

Valparaiso University

Radio Frequency Identification

RFID Independent Research CS492 & ME499

Stephen Dolph, Dan Roggendorf, Kirsten Swanson, & Adam Watkins

05/05/09

I have neither given or received nor have I tolerated others' use of unauthorized aid.

TABLE OF CONTENTS

BACKGROUND

PASSIVE RFID TAGS	3
ACTIVE RFID TAGS	3

RESEARCH

CHOOSING A TAG SYSTEM	4
EXPERIMENTS	5

PROGRAMMING

CURRENT PROGRAM	7
ADVANCED PROGRAM	8
FINAL PROGRAM	9

Background

Passive RFID Tags

The radio frequency identification (RFID) tag is considered to be the most sophisticated piece of equipment within an RFID system, even though it is often the least expensive component. There are two major types of RFID tags, passive and active, each with their own strengths and weaknesses. The majority of the tags utilized in today's marketplace are passive tags. Passive tags have a distinct advantage over active tags as they are able to transmit information without the presence of an external power source, thus giving them an infinite lifetime (Glover 5). Passive tags can also vary in size from several inches across to smaller than a grain of rice. However, passive tags have very short read ranges, typically ranging from several inches to just over two feet. Passive tags are also extremely limited as to the amount of information that can be stored on them. The majority of passive tags can hold between 32 and 128 bits ("What is RFID?"). Although information capacity does place a limitation on the applications of passive identification tags, this capacity exceeds what is needed in many applications, including the tracking of packages between the supplier and store shelves.

Active RFID Tags

An active tag is often utilized in environments where a passive tag would be ineffective. They are often utilized in situations where the range between the tag and the antenna is of the utmost importance. Active tags have a much longer read range and in ideal conditions, can be read as far as 300 feet from the antenna (Huber 173). Active tags differ from passive tags as they require an external power source, usually obtained from an external battery. The electrical components within the active tags prove to be beneficial and yet detrimental to the success of the

active tag industry. Active tags are able to store greater amounts of information than their passive counterparts, as much as one megabyte. The introduction of power to the tag also forces the tags to be larger in size than passive tags. The power requirement introduces an expiration date for the tag as the battery has an operational life of roughly ten years (Hunt 7).

Research

Choosing a Tag System

Initial research of different passive and active RFID tag systems revealed several options. We first explored the options provided by Texas Instruments. Through Texas Instruments' free product sampling program, we were able to obtain passive Tag-IT RFID tags. However, the free samples did not provide enough equipment to analyze or program the tags. We explored the option of purchasing additional equipment in order to continue to use the free samples from TI. The purchase price for the interrogator, antenna, and software for the Tag-IT series was \$4000 and no educational discount was offered. In addition, Eddie LaCost, an engineer from TI's RFID research team, informed us TI only produced passive tag systems. This indicated we would have to purchase two separate systems, one passive and one active, in order to conduct our research. Since this option was not fiscally feasible, we decided to explore other options.

An evaluation kit from Dynasys was also considered for purchase. This kit contained active tags and an active tag reader. Dynasys' website quoted the kit for \$995. When Dynasys was contacted, they were willing to discount the kit by \$15, but they did offer an educational discount. This system was not chosen because it did not provide many opportunities for expansion of the system as additional software would have to be purchased in order to utilize all

functions of the tags.

The final company considered was Identec. Identec offers several systems which read both active and passive tags. The system we considered had a list price of \$2500, however with the educational discount the system was purchased for \$1500. The system included an i-CARD 3, 2 antennas, 6 active tags, and demo software. The evaluation system also allowed for the option to upgrade or change the types of tags used at a later date as all Identec tags are compatible with this reader. The package offered by Identec offered the greatest read range out of current RFID manufacturer and offered the greatest flexibility in the design and alteration of our final system.

