**TABLE OF CONTENTS**

I. PROJECT DESCRIPTION........................................................................................... 3

II. HOME SECURITY SYSTEM...................................................................................... 4

III. FLOW CHART............................................................................................................. 5

IV. PROGRAM................................................................................................................... 6

V. COMPONENT……………………………………………………………………… 12

V.I. Microcontroller PIC 16F887........................................................................... 12

V.I.I. PORTC................................................................................................ 12

V.I.II. PORTB................................................................................................ 13

V.I.II.I. Interrupt on change................................................................ 13

V.I.II.II. INTCON Register................................................................. 13

V.I.II.III. ANSELH Register............................................................... 13

V.I.II.IV. IOCB Register…................................................................. 13

V.II. LCD……………………………………......................................................... 14

VI. DISCUSSION………………..................................................................................... 15

VII. CONCLUSION........................................................................................................... 16

VIII. REFERENCES............................................................................................................ 17

**LIST OF FIGURES**

Fig I: Block Diagram.................................................................................................. 3

Fig II: Circuit Diagram................................................................................................. 4

Fig III: Flow Chart......................................................................................................... 5

Fig V.I: PIC 16F887...................................................................................................... 12

Fig V.II: PORTC Registers............................................................................................ 12

Fig V.III: Registers Associated with Interrupt on change............................................... 14

Fig V.IV: LCD................................................................................................................. 14

**I. PROJECT DESCRIPTION**

Home security is one of the prime concerns in our life. A secure home provides peace to our mind both when we are at home and when we are away from home. So we designed a model of a simple home security system to keep the home safe from any intrusion or theft.

This project is based on a PIC16F887 microcontroller, switches and LCD. Microcontroller acts as a brain of the system. It takes input from the switches and display massages on the LCD accordingly. Switches are used to indicate inputs from different sensors. Block diagram of this project is shown in Fig I.I. We used the software mikroC and proteus.

PIC 16F887

LCD

Input from switches

Fig I: Block Diagram

**II. HOME SECURITY SYSTEM**

The house has three zones. Each zone is represented by a switch/pushbutton in our design. When a switch/pushbutton is pressed, it means that that zone is ready to be armed. If no zone is armed the message “No Zone Armed” is displayed on the LCD display. As each zone is armed or disarmed a message indicating which zones are armed is displayed on the LCD display. For example “Zone 1 Armed”, “Zones 1 & 3 Armed”, “Zones 2 & 3 Armed” etc. is displayed on the LCD display. When all three zones are armed the message “Ready To Arm” is displayed on the LCD display.

When all zones are armed a fourth switch/pushbutton is used to arm the entire system. When it is pressed the message “System Armed in 10 Sec” is displayed on LCD. Then the countdown begins and there is a period of 10 seconds in which any of the zones can be disarmed and armed again. This time is required to allow the person in the house to leave the house without the system detecting it as a security breach.

After the 10-second period the LCD displays the message “System Armed” and an LED is turned on indicating that the system is armed. If after expiration of the 10 seconds any zone is disarmed it is considered as a security breach and the LED for security breach lights up and the message “Security breach” is displayed on the LCD display until the system is reset.

After a successful system arming if any zone is disarmed. It causes a countdown of 10 seconds to appear on the LCD display indicating the period that the person entering the house can safely disarm the system. If the person releases the push button (used to arm the system) then system goes back to step 1. If after 10 seconds the system is not safely disarmed the message “Breach” is displayed on the LCD display and an LED is turned on indicating the beach in the security. The circuit diagram is shown in Fig II.I

