RFID Anti-Collision System Using the Spread Spectrum Technique

Document ID: PG-TR-050426-AR
Date: 26 April 2005

Anil Rohatgi
777 Atlantic Ave. Atlanta GA 30332-0250
Voice: (404)894-8169 Fax: (404)894-5935
http://www.propagation.gatech.edu

RFID Anti-Collision System Using the Spread Spectrum Technique

By: Anil Rohatgi
Supervisor: Prof. Gregory D. Durgin
Georgia Institute of Technology Propagation Group
4/26/2005

Introduction

The abbreviation RFID stands for “Radio Frequency Identification”, and is a technology that is vastly and rapidly advancing. An RFID system creates a situation where any product can be labeled with essentially any information, and the owner can access this information at any time without having direct contact with the item. The immediate and most prominent goal of RFID is to allow the owner of the system to keep track of his inventory without manually having to handle each individual item. There are also numerous other applications that can be extended for these systems. Essentially, any pertinent information about an item can be stored on the item itself, making automation and control incredibly simple, fast, and noninvasive. The possibilities are endless.

The goal of this project was to design and fabricate a fully functional RFID system from scratch. The system was designed to sense and identify 255 mutually exclusive items. The tag was designed to encode a variable identity and transmit this information. On the receiver end, an algorithm was developed in Matlab to process, detect, and decrypt the incoming information to recover the identity of the chip it is currently sensing.

How RFID works

The basic RFID system consists of two essential parts: the RFID interrogator, and the RFID tag. The purpose of the RFID interrogator is to transmit a set frequency for detection, and then receive this waveform back in a modulated form. The RFID tag is the part that is placed upon the desired item. Inside this tag is where the information and identity of the item is stored. Once the tag senses an incoming waveform, the data logic

inside the chip modulates the waveform and sends the modulated signal back to the interrogator. The modulated signal is in the form of a bit sequence, inside of which is the desired information and the identity of the item. Figure 1 shows the basic components of an RFID system, and how they interact with each other.

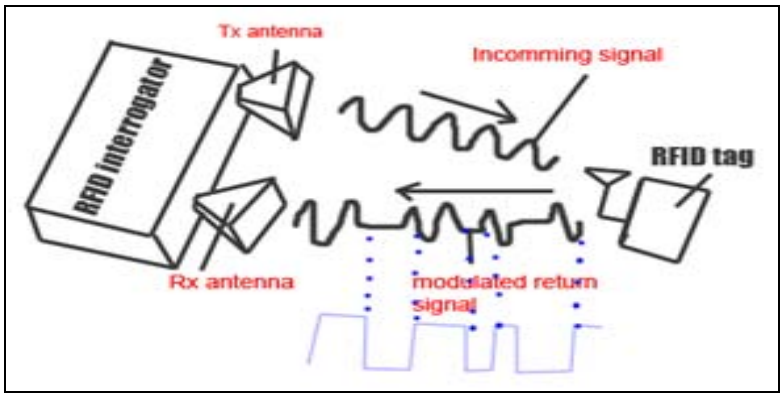


Figure 1 Basic RFID components

The way the RFID tag modulates and reflects the incoming waveform into a bit sequence is through the use of a diode and the reflection properties of a transmission line. Located between the tag's antenna and the data logic circuits used to control the modulation is a transmission line with a diode attached to the end. When the data logic circuits desire to transmit a bit one, a high voltage is output across the diode, forcing it to act like an open circuit. The reflection coefficient of an open circuit transmission line is one, and all of the incoming waveform is reflected back to the interrogator. However, when the control logic desires to send a bit zero back to the interrogator, it outputs a low voltage across the diode and forces it to look like a short circuit. This short circuit transmission line has a reflection coefficient of zero, and none of the incoming waveform is reflected back. By

toggling the voltage seen across the diode, the data control logic circuit can effectively transmit any desired bit sequence back to the interrogator.

Anti-Collision

The system described above works effectively if there is only a single tag in the sensing environment; however, problems arise when multiple objects modulate and reflect the incoming signal. The ability for the receiver to identify which item it is receiving information from in the presence of multiple responses is known as Anti-Collision. The method implemented in the proposed system is known as Spread Spectrum technology.

