ANNA UNIVERSITY-CHENNAI

MADRAS INSTITUTE OF TECHNOLOGY

CHROMPET, CHENNAI – 600 044

DEPARTMENT OF PRODUCTION TECHNOLOGY

MR7212-MICROCONTROLLERS LABORATORY

NAME	
REG NO	
YEAR	
SEMESTER	
BRANCH	**ME MECHATRONICS**
DATE OF END SEM EXAMINATION	

BONAFIDE CERTIFICATE

NAME :

REGISTER NO. :

SUBJECT :

DEPARTMENT :

Certified to be bonafide record of practical work done by Mr./Miss. ..…………………………………………………… in the …………………………………………….. Laboratory during the period …………………….20…...

Date: **Staff-In-Charge**

Submitted for the practical examination held on …………….

LIST OF EXPERIMENTS

MICROCONTROLLERS LABORATORY

Ex.No : 1
Date :

Assembly language programming and simulation of 8051 in Keil IDE

A. Finding the average of numbers

Aim

To find the average of a set of input hexadecimal numbers using assembly language.

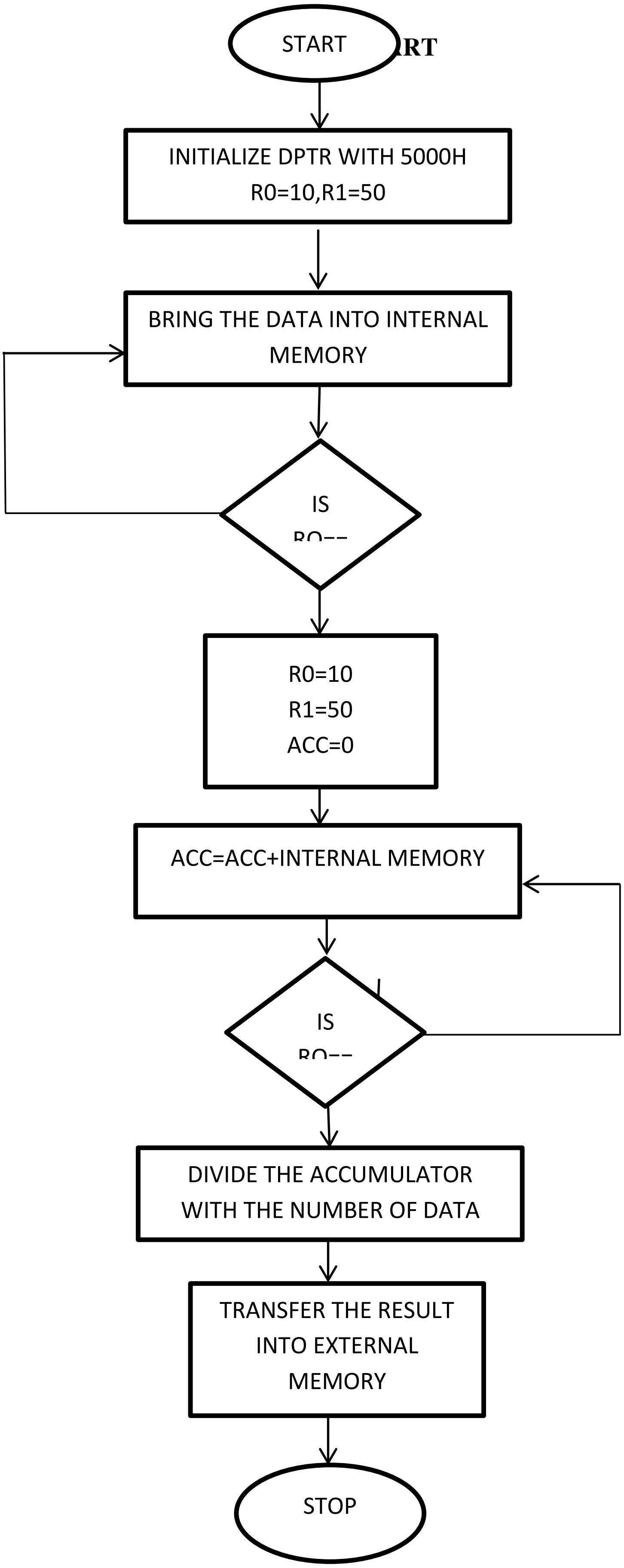
Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision software

Procedure

1. Create a new project in Keil software.
2. Select the Controller as AT89C51
3. Open a new script and type the program.
4. Save the program as '.asm' file.
5. Add this file to the Source Group folder in the Target folder.
6. Build a target location for the program by clicking 'Build Target' option in Project tab.
7. Now start executing the program by clicking 'Start Debug Session' option in Debug tab.
8. Check for the errors and warning and finally Run the Program
9. The output can be viewed from the Project status window.

FLOW CHART


START
RT
INITIALIZE DPTR WITH 5000H
R0=10,R1=50
BRING THE DATA INTO INTERNAL MEMORY
IS
RO==
R0=10
R1=50
ACC=0
ACC=ACC+INTERNAL MEMORY
IS
RO==
DIVIDE THE ACCUMULATOR WITH THE NUMBER OF DATA
TRANSFER THE RESULT INTO EXTERNAL MEMORY
STOP

PROGRAM

```
MOV DPTR,#4200H
MOVX A,@DPTR
MOV R0,A
MOV B,#00H
MOV R1,B
INC DPTR
LOOP1:  CLR C
MOVX A,@DPTR
ADD A,B
MOV B,A
JNC LOOP2
INC R1
LOOP2: INC DPTR
DJNZ R0, LOOP1
MOV DPTR,#4500
MOV A,R1
MOVX @DPTR,A
INC DPTR
MOVX @DPTR,#4200H
MOVX A,@DPTR
MOV R2,A
MOV A,B
MOV B,R2
DIV AB
MOV DPTR,#5000H
MOV A,@DPTR
END
```

Inference:

1.
2.
3.
4.

Result

Thus average number for a given set of number is computed and verified.

****

Ex.No : 1
Date :

Assembly language programming and simulation of 8051 in Keil IDE

B. Arranging numbers in ascending and descending order

Aim

To arrange numbers in ascending and descending order from a given set of input hexadecimal numbers using assembly language.

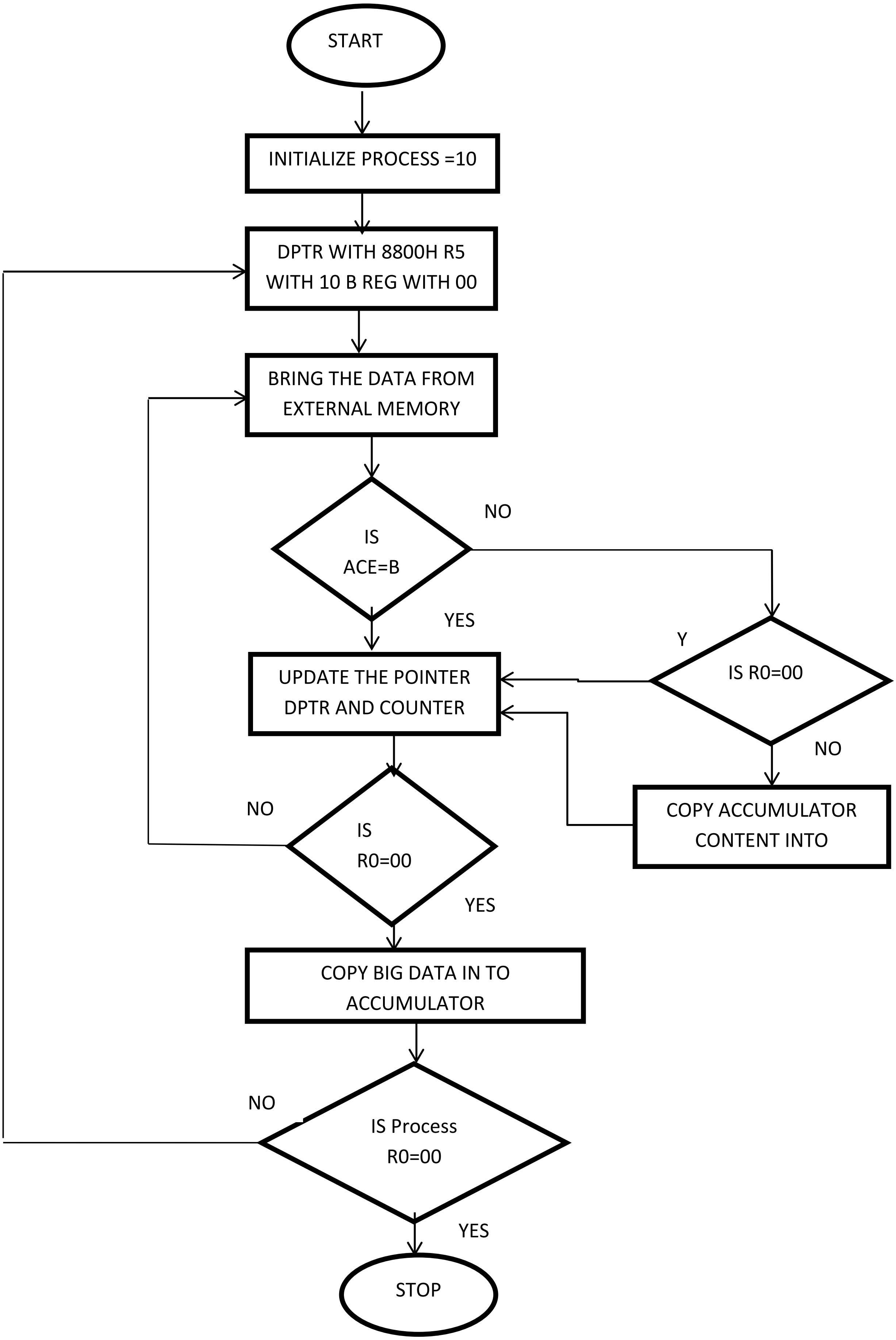
Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision software

