1. **LED INTERFACING**

**LED FLASHING**

#include<Reg51.h>

void delay (unsigned int time);

void main ()

{

P0=0X00;

while(1)

{

P0=0X01;

delay(25);

P0=0X00;

delay(25);

}

}

void delay(unsigned int time)

{

int i,j;

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

for(j=0;j<155;j++);

}

**LED UP COUNTER**

#include <reg51.h>

void delay(unsigned int time);

void main()

{

int count;

P0 = 0X00;

while(1)

{

for(count =0; count<=15; count++)

{

P0 = count;

delay(300);

}

}

void delay(unsigned int time)

{

int i,j;

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

for(j=0; j<=155; j++);

}

**LED DOWN COUNTER**

#include <reg51.h>

void delay(unsigned int time);

void main()

{

int count;

P0 = 0X00;

while(1)

{

for(count =15; count >=0; count--)

{

P0 = count;

delay(300);

}

}

void delay(unsigned int time)

{

int i,j;

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

for(j=0; j<=155; j++);

}

**BCD UP COUNTER**

#include <reg51.h>

void delay(unsigned int time);

void main()

{

int count;

P0 = 0X0B00000000;

while(1)

{

for(count =0; count<=9; count++)

{

P0 = count;

delay(300);

}

}

void delay(unsigned int time)

{

int i,j;

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

for(j=0; j<=155; j++);

}

**BCD DOWN COUNTER**

#include <reg51.h>

void delay(unsigned int time);

void main()

{

int count;

P0 = 0X0B00000000;

while(1)

{

for(count =9; count >=0; count--)

{

P0 = count;

delay(300);

}

}

void delay(unsigned int time)

{

int i,j;

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

for(j=0; j<=155; j++);

}

**2. Stepper motor Interfacing**

**1. stepper motor using timer full step in clockwise:**

#include < reg51 .h >

void delay(unsigned int time);

void main()

{

P0=0x00;

while(1)

{

P0=0x09;

delay();

P0=0X0A;

delay();

P0=0x06;

delay();

P0=0X05;

delay();

}

}

void delay()

{

TMOD=0x90;

TH1=0xFF;

TL1=0x49 ;

TR1=1;

while(!TF1) ;

TR1 = 0;

TF1 = 0;

}

**2.Full step in clockwise**

#include<Reg51.h>

void delay (unsigned int time);

void main ()

{

P0=0X00;

while(1)

{

P0=0X09;

delay(25);

P0=0X05;

delay(25);

P0=0X06;

delay(25);

P0=0X0A;

delay(25);

}

}

void delay(unsigned int time)

{

int i,j;

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

for(j=0; j<155; j++);

}

**3.Full step in anticlockwise**

#include<Reg51.h>

void delay (unsigned int time);

void main ()

{

P0=0X00;

while(1)

{

P0=0X09;

delay(25);

P0=0X0A;

delay(25);

P0=0X06;

delay(25);

P0=0X05;

delay(25);

}

}

void delay(unsigned int time)

{

int i,j;

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

for(j=0; j<155; j++);

}

**4.Half step in clockwise**

#include<Reg51.h>

void delay (unsigned int time);

void main ()

{

P0=0X00;

while(1)

{

P0=0X09;

delay(25);

P0=0X01;

delay(25);

P0=0X05;

delay(25);

P0=0X04;

delay(25);

P0=0X06;

delay(25);

P0=0X02;

delay(25);

P0=0X0A;

delay(25);

P0=0X08;

delay(25);

}

}

void delay(unsigned int time)

{

int i,j;

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

for(j=0; j<155; j++);

}

**5.Half step in anticlockwise**

#include<Reg51.h>

void delay (unsigned int time);

void main ()

{

P0=0X00;

while(1)

{

P0=0X09;

delay(25);

P0=0X08;

delay(25);

P0=0X0A;

delay(25);

P0=0X02;

delay(25);

P0=0X06;

delay(25);

P0=0X04;

delay(25);

P0=0X05;

delay(25);

P0=0X01;

delay(25);

}

}

void delay(unsigned int time)

{

int i,j;

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

for(j=0; j<155; j++);

