Final Project Report
E3390 Electronic Circuits Design Lab

RFID Access Control System

Jeffrey Mok
Joseph Kim

Submitted in partial fulfillment of the requirements for the
Bachelor of Science Degree

May 11, 2007

Department of Electrical Engineering
Columbia University

Table of Contents

1. Executive Summary

RFID is a contactless identification technology based on the transmission of radio frequency waves. Its advantage over its predecessor, the barcode system, is its increased range and increased data storage capacity. The typical RFID system consist of three main components, the transponder (or tag), the reader, and the application.

The tag is the data storage component. The tags we will use in this project will be passive tags, meaning they do not have an internal power supply. The reader activates, powers, and communicates with the tag using electromagnetic waves. Once activated, the tag will respond to the reader with the information that is stored in its memory. The reader extracts this information and sends it the application component for processing.

Our project demonstrates a low-cost RFID access control application. Tags will be used as keys, with the system able to configure tags to be “allowed” or “denied”.

2. Block Diagram, Design Targets, and Specifications

Block Diagram

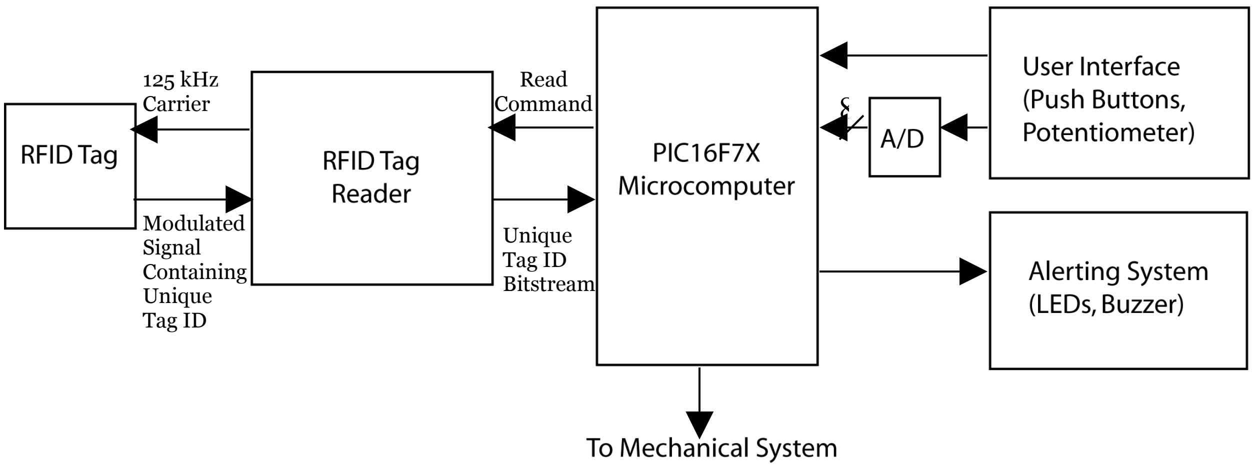


Figure 1: RFID Access Control System Block Diagram

Design Targets and Specifications

RFID Tag – Purchased since a practical (small and portable) tag is out of our manufacturing capabilities.
RFID Tag Reader – Constructed using discrete components and IC's.
Microcomputer programming – Programmed on PIC16F7X MCU in assembly language using Microchip's MPLAB.
User Interface – This includespushbuttons (read command), switches (configure, change operation mode – normal or setup).
Alerting System – This includes LEDs to indicate "accept" or "reject", error indicator (or might have it just blink between accept and reject lights), display RFID's unique code.
Mechanical System – Locking mechanism. Not implemented at this time.

-

3. Individual Block Descriptions

RFID Tag

Atmel read-only TK5530 tags were chosen for this system. These tags respond to a 125 kHz wave with an 125 kHz AM wave containing a 64-bit rolling code at 3.9kbps. The code contains an 8 bit header followed by a unique ID code. The data is encoded using Manchester encoding.

These tags were chosen because of our knowledge of how to demodulate AM compared to tags that use other kinds of schemes such as FSK or PSK. Also, our application did not require. Also, we did not require the increased functionalities of more expensive Read/Write tags.

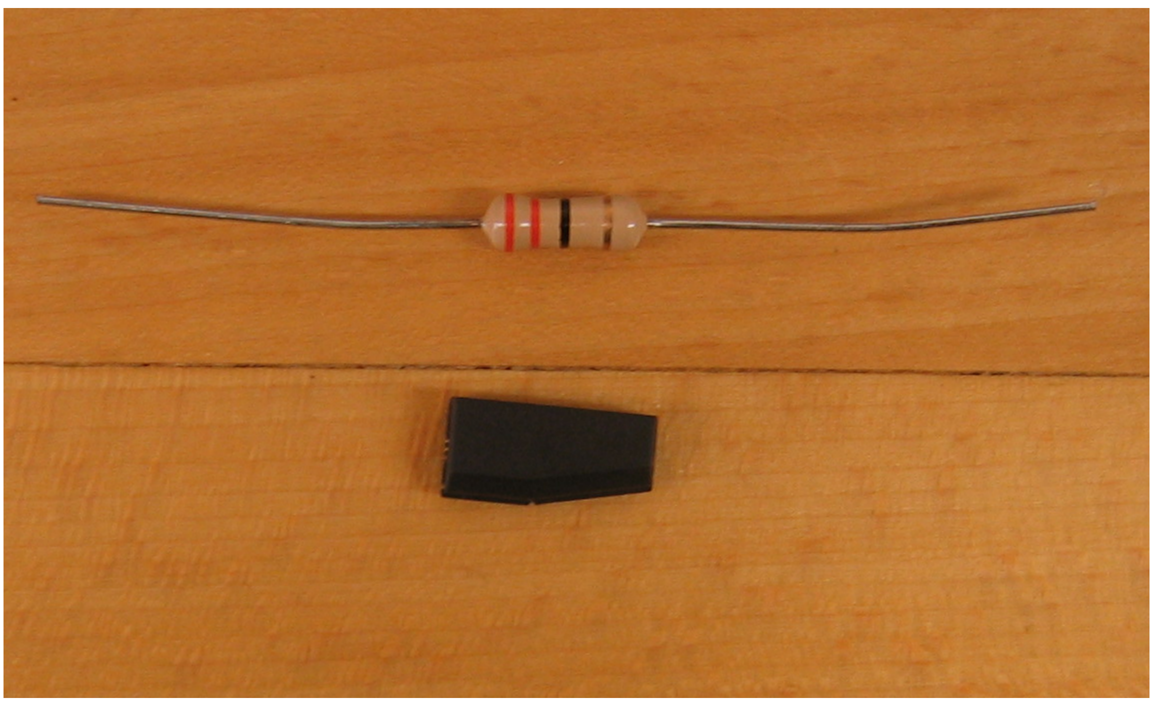


Figure 2: Atmel TK5530 Tag (with resistor for size comparison)

RFID Tag Reader

The purpose of the Reader component is to activate and power the tag, demodulate the response, and prepare the signal for the microcontroller. The components of this reader are: the antenna, signal generator, peak detector, low pass filter, and voltage comparator.

