RV COLLEGE OF ENGINEERING®

BENGALURU – 560059

(Autonomous Institution Affiliated to VTU, Belagavi)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

"ELEVATOR SYSTEM"

a smart door opening and closing control system that can be employed in houses and offices for security purpose with the help of the controller LPC 2148.

MINI-PROJECT REPORT MICROCONTROLLERS AND EMBEDDED SYSTEMS IV SEMESTER

2020-21

Submitted by

PHALAKSHA CG	**1RV19CS112**
MOHAMED MOIN IRFAN	**1RV19CS089**

RV COLLEGE OF ENGINEERING®, BENGALURU - 560059

(Autonomous Institution Affiliated to VTU, Belagavi)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the **Mini-**project work titled "ELEVATOR SYSTEM" has been carried out by Phalaksha CG and Mohamed Moin Irfan bonafide students of RV College of Engineering, Bengaluru, have submitted in partial fulfillment for the **Assessment of Course: Microcontrollers and Embedded Systems** during the year 2020-2021. It is certified that all corrections/suggestions indicated for the internal assessment have been incorporated in the report.

Faculty Incharge
Department of CSE,
RVCE., Bengaluru –59

Head of Department
Department of CSE,
RVCE, Bengaluru–59

RV COLLEGE OF ENGINEERING®, BENGALURU - 560059
(Autonomous Institution Affiliated to VTU)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We, **Phalaksha CG <1RV19CS112> and Mohamed Moin Irfan <1RV19CS089> the students of 4th Semester B.E., Department of Computer Science and Engineering, RV College of Engineering, Bengaluru hereby declare that the Mini-Project titled "Elevator System" has been carried out by us and submitted in partial fulfillment for the Assessment of Course: Microcontrollers and Embedded Systems during the year 2020-2021.**

Place: Bengaluru **Phalaksha CG**
Date: 09.08.2021 **Mohamed Moin Irfan**

INTRODUCTION

Elevators are used in our day-to-day life, and every day we use elevators in many place and sometimes.An **elevator** or **lift** is a type of cable-assisted, hydraulic cylinder-assisted, or roller-track assisted machine that vertically transports people or freight between floors, levels, or decks of a building, vessel, or other structure. They are typically powered by electric motors that drive traction cables and counterweight systems such as a hoist, although some pump hydraulic fluid to raise a cylindrical piston like a jack.In agriculture and manufacturing, an elevator is any type of conveyor device used to lift materials in a continuous stream into bins. Some elevators can also travel horizontally in addition to the usual vertical motion.

It is important to have a proper functioning, so that when there is a error or emergency stop in lift it can continue functioning again.we have added calling function to the lift so that its better to contact ouside lift during an emergency.A fully functional elevator system using LPC2148,LCD display which shows the info,a 7 segment display to display the floor level and keyboard matrix for operating the lift or to contact someone during an emergency.

DESCRIPTION

We have done the basic structure of a elevator system that can be employed in houses, offices, etc with the help of the controller LPC 2148, keyboard matrix, an LCD, 5 seven segment displays.

In this project we have made a fully functional elevator system using LPC2148,LCD display which shows the info,a 7 segment display to display the floor level and keyboard matrix for operating the lift or to contact someone during an emergency.

CMOPUNENTS

1. LPC2148 Controller
2. Keyboard matrix
3. LCD display 32x4
4. 5 7 segment displays

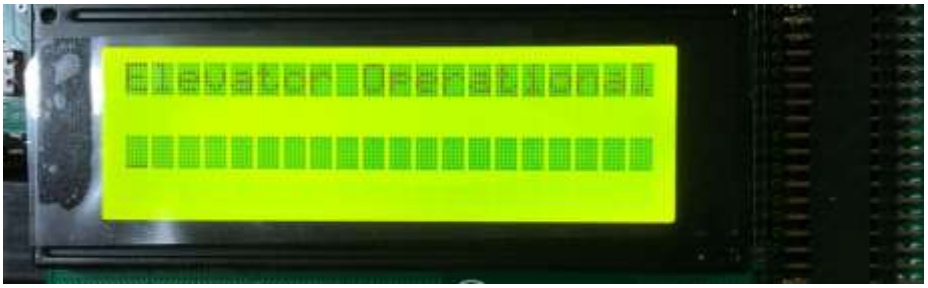
1. LPC2148 Controller

The LPC2148 microcontrollers are based on a 16-bit/32-bit ARM7TDMI-S CPU with real-time emulation and embedded trace support, that combine the microcontroller with embedded high-speed flash memory ranging from 32 kB to 512 kB. A 128-bit wide memory interface and a unique accelerator architecture enable 32-bit code execution at the maximum clock rate. For critical code size applications, the alternative 16-bit Thumb mode reduces code by more than 30 % with minimal performance penalty

