RFIDIoT: RFID as the data link layer for the Internet of Things

by

Erick William Fuentes

S.B. Aeronautics and Astronautics (2011)
Massachusetts Institute of Technology

Submitted to the Department of Mechaninal Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2015

Signature redacted

Author .....
Department of Mechaninal Engineering
August 19, 2015

Signature redacted

Certified by .....
Sanjay E. Sarma
Professor of Mechanical Engineering
Thesis Supervisor

Signature redacted

Accepted by .....
David E. Hardt
Chairman, Department Committee on Graduate Theses

RFIDIoT: RFID as the data link layer for the Internet of Things

by

Erick William Fuentes

Submitted to the Department of Mechaninal Engineering
on August 19, 2015, in partial fulfillment of the
requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

In this thesis, the suitability of RFID as a platform for real time applications in the Internet of Things is investigated. Two sample applications of RFID technology identify throughput and latency as key figures of merit. An experiment to measure these parameters in an existing implementation is devised and executed. The sources of latency are identified with a high degree of accuracy. Recommendations for improvement of the existing implementation are guided by the latency measurements.

Thesis Supervisor: Sanjay E. Sarma
Title: Professor of Mechanical Engineering

Acknowledgments

The amount of that help I have received to get me to where I am today is so much more than I deserve. It isn't my intention to diminsh what everyone has done for me but I wanted to especially thank the following people who most closely involved in this endeavor:

First, I want to thank my advisor, Professor Sanjay Sarma. He has provided me unbridled freedom to pursue my interests inside and outside of graduate school. From building autonomous robots and exploring entreprenuership to allowing me to participate in a summer internship, I now realize how lucky I really am.

I also want to thank my labmates: Jason Ku, Yongbin Sun, Pranay Jain, Partha Bhattacharrjee, Josh Siegel, Dylan Erb and Stephen Ho. You have made my time here so much more enjoyable than I expected. Isaac Ehrenberg, Rahul Bhattacharyya and Khaled ElMahgoub were especially invaluable as sounding boards and in providing new directions to explore when I thought I had reached a dead end.

I want to thank my sisters, Sandy and Claudia, and my girlfriend, Katie Inman, for taking part in my antics and being there when I most needed it.

Finally, I want to thank my parents, Jose and Martha. There are no words that can express how deeply, truly thankful I am for the sacrifices that they have made for the mere possibility of providing better lives for my sisters and I. Their tireless work ethic, boundless selflessness, and limitless curiosity are unending sources of inspiration which drive me to do more. I can only hope to make their sacrifices worthwhile by making them proud.

Contents

List of Figures

Chapter 1

Background and Motivation

1.1 The Internet of Things

The Internet of Things (IoT) is a network that allows objects to be remotely sensed and operated. This connectivity has ushered in a new age of automation that impacts all aspects of human life and industry. A smart grid enabled by the IoT dynamically adjusts power generation as demand shifts over time and across regions. Connected vehicles monitor their own state of health and communicate with other vehicles on the road to mitigate accidents and traffic jams. Health tracking devices empower their users and the users' doctors to make the best decisions to improve overall well being. It is expected that this network will encompass 30 billion devices in 2020, driving $6.2 trillion in value in 2025 [1].

One of the key enablers of the IoT is low power wireless technologies that have been developed in the last 20 years. These technologies enable connectivity without requiring a mechanical connection. The first of these modern low power networks is the IEEE 802.11 standard which was initially released in 1997 and provides the specification for Wi-Fi. IEEE 802.15.1 defined Bluetooth, and 802.15.4 is the basis for ZigBee and 6LoWPAN. In addition, there are other wireless protocols spearheaded by various industry players such as ANT and Z-Wave. Finally, the proliferation of cellular networks across the country has provided connectivity out in the field.

The exploding proliferation of these networks allows devices to connect at a wide

variety of ranges and power. However, all of these solutions suffer from the same drawback; they require all remote nodes to have radio transmitters. Radio transmitters drive up the cost of the remote node not only due to the expensive radio frequency (RF) components but also due to the increased power requirements that necessitate the use of onboard power storage or external power.

For high valued goods, like mobile phones and automobiles, where the added expense is a small part of the bill of materials, this cost can be justified. However, there is the potential for great value to be derived from connecting a large number of low cost devices. These low margin devices cannot afford to include an active radio and as a result must seek alternative solutions.

1.2 Backscatter Communication

In a backscatter communication system, a single transmitter provides the power for communication to all remote devices in its field of view. In order to enable communication, each node only requires a means to reflect power, a much lower cost solution. In addition, if the devices consume a small amount of power, it is possible for the node to harvest all of its power from the incident power. Backscatter communication challenges the notion that an active transmitter is required to participate in the IoT by significantly lowering the cost to communicate. It is a cost effective way of bringing a very large number of devices online.

A number of backscatter radio communication standards have been developed to meet the needs of different applications and industries, but the most successful implementation is the GS1 radio frequency identification (RFID) Air Protocol accepted by the International Organization for Standards (ISO) as ISO 18000-6 . As its name implies, the primary purpose of this protocol is to cheaply and cost effectively identify objects. The remote devices are known as tags and they communicate an identifying number when prompted. An interrogator serves as a power source and facilitates communication between the tags and the clients who benefit from the identification of items. By shifting the cost of communication to a single device, the marginal cost

of identifying a new device approaches cents as the cost of a tag falls. RFID has seen adoption by retail industry giants, causing the price of a tag to drop well below $0.25 USD [2].

With 8 billion devices connected the internet in 2013 [1], the combination of low cost, low power and wide adoption make RFID a very attractive platform on which to connect the next 22 billion devices.

Given the potential of RFID's role in the IoT, this thesis is structured as follows: Chapter 3 will explore how RFID is used. Chapter 2 will explain how backscatter communication works. Finally, Chapter 4 will explore how RFID can enable the IoT.

Chapter 2

Backscatter Fundamentals

This chapter will answer the following questions with enough depth so that the rest of the thesis can be understood: What problems can be solved by backscatter communication? What is backscatter communication and how does it work? How does RFID utilize backscatter communication? How does a client interface with an iterrogator?

2.1 Brief History

The manipulation of a backscattered radio signal as a means of conveying information dates back to World War II. In early attempts to distinguish friend from foe, German Luftwaffe pilots would roll their aircraft when illuminated by a radar system. This rolling maneuver changes how the radar energy reflects off of the aircraft, appearing as a blip on the radar operators screen [3]. While this system was very rudimentary, it demonstrates how radar backscatter could be used to distinguish objects. However, this system is very insecure as any aircraft can be rolled and the resulting backscatter does not identify the type of plane without further processing of the backscattered signal.

The solution was to equip all aircraft with transponders. When the planes were illuminated by radar pulses, the transponders, containing active transmitters, would respond back with an identifying number. An implementation of this system is used to this day on all aircraft. Air traffic controllers use these transponders to identify

aircraft in the area for which they are responsible. In addition, aircraft use these transponders for anticollision systems which operate independently of ground based systems. The use of these systems is critical to maintaining the safety of private and commercial aviation.

The need to automatically and cheaply identify objects is not unique to the aircraft industry. This problem exists in a number of industries ranging from the electronic toll collection to tracking of livestock. Due to the unique challenges in each instance of the automatic identification problem, different, incompatible solutions have been developed. These applications will be explored in Chapter 3.

In the late 1990's, research problems in robot localization and industry needs for supply chain management met in a way that caused the automatic identification problem to be approached in a generic and unified manner. This collaboration between academia and industry launched the MIT AutoID Center. The goal of this collaboration was identify and develop standards in order to practically identify objects. Three of the many innovations that came about are the electronic product code (EPC), the standardization of the communication between a tag and the interrogator, and the standardization of the communication between a client and an interrogator. In order to enable the development of neutral standards which do not favor a particular vendor or customer, the standards are maintained by an independent organization known as GS1.

2.2 Electronic Product Code

At the heart of the efforts of the AutoID Center is the concept of the EPC. It is a number that is assigned to every object, group of objects, location and document. On a tag, the EPC is stored as a 96 bit number. The GS1 Tag Data Standard [4] defines how this number should be interpreted. A number of bits are reserved to identify the format of the EPC, the company, and product that the EPC describes. As a result, although maximum number of items that could be identified with a 96 bit number is $7.923 \cdot 10^{28}$, the actual number objects that could be identified is fewer due to the use

of reserved bits.

2.3 Basic Operation

The most widely adopted GS1 RFID standard is in the Industrial, Scientific and Medical (ISM) ultra high frequency (UHF) band. In the Americas, the UHF ISM band has a center frequency of 915 MHz and a bandwidth of 26 MHz. At this frequency power transfer occurs through radiative coupling, as opposed to inductive coupling typically seen at much lower frequencies. The range as a result is much greater than what is seen in inductive coupling and power is radiated whether or not there is a remote device harvesting.

In order to develop an intuition for how RF radiation can be used in backscatter communication, it is useful to draw an analogy between the RF radiation and visible light. Imagine Alice, a ship captain and her crew, Bob, who has been washed overboard in a late night storm. Unfortunately, Bob is very difficult to see from the boat, even with the use of the onboard searchlight. Luckily, Bob remembers that he has a mirror is his pocket. When Alice points her spotlight at Bob, Bob uses the mirror to reflect some of the light back at Alice. Being well versed in morse code, Bob uses the mirror and starts to tell Alice that he is tiring and to come quickly. Alice sees the reflected light and recognizes it as morse code. Alice responds to Bob by turning the searchlight on and off, telling him to remain calm and that she will be right over.

Exploring the analogy a little further, it becomes clear that Bob cannot communicate with Alice at the same time that Alice is trying to communicate with Bob. A refinement of the analogy would be that instead of Alice seeing Bob reflecting light, she is observing the output of a single pixel camera, or a photodetector at which Bob aims his mirror. With this refinement, it becomes clear that if multiple people, Bob and Charlie, are overboard, they cannot talk to Alice at the same time as their communication would overlap at the photodetector. Finally, since Bob and Charlie are aiming their mirrors at the photodetector, they are not able to see when the other is attempting to communicate with Alice.

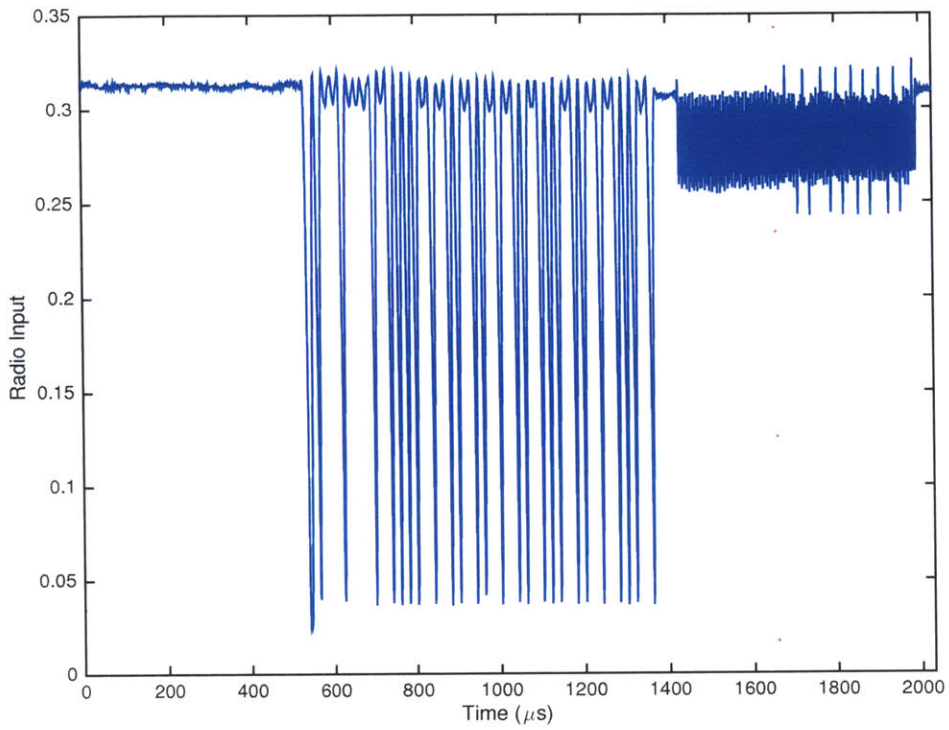


Figure 2-1: Example of backscatter communication

With this high level understanding, we can start to assign names to different parts of the analogy. Alice's searchlight and photodetector pair is known as the interrogator in an RFID system. It radiates power while listening for a tag and communicates with all tags simultaneously. Bob and Charlie's mirror is the analog of a tag antenna. It communicates with the interrogator by reflecting, or backscattering, power back to the interrogator. However, since the power reflected by the tag is much lower than that radiated by the interrogator, tags communicating via backscatter do not interfere with nearby tags attempting to communicate. Figure 2-1 shows what the output of the photodetector may look like. The high section on the left is the interrogator providing power for tags. In the middle section, the interrogator modulates between high and low power states communicating to all tags in its field of view. Finally, on the right side is the tag responding via backscatter.

In the analogy, Bob and Charlie sent information by physically moving the mirror. Tags achieve the same result by modulating its antenna's reflectivity with the help of the on board integrated circuit (IC). When the antenna is well matched, the incident

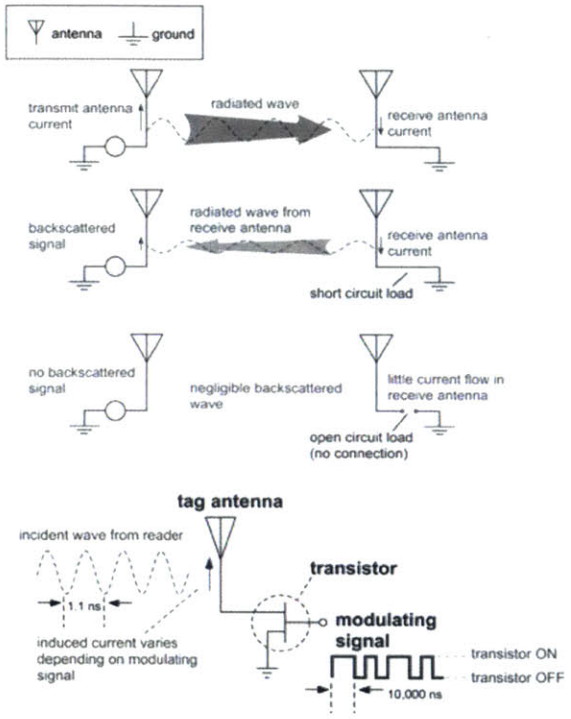


Figure 2-2: Tag Modulation of Backscatter (From: The RF in RFID [3])

radiation is absorbed and can be used by the IC to power internal logic. In order to reflect power, the antenna can be detuned by shorting it to ground. A transistor can be used to electrically switch the short circuit path on and off. Figure 2-2 shows how this works in practice.

2.4 GS1 UHF RFID Air Protocol

In order to exchange information, it is important to understand and define the symbols that will be used to facilitate this exchange. Interrogators use pulse interval encoding (PIE) to transmit information. As the name implies, the difference between a binary 0 and a binary 1 is how long the pulse is. The basic unit of timing is called a tari and corresponds to the length of a binary 0. A binary 0 is defined as a period high power followed by a period of low power. A binary 1 is defined as a high power pulse greater than a tari in length and a period of low power. Figure 2-3 depicts how these pulses are defined.

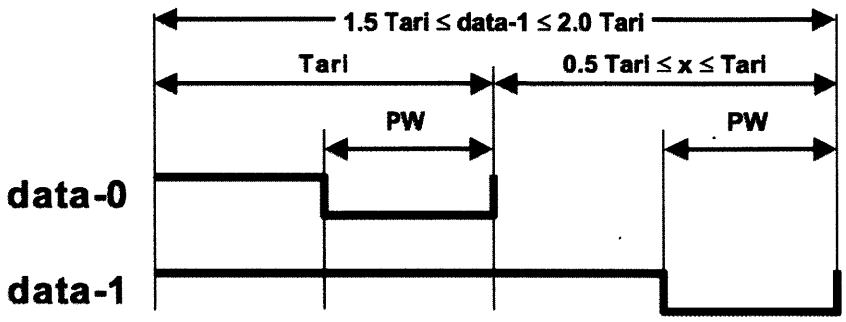


Figure 2-3: Interrogator Pulse Interval Encoding (From: EPC UHF Air Protocol Standard [5])

There are many more details involved in the interrogator/process, but they are not necessary for the understanding of the work in this thesis. For more details, please refer to Tom Scharfeld's thesis [6], the RF in RFID [3], and the GS1 Class 1 Generation 2 (C1G2) Air Protocol [5].

2.4.1 Inventory Process

The inventory process is how an interrogator determines what tags are in its field of view. Due to the fact that only the interrogator can see the tags, it is up to the interrogator to decide when each tag communicates with the interrogator.

The inventory process takes place in rounds. Using the query command, the interrogator announces how many slots there are in the current round, n. When the start of a round is announced, each tag will pick a random number that is between zero and $n-1$. After a certain wait period the interrogator will announce the start of a new slot. The tags will then decrement the random number they chose by one. When the random number is equal to zero, the tag will backscatter a random 16 bit number. If there are two tags that transmit a random number in that slot, the interrogator will detect the collision and ignore the backscattered signal. If only one tag responds, the interrogator will acknowledge the tag by responding with the random number that the tag backscattered. Once the tag has been acknowledged, it will backscatter its EPC. When all of the slots have been iterated through, a new round is started.

This procedure is known as the Q protocol and it is a variant of the slotted Aloha Protocol [6].

For this protocol to work efficiently, it is very import to pick the number of slots in a manner that is proportional to the number of tags that are present in the field of view. If the tag population is very large and the number of slots is very small, it is unlikely that there will only be one tag responding in a time slot, resulting in a large number of collisions. If the tag population is small and the number of slots is large, it will take a long time to complete an inventory round.

The number of slots present in a round is determined by the Q parameter after which the protocol is named [7]. It is a 4 bit number that ranges between 0 and 15. The number of slots is determined by 2^Q. This allows the number of slots to range from 1 slot all the way to 32768 slots, large enough to handle even the most demanding applications.

2.4.2 Interrogator Commands

With an understanding of the theory of operation of backscatter communication and how the inventory process works at a high level, the remaining portion of this chapter will delve into some of the implementation details of the GS1 RFID Air Protocol.

Select Commands

Before an inventory round begins, the interrogator has the ability to limit which tags can participate in the inventory round. For example, consider an interrogator mounted at the entrance of a warehouse. Pallets contain 20 boxes and each box contains 20 light bulbs. If the supply chain software only needs to know the EPC of the pallet, it would be very time inefficient to inventory each light bulb and box. By selecting only EPC's which correspond to pallets, the inventory process can be greatly expedited.

In a select command, the interrogator specifies a bit pattern and mask. A tag that matches the select command can be instructed to participate or abstain from

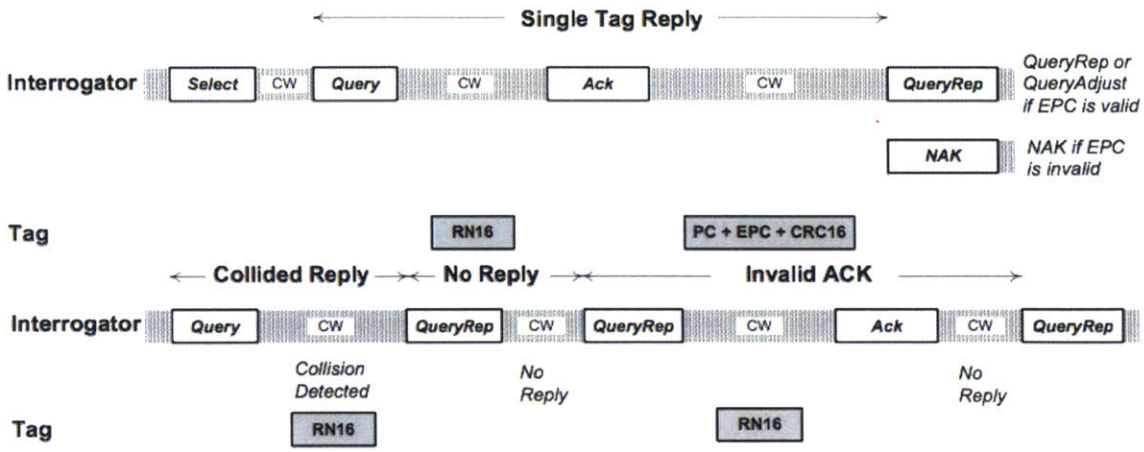


Figure 2-4: Inventory Process (From: GS1 C1G2 Air Protocol [5])

the following inventory round. By issuing multiple select commands, an interrogator can effectively winnow the tags that participate in an inventory round.

Inventory Commands

An interrogator has the following commands to control the inventory process: Query, QueryRep, QueryAdjust, ACK, NAK. Of these commands, the most relevant commands are the Query, QueryRep and ACK commands.

In a Query command, the interrogator announces the start of a new round, sets the number of slots in this round, and sets the communication parameters of the tag backscatter reply.

An interrogator uses the QueryRep command to delineate the slots in an inventory round. Issuing this command causes each tag to decrement their internal counter and backscatter a random 16 bit number, an RN16, when that counter reaches 0.

The ACK command is used by the interrogator to acknowledge a tag that has backscattered a random number. The interrogator will repeat this random number back to the tag. If the random number sent by the interrogator is the same as the random number sent by the tag, the tag will backscatter its EPC.

Figure 2-4 shows how these commands work with tag responses to efficiently transfer the EPC to the interrogator.

