8-2022

An Evaluation of Wi-Fi 802.11b Backscatter

Anthony Chen
ajchen@g.clemson.edu

An Evaluation of Wi-Fi 802.11b Backscatter

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Electrical Engineering

---

by
Anthony Jye-Ru Chen
August 2022

Accepted by:
Dr. Linke Guo, Committee Chair
Dr. Harlan Russell
Dr. Carl Baum

Abstract

Internet of Things (IoT) devices are in need of low-power communications systems with longevity and reliability. With the use of backscatter technology, IoT devices can communicate at the cost of almost no power and can last for up to a decade. Furthermore, backscatter technology is compatible with everyday wireless signals such as Wi-Fi and Bluetooth, allowing for easy communication without specific hardware constraints. This thesis aims to evaluate a Wi-Fi backscatter system and analyze its ease in triggering off of such ambient signals and sources. The system will utilize Wi-Fi 802.11b as a backscatter source to trigger the backscatter system to transmit its own data as a modified Wi-Fi signal 802.11b. This thesis will establish the effectiveness of a particular backscatter implementation, provide an overview of its systems, and demonstrate the efficacy of each system's abilities.

Dedication

This thesis is dedicated to my parents, Henry and Susan Chen, for their dedication to helping me succeed in my life and education.

Acknowledgments

I would like to thank my advisors Dr. Linke Guo, Dr. Harlan Russell, and Dr. Carl Baum for taking their time in aiding and guiding me in my studies. They offered advice and insights into my research and education when I needed it the most. I would also like to thank the students that have dedicated their time to working with my on my project, Chun Chih and Daniel Pendleton. I would not have been able to complete my research without them.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

With the advent and rising popularity of Internet of Things (IoT) technology, more and more household devices are having stricter power and communication restraints. By utilizing backscatter communications and hardware, devices are able to achieve close to zero power usage to communicate their data. Due to the extremely low power usage of backscatter technology, wireless sensors can receive longevity, lasting for up to decades with reliable communications.

1.1 Backscatter Communications

Backscatter, unlike typical communications systems, does not generate its own signals. Instead, backscatter systems will 'backscatter' off of currently existing signals around it to transmit the data it needs to convey. These backscatter systems will achieve this by harvesting the wireless signals around it and regurgitating the signals back out, but with slight modifications. During this harvesting of ambient signals, the backscatter system will receive most, if not all, of the power it needs to operate and communicate. This allows for backscatter communications systems to support IoT devices and sensors for years through the use a small battery or even with just the ambient signals themselves.

1.2 IoT Devices and Sensors

In the recent years, the use of IoT devices and wearables has been increasing rapidly as well as research into its field. These IoT devices improve our daily life by automating and controlling menial work and chores, such that people have more convenience and work less. These devices can range from anything as large as a car to something as small as a heartbeat monitor. As long as a device is 'smart' enough to communicate with other devices and share their data and information, they can be considered part of the IoT. However, as described in [7], the IoT faces a multitude of challenges including, security, privacy, connectivity, and limited energy resources. By and large, communications and energy management are a large portion of what complicates IoT devices. However, through the use of backscatter technology, reliable communications for almost no energy becomes available, solving some of the main challenges of IoT.

1.3 Statement of Work

This thesis evaluates the overall capabilities of the Wi-Fi backscatter system inspired by [12], analyzing its excitation trigger sensitivity, and Wi-Fi backscattering proficiency. The scope of this thesis mainly involves the implementation and testing of the backscatter tag itself, while the receiver and transmitter design are outside of the scope of this project. The rest of this thesis is as follows. Chapter 2 will give background on backscatter systems and their signal sources. Chapter 3 will discuss the related works. Chapter 4 will discuss the backscatter system model and testing procedures. Chapter 5 will discuss the result obtained from experimentation. Finally, chapter 6 will conclude this thesis.

Chapter 2

Background

This chapter provides background on the current implementations of backscatter systems and the application of backscatter systems in wireless communications.

2.1 Energy Harvesting

One of the core concepts within backscatter is energy harvesting, where like the name suggests, energy is harvested from the environment in order to power a device. This has many applications, such as for powering various home electronics, medical implants, vehicles, and IoT devices [10]. Within energy harvesting, there are various techniques such as wireless power transfer [6], which transfers power without information, wireless power communication networks, which allows devices to harvest energy and use that energy to transmit data [10], and simultaneous wireless information and power transfer (SWIPT) [10], which allows for transmission of both information and power at the same time. However, unlike most energy harvesting devices, backscatter technology is extremely low power itself and is able to both power itself and receive data from signals. Because of this, backscatter technology has longevity and is self-sustainable for long periods of time with little to no outside power source.

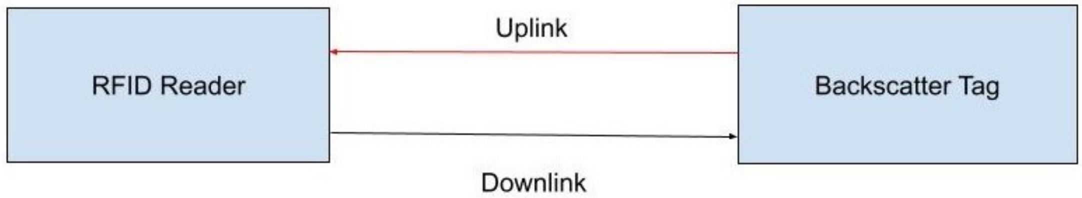


Figure 2.1: Traditional Backscatter

2.2 Traditional Backscatter

Backscatter systems usually are composed of three parts: an excitation signal generator, the backscatter tag, and a receiver to decode the backscattered signal. Traditionally, backscatter implementations utilize specialized hardware in order to be able to generate excitation signals for the backscatter to be able to reflect as well as specialized hardware to decode the backscattered signals, making it incapable of widespread implementation for data-intensive communications systems. This is because the backscatter transmitters must be placed next to the RF signal sources, limiting device usage and coverage area, while the backscatter transmitters must be within the same device, usually a Radio Frequency Identification reader (RFID reader), creating self-interference and reducing communications performance [10]. Furthermore, these systems will only operate when prompted by a backscatter transmitter. These limitations can be seen from Fig. 2.1, where both the excitation signal generator (the RF source) as well as the signal receiver are in the same device, shown by the RFID reader. On the other hand, the tag is separate, and backscattering off of the specialized signals sent by the reader and sending its data back to the RFID reader.

2.3 Current Backscatter Implementations

Unlike traditional backscatter systems, current implementations of backscatter avoid the need for a dedicated RF signal source, implementing backscatter systems such as Ambient Backscatter (Wi-Fi, Bluetooth, etc.), Bistatic Backscatter, Full-Duplex Backscatter, Inter-technology Backscatter, LoRa Backscatter, LIS-aided backscatter (Large Intelligent Surface), and Piezo-Acoustic Backscatter (PAB) [11].

Current implementations of Ambient Backscatter make use of existing ambient wireless

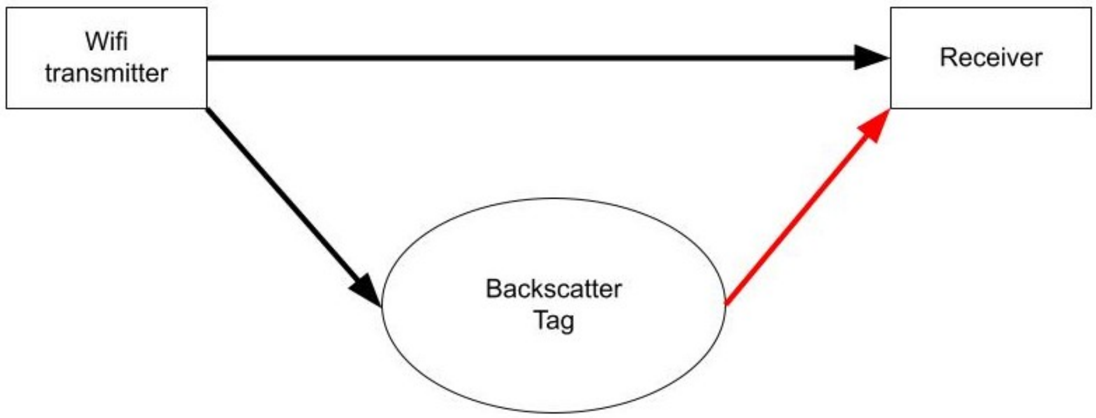


Figure 2.2: Wi-Fi Backscatter System

signals, such as Wi-Fi, ZigBee, and Bluetooth. Among these implementations, some backscatter systems are designed to only utilize a single type of ambient signal such as [12] utilizing only Wi-Fi 802.11b and [2] utilizing only Bluetooth. On the other hand, some backscatter implementations are set to utilize any such ambient signal such as in [13]. These backscatter systems usually operate by backscattering or absorbing ambient signals to indicate whether a '1' or a '0' is being transmitted. On the receiver end, readers utilize various signal processing methods to detect the state of the tag. Conversely, Inter-Technology backscatter works similarly to ambient backscatter, but converts one type of signal to a different one. For example, a Bluetooth transmitter can transmit a signal to the backscatter tag, which will transform the signal into a Wi-Fi signal for a Wi-Fi receiver. Similar to the Ambient Backscatter, the LoRa Backscatter implementations also utilize a RF signal which is backscattered by the tag, but instead has long-distance communication capabilities due to the implementation of spread-spectrum coding and the high sensitivity of LoRa signals [11].

Full-Duplex Backscatter is a backscatter system that is based on full-duplex Wi-Fi gateways with a large number of antennas simultaneously communicating with laptops and smartphones while collecting data from backscatter tags. When these Wi-Fi gateways transmit to the laptops or smartphones, the signals can be reflected by the backscatter tags and the Wi-Fi gateways would recover the tag's data [11].

LIS-aided backscatter utilizes many almost passive reflectors in order to alter the phase of incoming signals [11]. This backscatter implementation is essentially a node with large backscatter units, enabling high data rate transmissions with the ability to even strengthen signals. Finally, PAB is the first underwater backscatter, which utilizes underwater acoustics to harvest energy and backscatter communications. A PAB would be set up with a projector to project an underwater acoustic signal, which the tag would backscatter to a hydrophone to pick up the data which has been sent by the backscatter. In general, most implementations of backscatter systems utilize similar methods to communicate their data through the use of passive reflection and absorption by the tag and backscattering off of various types of transmitted signals to some receiver which decodes the data. With all of these diverse types of backscatter implementations, backscatter research has much potential in enabling extremely low-power communications.

2.4 Backscatter Hardware

Backscatter tags in general passively transmit data by harvesting energy from received signals, modify these signals, then re-transmit them. Traditionally, these backscatter systems achieve this by controlling the backscatter tag's reflection coefficient $F_T(t)$ to adjust the frequency, amplitude, and phase of the signal it receives in order to encode its data. This reflection coefficient is controlled by the antenna impedance R_A in combination with the load impedance R_L, where it can be calculated by [11]:

$$F_T(t) = (R_L - R_A)/(R_L + R_A) \tag{2.1}$$

These backscatter tags typically employ either analog or digital modulation to modulate their data onto signals. Generally, these tags will control their reflection coefficient through the use of analog circuits for continuously time-varying reflection coefficients as shown in Fig. 2.4 or a controller with several different fixed states of reflection coefficients as shown in Fig. 2.3 [11], where the backscatter tag can modulate its own data onto existing signals by changing to different reflection coefficients.

2.5 Backscatter Power Usage

One of the main benefits of utilizing backscatter is its low power consumption. As described in [12], Wi-Fi backscatter can utilize as little as on the order of tens of µW to operate and transmit its data. Other backscatter types such as LoRa backscatter [8], can even use single digit amounts of µW to operate and transmit their data. This power usage is far less than the current power consumption of IoT communication devices and most communication devices in general.

