Final Project: Security System

EE 371

Jake A. Cole

12/16/2013

Table of Contents:

Final Project Code

```
/*
Jake A. Cole
Final Project: Security System
12/16/2013

The Security System project is a building security system
that covers three zones, each controlled by a switch.
The project displays the status of the system, a timer
that allows the user to leave the building in 10 seconds,
and uses LEDs to show if the system is armed or
if the security has been breached.
*/

void convert(int n, char s[]);
void zonecheck(void);
void interrupt (void);

int counter = 0, dflag = 0, breachv = 0, vclear = 0, zone1 = 0, zone2 = 0, zone3 = 0;
char timer[10], string[4], breachs[15];

void main () {
  TMR1H = 0xF6;  //Load Timer for a hundredth of a second interrupts
  TMR1L = 0x3B;

  PIR1.TMR1IF = 0;      //Clear Timer overflow flag

  T1CON.TMR1ON = 1;     //Enable Timer
  T1CON.T1CKPS1 = 1;    //Set Prescaler
  T1CON.T1CKPS0 = 1;    //to 1:8
  T1CON.T1OSCEN = 0;    //Turn off oscillator

  PIE1.TMR1IE = 1;      //Enable timer overflow interrupts

  ANSEL  = 0x00;        //Clear analog input register
  ANSELH = 0x00;

  CM1CON0 = 0x00;       //Clear the comparators
  CM2CON0 = 0x00;

  IOCB = 0x00;          //Ensures mismatch interrupts are not enabled
  TRISB = 0x00;         //Sets TRISB as an output
  TRISC = 0xFF;         //Sets TRISC as an input
  TRISD = 0x00;         //Sets TRISD as an output

  PORTD = 0x00;         //All Leds are set off in PORTD

  INTCON.GIE = 1;       //Global interrupt enabled
  INTCON.PEIE = 1;      //Peripheral interrupt enabled

  Lcd_Config(&PORTB, 4, 5, 6, 3, 2, 1, 0); //Configures the LCD screen to PORTB
  LCD_Cmd(LCD_CURSOR_OFF);
  LCD_Cmd(LCD_CLEAR);

  while (1) {     //infinite loop
```

```
if(zone1 == 0 && zone2 == 0 && zone3 == 0 && dflag == 0) {  //No zones are armed
  Lcd_Out(1, 1, "No Zones Armed ");
}

else if(zone1 == 1 && zone2 == 1 && zone3 == 1 && dflag == 0) {  //All zones are armed
  Lcd_Out(1, 1, "Ready to Arm   ");
  if (PORTC.F4 == 1) { //RC4 Finalizes the arming of all three zones
    LCD_Cmd(LCD_CLEAR);
    counter = 1000;  //Starts the timer
    dflag = 1;
   }
}

else if(dflag == 0) {
  zonecheck();
  Lcd_Out(1, 1, "Zones Armed:   ");     //A combination of zones other than all three are armed
  Lcd_Out(2, 1, string);
}

if (dflag == 1 || dflag == 2) {
  Lcd_Out(1, 1, "System Armed   ");  //Arms the system, allows the user to exit the zones in ten seconds
  if (dflag == 1) Lcd_Out(2, 1, timer);

  if(counter == 0 && dflag == 1) {
     LCD_Cmd(LCD_CLEAR);
     PORTD = 0x01; //Sets RD0 on signifying that the system is armed
     dflag = 2;
     zone1 = 1;
     zone2 = 1;
     zone3 = 1;
  }

}

if(breachv == 1) {    //Sets the alarm off if the system is not disarmed

  if (counter > 0) {       //If the counter is above 0 the user still has time to disarm
     Lcd_Out(2, 1, timer);
     breachs[15] = '\0';
     breachs[14] = ' ';
     breachs[13] = ' ';
     breachs[12] = 'm';
     breachs[11] = 'e';
     breachs[10] = 't';
     breachs[9] = 's';
     breachs[8] = 'y';
     breachs[7] = 'S';
     breachs[6] = ' ';
     breachs[5] = 'm';
     breachs[4] = 'r';
     breachs[3] = 'a';
     breachs[2] = 's';
     breachs[1] = 'i';
     breachs[0] = 'D';
     dflag = 3;
  }
```

```
  else {              //The disarm time has run out
    if (vclear == 0) LCD_Cmd(LCD_CLEAR);
    vclear = 1;
    breachs[15] = '\0';
    breachs[14] = 'h';
    breachs[13] = 'c';
    breachs[12] = 'a';
    breachs[11] = 'e';
    breachs[10] = 'r';
    breachs[9] = 'B';
    breachs[8] = ' ';
    breachs[7] = 'y';
    breachs[6] = 't';
    breachs[5] = 'i';
    breachs[4] = 'r';
    breachs[3] = 'u';
    breachs[2] = 'c';
    breachs[1] = 'e';
    breachs[0] = 'S';
    PORTD = 0x02; //Sets the second Led on showing the user that the security is breached
    dflag = 3;
  }
  Lcd_Out(1, 1, breachs);
}

while (PORTC.F7 == 1) {     //RC7 Controls zone 1
    LCD_Cmd(LCD_CLEAR);
    if (PORTC.F7 == 0) {
      if(zone1 == 1) {
        zone1 = 0;
        if(dflag == 2) {
          breachv = 1;
          counter = 1000;
        }
      }
      else {
        zone1 = 1;
      }
    }
}

while (PORTC.F6 == 1) {    //RC6 Controls zone 2
    LCD_Cmd(LCD_CLEAR);
    if (PORTC.F6 == 0) {
      if(zone2 == 1) {
        zone2 = 0;
        if(dflag == 2) {
          breachv = 1;
          counter = 1000;
        }
      }
      else {
        zone2 = 1;
      }
    }
```

```
    }

    while (PORTC.F5 == 1) {   //RC5 Controls zone 3
        LCD_Cmd(LCD_CLEAR);
        if (PORTC.F5 == 0) {
          if(zone3 == 1) {
            zone3 = 0;
            if(dflag == 2) {
              breachv = 1;
              counter = 1000;
            }
          }
          else {
            zone3 = 1;
          }
        }
    }

    if (PORTC.F3 == 1) {    //RC3 Resets the alarm, would be the alarm code
        LCD_Cmd(LCD_CLEAR);
        zone1 = 0;
        zone2 = 0;
        zone3 = 0;
        dflag = 0;
        breachv = 0;
        vclear = 0;
        PORTD = 0x00;
    }
  }
}

/*
zonecheck creates the combinations of zones armed to be displayed on the LCD
*/
void zonecheck(void) {
  string[3] = '\0';
  string[2] = ' ';
  string[1] = ' ';
  string[0] = ' ';

  if(zone1 == 0 && zone2 == 0 && zone3 == 1) string[0] = '3';
  if(zone1 == 0 && zone2 == 1 && zone3 == 0) string[0] = '2';
  if(zone1 == 1 && zone2 == 0 && zone3 == 0) string[0] = '1';

  if(zone1 == 0 && zone2 == 1 && zone3 == 1) {
    string[2] = '3';
    string[1] = '&';
    string[0] = '2';
  }

  if(zone1 == 1 && zone2 == 0 && zone3 == 1) {
    string[2] = '3';
    string[1] = '&';
    string[0] = '1';
  }
```

```
  if(zone1 == 1 && zone2 == 1 && zone3 == 0) {
    string[2] = '2';
    string[1] = '&';
    string[0] = '1';
  }
  return;
}

/*
Interrupt reloads the timer when it overflows and
increments the counter every hundredth of a second
*/
void interrupt (void) {
  if (PIR1.TMR1IF == 0) { //Returns if the timer hasnt overflown
    dflag = 0;  //Flags that the board doesn't need to display the Clock
    return;
  }

  TMR1H = 0xF6;  //Load Timer
  TMR1L = 0x3B;
  PIR1.TMR1IF = 0;      //Clear Timer overflow flag

  convert(counter, timer);
  counter--;   //Decrements the counter

  if (counter < 0) {
    counter = 0;   //stops the counter if it hits 0
  }
  return;
}

/*
Convert converts the timer to a string so it can be displayed on the LCD
*/
void convert(int n, char s[]) {
  s[9] = '\0';
  s[8] = n % 10 + '0';
  n /= 10;
  s[7] = n % 10 + '0';
  n /= 10;
  s[6] = '.';
  s[5] = n % 10 + '0';
  n /= 10;
  s[4] = n % 10 + '0';
  s[3] = ' ';
  s[2] = ':';
  s[1] = 'n';
  s[0] = 'i';
  return;
}
```

