

IOT CAMERA SYSTEM FOR MONITORING STRAWBERRY FIELDS

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Electrical Engineering

by

Simon Schoennauer

December 2020

© 2020
Simon Schoennauer
ALL RIGHTS RESERVED

COMMITTEE MEMBERSHIP

TITLE: IoT Camera system for Monitoring Strawberry Fields

AUTHOR: Simon Schoennauer

DATE SUBMITTED: December 2020

COMMITTEE CHAIR: Vladimir Prodanov, Ph.D.
Professor of Electrical Engineering

COMMITTEE MEMBER: Jane Zhang, Ph.D.
Professor of Electrical Engineering

COMMITTEE MEMBER: Benjamin Hawkins, Ph.D.
Assistant Professor of Electrical/Biomedical Engineering

ABSTRACT

IoT Camera system for Monitoring Strawberry Fields

Simon Schoennauer

A wireless imaging system for monitoring strawberry fields provides enough quality image data for computer vision algorithms to make meaningful yield predictions. This report contains a design for a wireless sensor network modified with mesh networking techniques to extend coverage range and a solar energy harvesting system to improve sensor node lifetime. A two hop system with six nodes is implemented in a laboratory environment validating the communication systems integrity over an 800' range. Moving from a primary battery system to solar energy harvesting increases the module lifetime indefinitely.

ACKNOWLEDGMENTS

A special thanks to my advisor Dr. Prodanov, and my committee members Dr. Zhang and Dr. Hawkins for providing guidance and leadership in spite of the challenges associated with COVID19.

Thanks to my mom, Brenda Schoennauer for her compassion, love, and support. Thank you to my Dad, Larry Schoennauer along with all of the other engineers in our family who are my inspiration.

Thank you to my siblings Eli and Emily, my roommates Zac and Mae, and my significant other, Katie.

Thank you to my friends and classmates Logan, Jessica, Maria, Eddy, Nick, Riley, and Carson.

Finally, thank you to all of the teachers, coaches, and mentors that have helped me reach where I'm at today.

TABLE OF CONTENTS

	Page
LIST OF TABLES	viii
LIST OF FIGURES	ix
CHAPTER	
1 Introduction	1
1.1 Application	1
1.2 Mesh Networking	2
1.3 Solar Energy Harvesting	3
1.4 Organization	4
2 Literature Review	5
2.1 Self Powered Environmental Sensors	5
2.1.1 System Overview	5
2.1.2 Comparative Discussion	7
2.2 Low-Cost IoT Remote Sensor Mesh Orchard Monitoring	8
2.2.1 System Summary	8
2.2.2 Comparative Discussion	10
2.3 Routing for Energy Sensitive Mesh Networks	10
2.3.1 System Summary	10
2.3.2 Comparative Discussion	12
3 Design	13
3.1 Image Capture System	13
3.2 Proposed Communication System	14
3.2.1 Hardware	15

3.2.2	Communication Protocols	16
3.2.3	Firmware Platforms	21
3.2.4	System Modules	22
3.2.5	Placement and Data-path	23
3.2.6	Firmware	24
3.3	Power System	26
3.3.1	Energy Needs	27
3.3.2	Solar Cells	27
3.3.3	Super-capacitors	28
3.3.4	Schematic	29
4	Performance	31
4.1	Communication System Performance	31
4.1.1	Communication System Proof of Concept	31
4.1.2	Range	32
4.2	Performance of Power System	35
4.2.1	Power Evaluation	35
4.2.2	Supercapacitor Characterization	35
4.2.3	Solar Cell Characterization	38
4.2.4	Sensor Node Recovery	41
4.2.5	Sensor Node Size	41
5	Conclusion	43
5.1	Results	43
5.2	Future Work	43
	BIBLIOGRAPHY	45

LIST OF TABLES

Table		Page
2.1	Environmental Sensor Node Energy Requirements	6
3.1	802.11b and BLE Comparison	17
4.1	Sensor Node Hardware Size	42

LIST OF FIGURES

Figure		Page
1.1	Mesh Network Example	2
2.1	Environmental Wireless Sensor Node	6
2.2	Sensor Node Supercapacitor Voltage	7
2.3	Sensor Node Supercapacitor Voltage Vs. Battery	8
2.4	Sensor and Relay Node Prototype	9
2.5	Example Simulated Mesh Network	11
2.6	Simulated energy efficiency results	12
3.1	OV2640 Camera Module	13
3.2	Arducam ESP32S UNO Development Board	15
3.3	ESP32S Ai Thinker Module	16
3.4	BLE High Level Topology	18
3.5	BLE GATT Structure	18
3.6	Example AP and station	19
3.7	Socket Connection Procedure	20
3.8	Summary of System Modules	22
3.9	Communication System Diagram	23
3.10	Physical System Diagram	24
3.11	Trunk Firmware Flow Diagram	25
3.12	Branch Firmware Flow Diagram	25
3.13	Leaf Firmware Flow Diagram	26
3.14	Solar Cell Comparison	28

3.15	AVX 7.5F/5.4V Supercapacitor	29
3.16	Power System Schematic	29
4.1	Communication System Test Setup	32
4.2	Range Test Setup Results	33
4.3	Close image transmission	33
4.4	Mid-range image transmission	34
4.5	Max-range image transmission	34
4.6	Leaf Node Current Screen Capture	36
4.7	Branch Node Current Screen Capture	36
4.8	MPT 75x75mm Solar Cell	39
4.9	SUNYIMA 68x37mm Solar Cell	39

Chapter 1

INTRODUCTION

Sophisticated methods of statistical modeling demand large amounts of quality data. Wireless sensor networks (WSN) are used for data collection, processing, and analysis to help fill the growing demand for information. Today, WSN are common with the emergence of the internet of things (IoT). For agricultural applications, sensor nodes are typically battery powered and designed to operate for several years [1] [2]. Some of the challenges associated with designing IoT networks include signal range, lifetime, and cost. Mesh networking provides a solution that extends signal range but is difficult to implement as a low energy network, in this work a simplified version of mesh networking is presented as an improvement to the range performance over the standard point-to-point system. A solar energy harvesting system improves the systems maximum operation time and removes the need to periodically service battery powered sensor nodes.

1.1 Application

Creating a system to capture daily images of strawberry plants in a field and transporting this data to an accessible location for processing is the goal of this project. The sensor nodes need to cover a wide range to get a meaningful average representing the entire strawberry field. Data is taken over the entire growth cycle of the strawberry plants to form complete models, to achieve this repeatedly, sensor nodes are designed to operate over many years.

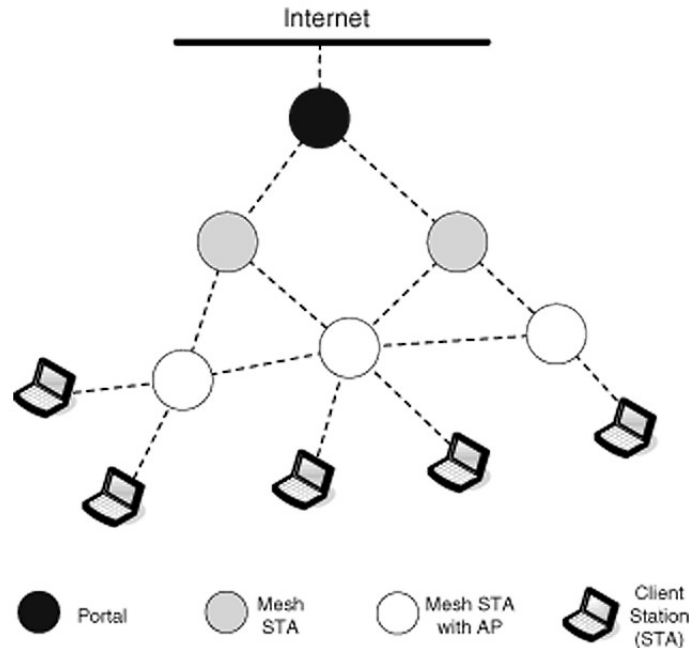


Figure 1.1: Mesh Network Example [4]

Jake Wahl did research into existing sensor networks and presented a strawberry imaging system in spring of 2020 that reflects standard IoT characteristics [3]. Wahl made a single point to point WSN using bluetooth low energy (BLE) as the communication protocol and implemented this on a battery powered Arducam ESP32S using the Arduino IDE and framework.

1.2 Mesh Networking

True mesh networks are self-provisioning, self-forming, and self-healing wireless communication systems where each node is capable of relaying information to any adjacent nodes. Figure 1.1 shows how station nodes, access point nodes, and hybrid nodes form a mesh network connecting each node to a portal or the root node [4]. The benefits of mesh networks include additional range with the same number of devices, meaning more range for less cost. However, mesh networks are complex and challenging to implement they typically consume more energy than a point-to-point system because

of the extra receptions and re-transmissions required for the edge nodes to communicate. Mesh networks automatically configure the data-path and handle dropped and new connections, this requires a provisioning algorithm that consumes energy during operation. As seen in [5], a wireless mesh network (WMN) topology for IoT applications that includes battery life into the cost function governing the algorithm for node provisioning attempts to mitigate the high energy demands of a mesh network. However, this topology is only an optimization of the fundamentally high energy demands of a mesh network. Presented in this report is an alternative to a mesh network. By taking advantage of the parameters specific to our application we achieve a WSN that has the desirable properties of a WMN eg. long range, low cost, without the complexity and downsides of a true WMN. For our application, the most relevant feature of mesh networking is the ability to relay messages between nodes, this feature improves the coverage area indefinitely. To achieve this with our system, a predetermined data path and schedule is defined where groups of sensor nodes wake up from deepsleep and form a bucket brigade extending their range to access the root node. Since sensor node location and data are deterministic, the optimum datapath is static and self-provisioning is an unnecessary energy expense.

1.3 Solar Energy Harvesting

There are many energy harvesting techniques [6] [7] [8], it just so happens that solar energy harvesting works best for our strawberry field application. This is because the sensor is outdoors and only needs to operate during daylight hours when the camera can see the strawberry plant. Also, solar cells are widely available and present a low complexity, low cost solution to improving the lifetime of the device. However, solar cells produce inconsistent and limited output power. Typically to operate as a power system solar cells use external hardware to store energy, buffer power, regulate output

voltage, and protect circuitry [9] [7] [10]. Our solar energy harvesting system uses a supercapacitor for energy storage and power buffering. Similarly in [5], a WSN is powered with a solar energy system using a supercapacitor for energy storage and power buffering. The authors show that for low energy applications supercapacitors function as longer lasting alternatives to rechargeable batteries because they endure millions of charge cycles.

1.4 Organization

The rest of the report is organized as follows, Chapter 2 outlines the system design and describes design decisions made for each section of the system and the trade offs associated with them. Chapter 3 describes the characterization of the implemented system, detailing the tests and results describing the implemented systems performance. Chapter 4 consolidates results and reviews the state of the project in a conclusion section, and also includes a section describing opportunities for future work.