Due to their tiny size and low power consumption, LPC2148 are ideal for applications where miniaturization is a key requirement, such as access control and point-of-sale. Serial communications interfaces ranging from a USB 2.0 Full-speed device, multiple UARTs, SPI, SSP to I2C-bus and on-chip SRAM of 8 kB up to 40 kB, make these devices very well suited for communication gateways and protocol converters, soft modems, voice recognition and low end imaging, providing both large buffer size and high processing power. Various 32-bit timers, single or dual 10-bit ADC(s), 10-bit DAC, PWM channels and 45 fast GPIO lines with up to nine edge or level sensitive external interrupt pins make these microcontrollers suitable for industrial control and medical systems

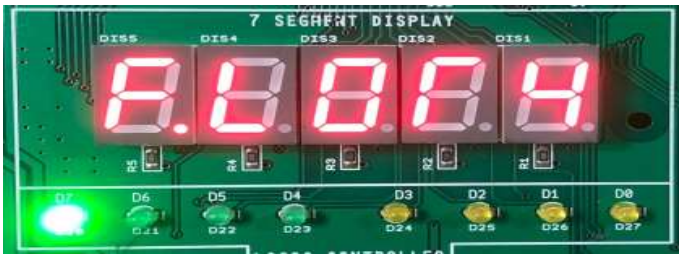
2.LCD Display 32x4

The PCA85162 is a peripheral device which interfaces to almost any Liquid Crystal Display (LCD) with low multiplex rates. It generates the drive signals for any static or multiplexed LCD containing up to four backplanes and up to 32 segments. It can be easily cascaded for larger LCD applications. The PCA85162 is compatible with most microcontrollers and communicates via the two-line bidirectional I²C-bus. Communication overheads are minimized by a display RAM with auto-incremented addressing, by hardware subaddressing, and by display memory switching (static and duplex drive modes).



