AUTOMATIC SILENCING OF CELLULAR PHONES

USING BLUETOOTH COMMUNICATION

by

CHAD DANIEL LARSH, B.S.

A THESIS

IN

ELECTRICAL ENGINEERING

Submitted to the Graduate Faculty
of Texas Tech University in
Partial Fulfillment of
the Requirements for
the Degree of

MASTER OF SCIENCE

IN

ELECTRICAL ENGINEERING

Approved

Brian Nutter
Chairperson of the Committee

Tanja Karp

Accepted

John Borrelli
Dean of the Graduate School

May, 2006

ACKNOWLEDGEMENTS

First of all, I would like to thank my Lord and Savior, Jesus Christ, for showing me the idea for this topic and for sustaining me as I struggled through the project. I would like to thank Dr. Brian Nutter for all of his help and encouragement. His intelligence allows him to understand and solve problems more efficiently than anyone I have ever met. I would also like to thank Dr. Tanja Karp for her willingness to be on my thesis committee and for evaluating my work.

Finally, I would like to thank my family and friends, for supporting me as I work and for encouraging me every step of the way.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS i

LIST OF FIGURES v

CHAPTER

I. INTRODUCTION 1

II. BLUETOOTH ESSENTIALS 2

Piconet 2

Frequency Hopping 3

Bluetooth Channels 4

Low Power Consumption 5

Range of Communication 5

Protocols and Protocol Stacks 5

Scatternet 6

Device Inquiry 7

Time to Acquire 8

III. CURRENT SOLUTION 10

Problems 11

IV. SYSTEM DESCRIPTION 13

Timing Concerns 13

Walking Speeds 14

Solution: A Scatternet 14

Physical Specifications 15
Bluetooth Antenna 15
Zone Dimensions 19
Software Algorithm 20
Buffer Entry Structure 20
Software Flowchart 22
An Example 24
Possible Walk Patterns 27
Final Thoughts 28
Why Bluetooth? 28
Why communicate between the two separate zones? 29
V. WINSOCK IMPLEMENTATION 30
Winsock Limitations 30
Accomplishments 31
Conclusion 34
VI. FUTURE WORK 35
REFERENCES 37
APPENDIX 38

LIST OF FIGURES

2.1 – Example Piconet 3

2.2 – Bluetooth Protocol Stack 5

2.3 – Scatternet 7

2.4 - Inquiry Scan & Page Scan Timings 8

3.1- Q-Node 10

3.2 - Example Q-Node Coverage 11

4.1 - Antenna Radiation Pattern 16

4.2 - Communication Zones (Perspective) 17

4.3 - Communication Zones (Side) 18

4.4 - Zone Dimensions 19

4.5 - Buffer Entry Structure 20

4.6 - Software Flowchart for Inside Device 22

4.7 - An Example 27

5.1 - Laptop to Cell Phone Connection 32

5.2 – Activate Example 33

5.3 – Silence Example 34

6.1 - Smart Modular Bluetooth Minimodule Development Kit 35

CHAPTER 1

INTRODUCTION

As of October of 2005, there were 195 million cell phone subscribers in the United States alone[4]. In today's digital age, people are never far from their cell phone or PDA. These devices, unfortunately, have audible alerts that are not welcome in certain locations. For example, in movie theaters, concert halls and houses of worship, a ringing cell phone is an embarrassment to the person and a distraction to all those around. These people are usually not intending to be a distraction; they simply forgot to change to a silent profile upon entering the venue. Bluetooth is quickly becoming a standard in the communication industry especially in the area of cell phones. This paper will discuss using Bluetooth to automatically silence a cell phone or other device when the user enters a room where these audible alerts are a distraction. The device will then reactivate the phone upon leaving the room. This will be an improvement to the Q-Node created by Bluelinx Inc. For clarity in the remainder of this paper, the device being described in this paper will be referred to as the device, and any Bluetooth enabled device that could be silenced or activated will be referred to as the phone.

CHAPTER 2

BLUETOOTH ESSENTIALS

"Bluetooth wireless technology is a short-range communications system intended to replace the cable(s) connecting portable and/or fixed electronic devices. The key features of Bluetooth wireless technology are robustness, low power, and low cost. Many features of the core specification are optional, allowing product differentiation [1]." As Bluetooth technology continues to grow and become more prevalent in society, it has become a staple aspect of most new cell phones being manufactured and sold. Bluetooth has built-in support for streaming audio and thus offers a great method to connect hands-free devices to a cell phone or even to a PDA.

Piconet

Bluetooth is a wireless protocol, and thus it is only capable of communicating with other Bluetooth devices that are within its range. A group of Bluetooth devices that are in range and communicating with each other is called a piconet. In a piconet, one of the Bluetooth devices is designated as the master, and all of the other devices are slaves. Bluetooth communication is based on time slots, and in a piconet, the master assigns the timeslots for the entire piconet. As a result, only masters-slave communication can be achieved. It is impossible for a master to talk to a master or for a slave to talk to a slave. An example of a Bluetooth piconet is shown in figure 2.1. In order for a piconet to exist, all of the devices must be synchronized with each other in both clock and frequency hopping pattern.

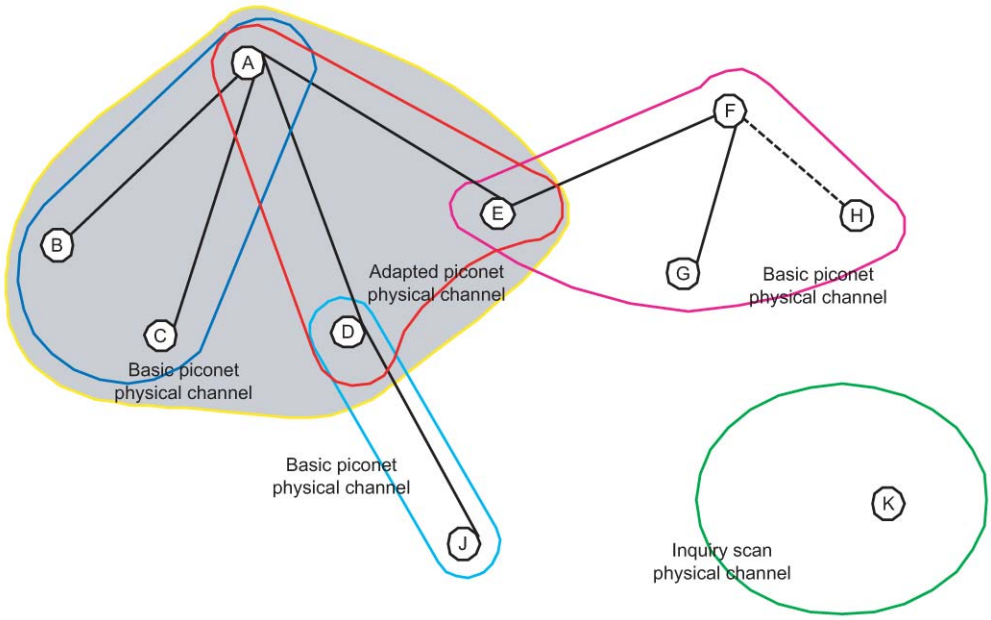


Figure 2.1 – Example Piconet[1]

Frequency Hopping

The Bluetooth protocol operates in the 2.4GHz region of the industrial, scientific and medical (ISM) band specified by the International Telecommunication Union (ITU). This region of frequencies is used by many different types of devices, so interference is a common problem with devices that operate in this frequency band. In an effort to combat this problem, Bluetooth uses a frequency hopping pattern that constantly changes frequencies to prevent interruption caused by interference. Bluetooth devices hop between 79 different frequencies that all reside in the ISM band in a pseudo-random pattern based on the unique Bluetooth address of the master. When a piconet is formed, the device that initiated the inquiry scan is decided to be the master and creates the frequency hopping pattern. As other slaves join the piconet, they are able to follow the pattern by simply knowing the address of the master and knowing the algorithm to follow

the hopping pattern. The master device then claims the master clock for the piconet, and all of the slave devices must synchronize their clock to the clock of the master for the remainder of the time in that particular piconet. This clock synchronization is not exact; the devices are able to synchronize close enough by setting a clock offset between their own clock and the communicated clock of the master.

Bluetooth Channels

Four different physical channels are used in the Bluetooth environment to communicate at the hardware level. Each of these channels has a unique purpose and is designed with characteristics to achieve this purpose. The four Bluetooth channels are the basic piconet channel, the adapted piconet channel, the inquiry scan channel and the page scan channel. Each Bluetooth device can only communicate on one physical channel at any given time. The inquiry scan channel is used by devices to locate other Bluetooth devices in the area. This channel limits the number of channels in the hopping pattern to 32. Once a Bluetooth device is located using the inquiry scan channel, the page scan channel is used to initiate a connection with the located device. Once a device is connected and the clock and hopping patterns are synchronized, then the device can enter either the basic or the adapted piconet channel. The adapted piconet channels main difference lies in its ability to limit the number of frequencies in the hopping pattern to less than 79.