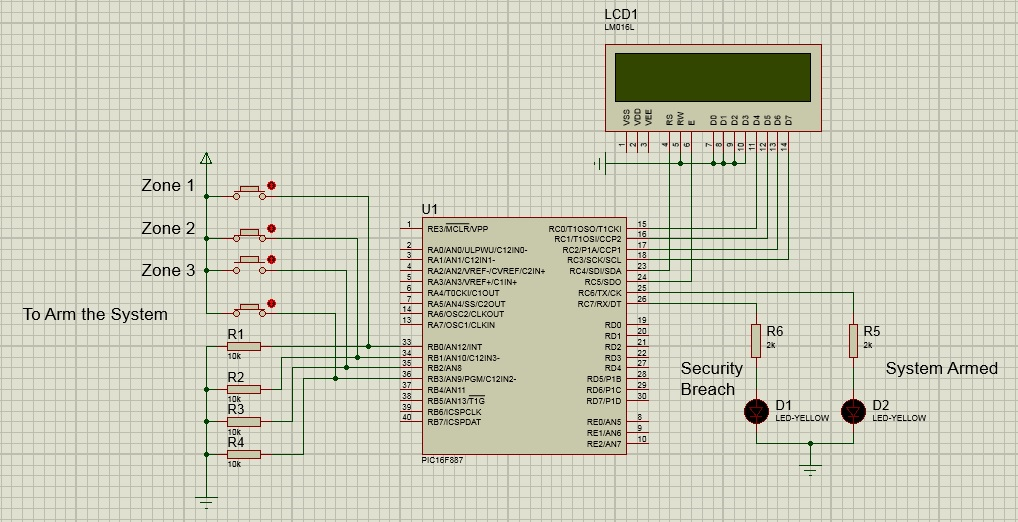


Fig II: Circuit Diagram

**III. FLOW CHART**



Fig III: Flow Chart

**IV. PROGRAM**

sbit LCD\_RS at RC4\_bit;

sbit LCD\_EN at RC5\_bit;

sbit LCD\_D4 at RC0\_bit;

sbit LCD\_D5 at RC1\_bit;

sbit LCD\_D6 at RC2\_bit;

sbit LCD\_D7 at RC3\_bit;

sbit LCD\_RS\_Direction at TRISC4\_bit;

sbit LCD\_EN\_Direction at TRISC5\_bit;

sbit LCD\_D4\_Direction at TRISC0\_bit;

sbit LCD\_D5\_Direction at TRISC1\_bit;

sbit LCD\_D6\_Direction at TRISC2\_bit;

sbit LCD\_D7\_Direction at TRISC3\_bit;

// LCD module connections

#define ARMED PORTC.RC6

#define BREACH PORTC.RC7

unsigned char time,t1,q1,w1,e1,i,a=1;

void system(void);

void RBINT\_ISR(void);

void interrupt(void)

{

if(INTCON.RBIF==1)

{

RBINT\_ISR();

}}

void RBINT\_ISR() // Interrupt service routine for PORTB interrupt on change

{

i=PORTB; // For every interrupt save the value of PORTB in variable "i"

INTCON.RBIF=0; // Must clear PORTB interrupt on change flag

}

char txt1[] = "No Zone Armed";

char txt2[] = "Zone 1 Armd";

char txt3[] = "Zone 2 Armd";

char txt4[] = "Zone 3 Armd";

char txt5[] = "Zone 1&2 Armd";

char txt6[] = "Zone 1&3 Armd";

char txt7[] = "Zone 2&3 Armd";

char txt8[] = "Ready to Arm";

char txt9[] = "System Armed in";

char txt10[] = " Sec";

char txt11[] = "System Armed";

char txt12[] = "Breach";

void main()

{

char b=0;

time=10;

ANSELH=0x00; // Enable PORTB for Digital input

TRISC = 0; // PORTC as output

TRISB = 0xFF; // PORTB as input

PORTB = 0xFF; // Initial value of PORTB on Reset

IOCB=0xFF; // Enable the bits of PORTB for interrupt

OPTION\_REG=0x80;

INTCON=0xC8; // PORTB Interrupt on change enabled

Lcd\_Init(); // Initialize LCD, Built in function

Lcd\_Cmd(\_LCD\_CURSOR\_OFF); // Cursor off

i=PORTB; // Save PORTB value in variable

abc: // Label, used to jump back to the start

time=10;

b=1;

ARMED=0;

BREACH=0;

while(1)

{

while(time>0)

{

t1=time/10; //////////////////////////////////////////////////////

q1=time%10; //////////////////////////////////////////////////////

w1=t1%10;

e1=t1/10; ////// Conversion of decimal value of ///////////////

q1=q1|0x30; ////// countdown timer to Ascii to /////////////////

w1=w1|0x30; ////// display it on LCD ////////////////////////////

e1=e1|0x30; //////////////////////////////////////////////////////

if(i>8)

--time; // Decrease the timer value if i>8

system(); // Function to display the status of security on LCD according to input

Delay\_ms(999); // To produce a delay of 1 sec

Lcd\_Cmd(\_LCD\_CLEAR); // Clear display

Lcd\_Cmd(\_LCD\_CURSOR\_OFF); // Cursor off

}

if(i==15 && a>0) // Condition to check that system is armed or not

{

Lcd\_Out(1,1,txt11); // Write "System Armed" on LCD

ARMED=1; // LED to indicate that system is armed

b=0; // Controls the security breach loop

}

while(b) // start of security breach loop

{

if(i<15 && i>8)