Description of the Project:

The Security System project is a security system that monitors three zones of a building. The three zones are represented by three switches RC7, RC6, and RC5. If all switches are all set to be armed, there is a master switch RC4 that arms the entire building. Once this switch is pressed, the LCD displays a timer of 10 seconds that allows the user to exit the building without setting off the alarm. A LED RD0 is then turned on to show that the system is armed. If a zone is breached, it allows the user 10 seconds to reset the alarm. If the alarm is not reset, the security is breached and a LED RD1 is turned on until the user resets the system. There is a reset switch RC3 which represents the code to turn off the alarm.

Materials used:

Pic16F887 Board, USB Link Cable, Mikro C compiler

Solving the Problem:

To solve this problem I made sure to initialize the Pic16F887 board so that it could utilize Timer1 for the interrupt service routine. I also initialized PORTB for the LCD screen output, PORTC for input switches, and PORTD for LED outputs.

I used the switches RC7, RC6, and RC5 to represent zone one, two and three respectively. I also used the switch RC4 as the master arm switch to start the arming timer. The switch RC3 is used to represent the code that would reset the system. I also used the LCD to display the status of the security system. The timer is used to decrement a counter to represent a 10 second countdown. I used the LED RD0 to turn on if the system is on, and the LED RD1 to turn on if the security has been breached.