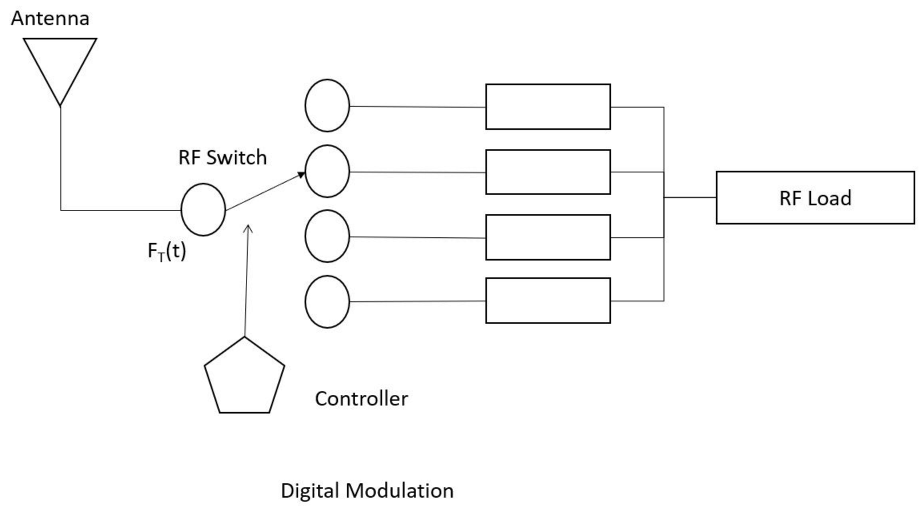


Figure 2.3: Digital Modulation Backscatter

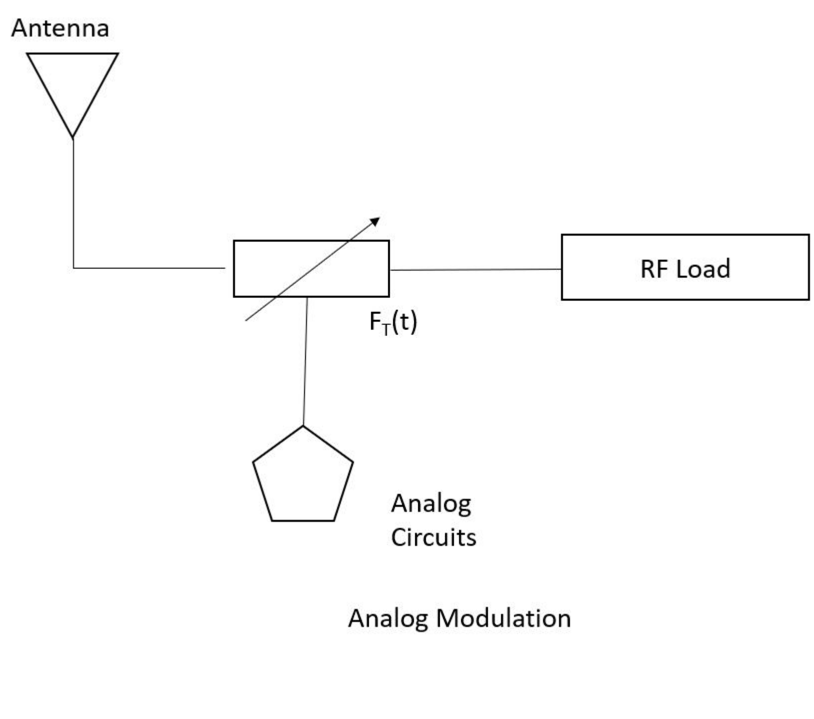


Figure 2.4: Analog Modulation Backscatter

2.6 Wi-Fi 802.11b

The evaluation in this thesis utilizes Wi-Fi 802.11b to transmit and backscatter signals. As such, it is important to understand the characteristics of Wi-Fi 802.11b. Wi-Fi 802.11b consists of thirteen 22MHz wide channels along the 2.4 GHz frequency band [3]. This can be sent with 1 Mb/s, 2 Mb/s, 5.5 Mb/s, and 11 Mb/s data rates, with the first two utilizing DBPSK and DQPSK, and the last two data rates utilizing CCK. In the implementation of this thesis's evaluation, 1 Mb/s DSSS is assumed to allow for backscatter implementation. The format of the Wi-Fi packet can be seen in Fig. 2.5, consisting of a preamble, a header, and variable bits depending on the type of modulation. Within the header are the synch field, which is utilized for synchronization, and the Start Frame Delimiter (SFD) which is a pattern of 16 bits to signal the beginning of a frame (1111 0011 1010 0000 in the case of this thesis). Next is the header, which consists of the signal field which specifies the desired modulation scheme and its desired rate (i.e., 0x0A for 1 Mbps with DBPSK), and the Cyclic Redundancy Check (CRC), which is utilized for error detection. Both the PLCP preamble and header are transmitted at 1 Mbps with the PSDU transmitted at different data rates depending on the modulation schemes described before [3].

PLCP Preamble (144 bits)	PLCP Header (48 bits)	PSDU (Variable bits)

Figure 2.5: Wi-Fi 802.11b Packet Format

2.6.1 Wi-Fi Codewords

To encode its packets, Wi-Fi 802.11b utilizes a fixed and finite set of codewords for each type of modulation scheme. With 1 Mb/s DBPSK, Wi-Fi 802.11b uses two codewords for encoding. With 2Mb/s DQPSK, Wi-Fi 802.11b uses four codewords for encoding. With 5.5 Mb/s CCK, Wi-Fi 802.11b uses 16 codewords for encoding. And with 11 Mb/s CCK, Wi-Fi 802.11b uses 64 codewords [12].

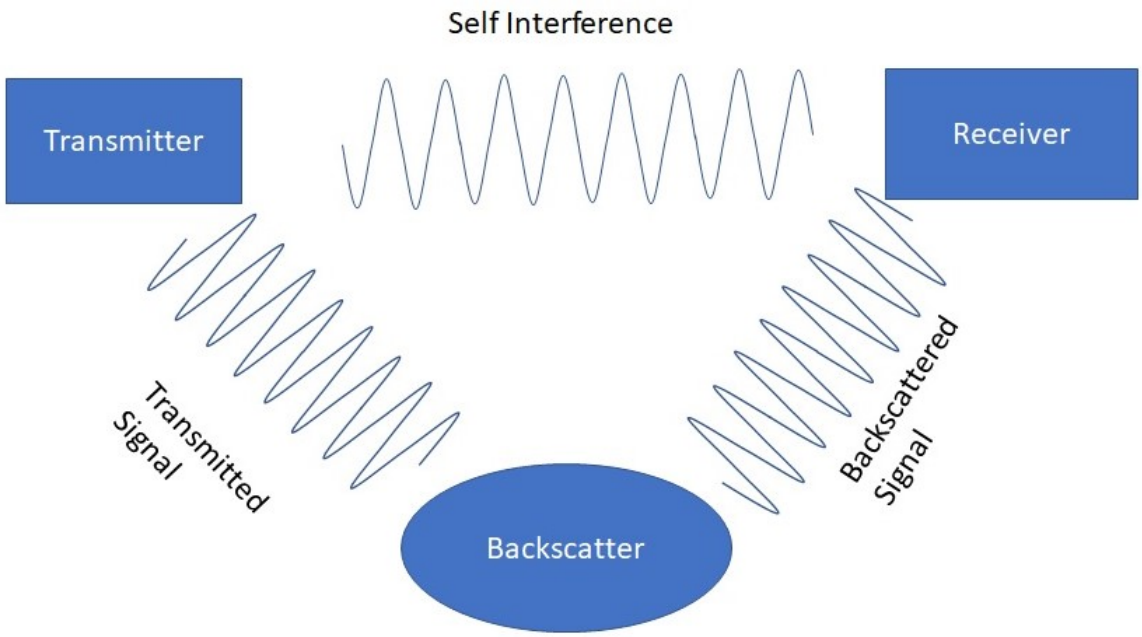


Figure 2.6: Self Interference from transmitter

2.7 Interference

In the sector of communications, one of the largest limiting factors to stable communications among devices is interference, whether it be from other signal sources or even from the device itself. One big problem with utilizing backscatter communications is cross technology interference due to sharing the same frequency band as many other signal types in the 2.4 GHz ISM band (Industrial, Scientific, and Medical). Furthermore, due to the fact that backscatter is generating signals that usually match the signal types that they are backscattering, these backscatter systems often will suffer from the similar interference issues to the signal type they backscatter or even from self-interference. For example, the Wi-Fi backscatter from [12] causes self-interference with 802.11b Wi-Fi if not properly modulated, and in [8], the backscattered signal even suffers from interference from the original RF signal source. This self-interference can be demonstrated from Fig. 2.6, where the backscattered signal becomes interfered with by the original transmitted ambient signal. Similar to the Zigbee devices described in [14], backscatter is utilized in low power wireless communications, being vulnerable to both packet loss or even total communication failure. As backscatter usually operates at extremely low power around the µW range, it is often overpowered by much stronger signals. Furthermore, many ambient backscatter implementations are tuned to operate only off of one type of trigger signal. If that signal is interfered with, then the backscatter may not even operate, much less transmit correctly.

Chapter 3

Related Work

This chapter provides background on backscatter systems being utilized in this thesis as well as other relevant implementations of backscatter.

3.1 Hitchhike

The core implementation of wifi backscatter utilized in this thesis originates from the study in [12] called HitchHike, which also utilizes WiFi 802.11b backscatter. The backscatter tag hardware utilized in this thesis originates from [12]'s implementation. [12]'s main contribution is its use of WiFi codewords and utilization of the backscatter tag to translate from one WiFi codeword to another. This allows for easy decoding at the receiver and removes the need for specialized or dedicated readers at the receiver or waveform generator at the transmitter due to the backscattered signal being WiFi 802.11b signals except for some slight processing for the XOR decoder which will be explained further on.

3.1.1 Codeword Translation in WiFi 802.11b

One of the core mechanics to the WiFi 802.11b backscatter being evaluated in this thesis is the use of codeword translation, which allows backscatter tags to embed their data onto standard wifi packets. In the case of WiFi 802.11b, every wifi packet is an encoded series of codewords that originate from a codebook. Due to the amount of wifi codewords being finite, where the only difference between select codewords is through a phase offset, backscatter systems can easily implement codeword translation by utilizing those phase offsets to translate one codeword to another.

With regards to Wifi 802.11b, there can be between 2 to 64 codewords (in powers of 2) and having a phase offset of $\frac{360°}{\text{Number of Codewords}}$ between each codeword.

$$\begin{aligned} \text{Codeword } 0 &= \text{Codeword } 1 * e^{j\pi} \\ \text{Codeword } 1 &= \text{Codeword } 0 * e^{j\pi} \end{aligned} \tag{3.1}$$

One example is Equation 3.1, which demonstrates the codebook relations of Wifi 802.11b 1 Mbps codewords. Hitchhike utilizes the codeword translation in 3.1 in their implementation of backscatter to allow for easy decoding at the receiver and to act as a simple demonstration [12].

$$\begin{aligned} \text{Tag data } 0 &= \text{Original 802.11b data} \\ \text{Tag data } 1 &= \text{Original 802.11b data} * e^{j\pi} \end{aligned} \tag{3.2}$$

In order to physically implement this in the backscatter tag, [12] the backscatter tag simply reflects the incoming 802.11b WiFi signal to create the 180°phase shift.