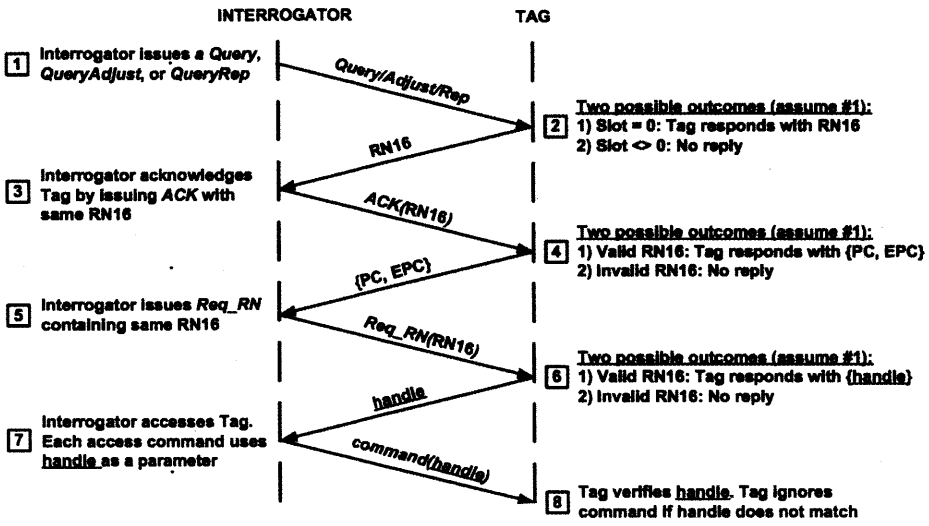


Figure 2-5: Tag Access Procedure (From: GS1 C1G2 Air Protocol [5])

Access Commands

Access commands allow the interrogator to access other parts of the onboard eeprom. In order to accomplish these tasks, the interrogator has the RequestRN, Read, Write, and BlockWrite commands.

The RequestRN command is used to have the tag backscatter a new random number. This number can be used as a handle to refer to the tag or it can serve as a key to encode information.

The Read command instructs the tag to backscatter a number of bits of its memory starting at a specified memory location.

The Write and BlockWrite commands instruct the tag to commit a 16 bit number to memory at a specified memory location. The difference between the Write and BlockWrite commands will be explained in the following section.

2.4.3 Reading and Writing Process

For any access command, the RequestRN command is sent to the tag, causing it to backscatter a new random number. This random number is used as the handle to reference the tag. This handle is used in all subsequent access commands. This

process is depicted in Figure 2-5.

For a Read command, the interrogator includes the address of where the read should start, the length of the read and the handle of the tag.

For a write, after the handle is requested, a new random number is requested. This random number is the key. This key is XOR'd with the data to be written. The reason for this is that the interrogator radiates a large amount of power which can easily be picked up outside of a building. Passive and semipassive tags, on the other hand, are quiet. This means that the key is less likely to be leaked if eavesdroppers are kept a reasonable distance away. As a result, an eavesdropper will not be able to decode what written to the tag. This process is repeated for every 16 bits that are written to the tags memory.

A BlockWrite is similar to a write but no cover coding is used. As a result, the writes tend to be faster because a new random number does not need to be requested for every word that is written.

2.5 Low Level Reader Protocol (LLRP)

LLRP defines how clients suchs as computers and mobile devices can communicate with a interrogator. However, not all interrogators implement LLRP; devices such as the Nordic ID Stix, the Impinj Indy RS400, and the Magic RF RFID module implement their own proprietary interfaces.

When dealing with a interrogator that communicates using the LLRP protocol, there are three messages that the user needs to be aware of. The first is the Reader Operation Specification (ROSpec). This defines the parameters of operation, such as transmit power, number of antennas, and data rates to use. The second message is the Access Specification (AccessSpec). This defines what memory regions on a tag to read and write. It is also possible to specify a target which allows the use to specify which tags to write to. The last message is the RO Access Report. Unlike the previous message which are sent form the client to the interrogator, this message is sent from the interrogator to the client. It contains information such as the EPC

of an observed tag, the time when that tag was seen and the recieved signal strength indicator (RSSI). It also contains the result of any completed reads and writes. Details of the LLRP protocol are available in the GS1 LLRP Specification [8].

Chapter 3

RFID Applications

Chapter 2 focused on how RFID works and this chapter will focus on how RFID is used. First, traditional use cases will be examined, then, potential future applications relying on RFID technology will be explored.

3.1 Standard use of RFID

The inventory process has been optimized to transmit the EPC as quickly as possible from the tag to the interrogator. Additional accesses to tag memory are considerably slower. As a result, a tag will rarely contain any information other than that which is required to identify the object. While applications in supply chain and inventory management are relatively well known, applications outside of these areas will be mentioned in order to show the versatility of RFID.

One of the earliest examples of the application of RFID technology is in electronic toll collection. Human operated toll booths are known to increase congestion because each car must stop for a small period of time. Increasing throughput requires more human operated toll lanes to be added, an expensive proposition. Automated toll lanes allow vehicles to travel through the toll gate, at times without stopping. As a result, fewer automated toll lanes are required to serve a given level of traffic and they are also cheaper to run, as no operator is required. The method of operation is very simple. An interrogator is mounted in the toll booth and it looks for a transponder

that is affixed to a car. The transponder will send its identifying number which the interrogator will forward to the electronic toll collection software. The software will associate the ID with an account and it will automatically deduct the toll. Automated toll collection also enables dynamic pricing, allowing the toll to vary with time of day or level of congestion.

RFID is also used in a variety of timing applications, such as marathons. For short distance races on a track a computer vision system will be able to determine who won the race. In larger races, such as the Boston Marathon, gun timing was used until 1996. Gun timing, or time after the starting gun goes off, is accurate for those who are near the start line, but as marathons draw more participants, it become less accurate for those near the back. As waves were introduced to prevent congestion on the course, this method of timing only became more inaccurate. RFID tags are now affixed to the runners' bibs, allowing the time that the runner crosses the start and finish to be accurately measured. With over 30,000 participants, RFID is the only scalable way to provide accurate timing information.

RFID technology is also used in libraries to manage their massive collections of books. Before RFID was implemented, the only information that librarians would have are whether the books are checked out, received and waiting to be sorted, or in the stacks. With visitors coming and going, it is not guaranteed that any book will be in the place that it belongs. To mitigate this issue, some libraries have affixed RFID tags to the inside cover of their books. This allows the books to be quickly scanned by a person who has a hand held interrogator. This is still very tedious and a libararians time is better spent helping people. Bookbot aims to solve this problem [9] by scanning on behalf of the librarian. Bookbot traverses the aisles in a library, cataloguing where books are in the library and is able to pinpoint any misplaced books with centimeter accuracy. The books can also be used to help bookbot determine its position within the library.

3.2 Changing Landscape

Since the standardization of RFID, there has been significant interest in wireless sensor and actor networks (WSAN). These networks are made of small nodes which range in cost from a few to several hundred dollars. They have some onboard computation power, a communications interface, a power management system, and some way to interact with the enviroment. This interaction occurs in two methods, sensing and actuation. Examples of common sensors are temperature, humidity, chemical concentration, light, and sound. The actors in this network are often mobile robots, such as R/C cars or unmanned aerial vehicles (UAV's). These actors could also be fixed modules, such as a a valve or air vent.

When RFID was first standardized, the technology that implemented the specification was very barebones. Reader sensitivity was poor and tag power consumption was very high. Nulls, locations where the reader radiation has destructively interfered, would cause tags to not be read. As the technology has matured, however, these problems are not as limiting as they once were. This has allowed for RFID to be considered for WSAN applications. While RFID does have some mechanisms for transmitting information to and from the tag, they are not nearly as developed as inventorying process. As a result, serveral nonstandard extensions to the protocol have been developed.

3.3 Physical Layer Extensions

Tag Antenna Based Sensing (TABS) relies on physical phenomenon affecting antenna performance in a way that can be detected at the interrogator. The mechanism through which the antenna becomes detuned can vary dramatically. For a fluid level sensor, it may be enough for the presence of liquid to detune an antenna [10]. Off the shelf circuit components whose impedance changes in response to external stimuli is another way to do it, as was done for a light detecting RFID Tag [11]. Finally, circuit elements can be constructed to respond external stimuli, as was done for moisture

sensing in soil [12].

It is clear that in all of these cases, information is conveyed by means that are outside of what is defined by the Class 1 Gen 2 protocol. It comes from the presence or absence of a tag, the signal strength of a tag relative to a reference, or difference in timing. For the purposes of this thesis, these means of communication are classified as extensions of the physical layer.

3.4 Air Protocol Layer Extensions

Contrary to the physical layer extensions, which work outside of the RFID protocol, air protocol extensions work within the protocol, but use certain mechanisms in ways that they were not meant to be used.

The University of Washington has developed the Wireless Identification and Sensing Platform (WISP) [13]. The WISP is a tag where the air protocol is implemented by a low power microcontroller. Given that there is an onboard microcontroller, it allows the programmer to include features that were not originally in the specification. An example of how this might be done is by a method they have named EPC Modulation. In order to indicate an integer value, the tag may transmit a different EPC for each state [14]. Another possibility is to reserve a number of bits in the EPC in order to transmit information. When the tag is inventoried and the EPC is back scattered, the data bytes will also be sent. The WISP will queue the next bytes to be transmitted in the next inventory round.

Applications built around the WISP include transmitting accelerometer and temperature information [15], transmitting audio [16], and transmitting images [17].

3.5 Quantifiying the existing mechanisms

Before suggesting changes to the protocol, it is important to understand what is possible within the current specification. Identifying the bottlenecks would allow a high performance system to be developed, and resolving them in a backwards compatible

manner would allow already deployed systems to participate with lower performance.

As all of the above described applications involve information transfer from the tag to the interrogator, the author proposes to reverse the flow of information. In order to test the extremes, two projects were pursued, an RFID controlled price display, and an RFID controlled robot. These projects push the limits of what is possible using the protocol in the amount of information that is transferred and how quickly that information is transferred.

3.5.1 RFID Controlled Price Display

Whenever a manager in a retail store wants to change the price on an item, they must change the price in the store database, and an employee must find the shelf or rack where the item is displayed and change the label. As the number of items becomes larger, the cost of changing an item's price becomes much more significant. In addition, if there is a discrepancy between the price at the register and the price on the shelf, many municipalities have declared that the consumer will receive the lowest of the two prices. With items going on sale for one week and back to normal price the following week, a mistake in changing the labels has an even greater cost for store management.

As a result of all of these challenges, stores are beginning to adopt electronic shelf labels. These devices sport an LCD or e-paper display [18]. They often contain a coin cell battery and communicate using a Zigbee radio or may use an optical communication method. These tags start at $5 dollars for an LCD based display to slightly more for a more legible e-paper display. With an average super market containing 20,000 price tags, this is a significant investment for store management to make. If the tags used RFID instead of an active radio, the price for each label could be reduced as a radio transmitter would no longer be required and the battery could be eliminated.

In order to prototype the device, the following components were used. An NXP RFID IC, SLS4021FHK [19], was used to act as the communication interface. It acts like a normal tag, but it also provides a I^2C interface for a microcontroller and

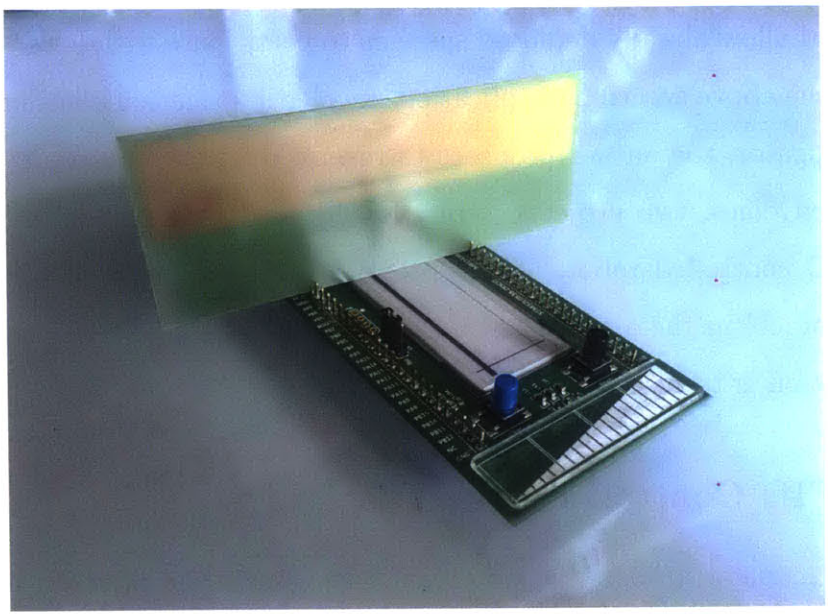


Figure 3-1: Electronic Shelf Label Prototype

appears as an external memory device. For the rest of the hardware platform, the STM32L0538 Discovery Board [20] was used. The board has an STM32L053, a low power microcontroller based on a 32-bit ARM Cortex-M0+ core. There is also a 2 inch e-paper display that has a resolution of 172 x 72 pixels and it has a depth of 2 bits per pixel. The means that a frame buffer of 3096 bytes is required to hold the contents of screen. Figure 3-1 shows the hardware platform used to develop this application.

The RFID IC offers two registers that can utilized for communication between an interrogator and a microcontroller, a bridge register, and a configuration register. When data is written to the bridge register from the radio interface, a flag known as the download flag in the configuration register is set. A microcontroller can monitor this flag and read the download register once data has been written by the interrogator, receiving the word sent by the interrogator and clearing the download flag. An interrogator can also monitor for when this flag is cleared so that data is not accidentally overwritten before it has been read. A similar process can take place in the reverse direction utilizing the upload flag. The use of tag memory as shared memory enables communication between remote devices and an interrogator.

In order to test the throughput of this system, a simple test was devised. A word is written by the interrogator to the bridge register, at which point it is read by the microcontroller, incremented, and written back to the bridge register. Using an imprecise measurement setup, it was found that it took roughly a minute to complete 50 such transactions. This corresponds to a throughput of 26 bits per second, an abysmal performance compared to the hundreds to thousands of kilobits per second available using the wireless communication schemes mentioned in Chapter 1.

It is the author's belief that one of the major contributors to this poor performance is how arbitration between the radio and I^2C interface is handled. In order to avoid a race condition, the tag memory cannot be read or written by one interface when it is in use by the other interface. Because the configuration register is part of the tag memory, polling this register on one interface prevents writing to the bridge register on the other interface.

Using the estimate of 26 bits per second means that it would take roughly 950 seconds to transmit 3096 byte screen buffer for the e-paper display. This is an unreasonable amount of time and it must determined why the throughput of this interface is so poor.

3.5.2 RFID Controlled Roomba

A robot is not subject to many of the design constraints of a sensor node. Robots often contain high power actuators that demand onboard power storage. Cost with robots is typically significantly higher than a sensor node, making the cost savings of using backscatter radio not as relevant. While this may be the case for many applications, new research into low cost robotics has brought the issue back to the forefront.

For this demonstration, the iRobot Create [21] robot is remotely operated through the use of an Xbox 360 gamepad [22]. An Arduino Mega 2560 [23] connects to the serial interface exposed by the Create. Using this serial interface, the arduino is able to command the left and right motor speeds, and read the state of various onboard sensors. The Arduino Mega 2560 then connects to the same NXP SL3S4021FHK IC

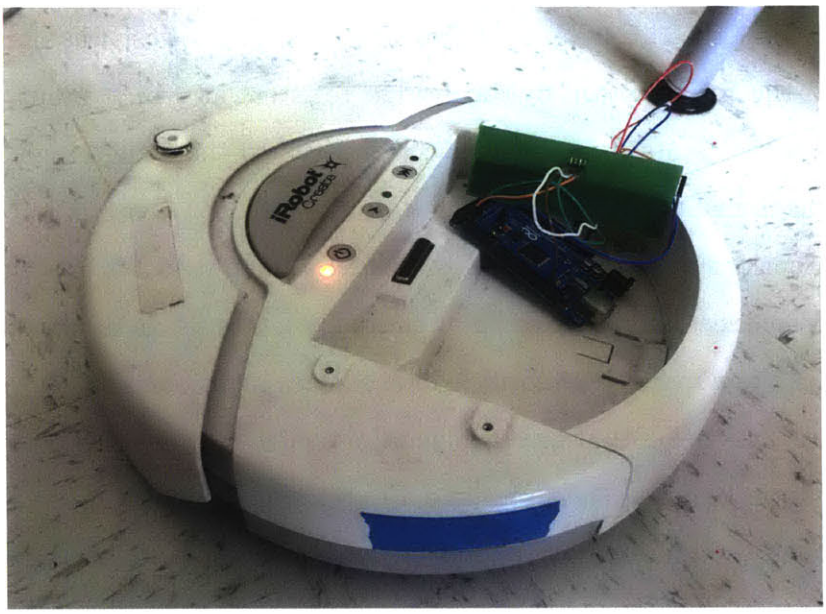


Figure 3-2: RFID controlled Roomba prototype

used in the price display detailed in the previous section. The Arduino then interprets bytes written to the bridge register as motor commands, allowing the Create to be controlled by an interrogator. For the sake of convenience, a laptop interprets the state of the gamepad and translates them to motor commands and uses a python library [24] to forward those commands to the Create through an interrogator. Figure 3-2 depicts the hardware platform that was used to develop this functionality.

While the initial goal of building an RFID controlled robot was accomplished, it did not perform entirely as expected; when driving straight and a right turn is suddenly commanded, 1 to 2 seconds of delay was noticed before the Create responded to the input command. A second test where the RFID interface was removed by connecting the laptop over USB to the Arduino yielded a much better experience with no perceptible delay. Clearly the RFID interface introduced a noticable amount of latency into the system. For systems where real time control is important, time delays limit the gain crossover frequency and ultimately the control bandwidth available. As a result, quantifying the latency introduced by RFID is critical to understanding what kinds of applications can benefit from the use of RFID as a control interface.

Chapter 4

GS1 Class 1 Gen 2 Protocol Performance

Through the motivating examples in the previous chapter, it became clear that if RFID is to be used for control, it is important to fully understand what drives the latency and throughput of the air protocol. While it was not clear where latency is introduced, this chapter will serve to shed some light on the issue. The background in chapter 2 is critical to understanding the different stages in communication. By identifying and measuring the different steps, it is possible to determine where the bottlenecks exist and devise methods to work around them.

4.1 Latency Chain

In order for a write command to be successfully processed, the steps outlined in Figure 4-1 must occur sequentially. The write process involves four systems: the software application, client library, the interrogator, and the tag. The application software dictates the high level logic and instructs the client library what procedures it wants exectued. The client library facilitates communication between the application software and the interrogator. Given that many procedures require multiple messages to go between the interrogator and the client, the client library abstracts these details away from the application software. The interrogator facilitates communication be-

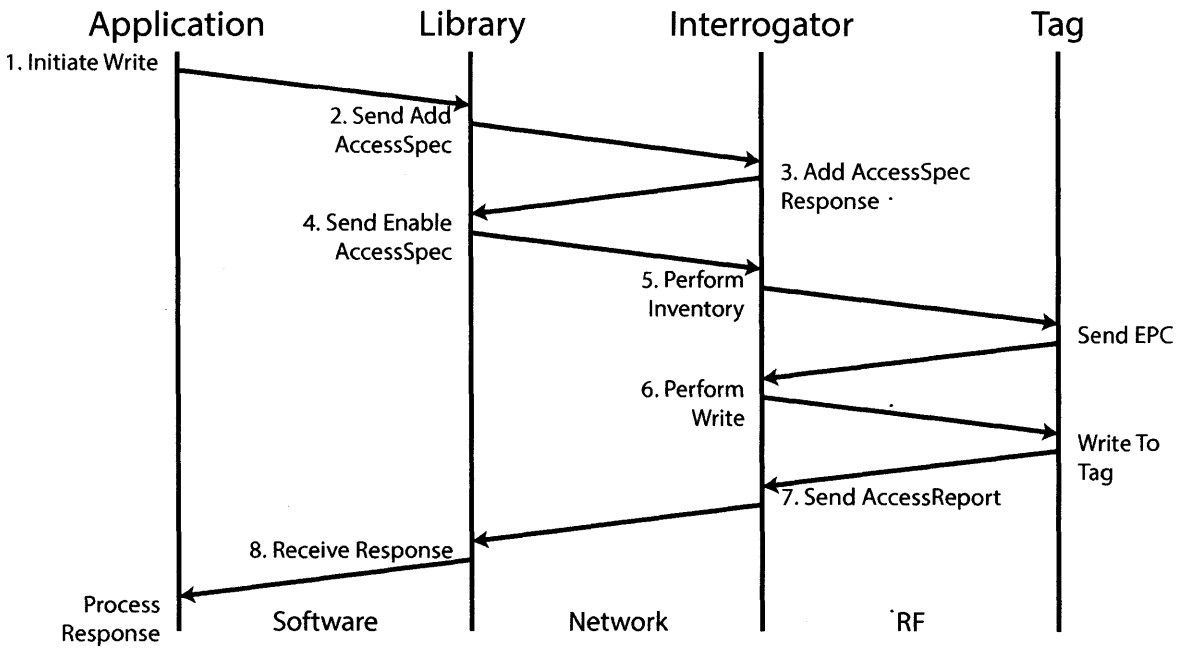


Figure 4-1: Write Operation Communication

tween a client and the tags. It notifies the client of any tags that are in its field of view and also performs read and write operations as instructed by the client. Finally, the tag listens for interrogator commands and communicates as required.

These four systems communicate over three layers: software, network and radio. This poses two challenges, the first is how to instrument each of the layers. The second challenge is how to synchronize the measurements made across layers. In the following sections, these challenges will be addressed.

4.2 Instrumentation

The write process spans three layers: software, network, and RF. In order to accurately time this process, coverage of these three layers much be achieved. Fortunately, there are a number of tools that could be used to take measurements in each of these layers.