Low Power Consumption

Most Bluetooth devices, especially those that are used in cell phones, are Class III Bluetooth devices. The power consumption for a Class III Bluetooth device is approximately 1 mW per transmission. This makes it a very useful communication method for short range communication.

Range of Communication

The communication range for a Class III Bluetooth device is 16-33 ft. Within this range, a device can reliably establish a piconet and communicate effectively.

Protocols and Protocol Stacks

Just like any other communication system, Bluetooth uses a protocol stack that is made of several different protocols. The protocol stack for Bluetooth can be seen in figure 2.2.

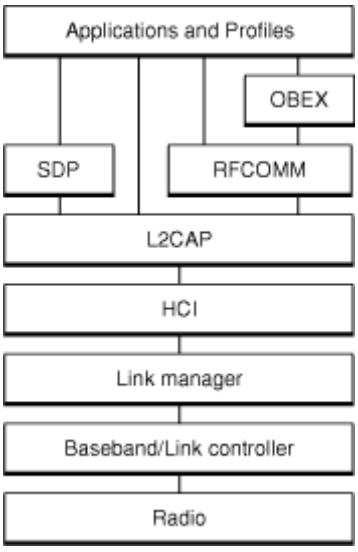


Figure 2.2 – Bluetooth Protocol Stack[6]

Above the physical protocols is the Logical Link Control and Adaptation Layer Protocol (L2CAP). The L2CAP protocol is responsible for both connection-oriented and connectionless communication over a Bluetooth link. In a connection-oriented environment, the L2CAP protocol manages the connection, and in either method, L2CAP is responsible for the segmentation and reassembly of information as it is sent over the physical channel.

Above the L2CAP protocol is the Radio Frequency Communication (RFCOMM) protocol. This protocol is used to emulate a serial port over a Bluetooth connection.

Bluetooth is still a fairly new protocol and, as a result, has not been fully standardized. At the development stage, there are two primary protocol stacks supported by two different companies. Microsoft has their protocol stack and uses it in their Windows programming environment as well as with Winsock. Broadcom (Widcomm) also has a protocol stack. Most non-Microsoft companies use the Broadcom stack, and most of the industry claims that the Broadcom stack is superior. Microsoft provides their Software Development Kit (SDK) for free, while Broadcom charges over $1,000 for their SDK.

Scatternet

It is possible for a Bluetooth device to communicate on more than one piconet at any given time. When two piconets exist in the same region and are connected by a common node, a scatternet forms. It is possible for a device on one piconet to communicate with a device on the other by using the common node as a repeater. In

order for a scatternet to form, one of the slaves on the first piconet must undergo a master/slave switch. When this happens, the slave device switches and becomes a master and begins an inquiry scan of its own to look for devices to join its piconet. The frequency hopping pattern and clock synchronization for this new piconet will be based on the address of the new master. Once the new piconet is formed, the device that switched from master to slave now has a dual role. It exists as a slave on one piconet and a master on another.

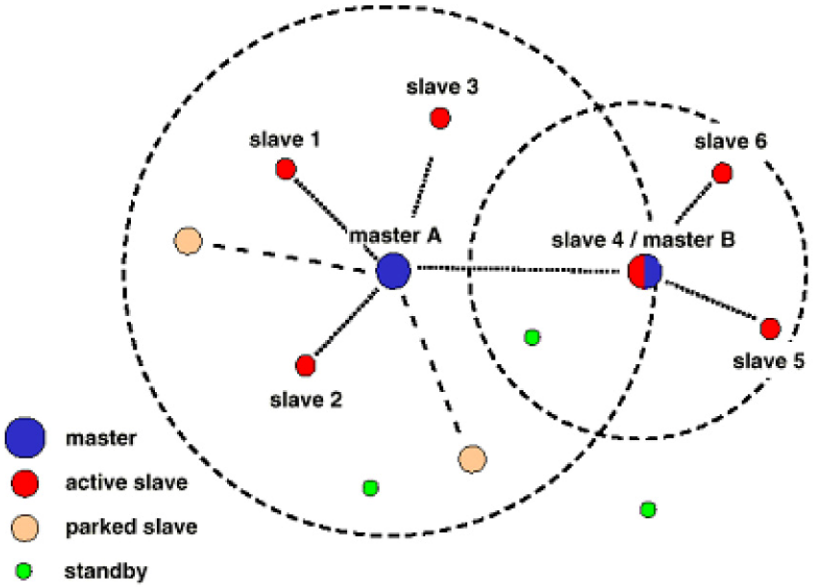


Figure 2.3 – Scatternet[7]

Device Inquiry

When a Bluetooth device looks for other devices in its area, it begins an inquiry scan on the inquiry scan physical channel. An access code is sent on each of the 32 frequencies of the inquiry scan channel and the device waits for a reply. A slave device, when listening on the inquiry scan channel, changes its listening frequency every 1.28

seconds. When the slave device receives the access code, it replies to the master with its own device address. At this point, both devices switch to the page scan channel in order to establish a connection. In the page scan channel, the master continues to hop the same 32 frequencies as the inquiry scan channel, except the page scan channel hops based on the address of the slave. The slave hops along the same pattern, again changing its listening frequency every 1.28 seconds. After the page scan is complete, the devices have formed a piconet and basic communication can begin.

Time to Acquire

The time required for both the inquiry scan and the page scan can obviously vary. While studying a new scatternet formation protocol, the following results show average timings that were found for both the inquiry scan and the page scan.

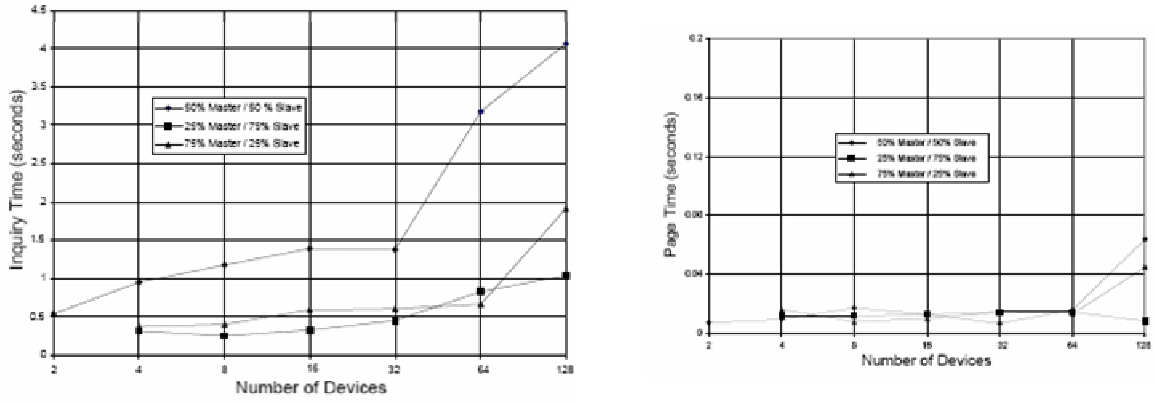


Figure 2.4 - Inquiry Scan & Page Scan Timings[2]

As mentioned, these times are for the formation of a scatternet, so it can be assumed that the time to create a simple piconet will be less than or equal to the times above. The inquiry scan typically lasts less than one second, while the page scan usually lasts under

0.04 seconds. According to the tests above, an average time for a complete Bluetooth connection is 1.4 seconds.

CHAPTER 3

CURRENT SOLUTION

Bluelinx Inc. is a company based in North Carolina that has designed a solution to automatically silence cell phones, and they call it a Q-Node.

Figure 3.1- Q-Node[5]

A Q-Node is a Bluetooth device that scans its communication region for any Bluetooth device that is configured to respond to it. Once a device enters the Q-Node region, the Q-Node sends the device a Bluetooth message notifying the device that it is in a Q-Zone and that it needs to change itself to a predefined mode with an undisturbing notification, whether this is a vibrating alert or simply no alert at all. Once the device leaves the Q-Zone, the device returns to its original state.

Problems

While the Q-Node does silence a cell phone as desired, it does have many potential problems and room for improvement. The Q-Node would be a great solution for a conference room where all users are located around a table or in a smaller area. But in a larger room like a movie theater or even a concert hall, the 10 meter range of a Q-Node is going to require several Q-Nodes scattered throughout the room. An example of a larger room with spaced Q-Nodes can be seen below. It would take 11 Q-Nodes to cover this very average sized church sanctuary.

