. The

researchers used the Controlled Area Network (CAN) for communication and the Raspberry Pi acts as a microcontroller in this system. Data from the Raspberry Pi can be sent on a web server directly. Owncloud, Adafruit IO web server is an open-source that can be used for web hosting. This can operate without a private server. For access to the server, the user needs to authenticate its credentials. Any user can remotely access the data in the cloud by logging in to the server credential. (Tavade and Nasikkar, 2017)

2.3.5 Remote Monitoring of Energy Consumption Using Zigbee and GSM Networking

This paper deals with a power or energy monitoring system using Zigbee and GSM communication technology. A hardware unit or portable display (PD) is developed which communicates with an energy meter installed with the consumer's preference. The PD can communicate with the energy meter using a 2.4 GHz Zigbee Module. The meters must also be customized with the incorporation of a Zigbee module that allows communication with the PD within coverage ranges of a few meters around the energy meter's location. From this PD, the consumers can get the information on energy consumption, voltage, current, active, and reactive power corresponding to the real-time basis and also the date. Knowing the status, the consumer can take part in improving the power system reliability by consuming more energy when the power demand is less and wastage of power. (Mondal et al., 2014)

2.3.6 A wireless approach to real-time remote monitoring system examining environmental parameters using the feasibility of a GSM module.

This study came up with a wireless approach for monitoring a remote area that is difficult to manage by individuals in inaccessible areas. The researchers used a thermistor and a humidity sensor to sense the temperature and humidity around. Each sensor is connected to a GSM module. Whenever the parameters change, a notification is automatically sent to a number that contains the parameters. This study can be useful in the agricultural industry to save time and cost in monitoring environmental parameters. (Metha and Punetha, 2014)

2.4 Related studies on the automated pesticide sprinkler system

2.4.1 Solar Powered Automatic Pesticides Sprayer

This study was developed in India for agricultural purposes. Solar energy is used to supply the system by using solar-powered batteries. This system helps the remote areas that have no enough source of electricity and energy. A 3 watts solar panel, Arduino Uno, Relay driver, and GSM module are the main hardware in this system. This system is a four-wheeled vehicle with a nozzle and DC pump that sprays pesticide on a specified area. This vehicle is controlled by a GSM or SMS provided by the farmer. This can ensure the safety and direct contact of the farmer to the pesticides. (Vasanth et al., 2017)

2.4.2 Agriculture Robotic Vehicle-Based Pesticide Sprayer with Efficiency Optimization

This study focuses on developing a smart machine to aid the farmers in detecting pests, spraying fertilizers, and spraying pesticides on their farms. This study aims to provide safety to the farmers and a precise agriculture system. The proponents use microcontroller, cameras, motors, and terminal equipment as cost-effective equipment in designing the system. This agricultural vehicle helps the farmers in hitting their target production. This system is easy to operate and user friendly. This project is essential in countries where agriculture is the main source of economy. (Aishwarya, Archana & Umayal, 2015)

2.4.3 Automated Sprayer System for Variable Rate Application of Pesticides

This study focuses on the low-cost hardware and software architecture for the automatic application of pesticides, through tree detection in citrus plantations. The system is composed of two distributed subsystems. The first one is embedded in the tractor which is responsible for the positioning and tree inventory update. The second one is embedded in the air-assisted sprayer which detects the tree presence using ultrasonic methods. These experiments showed a suitable acquisition of the position of the system. It was confirmed that the designed prototype allows the detection of plants with an accuracy of 98%. The proposed project makes it possible to reuse existing vehicles by improving spraying. This will help to save pesticides by reduced cost and

pollution. The results of the system are extendable to other fruit fields that employ a spraying system. (Duran-Faundez et al., 2018)

2.4.4 Agricultural Robot for Automatic Ploughing and Seeding

This paper presents the development of a robot capable of performing operations such as plowing, seed dispensing, fruit picking, and pesticide spraying. The main component of this study is AVR At Mega microcontroller that supervises the whole process. The robot tills first the entire field and proceeds to plow, whilst simultaneously dispensing side by side. The device used for navigation is an ultrasonic sensor which continuously sends data to the microcontroller. The robot operates on automated mode inside the field, but outside, the field is strictly operated in manual mode. For manual control, the robot uses the Bluetooth pairing app as a control device and helps in the navigation of the robot outside the field. The camera and nozzle are installed at the end of the robot arm. On finding a plant the robot stops moving and the arms inspection of the plant from bottom to top is done using a binocular stereo vision system using one camera. Through the depth of information obtained, the robot can control the arm and to how much distance it should extend to focus the pest in the scope of the spray nozzle. A DSP board is used to control the spray nozzle. A sprinkler similar to a water sprinkler is used to spray pesticides in the field. The spraying of pesticides is done after the seeds are placed in the soil. (Abirami et al., 2015)

2.4.5 Automated Paraquat Sprayer by Using Raspberry-Pi

This paper presents an image processing algorithm that is used to take images of the plantation rows at regular intervals. Upon identifying the weeds in the image, the herbicide is sprayed directly on the weeds. The algorithm uses an erosion and dilation approach to detect weeds. The image of the color is converted into binary by extracting the green parts of the image. The amount of white pixels present in the region of interest is determined and the regions with a higher white pixel count than the predefined threshold are considered as weeds. The herbicide is stored in a container that is fitted with water pump motors attached to spray nozzles. A signal is sent from the Raspberry-Pi to the motor driver IC once the weeds are detected. The IC controls the water pump motors to spray the chemicals over the weed. (Swathi V et al., 2017)

CHAPTER 3

METHODOLOGY

3.1 Conceptual Framework

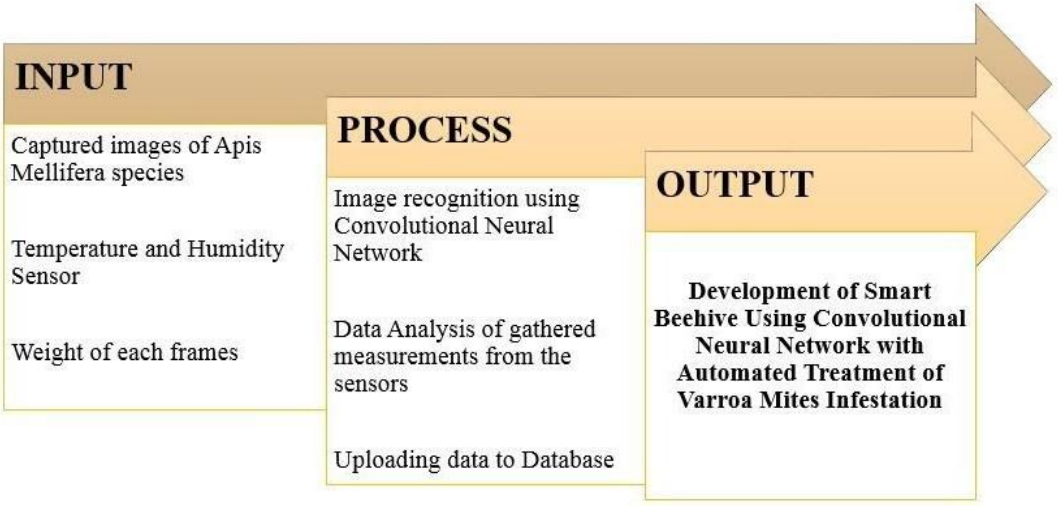


Figure 3.1. Input-Process-Output Flow

Figure 3.1 shows the Input-Process-Output Flow of the system. It shows how the study will be done. First, the input will be the captured images of species and the data gathered from the sensors. Subsequently, the process will be the data analysis of the captured images and the gathered data. Lastly, the output will be stored in the cloud that can be accessed through a mobile application and an alert notification through GSM.

3.2 Research Process Flow

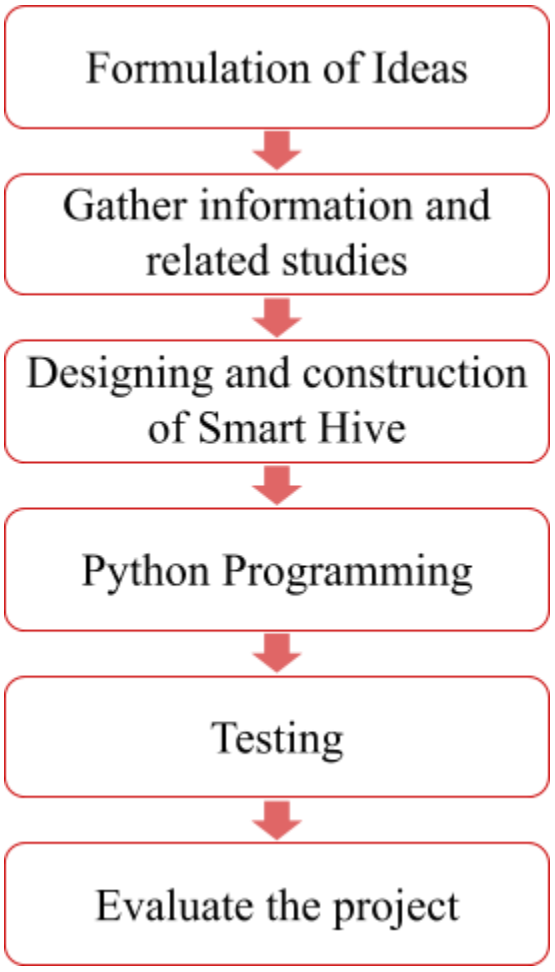


Figure 3.2. Research Process Diagram

Figure 3.2 shows the Research Process Diagram of the study. The first stage focuses on the formulation of the ideas of the study. The second stage focuses on gathering literature, articles, and previous studies. The third stage shall concentrate on designing and constructing the smart hive. For the fourth stage, programming shall be implemented using Python language. After the testing stage, the proponents shall evaluate the result.

In developing the specific objectives of this study, the proponents shall carry out the following processes:

3.3 To construct a smart beehive with multiple entrances, having a Sensor Network System which will acquire the thermal reading, atmospheric moisture, and weighing system of a honey frame using Raspberry Pi.

3.3.1 Design Conceptualization

3.3.1.1 Design a smart beehive based on the existing Langstroth hive that would satisfy the system's functionality.

3.3.1.2 Find and purchase a suitable temperature, humidity, and weight sensor needed. Buy Raspberry Pi that will serve as the microcontroller of the system, GSM module for SMS notification, and camera for image recognition system.

3.3.2 Design Assembly and Simulation

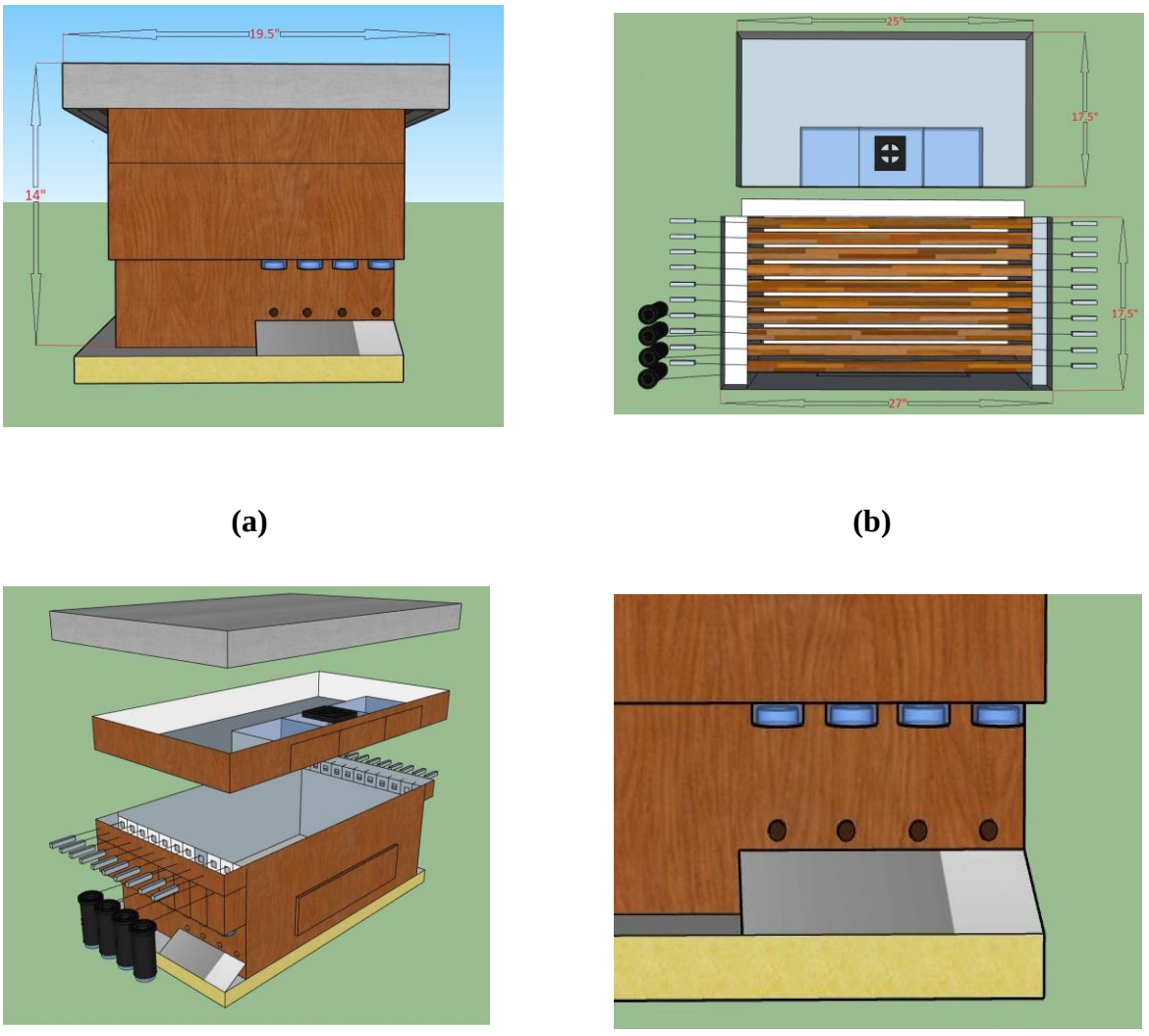
3.3.2.1 Mount the sensors and camera to their respective place in the beehive.

3.3.2.2 Connect the sensors to the Digital Input/Output (DI/O) of the Arduino Mega.

3.3.4.2 Design a system that will process the data gathered from all the sensors used.

3.3.3 Functionality Test

3.3.3.1 Test whether there is an output obtained from the condition of the beehive.



(a) **(b)**

(c) **(d)**

Figure 3.3. Smart Beehive

(a) Front View **(b)** Top View
(c) Isometric Left View **(d)** Camera Module at the Entrance

The design structure of the modified smart beehive includes:

1. An additional casing added on top of the beehive to serve as an enclosure for the treatment of varroa mites that are connected to the main hive via a tube.
2. USB cameras are placed above the entrance of the beehive for capturing the images of bees infected with mites. The data gathered from the cameras are

delivered to the image recognition system that will determine whether the hive is already infested and requires treatment or not.

3. A Sensor Network System which acquires the thermal reading and atmospheric moisture of the beehive, and the weight of the honey frames. The data gathered from the sensors are sent automatically to the database.
4. An Android application and a database are developed for remote monitoring and easy access to data results. Recent thermal and atmospheric moisture readings, the weight of each frame, varroa mites count threshold, and treatment options are shown in the application and database.

3.4 To develop an image recognition system using Convolutional Neural Network which will detect the early infestation of Varroa mites on the species Apis Mellifera within the beehive.

3.4.1 Simulation

3.4.1.1 Examine the proper positioning of the camera and mirror to have an accurate detection of the bees entering the hive.

3.4.1.2 Gather multiple images of actual bees with mites.

3.4.1.3 Design an algorithm for the control system of the camera using the Convolutional Neural Network.

3.4.2 Functionality Test

3.4.2.1 Test the accuracy of the camera and the effectiveness of the mirror.

3.5 To develop a notification system using the GSM module and enhance the mobile application that will aid the beekeepers in monitoring the condition of the beehive.

3.5.1 Design Simulation

3.5.1.1 Find a cloud server that can easily be accessed by the beekeeper.

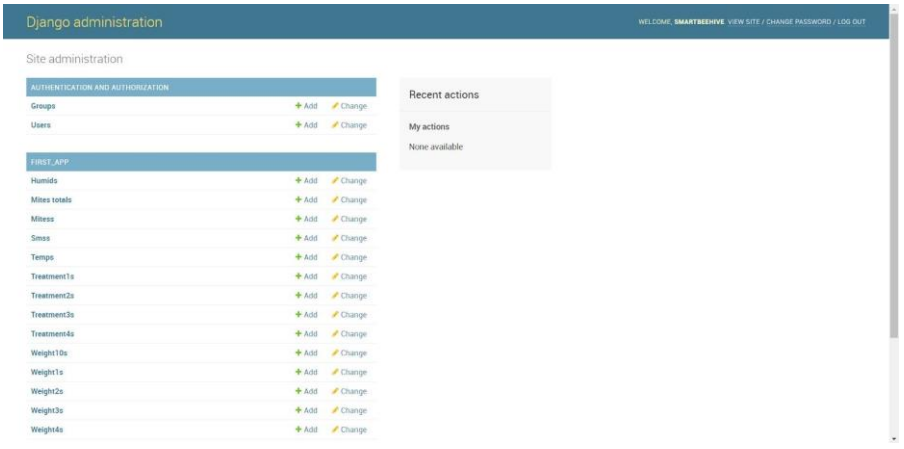


Figure 3.4. Database

3.5.1.2 Develop a code that will notify the beekeeper through their mobile phone via GSM.

3.5.1.3 Develop a code that will send the data gathered from the hives to the database.

3.5.1.4 Enhance the mobile application to monitor the temperature, humidity, weight of the frames, and a number of mites inside the beehive.

3.5.2 Functionality Test

3.5.2.1 Perform a dry-run to assess if the data acquired are sent in the database and the notifications are received by the mobile phone.

3.5.2.2 Launch the mobile application and compare whether the data stored in the database is the same as the data displayed in the mobile application.

3.6 To design an automated system for the treatment of the beehive infested with varroa mites.

3.6.1 Design Conceptualization

3.6.1.1 Design an automated treatment consisting of Alagao Leaves, Ultrasonic Mist maker, and API Strips.

3.6.1.2 Purchase all the materials needed for the automated treatment.

3.6.2 Design Assembly and Simulation

3.6.2.1 Assemble the materials inside the cover of the beehive.

3.6.2.2 Develop a code that will automate the system.

3.6.3 Functionality Test

3.6.3.1 Test the accuracy and effectiveness of alagao leaves, mist, and API strips as a treatment for infected beehives.

3.7 To test and evaluate the efficiency of the system.

CHAPTER 4

RESULTS AND DISCUSSIONS

This chapter contains the project technical description, project structural description, and tabulation and analysis of results relative to the tests conducted.

4.1 Project Technical Description

The technology developed for the project entitled “Development of Smart Beehive Using Convolutional Neural Network With Automated Treatment of Varroa Mites Infestation” aims to detect Varroa Mites Infestation on a beehive and to treat the infestation using the automated treatment system. The detection uses Convolutional Neural Network for image recognition of the bees entering the beehive. The technology also monitors the condition of the beehive. The parameters include the temperature, humidity, and weight of each frame inside the hive. To measure the weight of the frames, 20 load cells, (2 on each frame) are utilized. All the data gathered will be processed by the Raspberry Pi and can be accessed through a mobile application. Also, a notification message will be sent to the beekeeper if mites are detected.

4.2 Project Structural Design

Figure 4.1. Smart Beehive Prototype

Figure 4.1 shows the fabricated 10-frame Langstroth beehive of the study. The hive measures 17.5 inches in width, 25 inches in length, and 14 inches in height which corresponds to the standard size of the beehive.

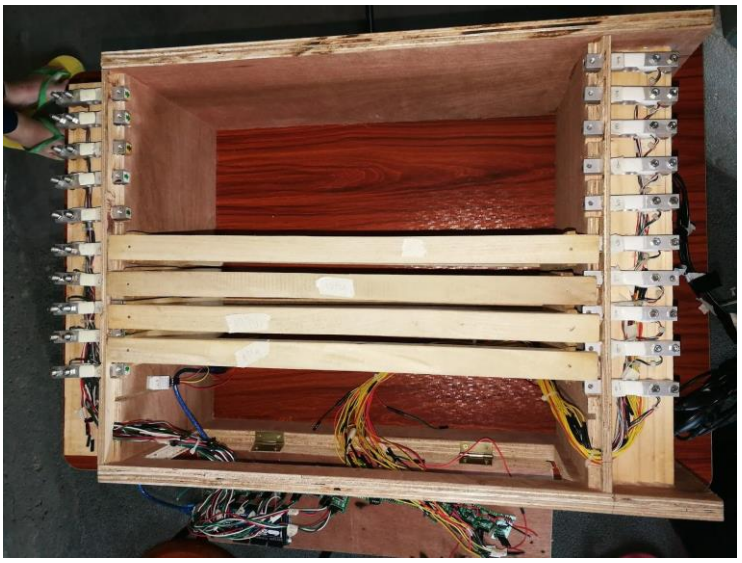


Figure 4.2. The actual placement of the sensors

Figure 4.2 shows the placement of the HX711 with a load cell on the frames of the beehive. A load cell is placed on each side of the frame having a total of 20 load cells for the 10 frames. The DHT22 is placed at the bottom of the beehive.

Figure 4.3. Setup at the entrance of the beehive

Figure 4.3 shows the multiple entrances of the hive and the mirror placed below the entrance with an angle of 30 degrees. The four USB microscope cameras capture the images of bees entering the hive. The images will then be sent to the Raspberry Pi and will be recorded on the database.

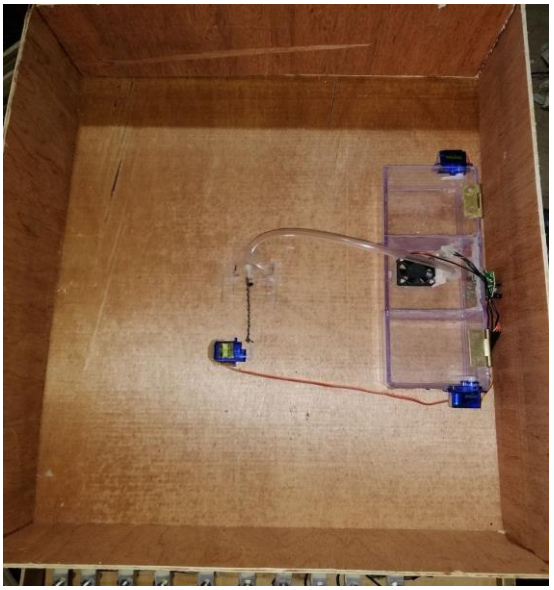
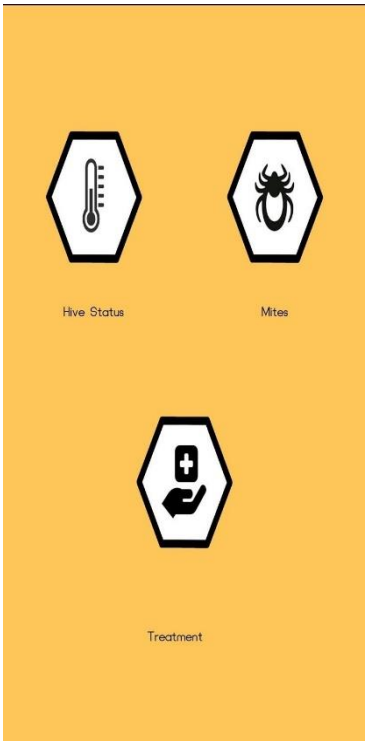


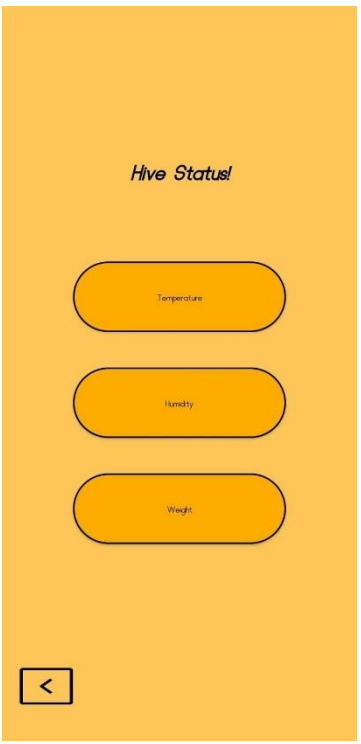
Figure 4.4. The automated treatment

The figure shows the automated treatment system for Varroa mites infestation in the hive. The treatment is designed with 3 servo motors. The first treatment is the *Alagaw* Leaves, the second is miticide strips and the third treatment is the ultrasonic mist maker with formic acid. The automated treatment is connected to the node MCU and Arduino Mega. The treatment can only be accessed through a mobile application.