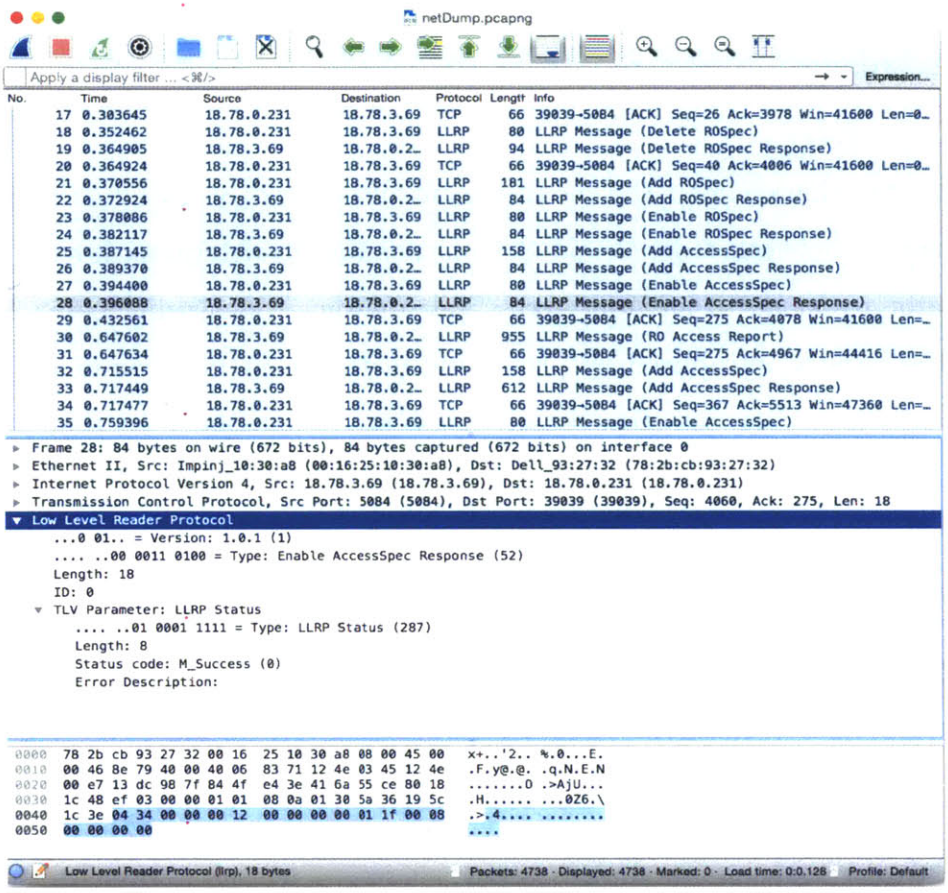


Figure 4-2: Screenshot of Wireshark

4.2.1 Software

Timing in the software layer is the simplest of the three. Modern computers have accurate clocks so it is sufficient to compute the elapsed time. In Figure 4-1, software timing would allow the measurement of time between steps 1 and 8. This information, while very easy to collect, does not give very much insight to where bottlenecks occur.

4.2.2 Network

The next layer is the network layer. Communication between the interrogator and the client library occurs using the client's network interface. This allows the use of tools such as Wireshark, a network protocol analyzer. As shown in Figure 4-2, Wireshark monitors the network interface and records any traffic that passes through. It can

filter for traffic coming from particular sources and also has the capability to decode the traffic. Wireshark offers a layer of detail that would be difficult to achieve in the application software.

Using Wireshark allows the measurement of steps 2, 3, 4 and 7 in figure 4-1. By measuring between steps 2 and 3, it is possible to determine how quickly the interrogator responds to an Add AccessSpec message. Measuring between steps 3 and 4, the latency introduced by processing in the client library can be analyzed. Finally, measuring between steps 4 and 7 gives the time required by the interrogator to complete a write command.

4.2.3 Radio

The last layer is the RF layer. This layer is the most difficult to measure because tools are not readily available. A spectrum analyzer can calculate received power at a particular frequency, but it does not allow for the decoding of packets. National Instruments can provide a solution for this decoding, but the hardware components cost over $50,000 USD [25]. Fortunately, recent interest by the Maker community in Software Defined Radio (SDR) equipment has brought a number of high quality and low cost devices on to the market. These devices allow a certain region of the RF spectrum to be recorded and stored to disk for later processing. Decoders and analyzers can then be built using traditional tools, like C++, or scientific computing platforms such as MATLAB. For details on how this was done, the reader is invited to read Appendix A.

The BladeRF SDR [26], shown in figure 4-3, is one of these new devices. It is a direct conversion transceiver that can tune between 300 MHz and 3.8 GHz and it can provide up to 28 MHz of bandwidth. As was mentioned in Chapter 2, the 915 MHz ISM band is 26 MHz wide meaning that it can capture the entire band. During experimentation, it was found that using the SDR at its full performance is very taxing on the system. A stream of 28 MSPS of 64 bit complex samples corresponds to a flow of 224 MB/s. Commercially available hard disk drives are incapable of keeping up at this data rate [27]. Modern solid state drives were found to be able to

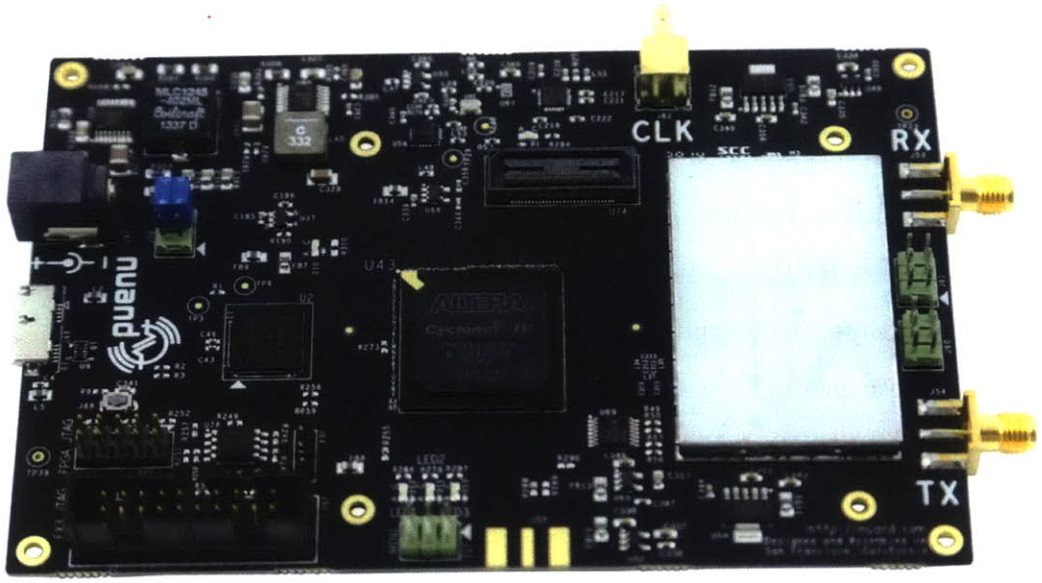


Figure 4-3: BladeRF SDR [26]

cope with this stream of information, but no other processing could be done on the same computer. It was found that running the SDR software affected the performance of the RFID application software to communicate with the interrogator. Due to these considerations, the SDR software and the RFID application software is run on two different computers.

Sampling the RF radiation exposes the communication between the interrogator and the tag which occurs in steps 5 and 6 in figure 4-1. Measuring the time between steps 5 and 6 reveals the time required for the tag to be identified in the inventory process. Measuring step 6 reveals how much time is required to complete the write command. It is this elapsed time that determines the throughput. The rest of the elapsed time feeds into the latency of the system.

4.3 Synchronization

Each tool in the previous section functions independently. As a result, it is straightforward to compare when certain events occured within each layer, but comparing

events in different layers requires more thought.

Synchronization between the software and network layer is simple because both the application software and Wireshark get their timestamp information from the system kernel.

Synchronization between the software and RF layers could also be similarly implemented if the RFID application software and the SDR software were running on the same machine. Unfortunately, performance requirements of the SDR make this impractical. Accordingly, a mechanism that allows the computer running the RFID application software to synchronize with the computer running the SDR software is needed. The solution is a radio that allows the RFID application to emit a "ping" in the RF layer when a write is initiated. The device that gives the RFID application this ability is based on the STM32F4 Nucleo [28] and the CC110L Booster Pack [29]. This device has a serial interface and a radio that transmits in the 915 MHz ISM Band. The GS1 Air Protocol defines 500 kHz wide channels starting at 902.75 MHz and ending at 927.25 MHz for a total of 50 channels [5]. To avoid interfering with communications between the interrogator and tag, the CC110L transmits on an unused channel centered at 902.25 MHz. This method has the advantage of being embedded in the stream of sampled RF data where it can be extracted in post processing.

4.4 Equipment overview

The pieces of hardware involved in this setup are outlined in Figure 4-4.

The client is a Linux machine running the Linux Mint 17.1 distribution. It has a dual core 3.3 GHz Intel Core i3-2120 CPU and 8 GB of RAM.

The interrogator is an Impinj Speedway R420. It is capable of outputting up to 30 dBm and has a network interface which implements LLRP.

The microcontroller is an STM32F401 running at 84 MHz and it contains 512 kB of flash and 96 kB of RAM. It uses a UART peripheral configured at a baud rate of 115200 baud and communicates over SPI with the CC110L radio.

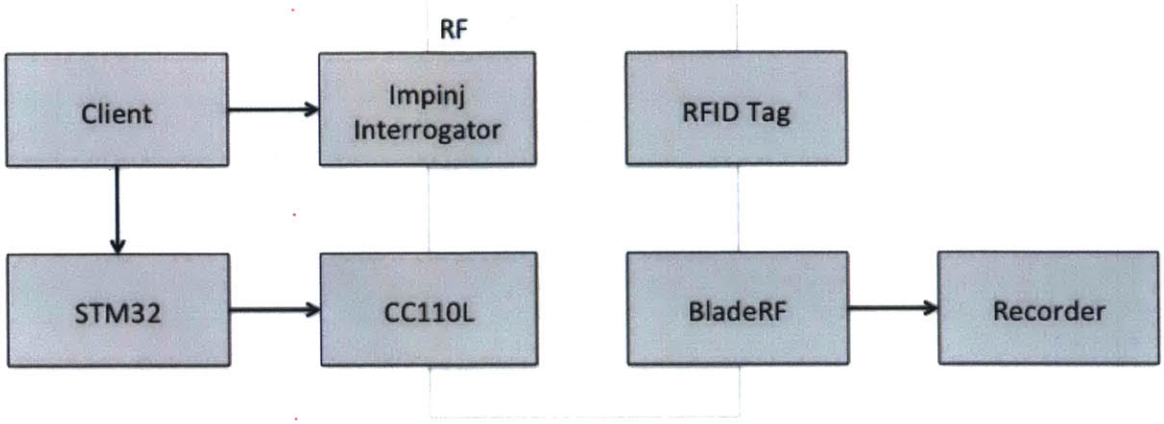


Figure 4-4: Diagram of test setup

The Texas Instruments CC110L radio transceiver is designed to work in a number of sub-GHz frequency bands, including the 915 MHz US ISM band. It can be configured to operate using many different modulation schemes including, 2-FSK, 4-FSK and OOK. It has a maximum transmit power of 12 dBm.

The RFID tag which is written to is an Alien Squiggle Short 9662 with an Alien Higgs 3 RFID IC. The tag is on a clear inlay and has an adhesive backing. The tag contains a 512-bit user memory.

The BladeRF SDR is a direct conversion radio transceiver. It has a 28 MHz bandwidth and a tuning range from 300 kHz to 3.8 GHz. The RF portion of the board is based on the Lime Microsystems LMS6002D which contains all of the amplifiers, mixers, oscillators and filters to implement a flexible RF frontend. This IC is connected to an FPGA to through a two 12 bit parallel interfaces for receiving and transmitting samples and is configured through an SPI interface. The FPGA provides buffering between the LMS6002D and the Cypress FX3 USB 3.0 peripheral controller which provides USB 3.0 connectivity.

The Recorder is a MacBook Air runing Mac OS X 10.10.4. It has a dual core 1.3 GHz Intel Core i5-4250 CPU and 8 GB of RAM. It has two USB 3.0 ports which are required for receiving the full 28 MHz bandwidth from the bladeRF. It also contains a 256 GB SSD flash drive which is capable of sustained writes in excess of 700 MB/s.

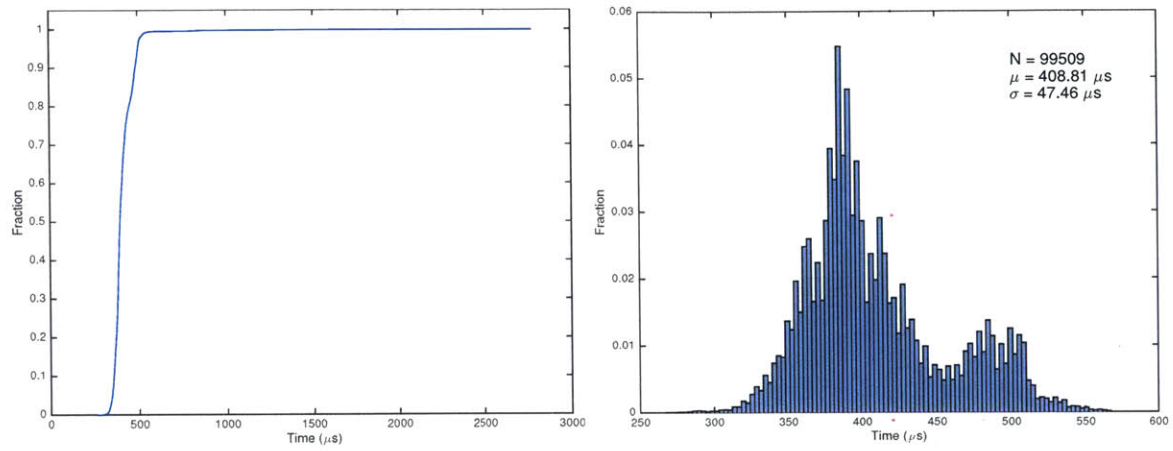


Figure 4-5: Serial Ping Times. Left: Cumulative distribution of ping times. Right: Histogram of ping times less than 600 μs.

4.5 Synchronization Latency Characterization

Before delving into the experiments, it is important to measure the elapsed time from the moment that a byte is sent to the client until that byte is transmitted by the CC110L radio. This process can be broken down into two steps: the time required for a byte to be sent from the application software to the UART peripheral and the time from byte reception at the microcontroller to RF packet transmission.

4.5.1 Serial Latency

The time required for the byte to be sent from the application software to the microcontroller can be approximated by measuring the round trip time for a byte to be transmitted to the microcontroller and back. The latency that is introduced is approximately half of this value. This procedure was repeated 100000 times and the results are shown in figure 4-5. The left chart shows the cumulative distribution of ping times. This plot shows that while the vast majority of serial pings take less than 500 μs, the measured worst case is approximately 5 times worse. This is the nature of the USB as it is instrinsically a nondeterministic protocol. There is not much that can be done to mitigate this latency, but it must be taken into account when measuring against this reference signal. In the chart on the right a histogram containing the

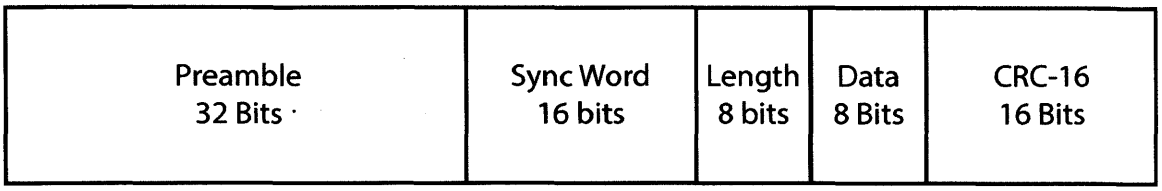


Figure 4-6: CC110L Packet Structure

ping times of the 99.5th percentile and below are plotted. Of these samples, it takes 408.81 μs on average which corresponds to an average latency of 204.21 μs for a byte to be transmitted over USB.

4.5.2 Radio Latency

The second component of the latency in the synchronization signal comes from the reception of the serial byte and transmission of that byte over the radio. In order to determine when the radio is transmitting, the CC110L can be configured to assert one of its outputs after the sync word has been transmitted and deasserted once the CRC-16 has been sent. Figure 4-6 illustrates the packet structure for a radio transmission from the CC110L. With the knowledge of when the output is asserted and the packet structure, it is possible to determine when the start of the preamble has been sent. This output can be monitored using the Saleae Logic 16 [30] logic analyzer. This logic analyzer is capable of sampling 2 inputs at up to 100 MHz, offering extremely fine measurement of digital signals. The logic analyzer can also be used to sample the serial stream from the USB to Serial converter. The difference in timing between the serial stream and when the CC110L output is asserted can be used to determine the latency introduced by this process.

This test was repeated 100,000 times and the results are shown in figure 4-7. It takes 376 μs from the falling edge of the serial byte start bit to the start of the RF packet preamble. The standard deviation of the distribution is 174 ns and a range of 1.16 μs. Unlike the USB communication, this section is very predictable.

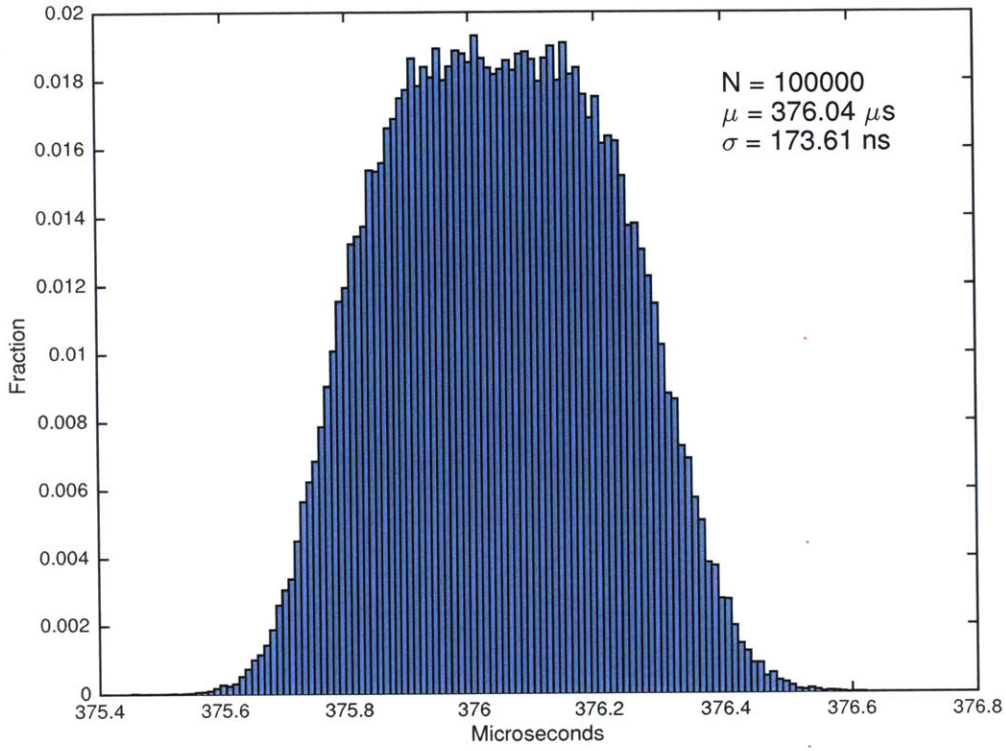


Figure 4-7: Latency from RF Packet Transmission

4.5.3 Synchronization Adjustment

In order to account for the latency in this synchronization symbol, all elapsed time periods were adjusted upwards by half the average serial latency plus the radio latency. Using the measurements made in the previous sections, this amounts to an adjustment of 580 μs.

4.6 Experiments

It is expected that the performance of the C1G2 protocol is dependent on the number of tags in the field of view of the antenna. In order to control for this, experiments were run in the presence of 1, 5, 10, and 20 tags. For a latency test, the interrogator was commanded to write 2 words to the tag's EPC memory 500 times. For the throughput test, the interrogator was commanded to fill the 64 byte user memory 250 times. There are two ways to write to user memory, a Write command and a

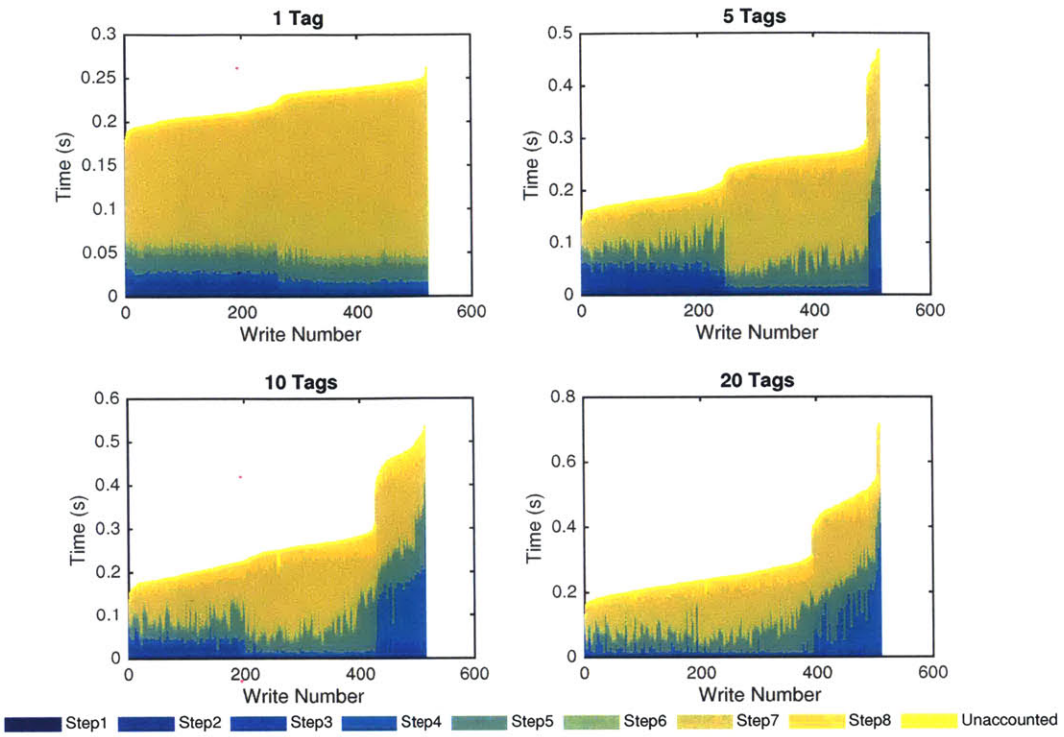


Figure 4-8: Latency Test Breakdown

BlockWrite command. Both methods were measured.