Chapter 2

LITERATURE REVIEW

This chapter outlines three systems reported in the recent literature. These three works are chosen out of many from the open literature because either the application or design constraints resemble that of the proposed system. After a brief description of the system, a discussion regarding the relevance of the work is presented.

2.1 Self Powered Environmental Sensors

2.1.1 System Overview

The authors in [7] present a self powered wireless environmental sensor network. Their wireless sensor network collects outdoor environmental data using temperature, humidity, CO , and CO_2 sensors. This data is collected using a XBee wireless transceiver module (XBee-Pro 900HP). The sensor nodes employ a solar energy harvesting system to power the device. Separate sensor nodes are implemented using supercapacitors and rechargeable batteries for energy storage to verify that supercapacitors are a suitable alternative to rechargeable batteries.

Figure 2.1 shows the authors' sensor node hardware. The sensor nodes power system is comprised of a 55x67.5mm solar cell, two 100F/2.7V supercapacitors, a DC-DC converter (LTC3105), and a low dropout voltage regulator. Some nodes use a 2600mAh (ICR18650-26F) battery is used in place of the supercapacitors. The power system supports all of the environmental sensors, an ATmega328p MCU, and the XBee RF module.

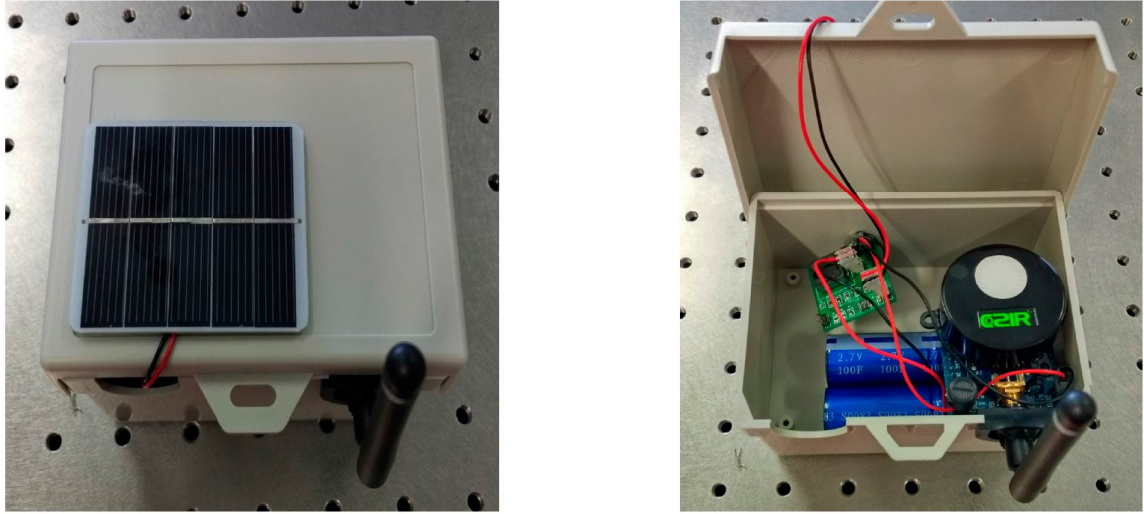


Figure 2.1: Environmental Wireless Sensor Node [7]

Table 2.1: Environmental Sensor Node Energy Requirements [7]

Stage	Mode	Current (mA)	Energy (mJ)
1	Idle(0-10s, 10.1-10.5s)	65.8	2258.26
2	Transmit (10-10.1s)	156.2	51.55
3	Idle(10.5-15s)	35.1	521.24
4	Sleep(15-16s)	4.3	14.19
5	Sleep(16 s+)	0.2	781.44

The sensor nodes operate continuously on a 1.25% duty cycle, waking for 15 seconds every 20 minutes to take and transmit sensor data. Table 2.1 shows the current and energy breakdown of the sensor nodes. In figure 2.2 the 55x67.5mm solar cell provides a maximum of 300mW, charging the 50F supercapacitor bank until night comes. The sensor node reduces the supercapacitor banks voltage in a slanted staircase fashion, due to the alternating sleep and operating modes, to around 3.6V during the night then quickly recharges to full capacity at daybreak. Figure 2.3 compares the sensor nodes using supercapacitors to the sensor nodes using rechargeable battery's. Although both energy storage devices effectively power the sensor node through the night, the batteries have more energy capacity. Without any power from the solar cell the fully charged supercapacitor bank is only capable of supporting the device

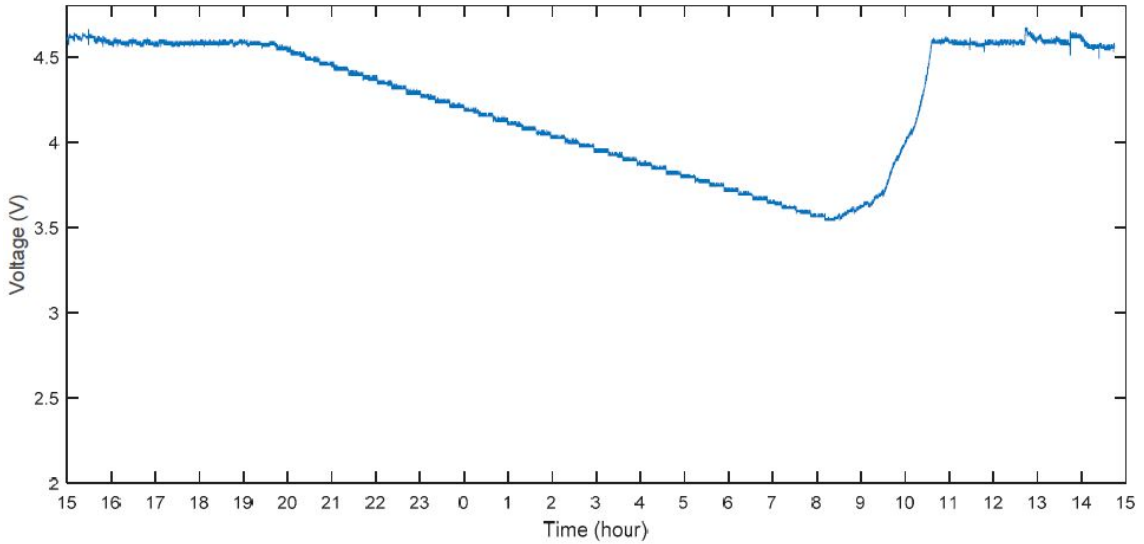


Figure 2.2: Sensor Node Supercapacitor Voltage [7]

for 12 hours. With 2600mAh the battery nodes could survive for many days without power, furthermore the excess energy generated by the solar cell during peak daylight hours is capable of repaying energy debt generated by prolonged no-light situations.

2.1.2 Comparative Discussion

The authors' environmental sensor network faces many of the same challenges that we face in our project (e.g. low energy, outdoors).

Although their application requires a 1.25% duty cycle where ours is approximately 0.025% their reduced data payload brings the net energy requirements of our projects closer together. The authors implemented solar energy harvesting system performs similarly to ours. From their comparative analysis using supercapacitors and batteries they concluded that supercapacitors are a suitable alternative device with a longer overall lifetime. However, the superior energy density of rechargeable batteries allows them to buffer more energy making the power system more resilient to poor weather conditions. A robust power system should allow for a 2-10 times factor of safety for the

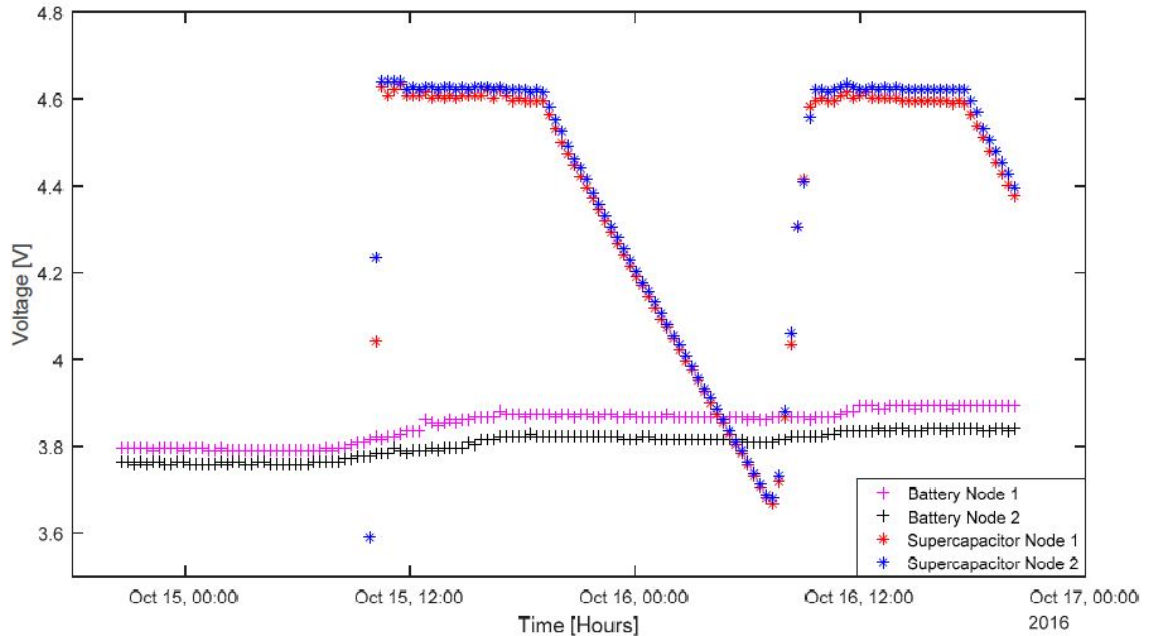


Figure 2.3: Sensor Node Supercapacitor Voltage Vs. Battery [7]

energy storage requirement. This could be difficult to achieve using supercapacitors because of their low energy density. The type of wireless network used was not specified by the authors, we've assumed the topology is not a mesh because the network is referred to as a WSN as opposed to a WMN. Although the network is not a mesh the authors have little coverage range limitation because the XBee RF modules have a specified transmit range of 28 miles. However, the XBee wireless modules have a max datarate of 200kbps which is reduced when being used for long range applications.

2.2 Low-Cost IoT Remote Sensor Mesh Orchard Monitoring

2.2.1 System Summary

The authors in [9] present a self powered wireless network for monitoring orchard fields. The sensor nodes contain internal and external temperature and humidity

sensors with a gas and moisture sensor. They use an ATmega328P MCU and a E32433k30d LoRa transmitter. Figure 2.4 shows the physical prototypes for their sensor and relay nodes. The function of the relay node is to connect the LoRaWAN signal from the sensor node to a WiFi access point. Although the design calls for a mesh network of sensor nodes, the author did not specify if this was implemented. Their results are derived from one sensor node implying that for the prototype system a mesh network has not been implemented.