}

**3. DAC INTERFACING**

**Triangular Wave**

#include<reg51.h>

void main()

{

int count;

P0=0X00;

While(1)

{

P0=0X00;

for(count=0X00; count<0Xff; count++)

{

P0=count;

}

P0=0Xff;

for(count=0Xff; count>0X00; count--)

{

P0=count;

}

}

}

**Forward Ramp**

#include<Reg51.h>

void main()

{

int count =0XFF;

int i=0;

P0=0X00;

while(1)

{

P0=0X00;

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

{

P0=i;

}

P0=0X00;

}

}

**Reverse Ramp**

#include<Reg51.h>

void main()

{

int count =00;

int i;

P0=0XFF;

while(1)

{

P0=0XFF;

for (i=0XFF; i>=count; i--)

{

P0=i;

}

P0=0XFF;

}

}

**Square Wave**

#include<Reg51.h>

void delay (unsigned int time);

void main ()

{

P0=0X00;

while(1)

{

P0=0Xff;

delay(25);

P0=0X00;

delay(25);

}

}

void delay(unsigned int time)

{

int i,j;

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

for(j=0; j<155; j++);

}

**square wave generation using timer :**

#include < reg51 .h >

sbit square = P2^0 ;

void timer1\_ISR (void) interrupt 1

{

square = ~ square;

TH1=0xFF;

TL1=0x49;

}

void main()

{

TMOD=0x90;

TH1 = 0xFF;

TL1 = 0x49;

ET1 =1;

TR1 = 1;

while (1);

 }

**Staircase Wave**

#include <reg51.h>

void delay(unsigned int time);

void main()

{

int values [] = {0, 50, 100, 180, 156, 180, 156, 100, 50};

P0 = 0x00;

while(1)

{

int i;

for (i =0; i<9; i++)

{

P0 = values[i];

delay(155);

}

}

}

void delay (unsigned int time)

{

int i,j;

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

for (j = 0; j<155; j++);

}

**4. LED,Buzzer Relay Interfacing With PIC18**

#include<p18f4550.h>

#pragma config FOSC = HS

#pragma config WDT = OFF

#pragma config LVP = OFF

#pragma config PBADEN = OFF

#define sw1 PORTBbits.RB5

#define sw2 PORTBbits.RB5

#define buzzer PORTCbits.RC2

#define relay PORTCbits.RC1

void msdelay(unsigned int time);

void main()

{

unsigned char val = 0;

INTCON2bits.RBPU=0;

ADCON1 = 0x0F;

TRISBbits.TRISB4=1;

TRISBbits.TRISB5=1;

TRISCbits.TRISC1=0;

TRISCbits.TRISC2=0;

TRISD = 0x00;

PORTD = 0x00;

buzzer=0;

relay =0;

while(1)

{

if(!(sw1))

val=1;

if(!(sw2))

val=2;

if(val==1)

{

buzzer=1;

relay=1;

PORTD=PORTD>>1;

if(PORTD==0x00)

PORTD=0x00;

msdelay(250);

}

if(val==2)

{

buzzer=0;

relay=0;

PORTD=PORTD<<1;

if(PORTD==0x00)

PORTD=0x01;

msdelay(250);

}

}

}

void msdelay(unsigned int time)

{

unsigned int i,j;

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

for(j=0;j<275;j++);

}

**5.LCD interfacing**

#include <p18f4550.h>

#pragma config FOSC = HS

#pragma config WDT = OFF

#pragma config PBADEN = OFF

#pragma config LVP = OFF

#define en PORTEbits.RE2

#define rw PORTEbits.RE1

#define rs PORTEbits.RE0

void LCD\_cmd (unsigned char cmd);

void init\_LCD (void);

void LCD\_write (unsigned char data);

void LCD\_write\_string (static char \*str);

void myMsDelay (unsigned int time)

{

unsigned int i, j;

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

for (j = 0; j < 710; j++);

}

void display\_string\_LCD(static char \*pstring1, static char \*pstring2)

{

ADCON1 = 0x0F;

TRISD = 0x00;

TRISE = 0x00;

init\_LCD();

myMsDelay(50);

LCD\_write\_string(pstring1); myMsDelay(15);