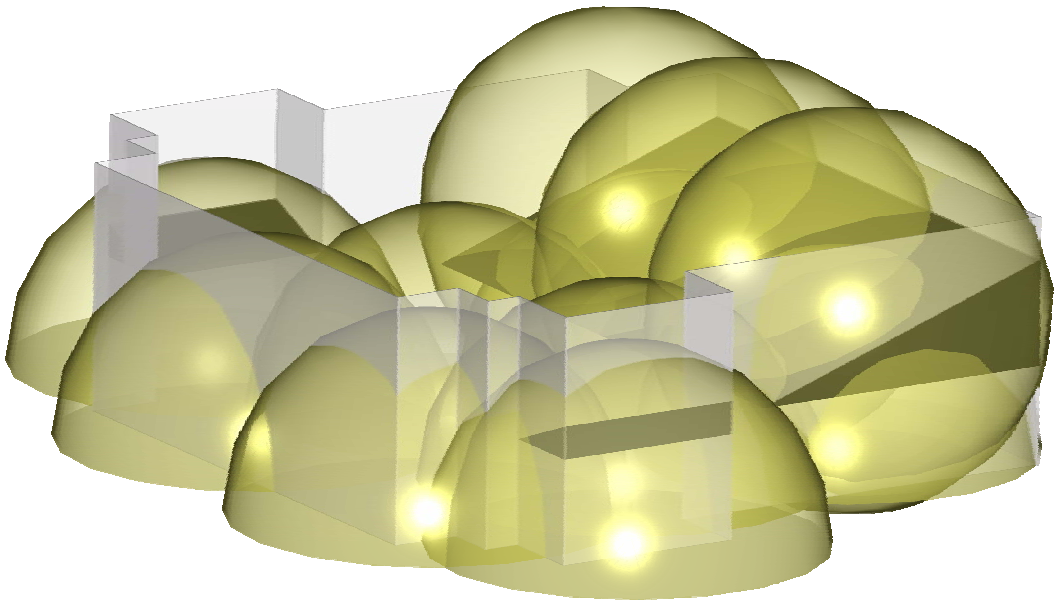


Figure 3.2 - Example Q-Node Coverage

Also, because the Q-Node is an electronic device, a power supply of some sort is required to power the Q-Node. This could either come in the form of batteries which would require changing, or the facility will have to run power to all of the devices scattered throughout the room. This is not very convenient and could possibly involve a major project to install a Q-Node system into an existing building.

Another place for improvement can be seen in the initialization process for a Bluetooth device to join a piconet. If the Q-Nodes are scattered throughout the room, they are probably going to each operate on their own piconet and will have their own clock synchronization and frequency hopping pattern. If a person were to walk around in the room, this would mean that every time they walked from one Q-Zone to another, the phone would have to go through the inquiry scan and page scan sequence to join a new piconet. Depending on how much moving around the user has to do, this could provide a significant battery drain on the phone. It would be much more convenient for the phone to be silenced and then no longer need to communicate with the Q-Node.

CHAPTER 4

SYSTEM DESCRIPTION

Whereas the Q-Node operates on the principle that as long as a device is within the Q-Zone, it will continue to be silenced, the goal of this project is to design a device that will monitor the entrances and exits to a particular room and make the appropriate changes to the phone as it either enters or leaves the room. This device would be much more practical for larger venues, as usually there are many fewer doors than there is area to cover in a given room. A movie theater, for example, may have several thousand square feet inside but usually only one or two doors to enter or leave. The device must not only be able to discern when the user walks through the door, it must also be able to determine the direction of travel through the door. It makes a difference whether the person is coming into the room or leaving the room. The goal is to silence the device as the user enters the room, and then return the device to its original state upon exiting the room. The project described below is a two part device. One piece of the device will mount on the wall on the outside of the door of the room to be controlled, and the other part will mount on the inside. Mounting these devices on opposite sides of the wall will provide isolated regions in which to operate and will also allow the devices to communicate with each other.

Timing Concerns

The most important concern in this project is timing. The Bluetooth device in a Q-Zone setting is in constant communication with the Q-Node. If for some reason the

first communication with the device is not successful, it would only mean that the phone remain active for a couple of seconds until the Q-Node tries again. In the project described here, however, the device must be able to establish a connection with the device and silence it while the user is walking through the door. If for some reason the exchange is unsuccessful, the device would remain active the entire time it is in the room, or at least until the user left and came back.

Walking Speeds

Richard L. Knoblauch, Martin T. Pietrucha, and Marsha Nitzburg performed a study, and their results were reported in the paper "Field Studies of Pedestrian Walking Speed and Start-Up Time[3]." This study was designed to understand people's response time and walking speed in order to determine timing for pedestrian crosswalks. In their study of 7,123 pedestrians it was determined that the average walking speed is 4.11 feet/second for people over the age of 65 and 4.95 feet/second for people under 65. A crosswalk is usually a place where people are in a hurry, whether it is because they are late or because they simply want to get out of the street, so for the remainder of this project, a normal walking speed of 5 feet/second will be used.

Solution: A Scatternet

As a Bluetooth phone enters into one of the zones, whether it be the inside zone or the outside zone, it must join a piconet in order to communicate with the device. If both parts of the device are a Bluetooth master, there will be no way for the devices to communicate with each other, because the Bluetooth specification does not allow for

communication between masters. The solution to the problem is a scatternet. The outside device will create a piconet and the inside device will be a slave on this piconet. The inside device will then perform the master/slave switch and create a piconet inside the door. The inside device will maintain its dual role, as the master of its own piconet and a slave to the piconet located outside the door. This will allow a phone to communicate with a master no matter which zone it enters, and it will also allow the devices to communicate with each other.

Physical Specifications

Since the device is designed to be mounted above the door, and most venues that would use a device similar to this have double wide doors, all of the drawings used in this project show a double wide door with standard dimensions of 64 inches wide and 80 inches tall. The device is mounted on the wall above the door, with one piece on each side of the door at a height of 90 inches.

Bluetooth Antenna

The Bluetooth device described requires a specific Bluetooth antenna. Most Bluetooth devices use an omni directional antenna in order to provide the largest coverage area and allow communication with the most devices. In this specific implementation, the goal is to create two separate zones in order to distinguish the direction of travel. This goal requires a directional antenna of some sort. The primary goal is to create two separate zones, but it would also be beneficial for the antenna to be able to communicate with the other device mounted on the other side of the wall. If this

can not be accomplished with one antenna, the device will either require two antennas or a wire between the devices. At first, a parabolic antenna was tried, but this created a very elongated region and did not provide the maximum area. It did not allow the two devices to communicate with each other. Also, parabolic antennas have a large physical size, and to obtain the beam width that was desired, it would have required a fairly large antenna.

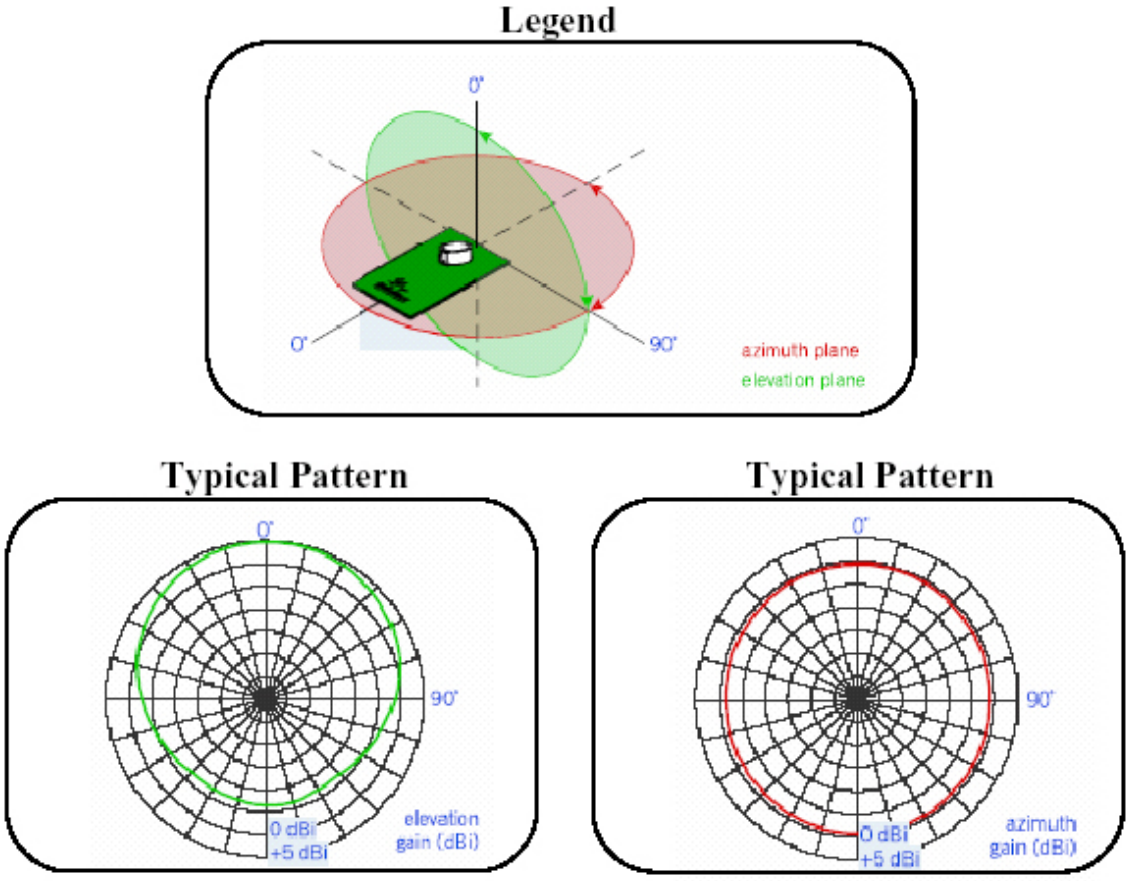


Figure 4.1 - Antenna Radiation Pattern[8]

