Documentation on

PROJECTS MADE EASY

Prepared by

Dr. M.CHAKRAVARTHY

Prof., HOD-EEE

&

G.Sandhya Rani

Asst. Professor

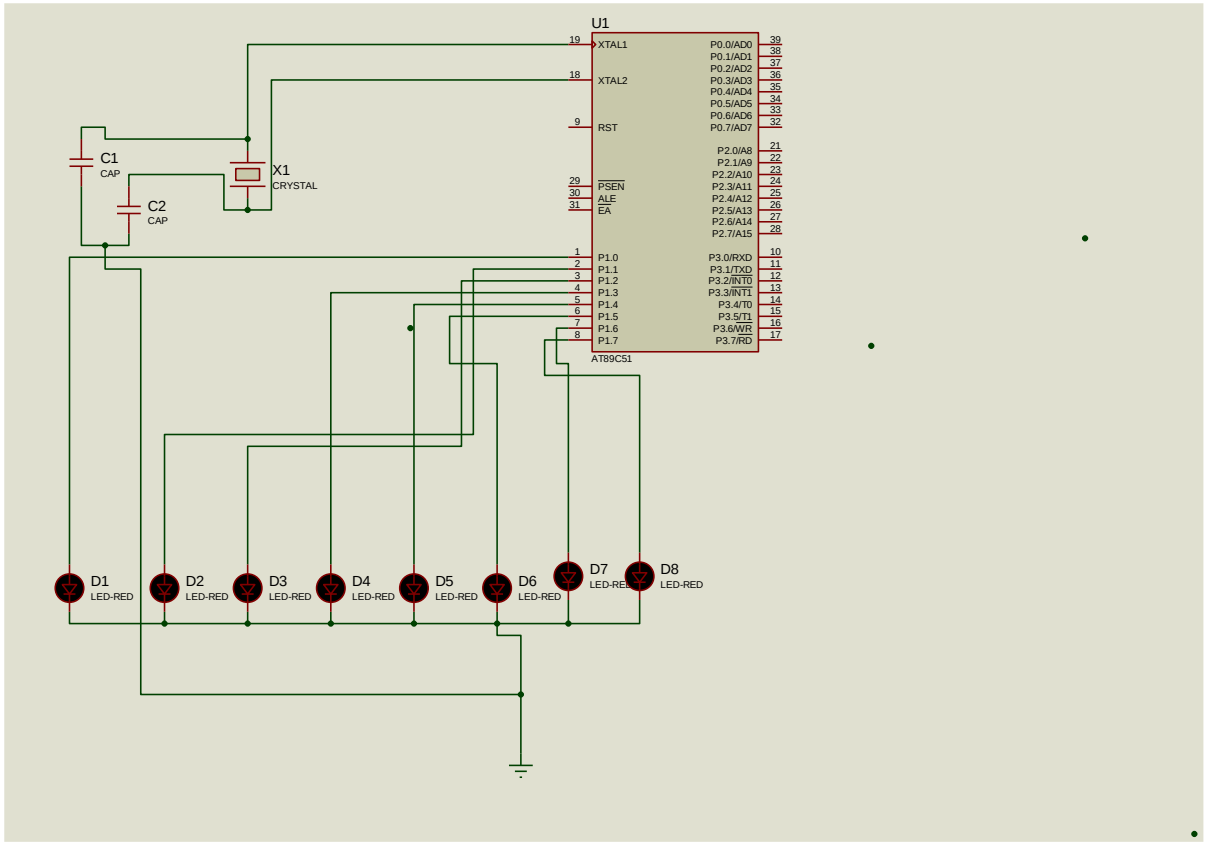
Department of Electrical and Electronic Engineering

List of Experiments

1. LED INTERFACING
2. LCD INTERFACING
3. KEYBOARD AND LCD INTERFACING
4. STEPPER MOTOR INTERFACING
5. WATER LEVEL CONTROLLER
6. DIGITAL THEMOMETER
7. FASTEST FINGER FIRST USING
8. MICROCONTROLLER BASED AUTOMATIC BUZZER
9. LOAD SHEDDER
10. DATA LOGGER
11. INDUSTRIAL TIMER
12. DC MOTOR INTERFACING
13. DC MOTOR SPEED CONTROL
14. SOUND ACTIVATED ROBOT
15. TEMPERATURE MONITORING
16. DISPLAYING OF TIME
17. AUTOMATIC ROOM LIGHT CONTROL WITH VISITOR COUNTER
18. POWER FACTOR CONTROL
19. SPEED CONTROL OF INDUCTION MOTOR USING IGBT
20. OBSTACLE DETECTION FOR VISUALLY HANDICAPPED PERSONS

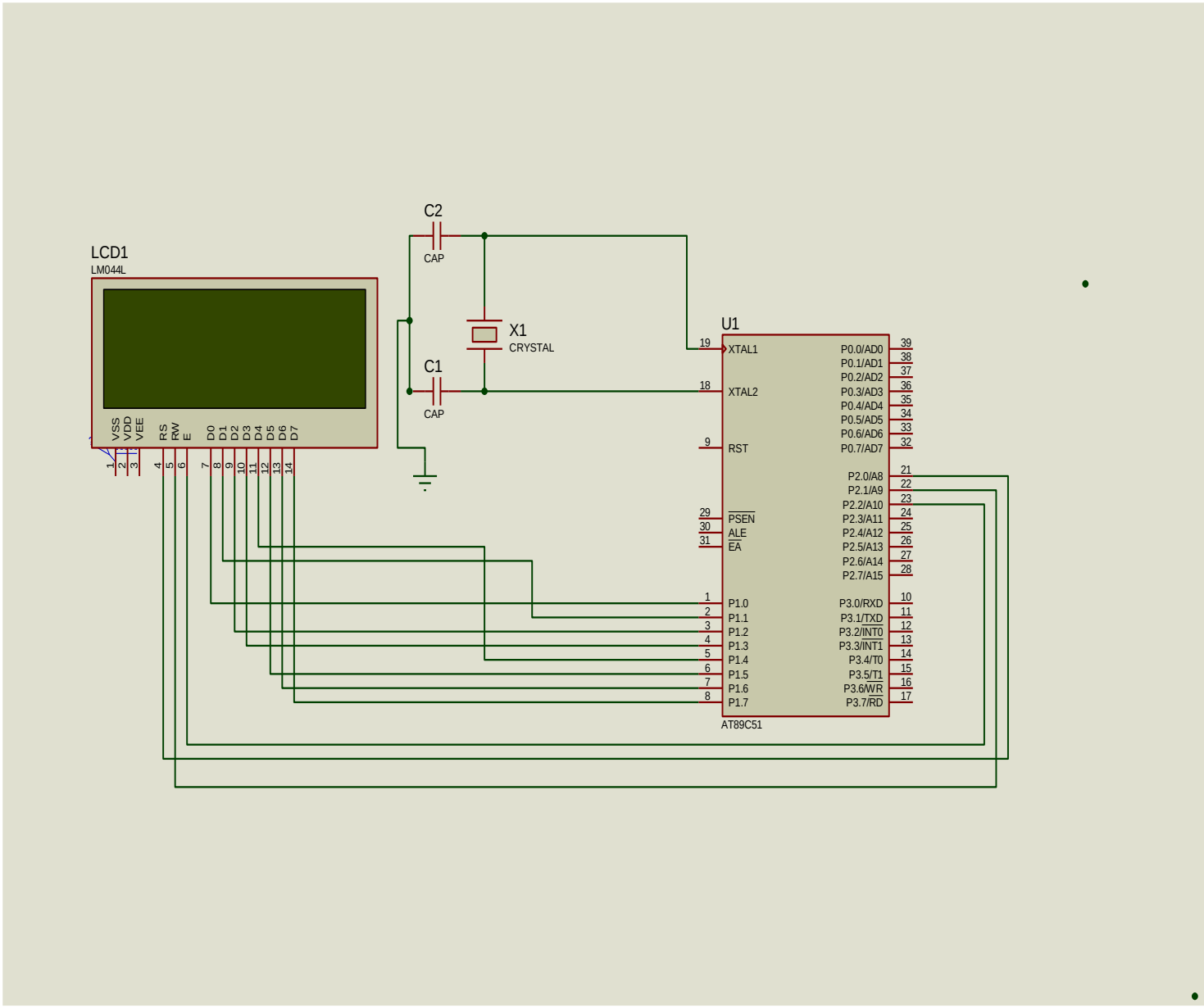
Chapter 1: LED INTERFACING

```
#include<reg51.h>
#define first P1
#define second P2
//sbit first P1^0
//sbit second P2^0
void wait()
{
  int i,j;
  for(i=0;i<=5000;i++)
    for(j=0;j<=100;j++);
 }
main()

 {
  unsigned char n1=0xAA,n2=0x55;          /* Delay var */
 // unsigned char j,m;                     /* LED var */

  while (1)
  {
    P1=n1;
    wait();
    P1=n2;
wait();

}
}
```


U1
XTAL1
XTAL2
RST
PSEN
ALE
EA
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
AT89C51
C1
CAP
C2
CAP
X1
CRYSTAL
D1
LED-RED
D2
LED-RED
D3
LED-RED
D4
LED-RED
D5
LED-RED
D6
LED-RED
D7
LED-RED
D8
LED-RED

Chapter 2:LCD INTERFACING

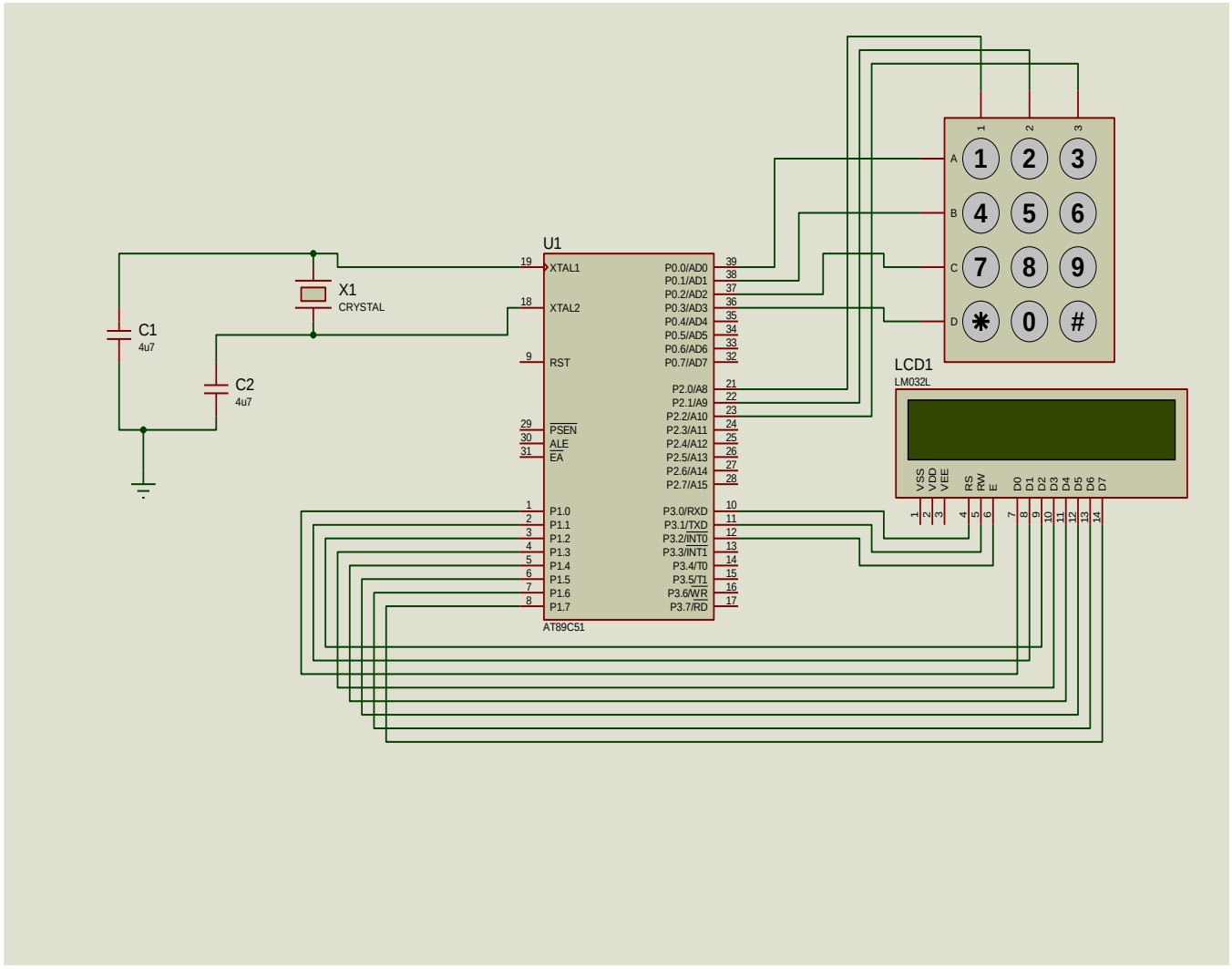
```
#include<reg51.h>
sfr ldata=0x90;
sbit rs=P2^0;
sbit rw=P2^1;
sbit en=P2^2;
void delay(unsigned int itime);
void lcdcmd(unsigned char value);
void lcddata(char *value);
int m=0,n=0;

void main()
{
  lcdcmd(0x38);/* selection of 2 lines of 5*7 matrix */
  delay(250);
  lcdcmd(0x0e);/* Display On cursor blinking */
  delay(250);
lcdcmd(0x01);/* Clear display screen */
  delay(250);
lcdcmd(0x06);/* shift cursor left */
  delay(250);
lcdcmd(0x80);/* force cursor to beginning of first line */
  delay(250);
lcddata("This is chakri");
//lcdcmd(0x07);
}


void lcdcmd(unsigned char value)
 {
   ldata=value;
   rs=0;
   rw=0;
   en=1;
   delay(1);
   en=0;
   return;
 }
void lcddata(char *name1)
 {
    int i;
   for(i=0;name1[i]!='\0';i++)
   {
   ldata=name1[i];
   rs=1;
   rw=0;
   en=1;
   delay(1);
   en=0;
   }
   return;
 }
 void delay(unsigned int itime)
 {
   unsigned int i,j;
    for(i=0;i<=itime;i++)
     for(j=0;j<=1257;j++); }
```


LCD1
LM044L
VSS
VDD
VEE
RS
RW
E
D0
D1
D2
D3
D4
D5
D6
D7
C2
CAP
X1
CRYSTAL
C1
CAP
U1
XTAL1
XTAL2
RST
PSEN
ALE
EA
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
AT89C51

Chapter 3:KEYBOARD AND LCD INTERFACING

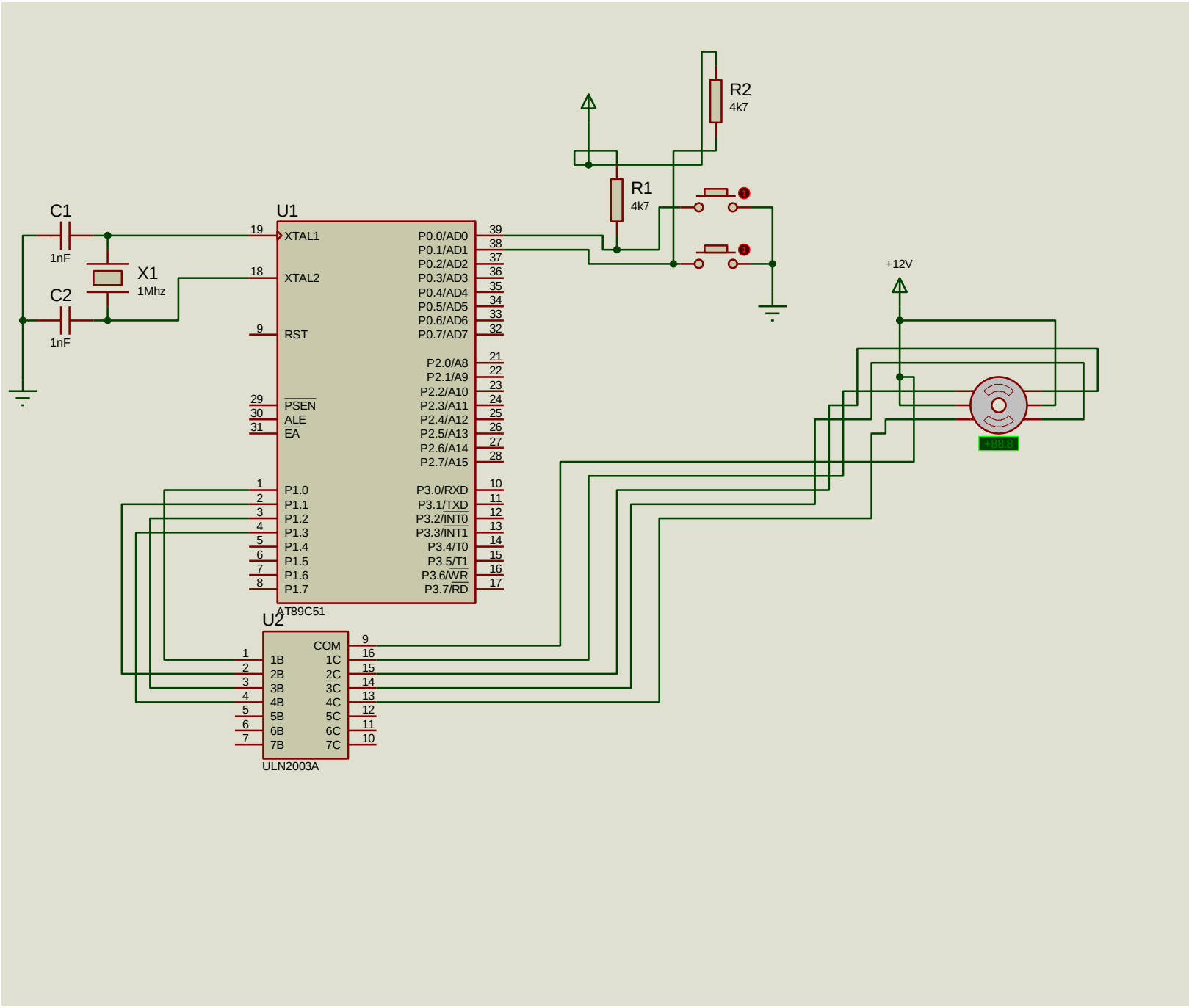
```
#include<reg51.h>
sbit rs = P3^0;
sbit rw = P3^1;
sbit e  = P3^2;

unsigned char keys[4][3]={ '1','2','3',

                           '4','5','6',

                           '7','8','9',

                           '*','0','#',};

unsigned char a,b;

void lcdcmd(unsigned char);
void lcddata(unsigned char);
void delay(int);

void main()
{
    lcdcmd(0x38);
      lcdcmd(0x01);
      lcdcmd(0x0e);
      lcdcmd(0x14);


      P2 = 0x07;
      y:  P0 = 0x00;
          delay(20);
            if( P2 == 0x07)
            goto y;
      delay(20);
      if( P2 == 0x07)
      goto y;
      a = P2;

      P0 = 0x0e;
      if( P2 != 0x07)
      goto row1;
      P0 = 0x0d;
      if( P2 !=0x07)
      goto row2;
      P0 = 0x0b;
      if( P2 !=0x07)
      goto row3;
      P0 = 0x07;
      if( P2 != 0x07)
      goto row4;
      else
      goto y;

      row1:  if( P2 == 0x06)
             lcddata( keys[0][0]);
               else if( P2 == 0x05)
               lcddata( keys[0][1]);
               else if( P2 == 0x03)
               lcddata( keys[0][2]);
```

```
                goto y;

        row2:  if( P2 == 0x06)
               lcddata( keys[1][0]);
                 else if( P2 == 0x05)
                 lcddata( keys[1][1]);
                 else if( P2 == 0x03)
                 lcddata( keys[1][2]);
                 goto y;

        row3:  if( P2 == 0x06)
               lcddata( keys[2][0]);
                 else if( P2 == 0x05)
                 lcddata( keys[2][1]);
                 else if( P2 == 0x03)
                 lcddata( keys[2][2]);
                 goto y;

        row4:  if( P2 == 0x06)
               lcddata( keys[3][0]);
                 else if( P2 == 0x05)
                 lcddata( keys[3][1]);
                 else if( P2 == 0x03)
                 lcddata( keys[3][2]);

        goto y;
}

void lcdcmd(unsigned char value)
{    rs = 0;
     rw = 0;
       e = 1;
       delay(1);
       P1 = value;
       e = 0;
}

void lcddata(unsigned char value)
{    rs = 1;
     rw = 0;
       e = 1;
       delay(1);
       P1 = value;
       e = 0;
}

void delay(int time)
{  int i,j;
   for( i=0;i<=time;i++)
   for(j=0;j<=1275;j++);
}
```