How Spread Spectrum Works:

The theory behind using the Spread Spectrum exclusion process is as follows. The data that is to be transmitted is multiplied by a pseudo random sequence. Each tag contains a unique sequence. The bit rate of the pseudorandom sequence is much higher than the bit rate of the data. The output of this multiplication (high frequency) is then used to toggle the diode, and a high bit rate waveform is sent back to the interrogator. Figure 2 shows this process and the result.

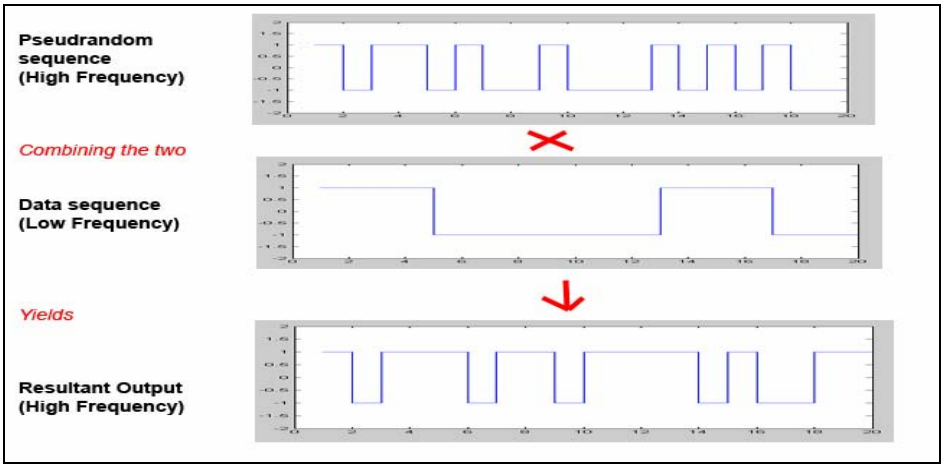


Figure 2. Tag combination process

On the interrogator side, what it sees is a combination of multiple high frequency waves arriving at the same time. In order to recover the information, the receiver must have a way to process the data. The key to this recovery process lies within the unique pseudorandom sequence that was combined with the data from the desired tag. If the interrogator knows the pseudorandom sequence associated with the desired tag information, it can recover the data. To do this, theoretically all that needs to be done is to multiply the incoming signal by the same pseudorandom sequence that is combined with the desired data. Assuming the incoming waveform is centered on zero and has peaks that range from negative one to positive one, doing this operation would force the modulating sequence to converge to one for the desired wave, thus only leaving behind the low frequency data wave. When this unique pseudorandom sequence is multiplied with the other waves comprising the incoming waveform received by the interrogator, other high frequency waves are created. Therefore the only low frequency wave left behind is the desired original data. Figure 3 demonstrates the recombination process.

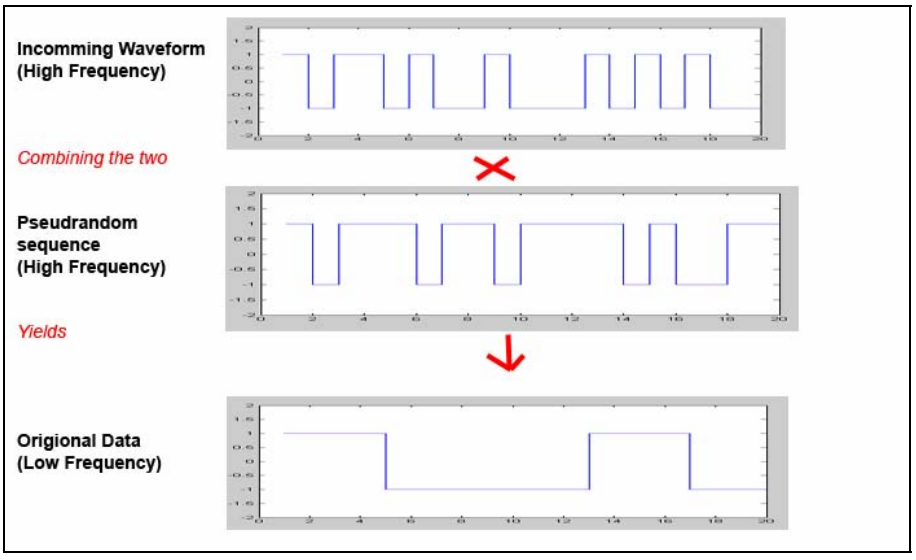


Figure 3. Recombination process

Once the multiplication is completed, the original data can be recovered by passing the resulting waveform through a low pass filter. All the high frequency waves generated from other tags will be removed.