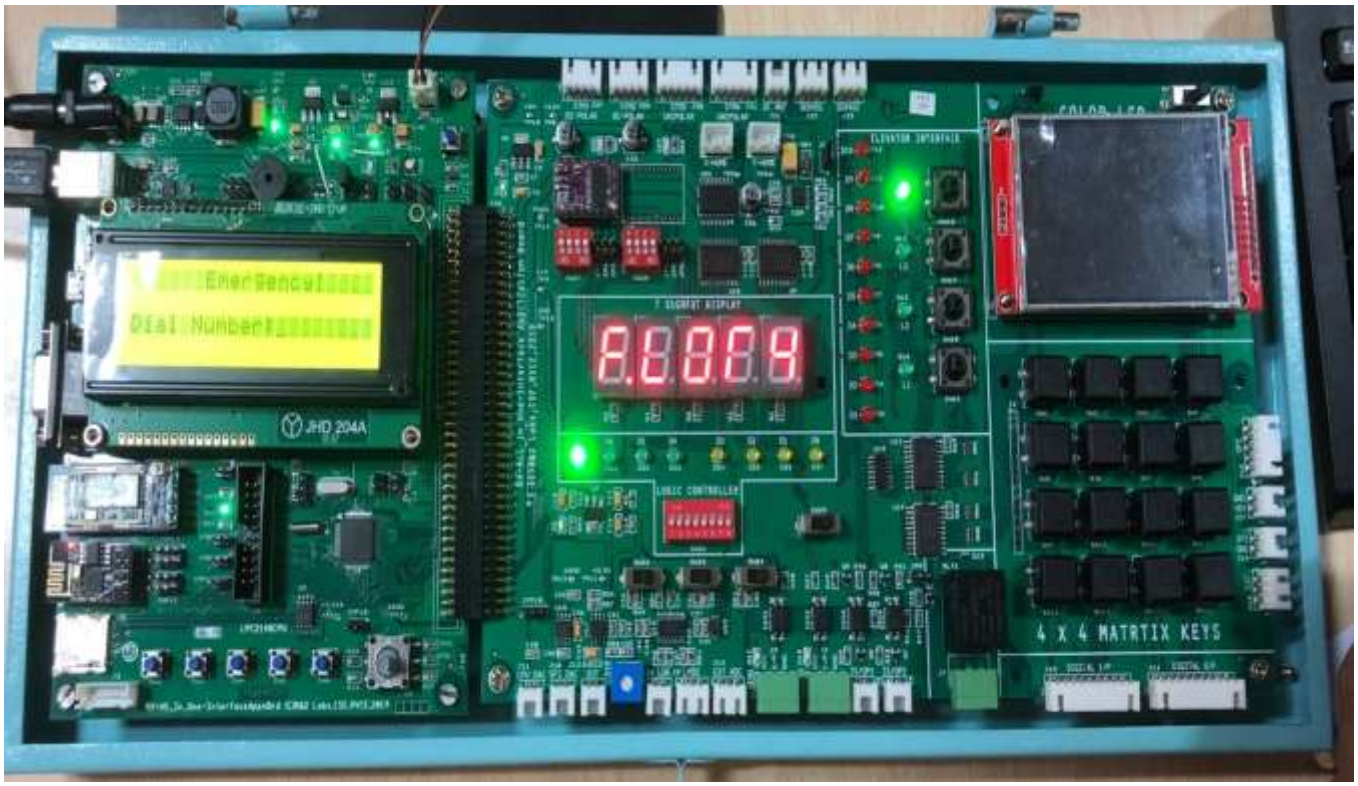
3.5x7 Segment Display

A seven-segment display is a form of electronic display device for displaying decimal numerals that is an alternative to the more complex dot matrix displays. Seven-segment displays are widely used in digital clocks, electronic meters, basic calculators, and other electronic devices that display numerical information.



4.keyboard matrix:

a matrix of 8 columns and 8 rows of wires, with a switch at every intersection. The keyboard controller scans the columns. If a key has been pressed, the controller scans the rows the controller determines the row-column combination at which a key has been pressed, and generates a note corresponding to that key.



IMPLEMENTATION DETAILS:

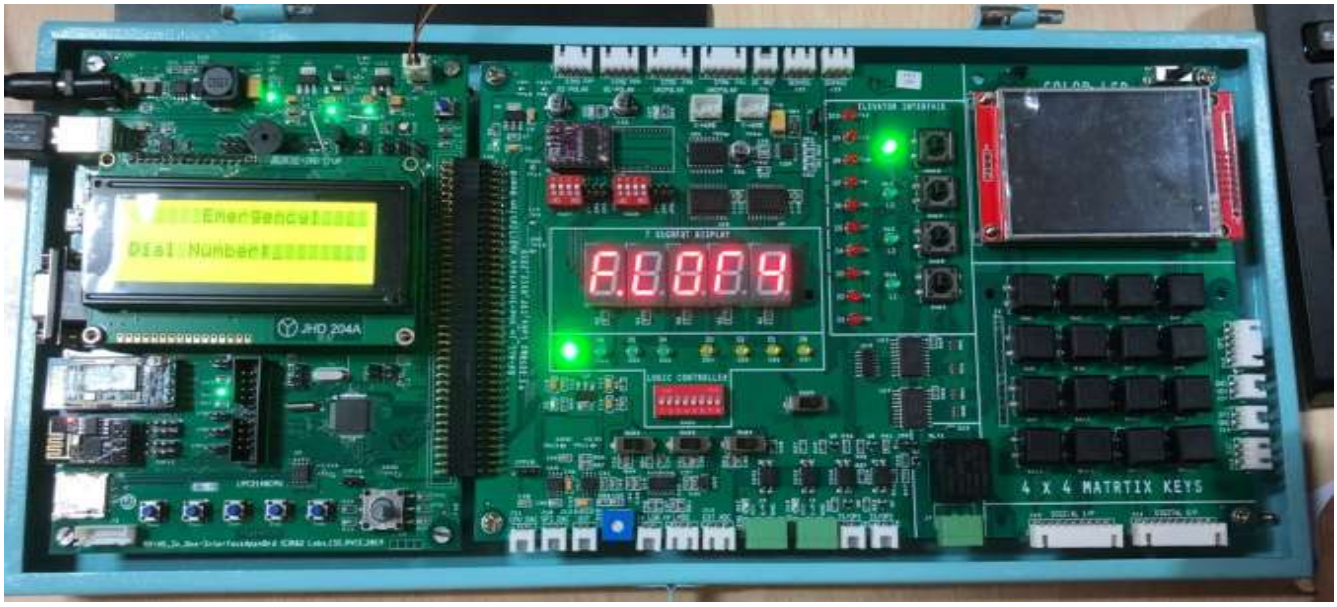
This project we have been implemented using with the help LPC 2148 controller and using martix keyboard we are going to take inputs for our elevator system.when a key is pressed it takes to the desired floor.for example, when key one is pressed it takes to floor one.the 7 segment display shows the floor which the lift has reached.in this project we have given an emergency alert after reaching 4th floor.so when the lift get stuck.the LCD displays an emergency and when the lift is running it shows operational,later we can use the keyboard matrix to call someone and the lift can br fixed and again can back up to be operational.