The antenna that was chosen for this project is a hemispherical antenna. An example of such an antenna is the Tyco 1513151-1. The radiation pattern from the datasheet can be seen above, and the entire datasheet can be found in appendix A. The hemispherical antenna allows for a very broad region to locate and acquire a Bluetooth

device as the user is passing by, but it also allows for the two parts of the device to communicate with each other, eliminating the need for a secondary antenna. With the two antennas located on opposite sides of the door, the two regions would be generated as shown below.

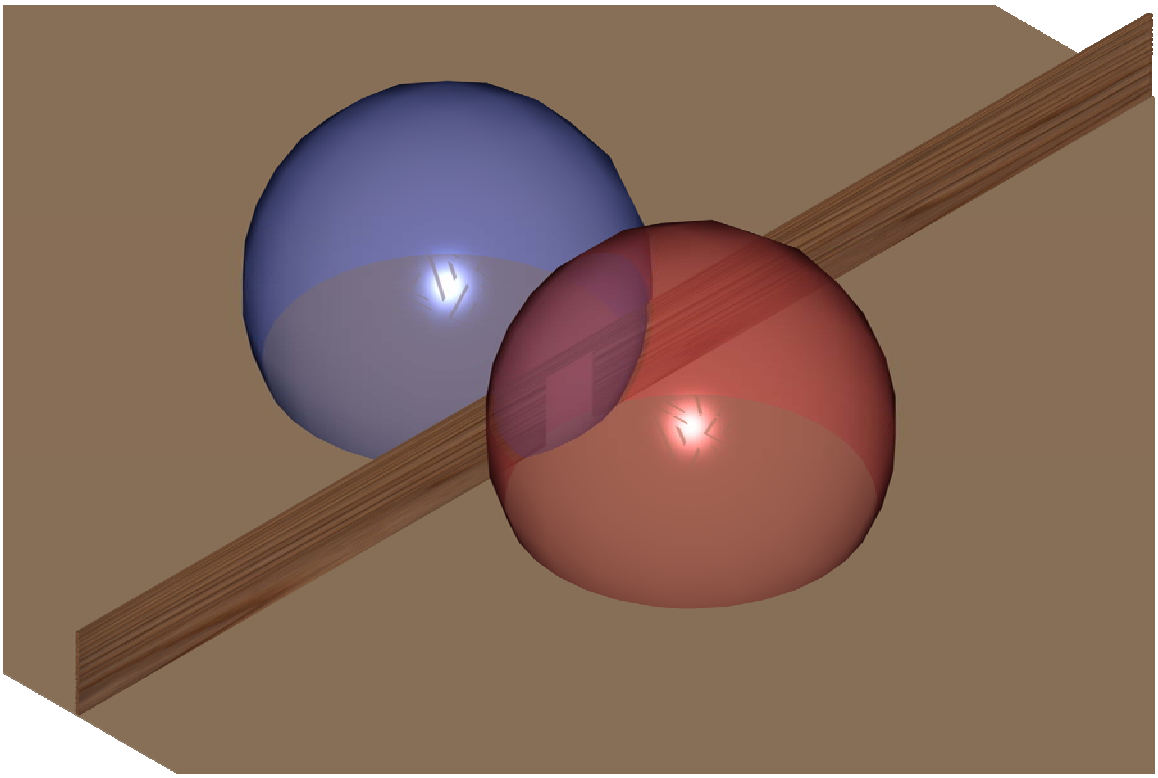


Figure 4.2 - Communication Zones (Perspective)

These renderings were created using the actual power ratios from the datasheet and are all drawn to scale. As can be seen by figure 4.2, the zones are quite distinctive and very large compared to the double wide door that is in the wall. Figure 4.3 clearly shows that while the two antennas do create two distinct zones, they still allow for communication between the two parts of the device.

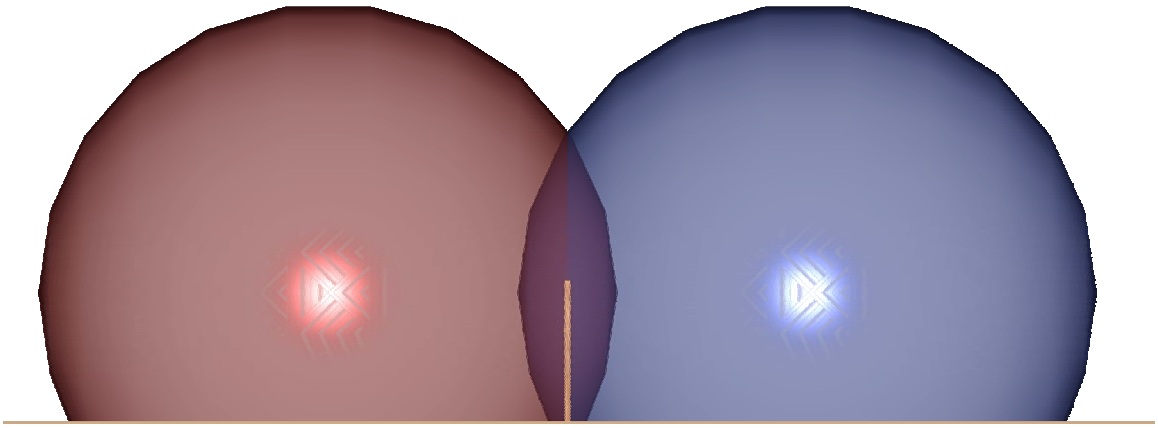


Figure 4.3 - Communication Zones (Side)

Zone Dimensions

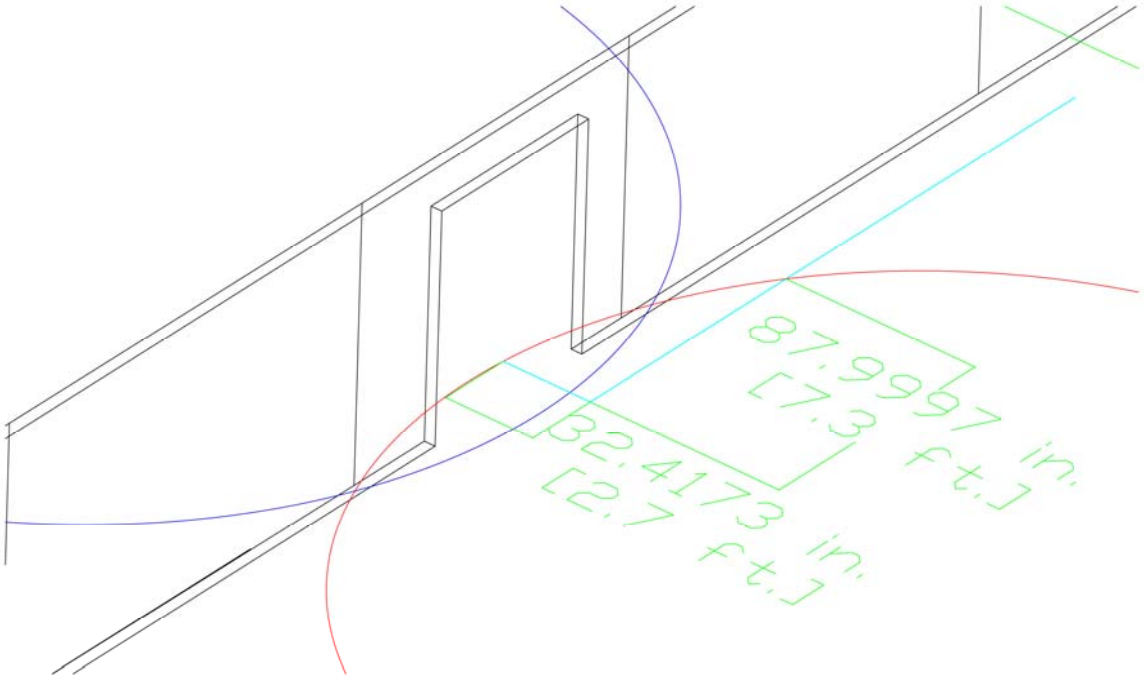


Figure 4.4 - Zone Dimensions

As mentioned earlier, the primary concern for this particular implementation is that the user remains in the communication zone long enough for the entire transmission to take place. Based on the research by Knoblauch, Pietrucha, and Nitzburg the average walking speed is 5 feet/second. This number will be used for the calculations in this section. The communication zones were created using the power ratio taken from the antenna datasheet, and then applying the ratios so that the maximum point from the antenna is 33 feet, based on the Bluetooth specification. This generates two zones, with a total distance of 56.9 feet. At an average walking speed of 5 ft/sec, this is sufficient time to locate the device, establish a connection, and silence the phone. In order to test the worse case, it was assumed that a person would not walk closer than 18 inches to a wall

perpendicular to the door that they are walking through. This possible walking path can be seen by the light blue lines in the drawings. This provides a total of 10 feet when the user is in the communication zone. This still provides 2 seconds of communication time with the device at the average walking speed. With the average time to establish a connection at 1.4 seconds, this still leave adequate time to silence or activate the device while the user remains in the zone.