4.7 Results

The results of the three different tests can be seen in Figures 4-8, 4-9, and 4-10. The results are shown in a stacked bar chart with the amount of time spent by each step appearing as a different colored segment. The cumulative sum is represented as the height of each bar. There is also a segment of each bar that is meant to show the amount of time that has not been accounted for in any of the measurements. For almost all cases, the unaccounted for time is on the order of milliseconds, but in some cases, it can be as high as 30 milliseconds. The steps in the figures below refer to the steps found in Figure 4-1. Histograms of the individual steps can be found in Appendix B.

All of the charts share some similarities. One of the most striking features is how long step 7, which is the time from when a write is completed to the time the RO

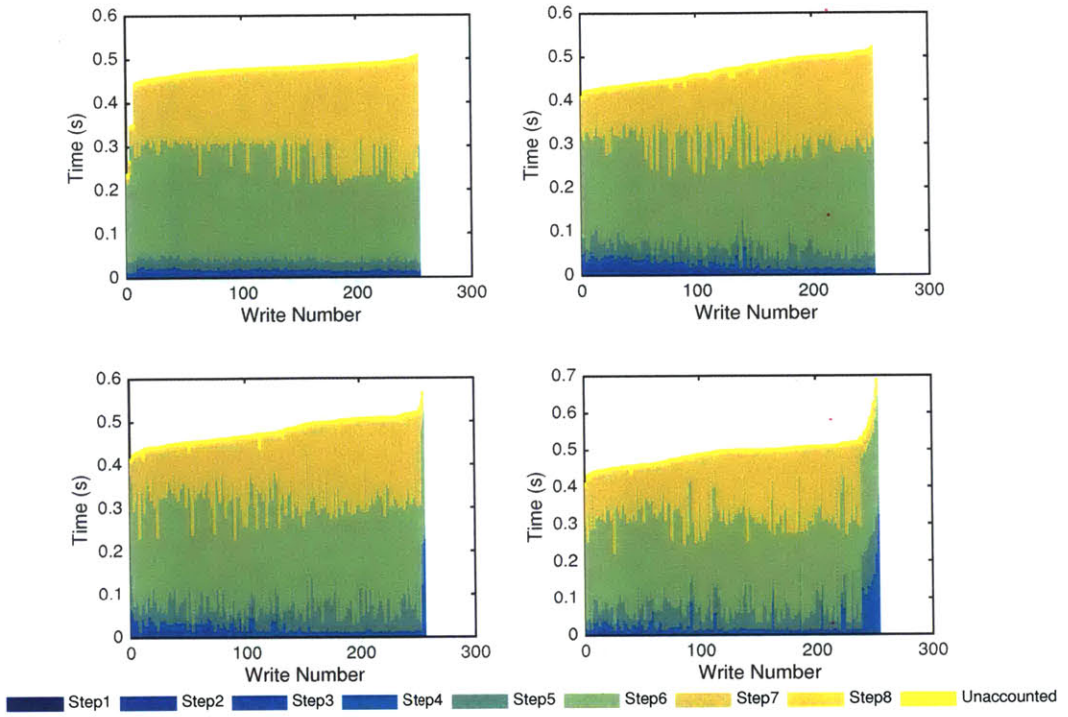


Figure 4-9: Write Test Breakdown

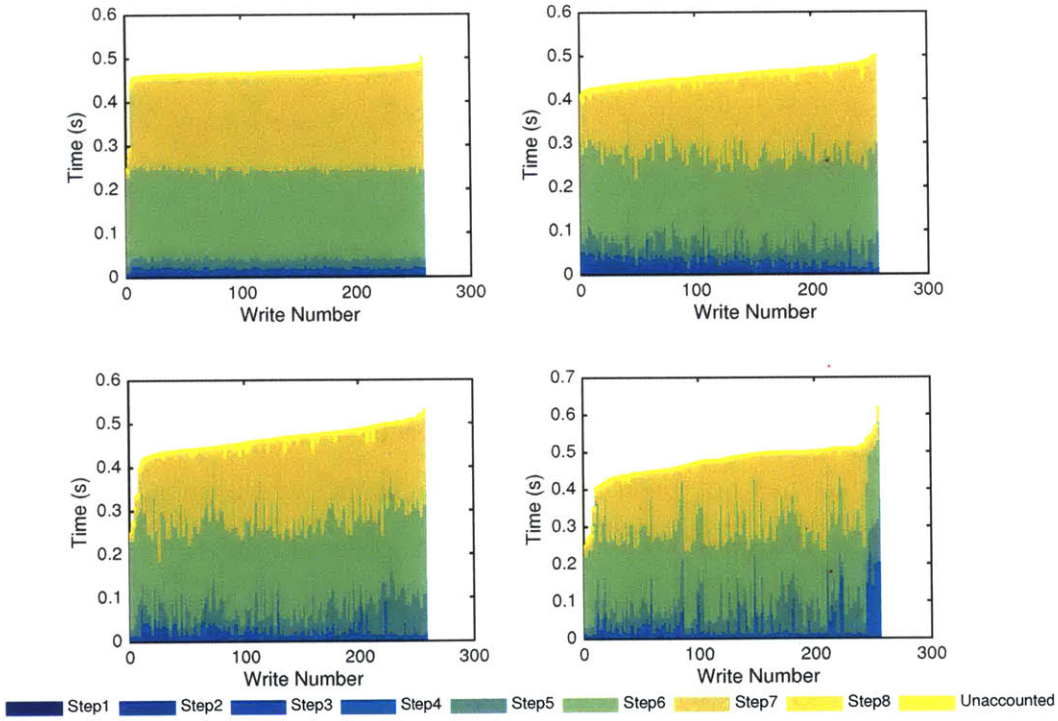


Figure 4-10: BlockWrite Test Breakdown

AccessReport is emitted by the interrogator, takes to complete. The next observation is how the time to complete step 5 increases as the number of tags increases. This is expected as step 5 is the inventory step. In addition, there appears to be a large amount of variability in step 3. This is the amount of time it takes for the Add AccessSpec Response to be sent from the reader.

For the latency test, time spent writing is significantly shorter given the fewer number of words written. Comparing a Write operation to a BlockWrite operation, we find that the BlockWrite spends less time in the writing portion. For every word written, a BlockWrite requires 6.65 ms and a normal Write requires 8.75 ms. Over the 32 words written, this amounts to 67 ms in savings.

4.8 Implications for RFID IoT Applications

This work has shown that RFID is not the bottleneck for current implementations of RFID IC's with microcontroller interfaces. This work suggests that the current method of arbitration of access to tag memory may also be a significant source of latency. New IC's with a better arbitration process would significantly improve the performance of applications that use RFID as a communication method.

Chapter 5

Conclusion

5.1 Contributions

RFID is a versatile platform upon which many new applications can be developed. In order to maximize the number of applications built on RFID, it is important to provide a low power, low latency, and high throughput connection. In order to demonstrate the possibilities, these are the main contributions of this thesis:

- It has been demonstrated that RFID can be used to control remote nodes in applications that are sensitive to latency and throughput.
- The protocol has been analyzed; bottlenecks have been identified and quantified.

5.2 Sources of Improvement and Future Work

While there are improvements that could be made to both LLRP and the Air Protocol, a large source of latency is in the reader implementation. It is the author's suspicion that there is a periodic task on the interrogator that runs every 200 to 250 ms. This task collects tag observations, manages writing to the tag and manages communication with the client. This periodic task would explain the stair step behavior that is especially prominent in the latency test.

When considering improvement to LLRP, latency can be reduced by minimizing

the number of messages required to execute a write. Including an "execute immediately" option in the Add AccessSpec message eliminates the need for the Enable AccessSpec message. There is precedent in the LLRP protocol for similar behavior in starting the inventory process automatically.

While the Air Protocol does not seem to be an immediate bottleneck, improvements can be made to reduce the overhead required. Figure 2-5 depicts the six steps that must occur each and every time a tag access is performed. The result of these six steps is the acquisition of a handle that is used in the next interrogator access command, after which the handle is ignored by the tag. If instead the handle is maintained for some time, the latency cost of identifying a tag before tag access can be amortized over multiple accesses. Furthermore, this change can be made in a backwards compatible manner.

Directions for future investigation are plentiful. Software defined implementations of the tag and the interrogator allow changes in the air protocol to be quickly prototyped and validated. These will become invaluable tools when attempting to resolve the issues raised in this work. Groups in the field are investigating what is possible with a clean sheet redesign of the air protocol using similar tools [31]. Ideas from this exploratory work is invaluable as it can inform how the air protocol evolves. However, it is important to keep an eye towards creating a protocol which facilitates as many use cases as possible, not just those requiring high throughput.

5.3 Beyond ID

RFID has had a tremendous impact and has seen adoption in many different industries. It is the author's belief that the next frontier for RFID is enabling communication in the IoT. In this future, interrogators move from being devices that find tagged objects, to gateways which facilitate the transfer of information from Internet Protocol (IP) to the remote tags. These tags become more than a means to identify objects; they are communication interfaces for devices such as the electronic shelf labels and autonomous robots detailed in Chapter 3.

As the devices and applications that use the Air Protocol evolve, the protocol itself must also evolve. The inventory process can be integrated in to a larger Media Access Control (MAC) layer as a device discovery mechanism. The EPC would then become a MAC address that can be used to route packets to the remote devices. If a more deterministic structure is required, Time Division Multiple Access (TDMA) methods can be used to assign a remote device a slot in which it can transmit in. These changes will require a more complex tag implementation, but Moore's law should allow these devices to become smaller and more power efficient.

Appendix A

Software Defined Listener

A software defined listener is a device which can eavesdrop on the communication between the interrogator and a tag and automatically decode it. A software defined RFID interrogator was built by Buettner [32], but it does not implement frequency hopping, uses a different hardware platform, and it is incompatible with the recent versions of GNURadio. This chapter will describe the processing chain used to decode interrogator and tag transmissions.

A.1 How the ISM Band is Used

As mentioned in Chapter 2, RFID operates in the UHF ISM band. FCC regulations allow non-licensed intential radiator to transmit up to 1 watt of power if frequency hopping with at least 50 channels are used [33]. As a result, the GS1 RFID Air Protocol uses 50 channels with the first center at 902.75 MHz and a channel spacing of 500 kHz [5]. The last channel is found at 927.25 MHz. The channels at 902.25 MHz and 927.75 MHz are not used to in order to limit the amount of radiation emitted outside of the ISM band.

The waterfall plot in Figure A-1 depicts the power contained at each frequency over time. Figure A-2 contains a small portion of the captured signal.

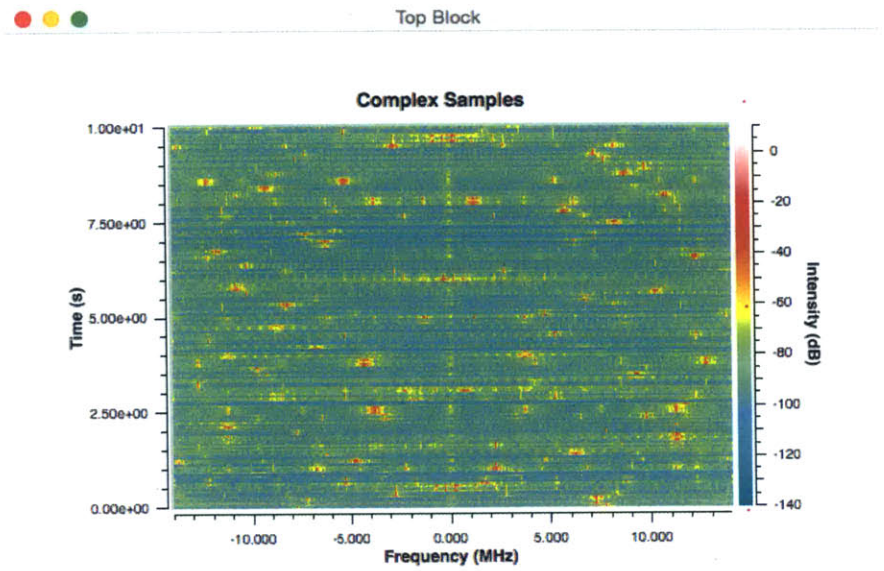


Figure A-1: Waterfall plot of signal frequency over time

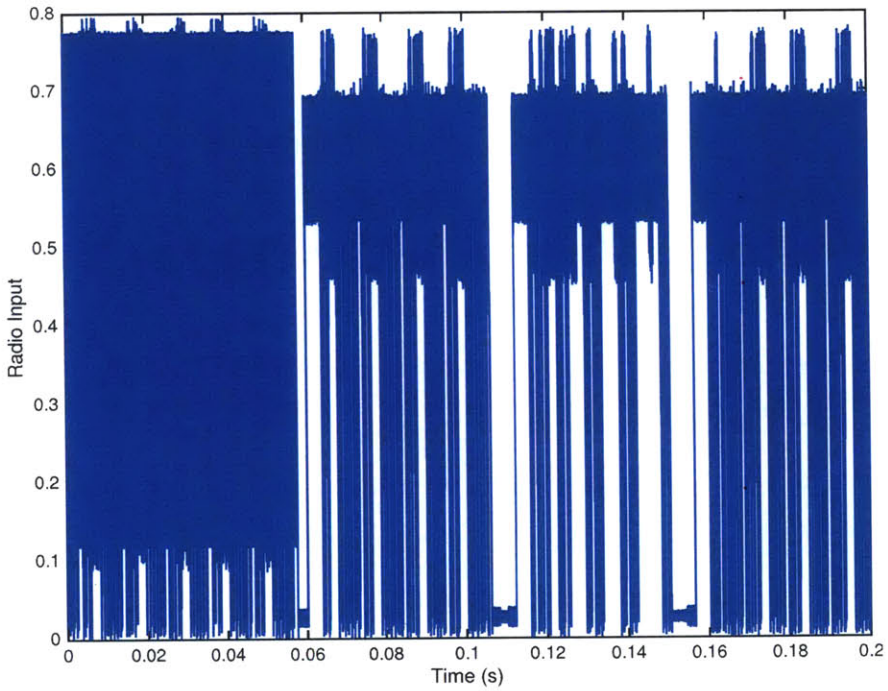


Figure A-2: Time series of signal magnitude

A.2 Receiving the signal

An antenna which is tuned to the ISM band is required. In addition, due to the unknown frequency hopping pattern of the interrogator, the entire ISM band must be captured. The BladeRF provides 28 MS/s of I/Q samples, which corresponds to 28 MHz bandwidth which is just enough to cover the 25 MHz of the used spectrum.

A.3 Shifting to Baseband

When building a software defined listener, the first step is shift the active channel down to baseband frequency. This is performed in two steps. First the active channel must be detected. Next, the signal must be shifted down to baseband by the channel center frequency.

Detecting the active channel can be done by computing the fourier coefficients of a signal. While this is typically done by utilizing the fast fourier transform, significant computational savings can be realized by recognizing that only the coefficients at the center frequencies of each channel are of interest. Using this knowledge, the discrete fourier transform is calculated at each of the channel centers and the channel with the strongest contribution is the active channel.

Shifting the signal down to baseband is achieved by using the frequency shift property of the fourier transform. The signal is multiplied by a complex exponential that is the negative of the channel frequency. While the actual center frequency may not exactly coincide with the expected center frequency, in practice it is close enough to not pose a significant issue. GNURadio provides a rotator class which utilizes optimized routines to perform this frequency translation.

Now that it is understood what computations are required to perform the required shift, it must be considered when these computations are performed. While periodic sampling is an option, a far more robust method is to determine when a frequency hop has occured and then perform the required calculation.

How is it possible to determine when a frequency hop has occured without com-

puting the fourier transform? Interrogator commands are visible in Figure A-2 as dips in radiated power, however there are also longer dips in power. These dips in power are known as blanking intervals and they correspond to the time required by the Interrogator's internal RF hardware to adjust its local oscillator to a new frequency when frequency hopping. If the signal power can be monitored and these long dips in power identified, it stands to reason that significant computation reduction can be achieved. Empirically, these dips in power are 2 ms in length. Dips in power due to interrogator communication are significantly shorter. A low pass filter with a cutoff frequency of 600 Hz should respond to the blanking intervals and reject the higher frequency interrogator communication.

A further improvement can be made by recognizing that the full 28 MHz does not need to be passed through the low pass filter. If one out of every 10000 samples is passed through the low pass filter, the blanking intervals are still identifiable, but the number of computations required is reduced. This process is a downsampling of the signal which means that aliasing must be considered. Aliasing does occur, but because the majority of the signal power in the magnitude of the received RF is near baseband, the aliasing does not do much to distort the low frequency components of the signal. However, aliasing does affect the much weaker tag response which is why the frequency shifting is required. Figure A-3 shows a waterfall plot of a complex signal and its magnitude. The concentration of power centered on 0 Hz shows why aliasing does not significantly distort the output.

A.4 Signal Magnitude

Once the signal has been shifted to baseband, the magnitude of the signal is calculated.

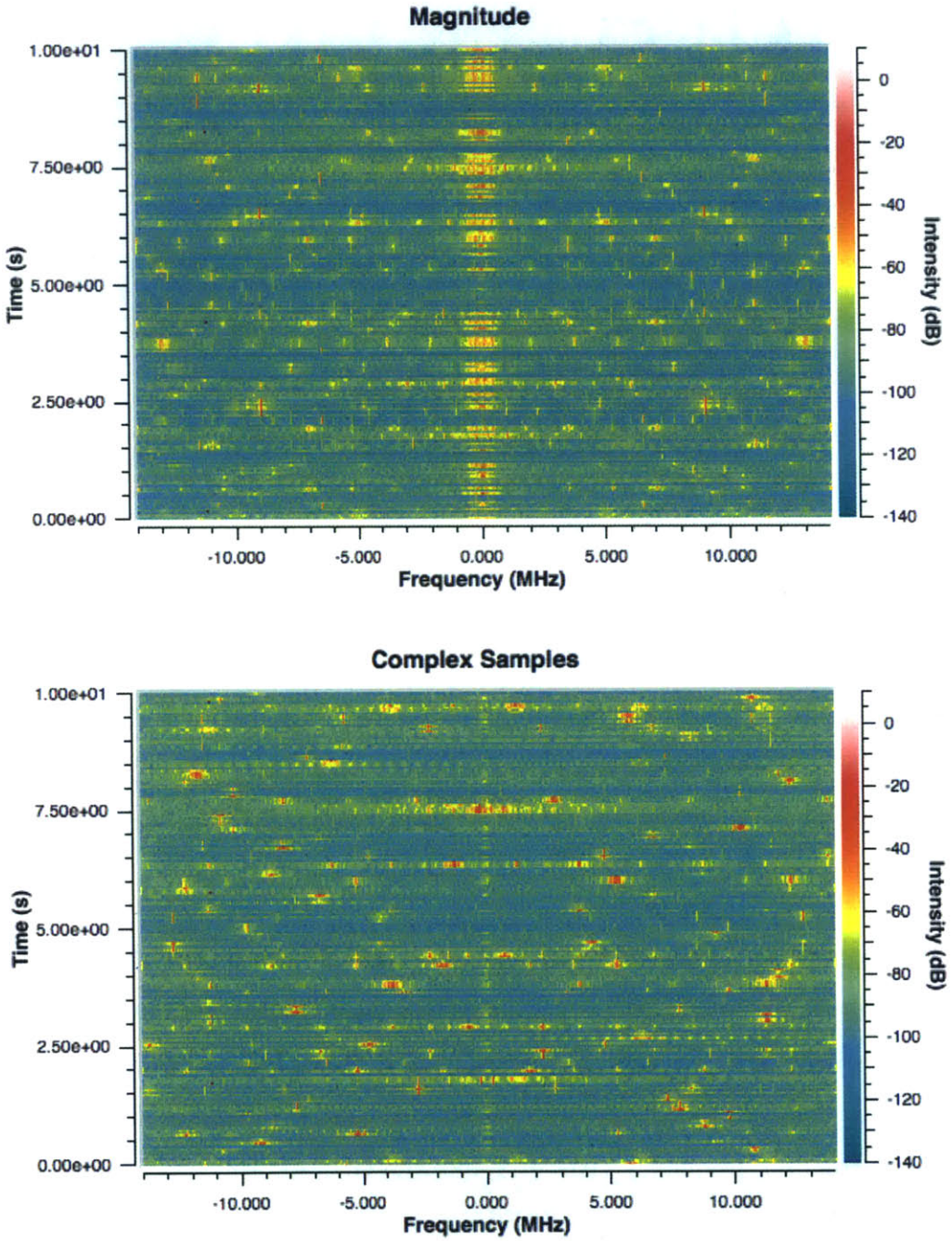


Figure A-3: Time series of signal magnitude

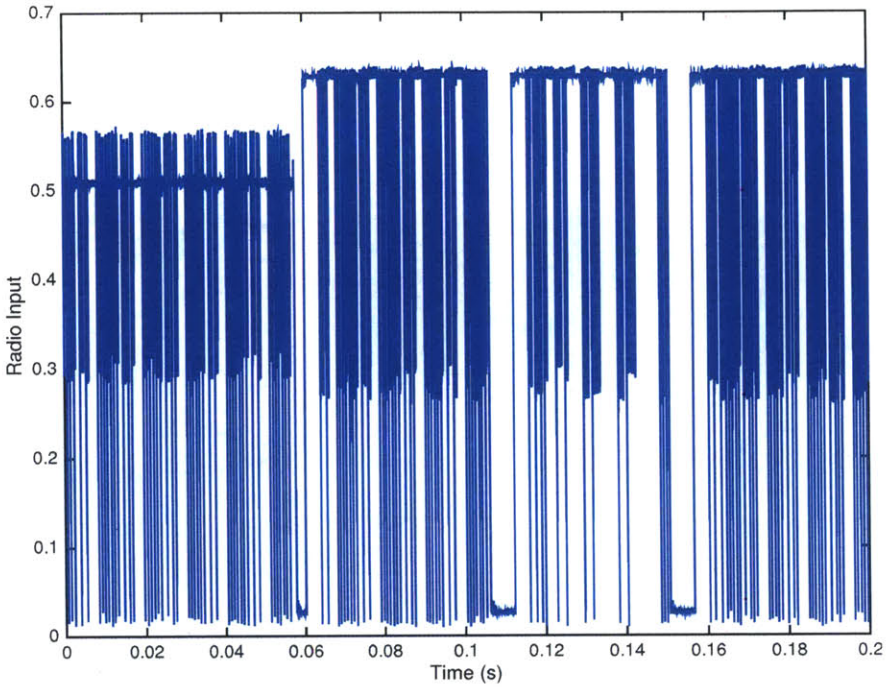


Figure A-4: Time series of decimated signal

A.5 Decimation

We now have a magnitude signal stream that is 28 MHz wide. In order to reduce the computation needed in later steps, the sample must be reduced through a decimation process. This requires passing the signal stream through an anti-aliasing filter and then a downsampling step. The width of the antialiasing filter should be informed by the backscatter link frequency (BLF). While the GS1 Air Protocol allows for BLF's greater than 1 MHz, in practice, a BLF in the 100 to 300 kHz range is standard. An antialias filter with a passband of to 400 kHz is more than sufficient. This allows the signal to be downsampled to 1 MHz without any significant concerns of aliasing. Figure A-4 depicts the same signal found in Figure A-2 after the shift, filter, and decimations have been applied.