U1
AT89C51
XTAL1
XTAL2
RST
PSEN
ALE
EA
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
X1
CRYSTAL
C1
4u7
C2
4u7
LCD1
LM032L
VSS
VDD
VEE
RS
RW
E
D0
D1
D2
D3
D4
D5
D6
D7
1
2
3
4
5
6
7
8
9
*
0
#
A
B
C
D

Chapter:4 STEPPER MOTOR INTERFACING

```
#include<reg51.h>
#define no_of_rotations 4
sbit m= P0^0;
sbit n=P0^1;
void delay()
{
  int i,j;
   for(i=0;i<=1000;i++)
     for(j=0;j<=100;j++);
 }

main()
{
  int forward,reverse,x;
  forward=0x01;
  reverse=0x08;
  x=no_of_rotations;

  while(1)
  {

   if(m==0)
    {
      while(x!=0)
      {
       P1=forward;
         forward<<=1;
       delay();

       if(forward==0x10)
         forward=0x01;
       }
      x--;
    }
    if(n==0)
    {
      while(x!=0)
      {
       P1=reverse;
         reverse>>=1;
       delay();

       if(reverse==0x00)
         reverse=0x08;
       }
      x--;
    }
  }
}
```

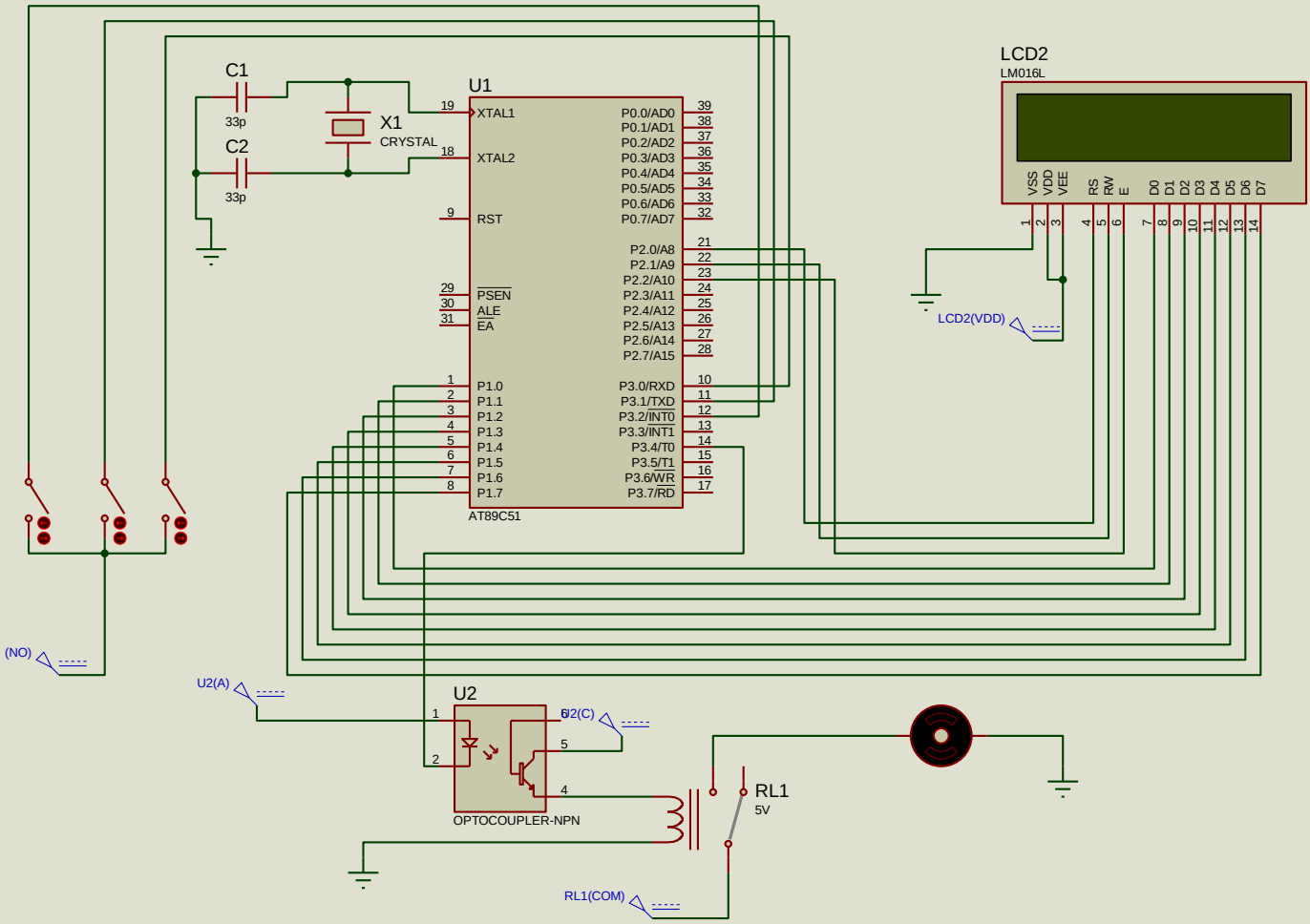

C1
1nF
X1
1Mhz
C2
1nF
U1
XTAL1
XTAL2
RST
PSEN
ALE
EA
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
AT89C51
R1
4k7
R2
4k7
+12V
+88.8
U2
COM
1B
2B
3B
4B
5B
6B
7B
1C
2C
3C
4C
5C
6C
7C
ULN2003A

Chapter 5:WATER LEVEL CONTROLLER

```
#include<reg51.h>
sbit rs=P2^0;
sbit rw=P2^1;
sbit en=P2^2;
sbit r1bit=P3^0;
sbit r2bit=P3^1;
sbit r3bit=P3^2;
sbit relay=P3^4;
//sbit l=P3^0;
//sbit m=P3^1;
//sbit n=P3^2;
void delay(unsigned int time);
void lcdcmd(unsigned char value);
void lcddata( char *value);

//char *name1="motor on";
//char *name2="half filled";
//char *name3="dry sump";
//char *name4="tank full halt the motor";
//char *name="bye";
//int a=0,b=0,c=0,d=0;
unsigned char variable;

void main()
{

while(1)
{

 TMOD=0xf1;
  lcdcmd(0x08);
  delay(50);
  lcdcmd(0x0e);
  delay(50);
  lcdcmd(0x01);
  delay(50);
  lcdcmd(0x06);
  delay(50);
  lcdcmd(0x81);
  delay(50);
P3=0x00;
delay(100);
variable=P3;
relay=1;

//m=0x00;
//n=0x00;
delay(1);

if((variable&0x0f)==0x00)
{

lcddata("motor on");
relay=0;
delay(500);
lcdcmd(0x80);
}
```

```
else if((variable&0x0f)==0x01)
{
lcddata("minimum level");
relay=0;
delay(500);
lcdcmd(0x80);
delay(500);
/* if(P3!=0x03)
{
delay(50000);
lcddata("dry state warning");
delay(50);
lcdcmd(0x80);
delay(1);
}
else
{
delay(50);
}      */
}
else if((variable&0x0f)==0x03)
{
lcddata("half filled");
relay=0;
delay(500);

lcdcmd(0x80);
 delay(500);
}
else if((variable&0x0f)==0x07)
{
lcddata("Tank full");
  delay(500);
lcdcmd(0x80);
  delay(500);}
else
{delay(1);
}}}
 void lcdcmd(unsigned char value)
 {
   P1=value;
   rs=0;
   rw=0;
   en=1;
   delay(1);
   en=0;
   return;
 }
void lcddata(char *value)
 {
   int i;
    for(i=0;value[i]!='\0';i++)
      {
   P1=value[i];
   rs=1;
   rw=0;
   en=1;
   delay(1);
```

```
  en=0;
  }
  return;
}

void delay(unsigned int itime)
{
  unsigned int i,j;
   for(i=0;i<=itime;i++)
    for(j=0;j<=1275;j++);
}
```



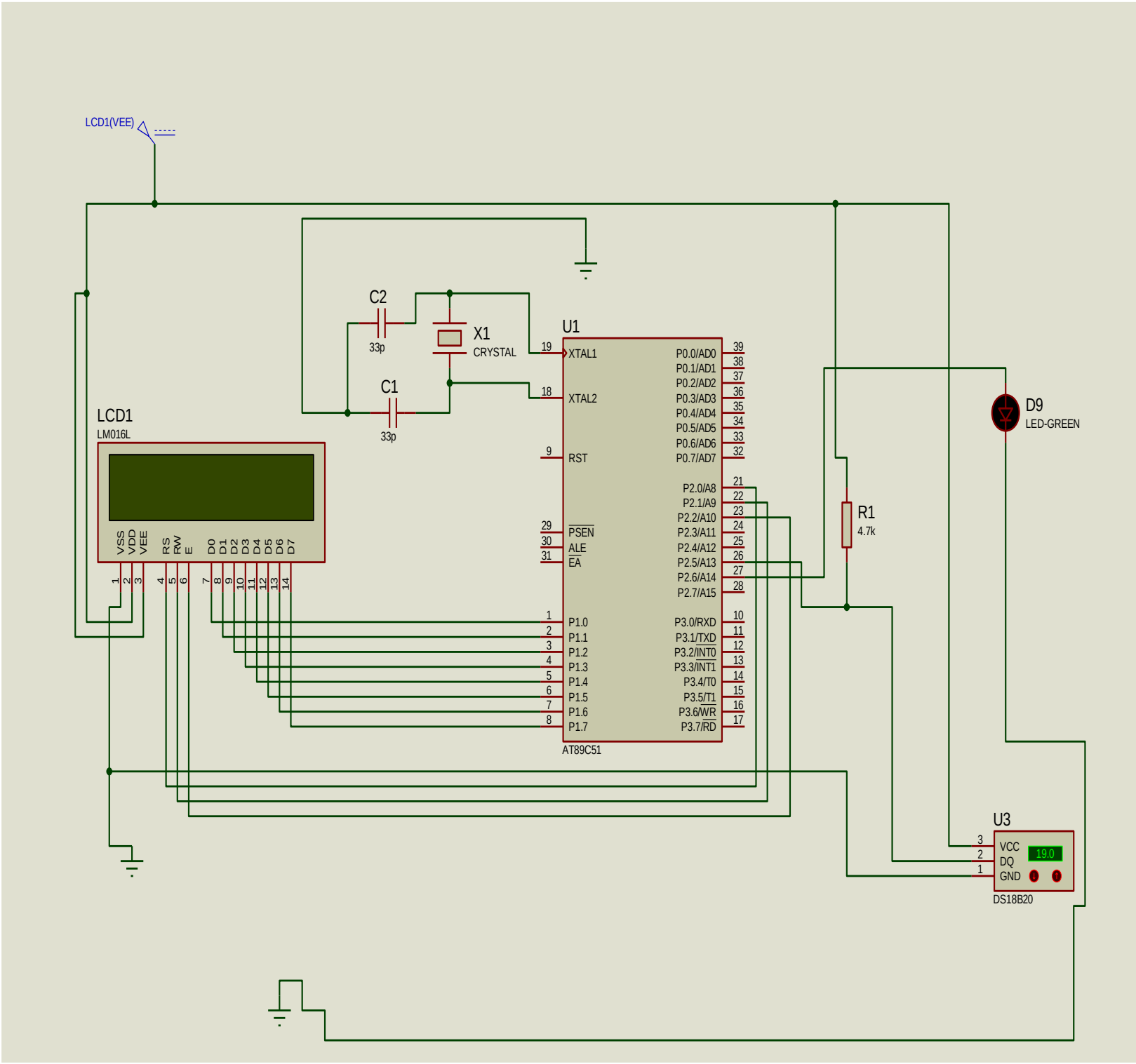
chapter:6 DIGITAL THEMOMETER

```
#include<reg51.h>
sfr ldata=0x90;
sbit rs=P2^0;
sbit rw=P2^1;
sbit en=P2^2;
sbit DQ=P2^5;
sbit output=P2^6;
void lcdcmd(unsigned char value);
void lcddata_number(unsigned char value);
void lcddata(unsigned char *value);
void lcddelay(unsigned int time);
bit reset();
void delay(int us);
bit readbit(void);
void writebit(bit dbit);
unsigned char readbyte(void);
void writebyte(unsigned char dout);
void readtemp();
unsigned char mydata1,mydata2,mydata3;


 void main()
 {

  lcdcmd(0x38);
  lcddelay(10);

  lcdcmd(0x0e);
  lcddelay(10);

  lcdcmd(0x01);
  lcddelay(10);

  lcdcmd(0x06);
  lcddelay(10);

 lcddata("temperature");
while(1)
{
  readtemp();
 }
}

bit reset()
{
  bit presense;
  DQ=0;
  delay(29);
  DQ=1;
  delay(3);
  presense=DQ;
  delay(25);
  output=presense;
  return(presense);
   }
```

```
bit readbit(void)
{
unsigned char i=0;
DQ=0;
DQ=1;
for(i=0;i<3;i++);
return(DQ);
}

void writebit(bit Dbit)
{
unsigned char i=0;
DQ=0;
DQ=Dbit?1:0;
delay(5);
DQ=1;
}

unsigned char readbyte(void)
{
unsigned char i;
unsigned char din=0;
for(i=0;i<8;i++)
{
din|=readbit()?0x01<<i:din;
delay(6);
}
//B=din;
return(din);
}

void writebyte(unsigned char dout)
{
unsigned char i;
for(i=0;i<8;i++)
{
writebit((bit)(dout&0x01));
dout=dout>>1;
}
delay(5);
}

void readtemp()
{

 unsigned char n;
 unsigned char buff[9];
 EA=0;
 reset();
 writebyte(0xcc);
 writebyte(0x44);
 while(readbyte()==0xff);
 reset();
 writebyte(0xcc);
 writebyte(0xbe);
 for(n=0;n<9;n++)
 {

  buff[n]=readbyte();
  delay(100);
```

```
 }
  mydata1=buff[0];
  delay(10);
  mydata1=mydata1>>4;

  mydata2=buff[1];
  delay(10);
  mydata2=mydata2<<4;
  mydata3=mydata2|mydata1;

lcddata_number(mydata3);
//lcddelay(1);
delay(2500);
EA=1;

}

 void lcdcmd(unsigned char value)
 {
  ldata=value;
  rs=0;
  rw=0;
  en=1;
  lcddelay(1);
  en=0;
  return;
 }


 void lcddata_number(unsigned char value)

 {
   char data1;
   lcdcmd(0x8c);
   delay(1000);
   B=value;
data1=value%10;
value=value/10;
if(value>9)
{
ldata=1+0x30;

  for(;value!=0;)
  {
  data1 =value%10;
 ldata=data1+0x30;
 rs=1;
 rw=0;
 en=1;
 lcddelay(100);
 en=0;
 ldata=value%10+0x30;
 rs=1;
 rw=0;
 en=1;
 lcddelay(10);
 en=0;
 }
 }
```

```
else
 {
 ldata=value+0x30;
 lcddelay(10);
 rs=1;
 rw=0;
 en=1;
 lcddelay(10);
 en=0;}
ldata=data1+0x30;
 rs=1;
 rw=0;
 en=1;
 lcddelay(100);
 en=0;
 ldata='\0';
 rs=1;
 rw=0;
 en=1;
 lcddelay(100);
 en=0;
 return;
}
void lcddata(unsigned char *value)
{
 char i;
 for(i=0;value[i]!='\0';i++)
{
ldata=value[i];
 rs=1;
 rw=0;
 en=1;
 lcddelay(100);
 en=0;
 }
 return;
}
 void lcddelay(unsigned int time)
 {
 unsigned int i,j;
 for(i=0;i<=time;i++)
 {
 for(j=0;j<=1000;j++);
 }}


void delay(int us)
{
 int i;
 for(i=0;i<us;i++);
}
```

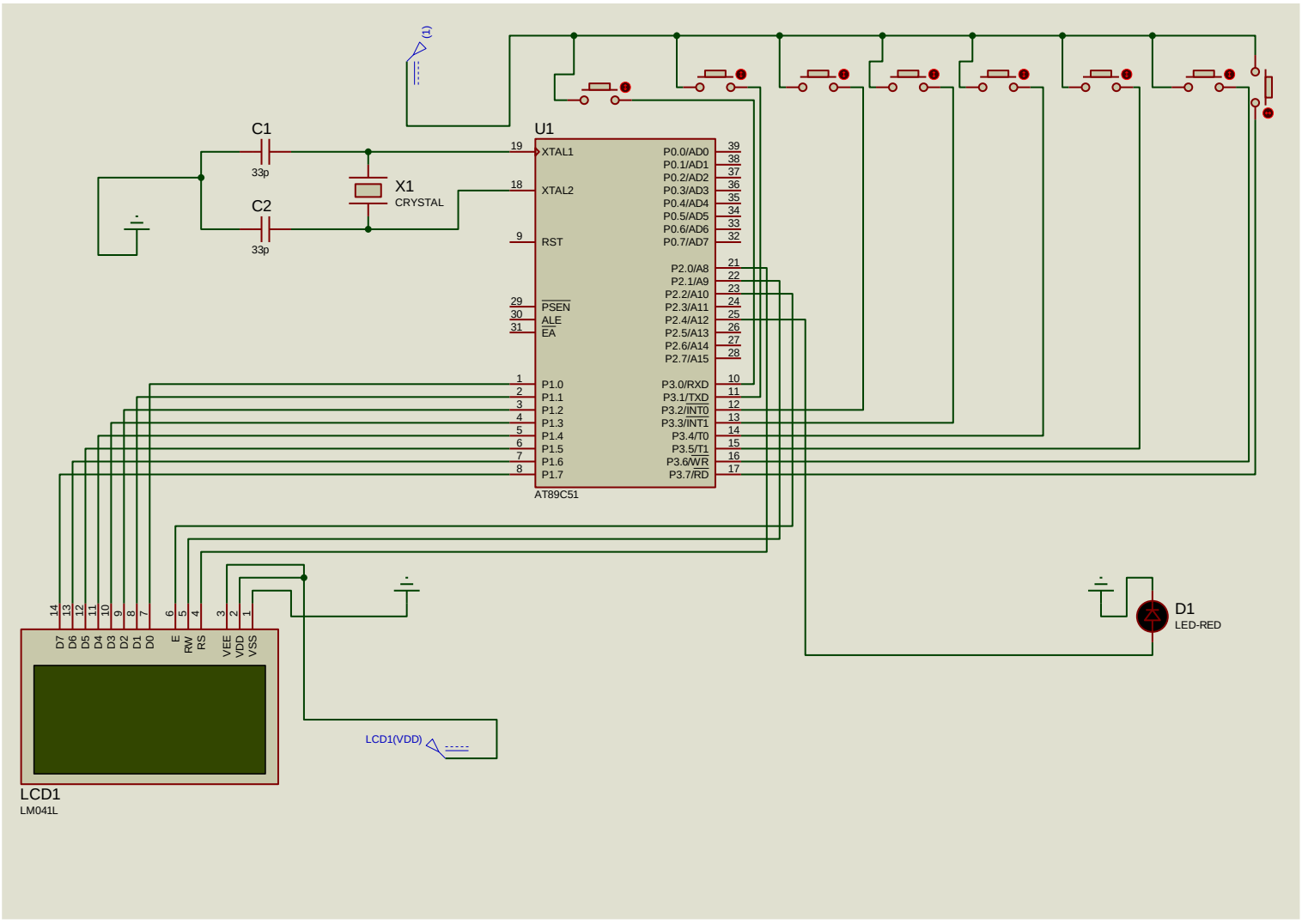

LCD1(VEE)
C2
33p
X1
CRYSTAL
C1
33p
U1
XTAL1
XTAL2
RST
PSEN
ALE
EA
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
AT89C51
LCD1
LM016L
VSS
VDD
VEE
RS
RW
E
D0
D1
D2
D3
D4
D5
D6
D7
D9
LED-GREEN
R1
4.7k
U3
VCC
DQ
GND
19.0
DS18B20