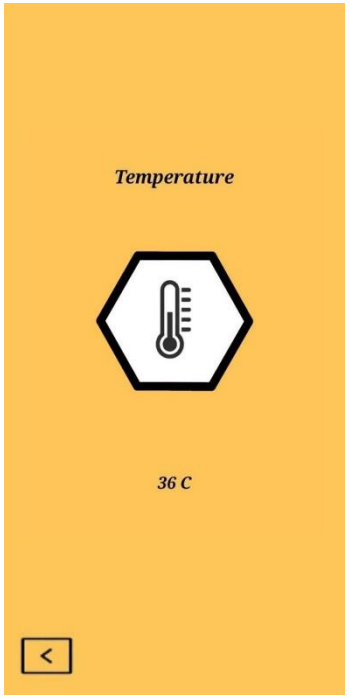
(a) App Log in



(b) GUI display

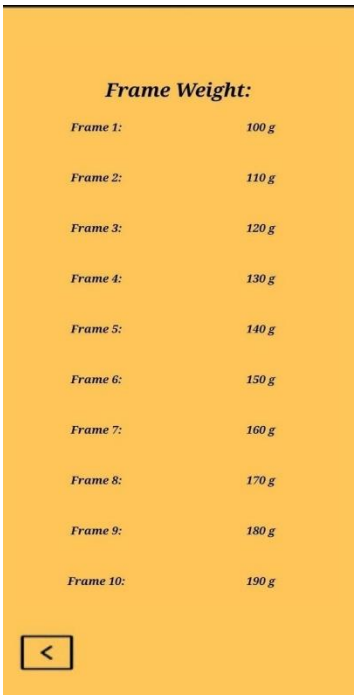


(c) Hive status display

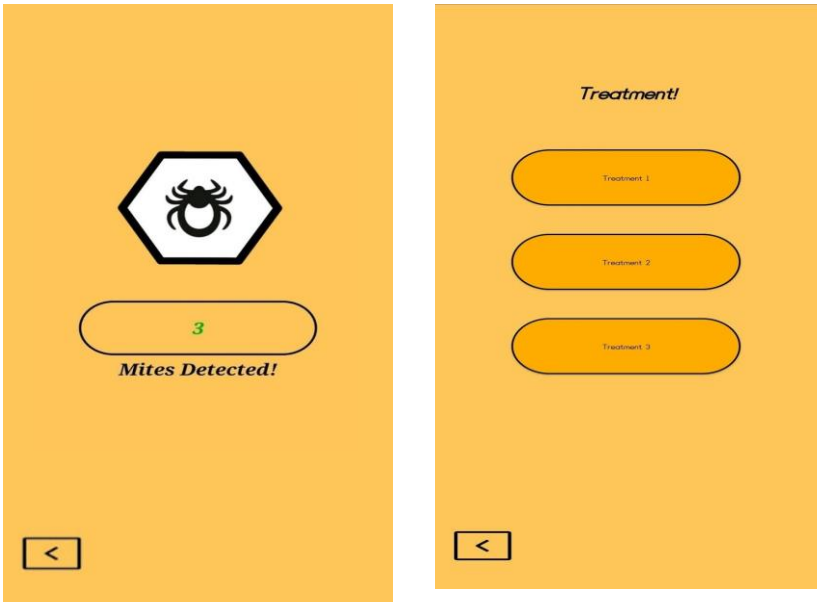


(d) Temperature data display

(e) Humidity data display



(f) Weight data display



(g) Mites data display **(h)** Treatment display

Figure 4.5. Mobile Application

The figure shows the mobile application containing the status of the beehive which includes the number of mites detected, the temperature and humidity status, the weight of each frame, the automated treatment for the infestation. The Basic 4 Android (B4A), a rapid development tool for android application, is used for building user interfaces.

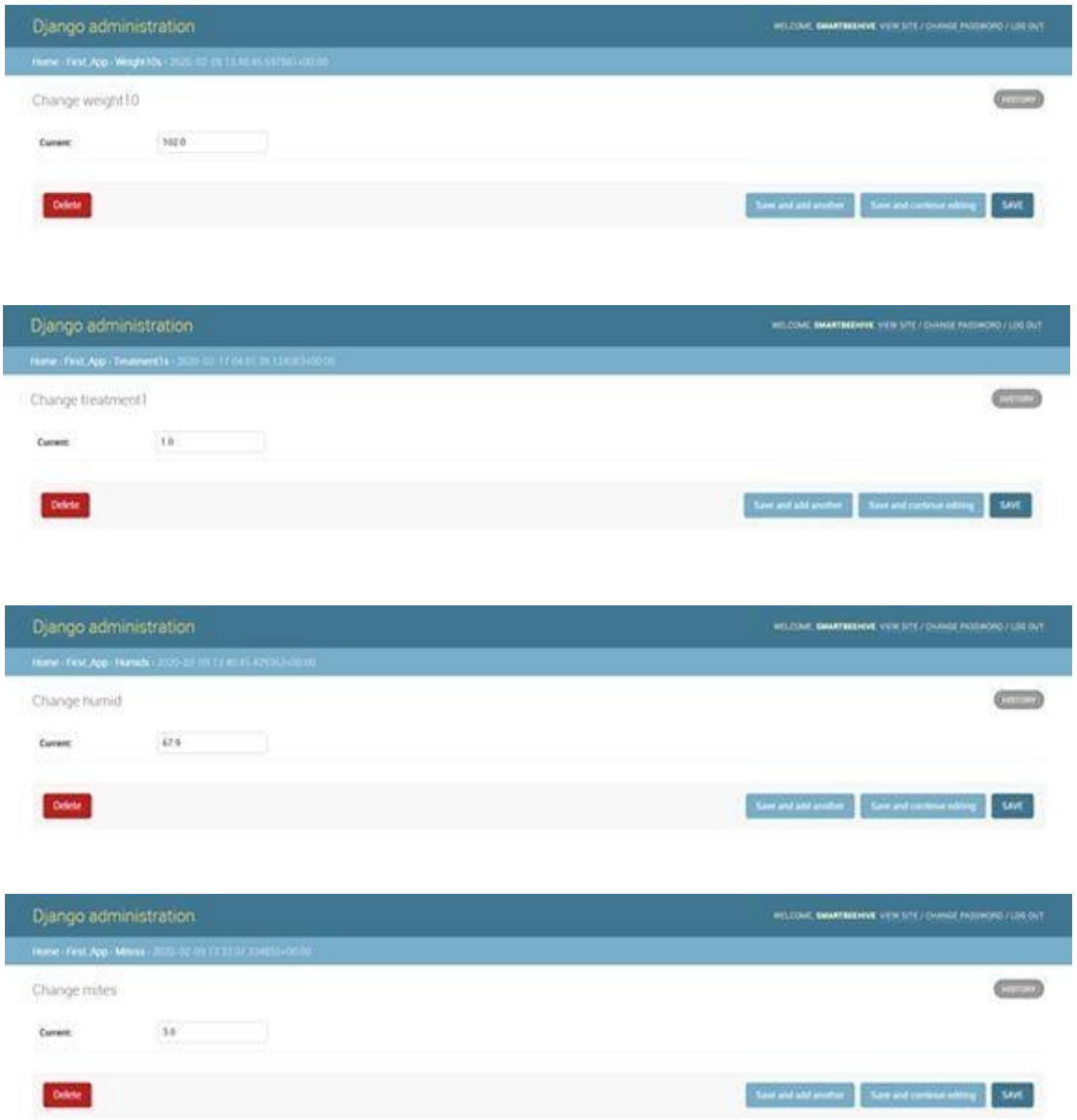


Figure 4.6. Data Received by the Webserver (Database)

The figures show the data received by the web server containing the status of the beehive which includes the number of mites detected, the temperature and humidity status, the weight of each frame, and the automated treatment for the infestation. The proponents used Django, a high-level web platform from Python that allows for the rapid creation of stable and maintainable websites.

4.3 Project Evaluation

4.3.1 Weight Measurement Evaluation

4.3.1.1 Weight Sensors Evaluation for paired load cells at Frame 1.

Table 4.1. Data for Digital Weighing Scale and Load cells for Frame 1.

Trial	Digital Weighing Scale (kg)	Load cells for Frame 1 (kg)
1	0.94	0.947
2	0.90	0.955
3	0.95	0.957
4	0.99	0.966
5	0.98	0.979
6	0.95	0.943
7	0.99	0.97
8	0.90	0.911
9	0.97	0.972
10	0.97	0.963

Table 4.1 shows the acquired measurements of a digital weighing scale and a load cell sensor for Frame 1. Ten trials were made to compare the significant difference between using a digital weighing scale and a load cell sensor. A paired t-test was used to compare the two methods to gather statistical data analysis.

The null hypothesis is that there is no significant difference between the measurement gathered by a digital weighing scale and a load cell sensor. The level of significance is 0.01 and the hypothesized mean difference is 0.

Table 4.2. Paired T-test for Weight Measurements of Frame 1.

SUMMARY			$\alpha = 0.01$	Hypothesized Mean Difference = 0		
Groups	*Count*	*Mean*	*Standard Deviation*	*Standard Error*	*t*	*df*
Digital Weighing Scale	10	0.954	0.031			
Load cells at (Frame 1)	10	0.9563	0.018			
Difference		-0.0023	0.013	0.0221	0.0023	9

t test		
	p-value	*t-critical*
One tail	0.373	2.8214
Two tail	0.746	3.2498

Based on the summary shown in Table 4.2, by using a paired t-test, the two-tailed p-value is 0.746 which is greater than the 0.01 significance level. Thus, the null hypothesis is accepted. The load cells and digital weighing scale have no significant difference. Therefore, the load cell sensor can be used to gather weight measurements of the frames inside the beehive.

4.3.1.2 Weight Sensors Evaluation for paired load cells at Frame 2.

Table 4.3. Data for Digital Weighing Scale and Load cells for Frame 2.

Trial	Digital Weighing Scale (kg)	Load cells for Frame 2 (kg)
1	1.20	1.185
2	1.20	1.192
3	1.25	1.245
4	1.27	1.265
5	1.25	1.252
6	1.30	1.311
7	1.31	1.315
8	1.32	1.323
9	1.32	1.333
10	1.32	1.327

Table 4.3 shows the acquired measurements of a digital weighing scale and a load cell sensor for Frame 2. Ten trials were made to compare the significant difference between using a digital weighing scale and a load cell sensor. A paired t-test was used to compare the two methods to gather statistical data analysis.

The null hypothesis is that there is no significant difference between the measurement gathered by a digital

weighing scale and a load cell sensor. The level of significance is 0.01 and the hypothesized mean difference is 0.

Table 4.4. Paired T-test for Weight Measurements of Frame 2.

SUMMARY			$\alpha = 0.01$	*Hypothesized Mean Difference = 0*		
Groups	*Count*	*Mean*	*Standard Deviation*	*Standard Error*	*t*	*df*
Digital Weighing Scale	10	1.274	0.055			
Load cells at (Frame 2)	10	1.2748	0.047			
Difference		-0.0008	0.008	0.0028	2.857	9

t test

	p-value	*t-critical*
One tail	0.3910	2.821
Two tail	0.7821	3.249

Based on the summary shown in Table 4.4, by using a paired t-test, the two-tailed p-value is 0.7821 which is greater than the 0.01 significance level. Thus, the null hypothesis is accepted. The load cells and digital weighing scale have no significant difference. Therefore, the load cell sensor can be used to gather weight measurements of the frames inside the beehive.

4.3.1.3 Weight Sensors Evaluation for paired load cells at Frame 3.

Table 4.5. Data for Digital Weighing Scale and Load cells for Frame 3.

Trial	Digital Weighing Scale (kg)	Load cells for Frame 3 (kg)
1	1.1	1.158
2	1.12	1.119
3	1.15	1.162
4	1.13	1.136
5	1.17	1.167
6	1.19	1.192
7	1.2	1.245
8	1.17	1.169
9	1.14	1.138
10	1.1	1.213

Table 4.5 shows the acquired measurements of a digital weighing scale and a load cell sensor for Frame 3. Ten trials were made to compare the significant difference between using a digital weighing scale and a load cell sensor. A paired t-test was used to compare the two methods to gather statistical data analysis.

The null hypothesis is that there is no significant difference between the measurement gathered by a digital

weighing scale and a load cell sensor. The level of significance is 0.01 and the hypothesized mean difference is 0.

Table 4.6. Paired T-test for Weight Measurements of Frame 3.

SUMMARY			*α= 0.01*	*Hypothesized Mean Difference = 0*		
Groups	*Count*	*Mean*	*Standard Deviation*	*Standard Error*	*t*	*df*
Digital Weighing Scale	10	1.147	0.035			
Load cells at (Frame 3)	10	1.170	0.038			
Difference		-0.023	-0.003	-0.001	-5.270	9

t test		
	p-value	*t-critical*
One tail	0.0451	2.821
Two tail	0.0902	3.250

Based on the summary shown in Table 4.6, by using a paired t-test, the two-tailed p-value is 0.0902 which is greater than the 0.01 significance level. Thus, the null hypothesis is accepted. The load cells and digital weighing scale have no significant difference. Therefore, the load cell sensor can be used to gather weight measurements of the frames inside the beehive.

4.3.1.4 Weight Sensors Evaluation for paired load cells at Frame 4.

Table 4.7. Data for Digital Weighing Scale and Load cells for Frame 4.

Trial	Digital Weighing Scale (kg)	Load cells for Frame 4 (kg)
1	1.2	1.232
2	1.22	1.268
3	1.2	1.295
4	1.25	1.247
5	1.23	1.273
6	1.26	1.257
7	1.24	1.241
8	1.27	1.273
9	1.25	1.259
10	1.21	1.215

Table 4.7 shows the acquired measurements of a digital weighing scale and a load cell sensor for Frame 4. Ten trials were made to compare the significant difference between using a digital weighing scale and a load cell sensor. A paired t-test was used to compare the two methods to gather statistical data analysis.

The null hypothesis is that there is no significant difference between the measurement gathered by a digital

weighing scale and a load cell sensor. The level of significance is 0.01 and the hypothesized mean difference is 0.

Table 4.8. Paired T-test for Weight Measurements of Frame 4.

SUMMARY			*α= 0.01*	*Hypothesized Mean Difference = 0*		
Groups	*Count*	*Mean*	*Standard Deviation*	*Standard Error*	*t*	*df*
Digital Weighing Scale	10	1.233	0.025			
Load cells at (Frame 4)	10	1.256	0.023			
Difference		-0.023	-0.003	-0.001	15.527	9

t test		
	p-value	*t-critical*
One tail	0.024	2.821
Two tail	0.047	3.250

Based on the summary shown in Table 4.8, by using a paired t-test, the two-tailed p-value is 0.047 which is greater than the 0.01 significance level. Thus, the null hypothesis is accepted. The load cells and digital weighing scale have no significant difference. Therefore, the load cell sensor can be used to gather weight measurements of the frames inside the beehive.

4.3.1.5 Weight Sensors Evaluation for paired load cells at Frame 5.

Table 4.9. Data for Digital Weighing Scale and Load cells for Frame 5.

Trial	Digital Weighing Scale (kg)	Load cells for Frame 5 (kg)
1	0.95	0.952
2	1	1.023
3	0.93	0.938
4	0.95	0.951
5	0.93	0.928
6	0.9	0.964
7	1	1.017
8	0.92	0.923
9	0.95	0.948
10	0.97	0.968

Table 4.9 shows the acquired measurements of a digital weighing scale and a load cell sensor for Frame 5. Ten trials were made to compare the significant difference between using a digital weighing scale and a load cell sensor. A paired t-test was used to compare the two methods to gather statistical data analysis.

The null hypothesis is that there is no significant difference between the measurement gathered by a digital

weighing scale and a load cell sensor. The level of significance is 0.01 and the hypothesized mean difference is 0.

Table 4.10. Paired T-test for Weight Measurements of Frame 5.

SUMMARY			$\alpha = 0.01$	*Hypothesized Mean Difference = 0*		
Groups	*Count*	*Mean*	*Standard Deviation*	*Standard Error*	*t*	*df*
Digital Weighing Scale	10	0.95	0.033			
Load Cell	10	0.9612	0.034			
Difference		-0.0112	0.001	0.0074	-1.50	9

t test	*p-value*	*t-critical*
One tail	0.058	1.833
Two tail	0.117	2.262

Based on the summary shown in Table 4.10, by using a paired t-test, the two-tailed p-value is 0.117 which is greater than the 0.01 significance level. Thus, the null hypothesis is accepted. The load cells and digital weighing scale have no significant difference. Therefore, the load cell sensor can be used to gather weight measurements of the frames inside the beehive.

4.3.1.6 Weight Sensors Evaluation for paired load cells at Frame 6.

Table 4.11. Data for Digital Weighing Scale and Load cells for Frame 6.

Trial	Digital Weighing Scale (kg)	Load cells for Frame 6 (kg)
1	0.72	0.723
2	0.7	0.689
3	0.74	0.744
4	0.69	0.692
5	0.75	0.758
6	0.7	0.703
7	0.73	0.733
8	0.69	0.694
9	0.78	0.782
10	0.75	0.757

Table 4.11 shows the acquired measurements of a digital weighing scale and a load cell sensor for Frame 6. Ten trials were made to compare the significant difference between using a digital weighing scale and a load cell sensor. A paired t-test was used to compare the two methods to gather statistical data analysis.

The null hypothesis is that there is no significant difference between the measurement gathered by a digital

weighing scale and a load cell sensor. The level of significance is 0.01 and the hypothesized mean difference is 0.

Table 4.12. Paired T-test for Weight Measurements of Frame 6.

SUMMARY $\alpha = 0.01$ *Hypothesized Mean Difference = 0*

Groups	*Count*	*Mean*	*Standard Deviation*	*Standard Error*	*t*	*df*
Digital Weighing Scale	10	0.725	0.030			
Load Cells	10	0.7275	0.033			
Difference		-0.0025	-0.003	-0.002	-1.369	9

t test

	p-value	*t-critical*
One tail	0.079	1.833
Two tail	0.159	2.262

Based on the summary shown in Table 4.12, by using a paired t-test, the two-tailed p-value is 0.159 which is greater than the 0.01 significance level. Thus, the null hypothesis is accepted. The load cells and digital weighing scale have no significant difference. Therefore, the load cell sensor can be used to gather weight measurements of the frames inside the beehive.

4.3.1.7 Weight Sensors Evaluation for paired load cells at Frame 7.

Table 4.13. Data for Digital Weighing Scale and Load cells for Frame 7.

Trial	Digital Weighing Scale (kg)	Load cells for Frame 7 (kg)
1	1.02	1.125
2	1.10	1.099
3	1.03	0.901
4	1.02	1.130
5	1.03	1.200
6	1.10	1.123
7	0.99	1.153
8	0.99	1.110
9	1.01	1.099
10	1.01	0.987

Table 4.13 shows the acquired measurements of a digital weighing scale and a load cell sensor for Frame 7. Ten trials were made to compare the significant difference between using a digital weighing scale and a load cell sensor. A paired t-test was used to compare the two methods to gather statistical data analysis.

The null hypothesis is that there is no significant difference between the measurement gathered by a digital

weighing scale and a load cell sensor. The level of significance is 0.01 and the hypothesized mean difference is 0.

Table 4.14. Paired T-test for Weight Measurements of Frame 7.

SUMMARY		$\alpha = 0.01$	*Hypothesized Mean Difference = 0*			
Groups	*Count*	*Mean*	*Standard Deviation*	*Standard Error*	*t*	*df*
Digital Weighing Scale	10	1.03	0.039			
Load Cell	10	1.093	0.086			
Difference		-0.063	0.094	0.030	2.09	9

t test

	p-value	*t-critical*
One tail	0.022	2.896
Two tail	0.044	3.355

Based on the summary shown in Table 4.14, by using a paired t-test, the two-tailed p-value is 0.044 which is greater than the 0.01 significance level. Thus, the null hypothesis is accepted. The load cells and digital weighing scale have no significant difference. Therefore, the load cell sensor can be used to gather weight measurements of the frames inside the beehive.

4.3.1.8 Weight Sensors Evaluation for paired load cells at Frame 8.

Table 4.15. Data for Digital Weighing Scale and Load cells for Frame 8

Trial	Digital Weighing Scale (kg)	Load cells for Frame 8 (kg)
1	0.95	0.965
2	0.96	0.986
3	0.99	1.09
4	1.02	1.15
5	0.97	0.965
6	0.96	0.998
7	0.98	0.973
8	0.98	0.993
9	1.01	1.03
10	0.97	0.991

Table 4.15 shows the acquired measurements of a digital weighing scale and a load cell sensor for Frame 8. Ten trials were made to compare the significant difference between using a digital weighing scale and a load cell sensor. A paired t-test was used to compare the two methods to gather statistical data analysis.

The null hypothesis is that there is no significant difference between the measurement gathered by a digital weighing scale and a load cell sensor. The level of significance is 0.01 and the hypothesized mean difference is 0.

Table 4.16. Paired T-test for Weight Measurements of Frame 8

SUMMARY			$\alpha = 0.01$	*Hypothesized Mean Difference = 0*		
Groups	*Count*	*Mean*	*Standard Deviation*	*Standard Error*	*t*	*df*
Digital Weighing Scale	10	1.03	0.039			
Load Cell	10	1.093	0.086			
Difference		-0.063	0.094	0.030	2.09	9

t test

	p-value	*t-critical*
One tail	0.022	2.896
Two tail	0.044	3.355

Based on the summary shown in Table 4.16, by using a paired t-test, the two-tailed p-value is 0.044 which is greater than the 0.01 significance level. Thus, the null hypothesis is accepted. The load cells and digital weighing scale have no significant difference. Therefore, the load cell sensor can be used to gather weight measurements of the frames inside the beehive.

4.3.1.9 Weight Sensors Evaluation for paired load cells at Frame 9.

Table 4.17. Data for Digital Weighing Scale and Load cells for Frame 9

Trial	Digital Weighing Scale (kg)	Load cells for Frame 9 (kg)
1	1.18	1.2
2	1.15	1.1
3	1.12	1.1
4	1.16	1.2
5	1.3	1.2
6	1.25	1.1
7	1.05	1.1
8	1.15	1.2
9	1.15	1.2
10	1.04	1.1

Table 4.17 shows the acquired measurements of a digital weighing scale and a load cell sensor for Frame 9. Ten trials were made to compare the significant difference between using a digital weighing scale and a load cell sensor. A paired t-test was used to compare the two methods to gather statistical data analysis.

The null hypothesis is that there is no significant difference between the measurement gathered by a digital weighing scale and a load cell sensor. The level of significance is 0.01 and the hypothesized mean difference is 0.

Table 4.18. Paired T-test for Weight Measurements of Frame 9

SUMMARY			*α= 0.01*	*Hypothesized Mean Difference = 0*		
Groups	*Count*	*Mean*	*Standard Deviation*	*Standard Error*	*t*	*df*
Digital Weighing Scale	10	1.15	0.079			
Load cells at (Frame 9)	10	1.155	0.053			
Difference		-0.005	0.026	0.0083	-0.22	9

t test		
	p-value	*t-critical*
One tail	0.42	2.82
Two tail	0.83	3.25

Based on the summary shown in Table 4.18, by using a paired t-test, the two-tailed p-value is 0.83 which is greater than the 0.01 significance level. Thus, the null hypothesis is accepted. The load cells and digital weighing scale have no significant difference. Therefore, the load cell sensor can be used to gather weight measurements of the frames inside the beehive.

4.3.1.10 Weight Sensors Evaluation for paired load cells at Frame 10.

Table 4.19. Data for Digital Weighing Scale and Load cells for Frame 10

Trial	Digital Weighing Scale (kg)	Load cells for Frame 10 (kg)
1	0.826	0.82
2	0.937	0.93
3	0.905	0.9
4	0.918	0.91
5	0.824	0.82
6	0.815	0.81
7	0.804	0.8
8	0.982	0.98
9	0.866	0.86
10	0.857	0.85

Table 4.19 shows the acquired measurements of a digital weighing scale and a load cell sensor for Frame 10. Ten trials were made to compare the significant difference between using a digital weighing scale and a load cell sensor. A paired t-test was used to compare the two methods to gather statistical data analysis.

The null hypothesis is that there is no significant difference between the measurement gathered by a digital weighing scale and a load cell sensor. The level of significance is 0.01 and the hypothesized mean difference is 0.

Table 4.20. Paired T-test for Weight Measurements of Frame 10

SUMMARY			*α= 0.01*	*Hypothesized Mean Difference = 0*		
Groups	*Count*	*Mean*	*Standard Deviation*	*Standard Error*	*t*	*df*
Digital Weighing Scale	10	0.868	0.0597			
Load cells at (Frame 10)	10	0.873	0.0598			
Difference		-0.005	-0.0001	-0.0001	-9.61	9

t test		
	p-value	*t-critical*
One tail	2.482	2.821
Two tail	4.965	3.25

Based on the summary shown in Table 4.20, by using a paired t-test, the two-tailed p-value is 4.965 which is greater than the 0.01 significance level. Thus, the null hypothesis is accepted. The load cells and digital weighing scale have no significant difference. Therefore, the load cell sensor can be used to gather weight measurements of the frames inside the beehive.

4.3.2 Temperature Evaluation

Table 4.21. Data for Digital Thermometer and DHT22 Temperature Sensor

Trial	Digital Thermometer (°C)	DHT22 Temperature Sensor (°C)
1	29.2	29.4
2	29.2	29.4
3	29.0	29.5
4	29.8	29.4
5	28.1	29.4
6	30.0	29.5
7	30.1	29.5
8	29.9	29.5
9	29.7	29.5
10	30.0	29.6

Table 4.21 shows the acquired measurements of a digital thermometer and DHT22 temperature sensor. Ten trials were made to compare the significant difference between using a digital thermometer and a temperature sensor. A paired t-test was used to compare the two methods to gather statistical data analysis.

The null hypothesis is that there is no significant difference between the measurement gathered by a digital

thermometer and DHT22 temperature sensor. The level of significance is 0.01 and the hypothesized mean difference is 0.

Table 4.22. Paired T-test for Temperature inside the Smart Beehive

SUMMARY			$\alpha = 0.01$	*Hypothesized Mean Difference = 0*		
Groups	*Count*	*Mean*	*Standard Deviation*	*Standard Error*	*t*	*df*
Digital Thermometer	10	29.5	0.067			
DHT22 Temperature Sensor	10	29.47	0.627			
Difference		0.03	-0.56	0.186	-0.505	9

t test		
	p-value	*t-critical*
One tail	0.44	1.83
Two tail	0.88	2.26

Based on the summary shown in Table 4.22, by using a paired t-test, the two-tailed p-value is 0.88 which is greater than the 0.01 significance level. Thus, the null hypothesis is accepted. DHT22 and digital thermometer have no significant difference. Therefore, the DHT22 can be used to gather temperature measurements inside the beehive.