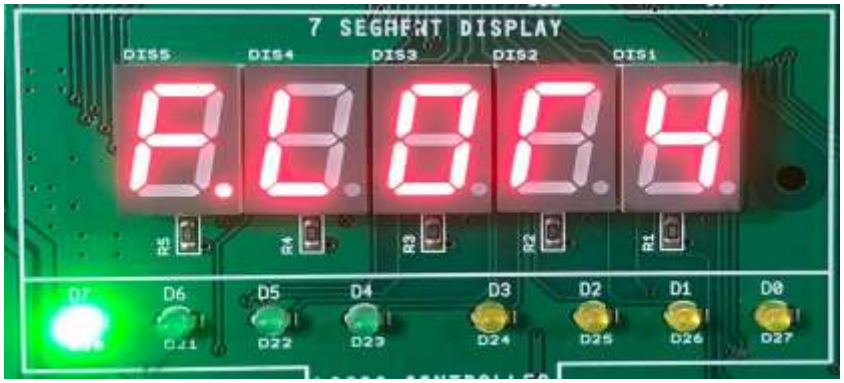
DEMO VIDEO DRIVE LINK:

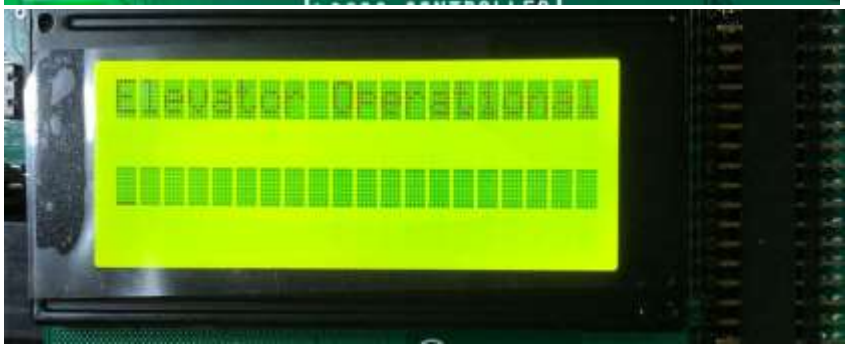
https://drive.google.com/drive/folders/1-DU4-zU5D1g2pQjUA3rmuFGU2UcyYMWn