}

void init\_LCD(void)

{

LCD\_cmd(0x38); myMsDelay(15);

LCD\_cmd(0x01); myMsDelay(15);

LCD\_cmd(0x0C); myMsDelay(15);

LCD\_cmd(0x80); myMsDelay(15);

return;

}

void LCD\_cmd(unsigned char cmd)

{

PORTD = cmd;

rs = 0;

rw = 0;

en = 1;

myMsDelay(15);

en = 0;

myMsDelay(15);

return;

}

void LCD\_write(unsigned char data) {

PORTD = data;

rs = 1;

rw = 0;

en = 1;

myMsDelay(15);

en = 0;

myMsDelay(15);

return ;

}

void LCD\_write\_string(static char \*str)

{

int i = 0;

while (str[i] != 0)

{ LCD\_write(str[i]);

myMsDelay(15);

i++;

}

return;

}

void main(void)

{

char var1[] = "welcome";

char var2[] = "T";

init\_LCD();

while (1)

{

display\_string\_LCD(var1,var2);

}

}

**6. Keypad Interfacing**

#include <p18f4550.h>

#include <stdio.h>

#pragma config PLLDIV = 5

#pragma config CPUDIV = OSC1\_PLL2

#pragma config USBDIV = 2

#pragma config FOSC = HSPLL\_HS

#pragma config VREGEN = ON

#pragma config WDT = OFF

#pragma config PBADEN = OFF

#pragma config LVP = OFF

#define LCD\_DATA PORTD

#define ctrl PORTE

#define en PORTEbits.RE2

#define rw PORTEbits.RE1

#define rs PORTEbits.RE0

#define BUSY PORTDbits.RD7

void LCD\_Busy(void);

void LCD\_cmd(unsigned char cmd);

void init\_LCD(void);

void LCD\_write(unsigned char data);

void LCD\_write\_string(static char \*str);

void display\_string\_LCD(static char \*pstring1);

void myMsDelay (unsigned int time)

{

unsigned int i, j;

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

for (j = 0; j < 710; j++);

}

void PORTs\_init(void)

{

INTCON2bits.RBPU=0;

TRISB = 0x0F;

TRISD = 0x00;

TRISE =0x00;

ADCON1 = 0x0E;

}

void main()

{

char msg1 [] = "i2it";

char msg2 [] = "KEYPAD TEST";

char msg3 [] = "Press Key";

char msg4 [] = "Pressed Key:";

int key;

PORTs\_init();

init\_LCD();

LCD\_write\_string (msg1);

LCD\_cmd (0xC0);

LCD\_write\_string (msg2);

myMsDelay(350);

LCD\_cmd (0x01);

LCD\_write\_string (msg3);

LCD\_cmd (0xC0);

LCD\_write\_string (msg4);

myMsDelay(350);

while(1)

{

PORTB = 0xE0;

myMsDelay(10);

switch(PORTB & 0x0F)

{

case 0x0E:

myMsDelay (100);

if((PORTB & 0x0F) == 0x0E)

{

key='0';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

case 0x0D:

myMsDelay (100);

if((PORTB & 0x0F) == 0x0D)

{

key='4';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

case 0x0B:

myMsDelay (100);

if((PORTB & 0x0F) == 0x0B)

{

key='8';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

case 0x07:

myMsDelay (100);

if((PORTB & 0x0F) == 0x07)

{

key='C';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

}

PORTB = 0xD0;

myMsDelay(10);

switch(PORTB & 0x0F)

{

case 0x0E:

myMsDelay (100);

if((PORTB & 0x0F) == 0x0E)

{

key='1';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

case 0x0D:

myMsDelay (100);

if((PORTB & 0x0F) == 0x0D)

{

key='5';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

case 0x0B:

myMsDelay (100);

if((PORTB & 0x0F) == 0x0B)

{

key='9';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

case 0x07:

myMsDelay (100);

if((PORTB & 0x0F) == 0x07)

{

key='D';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

}

PORTB = 0xB0;

myMsDelay(10);

switch(PORTB & 0x0F)

{

case 0x0E:

myMsDelay (100);

if((PORTB & 0x0F) == 0x0E)