Chapter:7 Fastest Finger First Using At89c51

```
#include<reg51.h>
void lcdcmd(unsigned char value);
void lcddata(char *value);
void delay(unsigned int itime);
sbit rs=P2^0;
sbit rw=P2^1;
sbit en=P2^2;
sbit buzzer=P2^4;
sbit start=P3^0;
sbit player_1=P3^1;
sbit player_2=P3^2;
sbit player_3=P3^3;
sbit player_4=P3^4;
sbit player_5=P3^5;
sbit player_6=P3^6;
sbit player_7=P3^7;

void main()
{
lcdcmd(0x38);
delay(250);
lcdcmd(0x0e);
delay(250);
lcdcmd(0x01);
delay(250);
lcdcmd(0x06);
delay(250);
lcdcmd(0x86);
delay(250);

P3=0x00;
buzzer=0;

while(1)
{
buzzer=0;
if(start==1)
{
lcddata("start");
lcdcmd(0x81);
buzzer=1;
delay(100);
lcdcmd(0x01);

}

else if(player_1==1)
{
lcddata("player1");
lcdcmd(0x81);
buzzer=1;
delay(100);
lcdcmd(0x01);
}

else if(player_2==1)
{
```

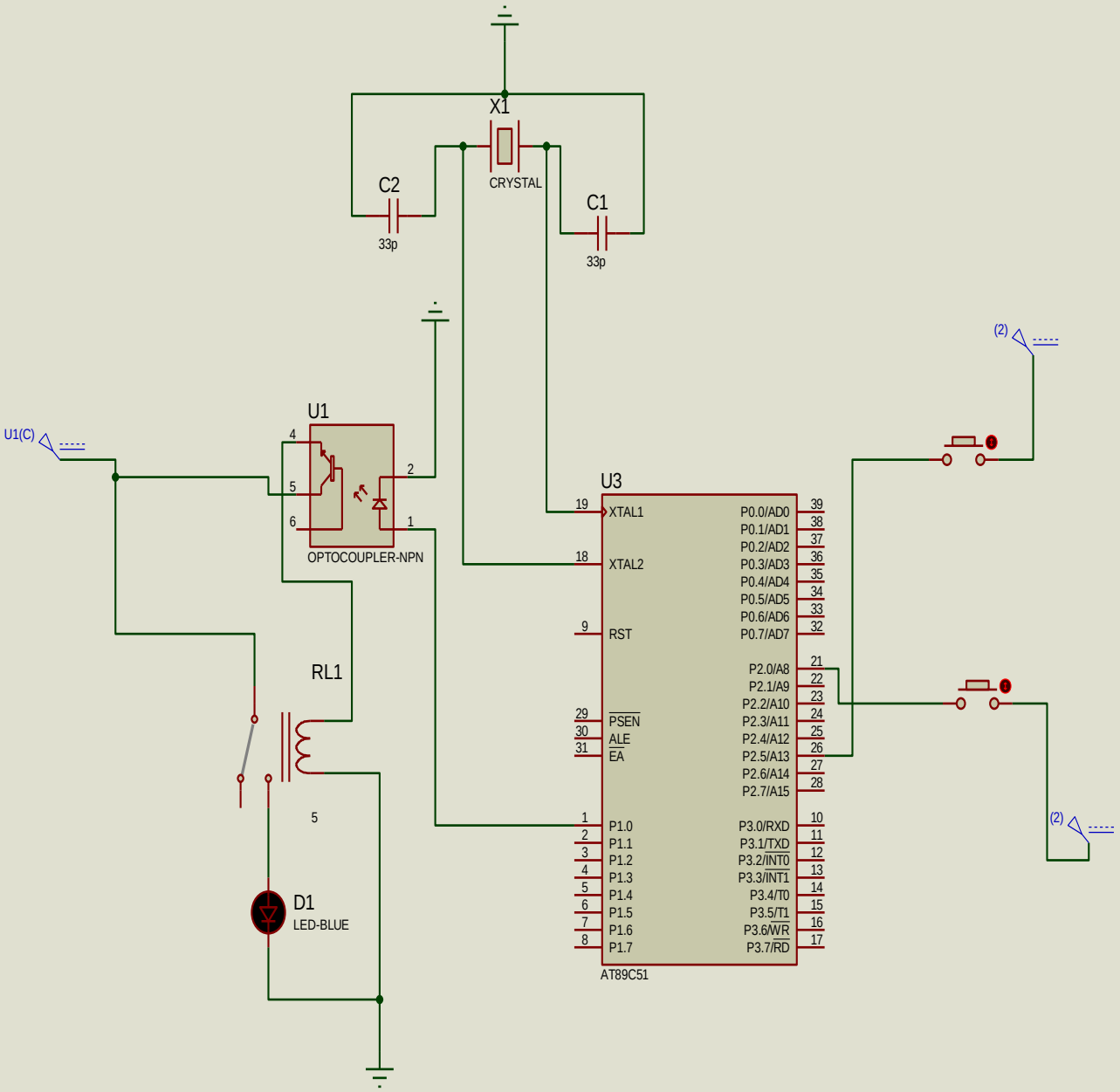
```
lcddata("player2");
lcdcmd(0x81);
buzzer=1;
delay(100);
lcdcmd(0x01);
}

else if(player_3==1)
{
lcddata("player3");
lcdcmd(0x81);
buzzer=1;
delay(100);
lcdcmd(0x01);
}

else if(player_4==1)
{
lcddata("player4");
lcdcmd(0x81);
buzzer=1;
delay(100);
lcdcmd(0x01);
}

else if(player_5==1)
{
lcddata("player5");
lcdcmd(0x81);
buzzer=1;
delay(100);
lcdcmd(0x01);
}

 else if(player_6==1)
{
lcddata("player6");
lcdcmd(0x81);
buzzer=1;
delay(100);
lcdcmd(0x01);
}

else if(player_7==1)
{
lcddata("player7");
lcdcmd(0x81);
buzzer=1;
delay(100);
lcdcmd(0x01);
}

else
{
lcddata("nop");
lcdcmd(0x81);
buzzer=0;
lcdcmd(0x01);}}}
void lcdcmd(unsigned char value)
{
```

```
P1=value;
rs=0;
rw=0;
en=1;
delay(1);
en=0;}
void lcddata(char *value)
{
int i;
for(i=0;value[i]!='\0';i++)
{
P1=value[i];
rs=1;
rw=0;
en=1;
delay(1);
en=0;}}
void delay(unsigned int itime)
{
unsigned int i,j;
for(i=0;i<itime;i++)
for(j=0;j<1275;j++);
}
```



chapter 8:MICROCONTROLLER BASED AUTOMATIC BUZZER

```
#include<reg51.h>
sbit buzzer=P1^0;
sbit power_on =P2^0;
sbit power_off=P2^5;

void timer();
   int i;
void main()
{
  int variable1=1,variable2=1;
  buzzer=0;
  power_on=0;
  power_off=0;
  while(1)
  {

  if(power_on==1&&variable1==1)
    {
      buzzer=1;
      timer();
      buzzer=0;
      variable1=0;
      variable2=1;
     }

  else if(power_off==1&&variable2==1)
    {
      buzzer=1;


      timer();
      buzzer=0;
      variable1=1;
      variable2=0;
    }
  }
}

void timer()
{
 for(i=0;i<200;i++)
       {
TMOD=0x01;
TL0=0xfd;
TH0=0x4a;
TR0=1;
while(TF0==0);
TF0=0;
}
return;
}
```

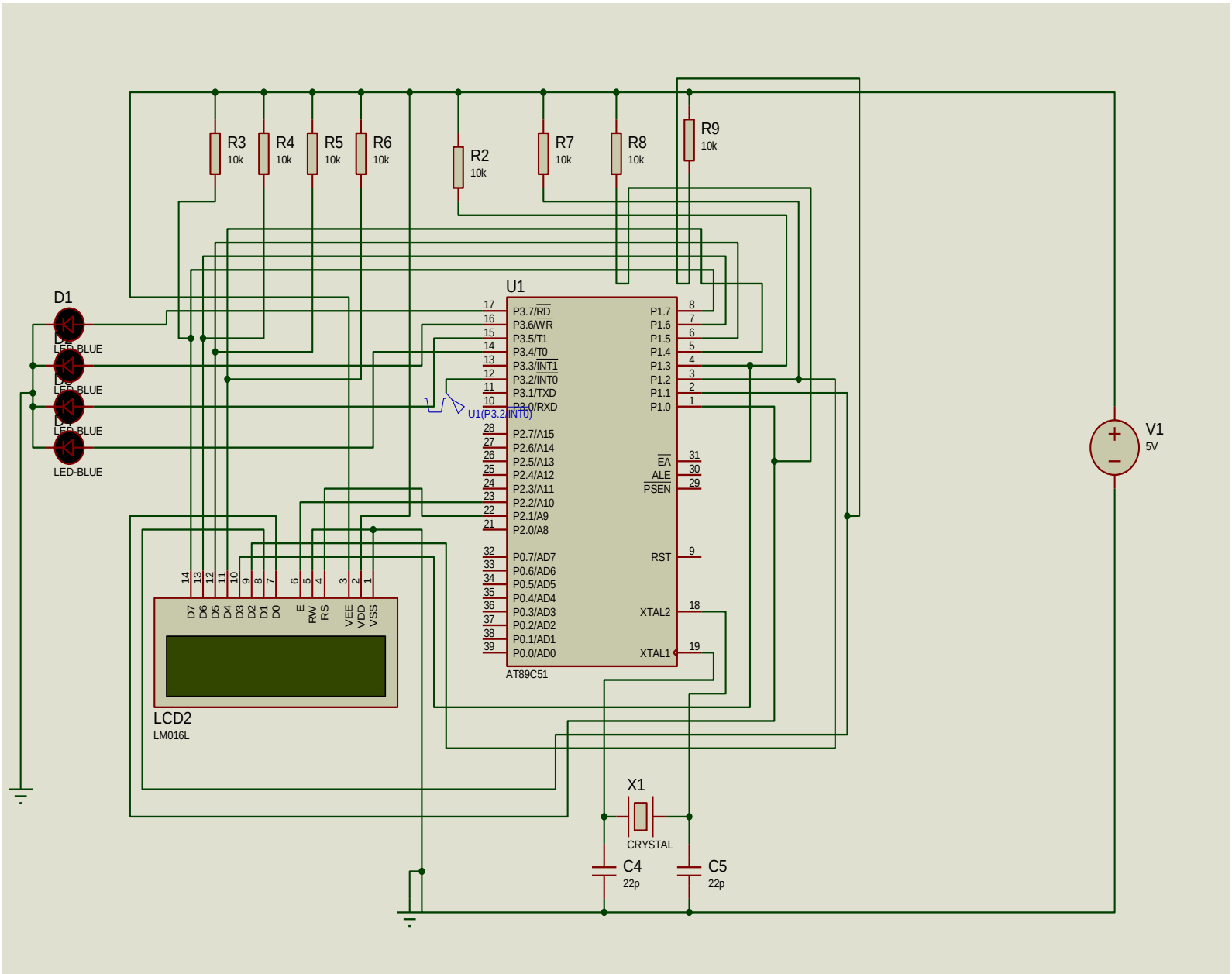

X1
CRYSTAL
C2
33p
C1
33p
U1
U1(C)
OPTOCOUPLER-NPN
U3
XTAL1
XTAL2
RST
PSEN
ALE
EA
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
AT89C51
RL1
D1
LED-BLUE
(2)
(2)

Chapter 9:LOAD SHEDDER

```
#include <reg51.h>
#include<string.h>
sbit L1=P3^7;
sbit L2=P3^6;
sbit L3=P3^5;
sbit L4=P3^4;
sbit en=P2^2;
sbit rs=P2^1;
sbit pulsein=P3^2;
sfr ldata=0x90;
float n=0x00;
int variable1=1,variable2=15,b;
void delay(unsigned int itime);
void lcdcmd(unsigned int value);
void lcddata(unsigned int value);
void timer0() interrupt 1
{
 TMOD=0x01;
 TH0=0x0d;
 TL0=0x01;
 IE=0x82;
 TR0=1;
 variable2--;
  if(variable2==0)
 {
 variable2=15;
 TMOD=0x10;
 TH1=0X08;
 TL1=0Xff;
 IE=0X88;
 TR1=1;
 }
 }
 void timer1() interrupt 3
 {
  lcdcmd(0x38);
  delay(50);
  lcdcmd(0x01);
  delay(50);
  lcdcmd(0x0e);
  delay(50);
  lcdcmd(0x81);
          if(n==48)
             {
               L1=1;
             delay(100);
            }
                  else if(n==49)
                  {
                   L1=1;
                   L2=1;
                 delay(100);
                  }
                  else if(n==50)
                         {
```

```
                        L1=1;
                        L2=1;
                        L3=1;
                        delay(100);
                       }



                            else if(n==51)
                                  {
                                        L1=1;
                                        L2=1;
                                        L3=1;
                                        L4=1;
                                        delay(10);
                                  }
lcddata(n);
delay(1);
}
void delay(unsigned int itime)
 {
   unsigned int i,j;
    for(i=0;i<=itime;i++)
     for(j=0;j<=1275;j++);

 }

 void main()
{
P3=0x00;
TMOD=0x01;
TH0=0x00;
TL0=0x00;
//delay(100);
IE=0x82;
TR0=1;
                  while(1)
                  {
                  if(pulsein==1&&variable1==1)
                  {
                  n++;
                   variable1=0;
                   //variable2=1;
                  }
                  else if(pulsein==0&&variable1==0)
                  {
                   variable1=1;
                  // variable2=0;}}}
 void lcdcmd(unsigned int value)

 {    ldata=value;
   rs=0;
   en=1;
   delay(1);
   en=0;
   return;}
  void lcddata(unsigned int value)
 {   int a[4],i,send=0,n=0;
   for( i=0;value != 0; i++)
```

```
{     send=value%10;
      value=value/10;
     a[i]=send;
     n++; }
     for(i=n-1;i>=0;i--)
     {      ldata = a[i] + 0x30;
      rs=1;
    en=1;
    delay(1);
    en=0;
      P3=0x00;
      if(ldata==0x31)
     {      P3=0xff;
     delay(10);}}
     return;}
```



Chapter:10 DATA LOGGER

```
#include<reg51.h>
#include<stdio.h>
#include<string.h>
sbit CS=P2^0;
sbit SCLK=P2^1;
sbit DIN=P2^2;
sbit DOUT=P2^3;
sbit MSBRA=B^7;
sbit LSBRA=B^0;
sbit out=P1;
void lcdcmd(unsigned char value);
void lcddata(char *value);
void Delay(unsigned int);
sfr ldata=0x90;
sbit rs=P2^4;
sbit rw=P2^5;
sbit en=P2^6;
void Delay(unsigned int itime)
 {
  unsigned int i,j;
 for(i=0;i<itime;i++)
   for(j=0;j<1275;j++);
 }
void main()
{
int i=0;
unsigned char conbyte=0xc0;
   char y[4],f[5];
        char rema,m;
            int queti;
            int x;
            float n;
            int add,e1,r1,q2,r2,e2;

      lcdcmd(0x38);
      Delay(20);        /*time delay 250ms*/
      lcdcmd(0x0e);
      Delay(20);
      lcdcmd(0x01);
      Delay(20);
      lcdcmd(0x06);
      Delay(20);
      lcdcmd(0x81);
      Delay(20);
      CS=0;
while(1)
{
            for(conbyte=0xc0;conbyte!=0x00;conbyte=conbyte+0x10)
            {
            CS=0;
             B=conbyte;
             CS=0;
             Delay(20);
            // lcdcmd(0x01);
```

