RFID trace Lab Report

Ruiling Zhang(4501322)
Yuzhu Yan(4468023)

March 2016

1 Introduction

In this lab,the task is to interpret a RFID trace into binary code. This trace file includes bidirectional messages between reader and tag. How to identify and measure the length of "Tari" (Figure 1) is the key of this experiment. In this lab, length of "Tari" should be a threshold. If the horizontal length of a pulse satisfy this threshold, then it could be regarded as a "zero" data. If it is between "1.5Tari" and "2.0Tari", it could be taken as a "1" signal. However the encode method for backscattered is different which we can see in Figure 2. The data depend on whether the amplitude changed within one period. As a result, threshold could also be used to formulate the method of preamble identification. In a nutshell, threshold should be the key benchmark of this experiment. In the end, it is also necessary to "read" the binary code and translate it into commands and corresponding reply. The tools we used is matlab.

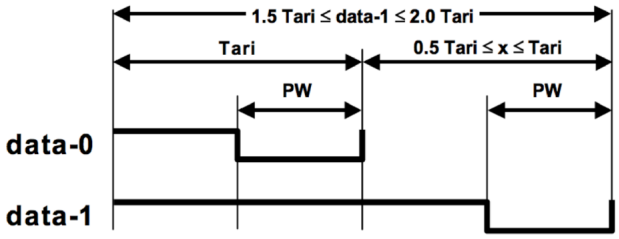

Figure 1: PIE symbols

2 Implementation

We firstly imported trace file into matlab, and received Figure 3.
As we can seen, this graph consists two kinds of pulses, one from reader ("bigger one") and the other from label("smaller one"). In this way, different thresholds should be chosen and this trace file should be analysed into several parts.

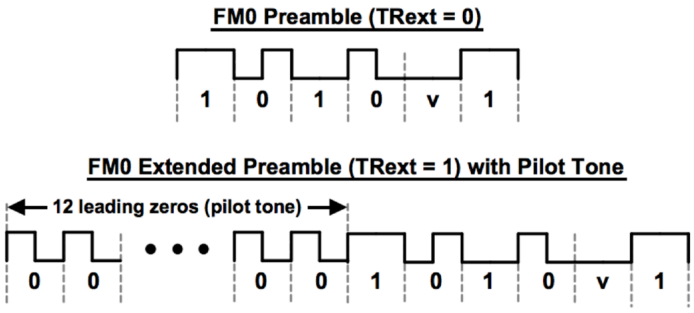

Figure 2: FM0 $T => R$ preamble

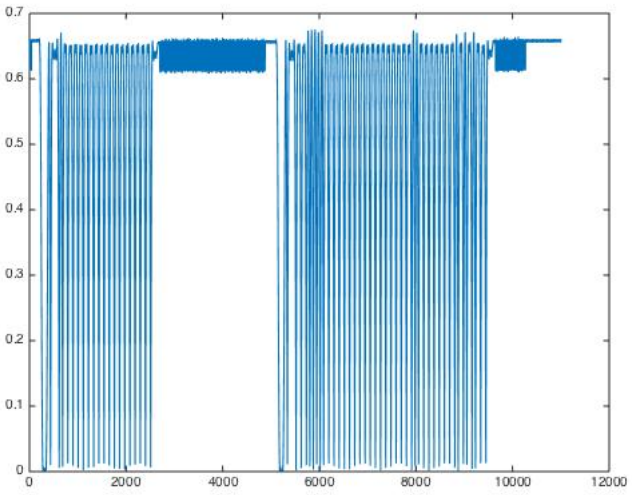

Figure 3: Matlab output of RFID trace

For the command from reader to label, we collected the nodes whose vertical value between 0.3 and 0.35. Threshold based on horizontal value. A threshold for "data-0" is set from 10 to 70. Threshold for "data-1" is set from 70 to 150. With the help of threshold, we differentiate signal into "0" and "1". It can be seen that reply signal from label to read is smaller than command signal. So new threshold should be used. Threshold of "data-0" is set less than 10. Threshold of "data-1" is set from 10 to 20. In order to identify preamble, we set threshold from 20 to 25 for it. As a convenience, a special threshold, which is used for to identify the start of reply and the end of reply is set as more than 50. A matlab program was written to implement above mentioned information. Details of code can be seen in the appendix of code.

After running the code, we derived following array:" 020111111111111111113 1010v10110101101010110111010101011011110110111101011010130211000001111 11111111111111001111111010101 1"

Like what we mentioned before, 0 means "data-0", 1 means "data-1", "v" are parts of preampble ("v" resembles the red blocked pulse in Figure 4), 2 presents longer pulse, 3 means the start and the end of reply. This array will be divided into three parts in order to investigate deep meaning behind it.