System Tag Design

The system that was implemented was designed to generate 255 unique pseudorandom sequences based on a variable phase shift located on the tag. This phase shift is where the identity of the tag will be encoded. Two 8-bit shift registers were connected in a feedback sequence in order to generate their maximum length pseudo random sequence of 255 bits. These two sequences were then combined through an XOR gate, and the resulting output is used as the modulating high frequency signal for the data. Figure 4 shows the circuit diagram for these two pseudorandom sequence generators.

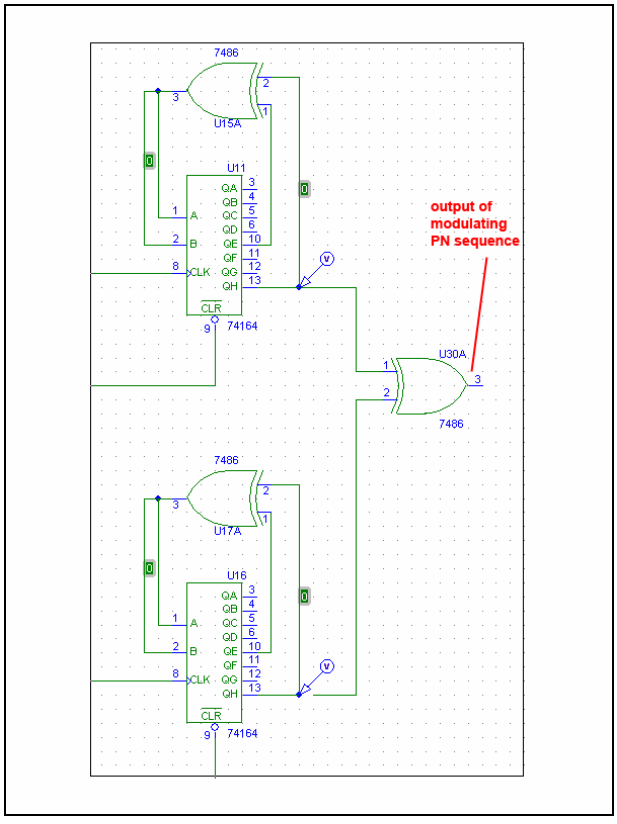


Figure 4. Schematic for the two PN generators

The phase shift, in bits, between the two pseudorandom sequences before they are combined, determines the identity of the tag. Thus, there are 255 possible phase shifts between the signals before the sequences begin to repeat, and thus there are 255 possible tags that can be implemented using this schematic. Figure 5 shows the two pseudorandom sequences with no phase shift between them. Figure 6, shows the same two sequences delayed apart by 241 ticks.

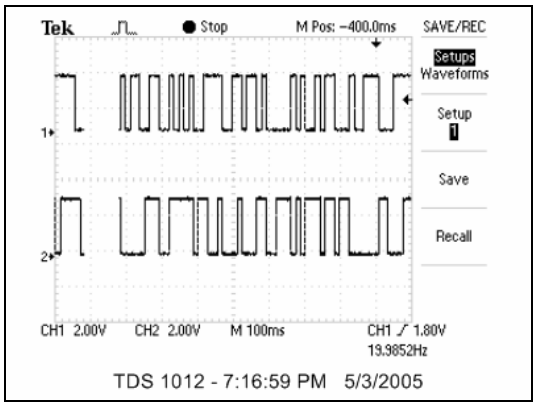


Figure 5. Two PN sequences with 241 bit delay

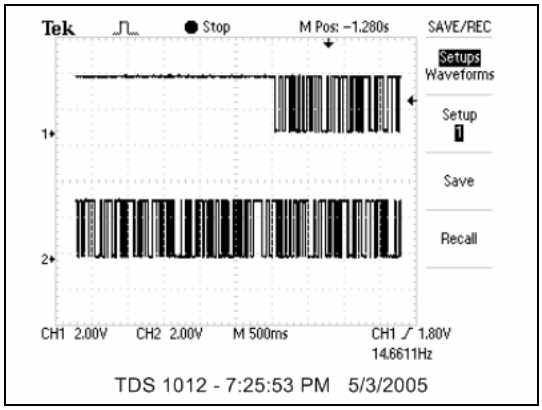


Figure 6. Two PN sequences with no delay.