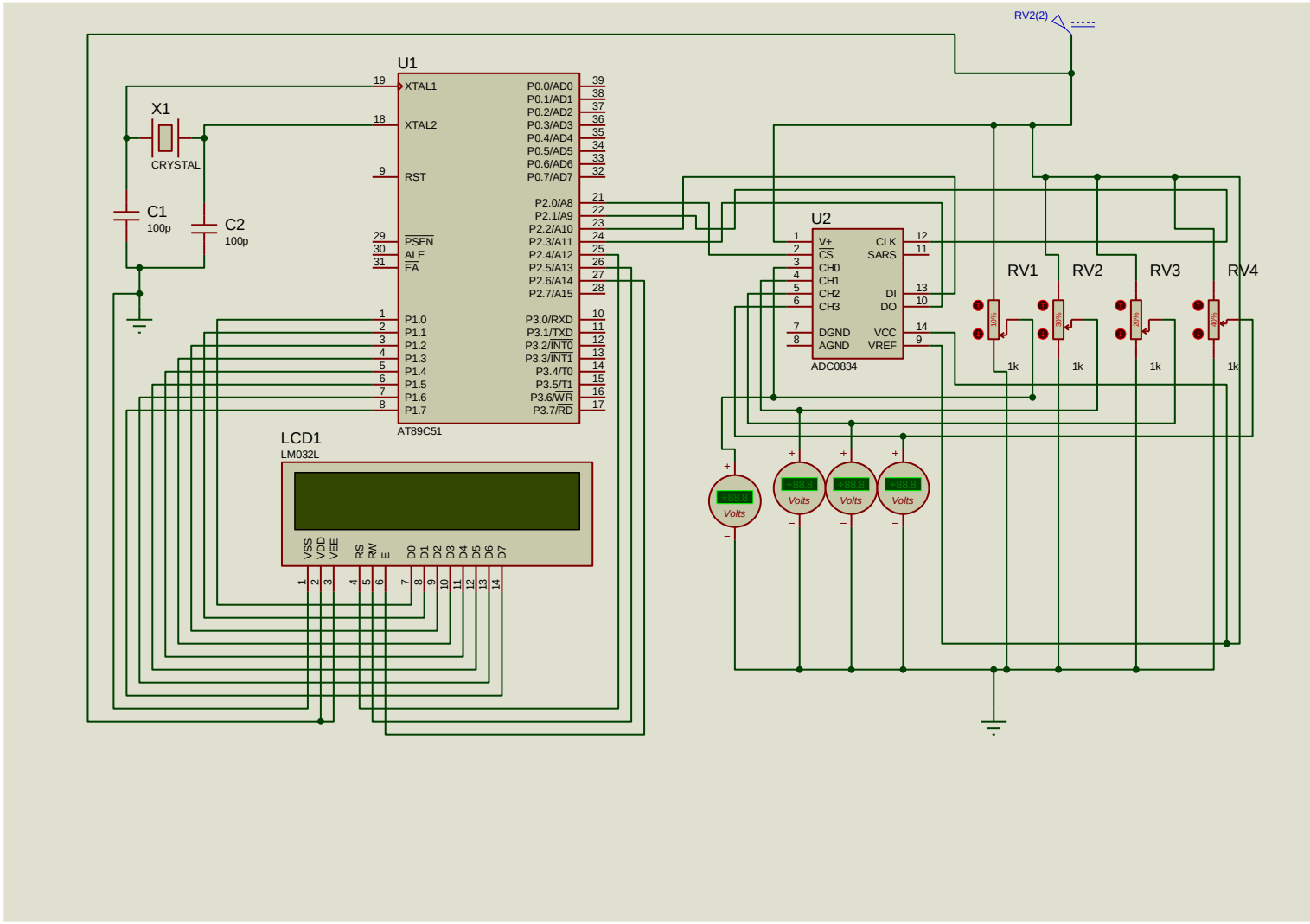
```
     Delay(25);
   if(conbyte==0xc0)                  /*ac voltage signal conversion*/
   {
          for(x=0;x<5;x++)
          {
          SCLK=1;
          Delay(20);                  /*time delay 20ms*/
                DIN=MSBRA;
          SCLK=0;
          Delay(20);
                B=B<<1;
          }
          Delay(20);
          SCLK=0;
          Delay(20);
          SCLK=1;
          Delay(20);
          SCLK=0;
          Delay(20);
   for(x=0;x<8;x++)
   {
          SCLK=1;
   Delay(20);
          LSBRA=DOUT;
          SCLK=0;
   Delay(20);
   B=B<<1;
          Delay(20);
   }
   queti=B;
          lcdcmd(0x81);
          Delay(10);
          n=(float)B;
   for(m=0;n!=0x00;m++)
{
          rema=(char)n%10;
   y[m]=rema+0x30;
          Delay(20);
          n=n/10;
}
   y[m]='\0';

    add=(122*(y[1]-48))+((y[2]-48)*122*10);
     e1=add/1000;
     f[0]=e1+48;
     r1=add%1000;
     e2=r1/100;
     f[1]=e2+48;
     r2=r1%100;
     q2=r2/10;
     f[2]=q2+48;
     f[3]='\0';
     if(i==0)
     lcddata("DATA LOGGING");
     if(i==1)
     {
     lcddata("suply volt=");
     lcddata(f);
     lcddata("V");
     Delay(500);
```

```
        }
        i=1;
}                                       *end of ac voltage signal conversion*/

             Delay(20);
      if(conbyte==0xd0)                 /*ac current signal conversion*/
      {
            for(x=0;x<5;x++)
            {
            SCLK=1;
            Delay(20);                  /*time delay 10ms*/
                  DIN=MSBRA;
            SCLK=0;
            Delay(20);
                  B=B<<1;
            }
            Delay(20);
            SCLK=0;
            Delay(20);
            SCLK=1;
            Delay(20);
            SCLK=0;
            Delay(20);
      for(x=0;x<8;x++)
      {
            SCLK=1;
      Delay(20);
            LSBRA=DOUT;
            SCLK=0;
      Delay(20);
      B=B<<1;
            Delay(20);
      }
      queti=B;
            lcdcmd(0x81);
            Delay(10);
            n=(float)B;
      for(m=0;n!=0x00;m++)
   {
            rema=(char)n%10;
      y[m]=rema+0x30;
            Delay(20);
            n=n/10;
   }
            y[m]='\0';
        add=((y[1]-48)*2)+((y[2]-48)*2*10);
        e1=add/10;
        f[0]=e1+48;
        f[1]='.';
        r1=add%10;
        f[2]=r1+48;
        f[3]='\0';

        lcddata("suply curr=");
        lcddata(f);
        lcddata("A");
            Delay(500);
}                                        /*end of ac current signal conversion*/

                  Delay(500);
```

```
if(conbyte==0xe0)                /*dc voltage signal conditioning*/
{
       for(x=0;x<5;x++)
       {
      SCLK=1;
      Delay(20);                 /*time delay 20ms*/
            DIN=MSBRA;
      SCLK=0;
      Delay(20);
            B=B<<1;
      }
            Delay(20);
            SCLK=0;
            Delay(20);
            SCLK=1;
            Delay(20);
            SCLK=0;
            Delay(20);
      for(x=0;x<8;x++)
      {
            SCLK=1;
      Delay(20);
            LSBRA=DOUT;
            SCLK=0;
      Delay(20);
      B=B<<1;
            Delay(20);
      }
queti=B;
      lcdcmd(0x81);
      Delay(10);
      n=(float)B;
      for(m=0;n!=0x00;m++)
      {
            rema=(char)n%10;
      y[m]=rema+0x30;
            Delay(20);
            n=n/10;
      }
            y[m]='\0';

 add=(123*(y[1]-48))+((y[2]-48)*123*10);
  e1=add/1000;
  f[0]=e1+48;
  r1=add%1000;
  e2=r1/100;
  f[1]=e2+48;
  r2=r1%100;
  q2=r2/10;
  f[2]=q2+48;
  f[3]='\0';

            lcddata("armat volt=");
            lcddata(f);
            lcddata("V");
            Delay(500);
 }                               /*end of dc voltage signal conversion*/
             Delay(50);
if(conbyte==0xf0)                /*dc current signal conversion*/
{
```

```
            for(x=0;x<5;x++)
            {
            SCLK=1;
            Delay(20);                  /*time delay 10ms*/
                  DIN=MSBRA;
            SCLK=0;
            Delay(20);
            B=B<<1;
       }
            Delay(20);
            SCLK=0;
                  Delay(20);
            SCLK=1;
            Delay(20);
            SCLK=0;
            Delay(20);
      for(x=0;x<8;x++)
      {
            SCLK=1;
      Delay(20);
            LSBRA=DOUT;
            SCLK=0;
      Delay(20);
      B=B<<1;
            Delay(20);
      }
      queti=B;
            lcdcmd(0x81);
            Delay(10);
            n=(float)B;
    for(m=0;n!=0x00;m++)
   {
            rema=(char)n%10;
      y[m]=rema+0x30;
            Delay(20);
            n=n/10;
   }
            y[m]='\0';
        add=((y[1]-48)*2)+((y[2]-48)*2*10);
        e1=add/10;
        f[0]=e1+48;
        f[1]='.';
        r1=add%10;
        f[2]=r1+48;
        f[3]='\0';
          lcddata("armatr cur=");
            lcddata(f);
            lcddata("A");
            Delay(500);
      }
                  /*end of dc current signal conversion*/
             Delay(50);
             CS=1;
               }
            conbyte=0xc0;               /*end of for loop*/
}
                  /*end of while loop*/
Delay(10);
 }
                  /*end of main*/
```

```
      void lcdcmd(unsigned char value)
       {
         P1=value;
         rs=0;
         rw=0;
         en=1;
         Delay(1);
         en=0;
         return;
       }

      void lcddata(char *value)
       {
       int i;
            for(i=0;value[i]!='\0';i++)
         {
         P1=value[i];
         rs=1;
         rw=0;
         en=1;
         Delay(1);
         en=0;
}
return;}
```



Chapter11:Industrial Timer

TIMER PROGRAM

```
 #include<reg51.h>
sbit start1 =P3^0;
sbit stop1  =P3^1;
sbit up1    =P3^2;
sbit down1  =P3^3;
sbit reset1 =P1^7;
sbit led=P3^7;
sbit store1=P1^6;
sbit password=P3^6;


void delay();
void delay1();

void first();

unsigned int n,m,k,l,s;
int q,w,i,j,z,x;
 unsigned char a,b;

  void delay()
{     int i,j;
            for(i=0;i<=178;i++)
            for(j=0;j<1000;j++);
}

 void delay1()
{     int i,j;
            for(i=0;i<=50;i++)
            for(j=0;j<500;j++);
}


 void main()
{
                  led=0;

              P1=0x00;
            P2=0x00;
                  if(password==0)
            first();
            }

void first()
{
begin:
              if(up1==0)
             goto up;
              if(start1==0)
             goto start;
              if(stop1==0)
             goto stop;
              if(reset1==1)
             goto reset;
              if(down1==0)
             goto down;
```

```
              else
             goto begin;

stop:
           led=0;
         if(store1==1)
         goto store;
              if(reset1==1)
          goto reset;
           else
           goto stop;



 reset:
       while(1)
         {
           n=0;
           i=0;
           P1=0x00;
           P2=0x00;
           led=0;
           if(store1==1)
         goto store;
         if(start1==0)
           goto begin;
           }
 up:
           a=P0;

           m=(int)a;

          n=n+1;

           if(n>9)
           {
           s=m+1;
           n=0;
           if(m>9)
           goto stop;
           if(s>m)
           {
           m=s;
          P2=m;
      }
            }

          if(store1==1)
           goto store;

          delay1();
           P2=m;
           P1=n;

           q=n;
           w=m;
           up1=1;
           delay1();

          goto begin;
```

```
down:   q=P1;
        w=P2;
            q=q-1;
            if(q<0)
            {
            w=w-1;
            q=9;
            if(w<0)

            {
            q=0;
            w=0;
            }
            }
            P1=q;
            P2=w;
            if(w==0&&q==0)
            goto stop;

            delay1();
            if(store1==1)
         goto store;
            down1=1;
            goto begin;

 start:
            k=m;
            l=n;
            for(i=k;i>=0x00;i--)
            {
             led=1 ;


            for(j=l;j>=0x00;j--)
            {

            led=1;
            P1=j;
            P2=i;
            delay();
            z=P1;x=P2;
          if(P1==0&&P2==0)
            goto reset;

            if(i!=0x00&&j==0x00)
         {
          l=0x09;
            break;
            }
            while(z==0&&x==0)
            goto reset;
            if(stop1==0)
            goto stop;} }
            start1=1;
             goto begin;

 store:
            P2=m;
                  P1=n;
```

```
                    store1=0;
                    goto begin;
  }
```

PASSWORD PROGRAM

```
#include<reg51.h>
#include<string.h>
#define COL P2
#define ROW P0
sbit rs = P3^0;
sbit rw = P3^1;
sbit e  = P3^2;
void keyboard_reading(char );
void keyboard_key_press();
sbit access=P3^6;

char temporary_storage[6];
char temp_store[3];
int i;
char ch,l;
unsigned char a,b;

char master_password[6]="11111";


unsigned char keys[4][3]={ '1','2','3',

                           '4','5','6',

                           '7','8','9',

                           '*','0','#',};
unsigned char keyboard[4][3]={ '1','2','3',

                               '4','5','6',

                               '7','8','9',

                               '*','0','#',};

unsigned char a,b;

void lcdcmd(unsigned char);
void lcddata(unsigned char);

void delay(int);
 char *name1="ENTER PASSWORD";
 char *name2="ACCESS GRANTED";
 char *name3="ENTER 1ST DIGIT";
 char *name4="SET THE TIMER";
 char *name5="ACCESS DENIED";
   char *name6="PLZ TRY AGAIN";
   char *name7="PASSWORD:";
 int q=0;w=0;g=0;r=0;t=0;y=0;u=0;


void main()
{              access=1;
```

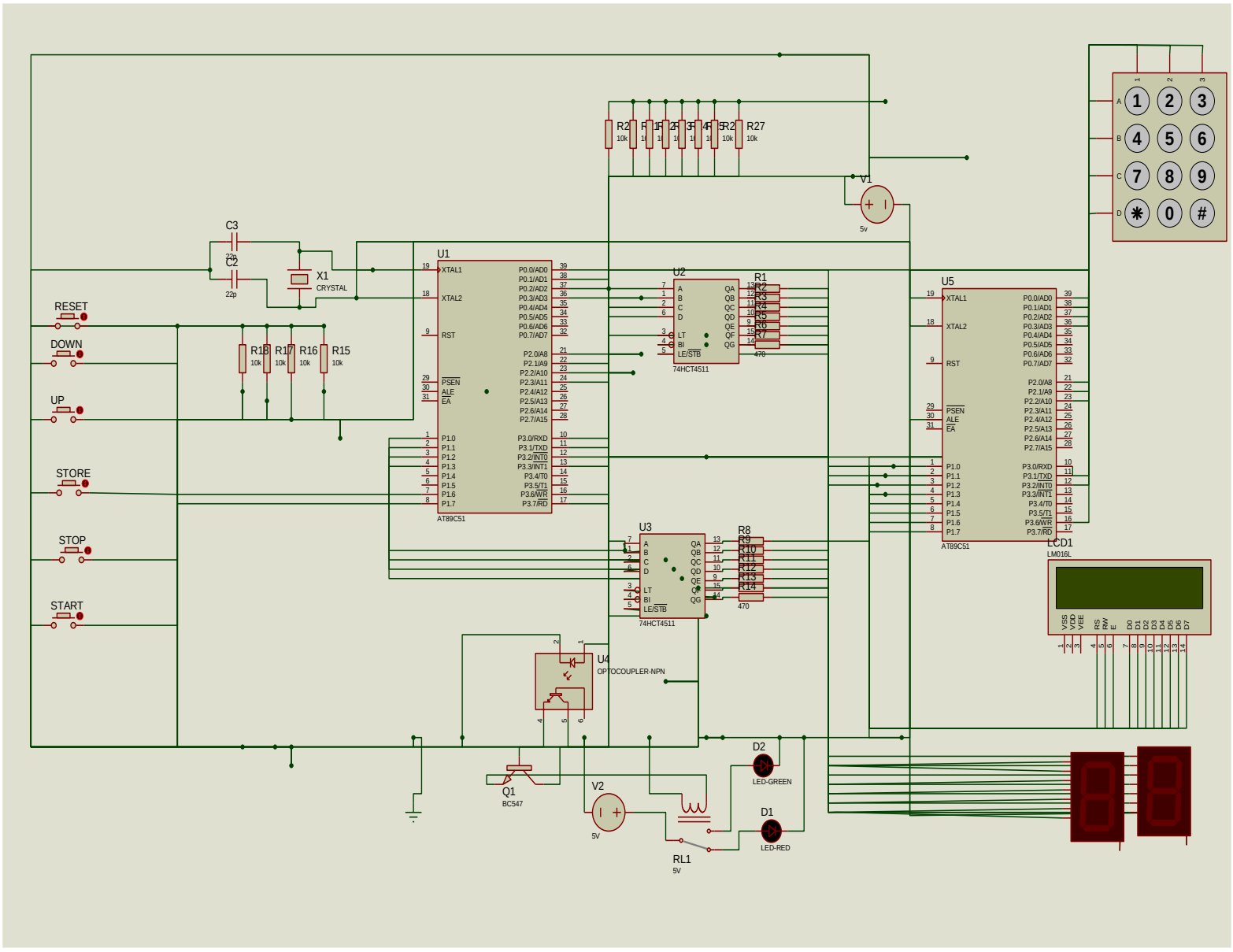
```
                lcdcmd(0x38);
      lcdcmd(0x01);
      lcdcmd(0x0e);
      lcdcmd(0x14);

        while(name1[q]!='\0')
     {
lcddata(name1[q++]);

      }

         delay(100);
            ch=0;
             keyboard_key_press();


      goto check;

check:

      if(strcmp(master_password,temporary_storage)==0)
{
            lcdcmd(0x01);
      lcdcmd(0x81);
            delay(25);
                  lcddata(' ');

               while(name2[w]!='\0')
     {
lcddata(name2[w++]);
  delay(1);
      }

            delay(100);
            lcdcmd(0x01);

             while(name3[g]!='\0')
     {
lcddata(name3[g++]);
  delay(1);
      }

            delay(100);
            lcdcmd(0x01);


              while(name4[r]!='\0')
     {
lcddata(name4[r++]);
  delay(1);
      }

         delay(100);

            access=0;
}//if



      else
```

```
{
             for(i=1;i<=3;i++)
       {
             lcdcmd(0x01);
              while(name5[t]!='\0')
      {
lcddata(name5[t++]);
  delay(1);
       }
             delay(100);
             lcdcmd(0x01);
          while(name6[y]!='\0')
      {
lcddata(name6[y++]);
  delay(1);
       }
             delay(1);

              lcdcmd(0x01);

              while(name7[u]!='\0')
      {
lcddata(name7[u++]);
  delay(1);
       }


             lcdcmd(0x01);
       ch=0;
        keyboard_key_press();
               goto check;



     }//for

 }//else

keypad:
   lcdcmd(0x01);
       P2 = 0x07;
       y:  P0 = 0x00;
           delay(8);
             if( P2 == 0x07)
             goto y;
       delay(8);
       if( P2 == 0x07)
       goto y;
       a = P2;

       P0 = 0x0e;
       if( P2 != 0x07)
       goto row1;
       P0 = 0x0d;
       if( P2 !=0x07)
       goto row2;
       P0 = 0x0b;
       if( P2 !=0x07)
       goto row3;
       P0 = 0x07;
```

```
      if( P2 != 0x07)
      goto row4;
      else
      goto y;

      row1:  if( P2 == 0x06)
             lcddata( keys[0][0]);
               else if( P2 == 0x05)
               lcddata( keys[0][1]);
               else if( P2 == 0x03)
               lcddata( keys[0][2]);
               goto y;

      row2:  if( P2 == 0x06)
             lcddata( keys[1][0]);
               else if( P2 == 0x05)
               lcddata( keys[1][1]);
               else if( P2 == 0x03)
               lcddata( keys[1][2]);
               goto y;

      row3:  if( P2 == 0x06)
             lcddata( keys[2][0]);
               else if( P2 == 0x05)
               lcddata( keys[2][1]);
               else if( P2 == 0x03)
               lcddata( keys[2][2]);
               goto y;

      row4:  if( P2 == 0x06)
             lcddata( keys[3][0]);
               else if( P2 == 0x05)
               lcddata( keys[3][1]);
               else if( P2 == 0x03)
               lcddata( keys[3][2]);

      goto y;
}

//main




   void keyboard_reading(char value)
{
   if(ch==0)
  {
       lcdcmd(0x01);
             while(name7[u]!='\0')
     {
lcddata(name7[u++]);

      }
 // lcdcmd(0x89);
  }
   temporary_storage[ch]=value;
    ch++;
      delay(5);
      lcddata('*');
```

```
   if(ch==5)
    temporary_storage[ch]='\0';
   return;
}

   void keyboard_key_press()
{
 unsigned char colloc,rowloc;
 int m;



 for(m=0;m<=4;m++)
 {
  do
   {

     ROW=0x00;
       colloc=COL;
       colloc&=0x0f;
       }
        while(colloc!=0x0f);
  do
   {
    do
     {
      delay(2);
      colloc=COL;
      colloc&=0x0f;
      }while(colloc==0x0f);
      delay(2);
      colloc=COL;
      colloc&=0x0f;
      } while(colloc==0x0f);

    while(1)
     {
       ROW=0xfe;
       colloc=COL;
       colloc&=0x0f;

       if(colloc!=0x0f)
        {
          rowloc=0;
          break;
         }
         ROW=0xfd;
       colloc=COL;
       colloc&=0x0f;
       if(colloc!=0x0f)
        {
          rowloc=1;
          break;
         }
        ROW=0xfb;
       colloc=COL;
       colloc&=0x0f;

       if(colloc!=0x0f)
        {
```

```
          rowloc=2;
          break;
         }
        ROW=0xf7;
       colloc=COL;
       colloc&=0x0f;
        rowloc=3;
          break;
       }
        if(colloc==0x0e)
         keyboard_reading(keyboard[rowloc][0]);
        else if(colloc==0x0d)
         keyboard_reading(keyboard[rowloc][1]);
         else if(colloc==0x0b)
         keyboard_reading(keyboard[rowloc][2]);
          else
         keyboard_reading(keyboard[rowloc][3]);

        }
}


void lcdcmd(unsigned char value)
{   rs = 0;
    rw = 0;
      e = 1;
      delay(1);
      P1 = value;
      e = 0;
}

void lcddata(unsigned char value)
{   rs = 1;
    rw = 0;
      e = 1;
      delay(1);
      P1 = value;
      e = 0;
}

void delay(int time)
{  int i,j;
   for( i=0;i<=time;i++)
   for(j=0;j<=1275;j++);
}
```