Experiments

Shortly after receiving the shipment from Identec, short range testing on the tags began. The ILR i-Q8 and ILR i-Q32T tags were extensively tested as they have a longer read range than the other tags included in the kit. Initial indoor testing with the Identec antenna reported a range of 100 feet. This made isolating the tags difficult as only one tag could remain in the room at any given time. Once the tags were isolated, the call numbers for each tag were verified. This also allowed for an assignment of a numerical value to each of the tags in Dan Roggendorf's program.

Testing continued with outdoor testing at Eastgate field in an attempt to develop an antenna profile. It is important to develop the antenna profile due to the possible directionality of antennas. If the antenna were to be mounted at an undesirable angle, it would be possible for a vehicle to remain within the tag's dead zone. As shown in Figure 1, testing located several weak signal coordinates; however, better equipment is required to develop a full antenna profile. This experiment was also beneficial as it provided an opportunity to experiment with the true

range of the tags. When outside, the range of the tags dropped significantly, as the maximum distance attainable between the tag and the antenna shrinking to just over forty feet. Indoors, the high frequencies bounced off of the brick walls, providing excellent read ranges. The outdoor testing brought up new concerns about whether a new antenna would have to be purchased before being placed in an automobile as the signal strength from the antenna we currently have will not be strong enough for long distance reading of the signal. Any obstruction of the signal would also cause the tag to become unreadable.

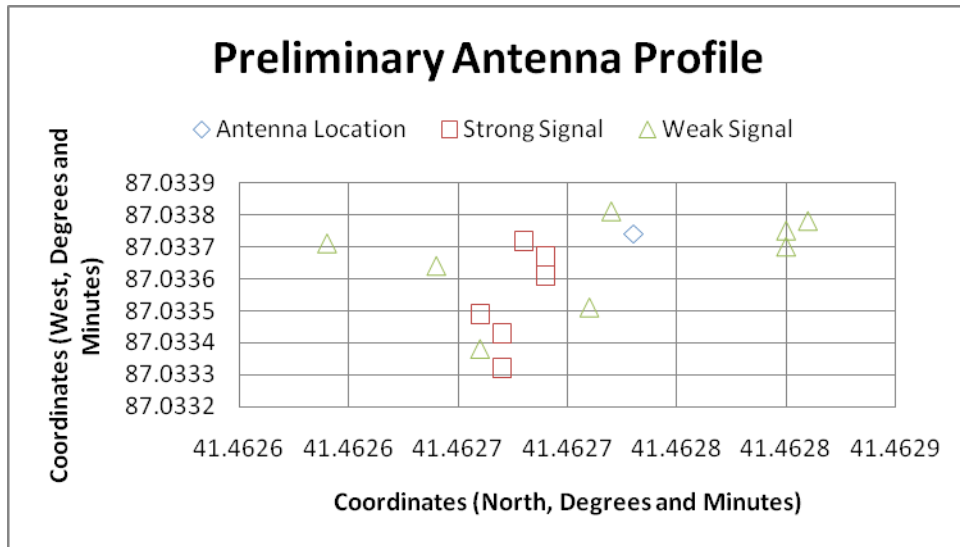


Figure 1: Antenna Profile developed at Eastgate Field, Valparaiso, Indiana

The final experiment involved the speed at which the program could identify and display the correct speed limit. Initially, the tag was walked, at approximately five miles per hour, past the antenna to determine the distance at which the tags could be read. Bicycles were then utilized to increase the velocity to roughly ten miles per hour. This testing proved the program could read and properly identify the tags from a 40 foot range. In this experiment, the position of the antenna was also adjusted: stationary antenna placed on the ground, rotating the antenna following the rider, and stationary antenna held approximately four feet above the ground. The

antenna was least efficient on the ground as the antenna was unable to read the tags from more than a 20 foot read range. Also, the dropout zones for the antenna seemed to be accentuated when the antenna was left on the ground.