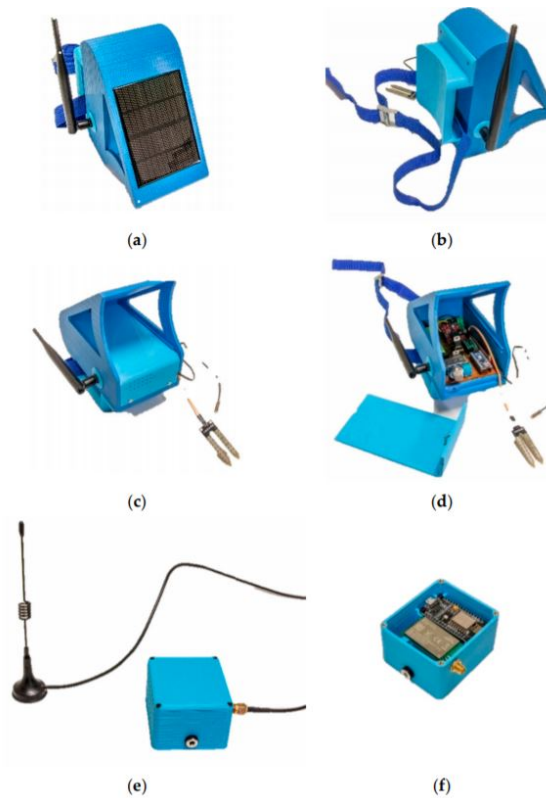


Figure 2.4: Sensor and Relay Node Prototype (a,b,c,d are the sensor node e,f are the relay node) [9]

The sensor nodes are powered with a 1.5W 115x85mm solar panel using two separate buck boost converters to charge the 2600mAh battery and to power the device. The LoRaWAN wireless communication protocol is designed for low power, low data rate, long range communication. The maximum range reported by the author for this

protocol is 15km, the maximum data rate is 50kbps this reduces, as the maximum range reached, to 290bps.

2.2.2 Comparative Discussion

Although the authors report a design for a low cost LoRaWAN mesh network, the author did not specify the network topology of their prototype system. Furthermore the LoRaWAN communication protocol only supports mesh networks without energy constrained relay nodes [11]. Monitoring an olive orchard is comparable to our application; the authors conclude that a mesh network can be used to improve coverage range. Regardless of their single node implementation, their mesh network design considerations support our predictions. The authors provide little analysis of their power systems performance, with a 1.5W solar cell and a 2600mAh battery energy issues are unlikely considering the low power requirement of their LoRa transceiver and their data payload. A key difference between our applications is the data payload and duty cycle. The sensor data recorded in their prototype system is sent every 15 seconds. The data size is not reported however it is unlikely the information recorded in their application requires much data because of the types of sensors being used.

2.3 Routing for Energy Sensitive Mesh Networks

2.3.1 System Summary

The authors in [5] develop and simulate a design a long range WMN for energy constrained environments. They claim that the mesh networks are well suited for outdoor, agricultural applications because of clean communication environment compared to rural areas. However, outdoor agricultural applications typically include

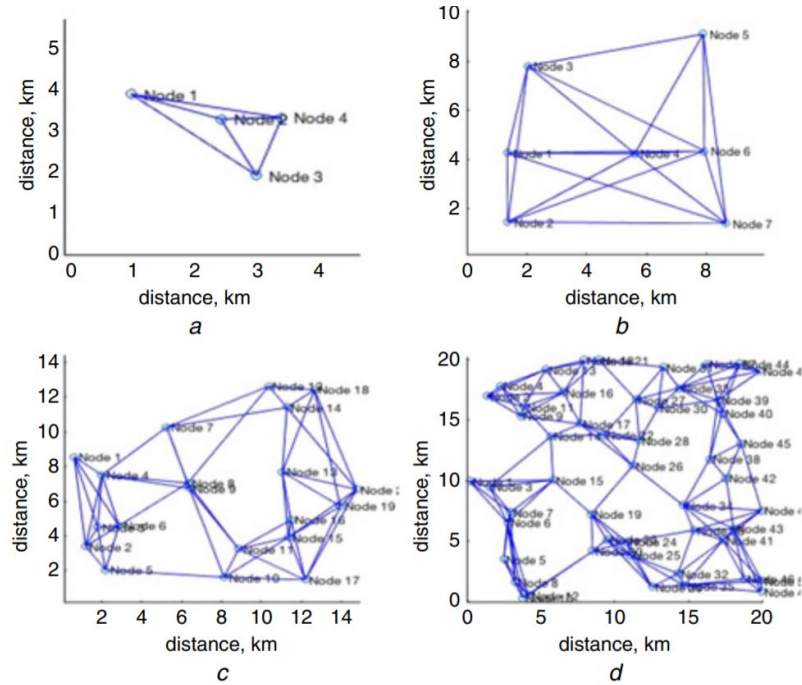


Figure 2.5: Example Simulated Mesh Network Coverage Area (a=4node, b=7node, c=20node, d=50node) [5]

energy constrained sensor nodes as a solution the authors present their work on a reinforcement learning algorithm that optimizes sensor node battery life. Figure 2.5 shows four example networks generated from a randomly placed model. Each sensor node is initialized with 15 Wh of energy and is periodically recharged in a 1000s charge cycle. The transmission bandwidth and energy requirements are modeled after LoRa transceivers.

Figure 2.6 shows the results from their simulations where the authors reinforcement learning algorithm is compared to the ideal centralised SPF and to random routing. The authors show that a RL algorithm approaches the ideal centralized SPF as the size of the sensor network increases.

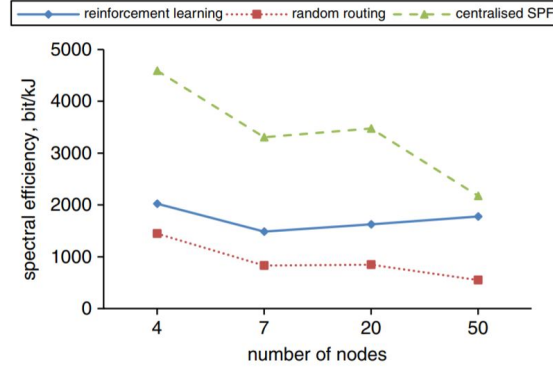


Figure 2.6: Simulated energy efficiency results [5]

2.3.2 Comparative Discussion

The work in [5] addresses the issues associated with mesh network scalability. Their results show that as a mesh network increases in size it becomes less reliable and less energy efficient. To mitigate this their reinforcement learning routing algorithm takes into account sensor node battery life, optimizing energy efficiency. Although improving the energy efficiency of a mesh network is one of our projects main goals, the methods used in this work alone are not adequate for our application. The simulated results assume a constant recharge cycle every 1000 seconds which is far from our sensor node lifetime goals. Also no accommodations are made for sleeping sensor nodes in their simulation, allowing sensor nodes to be in sleep mode is a critical part of low energy wireless sensor network design.

Chapter 3

DESIGN

3.1 Image Capture System

Sensor nodes capture images and report this data back to the base-station. Although critical, this section of the system design is kept relatively short due to the availability of an effective low cost solution. The application demands high resolution images in an outdoors environment during the day; the OV2640 camera module achieves this.

The OV2640 camera module, shown in figure 3.1, is the camera module built into the Arducam ESP32 UNO development board. The OV2640 is a small form factor 2 Megapixel camera, with configurable white-balance and saturation settings. The OV2640 communicates using a standard serial camera control bus (SCCB) and has a built in image compression engine that reduces the amount of transmitted data per image [13].

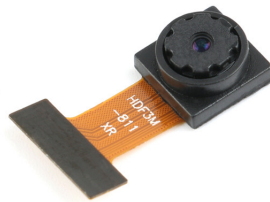


Figure 3.1: OV2640 Camera Module [12]

Due to the OV2640s compression engine, jpeg files vary in size between 100kB-250kB depending on image content. From a communication standpoint, handling this amount of data requires either a large amount of dynamic memory or a static memory buffering algorithm.

Although the compression techniques used in the creation of .jpeg files is not lossless. The compression algorithm preserves most of the information in the image and significantly reduces the data payload. The image recognition algorithm is highly statistical in nature, the work done by Fitter in [14] indicates that compressed image quality is sufficient for the application.

3.2 Proposed Communication System

The communication system design below is an alternative to a true WMN, where some of the benefits like cost, efficiency, and range are achieved without the undesirable complexity and energy requirements of a traditional mesh network. The Arducam ESP32S development board has WiFi and bluetooth low energy (BLE) capability and is used to implement the communication and imaging systems. The implemented design uses 802.11b, a version of WiFi, as the low level communication protocol [15]. The standard station and access point (ap) topology along with transmission control protocol (TCP) are used to implement the high level system design. The communication system schedules and conducts data transactions between trunk, branch, and leaf nodes shown in figure 3.8. The proposed system has one trunk node that acts as the base-station where all the image data ends up. Then, there are parallel groups of branch and leaf node pairs, with one leaf per branch, where each branch acts as a conduit from its leaf to the trunk. These branch leaf pairs operate on a schedule to wake up, form connections, transport data, and to synchronise their clocks.

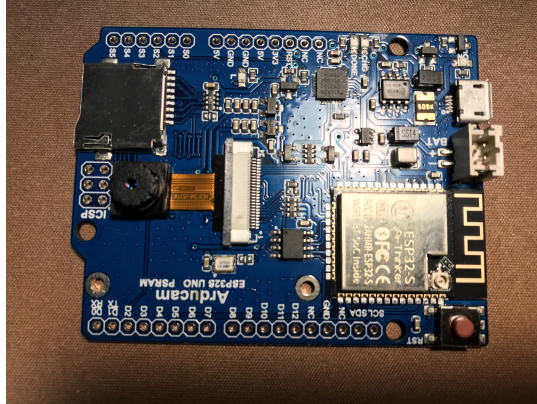


Figure 3.2: Arducam ESP32S UNO Development Board

3.2.1 Hardware

The Arducam ESP32 UNO development board shown in figure 3.2 contains all the communication and imaging hardware used by the sensor nodes in this design. The notable components on the Arducam ESP32 development board include the OV2640 camera module, the ESP32S module, and basic power supply circuitry. The Arducam ESP32 UNO board was used in Wahls point-to-point data system. Despite poor documentation (no schematic), it has proven to be an adequate development platform for this application.

Found on the Arducam ESP32S UNO board is the ESP32S module, shown in figure 3.3. The ESP32S module contains the ESP32 SoC and additional transceiver hardware including the 2.4GHz meander line inverted F PCB antenna. The relevant features of the ESP32S module are as follows: Wifi and bluetooth functionality, internal SRAM with provision for additional external SRAM, a low power coprocessor, SPI and I2C peripherals, and an SD card interface.