The phase shift was accomplished by delaying the clock signal fed into the second shift register. A counter was employed as a mechanism for timing the number of bits needed to delay the clock signal sent to the second generator. The value loaded initially into the counter, determines how many ticks the counter will wait before allowing the second shift register to begin, and therefore how many bits the signal will be delayed by. This initial value is established using a set of dipswitches, and can be changed to mimic any of the 255 possible tags. The numeric identity of the tag is thus encoded and related to initial counter value by the equation:

tag ID number = 255 – initial counter value **Equation 1.**

In order to keep the counter for resetting itself every time it rolled over and consequently restarting the delay sequence, the outputs of the counter were fed through a series of AND gates set to trigger once the number 254 had been reached. On the next clock tick, the output of the AND gates will be high. This value was fed directly into the count enable input of the counter forcing the counter to freeze at 255. The output of the AND gates was also inverted and used to AND together with the original clock signal being fed into the second shift register. This ensured that the clock signal would remain zero until the counter had reached its end value of 255. Once this number has been attained, output of the AND gates is one, and the original clock signal is allowed to pass into the second shift register. Figures 7 and 8 show the original clock signal and the second clock signal with a delay. In Figure 7, the delay and ID number of the tag is three, and in figure 8 the delay and ID number is 252.

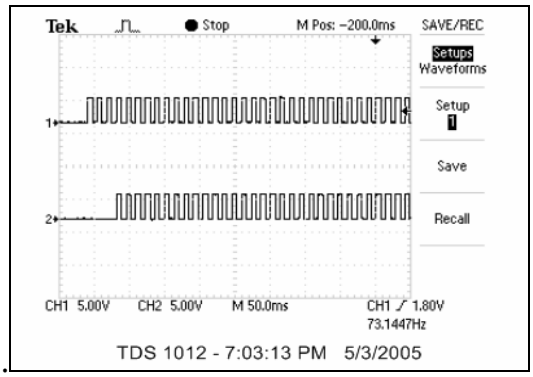


Figure 7. Clock signal delayed by 3 bits

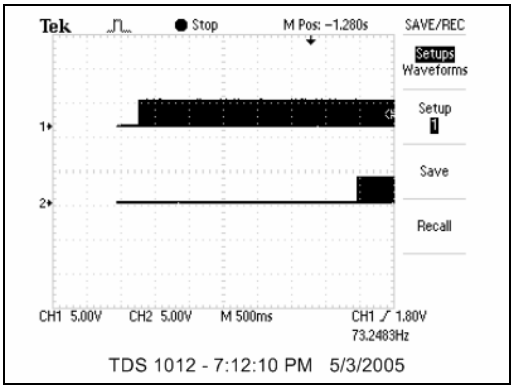


Figure 8. Clock signal delayed by 252 bits

The schematic of the phase delay circuit is shown in figure 9 A full schematic of the entire tag circuit can be found as an appendix to the report.