4.3.3 Relative Humidity Evaluation

Table 4.23. Data for Digital Hygrometer and DHT22 Humidity Sensor

Trial	Digital Thermometer (°C)	DHT22 Temperature Sensor (°C)
1	71.2	71.4
2	71.2	71.4
3	71.3	71.4
4	71.5	71.4
5	71.5	71.3
6	71.2	71.3
7	71.2	71.3
8	71.9	71.4
9	71.8	71.4
10	71.9	71.4

Table 4.23 shows the acquired measurements of a digital hygrometer and DHT22 humidity sensor. Ten trials were made to compare the significant difference between using a digital hygrometer and a humidity sensor. A paired t-test was also used to compare the two methods to gather statistical data analysis.

The null hypothesis is that there is no significant difference between the measurement gathered by a digital hygrometer and the DHT22 humidity sensor. The level of significance is 0.01 and the hypothesized mean difference is 0.

Table 4.24. Paired T-test for Relative Humidity inside the Smart Beehive

SUMMARY			$\alpha = 0.01$	*Hypothesized Mean Difference = 0*		
Groups	*Count*	*Mean*	*Standard Deviation*	*Standard Error*	*t*	*df*
Digital Hygrometer	10	71.47	0.283			
DHT22 Humidity Sensor	10	71.37	0.045			
Difference		0.1	0.238	0.100	-3.16	9

t test	*p-value*	*t-critical*
One tail	0.146	1.833
Two tail	0.292	2.262

Based on the summary shown in Table 4.24, by using a paired t-test, the two-tailed p-value is 0.292 which is greater than the 0.01 significance level. Thus, the null hypothesis is accepted. DHT22 and digital hygrometer have no significant difference. Therefore, the DHT22 can be used to gather relative humidity measurements inside the beehive.

4.3.4 Varroa Mite Detection Evaluation

Convolutional Neural Network (CNN) was used to detect the varroa mites on the bees. One thousand data sets were used to train the system. The accuracy of detection was shown in percentage.

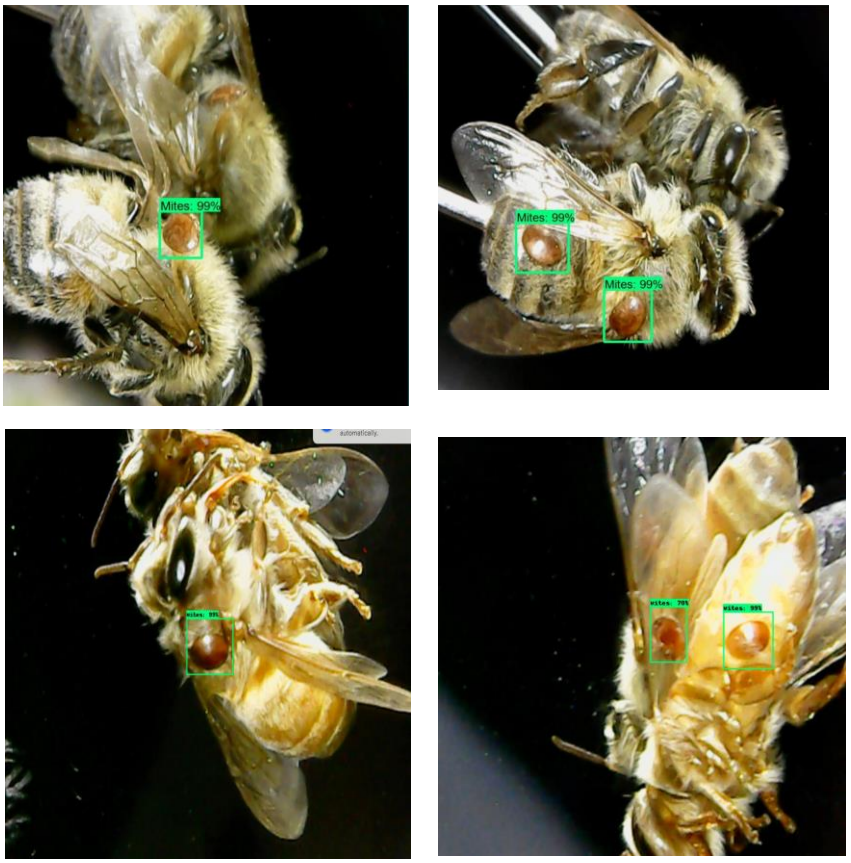


Figure 4.7. Captured Images with Accurate Detection

Figure 4.7. shows the captured images of the system. The mites were detected with 99% accuracy. Moreover, the reflected image of the mites was also detected.

CHAPTER 5

SUMMARY OF FINDINGS, CONCLUSION, AND RECOMMENDATION

This chapter presents a summary of findings, conclusions, and recommendations relative to the results of the conducted study.

5.1 Summary of Findings

1. The p-values obtained are greater than the 0.01 significance level. Thus, the null hypothesis is accepted. There is no significant difference between the system's sensor network system and measurement devices.
2. The system can detect varroa mites with 99% accuracy.
3. The system can provide different treatment options that are convenient to the beekeeper. It is known that continuous use of the same treatment may tend to immunize the mites. The automated treatment process consumes lesser time and labor as compared to manual treatment.

5.2 Conclusion

Based on the data gathered, outcomes of the tests conducted and the analysis of findings, the following statements are inferred:

1. The weight monitoring system and the temperature and humidity sensor produced results with high accuracy. Therefore, the system can be used to measure the beehive

parameters efficiently. The Arduino Mega was programmed to gather and process all these data every day, making the monitoring of these beehive parameters possible.

2. The early infestation of varroa mite is determined every time the image recognition system detects 10 or more captured images of bees with mites on their bodies.
3. The Smart Beehive mobile application displays all the parameters from the database. The application, which includes the beehive's status, can only be accessed by the beekeeper.
4. The automated treatment with the option of using mist, alagaw leaves, and API strips can be controlled using the mobile application. The beekeeper can treat the beehive if the system detects 10 or more varroa mites. Once a treatment is chosen, the number of mites displayed on the mobile application will then be reset to 0.
5. The system is effective and saves time and effort as compared to manual inspection and monitoring of the beehive.

5.3 Recommendation

For further improvement of the study, the proponents recommend the following:

1. Use separate Raspberry Pi for the USB cameras for faster processing of detection.

2. Use a high-quality mist maker to ensure that sufficient mist will reach the inside of the beehive. Additional treatment options that ensure the safety of bees and the quality of the honey should also be considered.
3. Add other pest detection in the system that is necessary for the health of the beehive. Birds are one of the main pests that kill most of the bees according to beekeepers.
4. Have a research about future technologies related to the extraction of honey that can be added to the system.
5. Improve the design of the system to make it more robust.
6. Conduct research on other behavioral patterns of the bees that may indicate an opportunity to innovate existing technologies to cope up with the demands of the apiculture industry in the country,

REFERENCES

A. Zacepins and J. Meitalovs, (2014, June 26) "Implementation of multi-node temperature measurement system for bee colonies online monitoring," *Proceedings of the 2014 15th International Carpathian Control Conference (ICCC*), Velke Karlovice, 2014, pp. 695-698. Retrieved from https://ieeexplore.ieee.org/document/6843694

W. Chen, C. Wang, J. Jiang, and E. Yang, (2016, March 24) "Development of a monitoring system for honeybee activities," *2015 9th International Conference on Sensing Technology (ICST)*, Auckland, 2015, pp. 745-750. Retrieved from https://ieeexplore.ieee.org/document/7438495

A. Zacepins, A. Kviesis, P. Ahrendt, U. Richter, S. Tekin and M. Durgun, (2016, June 30) "Beekeeping in the future — Smart apiary management," *2016 17th International Carpathian Control Conference (ICCC)*, Tatranska Lomnica, 2016, pp. 808-812. Retrieved from https://ieeexplore.ieee.org/document/7501207

G. C. Seritan, B. Enache, F. C. Argatau, F. C. Adochiei and S. Toader, (2018, July 05) "Low-cost platform for monitoring honey production and bees health," *2018 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR*), Cluj-Napoca, 2018, pp. 1-4. Retrieved from https://ieeexplore.ieee.org/document/8402704

D. W. Fitzgerald, F. E. Murphy, W. M. D. Wright, P. M. Whelan, and E. M. Popovici, (2015, July 23) "Design and development of a smart weighing scale for beehive monitoring," *2015 26th Irish Signals and Systems Conference (ISSC)*, Carlow, 2015, pp. 1-6. Retrieved from https://ieeexplore.ieee.org/document/7163763

I. F. Rodriguez, R. Megret, E. Acuna, J. L. Agosto-Rivera and T. Giray, (2018, May 07) "Recognition of Pollen-Bearing Bees from Video Using Convolutional Neural Network," *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Lake Tahoe, NV, 2018, pp. 314-322. Retrieved from https://ieeexplore.ieee.org/document/8354145

L. Chazette, M. Becker and H. Szczerbicka, (2017, February 13) "Basic algorithms for beehive monitoring and laser-based mite control," *2016 IEEE Symposium Series on Computational Intelligence (SSCI*), Athens, 2016, pp. 1-8. Retrieved from https://ieeexplore.ieee.org/document/7850001

G. Yue and L. Lu, (2018, November 12) "Face Recognition Based on Histogram Equalization and Convolution Neural Network," *2018 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC*), Hangzhou, 2018, pp. 336-339. Retrieved from https://ieeexplore.ieee.org/document/8530341

Z. Rian, V. Christanti and J. Hendryli, (2019, August 26) "Content-Based Image Retrieval using Convolutional Neural Networks," *2019 IEEE International Conference on Signals and Systems (ICSigSys*), Bandung, Indonesia, 2019, pp. 1-7. Retrieved from https://ieeexplore.ieee.org/document/8811089

A. R. de la Concepcion, R. Stefanelli, and D. Trinchero, (2014. June 05) "Adaptive wireless sensor networks for high-definition monitoring in sustainable agriculture," *2014 IEEE Topical Conference on Wireless Sensors and Sensor Networks (WiSNet)*, Newport Beach, CA, 2014, pp. 67-69. Retrieved from https://ieeexplore.ieee.org/document/6825511

S. E. Princy and K. G. J. Nigel, (2016, April 19) "Implementation of cloud server for real-time data storage using Raspberry Pi," *2015 Online International Conference on Green Engineering and Technologies (IC-GET*), Coimbatore, 2015, pp. 1-4. Retrieved from https://ieeexplore.ieee.org/document/7453790

B. V. S. Krishna, J. Oviya, S. Gowri, and M. Varshini, (2016, September 08) “Cloud robotics in the industry using Raspberry Pi," *2016 Second International Conference on Science Technology Engineering and Management (ICONSTEM)*, Chennai, 2016, pp. 543-547. Retrieved from https://ieeexplore.ieee.org/document/7560952

N. P. Kumar and R. K. Jatoth, (2015, July 09) "Development of cloud-based light intensity monitoring system using Raspberry Pi," *2015 International Conference on Industrial Instrumentation and Control (ICIC)*, Pune, 2015, pp. 1356-1361. Retrieved from https://ieeexplore.ieee.org/document/7150959

T. Tavade and P. Nasikkar, (2017, October 17) "Raspberry Pi: Data logging IOT device," *2017 International Conference on Power and Embedded Drive Control (ICPEDC)*, Chennai, 2017, pp. 275-279. Retrieved from https://ieeexplore.ieee.org/document/8081100

B. Mondal, A. Mahata, S. Chowdhuri, J. N. Bera and G. Sarkar, (2014, November 20) "Remote monitoring of energy consumption using Zigbee and GSM networking," *Proceedings of The 2014 International Conference on Control, Instrumentation, Energy and Communication (CIEC*), Calcutta, 2014, pp. 529-532. Retrieved from https://ieeexplore.ieee.org/document/6959145

D. Punetha and V. Mehta, (2015, January 19) "A Wireless Approach to Real-Time Remote Monitoring System examining Environmental Parameters Using Feasibility of a GSM Module," *2014 International Conference on Advances in Engineering & Technology Research (ICAETR - 2014)*, Unnao, 2014, pp. 1-6. Retrieved from https://ieeexplore.ieee.org/document/7012918

N. Vasanth, G. Akash, K. R. Srikanth, S. T. N. Pavan, and R. Sinha, (2018, June 21) "Solar-powered automatic pesticides sprayer," *2017 International*

Conference on Energy, Communication, Data Analytics, and Soft Computing (ICECDS), Chennai, 2017, pp. 3438-3441. Retrieved from https://ieeexplore.ieee.org/document/8390099

Aishwarya. B.V, Archana. G and C. Umayal, (2015, December 17) "Agriculture robotic vehicle-based pesticide sprayer with efficiency optimization," *2015 IEEE Technological Innovation in ICT for Agriculture and Rural Development (TIAR)*, Chennai, 2015, pp. 59-65. Retrieved from https://ieeexplore.ieee.org/document/7358532

R. Sosa, L. Vazquez, I. Santana, E. Rubio and C. Duran-Faundez, (2019, January 14) Automated Sprayer System for Variable Rate Application of Pesticides," *2018 IEEE International Conference on Automation/XXIII Congress of the Chilean Association of Automatic Control (ICA-ACCA*), Concepcion, 2018, pp. 1-6. Retrieved from https://ieeexplore.ieee.org/document/8609846

Amrita Sneha.A, Abirami.E, Ankita.A, R. Praveena and R. Srimeena, (2015, December 17) "Agricultural Robot for automatic ploughing and seeding," *2015 IEEE Technological Innovation in ICT for Agriculture and Rural Development (TIAR)*, Chennai, 2015, pp. 17-23. Retrieved from https://ieeexplore.ieee.org/document/7358525

Sivaranjani M, Swathi V, Vasanthi G, Revathi Y (2017, March) “Automated Paraquat Sprayer By Using Raspberry-Pi” *International Journal of Advanced Research in Basic Engineering Sciences and Technology (IJARBEST)*

M. Roth, J. Wilson, K.Tignor and A.Gross (2020, January 5) "Biology and Management of Varroa destructor (Mesostigmata: Varroidae) in Apis mellifera (Hymenoptera: Apidae) Colonies" *Journal of Integrated Pest Management*, Blacksburg, Volume 11, Issue 1, 2020, 1. Retrieved from https://academic.oup.com/jipm/article/11/1/1/5692075

E. Lopez-Tagle, E. Siquieros, and H. Ponce, (2017 September 20-22) “Design of an Automated Hive for Bee Proliferation and Crop Betterment” 2*017 14th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE),* Mexico, pp. 1-2. Retrieved from https://ieeexplore.ieee.org/document/8108878

V.G. Rybin, D.N. Butusov, T.I. Karimov, D.A. Belkin, and M.N. Kozak, “Embedded Data Acquisition System for Beehive Monitoring” *2017 IEEE II International Conference on Control in Technical Systems (CTS),* Russia, pp. 387-389. Retrieved from https://ieeexplore.ieee.org/document/8109576

APPENDIX A

BILL OF MATERIALS

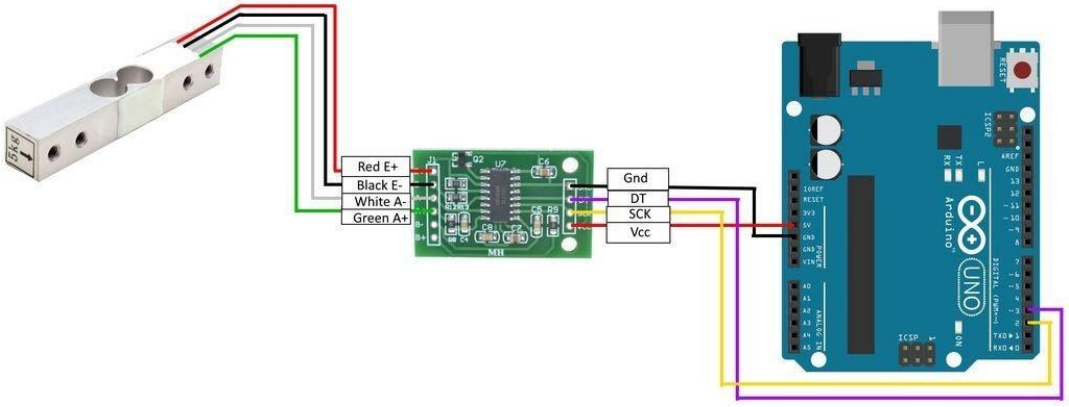
QUANTITY	COMPONENT	PRICE
1	Raspberry Pi 3B+	Php 2,300.00
1	Raspberry Pi 3B+ power connector	Php 399.00
1	Arduino Mega 2560	Php 481.00
1	DHT22 Temperature and Humidity Sensor	Php 250.00
20	HX711 Weight Sensor Module	Php 2,780.00
4	USB Microscope Camera	Php 2,796.00
1	GSM Module	Php 449.00
1	Starter Beehive	Php 5,000.00
1	10-Frame Beehive	Php 5,000.00
1	Colony with Queen	Php 4,500.00
1	Beehive stand	Php 1,000.00
1	16 GB SD card	Php 550.00
3	Servo motor	Php 327.00
1	Ultrasonic Mist maker	Php 295.00
1	Panasonic Battery 12V 7.2A	Php 890.00
2	ESP8266 WiFi Module	Php 350.00

1	DC to DC Buck Converter	Php 300.00
	Miscellaneous (Wires, connectors, etc.)	Php 2,000.00
Total:		**Php 29,667.00**

APPENDIX B

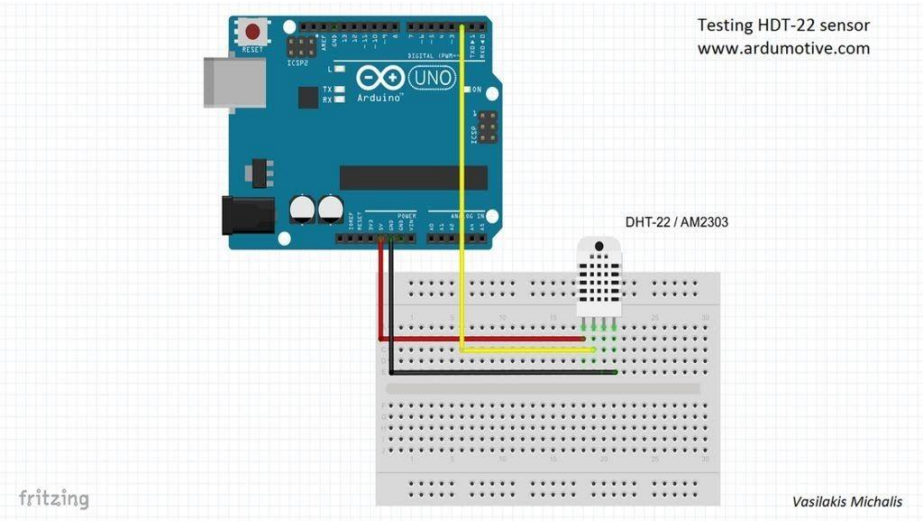
SCHEMATIC DIAGRAM AND PROGRAM CODES

1. Load cell and HX711



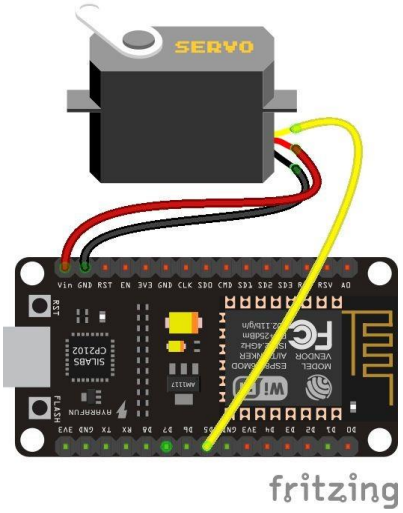
Source:https://www.instructables.com/id/Arduino-Scale-With-5kg-Load-Cell-and-HX711-Amplifi/

2. DHT22

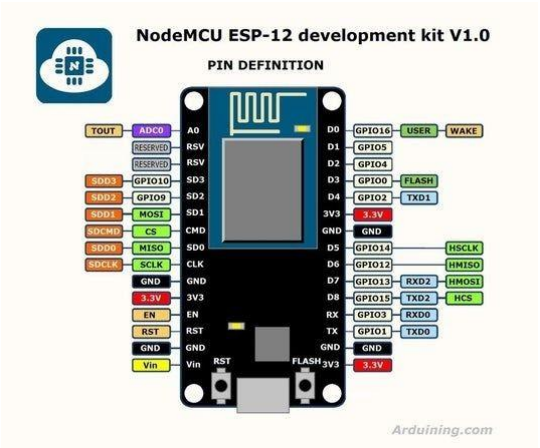


Source:https://www.instructables.com/id/How-to-use-DHT-22-sensor-Arduino-Tutorial/