3.1.2 XOR Decoding

Table 3.1: Decoded Data

Back-scattered Bits	Original Wifi 802.11b Data	Decoded Data
0	0	0
0	1	1
1	0	1
1	1	0

As can be seen from Table 3.1, to decode the backscatter data at the receiver, [12] just performs a simple XOR operation between the bits it receives from the original WiFi transmission and the bits it receives from the backscatter tag. As described in the previous section, the backscatter will flip the WiFi data if it wishes to transmit a '1', and if it wishes to transmit a '0', then it will retransmit the WiFi data as is. This relation can be observed from the table, where the decoded data is just an XOR between the data received from the backscatter and the data received from the original WiFi.

3.2 Other Backscatter Systems

One of the original works to utilize WiFi as a backscatter signal source was the study in [4], or Passive WiFi. In this implementation, the backscatter reduced the need for specialized hardware by enabling backscatter decoding in standard WiFi 802.11b radios. However, unlike in [12], a specialized single tone RF carrier emitter to trigger the backscatter tag. This methodology also requires more spectrum by utilizing two WiFi channels rather than one.

Another work to utilize WiFi as a backscatter signal source was BackFi in [1], which utilizes full-duplex WiFi radios to generate and receive backscattered signals. However, unlike in [12], the WiFi radios would need to be modified in order to generate excitation signals and receive backscattered signals.

While this thesis does evaluate a backscatter system very similar to that of [12], there are many other WiFi backscatters that have been implemented that provide various benefits over [12], such as improvements in backscatter throughput, range, versatility, and power consumption ([9, 10, 13, 5]. However, many of these implementations required specialized hardware at either the transmitter or receiver for their backscatter implementation in order to generate the excitation signal or decode the backscattered signal. With regards to equipment, [12] was the first to allow any WiFi 802.11b radio to be backscattered with any receiver that can receive WiFi 802.11b and not need specialized equipment like the implementations in [1, 4].

Chapter 4

Research Design and Methods

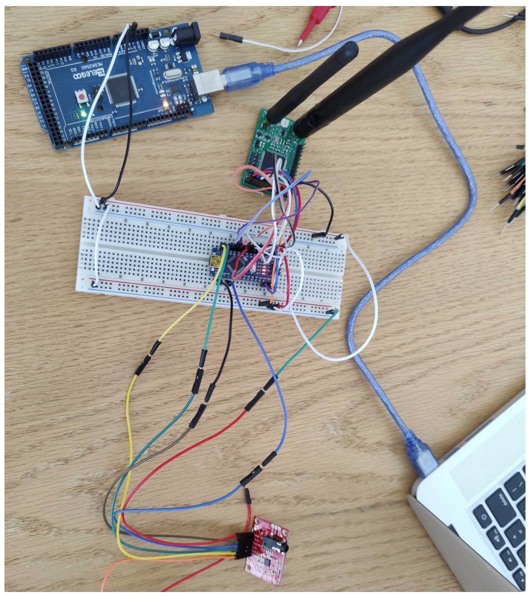


Figure 4.1: Whole Backscatter Implementation

The implementation and design of the backscatter system are described in this chapter. The sections discussed in this chapter will cover the backscatter tag's hardware and functions, the limitations and challenges faced in this implementation, and the testing procedures. The overall implementation can be seen from Fig. 4.1, where each individual portion of the system design will be covered in various separate sections.

4.1 Backscatter Tag Hardware and Functions

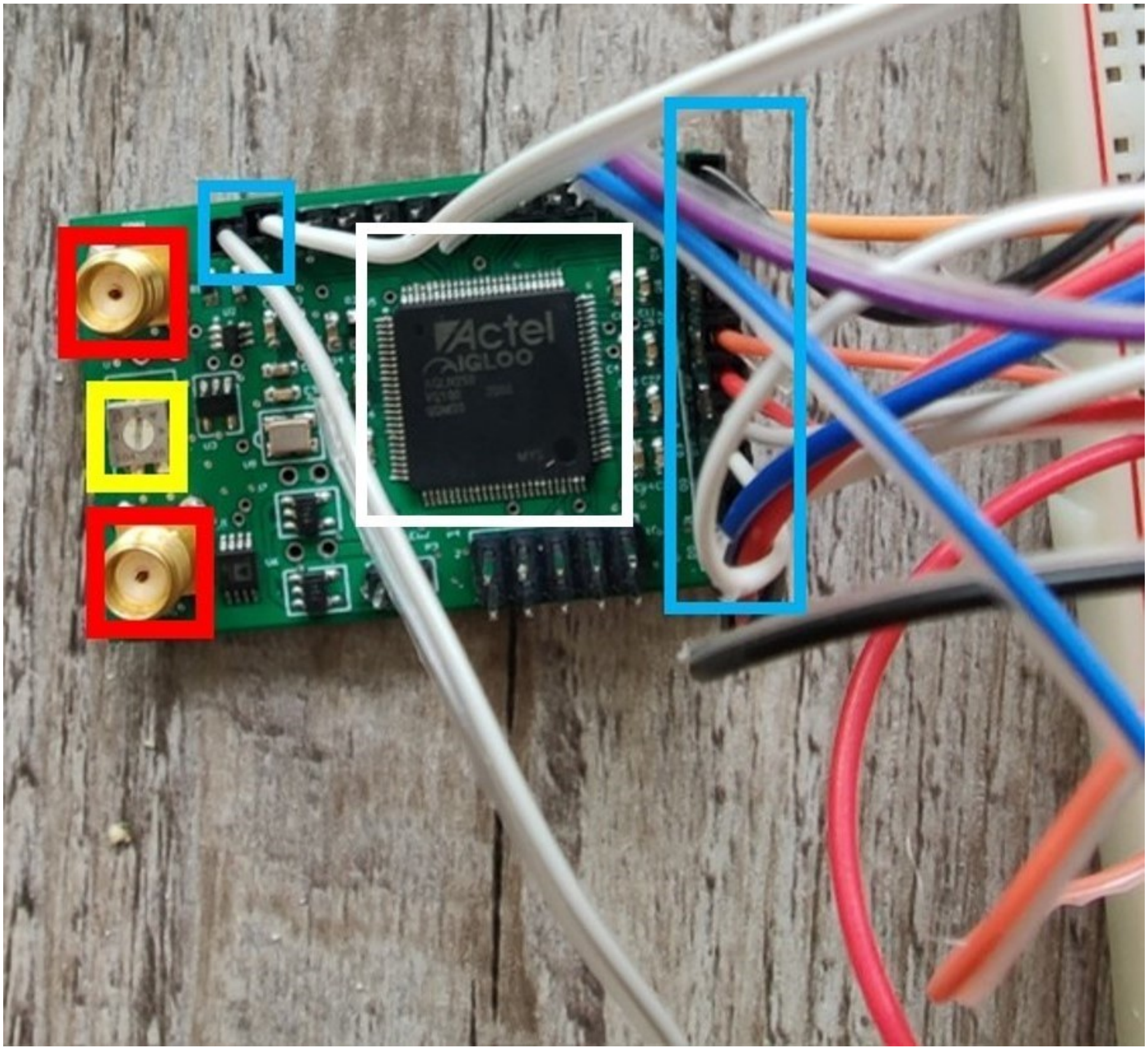


Figure 4.2: Backscatter tag

The hardware and components in this implementation are vastly the same as those in [12] with slight differences in this implementation's excitation signal generator and receiver. First in this implementation is the backscatter tag, as can be seen in Fig. 4.2. This backscatter tag was a custom non-generic FPGA manufactured according to the blueprint of the demonstrative board from [12]. The main locations of importance are boxed within the picture. First is the actual chip which implements the code which is boxed in white. This implementation utilizes the Actel Igloo AGN250, which runs the Verilog code that has been flashed on the board. Next are the pins that are boxed in blue. The pair of pins on the left are the data and comp pins, which are used to visualize the data that is being output by the board into the receiver and the value of the trigger signal to determine if the backscatter tag has been excited. Although the backscatter board's receiver is equipped to pick up trigger signals, the backscatter board is not able to differentiate between signals and is triggered by the strongest signal received. Furthermore, the implementation does not allow the backscatter tag to effectively utilize the signal obtained from the receiver, as the signal from the receiver has no path to any pins utilized by the chip. The large line of pins on the right side of the backscatter tag are used to power the backscatter board and is connected to an Arduino Nano which originally was to be used for gathering data from a heartbeat sensor, but later became unused in the implementation. In the red boxes on the left are the locations for the receiver and transmitter of the backscatter tag, with the receiver located at the top of the board and the transmitter located at the bottom of the board. Finally, one of the main actors of our evaluation is the variable resistor located on the left in the yellow box (the 3314G-1) with the ability to output resistances between $0\,\Omega$ to $35\,\mathrm{k}\Omega$. This variable resistor is connected to a 3.3 volt source, as well as a comparator. In essence, this resistor will change the voltage being led into the comparator and determine whether or not the trigger signal is strong enough to trigger the board. By increasing the resistance of the variable resistor, the voltage into the comparator becomes higher, and the minimum trigger voltage becomes higher, however with lower resistance, less voltage is led into the comparator and the backscatter board becomes easier to trigger.

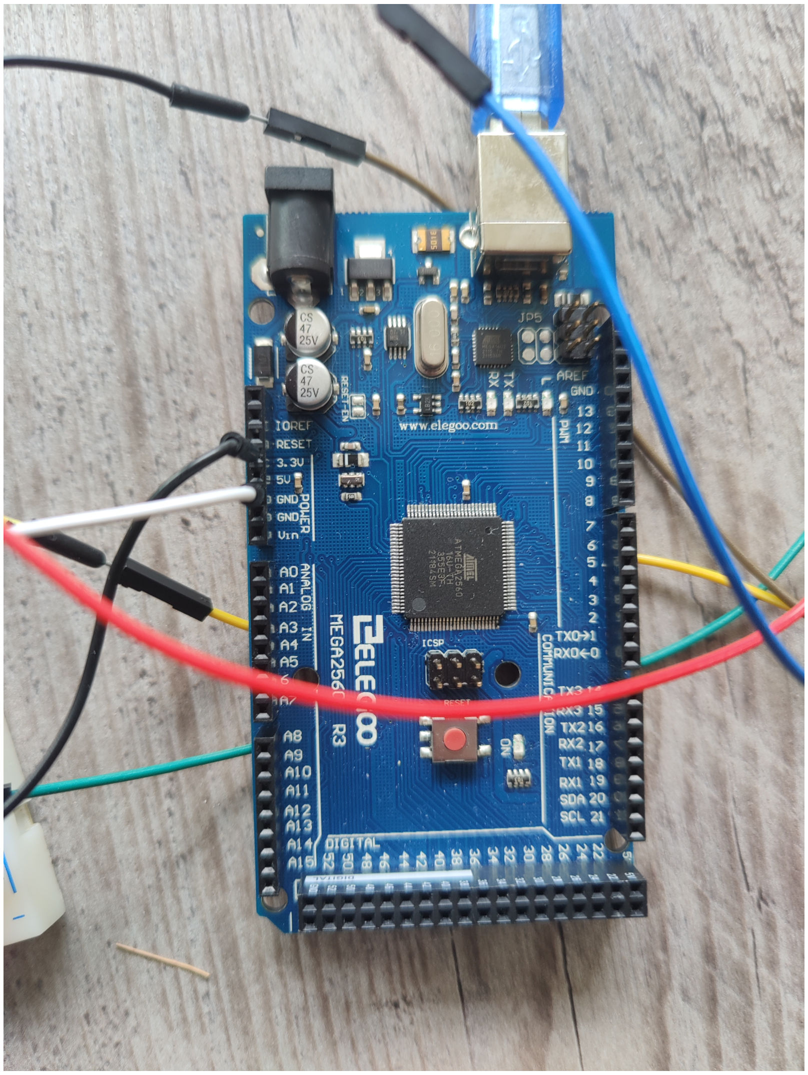


Figure 4.3: 3.3V Supply Arduino

In the design from [12], the board was meant to be a demonstrative board to display the validity of their codeword translation, making this board re-programmable and more versatile than most backscatter systems with single functions. Due to this, the board requires much more power than can be given by ambient signals and requires 3.3V of power being supplied by an Arduino Mega, as can be seen from Fig. 4.3, where the only two connections are from the 3.3V and ground, leading to the backscatter board.