{

key='2';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

case 0x0D:

myMsDelay (100);

if((PORTB & 0x0F) == 0x0D)

{

key='6';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

case 0x0B:

myMsDelay (100);

if((PORTB & 0x0F) == 0x0B)

{

key='A';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

case 0x07:

myMsDelay (100);

if((PORTB & 0x0F) == 0x07)

{

key='E';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

}

PORTB = 0x70;

myMsDelay(10);

switch(PORTB & 0x0F)

{

case 0x0E:

myMsDelay (100);

if((PORTB & 0x0F) == 0x0E)

{

key='3';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

case 0x0D:

myMsDelay (100);

if((PORTB & 0x0F) == 0x0D)

{

key='7';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

case 0x0B:

myMsDelay (100);

if((PORTB & 0x0F) == 0x0B)

{

key='B';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

case 0x07:

myMsDelay (100);

if((PORTB & 0x0F) == 0x07)

{

key='F';

LCD\_cmd(0xCC);

LCD\_write (key);

}

break;

}

}

}

void display\_string\_LCD(static char \*pstring1)

{

int count=0,len;

TRISE = 0x00;

TRISD = 0x00;

LCD\_write\_string(pstring1);

myMsDelay(15);

return;

}

void init\_LCD(void)

{

LCD\_cmd(0x38);

myMsDelay(15);

LCD\_cmd(0x01);

myMsDelay(15);

LCD\_cmd(0x0C);

myMsDelay(15);

LCD\_cmd(0x80);

myMsDelay(15);

return;

}

void LCD\_cmd(unsigned char cmd)

{

LCD\_DATA = cmd;

rs = 0;

rw = 0;

en = 1;

myMsDelay(15);

en = 0;

myMsDelay(15);

return;

}

void LCD\_write(unsigned char data)

{

LCD\_DATA = data;

rs = 1;

rw = 0;

en = 1;

myMsDelay(15);

en = 0;

myMsDelay(15);

return ;

}

void LCD\_write\_string(static char \*str)

{

int i = 0;

while (str[i] != 0)

{

LCD\_write(str[i]);

myMsDelay(15);

i++;

}

return;

}

**8. DC MOTOR CONTROL**

#include<p18f4550.h>

#pragmaconfig FOSC = HS

#pragmaconfig PWRT = OFF

//#pragma config BOREN = OFF

#pragmaconfig WDT = OFF

#pragmaconfig LVP = OFF

#pragmaconfig PBADEN = OFF

voidmyMsDelay (unsignedint time);

void main()

{

TRISCbits.TRISC2 = 0 ;// Set PORTC, 2 as output

TRISCbits.TRISC6 = 0 ;

TRISCbits.TRISC7 = 0 ;

PR2 = 0XBA; // set PWM period to Maximum value

CCPR1L = 0x4B; // Initalise PWM duty cycle to 00

CCP1CON = 0x0C; // Configure CCP1CON

T2CON = 0x07;

PORTCbits.RC6 = 1 ;

PORTCbits.RC7 = 0 ;

while(1)

{

CCPR1L = 0x4B; //40%

myMsDelay(1500);

CCPR1L = 0x4B; //80%

myMsDelay(1500);

}

}

voidmyMsDelay (unsignedint time)

{

unsignedint i, j;

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

for (j = 0; j < 710; j++);/\*Calibrated for a 1

}

**7.ADC PIC 18**

#include <p18f4550.h> //Include controller specific .h file

#pragma config FOSC = HS //Oscillator Selection

#pragma config WDT = OFF //Disable Watchdog timer

#pragma config LVP = OFF //Disable Low Voltage Programming

#pragma config PBADEN = OFF //Disable PORTB Analog inputs

//Declarations for LCD Connection

#define LCD\_DATA PORTD //LCD data port

#define en PORTEbits.RE2 // enable signal

#define rw PORTEbits.RE1 // read/write signal

#define rs PORTEbits.RE0 // register select signal

//Function Prototypes

void ADC\_Init(void); //Function to initialize the ADC

unsigned int Get\_ADC\_Result(void); //Function to Get ADC result

void Start\_Conversion(void); //Function to Start of Conversion

void msdelay (unsigned int time); //Function to generate delay

void init\_LCD(void); //Function to initialise the LCD

void LCD\_command(unsigned char cmd);

void LCD\_data(unsigned char data);

void LCD\_write\_string(static char \*str);