C3
22p
C2
22p
X1
CRYSTAL
U1
AT89C51
XTAL1
XTAL2
RST
PSEN
ALE
EA
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
R18 10k
R17 10k
R16 10k
R15 10k
RESET
DOWN
UP
STORE
STOP
START
U2
74HCT4511
A
B
C
D
LT
BI
LE/STB
QA
QB
QC
QD
QE
QF
QG
R1
R2
R3
R4
R5
R6
R7
470
U3
74HCT4511
R8
R9
R10
R11
R12
R13
R14
470
U4
OPTOCOUPLER-NPN
Q1
BC547
V1
5v
V2
5V
RL1
5V
D2
LED-GREEN
D1
LED-RED
U5
AT89C51
LCD1
LM016L
VSS
VDD
VEE
RS
RW
E
D0
D1
D2
D3
D4
D5
D6
D7
A
B
C
D
1
2
3
4
5
6
7
8
9
*
0
#

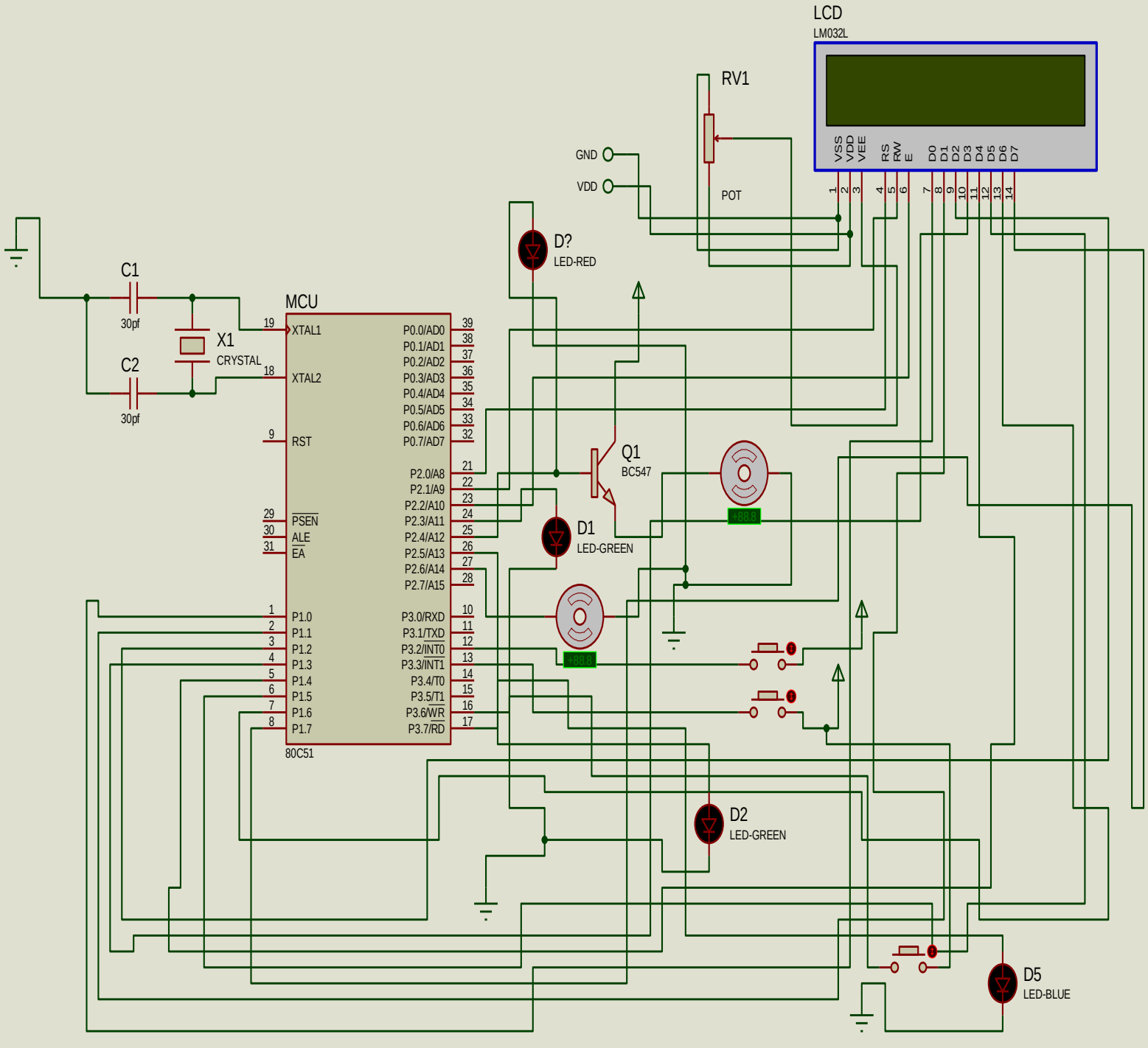
```
                              Chapter 11
//   TITLE:       POWER SAVING SYTEM FOR SHOPPING MALLS
//     LCD DATA PORT:: P0
//                     RS:: P2.7
//                     RW:: P2.6
//                     EN:: p2.5

#include "reg51.h"

sfr ldata               = 0X90;
sbit rs                 = P2^0;
sbit rw                 = P2^1;
sbit en                 = P2^2;

sbit s1_entry   = P3^2;
sbit s1_exit      = P3^3;

sbit init_lt1   = P2^3;
sbit init_fan   = P2^4;


sbit init_lt2   = P2^5;
sbit init_fan2  = P2^6;

sbit s1_pir       = P3^6;
sbit init_buzzer= P3^7;


unsigned int count1=0,count2=0,count3,values;
int cnt();
char cont[14]="CNT IN HALL=";

void delay1(unsigned int dtime)
{
   unsigned int i,j;
   for (i=0; i<=dtime; i++)
     for (j=0; j<=10; j++);
}
 void lcdcmd(unsigned char value)
{
   ldata=value;
   rs=0;
   en=1;
   rw=0;
   delay1(1000);
   en=0;
   return;
}
void lcddata(unsigned char value)
{
   ldata=value;
   rs=1;
   en=1;
   rw=0;
   delay1(1000);
```

```
   en=0;
   return;
}
/*int counting(unsigned int count2)
{
int count3=count2-10;
return count3;
}
      */
void count()
 {    int a;

      for(a=0;a<=13;a++)
        {
        lcddata(*(cont+a));//COUNT=
        }
 }
unsigned int check()
   {
   if(count2==5)
   values=1;
   return values;
   }

void main()
{
      char x,d1,d2,d3;
      int a,i,j;


      char disp= '0';
    char string[16]= "POWER SAVING FOR";
      char string1[14]="SHOPPING MALL";
      char string2[18]="INITIALLY LIGHT ON";
      char string3[8]="FAN ON";
      char string4[8]="LED ON";
      char string5[11]="PIR ENABLED";
      char string6[13]="PIR DISABLED";
      char space[2]="  ";

      init_lt1 = 0;
      init_lt2 = 0;
      init_fan = 0;
      init_fan2= 0;
      s1_entry = 0;
      s1_exit  = 0;
      s1_pir=1;
      init_buzzer=0;
      //    delay1(100);
            lcdcmd(0X38);
      //    delay1(100);
            lcdcmd(0X0E);
      //    delay1(100);
            lcdcmd(0X01);
      //    delay1(100);
            lcdcmd(0X06);
      //    delay1(100);
            lcdcmd(0X86);
      //    delay1(100);                                                    1
```

```
lcdcmd(0X01);




for(i=0;i<=15;i++)
{
lcddata(*(string+i)); //POWER SAVING FOR SHOPPING MALL
}
lcdcmd(0XC0);//next line command
lcddata(*(space));

for(j=0;j<=13;j++)
{
lcddata(*(string1+j));//GIVE SPACE IN 2ND ROW
}
lcdcmd(0X01);//clears the lcd

for(a=0;a<=17;a++)
 { lcddata(*(string2+a));}     //INITIALLY LIGHT ON
  lcdcmd(0XC0);   //NEXT LINE COMMAND
             for(a=0;a<=15;a++)
              {
              lcddata(*(cont+a));   //COUNT=
              }
             lcddata(disp);   //0
             init_lt1 = 1; init_fan = 0; init_lt2 = 0;rs=0;delay1(100);
             lcdcmd(0X01);//CLEARS THE LCD


      count();

/*   while(!(s1_pir))
      {
      init_buzzer=1;
      for(i=0;i<=10;i++)
      lcddata(*(string5+i));
      }
 */


   while(1)//CONTINOUS LOOP
   {
      while(!(s1_entry || s1_exit));
      count1=cnt();



           x  = count1/10;
        d1 = count1%10;
        d2 = x % 10;
        d3 = x / 10;
           count2 = 100*d3+10*d2+d1;

           if (count2>=0 && count2<=5)
            {
            init_lt1 = 1; init_fan = 0; init_fan2 = 0; init_lt2 = 0;rs=0;
```

```
             lcddata(disp+count2);//count2 value
                  lcdcmd(0X10); //BACK SPACE

             delay1(100);
             }

            if (count2>=5 && count2<10)
            {

             check();
             if(values==1)
             {
              lcdcmd(0X01);
              for(a=0;a<=6;a++)
               {
                lcddata(*(string3+a));
             }
              values=0;
              lcdcmd(0X01);
             }

             lcdcmd(0x80);
             for(a=0;a<=15;a++)
             {
              lcddata(*(cont+a));
             }
             init_lt1 = 1; init_fan = 1;init_fan2 = 0; init_lt2 = 0;rs=0;

             lcddata(disp+count2);
             lcdcmd(0X10);
             delay1(100);


            }

            if(count2>=10 && count2<15)
               {
                /*counting(count2);
                  lcddata(disp+count3);
                lcddata(disp);*/
               }
            if (count2>=15 && count2<25)
            { lcdcmd(0X01);for(a=0;a<=5;a++)  {lcddata(*(cont+a));} init_lt1  =
1;init_fan2 = 1;  init_fan = 1;  init_lt2 =
0;delay1(100);lcddata(disp+count2);delay1(100);}

            if (count2>=25) n
            {lcdcmd(0X01); for(a=0;a<=5;a++)  {lcddata(*(cont+a));}  init_lt1  =
1;    init_fan = 1;  init_lt2 = 1; init_fan2 =
1;delay1(100);lcddata(disp+count2);delay1(100);}}}
int cnt()
{
      if (s1_entry == 1)
      {
   delay1(10);
            count1 = count1 + 1;
            delay1(10);
      }
      if (s1_exit == 1)
```

```
        {
                delay1(10);
                count1 = count1 - 1;
                delay1(10); }
return count1;}
```

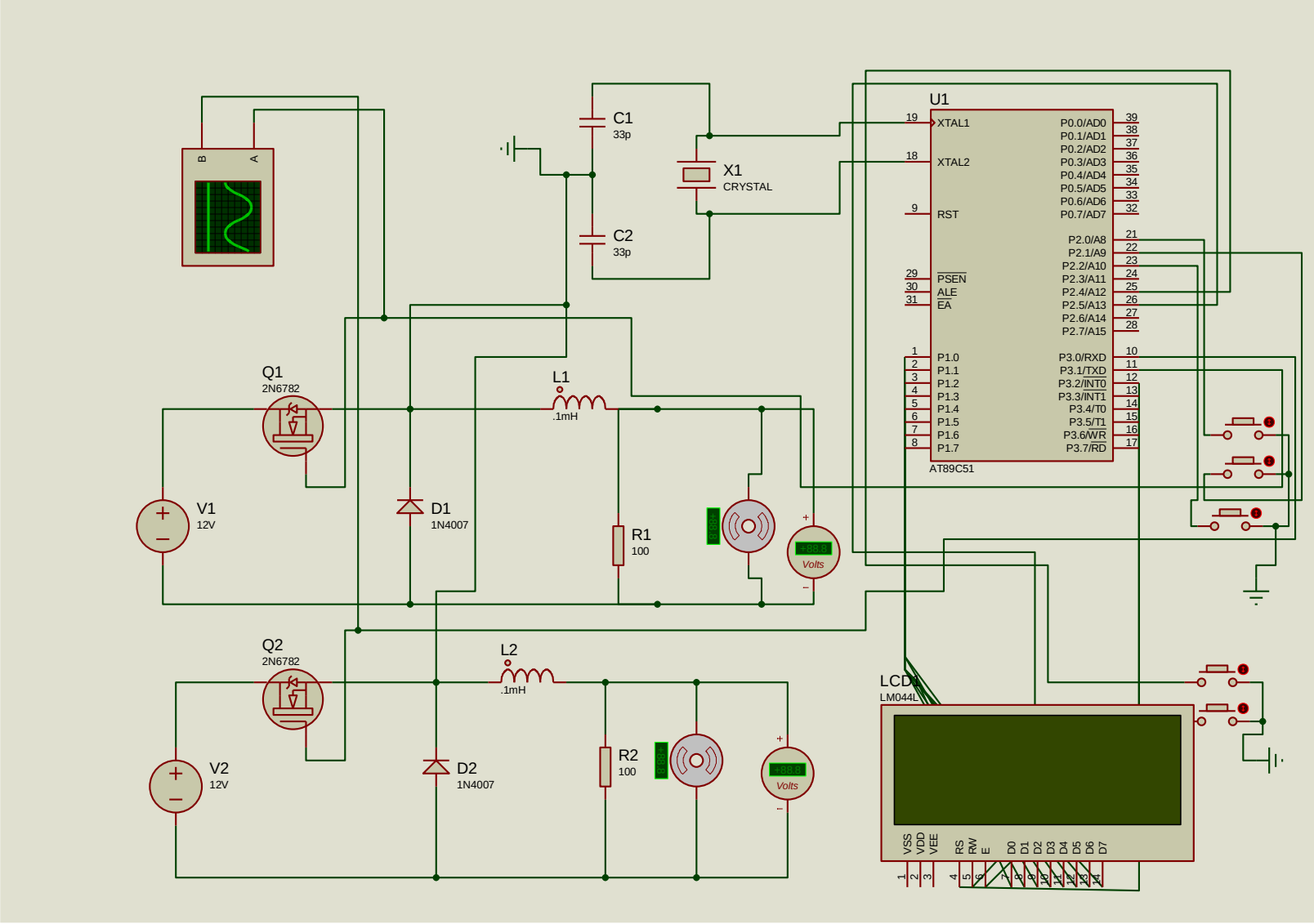



Chapter 11:DC MOTOR INTERFACING

```
#include<reg51.h>
sfr ldata=0x90;
sbit rs=P3^2;
sbit rw=P3^3;
sbit en=P3^4;
sbit a=P2^0;
sbit b=P2^1;
sbit c=P2^2;
sbit d=P2^3;
sbit e=P2^4;
sbit f=P2^5;
sbit w=P3^0;
sbit s=P3^1;
void delay(unsigned int itime);
void lcdcmd(unsigned char value);
void lcddata(char *value);

void delay(time)
{
int i,j;
for(i=0;i<=time;i++)
{
for(j=0;j<60;j++);
}
}
main()
{
int m,n,k,g,z;
P1=0x00;
P3=0x00;
m=19;
n=19;
g=19;
z=19;
k=1;
if(a==0)
{
while(1)
{
w=1;
delay(n);
w=0;
delay(m);

if(b==0)
{
n=n+k;
m=m-k;
delay(100);
}
else if(c==0)
{
n=n-k;
```

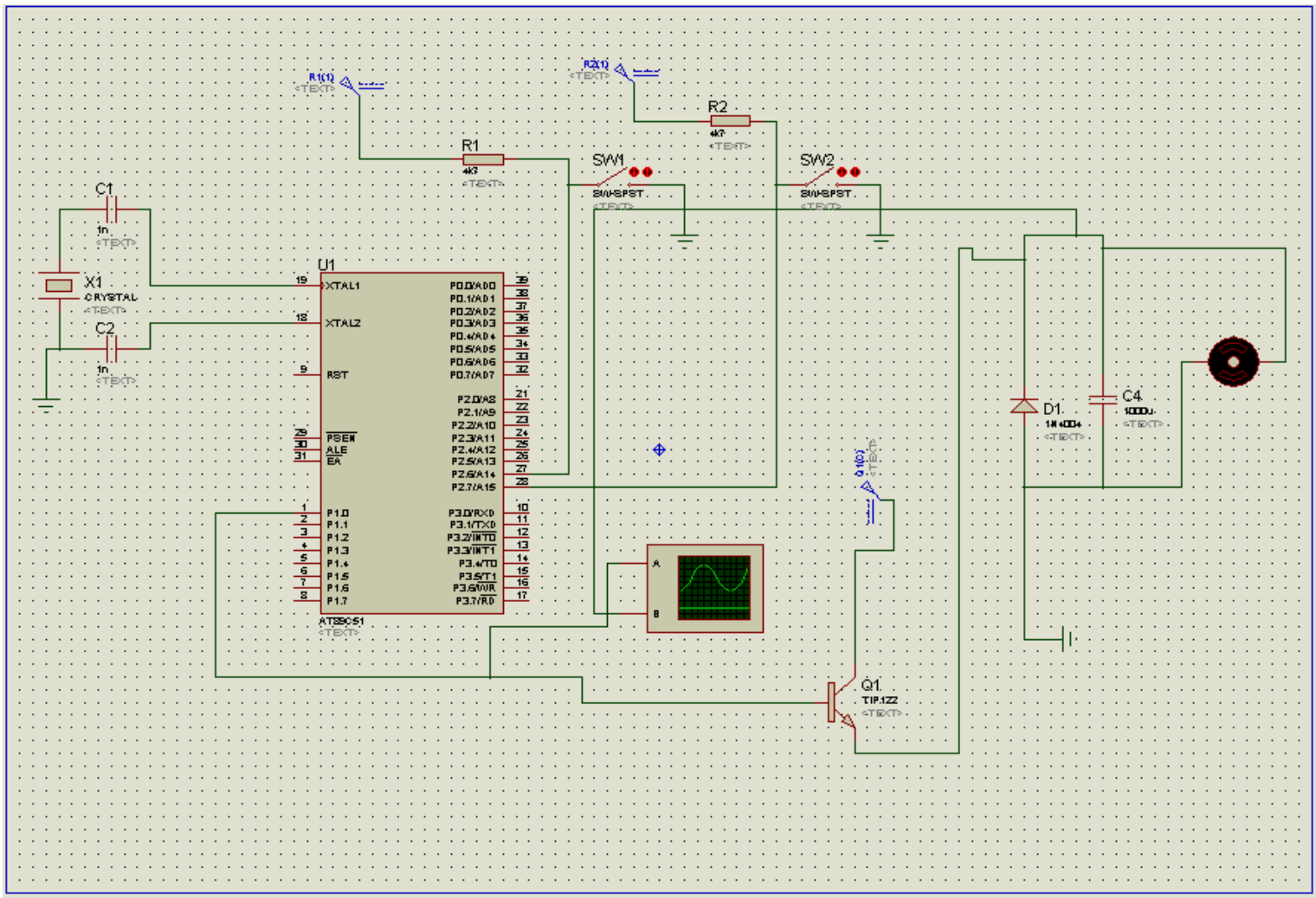
```
m=m+k;
delay(100);
}
s=1;
delay(g);
s=0;
delay(z);

if(e==0)
{
g=g+k;
z=z-k;
delay(100);
}
else if(f==0)
{
g=g-k;
z=z+k;
delay(100);
}
else if(n==g)
{
 lcdcmd(0x38);

  lcdcmd(0x0e);

lcdcmd(0x01);

lcdcmd(0x06);

lcdcmd(0x80);

lcddata("synchronous");
}
else if(n!=g)
{
 lcdcmd(0x38);

  lcdcmd(0x0e);

lcdcmd(0x01);

lcdcmd(0x06);

lcdcmd(0x80);

lcddata("asynchronous");
}
else
{
continue;
}
}
}
 }



 void lcdcmd(unsigned char value)
 {
```

```
    ldata=value;
    rs=0;
    rw=0;
    en=1;
    delay(1);
    en=0;
    return;
  }

void lcddata(char *name1)
  {
     int i;
    for(i=0;name1[i]!='\0';i++)
    {
    ldata=name1[i];
    rs=1;
    rw=0;
    en=1;
    delay(1);
    en=0;
    }
    return;
  }
```


