

**EXP –**

**01 : MEMORY TRANSFER**

- **INTERNAL TO INTERNAL :-**

```
ORG 0000H
MOV R0, #40H
MOV R1, #60H
MOV R2, #10
BASIC:
MOV A,@R0
MOV@R1, A
INC R0
INC R1
DJNZ R2,BASIC END
```

- **INTERNAL TO EXTERNAL :-**

```
ORG 0000H
MOV R1, #05H
MOV R0, #30H
MOV DPTR, #2000H
BACK :MOV A, @R0
MOVX @DPTR,A
INC R0
INC DPTR
DJNZ R1,BACK
END
```

- **EXTERNAL TO INTERNAL :-**

```
ORG 0000H
MOV R1, #05H
MOV R0, #40H
MOV DPTR, #2000H
BACK :MOVC A, @A+DPTR
MOVX @R0,A
INC R0
INC DPTR
CLR A
```

**EXP - :**

```
DJNZ R1,BACK  
END
```

**02 P1 PORT LED FLASHING :-**

```
#include <reg51.h>

//delay function declaration void
delay(void);

void main(void)
{
// an infinite loop while(1)
{
// Turn ON all LED's connected to Port1
P1 = 0xFF; delay();

// Turn OFF all LED's connected to Port1
P1 = 0x00; delay();
}

//delay function definition void
delay(void)
{ int i,
j;

for(i=0; i<0xFF; i++) for(j=0;
j<0xFF; j++);
}
```

**EXP – :**

**02 TOGGLE P2 CONTINUOUSLY USING TIMER 1 MODE 1 DELAY :-**

```
#include <reg51.h>

//delay function declaration void
T1M1Delay(void);

void main(void)
{
    unsigned char x;

    P2 = 0x55;

    while(1)
    {
        // Toggle Port2
        P2 = ~P2;

        // Repeat delay 20 times for(x=0;
        x<20; x++)
        T1M1Delay();
    }
}

//delay function definition void
T1M1Delay(void)
{
    TMOD = 0x10;
    TL1 = 0xFE;
    TH1 = 0xA5;
    TR1 = 1;

    while(TF1 == 0);

    TR1 = 0;
    TF1 = 0;
}
```

**EXP – :**

**03 DAC :-**

• **SQUARE WAVE :-**

```
#include <reg51.h> void  
MSDelay(unsigned int); void  
main(void)  
{  
    while (1) //repeat forever  
{  
        P1=0x55;  
        MSDelay(250);  
        P1=0xAA;  
        MSDelay(250);  
    }  
}  
  
void MSDelay(unsigned int itime)  
{  
    unsigned int i,j; for  
(i=0;i<itime;i++) for  
(j=0;j<1275;j++);  
}
```

### **TRIANGULAR WAVE :-**

```
#include<reg51.h>

void delay(unsigned int);

void main(void)
{
    while(1)          // infinite loop
    {
        unsigned int x;
        for(;;)           //repeat forever
        {

            for(x=0;x<250;x++)
            {
                P1=x;
                delay(100);
            }
            for(x=250;x>0;x--)
            {
                P1=x;
                delay(100);
            }
        }
    }
}

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

- RAMP WAVE :-

```
#include <reg51.h>
void main() {unsigned
int i; while(1)
{for(i=0;i<256;i++)
{
P2=i;
}
}
}
```

#### EXP – 04 : STEPPER MOTOR :-

- CLOCKWISE DIRECTION :-

```
#include <reg51.h>
void    delay(void);
void main(void)
```

```

{
while(1)
{
P1=0xCC; delay();
P1=0x66; delay();
P1=0x33; delay();
P1=0x99; delay();
}
}

void delay()
{
TMOD=0x01; // timer 0 ,mode 1(16 bit)
TL0=0xFD;// load TL0
TH0=0x4B; TR0=1;
while(TF0==0);// wait for TF0 to roll over TR0=0;
TF0=0;
}

```

- **ANTICLOCKWISE DIRECTION :-**

```

#include<reg51.h>
void delay(void); void
main(void)
{
while(1)
{
P1=0x33;
delay();
P1=0x66;
delay();
P1=0x99;
delay();
}

```

```

P1=0xCC; delay()
);
}
}

void delay()
{
    TMOD=0x01; // timer 0 ,mode 1(16 bit)
    TL0=0xFD;// load TL0
    TH0=0x4B; TR0=1;
    while(TF0==0);// wait for TF0 to roll over TR0=0;
    TF0=0;
}

```

#### **EXP – 05 : LED WITH PIC18F4550 :-**

```

#include<P18F4550.h> void
delay()
{
    unsigned int i; for(i=0;i<30000;i++);
}

void main()
{
    unsigned char i, key = 0;

    TRISB = 0x00; // LED pins as output
    PORTB = 0x00;

    TRISDbits.TRISD0 = 1; // Set RD0 as input
    TRISDbits.TRISD1 = 1; // Set RD1 as input
}

```

```

TRISDbits.TRISD2 = 0; // Set Buzzer pin RD2 as output

TRISAbits.TRISA4 = 0; // Set Relay pin RA4 as output while(1)