3. Servo motor and Node MCU



Source:https://github.com/lvidarte/esp8266/wiki/MicroPython:-Servo-sg90



Source: https://www.instructables.com/id/Interfacing-Servo-Motor-With-NodeMCU/

PROGRAM CODES

1. **Sensors**

```
#include <DHT.h>;

#include "HX711.h"


#define DHTPIN 53

#define DHTTYPE DHT22

DHT dht(DHTPIN, DHTTYPE);


int chk;

float hum;

float temp;


const int LOADCELL_DOUT_PIN1 = 2;

const int LOADCELL_SCK_PIN1 = 3;


const int LOADCELL_DOUT_PIN2 = 4;

const int LOADCELL_SCK_PIN2 = 5;


const int LOADCELL_DOUT_PIN3 = 6;

const int LOADCELL_SCK_PIN3 = 7;


const int LOADCELL_DOUT_PIN4 = 8;
```

```
const int LOADCELL_SCK_PIN4 = 9;

const int LOADCELL_DOUT_PIN5 = 10;
const int LOADCELL_SCK_PIN5 = 11;

const int LOADCELL_DOUT_PIN6 = 12;
const int LOADCELL_SCK_PIN6 = 13;

const int LOADCELL_DOUT_PIN7 = 14;
const int LOADCELL_SCK_PIN7 = 15;

const int LOADCELL_DOUT_PIN8 = 16;
const int LOADCELL_SCK_PIN8 = 17;

const int LOADCELL_DOUT_PIN9 = 18;
const int LOADCELL_SCK_PIN9 = 19;

const int LOADCELL_DOUT_PIN10 = 42;
const int LOADCELL_SCK_PIN10 = 43;

const int LOADCELL_DOUT_PIN11 = 22;
const int LOADCELL_SCK_PIN11 = 23;

const int LOADCELL_DOUT_PIN12 = 24;
```

```
const int LOADCELL_SCK_PIN12 = 25;

const int LOADCELL_DOUT_PIN13 = 26;
const int LOADCELL_SCK_PIN13 = 27;

const int LOADCELL_DOUT_PIN14 = 28;
const int LOADCELL_SCK_PIN14 = 29;

const int LOADCELL_DOUT_PIN15 = 30;
const int LOADCELL_SCK_PIN15 = 31;

const int LOADCELL_DOUT_PIN16 = 32;
const int LOADCELL_SCK_PIN16 = 33;

const int LOADCELL_DOUT_PIN17 = 34;
const int LOADCELL_SCK_PIN17 = 35;

const int LOADCELL_DOUT_PIN18 = 36;
const int LOADCELL_SCK_PIN18 = 37;

const int LOADCELL_DOUT_PIN19 = 38;
const int LOADCELL_SCK_PIN19 = 39;

const int LOADCELL_DOUT_PIN20 = 40;
```

```
const int LOADCELL_SCK_PIN20 = 41;


HX711 scale1;

HX711 scale2;

HX711 scale3;

HX711 scale4;

HX711 scale5;

HX711 scale6;

HX711 scale7;

HX711 scale8;

HX711 scale9;

HX711 scale10;

HX711 scale11;

HX711 scale12;

HX711 scale13;

HX711 scale14;

HX711 scale15;

HX711 scale16;

HX711 scale17;

HX711 scale18;

HX711 scale19;

HX711 scale20;
```

```
void setup() {
  Serial.begin(9600);
  dht.begin();

  scale1.begin(LOADCELL_DOUT_PIN1, LOADCELL_SCK_PIN1);
  scale2.begin(LOADCELL_DOUT_PIN2, LOADCELL_SCK_PIN2);
  scale3.begin(LOADCELL_DOUT_PIN3, LOADCELL_SCK_PIN3);
  scale4.begin(LOADCELL_DOUT_PIN4, LOADCELL_SCK_PIN4);
  scale5.begin(LOADCELL_DOUT_PIN5, LOADCELL_SCK_PIN5);
  scale6.begin(LOADCELL_DOUT_PIN6, LOADCELL_SCK_PIN6);
  scale7.begin(LOADCELL_DOUT_PIN7, LOADCELL_SCK_PIN7);
  scale8.begin(LOADCELL_DOUT_PIN8, LOADCELL_SCK_PIN8);
  scale9.begin(LOADCELL_DOUT_PIN9, LOADCELL_SCK_PIN9);
  scale10.begin(LOADCELL_DOUT_PIN10, LOADCELL_SCK_PIN10);
  scale11.begin(LOADCELL_DOUT_PIN11, LOADCELL_SCK_PIN11);
  scale12.begin(LOADCELL_DOUT_PIN12, LOADCELL_SCK_PIN12);
  scale13.begin(LOADCELL_DOUT_PIN13, LOADCELL_SCK_PIN13);
  scale14.begin(LOADCELL_DOUT_PIN14, LOADCELL_SCK_PIN14);
  scale15.begin(LOADCELL_DOUT_PIN15, LOADCELL_SCK_PIN15);
  scale16.begin(LOADCELL_DOUT_PIN16, LOADCELL_SCK_PIN16);
  scale17.begin(LOADCELL_DOUT_PIN17, LOADCELL_SCK_PIN17);
  scale18.begin(LOADCELL_DOUT_PIN18, LOADCELL_SCK_PIN18);
  scale19.begin(LOADCELL_DOUT_PIN19, LOADCELL_SCK_PIN19);
  scale20.begin(LOADCELL_DOUT_PIN20, LOADCELL_SCK_PIN20);
```

```
scale1.set_scale(174.f);
scale1.tare();
scale2.set_scale(210.f);
scale2.tare();
scale3.set_scale(17.f);
scale3.tare();
scale4.set_scale(210.f);
scale4.tare();
scale5.set_scale(2280.f);
scale5.tare();
scale6.set_scale(2280.f);
scale6.tare();
scale7.set_scale(2280.f);
scale7.tare();
scale8.set_scale(2280.f);
scale8.tare();
scale9.set_scale(2280.f);
scale9.tare();
scale10.set_scale(2280.f);
scale10.tare();
scale11.set_scale(-230.f);
scale11.tare();
```

```
  scale12.set_scale(167.f);
  scale12.tare();
  scale13.set_scale(140.f);
  scale13.tare();
  scale14.set_scale(2280.f);
  scale14.tare();
  scale15.set_scale(2280.f);
  scale15.tare();
  scale16.set_scale(81.f);
  scale16.tare();
  scale17.set_scale(101.f);
  scale17.tare();
  scale18.set_scale(25.f);
  scale18.tare();
  scale19.set_scale(55.f);
  scale19.tare();
  scale20.set_scale(2280.f);
  scale20.tare();
}

void loop() {

  hum = dht.readHumidity();
  temp = dht.readTemperature();
```

```
//Serial.print("Humidity: ");
//Serial.print(hum);
//Serial.print(" %, Temp: ");
//Serial.print(temp);
//Serial.println(" Celsius");
int weight1 = scale1.get_units() + scale2.get_units();
//Serial.print("load cell one reading:\t");
//Serial.println(scale1.get_units(), 1);
//Serial.print("load cell two reading:\t");
//Serial.println(scale2.get_units(), 1);
int weight2 = scale3.get_units() + scale4.get_units();
//Serial.print("load cell three reading:\t");
//Serial.println(scale3.get_units(), 1);
//Serial.print("load cell four reading:\t");
//Serial.println(scale4.get_units(), 1);
int weight3 = scale5.get_units() + scale6.get_units();
//Serial.print("load cell five reading:\t");
//Serial.println(scale5.get_units(), 1);
//Serial.print("load cell six reading:\t");
//Serial.println(scale6.get_units(), 1);
int weight4 = scale7.get_units() + scale8.get_units();
//Serial.print("load cell seven reading:\t");
//Serial.println(scale7.get_units(), 1);
//Serial.print("load cell eight reading:\t");
```

```
//Serial.println(scale8.get_units(), 1);
int weight5 = scale9.get_units() + scale10.get_units();
//Serial.print("load cell nine reading:\t");
//Serial.println(scale9.get_units(), 1);
//Serial.print("load cell ten reading:\t");
//Serial.println(scale10.get_units(), 1);
int weight6 = scale11.get_units() + scale12.get_units();
//Serial.print("load cell eleven reading:\t");
//Serial.println(scale11.get_units(), 1);
//Serial.print("load cell twelve reading:\t");
//Serial.println(scale12.get_units(), 1);
int weight7 = scale13.get_units() + scale14.get_units();
//Serial.print("load cell thirteen reading:\t");
//Serial.println(scale13.get_units(), 1);
//Serial.print("load cell fourteen reading:\t");
//Serial.println(scale14.get_units(), 1);
int weight8 = scale15.get_units() + scale16.get_units();
//Serial.print("load cell fifteen reading:\t");
//Serial.println(scale15.get_units(), 1);
//Serial.print("load cell sixteen reading:\t");
//Serial.println(scale16.get_units(), 1);
int weight9 = scale17.get_units() + scale18.get_units();
//Serial.print("load cell seventeen reading:\t");
//Serial.println(scale17.get_units(), 1);
```

```
//Serial.print("load cell eighteen reading:\t");
//Serial.println(scale18.get_units(), 1);
int weight10 = scale19.get_units() + scale20.get_units();
//Serial.print("load cell nineteen reading:\t");
//Serial.println(scale19.get_units(), 1);
//Serial.print("load cell twenty reading:\t");
//Serial.println(scale20.get_units(), 1);

scale1.power_down();
scale2.power_down();
scale3.power_down();
scale4.power_down();
scale5.power_down();
scale6.power_down();
scale7.power_down();
scale8.power_down();
scale9.power_down();
scale10.power_down();
scale11.power_down();
scale12.power_down();
scale13.power_down();
scale14.power_down();
scale15.power_down();
scale16.power_down();
```

```
scale17.power_down();
scale18.power_down();
scale19.power_down();
scale20.power_down();
delay(5000);
scale1.power_up();
scale2.power_up();
scale3.power_up();
scale4.power_up();
scale5.power_up();
scale6.power_up();
scale7.power_up();
scale8.power_up();
scale9.power_up();
scale10.power_up();
scale11.power_up();
scale12.power_up();
scale13.power_up();
scale14.power_up();
scale15.power_up();
scale16.power_up();
scale17.power_up();
scale18.power_up();
scale19.power_up();
```

```
  scale20.power_up();

Serial.println("temp="+String(temp)+"&humid="+String(hum)+"&weight1="
+String(weight1)+"&weight2="+String(weight2)+"&weight3="+String(weigh
t3)+"&weight4="+String(weight4)+"&weight5="+String(weight5)+"&weight
6="+String(weight6)+"&weight7="+String(weight7)+"&weight8="+String(w
eight8)+"&weight9="+String(weight9)+"&weight10="+String(weight10));
}
```

2. Load Cell Calibration

```
#include "HX711.h"

#define DOUT 8
#define CLK 9


HX711 scale;

float calibration_factor = -7050; //-7050 worked for my 440lb max scale setup

void setup() {
  Serial.begin(9600);
  Serial.println("HX711 calibration sketch");
```

```
  Serial.println("Remove all weight from scale");
  Serial.println("After readings begin, place known weight on scale");
  Serial.println("Press + or a to increase calibration factor");
  Serial.println("Press - or z to decrease calibration factor");

  scale.begin(DOUT, CLK);
  scale.set_scale();
  scale.tare(); //Reset the scale to 0

  long zero_factor = scale.read_average(); //Get a baseline reading
   Serial.print("Zero factor: "); //This can be used to remove the need to tare the
scale. Useful in permanent scale projects.
  Serial.println(zero_factor);
}

void loop() {

  scale.set_scale(calibration_factor); //Adjust to this calibration factor
  Serial.print("Reading: ");
  Serial.print(scale.get_units(), 1);
   Serial.print("g"); //Change this to kg and re-adjust the calibration factor if
you follow SI units like a sane person
  Serial.print(" calibration_factor: ");
  Serial.print(calibration_factor);
```

```
  Serial.println();


  if (Serial.available())
  {
    char temp = Serial.read();
    if (temp == '+' || temp == 's')
      calibration_factor += 10;
    else if (temp == '-' || temp == 'x')
      calibration_factor -= 10;
    else if (temp == 'd')
    calibration_factor += 100;
    else if (temp == 'c')
    calibration_factor -= 100;
    else if (temp == 'f')
    calibration_factor += 1000;
    else if (temp == 'v')
    calibration_factor -= 1000;
    else if (temp == 'a')
    calibration_factor += 1;
    else if (temp == 'z')
      calibration_factor -= 1;
  }
}
```

3. Object Detection

a) Image

```
# Import packages

import os

import cv2

import numpy as np

import tensorflow as tf

import sys

from matplotlib import pyplot as plt

from PIL import Image

# This is needed since the notebook is stored in the object_detection folder.

sys.path.append("..")


# Import utilities

from utils import label_map_util

from utils import visualization_utils as vis_util

"%matplotlib inline"

# Name of the directory containing the object detection module we're using

MODEL_NAME = 'inference_graph'

IMAGE_NAME = 'dataset/73.png'


# Grab path to current working directory

CWD_PATH = os.getcwd()
```

```
# Path to frozen detection graph .pb file, which contains the model that is used
# for object detection.
PATH_TO_CKPT =
os.path.join(CWD_PATH,MODEL_NAME,'frozen_inference_graph.pb')

# Path to label map file
PATH_TO_LABELS = os.path.join(CWD_PATH,'training','labelmap.pbtxt')

# Path to image
PATH_TO_IMAGE = os.path.join(CWD_PATH,IMAGE_NAME)

# Number of classes the object detector can identify
NUM_CLASSES = 1

IMAGE_SIZE = (1280, 720)

# Load the label map.
# Label maps map indices to category names, so that when our convolution
# network predicts `5`, we know that this corresponds to `king`.
# Here we use internal utility functions, but anything that returns a
# dictionary mapping integers to appropriate string labels would be fine
label_map = label_map_util.load_labelmap(PATH_TO_LABELS)
```

```
categories = label_map_util.convert_label_map_to_categories(label_map,
max_num_classes=NUM_CLASSES, use_display_name=True)
category_index = label_map_util.create_category_index(categories)

# Load the Tensorflow model into memory.
detection_graph = tf.Graph()
with detection_graph.as_default():
    od_graph_def = tf.GraphDef()
    with tf.gfile.GFile(PATH_TO_CKPT, 'rb') as fid:
        serialized_graph = fid.read()
        od_graph_def.ParseFromString(serialized_graph)
        tf.import_graph_def(od_graph_def, name='')

    sess = tf.Session(graph=detection_graph)

# Define input and output tensors (i.e. data) for the object detection classifier

# Input tensor is the image
image_tensor = detection_graph.get_tensor_by_name('image_tensor:0')

# Output tensors are the detection boxes, scores, and classes
# Each box represents a part of the image where a particular object was
detected
detection_boxes = detection_graph.get_tensor_by_name('detection_boxes:0')
```

```
# Each score represents a level of confidence for each of the objects.
# The score is shown on the result image, together with the class label.
detection_scores = detection_graph.get_tensor_by_name('detection_scores:0')
detection_classes =
detection_graph.get_tensor_by_name('detection_classes:0')

# Number of objects detected
num_detections = detection_graph.get_tensor_by_name('num_detections:0')

# Load image using OpenCV and
# expand image dimensions to have shape: [1, None, None, 3]
# i.e. a single-column array, where each item in the column has the pixel RGB
value
image = cv2.imread(PATH_TO_IMAGE)
image = cv2.resize(image,(800,800))
image_expanded = np.expand_dims(image, axis=0)

(boxes, scores, classes, num) = sess.run(
    [detection_boxes, detection_scores, detection_classes, num_detections],
    feed_dict={image_tensor: image_expanded})

# Draw the results of the detection (aka 'visualize the results')
```

```
vis_util.visualize_boxes_and_labels_on_image_array(
    image,
    np.squeeze(boxes),
    np.squeeze(classes).astype(np.int32),
    np.squeeze(scores),
    category_index,
    use_normalized_coordinates=True,
    line_thickness=5,
    min_score_thresh=0.5)

while 1:
# All the results have been drawn on image. Now display the image.
    cv2.imshow('Object detector', image)

# Press any key to close the image
    if cv2.waitKey(0) == 32:
        break

    #Clean up cv2.destroyAllWindows()
```

b) Video

Description:

```
# This program uses a TensorFlow-trained classifier to perform object
detection.
# It loads the classifier and uses it to perform object detection on a video.
# It draws boxes, scores, and labels around the objects of interest in each
# frame of the video.

## Some of the code is copied from Google's example at
##
https://github.com/tensorflow/models/blob/master/research/object_detection/o
bject_detection_tutorial.ipynb

## and some is copied from Dat Tran's example at
##
https://github.com/datitran/object_detector_app/blob/master/object_detection_
app.py

## but we changed it to make it more understandable.

# Import packages
import os
import cv2
import numpy as np
import tensorflow as tf
import sys
```

```
# This is needed since the notebook is stored in the object_detection folder.
sys.path.append("..")

# Import utilities
from utils import label_map_util
from utils import visualization_utils as vis_util

# Name of the directory containing the object detection module we're using
MODEL_NAME = 'inference_graph'
VIDEO_NAME = 'test.mov'

# Grab path to current working directory
CWD_PATH = os.getcwd()

# Path to frozen detection graph .pb file, which contains the model that is used
# for object detection.
PATH_TO_CKPT =
os.path.join(CWD_PATH,MODEL_NAME,'frozen_inference_graph.pb')

# Path to label map file
PATH_TO_LABELS = os.path.join(CWD_PATH,'training','labelmap.pbtxt')

# Path to video
```

```
PATH_TO_VIDEO = os.path.join(CWD_PATH,VIDEO_NAME)

# Number of classes the object detector can identify
NUM_CLASSES = 6

# Load the label map.
# Label maps map indices to category names, so that when our convolution
# network predicts `5`, we know that this corresponds to `king`.
# Here we use internal utility functions, but anything that returns a
# dictionary mapping integers to appropriate string labels would be fine
label_map = label_map_util.load_labelmap(PATH_TO_LABELS)
categories = label_map_util.convert_label_map_to_categories(label_map,
max_num_classes=NUM_CLASSES, use_display_name=True)
category_index = label_map_util.create_category_index(categories)

# Load the Tensorflow model into memory.
detection_graph = tf.Graph()
with detection_graph.as_default():
    od_graph_def = tf.GraphDef()
    with tf.gfile.GFile(PATH_TO_CKPT, 'rb') as fid:
        serialized_graph = fid.read()
        od_graph_def.ParseFromString(serialized_graph)
        tf.import_graph_def(od_graph_def, name='')
```

```
    sess = tf.Session(graph=detection_graph)

# Define input and output tensors (i.e. data) for the object detection classifier

# Input tensor is the image
image_tensor = detection_graph.get_tensor_by_name('image_tensor:0')

# Output tensors are the detection boxes, scores, and classes
# Each box represents a part of the image where a particular object was
detected
detection_boxes = detection_graph.get_tensor_by_name('detection_boxes:0')

# Each score represents a level of confidence for each of the objects.
# The score is shown on the result image, together with the class label.
detection_scores = detection_graph.get_tensor_by_name('detection_scores:0')
detection_classes                                                    =
detection_graph.get_tensor_by_name('detection_classes:0')

# Number of objects detected
num_detections = detection_graph.get_tensor_by_name('num_detections:0')

# Open video file
video = cv2.VideoCapture(PATH_TO_VIDEO)
```

```
while(video.isOpened()):

    # Acquire frame and expand frame dimensions to have shape: [1, None,
None, 3]
    # i.e. a single-column array, where each item in the column has the pixel
RGB value
   ret, frame = video.read()
   frame_expanded = np.expand_dims(frame, axis=0)

   # Perform the actual detection by running the model with the image as input
   (boxes, scores, classes, num) = sess.run(
      [detection_boxes, detection_scores, detection_classes, num_detections],
      feed_dict={image_tensor: frame_expanded})

   # Draw the results of the detection (aka 'visualize the results')
   vis_util.visualize_boxes_and_labels_on_image_array(
      frame,
      np.squeeze(boxes),
      np.squeeze(classes).astype(np.int32),
      np.squeeze(scores),
      category_index,
      use_normalized_coordinates=True,
      line_thickness=8,
      min_score_thresh=0.60)
```

```
    # All the results have been drawn on the frame, so it's time to display it.
    cv2.imshow('Object detector', frame)

    # Press 'q' to quit
    if cv2.waitKey(1) == ord('q'):
        break

# Clean up
video.release()
cv2.destroyAllWindows()
```

c) Webcam

```
# Description:
# This program uses a TensorFlow-trained classifier to perform object detection.
# It loads the classifier and uses it to perform object detection on a webcam feed.
# It draws boxes, scores, and labels around the objects of interest in each frame
# from the webcam.

## Some of the code is copied from Google's example at
```

```
##
https://github.com/tensorflow/models/blob/master/research/object_detection/o
bject_detection_tutorial.ipynb

## and some is copied from Dat Tran's example at
##
https://github.com/datitran/object_detector_app/blob/master/object_detection_
app.py

## but I changed it to make it more understandable to me.



# Import packages
import os
import cv2
import numpy as np
import tensorflow as tf
import sys

# This is needed since the notebook is stored in the object_detection folder.
sys.path.append("..")

# Import utilities
from utils import label_map_util
```

```
from utils import visualization_utils as vis_util

# Name of the directory containing the object detection module we're using
MODEL_NAME = 'inference_graph'

# Grab path to current working directory
CWD_PATH = os.getcwd()

# Path to frozen detection graph .pb file, which contains the model that is used
# for object detection.
PATH_TO_CKPT =
os.path.join(CWD_PATH,MODEL_NAME,'frozen_inference_graph.pb')

# Path to label map file
PATH_TO_LABELS = os.path.join(CWD_PATH,'training','labelmap.pbtxt')

# Number of classes the object detector can identify
NUM_CLASSES = 1

## Load the label map.
# Label maps map indices to category names, so that when our convolution
# network predicts `5`, we know that this corresponds to `king`.
# Here we use internal utility functions, but anything that returns a
# dictionary mapping integers to appropriate string labels would be fine
```

```
label_map = label_map_util.load_labelmap(PATH_TO_LABELS)
categories = label_map_util.convert_label_map_to_categories(label_map,
max_num_classes=NUM_CLASSES, use_display_name=True)
category_index = label_map_util.create_category_index(categories)

# Load the Tensorflow model into memory.
detection_graph = tf.Graph()
with detection_graph.as_default():
    od_graph_def = tf.GraphDef()
    with tf.gfile.GFile(PATH_TO_CKPT, 'rb') as fid:
        serialized_graph = fid.read()
        od_graph_def.ParseFromString(serialized_graph)
        tf.import_graph_def(od_graph_def, name='')

    sess = tf.Session(graph=detection_graph)


# Define input and output tensors (i.e. data) for the object detection classifier

# Input tensor is the image
image_tensor = detection_graph.get_tensor_by_name('image_tensor:0')

# Output tensors are the detection boxes, scores, and classes
```

```
# Each box represents a part of the image where a particular object was
detected
detection_boxes = detection_graph.get_tensor_by_name('detection_boxes:0')

# Each score represents a level of confidence for each of the objects.
# The score is shown on the result image, together with the class label.
detection_scores = detection_graph.get_tensor_by_name('detection_scores:0')
detection_classes =
detection_graph.get_tensor_by_name('detection_classes:0')

# Number of objects detected
num_detections = detection_graph.get_tensor_by_name('num_detections:0')

# Initialize webcam feed
video = cv2.VideoCapture(1)
ret = video.set(3,1280)
ret = video.set(4,720)

while(True):

    # Acquire frame and expand frame dimensions to have shape: [1, None,
None, 3]
    # i.e. a single-column array, where each item in the column has the pixel
RGB value
```

```
ret, frame = video.read()
frame_expanded = np.expand_dims(frame, axis=0)

# Perform the actual detection by running the model with the image as input
(boxes, scores, classes, num) = sess.run(
    [detection_boxes, detection_scores, detection_classes, num_detections],
    feed_dict={image_tensor: frame_expanded})

# Draw the results of the detection (aka 'visualize the results')
vis_util.visualize_boxes_and_labels_on_image_array(
    frame,
    np.squeeze(boxes),
    np.squeeze(classes).astype(np.int32),
    np.squeeze(scores),
    category_index,
    use_normalized_coordinates=True,
    line_thickness=8,
    min_score_thresh=0.5)

# All the results have been drawn on the frame, so it's time to display it.
cv2.imshow('Object detector', frame)

# Press 'q' to quit
if cv2.waitKey(1) == 32:
```

```
    break

# Clean up
video.release()
cv2.destroyAllWindows()
```

4. Notification System

a) Send notification

```
#include <SoftwareSerial.h>
#include <Servo.h>

//SIM800 TX is connected to Arduino D8
#define SIM800_TX_PIN 3

//SIM800 RX is connected to Arduino D7
#define SIM800_RX_PIN 2

//Create software serial object to communicate with SIM800
SoftwareSerial serialSIM800(SIM800_TX_PIN, SIM800_RX_PIN);

String x;
```

```
int servoPin1 = 3;

int servoPin2 = 3;

int servoPin3 = 3;

int servoPin4 = 3;

int servoPin5 = 3;

int servoPin6 = 3;

int servoPin7 = 3;

int servoPin8 = 3;

int servoPin9 = 3;

int servoPin10 = 3;

int servoPin11 = 3;


Servo Servo1;

Servo Servo2;

Servo Servo3;

Servo Servo4;

Servo Servo5;

Servo Servo6;

Servo Servo7;

Servo Servo8;

Servo Servo9;

Servo Servo10;

Servo Servo11;
```

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);

  //Being serial communication with Arduino and SIM800
  serialSIM800.begin(9600);
  delay(1000);

  //Set SMS format to ASCII
  serialSIM800.write("AT+CMGF=1\r\n");
  delay(1000);
  Servo1.write(0);
  Servo2.write(0);
  Servo3.write(0);
  Servo4.write(0);
  Servo5.write(0);
  Servo6.write(0);
  Servo7.write(0);
  Servo8.write(0);
  Servo9.write(0);
  Servo10.write(0);
  Servo11.write(0);
}
```

```
void loop() {
  // put your main code here, to run repeatedly:
  while (Serial.available()) {
    x = Serial.readString();
    if (x == "1") {
      //Set SMS format to ASCII
      serialSIM800.write("AT+CMGF=1\r\n");
      delay(1000);

      //Send new SMS command and message number
      serialSIM800.write("AT+CMGS=\"09566584845\"\r\n");
      delay(1000);

      //Send SMS content
      serialSIM800.write("Mites have been detected!!!");
      delay(1000);

      //Send Ctrl+Z / ESC to denote SMS message is complete
      serialSIM800.write((char)26);
      delay(1000);

      digitalWrite(buzzer, HIGH);
      delay(10);
    }
```

```
else if (x == "a") {
  Servo1.write(90);
  delay(1000);
  Servo1.write(0);
}
else if (x == "b") {
  Servo2.write(90);
  delay(1000);
  Servo2.write(0);
}
else if (x == "c") {
  Servo3.write(90);
  delay(1000);
  Servo3.write(0);
}
else if (x == "d") {
  Servo4.write(90);
  delay(1000);
  Servo4.write(0);
}
else if (x == "e") {
  Servo5.write(90);
  delay(1000);
  Servo5.write(0);
```

```
}
else if (x == "f") {
  Servo6.write(90);
  delay(1000);
  Servo6.write(0);
}
else if (x == "g") {
  Servo7.write(90);
  delay(1000);
  Servo7.write(0);
}
else if (x == "h") {
  Servo8.write(90);
  delay(1000);
  Servo8.write(0);
}
else if (x == "i") {
  Servo9.write(90);
  delay(1000);
  Servo9.write(0);
}
else if (x == "j") {
  Servo10.write(90);
  delay(1000);
```

```
    Servo10.write(0);
   }
   else if (x == "k") {
    Servo11.write(90);
    delay(1000);
    Servo11.write(0);
   }
  }
}
```

b) Receive notification

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>

const char* ssid = "BEEKEEP";
const char* password = "bestthesis2019";

void setup () {

 Serial.begin(9600);
 WiFi.begin(ssid, password);

 while (WiFi.status() != WL_CONNECTED) {
```

```
    delay(1000);
    Serial.print("Connecting..");

  }
}

void loop() {

  if (WiFi.status() == WL_CONNECTED) { //Check WiFi connection status

    HTTPClient http; //Declare an object of class HTTPClient
    http.begin("http://smartbeehive.pythonanywhere.com/first_app/GetSMS");
    int httpCode = http.GET();                                  //Send
the request

    if (httpCode > 0) { //Check the returning code

      String payload = http.getString(); //Get the request response payload
      Serial.println(payload);                    //Print the response payload
      if (payload == "0.0") {
        Serial.println("not send");
      }
      else {
```

```
        SendMessage();

http.begin("http://smartbeehive.pythonanywhere.com/first_app/SendSMS?cou
nt=0");
                                        int     httpCode    =    http.GET();
//Send the request

        if (httpCode > 0) { //Check the returning code

          String payload = http.getString(); //Get the request response payload
          Serial.println(payload);
        }

      }

      http.end(); //Close connection

    }

    delay(30000);    //Send a request every 30 seconds

  }
}
```

```
 void SendMessage() {
   Serial.print("AT"); //Start Configuring GSM Module
   delay(1000);         //One second delay
   Serial.println();
   Serial.println("AT+CMGF=1"); // Set GSM in text mode
   delay(1000);                 // One second delay
   Serial.println();
   Serial.print("AT+CMGS=");     // Enter the receiver number
   Serial.print("\"+639171836417\"");
   Serial.println();
   delay(1000);
   Serial.print("Beehive Mites Detected > 10"); // SMS body - Sms Text
   delay(1000);
   Serial.println();
   Serial.write(26);
 }
```

5. Automated Treatment System

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>

#include <Servo.h>
```

```
int servoPin1 = D7;

int servoPin2 = D6;

int servoPin3 = D5;


Servo Servo1;

Servo Servo2;

Servo Servo3;


const char* ssid = "BEEKEEP";

const char* password = "bestthesis2019";


void setup () {


  Serial.begin(9600);

  Servo1.attach(servoPin1);

  Servo2.attach(servoPin2);

  Servo3.attach(servoPin3);

  WiFi.begin(ssid, password);


  while (WiFi.status() != WL_CONNECTED) {


    delay(1000);

    Serial.print("Connecting..");
```

```
  }
}

void loop() {

  if (WiFi.status() == WL_CONNECTED) { //Check WiFi connection status

    HTTPClient http; //Declare an object of class HTTPClient

http.begin("http://smartbeehive.pythonanywhere.com/first_app/GetTreatment1
");
     int httpCode  = http.GET();                                          //Send
the request

    if (httpCode > 0) {

      String payload = http.getString(); //Get the request response payload
      //Serial.println(payload);                    //Print the response payload
      if (payload == "1.0") {
        Servo1.write(0);
        delay(1000);
        Servo1.write(90);
        delay(1000);
```

```
http.begin("http://smartbeehive.pythonanywhere.com/first_app/SendTreatment
1?count=0");
      int httpCode = http.GET();
    }
  }

http.begin("http://smartbeehive.pythonanywhere.com/first_app/GetTreatment2
");                                                    //Send the request
  int httpCode2 = http.GET();
  if (httpCode2 > 0) {