WORKING IMAGES

Circuit Layout




7 SEGMENT DISPLAY
DIS5
DIS4
DIS3
DIS2
DIS1
D7
D6
D5
D4
D3
D2
D1
D0
D22
D23
D24
D25
D26
D27




JHD 204A


JHD 204A


JHD 204A
LPC2148CPU

SOURCE CODE

```
//Elevator Program:
// P0.16 - P0.19 are connected to decoder inputs, it makes one of the o/p LEDs 0 to 9 on
// P0.20-P0.23 are connected to *CLR pins of latches: make it '0' and then '1' to clear
// elevator keys: *Q outputs of latches connected to P1.16 TO P1.19

#include <lpc214x.h>
#include <stdio.h>
#define LED_OFF (IO0SET = 1U << 31)
#define LED_ON (IO0CLR = 1U << 31)
#define PLOCK 0x00000400
void alphadisp7SEG(char* buf);
void delay_ms(unsigned int j);
void elevator_run(void);
#define RS_ON  (IO0SET =  1U << 20)
#define RS_OFF (IO0CLR = 1U << 20)
#define EN_ON  (IO1SET = 1U << 25)
#define EN_OFF (IO1CLR = 1U << 25)
#define COL0 (IO1PIN & 1 <<19)
#define COL1 (IO1PIN & 1 <<18)
#define COL2 (IO1PIN & 1 <<17)
#define COL3 (IO1PIN & 1 <<16)
unsigned char rowsel=0,colsel=0;
void LCD_SendCmdSignals();
static void LCD_SendHigherNibble(unsigned char dataByte);
void LCD_SendDataSignals();
```

```
void keyboard();
static void delay_us(unsigned int count);
unsigned char lookup_table[4][4]={{'0', '1', '2','3'},
                                  {'4', '5', '6','7'},
                                  {'8', '9', 'a','b'},
                                  {'c', 'd', 'e','f'}};
static void LCD_CmdWrite( unsigned char cmdByte)
{
    LCD_SendHigherNibble(cmdByte);
    LCD_SendCmdSignals();
    cmdByte = cmdByte << 4;
    LCD_SendHigherNibble(cmdByte);
    LCD_SendCmdSignals();

}
static void LCD_DataWrite( unsigned char dataByte)
{
    LCD_SendHigherNibble(dataByte);
    LCD_SendDataSignals();
    dataByte = dataByte << 4;
    LCD_SendHigherNibble(dataByte);
    LCD_SendDataSignals();

}
static void LCD_Reset(void)
{
  /* LCD reset sequence for 4-bit mode*/
    LCD_SendHigherNibble(0x30);
```

```
    LCD_SendCmdSignals();
    delay_ms(100);
    LCD_SendHigherNibble(0x30);
    LCD_SendCmdSignals();
    delay_us(200);
    LCD_SendHigherNibble(0x30);
    LCD_SendCmdSignals();
    delay_us(200);
    LCD_SendHigherNibble(0x20);
    LCD_SendCmdSignals();
    delay_us(200);
}
static void LCD_SendHigherNibble(unsigned char dataByte)
{
//send the D7,6,5,D4(uppernibble) to P0.16 to P0.19
    IO0CLR = 0X000F0000;IO0SET = ((dataByte >>4) & 0x0f) << 16;
}
static void LCD_SendCmdSignals(void)
{
     RS_OFF;// RS - 1
     EN_ON;delay_us(100);EN_OFF;// EN - 1 then 0
}
static void LCD_SendDataSignals(void)
{
     RS_ON;// RS - 1
     EN_ON;delay_us(100);EN_OFF;// EN - 1 then 0
}
static void LCD_Init(void)
```

```
{
delay_ms(100);
LCD_Reset();
LCD_CmdWrite(0x28u);//Initialize the LCD for 4-bit 5x7 matrix type
LCD_CmdWrite(0x0Eu);// Display ON cursor ON
LCD_CmdWrite(0x01u);//Clear the LCD
LCD_CmdWrite(0x80u);//go to First line First Position
}
void LCD_DisplayString(const char *ptr_string)
{
// Loop through the string and display char by char
   while((*ptr_string)!=0)
      LCD_DataWrite(*ptr_string++);
}

static void delay_us(unsigned int count)
{
 unsigned int j=0,i=0;