Antenna

Many antenna configurations were constructed for testing. Each had limited range and were difficult to use because the coils would come out of place.
In the end, we settled on a pre-made antenna that consisted of two coils wrapped around a ferrite coil in a transformer configuration. The inductance of the coils were measured, and an appropriate capacitor was chosen to tune the antenna to the resonant frequency using the parallel tank circuit equation:

$$f = \frac{1}{2\pi\sqrt{LC}}$$

This antenna still had very limited range. The range was no farther than one. But with this configuration it was possible to rest the tag directly on the antenna, allowing for a consistently good signal.

Signal Generator

A 125 kHz square wave signal generator is required to drive the antenna. We generated a signal from the MCU for this purpose, but due to time constrictions we did not have time to build a circuit to make the signal have the necessary voltage. For now, we are using a function generator as the signal generator. It is set to output a square wave at 125 kHz, 10 Vpp.

Peak Detector

The peak detector is used to extract the envelop of the AM signal. Figures 3 and 4 show the antenna input without and with the tag in proximity. Figure 5 shows the signal after the peak detector.

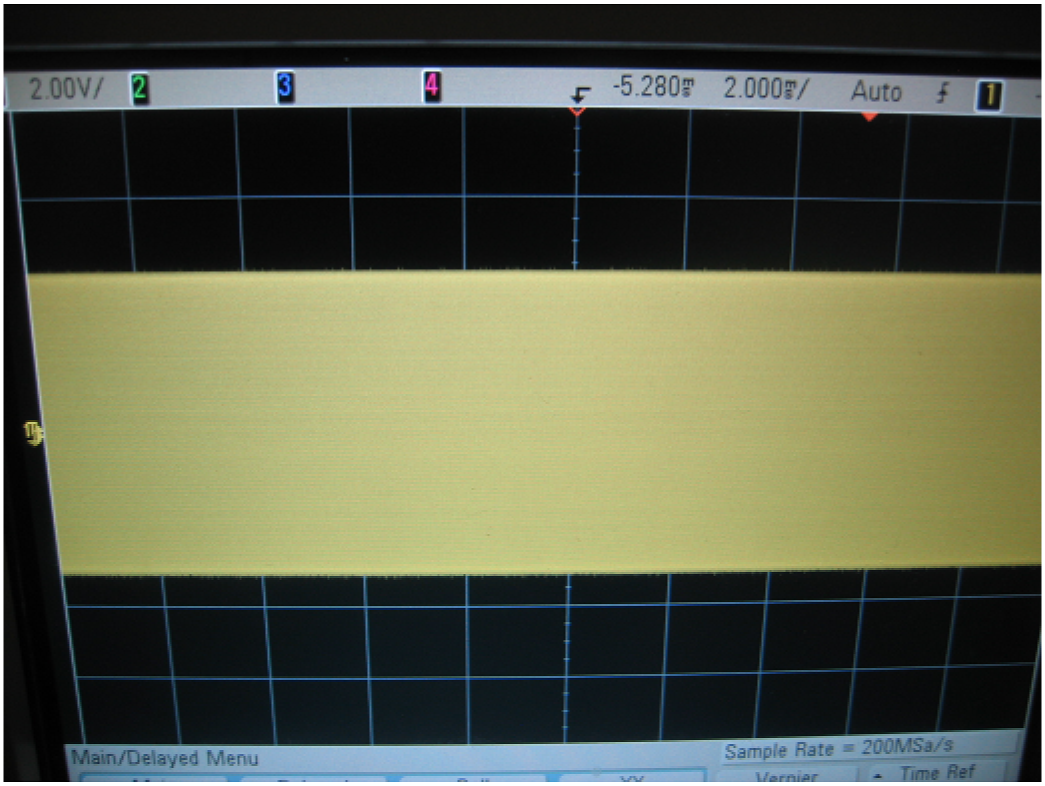


Figure 3: 125 kHz square wave

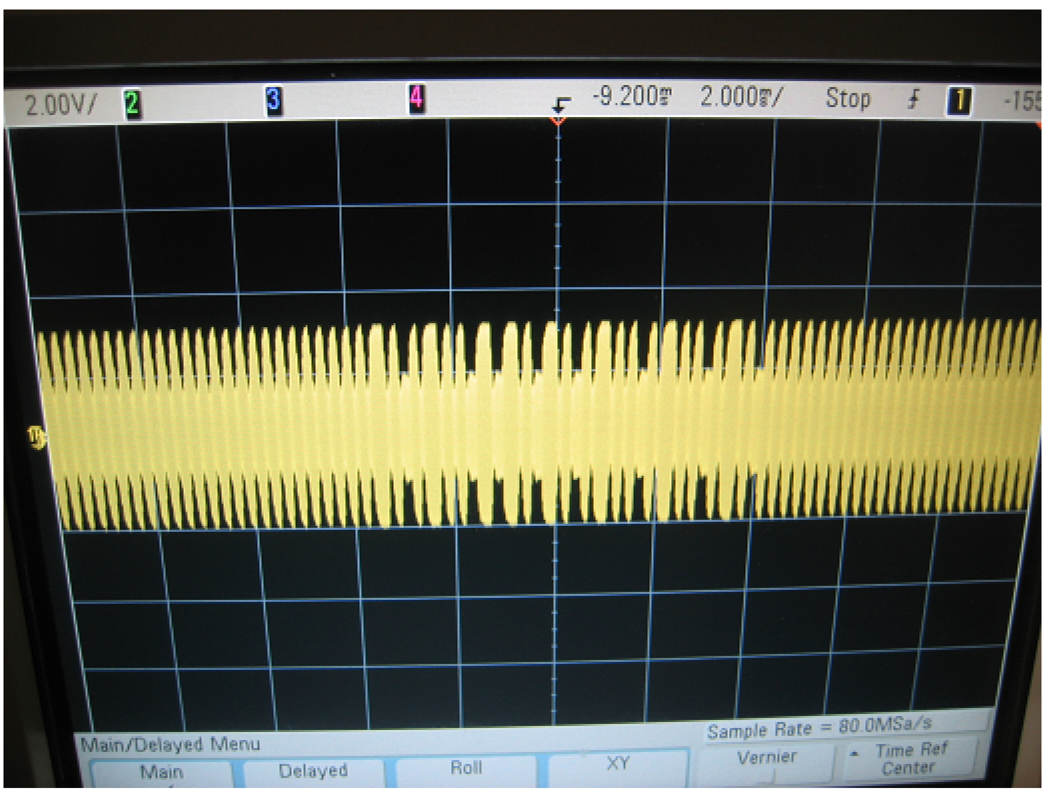


Figure 4: AM response from tag

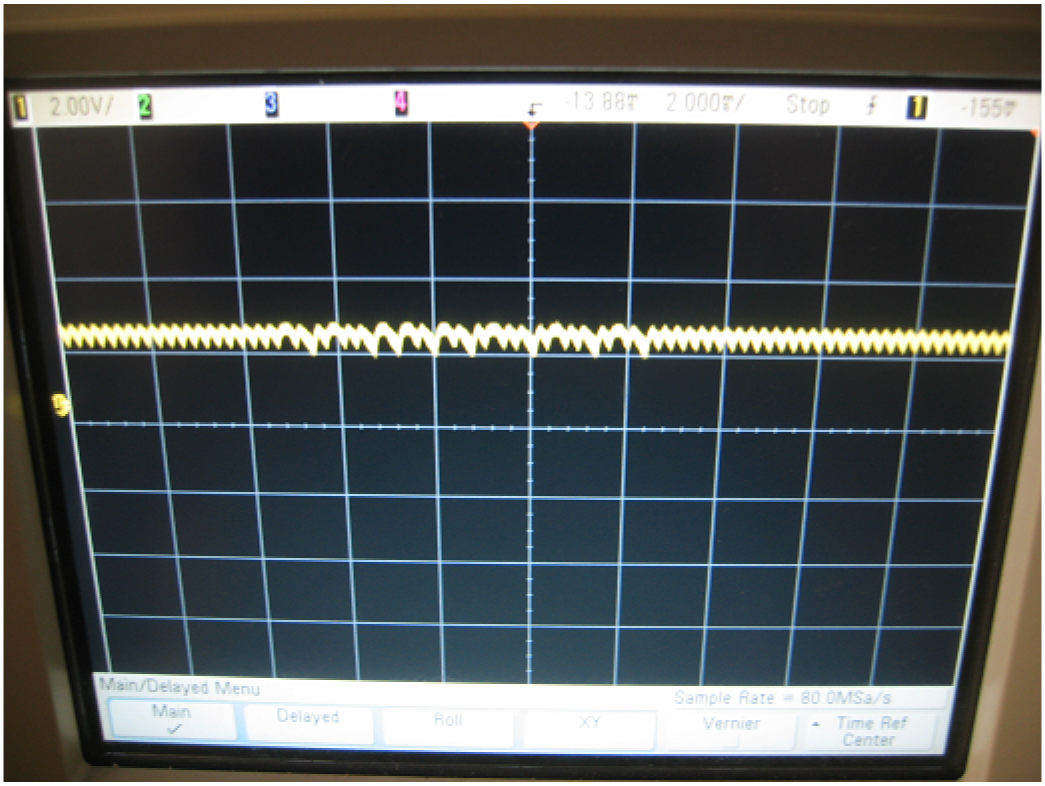


Figure 5: Output of peak detector

Low Pass Filter

A first order low pass filter with a cutoff of 10 kHz was constructed to reduce the carrier frequency. The data is at 3.9 kHz.

Voltage Comparator

The envelop signal is converted to a square wave in preparation for sending to the microcontroller. The LM411 comparator was used. Notice the noise in the signal. This noise greatly affected what the MCU was reading, causing inconsistent results in our application.

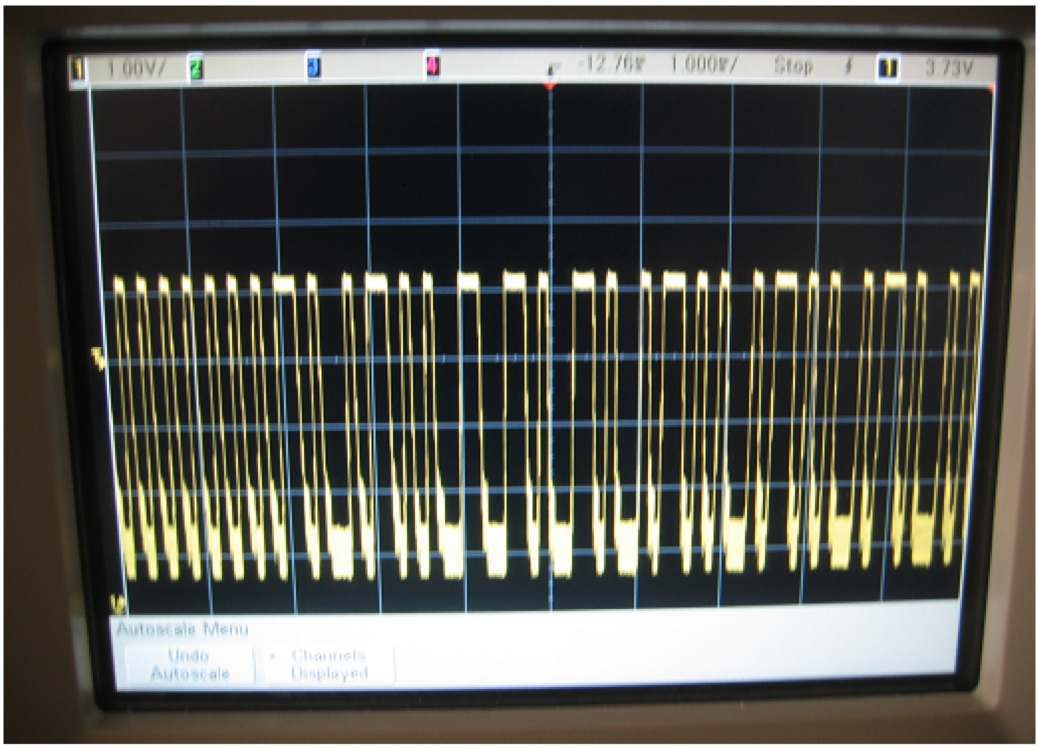


Figure 6: Output of Voltage Comparator

Two inverted Schmitt triggers were used to smooth out the edges. The resulting output was sent into the MCU.