    String payload = http.getString(); //Get the request response payload
    //Serial.println(payload);                  //Print the response payload
    if (payload == "1.0") {
      Servo2.write(0);
      delay(1000);
      Servo2.write(90);
      delay(1000);

http.begin("http://smartbeehive.pythonanywhere.com/first_app/SendTreatment
2?count=0");
      int httpCode2 = http.GET();
    }
```

```
    }

http.begin("http://smartbeehive.pythonanywhere.com/first_app/GetTreatment3
");                                                  //Send the request
    int httpCode3 = http.GET();
    if (httpCode3 > 0) {

      String payload = http.getString(); //Get the request response payload
      //Serial.println(payload);                    //Print the response payload
      if (payload == "1.0") {
        Servo3.write(0);
        delay(1000);
        Servo3.write(90);
        delay(1000);

http.begin("http://smartbeehive.pythonanywhere.com/first_app/SendTreatment
3?count=0");
        int httpCode3 = http.GET();
      }
    }

    if(Serial.available()){
      String x = Serial.readStringUntil('\n');
      x.trim();
```

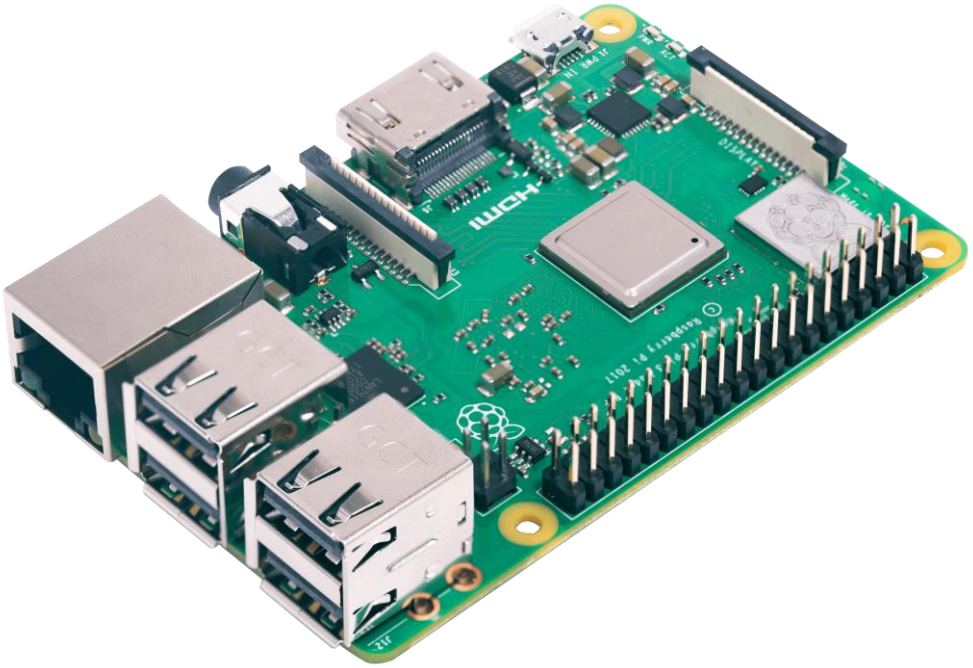
```
http.begin("http://smartbeehive.pythonanywhere.com/first_app/save_count_ar
duino?"+ String(x));
      int httpCode4 = http.GET();
      Serial.println(x);
      if (httpCode4 > 0) {