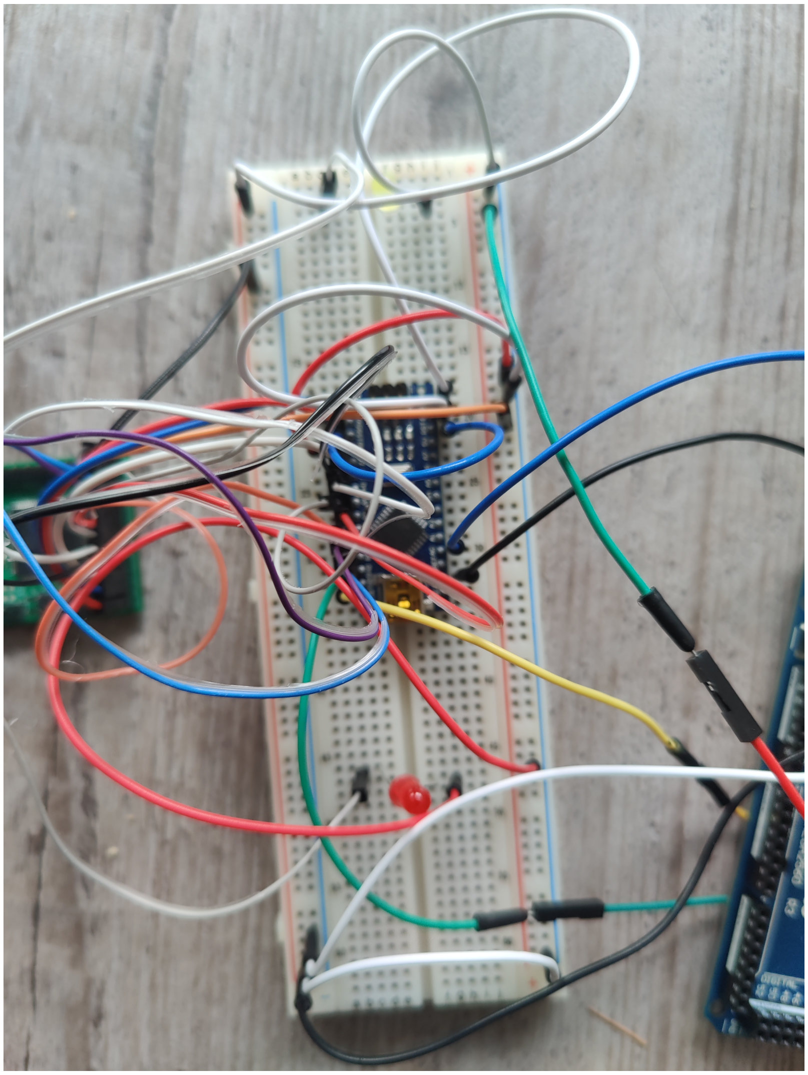


Figure 4.4: Breadboard

Finally, the all of the components are connected through the use of a standard breadboard as shown in Fig. 4.4. As shown in the figure, an Arduino Nano is used rather than the Arduino Pro Mini from the implementation in [12], as for the purposes of backscatter implementation, they have essentially the same functionality. Furthermore, from Fig. 4.4, two LEDs can be seen, one red and one yellow. These LEDs were used in testing to determine the state of the data pin (red LED), and the state of the comp pin (yellow LED) in order to easily observe when the backscatter identifies a trigger signal and when the data pin is attempting to send data to the transmitter. The Arduino Nano on the breadboard is also powered through the Arduino Mega from Fig. 4.3.

4.2 Backscatter Software

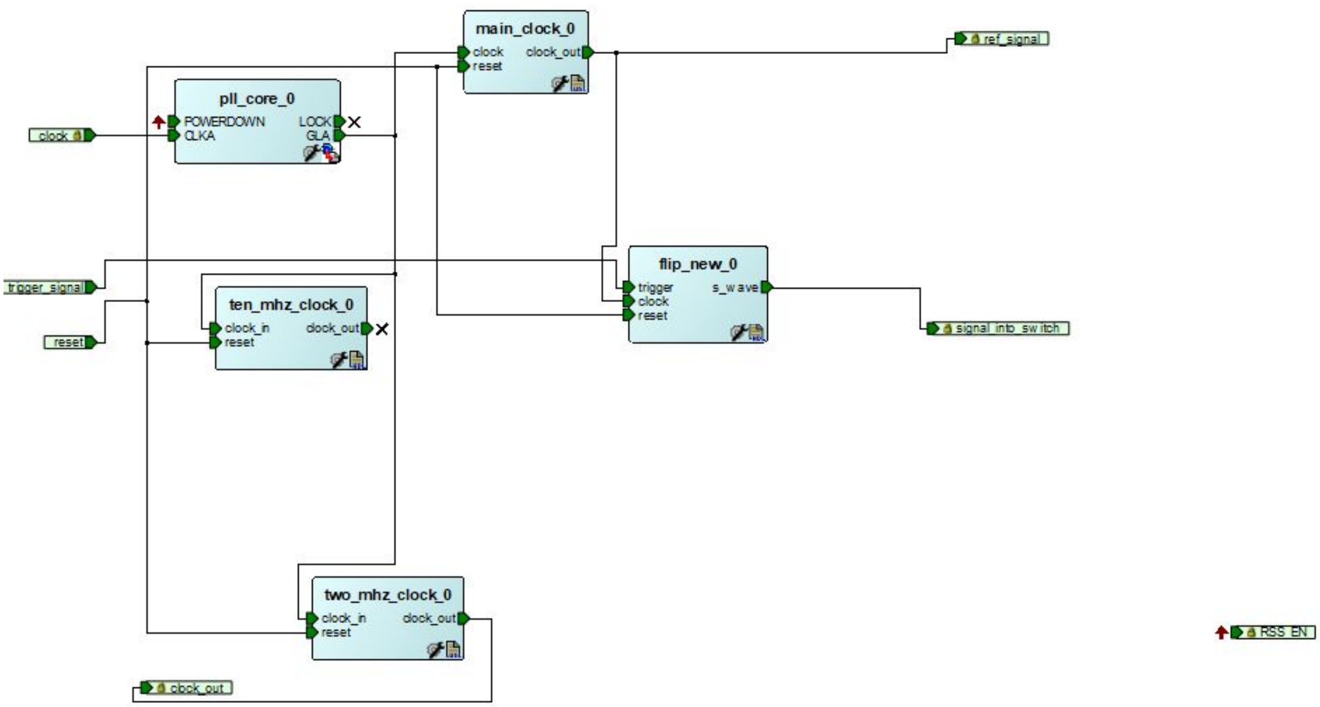


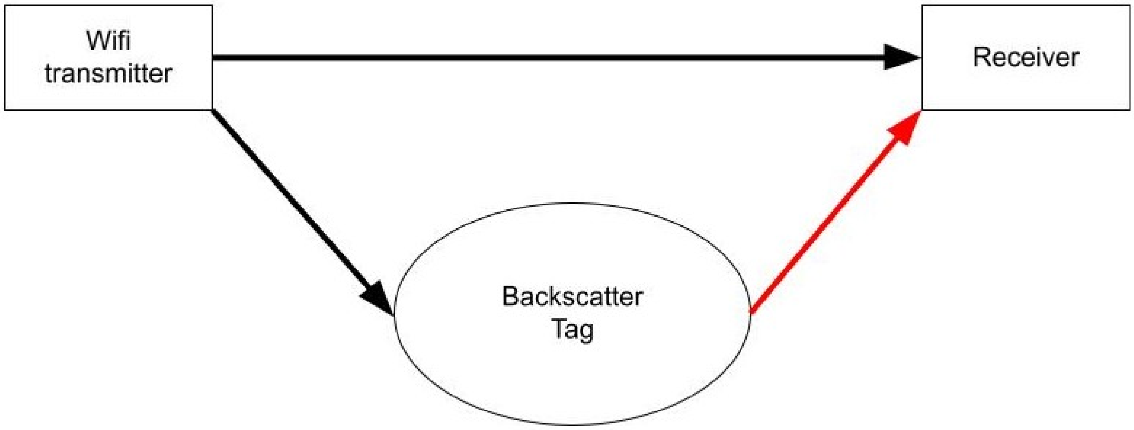
Figure 4.5: Libero Modules

Our backscatter tag was coded through the use of Libero SoC v11.9 and flashed onto the chip through the use of Flashpro. Because this design utilizes an Actel IGLOO, Libero SoC's later than version 11.9 are non-compatible and unable to be properly flashed onto the chip. In order to implement codeword translation on the backscatter board like in [12], all of the code is also in Verilog. This can be seen from Fig. 4.5, where the code utilizes a single module to implement the codeword translation and backscatter transmission. This is implemented mostly in the module labeled flip_new, which implements codeword translation and enables the system in the presence of a trigger signal. The other modules represent clocks where the only clock in use is a 50 MHz clock. Although the system in [12] states that the backscatter board modulates the signal into RF, the backscatter board implementation in the design does not actually have the capability to modulate from base-band to RF without extra components. Originally, the ensure the backscatter system performs its functions, code was developed to have the backscatter system transmit square wave, then Wi-Fi bits, then finally conduct codeword translation. The code for the backscatter software can be seen in Appendix

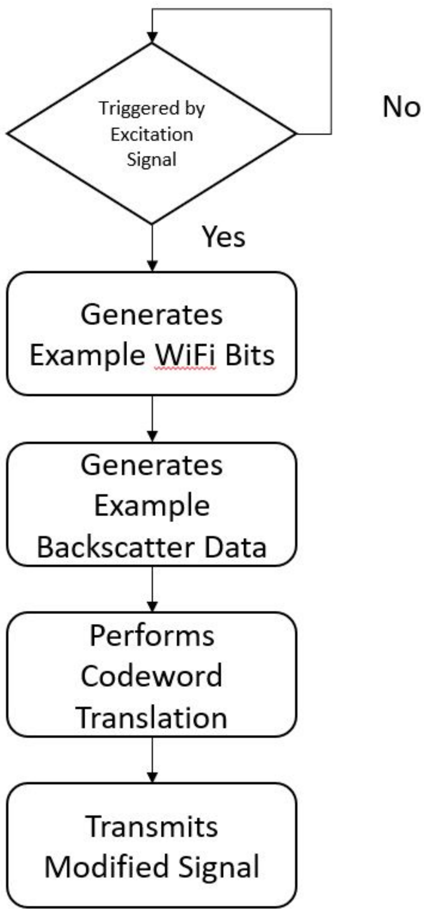
4.3 Testing

A majority of testing was done through the use of an Analog Discovery 2 (AD2), utilizing both its waveform generator and oscilloscope capabilities. Although the implementation in [12] utilizes an Intel NUC transmitter and MacBook Pro receiver, this thesis' implementation did not have a suitable 802.11b transmitter available and used the AD2's waveform generator instead. Originally, a Lenovo ThinkPad with an Intel 5300 NIC equipped was utilized to transmit the 802.11b signal for the backscatter tag to utilize, however, due to the signal strength of the transmitter being too low compared to the all of the background noise even at short distances, the design was forced to change to employ wired transmission, where the signal was transmitted from the AD2 directly into the receiver of the backscatter tag. Furthermore, because the tag is incapable of modulation from baseband to RF, the AD2 as well as an USRP N210 SDR was used to directly analyze the data being sent from the transmitter of the backscatter tag directly using a wire or coax cable. Due to all of these constraints, the general backscatter system was implemented as shown in Fig. 4.6. As can be seen from Fig. 4.6a, the AD2 provides the excitation signal and acts as an oscilloscope to observe the transmitted signal. On the other hand, Fig. 4.6b shows that the excitation signal is only used to trigger the backscatter and none of the actual transmitted data is reliant on the trigger signal itself. Once triggered, the system uses the demonstrative Wi-Fi data and backscatter data that has been flashed onto the system, performs codeword translation, then transmits the modified signal back to the AD2.