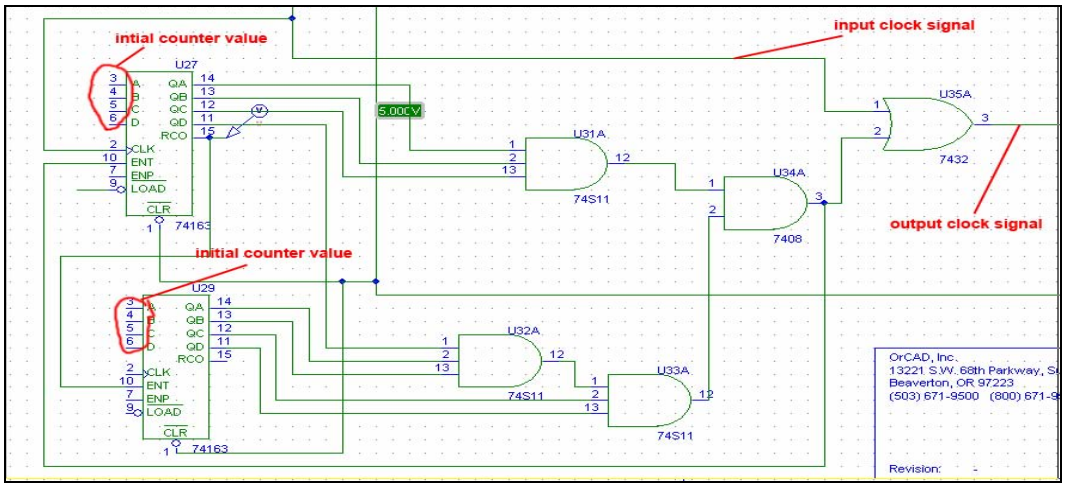


Figure 8. Phase delay circuit

System Tag Hardware

The tag created for this system was built on a proto board. The chips were powered by a 5 Volt power supply, and drew a current of .02Amps. The overall power consumption of the entire tag came out to be 0.1Watts. The chips laid out on the board consisted of two 8-bit shift registers, one 555D timer chip, two 4 bit counters, two quad AND gate chips, one inverter chip, one XOR gate chip, one 8 bit dip switch, and 5 toggle switches. A picture of the board lay out is included in Figure 9.

The dip switches in the bottom of the board are where the initial conditions for the counter are set. The set of toggle switches on the right side of the board are used when the ID of the tag is changed. In order to reset the tag ID, first set the dip switches to the desired ID by using the inverse of equation 1. Then set all the switches to the upward position in order starting from the bottom switch to the top switch. Once the top switch is flipped, the clock is stopped. Next, flip all the switches back downward starting again from the bottom switch. Once the top switch has been flipped downward, the chip has

been reset, and is transmitting the new ID. A larger version of figure 9 can be found in the appendix.

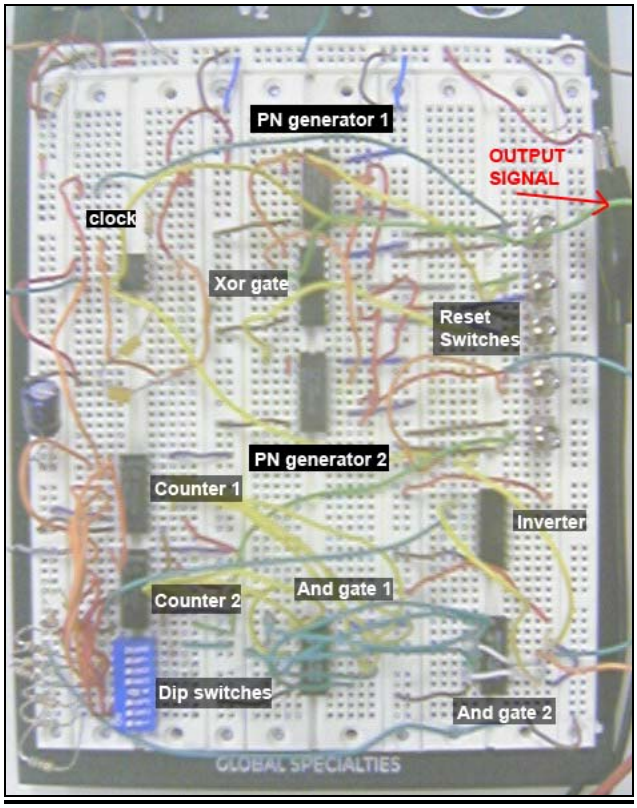


Figure 9. Tag Hardware layout

System Interrogator Design

The design for the interrogator for the RFID system consisted of multiple parts. The transmission and receiver antennas were two copper prefabricated horn and rectangular waveguides. There were to be placed next to each other, in order to use their similar radiation patters to essentially narrow the direction of transmission and reception by nullifying the noise of the adjacent antenna. The transmission antenna was to be attached to a function generator that would output a sine wave of 915 MHz, the frequency

commonly associated with UHF RFID. This wave is what irradiates the tag, and what gets modulated and reflected by the tag.