Procedure

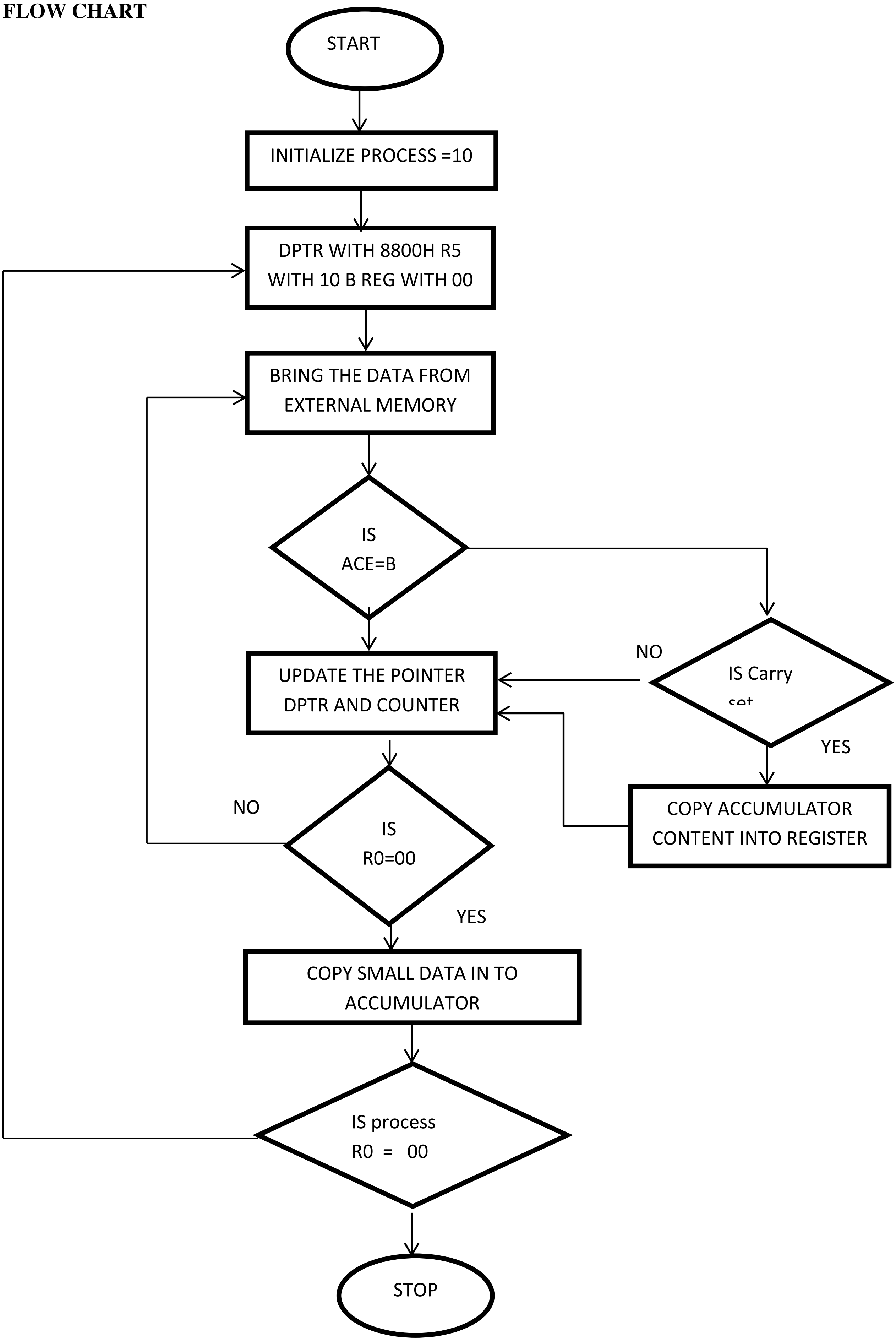
1. Create a New project in Keil software.
2. Select the Controller as AT89C51
3. Open a new script and type the program.
4. Save the program as '.asm' file.
5. Add this file to the Source Group folder in the Target folder.
6. Build a target location for the program by clicking 'Build Target' option in Project tab.
7. Now start executing the program by clicking 'Start Debug Session' option in Debug tab.
8. Check for the errors and warning and finally Run the Program.
9. The output can be viewed from the Project status window.

FLOW CHART



START
INITIALIZE PROCESS =10
DPTR WITH 8800H R5 WITH 10 B REG WITH 00
BRING THE DATA FROM EXTERNAL MEMORY
IS ACE=B
NO
YES
Y
IS R0=00
NO
UPDATE THE POINTER DPTR AND COUNTER
COPY ACCUMULATOR CONTENT INTO
NO
IS R0=00
YES
COPY BIG DATA IN TO ACCUMULATOR
NO
IS Process R0=00
YES
STOP

FLOW CHART


START
INITIALIZE PROCESS =10
DPTR WITH 8800H R5 WITH 10 B REG WITH 00
BRING THE DATA FROM EXTERNAL MEMORY
IS ACE=B
NO
IS Carry set
UPDATE THE POINTER DPTR AND COUNTER
YES
COPY ACCUMULATOR CONTENT INTO REGISTER
NO
IS R0=00
YES
COPY SMALL DATA IN TO ACCUMULATOR
IS process R0 = 00
STOP

PROGRAM FOR ASCENDING

```
        org 8400h
        mov dptr,#8800h
        mov r0,#40h
        mov r5,#0ah
next:   movx a,@dptr
        mov @r0,a
        inc dptr
        inc r0
        djnz r5,next
        mov r4,#09h
top:    mov r0,#40h
        mov a,r4
        mov r5,a
again:  clr c
        mov a,@r0
        inc r0
        subb a,@r0
        jc next1
        mov a,@r0
        dec r0
        xch a,@r0
        inc r0
        mov @r0,a
next1:  djnz r5,again
        djnz r4,top
        mov r0,#40h
        mov dptr,#8700h
        mov r7,#0ah
next2:  mov a,@r0
        movx @dptr,a
        inc dptr
        inc r0
        djnz r7,next2
sun:    sjmp sun
        end
```

PROGRAM FOR DESCENDING ORDER

```
        org 00h
l1:     mov r3,#04h
        mov r2,#04h
        mov r1,#40h
l2:     mov a,@r1
        mov b,a
        inc r1
        mov a,@r1
        mov r4,a
        subb a,b
        mov a,r4
        jc l4
        dec r1
        mov @r1,a
        inc r1
        mov a,b
        mov @r1,a
l4:     djnz r1,l1
        djnz r3,l2
l3:     sjmp l3
        org 0300h
data:   db 45h,64h,99h,33h
        end
```

Inference:

1.
2.
3.
4.

Result

Thus the given set of numbers is arranged in ascending, descending order and output is verified.

****

Ex.No : 2
Date :

A. Alphanumeric LCD interfacing using 8051 and PIC Microcontroller

Aim

To interface alphanumeric LCD(16X2) using 8051 and PIC microcontroller

Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision software (µC)
3	EZ Downloader
4	MPLAB software (PIC)
5	8051 trainer kit
6	PIC trainer kit
7	Serial cable

Procedure

For AT89C51,

1. Create a new project in keil software.
2. Select the controller as AT89C51.
3. Open a new script and type the program.
4. Save the program as '.asm' file.
5. Add this file to the source group folder in the target folder.
6. Build a target location for the program by clicking 'build target' option in project tab.
7. Now start executing the program by clicking 'start debug session' option in debug tab.
8. Check for the errors and warning and finally run the program.
9. The output can be viewed from the project status window and hex files also generated.
10. Connect kit and system using serial cable and embed the .hex file in the trainer kit using EZ Downloader then RESET AT89C51.

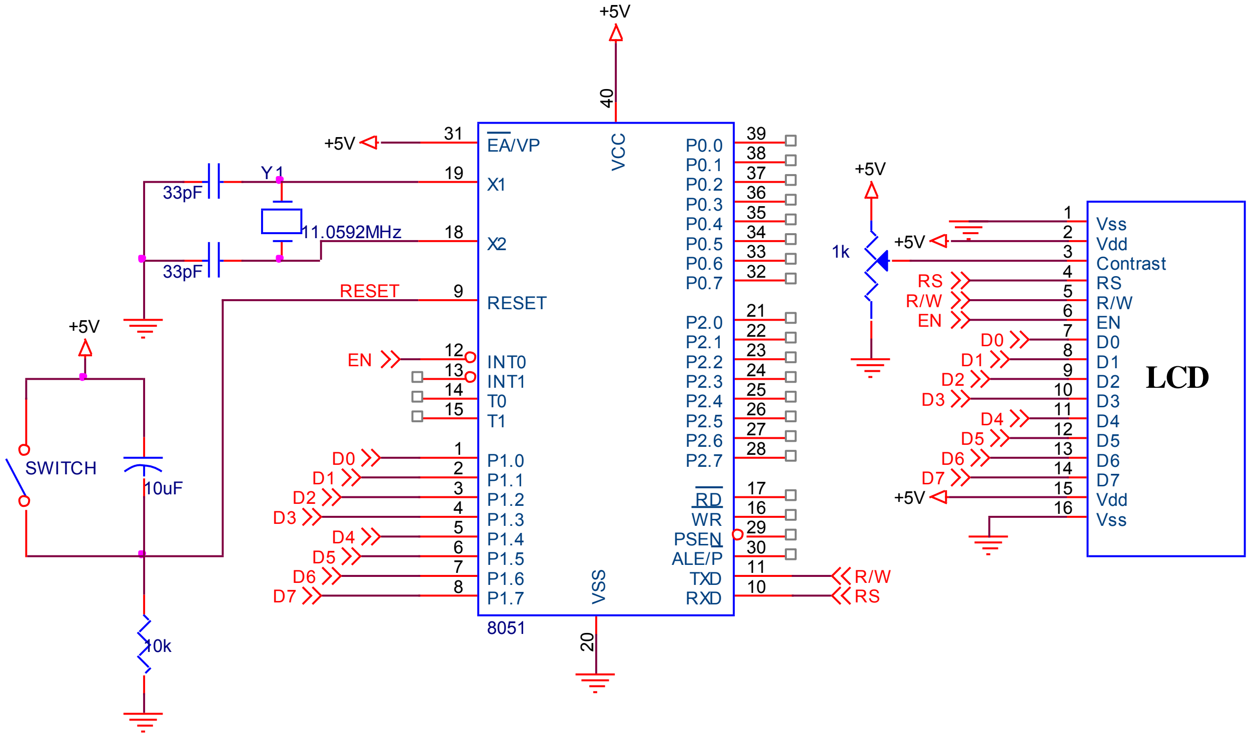
For PICF458,

1. Create a new project in MPLAB software.
2. Select the controller as PIC18F458
3. Open a new script and type the program.
4. Save the program as '.c' file.
5. Add this file to the Source Group folder in the Target folder.
6. Build a target location for the program by clicking 'Build Target' option in Project tab.
7. Now start executing the program by clicking 'Start Debug Session' option in Debug tab.
8. Check for the errors and warning and finally Run the Program
9. The output can be viewed from the Project status window.

PROGRAM : Alphanumeric LCD interfacing using 8051

```
#include<reg51.h>

sfr lcd=0x90;     // data of lcd at port 2
sbit rs=P3^0;     // rs pin at P3.0
sbit rw=P3^1;     // rw pin at P3.1
sbit en=P3^2;     // en pin at P3.2

void delay();     // for delay
void cmd();       // lcd in command mode
void display();   // lcd is in display mode
void main()
  {
          while(1)
           {
            // LCD INITIALIZE START
                    lcd=0x38;
                    cmd();
                    lcd=0x0e;
                    cmd();
                    lcd=0x01;
                    cmd();
                    lcd=0x06;
                    cmd();
                    lcd=0x80;
                    cmd();
            // LCD INITIALIZE END
            // DATA DISPLAYING ON LCD
                    lcd='W';
                    display();
                    lcd='E';
                    display();
                    lcd='L';
                    display();
                    lcd='C';
                    display();
                    lcd='O';
                    display();
                    lcd='M';
                    display();
                    lcd='E';
                    display();
                    lcd=' ';
                    display();
                    lcd='T';
                    display();
                    lcd='O';
                    display();
                    lcd=0xc0;  //NEXT LINE COMMAND
                    cmd();
                    lcd='M';
                    display();
                    lcd='E';
                    display();
                    lcd='C';
                    display();
                    lcd='H';
                    display();
                    lcd='A';
                    display();
                    lcd='T';
                    display();
                    lcd='R';
                    display();
                    lcd='O';
                    display();
                    lcd='N';
                    display();
                    lcd='I';
                    display();
                    lcd='C';
                    display();
                    lcd='S';
                    display();

                }

    }
void cmd()
 {
     unsigned char i;
     rs=0;
     rw=0;
     en=1;
     for(i=0;i<2;i++);
     en=0;
     delay();
 }
void display()
  {
     unsigned char i;
     rs=1;
     rw=0;
     en=1;
     for(i=0;i<2;i++);
     en=0;
     delay();
  }
void delay()
  {
      unsigned int i,j;
      for(i=0;i<1275;i++)
         for(j=0;j<1275;j++);
  }
```