A large portion of this research was dedicated to ensuring the board properly functions and could correctly backscatter signals. To ensure its effectiveness, the board was first tested with the original software and hardware utilized from the HitchHike GitHub. However, as described before, the Wi-Fi transmitter was insufficiently powerful and the project was forced to resort to wired transmission, and code modifications were made to ensure the board could conduct basic functions. As such, testing with having the FPGA transmit a basic square wave with the AD2 as the signal generator and receiver, then testing with Wi-Fi bits, then testing with codeword translation. Only after ensuring that all of the board was functioning correctly and determining the major limitations of the backscatter system, the various excitation signals were tested against each other for the final



(a) Backscatter General Setup



(b) Backscatter System Implementation

Figure 4.6: Backscatter Overall Implementation

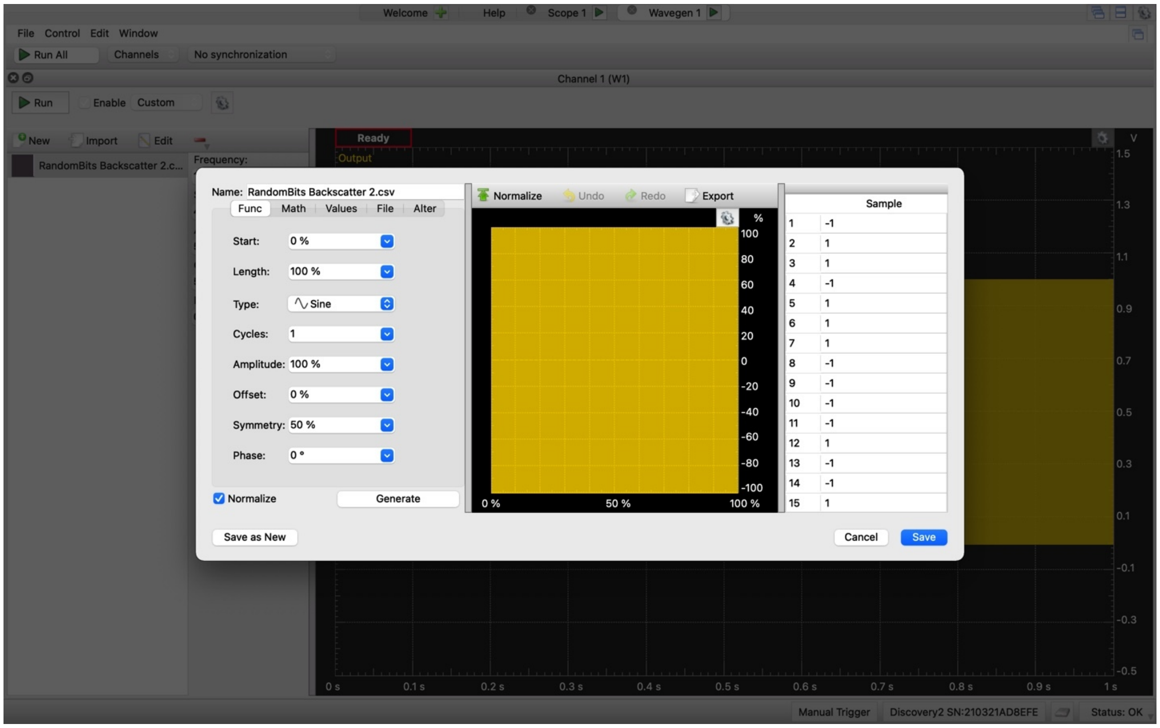


Figure 4.7: Sample 802.11b Packet in WaveForms

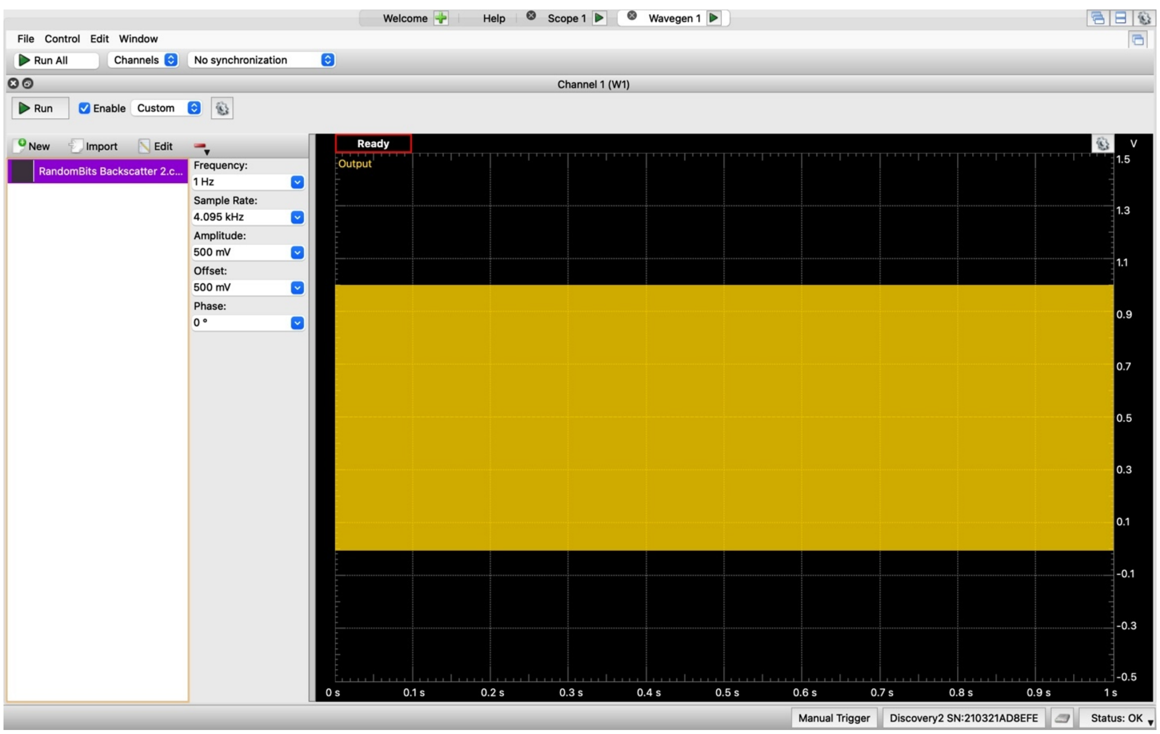


Figure 4.8: 802.11b Packet Configuration in WaveForms

evaluation.

For evaluation of the backscatter board, the values of the variable resistor and minimum trigger voltage from the variable resistor were measured to determine the effect of various excitation signals on the backscatter board. To simulate various excitation signals, many sample trigger signals were generated in WaveForms for the AD2 to transmit to the board including Wi-Fi 802.11b, DC signals, and square waves. This can be seen from Fig. 4.7, where a sample 802.11b packet was generated in MATLAB and transferred to the WaveForms software to be generated. In Fig. 4.8, the waveform is generated and ready to run at any specified frequency, sample rate, amplitude, offset, and phase offset. The same could be said for the square wave as shown in Fig. 4.9, and the DC signal as shown in 4.10.

4.4 Ensuring Board and Software Functionality

When using the backscatter board, the first thing to do is to ensure it functions correctly. To do so, we first observed each pin on the backscatter system using an oscilloscope in order to determine if it output expected results without the presence of received signals (i.e., the various clock pins or data pin). Then, the overall functionality of the board was tested to guarantee the backscatter system correctly operates. Namely, we had to ensure the board would activate only in the presence of an excitation signal, perform codeword translation, and transmit the backscattered bits. To ensure the board could successfully transmit its data and implement codeword translation, we generated an 802.11b Wi-Fi packet through MATLAB, performed codeword translation on the MATLAB-generated Wi-Fi packet, and verified the validity of the output from the transmitter/data pin. As can be seen from Fig. 4.11, The blue signal is the output of the data pin which sends the signal to the transmitter of the backscatter board. These are the backscattered bits that have gone through codeword translation and are to be transmitted and decoded at the receiver. The blue signal is a one-to-one copy of the white bits on top, the green bits are the data the backscatter wishes to transmit, and the yellow bits represent the generated MATLAB packet. However, as can be noticed, some bits are highlighted in red to better demonstrate where codeword translation is taking place. From the previous chapter, [12] mentioned that with codeword translation, if the backscatter board wishes to transmit a '0', then it would retransmit the Wi-Fi bits, and if the backscatter board wishes to transmit a '1', then the backscatter board would flip the Wi-Fi bit to be transmitted.