 for(j=0;j<count;j++)
 {
  for(i=0;i<10;i++);
 }
}
int main()
{
 IO0DIR |= 1U << 31 | 0x00FF0000 ; // to set P0.16 to P0.23 as o/ps
 IO1DIR |= 1U << 25; // to set P1.25 as o/p used for EN
 IO0DIR |= 1U << 31 | 0XFF << 16  | 1U<<30 ; // to set P0.31 & P1.20 to P1.23 as outputs
```

```
 IO1DIR |= 1 << 24; // to set P1.24 as output
//keyboard();
LCD_Reset();
 LCD_Init();
 delay_ms(100);
LCD_Reset();LCD_CmdWrite(0x80); LCD_DisplayString("Elevator Operational");
 LED_ON; // make D7 Led on .. just indicate the program is running
 elevator_run();
 while(1);
}
void elevator_run(void)
{
int i,val;
unsigned int counter;
IO1CLR = 1 << 24;// enable elevator section in the application board : 0 to enable
IO0CLR = 0X000F0000;//to set the  elevator led for first floor
do{
IO0CLR = 0X00F00000;IO0SET = 0X00F00000; // clear all the latches *CLR
//waiting for floor key
do{
counter = (IO1PIN >> 16) & 0X0000000F ;// wait for any lift/elevator key press
}while(counter == 0x0F);
if(counter == 0x0e){      val=3;
alphadisp7SEG("flor1");
}
//1110 - floor 1 key pressed
else if(counter == 0x0d) {      val=6;
alphadisp7SEG("flor2");
```

```
}//1101 - floor 2 key pressed
else if(counter == 0x0b) {      val=8;
alphadisp7SEG("flor3");
}//1011 - floor 3 key pressed
else if(counter == 0x07) {      val=10;
alphadisp7SEG("flor4");
}//0111- floor 4 k
//elevator movement-UP
for(i=0;i<val;i++)
{
IO0CLR = 0X000F0000;IO0SET = i << 16;
delay_ms(1000);
}
//elevator movement-DN
for(i=val-1;i>=0;i--)
{
IO0CLR = 0X000F0000;IO0SET = i << 16;
delay_ms(1000);
}
if(counter==0x07)
keyboard();
}while(1);
}
void SystemInit(void)
{
  PLL0CON = 0x01;
  PLL0CFG = 0x24;
  PLL0FEED = 0xAA;
```

```
  PLL0FEED = 0x55;
  while( !( PLL0STAT & PLOCK ))
  {
;
}
  PLL0CON = 0x03;
  PLL0FEED = 0xAA; // lock the PLL registers after setting the required PLL
  PLL0FEED = 0x55;
}
void keyboard(){
SystemInit();
char  buf[10] ;
LCD_Reset();
LCD_CmdWrite(0x80); LCD_DisplayString("      Emergency!    ");
LCD_CmdWrite(0x94); LCD_DisplayString("Dial Number:");


//initialize UART0 port
IO0DIR |= 1U << 31 | 1U << 19 | 1U << 20 | 1U << 30 ; // to set as o/ps
 IO0DIR |= 1U << 31 | 0x00FF0000; // to set P0.16 to P0.23 as o/ps
int i=0;
//make D7 Led on off for testing
 LED_ON; delay_ms(500);LED_OFF;delay_ms(500);
 do
{
while(1){

//check for keypress in row0,make row0 '0',row1=row2=row3='1'
```

```
rowsel=0;IO0SET = 0X000F0000;IO0CLR = 1 << 16;
if(COL0==0){colsel=0;break;};if(COL1==0){colsel=1;break;};
    if(COL2==0){colsel=2;break;};if(COL3==0){colsel=3;break;};
 //check for keypress in row1,make row1 '0'
rowsel=1;IO0SET = 0X000F0000;IO0CLR = 1 << 17;
if(COL0==0){colsel=0;break;};if(COL1==0){colsel=1;break;};
    if(COL2==0){colsel=2;break;};if(COL3==0){colsel=3;break;};
//check for keypress in row2,make row2 '0'
rowsel=2;IO0SET = 0X000F0000;IO0CLR = 1 << 18;//make row2 '0'
if(COL0==0){colsel=0;break;};if(COL1==0){colsel=1;break;};
    if(COL2==0){colsel=2;break;};if(COL3==0){colsel=3;break;};
//check for keypress in row3,make row3 '0'
rowsel=3;IO0SET = 0X000F0000;IO0CLR = 1 << 19;//make row3 '0'