      String payload = http.getString(); //Get the request response payload
      Serial.println(payload);
    }
    }
    http.end();
    delay(1000);    //Send a request every 30 seconds
  }
}
```

APPENDIX C

DATASHEETS

Raspberry Pi Compute Module 3+ Raspberry Pi

Compute Module 3+ Lite



Release 1, January 2019

Table 1: Release History

Release	Date	Description
1	28/01/2019	First release

The latest release of this document can be found at
`https://www.raspberrypi.org`

1 Introduction

The Raspberry Pi Compute Module 3+ (CM3+) is a range of DDR2-SODIMM-mechanically-compatible System on Modules (SoMs) containing processor, memory, eMMC Flash (on non-Lite variants) and supporting power circuitry. These modules allow a designer to leverage the Raspberry Pi hardware and software stack in their own custom systems and form factors. In addition these modules have extra IO interfaces over and above what is available on the Raspberry Pi model A/B boards, opening up more options for the designer.

The CM3+ contains a BCM2837B0 processor (as used on the Raspberry Pi 3B+), 1Gbyte LPDDR2 RAM and eMMC Flash. The CM3+ is currently available in 4 variants, CM3+/8GB, CM3+/16GB, CM3+/32GB and CM3+ Lite, which have 8, 16 and 32 Gigabytes of eMMC Flash, or no eMMC Flash, respectively.

The CM3+ Lite product is the same as CM3+ except the eMMC Flash is not fitted, and the SD/eMMC interface pins are available for the user to connect their own SD/eMMC device.

Note that the CM3+ is electrically identical and, with the exception of higher CPU z-height, physically identical to the legacy CM3 products.

CM3+ modules require a software/firmware image dated November 2018 or newer to function correctly.

2 Features

2.1 Hardware

- Low cost
- Low power
- High availability
- High reliability
  - Tested over millions of Raspberry Pis Produced to date
  - Module IO pins have 15 micro-inch hard gold plating over 2.5 micron Nickel

2.2 Peripherals

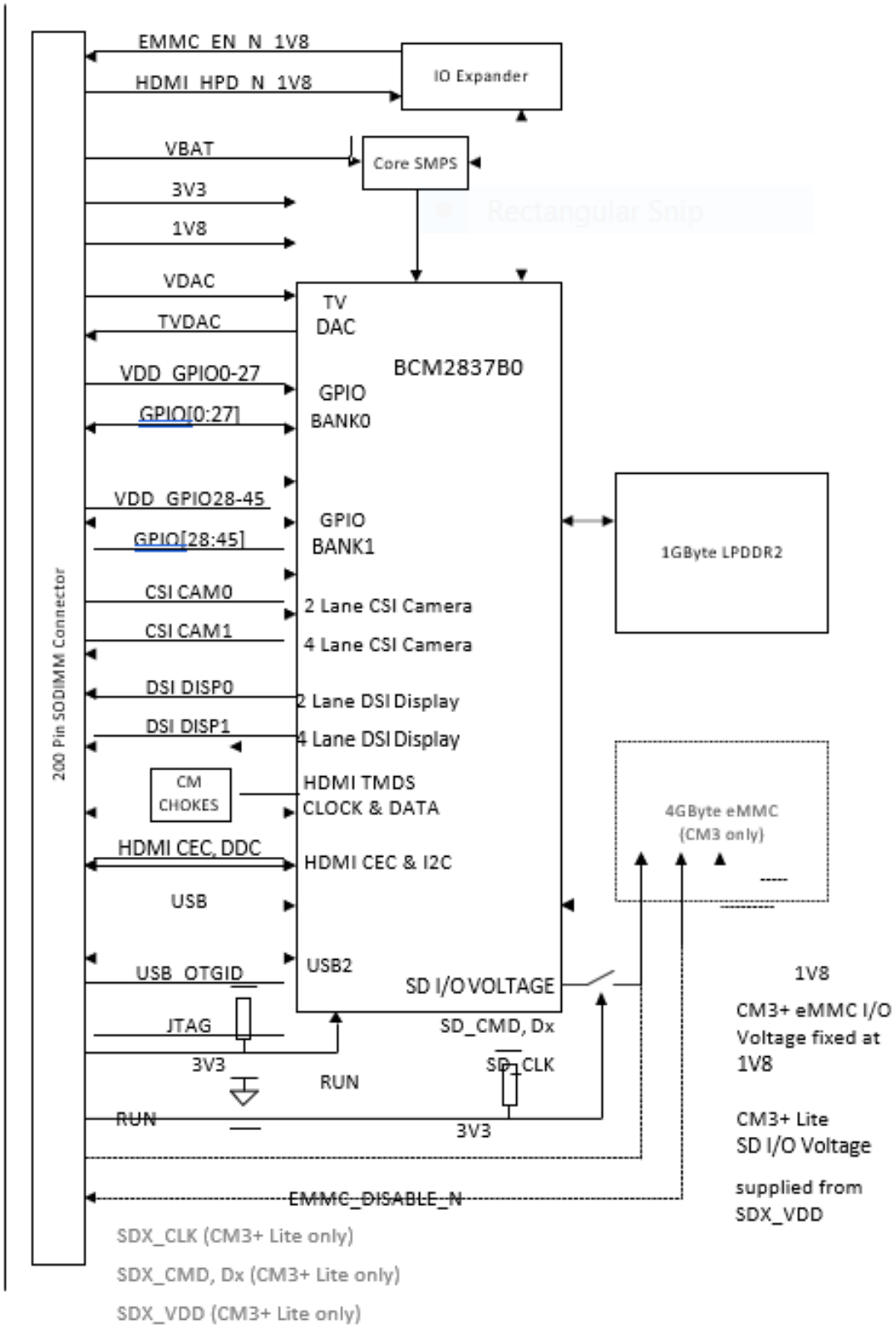
- 48x GPIO
- 2x I2C
- 2x SPI
- 2x UART
- 2x SD/SDIO
- 1x HDMI 1.3a
- 1x USB2 HOST/OTG
- 1x DPI (Parallel RGB Display)
- 1x NAND interface (SMI)
- 1x 4-lane CSI Camera Interface (up to 1Gbps per lane)
- 1x 2-lane CSI Camera Interface (up to 1Gbps per lane)
- 1x 4-lane DSI Display Interface (up to 1Gbps per lane)
- 1x 2-lane DSI Display Interface (up to 1Gbps per lane)

2.3 Software

- ARMv8 Instruction Set
- Mature and stable Linux software stack
  - Latest Linux Kernel support
  - Many drivers upstreamed
  - Stable and well supported userland
  - Full availability of GPU functions using standard APIs

3 Block Diagram

Figure 1: CM3+ Block Diagram



4 Mechanical Specification

The CM3+ modules conform to JEDEC MO-224 mechanical specification for 200 pin DDR2 (1.8V) SODIMM modules and therefore should work with the many DDR2 SODIMM sockets available on the market. **(Please note that the pinout of the Compute Module is not the same as a DDR2 SODIMM module; they are not electrically compatible.)**

The SODIMM form factor was chosen as a way to provide the 200 pin connections using a standard, readily available and low cost connector compatible with low cost PCB manufacture.

The maximum component height on the underside of the

Compute Module is 1.2mm. The maximum component height

on the top side of the Compute Module is 2.5mm. The Compute

Module PCB thickness is 1.0mm +/- 0.1mm.

Note that the location and arrangement of components on the Compute Module may change slightly over time due to revisions for cost and manufacturing considerations; however, maximum component heights and PCB thickness will be kept as specified.

Figure 2 gives the CM3+ mechanical dimensions.

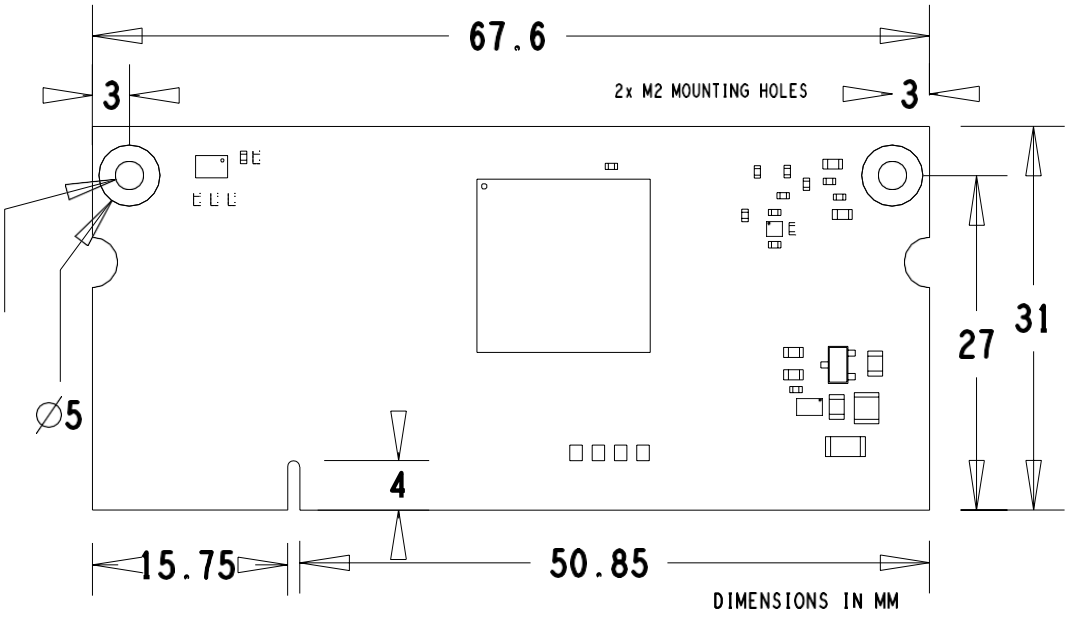


Figure 2: CM3+ Mechanical Dimensio

5 Pin Assignments

CM3+	CM3+ Lite	PIN		PIN	CM3+	CM3+ Lite
GND		1		2	EMMC_DISABLE_N	
GPIO0		3		4	NC	SDX_VDD
GPIO1		5		6	NC	SDX_VDD
GND		7		8	GND	
GPIO2		9		10	NC	SDX_CLK
GPIO3		11		12	NC	SDX_CMD
GND		13		14	GND	
GPIO4		15		16	NC	SDX_D0
GPIO5		17		18	NC	SDX_D1
GND		19		20	GND	
GPIO6		21		22	NC	SDX_D2
GPIO7		23		24	NC	SDX_D3
GND		25		26	GND	
GPIO8		27		28	GPIO28	
GPIO9		29		30	GPIO29	
GND		31		32	GND	
GPIO10		33		34	GPIO30	
GPIO11		35		36	GPIO31	
GND		37		38	GND	
GPIO0-27_VDD		39		40	GPIO0-27_VDD	
		KEY				
GPIO28-45_VDD		41		42	GPIO28-45_VDD	
GND		43		44	GND	
GPIO12		45		46	GPIO32	
GPIO13		47		48	GPIO33	
GND		49		50	GND	
GPIO14		51		52	GPIO34	
GPIO15		53		54	GPIO35	
GND		55		56	GND	
GPIO16		57		58	GPIO36	
GPIO17		59		60	GPIO37	
GND		61		62	GND	
GPIO18		63		64	GPIO38	
GPIO19		65		66	GPIO39	
GND		67		68	GND	
GPIO20		69		70	GPIO40	
GPIO21		71		72	GPIO41	
GND		73		74	GND	
GPIO22		75		76	GPIO42	
GPIO23		77		78	GPIO43	
GND		79		80	GND	
GPIO24		81		82	GPIO44	
GPIO25		83		84	GPIO45	
GND		85		86	GND	
GPIO26		87		88	HDMI_HPD_N_1V8	
GPIO27		89		90	EMMC_EN_N_1V8	
GND		91		92	GND	
DSI0_DN1		93		94	DSI1_DP0	
DSI0_DP1		95		96	DSI1_DN0	
GND		97		98	GND	
DSI0_DN0		99		100	DSI1_CP	
DSI0_DP0		101		102	DSI1_CN	
GND		103		104	GND	
DSI0_CN		105		106	DSI1_DP3	
DSI0_CP		107		108	DSI1_DN3	
GND		109		110	GND	
HDMI_CLK_N		111		112	DSI1_DP2	
HDMI_CLK_P		113		114	DSI1_DN2	

GND	115	116	GND
HDMI_D0_N	117	118	DSI1_DP1
HDMI_D0_P	119	120	DSI1_DN1
GND	121	122	GND
HDMI_D1_N	123	124	NC
HDMI_D1_P	125	126	NC
GND	127	128	NC
HDMI_D2_N	129	130	NC
HDMI_D2_P	131	132	NC
GND	133	134	GND
CAM1_DP3	135	136	CAM0_DP0
CAM1_DN3	137	138	CAM0_DN0
GND	139	140	GND
CAM1_DP2	141	142	CAM0_CP
CAM1_DN2	143	144	CAM0_CN
GND	145	146	GND
CAM1_CP	147	148	CAM0_DP1
CAM1_CN	149	150	CAM0_DN1
GND	151	152	GND
CAM1_DP1	153	154	NC
CAM1_DN1	155	156	NC
GND	157	158	NC
CAM1_DP0	159	160	NC
CAM1_DN0	161	162	NC
GND	163	164	GND
USB_DP	165	166	TVDAC
USB_DM	167	168	USB_OTGID
GND	169	170	GND
HDMI_CEC	171	172	VC_TRST_N
HDMI_SDA	173	174	VC_TDI
HDMI_SCL	175	176	VC_TMS
RUN	177	178	VC_TDO
DD_CORE (DO NOT CONNECT	179	180	VC_TCK
GND	181	182	GND
1V8	183	184	1V8
1V8	185	186	1V8
GND	187	188	GND
VDAC	189	190	VDAC
3V3	191	192	3V3
3V3	193	194	3V3
GND	195	196	GND
VBAT	197	198	VBAT
VBAT	199	200	VBAT

Table 2: Compute Module 3+ SODIMM Connector Pinout

Table 2 gives the Compute Module 3+ pinout and Table 3 gives the pin function

Pin Name	DIR	Voltage Ref	PDN[a] State	If Unused	Description/Notes
RUN and Boot Control (see text for usage guide)					
RUN	I	3V3[b]	Pull High	Leave open	Has internal 10k pull up EMMC
DISABLE N	I	3V3[b]	Pull High	Leave open	Has internal 10k pull up EMMC EN N 1V8 O
1V8			Pull High	Leave open	Has internal 2k2 pull up
GPIO					
GPIO[27:0]	I/O	GPIO0-27_VDD	Pull or Hi-Z[c]	Leave open	GPIO Bank 0
GPIO[45:28]	I/O	GPIO28-45_VDD	Pull or Hi-Z[c]	Leave open	GPIO Bank 1
Primary SD Interface[d,e]					
SDX_CLK	O	SDX_VDD	Pull High	Leave open	Primary SD interface CLK SDX
CMD	I/O	SDX_VDD	Pull High	Leave open	Primary SD interface CMD SDX
Dx	I/O	SDX_VDD	Pull High	Leave open	Primary SD interface DATA
USB Interface					
USB_Dx	I/O	-	Z	Leave open	Serial interface
USB_OTGID	I	3V3		Tie to GND	OTG pin detect
HDMI Interface					
HDMI_SCL	I/O	3V3[b]	Z[f]	Leave open	DDC Clock (5.5V tolerant)
HDMI_SDA	I/O	3V3[b]	Z[f]	Leave open	DDC Data (5.5V tolerant)
HDMI_CEC	I/O	3V3	Z	Leave open	CEC (has internal 27k pull up)
HDMI_CLKx	O	-	Z	Leave open	HDMI serial clock
HDMI_Dx	O	-	Z	Leave open	HDMI serial data
HDMI_HPD_N_1V8	I	1V8	Pull High	Leave open	HDMI hotplug detect
CAM0 (CSI0) 2-lane Interface					
CAM0_Cx	I	-	Z	Leave open	Serial clock
CAM0_Dx	I	-	Z	Leave open	Serial data
CAM1 (CSI1) 4-lane Interface					
CAM1_Cx	I	-	Z	Leave open	Serial clock
CAM1_Dx	I	-	Z	Leave open	Serial data
DSI0 (Display 0) 2-lane Interface					
DSI0_Cx	O	-	Z	Leave open	Serial clock
DSI0_Dx	O	-	Z	Leave open	Serial data
DSI1 (Display 1) 4-lane Interface					
DSI1_Cx	O	-	Z	Leave open	Serial clock
DSI1_Dx	O	-	Z	Leave open	Serial data
TV Out					
TVDAC	O	-	Z	Leave open	Composite video DAC output
JTAG Interface					
TMS	I	3V3	Z	Leave open	Has internal 50k pull up
TRST_N	I	3V3	Z	Leave open	Has internal 50k pull up
TCK	I	3V3	Z	Leave open	Has internal 50k pull up
TDI	I	3V3	Z	Leave open	Has internal 50k pull up
TDO	O	3V3	O	Leave open	Has internal 50k pull up

[a] The PDN column indicates power-down state (when RUN pin LOW)
[b] Must be driven by an open-collector driver
[c] GPIO have software enabled pulls which keep state over power-down
[d] Only available on Lite variants
[e] The CM will always try to boot from this interface first
[f] Requires external pull-up resistor to 5V as per HDMI spec

Table 3: Pin Functions

6 Electrical Specification

Caution! Stresses above those listed in Table 4 may cause permanent damage to the device. This is a stress rating only; functional operation of the device under these or any other conditions above those listed in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability

Symbol	Parameter	Minimum	Maximum	Unit
VBAT	Core SMPS Supply	-0.5	6.0	V
3V3	3V3 Supply Voltage	-0.5	4.10	V
1V8	1V8 Supply Voltage	-0.5	2.10	V
VDAC	TV DAC Supply	-0.5	4.10	V
GPIO0-27_VDD	GPIO0-27 I/O Supply Voltage	-0.5	4.10	V
GPIO28-45_VDD	GPIO28-45 I/O Supply Voltage	-0.5	4.10	V
SDX_VDD	Primary SD/eMMC Supply Voltage	-0.5	4.10	V

Table 4: Absolute Maximum

Ratings DC Characteristics are defined in Table 5

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{IL}	Input low voltage[a]	VDD_IO= 1.8V	-	-	0.6	V
		VDD_IO= 2.7V	-	-	0.8	V
		VDD_IO= 3.3V	-	-	0.9	V
V_{IH}	Input high voltage[a]	VDD_IO= 1.8V	1.0	-	-	V
		VDD_IO= 2.7V	1.3	-	-	V
		VDD_IO= 3.3V	1.6	-	-	V
I_{IL}	Input leakage current	TA= +85°C	-	-	5	µA
C_{IN}	Input capacitance	-	-	5	-	pF
V_{OL}	Output low voltage[b]	VDD_IO = 1.8V, IOL = -2mA	-	-	0.2	V
		VDD_IO = 2.7V, IOL= -2mA	-	-	0.15	V
		VDD_IO = 3.3V, IOL= -2mA	-	-	0.14	V
V_{OH}	Output high voltage[b]	VDD_IO = 1.8V, IOH= 2mA	1.6	-	-	V
		VDD_IO = 2.7V, IOH= 2mA	2.5	-	-	V
		VDD_IO = 3.3V, IOH= 2mA	3.0	-	-	V
I_{OL}	Output low current[c]	VDD_IO = 1.8V, VO= 0.4V	12	-	-	mA
		VDD_IO = 2.7V, VO= 0.4V	17	-	-	mA
		VDD_IO = 3.3V, VO= 0.4V	18	-	-	mA
I_{OH}	Output high current[c]	VDD_IO = 1.8V, VO= 1.4V	10	-	-	mA
		VDD_IO = 2.7V, VO= 2.3V	16	-	-	mA
		VDD_IO = 3.3V, VO= 2.3V	17	-	-	mA
R_{PU}	Pullup resistor	-	50	-	65	kΩ
R_{PD}	Pulldown resistor	-	50	-	65	kΩ

[a] Hysteresis enabled
[b] Default drive strength (8mA)
[c] Maximum drive strength (16mA)

Table 5: DC Characteristics

AC Characteristics are defined in Table 6 and Fig. 3.

Pin Name	Symbol	Parameter	Minimum	Typical	Maximum	Unit
Digital outputs	t_{rise}	10-90% rise time[a]	-	1.6	-	ns
Digital outputs	t_{fall}	90-10% fall time[a]	-	1.7	-	ns
GPCLK	t_{JOSC}	Oscillator-derived GPCLK cycle-cycle jitter (RMS)	-	-	20	ps
GPCLK	t_{JPLL}	PLL-derived GPCLK cycle-cycle jitter (RMS)	-	-	48	ps

[a] Default drive strength, CL = 5pF, VDD_IOx = 3.3V

Table 6: Digital I/O Pin AC Characteristics

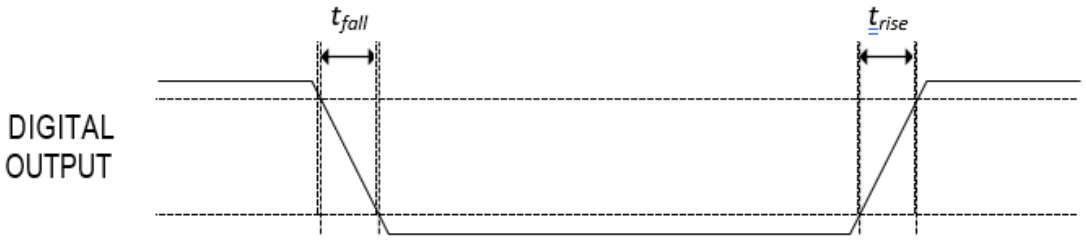


Figure 3: Digital IO Characteristi

7 Power Supplies

The Compute Module 3+ has six separate supplies that must be present and powered at all times; you cannot leave any of them unpowered, even if a specific interface or GPIO bank is unused. The six supplies are as follows:

1. VBAT is used to power the BCM2837 processor core. It feeds the SMPS that generates the chip core voltage.
2. 3V3 powers various BCM2837 PHYs, IO and the eMMC Flash.
3. 1V8 powers various BCM2837 PHYs, IO and SDRAM.
4. VDAC powers the composite (TV-out) DAC.
5. GPIO0-27 VREF powers the GPIO 0-27 IO bank.
6. GPIO28-45 VREF powers the GPIO 28-45 IO bank.

Supply	Description	Minimum	Typical	Maximum	Unit
VBAT	Core SMPS Supply	2.5	-	5.0 + 5%	V
3V3	3V3 Supply Voltage	3.3 - 5%	3.3	3.3 + 5%	V
1V8	1V8 Supply Voltage	1.8 - 5%	1.8	1.8 + 5%	V
VDAC	TV DAC Supply[a]	2.5 - 5%	2.8	3.3 + 5%	V
GPIO0-27_VDD	GPIO0-27 I/O Supply Voltage	1.8 - 5%	-	3.3 + 5%	V
GPIO28-45_VDD	GPIO28-45 I/O Supply Voltage	1.8 - 5%	-	3.3 + 5%	V
SDX_VDD	Primary SD/eMMC Supply Voltage	1.8 - 5%	-	3.3 + 5%	V

[a] Requires a clean 2.5-2.8V supply if TV DAC is used, else connect to 3V3

Table 7: Power Supply Operating Ranges

7.1 Supply Sequencing

Supplies should be staggered so that the highest voltage comes up first, then the remaining voltages in descending order. This is to avoid forward biasing internal (on-chip) diodes between supplies, and causing latch-up. Alternatively supplies can be synchronised to come up at exactly the same time as long as at no point a lower voltage supply rail voltage exceeds a higher voltage supply rail voltage.

7.2 Power Requirements

Exact power requirements will be heavily dependent upon the individual use case. If an on-chip subsys- tem is unused, it is usually in a low power state or completely turned off. For instance, if your application does not use 3D graphics then a large

part of the core digital logic will never turn on and need power. This is also the case for camera and display interfaces, HDMI, USB interfaces, video encoders and decoders, and so on.

Powerchain design is critical for stable and reliable operation of the Compute Module 3+. We strongly recommend that designers spend time measuring and verifying power requirements for their particular use case and application, as well as paying careful attention to power supply sequencing and maximum supply voltage tolerance.

Table 8 specifies the recommended minimum power supply outputs required to power the Compute Module 3+

Supply	Minimum Requirement	Unit
VBAT (CM1)	2000[a]	mW
VBAT (CM3,3L)	3500[a]	mW
3V3	250	mA
1V8	250	mA
VDAC	25	mA
GPIO0-27 VDD	50[b]	mA
GPIO28-45 VDD	50[b]	mA
SDX VDD	50[b]	mA

Recommended minimum. Actual power drawn is very dependent on use-case
Each GPIO can supply up to 16mA, aggregate current per bank must not exceed 50 mA

Table 8: Mimimum Power Supply Requirements

8 Booting

The eMMC Flash device on CM3+ is directly connected to the primary BCM2837 SD/eMMC interface. These connections are not accessible on the module pins.

- On CM3+ Lite this SD interface is available on the SDX pins.

When initially powered on, or after the RUN pin has been held low and then released, the BCM2837 will try to access the primary SD/eMMC interface. It will then look for a file called bootcode.bin on the primary partition (which must be FAT) to start booting the system. If it cannot access the SD/eMMC device or the boot code cannot be found, it will fall back to waiting for boot code to be written to it over USB; in other words, its USB port is in slave mode waiting to accept boot code from a suitable host.

A USB boot tool is available on Github which allows a host PC running Linux to write the BCM2837 boot code over USB to the module. That boot code then runs and provides access to the SD/eMMC as a USB mass storage device, which can then be read and written using the host PC. Note that a Raspberry Pi can be used as the host machine. For those using Windows a precompiled and packeged tool is available. For more information see here.

The Compute Module has a pin called EMMC DISABLE N which when shorted to GND will disable the SD/eMMC interface (by physically disconnecting the SD CMD pin), forcing BCM2837 to boot from USB. Note that when the eMMC is disabled in this way, it takes a couple of seconds from powering up for the processor to stop attempting to talk to the SD/eMMC device and fall back to booting from USB.

Note that once booted over USB, BCM2837 needs to re-enable the SD/eMMC device (by releasing EMMC DISABLE N) to allow access to it as mass storage. It expects to be able to do this by driving the EMMC EN N 1V8 pin LOW, which at boot is initially an input with a pull up to 1V8. If an end user wishes to add the ability to access the SD/eMMC over USB in their product, similar circuitry to that used on the Compute Module IO Board to enable/disable the USB boot and SD/eMMC must be used; that is, EMMC DISABLE N pulled low via MOSFET(s) and released again by MOSFET, with the gate controlled by EMMC EN N 1V8. **Ensure you use MOSFETs suitable for switching at 1.8V (i.e. use a device with gate threshold voltage, Vt, suitable for 1.8V switching).**

9 Peripherals

9.1 GPIO

BCM2837 has in total 54 GPIO lines in 3 separate voltage banks. All GPIO pins have at least two alternative functions within the SoC. When not used for the alternate peripheral function, each GPIO pin may be set as an input (optionally as an interrupt) or an output. The alternate functions are usually peripheral I/Os, and most peripherals appear twice to allow flexibility on the choice of I/O voltage.

GPIO bank2 is used on the module to connect to the eMMC device and for an on-board I2C bus (to talk to the core SMPS and control the special function pins). On CM3+ Lite most of bank2 is exposed to allow a user to connect their choice of SD card or eMMC device (if required).

Bank0 and 1 GPIOs are available for general use. GPIO0 to GPIO27 are bank0 and GPIO28-45 make up bank1. GPIO0-27 VDD is the power supply for bank0 and GPIO28-45 VDD is the power supply for bank1. SDX VDD is the supply for bank2 on CM3+ Lite. These supplies can be in the range 1.8V-3.3V (see Table 7) and are not optional; each bank must be powered, even when none of the GPIOs for that bank are used.

Note that the HDMI HPD N 1V8 and EMMC EN N 1V8 pins are 1.8V IO and are used for special functions (HDMI hot plug detect and boot control respectively). Please do not use these pins for any other purpose, as the software for the module will always expect these pins to have these special functions. If they are unused please leave them unconnected.

All GPIOs except GPIO28, 29, 44 and 45 have weak in-pad pull-ups or pull-downs enabled when the device is powered on. It is recommended to add off-chip pulls to GPIO28, 29, 44 and 45 to make sure they never float during power on and initial boot.

9.1.1 GPIO Alternate Functions

GPIO	Default Pull	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5
0	High	SDA0	SA5	PCLK	-	-	-
1	High	SCL0	SA4	DE	-	-	-
2	High	SDA1	SA3	LCD_VSYNC	-	-	-
3	High	SCL1	SA2	LCD_HSYNC	-	-	-
4	High	GPCLK0	SA1	DPI_D0	-	-	ARM_TDI
5	High	GPCLK1	SA0	DPI_D1	-	-	ARM_TDO
6	High	GPCLK2	SOE_N	DPI_D2	-	-	ARM_RTCK
7	High	SPI0_CE1_N	SWE_N	DPI_D3	-	-	-
8	High	SPI0_CE0_N	SD0	DPI_D4	-	-	-
9	Low	SPI0_MISO	SD1	DPI_D5	-	-	-
10	Low	SPI0_MOSI	SD2	DPI_D6	-	-	-
11	Low	SPI0_SCLK	SD3	DPI_D7	-	-	-
12	Low	PWM0	SD4	DPI_D8	-	-	ARM_TMS
13	Low	PWM1	SD5	DPI_D9	-	-	ARM_TCK
14	Low	TXD0	SD6	DPI_D10	-	-	TXD1
15	Low	RXD0	SD7	DPI_D11	-	-	RXD1
16	Low	FL0	SD8	DPI_D12	CTS0	SPI1_CE2_N	CTS1
17	Low	FL1	SD9	DPI_D13	RTS0	SPI1_CE1_N	RTS1
18	Low	PCM_CLK	SD10	DPI_D14	-	SPI1_CE0_N	PWM0
19	Low	PCM_FS	SD11	DPI_D15	-	SPI1_MISO	PWM1
20	Low	PCM_DIN	SD12	DPI_D16	-	SPI1_MOSI	GPCLK0
21	Low	PCM_DOUT	SD13	DPI_D17	-	SPI1_SCLK	GPCLK1
22	Low	SD0_CLK	SD14	DPI_D18	SD1_CLK	ARM_TRST	-
23	Low	SD0_CMD	SD15	DPI_D19	SD1_CMD	ARM_RTCK	-
24	Low	SD0_DAT0	SD16	DPI_D20	SD1_DAT0	ARM_TDO	-
25	Low	SD0_DAT1	SD17	DPI_D21	SD1_DAT1	ARM_TCK	-
26	Low	SD0_DAT2	TE0	DPI_D22	SD1_DAT2	ARM_TDI	-
27	Low	SD0_DAT3	TE1	DPI_D23	SD1_DAT3	ARM_TMS	-

Table 9: GPIO Bank0 Alternate Functions

GPIO	Default Pull	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5
0	None	SDA0	SA5	PCM_CLK	FL0	-	-
1	None	SCL0	SA4	PCM_FS	FL1	-	-
2	Low	TE0	SA3	PCM_DIN	CTS0	-	CTS1
3	Low	FL0	SA	PCM_DOUT	RTS0	-	RTS1
4	Low	GPCLK0	SA	RING_OCLK	TXD0	-	TXD1
5	Low	FL1	SA0	TE1	RXD0	-	RXD1
6	High	GPCLK0	SOE_N	TE2	SD1_CLK	-	-
7	High	SPI0_CE1_N	SWE_N	-	SD1_CMD	-	-
8	High	SPI0_CE0_N	SD0	TXD0	SD1_DAT0	-	-
9	Low	SPI0_MISO	SD1	RXD0	SD1_DAT1	-	-
10	Low	SPI0_MOSI	SD2	RTS0	SD1_DAT2	-	-
11	Low	SPI0_SCLK	SD3	CTS0	SD1_DAT3	-	-
12	Low	PWM0	SD4	-	SD1_DAT4	SPI2_MISO	TXD1
13	Low	PWM1	SD5	TE0	SD1_DAT5	SPI2_MOSI	RXD1
14	Low	GPCLK1	SD6	TE1	SD1_DAT6	SPI2_SCLK	RTS1
15	Low	GPCLK2	SD7	TE2	SD1_DAT7	SPI2_CE0_N	CTS1
16	None	GPCLK1	SDA0	SDA1	TE0	SPI2_CE1_N	-
17	None	PWM1	SCL0	SCL1	TE1	SPI2_CE2_N	-

Table 10: GPIO Bank1 Alternate Functions

Table 9 and Table 10 detail the default pin pull state and available alternate GPIO functions. Most of these alternate peripheral functions are described in detail in the Broadcom Peripherals Specification document and have Linux drivers available.

9.1.2 Secondary Memory Interface (SMI)