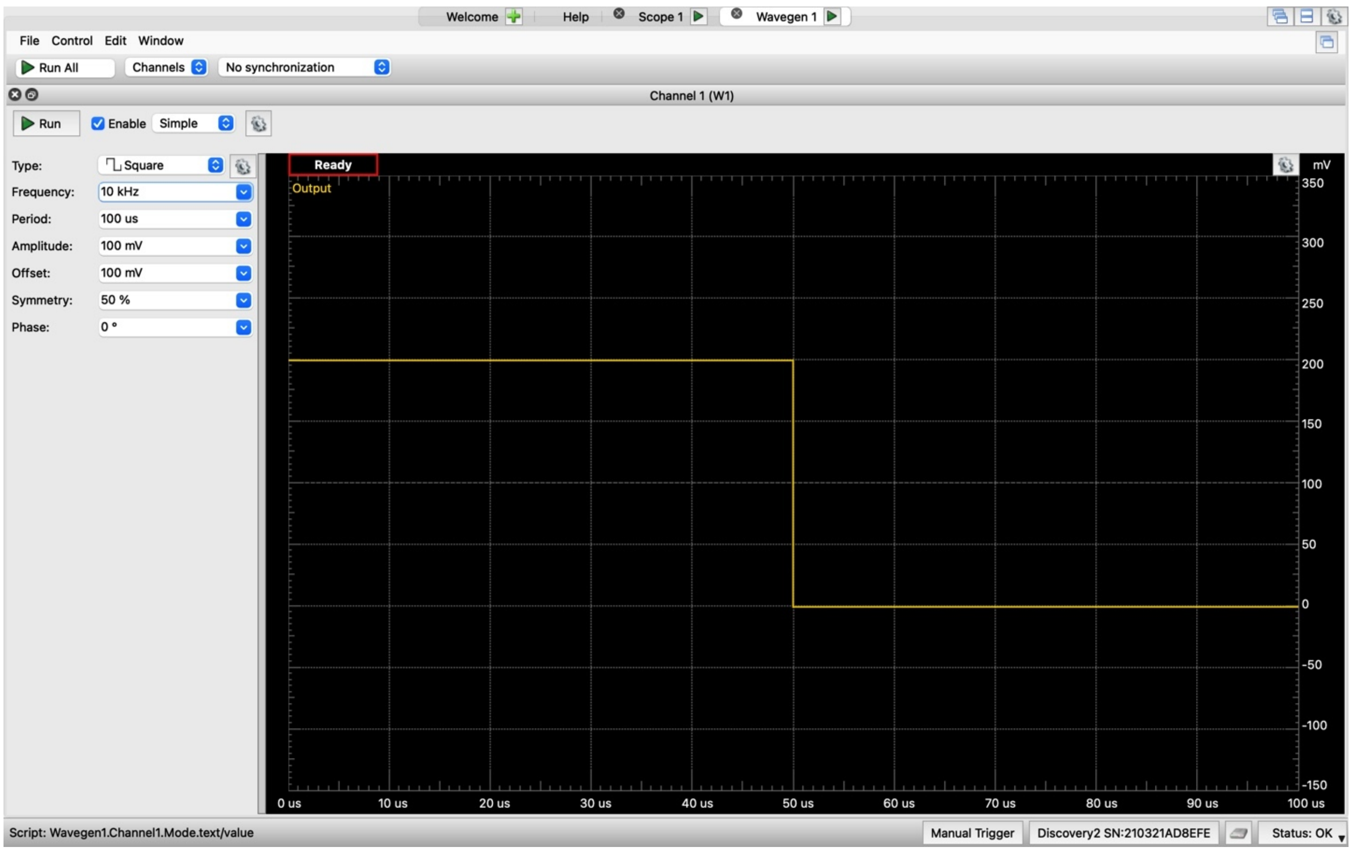


Figure 4.9: Square Wave from WaveForms

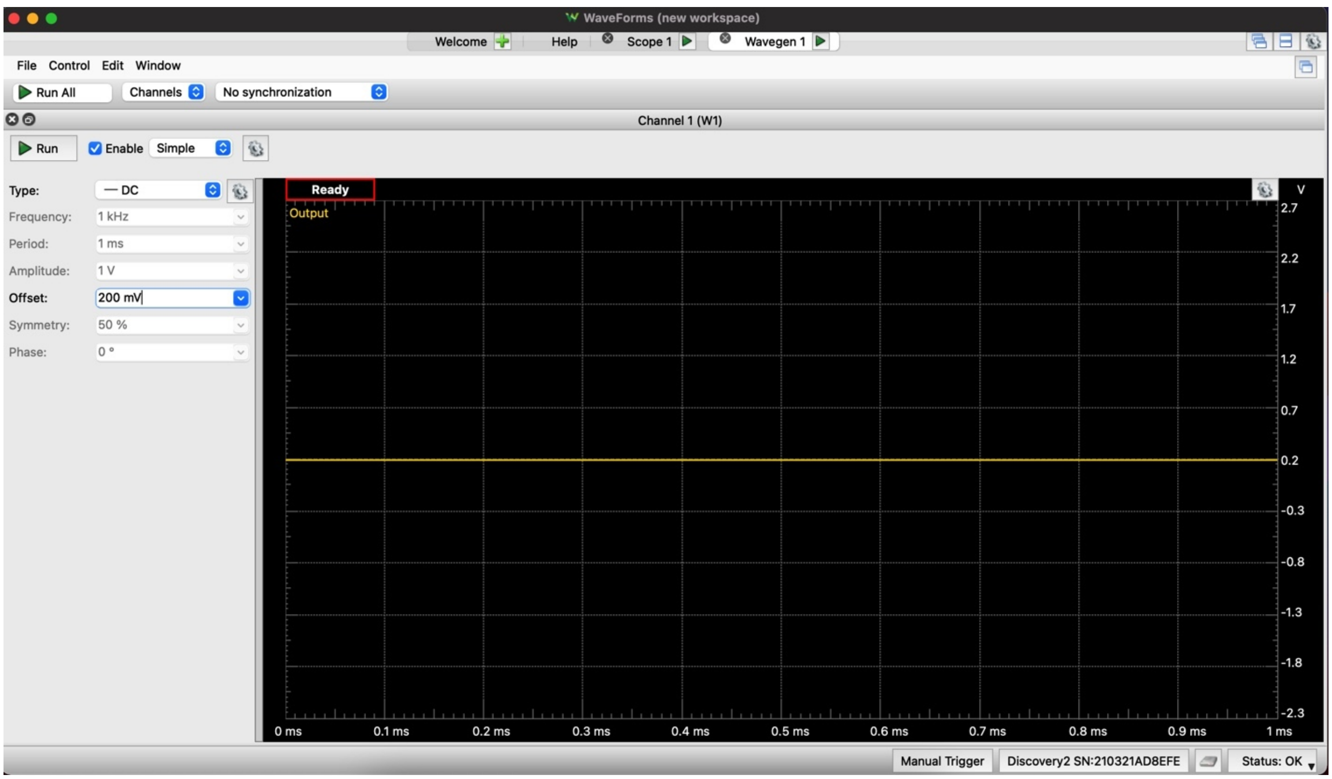


Figure 4.10: DC Signal from WaveForms

This experimentation was conducted with multiple test signals and backscatter data to corroborate our findings. The signals were correctly produced and translated, as can be clearly seen from Fig. 4.11 where the outputted backscattered bits completely match the signal as well as the codeword translation, demonstrating that the coding and hardware work as intended.

After analyzing the output from the Data pin, the transmitter's output was also analyzed, which can be seen from Fig. 4.12, where the data pin output is the blue signal, and the transmitted signal is the lower power flipped version in yellow. This is due to the data going into an RF switch which flips the input before transmission, creating a flipped version of the backscattered bits.

4.5 Trigger Thresholds

The main point of evaluation in this thesis is the trigger threshold for the backscatter system through the use of various types of excitation signals. Although the backscatter system is designed to output 802.11b baseband signals, it is able to receive any kind of signal, including non-802.11b signals. As such, the backscatter system is easily influenced by any and all signals that it can receive. In that regard, the variable resistor which controls the trigger voltage comparator is one point of contest. The resistance provided by this variable resistor is able to be adjusted through the use of a flathead screwdriver. Additionally, the minimum trigger voltage set by the variable resistor itself is another point of comparison to be judged among the different types of excitation signals. For the sake of this evaluation, an 802.11b Wi-Fi packet, a square wave, and a DC signal as a control signal, have been judged against each other to determine their effectiveness at triggering the backscatter system as well as their effects on the board itself. As these signals were to be generated through the use of the AD2, their frequency output was also limited by the AD2, as the software could only generate custom signals up to 10 MHz. However, through manually editing the samples, such that multiple bits were output at a select frequency (i.e., 4000 bits), we created a much higher frequency signal than was theoretically supposed to be generated from the AD2 as shown in Fig. 4.7, creating an 802.11b format signal with up to 40 MHz, which although was lower frequency than an actual 802.11b signal with 2.4 GHz, was the highest reasonable frequency we could generate with the equipment given. Other than the 802.11b signal, the other signals tested utilized normal signal generation from the AD2.

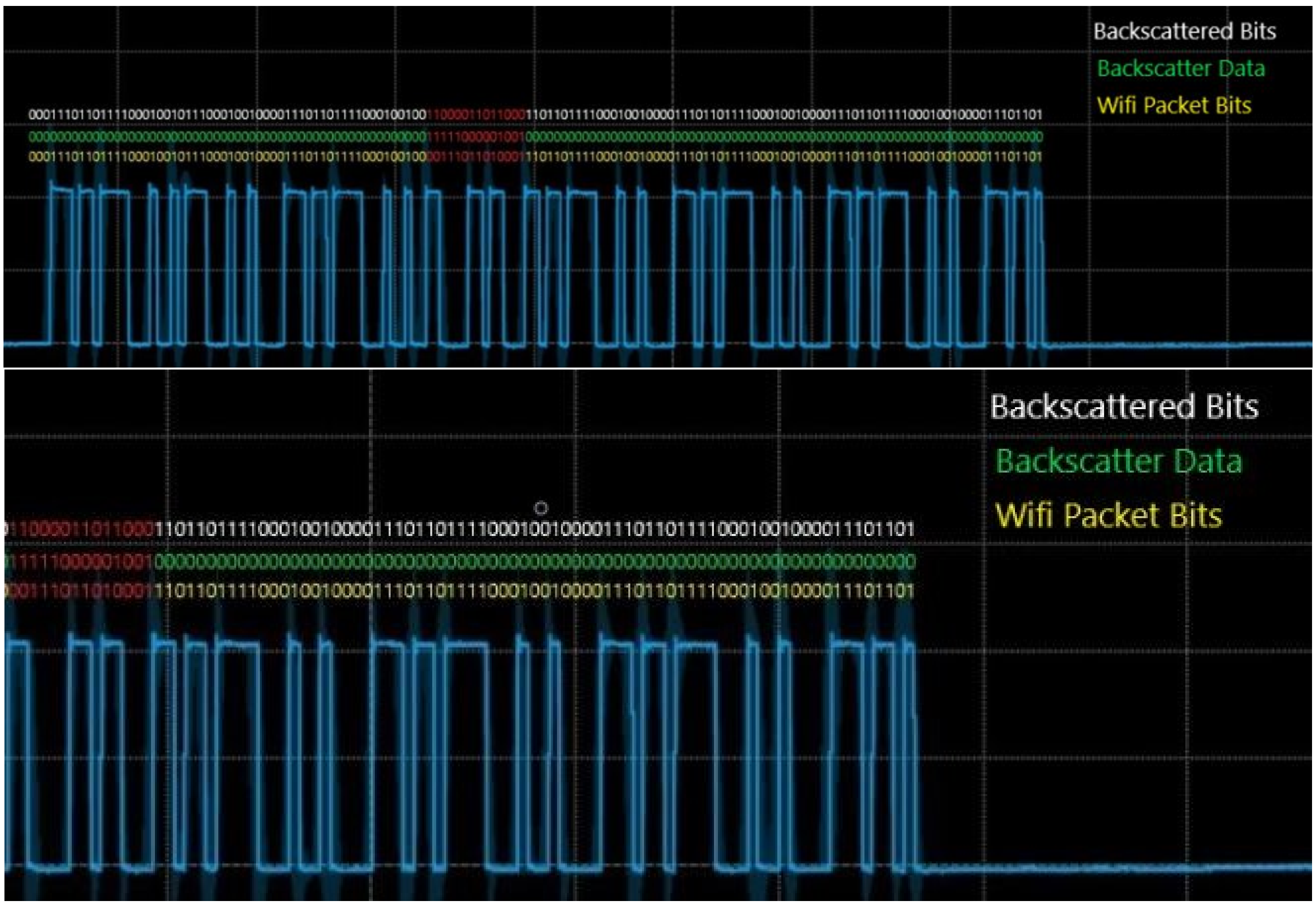


Figure 4.11: Backscattered bits from the data pin

Figure 4.12: Backscattered bits from the data pin and transmitter

Chapter 5

Evaluation

In this chapter, results will be discussed to evaluate the influence of different excitation signals on the backscatter system. The results provided are those that provided useful comparisons and meaningful discussion.

5.1 Trigger Thresholds

In these results, various excitation signals were compared to determine how they would affect the backscatter system and at what point they would trigger the backscatter system to begin transmission. Within the backscatter system, an adjustable variable resistor with resistance R_t (to signify trigger resistance) controls voltage leading into a comparator, which will be called V_t (to signify trigger voltage). Therefore, with a higher R_t, a higher V_t is created and the harder it will be to trigger the board. The results gathered are such that the highest for each excitation signal, the highest V_t and R_t are recorded such that the resulting transmitted signal from the backscatter is clear. This is because any V_t and R_t below the recorded threshold will still trigger the board (lower trigger value), and any higher value will not keep the trigger on.