if(COL0==0){colsel=0;break;};if(COL1==0){colsel=1;break;};
    if(COL2==0){colsel=2;break;};if(COL3==0){colsel=3;break;};
 };
delay_ms(50);//allow for key debouncing
while(COL0==0 || COL1==0 || COL2==0 || COL3==0);//wait for key release
delay_ms(50);//allow for key debouncing
  IO0SET = 0X000F0000; //disable all the rows
//send to serial port(check on the terminal)
IO0SET = 0X0FF0000;//IO0CLR = lookup_table[rowsel][colsel] << 16;
  buf[i] = lookup_table[rowsel][colsel];
buf[i+1]=0;
//buf[i]=lookup_table[rowsel][colsel];
i++;
LCD_Reset();
LCD_CmdWrite(0x80); LCD_DisplayString("                    ");
```

```
LCD_CmdWrite(0x94); LCD_DisplayString("                    ");
LCD_Reset();LCD_CmdWrite(0x80); LCD_DisplayString("Dial : ");
LCD_CmdWrite(0x94); LCD_DisplayString(buf);
 }
 while(i<10);
 LCD_Reset();
LCD_CmdWrite(0x80); LCD_DisplayString("                    ");
LCD_CmdWrite(0x94); LCD_DisplayString("                    ");
 LCD_Reset();LCD_CmdWrite(0x80); LCD_DisplayString("Ringing");
 delay_ms(2000);

 LCD_Reset();LCD_CmdWrite(0x80); LCD_DisplayString("Ringing.");
 delay_ms(2000);

 LCD_Reset();LCD_CmdWrite(0x80); LCD_DisplayString("Ringing..");
 delay_ms(200);
 LCD_Reset();LCD_CmdWrite(0x80); LCD_DisplayString("Ringing...");
 delay_ms(500);
 LCD_Reset();LCD_CmdWrite(0x80); LCD_DisplayString("Elevator Operational");
 delay_ms(3000);
 }

void delay_ms(unsigned int j)
{
  unsigned int x,i;
for(i=0;i<j;i++)
{
    for(x=0; x<600; x++);    /* loop to generate 1 milisecond delay with Cclk = 60MHz */
```

```
}
}

unsigned char getAlphaCode(unsigned char alphachar)
{
switch (alphachar)
{
// dp g f e d c b a - common anode: 0 segment on, 1 segment off
case 'f': return 0x8e;
case 'i': return 0xf9;
case 'r': return 0xce;
case 'e': return 0x86;  // 1000 0110
case 'h': return 0x89;
case 'l': return 0xc7;
case 'p': return 0x8c;
case ' ': return 0xff;
case 'a' : return 0x88;
case 'g' : return 0x82;
case 'o' : return 0xc0;
case '1' : return 0xcf;
case '2' : return 0xa4;
case '3' : return 0xb0;
case '4' : return 0x99;//10011001
//simmilarly add for other digit/characters
default : return 0x99;
break;
}
return 0xff;
```

```
}

void alphadisp7SEG(char *buf)
{
unsigned char  i,j;
//char seg7_data;

unsigned char seg7_data,temp=0;
for(i=0;i<5;i++) // because only 5 seven segment digits are present
{
seg7_data = getAlphaCode(*(buf+i)); //instead of this look up table can be used
   //to shift the segment data(8bits)to the hardware (shift registers) using Data,Clock,Strobe
  for(j=0;j<8;j++)
{
//get one bit of data for serial sending
temp = seg7_data & 0x80; // shift data from Most significan bit (D7)
if(temp == 0x80)
IOSET0 |= 1 << 19; //IOSET0 | 0x00080000;
else
IOCLR0 |= 1 << 19; //IOCLR0 | 0x00080000;
if(i==7&&j==7){
IOSET0 |= 1 << 19;
}
//send one clock pulse
IOSET0 |= 1 << 20;  //IOSET0 | 0x00100000;
 delay_ms(1);       // nop();
IOCLR0 |= 1 << 20;  //IOCLR0 | 0x00100000;
seg7_data = seg7_data << 1; // get next bit into D7 position
```

```
}

}
IOSET0 |= 1 << 30; //IOSET0 | 0x40000000;
delay_ms(1);      //nop();
IOCLR0 |= 1 << 30; //IOCLR0 | 0x40000000;
}
```