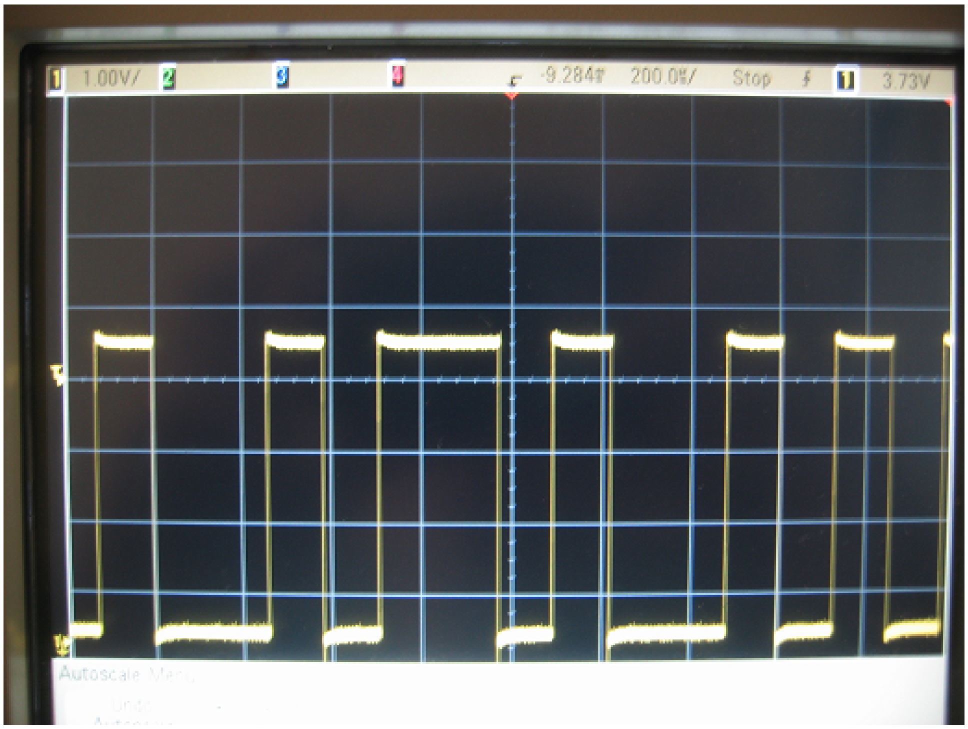


Figure 7: Output of Schmitt Triggers

Microcontroller

The PIC16F7X MCU was programmed in assembly language. The MCU is responsible for decoding the Manchester encoded data, extracting the data, controlling the LED's that indicate the ID, and managing the access control.

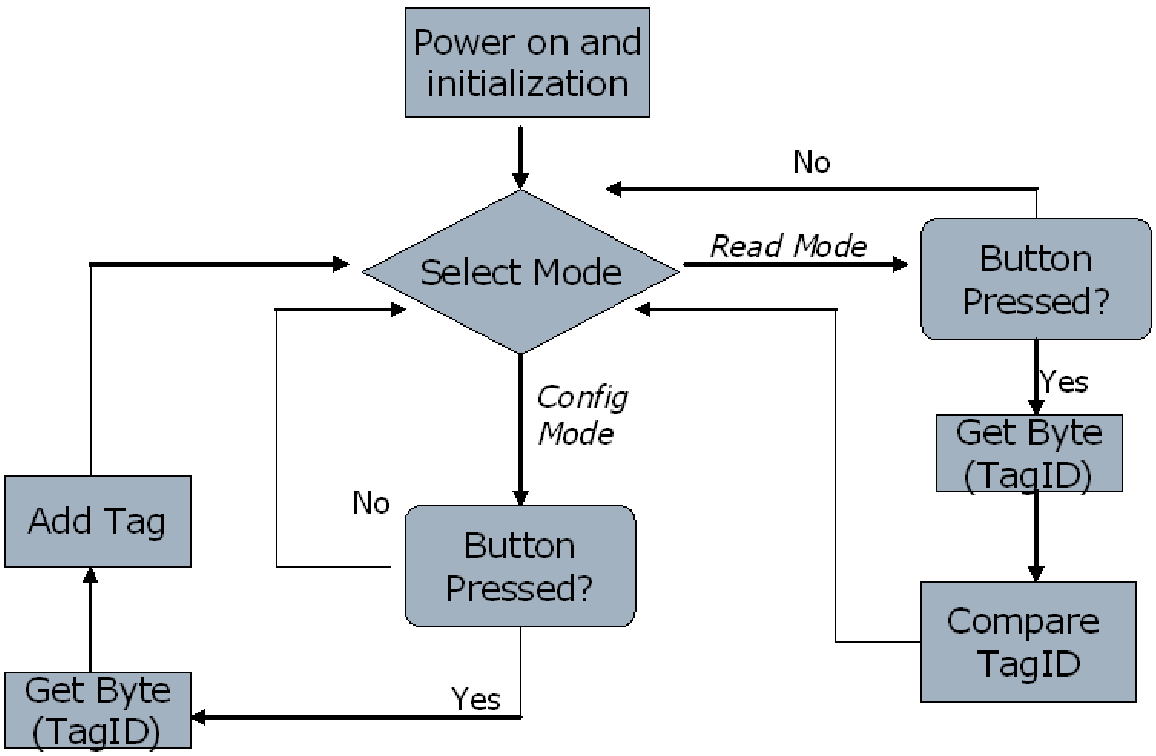


Figure 8: MCU Control Diagram

ID Extraction

The first step in reading the data is to find the header of the code. The Atmel chips have a header of E6 (11100110)

We devised a scheme to find the header as follows:

- First, phase correction:
  - -Keep sampling input pin (every two usec) until a high is read
  - -Next, keep sampling input pin until a low is read
- Finally, keep sampling input until a high is read

- Second, wait just over half a period to adjust for Manchester encoding and sample there at 3.91 kHz
  - -Sample 8-bits and check if all zeros; if not, rotate bits left and sample the next bit; repeat until all zeros
  - - Now keep shifting 8-bit window until the first high-level is found; this bit and the next 7 bits make up the header
  - - After the header, sample another 8-bits: this is the unique tag ID