Software Algorithm

Buffer Entry Structure

Each of the devices mounted above the door are responsible for detecting the Bluetooth phones that enter their zone. Each device will manage a buffer of the phones that are currently in its zone, or that have been in their zone in the recent past. The two parts will be in communication with each other as to who is in their zone. Each buffer entry will contain all of the following information about each device in its zone.

ID	Bluetooth Address	Active	Timeout	Status
8 bits	48 bits	1 bit	8 bits	8 bits

Figure 4.5 - Buffer Entry Structure

The Bluetooth address is the unique identifier for each phone in the zone. The Bluetooth address is a 48-bit identifier that is assigned to both the device and phones by the manufacturer, similar to a MAC address. The system uses the unique Bluetooth address to maintain a list of phones in the particular zone.

The active field is used to store whether the device is currently in the zone or if it has been in the zone in the recent past. The algorithm must be able to maintain a list of those phones that are currently in the zone, or have recently been in the zone, and it must be able to distinguish between the two. The active field is stored as a Boolean value.

The timeout field is used to determine how long a phone remains in the buffer after it has gone inactive. When a new phone is discovered, its timeout value is set to the maximum timeout value and remains at the maximum value until the device goes inactive. When the phone is no longer active, the timeout value decrements every time. The inquiry scan is performed until its value reaches zero, at which point the device is removed from the buffer.

The status field is used to store the current status of the phone, as far as the system is concerned. When a new phone is discovered and it is added to the buffer, its status value is set to 'U' for unknown, because while the phone has a current state, the system is unaware of what that state is. As the system changes the status of a phone and receives a confirmation, it stores the updated status so as to not send repeat messages to the phone to conserve power.

Software Flowchart

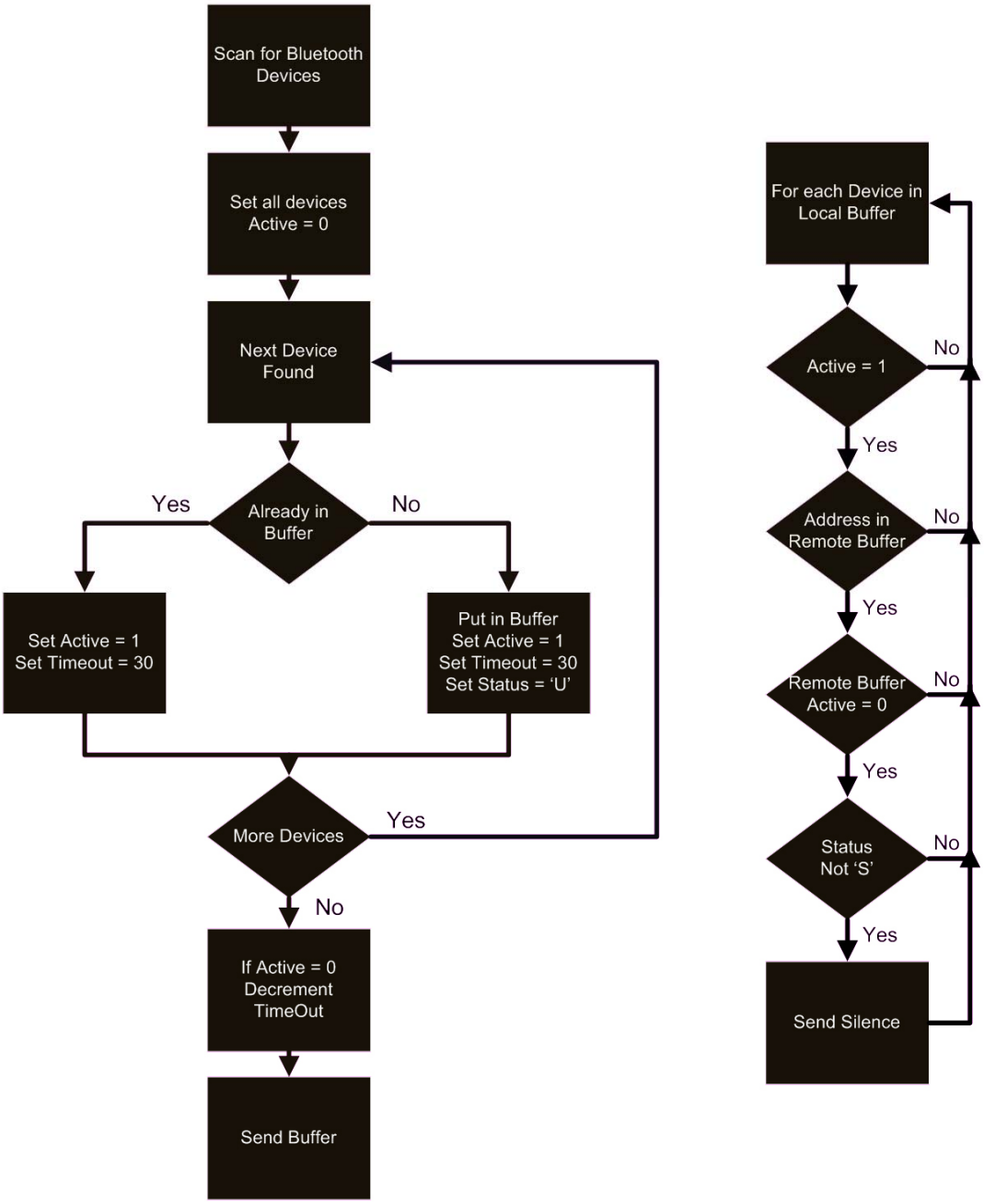


Figure 4.6 - Software Flowchart for Inside Device

The software flowchart shows the device algorithm for the inside mounted part of the device, as can be seen by the silence operation near the bottom of the flowchart. The

outside zone would be very similar to the one shown, with minor modifications. The software performs an infinite loop to scan for phones and determine if there are any phones to be silence. The program begins by looking for any Bluetooth phones that are currently in the zone. The inquiry itself does not distinguish what phones have been in the zone before and what devices are new, it simply reports a list of all the Bluetooth address that are currently in the zone. Then the stored buffer of entries changes so that all devices are inactive. The program then looks at each Bluetooth phone found to see if it already exists in the buffer. If it already exists in the buffer, the program sets the active field back to 1. It also sets the timeout back to the maximum value because the phone may have not been in the zone for a while, but it has reappeared so the timeout value needs to be reset. If the address is not found in the buffer, this means the phone is new to the zone, and a new entry needs to be added to the buffer. The address is added to the next available slot in the buffer, the active field is set to 1, the timeout is set to the maximum value, and the status is set to 'U', because the phone has just appeared so its status cannot be known. The program then looks at the buffer after all phones have been updated, and any phone that is not active, the timeout field is decremented.

At this point, the local buffer has been updated and contains an accurate list of all those Bluetooth phones that are currently in the zone or have been in the zone in the last 30 inquiry scans. This device that is inside then sends its complete buffer to the outside device. Each device is constantly storing a buffer of both the phones that are in its zone, or that have been in its zone, and also a list of the phones that either are or have been in the other zone as well. Once the devices have swapped buffers it is time to look and see

if there are any phones to be silence or activated. In order to accomplish this, the device looks at each phone in its local buffer. There is no reason to scan through the remote buffer, because of the two separate zones, the device is only capable of sending messages to the phone that is in its zone and those addresses are stored in the local buffer. For each address in the local buffer, the program looks through the addresses in the remote buffer. If a match is found, the program looks to see if that phone is inactive. If the phone is still active in the local buffer of both parts of the device, this means that it is in the small window directly under the device and thus nothing can be determined. If the phone in the remote buffer is inactive, that means that this particular phone used to be in the outside zone, but now it is in the inside zone. Lastly it checks the status of the phone in the local buffer. If the status of the phone is 'S' that means that it was already silenced in a previous iteration and the device received a confirmation so there is no need to send the message again. So in summary, if the phone is present in the local buffer, is present in the remote buffer, is inactive in the remote buffer, and has any value other than 'S' as a status, the device sends a silence command to the phone and waits for a response. If the phone responds that it has been silenced, the status field is updated in both the local and remote buffers, because there is actually only one physical phone, and while it may only exist in one of the zones, its status must be the same regardless.