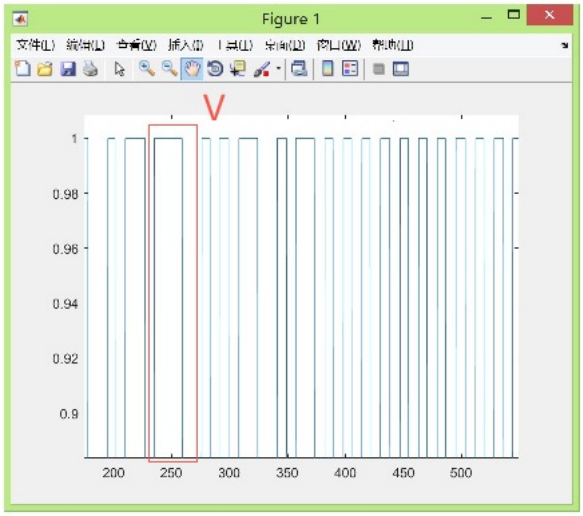

Figure 4: Illustration of "v"

The first part is "02011111111111111111", if we put preamble aside, it can be seen that this command starts with "01", which means it is a ACK command. And the length of this command is 18 bits. According to [1], the details of ACK command is showed in figure 5.

The response from label is "1010v10110101101010110111010101011011110110 11101010110101". It is a 56 bits reply.

	Command	RN
# of bits	2	16
description	01	Echoed RN16 or handle

Figure 5: ACK command

In the end, reader sends a 44 bits message to label again. It is "021100000111 11111111111110011111110101011". The start of this array is 11000001. So we consider it as a Req_RN command. According to following table from book[1], a successful ACK command should between 21 bits to 33328 bits, and what we derived meet this condition. (See Figure 6)

	Reply
# of bits	21 to 33,328
description	See Table 6.17

Figure 6: Tag Reply to a successful ACK command

According to the three times communications we derived, we can see it is a part of the following showed communication between reader and tag, see figure 7.

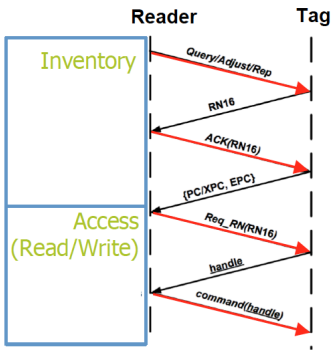

Figure 7: Communication Between Reader and Tag

3 Conclusion

In this lab, we analysed a RFID trace file with help of a matlab program. Three times communication between reader and tag are identified as ACK(RN16), reply,Req_RN.

References

[1] EPC^{TM} Radio-Frequency Identity Protocols Generation-2 UHF RFID

[2] Shengli Wang, Shan Qiao, Shaoyuan Zheng. Simulation Study for the Decoding of UHF RFID Signals, Vol 3, 2007

4 Appendix

```
data = textread('F:\slide\Q3\wireless networking\signal');
figure(1), plot(data);
% Signal encoding part
[m n] = size(data);
%signal size
pth = find(0.35<data<0.3);
%set the threshold between 0.35 and 0.3
[m1 n1] = size(pth);
%get the threshold point
t=1;
signal=[];
%built an empty array to put signal code sequence
for(i = 2:m1-1)
    level = pth(i+1)-pth(i);
    if(level>300)
% FM0 decoder
        a=pth(i);
        b=pth(i+1);
        data_bs = data(a:b);
        a = find(0.625<data_bs);
        data_bs(a)=1;
        b=find(0.625>data_bs);
        data_bs(b)=0;
        btag=(diff(data_bs)==1|diff(data_bs)==-1);
        flip = find(btag == 1);
        m2 = size(flip);
        for(j = 1:m2-1)
            d = flip(j+1)-flip(j);
            if((d<10)&&(d>6))
                m=signal(1,t-1);
                if(m==0)
%0 corresponding two times voltage change in a period so we should
                else
%skip one when the last code already is 0

                signal(1,t)=0;

                t=t+1;
                end
            end
            if((d>10)&&(d<19))
                signal(1,t)=1;
                t=t+1;
```

```
            end
            if((d>19)&&(d<30))
                signal(1,t)='v';
                t=t+1;
            end
            if(d>30)
                signal(1,t)=3;
                t=t+1;
            end

        end

    end
    if((level<500)&&(level>150))
% reader signal tell the real code begin to transmit
        for(j=pth(i):pth(i+1))
            data(j)=2;
        end
        signal(1,t)=2;
        t=t+1;
    end
    if((level<150)&&(level>70))
% reader signal 1
        for(j=pth(i):pth(i+1))
            data(j)=1;
        end
        signal(1,t)=1;
        t=t+1;
    end
    if((10<level)&&(level<70))
%reader signal 0
        for(j=pth(i):pth(i+1))
            data(j)=0;
        end;
        signal(1,t)=0;
        t=t+1;
    end
    if(level<10)
        for(j=pth(i):pth(i+1))
            data(j)=0.5;
        end
    end

end
fid = fopen('code.txt','w');
% put the signal code in txt file
```

```
fprintf(fid,'%g\t',signal);
fclose(fid);
```