CIRCUIT DIAGRAM: ALPHANUMERIC LCD INTERFACING USING 8051



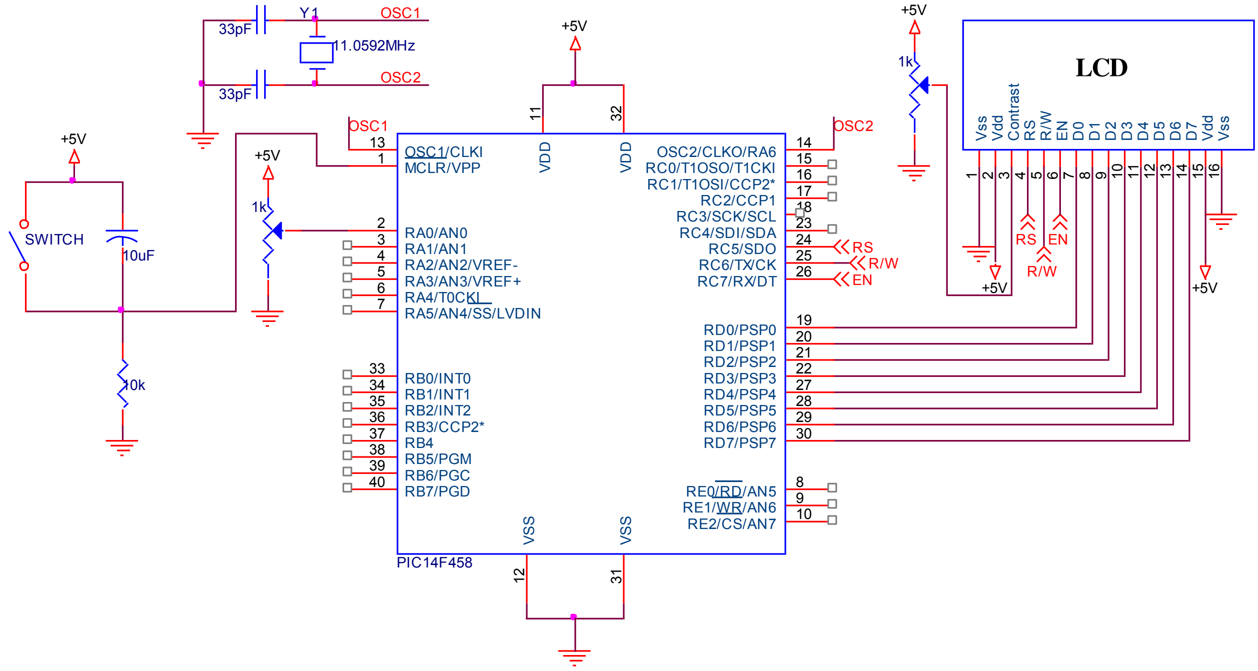
PROGRAM: ALPHANUMERIC LCD INTERFACING USING PIC MICROCONTROLLER

```
/* header file used in this program is already included in software MPLAB for pic*/

#define lcd PORTD      // data of lcd at port d
#define rs portc.f5    // rs pin at 5 pin of portc
#define rw portc.f6    // rw pin at 6 pin of portc
#define en portc.f7    // en pin at 7 pin of portc

void delay();    // for delay
void cmd();      // lcd in command mode
void display();  // lcd is in display mode


void main()
  {
     TRISC=0X00;
     TRISD=0X00;
    while(1)
         {
                  lcd=0x38;
                  cmd();
                  lcd=0x0e;
                  cmd();
                  lcd=0x01;
                  cmd();
                  lcd=0x06;
                  cmd();
                  lcd=0x80;
                  cmd();
                  lcd='W';
                  display();
                  lcd='E';
                  display();
                  lcd='L';
                  display();
                  lcd='C';
                  display();
                  lcd='O';
                  display();
                  lcd='M';
                  display();
                  lcd='E';
                  display();
                  lcd=' ';
                  display();
                  lcd='T';
```

```
                        display();
                        lcd='O';
                        display();
                        lcd=0xc0;
                        cmd();
                        lcd='M';
                        display();
                        lcd='E';
                        display();
                        lcd='C';
                        display();
                        lcd='H';
                        display();
                        lcd='A';
                        display();
                        lcd='T';
                        display();
                        lcd='R';
                        display();
                        lcd='O';
                        display();
                        lcd='N';
                        display();
                        lcd='I';
                        display();
                        lcd='C';
                        display();
                        lcd='S';
                        display();

          }
    }

void cmd()
  {
     unsigned char i;
     rs=0;
     rw=0;
     en=1;
     for(i=0;i<2;i++);
     en=0;
     delay();
   }

void display()
  {
     unsigned char i;
     rs=1;
     rw=0;
```

```
      en=1;
      for(i=0;i<2;i++);
      en=0;
      delay();
   }

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

CIRCUIT DIAGRAM: ALPHANUMERIC LCD INTERFACING USING PIC MICROCONTROLLER



Inference:

1.
2.
3.
4.

Result

Thus interfacing alphanumeric LCD (16X2) using 8051 and PIC microcontroller is done successfully.

****

Ex.No : 2
Date :

B. Graphical LCD interfacing using 8051

Aim

To interface GLCD(128X64 pixles) using 8051 and PIC microcontroller

Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision software (µC)
3	EZ Downloader
4	GLCD
5	Serial cable

Procedure

1. Create a new project in keil software.
2. Select the controller as AT89C51.
3. Open a new script and type the program.
4. Save the program as '.asm' file.
5. Add this file to the source group folder in the target folder.
6. Build a target location for the program by clicking 'build target' option in project tab.
7. Now start executing the program by clicking 'start debug session' option in debug tab.
8. Check for the errors and warning and finally run the program.
9. The output can be viewed from the project status window and hex files also generated. Connect kit and system using serial cable and embed the .hex file in the trainer kit using EZ Downloader then RESET AT89C51.

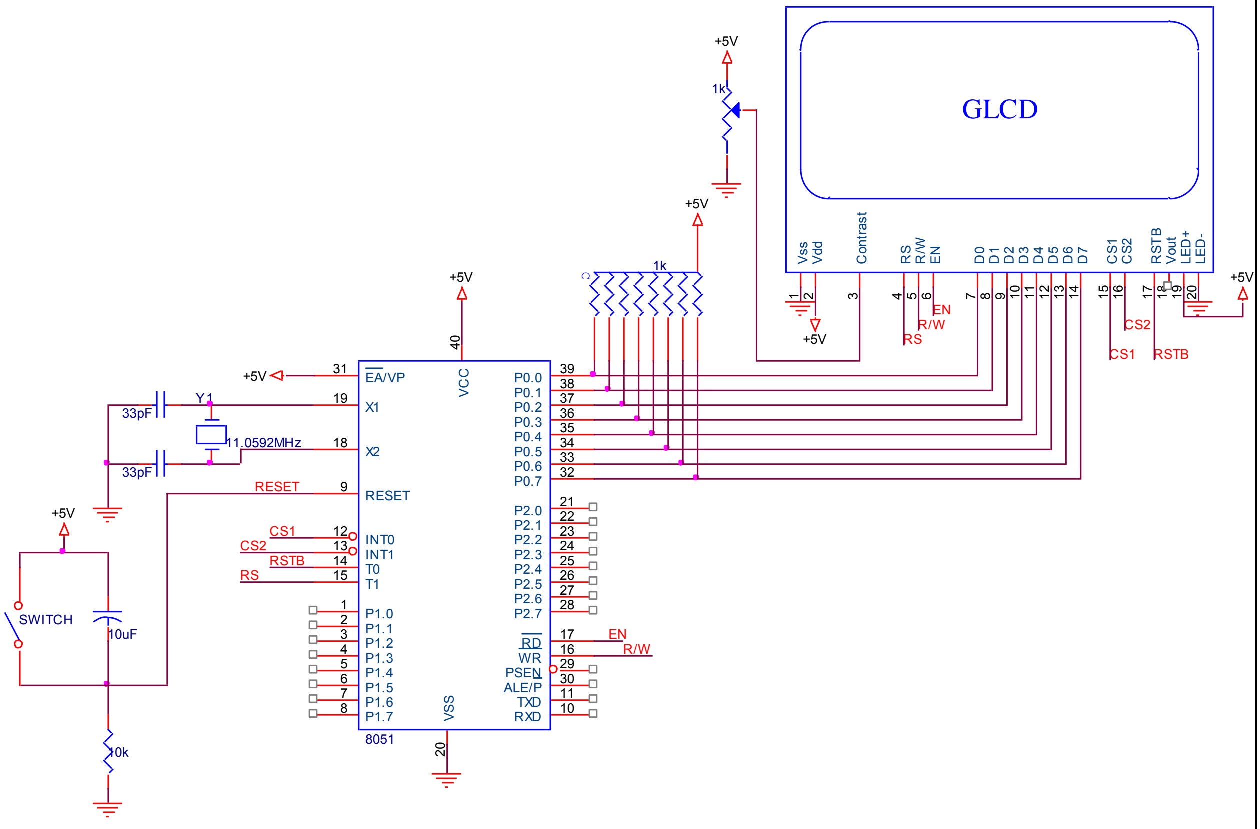
PROGRAM : Graphical LCD interfacing using 8051

```
#include <reg51.h>
#include <stdio.h>
#define DATA P0
sbit CS1     = P3^3;
sbit CS2     = P3^2;
sbit RS     = P3^5;
sbit RW = P3^6;
sbit lcd_e   = P3^7;
 sbit RST     = P3^4;

void GLCD_PutPicture(const unsigned char *);
void Select_page(unsigned char);

int i;

code unsigned char const AU_LOGO[1024]=
{
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x80,
0xc0,0x40,0x20,0x20,0x60,0xc0,0x00,0x00,0x00,0x00,0x00,0x00,0x80,0xf8,0x06,0x06,
0x06,0x06,0x06,0xfc,0x80,0x80,0x00,0x00,0x00,0x00,0x80,0xc0,0x20,0x30,0x30,0x20,
0x40,0xc0,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x80,0x80,0x80,0x00,0x00,0x00,0x00,0x80,0xc3,
0x6c,0x30,0x00,0xc0,0x40,0x81,0x83,0x02,0x3a,0x01,0x0d,0xb1,0x81,0x9c,0x8c,0x9c,
0x80,0x98,0x80,0x80,0xbc,0x91,0x8d,0x21,0x5b,0x0b,0x03,0xc0,0x60,0xa0,0x00,0x18,
0x3e,0x63,0xc0,0x80,0x00,0x00,0x80,0x80,0x80,0x80,0x80,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x80,0xc0,0x80,0x10,0x00,0x10,0x18,
0x0c,0x18,0x00,0x80,0x06,0x00,0x0e,0x00,0x0c,0x0e,0x00,0x08,0x14,0x30,0x28,0x60,
0x10,0x00,0x80,0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x0c,0x3e,0x61,0x41,0xc1,0x01,0x03,0x42,0xd3,0x51,0x64,
0x2c,0x97,0x46,0x31,0x19,0x08,0x04,0x02,0x02,0x01,0x01,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x04,0xfb,0x09,0x0b,0x0b,0x02,0x06,0x0c,0x08,0x30,0x64,0xca,
0x01,0x12,0x88,0x49,0x63,0x01,0x01,0xc0,0xe0,0x31,0x1b,0x0e,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x60,0x80,0x00,0x80,0x60,0x00,0x03,0x00,0x10,0xc0,0x10,0x10,
0x48,0x08,0xa0,0xe7,0x07,0x1f,0xfe,0xde,0xc0,0x08,0x70,0x00,0x10,0x10,0x00,0xe0,
0x90,0xc1,0x82,0x81,0x02,0x40,0x40,0x00,0x40,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0xc0,0x60,0x20,0x20,0x20,0x30,0x3e,0x03,0x00,0x06,0x0a,0x16,0x02,0xf8,
0xfe,0xb9,0xf8,0xfc,0xbe,0xfe,0xf0,0xf8,0xb8,0xfc,0xfe,0xbe,0xff,0xf0,0xd8,0xf8,
0xfc,0xde,0xfe,0xff,0xf8,0x9f,0xfc,0xfc,0xde,0xff,0xdf,0xf8,0xd8,0xfc,0xdc,0xdf,
0xff,0xfc,0x80,0x00,0x04,0x0e,0x08,0x01,0x0f,0x3c,0x30,0x30,0x30,0x60,0xc0,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x14,0x84,0xa0,0x63,0xff,0x00,0x00,0x00,0x00,0x00,0xff,0x1c,0x1c,
0x00,0x00,0xff,0xff,0x00,0xf8,0xff,0xff,0xff,0x00,0xff,0xfe,0x3e,0x0e,0x00,0xff,
0x7f,0x07,0x00,0x00,0x00,0xc3,0xce,0x42,0x09,0x30,0x00,0x00,0x00,0x00,0x00,0x80,