Programming

CURRENT PROGRAM

During the planning stages of this project, we determined that a program was needed that could take the signals from an antenna and determine which tag, among many, was the closest to the antenna. This program would use the antenna power to determine which tag was using the least power. By finding the tag that was using the least power it was reasoned that it must be the one which was closest to the antenna. This was found to be a reasonable assumption. Later, it was also determined that by associating tag numbers with speed limits, the program could report the proper speed limit.

While this program was never intended to be developed into a final program, many flaws were found with it. In order for the program to run properly, it needs the Identec software to run in parallel with it. This was known before the program was written and was considered at the time to be a good thing because it would save programming time. The drawback was that when both programs ran together, they would command the majority of the system resources, making the system slow down significantly. This also prevented the program from processing tags in real time. Results would take several seconds to get at times. A possible fix to this problem was discussed but was never implemented. Another drawback for this program was that it determined which tag was closest to the antenna based on the signal power. This seemed reasonable at the time, but during testing it was found that there were more factors involved with

the signal power than just distance. This method worked most of the time but it was realized that a better solution needed to be found.

ADVANCED PROGRAM

After seeing the results of the first program, it became apparent that a new program was needed that would expand upon our current program. This program would implement a more accurate way for determining which tag was the closest to the antenna. An external display would also be used to more visually convey the program's results. Also, this program would need to address the shortcomings of the previous program, such as overusing computer resources.

Rather than starting from scratch, the advanced program would use the current program as a base to build from. One of the first changes that would need to be addressed is the use of system resources. One way that was discussed to solve this issue was to use wait statements to free up the system processor. This should solve the problem but would also introduce another. By inserting wait statements we would effectively be decreasing the polling rate. For example, instead of the program checking which tag was closest every second, it would be set to check every six seconds. The impact this change would have on our research was discussed and we concluded that the change was more beneficial than the issues it brought with it.

Another improvement that needs to be made from the current program is to find a better variable to use to find which tag to use. One of the better ideas that was discussed was to use cardinal directions as a variable. The advantage of this was that when testing certain scenarios, such as the one in figure two, the program would better be able to choose the correct tag. In this case, the tag for the North South road will be picked up as the tag nearest to the antenna, which is in the car. This is not the tag that belongs with the road the car is on. With the new method, each

tag will be programmed with a speed limit and a cardinal direction. This way, when the vehicle approaches the North South tag, it will be able to disregard it in favor of the East West tag.



Figure 2: Graphic illustrating the issues of using tag distance to determine which tag to use and why adding cardinal directions to the tags will solve this issue.

There were several setbacks that delayed the writing of the advanced program.

Programming the tags was by far the biggest setback. In order to program these tags, the computer needed to be running Windows CE. There were several attempts made to get a version of Windows CE installed on a laptop, but to no avail.

FINAL PROGRAM

When the research project first began, we brainstormed what we wanted our final product to look like. This included what would it do, what would it not do and how would it do it. Our vision for the final product was that any software needed would be independent from a computer. This was mainly for cost reasons, since it was decided that it would not be practical to have our product depend on a computer. To take the computer's place, a series of filters would support a microcontroller based system. This was an ambitious decision that we knew could easily be turned into a senior project. With this in mind, we started to make a list of what it would take to

make this project work.

Unfortunately, this idea did not get off the ground. It was found early on that we were missing two critical pieces of information, the tag's exact frequency and the protocol the tags use to communicate. If the frequency could be obtained, the protocol could be experimentally determined. Unfortunately, Valparaiso University does not own equipment powerful enough to measure the frequency of the tag. We are currently looking into local companies that would have either a powerful oscilloscope or a frequency analyzer. The future success of this project will depend on whether or not we can get the frequency and the protocol.

Works Cited

Glover, Bill, and Bhatt Himanshu. RFID Essentials. Sebastopol: O'Reilly, 2006.

Huber, Alexander, and Josef Huber. UMTS and Mobile Computing. Norwood: Artech House, Inc., 2002.

Hunt, Daniel, Albert Puglia, and Mike Puglia. RFID: a Guide to Radio Frequency Identification. Hoboken: John Wiley & Sons Inc., 2007.