A.6 Interrogator Symbol Detection

The interrogator symbols are defined in Chapter 2. Two methods of interrogator symbol detection are possible, a matched filter, and a threshold crossing detector.

A.6.1 Matched Filter

A matched filter is a filter that attempts to find a signature in a signal by correlating the received signal with the signature. When the signature is found, the output of the matched filter is maximal. A matched filter is implemented as a finite impulse response filter with the taps set to the copy of the signal. The sum of the taps correspond to the DC offset of the filter output and the sum of the absolute value of the taps correspond to the gain of the filter.

The filter taps are set to match the the symbols in Figure 2-3. The output of the filter can be seen in Figure A-5. Due to the shape of the symbols, a one is also detected as a zero. However, while a zero does create a peak in the output of the matched filter for a one, that peak is not as high as when a one is matched. The symbols can then be decoded by finding the peaks in the respective filter output. Figure A-6 shows the matched filter output depicted in Figure A-5 and also shows the result of the bit detection algorithm. A one is represented as a green circle and a zero is represented as a red circle. The reader data in binary is 10001101011010010011001.

A.6.2 Threshold Crossing

While the matched filter is very robust, it is computationally expensive. If the signature contains n taps, n multiplications and $n-1$ additions are required to compute a single output. A simpler approach is a threshold crossing detector. The threshold is determined by passing the magnitude signal through a low pass filter and setting the threshold at some fraction of the low pass filter output. The time between the positive crossings of the threshold yields the length of the interrogator symbols. Given that a one is longer than a zero, a threshold on the symbol length will yield whether or not a particular symbol is a zero or a one.

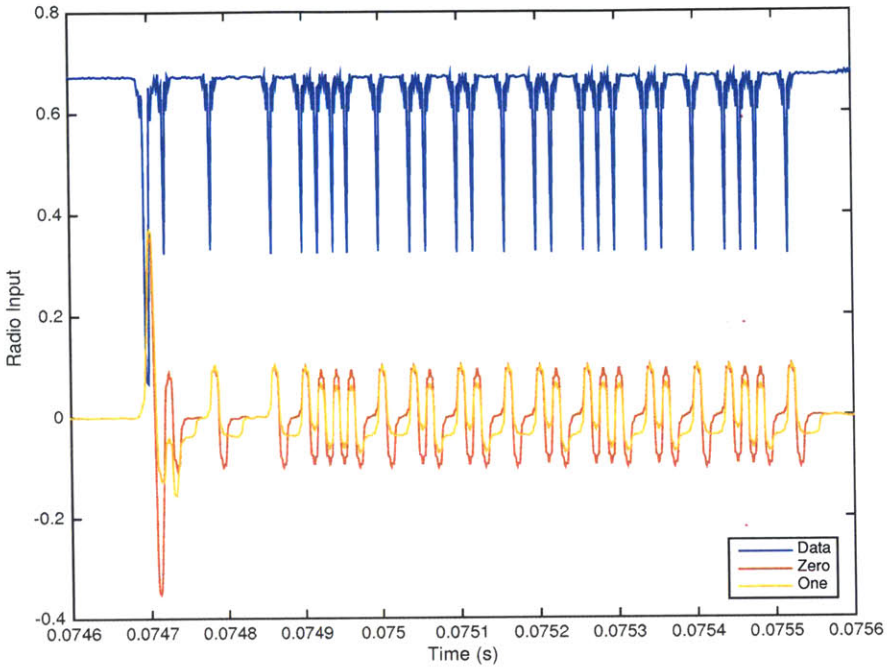


Figure A-5: Matched Filter Output

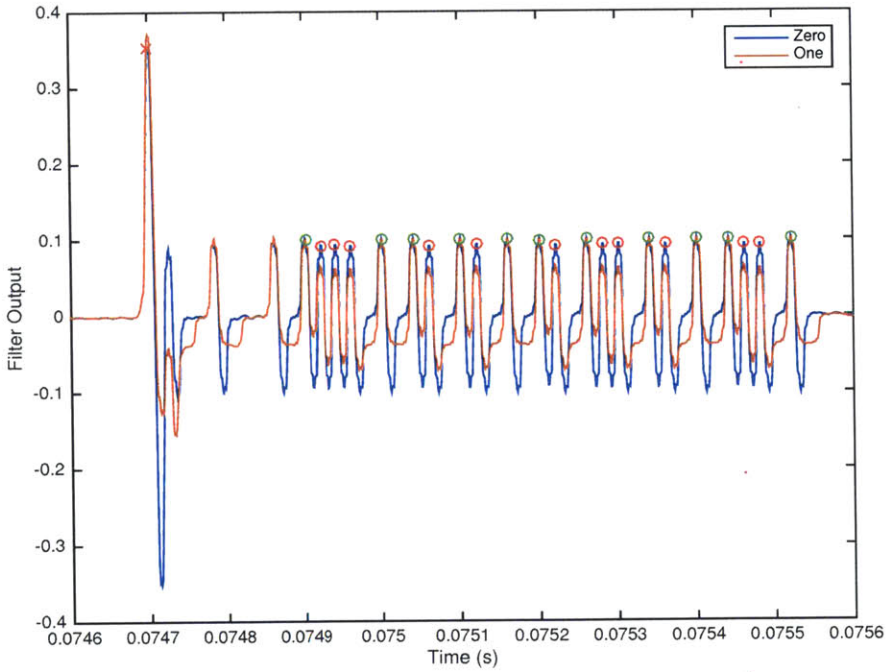


Figure A-6: Matched Filter Symbol Recognition

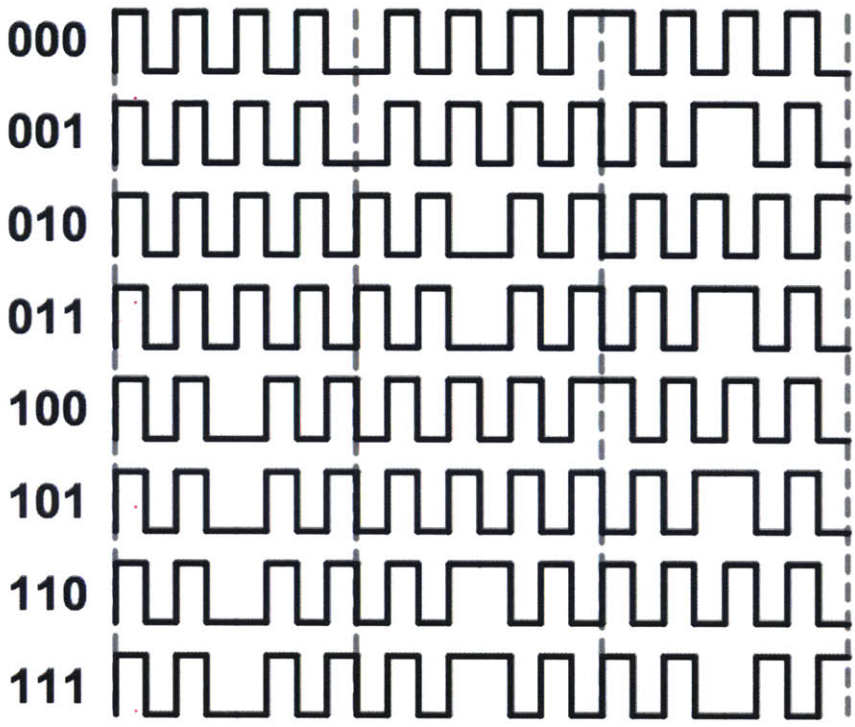


Figure A-7: Miller Subcarrier Sequences (From: GS1 C1G2 Air Protocol [5])

While this approach is simpler, it can yield unreliable results in noisy environments or in the presence of a strong tag backscatter signal. The matched filter approach will almost always provide a reliable output.

A.7 Interrogator Command Decoding

Once the reader symbols have been correctly decoded, the command decoding is completed by identifying the command pattern and decoding the bits as described in the GS1 UHF RFID Air Protocol [5]. In the case of the reader command depicted in Figure A-6, this is a Query command with a Q value of 2, indicating that there will be 4 slots in the following inventory round.

A.8 Tag Symbol Detection

The tag symbols are defined in the GS1 UHF Air Protocol [5]. While there have been many methods developed to decode the tag symbols with varying level of accuracy and robustness, the simplest method is using a threshold crossing detector. As was done in the threshold crossing detector for the interrogator, the tag backscatter signal is low pass filtered. However, unlike the interrogator decoding, the filter output is directly used as the threshold and the time between crossings can be used to detect the tag symbols.

The pilot tone can be detected as a series of crossing that take a similar amount of time. If a rolling standard deviation of the crossings is calculated, it can be seen that it drops off very quickly. The pilot tone can be used to fine tune the BLF.

The tag uses miller encoding which encodes the bits as 180° phase transitions. These phase transitions appear as crossings that take twice as long as expected given the BLF. The center of these phase transitions can stored for the next phase of processing.

In order to decode a Miller encoded signal, it is necessary to determine where the symbol boundaries occur. As shown in Figure A-7, a phase reversal occurs between two consecutive zeros and in the middle of a one. However, only the location of the phase reversals are currently known. If we instead focus on where phase reversals occur, it can be seen that transitions occur at 1, $\frac{3}{2}$, or 2 symbol widths away from each other. If the transitions are one symbol width apart, the last decoded symbol is repeated. If the transitions are $\frac{3}{2}$ symbol width apart and if the last decoded symbol is a one, then a zero was just decoded. If the last decoded symbol was a zero, then a zero and a one was just decoded. Finally, if the transitions are two symbol widths apart and the last decoded symbol was a one, then a zero and a one were just decoded. A transition that is two symbol widths apart after a zero has just been decoded is an invalid state and points to an erroneous decoding of the signal.

Given that a tag response will always begin with the same preamble of 010111 [5], this appears as a set of 4 transitions. The time between the first and second

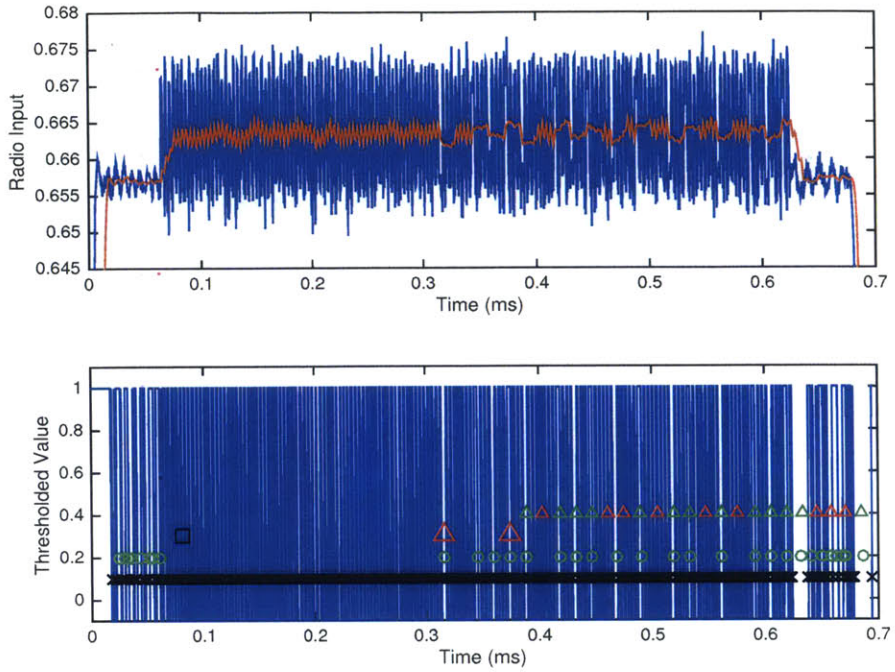


Figure A-8: Decoded Tag Backscatter

transition should be twice as long as the length of time between the second and third transition as well as the time between the third and fourth transition. Figure A-8 shows the output of this decoding process. The top plot contains the raw tag response in blue. The orange line is the output of a 25 μs moving average which serves as the threshold. The bottom plot contains the thresholded tag response; values greater than the threshold are set to one and values below the threshold are set to negative one. The row of black "x"'s indicate the detected edges. The black square at 750 μs indicates the detection of the pilot signal. The large red triangles indicate the detection of the preamble. The smaller triangles are the bit decodings, a red triangle is a zero and a green triangle is a one.

A.9 Interrogator Command Pattern Matching

The reader decoding was found to be very slow as it was implemented in MATLAB. In addition, a bug in the channel selection code and a buffer overflow in the SDR interface caused the interrogator decoding to be unreliable. By the time these issues

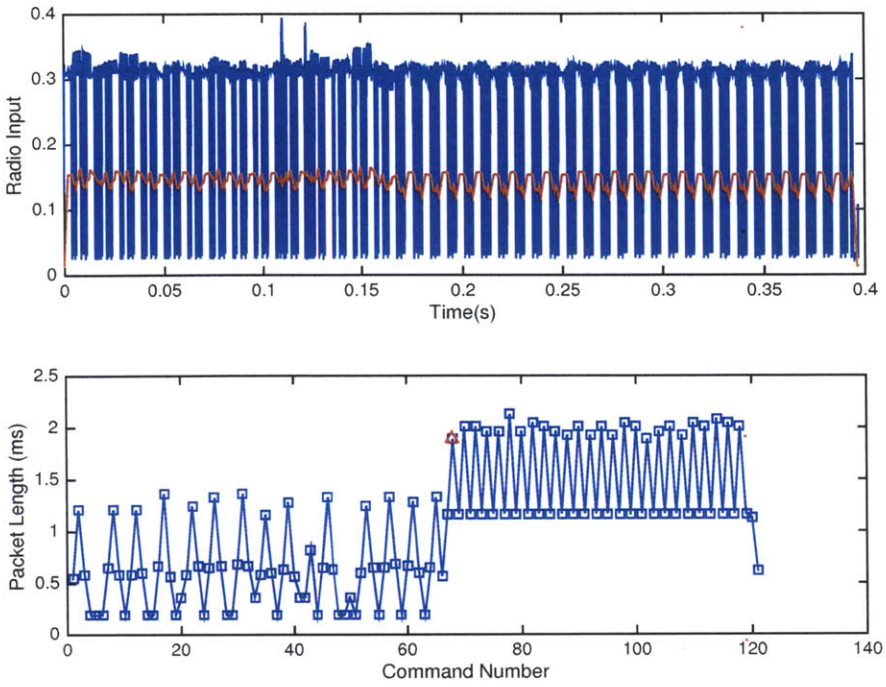


Figure A-9: Interrogator Command Lengths

were remedied, an insufficient amount of time was available to translate the reader decoding to C++. As a result a different approach was used to detect the read and write process.

It was observed that the length of a Write and a BlockWrite command are significantly longer than the commands that are typically used in the inventory process. If the length of an interrogator command could be accurately measured, it stands to reason that a write can be detected without explicitly decoding the interrogator commands.

In order to detect the length of an interrogator command, a process very similar to the threshold crossing method was used. Instead of storing every threshold crossing however, any threshold crossings that occured within 500 μs of a previous cross was deemed to be part of the same interrogator command. Once more than 500 μs has elapsed without encountering a threshold crossing, the interrogator command was complete and the process would be repeated. Once the interrogator command lengths have been obtained, a pattern in the command lengths is searched for. The top plot

in Figure A-9 depicts a Write command in the presence of 20 tags. The red line is the threshold that is used to find the edges of a reader symbol. The bottom plot shows the detected lengths of the reader commands. The irregular command lengths that occur below command number 70 are the result of the inventory process. Once the desired tag has been found, the write process is initialized, leading to a regular pattern of alternating Write and ReqRN commands. The red triangle marker indicates that the algorithm has found the start of a write process.

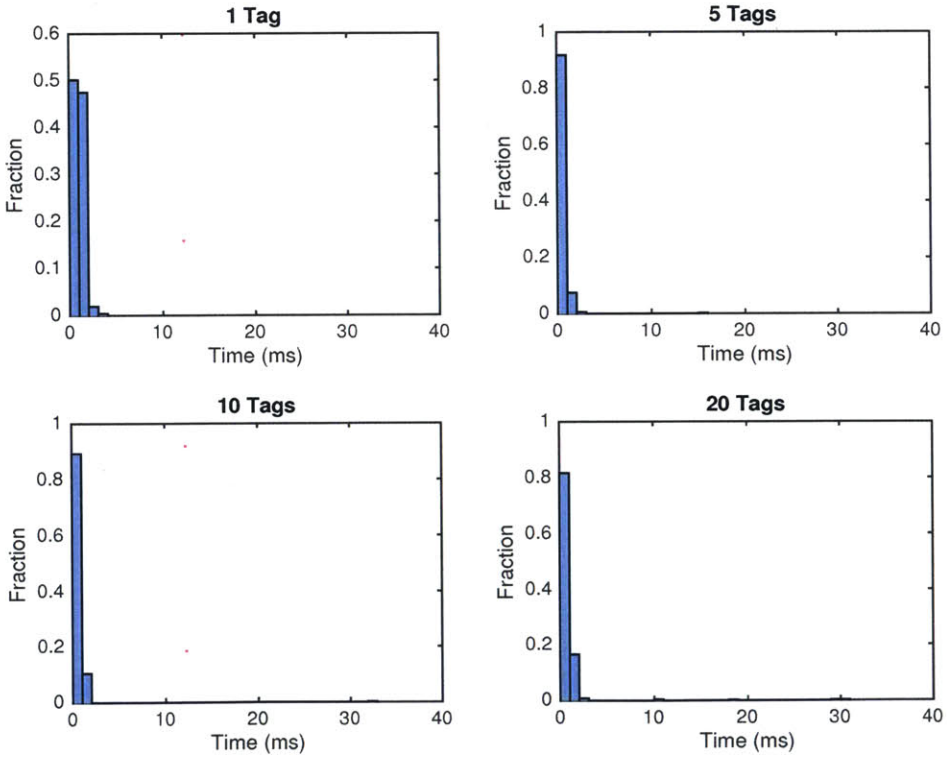
The procedure proved to be very reliable. All of the observed cases that yielded a false negative were found to occur in the presence of a strong tag backscatter signal, which crossed the threshold and interfered with the interrogator command length detection. The occurence of this was very low and was deemed acceptable for the purposes of the experiments carried out in this thesis.