0x00,0x00,0x07,0x04,0x08,0x08,0x08,0x08,0xf0,0x00,0x00,0x00,0x00,0x00,0x00,0x3f,
0xc7,0x07,0x07,0x07,0x07,0x07,0xf7,0x17,0x17,0xb7,0x07,0x07,0x07,0x07,0x07,0x07,
0xf7,0x07,0x8f,0x0f,0x4f,0xef,0x87,0x07,0x07,0x07,0xf7,0x07,0x07,0x07,0x07,0x07,
0xc7,0x3f,0x00,0x00,0x00,0x00,0x00,0x00,0xf0,0x38,0x08,0x08,0x08,0x0c,0x07,0x00,
0x00,0x00,0xc0,0xc0,0xc0,0xc0,0x7c,0x3c,0x18,0x18,0x0e,0x0f,0x0f,0xfe,0x84,0x8c,
0xaf,0x3f,0x1e,0x19,0x3c,0x7c,0xfc,0xcd,0xc6,0xd8,0xe0,0x40,0x80,0x3f,0x00,0x00,
0x20,0x20,0x0f,0x7f,0xc0,0xff,0xff,0xff,0x1f,0x00,0x3f,0x00,0x00,0x00,0x40,0x39,
0x80,0x40,0xe0,0xd0,0xcc,0x9d,0xfc,0x3c,0x1f,0x18,0x8f,0x87,0x87,0x86,0x86,0x07,

0x00,0x00,0x00,0x00,0x00,0x60,0xf0,0x08,0x0c,0x07,0x00,0x00,0x80,0x80,0x00,0x00,
0x01,0x03,0x0c,0x18,0x30,0x60,0x41,0xbe,0xa0,0x20,0x22,0x20,0x20,0x2e,0x28,0x60,
0x7f,0x20,0x2d,0x20,0x20,0x20,0x2b,0x22,0xa0,0xbe,0x41,0x60,0x30,0x18,0x0c,0x03,
0x01,0x00,0x00,0x80,0x80,0x80,0x00,0x07,0x07,0x0c,0x9c,0x78,0x00,0x00,0x00,0x00,
0x00,0x18,0xb8,0xfd,0x07,0x01,0x00,0x00,0x00,0x00,0xf8,0xaa,0xf5,0x48,0xee,0x74,
0x3c,0x41,0x02,0x08,0x30,0x40,0x40,0x01,0xff,0xff,0x38,0x38,0x04,0x03,0x02,0x00,
0x20,0x00,0x30,0x40,0x71,0xf3,0x93,0x10,0x00,0x30,0x00,0x78,0x10,0x20,0x00,0x03,
0x04,0x04,0x30,0x73,0xff,0xc7,0x00,0x00,0x00,0x38,0x5a,0xcb,0x8b,0x87,0xef,0xfa,

0x00,0x00,0xe0,0xf0,0x30,0x30,0x31,0x33,0x32,0x32,0x22,0x33,0x31,0x21,0x36,0x34,
0x38,0x38,0x30,0x30,0x30,0x30,0x30,0x30,0x31,0x31,0x31,0x33,0x32,0x32,0x32,0x32,
0x32,0x32,0x32,0x32,0x32,0x31,0x31,0x31,0x30,0x30,0x30,0x30,0x30,0x30,0x30,0x38,
0x38,0x36,0x33,0x31,0x30,0x31,0x31,0x33,0x33,0x33,0x33,0x30,0x30,0x30,0xf0,0x00,
0x00,0x03,0x07,0x27,0x7e,0x78,0x70,0x70,0xdc,0x9d,0x84,0x84,0x01,0x00,0x00,0x04,
0x04,0x80,0x84,0xc8,0xc0,0x60,0x70,0x7c,0x67,0x07,0x07,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x06,0x0f,0x0f,0x6e,0xf8,0x70,0x60,0x46,0xc2,0xc2,0x82,0x86,0x8a,

0x00,0x00,0x3f,0x3f,0x20, 0x27,0x23,0x20,0x23,0x26,0x25,0x27,0x67,0x67,0x20,0x23,
0x25,0x27,0x20,0x27,0x25,0x27,0x27,0x20,0x27,0x27,0x22,0x20,0x22,0x27,0x27,0x25,
0x23,0x24,0x22,0x24,0x26,0x22,0x27,0x20,0x23,0x24,0x23,0x27,0x23,0x24,0x23,0x26,
0x27,0x21,0x24,0x20,0x24,0x27,0x24,0x20,0x24,0x27,0x27,0x24,0x20,0x30,0x3f,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0x03,0x01,0x01,0x01,0x07,0x07,0x03,
0x01,0x01,0x03,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x03,0x01,0x00,0x00,

};
```

```
void GLCD_Init();
void GLCD_Data(unsigned char);
void GLCD_Comd(unsigned char);
void DelayMs(int);

void GLCD_Comd(unsigned char cmnd)
{
   DATA = cmnd;     //send command to port
   RS = 0;              //make it RS to Low
   RW = 0;               //make it RW to low
   lcd_e = 1;           //enbale high
   DelayMs(10);
   lcd_e = 0;           //enable low
}
void GLCD_Data(unsigned char dat)
{
   DATA = dat;          //send command to port
   RS = 1;              //make it RS to high
   RW = 0;               //make it RW to low
   lcd_e = 1;           //enbale high
   DelayMs(10);
   lcd_e = 0;           //enbale low
}
void DelayMs(int k)
   {
       unsigned int a;
       for(a=0;a<=k;a++);
   }
void GLCD_Init()
{
   unsigned char Comd[5]={0x3f,0xc0,0xb8,0x40};//LCD Command list
   Select_page(1);          //send commands to page1
   for(i=0;i<4;i++)
   GLCD_Comd(Comd[i]);
   Select_page(0);          //send commands to page0
   for(i=0;i<4;i++)
   GLCD_Comd(Comd[i]);
}
void Select_page(unsigned char Page)
{
   if(Page)
   {
       CS1=0;      //Page 0 LCD IC1
       CS2=1;
   }
   else
       {
          CS1=1;   //Page 1 LCD IC2
```

```
            CS2=0;
        }
}
void GLCD_PutPicture(const unsigned char *ip) //Change here for method 1,2 and 3
{
  int Page=0,i=0;
  int Column=0;

  for (Page = 0; Page < 8; Page++)
   {
        Select_page(1);                      //Display part of image to Page1
        GLCD_Comd(0xb8 | Page);
        GLCD_Comd(0x40);
    for (Column = 0; Column < 128; Column++)
        {
          if (Column == 64)
          {
                Select_page(0);                  //Display part of image to Page0
                GLCD_Comd(0xb8 | Page);
                GLCD_Comd(0x40);
          }
            GLCD_Data(*ip++);
       }
   }
}
void main(void)
{
    DelayMs(2);
    RST = 1;
    DelayMs(5);
    RST =   0;
    DelayMs(5);
    RST =    1;
    DelayMs(5);
    GLCD_Init();             //Initialize GLCD
    DelayMs(15);
    GLCD_PutPicture(AU_LOGO);//Display Image
                      while(1);                    //wait forever
}
```

CIRCUIT DIAGRAM: Graphical LCD interfacing using 8051



Inference:

1.
2.
3.
4.

Result

Thus the GLCD is interfaced with 8051 successfully.

****

Ex.No : 3
Date :

Sensor interfacing with ADC to 8051 and PIC

Aim

To interface sensor with ADC to 8051 and PIC

Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision Software
3	MPLAB Software
4	8051 trainer kit
5	PIC trainer kit
6	LDR
7	10k POT

Procedure

For AT89C51,

1. Create a new project in keil software.
2. Select the controller as AT89C51.
3. Open a new script and type the program.
4. Save the program as '.asm' file.
5. Add this file to the source group folder in the target folder.
6. Build a target location for the program by clicking 'build target' option in project tab.
7. Now start executing the program by clicking 'start debug session' option in debug tab.
8. Check for the errors and warning and finally run the program.
9. The output can be viewed from the project status window and hex files also generated. Connect kit and system using serial cable and embed the .hex file in the trainer kit using EZ Downloader then RESET AT89C51.

For PICF458,

1. Create a New project in MPLAB software.
2. Select the Controller as PIC18F458
3. Open a new script and type the program.
4. Save the program as '.c' file.
5. Add this file to the Source Group folder in the Target folder.
6. Build a target location for the program by clicking 'Build Target' option in Project tab.
7. Now start executing the program by clicking 'Start Debug Session' option in Debug tab.
8. Check for the errors and warning and finally Run the Program.
9. The output can be viewed from the Project status window.

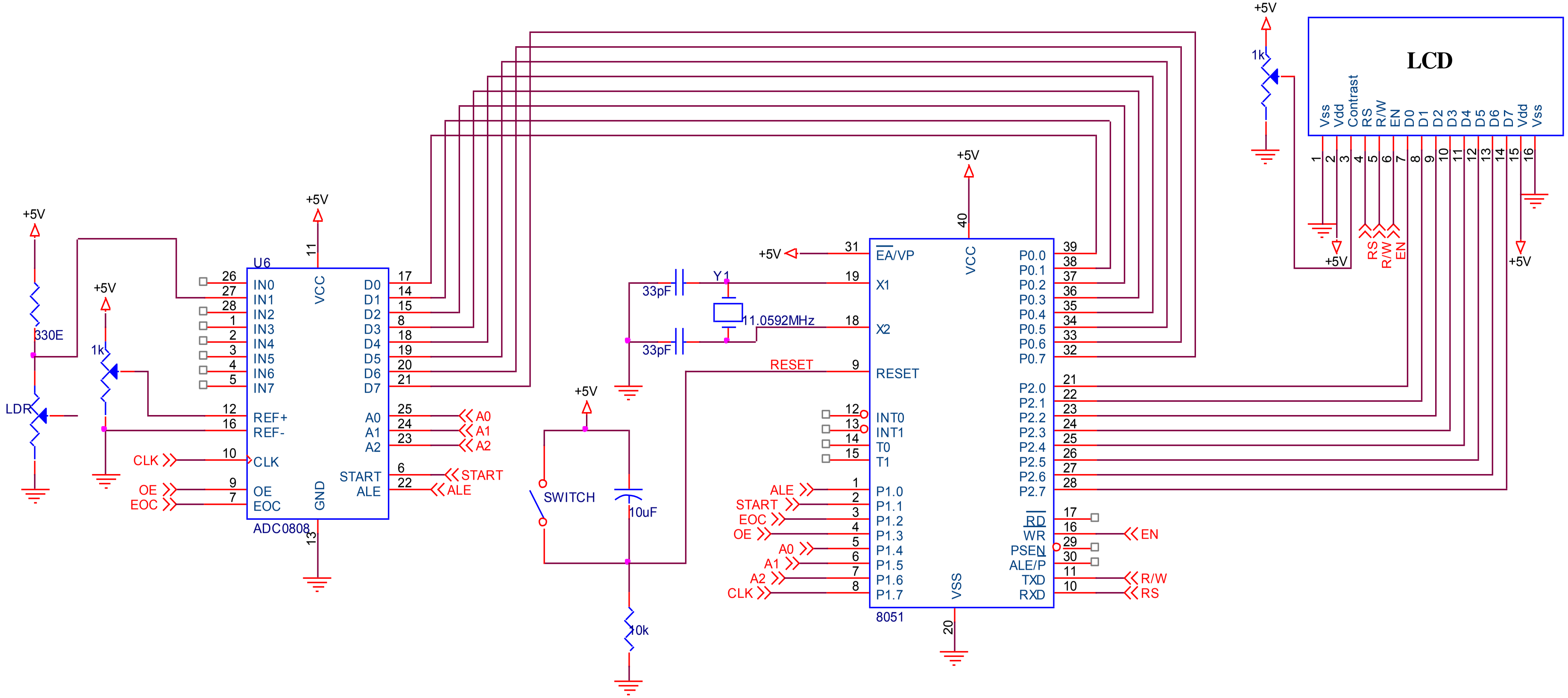
PROGRAM: SENSOR INTERFACING WITH ADC TO 8051

```
// Program to interface LDR using ADC 0808. The output of LDR is displayed on LCD.
Controller interrupt is used to generate the clock for driving ADC 0808.