An Example

The example below demonstrates a typical situation with two cell phones. Both the local and the remote buffers for each part of the device are shown. Any changes to the buffer since the last iteration are shown in red. In this situation, cell phone one enters

the outside zone and is detected by the outside device. That device then sends its information to the inside device, which stores it in its remote buffer. Cell phone 2 enters the outside zone and stays there for the remainder of this example. Cell phone 1 then moves to the center region where it is in both zones local buffer, and no change occurs. As soon as it moves into the inside zone, the outside zones local buffer status changes to a zero and when the inside remote buffer is updated, cell phone 1 is silenced, and the status field is updated.

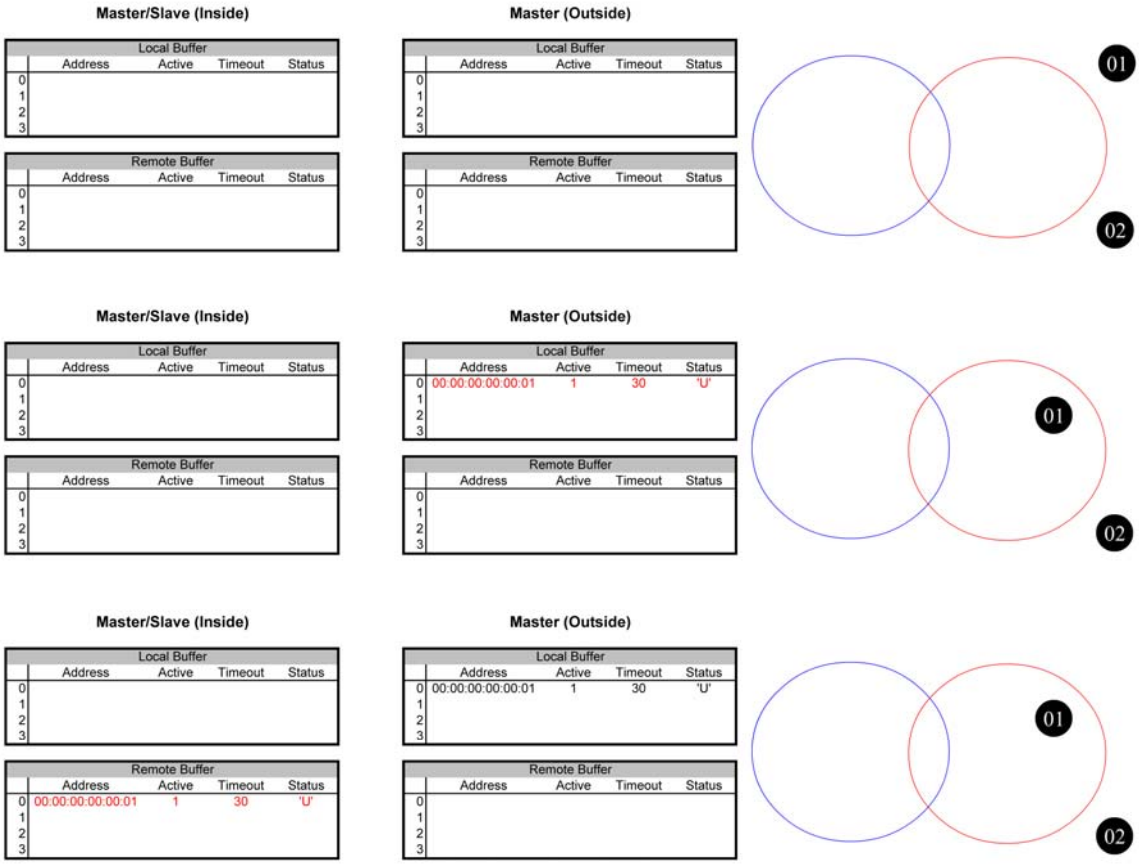


Figure 4.7 – An Example

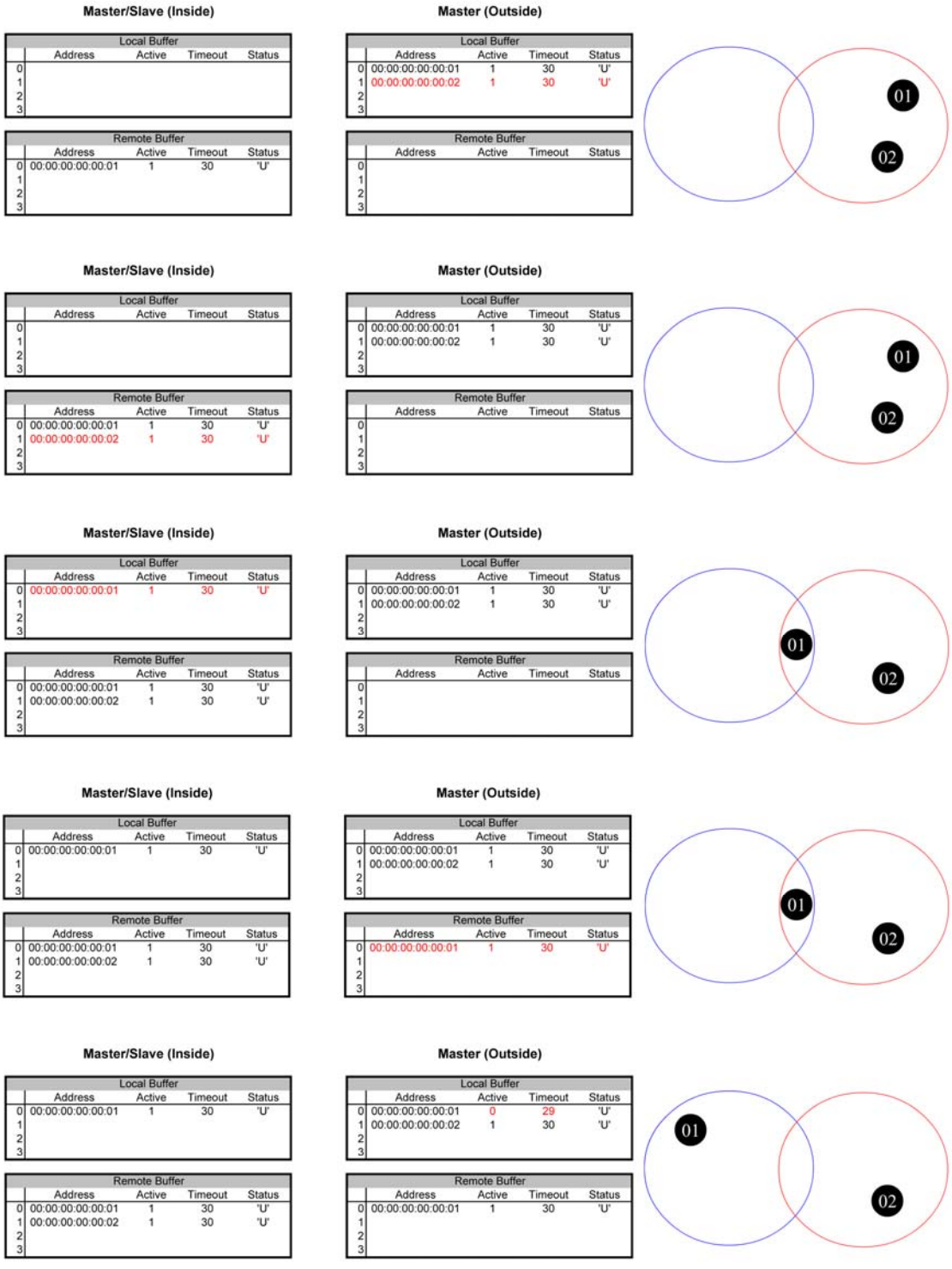


Figure 4.7 – An Example

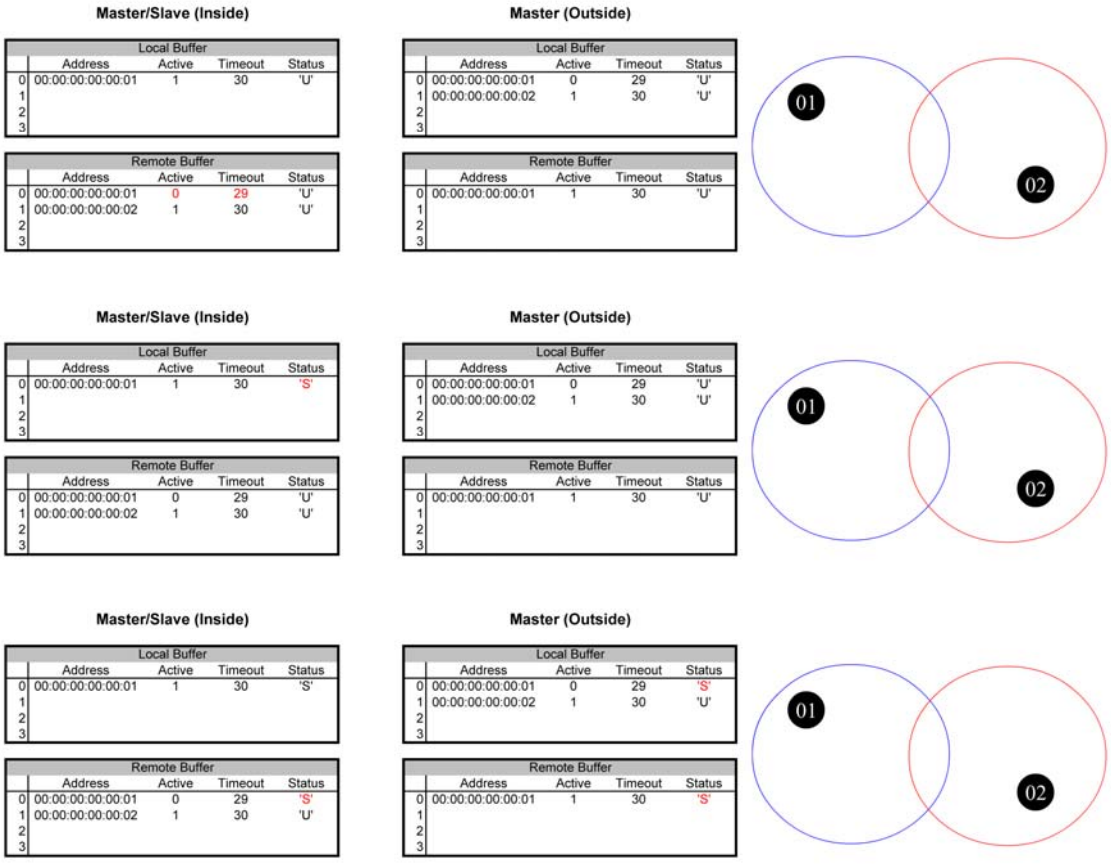


Figure 4.7 - An Example

Possible Walk Patterns

The above example demonstrates how the device operates successfully if the user walks directly through the two zones as intended, but what happens if the user does not walk through as intended? The algorithm must be able to handle this sort of event, as it is very possible for a device to move through the zones, and it would not be good if it accidentally silenced a phone without the user expecting it. These devices will probably be mounted so that at least the outside zone is in some sort of lobby or waiting area. These are usually very crowded places, and so it would be very possible for a person to

just walk across the outside zone, and never enter the room. The same is possible for someone inside the room. In either case, the phone will be connected to the piconet, but as long as it only shows up in one zone's local buffer, then no message will ever be sent to change the mode. Even if the phone were to enter the region at the door, where both zones overlap, the phone would still not receive a message until it is all the way inside or outside the room.

If the user were to walk into the room and right back out quickly, long enough for the local buffer to detect the presence of the phone, and the remote buffer to change to inactive, but before the message to silence the phone could be sent, the phone would not be effected either.

Final Thoughts

Why Bluetooth?

The Bluetooth communication protocol provides some great advantages to this project verses some other communication methods. First of all, Bluetooth is readily available on almost all new phones, which would mean that it would be easy for phone manufacturers to adopt this new idea. In the year 2001 alone, over 200 million Bluetooth equipped devices were shipped from manufacturers. On the actual phone, it is fairly easy to implement an algorithm to silence a phone when you receive a Bluetooth message to silence and to activate when you receive a message to do so. The processing all happens in the device above the door. Radio Frequency Identification (RFID) would be a possible solution to silence phones upon entering a room. While RFID devices are simple and

very small, the goal of cell phone technology is to make them as small as possible and cell phone manufacturers would be reluctant to add any more hardware to their device.

Why communicate between the two separate zones?

It would be possible to accomplish this problem by simply putting a Bluetooth zone outside the door that constantly broadcasts messages to activate all devices that are within its zone and for the inside zone to constantly broadcast messages to silence the devices. If this implementation were chosen, the phone would have to be constantly listening for and accepting messages from the device. Especially since the outside device will probably generate a zone that lies in a lobby of some sort, it would be possible for a phone to reside in that particular zone for a long period of time. If the device were constantly broadcasting, this would cause a significant power drain on the phone. If the two devices are in communication with each other, and only talk to the phone when necessary to either silence or activate the phone, this places most of the power drain on the device that has a power supply and alleviates the drain on the battery operated phone. In this case, the phone can be discovered and synchronized and then placed in a parked state until a transmission is required.