U1
AT89C51
XTAL1
XTAL2
RST
PSEN
ALE
EA
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
C1
33p
C2
33p
X1
CRYSTAL
Q1
2N6782
Q2
2N6782
L1
.1mH
L2
.1mH
V1
12V
V2
12V
D1
1N4007
D2
1N4007
R1
100
R2
100
Volts
Volts
LCD1
LM044L
VSS
VDD
VEE
RS
RW
E
D0
D1
D2
D3
D4
D5
D6
D7

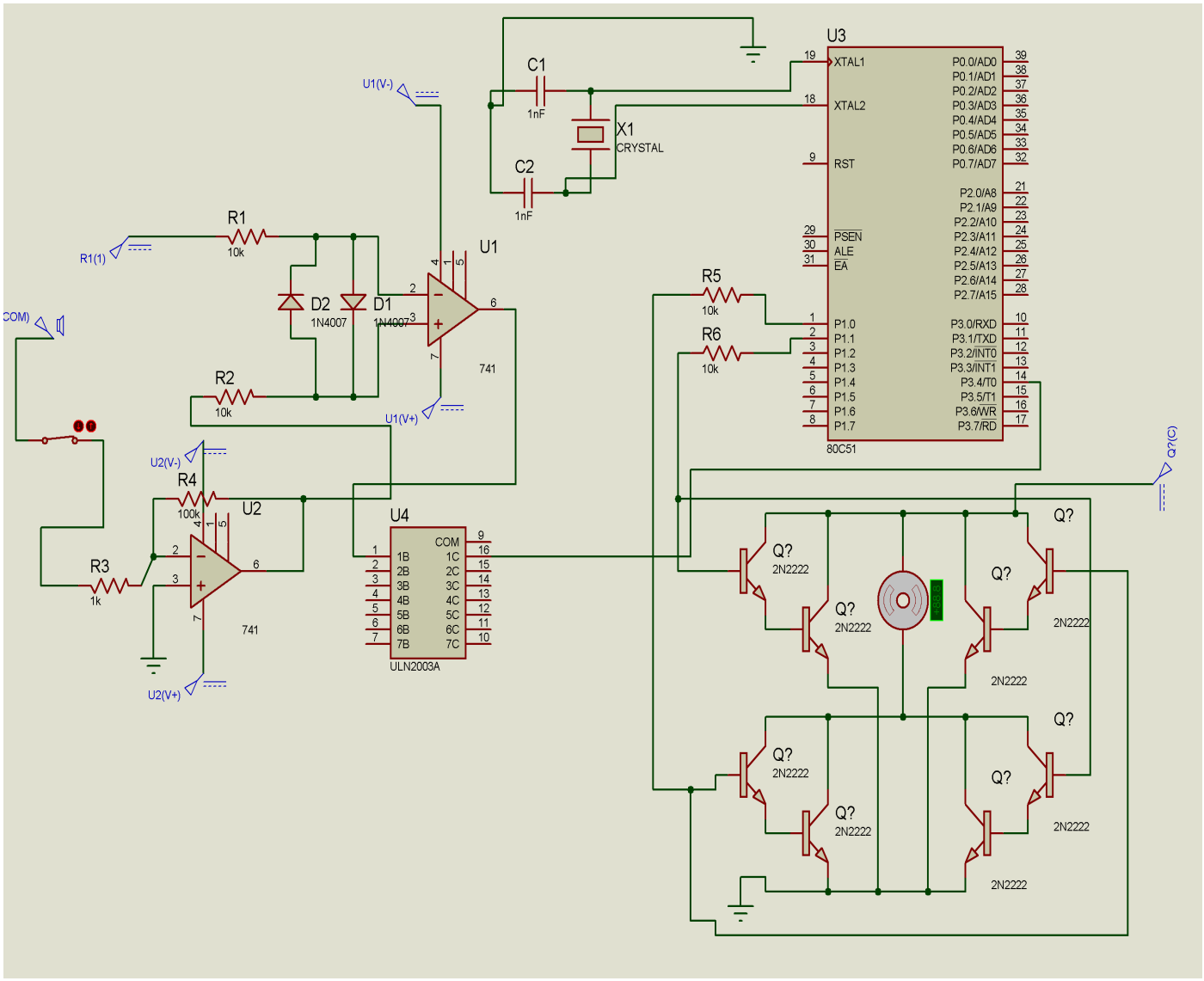
Chapter12: DC Motor speed control

```
#include<reg51.h>
sbit SW_1 = P2^6;
sbit SW_2 = P2^7;
sbit MTR = P1^0;
 void main()
{
int x;
while(1)
{
if(SW_1==0 & SW_2==0)
{
MTR=1;
for(x=0;x<150;x++);
MTR=0;
for(x=0;x<600;x++);
}
if(SW_1==0 & SW_2==1)
{
MTR=1;
for(x=0;x<300;x++);
MTR=0;
for(x=0;x<300;x++);
}
if(SW_1==1 & SW_2==0)
{
MTR=1;
for(x=0;x<600;x++);
MTR=0;
for(x=0;x<150;x++);
}
if(SW_1==1 & SW_2==1)
{
MTR=1;
}
}
}
```


R1(1)
R2(1)
R2
4k7
R1
4k7
SW1
SW-SPST
SW2
SW-SPST
C1
1n
X1
CRYSTAL
C2
1n
U1
XTAL1
XTAL2
RST
PSEN
ALE
EA
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
AT89C51
D1
C4
Q1
TIP122

Chapter 13: Sound activated robot

```
#include "reg51.h"
sbit sw1=P0^0;
sbit sw2=P0^1;
sbit ind1=P1^0;
sbit ind2=P1^1;
sbit r=P3^4;
void main()
{
  int x;
  TMOD=0x15;
  r=1;
  TL0=0;
  TH0=0;
  TR0=1;
 for(x=1;x<=28;x++)
  {
   TL1=0;
   TH1=0;
   TR1=1;
    while(TF1==1)
   {
    TF1=0;
    TR1=0;
   }
  }

ind1=0;
ind2=0;
while(1)
{
if(TL0<0xFF&&TL0>0xF0)
ind2=1;
else
ind2=1;
}

}
```


U3
C1
1nF
C2
1nF
X1
CRYSTAL
XTAL1
XTAL2
RST
PSEN
ALE
EA
80C51
U1(V-)
U1
741
U1(V+)
R1
10k
R1(1)
D2
1N4007
D1
1N4007
R2
10k
R5
10k
R6
10k
COM)
U2(V-)
U2
R4
100k
R3
1k
741
U2(V+)
U4
ULN2003A
Q?
2N2222
Q7(C)

Chapter 14:Temperature monitoring

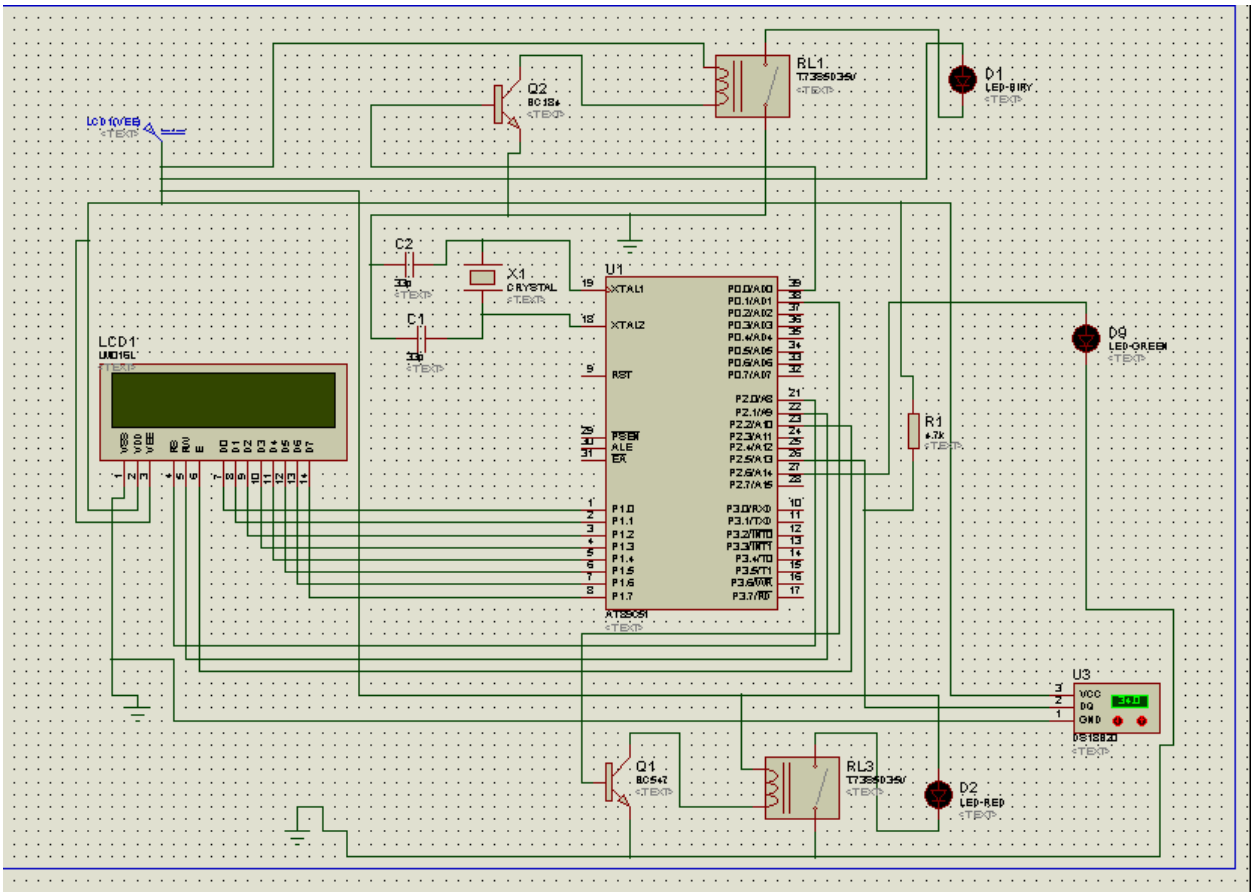
```
#include<reg51.h>
sfr ldata=0x90;
sbit rs=P2^0;
sbit rw=P2^1;
sbit en=P2^2;
sbit DQ=P2^5;
sbit output=P2^6;
sbit rel1=P0^0;
sbit rel2=P0^1;
void lcdcmd(unsigned char value);
void lcddata_number(unsigned char value);
void lcddata(unsigned char *value);
void lcddelay(unsigned int time);
bit reset();
void delay(int us);
bit readbit(void);
void writebit(bit dbit);
unsigned char readbyte(void);
void writebyte(unsigned char dout);
void readtemp();
unsigned char mydata1,mydata2,mydata3;


 void main()
 {

  lcdcmd(0x38);
  lcddelay(10);

  lcdcmd(0x0e);
  lcddelay(10);

  lcdcmd(0x01);
  lcddelay(10);

  lcdcmd(0x06);
  lcddelay(10);

 lcddata("temperature");


while(1)
{

 readtemp();
if(mydata3<=35)
 rel1=0x01;
else
```

```
 rel1=0x00;
if(mydata3>35&mydata3<=45)
 rel2=0x01;
else
 rel2=0x00;

  }
 }

bit reset()
{
  bit presense;
  DQ=0;
  delay(29);
  DQ=1;
  delay(3);
  presense=DQ;
  delay(25);
  output=presense;
  return(presense);
   }



bit readbit(void)
{
unsigned char i=0;
DQ=0;
DQ=1;
for(i=0;i<3;i++);
return(DQ);
}

void writebit(bit Dbit)
{
unsigned char i=0;
DQ=0;
DQ=Dbit?1:0;
delay(5);
DQ=1;
}

unsigned char readbyte(void)
{
unsigned char i;
unsigned char din=0;
for(i=0;i<8;i++)
{
din|=readbit()?0x01<<i:din;
delay(6);
```

```
}
//B=din;
return(din);
}

void writebyte(unsigned char dout)
{
unsigned char i;
for(i=0;i<8;i++)
{
writebit((bit)(dout&0x01));
dout=dout>>1;
}
delay(5);
}

void readtemp()
{

 unsigned char n;
 unsigned char buff[9];
 EA=0;
 reset();
 writebyte(0xcc);
 writebyte(0x44);
 while(readbyte()==0xff);
 reset();
 writebyte(0xcc);
 writebyte(0xbe);
 for(n=0;n<9;n++)
 {

  buff[n]=readbyte();
  delay(100);

 }
  mydata1=buff[0];
  delay(10);
  mydata1=mydata1>>4;

  mydata2=buff[1];
  delay(10);
  mydata2=mydata2<<4;
  mydata3=mydata2|mydata1;

lcddata_number(mydata3);
//lcddelay(1);
delay(2500);
EA=1;
```

```
}

 void lcdcmd(unsigned char value)
 {
  ldata=value;
  rs=0;
  rw=0;
  en=1;
  lcddelay(1);
  en=0;
  return;
 }


 void lcddata_number(unsigned char value)
 {
   char data1;
   lcdcmd(0x8c);
   delay(1000);
   B=value;
data1=value%10;
value=value/10;
if(value>9)
{
ldata=1+0x30;

  for(;value!=0;)
  {
  data1 =value%10;
 ldata=data1+0x30;
 rs=1;
 rw=0;
 en=1;
 lcddelay(10);
 en=0;
 ldata=value%10+0x30;
 rs=1;
 rw=0;
 en=1;
 lcddelay(10);
 en=0;
 }
 }
else
 {
 ldata=value+0x30;
 lcddelay(10);
 rs=1;
 rw=0;
 en=1;
```

```
 lcddelay(10);
 en=0;}
ldata=data1+0x30;
 rs=1;
 rw=0;
 en=1;
 lcddelay(10);
 en=0;
 ldata='\0';
 rs=1;
 rw=0;
 en=1;
 lcddelay(10);
 en=0;
 return;

}
void lcddata(unsigned char *value)
{
 char i;
 for(i=0;value[i]!='\0';i++)
{
ldata=value[i];
 rs=1;
 rw=0;
 en=1;
 lcddelay(10);
 en=0;
 }
 return;
}
 void lcddelay(unsigned int time)
 {
 unsigned int i,j;
 for(i=0;i<=time;i++)
 {
 for(j=0;j<=1000;j++);
 }}


void delay(int us)
{
 int i;
 for(i=0;i<us;i++);
}
```


RL1
D1
LED-BIBY
Q2
C2
X1
CRYSTAL
U1
C1
LCD1
D9
LED-GREEN
R1
U3
Q1
RL3
D2
LED-RED

Chapter 15: Displaying of time

```
;SET CORRECT MINUTE AND HOUR DATA  AT LINE NOS 120 and  122
RB0     EQU     000H    ; Select Register Bank 0
RB1      EQU      008H     ; Select Register Bank 1  ...poke to PSW
to use

;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%
;          PORT DECLERATION
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%

SDA  EQU  P3.4;SDA=PIN5
SCL  EQU  P3.5 ;SCL=PIN6

RTC_W   EQU    0D0H     ; SLAVE ADDRESS 1101 000 + 0 TO WRITE
RTC_R   EQU    0D1H     ; SLAVE ADDRESS 1101 000 + 1 TO READ

RS11      EQU P2.0   ;ASSIGNING  NAMES  TO  THE  PORT  PINS  AS
MENTIONED
RW11      EQU P2.1
EN        EQU P2.2
;Sfr LCDVAL =0X90




FLAGS     DATA   20H
LASTREAD  BIT    FLAGS.0
SQW    BIT    FLAGS.4
ACK       BIT    FLAGS.5
BUS_FLT   BIT    FLAGS.6
_2W_BUSY  BIT    FLAGS.7
BITCNT    DATA   21H
BYTECNT   DATA   22H
SECS      DATA   24H      ;   '   SECONDS STORAGE RAM
MINS      DATA   25H      ;   '   MINUTES   '     '
HRS       DATA   26H      ;   '   HOURS     '     '

COUNT       DATA     27H
SPEED       DATA     28H

VALUE_1   DATA 30H
VALUE_2     DATA    31H
VALUE_3     DATA    32H
VALUE_4     DATA    33H

VALUE_5     DATA    34H
```

```
VALUE_6     DATA    35H

SCL_HIGH  MACRO
          SETB   SCL       ; SET SCL HIGH
          JNB    SCL,$     ; LOOP UNTIL STRONG 1 ON SCL
          ENDM


;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%


     ORG     00H              ; Reset
     JMP     MAIN

;    ORG  000BH          ;Timer Interrupt0
;    JMP  REFRESH
MAIN:
     MOV PSW,#RB0        ; Select register bank 0
    MOV SP,#40H
    MOV SPEED,#00H
    MOV COUNT,#00H
       MOV SCON,#50H;// to select the mode2 and reciver enable
        MOV TMOD,#20H;// to select the timer1 in auto relod mode
        MOV TH1,#0FDH; // to get the 9600 baud rate
        SETB TR1;    // start the timer

; ***********************************************************
;         INITILIAZE RTC
; ***********************************************************
          SETB SDA            ; ENSURE SDA HIGH
        SCL_HIGH               ; ENSURE SCL HIGH
        CLR ACK                 ; CLEAR STATUS FLAGS
        CLR BUS_FLT
        CLR _2W_BUSY
         CLR SQW

         CALL OSC_CONTROL          ;Initilize the RTC

;((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
((((((((((((((
;         STORE THE TIME TO RTC CHIP
;((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
((((((((((((((

          LCALL SEND_START
        MOV A,#RTC_W              ; LOAD RTC WRITE COMMAND
         LCALL SEND_BYTE          ; SEND WRITE COMMAND
```

```
         MOV A,#08H                    ;  SET  DS1307  DATA  POINTER
TO BEGINNING
         LCALL SEND_BYTE            ; OF USER RAM 08H

         ACALL       SEND_STOP
         ACALL       SEND_START
         MOV         A,#RTC_R
         ACALL       SEND_BYTE
          ACALL       READ_BYTE
          ACALL       SEND_STOP
          CJNE       A,#41H, DOWN1
          JMP         START_PROGRAM


DOWN1:    LCALL SEND_START          ; SEND I2C START CONDITION

     MOV A,#RTC_W              ; LOAD RTC WRITE COMMAND
     LCALL SEND_BYTE           ; SEND WRITE COMMAND

     MOV A,#01H                    ;  SET  DS1307  DATA  POINTER  TO
MINUT ADDR
     LCALL SEND_BYTE
     MOV A,#22H                ;SET CORRECT MINUTES DATA HERE
     LCALL SEND_BYTE
    MOV A,#03H                 ;SET CORRECT HOUR DATA HERE
     LCALL SEND_BYTE

     LCALL SEND_STOP           ; SEND 2WIRE STOP CONTION*/

     LCALL SEND_START
   MOV A,#RTC_W                ; LOAD RTC WRITE COMMAND
     LCALL SEND_BYTE           ; SEND WRITE COMMAND

     MOV A,#08H                    ;  SET  DS1307  DATA  POINTER  TO
BEGINNING
     LCALL SEND_BYTE           ; OF USER RAM 08H
     MOV A,#41H                ; WRITE BYTE TO ENTIRE RAM SPACE
     LCALL SEND_BYTE


     LCALL SEND_STOP           ; SEND 2WIRE STOP CONTION*/

;*****************************************************************
**************
;         MAIN PROGRAM
;*****************************************************************
**************
START_PROGRAM:
     CALL READ_CLOCK
```

```
ACALL LCDINIT
MOV A,#080H
ACALL COMMAND
ACALL DELAY1
MOV R1,#26H
MOV A,@R1
ANL A,#0F0H
SWAP A
ADD A,#30H
ACALL DATA1
ACALL DELAY1
MOV VALUE_1,A
MOV A,#081H
ACALL COMMAND
ACALL DELAY1
MOV R1,#26H               ;GET HOUR AND DISPLAY
MOV A,@R1
ANL A,#0FH
   ADD A,#30H
   ACALL DATA1
ACALL DELAY1
MOV VALUE_2,A


MOV VALUE_3,A
MOV A,#082H
ACALL COMMAND
ACALL DELAY1
MOV A,#3AH
ACALL DATA1
ACALL DELAY1
MOV A,#083H
ACALL COMMAND
ACALL DELAY1
MOV R1,#25H
MOV A,@R1
ANL A,#0F0H
SWAP A
ADD A,#30H
ACALL DATA1
ACALL DELAY1


MOV A,#084H
ACALL COMMAND
ACALL DELAY1
MOV R1,#25H               ;GET MIN AND DISPLAY
MOV A,@R1
ANL A,#0FH
ADD A,#30H
```

```
      ACALL DATA1
      ACALL DELAY1
      MOV VALUE_4,A
      MOV A,#085H
      ACALL COMMAND
      ACALL DELAY1
      MOV A,#3AH
      ACALL DATA1
      MOV A,#086H
      ACALL COMMAND
      ACALL DELAY1
      MOV R1,#24H
      MOV A,@R1
      ANL A,#0F0H
      SWAP A
      ADD A,#30H
      ACALL DATA1
      ACALL DELAY1
      MOV VALUE_5,A

      MOV A,#087H
      ACALL COMMAND
      ACALL DELAY1
      MOV R1,#24H              ;GET SEC AND DISPLAY
      MOV A,@R1
      ANL A,#0FH
      ADD A,#30H
         ACALL DATA1
      ACALL DELAY1
      MOV VALUE_6,A

      ;MOV DPTR,#CMD      ;INITIALIZATION OF LCD COMMANDS
LCDINIT:CLR A
      MOV A,#38H
      ACALL COMMAND
      ACALL DELAY1
      MOV A,#0EH
      ACALL COMMAND
      ACALL DELAY1
      MOV A,#01H
      ACALL COMMAND
      ACALL DELAY1
      MOV A,#06H
      ACALL COMMAND
      ACALL DELAY1
         RET

COMMAND:  MOV P1,A
           CLR RS11
        CLR RW11
```

```
        CLR P2.2
        SETB P2.2
        ACALL DELAY1
        CLR P2.2
           ;    ACALL DELAY1
         RET

DATA1:     MOV P1,A
        SETB RS11
         CLR RW11
         CLR P2.2
         SETB P2.2
        ACALL DELAY1
         CLR P2.2
              ;ACALL DELAY1
         RET

DELAY1:     MOV R6,#255
HERE1:        MOV R7,#50
          DJNZ R7,$
          DJNZ R6,HERE1
          RET

        ;JMP LOOP_SERIAL1
         ; INC R1
         ; DJNZ R3,LOOP_SERIAL

     JMP START_PROGRAM
;(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
DEALAY:
     PUSH ACC
     MOV R6,#20
REPP2:  MOV A,#0A6H
MD_OLP:
          INC  A
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          JNZ MD_OLP
          NOP
            DJNZ R6,REPP2
           POP ACC
           RET
; ***********************************************************
 ; SUB SETS THE DS1307 OSCILLATOR
```