#include<reg51.h>
sbit ale=P1^0;  //address latch enable
sbit oe=P1^3;  //output enable
sbit sc=P1^1;  //start conversion
sbit eoc=P1^2;  //end of conversion
sbit clk=P1^7;  // clock
sbit ADD_A=P1^4;  // Address pins for selecting input channels.
sbit ADD_B=P1^5;
sbit ADD_C=P1^6;
sfr lcd_data_pin=0xA0;  //P2 port
sbit rs=P3^0;
sbit rw=P3^1;
sbit en=P3^6;
sfr input_port=0x80;  //P0 port
unsigned int bitvalue,decimal_value,key,left_value,value,number,ascii1,ascii2,ascii3,flag,key1;

void timer0() interrupt 1  // Function to generate clock of frequency 500KHZ using Timer 0
interrupt.
{
        clk=~clk;
}

void delay(unsigned int count)  // Function to provide time delay in msec.
{
        int i,j;
        for(i=0;i<count;i++)
        for(j=0;j<1275;j++);
}

void lcd_command(unsigned char comm)  //Function to send command to LCD.
{
        lcd_data_pin=comm;
        en=1;
        rs=0;
        rw=0;
        delay(10);
        en=0;
}

void lcd_data(unsigned char disp)  //Function to send data to LCD.
{
        lcd_data_pin=disp;
        en=1;
        rs=1;
        rw=0;
        delay(10);
        en=0;
}
```

```
lcd_dataa(unsigned char *disp)  //Function to send string data to LCD.
{
int x;
for(x=0;disp[x]!=0;x++)
{
               lcd_data(disp[x]);
}
}

void lcd_ini()  //Function to inisialize the LCD
{
             lcd_command(0x38);
             delay(5);
             lcd_command(0x0F);
             delay(5);
             lcd_command(0x80);  //Force cursor to blink at line 1 positon 0
             delay(5);
}

  void BCD()  // Binary to decimal conversion to send the data to LCD
{
               key1++;
               key=0;
               flag=0;
                 number=input_port;
                 value=number%10;
             number=number/10;
             ascii1=value+48;
if(number!=0)
{
               value=number%10;
               number=number/10;
               ascii2=value+48;
                 flag=1;
}
else
{
                ascii2=48;
                flag=1;
}
if(number!=0)
{
               value=number%10;
                 number=number/10;
                 ascii3=value+48;
                 key=2;
}
else
{
               ascii3=48;
               key=2;
}
             if(key==2)
             lcd_data(ascii3);
             if(flag==1)
```

```
            lcd_data(ascii2);
            lcd_data(ascii1);
            if(key1==3)
{
            key1=0;
            ascii3=0;
            ascii2=0;
            ascii1=0;
            delay(10);
}
}

void adc()  //Function to drive ADC
{
while(1)
{
              ADD_C=0;  // Selecting input channel 2 using address lines
              ADD_B=0;
              ADD_A=1;
              delay(2);
              ale=1;
              delay(2);
              sc=1;
              delay(1);
              ale=0;
              delay(1);
              sc=0;
              while(eoc==1);
              while(eoc==0);
              oe=1;
              BCD();
              lcd_command(0x88);
              delay(2);
              oe=0;
}
}

void main()
{
            eoc=1;
            ale=0;
            oe=0;
            sc=0;
            key1=0;
            TMOD=0x02;  //timer0 setting for generating clock of 500KHz using interrupt enable
            mode.
            TH0=0xFD;
            IE=0x82;
            TR0=1;
            lcd_ini();
            lcd_dataa("Value : ");
            lcd_command(0x88);
            adc();
}
```

CIRCUIT DIAGRAM: SENSOR INTERFACING WITH ADC TO 8051



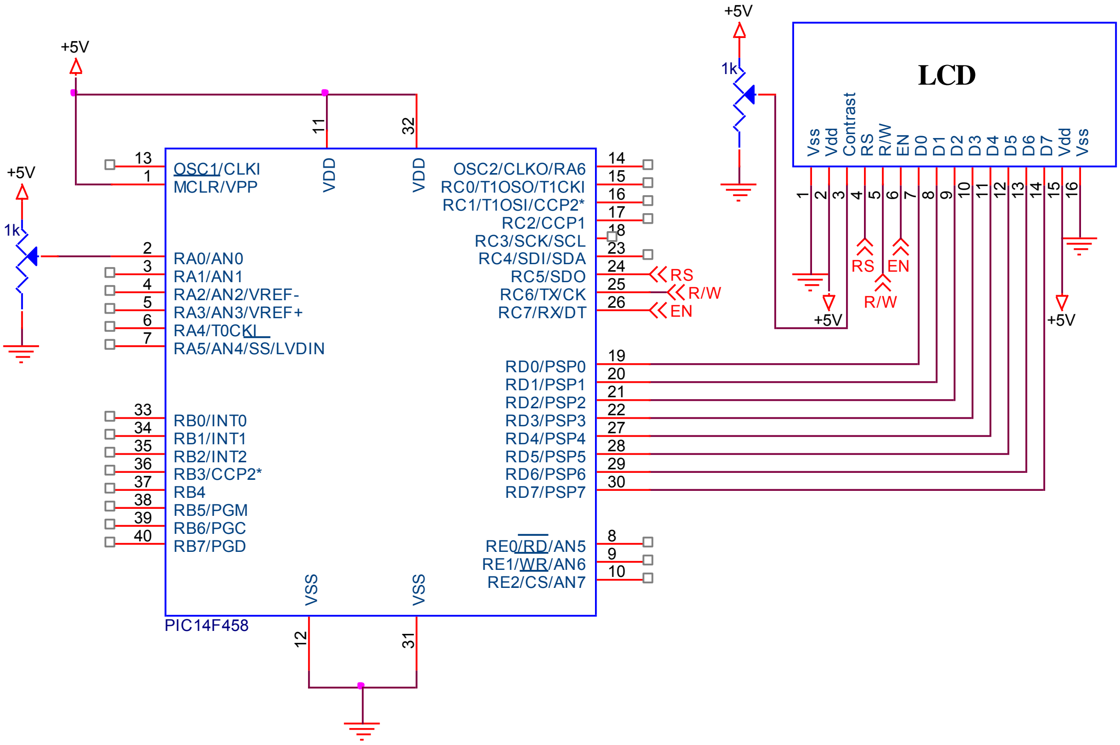
PROGRAM: SENSOR INTERFACING WITH ADC TO PIC18F458

```
/* header file used in this program is already included in software MPLAB for pic*/

#define lcd PORTD     // data of lcd at port d
#define rs portc.f5     // rs pin at 5 pin of portc
#define rw portc.f6     // rw pin at 6 pin of portc
#define en portc.f7     // en pin at 7 pin of portc

void delay();     // for delay
void cmd(unsigned char);       // lcd in command mode
void display(unsigned char);   // lcd is in display mode
int lcd_pos(int x,int y);
void lcd_str(unsigned char *);
void lcd_ini();
void dec_hex(unsigned long);
unsigned long int adc_re();


void main()
  {
     TRISC=0X00;
     TRISD=0X00;
     TRISA.TRISA0=0;
     ADCON0=0X81;
     ADCON1=0XCE;
     lcd_ini();
     lcd_str("VALUE OF ADC IS");
     while(1)
      {
        dec_hex(adc_re());
        delay_ms(300);

      }
   }

unsigned long adc_re()
  {
    float x;
    ADCON0.GO=1;
    while(ADCON0.DONE==1);
    x=ADRES;
    return x;
  }

void cmd(unsigned char x)
  {
     unsigned char i;
     lcd=x;
     rs=0;
     rw=0;
     en=1;
     for(i=0;i<2;i++);
     en=0;
     delay();
   }
```

```
void display(unsigned char x)
  {
     unsigned char i;
     lcd=x;
     rs=1;
     rw=0;
     en=1;
     for(i=0;i<2;i++);
     en=0;
     delay();
  }

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

int lcd_pos(int x,int y)
   {
     if(x==0)
          cmd(0x80+y);
     else if(x==1)
          cmd(0xc0+y);
   }

void lcd_str(unsigned char *x)
   {
    while(*x!='\0')
      {
          display(*x);
          x++;
      }
   }

void lcd_ini()
  {
    cmd(0x38);
    cmd(0x0e);
    cmd(0x01);
    cmd(0x06);
    cmd(0x80);
  }

void dec_hex(unsigned long temp)
   {
              unsigned char first,second,third,fourth;

              first=temp%10+'0';
              temp=temp/10;
              second=temp%10+'0';
              temp=temp/10;
              third=temp%10+'0';
              temp=temp/10;
```