In addition to handling on-board communications to hardware peripherals and wireless communication between camera nodes, the ESP32S buffers large image files using 4MB of external PSRAM. An onboard USB to UART converter from SI-LABS inter-

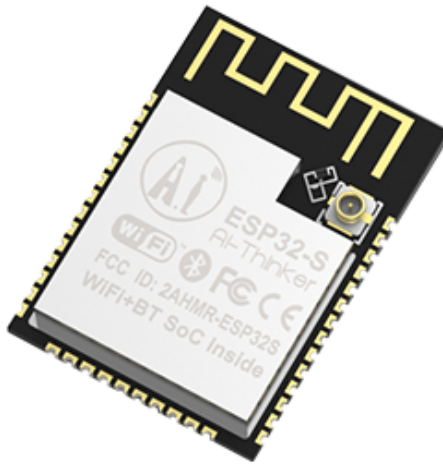


Figure 3.3: ESP32S Ai Thinker Module [16]

faces programming capability to the module the from a PC. Since camera nodes spend a majority of their time asleep, the low power coprocessor allows the ESP32S to enter a deep-sleep mode where the module consumes $5\mu\text{A}$. The ESP32S module supports a variety of wireless communication protocols in the 2.4GHz ISM band, including BLE and 802.11b or Wifi.

3.2.2 Communication Protocols

Among the wireless communication protocols the ESP32S supports, the 2 most practical are 802.11b and BLE. Table 3.1 shows a comparison between the two wireless communication protocols.

Sending a few images a day (1-2) demands low data rates on average. However, outgoing data arrives in large chunks (100-250k bytes) all at once. Although the

Table 3.1: 802.11b and BLE Comparison

Protocol	Power Consumption	Baud-rate	MTU	Range
802.11b	200mA	11MBps	2312	400ft
BLE	200mA	1MBps	513	400ft

minimum data rate given the amount of time between image captures is extremely low, slower data rates require longer transmission times. This means longer operating sessions where the high power transceiver peripherals are active. To keep the energy demands low, higher data rates are preferred to reduce the amount of time the module spends in operation mode. BLE is used in Wahls system but we've since moved to 802.11b to achieve a higher throughput. In the following section we discuss BLE to illustrate the negative effects sending images has on the protocols energy efficiency.

Shown in figure 3.4, there are BLE central (Phone, Tablet, Computer) and peripheral devices. Generally speaking, peripherals hold data and centrals access the data [17]. To initiate communication, centrals take on the observer role and scan for any peripherals that are advertising. After a scan is complete, a connection to a specific peripheral is made [18].

The exchange of data from central to peripheral is governed by the Generic Attribute Protocol (GATT) shown in figure 3.5. Within a profile for a peripheral device, there are services and characteristics which contain data [17]. Typically a peripheral device updates its characteristic values to reflect sensor readings, and the central device reads a peripherals characteristic as needed. A peripheral device can also notify a central device that its characteristic value has been updated.

Similar to BLEs centrals and peripherals, WiFi has stations and access points (AP). For example, in a simple 2 node network where a phone connects to your home WiFi, the phone is a station and the router projecting the wifi signal is the AP. Figure 3.6

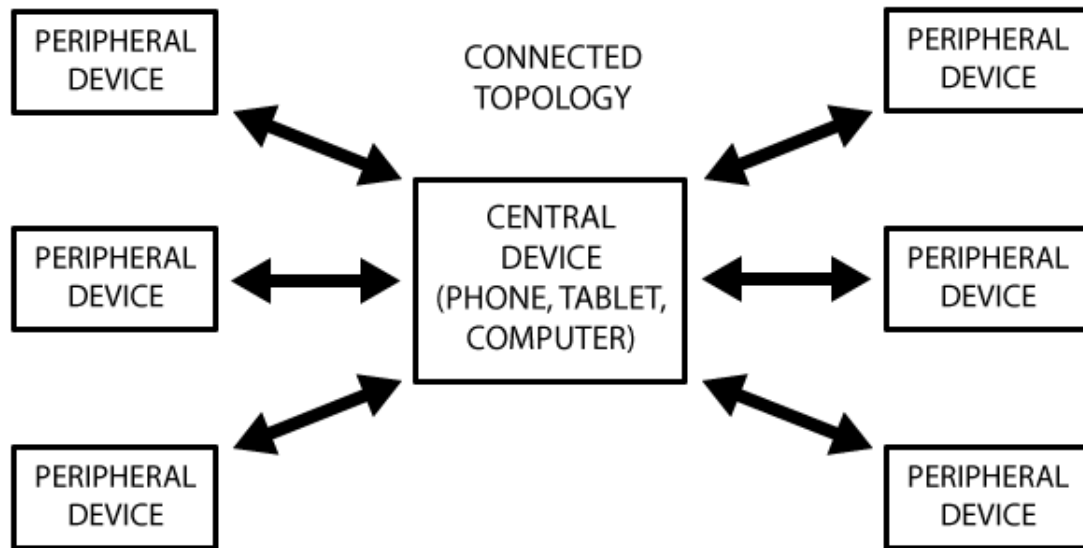


Figure 3.4: BLE High Level Topology [17]

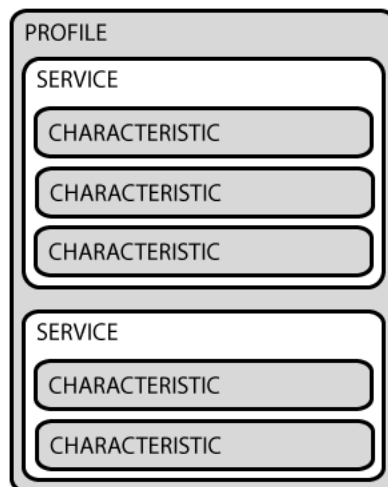


Figure 3.5: BLE GATT Structure [17]

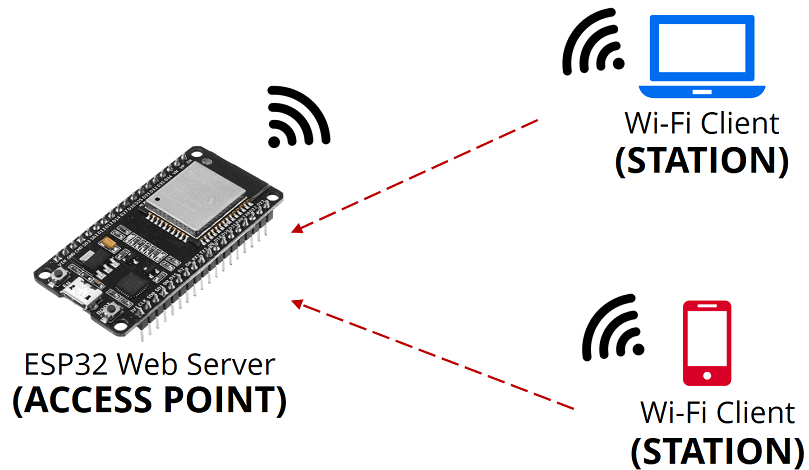


Figure 3.6: Example AP and station [19]

shows how an ESP32 can act as an access point. The connection procedure for the ESP32 AP is exactly the same as a phone to a home WiFi network.

ESP32S's act as stations or APs. To set up an AP with the ESP32S, a service set identifier (ssid) and password are defined and the AP is made active. Then the station node provides the ssid and password to connect within the APs range [20]. Once a WiFi connection is formed, internet protocol (IP) takes over and transmission control protocol (TCP) or similar interfaces are used to transmit data. Figure 3.7 below shows the formation of a TCP connection using sockets.

Typically TCP connections are one to one between a server and client. The connection can be thought of as a virtual wire where the connection points are called sockets. The server binds to a local ip address and port number, then a client connects to the server socket using this information [18].

Wahl uses BLE in his point-to-point system, and reports a maximum operating time of 17.5s, 10s of this time derives from the image transmit process. The reason the image transmit time is so long is because of the limited maximum transmission unit (MTU), table 3.1 shows the MTU for the BLE protocol implemented in both the

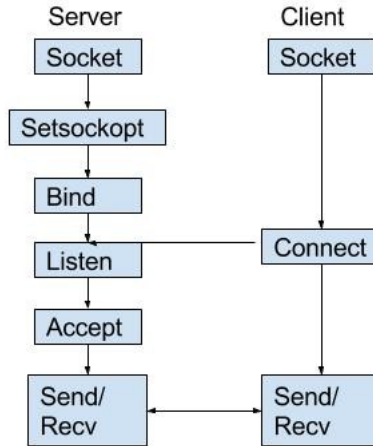


Figure 3.7: Socket Connection Procedure
[21]

Micropython and Arduino framework to be 23bytes. With a 23 Byte MTU, BLEs throughput is 0.226Mbps. This is a small portion of BLEs 1 Mbps baudrate [22]. Increasing the MTU increases the throughput until the MTU is greater than the size of data being sent per event. In our case, image files with sizes ranging from 100-250kB are sent, meaning the larger the MTU the better. BLEs connection and data transfer process manifests in the firmware entirely as interrupts. This is great for minimising system idle time but the added complexity at a high level creates reliability issues. BLE lends itself to transmitting small amounts of data on a short time interval, however our application requires large amounts of data on an extremely long time interval. Although BLE consumes less average power than WiFi, it uses more energy because of its limited throughput caused by the small MTU. WiFi takes less time to transmit images than BLE, this reduces the amount of time the sensor node spends in a high power state and ultimately reduces the sensor nodes net energy requirement. In addition to WiFi outperforming BLE in total energy consumption, its higher level networking protocol, TCP, simplifies the design while GATT includes many layers that are unnecessary for the application.

3.2.3 Firmware Platforms

To construct a high level communication system, high level firmware tools are needed. The two development languages considered are Micropython and C++. Micropython is a lean version of Python 3 that is meant to run on microcontrollers, and C++ is a higher level version of the standard C programming language. One key difference between these languages is that micropython is an interpreted language and C++ is a compiled language. Interpreted languages have a built in compiler on the microcontroller, while compiled languages need a separate compiler.

Because it is a compiled language, C++ runs faster than Micropython. C++ itself does not limit the project, however the availability and quality of the low level firmware drivers implemented in C++ do. The Arduino IDE which is a development environment and compiler for C++ has poor documentation and incorporates broken libraries into their official databases. Furthermore the Arduino IDE shows a simplified view of the compiling process which makes debugging and altering low level firmware difficult.

Micropython's built in compiler allows for the read evaluate print loop (REPL), which manifests as a console on one of the UART channels of the microcontroller. The REPL allows us to write and test code on the microcontroller in real time, which is a critical debugging tool, especially given the absence of a hardware debugger on the Arducam ESP32S Uno board. Micropython's documentation requirements for their official libraries are extensive and well defined. All of the low level hardware drivers required for this project are supported.