{

if(PORTDbits.RD0 == 0) key =0; // If Button1 pressed

if(PORTDbits.RD1 == 0) key =1; // If Button2 pressed if(key

== 0)

{

PORTAbits.RA4 = 1; // Relay OFF

PORTDbits.RD2 = 0; // Buzzer OFF for(i=0;i<8;i++)

// Chase LED Right to Left

{

PORTB = 1<<i; delay();

PORTB = 0x00;

delay();

}

}

if(key == 1)

{

PORTAbits.RA4 = 0; // Relay ON

PORTDbits.RD2 = 1; // Buzzer ON for(i=7;i>

0;i--) // Chase LED Left to Right

{

PORTB = 1<<i; delay();

PORTB = 0x00;

```

```
delay();  
}  
}  
}  
}
```

### **EXP – 06 : LCD WITH PIC18F4550 :-**

```
#include <p18f4550.h>  
  
#define LCD_EN LATCbits.LC1  
  
#define LCD_RS LATCbits.LC0  
  
#define LCDPORT LATB void  
  
lcd_delay(unsigned int time)  
  
{  
  
    unsigned int i , j ; for(i  
  
= 0; i < time; i++)  
  
{  
  
    for(j=0;j<100;j++);  
  
}  
  
}  
  
void SendInstruction(unsigned char command)  
  
{  
  
    LCD_RS = 0; // RS Low : Instruction  
  
    LCDPORT = command; LCD_EN  
  
= 1; // EN High lcd_delay(10);
```

```

LCD_EN = 0; // EN Low; command sampled at EN falling edge lcd_delay(10);

}

void SendData(unsigned char lcddata)
{

LCD_RS = 1; // RS HIGH : DATA

LCDPORT = lcddata; LCD_EN

= 1; // EN High

lcd_delay(10);

LCD_EN = 0; // EN Low; data sampled at EN falling edge lcd_delay(10);

}

void InitLCD(void)

{

TRISB = 0x00; // Set data port as output

TRISCbits.RC0 = 0; // EN pin

TRISCbits.RC1 = 0; // RS pin

SendInstruction(0x38); // 8 bit mode, 2 line,5x7 dots

SendInstruction(0x06); // Entry mode

SendInstruction(0x0C); // Display ON cursor OFF

SendInstruction(0x01); // Clear display

SendInstruction(0x80); // Set address to 1st line

}

*****

```

```
******/  
  
unsigned char *String1 = "SAE Kondhwa"; unsigned  
char *String2 = "TE E&TC"; void main(void)  
  
{  
  
    TRISB = 0x00; // Set data port as output  
  
    TRISCbits.RC0 = 0; // EN pin  
  
    TRISCbits.RC1 = 0; // RS pin  
  
    SendInstruction(0x38); // 8 bit mode, 2 line,5x7 dots  
  
    SendInstruction(0x06); // Entry mode  
  
    SendInstruction(0x0C); // Display ON cursor OFF  
  
    SendInstruction(0x01); // Clear display SendInstruction(0x80);  
  
    // Set address to 1st line while(*String1)  
  
    {  
  
        SendData(*String1);  
  
        String1++;  
  
    }  
  
    SendInstruction(0xC0); // Set address to 2nd line while(*String2)  
  
    {  
  
        SendData(*String2);  
  
        String2++;  
  
    }  
  
    while(1);  
  
}
```



## **EXP –**

### **07 : TIMER WITH PIC18F4550 :-**

#### **CALCULATIONS :- \***

Required time = 100ms

\* TMR value = 0xFFFF - (Required time) / (4 \* Tosc \* Prescaler)  
\* = 0xFFFF - (0.1 \* 48000000) / (4 \* 256)  
\* = 0xFFFF - 0x124F  
\* TMR = 0xEDB0  
\* TMRH = 0xED  
\* TMRL = 0xB0  
\*/

#### **PROGRAM :-**

```
#include<p18f4550.h>
volatile bit timer_set = 0;
void timerInit(void)
{
    // Timer0 configuration
    TOCON = 0b00000111; // Timer0 16-bit; prescaler 1:256
    TMROH = 0xED;
    TMROL = 0xB0;
}
void Interrupt_Init(void)
{
    RCONbits.IPEN = 1;
    INTCON = 0b11100000; // Enable global and Timer0 interrupts; Clear Timer0 interrupt flag
    INTCON2bits.TMROIP = 0;
}
void interrupt low_priority timerinterrupt(void)
{
    if(TMROIF == 1) // If Timer0 interrupt flag is set.....
```

```
{  
    TMROON = 0; // Stop the timer  
    TMROIF = 0; // Clear the interrupt flag  
    TMROH = 0xED; // Reload Timer0  
    TMROL = 0xB0;  
    LATB = ~LATB; // Toggle PORTB  
    TMROON = 1; // Start the timer  
}  
}  
  
void main(void)  
{  
    TRISB = 0x00;  
    LATB = 0xFF;  
    Interrupt_Init(); timerInit();  
    TMROON = 1; // Start the timer while(1);  
    //Loop forever; do nothing  
}
```

## **EXP –**

### **BAUD RATE GENERATION :-**