```
            fourth=temp%10+'0';
            temp=temp/10+'0';
            lcd_pos(1,0);
            lcd_str("    ");
             lcd_pos(1,0);
            display(fourth);
            display(third);
            display(second);
            display(first);
 }
```

CIRCUIT DIAGRAM: SENSOR INTERFACING WITH ADC TO PIC18F458



Inference:

1.
2.
3.
4.

Result

Thus LDR interfaced with ADC to 8051 and a 10k pot interfaced with PIC18F458 successfully.

****

Ex.No : 4

Date :

DAC interfacing to 8051

Aim

To interface DAC using 8051

Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision Software
3	EZ Downloader
4	CRO
5	Serial cable

Procedure

For AT89C51,

1. Create a new project in keil software.
2. Select the controller as AT89C51.
3. Open a new script and type the program.
4. Save the program as '.asm' file.
5. Add this file to the source group folder in the target folder.
6. Build a target location for the program by clicking 'build target' option in project tab.
7. Now start executing the program by clicking 'start debug session' option in debug tab.
8. Check for the errors and warning and finally run the program.
9. The output can be viewed from the project status window and hex files also generated. Connect kit and system using serial cable and embed the .hex file in the trainer kit using EZ Downloader then RESET AT89C51.

PROGRAM: DAC interfacing to 8051

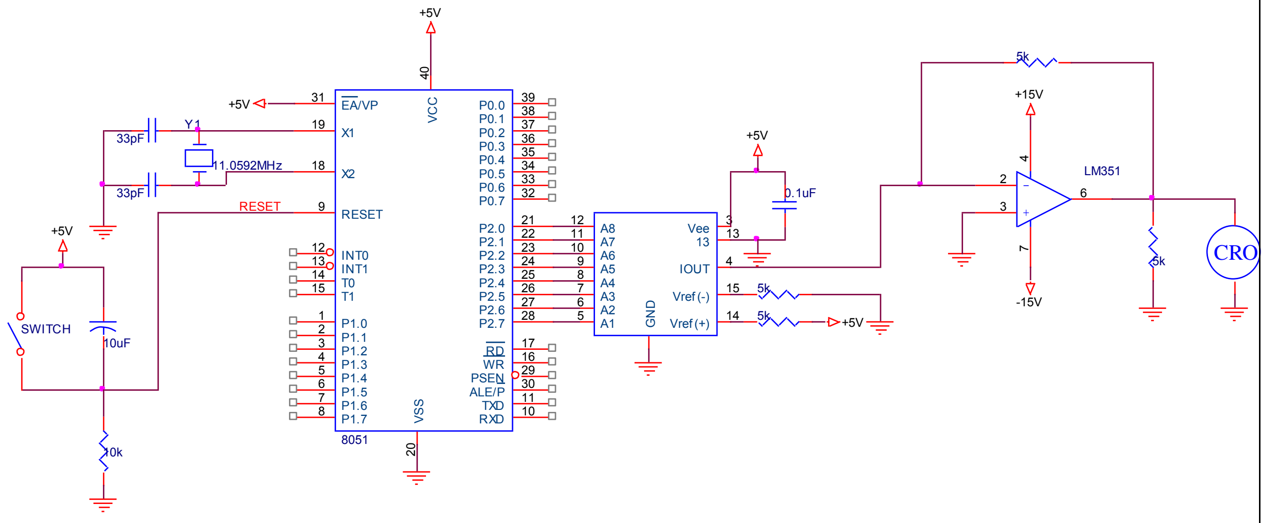
```
#include "Includes.h"

void InitDAC(void)
{
        DAC_Data_Bus = 0x00;  // Make Outputs
}

void Generate_DAC_Voltage(unsigned int mV)                    // Input should be in mV
{
         unsigned long V = ((unsigned long)mV * 25)/VREF;// Scale the input value
         V = V/98;                                            // Conversion factor

        DAC_Data_Bus = (unsigned char)V;                              // Assign proper
value to DAC inputs
}
```

```
#ifndef __INCLUDES_H
#define __INCLUDES_H

#include<reg51.h>
#include "DAC0808.h"

#endif
```

```
#include "Includes.h"

// Define Function Prototypes
void delay_sec(unsigned int);
void __delay_us(unsigned int);

// Main function
void main()
{
        P0 = 0x00;                          // Initialize all ports with a value of zero
        P1 = 0x00;
        P2 = 0x00;
        P3 = 0x00;

        InitDAC();                          // Initialize DAC0808 data bus

        while(1)
        {
                Generate_DAC_Voltage(1000);        // Generate 1000mV = 1v at output
                delay_sec(2);                              // Two second delay
                Generate_DAC_Voltage(2000);        // Generate 2000mV = 2v at output
                delay_sec(2);                              // Two second delay
                Generate_DAC_Voltage(3000);        // Generate 3000mV = 3v at output
                delay_sec(2);                              // Two second delay
        }
}


// Function Purpose: Produce approximate delay in Secs.
void delay_sec(unsigned int d)
```

```
{
  unsigned int i;

  for(i=0;i<(d*20);i++)
                        __delay_us(50000);
}



// Function Purpose: Produce approximate delay in given uSecs.
void __delay_us(unsigned int d)
{
  unsigned int i, limit;
  limit = d/15;

  for(i=0;i<limit;i++);
}
```

CIRCUIT DIAGRAM: DAC INTERFACING TO 8051



Inference:

1.
2.
3.
4.

Result

Thus interfacing DAC using 8051 done successfully and the output viewed in CRO.

****

Ex.No : 5

Date : **Timer, Counter and Interrupt program application for 8051 and PIC**

Aim

Write and test programs of timer, counter and interrupt program application for 8051 and PIC.

Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision Software
3	MPLAB Software
4	8051 trainer kit
5	PIC trainer kit
6	Serial Cable

Procedure

For AT89C51,

1. Create a new project in keil software.
2. Select the controller as AT89C51.
3. Open a new script and type the program.
4. Save the program as '.asm' file.
5. Add this file to the source group folder in the target folder.
6. Build a target location for the program by clicking 'build target' option in project tab.
7. Now start executing the program by clicking 'start debug session' option in debug tab.
8. Check for the errors and warning and finally run the program.
9. The output can be viewed from the project status window and hex files also generated. Connect kit and system using serial cable and embed the .hex file in the trainer kit using EZ Downloader then RESET AT89C51.

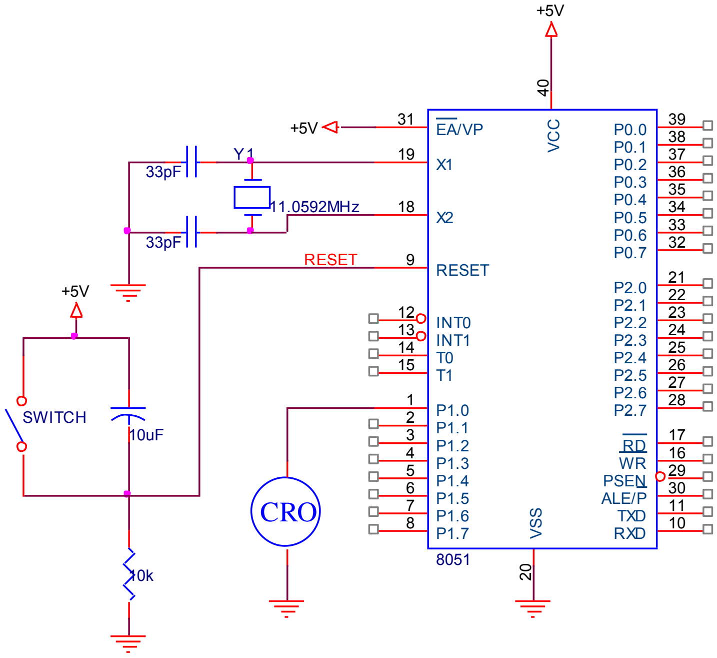
For PICF458,

1. Create a New project in MPLAB software.
2. Select the Controller as PIC18F458.
3. Open a new script and type the program.
4. Save the program as '.c' file.
5. Add this file to the Source Group folder in the Target folder.
6. Build a target location for the program by clicking 'Build Target' option in Project tab.
7. Now start executing the program by clicking 'Start Debug Session' option in Debug tab.
8. Check for the errors and warning and finally Run the Program.
9. The output can be viewed from the Project status window.

PROGRAM: GENERATING A SQUIRE WAVE OF 100 MICRO SECOND USING TIMER USING 8051

```
/*pulse of 100 micro second is created on pin p1.0*/

#include<reg51.h>
sbit pin=P1^0;
void timer_delay();
int i;

void main()
  {
          TMOD=0X01; // MODE 1 OF TIMER O IS SELECTED
          while(1)
            {
                    pin=0;
                    timer_delay();
                    pin=1;
                    timer_delay();
        }
  }

void timer_delay()
  {
          // time delay of 100 micro sec using 12MHz crystal oscillator
          TL0=0Xdb;
          TH0=0Xff;
          TR0=1;
          while(!TF0);
          TF0=0;
          TR0=0;
  }
```

CIRCUIT DIAGRAM: GENERATING A SQUIRE WAVE OF 100 MICRO SECOND USING TIMER

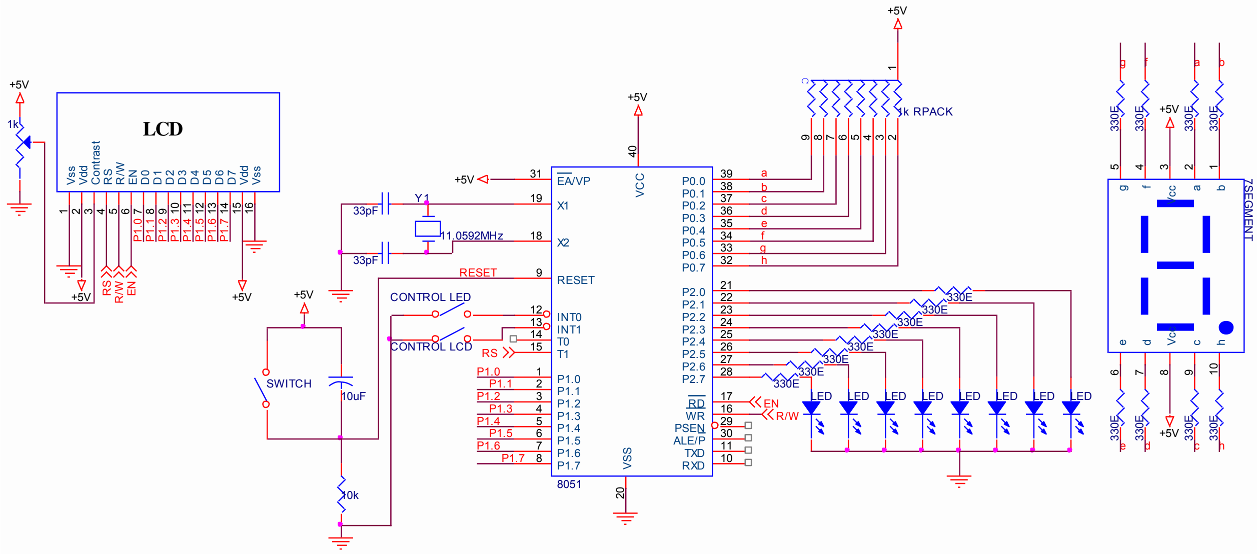


PROGRAM: INTERRUPT 0 LED BLINKING AND INTERRUPT 1 CONTROL LCD DISPLAY AND NORMAL MODE COUNTING OF 7 SEGMENT

```
#include<reg51.h>

sfr seven_seg=0x80; // 7segment at port 0
sfr led=0xa0;       // led at port 2
sfr lcd=0x90;       // data of lcd at port 1
sbit rs=P3^5;       // rs pin at P3.5
sbit rw=P3^6;       // rw pin at P3.6
sbit en=P3^7;       // en pin at P3.7


void delay();  // delay function

void cmd();

void display();

void main()
{
        int data1[]={0xc0,0xf9,0xa4,0xb0,0x99,0x92,0x82,0xf8,0x80,0x90}; // data of common
anode
        int i,j;
             IE=0x85; // extrnal interrupt are enabled
             while(1)
               {
                   if(i==10)
                     i=0;
                   seven_seg=data1[i];
                   delay();
                   delay();
                   delay();
                   i++;
        }
}

void led_control() interrupt 0   // interrupt(INT0) 0
     {
           led=0x00;
                          delay();
                          led=0xff;
                          delay();
     }

void lcd_control() interrupt 4   // interrupt(INT1) 1
    {
                    lcd=0x38;
                          cmd();
                          lcd=0x0e;
                          cmd();
                          lcd=0x01;
                          cmd();
                          lcd=0x06;
                          cmd();
                          lcd=0x80;
                          cmd();
                          lcd='W';
                          display();
                          lcd='E';
                          display();
                          lcd='L';
                          display();
                          lcd='C';
                          display();
                          lcd='O';
                          display();
                          lcd='M';
                          display();
                          lcd='E';
                          display();
                          lcd=' ';
                          display();
                          lcd='T';
                          display();
                          lcd='O';
                          display();
                          lcd=0xc0;
                          cmd();
                          lcd='M';
                          display();
                          lcd='E';
                          display();
                          lcd='C';
                          display();
                          lcd='H';
                          display();
                          lcd='A';
                          display();
                          lcd='T';
                          display();
                          lcd='R';
                          display();
                          lcd='O';
                          display();
                          lcd='N';
                          display();
                          lcd='I';
                          display();
                          lcd='C';
                          display();
                          lcd='S';
                          display();