Separate versions of micropython exist to reflect different hardware ports, a custom subset of micropython for the ESP32S port provided by [23] includes a built in library

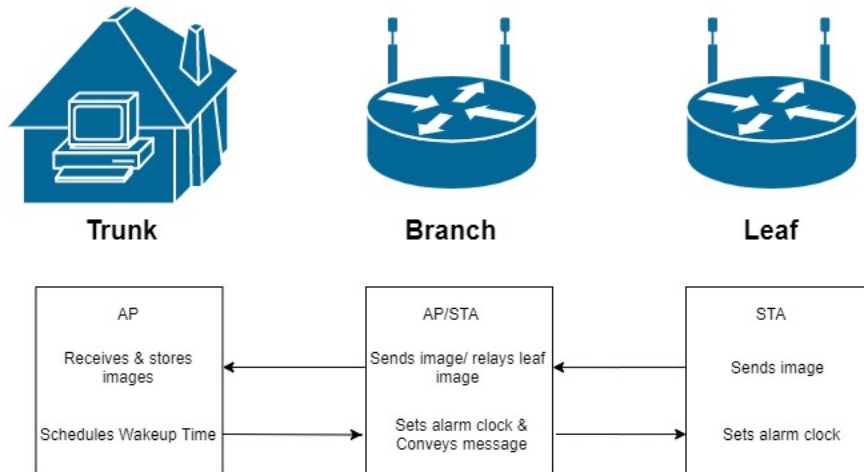


Figure 3.8: Trunk, Branch, Leaf Module Functional Diagram

giving high level functionality to the OV2640 module. WiFi and BLE networking libraries are included in micropythons official release.

3.2.4 System Modules

The trunk, branch, and leaf modules make up the communication system. Figure 3.8 depicts the basic functions for each module in addition to the WiFi node type (ie. station or AP).

A PC connected to a WiFi router acts as the sole trunk node of the network. A python script running on this PC manages the trunk’s functions. The connected WiFi router facilitates wireless connection to the many branch nodes. The trunk node synchronizes alarm clocks, receives image data, and stores the received data locally on the PC. The same hardware comprises both the branch and leaf nodes. This hardware includes the Arducam ESP32S development board and power system electronics. Branch nodes concurrently operate as an AP and a station to facilitate a simultaneous connection to the truck and leaf nodes. Branch nodes capture images

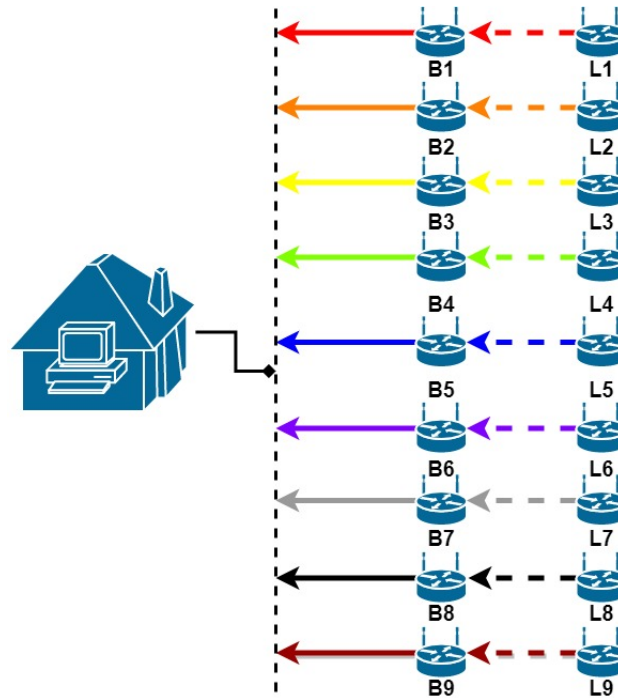


Figure 3.9: Communication System Diagram

and communicate directly to trunk. They also relay information between the trunk and leaf nodes. Leaf nodes capture images and send and receive data from the branch.

3.2.5 Placement and Data-path

Figure 3.9 shows the communication systems perspective of data flow, a simplified mesh network with predetermined data paths and numbers of hops. Each branch leaf pair wakes up one at a time to connect to the trunk.

Figure 3.10 below shows an example of the physical layout of the camera nodes where each branch connects to its corresponding leaf and the basestation. In spite of its simplicity this communication system achieves a large coverage area because of the maximum node to node transmission range. The physical placement in figure 3.10 ports directly to the simplified system in figure 3.9 but resembles the desired grid where node spacing reflects the needs of the application.

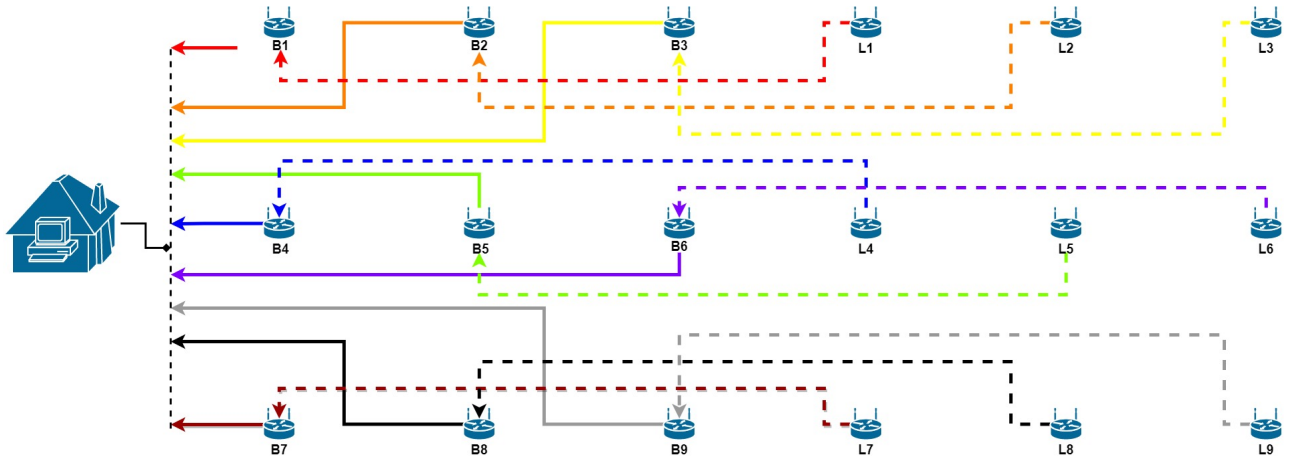


Figure 3.10: Physical System Diagram

3.2.6 Firmware

Figure 3.11 shows the flow diagram depicting the high level operation of the trunk node. The trunk opens a socket that awaits a connection from a branch through the WiFi routers local network, when the trunk receives a connection from a branch it initiates the image reception process for both the branch and leaf images. After collecting both images, the trunk node uses internet time to calculate the updated scheduling information and sends it to the branch and leaf nodes.

Figure 3.12 shows the high level firmware flow diagram for the branch node. The branch node wakes up on a schedule determined by the trunk during its previous waking session. The branch wakes at a scheduled time, configures itself as a station and connects to the trunk node. Then the branch sets up an AP in addition to the station and waits for its leaf to connect. After the leaf connects to the branch, the branch sends its image data to the trunk, relays the leaf image data, receives schedule data, and relays the schedule information to the leaf. Once finished the branch goes into deepsleep mode until woken at the time set by the trunk.

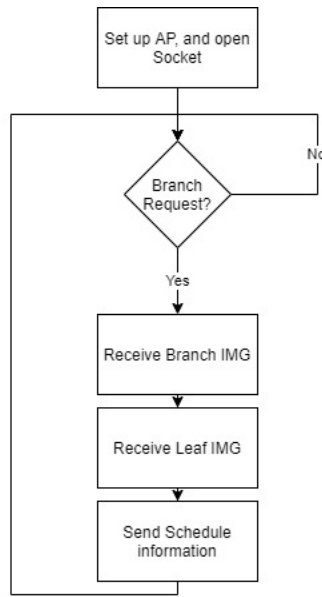


Figure 3.11: Trunk Firmware Flow Diagram

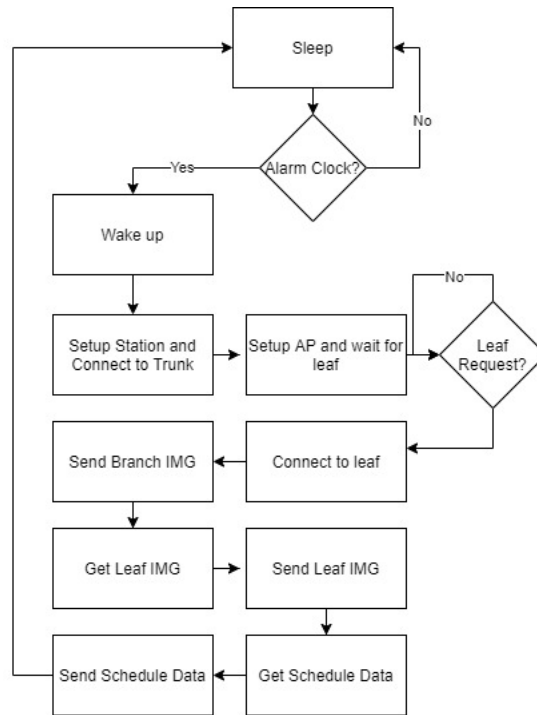


Figure 3.12: Branch Firmware Flow Diagram

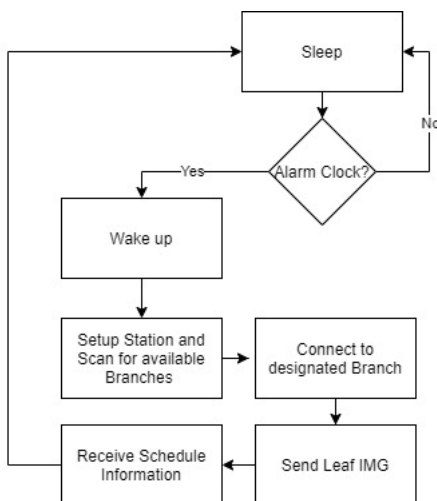


Figure 3.13: Leaf Firmware Flow Diagram

Figure 3.13 shows the firmware flow diagram for the leaf module. Similarly to the branch, the leaf wakes up at its scheduled time, and configures itself as a station. The leaf node needs to scan for available branches to make sure its branch is available to connect to and to prevent forming a connection to the wrong branch. After sending its image data and receiving scheduling information from the branch the leaf node goes into deep sleep mode.

3.3 Power System

A solar energy harvesting system improves the lifetime of sensor nodes compared to a standard primary battery system. The outdoors application lends itself to solar energy harvesting because the image capturing function already demands good lighting. Energy harvesting removes the need for a large energy bank, the rechargeable batteries lifetime then becomes the limiting factor for the systems lifetime. Supercapacitors have lower energy densities than batteries and much higher leakage currents but have long lifespans and low source impedance [7]. The power system uses so-

lar energy harvesting and supercapacitor energy storage to extend the sensor nodes lifetime indefinitely.

3.3.1 Energy Needs

Sensor nodes spend the majority of their time in a deepsleep mode, waking up briefly during the day to capture image data and complete communication tasks. During daylight hours sensor nodes charge while in sleep mode and discharge while in operating mode. At night, sensor nodes slowly discharge from the devices deepsleep current and the leakage current of the supercapacitor. The average operating power, average solar power, and operation time define the energy requirements during sensor node operation. The supercapacitor leakage, deepsleep power consumption, and no light duration determine the energy requirements of the device in deepsleep mode. Furthermore, the average solar power and deepsleep power determine the devices charge time. The power system must sustain the sensor node throughout an operation session and overnight in deepsleep mode. Also it needs to completely recharge before operating sessions, and end of day.