The SMI peripheral is an asynchronous NAND type bus supporting Intel mode80 type transfers at 8 or 16 bit widths and available in the ALT1 positions on GPIO banks 0 and 1 (see Table 9 and Table 10). It is not publicly documented in the Broadcom Peripherals Specification but a Linux driver is available in the Raspberry Pi Github Linux repository (bcm2835 smi.c in linux/drivers/misc).

9.1.3 Display Parallel Interface (DPI)

A standard parallel RGB (DPI) interface is available on bank 0 GPIOs. This up-to-24-bit parallel inter- face can support a secondary display. Again this interface is not documented in the Broadcom Peripher- als Specification but documentation can be found here.

9.1.4 SD/SDIO Interface

The BCM283x supports two SD card interfaces, SD0 and SD1.

The first (SD0) is a proprietary Broadcom controller that does not support SDIO and is the primary interface used to boot and talk to the eMMC or SDX x signals.

The second interface (SD1) is standards compliant and can interface to SD, SDIO and eMMC devices; for example on a Raspberry Pi 3 B+ it is used to talk to the on-board CYW43455 WiFi device in SDIO mode.

Both interfaces can support speeds up to 50MHz single ended (SD High Speed Mode).

9.2 CSI (MIPI Serial Camera)

Currently the CSI interface is not openly documented and only CSI camera sensors supported by the official Raspberry Pi firmware will work with this interface. Supported sensors are the OmniVision OV5647 and Sony IMX219.

It is recommended to attach other cameras via USB.

9.3 DSI (MIPI Serial Display)

Currently the DSI interface is not openly documented and only DSI displays supported by the official Raspberry Pi firmware will work with this interface.

Displays can also be added via the parallel DPI interface which is available as a GPIO alternate function

9.4 USB

The BCM2837 USB port is On-The-Go (OTG) capable. If using either as a fixed slave or fixed master, please tie the USB OTGID pin to ground.

The USB port (Pins USB DP and USB DM) must be routed as 90 ohm differential PCB traces.

Note that the port is capable of being used as a true OTG port however there is no official documentation. Some users have had success making this work.

9.5 HDMI

BCM283x supports HDMI V1.3a.

It is recommended that users follow a similar arrangement to the Compute Module IO Board circuitry for HDMI output.

The HDMI CK P/N (clock) and D0-D2 P/N (data) pins must each be routed as matched length 100 ohm differential PCB traces. It is also important to make sure

that each differential pair is closely phase matched. Finally, keep HDMI traces well away from other noise sources and as short as possible.

Failure to observe these design rules is likely to result in EMC failure.

9.6 Composite (TV Out)

The TVDAC pin can be used to output composite video (PAL or NTSC). Please route this signal away from noise sources and use a 75 ohm PCB trace.

Note that the TV DAC is powered from the VDAC supply which must be a clean supply of 2.5-2.8V. It is recommended users generate this supply from 3V3 using a low noise LDO.

If the TVDAC output is not used VDAC can be connected to 3V3, but it must be powered even if the TV-out functionality is unused.

10. Thermals

The BCM2837 SoC employs DVFS (Dynamic Voltage and Frequency Scaling) on the core voltage. When the processor is idle (low CPU utilisation), it will reduce the core frequency and voltage to reduce current draw and heat output. When the core utilisation exceeds a certain threshold the core votlage is increased and the core frequency is boosted to the maximum working frerquency of 1.2GHz. The voltage and frequency are throttled back when the CPU load reduces back to an 'idle' level OR when the silicon temperature as mesured by the on-chip temperature sensor exceeds 80C (thermal throttling).

A designer must pay careful attention to the thermal design of products using the CM3+ so that performance is not artificially curtailed due to the processor thermal throttling, as the Quad ARM complex in the BCM2837 can generate significant heat output under load.

10.1 Temperature Range

The operating temperature range of the module is set by the lowest maximum and highest minimum of any of the components used.

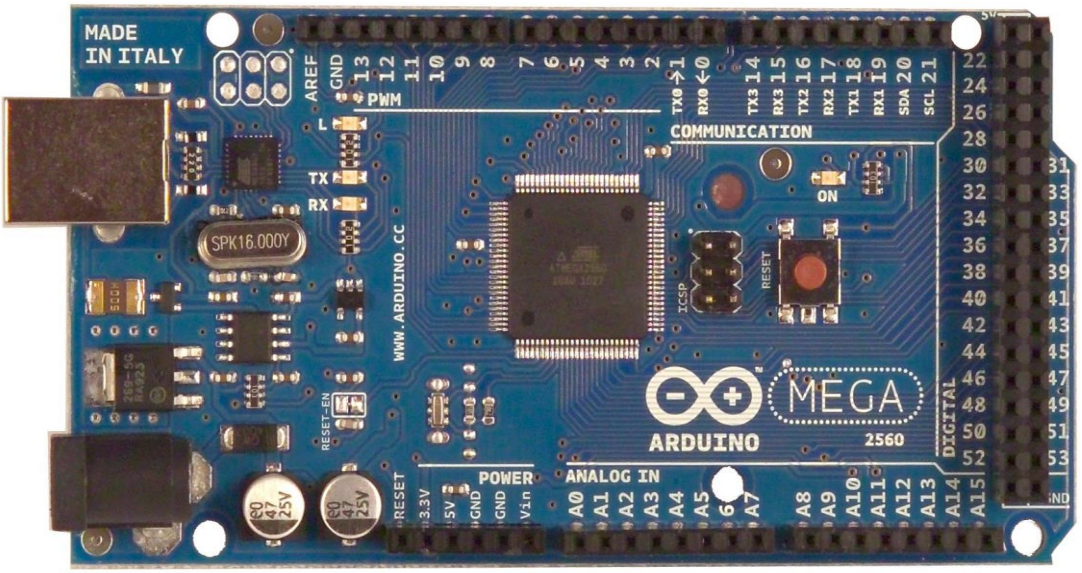
The eMMC and LPDDR2 have the narrowest range, these are rated for -25 to +80 degrees Celsius. Therefore the nominal range for the CM3+ and CM3+ Lite is -25C to +80C.

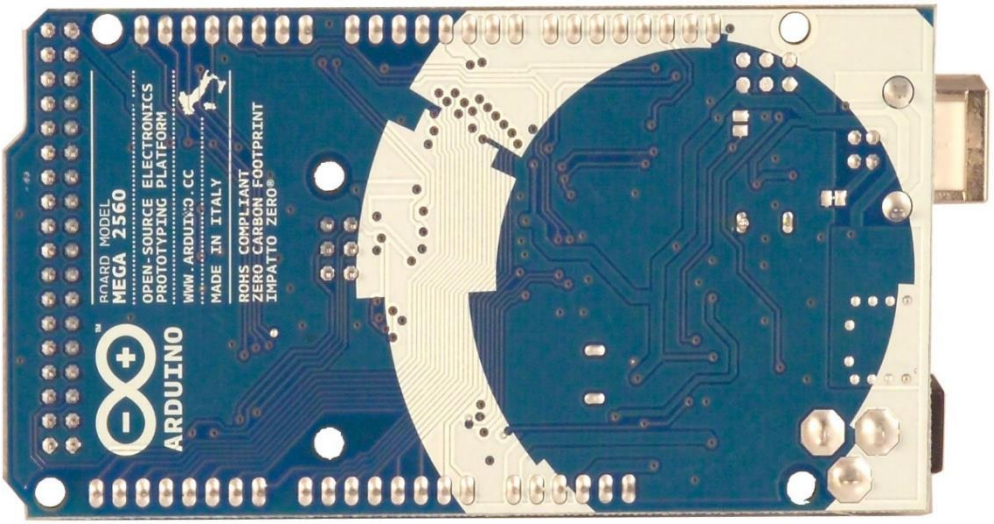
However, this range is the maximum for the silicon die; therefore, users would have to take into account the heat generated when in use and make sure this does not cause the temperature to exceed 80 degrees Celsius.

11 Availability

Raspberry Pi guarantee availability of CM3+ and CM3+ Lite until at least January 2026.

Arduino Mega 2560





Overview

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal

oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC- to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Table 11. Summary

Power

The Arduino Mega can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to- serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.

- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
  GND. Ground pins

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode()
, digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

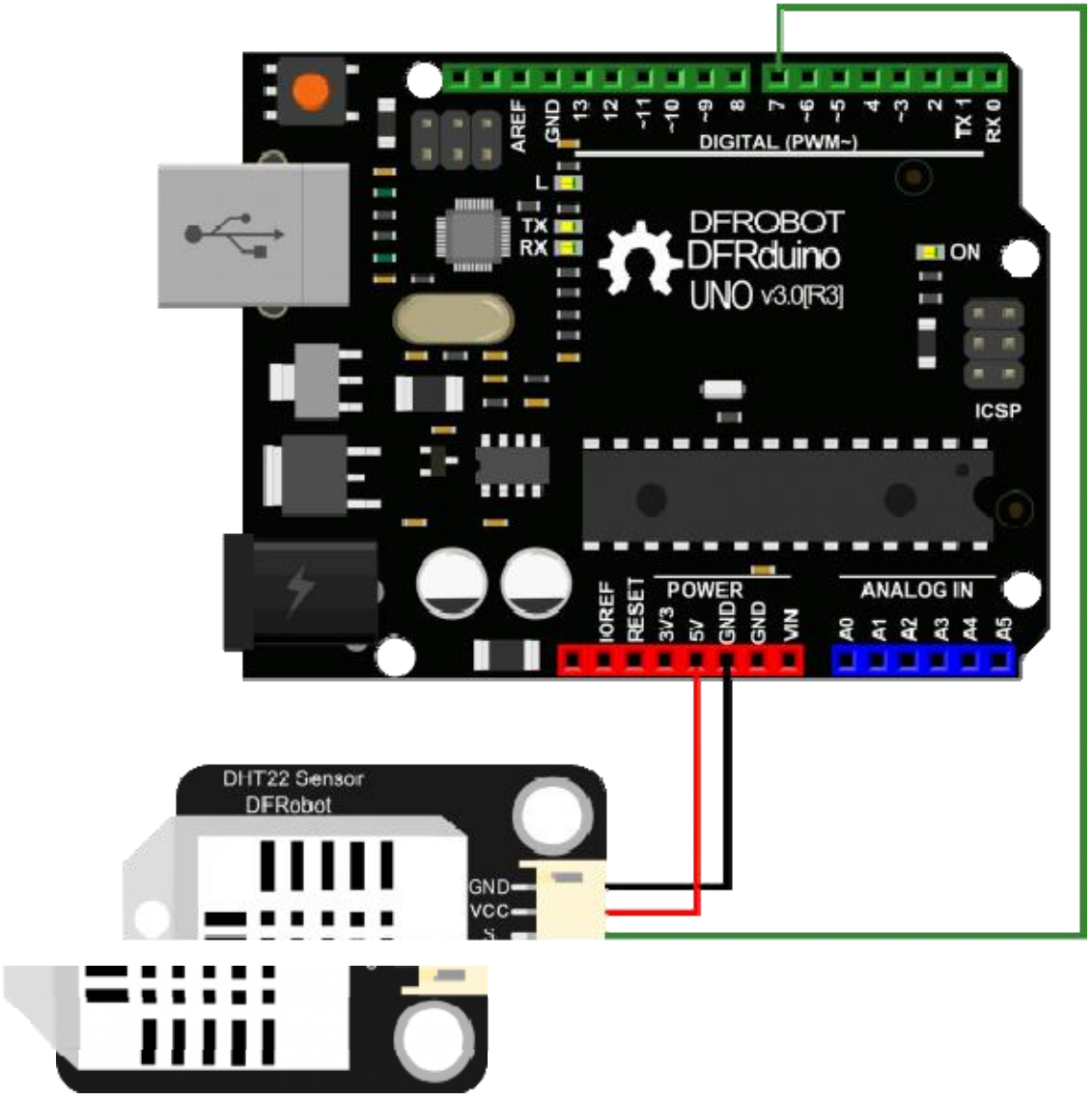
- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be
  configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. **PWM: 0 to 13.** Provide 8-bit PWM output with the analog function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication using the SPI library. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Uno, Duemilanove and Diecimila.
  LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- **I2C: 20 (SDA) and 21 (SCL).** Support I2C (TWI) communication using the Wire library. Note that these pins are not in the same location as the I_2C pins on the Duemilanove or Diecimila.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analog function.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs.
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

DHT22 Temperature and Humidity Sensor



Overview

DHT22 capacitive humidity sensing digital temperature and humidity module is one that contains the compound has been calibrated digital signal output of the temperature and humidity sensors.
Application of a dedicated digital modules collection technology and the temperature and humidity sensing technology, to ensure that the product has high reliability and excellent long-term stability.

The sensor includes a capacitive sensor wet components and a high-precision temperature measurement devices, and connected with a high-performance 8-bit microcontroller. The product has excellent quality, fast response, strong anti-jamming capability, and high cost.

Standard single-bus interface, system integration quick and easy. Small size, low power consumption, signal transmission distance up to 20 meters, making it the best choice of all kinds of applications and even the most demanding applications.

DHT22 has higher precision and can replace the expensive imported SHT10 temperature and humidity sensor.

It can measure the environment temperature and humidity to meet the high demand.The product has high reliability and good stability.If it's used and combined with special sensor Arduino expansion board,it will be easily implemented the interactive effect which related to the temperature and humidity perception.

Caution: DHT22 digital temperature and humidity sensor is designed for analog sensor interfaces.The analog port will be used as the digital which will not occupy the original digital port of the Arduino.The lines of the sensor which can transform the analog function to digital that can be use on digital port.

Specifications

- Supply voltage: 5V
- Output voltage: 0-3.3V
- Temperature range:-40-80ª resolution0.1ª error <±0.5ª
- Humidity range:0-100%RH resolution0.1%RH error±2%RH
- size: 38 x 20mm

HX711 Weight Sensor Module

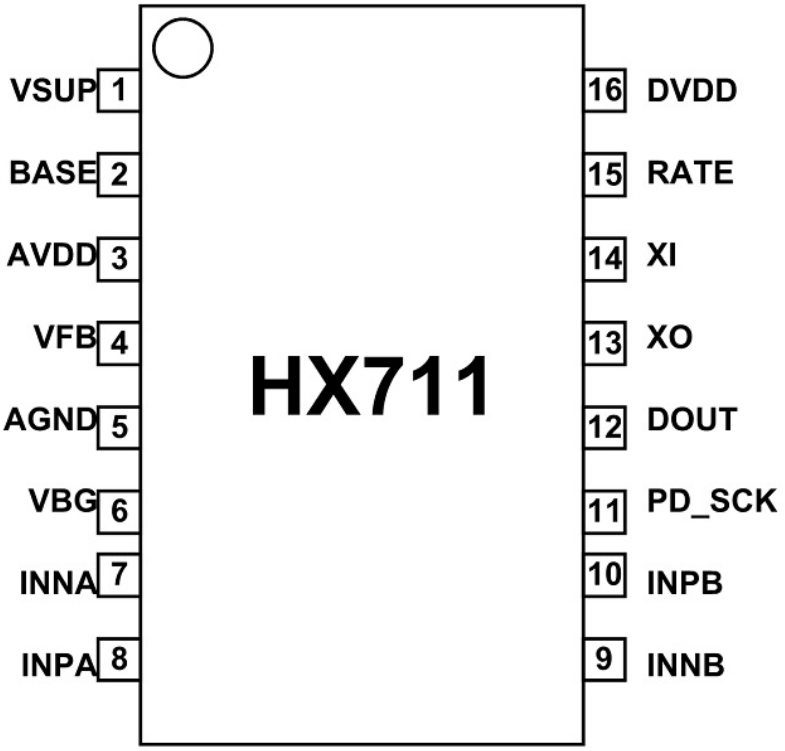
Overview

Based on Avia Semiconductor's patented technology, HX711 is a precision 24-bit analog- to-digital converter (ADC) designed for weigh scales and industrial control applications to interface directly with a bridge sensor.
The input multiplexer selects either Channel A or B differential input to the low-noise programmable gain amplifier (PGA). Channel A can be programmed with a gain of 128 or 64, corresponding to a full-scale differential input voltage of ±20mV or ±40mV respectively, when a 5V supply is connected to AVDD analog power supply pin. Channel B has a fixed gain of 32. On- chip power supply regulator eliminates the need for an external supply regulator to provide analog power for the ADC and the sensor. Clock input is flexible. It can be from an external clock source, a crystal, or the on-chip oscillator that does not require any external component. On-chip power- on-reset circuitry simplifies digital interface initialization.
There is no programming needed for the internal registers. All controls to the HX711 are through the pins.

Features

- Two selectable differential input channels
- On-chip active low noise PGA with selectable gain of 32, 64 and 128
- On-chip power supply regulator for load-cell and ADC analog power supply
- On-chip oscillator requiring no external component with optional external crystal
- On-chip power-on-reset
- Simple digital control and serial interface: pin-driven controls, no programming needed
- Selectable 10SPS or 80SPS output data rate
- Simultaneous 50 and 60Hz supply rejection
- Current consumption including on-chip analog power supply regulator: normal operation < 1.5mA, power down < 1uA
- Operation supply voltage range: 2.6 ~ 5.5V
- Operation temperature range: -40 ~ +85℃
- 16 pin SOP-16 package

Pin Description



Pin #	Name	Function	Description
1	VSUP	Power	Regulator supply: 2.7 ~ 5.5V
2	BASE	Analog Output	Regulator control output (NC when not used)
3	AVDD	Power	Analog supply: 2.6 ~ 5.5V
4	VFB	Analog Input	Regulator control input (connect to AGND when not used)
5	AGND	Ground	Analog Ground
6	VBG	Analog Output	Reference bypass output
7	INA-	Analog Input	Channel A negative input
8	INA+	Analog Input	Channel A positive input
9	INB-	Analog Input	Channel B negative input
10	INB+	Analog Input	Channel B positive input
11	PD_SCK	Digital Input	Power down control (high active) and serial clock input
12	DOUT	Digital Output	Serial data output
13	XO	Digital I/O	Crystal I/O (NC when not used)
14	XI	Digital Input	Crystal I/O or external clock input, 0: use on-chip oscillator
15	RATE	Digital Input	Output data rate control, 0: 10Hz; 1: 80Hz
16	DVDD	Power	Digital supply: 2.6 ~ 5.5V

Table 12 Pin Description

Parameter	Notes	MIN	TYP	MAX	UNIT
Full scale differential input range	V(inp)-V(inn)		±0.5(AVDD/GAIN)		V
Common mode input		AGND+1.2		AVDD-1.3	V
Output data rate	Internal Oscillator, RATE = 0		10		Hz
	Internal Oscillator, RATE = DVDD		80		
	Crystal or external clock, RATE = 0		f_{CLK}/1,105,920		
	Crystal or external clock, RATE = DVDD		f_{CLK}/138,240		
Output data coding	2's complement	800000		7FFFFF	HEX
Output settling time (1)	RATE = 0		400		ms
	RATE = DVDD		50		
Input offset drift	Gain = 128		0.2		mV
	Gain = 64		0.4		
Input noise	Gain = 128， RATE = 0		50		nV(rms)
	Gain = 128， RATE = DVDD		90		
Temperature drift	Input offset（Gain = 128）		±6		nV/℃
	Gain（Gain = 128）		±5		ppm/℃
Input common mode rejection	Gain = 128， RATE = 0		100		dB
Power supply rejection	Gain = 128， RATE = 0		100		dB
Reference bypass（V_{BG}）			1.25		V
Crystal or external clock frequency		1	11.0592	20	MHz
Power supply voltage	DVDD	2.6		5.5	V
	AVDD，VSUP	2.6		5.5	
Analog supply current (including regulator)	Normal		1400		μA
	Power down		0.3		
Digital supply current	Normal		100		μA
	Power down		0.2		

（1）Settling time refers to the time from power up, reset, input channel change and gain change to valid stable output data.

Table 13 Key Electrical Characteristics

Analog Inputs

Channel A differential input is designed to interface directly with a bridge sensor's differential output. It can be programmed with a gain of 128 or 64. The large gains are needed to accommodate the small output signal from the sensor. When 5V supply is used at the AVDD pin, these gains correspond to a full-scale differential input voltage of ±20mV or ±40mV respectively.

Channel B differential input has a fixed gain of
32. The full-scale input voltage range is ±80mV, when 5V supply is used at the

AVDD pin.

Clock Source Options

By connecting pin XI to Ground, the on-chip oscillator is activated. The nominal output data rate when using the internal oscillator is 10 (RATE=0) or 80SPS (RATE=1).

If accurate output data rate is needed, crystal or external reference clock can be used. A crystal can be directly connected across XI and XO pins. An external clock can be connected to XI pin, through a 20pF ac coupled capacitor. This external clock is not required to be a square wave. It can come directly from the crystal output pin of the MCU chip, with amplitude as low as 150 mV.

When using a crystal or an external clock, the internal oscillator is automatically powered down.

Output Data Rate and Format

When using the on-chip oscillator, output data rate is typically 10 (RATE=0) or 80SPS (RATE=1).

When using external clock or crystal, output data rate is directly proportional to the clock or crystal frequency. Using 11.0592MHz clock or crystal results in an accurate 10 (RTE=0) or 80SPS (RATE=1) output data rate.

The output 24 bits of data is in 2's complement format. When input differential signal goes out of the 24 bit range, the output data will be saturated at 800000h (MIN) or 7FFFFFh (MAX), until the input signal comes back to the input range.

Serial Interface

Pin PD_SCK and DOUT are used for data retrieval, input selection, gain selection and power down controls.

When output data is not ready for retrieval, digital output pin DOUT is high. Serial clock input PD_SCK should be low. When DOUT goes to low, it indicates data is ready for retrieval. By applying 25~27 positive clock pulses at the PD_SCK pin, data is shifted out from the DOUT output pin. Each PD_SCK pulse shifts out one bit, starting with the MSB bit first, until all 24 bits are shifted out. The 25th pulse at PD_SCK input will pull DOUT pin back to high (Fig.2).
Input and gain selection is controlled by the number of the input PD_SCK pulses (Table 3). PD_SCK clock pulses should not be less than 25 or more than 27 within one conversion period, to avoid causing serial communication error.

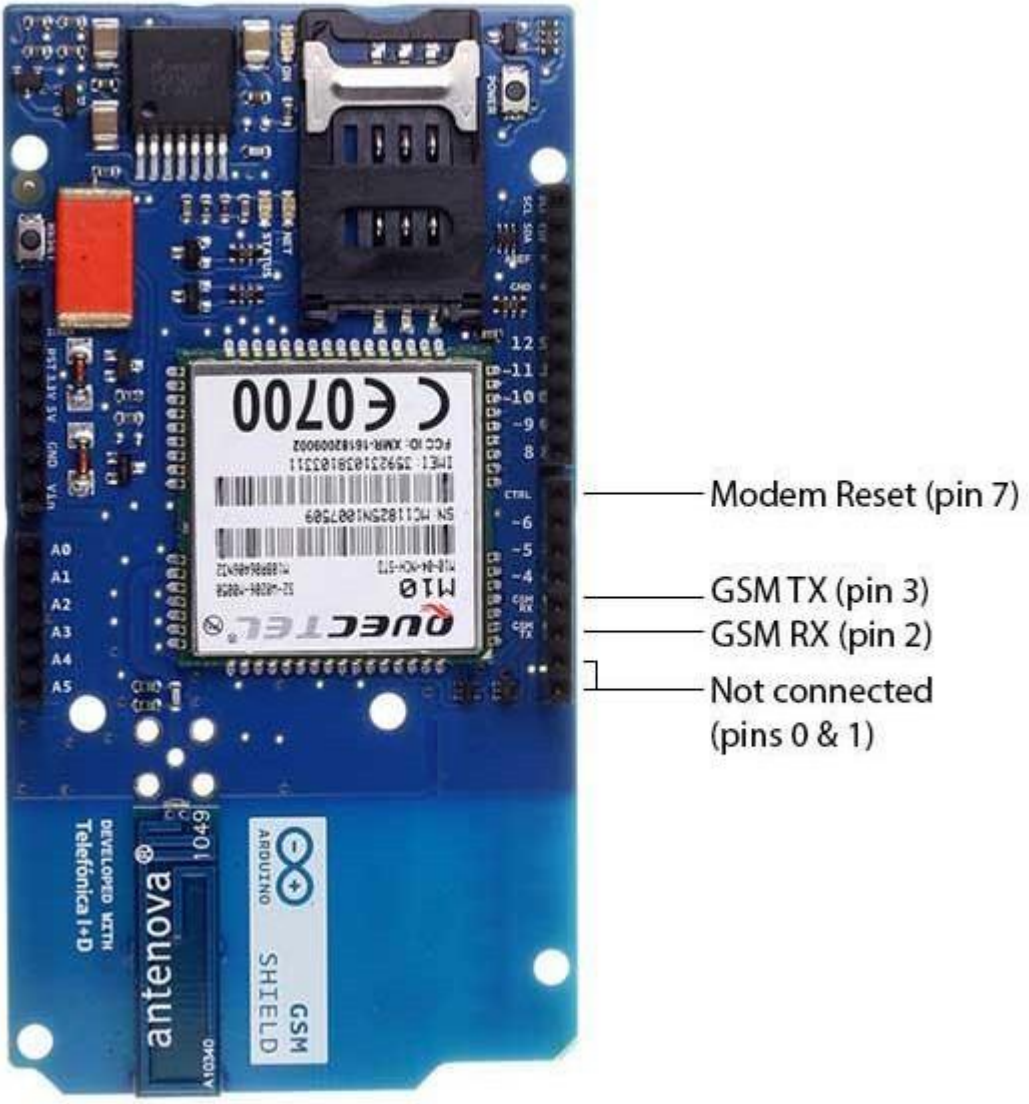
PD_SCK Pulses	Input channel	Gain
25	A	128
26	B	32
27	A	64

Table 14 Input Channel and Gain Selection

GSM Module

Overview

The Arduino GSM Shield connects your Arduino to the internet using the GPRS wireless network. Just plug this module onto your Arduino board, plug in a SIM card from an operator offering GPRS coverage and follow a few simple instructions to start controlling your world through the internet. You can also make/receive voice calls (you will need an external speaker and microphone circuit) and send/receive SMS messages.



- Requires and Arduino board (not included)
- Operating voltage 5V (supplied from the Arduino Board)
- Connection with Arduino Uno on pins 2, 3 (Software Serial) and 7 (reset).

Description

The Arduino GSM Shield allows an Arduino board to connect to the internet, make/receive voice calls and send/receive SMS messages. The shield uses a radio modem M10 by Quectel (datasheet). It is possible to communicate with the board using AT commands. The GSM library has a large number of methods for communication with the shield.
The shield uses digital pins 2 and 3 for software serial communication with the M10. Pin 2 is connected to the M10's TX pin and pin 3 to its RX pin. See these notes for working with an Arduino Mega, Mega ADK, or Leonardo. The modem's PWRKEY pin is connected to Arduino pin 7.
The M10 is a Quad-band GSM/GPRS modem that works at frequencies GSM850MHz, GSM900MHz, DCS1800MHz and PCS1900MHz. It supports TCP/UDP and HTTP protocols through a GPRS connection. GPRS data downlink and uplink transfer speed maximum is 85.6 kbps.
To interface with the cellular network, the board requires a SIM card provided by a network operator. See the getting started page for additional information on SIM usage.
The most recent revision of the board uses the 1.0 pinout on rev 3 of the Arduino Uno board.

Power requirements

It is recommended that the board be powered with an external power supply that can provide between 700mA and 1000mA. Powering an Arduino and the GSM shield from a USB connection is not recommended, as USB cannot provide the required current for when the modem is in heavy use.
The modem can pull up to 2A of current at peak usage, which can occur during data transmission. This current is provided through the large orange capacitor on the board's surface.

On board indicators

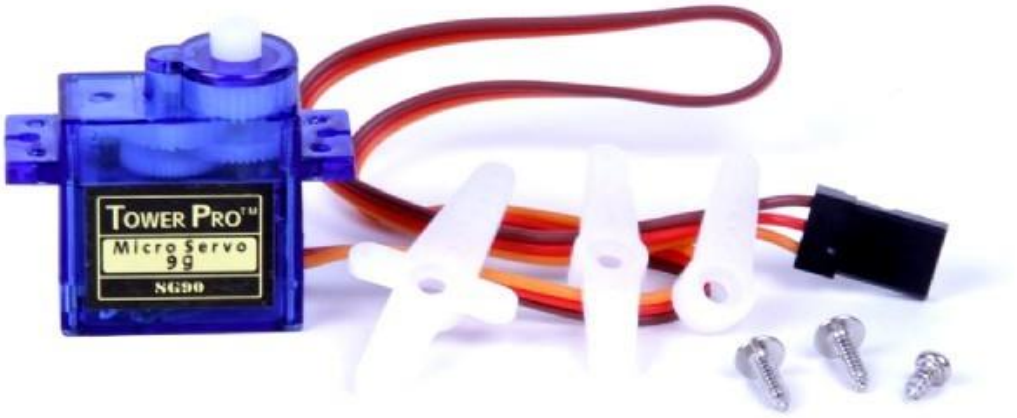
The shield contains a number of status LEDs:

- On: shows the Shield gets power.
- Status: turns on to when the modem is powered and data is being transferred to/from the GSM/GPRS network.
- Net: blinks when the modem is communicating with the radio network.

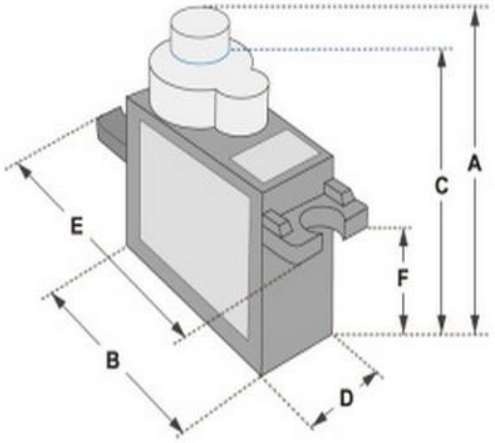
On board interfaces

The shield supports AIN1 and AOUT1 as audio interfaces; an analog input channel and an analog output channel. The input, exposed on pins MIC1P/MIC1N, can be used for both microphone and line inputs. An electret microphone can be used for this interface. The output, exposed as lines SPK1P/SPK1N, can be used with either a receiver or speaker. Through the modem, it is possible to make voice calls. In order to speak to and hear the other party, you will need to add a speaker and microphone.

Servo Motor

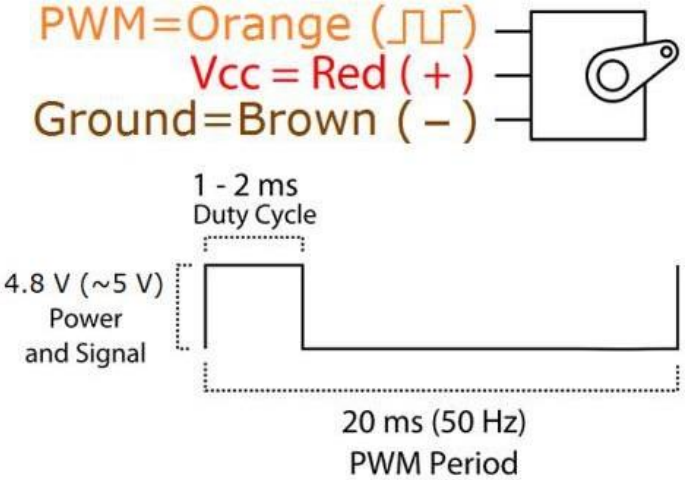


Tiny and lightweight with high output power. Servo can rotate approximately 180 degrees (90 in each direction), and works just like the standard kinds but smaller. You can use any servo code, hardware or library to control these servos. Good for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. It comes with a 3 horns (arms) and hardware.

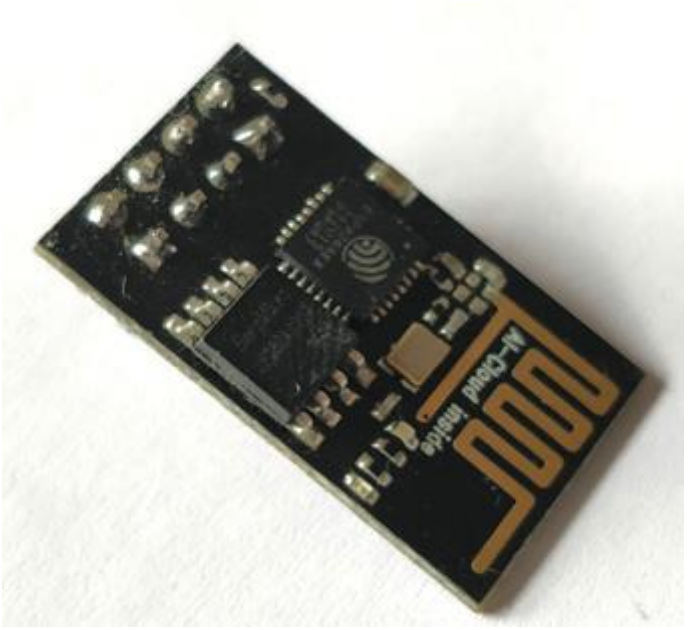


Dimensions & Specifications
A (mm) : 32
B (mm) : 23
C (mm) : 28.5
D (mm) : 12
E (mm) : 32
F (mm) : 19.5
Speed (sec) : 0.1
Torque (kg-cm) : 2.5
Weight (g) : 14.7
Voltage : 4.8 - 6

Position "0" (1.5 ms pulse) is middle, "90" (~2ms pulse) is middle, is all the way to the right, "-90" (~1ms pulse) is all the way to the left.



ESP8266 WIFI Module



Overview

The ESP8266 is a very user friendly and low cost device to provide internet connectivity to your projects. The module can work both as a Access point (can create hotspot) and as a station (can connect to Wi-Fi), hence it can easily fetch data and upload it to the internet making Internet of Things as easy as possible. It can also fetch data from internet using API's hence your project could access any information that is available in the internet, thus making it smarter. Another exciting feature of this module is that it can be programmed using the Arduino IDE which makes it a lot more user friendly. However this version of the module has only 2 GPIO pins (you can hack it to use upto 4) so you have to use it along with another microcontroller like Arduino, else you can look onto the more standalone ESP-12 or ESP-32 versions. So if you are looking for a module to get started with IOT or to provide internet connectivity to your project then this module is the right choice for you.

ESP8266 Pin Configuration

Pin Number	Pin Name	Alternate Name	Normally used for	Alternate purpose
1	Ground	–	Connected to the ground of the circuit	–
2	TX	GPIO – 1	Connected to Rx pin of programmer/uC to upload program	Can act as a General purpose Input/output pin when not used as TX