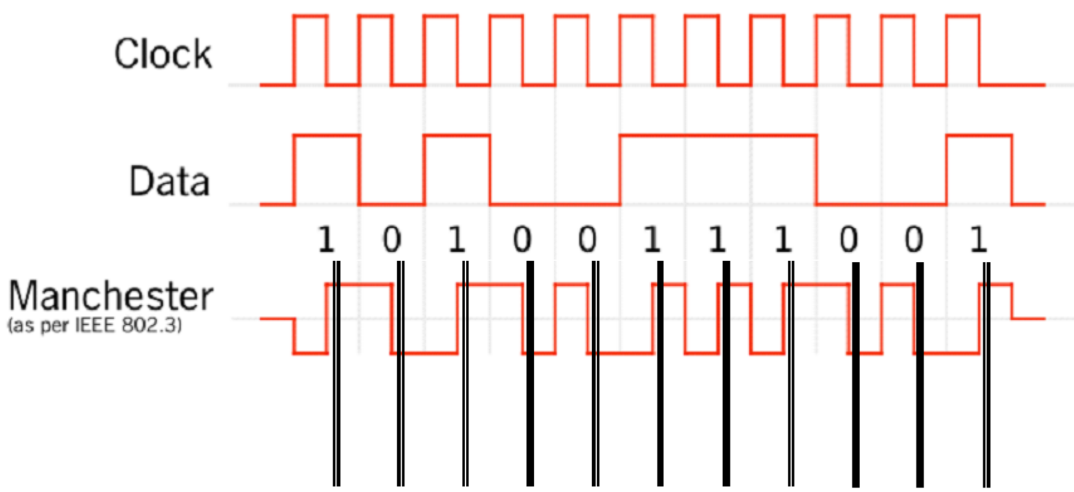


Figure 9: Manchester Encoding

Once decoded and extracted, the data is output to the LED's. See code and Schematics for more detail.

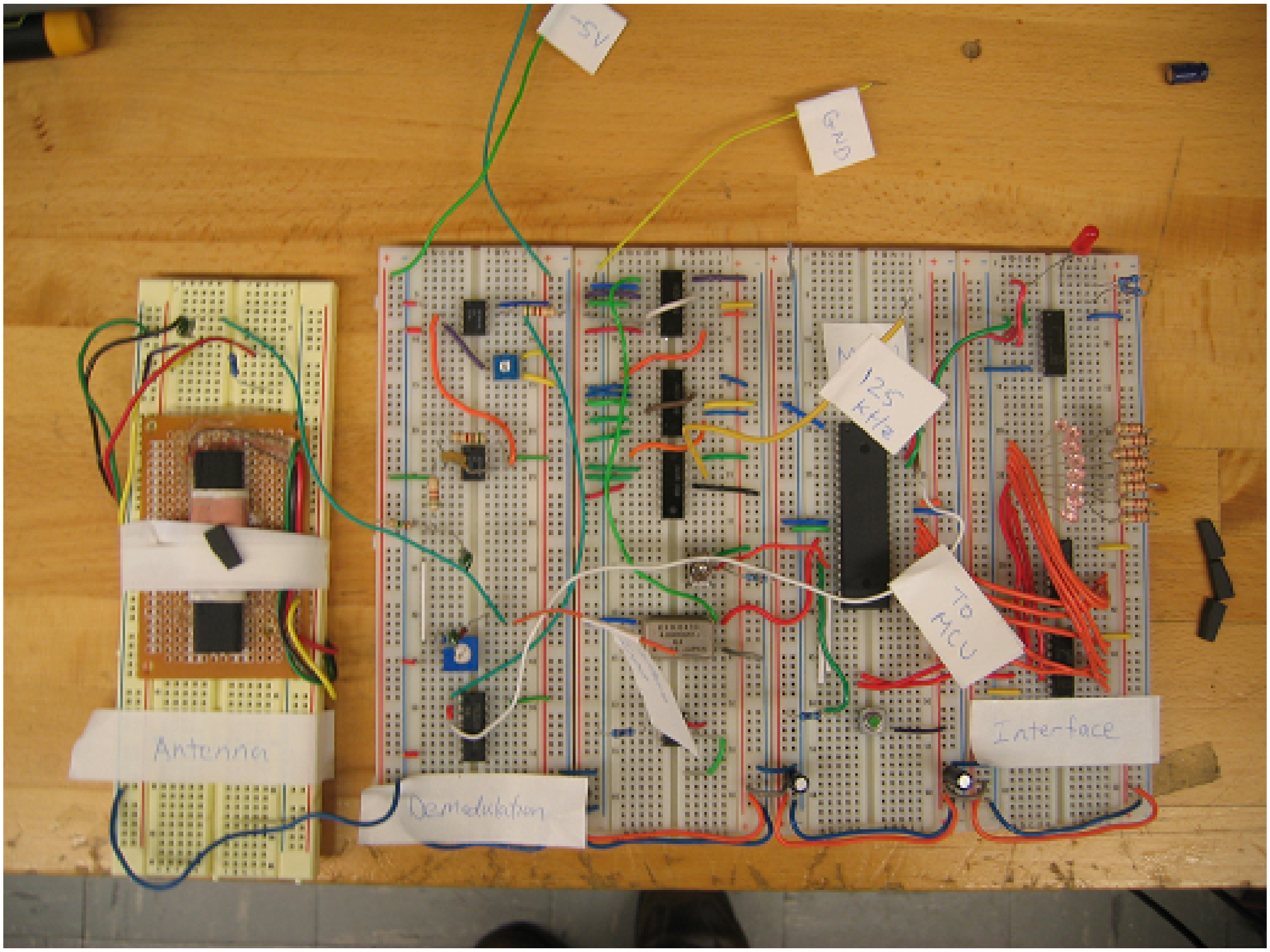
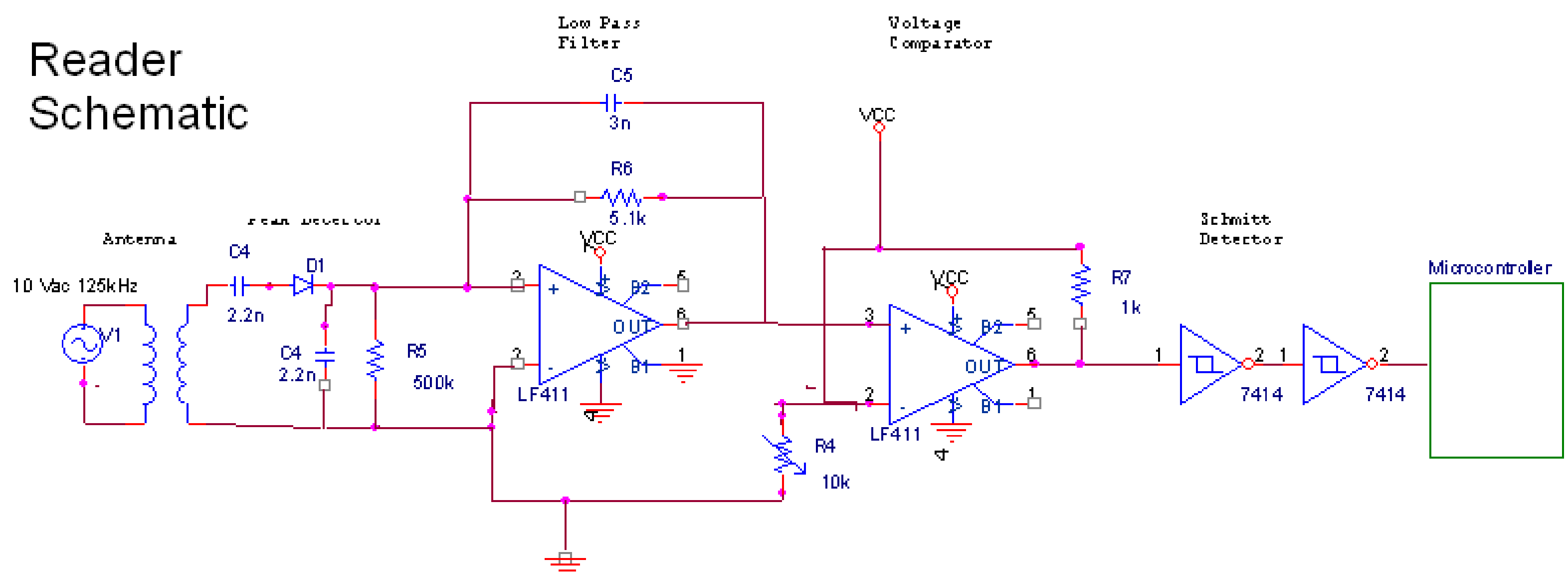