    }

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

```
void cmd()
  {
     unsigned char i;
     rs=0;
     rw=0;
     en=1;
     for(i=0;i<2;i++);
     en=0;
     delay();
  }

void display()
  {
      unsigned char i;
      rs=1;
      rw=0;
      en=1;
      for(i=0;i<2;i++);
      en=0;
      delay();
   }
```

PROGRAM: INTERRUPT 0 LED BLINKING AND INTERRUPT 1 CONTROL LCD DISPLAY AND NORMAL MODE COUNTING OF 7 SEGMENT USING 8051

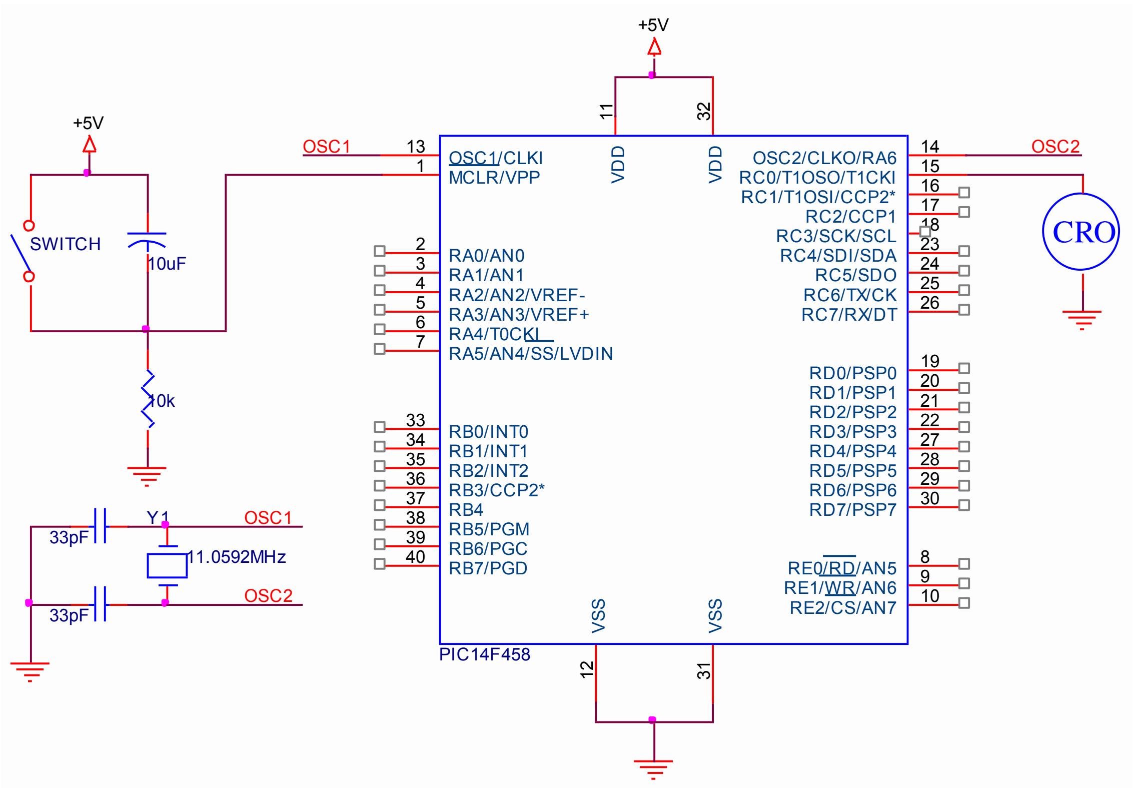


PROGRAM: CREATE A SQUARE WAVE OF 100 MS AT PORTC.0 OF PIC18F458

/* header file used in this program is already included in software MPLAB for pic*/

```
void timer_0();   // timer delay
#define PULSE portc.RC0
void main()
  {
     TRISC=0X00;    //PORTD AS OUTPUT MODE
     T0CON=0X08;    //TIMER0, 16 BIT MODE, NO PRESCALER
     PORTD=0X00;
     while(1)
      {
         PULSE=0;
         timer_0();       // CALLING FUNCTION
         PULSE=1;
         timer_0();
      }
   }

void timer_0()       //creating a delay of 1 sec
 {

          TMR0H=0XFF;               // LOAD TH0
          TMR0L=0X00;               //LOAD TL0
          T0CON|=(1<<7);            // TURN ON T0
          while((INTCON & 0X04)==0); //WAIT FOR OVERFLOW
          T0CON&=~(1<<7);           // TURN OFF T0
          INTCON&=~(1<<2);          //CLEAR TF0

 }
```

CIRCUIT DIAGRAM : CREATE A SQUARE WAVE OF 100 MS AT PORTC.0 OF PIC18F458



<u>**Inference:**</u>

1.
2.
3.
4.

Result

Thus timer, counter and interrupt program application for 8051 and PIC18F458 are done successfully.

****

Ex.No : 6
Date :

Step motor (unipolar & bipolar motor) and PWM servo motor control to interfacing with 8051

Aim

To interface of step motor (unipolar and bipolar) and control of PWM servo motor with 8051

Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision Software
3	8051 trainer kit
4	Stepper motor – Bipolar(5V)
5	Stepper motor – Unipolar(5V)
6	DC Servo motor (5V)
7	Serial cable

Procedure

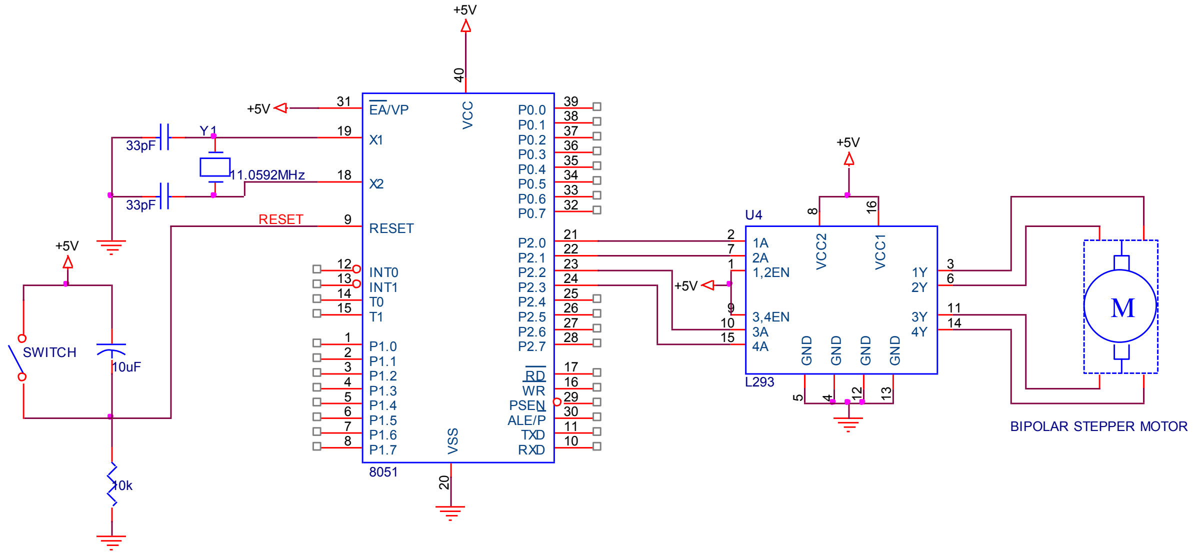
For AT89C51,

1. Create a new project in keil software.
2. Select the controller as AT89C51.
3. Open a new script and type the program.
4. Save the program as '.asm' file.
5. Add this file to the source group folder in the target folder.
6. Build a target location for the program by clicking 'build target' option in project tab.
7. Now start executing the program by clicking 'start debug session' option in debug tab.
8. Check for the errors and warning and finally run the program.
9. The output can be viewed from the project status window and hex files also generated. Connect kit and system using serial cable and embed the .hex file in the trainer kit using EZ Downloader then RESET AT89C51.

PROGRAM: STEPPER MOTOR INTERFACE USING 8051(BIPOLAR)

```
#include<reg51.h>
#include<stdio.h>

void delay(int);

void main()
{
 do
 {
   P2=0x01; //0001
   delay(1000);
   P2=0x04; //0100
   delay(1000);
   P2=0x02; //0010
   delay(1000);
   P2=0x08; //1000
   delay(1000);
 }while(1);
}

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

CIRCUIT DIAGRAM: STEPPER MOTOR INTERFACE USING 8051(BIPOLAR)



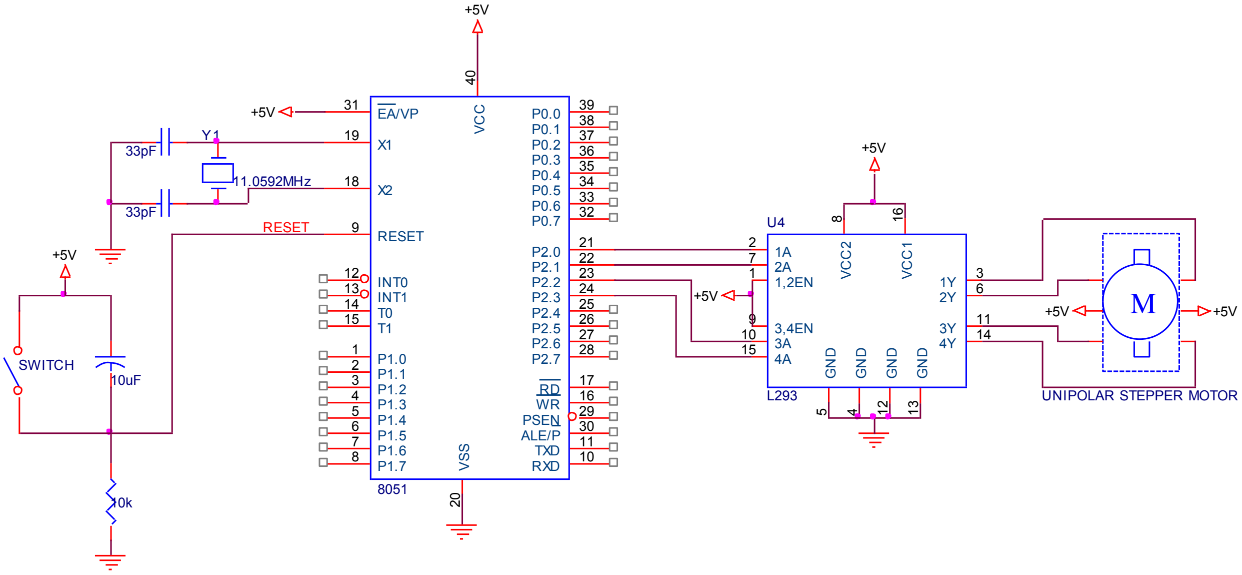
PROGRAM: STEPPER MOTOR INTERFACE USING 8051(UNIPOLAR-FULL DRIVE)