//Start of main program

void main()

{

char msg1[] = "welcome E&TC";

char msg2[] = "ADC O/P:";

unsigned char i, Thousands,Hundreds,Tens,Ones;

unsigned int adc\_val;

ADCON1 = 0x0F; //Configuring the PORT pins as digital I/O

TRISD = 0x00; //Configuring PORTD as output

TRISE = 0x00; //Configuring PORTE as output

ADC\_Init(); // Init ADC peripheral

init\_LCD(); // Init LCD Module

LCD\_write\_string(msg1); // Display Welcome Message

LCD\_command(0xC0); // Goto second line, 0th place of LCD

LCD\_write\_string(msg2); // Display Message "ADC O/P"

while(1)

{

Start\_Conversion(); //Trigger conversion

adc\_val= Get\_ADC\_Result();//Get the ADC output by polling GO bit

LCD\_command (0xC8); //Goto 9th place on second line of LCD

i = adc\_val/1000 ; //Get the thousands place

Thousands = i + 0x30; // Convert it to ASCII

LCD\_data (Thousands); // Display thousands place

i = (adc\_val%1000)/100; //Get the Hundreds place

Hundreds = i + 0x30; // Convert it to ASCII

LCD\_data (Hundreds); //Display Hundreds place

i = ((adc\_val%1000)%100)/10; //Get the Tens place

Tens = i + 0x30; // Convert it to ASCII

LCD\_data (Tens); //Display Tens place

i = adc\_val%10 ; //Get the Ones place

Ones = i + 30; // Convert it to ASCII

LCD\_data (i + 0x30); //Display Ones place

msdelay(300); }

}

//Function Definitions

void ADC\_Init()

{

ADCON0=0b00000100; //A/D Module is OFF and Channel 1 is selected

ADCON1=0b00001110; // Reference as VDD & VSS, AN0 set as analog pins

ADCON2=0b10001110; // Result is right Justified

//Acquisition Time 2TAD

//ADC Clk FOSC/64

ADCON0bits.ADON=1; //Turn ON ADC module

}

void Start\_Conversion()

{

ADCON0bits.GO=1;

}

//If you do not wish to use adc conversion interrupt you can use this

//to do conversion manually. It assumes conversion format is right adjusted

unsigned int Get\_ADC\_Result()

{

unsigned int ADC\_Result=0;

while(ADCON0bits.GO);

ADC\_Result=ADRESL;

ADC\_Result|=((unsigned int)ADRESH) << 8;

return ADC\_Result;

}

void msdelay (unsigned int time) //Function to generate delay

{

unsigned int i, j;

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

for (j = 0; j < 275; j++);//Calibrated for a 1 ms delay in MPLAB

}

void init\_LCD(void) // Function to initialise the LCD

{

LCD\_command(0x38); // initialization of 16X2 LCD in 8bit mode

msdelay(15);

LCD\_command(0x01); // clear LCD

msdelay(15);

LCD\_command(0x0C); // cursor off

msdelay(15);

LCD\_command(0x80); // go to first line and 0th position

msdelay(15);

}

void LCD\_command(unsigned char cmd) //Function to pass command to the LCD

{

LCD\_DATA = cmd; //Send data on LCD data bus

rs = 0; //RS = 0 since command to LCD

rw = 0; //RW = 0 since writing to LCD

en = 1; //Generate High to low pulse on EN

msdelay(15);

en = 0;

}

void LCD\_data(unsigned char data)//Function to write data to the LCD

{

LCD\_DATA = data; //Send data on LCD data bus

rs = 1; //RS = 1 since data to LCD

rw = 0; //RW = 0 since writing to LCD

en = 1; //Generate High to low pulse on EN

msdelay(15);

en = 0;

}

//Function to write string to LCD

void LCD\_write\_string(static char \*str)

{

int i = 0;

while (str[i] != 0)

{

LCD\_data(str[i]); // sending data on LCD byte by byte

msdelay(15);

i++;

}

}