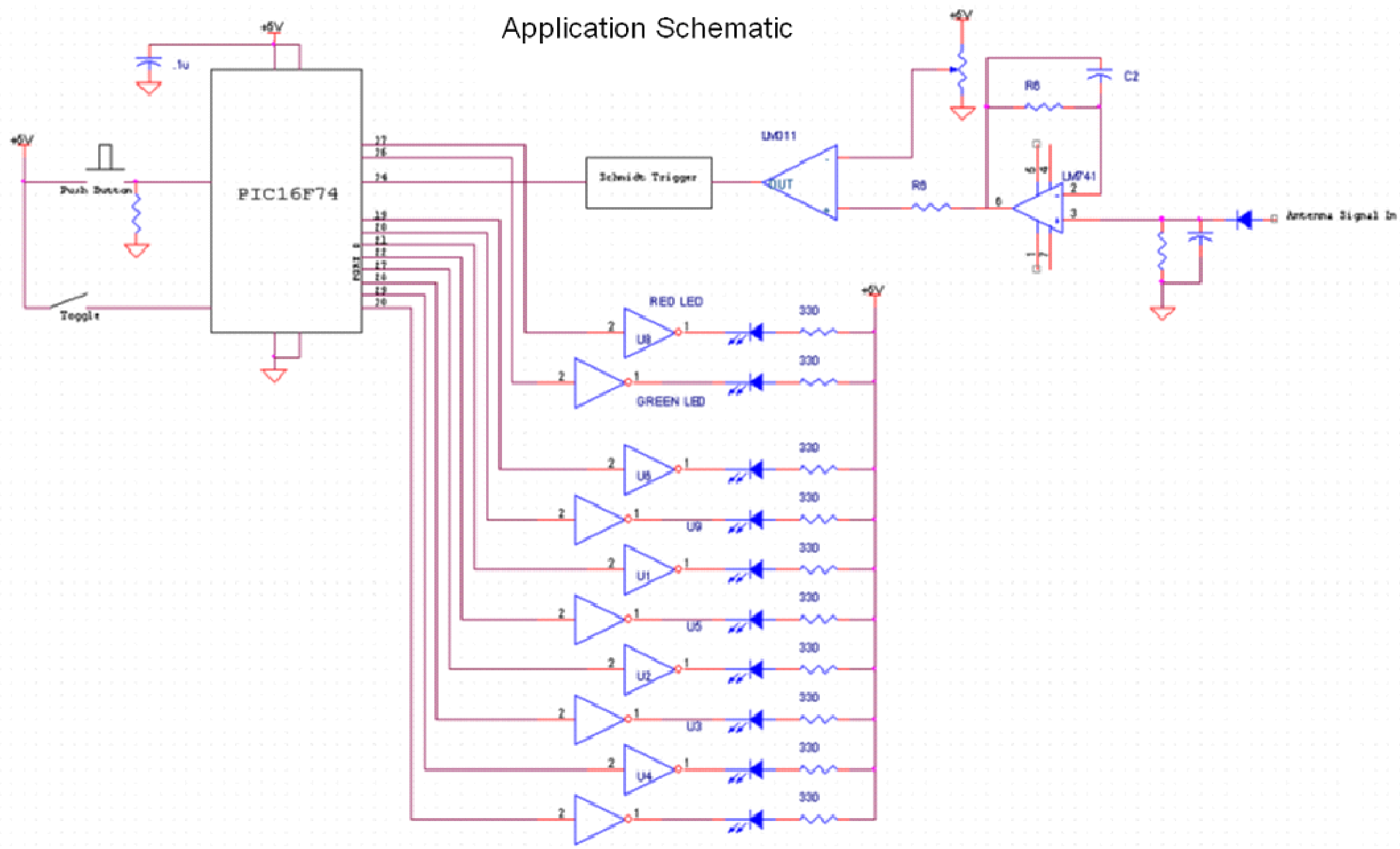
Figure 10: Final Completed System

Reader Schematic



Title	RFID Reader Stage			
Size A	Document Number <Doc>			
Date:	Thursday, May 03, 2007	Sheet	1	of 1