{

Lcd\_Cmd(\_LCD\_CLEAR); // Clear display

Lcd\_Cmd(\_LCD\_CURSOR\_OFF); // Cursor off

Lcd\_Out(1,1,txt12); // Write "Breach" on LCD

Delay\_ms(1000);

BREACH=1; // LED to indicate that system is breached

}}

if(i<15) // Condition to check that system is breached after successful system arming or not

{ --a;

goto abc; // Jump back to the start of code to check the security breach after successful system arming

} }}

void system()

{switch(i) // Compare the value of i and display message accordigly

{ case 1:

Lcd\_Out(1,1,txt2); // Write "Zone 1 Armed" on LCD

time=10; // Reset the value of timer

a=1;

break;

case 2:

Lcd\_Out(1,1,txt3); // Write "Zone 2 Armed" on LCD

time=10; // Reset the value of timer

a=1;

break;

case 3:

Lcd\_Out(1,1,txt5); // Write "Zone 1&2 Armed" on LCD

time=10; // Reset the value of timer

a=1;

break;

case 4:

Lcd\_Out(1,1,txt4); // Write "Zone 3 Armed" on LCD

time=10; // Reset the value of timer

a=1;

break;

case 5:

Lcd\_Out(1,1,txt6); // Write "Zone 1&3 Armed" on LCD

time=10; // Reset the value of timer

a=1;

break;

case 6:

Lcd\_Out(1,1,txt7); // Write "Zone 2&3 Armed" on LCD

time=10; // Reset the value of timer

a=1;

break;

case 7:

Lcd\_Out(1,1,txt8); // Write "Ready to Arm" on LCD

time=10; // Reset the value of timer

a=1;

break;

case 9:

Lcd\_Out(1,1,txt2); // Write "Zone 1 Armed" on LCD, when the button to arm system is pressed

Lcd\_Chr(2, 4, w1); // Write the MSB ASCii value of timer on LCD

Lcd\_Chr(2, 5, q1); // Write the LSB ASCii value of timer on LCD

Lcd\_Out(2,6,txt10); // Write "Sec" on LCD

break;

case 10:

Lcd\_Out(1,1,txt3); // Write "Zone 2 Armed" on LCD, when the button to arm system is pressed

Lcd\_Chr(2, 4, w1); // Write the MSB ASCii value of timer on LCD

Lcd\_Chr(2, 5, q1); // Write the LSB ASCii value of timer on LCD

Lcd\_Out(2,6,txt10); // Write "Sec" on LCD

break;

case 11:

Lcd\_Out(1,1,txt5); // Write "Zone 1&2 Armed" on LCD, when the button to arm system is pressed

Lcd\_Chr(2, 4, w1); // Write the MSB ASCii value of timer on LCD

Lcd\_Chr(2, 5, q1); // Write the LSB ASCii value of timer on LCD

Lcd\_Out(2,6,txt10); // Write "Sec" on LCD

break;

case 12:

Lcd\_Out(1,1,txt4); // Write "Zone 3 Armed" on LCD, when the button to arm system is pressed

Lcd\_Chr(2, 4, w1); // Write the MSB ASCii value of timer on LCD

Lcd\_Chr(2, 5, q1); // Write the LSB ASCii value of timer on LCD

Lcd\_Out(2,6,txt10); // Write "Sec" on LCD

break;

case 13:

Lcd\_Out(1,1,txt6); // Write "Zone 1&3 Armed" on LCD, when the button to arm system is pressed

Lcd\_Chr(2, 4, w1); // Write the MSB ASCii value of timer on LCD

Lcd\_Chr(2, 5, q1); // Write the LSB ASCii value of timer on LCD

Lcd\_Out(2,6,txt10); // Write "Sec" on LCD

break;

case 14:

Lcd\_Out(1,1,txt7); // Write "Zone 2&3 Armed" on LCD, when the button to arm system is pressed

Lcd\_Chr(2, 4, w1); // Write the MSB ASCii value of timer on LCD

Lcd\_Chr(2, 5, q1); // Write the LSB ASCii value of timer on LCD

Lcd\_Out(2,6,txt10); // Write "Sec" on LCD

break;

case 15:

Lcd\_Out(1,1,txt9); // Write "System Armed in" on LCD

Lcd\_Chr(2, 4, w1); // Write the MSB ASCii value of timer on LCD

Lcd\_Chr(2, 5, q1); // Write the LSB ASCii value of timer on LCD

Lcd\_Out(2,6,txt10); // Write "Sec" on LCD

break;

default:

Lcd\_Out(1,1,txt1); // Write "No Zone is Armed"

a=1;

break; }}

**V. COMPONENTS**

**V.I. Microcontroller PIC 16F887**

PIC 16F887 is an 8-bit microcontroller. It has 4 input/output ports. We will only discuss key features related to our project. Pin diagram of PIC16F887 is shown in Fig V.I[01].