"What is RFID?" Association for Automatic Identification and Mobility. 28 Jan. 2008

<http://www.aimglobal.org/technologies/RFID/what_is_rfid.asp>.

```

#include <iostream>
#include <fstream>
#include <string>
#include <windows.h>
#include <time.h>

using namespace std;

string doStuff(string tag, bool debug);
void wait(double seconds);

int main(){
    int i;
    bool debug = true;
    string tag = "NONE";
    cout << "RFID Quick and Dirty" << endl;
    cout << "Dan Roggendorf" << endl;
    cout << "October 27 2008" << endl;
    cout << endl << endl;
    cout << "DIRECTIONS" << endl;
    cout << "Please open Identec Solutions ILR Tag Diagnostics Utility" << endl;
    cout << "and press START before the countdown runs out." << endl;
    cout << endl << endl;
    cout << "Starting countdown";
    system("pause");
    for (i = 500; i>0; i--){
        wait(0.25);
        cout << ".";
        wait(0.25);
        cout << ".";
        wait(0.25);
        cout << ".";
        wait(0.25);
        cout << i;
    }//for i
    cout << endl << endl;
    while (1){
        tag = doStuff(tag, debug);
    }//while
    cout << endl << endl;
    system("pause");
    return 0;
}

string doStuff(string tag, bool debug){
    string tagArray[4]={"NONE","NONE","NONE","NONE"};

```

```

string line,temp;
    int antArray[4]={-99,-99,-99,-99};
    int i,j,tempA,flag;
ifstream ifs("tags.txt");
    i = 0;
    j = 0;
    //system("pause");
    while(getline(ifs,line)){
        if(i==3 || i==12 || i==21 || i==30){
            tagArray[j].assign(line,15,13);
        }//if

        else if(i==9 || i==18 || i==27 || i==36){
            temp.assign(line,16,3);
            antArray[j]=atoi(temp.c_str());
            j++;
        }//else if
        i++;
    }//while
flag = 1;
for(i=1;(i<=4)&&flag;i++){
    flag=0;
    for(j=0;j<3;j++){
        if(antArray[j+1] > antArray[j]){
            tempA = antArray[j];
            temp = tagArray[j];
            antArray[j] = antArray[j+1];
            tagArray[j] = tagArray[j+1];
            antArray[j+1] = tempA;
            tagArray[j+1] = temp;
            flag=1;
        }//if
    }//for j
}//for i
if(debug){
    cout << tagArray[0] << " " << antArray[0] << endl;
}//if
if(tagArray[0]=="NONE"){
    cout << endl << "No tags in range" << endl;
}//if
else if(tagArray[0]=="0.200.168.133" && tagArray[0]!=tag){
    cout << endl << "The speed limit is 25 MPH" << endl;
}//else if
else if(tagArray[0]=="0.200.168.134" && tagArray[0]!=tag){
    cout << endl << "The speed limit is 35 MPH" << endl;
}//else if

```

```

else if(tagArray[0]=="0.200.170.134" && tagArray[0]!=tag){
    cout << endl << "The speed limit is 55 MPH" << endl;
} //else if
else if(tagArray[0]=="0.200.170.135" && tagArray[0]!=tag){
    cout << endl << "The speed limit is 65 MPH" << endl;
} //else if
else{
    //Code that will execute if no NEW tags are found (or any tags)
} //else
tag = tagArray[0];
keybd_event(123,0,0,0);
wait(0.01);
keybd_event(123,0,2,0);
wait(0.01);
keybd_event(13,0,0,0);
wait(0.01);
keybd_event(13,0,2,0);
wait(0.01);
keybd_event(13,0,0,0);
wait(0.01);
keybd_event(13,0,2,0);
wait(1.5);
return tag;
} //doStuff

void wait(double seconds){
    clock_t endwait;
    endwait = clock () + seconds * CLOCKS_PER_SEC ;
    while (clock() < endwait) {}
} //wait

```