3.3.2 Solar Cells

The application involves taking images during the day outdoors; these conditions are ideal for solar cells. Solar cells eliminate the need for battery recharging or replacement. Due to the systems low energy requirements, solar cells fit for this application are commonly available and inexpensive. A solar cells open circuit voltage and short circuit current primitively characterize the device and define the maximum power point. The solar cells maximum power point matches the average load voltage to optimize power efficiency. Figure 3.14 shows the two types of solar cells considered

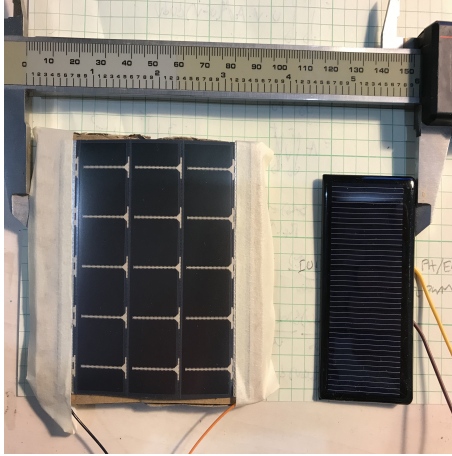


Figure 3.14: Solar Cell Comparison(left = MPT, Right = Sunyima)

for this project. On the left is the MPT solar cell, it has a short circuit current approximately equal to 60mA, and an open circuit voltage is around 5.5V. Similarly the Sunyima solar cell on the right has an short circuit current around 60mA, but a higher open circuit voltage around 6V.

3.3.3 Super-capacitors

Supercapacitors are polarized capacitors with a large enough capacitance to operate as energy storage devices for low energy density applications. The downsides of supercapacitors relative to batteries are higher leakage currents and lower energy densities. Advantages include longer lifetimes, and lower source impedance [10]. Typically supercapacitors have low voltage ratings, connecting modules in series gets around this at the expense of the net capacitance. The series connected modules used in this project shown in figure 3.15 are 7.5F supercapacitors rated for 5.4V.

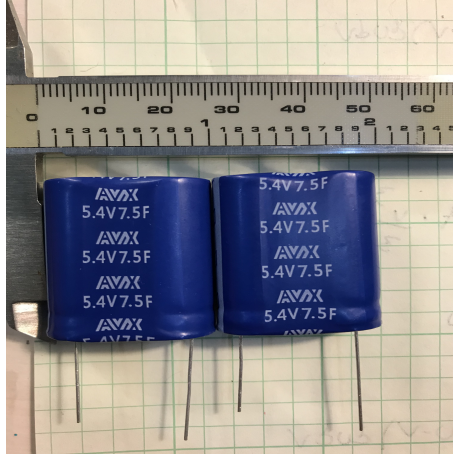


Figure 3.15: AVX 7.5F/5.4V Supercapacitor

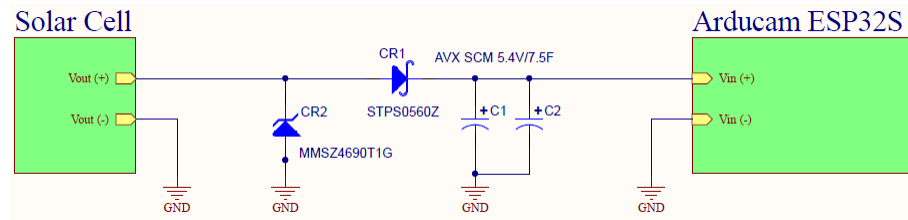


Figure 3.16: Power System Schematic

3.3.4 Schematic

Figure 3.16 shows the schematic for the power system design. Connected in series with the solar cell output is a low forward drop Schottky diode that prevents back-powering the solar cell. The two 7.5F supercapacitors are placed in parallel with the device to buffer current and store energy. A 5.6V Zener diode prevents the system from exceeding the supercapacitors rated voltage. The proposed power system prioritizes simplicity and cost. Similar works, like [7], use buck boost converters to implement maximum power point tracking systems for power efficiency purposes. Our application does not require maximised power efficiency because the solar cell generates 150-250mW of power during the day and the sensor node while in sleep mode at night requires around 2mW of power. This means that the solar cell takes 6-10min to collect enough energy for 12 hours of sleep operation.

At the input of the Arducam is a ME6211. The ME6211 is a high-speed LDO regulator from MicrOne and with an operating voltage range from 3.7-6V [24]. The total amount of energy stored on a supercapacitor is limited by the 7.5F capacitance and the available voltage discharge range. The voltage range that the SCMT32D755SRBB0 supercapacitors used in our system operate in is 3.7-5.4V. The minimum voltage of the LDO sets the bottom limit and the rated voltage for the supercapacitors sets the top. This maximum voltage defines the necessary clamping voltage of the Zener diode (5.6V) and the forward voltage drop of the Schottky diode (0.2V).

Chapter 4

PERFORMANCE

4.1 Communication System Performance

Two tests are completed to evaluate the communication systems performance, an accelerated image reception test and a range test. These tests are meant to validate the systems functionality and to evaluate the maximum coverage area.

4.1.1 Communication System Proof of Concept

Although the communication system architecture supports more than 2 hops, increasing the number of hops complicates the connection process and reduces reliability. Hence, this the system has only been implemented and evaluated using 2 hops. Due to limited funding and the applications long time scale, a full system evaluation is impossible. Six Arducam ESP32 boards are purchased and used for system evaluation purposes. Accelerated tests verify the systems integrity; six development boards are configured into three groups of branch and leaf node pairs. The trunk of the system runs on a laptop connected to a local WiFi network, figure 4.1 depicts the test setup.

The trunk schedules each branch leaf pair to wake up and conduct their data transactions once per minute. As shown is figure 4.1, branch leaf set 1 wakes up at the beginning of each minute, completes its data transactions then goes back to sleep, set 2 wakes up 20 seconds after set 1, and set 3 wakes 20 seconds after set 2. This process runs approximately 12 hours conveying around 2000 images to a local file on the trunk computer without failing. This test serves as an accelerated proof of

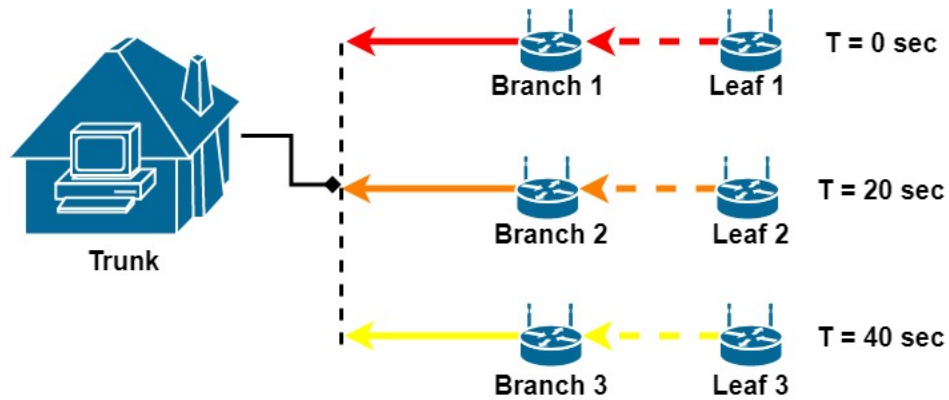


Figure 4.1: Communication System Test Setup

concept procedure validating the functionality and integrity of the image capture and communication system. Extending this test procedure to full scale involves resizing the time period to once per day and increasing the separation period between branch leaf pairs as an added factor of safety.

4.1.2 Range

Figures 4.3, 4.4, and 4.5 are images taken by the sensor node during the range test. Because of the large distance bicycle is used to estimate distance in an empty parking lot, the first few data points are taken 10 bicycle wheel rotations (69 feet) apart and the rest are taken 5 rotations (34.5) apart. Figure 4.2 shows the range test results. The test intends to evaluate the maximum distance between nodes. The branch node is kept close to the trunk and the leaf node moves linearly away from the branch until the signal is lost and images are no longer received. Figures 4.3, 4.4, 4.5 are 3 sample images sent by the leaf node during the range test that give a visual representation of how the test is conducted.

The number in dB below the distance measurement is a received signal strength indicator (RSSI), which is reported by the device and roughly indicates the received signal power. The node to node transmit distance is the achievable communication

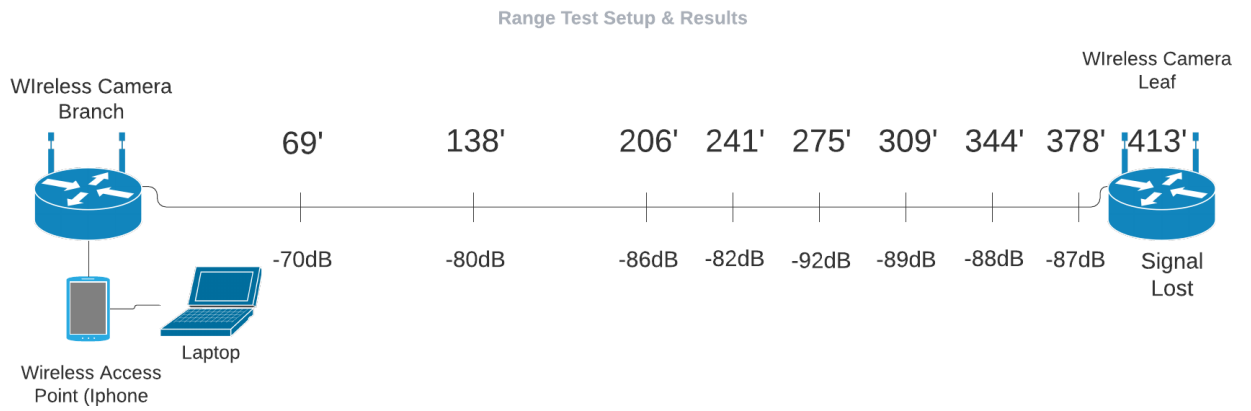


Figure 4.2: Range Test Setup Results



Figure 4.3: Sample image transmitted by leaf node at 138 feet



Figure 4.4: Image transmitted by leaf node at 275 feet



Figure 4.5: Image transmitted by leaf node at 413 feet

range from a single hop, where the total communication range in one dimension equals twice this distance because it is a two hop system.

4.2 Performance of Power System

In this section, the sensor modules power needs are measured giving context to the solar cell and supercapacitor performance evaluations. The goal of this section is to prove that the power system supports sensor node operation with a reasonable margin for error.

4.2.1 Power Evaluation

Figures 4.6 and 4.7 display the current waveforms during the operating period of the branch and leaf nodes. Average node currents derived from the datasets in these figures define the power requirements of the Arducam ESP32S Uno board during operation. For one branch leaf pair to wake up and transport one image per node to the trunk, the average current for the branch $I_{\text{branch}} = 159.9\text{mA}$ and for the leaf $I_{\text{leaf}} = 151.4\text{mA}$. The branch's higher average current makes sense because its responsible for relaying information between the leaf and trunk nodes. The following power calculations use branch node data because it is the limiting factor.