The receiving antenna would be attached to a spectrum analyzer that would be set to monitor the peaks at frequencies near 915MHz. If the system is functioning correctly, the peaks should rise and fall according to the data being reflected from the tag. This data would then be run through a threshold chip, most likely a Schmitt trigger, and then logged into a computer through a data acquisition board. From here, the recovery of the tag's identification would take place in Matlab.

Matlab Data Processing:

Once the bit sequence being received from the interrogator has been logged and recovered, an algorithm was written in Matlab in order to simulate all the possible combinations of pseudorandom sequences that could be output by the tag, and compare them to the incoming sequence. Because there are 255 possible phase shift combinations that can generate the output pseudorandom sequence of the tag, there needed to be 255 simulated sequences to compare with the input data stream. Once a match has been found, the computer logs the phase shift used in the simulation, and returns this number as the ID of the tag.

However, there is still another problem because the interrogator cannot tell when the sequence begins and when it ends. Although sampling 255 bits of the incoming waveform ensures that the entire sequence is captured, it does not ensure what bit the sequence begins and ends with. Thus if directly compared to the identical simulated

sequence, their phase differences would cause the comparison to give a false negative result. Thus, in order to ensure the correct tag ID, each simulated sequence needed to be bit shifted 255 times, and each time compared to the input data. Thus the total number of iterations for a 255 tag system turned out to be 65,025. The code for a rough implementation of this algorithm is included as an appendix.

Data Extension

The system described above is only designed to transmit the ID of a tag to the receiver, and not actual data. The tag built only contains chips to generate the pseudorandom sequence, but there is no data to mix it with. This system can easily be changed to include data, by simply multiplying the output pseudorandom sequence with a lower frequency data stream, and using the result to modulate the diode. However, in the implementation used for this demonstration, the pseudorandom sequence drove the modulating diode directly.

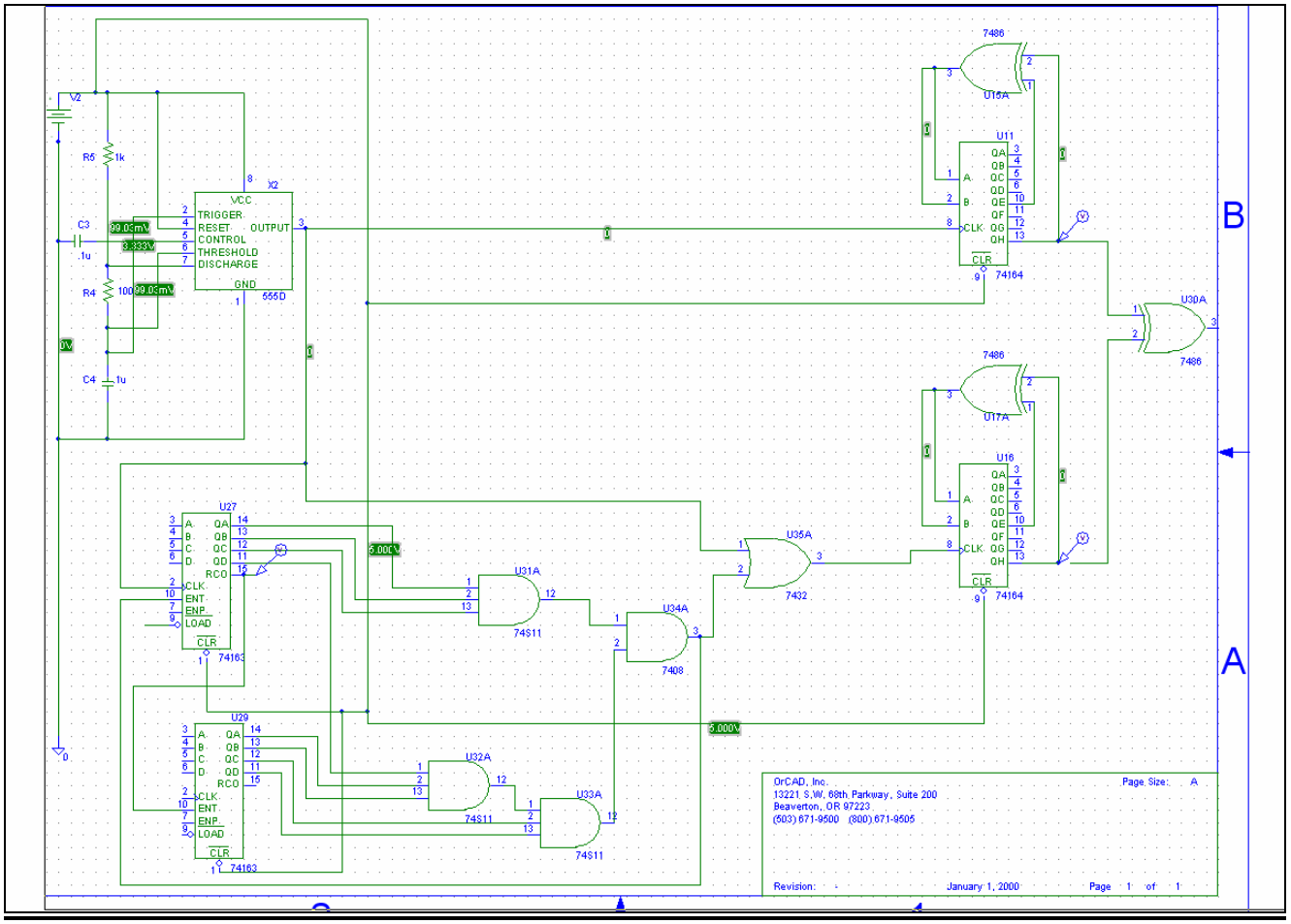
On the receiver end of the system, in order to be extended to include data, much more of the data processing would have to be done in real time, in order to perform low pass filtering to recover the data. The same algorithm can be used, but the condition for match would be different. Instead of looking for a matching bit to bit correspondence between a simulated sequence and the input data stream, the algorithm would look for the presence of a low frequency waveform. This would be the data in the tag, and the phase shift that generated the pseudorandom sequence that caused the low frequency waveform to appear is the ID of the tag.