Appendix B

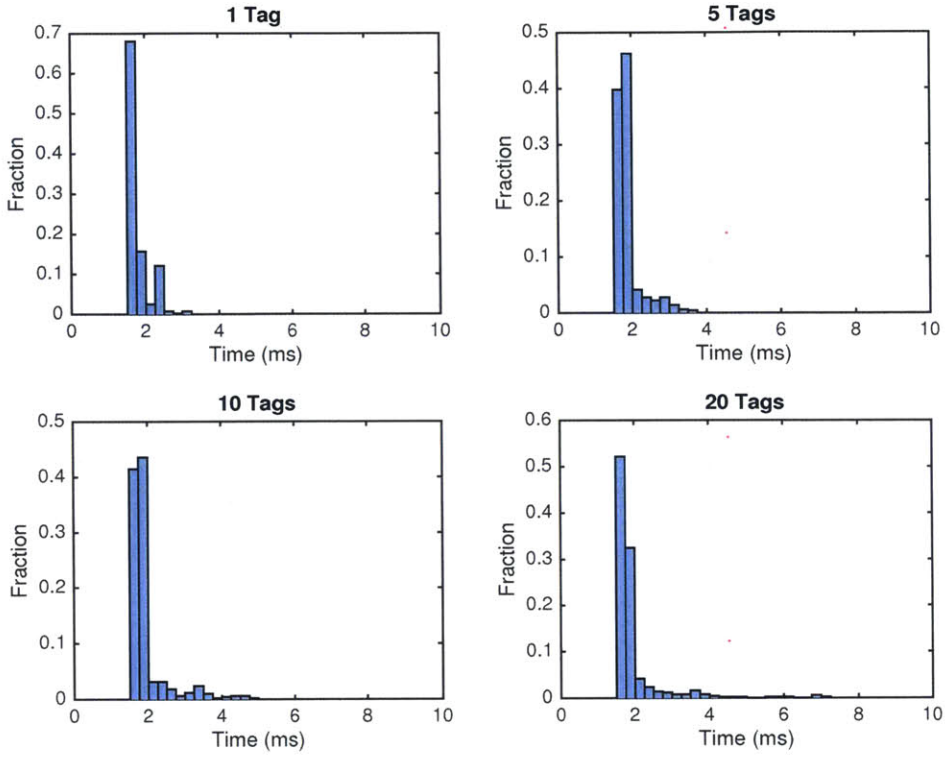
Extended Results

B.1 Latency Test

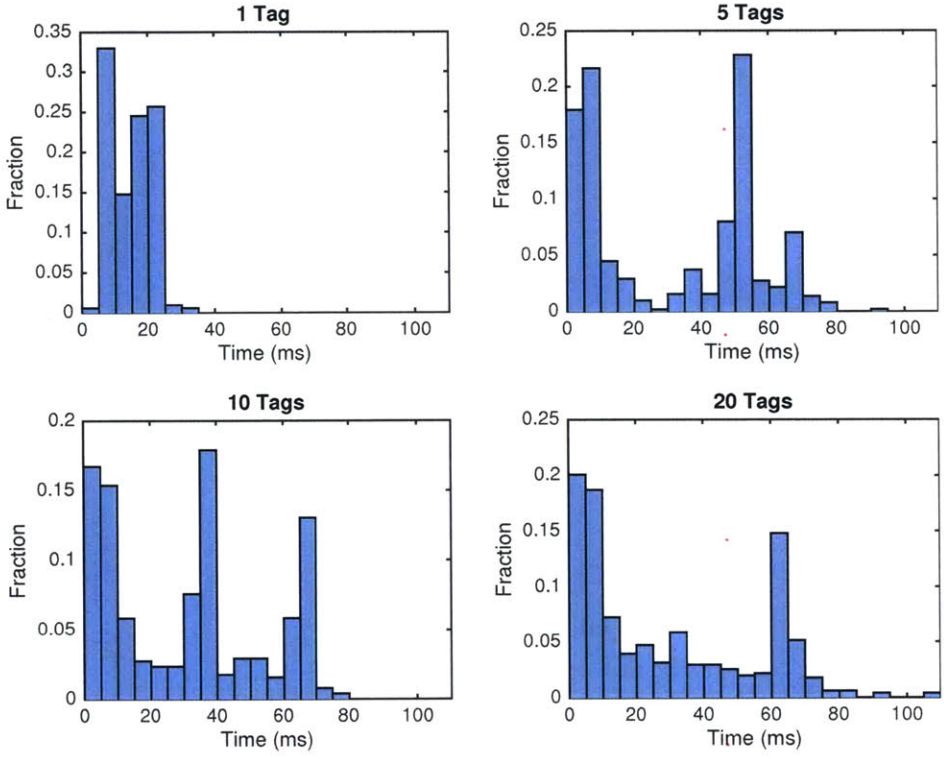
Step 1:



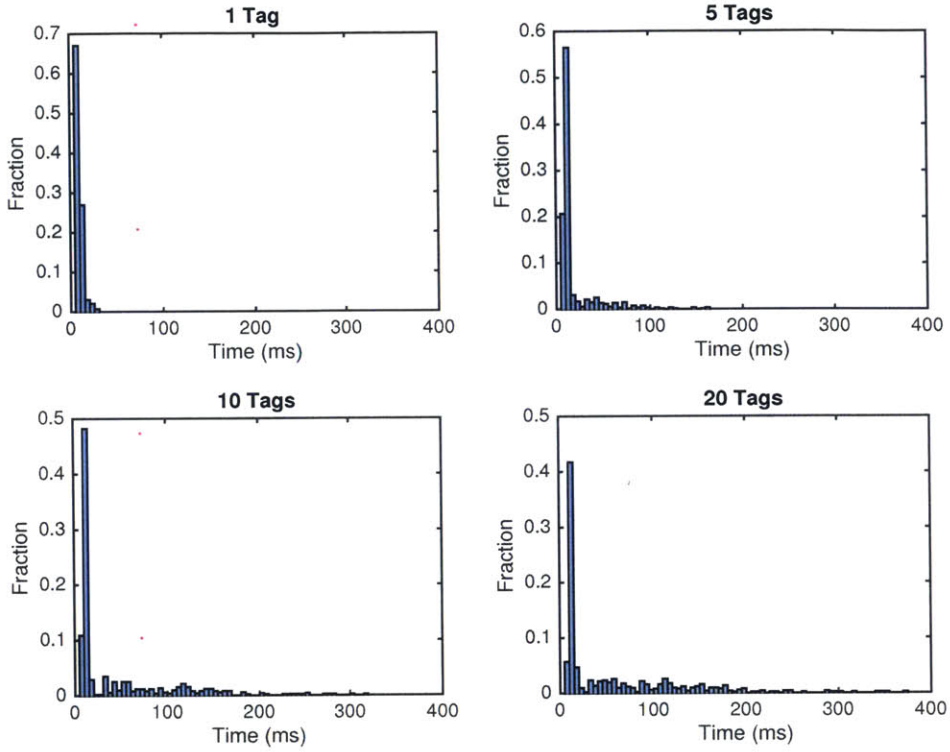
Step 2:


1 Tag
5 Tags
10 Tags
20 Tags
Fraction
Time (ms)

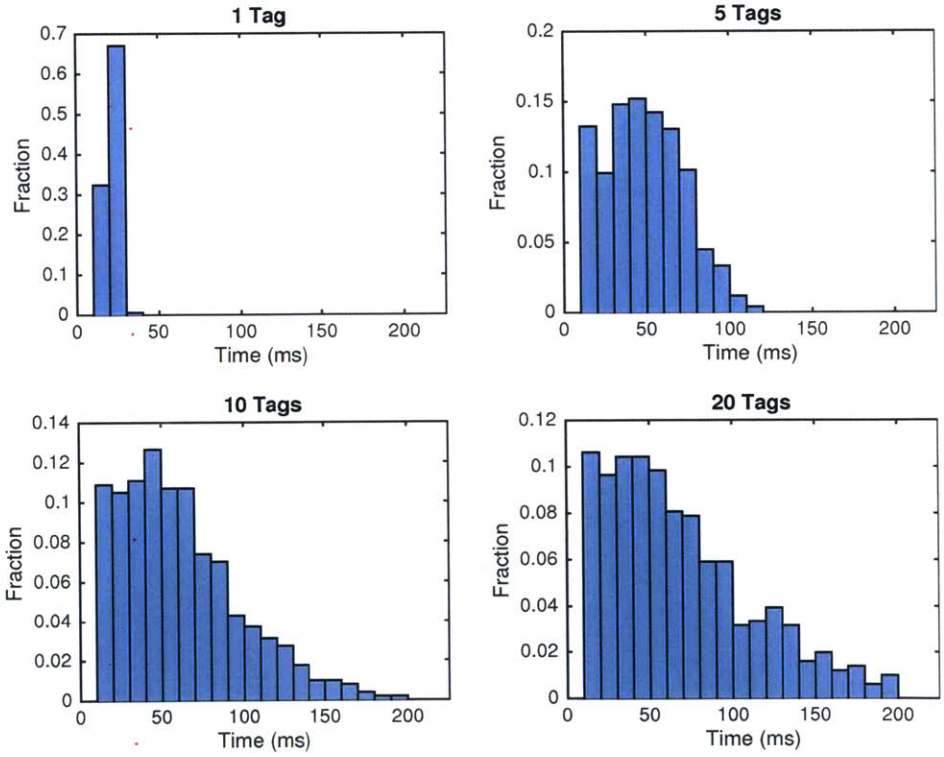
Step 3:


1 Tag
5 Tags
10 Tags
20 Tags
Fraction
Time (ms)

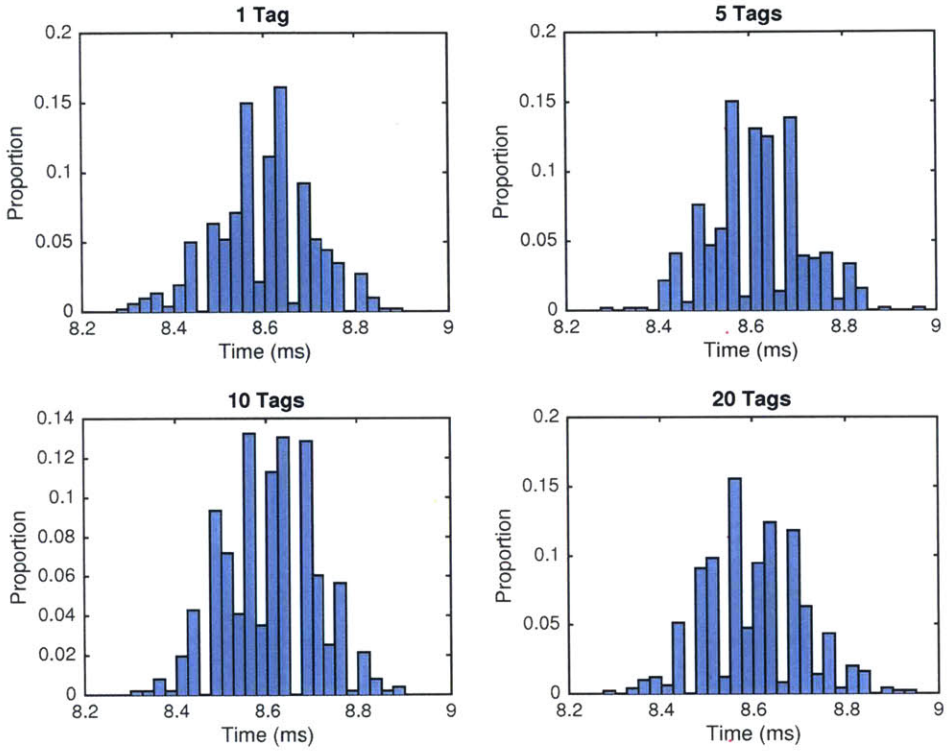
Step 4:


1 Tag
5 Tags
10 Tags
20 Tags
Fraction
Time (ms)

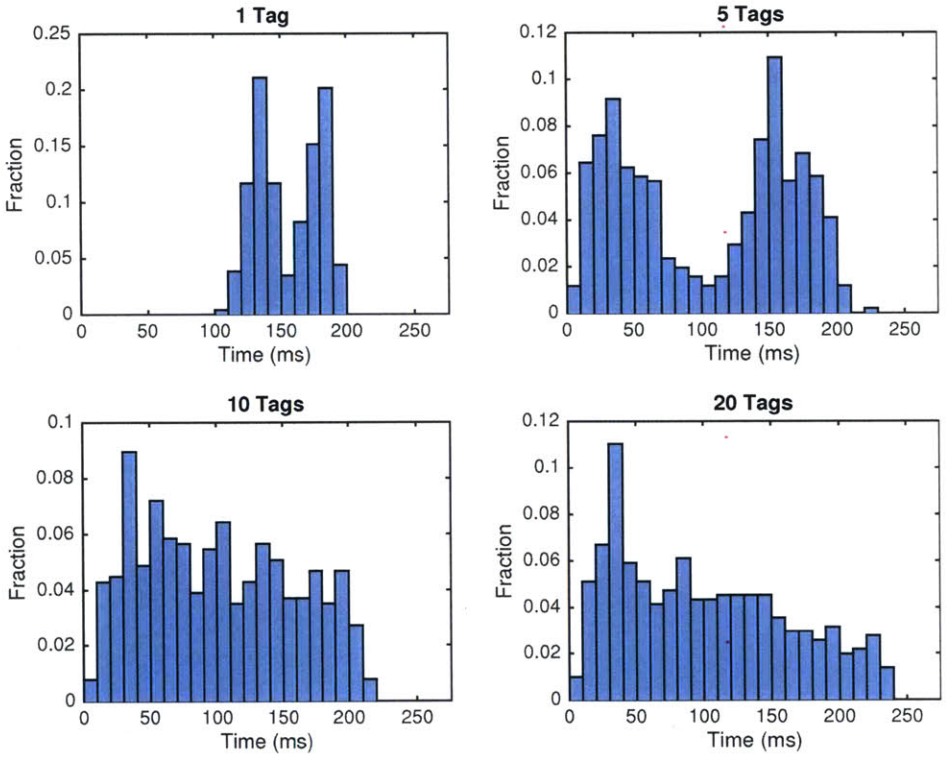
Step 5:


1 Tag
5 Tags
10 Tags
20 Tags
Fraction
Time (ms)

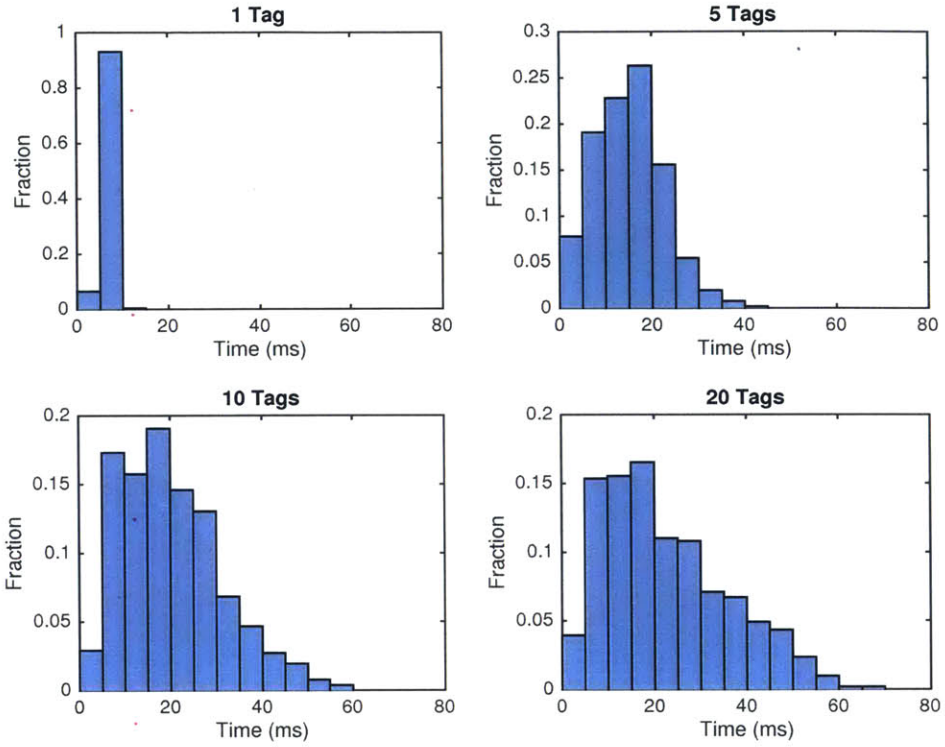
Step 6:


1 Tag
5 Tags
10 Tags
20 Tags
Proportion
Time (ms)

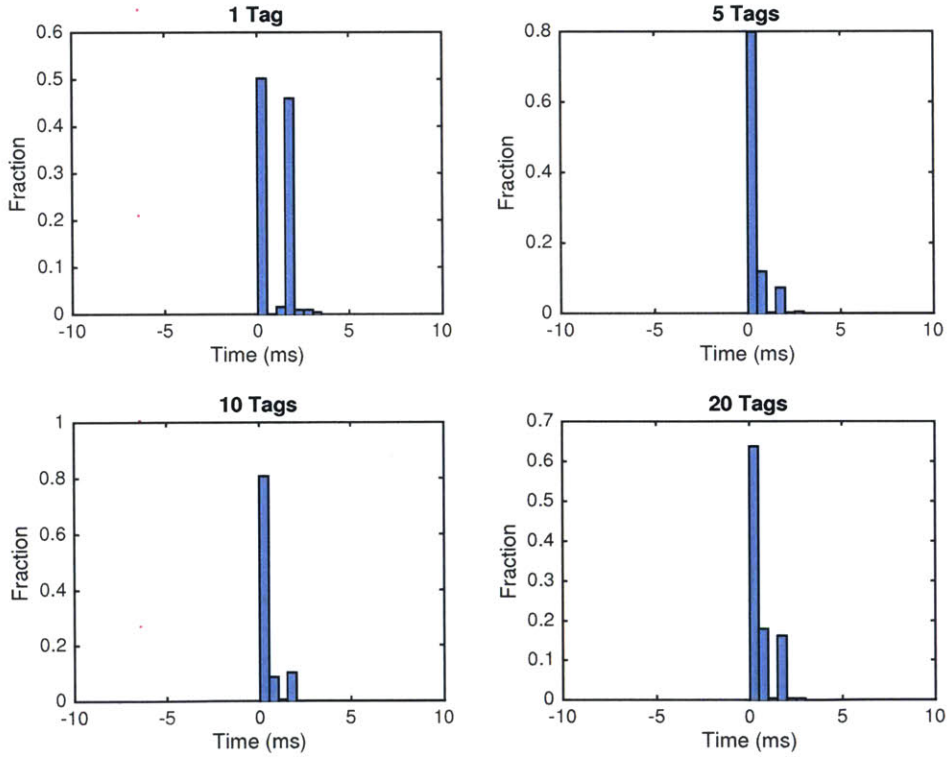
Step 7:


1 Tag
5 Tags
10 Tags
20 Tags
Fraction
Time (ms)

Step 8:

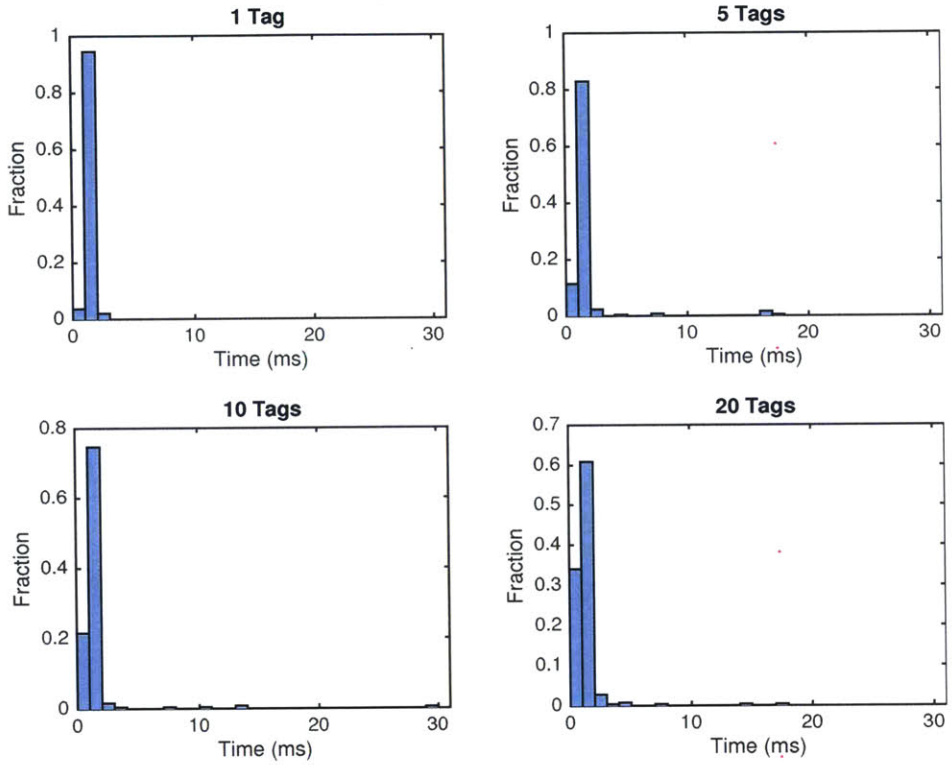

1 Tag
5 Tags
10 Tags
20 Tags
Fraction
Time (ms)

Step 9:

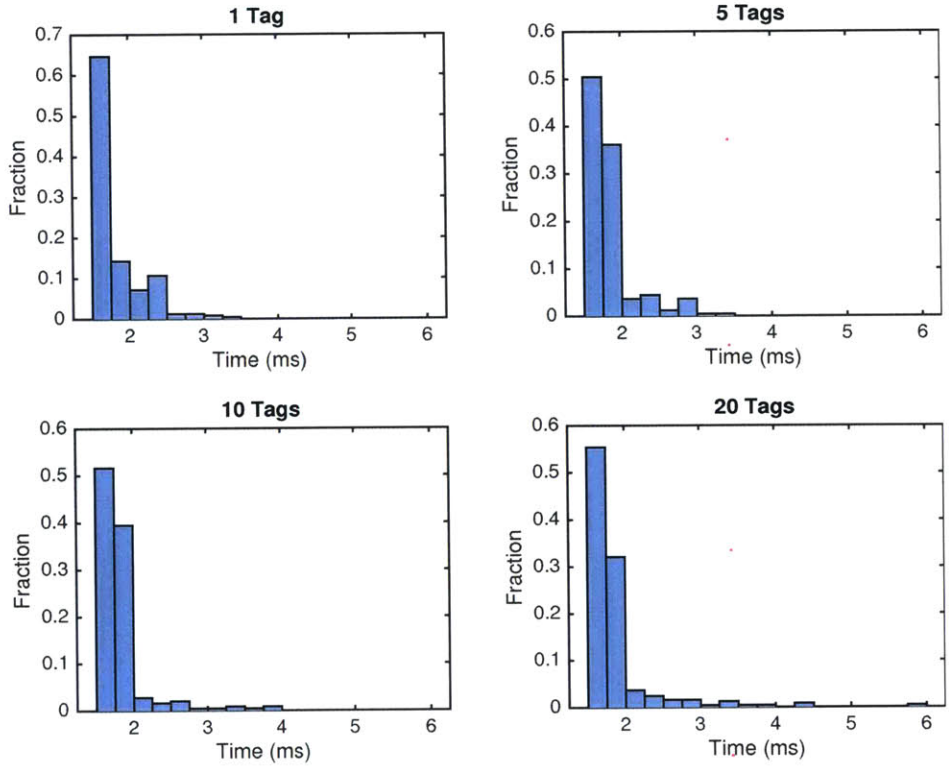

1 Tag
5 Tags
10 Tags
20 Tags
Fraction
Time (ms)

B.2 Write Test

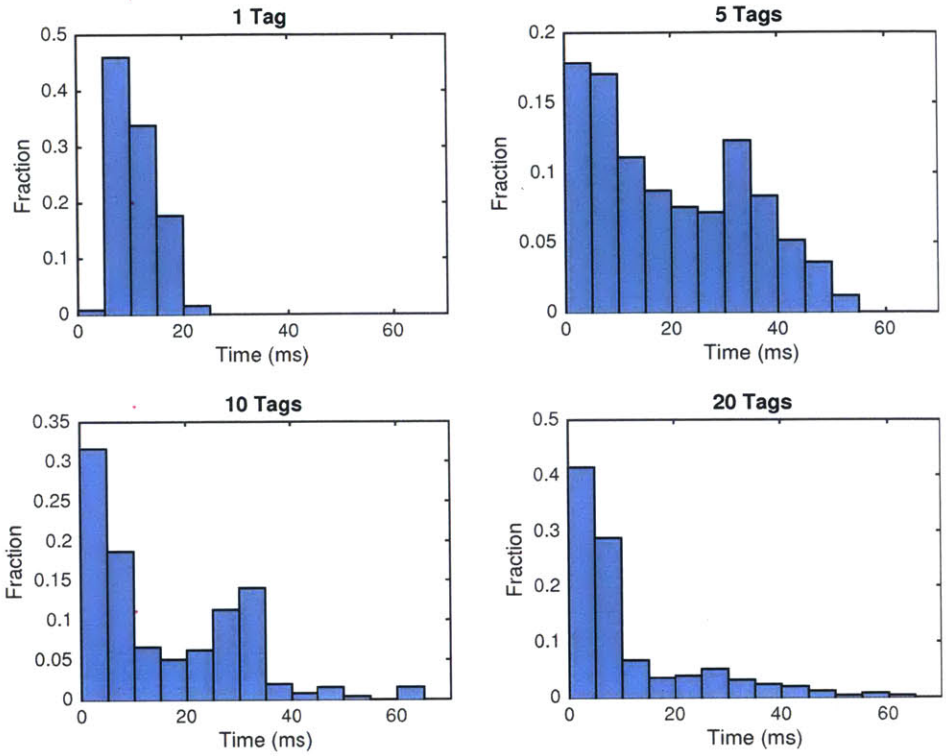
Step 1:



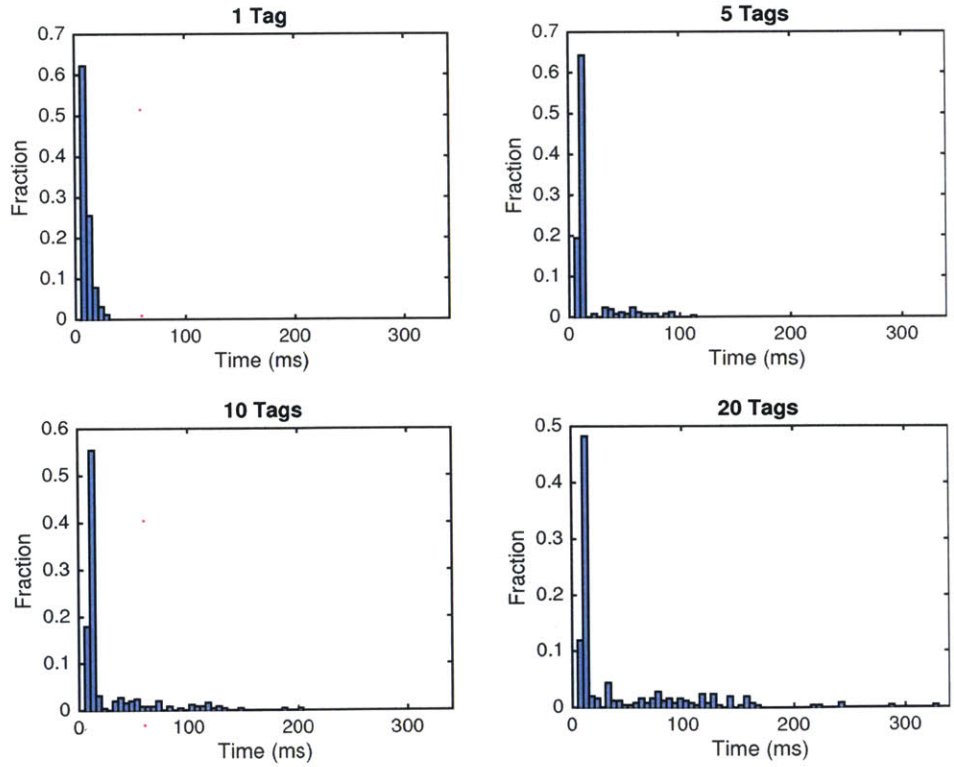
Step 2:



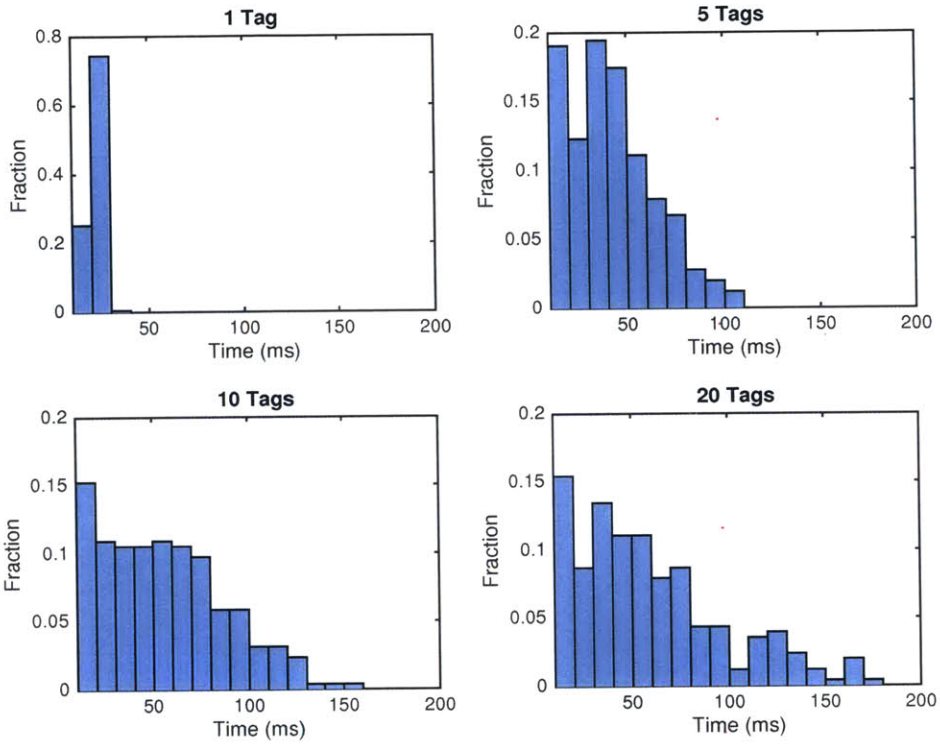
Step 3:


1 Tag
Fraction
Time (ms)
5 Tags
Fraction
Time (ms)
10 Tags
Fraction
Time (ms)
20 Tags
Fraction
Time (ms)

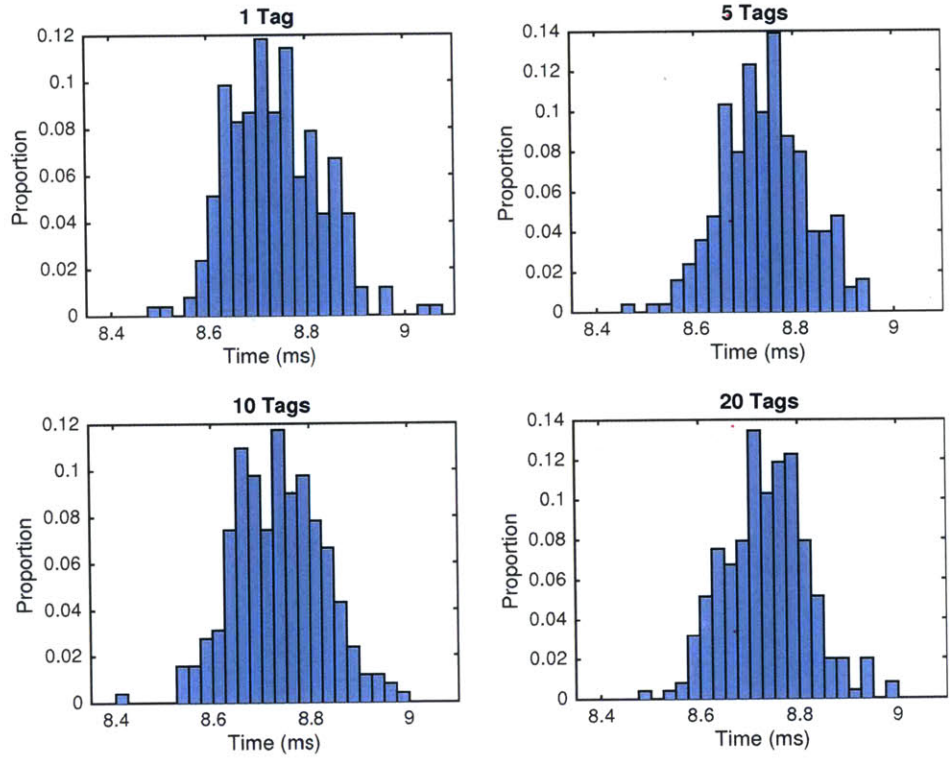
Step 4:


1 Tag
Fraction
Time (ms)
5 Tags
Fraction
Time (ms)
10 Tags
Fraction
Time (ms)
20 Tags
Fraction
Time (ms)

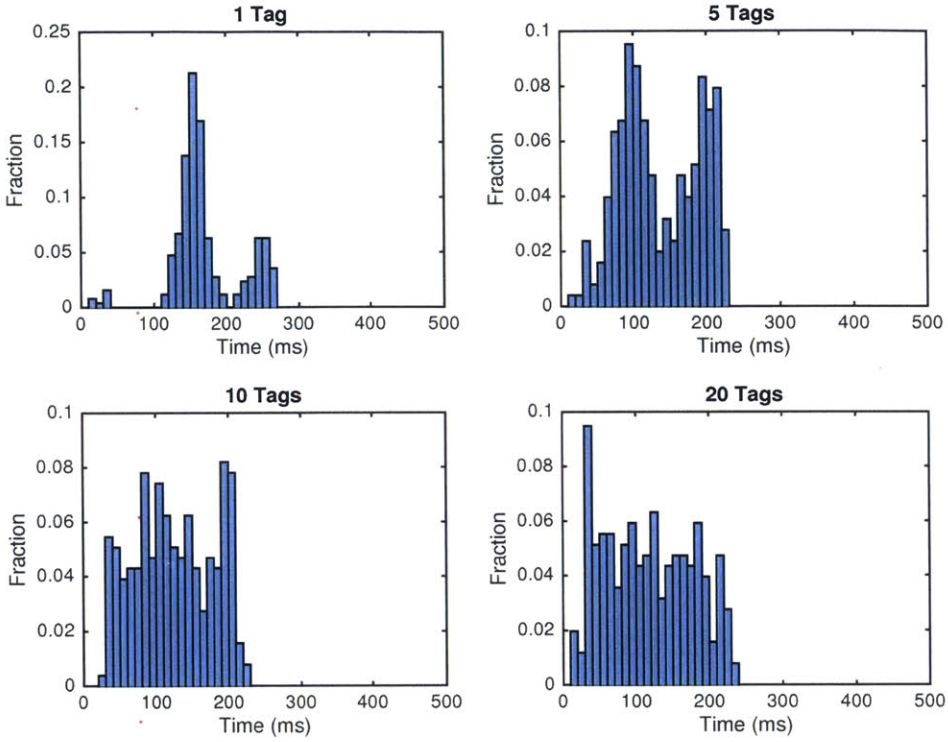
Step 5:


1 Tag
Fraction
Time (ms)
5 Tags
Fraction
Time (ms)
10 Tags
Fraction
Time (ms)
20 Tags
Fraction
Time (ms)

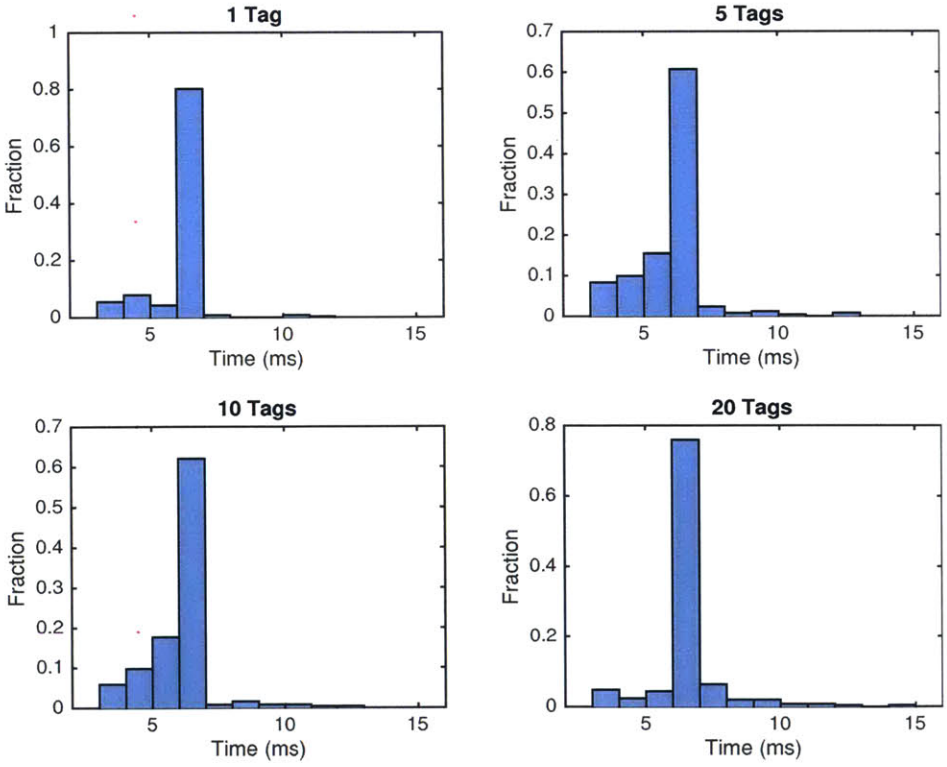
Step 6:


1 Tag
Proportion
Time (ms)
5 Tags
Proportion
Time (ms)
10 Tags
Proportion
Time (ms)
20 Tags
Proportion
Time (ms)

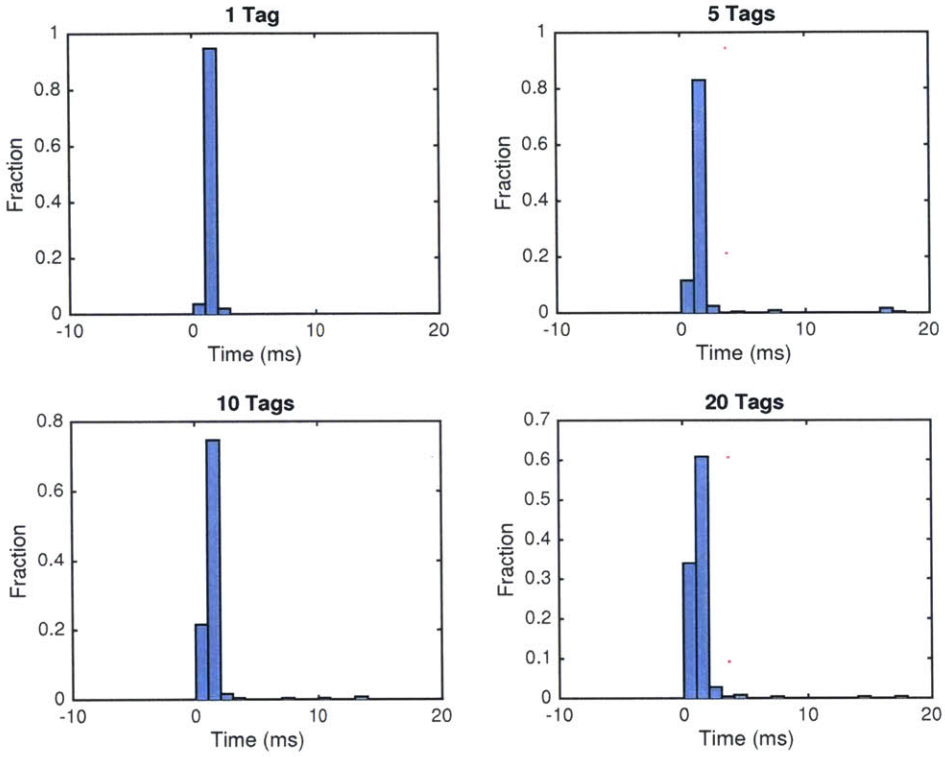
Step 7:


1 Tag
5 Tags
10 Tags
20 Tags
Fraction
Time (ms)

Step 8:

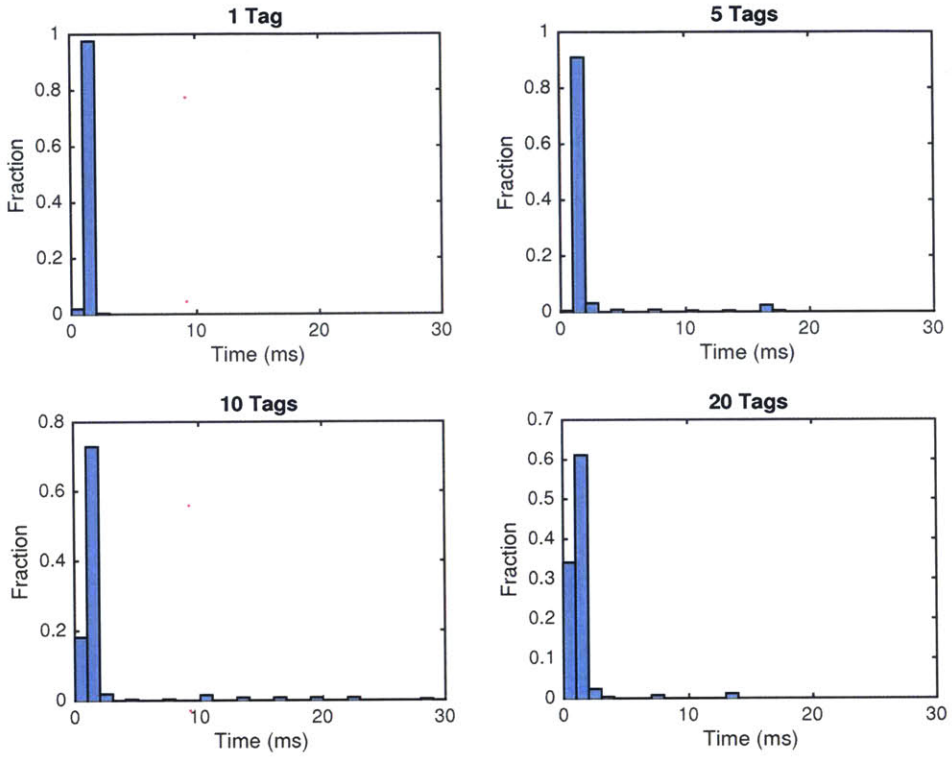

1 Tag
5 Tags
10 Tags
20 Tags
Fraction
Time (ms)

Step 9:

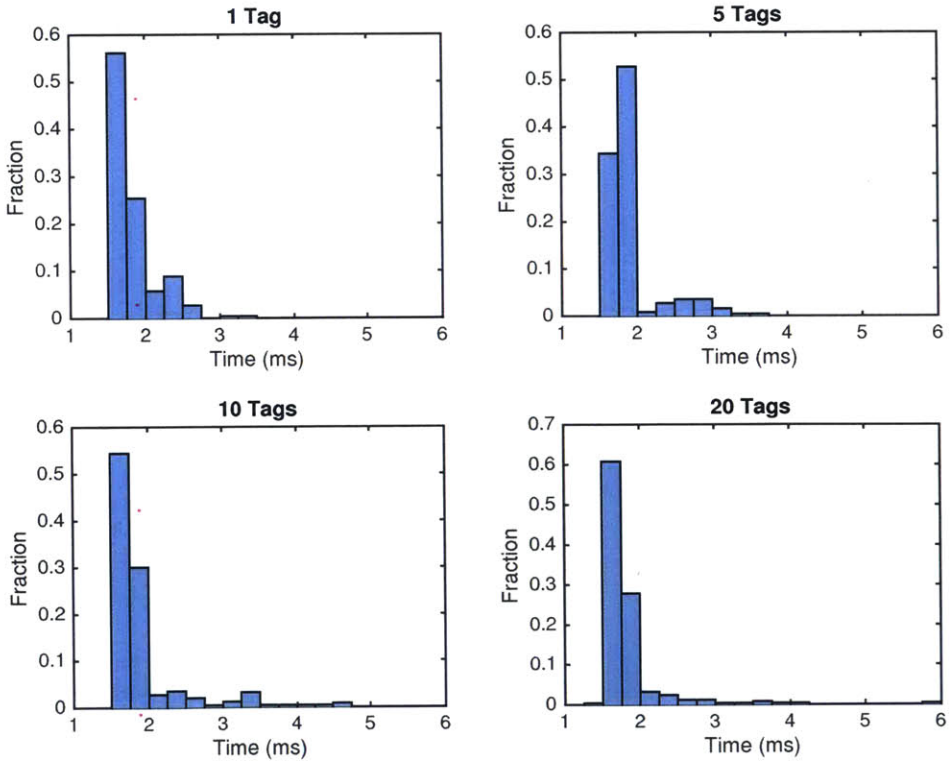


B.3 BlockWrite Test

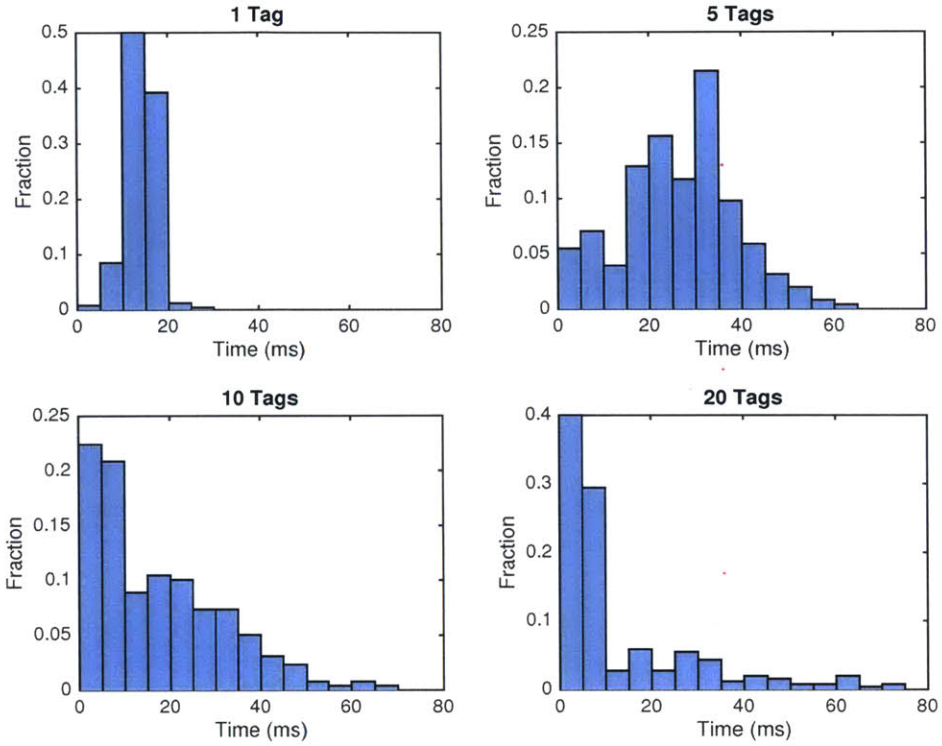
Step 1:



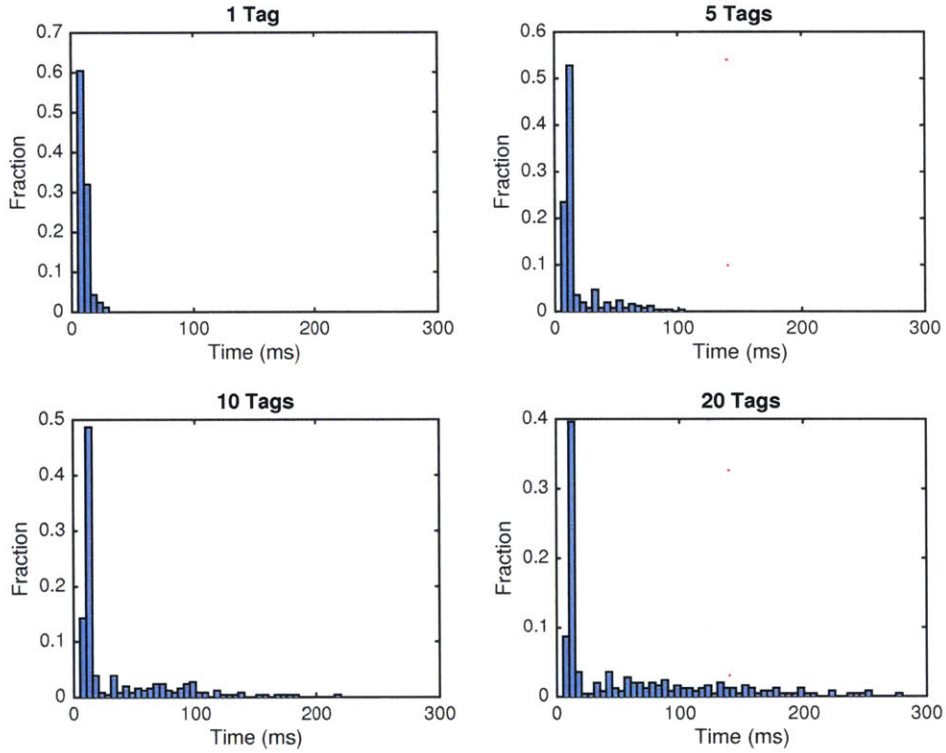
Step 2:



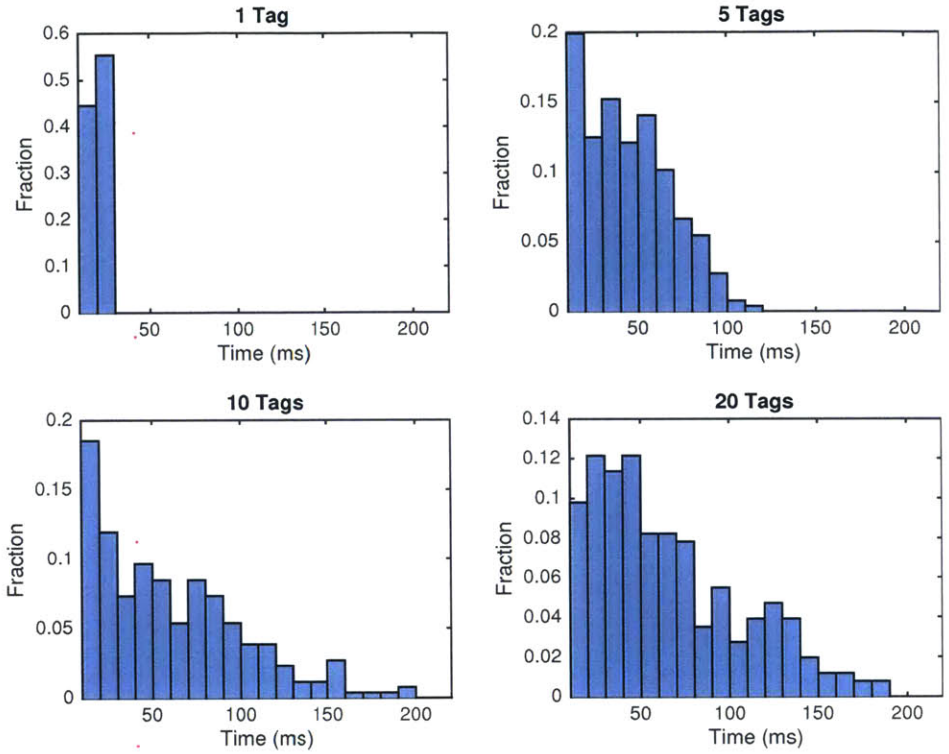
Step 3:


1 Tag
5 Tags
10 Tags
20 Tags
Fraction
Time (ms)

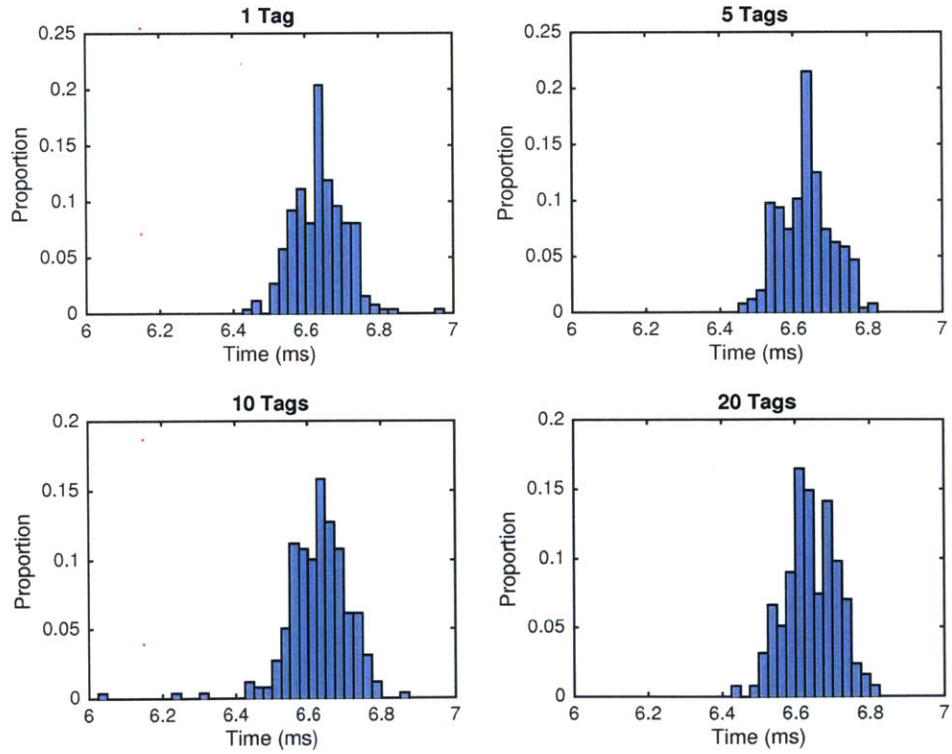
Step 4:


1 Tag
5 Tags
10 Tags
20 Tags
Fraction
Time (ms)

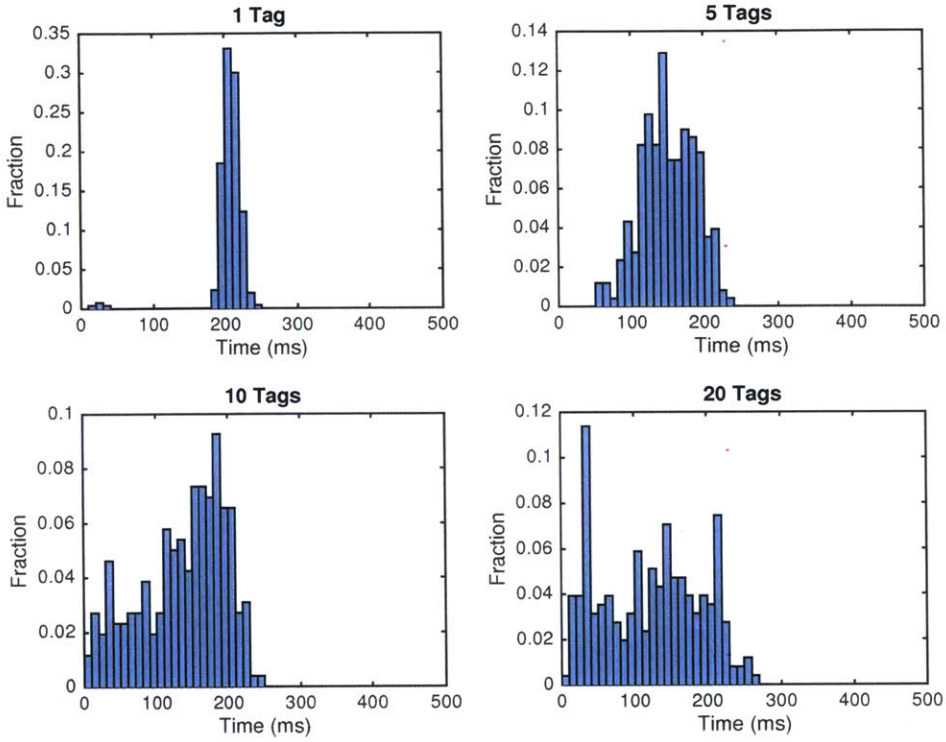
Step 5:


1 Tag
Fraction
Time (ms)
5 Tags
Fraction
Time (ms)
10 Tags
Fraction
Time (ms)
20 Tags
Fraction
Time (ms)

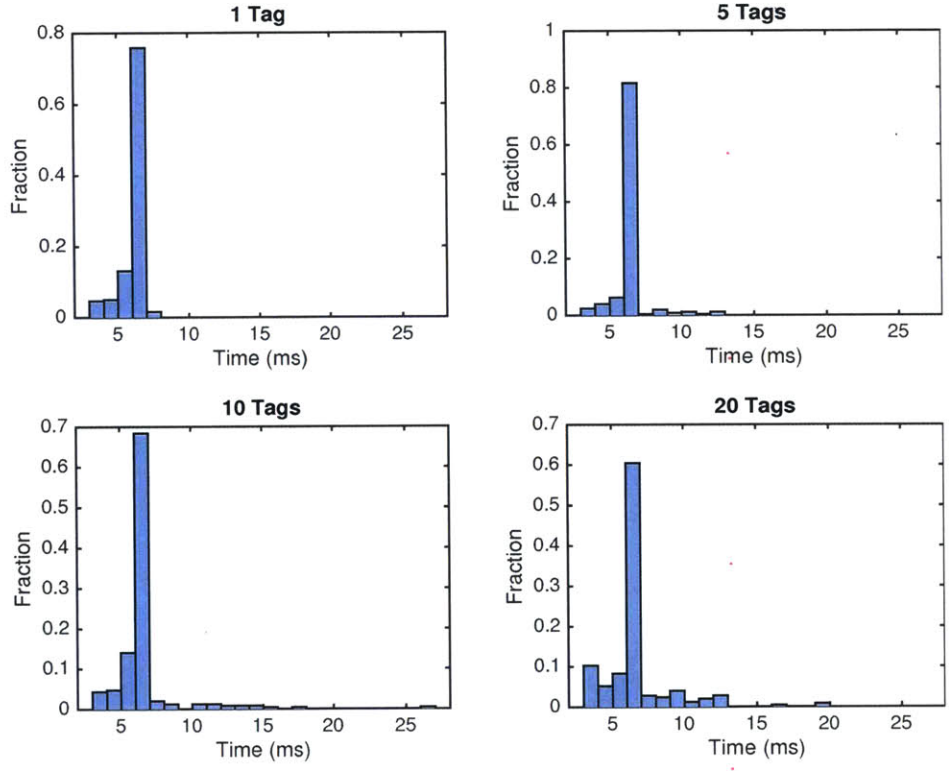
Step 6:


1 Tag
Proportion
Time (ms)
5 Tags
Proportion
Time (ms)
10 Tags
Proportion
Time (ms)
20 Tags
Proportion
Time (ms)

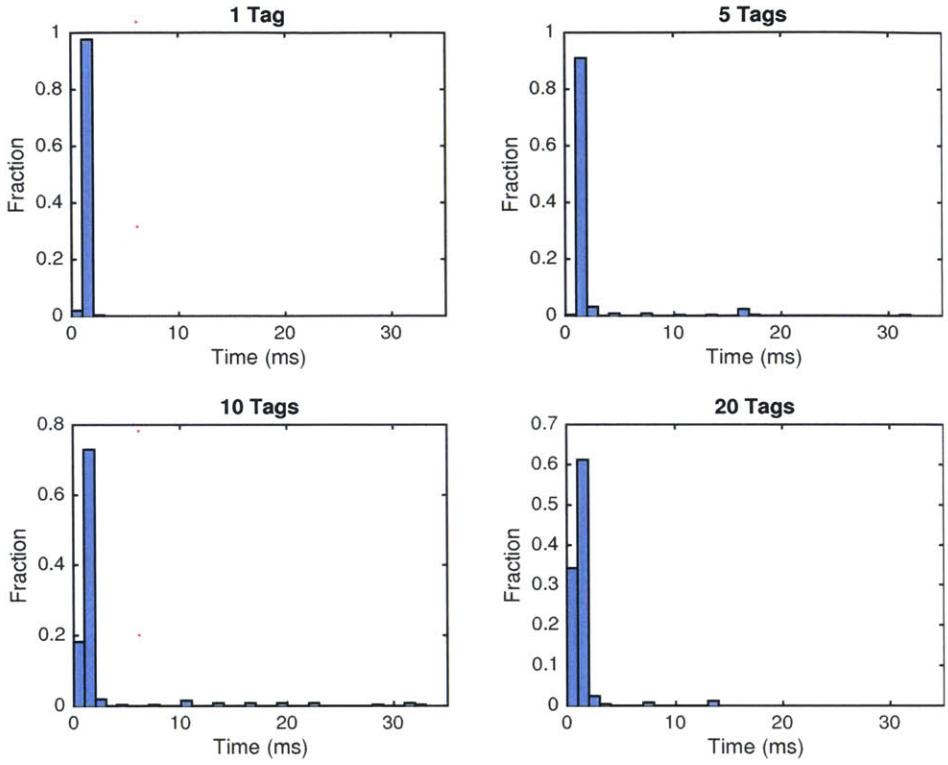
Step 7:


1 Tag
Fraction
Time (ms)
5 Tags
Fraction
Time (ms)
10 Tags
Fraction
Time (ms)
20 Tags
Fraction
Time (ms)

Step 8:


1 Tag
Fraction
Time (ms)
5 Tags
Fraction
Time (ms)
10 Tags
Fraction
Time (ms)
20 Tags
Fraction
Time (ms)

Step 9:



Bibliography

[1] M. P. Harald Bauer and J. Veira. (2014, December) The Internet of Things: Sizing up the opportunity. [Online]. Available: http://www.mckinsey.com/insights/high_tech_telecoms_internet/the_internet_of_things_sizing_up_the_opportunity

[2] How much does an RFID tag cost today? [Online]. Available: http://www.rfidjournal.com/faq/show?85

[3] D. M. Dobkin, *The RF in RFID*, 2nd ed. Elsevier Inc., 2013.

[4] "EPC Tag Data Standard," 2014.

[5] "EPC Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID," 2006.

[6] T. Scharfeld, "An analysis of the fundamental constraints on low cost passive radio-frequency identification system design," Master's thesis, Massachusetts Institute of Technology, September 2001.

[7] C. Floerkemeier, "Infrastructure support for rfid systems," Ph.D. dissertation, ETH Zurich, Zurich, Switzerland, Jul. 2006.

[8] "Low level reader protocol." [Online]. Available: http://www.gs1.org/epcrfid/epc-rfid-llrp/latest

[9] I. Ehrenberg, C. Floerkemeier, and S. Sarma, "Inventory management with an RFID-equipped mobile robot," in *Automation Science and Engineering, 2007. CASE 2007. IEEE International Conference on*, Sept 2007, pp. 1020–1026.

[10] R. Bhattacharyya, C. Floerkemeier, and S. Sarma, "RFID tag antenna based sensing: Does your beverage glass need a refill?" in *RFID, 2010 IEEE International Conference on*, April 2010, pp. 126–133.

[11] E. Md Amin, R. Bhattacharyya, S. Sarma, and N. Karmakar, "Chipless RFID tag for light sensing," in *Antennas and Propagation Society International Symposium, 2014 IEEE*, July 2014, pp. 1308–1309.

[12] A. Hasan, R. Bhattacharyya, and S. Sarma, "A monopole-coupled RFID sensor for pervasive soil moisture monitoring," in *Antennas and Propagation Society International Symposium, 2013 IEEE*, July 2013, pp. 2309–2310.

[13] J. R. Smith, A. P. Sample, P. S. Powledge, S. Roy, and A. Mamishev, *A Wirelessly-Powered Platform for Sensing and Computation.* Springer Berlin Heidelberg, 2006, vol. 4206, pp. 495–506. [Online]. Available: http://dx.doi.org/10.1007/11853565_29

[14] J. R. Smith, B. Jiang, S. Roy, M. Philipose, K. Sundara-Rajan, and A. Mamishev, *ID Modulation: Embedding Sensor Data in an RFID Timeseries.* Springer Berlin Heidelberg, 2005, vol. 3727, pp. 234–246. [Online]. Available: http://dx.doi.org/10.1007/11558859_18

[15] A. Sample, D. Yeager, P. Powledge, A. Mamishev, and J. Smith, "Design of an RFID-based battery-free programmable sensing platform," *Instrumentation and Measurement, IEEE Transactions on*, vol. 57, no. 11, pp. 2608–2615, Nov 2008.

[16] V. Talla and J. Smith, "Hybrid analog-digital backscatter: A new approach for battery-free sensing," in *RFID, 2013 IEEE International Conference on*, April 2013, pp. 74–81.

[17] S. Naderiparizi, A. Parks, Z. Kapetanovic, B. Ransford, and J. Smith, "WISP-Cam: A battery-free RFID camera," in *RFID, 2015 IEEE International Conference on*, April 2015, pp. 166–173.

[18] R. Stross, "Digital Tags Help Ensure the Price is Right," *The New York Times*, p. BU3, 02/09/2013. [Online]. Available: http://www.nytimes.com/2013/02/10/technology/digital-tags-help-ensure-that-the-price-is-right.html?_r=0

[19] "SL3S3021FHK." [Online]. Available: http://www.nxp.com/products/identification_and_security/nfc_and_reader_ics/connected_tag_solutions/SL3S4021FHK.html

[20] 32L0538Discovery. [Online]. Available: http://www.st.com/web/catalog/tools/FM116/SC959/SS1532/LN1848/PF260319?sc=stm32l0-discovery

[21] "Wikipedia - irobot create." [Online]. Available: https://en.wikipedia.org/wiki/IRobot_Create

[22] "Xbox 360 controller for windows." [Online]. Available: https://www.microsoft.com/hardware/en-us/p/xbox-360-wireless-controller-for-windows/JR9-00011

[23] "Arduino mega 2560." [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardMega2560

[24] "Sllurp." [Online]. Available: https://github.com/ransford/sllurp

[25] "Advanced RFID measurements: Basic theory to protocol conformance test." [Online]. Available: http://www.ni.com/tutorial/6645/en/

[26] "nuand BladeRF." [Online]. Available: https://nuand.com/

[27] "Hdd write throughput average." [Online]. Available: http://www.tomshardware.com/charts/enterprise-hdd-charts/-04-Write-Throughput-Average-h2benchw-3.16,3376.html

[28] "STM32F4 Nucleo." [Online]. Available: http://www.st.com/web/catalog/tools/FM116/SC959/SS1532/LN1847/PF260000?icmp=nucleo-ipf_pron_pr-nucleo_feb2014&sc=nucleoF401RE-pr

[29] "CC110L RF BoosterPack." [Online]. Available: http://www.ti.com/tool/430boost-cc110l

[30] "Saleae Logic. The logic analyzer you'll love to use." [Online]. Available: https://www.saleae.com/

[31] P. Zhang, P. Hu, V. Pasikanti, and D. Ganesan, "EkhoNet: High speed ultra low-power backscatter for next generation sensors," in *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '14. New York, NY, USA: ACM, 2014, pp. 557–568. [Online]. Available: http://doi.acm.org/10.1145/2639108.2639138

[32] M. Buettner and D. Wetherall, "A software radio-based uhf rfid reader for phy/mac experimentation," in *RFID (RFID), 2011 IEEE International Conference on*, April 2011, pp. 134–141.

[33] Semtech, "Application note: Fcc regulations for ism band devices: 902 - 928 mhz." [Online]. Available: https://www.semtech.com/images/promo/FCC_Part15_regulations_Semtech.pdf