CHAPTER 5

WINSOCK IMPLEMENTATION

A Bluetooth development is very expensive and not readily available, so in order to test this algorithm, two laptops were used with Bluetooth adaptors running Microsoft Visual Studio .NET along with the Platform Development Kit which contains the Bluetooth support for .NET. The algorithm described above was implemented with some exceptions.

Winsock Limitations

The primary limitation of Winsock is the lack of ability to debug Winsock functions. Winsock functions execute out of a Dynamic Link Library, and Microsoft does not provide debug version of the functions. As a result, any time the program calls a Winsock function, the programmer looses the ability to step or to set breakpoints within the function. The function is called and then returns, and what happens in between is not known to the programmer.

This limit caused a particular problem in this implementation of the program. The Winsock function to initiate a Bluetooth device inquiry returns after four seconds. The function allows the programmer to specify a timeout value as a multiple of 1.28 seconds. Even when this timeout value is set to 1.28 seconds, the function still required four seconds to return. As mentioned above, there is not a method to debug this problem, as it all lies within a Winsock function. This would require that the phone remain in the zone much longer for this particular implementation to operate correctly.

The next major limitation for this particular project lies in the Winsock implementation of Bluetooth. Winsock uses RFCOMM to create a virtual serial port and then uses all of the prepackaged functions to send data over this connection. This would work fine if the only goal of the project were to send data over a Bluetooth link, but with the complexity of this particular project, access to at least the L2CAP layer is required and even lower in the protocol stack would be desired.

The last major limitation is similar in that the Winsock implementation of Bluetooth does not allow for any programming below the RFCOMM layer. This is a major disadvantage as it does not allow the programmer any control over the Bluetooth device itself. Bluetooth supports so many different power states and other advanced mechanisms for controlling a Bluetooth radio, and the Winsock implementation does not allow for any programming of these parameters or even a method to view the current status of the radio.

Accomplishments

Using Bluetooth and Winsock, a successful connection was made between a standard, out-of-the-box cell phone. This demonstrates that it is possible to create a connection between a computer device and a Bluetooth enabled phone. An example of this connection can be seen in figure 5.1.

Figure 5.1 - Laptop to Cell Phone Connection

In the image on the left, the device name Z520a is the cell phone. No configuration changes were made to the cell phone. It was removed from its manufacturers packaging and the Bluetooth function was enabled. The Winsock software located the phone, discovered its address, and attempted a connection. The phone cane be seen asking to accept the connection, and once the connection has been accepted, it asks if the user would like to add the laptop to its list of preferred devices. As a side note, it can be seen that the device inquiry lasted just under four seconds, as described above.

The results of the algorithm can be seen in figures 5.2 and 5.3. The program displays the contents of both its local and remote buffer at the end of every inquiry scan. The last three updates can be seen in each picture. Any action that was taken by the device to either silence or activate a phone can be seen under the actions section. In both of these examples the device address 95034614882 is the cell phone.

In figure 5.2, the outside device activates a phone after the middle update in the screenshot. The cell phone is active in the local buffer, and is present in the remote buffer, but is inactive. The status in the local buffer is 'U,' meaning all of the requirements to activate a device have been met. The action to activate the cell phone is taken by the device.

In figure 5.3, the same occurs except on the inside device so the silence command is sent to the phone.