Application Schematic



4. Bill of Materials

Part	Manufacturer	#	Cost
TK5530 Tag	Atmel	5	5 * 2.60
Antenna		1	28
PIC16F7X	Microchip	1	1.50
LM411	National Semiconductor	2	2 * 1.50
7414 Schmitt Trigger	Texas Instruments	2	2 * .50
Capacitors		3	
Resistors		4	
Total Cost			Approx. 38.70

5. Health, Safety, and Environmental Issues

a. Product Dangers

No dangers related to the use of our project are noted. Care should be taken to hook up the circuit properly and use of correct voltages.

b. Health Hazards

No health hazards associated with RFID technology have been noted.

c. Environmental Hazards

i. FCC regulations cover RFID devices ranging in frequency from 9kHz to 64 GHz. According to FCC Part 15, Section 15.209, the maximum E field for a device operating between .009-.490 Mhz at a measuring distance of 300m is 2400/f uV/m.

ii. Electric Shock Problems. All wires are insulated,

6. Gantt Chart

RFID Reader

Jeffrey Mok, Joseph Kim

	30-Jan	6-Feb	13-Feb	20-Feb	**27-Feb**	6-Mar	13-Mar
	1	2	3	4	**5**	6	7
Research RFID Types, Existing Apps (Jeff, Joe)							
Research RFID Designs (Jeff, Joe)							
Determine which parts to buy (Jeff)							
Determine subsystems to design (Jeff, Joe)							
Meet with Prof Stolfi; working with MCU (Joe)							
Improve Antenna Design and Reader subsys (Jeff)							
Program Microcomputer (Joe)							
Design/Assemble User interface (Jeff, Joe)							
Mechanical Subsystem if time? (Joe)							
Form factor design (Jeff, Joe)							
System Debugging (Jeff, Joe)							
Project Presentation							
Final Report							

	27-Mar	3-Apr	10-Apr	17-Apr	**24-Apr**	**1-May**	**3-May**	**10-May**
	9	10	11	12	**13**	**14**	**15**	**16**
Research RFID Types, Existing Apps (Jeff, Joe)								
Research RFID Designs (Jeff, Joe)								
Determine which parts to buy (Jeff)								
Determine subsystems to design (Jeff, Joe)								
Meet with Prof Stolfi; working with MCU (Joe)								
Improve Antenna Design and Reader subsys (Jeff)								
Program Microcomputer (Joe)								
Design/Assemble User interface (Jeff, Joe)								
Mechanical Subsystem if time? (Joe)								
Form factor design (Jeff, Joe)								
System Debugging (Jeff, Joe)								
Project Presentation								
Final Report								

7. Criticism of this Course

The most positive thing about this course was the sense of achievement when the project was complete. We took a kind of technology that we did not any experience with before, but were able to use relatively simple ideas from our classes to implement commercial technology.

We may have spent too much time at the beginning of the semester defining our project. Perhaps this is good in that it reflects the detailed planning required in industry before a project is undertaken. But I think we would have benefited from a stricter schedule. Also, the possibility of this course becoming a two semester course should solve that problem.

A review of some electronic circuits material would have helped too. Again, a two semester course would help with this. It would also be interesting to see how some of the material from the other EE tracks could be part of the projects.