Conclusions

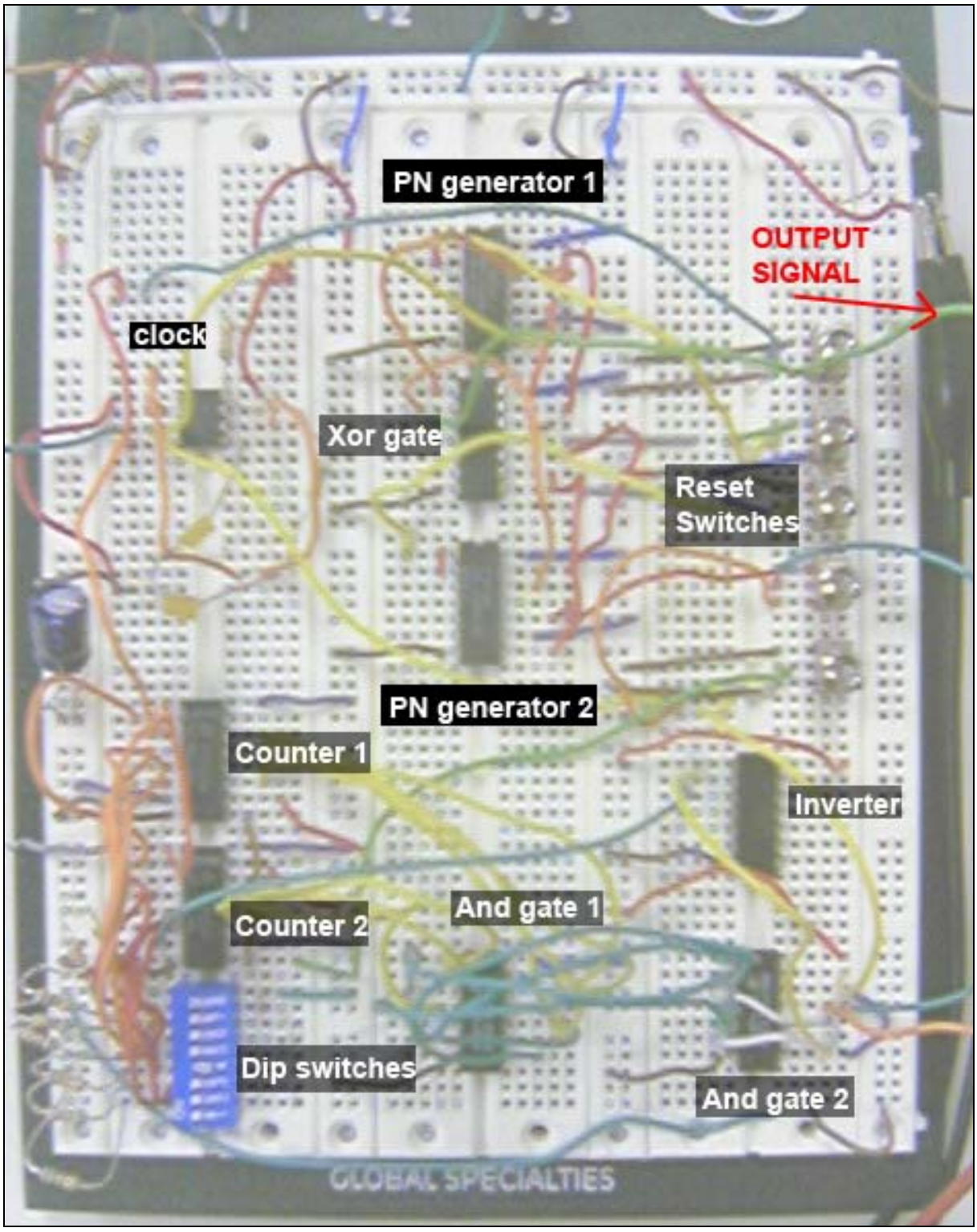
Radio Frequency Identification is a technology with the potential to change the world. More and more people are getting involved in this industry, and it is on the verge of flourishing. Major companies such as Wal-Mart are setting deadlines and mandates to switch their entire inventory to an RFID system in the near future. However, this fledgling technology is not yet developed to the point of a mass scale distribution. Problems involving anti-collision, read range and penetration depth have prevented RFID from hitting the market in a big way. But for every problem, there are numerous engineers working in the background to solve them.