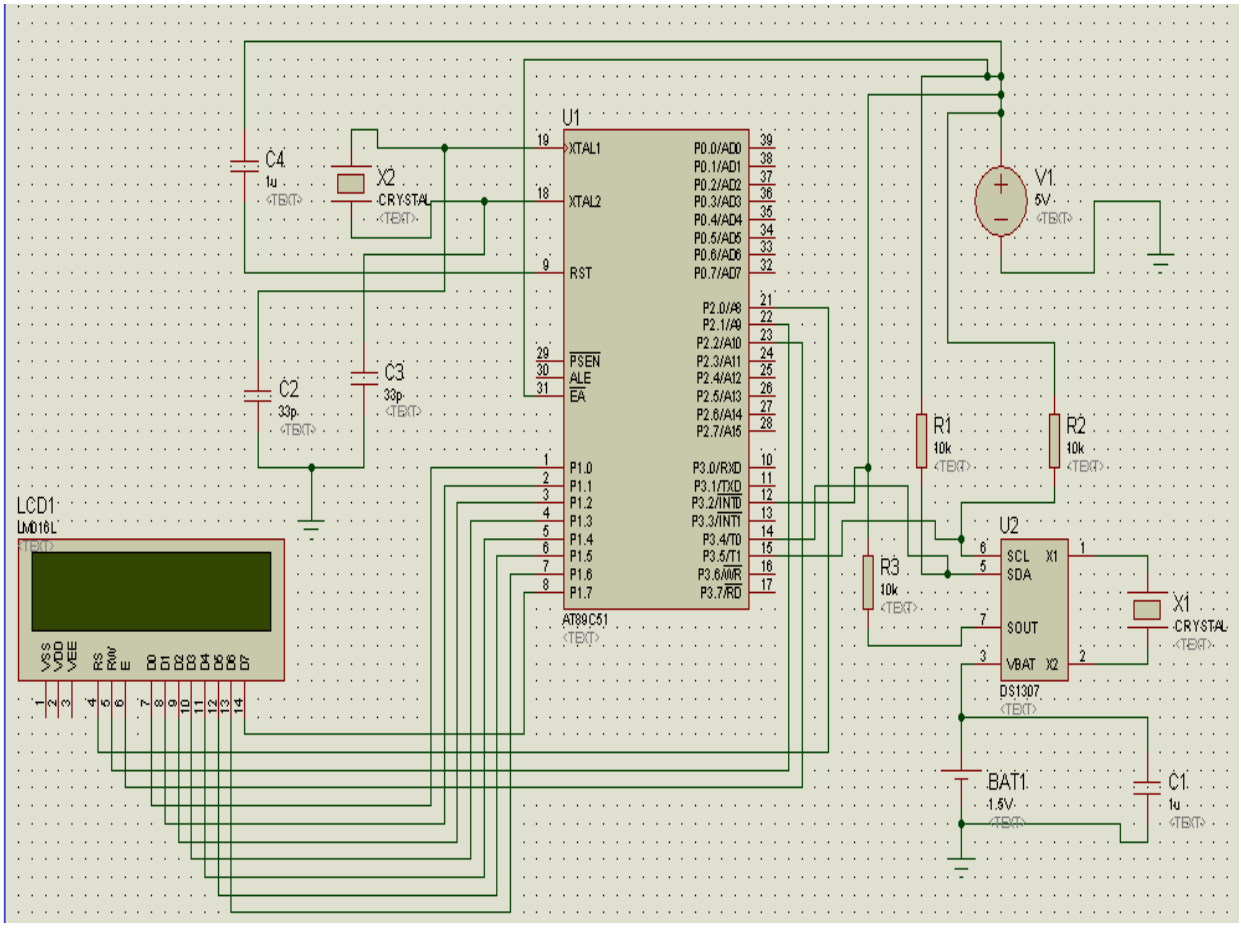
```
; *********************************************************

OSC_CONTROL:
         ACALL       SEND_START ; GENERATE START CONDITION

               MOV A,#RTC_W ; 1101 0000 ADDRESS + WRITE-BIT
         ACALL       SEND_BYTE  ; SEND BYTE TO 1307

               MOV A,#00H     ; ADDRESS BYTE TO REGISTER 00H
         ACALL       SEND_BYTE  ; SECONDS REGISTER, ALWAYS LEAVE
        SETB        LASTREAD   ; REG 00H-BIT #7 = 0 (LOW)

               ACALL          SEND_STOP   ; IF REG 00H-BIT #7 = 1
CLOCK

               ACALL       SEND_START ; OSCILLATOR IS OFF.

               MOV A,#RTC_R ; 1101 0001 ADDRESS + READ-BIT
                    ACALL       SEND_BYTE  ;

             ACALL       READ_BYTE  ; READ A BYTE FROM THE 1307
         CLR         ACC.7      ; CLEAR REG 00H-BIT #7 TO ENABLE
OSC_SET:                        ; OSCILLATOR.
         PUSH        ACC        ; SAVE ON STACK
         ACALL       SEND_STOP  ;

           ACALL       SEND_START ;

           MOV         A,#RTC_W ; SETUP TO WRITE
         ACALL       SEND_BYTE  ;

           MOV         A,#00H     ; REGISTER 00H ADDRESS
         ACALL       SEND_BYTE  ;

           POP         ACC        ; GET DATA TO START OSCILLATOR
         ACALL       SEND_BYTE  ; SEND IT

           ;MOV         A,#01H     ;
           ;ACALL       SEND_BYTE  ;

           ;MOV         A,#02H     ;
         ;ACALL       SEND_BYTE  ;


           ACALL       SEND_STOP
         RET
; *********************************************************
; THIS SUB CONTROLS THE SQW OUTPUT 1HZ
; *********************************************************
SQW_CONTROL_1HZ:
```

```
     LCALL SEND_START    ; SEND START CONDITION
     MOV A,#RTC_W        ; SET POINTER TO REG 07H ON
                    ; DS1307
     LCALL SEND_BYTE
     MOV A,#07H
     LCALL SEND_BYTE
     MOV A,#90H          ; SQW/OUT ON AT 1HZ
     JNB SQW,SQW_SET           ; JUMP IF SQW BIT IS ACTIVE
     MOV A,#80H          ; TURN SQW/OUT OFF – OFF HIGH
SQW_SET:
     LCALL SEND_BYTE
     LCALL SEND_STOP
     RET
; *********************************************************
; THIS SUB READS ONE BYTE OF DATA FROM THE DS1307
; *********************************************************

READ_BYTE:
         MOV         BITCNT,#08H; SET COUNTER FOR 8-BITS DATA
         MOV         A,#00H
         SETB        SDA         ; SET SDA HIGH TO ENSURE LINE
                                 ; FREE
READ_BITS:
         SCL_HIGH                ; TRANSITION SCL LOW-TO-HIGH
         MOV         C,SDA       ; MOVE DATA BIT INTO CARRY
         RLC         A           ; ROTATE CARRY-BIT INTO ACC.0
         CLR         SCL         ; TRANSITION SCL HIGH-TO-LOW
         DJNZ        BITCNT,READ_BITS
                                 ; LOOP FOR 8-BITS
         JB          LASTREAD,ACKN
                                 ; CHECK TO SEE IF THIS IS
                                 ; THE LAST READ
         CLR         SDA         ; IF NOT LAST READ SEND ACK-BIT

ACKN:
         SCL_HIGH                          ; PULSE SCL TO TRANSMIT
ACKNOWLEDGE
         CLR         SCL         ; OR NOT ACKNOWLEDGE BIT
         RET

; *********************************************************
; SUB SENDS START CONDITION
; *********************************************************

SEND_START:
         SETB        _2W_BUSY    ; INDICATE THAT 2-WIRE
         CLR         ACK         ; OPERATION IS IN PROGRESS
         CLR         BUS_FLT     ; CLEAR STATUS FLAGS
         JNB         SCL,FAULT
         JNB         SDA,FAULT
```

```
          SETB        SDA         ; BEGIN START CODITION
          SCL_HIGH
          CLR         SDA
          NOP
            NOP
            NOP
            NOP
            NOP
          CLR         SCL
          RET
FAULT:
          SETB        BUS_FLT
          RET

; *********************************************************
; SUB SENDS STOP CONDITION
; *********************************************************
SEND_STOP:
          CLR         SDA
          SCL_HIGH
          SETB        SDA
          CLR         _2W_BUSY
          RET

; THIS SUB SENDS 1 BYTE OF DATA TO THE DS1307
; CALL THIS FOR EACH REGISTER SECONDS TO YEAR
; ACC MUST CONTAIN DATA TO BE SENT TO CLOCK
; *********************************************************
SEND_BYTE:
          MOV         BITCNT,#08H; SET COUNTER FOR 8-BITS
SB_LOOP:
          JNB         ACC.7,NOTONE; CHECK TO SEE IF BIT-7 OF
          SETB        SDA         ; ACC IS A 1, AND SET SDA HIGH
          JMP         ONE
NOTONE:
          CLR         SDA         ; CLR SDA LOW
ONE:
          SCL_HIGH                ; TRANSITION SCL LOW-TO-HIGH
          RL          A           ; ROTATE ACC LEFT 1-BIT
          CLR         SCL         ; TRANSITION SCL LOW-TO-HIGH
          DJNZ        BITCNT,SB_LOOP; LOOP FOR 8-BITS
          SETB        SDA         ; SET SDA HIGH TO LOOK FOR
          SCL_HIGH                ; ACKNOWLEDGE PULSE
          CLR         ACK
          JNB         SDA,SB_EX   ; CHECK FOR ACK OR NOT ACK
          SETB        ACK         ; SET ACKNOWLEDGE FLAG FOR
                                  ; NOT ACK
SB_EX:
         NOP
           NOP
```

```
          NOP
          NOP
          NOP

          CLR         SCL         ; TRANSITION SCL HIGH-TO-LOW

          NOP
          NOP
          NOP
          NOP
          NOP

         RET
; ********************************************************
; SUB READS THE CLOCK AND WRITES IT TO THE SCRATCHPAD MEMORY
; ON RETURN FROM HERE DATE & TIME DATA WILL BE STORED IN THE
; DATE & TIME REGISTERS FROM 24H (SECS) TO 2AH (YEAR)
; ALARM SETTINGS IN REGISTERS 2CH(HRS) AND 2DH(MINUTES).
; ********************************************************
READ_CLOCK:
          MOV         R1,#24H     ; SECONDS STORAGE LOCATION
          MOV         BYTECNT,#00H
          CLR         LASTREAD
          ACALL       SEND_START
          MOV         A,#RTC_W
          ACALL       SEND_BYTE
          MOV         A,#00H
          ACALL       SEND_BYTE
          ACALL       SEND_STOP
          ACALL       SEND_START
          MOV         A,#RTC_R
          ACALL       SEND_BYTE

READ_LOOP:
          MOV         A,BYTECNT
          CJNE        A,#09H,NOT_LAST
          SETB        LASTREAD

NOT_LAST:
          ACALL       READ_BYTE
          MOV         @R1,A
          MOV         A,BYTECNT
          CJNE        A,#00H,NOT_FIRST
          MOV         A,@R1
          CLR         ACC.7       ; ENSURE OSC BIT=0 (ENABLED)
          MOV         @R1,A
NOT_FIRST:
          INC         R1
          INC         BYTECNT
          MOV         A,BYTECNT
```

```
        CJNE        A,#0AH,READ_LOOP
        ACALL       SEND_STOP
        RET


DELAY:
      MOV R1,#0CCH
REP2:MOV R7,#0FFH
REP1:NOP
      DJNZ R7,REP1
      DJNZ R1,REP2
      RET

 LDELAY   :
          LCALL DELAY
          LCALL DELAY
          LCALL DELAY
          LCALL DELAY
           RET
      END
```



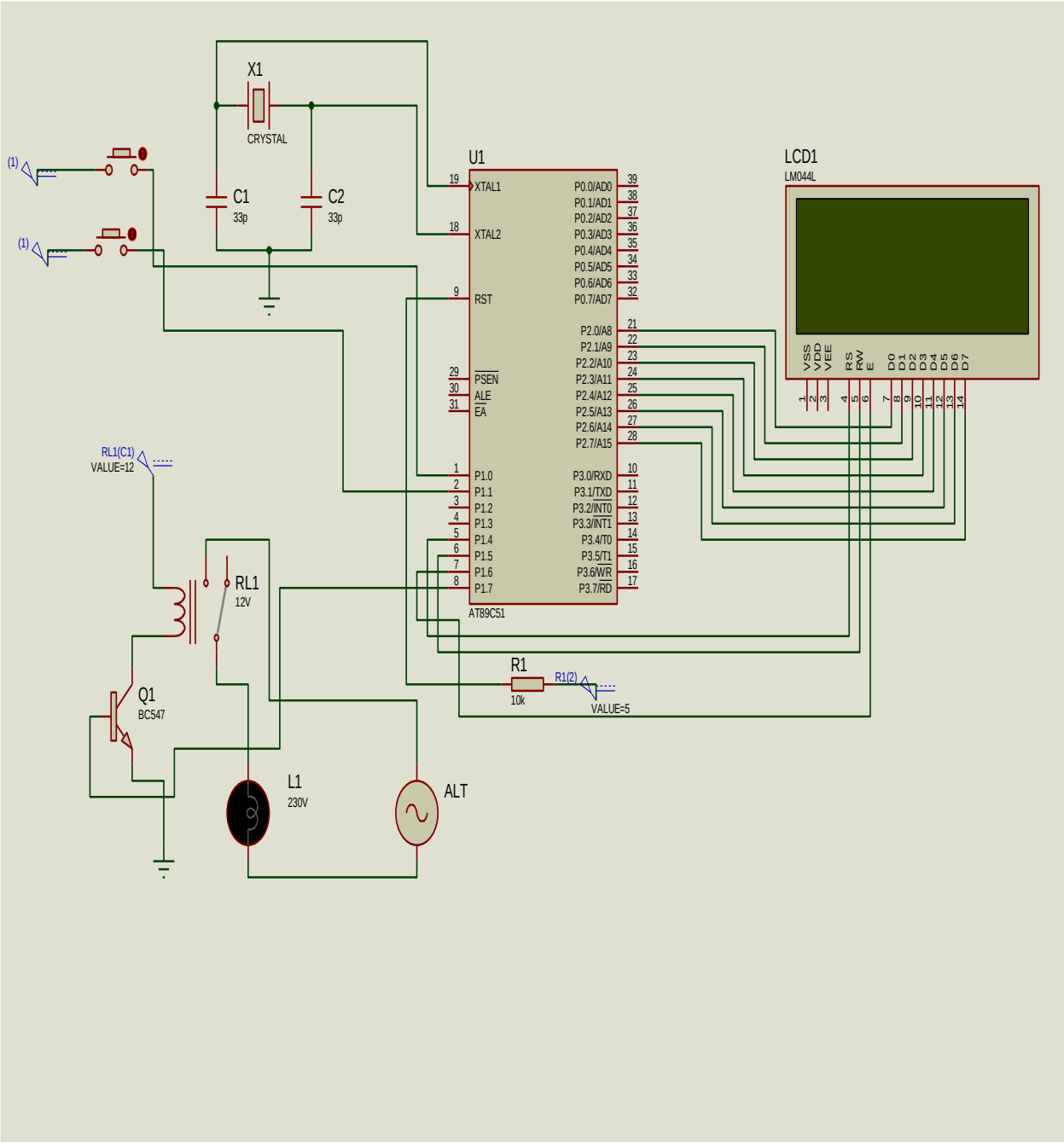
Chapter 16: Automatic room light control with visitor counter

```
#include <REGX51.H>
unsigned int i=0;
unsigned char temp,huns,tens,ones;
unsigned int count=0;
typedef unsigned char bit_8;
typedef unsigned int bit_16;

/**************************************
          LCD Port Configuration
**************************************/
sbit rs = P1^4;
sbit rw = P1^5;
sbit en = P1^6;

/*****************************************/
sbit ir1  = P1^0;
sbit ir2  = P1^1;
sbit light     = P1^7;
/*****************************************/

void lcd_clr(void);
void lcdDelay(int);
void lcd_init(void);          // Initialize LCD
void wrt_cmd(bit_8);          // Sending Command
void wrt_data(bit_8);         // Sending single character
void wrt_value(bit_16);       // Sending max 16-bit Decimal value
void wrt_string(bit_8*); // Sending String
void cursorxy(bit_8,bit_8);   // Bringing Cursor to (X,Y) location
X ->0,1 and Y -> 0-15
void delay(int);

void main()
{

     P1 = 0x00;
     P2 = 0x00;               // making P2 as output port
     P3 = 0x00;

     lcd_init();    // initilazing lcd
     lcd_clr();
     lcdDelay(500) ;
     cursorxy(0,0);
     wrt_string("Visitor Counter");
     cursorxy(1,0);
     wrt_string("Count:");

     while(1)
     {
```

```
            wrt_value(count);


            if(ir1==1)
            {

                  count++;
                  if(count>=999) count=0;
                  delay(25000);

            }
            if(count>=1)
            {
                  if(ir2==1)
                  {

                        count--;
                        if(count <=0) count=0;
                        delay(25000);

                  }

            }
            if(count==0)
            {
                  light=0;
            }
            if(count>=1)
            {
                  light=1;
            }


      }
}
void delay(int del)
{

      for(i=0;i<=del;i++);

}
void lcd_clr(void)
{

      wrt_cmd(0x01);
      lcdDelay(100);
}
```