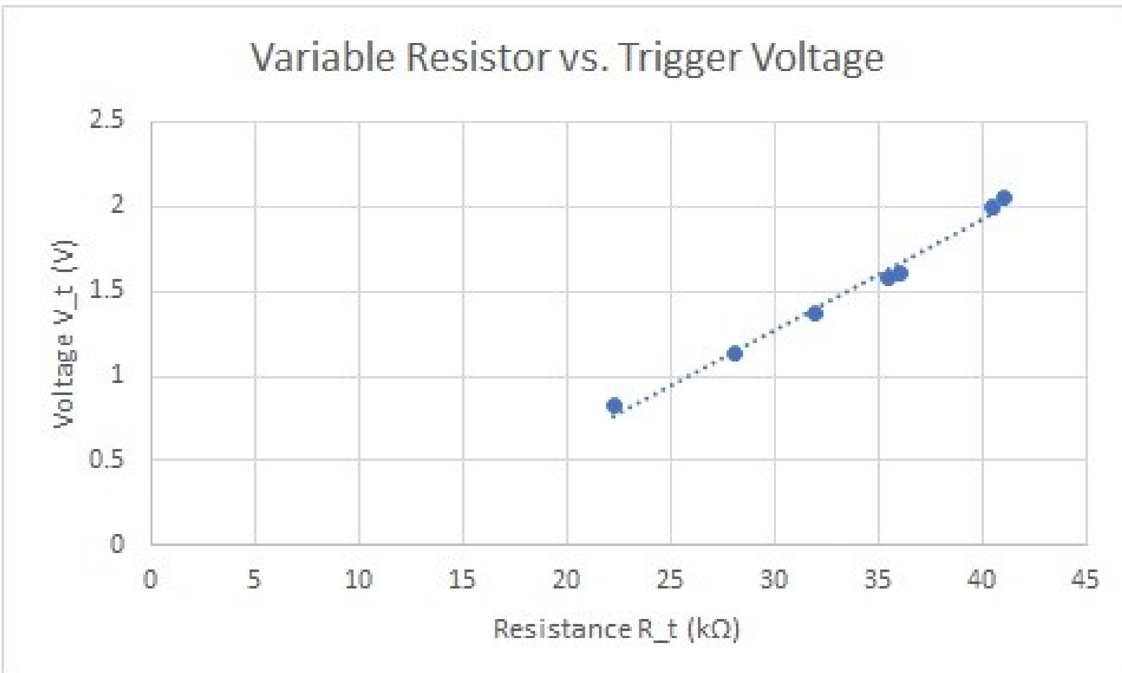


Figure 5.1: Variable Resistor vs. Trigger Voltage

5.1.1 Trigger Voltage to Variable Resistor Relation

The relation between the variable resistor and the trigger voltage can be seen from Fig.5.1, where from the data collected, we have an approximately linear relationship, where the trigger voltage increases with the variable resistor.

5.1.2 Excitation Signal Comparison

As described in the previous chapter, the types of excitation signals tested were the DC voltage, square, and 802.11b signals. With a high trigger voltage configuration from the backscatter board, only signals that provide a high comparative voltage would be able to act as an excitation signal, and with a low configured trigger voltage from the board, any signal above that trigger threshold would be able to excite the backscatter system. Therefore, the lower higher the voltage requirement shown in Fig. 5.3, the higher the comparative voltage the signal is able to generate to sufficiently trigger the signal. Fig. 5.3 displays our tested signals at various transmit voltages and frequencies. As can be seen and as predicted, the control test of DC signals triggered the board with the highest minimum trigger voltage. The trigger voltage stayed the same at 100 mV, 200 mV, and 400 mV DC signals due to the maximum trigger voltage being approximately 2.1 V. On the other hand, as can be seen from the figure, the 802.11b 4kHz as well as the 5 kHz square wave had the same trigger voltages no matter the transmit voltage. This was due to the 802.11b signal acting as essentially some kind of square wave and the lower frequency making the transmit voltage have less of an impact on the trigger voltage. Finally, at high frequencies, it can be seen that the transmit voltage actually makes a difference, where the trigger voltage increases with the transmit voltage as expected. Next is the resistance controlling the trigger voltage. As can be seen in Fig. 5.1, the resistances essentially show a linear relation to the voltage where the higher the variable resistance, then the higher the comparator voltage due to the backscatter system's setup. Due to this, the relations between the variable resistor and the excitation signal types are largely the same as the relations with voltage, shown in Fig. 5.2.

The excitation signals have been ranked in table 5.1 from the easiest to trigger the backscatter to the hardest to trigger the backscatter system. Essentially, if the backscatter system were to

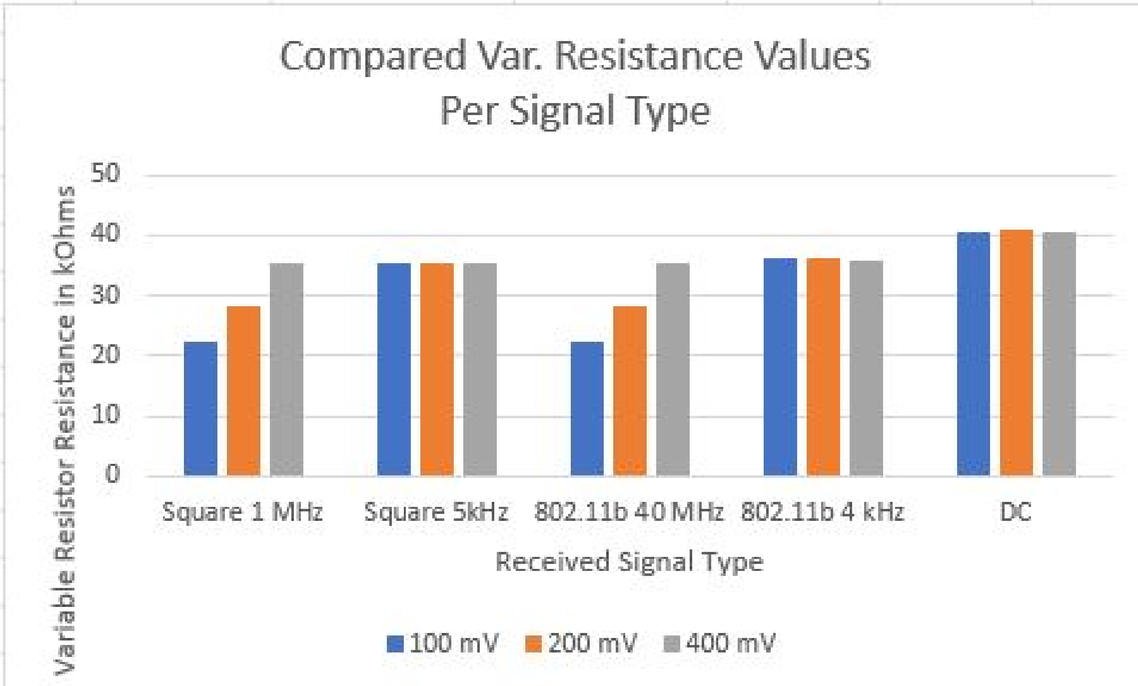


Figure 5.2: Comparison of Variable Resistor Values

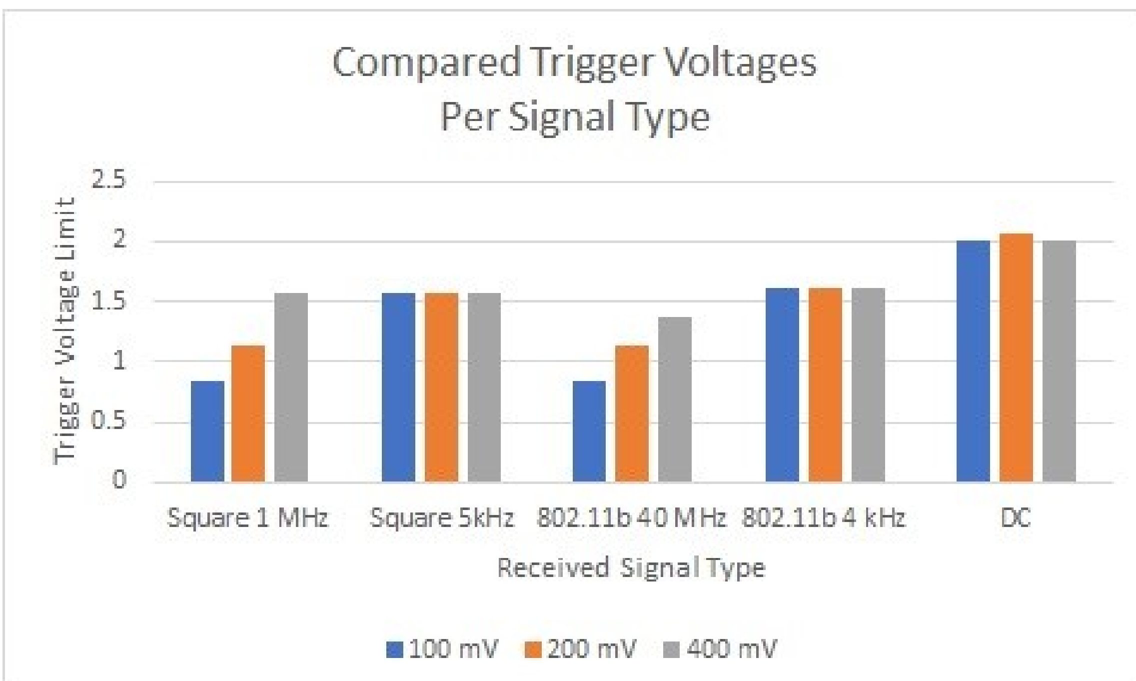


Figure 5.3: Excitation Signal Threshold Voltage Comparison

receive an excitation signal of higher rank, then it be more likely to trigger the system, while lower ranks would be less likely to trigger the system. My assumption as to the reason for the ordering is that at lower frequencies, the signals would stay 'high' for longer periods of time, essentially triggering the backscatter system longer even though it will also be 'low' for longer. Due to capacitors in the system, the 'high' trigger has a delay in turning 'low', making the longer 'low' periods less impactful. Therefore, the DC signal has the highest ranking, then the low frequency signals, then high frequency signals. If the backscatter system was to be triggered wirelessly than wired, as in the experiment, as can be seen from Table 5.1, High Frequency WiFi would have the hardest time triggering the backscatter system, as almost any lower frequency or higher power signal would trigger the backscatter system before the WiFi 802.11b could trigger the system. Furthermore, the backscatter would have to be tuned extremely finely such that it isn't triggered by signals in a similar frequency band, as the received signal strength makes a large difference at high frequencies, and signals in similar frequency bands have either similar on indistinguishable trigger values if they have similar signal strength.

Table 5.1: Excitation Signal Ranks in Triggering Backscatter System

Ranking	Signal Type
1	DC
2	Low Freq WiFi
3	Low Freq Square
4	High Freq WiFi
5	High Freq Square

5.2 Other Testing Attempts

With regards to our evaluation of the backscatter system created by [12], the system produced a vast number of failed results, vastly limiting the tests able to be performed. Among them, a few with significance to this evaluation will be discussed.

5.2.1 WiFi Transmission Attempts

When evaluating the trigger of the board when attempting to utilize antennas to transmit WiFi 802.11b, the backscatter system was unable to be affected by the transmission due to the low power of the WiFi transmission as well as the large amount of background noise, as the backscatter system's implementation made it extremely sensitive to noise. For example, the use of the oscilloscope, multimeter, or even a screwdriver had the potential to trigger the backscatter system. On the other hand, when attempting to transmit the WiFi 802.11b and tuning the backscatter board's variable resistor, the trigger voltage would either be too high for the WiFi signal to trigger, or too low such that the backscatter would be triggered by all of the noise other than the WiFi signal. This could be easily noted from Fig.4.4, where the yellow LED would light up once triggered. As such, either the LED would stay dormant (no trigger) no matter the transmission of WiFi 802.11b, light up in the presence of external forces (noise), or stay on (always triggered). This can be seen clearly from Fig. 5.4, where a screwdriver being present near the antenna of the backscatter system causes the backscatter system to mistake an excitation signal, as the trigger threshold has been tuned so low in order to pick up the WiFi signal, that almost anything will trigger the system.