The design that was proposed and developed for this project solves the anti-collision problem for up to 255 tags located in the sensing environment. Each part in the system was designed, developed and independently tested. However, due to time constraints, the components were never fully integrated together and tested as a whole unit. The scope of this project seemed too ambitious for a one semester time frame. Future work may include system integration and miniaturization of the components. Although the system was never fully implemented, the results from the testing are enough to show proof of concept.

Appendix 1: Complete Circuit Diagram



Appendix 2: Real Circuit Picture



Appendix 3: Matlab Code

```
%This program takes in an input waveform from a DAQ board
%and processes the signal to recover the ID of the chip
%the signal was sent from

%By Anil Rohatgi

load inputtest 5 %inputtest5 is the received signal from the Data Acquisition board
pickoff = 5;

%------------finished test data-------------------------

initial1=[1 1 1 1 1 1 1 1];   %initializing the chip

xor1=0;
xor2=0;
%---------------------------generating initial PN sequence----------
for i=1:256
  xor1=initial1(8);           %selecting pickoff points
  xor2=initial1(pickoff);
  out(i)=initial1(8);
      for j=1:7              %shifting
         index=abs(9-j);
         initial1(index)=initial1(index-1);
      end
      if(xor1==xor2)          %XOR
         initial1(1)=0;
      else
         initial1(1)=1;
      end
   end

 plot(out)
 %---------------------------phase shift and summation--------------------
 modulated=out;             %initializing the second signal that will be modulated
 answer=0;                 %Phase shift amount that generated input signal = RFID tag

 for shift=1:256
    templast=modulated(256);    %saves the original last value in the unshifted array to be assigned to the
first one
       for k=1:255           %shifting the signals before multiplying them together
          index2=abs(257-k);
          modulated(index2)=modulated(index2-1);
       end
     modulated(1)=templast;    %re-assigns the original last value to the first array position to loop the shift
     combined=modulated.*out;  %multiply the original signal with the shifted signal


            for phase=1:256          %now, shifts the combined signal and compares it to the input signal.
              templast2=combined(256) ;
                 for l=1:255
                    index3=abs(257-l);
                    combined(index3)=combined(index3-1);
                 end
```

```
            combined(1)=templast2;
            difference = combined-inputtest5;
               if (difference.*difference ==0)
                  answer = shift
               end
         end
end
```

Appendix 4: Miscellaneous Pictures