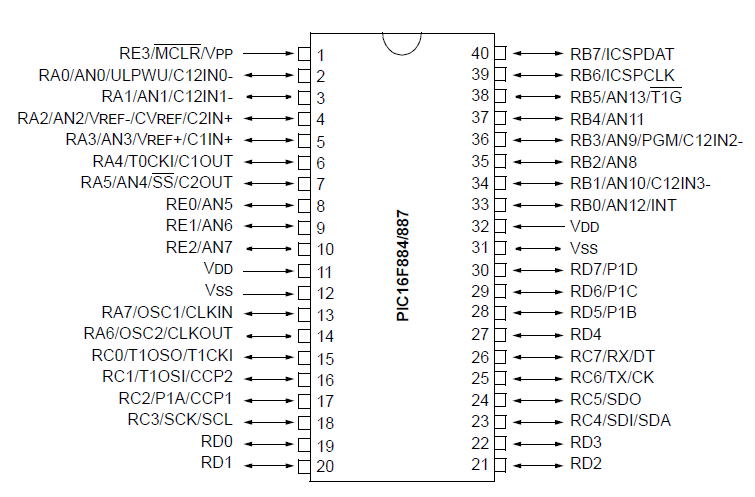
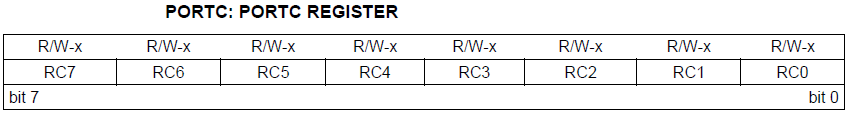


Fig V.I: PIC 16F887

**V.I.I. PORTC**

PORTC is a 8-bit wide, bidirectional port. The corresponding data direction register is TRISC. Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., put the corresponding output driver in a High Impedance mode). Clearing a TRISC bit (= 0) will make the corresponding PORTC pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 3-4 shows how to initialize PORTC. Registers associated with PORTC are shown in Fig V.II.

****

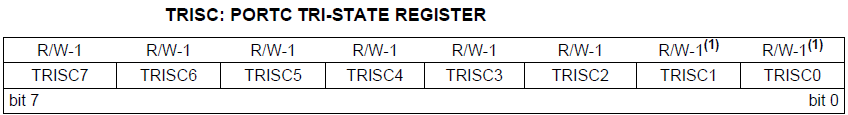
****

Fig V.II: PORTC Registers

**V.I.II. PORTB**

PORTB is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISB. Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e. enable the output driver and put the contents of the output latch on the selected pin).

**V.I.II.I. Interrupt-on-change**

All of the PORTB pins are individually configurable as an interrupt-on-change pin. Control bits IOCB<7:0> enable or disable the interrupt function for each pin. The interrupt-on-change feature is disabled on a Power-on Reset. For enabled interrupt-on-change pins, the present value is compared with the old value latched on the last read of PORTB to determine which bits have changed or mismatched the old value. The ‘mismatch’ outputs of the last read are OR’d together to set the PORTB Change Interrupt flag bit (RBIF) in the INTCON register. This interrupt can wake the device from Sleep. The user, in the Interrupt Service Routine, clears the interrupt by:

a) Any read or write of PORTB. This will end the mismatch condition.

b) Clear the flag bit RBIF.

A mismatch condition will continue to set flag bit RBIF. Reading or writing PORTB will end the mismatch condition and allow flag bit RBIF to be cleared. The latch holding the last read value is not affected by a MCLR or Brown-out Reset. After these Resets, the RBIF flag will continue to be set if a mismatch is present. Registers associated with PORTB interrupt on change are given in Fig V.III. We have to load corresponding values in the following registers to enable the PORTB interrupt on change.

**V.I.II.II. INTCON Register**

The INTCON register, shown in Fig V.III is a readable and writable register, which contains the various enable and flag bits for TMR0 register overflow, PORTB change and external INT pin interrupts. Each bit of INTCON register is explained below.

**V.I.II.III. ANSELH Register**

The ANSELH register is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELH bit high will cause all digital reads on the pin to be read as ‘0’ and allow analog functions on the pin to operate correctly. The state of the ANSELH bits has no affect on digital output functions. A pin with TRIS clear and ANSELH set will still operate as a digital output, but the Input mode will be analog. This can cause unexpected behavior when executing read-modify-write instructions on the affected port. ANSELH register is shown in the Fig V.III.