4.2.2 Supercapacitor Characterization

Equation 4.1 shows the calculation for the power requirement of the sensor nodes while asleep. P_{node} represents the average power of the Arducam ESP32S Uno board in sleep mode, P_{supercap} represents the power lost in the supercapacitor. Using equation 4.2 where $I_{\text{node}} = 300 \mu\text{A}$, $V_{\text{max}} = 5.4\text{V}$ and $V_{\text{min}} = 3.7\text{V}$, the average node power $P_{\text{node}} =$

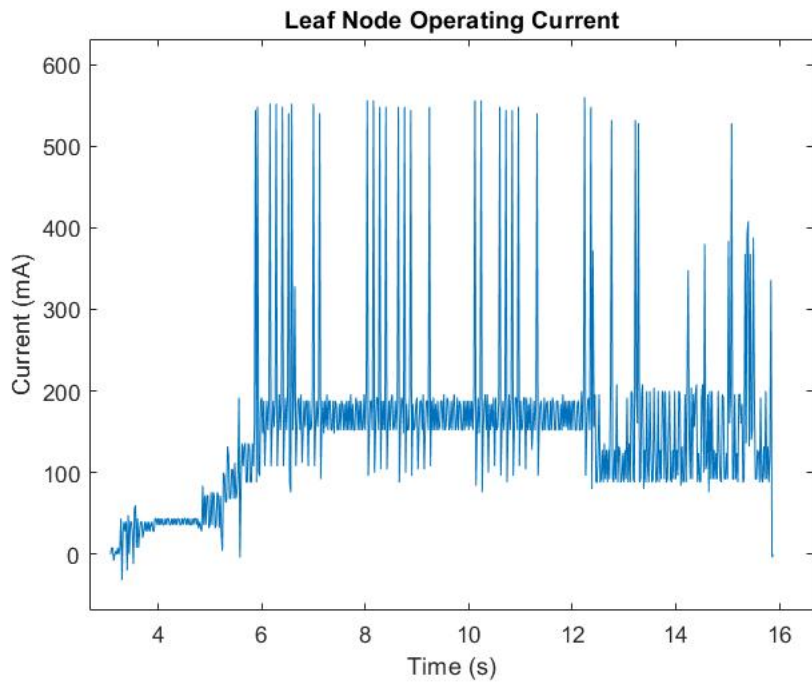


Figure 4.6: Leaf Node Current Screen Capture

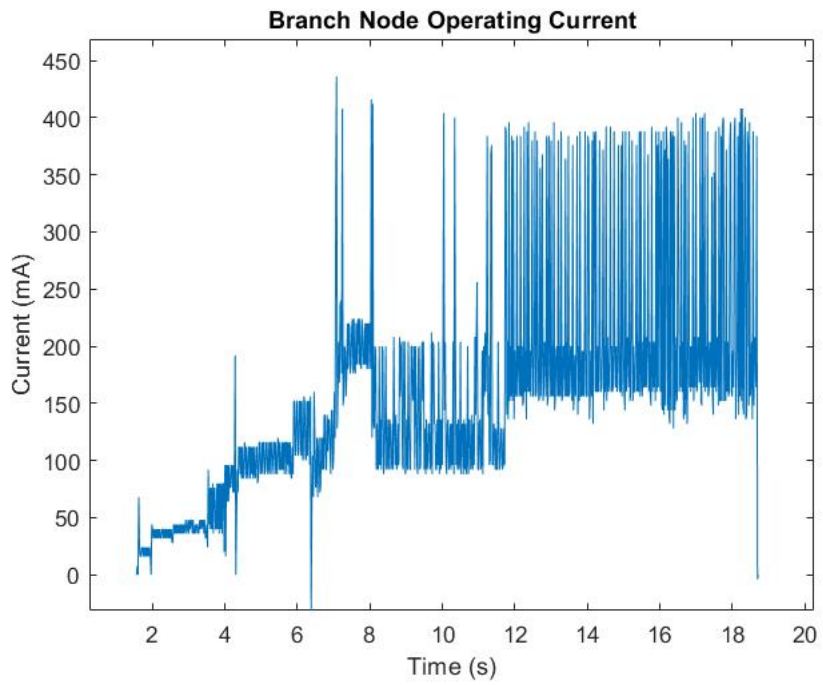


Figure 4.7: Branch Node Current Screen Capture

1.365mW. The power load associated with the supercapacitor, shown in equation 4.4, comes predominately from leakage current. To evaluate the supercapacitors leakage current a DC power supply charges the supercapacitor to its rated voltage of 5.4V and 24 hours later the voltage is measured. Equation 4.3 is used to calculate $I_{leak} = 86.8\mu A$ where $\Delta V = 1V$, $C = 7.5F$, and $\Delta T = 24hrs$. Since the leakage current is only evaluated for one capacitor, the value is doubled in equation 4.4 to account for the two capacitors connected in parallel. Using the leakage current calculated in equation 4.3 with equation 4.4, $P_{supercap}$ evaluates to 0.79mW. Finally using equation 4.1, $P_{sleep} = 2.16mW$.

$$P_{sleep} = P_{node} + P_{supercap} \quad (4.1)$$

$$P_{node} = I_{node} V_{avg} = \frac{I_{node}(V_{max} + V_{min})}{2} \quad (4.2)$$

$$I_{leak} = \frac{C * (\Delta V)}{\Delta T} \quad (4.3)$$

$$P_{supercap} = V_{avg} I_{leak} = \frac{2 * I_{leak}(V_{max} + V_{min})}{2} \quad (4.4)$$

Determining the system sleep power requirements allows us to solve for the total capacitance required to power the device overnight. Using equation 4.5, the necessary capacitance for 12 hours of sleepmode operation without solar power is 12F.

$$C = \frac{2 * P_{sleep} * t}{(V_{max}^2 - V_{min}^2)} \quad (4.5)$$

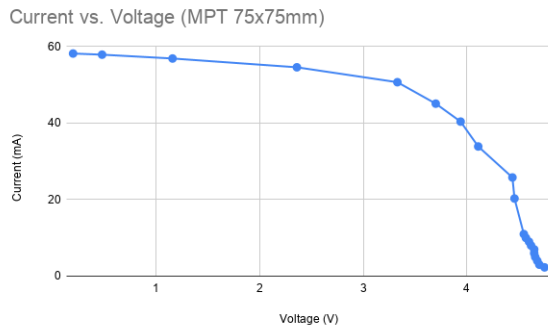
Two supercapacitors connected in parallel each with a capacitance of 7.5F makes a total capacity of 15F. The result from equation 4.5 shows that this amount of capacitance is sufficient for overnight operation with some margin for error. Given 15F of capacitance, equation 4.6 shows the device lasts in sleep mode while in darkness is operational for 14.9 hours, this metric exceeds the maximum duration of night in San Luis Obispo county which is 14.2hrs [25].

4.2.3 Solar Cell Characterization

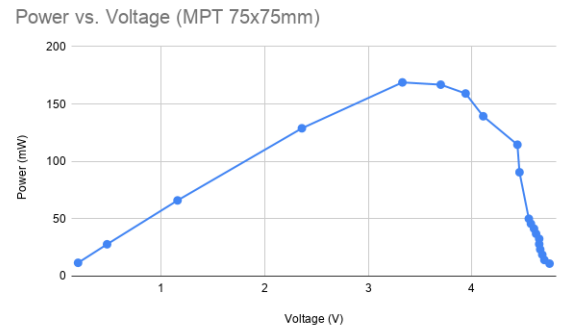
Figures 4.8a and 4.9a show the current vs. voltage characteristics of two solar cells purchased for this application. The MPT solar cell in figure 4.8a is specified with adequate short circuit current of 60mA and an open circuit voltage of 5.5V which is low enough with a 0.1V diode drop to prevent overvolutaging the supercapacitors which have a maximum operating voltage of 5.4V. The Sunyima solar cell in figure 4.9a has a 6V open circuit voltage but also a higher maximum power point than the MPT solar cell shown in figures 4.8b and 4.9b. The Sunyima solar cell is smaller and better suited for this application because its maximum power point is nearer to the maximum charge potential that the supercapacitor is rated for. Furthermore the Sunyima solar cells cost around one dollar each where the MPT solar cells cost five dollars.

$$T_{sleep} = \frac{C(V_{max}^2 - V_{min}^2)}{2 * P_{sleep}} \quad (4.6)$$

During the day, the solar cell provides power to the device. In full sunlight the Sunyima solarcell provides around 250mW at 5V. This amount is sufficient to power the sleeping device during the day and charge the supercapacitor. Equation 4.7 shows

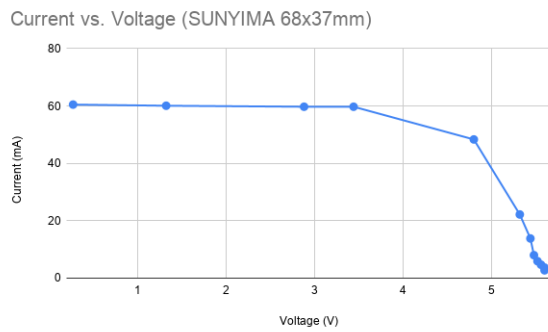


(a) Current vs. Voltage

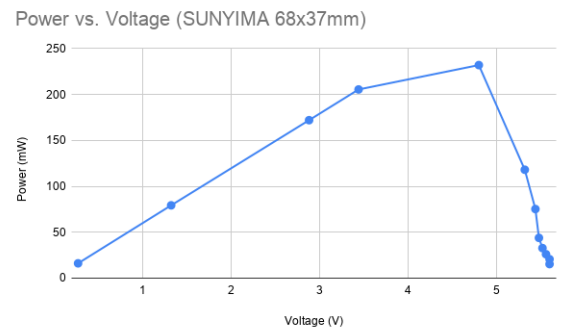


(b) Power vs. Voltage

Figure 4.8: MPT 75x75mm Solar Cell



(a) Current vs. Voltage



(b) Power vs. Voltage

Figure 4.9: SUNYIMA 68x37mm Solar Cell

that when the device sleeps during daylight hours it takes 7.8 minutes to charge the supercapacitor from 3.7V to full capacity.

$$T_{charge} = \frac{C(V_{max}^2 - V_{min}^2)}{2 * (P_{solar} - P_{sleep})} \quad (4.7)$$

The modules operating power $P_{op} = 727\text{mW}$, calculated in equation 4.8, exceeds the output power of the solar cell. This means that while the module is in operating mode the supercapacitor bank sources power to make up the difference.

$$P_{op} = V_{avg}I_{op} = \frac{I_{op}(V_{max} + V_{min})}{2} \quad (4.8)$$

Equation 4.9 gives the net operating power requirement from the supercapacitor bank, $P_{net} = 478.4\text{mW}$. Equation 4.10 uses P_{net} to calculate the amount of time the supercapacitor buffers the sensor module while in operating mode $T_{op} = 4$ minutes. The average amount of time the branch module takes to complete its objectives and return to sleep is 20s, so 4 minutes is sufficient.

$$P_{net} = P_{op} + P_{supercap} - P_{solar} \quad (4.9)$$

$$T_{op} = \frac{C(V_{max}^2 - V_{min}^2)}{P_{net}} \quad (4.10)$$

The calculations for charging and operating time do not take into account sub-optimal output power from the solar cell. As the supercapacitor bank charges past the maximum power point of the solar cell (5V) the solar cell begins to reduce the amount of power it produces to 100mW at 5.4V according to Figure 4.9. Given we take 1 image

per day and assuming 10 hrs of sunlight per day we would have approximately 5 hours to charge the device. The approximated value for charge time (7.8m) is around 40 times less than our approximate time window whereas the reduced power from sub-optimal solar cell harvesting is about 2.5 times less. Aside from the reduction in harvested power as the solar cell approaches a full charge, the weather effects the solar cells output power. The implemented system is designed to receive at least some power from the sun each day, around 9 minutes of charging in direct sunlight per day. If this requirement is not met the sensor nodes could fall offline until the sun returns.