```
* n => required baudrate  
* BRGH = 0  
* SPBRG = (Fosc / (64 * n)) -1  
* For 9600 baudrate, SPBRG ~=77  
*/
```

### **PROGRAM :-**

```
#include<p18F4550.h>  
#include<stdio.h> #define Fosc  
48000000UL void  
InitUART(unsigned int baudrate)  
{  
    TRISCbits.RC6 = 0; // TX pin set as output  
    TRISCbits.RC7 = 1; // RX pin set as input  
    SPBRG = (unsigned char)((Fosc /64)/baudrate)-1;  
    BAUDCON = 0b00000000; // Non-inverted data; 8-bit baudrate generator  
    TXSTA = 0b00100000; // Asynchronous 8-bit; Transmit enabled; Low speed baudrate select  
    RCSTA = 0b10010000; // Serial port enabled; 8-bit data; single receive enabled  
}  
void SendChar(unsigned char data)  
{  
    while(TXSTAbits.TRMT == 0); // Wait while transmit register is empty  
    TXREG = data; // Transmit data  
}  
void putch(unsigned char data)  
{  
    SendChar(data);
```

```
}

unsigned char GetChar(void)
{
    while(!PIR1bits.RCIF); // Wait till receive buffer becomes full return
    RCREG; // Returned received data
}

void main(void)
{
    InitUART(9600);
    printf("\r\nHello MicroPIC-18F: Enter any Key from Keyboard\r\n");
    while(1)
    {
        printf("%c",GetChar()); // Receive character from PC and echo back
    }
    while(1);
}
```

## EXP –

```
#include <p18f4550.h>
#include<stdio.h>

#define LCD_EN LATCbits.LC1
#define LCD_RS LATCbits.LC0
#define LCDPORT LATB

void lcd_delay(unsigned int time)
{
    unsigned int i , j ;

    for(i = 0; i < time; i++)
    {
        for(j=0;j<50;j++);
    }
}

void SendInstruction(unsigned char command)
{
    LCD_RS = 0;          // RS low : Instruction
    LCDPORT = command;
    LCD_EN = 1;           // EN High
    lcd_delay(10);
    LCD_EN = 0;           // EN Low; command sampled at EN falling edge
    lcd_delay(10);
}

void SendData(unsigned char lcddata)
{
    LCD_RS = 1;          // RS HIGH : DATA
    LCDPORT = lcddata;
```

```

LCD_EN = 1;           // EN High
lcd_delay(10);

LCD_EN = 0;           // EN Low; data sampled at EN falling edge
lcd_delay(10);

}

void InitLCD(void)
{
    TRISB = 0x00; //set data port as output
    TRISCbits.RC0 = 0; //EN pin
    TRISCbits.RC1 = 0; // RS pin

    SendInstruction(0x38); //8 bit mode, 2 line,5x7 dots
    SendInstruction(0x06); // entry mode
    SendInstruction(0x0C); //Display ON cursor OFF
    SendInstruction(0x01); //Clear display
    SendInstruction(0x80); //set address to 0
}

void ADCInit(void)
{
    TRISEbits.RE1 = 1;      //ADC channel 6 input
    TRISEbits.RE2 = 1;      //ADC channel 7 input

    ADCON1 = 0b00000111;   //Ref voltages Vdd & Vss; AN0 - AN7 channels Analog
    ADCON2 = 0b10101110;   //Right justified; Acquisition time 4T; Conversion clock
    Fosc/64

}

unsigned short Read_ADC(unsigned char Ch)
{

```

```

ADCON0 = 0b00000001 | (Ch<<2); //ADC on; Select channel;
GODONE = 1; //Start Conversion

while(GO_DONE == 1 ); //Wait till A/D conversion is complete
return ADRES; //Return ADC result
}

void DisplayResult(unsigned short ADCVal)
{
    unsigned char i,text[16];
    unsigned short tempv; tempv
    = ADCVal;

    SendInstruction(0x80); //set to 1st line
    for(i=0;i<10;i++) //Display the 10 bit ADC result on LCD
    {
        if(tempv & 0x200)
        {
            sendData('1');
        }
        else
        {
            sendData('0');
        }
        tempv = tempv<<1;
    }

    ADCVal = (5500/1024)*ADCVal; //Convert binary data to mV; 1 bit <=>
    (5500/1024)mV
    sprintf(text,"ADC value=%4dmv",ADCVal); //Convert integer data to string
}

```

```
SendInstruction(0xC0);           //set to 2nd line
for(i=0;i<16;i++)             //Display string on LCD
{
    SendData(text[i]);
}

}

void main()
{
    unsigned short Ch_result;

    TRISB = 0x00;           //PORTB connected to LCD is output
    ADCInit();
    InitLCD();

    while(1)
    {
        Ch_result = Read_ADC(7);
        DisplayResult(Ch_result);    lcd_delay(1000);
    }
}
```