**V.I.II.IV. IOCB Register**

The IOCB register is used to enable PORTB bits, required for interrupt on change in PORTB. IOCB register is shown in Fig V.III.

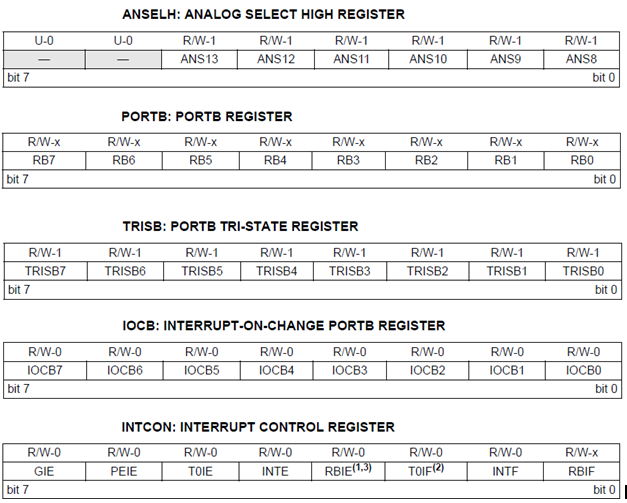


Fig V.III: Registers Associated with Interrupt on change

**V.II. LCD**

LCD used in this project is 2x16. It has 16 pins. Pin diagram of LCD is shown in Fig V.IV.[02]. It has 8 data pins. We used this LCD in 4-bit mode to engage minimum number of microcontroller pins as possible. For this purpose we connected the lower 4 bits D0-D3 to ground and send data at the higher bits D4-D7.

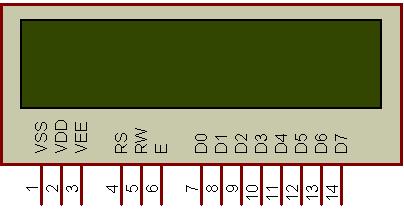


Fig V.IV: LCD

**VI. DISCUSSION**

This project is very simple model. It has many limitations due to which it cannot be employed in the practical life. This project is a building block of a well equipped home security system. It helped us to improve our programming and circuit design skills.

This design has a very serious drawback. The switch used to arm the system also helps in disarming the system. So if any person is aware of that switch then he could easily brake into your house without notifying when you are away. More over switches are not intelligent sensors, so they are easy to deceive. When you are away and someone gets into your house then it is impossible for you to get aware of intrusion with this design.

**VII. CONCLUSION**

This design of Home security system is just a model. In this design the switches are used to detect changes in the physical environment. In practical life switches or push buttons cannot provide smart information about the surroundings. So we can replace these switches with sensors to get smart information about the surroundings. We can use PIR motion sensors to detect any movement near the door. For more advance work we can place cameras and sensors inside and outside the house to avoid any theft. This project can be further expended by using GSM or internet modules to monitor information even when you are away from home.

**VIII. REFERENCES**

**[01]**https://www.google.com/search?biw=1332&bih=602&noj=1&tbm=isch&sa=1&q=pic+16f884%2F887&oq=pic+16f884%2F887&gs\_l=img.12...59957.78195.0.80097.9.9.0.0.0.0.715.2772.3-3j2j0j1.6.0....0...1c.1.32.img..8.1.499.njevk5m012A#facrc=\_&imgrc=hIlOwD9rKErcaM%3A%3Bpqof9jZdhBYKFM%3Bhttp%253A%252F%252Fcircuits.datasheetdir.com%252F19%252FPIC16F884-pinout.jpg%3Bhttp%253A%252F%252Fwww.datasheetdir.com%252FPIC16F884%252BPIC-Microcontrollers%3B570%3B356,Date 14 Dec,2013,Time 10:34 AM(PST).

**[02]**https://www.google.com/search?biw=1332&bih=602&noj=1&tbm=isch&sa=1&q=LCD+2X16&btnG=Submit#facrc=\_&imgdii=\_&imgrc=iZvI4t8m1BxLQM%3A%3BptGGvcjVvUcVM%3Bhttp%253A%252F%252F3.bp.blogspot.com%252FUfZLIAnwJPE%252FUR0q9Q2ZEkI%252FAAAAAAAAAHU%252FALPfG8ytEk%252Fs1600%252FPin%252Bdiag%252B16%252BX%252B2%252BLCD.jpg%3Bhttp%253A%22F%252Fwww.code2impress.com%252F2013%252F02%252Fdisplay-your-name-in-2x16lcdscreen.html%3B320%3B165,Date 15 Dec,2013,Time 9.00 PM(PST).