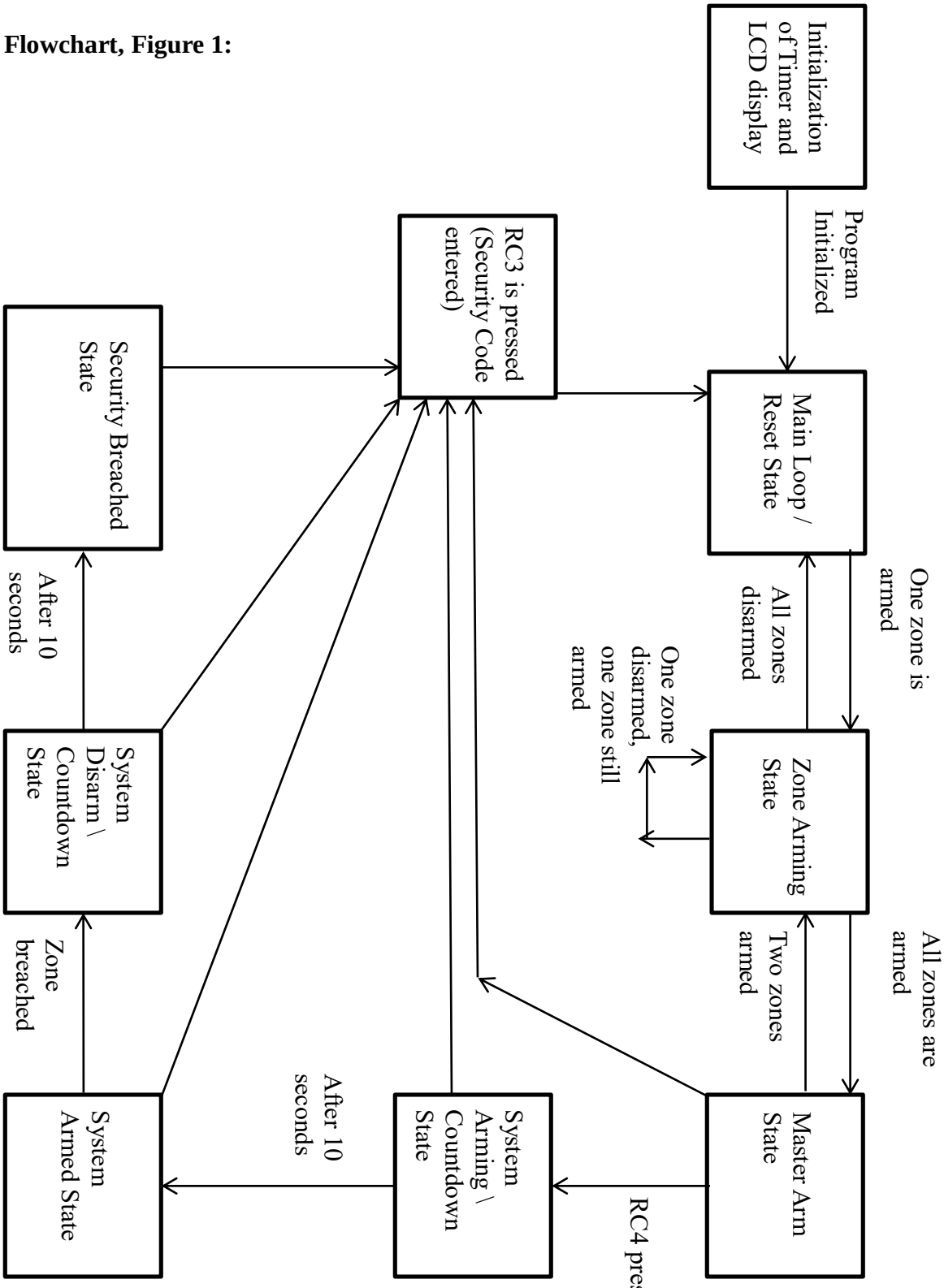
Explanation of the Modules:

The main function is used to initialize both the LCD display and Timer1. The main function also serves as a main loop that controls the entire security system.

The interrupt function is used to generate .01 second interrupts so that the security system can implement a ten second countdown to leave the building

The convert and zone check functions are used to format strings that are to be displayed on the LCD.

Flowchart, Figure 1:



Discussion of Results:

The Security System project is working as intended. It utilizes the interrupt service routine and a timer for a ten second countdown. The push button RC7 represents zone 1, the push button RC6 represents zone 2, the push button RC5 represents zone3, the push button RC4 represents the master arming switch, and the push button RC3 represents the master code or reset code. The LED RD0 is turned on when the system is on and the LED RD1 is turned on when the security is breached.

Conclusion:

After completing the Security System project I have learned how to apply the usage of a timer to create a 10 second countdown clock. I have become more proficient with the LCD display and the push buttons on the Pic16F887 board. I have learned how to design and implement a security system by using simple push switches, LEDs and a LCD screen. This project has helped to improve my methods of designing and mapping out flow charts. I also learned that when a program has to work in real time, the program has to be as efficient as possible to ensure that it can fulfill its purpose.

References:

Karimi, Bijan. "Embedded Systems Course Manual."
www.newhaven.edu. University of New Haven. Web.
4 Nov 2013.

"Pic16F887 Data Sheet."
http://www.microchip.com/. Microchip Technology Inc., n.d. Web.
4 Nov 2013.