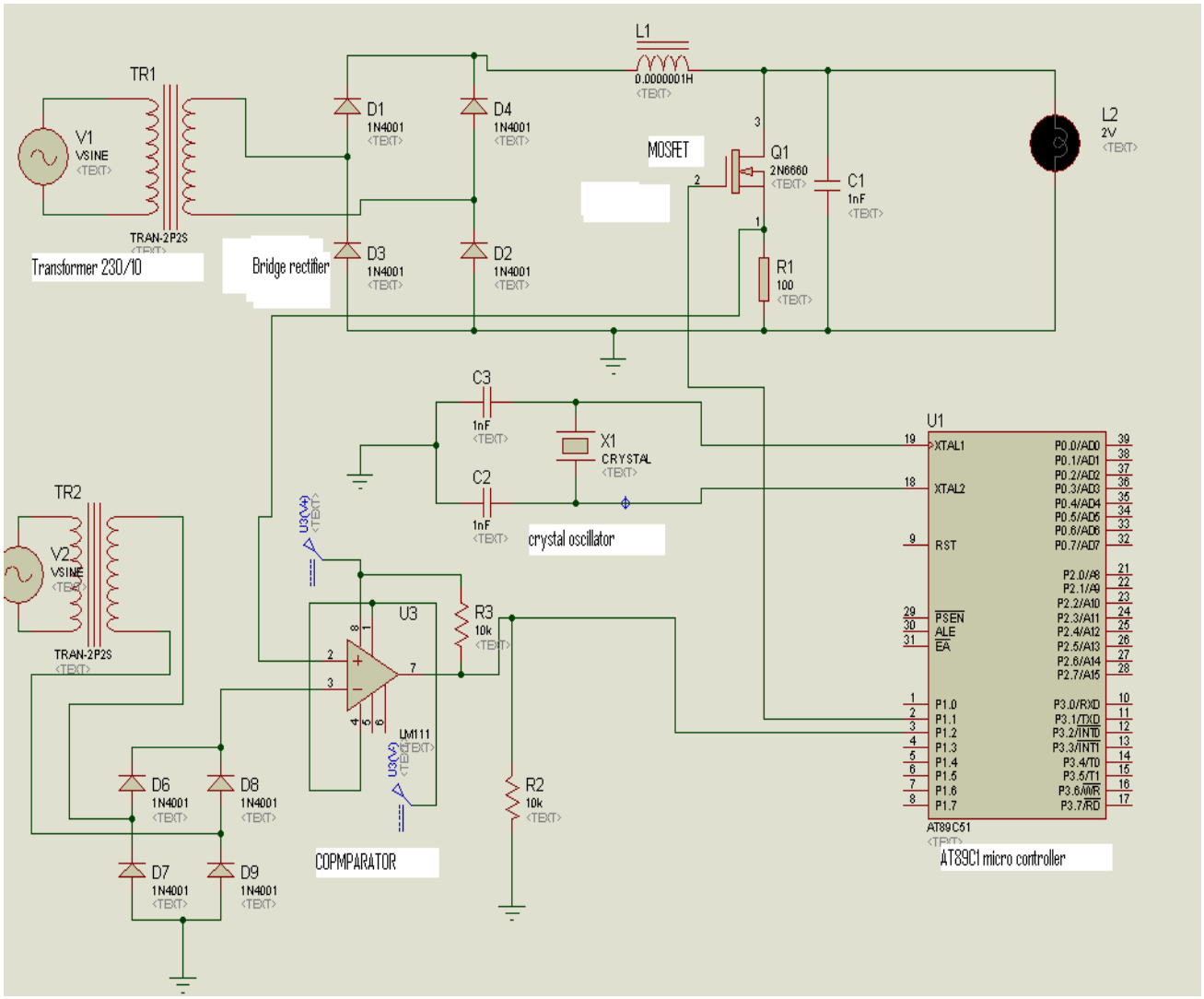
```
/*~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
     LCD Initialization
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~*/

void lcd_init(void)
{
     wrt_cmd(0x33); lcdDelay(100);
     wrt_cmd(0x32); lcdDelay(100);
     wrt_cmd(0x38); lcdDelay(100);
     wrt_cmd(0x0E); lcdDelay(100);
     wrt_cmd(0x01); lcdDelay(100);
     wrt_cmd(0x06); lcdDelay(100);

}


/*~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
     Writing command to LCD
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~*/

void wrt_cmd(bit_8 val)
{

     P2=val;
     rs=0;
     rw=0;
     en=1;
     en=0;
}

/*~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
     Writing string to LCD
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~*/

void wrt_string(bit_8 *string)
{

     while(*string)
          wrt_data(*string++);
}
/*~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
     Writing 8-bit Value to LCD
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~*/
void wrt_value(bit_16 x)
{
          temp=x;
          huns=temp/100;
          temp=temp%100;
          tens=temp/10;
          ones=temp%10;
```

```
          cursorxy(1,6);
          wrt_data(huns+0x30);
          wrt_data(tens+0x30);
          wrt_data(ones+0x30);
}


/*~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
     Writing data to LCD
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~*/

void wrt_data(bit_8 ch)
{

     P2 = ch;
     rs=1;
     rw=0;
     en=1;
     lcdDelay(500) ;
     en=0;
}

/*~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
     Bringing Cursor to (X,Y) location of LCD
          X -> 0,1
          Y -> 0,15
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~*/

void cursorxy(bit_8 x, bit_8 y)
{

     if (x == 0) wrt_cmd(0x80 + y);
     if (x == 1) wrt_cmd(0xC0 + y);

}

void lcdDelay(int ndel)
{
     int i1;

     for(i1=0 ; i1<ndel ; i1++);
}
```


X1
CRYSTAL
C1
33p
C2
33p
U1
XTAL1
XTAL2
RST
PSEN
ALE
EA
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
AT89C51
LCD1
LM044L
VSS
VDD
VEE
RS
RW
E
D0
D1
D2
D3
D4
D5
D6
D7
RL1(C1)
VALUE=12
RL1
12V
Q1
BC547
R1
10k
R1(2)
VALUE=5
L1
230V
ALT

```
//program for to generate pulses for mosfet and to control the
output current.

#include<reg51.h>
sbit a=P1^2;
sbit b=P1^3;
sbit WAVE=P1^1;

void timer0() interrupt 1
{
WAVE=~WAVE;
}
void serial0() interrupt 4
{
if(TI==1)
{
TI=0;
}

else
{
P0=SBUF;
RI=0;
}
}

void main()
{
a=1;
TMOD=0x22;
TH1=0xF6;
SCON=0x50;
TH0=0xA4;
IE=0x92;
TR1=1;
TR0=1;

while(1)
{
if(a==1)
{
b=0;
WAVE=a;
}
else
b=1;
```

```
}
}
//End of the program.
```

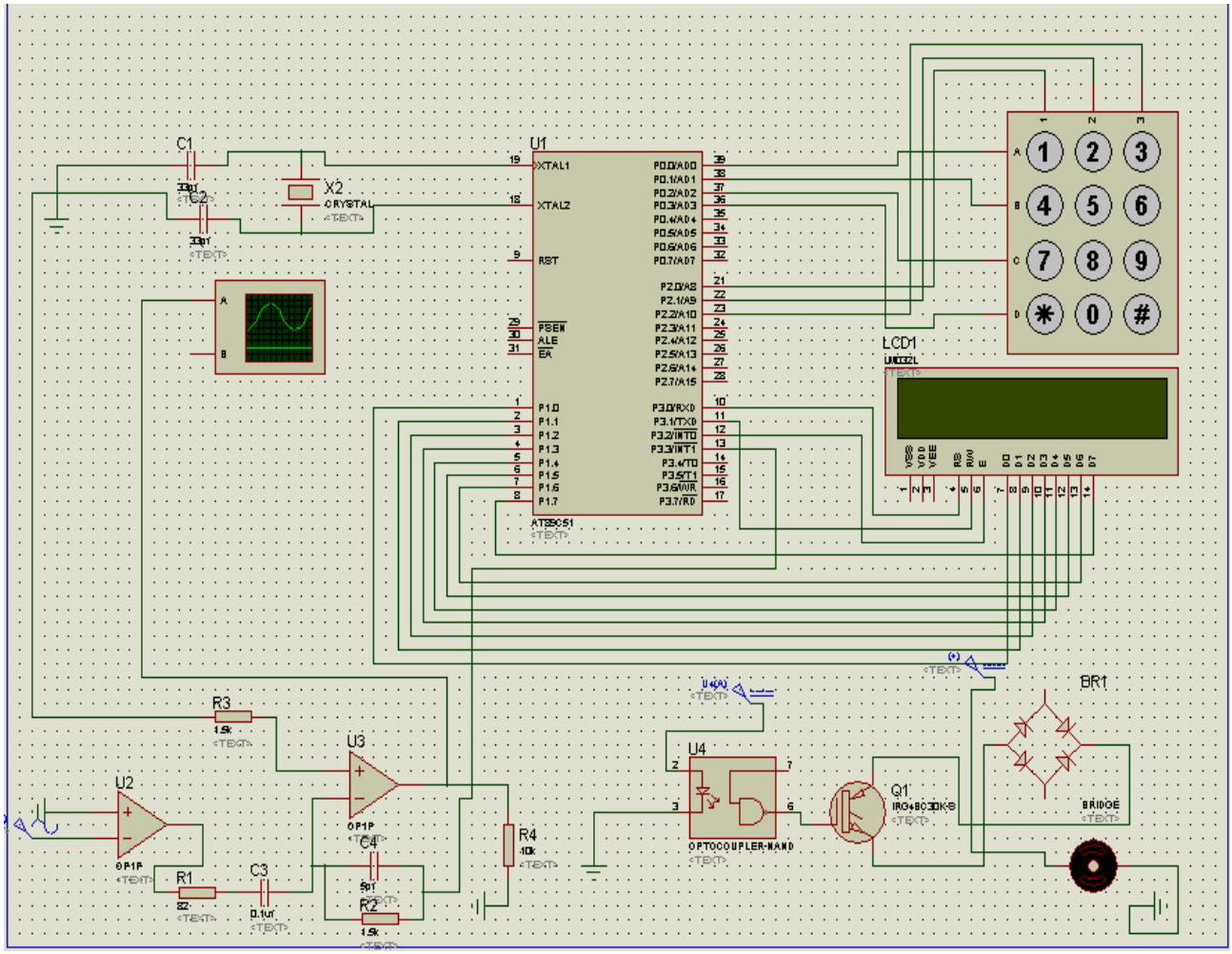


Chapter 18: Speed control of Induction motor using IGBT

```
#include<reg51.h>
#include<math.h>
#include<stdio.h>
sbit rs=P3^0;
sbit rw=P3^1;
sbit e=P3^2;
sbit in=P3^3;
sbit out=P3^4;

unsigned char keys [4][3]={'1','2','3',
'4','5','6',
'7','8','9',
'*','0','#',};
unsigned char a,b;
void lcdcmd(unsigned char);
void lcddata(unsigned char);
void delay(int);
void offdelay(toff);
void ondelay(ton);

void main()
{
float b,c,clr,a,x,y,t,n,v,d,ton,toff;
lcdcmd (0x38);
lcdcmd (0x01);
lcdcmd (0x0e);
lcdcmd (0x14);

in=1;
out=0;
P2=0x07;
b=0;
y:
P0=0x00;
delay(20);
if(P2=0x07)
goto y;
delay(20);
if(P2=0x07)
goto y;
a=P2;
P0=0x0e;
if(P2!=0x07)
goto row1;
P0=0x0d;

if(P2!=0x07)
goto row2;
P0=0x0b;
```

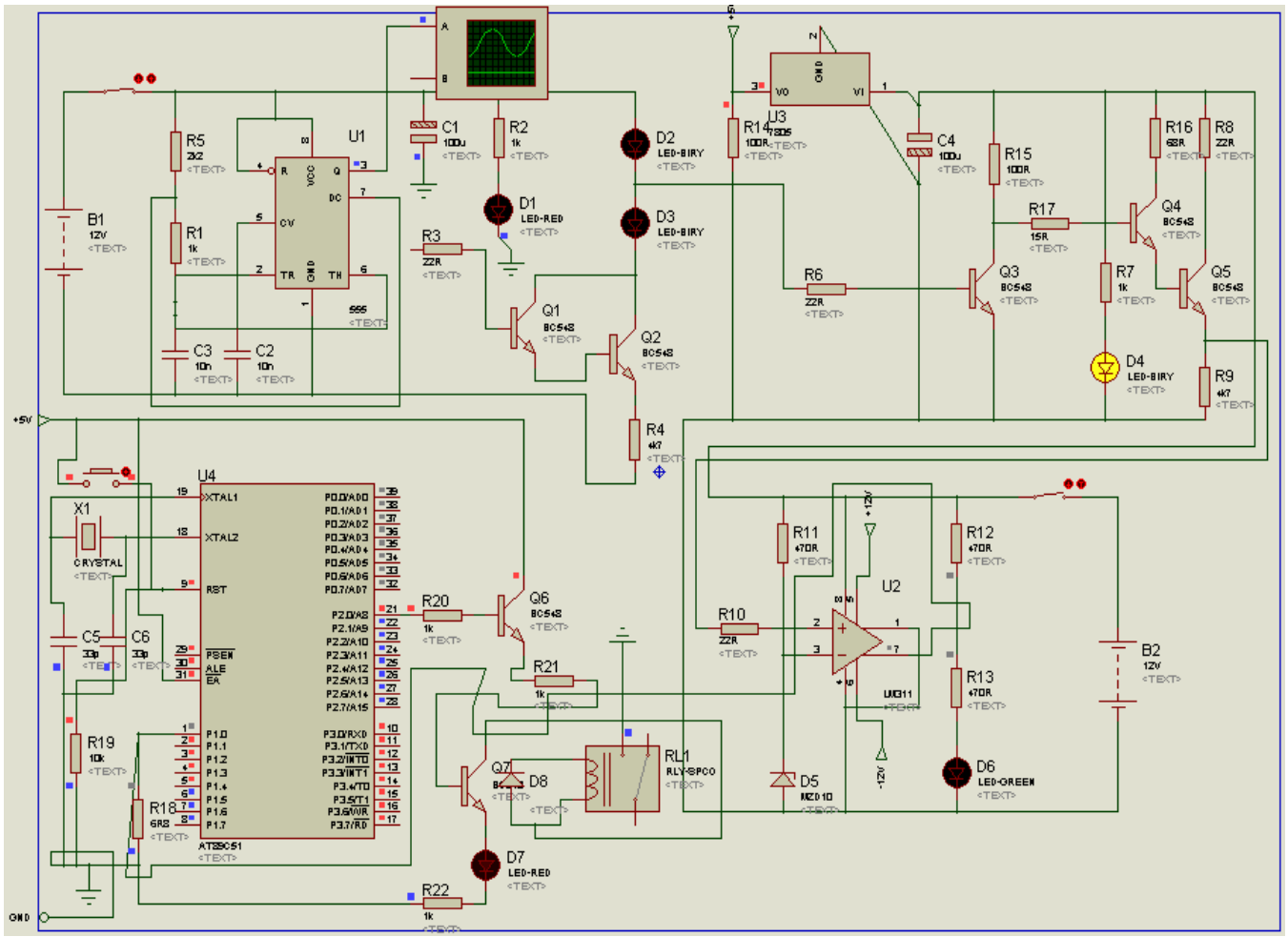
```
if(P2!=0x07)
goto row3;
P0=0x07;

if(P2!=0x07)
goto row4;
else goto y;

row1:
if(P2==0x06)
{b=1;lcddata(keys[0][0]);}
else if(P2==0x05)
{b=2;lcddata(keys[0][1]);}

else if(P2==0x03)
{b=3;lcddata(keys[0][2]);}
goto y;

row2:
if(P2==0x06)
{b=4;lcddata(keys[1][0]);}
else if(P2==0x05)
{b=5;lcddata(keys[1][1]);}

else if(P2==0x03)
{b=6;lcddata(keys[1][2]);}
goto y;

row3:
if(P2==0x06)
{b=7;lcddata(keys[2][0]);}
else if(P2==0x05)
{b=8;lcddata(keys[2][1]);}

else if(P2==0x03)
{b=9;lcddata(keys[2][2]);}
goto y;

row4:
if(P2==0x05)
{b=0;lcddata(keys[3][1]);goto y;}
 if(P2==0x06)
{c=1;lcddata(keys[3][0]);goto z;}

 if(P2==0x03)
{clr=1;lcddata(keys[3][2]);goto y;}

z:t=(20);
n=(b*175);
```

```
v=(n/(65/10));
d=v/230;
x=d*t;
y=(t-x);
toff=y;
ton=x;

while(1)
{
if (in==1)
{out=0;
offdelay(toff);
out=1;
ondelay(ton);
}
}
goto y;
}


void lcdcmd(unsigned char value)
{rs=0;rw=0;e=1;delay(1);P1=value;e=0;}

void lcddata(unsigned char value)
{rs=1;rw=0;e=1;delay(1);P1=value;e=0;}

void delay(int value)
{int i,j;
for (i=0;i<=value;i++)
for (j=0;j<=1275;j++)
}

void offdelay(toff)
{flot i;
for(i=0;i<=toff;i++);
}
void ondelay(ton)
{flot j;
for(j=0;j<=ton;j++);
}
```


U1
AT89C51
X2
CRYSTAL
C1
C2
LCD1
BR1
BRIDGE
U2
U3
U4
OPTOCOUPLER-NAND
Q1
R1
R2
R3
R4
C3
C4

```
#include<stdio.h>
#include<Regx51.h>
sbit t0=P1^0;
sbit t1=P1^1;
sbit t2=P1^2;
sbit t3=P1^3;
sbit t4=P2^0;
sbit t5=P2^1;
sbit t6=P2^2;
sbit t7=P2^3;
sbit t8=P2^4;
sbit t9=P1^5;
sbit t10=P1^6;
sbit t11=P1^7;
sbit t12=P2^5;
sbit t13=P2^6;
sbit t14=P2^7;

void main()
{
 t9=t10=t11=t12=t13=t14=0;
 t0=1;t1=1;t2=1;t3=1;//t3=t2=t1=t0=1;
 for(;;)
 {
  l1:if(t0==0)
  {
   t4=1;
   t5=t6=t7=t8=0;
   goto l1;
  }
  l2:if(t1==0)
  {
   t5=1;
   t4=t6=t7=t8=0;
   goto l2;
  }
  l3:if(t2==0)
  {
   t6=1;
   t4=t5=t7=t8=0;
   goto l3;
  }
  l4:if(t3==0)
  {
   t7=1;
   t4=t5=t6=t8=0;
   goto l4;
  }
```

```
    t8=1;
    t4=t5=t6=t7=0;
  }
}
```



For any queries kindly contact

Dr.M.Chakravarthy
Professor HOD-EEE
hodeee@staff.vce.ac.in
Cell No: 9849979136
OR
G. Sandhya Rani
Asst.Prof
g.sandhyarani@staff.vce.ac.in
9494771587