Appendix

Software Code

```
		LIST P=16F74
		title "Main Operator"
		__CONFIG B'11111110110010'

;*********************************
;
; RFID MCU Program
; 3-2007 Joseph Sungee Kim
; JSK2105@COLUMBIA.EDU
;
;*********************************
;	OSC1 freq (clock in) = 4MHz
;	Instruction cycle approx 1 usec
;
;
;*********************************

		#include	<P16F74.INC>

;
;	Variable Declarations
;

Count	equu		20h
Temp	equ		21h
State	equ		22h
TagID	equ		23h
Cycle1		equ		24h
Cycle2		equ		25h
Cycle3		equ		26h
Tag1	equ		27h
Count2		equ		28h
Count3		equ		29h

		org	00h			;Reset Vector

		goto	initPort

		org		04h			;Interrupt Vecotr
```

```
                goto    isrService      ;goto interrupt routine

                org             05h                     ;Beginning of Program Storage

;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
; Port Initialization
;
initPort
                clrf    PORTD                   ; LED displays (OUT)
                clrf    PORTC                   ; Push buttons (IN)
                clrf    PORTB                   ; DIN(b0-OUT),DOUT(b1-IN)
                bsf             STATUS,RP0
                clrf    TRISB                   ; set all PORTB as output
                bsf             TRISB,1                 ; set DOUT as input
                movlw           B'11111111'
                movwf           TRISC                   ; Port C - all inputs
                clrf    TRISD                   ; Port D - all outputs
                bcf             STATUS,RP0
                clrf    Count
                clrf    Temp
                movwf           Tag1            ;default: Tag1 = '11111111'
finished
;
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

;
; main driver

                call    cycleLED

ModeSelect
        ;       bsf             PORTB,2                         ; SCNTRL HIGH (slck
high)
                bcf             PORTB,3                         ; green led off
                bcf             PORTB,4                         ; red led off
                btfsc   PORTC,1                         ; check config
                goto    cMode                           ;if config high
                goto    initComm                ;if config low

;******
cMode
                btfss   PORTC,1
                goto    initComm
```

```
        btfss   PORTC,0                 ;check green button
        goto    cMode                   ;if low cycle
        call    SwitchDelay             ;debounce

        call    getTagID
        movfw   TagID
        movwf   Tag1
        bcf     PORTB,4                 ;red LED off

        movf    Tag1,W                  ; move TagID to W
        movwf   PORTD                   ; display on LEDs

        bsf     PORTB,3
        call    tDelay
        bcf     PORTB,3
        call    tDelay
        bsf     PORTB,3
        call    tDelay
        bcf     PORTB,3
        call    tDelay
        bsf     PORTB,3
        call    tDelay
        bcf     PORTB,3
        call    tDelay
        goto    cMode

IDreject
        bcf     PORTB,3                 ;green LED off
        bsf     PORTB,4                 ;red LED on
initComm
; first, flash LEDs on/off twice to indicate initComm start
        btfsc   PORTC,1
        goto    cMode

        btfss   PORTC,0                 ;check green button
        goto    initComm                ;if low, cycle
        call    SwitchDelay             ;debounce
        call    getTagID
        movfw   Tag1
        subwf   TagID,F
        incf    TagID,F
        decfsz  TagID,F
```

```
        goto    IDreject                ;ID rejected
        bcf             PORTB,4                 ;red LED off
        bsf             PORTB,3                 ;green LED on
        goto    initComm

;;;;;;;

getTagID
        movlw           B'11111111'
        movwf           Cycle1
;       movlw           9Bh
;       movwf           Cycle2
        bcf             State,1         ; clear tagFound bit

        movlw           B'11111111'
        movwf           PORTD
        call    tDelay
        movlw           B'00000000'
        movwf           PORTD
        call    tDelay
        movlw           B'11111111'
        movwf           PORTD
        call    tDelay
        movlw           B'00000000'
        movwf           PORTD

seq2
    ;;SYNCHRONIZE
Hscroll
        btfsc   PORTB,1
        goto    Hscroll
Lscroll
        btfss   PORTB,1
        goto    Lscroll

        movlw           D'64'
        movwf           Count2
    ;;move forward a half-period (manchester)
        call    hDelay
        call    grabByte
;       goto    check4header
;       goto    readTag                 ;DIAGNOSTIC!
        goto    diag1
```

```
;;;;;;;
nextBit
        decfsz Count2
        goto   ModeSelect
        movf   Temp,TagID
        rlf         TagID,F
        call   c2Delay
        bcf         TagID,0
        btfsc  PORTB,1              ; if DIN low, skip next
        bsf         TagID,0
        call   c2Delay
check4header
        movlw       b'11001110'     ;HEADER
        movf   TagID,Temp
        subwf  TagID,F
        incf   TagID,F
        decfsz TagID,F
        goto   nextBit
        call   c2Delay
;;;;;;;
;Scroll until byte is all zeroes:
diagNB
        rlf         TagID,F
        call   c2Delay
        bcf         TagID,0
        btfsc  PORTB,1
        bsf         TagID,0
diag1
        incf   TagID,F
        decfsz TagID,F
        goto   diagNB
        goto   diag2
;;find first high:
diagNB2
        rlf         TagID,F
        call   c2Delay
        bcf         TagID,0
        btfsc  PORTB,1
        bsf         TagID,0
diag2
        btfss  TagID,0
        goto   diagNB2
```

```
;;next 7:
        call    grabByte            ;DIAGNOSTIC
        call    c2Delay
        call    grabByte
        goto    dispID              ;DIAGNOSTIC
diag3
        movlw       D'7'
        movwf       Count3
diagNB3
        rlf         TagID,F
        call    c2Delay
        bcf         TagID,0
        btfsc   PORTB,1
        bsf         TagID,0
        decfsz Count3
        goto    diagNB3
        goto    dispID
;;;;;;

;256 cycles <==> 1/(125000/32)

readTag
        call    grabByte
        call    cDelay

;checkID
    ;   incfsz  TagID,W                 ; increment TagID
    ;   bsf         State,1             ; if TagID was not all high, set
tagFound
    ;   btfsc   State,1             ; if tagFound bit cleared, loop
    ;   goto    dispID              ; else display ID
    ;   decfsz Cycle1, F
    ;   goto    seq2

dispID
        movlw       B'10101010'
        movwf       PORTD
        call    tDelay
        movlw       B'01010101'
        movwf       PORTD
        call    tDelay
        movlw       B'10101010'
        movwf       PORTD
```

```
        call    tDelay
        movlw       B'01010101'
        movwf       PORTD

        movf    TagID,W                 ; move TagID to W
        movwf       PORTD                       ; display on LEDs
        return

;;;;;;;;;;;;;;;;;;;;

cycleLED
        movlw       B'00000001'
        movwf       PORTD
        call    tDelay
        movlw       B'00000010'
        movwf       PORTD
        call    tDelay
        movlw       B'00000100'
        movwf       PORTD
        call    tDelay
        movlw       B'00001000'
        movwf       PORTD
        call    tDelay
        movlw       B'00010000'
        movwf       PORTD
        call    tDelay
        movlw       B'00100000'
        movwf       PORTD
        call    tDelay
        movlw       B'01000000'
        movwf       PORTD
        call    tDelay
        movlw       B'10000000'
        movwf       PORTD
        call    tDelay
        movlw       B'00000000'
        movwf       PORTD
        call    tDelay
        return

; debounce switch:
SwitchDelay
        movlw       D'20'
```

```
            movwf       Temp
delay
            decfsz Temp,F                       ; 60 usec delay loop
            goto    delay
            return

;~tenth-second delay:
tDelay
            movlw       01h
            movwf       Cycle1
            movlw       98h
            movwf       Cycle2
tloop
            decfsz Cycle1, F
            goto    tloop
            decfsz Cycle2, F
            goto    tloop
            return

;~255 cycles
cDelay
            movlw       D'84'
            movwf       Cycle3
cloop
            decfsz Cycle3, F
            goto    cloop
            return

c2Delay
            movlw       D'81'
            movwf       Cycle3
c2loop
            decfsz Cycle3, F
            goto    c2loop
            return

c3Delay
            movlw       D'83'
            movwf       Cycle3
c3loop
            decfsz Cycle3, F
            goto    c3loop
            return
```

```
;half a period
hDelay
        movlw       D'41'
        movwf       Cycle3
hloop
        decfsz Cycle3,F
        goto    hloop
        return

grabByte
        clrf    TagID               ; clear TagID
        btfsc   PORTB,1                 ; if DIN low, skip next
        bsf             TagID,7
        call    cDelay
        btfsc   PORTB,1                 ; if DIN low, skip next
        bsf             TagID,6
        call    cDelay
        btfsc   PORTB,1                 ; if DIN low, skip next
        bsf             TagID,5
        call    cDelay
        btfsc   PORTB,1                 ; if DIN low, skip next
        bsf             TagID,4
        call    cDelay
        btfsc   PORTB,1                 ; if DIN low, skip next
        bsf             TagID,3
        call    cDelay
        btfsc   PORTB,1                 ; if DIN low, skip next
        bsf             TagID,2
        call    cDelay
        btfsc   PORTB,1                 ; if DIN low, skip next
        bsf             TagID,1
        call    cDelay
        btfsc   PORTB,1                 ; if DIN low, skip next
        bsf             TagID,0
        return

;;;;;
Fault
        bsf             PORTB,3                 ;green LED
        bsf             PORTB,4                 ;red LED
        goto    Fault
```

```
;****
isrService
        goto    isrService

        END
```