As described in the previous section, frequency and signal strength play a large part in triggering the backscatter system. Due to the presence of at least one signal that had a far lower frequency or higher signal strength, the low strength 802.11b signal we transmitted could not be utilized as a trigger. Therefore, no significant results could be gathered through the use of wireless transmission in the backscatter implementation, as there was no way to filter out the 802.11b WiFi signal.

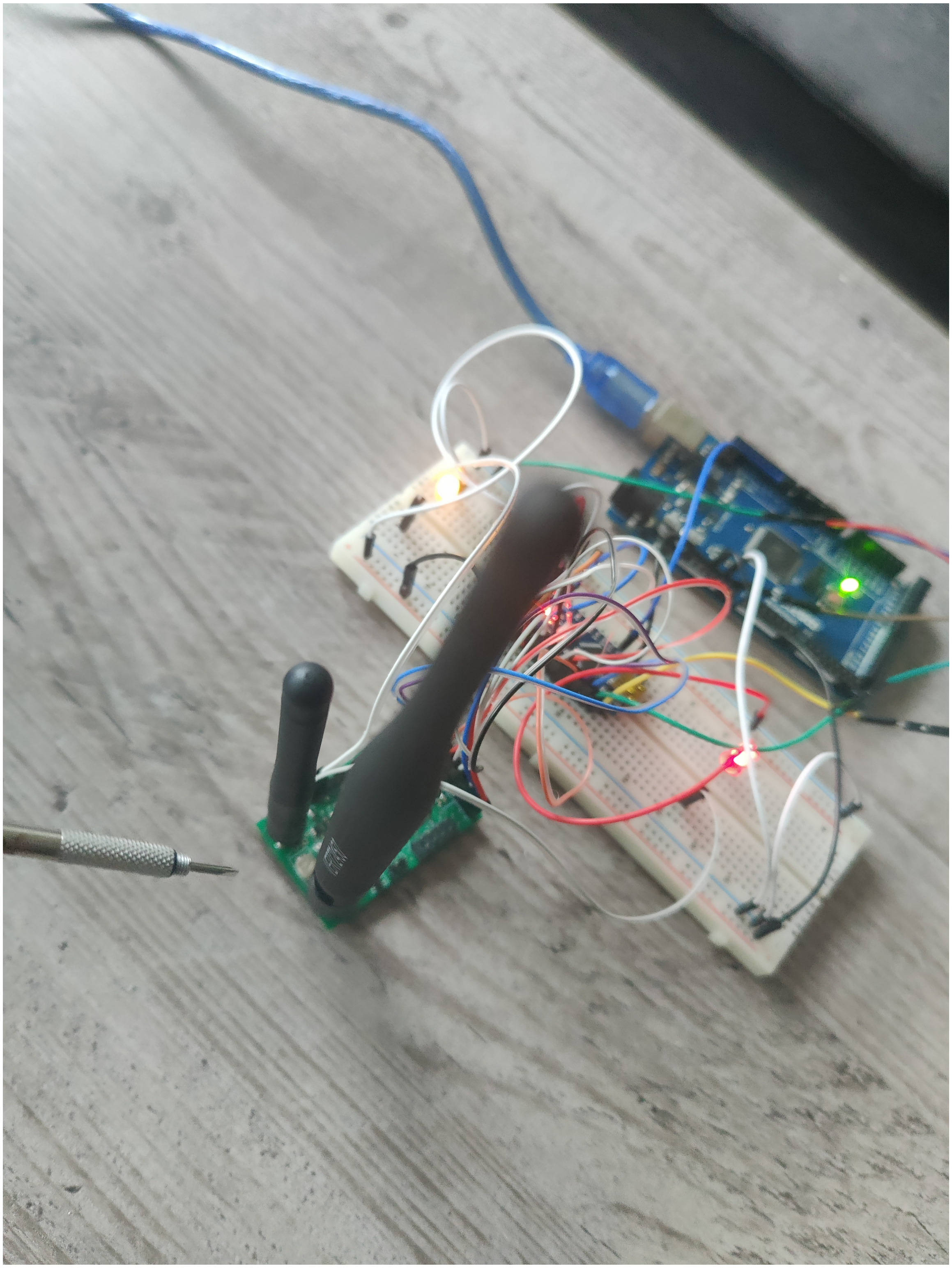


Figure 5.4: Incorrectly Triggered Backscatter Demonstration

Chapter 6

Conclusions and Discussion

Backscatter systems are a convenient low power solution to IoT sensing and communications. Through the use of ambient WiFi signals, a backscatter tag can reliably transmit data for close to zero power usage, allowing for device longevity. A system pipeline for the backscatter implementation involves transmitting an excitation signal, performing codeword translation on the WiFi 802.11b, backscattering the signal and re-transmitting it, and receiving/decoding the signal. It is shown that the backscatter tag being evaluated is still a demo board with limited backscattering functionality, but can be configured to be triggered by specific signal types with varying signal strengths. This thesis demonstrates that the demo board from [12] performs poorly if the WiFi transmission is low power and in the presence of noise, as false triggers will occur.

Future work would explore the use of a new, smaller backscatter board that would be able to fully backscatter signals with limited power usage and different types of ambient signals. Cheaper, dedicated backscatter systems can be tested in the future utilizing different signals such as Zigbee, LoRa, Bluetooth, or even all ambient signals, where the pros and cons of each type of implementation could be tested against each other. Through the development of new backscatter systems, IoT solutions can become more efficient with reliable communication systems.

Appendices

Appendix A   Codeword translation and Excitation code

```
module flipnew(trigger, clock, reset, swave );
input clock, reset, trigger;
output reg swave;

reg[26:0] counter;
reg[10:0] counter2;
reg[10:0] counter3;
reg[215:0] dataa;
reg[215:0] wifi;

initial begin
    swave = 1'b0;
    counter = 27'b0;
    counter2 = 11'b0;
    counter3 = 11'b0;
    wifi = 216'b00000000000000000000000000000000000
    000000000000000000000000000000000000
    010110111000010010001111011011100001
    001000111101101110000100100011110110
    111000101101110000100100011110110111
    000010010001110100100011110110111000;
    dataa = 216'b00000000000000000000000000000000000
    000000000000000000000000000000000000
    000000000000000000000000000000000000
    000000000000000000000000000000000000
    000100100000111110000000000000000000
    0000000000000000000000000000000000000;
end
```

```
always @(posedge clock or negedge reset)
begin
    if(!reset)
    begin
        counter<=0;
        counter2 <= 0;
        counter3 <= 0;
        swave <= 0;
        wifi <= 216'b0000000000000000000000000000000000000000
        0000000000000000000000000000000001011011100001001000
        1111011011100001001000111101101110000100100011110110
        1110001011011100001001000111101101110000100100011101
        00100011110110111000;
        dataa <= 216'b000000000000000000000000000000000000
        0000000000000000000000000000000000000000000000000
        0000000000000000000000000000000000000000000000000
        0000000000001001000001111100000000000000000000000
        000000000000000000000000000000000;
    end
    else if(trigger)
    begin
        if(counter == 2)
        begin
        counter <= 0;
            if (wifi[counter3])
            begin
                if(dataa[counter2])
                begin
                    swave <= ~wifi[counter3];
```

```
        end
        else
        begin
            swave <= wifi[counter3];
        end
        counter2 <= counter2 + 1;
        counter3 <= counter3 + 1;
        if(counter2 == 216)
        begin
            counter2 <= 0;
        end
        if(counter3 == 216)
        begin
            counter3 <= 0;
        end
    end
    else
    begin
        if(dataa[counter2])
        begin
            swave <= ~wifi[counter3];
        end
        else
        begin
            swave <= wifi[counter3];
        end
        counter3 <= counter3 + 1;
        counter2 <= counter2 + 1;
        if(counter2 == 216)
        begin
```

```
                    counter2 <= 0;
                end
                if(counter3 == 216)
                begin
                    counter3 <= 0;
                end
            end
        end
        else
        begin
            counter <= counter + 1;
        end
    end
    else
    begin
        swave <= 0;
        counter<=0;
        counter2 <= 0;
        counter3 <= 0;
    end
end

endmodule
```

Bibliography

[1] Dinesh Bharadia, Kiran Raj Joshi, Manikanta Kotaru, and Sachin Katti. Backfi: High throughput wifi backscatter. *SIGCOMM Comput. Commun. Rev.*, 45(4):283–296, aug 2015.

[2] Joshua F. Ensworth and Matthew S. Reynolds. Ble-backscatter: Ultralow-power iot nodes compatible with bluetooth 4.0 low energy (ble) smartphones and tablets. *IEEE Transactions on Microwave Theory and Techniques*, 65(9):3360–3368, 2017.

[3] IEEE. 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*, pages 1–96, 2000.

[4] Bryce Kellogg, Vamsi Talla, Joshua R. Smith, and Shyamnath Gollakot. Passive wi-fi: Bringing low power to wi-fi transmissions. *GetMobile: Mobile Comp. and Comm.*, 20(3):38–41, jan 2017.

[5] Taekyung Kim and Wonjun Lee. Anyscatter: Eliminating technology dependency in ambient backscatter systems. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pages 287–296, 2020.

[6] Xiaolin Mou and Hongjian Sun. Wireless power transfer: Survey and roadmap. In *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*, pages 1–5, 2015.

[7] Koustav Routh and Tannistha Pal. A survey on technological, business and societal aspects of internet of things by q3, 2017. In *2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU)*, pages 1–4, 2018.

[8] Vamsi Talla, Mehrdad Hessar, Bryce Kellogg, Ali Najafi, Joshua R. Smith, and Shyamnath Gollakota. Lora backscatter: Enabling the vision of ubiquitous connectivity. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 1(3), sep 2017.

[9] Ricardo Torres, Ricardo Correia, Nuno Borges Carvalho, Spyridon Daskalakis, George Goussetis, Yuan Ding, Apostolos Georgiadis, Aline Eid, Jimmy Hester, and Manos M. Tentzeris. Backscatter communications. *IEEE Journal of Microwaves*, 1(4):864–878, 2021.

[10] Nguyen Van Huynh, Dinh Thai Hoang, Xiao Lu, Dusit Niyato, Ping Wang, and Dong In Kim. Ambient backscatter communications: A contemporary survey. *IEEE Communications Surveys and Tutorials*, 20(4):2889–2922, 2018.

[11] Chaochao Yao, Yang Liu, Xusheng Wei, Gongpu Wang, and Feifei Gao. Backscatter technologies and the future of internet of things: Challenges and opportunities. *Intelligent and Converged Networks*, 1(2):170–180, 2020.

[12] Pengyu Zhang, Dinesh Bharadia, Kiran Joshi, and Sachin Katti. Hitchhike: Practical backscatter using commodity wifi. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, SenSys '16, page 259–271, New York, NY, USA, 2016. Association for Computing Machinery.

[13] Pengyu Zhang, Colleen Josephson, Dinesh Bharadia, and Sachin Katti. Freerider: Backscatter communication using commodity radios. In *Proceedings of the 13th International Conference on Emerging Networking EXperiments and Technologies*, CoNEXT '17, page 389–401, New York, NY, USA, 2017. Association for Computing Machinery.

[14] Zenghua Zhao, Xuanxuan Wu, Xin Zhang, Jing Zhao, and Xiang-Yang Li. Zigbee vs wifi: Understanding issues and measuring performances of their coexistence. In *2014 IEEE 33rd International Performance Computing and Communications Conference (IPCCC)*, pages 1–8, 2014.