4.2.4 Sensor Node Recovery

In the event that branch sensor node falls offline because of a loss of power, the leaf nodes connected to it would also lose connection. A sensor node that has fallen out of sync can resynchronise with the system by periodically attempting to make a connection (4s) and sleeping (20s) until a connection is made. The 20% duty cycle that an out of sync sensor node performs is low enough to extract a net positive amount of energy from each cycle while the solar cell is in full sunlight, the break even duty cycle is around 37%. Lost branch nodes easily reconnect to the always powered trunk node. Connected branch nodes wait 21 seconds in an idle state to ensure overlap time between them and any lost leaf nodes attempting to reconnect to the network.

4.2.5 Sensor Node Size

Given the measurements in table 4.1 the sensor nodes as is would fit in a minimum of a 2.75x2x0.85 inch box.

Table 4.1: Sensor Node Hardware Size

Hardware	Length (in)	Width (in)	Height (in)
Arducam ESP32 Dev. Board	2.75	2	0.25
AVX 7.5F/5.4V Supercapacitor	1.5	1	0.5
Sunyima 5V 250mW Solar Cell	2.75	1.5	0.1

Extending the packaging volume to a 3x3x1 inch box, allows for 2 Sunyima solar cells, and 6 AVX supercapacitors. Based off of the authors reported solar cell size, a 3x3x1 inch package would be around the same size or smaller than the sensor node prototypes presented in [7] and [11]. Increasing the package size has the potential to double the solar cell output power and triple the total energy storage of the device in comparison the existing sensor node. This provides a factor of safety for non ideal weather conditions.

Chapter 5

CONCLUSION

5.1 Results

A 802.11b sensor network is created using mesh networking techniques to extend range 400ft per link while maintaining relay node sleep mode functionality. A solar energy harvesting system improves sensor node lifetime. A two hop system with six nodes is implemented in a laboratory environment validating the communication systems integrity over an 800 foot range. Moving away from a primary battery system to an optimized solar energy harvesting increases the module lifetime indefinitely.

5.2 Future Work

The following items are considered for future work.

- Implementing the proposed system with more than 2 hops
- Building a custom board that integrates the power supply circuitry
- Packaging the sensor nodes for outdoor operation

Implementing the presented communication system with more than 2 hops improves maximum coverage area, but increases the risk for faults. Adding a more sophisticated error handling system improves the reliability of the communication system. Streamlining the connection process by implementing interrupt based event handling reduces the necessary operating time, further optimizing the sensor nodes energy

needs. At the front end of the arducam ESP32S module is an inefficient LDO. This device throws away excess power when the input voltage is greater than 3.3V and the device requires 3.7V to operate. This limits the minimum voltage the solar cell can drop to before ceasing to source power. Replacing this LDO with a buck boost converter system affords power savings while the source voltage is greater than 3.3V and allows the solar cell to continue to source power even when its voltage dips below 3.7V. Building a custom PCB that integrates the power electronics has the potential to improve power performance, reduce cost, and streamline the packaging process. Since the sensor nodes operate in a harsh outdoor environment, packaging the sensor nodes in a rugged water resistant container that protects the solar cell and camera lens is critical.

BIBLIOGRAPHY

- [1] A. Somov, D. Shadrin, I. Fastovets, A. Nikitin, S. Matveev, I. Seledets, and O. Hrinchuk, "Pervasive agriculture: Iot-enabled greenhouse for plant growth control," *IEEE Pervasive Computing*, vol. 17, no. 4, pp. 65–75, 2018.
- [2] B. Keswani, A. G. Mohapatra, A. Mohanty, A. Khanna, J. J. P. C. Rodrigues, D. Gupta, and V. H. C. de Albuquerque, "Adapting weather conditions based iot enabled smart irrigation technique in precision agriculture mechanisms." *Neural Computing Applications*, vol. 31, no. 1, pp. 277 – 292, 2019. [Online]. Available: <http://ezproxy.lib.calpoly.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=aph&AN=135751376&site=ehost-live&scope=site>
- [3] J. wahl, "Development and characterization of an iot network for agricultural imaging applications," Ph.D. dissertation, 2020.
- [4] S. Uludag, T. Imboden, and K. Akkaya, "A taxonomy and evaluation for developing 802.11-based wireless mesh network testbeds," *International Journal of Communication Systems*, vol. 25, no. 8, pp. 963–990, 2012. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/dac.1299>
- [5] Y. Liu, K. . Tong, and K. . Wong, "Reinforcement learning based routing for energy sensitive wireless mesh iot networks," *Electronics Letters*, vol. 55, no. 17, pp. 966–968, 2019.

- [6] X. Lu, P. Wang, D. Niyato, D. I. Kim, and Z. Han, “Wireless networks with rf energy harvesting: A contemporary survey,” *IEEE Communications Surveys Tutorials*, vol. 17, no. 2, pp. 757–789, 2015.
- [7] F. Wu, C. Rüdiger, and M. R. Yuce, “Real-time performance of a self-powered environmental iot sensor network system,” *Sensors (Basel, Switzerland)*, vol. 17, no. 2, Feb 01 2017, date completed - 2018-02-14; Date created - 2017-02-04; Date revised - 2020-01-12; SuppNotes - Cited By: Sensors (Basel). 2015 Feb 11;15(2):4072-96 25679312] *Sensors (Basel)*. 2015 Apr 22;15(4):9481-518 25912349] *Sensors (Basel)*. 2015 Dec 10;15(12):31005-22 26690433; Last updated - 2020-01-17. [Online]. Available: <http://ezproxy.lib.calpoly.edu/login?url=https://www-proquest-com.ezproxy.lib.calpoly.edu/docview/1865537122?accountid=10362>
- [8] G. Peruzzi and A. Pozzebon, “A review of energy harvesting techniques for low power wide area networks (lpwans),” *Energies*, vol. 13, no. 13, p. 3433, Jul 2020. [Online]. Available: <http://dx.doi.org/10.3390/en13133433>
- [9] L. Varandas, J. Faria, P. Gaspar, and M. Aguiar, “Low-cost iot remote sensor mesh for large-scale orchard monitorization,” *Journal of Sensor and Actuator Networks*, vol. 9, no. 3, p. 44, Sep 2020. [Online]. Available: <http://dx.doi.org/10.3390/jsan9030044>
- [10] X. Yue, M. Kauer, M. Bellanger, O. Beard, M. Brownlow, D. Gibson, C. Clark, C. MacGregor, and S. Song, “Development of an indoor photovoltaic energy harvesting module for autonomous sensors in building air quality applications,” *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2092–2103, 2017.

- [11] J. R. Cotrim and J. H. Kleinschmidt, "Lorawan mesh networks: A review and classification of multihop communication," *Sensors*, vol. 20, no. 15, p. 4273, Jul 2020. [Online]. Available: <http://dx.doi.org/10.3390/s20154273>
- [12] D. Cafe, "Ov2640 2.0 mega pixels camera module," 2015, [Online; accessed November, 2020]. [Online]. Available: <http://www.datasheetcafe.com/ov2640-datasheet-pdf/>
- [13] OmniVision, "Ov2640 color cmos uxga (2.0 megapixel) camerachip." [Online]. Available: <http://www.datasheetcafe.com/ov2640-datasheet-pdf/>
- [14] Y. Fitter, "Strawberry detection under various harvestation stages," Ph.D. dissertation, 2019.
- [15] "Ieee standard for information technology - telecommunications and information exchange between systems - local and metropolitan networks - specific requirements - part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications: Higher speed physical layer (phy) extension in the 2.4 ghz band," *IEEE Std 802.11b-1999*, pp. 1–96, 2000.
- [16] A. T. Manufacturing, "Esp32s module," 2016, [Online; accessed November, 2020]. [Online]. Available: http://www.ai-thinker.com/pro_view-25.html
- [17] K. Townsend, "Introduction to bluetooth low energy," 2020, [Online; accessed November, 2020]. [Online]. Available: <https://learn.adafruit.com/introduction-to-bluetooth-low-energy/gatt>
- [18] D. P. George, "ubluetooth — low-level bluetooth." [Online]. Available: <https://docs.micropython.org/en/latest/library/ubluetooth.html>

- [19] S. Santos, “How to set an esp32 access point (ap) for web server,” 2019, [Online; accessed November, 2020]. [Online]. Available: <https://randomnerdtutorials.com/esp32-access-point-ap-web-server/>
- [20] D. P. George, “Network basics.” [Online]. Available: https://docs.micropython.org/en/latest/esp8266/tutorial/network_basics.html
- [21] G. for Geeks, “Socket programming in c/c++,” 2019, [Online; accessed November, 2020]. [Online]. Available: <https://www.geeksforgeeks.org/socket-programming-cc/>
- [22] C. Coleman, “A practical guide to ble throughput.” [Online]. Available: <https://interrupt.memfault.com/blog/ble-throughput-primer#auditing-ble-throughput-limiting-factors>
- [23] tsaarni, “Micropython with esp32 cam.” [Online]. Available: <https://github.com/tsaarni/micropython-with-esp32-cam>
- [24] MicrOne, “High speed ldo regulators, high psrr, low noise, me6211 series.”
- [25] TimeandDate.com, “San luis obispo daylight data.” [Online]. Available: <https://www.timeanddate.com/sun/usa/san-luis-obispo>
- [26] “Cal Poly Github,” <http://www.github.com/CalPoly>.
- [27] A. Einstein, “Zur Elektrodynamik bewegter Körper. (German) [On the electrodynamics of moving bodies],” *Annalen der Physik*, vol. 322, no. 10, pp. 891–921, 1905.
- [28] M. Goossens, F. Mittelbach, and A. Samarin, *The L^AT_EX Companion*. Reading, Massachusetts: Addison-Wesley, 1993.
- [29] D. Knuth, “Knuth: Computers and typesetting.” [Online]. Available: <http://www-cs-faculty.stanford.edu/~{ }uno/abcde.html>