3	GPIO-2	–	General purpose Input/output pin	–
4	CH_EN	–	Chip Enable – Active high	–
5	GPIO – 0	Flash	General purpose Input/output pin	Takes module into serial programming when held low during start up
6	Reset	–	Resets the module	–
7	RX	GPIO – 3	General purpose Input/output pin	Can act as a General purpose Input/output pin when not used as RX
8	Vcc	–	Connect to +3.3V only	

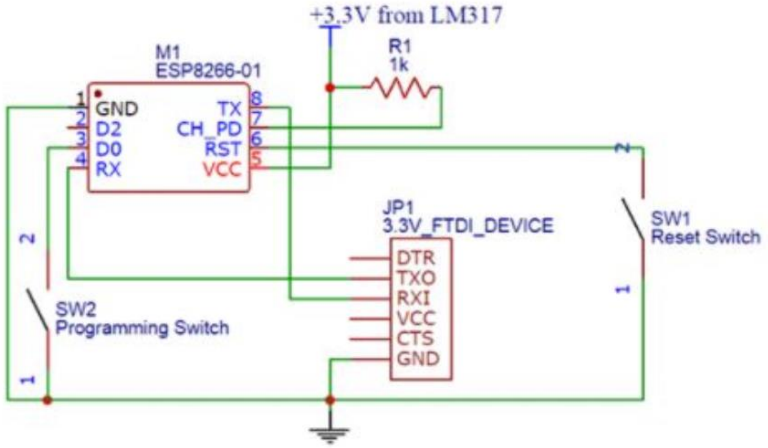
ESP8266-01 Features

- Low cost, compact and powerful Wi-Fi Module
- Power Supply: +3.3V only
- Current Consumption: 100mA
- I/O Voltage: 3.6V (max)
- I/O source current: 12mA (max)
- Built-in low power 32-bit MCU @ 80MHz
- 512kB Flash Memory
- Can be used as Station or Access Point or both combined

- Supports Deep sleep (<10uA)
- Supports serial communication hence compatible with many development platform like Arduino
- Can be programmed using Arduino IDE or AT-commands or Lua Script

How to use the ESP8266 Module

There are so many methods and IDEs available to with ESP modules, but the most commonly used on is the Arduino IDE. So let us discuss only about that further below.
The ESP8266 module works with 3.3V only, anything more than 3.7V would kill the module hence be cautions with your circuits. The best way to program an ESP-01 is by using the FTDI board that supports 3.3V programming. If you don't have one it is recommended to buy one or for time being you can also use an Arduino board. One commonly problem that every one faces with ESP-01 is the powering up problem. The module is a bit power hungry while programming and hence you can power it with a 3.3V pin on Arduino or just use a potential divider. So it is important to make a small voltage regulator for 3.31v that could supply a minimum of 500mA. One recommended regulator is the LM317 which could handle the job easily.
A simplified circuit diagram for using the ESP8266-01 module is given below



Applications

- IOT Projects
- Access Point Portals
- Wireless Data logging
- Smart Home Automation
- Learn basics of networking
- Portable Electronics
- Smart bulbs and Sockets

APPENDIX D

CURRICULUM VITAE

Cariño, Ian Jasper G.

Blk 67 Lot D6 P4 Elisa Homes Bacoor, Cavite
+639264847338
iancarino.ic@gmail.com

Personal Information

Birthday: January 1, 1999
Age: 21
Height: 5'7"
Weight: 122 lbs.
Citizenship: Filipino

Personal Skills

Has knowledge in troubleshooting and designing electronic system.

Has knowledge on Microsoft Applications (Word and PowerPoint).

Has knowledge on PCB Express and Circuit Wizard.

Has knowledge on basic programming in MATLAB, Octave and Python.

Achievements

Electronics Technician Board Passer (October 2018)

Career Objective

To grow capably by giving my skills to association and vice-versa. To work in demanding environment where all my skills and efforts to explore and familiar myself in varied fields and comprehend my potential and add to the growth of organization and inspiring performance.

Educational Attainment

Tertiary: *Technological University of the Philippines – Manila*
Ayala Blvd., Ermita, Manila
Bachelor of Science in Electronics Engineering
SY: 2015 – 2020

Secondary: *Bacoor National High School*
Molino 3 Bacoor, Cavite
SY: 2011 – 2015

Primary: *Likha Molino 4 Elementary School*
Molino 4 Bacoor Cavite
SY: 2003 – 2011

Seminars:

Research Themes and Trends in Electronics Engineering
Technological University of the Philippines - Ermita, Manila
July 30, 2018

APPRECIATE: Annual Presentation of Project Research in Electromechanical, Civil, Information and Telecommunications Engineering
Technological University of the Philippines - Ermita, Manila
2018 - 2019

IECEP MSC STUDENT SUMMIT 2020
Pamantasan ng Lungsod ng Maynila - Intramuros Manila/ Institute of Electronics Engineers of the Philippines, February 8, 2020

Organization Affiliations

Organization of Electronics Engineering Students
Member (2015 – 2020)

Institute of Electronics Engineers of the Philippines Manila
Member (2017 – Present)

Elaine Joy L. Elequin

568-D Purok 4 Cupang, Muntinlupa City
09566584845
elainejoy.elequin@tup.edu.ph

Personal Information

Birthday: November 12, 1998
Age: 22
Height: 5'5"
Weight: 136 lbs.
Citizenship: Filipino

Personal Skills

- Proficient in verbal and written communication (Filipino, English)
- Basic Programming knowledge (HTML, MATLAB, Octave, Python)
- Highly adaptable and flexible
- Leadership, People oriented and a Team Player

Achievements

- Licensed Electronics Technician
- CHED Scholar ng Bayan (2015-2018)
- Muntinlupa Business High School- Main – 8th Honorable Mention Batch 2015

Career Objective

It's my top priority to impart the knowledge and skills I've learned for the organization's best interest and maximize a position that will strengthen my potential as a more competent electronics engineer.

Educational Attainment

Tertiary: *Technological University of the Philippines – Manila*
Ayala Blvd., Ermita, Manila
Bachelor of Science in Electronics Engineering
SY: 2015 – 2020

Secondary: *Muntinlupa Business High School - Main*
146B Espeleta St., Cupang, Muntinlupa City
SY: 2011 – 2015

Primary: *Cupang Elementary School*
Manuel L. Quezon, Cupang, Muntinlupa 1770 Metro Manila
SY: 2005 – 2011

Seminars:

Student Summit 2020 Career Pathways: Walk with Prominence through Profession
IECEP- Manila Student Chapter, February 8, 2020

Cybersecurity Bureau Cybersecurity: The New Normal
Department of Information and Communications Technology, June 24, 2020

Power Electronics: Basic Concepts of Power Converters
Maxim Integrated, August 18, 2020

Organization Affiliations

Organization of Electronics Engineering Students
Public Relations Officer (2017-2018)

Organization of Electronics Engineering Students
Member (2015 – 2020)

Institute of Electronics Engineers of the Philippines Manila Student Chapter
Member (2017 – Present)

Cherrie Lynne R. Gangcuangco

2164 Caliraya St., Holiday Homes, San Antonio,
San Pedro City, Laguna
+63 (917) 183 6417
chegangcuangco@gmail.com

Personal Information

Birthday: August 19, 1999
Age: 21
Height: 5'0"
Weight: 60 kg.
Citizenship: Filipino

Personal Skills

Excellent written and verbal communication skills
Basic Programming Knowledge (Python, HTML and GNU Octave)
Proficient in Microsoft Office Applications (Word, Excel, PowerPoint)
Basic Electronics Skills
A team player and can work under pressure

Achievements

Licensed Electronics Technician
Top Team – Student Design Challenge

Career Objective

To work in a dynamic environment where I can fully utilize and share my skills and knowledge, contributing to the success of the company as I progress professionally and personally.

Educational Attainment

Tertiary: *Technological University of the Philippines – Manila*
Ayala Blvd., Ermita, Manila
Bachelor of Science in Electronics Engineering
SY: 2015 – 2020

Secondary: *Amazing Grace School*
San Vicente, San Pedro City, Laguna
SY: 2011 – 2015

Primary: *Amazing Grace School*
San Vicente, San Pedro City, Laguna
SY: 2005 – 2011

Seminars:

Student Design Challenge
DOST– UPSCALE Innovation Hub, Clark Pampanga, November 2019

Student Summit 2020 Career Pathways: Walk with Prominence through Profession
IECEP – Manila Student Manila, February 8, 2020

Power Electronics: Basic Concepts of Power Converters Maxim Integrated, August 18, 2020

Organization Affiliations

Organization of Electronics Engineering Students
Member (2015 – 2020)

Institute of Electronics Engineers of the Philippines Manila
Member (2017 – Present)

Organization of Electronics Engineering Students
Secretary-General (2018 – 2019)

Ellen Grace S. Jayomana

Block 12 Lot 43 Phase2A Southville 2
Brgy. Aguado, Trece Martires City, Cavite
+639422998261
ellen.jayomana@gmail.com

Personal Information

Birthday: November 29, 1995

Age: 24

Height: 5'8"

Weight: 58 kg.

Citizenship: Filipino

Personal Skills

Computer Literate (Proficient in Microsoft Office Word, Excel, Power Point)

Programming Languages (Basic Knowledge in Python, MATLAB and GNU Octave)

Basic electronic skills (soldering and troubleshooting techniques)

Can easily adapt new concepts

Initiator, leader and passionate about working

Achievements

Licensed Electronics Technician
October 2018

Top Team – Student Design Challenge

Career Objective

"To succeed in a stimulating and challenging environment, building the success of the company while I experience advancement opportunities."

Educational Attainment

Tertiary: *Technological University of the Philippines – Manila*
Ayala Blvd., Ermita, Manila
Bachelor of Science in Electronics Engineering
SY: 2015 – 2020
Technological University of the Philippines-Manila
Ayala Blvd., Ermita, Manila
Electronic Communications Engineering Technology (ECET)
SY: 2013-2016

Secondary: *Luis Aguado National High School*
Brgy. Aguado Trece Martires City, Cavite
SY: 2009 – 2012

Primary: *Southville Elementary School*
City, Province
SY: 2003 – 2009

Seminars:

Structured Cabling System and Design Workshop
TUP-MNL, August 2019

Student Design Challenge
UPScale, UP Diliman Quezon City, November 2019

IECEP-MSP Student Summit 2020 "Career Pathways", "Graduate Studies: To Pursue or Not To Pursue", "Redefining Success"
IECEP Manila, February 2020

AM, FM, TV BROADCASTING System and Station Design for Electronics Students and Professionals
OUES, TUP-MNL, January-May 2020

Mixed Signal IC Design for Students in and Professionals (54 hours)
OUES, TUP-MNL, June-July 2020

Publishing Extension Outputs in Research Journals
OUES, TUP-MNL, July 2020

Organization Affiliations

DOST – UPSCALE Innovation Hub, University of the Philippines

Organization of Electronics Engineering Students
Member (2015 – 2020)

Institute of Electronics Engineers of the Philippines Manila
Member (2017 – Present)

Lalaine B. Laman

B40 L5 Puffin St., Camella Homes Woodhills Subdivision,
Brgy. San Antonio, San Pedro City, Laguna
09053072824
lalainelaman@gmail.com

Personal Information

Birthday: November 8, 1998
Age: 22
Height: 5'3"
Weight: 125 lbs.
Citizenship: Filipino

Personal Skills

Excellent written and verbal communication skills
Fast learner with strong analytical skills
Team player and can work under pressure
Proficiently computer literacy in MS Office and G Suite
Strong technical capabilities

Achievements

Top team – Student Design Challenge
Electronics Technician Licensure Passer
Leadership Awardee

Career Objective

Enthusiastic & highly motivated Electronics Engineering graduate seeking to use her extensive knowledge to be part of the company's eminent success and further achieve its goal and excellence in the field.

Educational Attainment

Tertiary: *Technological University of the Philippines – Manila*
Ayala Blvd., Ermita, Manila
Bachelor of Science in Electronics Engineering
SY: 2015 – 2020

Secondary: *Sisters of Mary Immaculate School*
San Pedro City, Laguna
SY: 2011 – 2015

Primary: *Sisters of Mary Immaculate School*
San Pedro City, Laguna
SY: 2005 – 2011

Seminars:

Power Electronics: Basic Concepts of Power Converter
Maxim Integrated, August 18, 2020

Publishing Extension Outputs in Research Journals
Office of the Vice President for Research and Extension, July 3, 2020

Student Summit 2020 Career Pathways
IECEP – Manila Student Manila, February 8, 2020

Organization Affiliations

Organization of Electronics Engineering Students Member (2015 – 2020)

Institute of Electronics Engineers of the Philippines Manila Member (2017 – Present)

APPENDIX E

POWER CONSUMPTION COMPUTATION

Node MCU

$V_{supply\ max}$ =3.6Vdc

I_{max}=12mA

$P_{T\ Node\ MCU}$ = (V_{max})*(I_{max})*(# of Node MCU)

= (3.6V)*(12mA)*(2)

= **86.4mW**

Raspberry Pi 3B+

$V_{supply\ max}$ =5V

I_{max}=2A

$P_{T\ RaspPi}$= (V_{max})*(I_{max})

= (5V) *(2A)

= **10 W**

GSM Module (SIM800L)

$V_{supply\ max}$= 5Vdc

I_{max}= 1.5A

$P_{T\ GSM}$= (V_{max})*(I_{max})

= (5Vdc)*(1.5A)

= 7.5W

Ultrasonic Mist Maker

Rated power base on datasheet

P_{max}=2.5W

5V DC Cooling fan

$V_{supply\ max}$= 5Vdc

I_{max}= 500mA

$P_{T\ Fan}$= (V_{max})*(I_{max})

=(5Vdc)*(500mA)

=**2.5W**

Arduino MEGA

$V_{supply\ max}$= 5Vdc

I_{max}= 50mA

$P_{T\ Arduino}=(V_{max})*(I_{max})$

=(5Vdc)*(50mA)

=**250mW**

Load Cell

Input Resistance: 1000±15 Min.: 985

Typical: 1k

Max.: 1015

Excitation Voltage: 5V

Current Consumption: Min.: =4.93mA

Max.: =5.076mA

Typ: =5mA

$P_{T\ Loadcell}$ =(# of Load Cell)*(V_{Supply})*(I_{max})

= (20)*(5Vdc)*(5.076mA)

= **507.6mW**

HX711

V_{supply} =5V

I_{Supply} =1.6mA

$P_{T\ HX711}$ =(# of HX711)*(V_{Supply})*(I_{Supply})

= (20)*(5Vdc)*(1.6mA)

= **160mW**

DHT22

V_{Power} = 5V

I_{max}=2.5mA

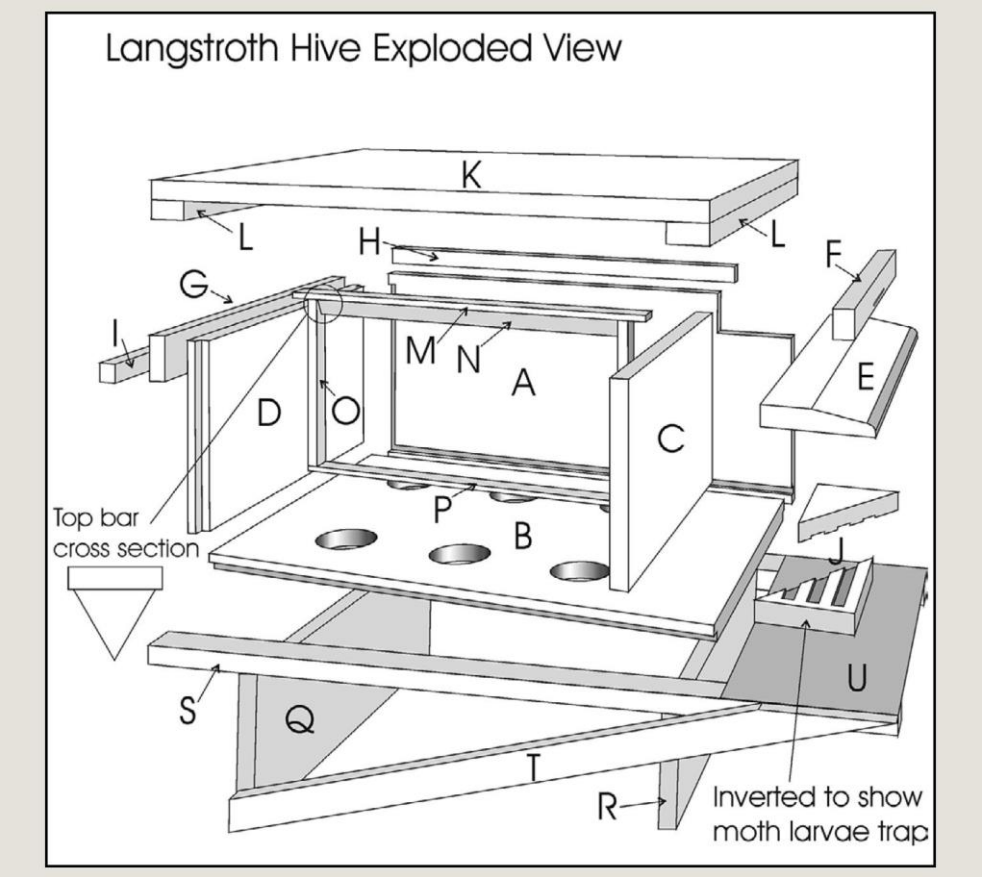
$P_{T\ DHT22}$ =(V_{Supply})*(I_{Supply})

=(5V)*(2.5mA)

=**12.5mW**

APPENDIX F

STANDARD BEEHIVE SIZE REFERENCE



Source: https://www.beeculture.com/build-original-langstroth-hive/

APPENDIX G

BEEFARM INTERVIEW

Dexter's Apiary Interview

1. How did you start culturing bees?

Out of Curiosity. Malimit kami kumain ng honey. We always buy honey from department store, fruit stand, or yung padala ng kamaganak from abroad. I got curious pano sila alagaan, iculture. So nagresearch ako sa net, sa forums then i found out na may mga bees pala na ppwede alagaan for honey production. So I start learning from the internet, sa mga mentor na nakasalamuha namin and they advised us na depende sa purpose mo. So if you're going into pollination, if you're going into honey production, or if you just want honey for your table. Dun nin nakilala yung ibang species na ppwede alagaan. So we started with european bees. From there, it cost us Php 20,000+, that was 2009.

2. Where do you obtain the beehives and the bees?

Almost 10 years, since 2009. We started with two colonies

3. Saan niyo po nakuha yung mga beehives, colony?

Yung unang beehive was from a beekeeper as well. Yung second colony namin is from UPLB.

4. Anong types po ng bee yung cinuculture nyo?

We have two types: The stingless and the european.

5. How often do you inspect the hives/bees?

Paano po yung process ng pagiinspect? once a week. Usually, we check on the frames to know the condition of the colony. if needed mag add ng frame, kung need ng food supplement.

6. What are the problems that you've experienced as a beekeeper?

Marami. First syempre yung maintenance, to check yung mga frames to make the colony improve and provide them kung anong kailangan nila. Second, condition ng weather pattern natin. And lastly shempre yung mga peste. Mga mites, ants, ibon and mga bee eaters. And mga mold na nanggagaling sa chalk wood. Lahat yan pinagdadaanan ng beekeepers

7. Sa mga naencounter nyo na pest, paano niyo po nattreat or naiiwasan?

Kailangan provide them kung ano yung mga kailangan ibigay sakanila. of course kailangan before or after ng season icheck kung may kailangan itreat. kung may mites, kung need palitaan frames or ireplace yung combs.

8. Ano po yung method pala malaman nyo kung may mites sa hives?

Isa yung ipm(integrate pest management) board, para macontrol, hindi maprevent. and paggamit ng natural products like alagao leaves, lemon leaves na nilagagay sa ibabaw ng frames. maliban don, yung chemical based, mga miticides.

9. When do you harvest honey? Is there an indication that the honey is ready for harvesting?

Harvest season is during the honey flow season na summer time. Depende sa colony location.

10. Nakakapekto po yung temperature and humidity sa mga bees?

Kapag humid ang paligid, madidisrupt ang temperature sa colony. Ayaw nila ng biglaang changes sa temp, may time na naaggitate sila. Kaya nagijnspect kami pag maganda yung weather condition

11. Gano po kalayo yung house niyo sa farm?

Ilang meters lang. Backyard farm kasi meron kami.

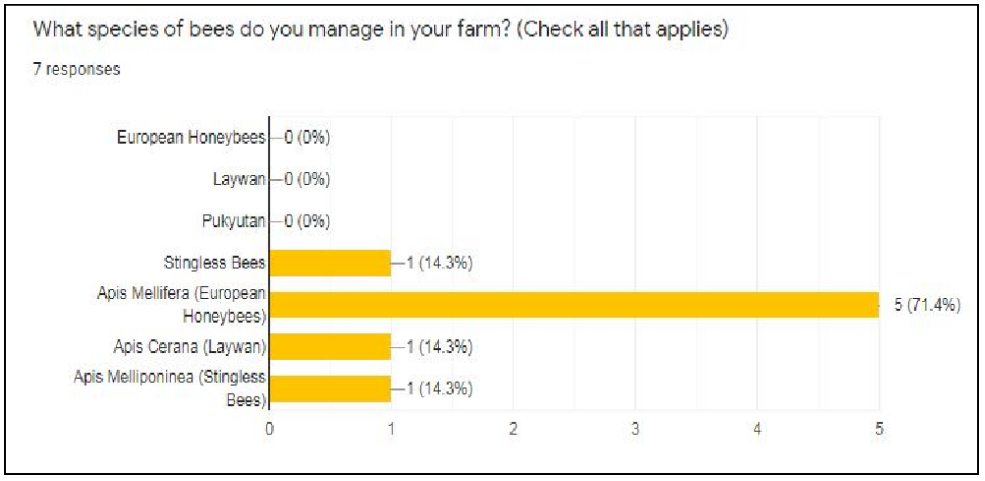
12. May suggestion po ba kayo about keeping?

Dapat magaral tungkol sa bees in order to improve and upgrade the beekeeping.

13. Are you open for innovations in your farm?

Yes of course dahil yun pala yung isa sa mga pwedeng gawin.

SURVEY



Number of bee colonies

7 responses

20
90
250
100
3
350
100

How much honey are produced per colony? (in liters)

7 responses

20
Pls don't use liters when it's Honey. Have it in kilos(30-40kgs/Colony)
22-30kg
1
30
Average of 1liters per year
201000

How much is the honey sold?

7 responses

400

600-800/kg

50NZD/kg

100

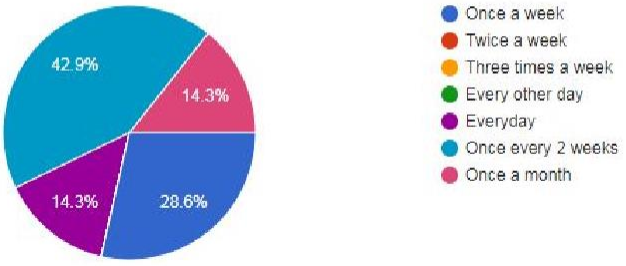
800

250 pesos per bottle of gin

1000 per kilo

How often do you inspect the beehives?

7 responses



How do you deal with these pests? What method do you use? (Please specify for every pest)

7 responses

Api strips

Apistan, Mitecide & other new mites strips

Early prevention treatment for varroa mites through strips(Apistan, Bayvarol, Apitraz)

Always check the beehive

Water pan and miticide

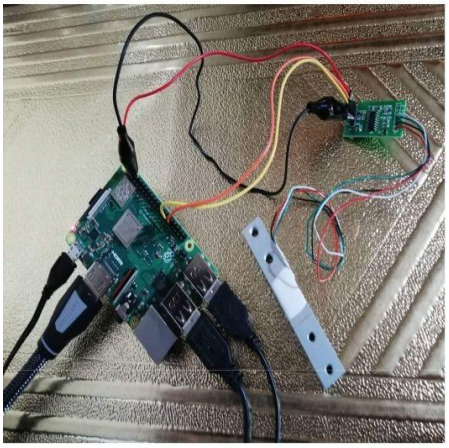
Making the colonies strong.
Proper sanitation when splitting the hive. (We do not try to spill honey or throw pollen or other hive parts on site to prevent pest from attacking the hive.)

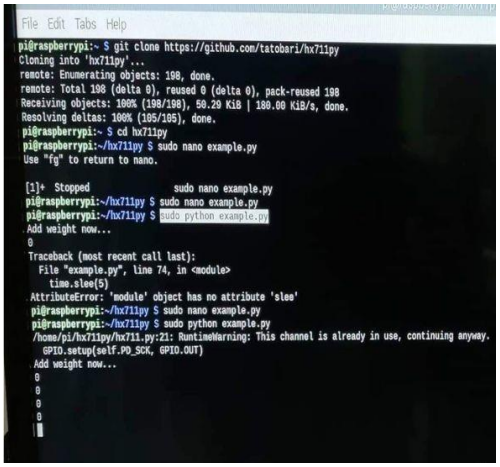
Mites with different miticides

APPENDIX H

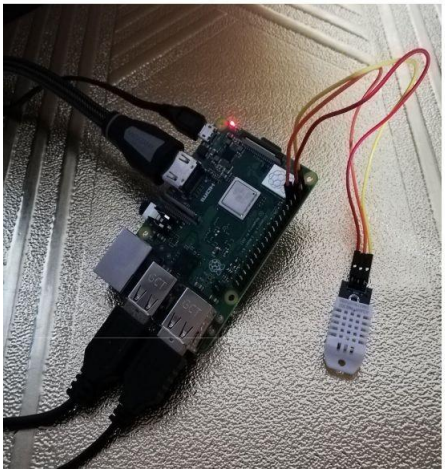
DOCUMENTATION

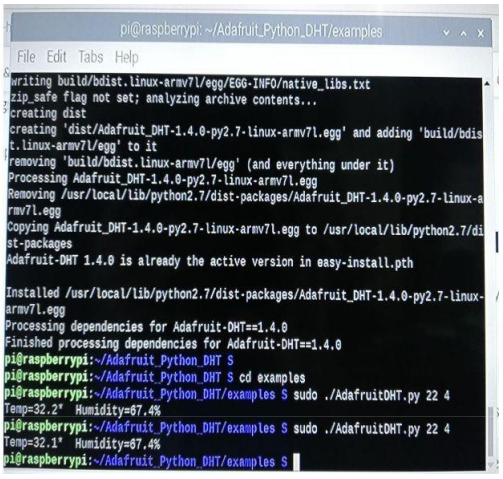
Testing of Load Cells



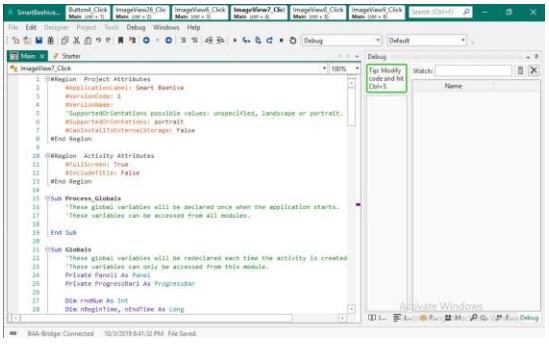


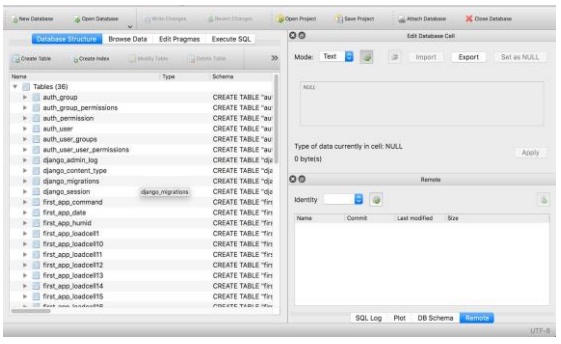
Testing of DHT22 Temperature and Humidity Sensor



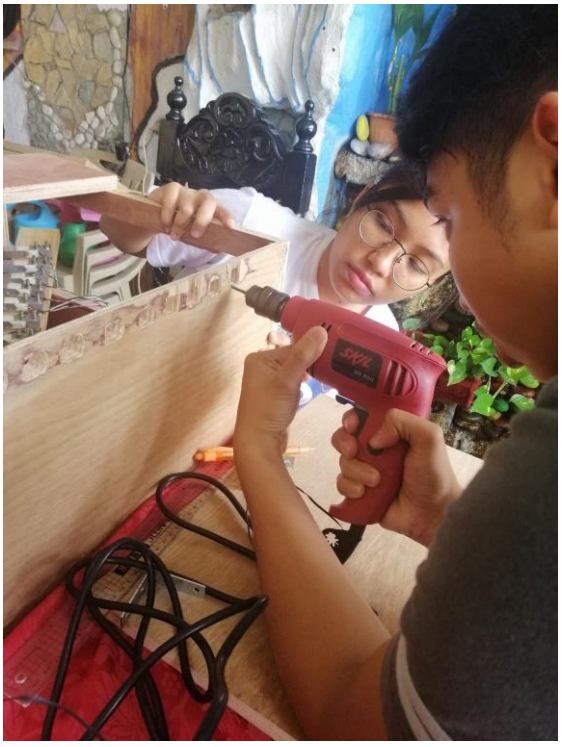


AndroidApplicationDevelopment

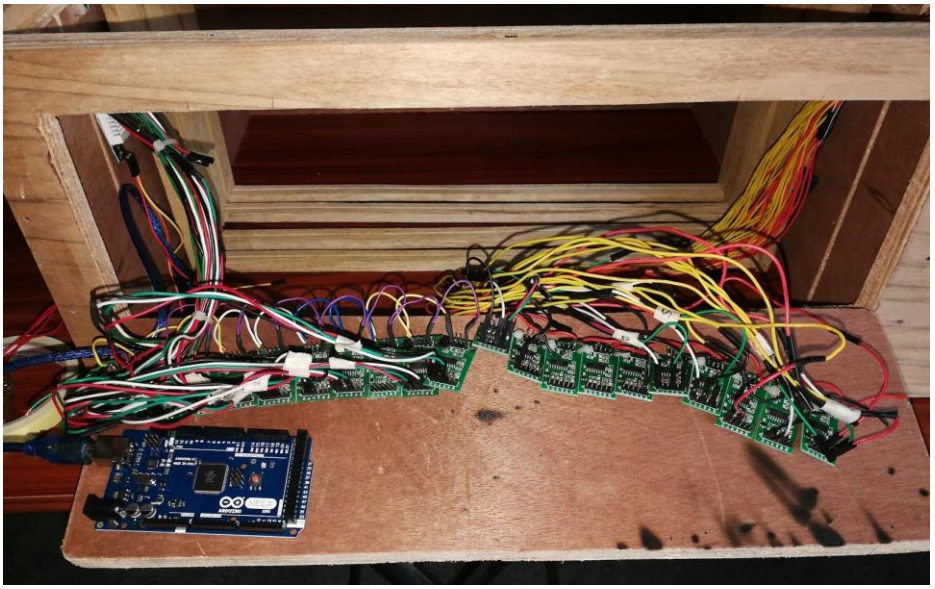




Prototype Construction



Wirings and Connections



Orientation in the bee farm

Deployment

Title Defense

Progress Defense

Pre-final Defense

APPENDIX I

GANTT CHART

Activities	2019								2020		
	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar
Design a smart beehive based on the existing Langstroth that would satisfy the system's functionality.											
Find and purchase the suitable temperature, humidity, and weight sensor needed. Buy Raspberry Pi that will serve as the microcontroller of the system, GSM module for SMS notification and camera for image recognition system.											
Mount the sensors and camera to their respective place in the beehive.											
Connect the sensors to the											

General-Purpose Input/Output (GPIO) of the Raspberry Pi.											
Design a system that will process the data gathered from all the sensors used.											
Mount the sensors to their respective place in the beehive.											
Test whether There is an output obtained from the condition of the beehive.											
Examine the proper positioning of the camera to have an accurate detection of the bees entering the hive.											
Gather multiple images of actual bees with and without mites.											

Design an algorithm for the control system of the camera using Convolutional Neural Network											
Test the accuracy of the camera and the effectiveness of the mirror.											
Find a cloud server that can easily be accessed by the beekeeper.											
Develop a code that will notify the beekeeper through their mobile phone via GSM.											
Develop a code that will send the data gathered from the hives to the cloud.											
Enhance the mobile application to monitor the temperature, humidity, weight of the frames and											

number of infected bees inside the beehive.											
Perform a dry-run to assess if the data acquired are sent in the cloud and the notifications are received by the mobile phone.											
Launch the mobile application and compare whether the data stored in the cloud is the same as the data displayed in the mobile application.											
Design an automated treatment system on the cover of the beehive.											
Purchase the materials needed in the treatment system.											
Assemble the treatment											

system inside the cover of the beehive.											
Develop a code that will automate the system.											
Test and evaluate the efficiency of the system.											