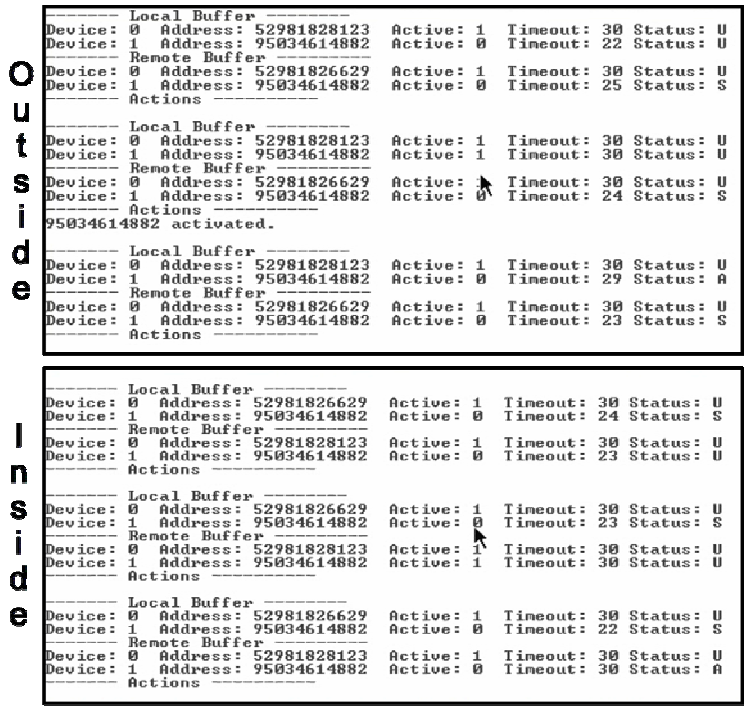


Figure 5.2 – Activate Example

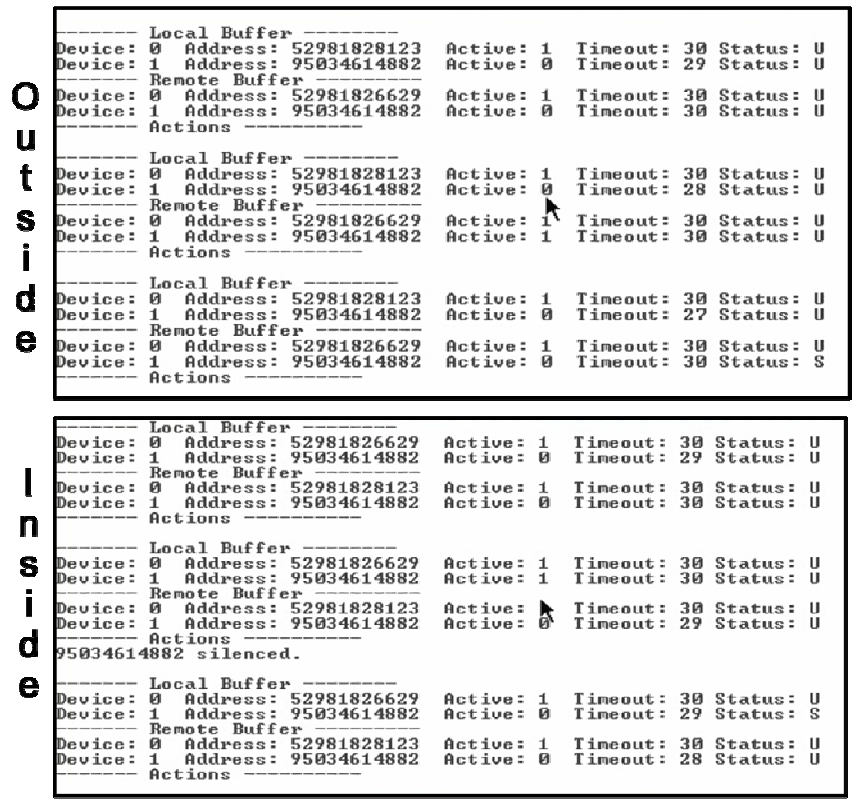


Figure 5.3 – Silence Example

Conclusion

While the Q-Node provides a great method to silence phones on a small scale, a large scale implementation just simply is not practical. A device like the one described above is a much more cost effective solution to the problem of automatically silencing cell phones. In addition to the cost benefit, installation into a pre-existing building is far more practical than an array of Q-Nodes. The physical description and software algorithm are in line with the Bluetooth specification and in first round tests are working properly. The device described in this paper is a viable solution to the problem of automatically silencing cell phones and deserves further investigation.

CHAPTER 6

FUTURE WORK

While the algorithm has been proven viable by the Winsock implementation, a much more comprehensive test environment is needed in order to perform any more advanced testing of the device. A Bluetooth development kit would be required to continuing testing and improving the device described above. An example of such a development kit is the Modular Bluetooth Minimodule Development Kit by Smart.

Figure 6.1 - Smart Modular Bluetooth Minimodule Development Kit[9]

While the price for this particular development kit was not listed on the website, most Bluetooth development kits range from $2,500 - $10,000. This kit offers the ability to program at either the HCI level or even at the radio level. It also claims full piconet and scatternet support which would be essential for this particular project. Lastly, the

external antenna connector would allow the connecting of a hemispherical antenna in order to create the two independent zones, also making testing more realistic.

REFERENCES

1. Bluetooth Specification Core v2.0 + EDR http://www.bluetooth.com

2. Ching Law. Amar K. Mehta, Kai-Yeung Siu, *Performance of. a New Bluetooth Scatternet Formation Protocol* Massachusetts Institute of Technology

3. Richard L. Knoblauch, Martin T. Pietrucha, and Marsha Nitzburg, "Field Studies of Pedestrian Walking Speed and Start-Up Time."

4. Wireless Quick Facts for October 2005; http://www.ctia.org

5. Bluelinx Inc.; http://www.bluelinx.com

6. Apple Developer Connection; http://developer.apple.com/documentation/DeviceDrivers/Conceptual/Bluetooth/art/bt_protocol_stack.gif

7. Bluetooth Protocol; www.baracoda.com/shared_docs/bluetooth_protocol.pdf

8. Tyco Electronics 1513151-1 Datasheet; http://catalog.tycoelectronics.com/TE/Presentations/100930r.pdf

9. Smart Modular Bluetooth MiniModule Development Kit; http://www.zbausa.com/bluetooth_adaptor_board.asp

tyco Electronics

Technical Data Sheet

Bluetooth 802.11b Antenna

Tyco Electronics P/N: 1513151-1

Features

- Wide bandwidth and high gain in a compact size
- Enhanced hemispherical pattern improves RF link reliability of portable devices
- Minimum matching circuits required

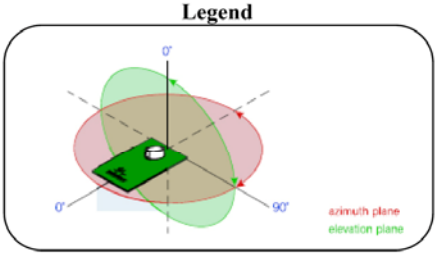
Electrical

Frequency Range	2400–2500 MHz
Peak Gain	> 4 dBi
VSWR	less than 2.5:1
Polarization	linear
Power Handling	10 Watt cw
Feed Point Impedance	50 Ohms unbalanced

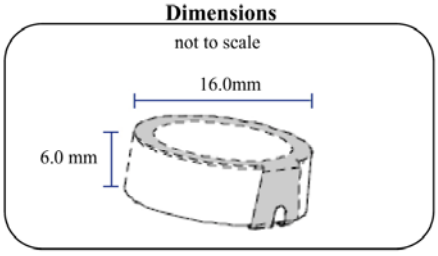
Mechanical

Size	16.0 mm x 6.0 mm
Weight	Less than 1 g

Legend

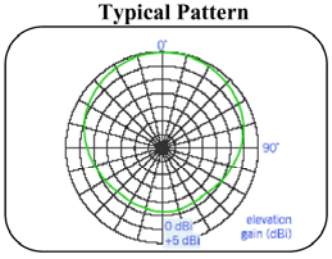


Dimensions

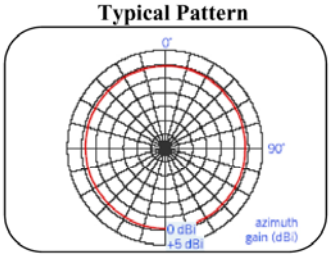


2400 MHz Band

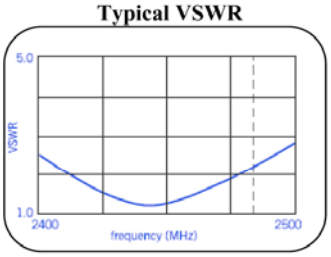
Typical Pattern



Typical Pattern



Typical VSWR



© 2004 by Tyco Electronics Corporation. Tyco RangeStar and RangeStar Wireless are trademarks.

PERMISSION TO COPY

In presenting this thesis in partial fulfillment of the requirements for a master's degree at Texas Tech University or Texas Tech University Health Sciences Center, I agree that the Library and my major department shall make it freely available for research purposes. Permission to copy this thesis for scholarly purposes may be granted by the Director of the Library or my major professor. It is understood that any copying or publication of this thesis for financial gain shall not be allowed without my further written permission and that any user may be liable for copyright infringement.

Agree (Permission is granted.)

_____Chad D Larsh_____ ____05/01/06_____
Student Signature Date

Disagree (Permission is not granted.)

_____ _____
Student Signature Date