```
#include<reg51.h>
#include<stdio.h>

void delay(int);

void main()
{
do
{
            P2 = 0x03; //0011
            delay(1000);
            P2 = 0x06; //0110
            delay(1000);
            P2 = 0x0C; //1100
            delay(1000);
            P2 = 0x09; //1001
            delay(1000);
 }
while(1);
}

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

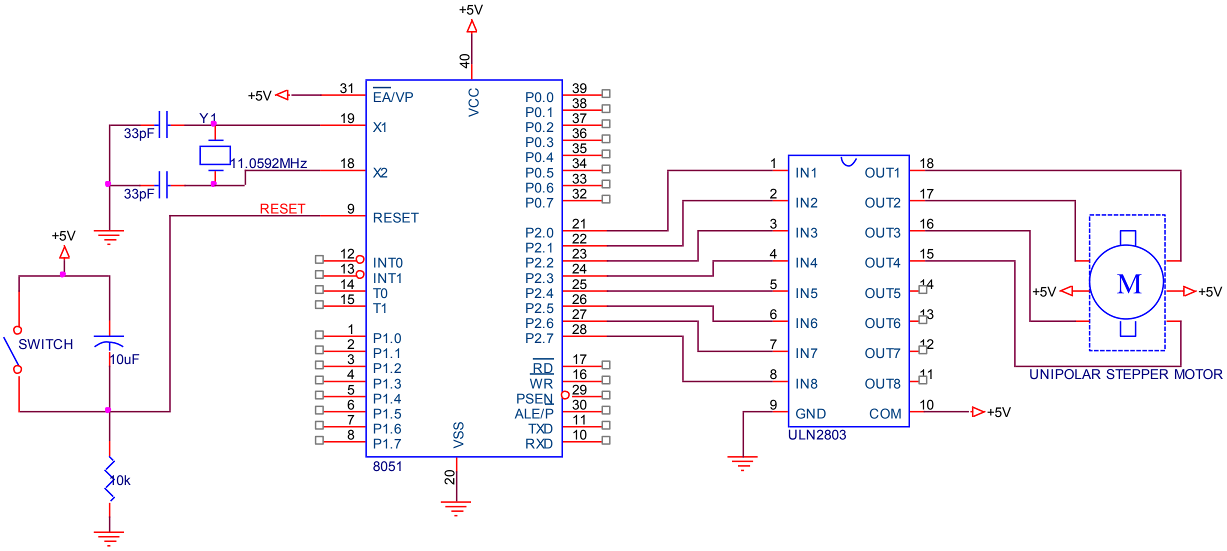
PROGRAM: STEPPER MOTOR INTERFACE USING 8051(UNIPOLAR-HALF DRIVE)

```
#include<reg51.h>
#include<stdio.h>

void delay(int);

void main()
{
 do
 {
   P2=0x01; //0001
   delay(1000);
   P2=0x03; //0011
   delay(1000);
   P2=0x02; //0010
   delay(1000);
   P2=0x06; //0110
   delay(1000);
   P2=0x04; //0100
   delay(1000);
   P2=0x0C; //1100
   delay(1000);
   P2=0x08; //1000
   delay(1000);
   P2=0x09; //1001
   delay(1000);
 } while(1);
}

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

CIRCUIT DIAGRAM: STEPPER MOTOR INTERFACE USING 8051(UNIPOLAR-HALF DRIVE AND FULL DRIVE) USING L293D



CIRCUIT DIAGRAM: STEPPER MOTOR INTERFACE USING 8051(UNIPOLAR-HALF DRIVE AND FULL DRIVE) USING ULN2803

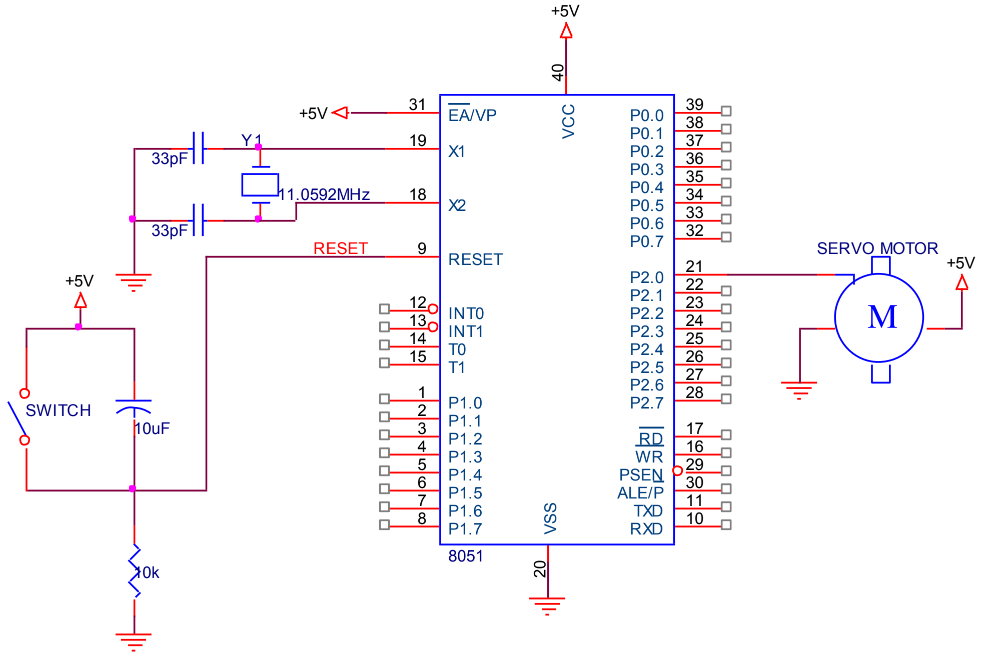


PROGRAM: SERVO MOTOR INTERFACE USING 8051

```
#include<reg51.h>
#include<stdio.h>
#include <intrins.h>

sbit motor_pin = P2^0;
void Delay(unsigned int);
void Delay_servo(unsigned int);
void main()
{
  motor_pin = 0;
 do
 {
   //Turn to 0 degree
             motor_pin = 1;
   Delay_servo(50);
   motor_pin = 0;
   Delay(1000);
             //Turn to 90 degree
   motor_pin=1;
             Delay_servo(82);
   motor_pin=0;
   Delay(1000);
             //Turn to 180 degree
   motor_pin=1;
   Delay_servo(110);
   motor_pin=0;
   Delay(1000);
 }while(1);
}
```

```
void Delay(unsigned int ms)
{
  unsigned long int us = ms*1000;
             while(us--)
             {
    _nop_();
  }
}
void Delay_servo(unsigned int us)
{
             while(us--)
             {
    _nop_();
  }
}
```

CIRCUIT DIAGRAM: SERVO MOTOR INTERFACE USING 8051



Inference:

1.
2.
3.
4.

Result

Thus interfacing of step motor (unipolar and bipolar) and control of PWM servo motor with 8051 is done successfully.

****

Ex.No : 7
Date :

UART serial programming in 8051 and PIC

Aim

Write program to test serial communication between PC and 8051, PIC

Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision Software
3	MPLAB Software
4	8051 trainer kit
5	PIC trainer kit
6	Serial cable

Procedure

For AT89C51,

1. Create a new project in keil software.
2. Select the controller as AT89C51.
3. Open a new script and type the program.
4. Save the program as '.asm' file.
5. Add this file to the source group folder in the target folder.
6. Build a target location for the program by clicking 'build target' option in project tab.
7. Now start executing the program by clicking 'start debug session' option in debug tab.
8. Check for the errors and warning and finally run the program.
9. The output can be viewed from the project status window and hex files also generated. Connect kit and system using serial cable and embed the .hex file in the trainer kit using EZ Downloader then RESET AT89C51.

For PICF458,

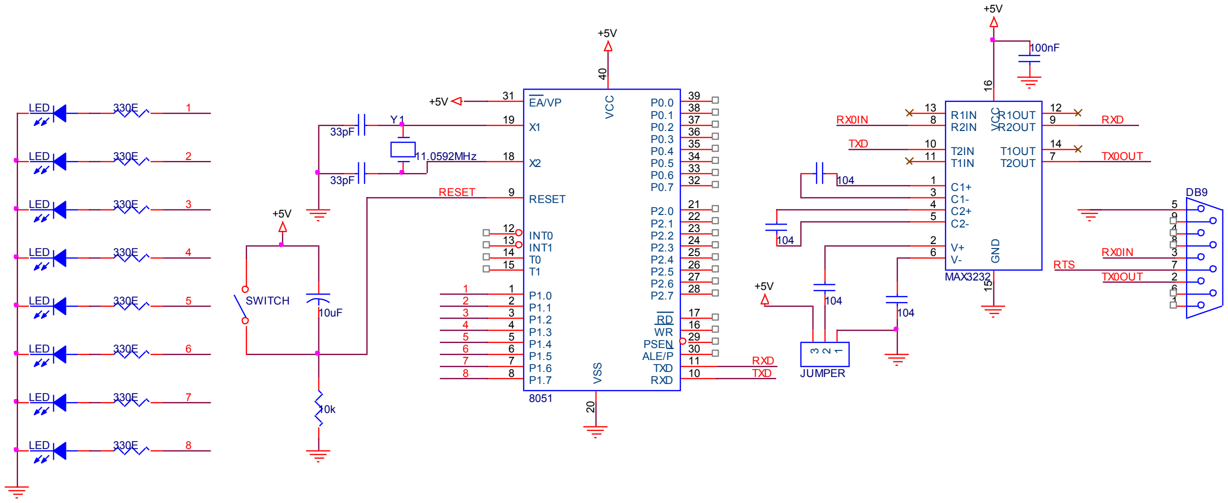
1. Create a New project in MPLAB software.
2. Select the Controller as PIC18F458
3. Open a new script and type the program.
4. Save the program as '.c' file.
5. Add this file to the Source Group folder in the Target folder.
6. Build a target location for the program by clicking 'Build Target' option in Project tab.
7. Now start executing the program by clicking 'Start Debug Session' option in Debug tab.
8. Check for the errors and warning and finally Run the Program.
9. The output can be viewed from the project status window.

PROGRAM: Serial Communication with 8051 Microcontroller

```
#include <reg51.h>

//DEFINE CONSTANT
#define Baud_rate 0xFD  // BAUD RATE 9600


//DEFINE PROTOTYPES
void SerialInitialize(void);
void SendByteSerially(unsigned char);
void cct_init(void);

sbit Appliance1 = P1^0;
sbit Appliance2 = P1^1;
sbit Appliance3 = P1^2;
sbit Appliance4 = P1^3;
sbit Appliance5 = P1^4;
sbit Appliance6 = P1^5;
sbit Appliance7 = P1^6;
sbit Appliance8 = P1^7;


void main()
{
        cct_init();
        SerialInitialize();

   EA = 1;
        ES = 1;

        while(1) {;}
}


void cct_init(void)   //initialize cct
{
        P0 = 0x00; //not used
        P1 = 0x00; //Used for Appliances
        P2 = 0x00; //not used
        P3 = 0x03; //used for serial

}

void SerialInitialize(void)                        // INITIALIZE SERIAL PORT
{
        TMOD = 0x20;                // Timer 1 IN MODE 2 -AUTO RELOAD TO
GENERATE BAUD RATE
        SCON = 0x50;                             // SERIAL MODE 1, 8-DATA BIT 1-
START BIT, 1-STOP BIT, REN ENABLED
        TH1 = Baud_rate;                         // LOAD BAUDRATE TO TIMER
REGISTER
        TR1 = 1;                                   // START TIMER
}
```

```
void SendByteSerially(unsigned char serialdata)
{
            SBUF = serialdata;                                    // LOAD DATA TO SERIAL
BUFFER REGISTER
            while(TI == 0);                                  // WAIT UNTIL TRANSMISSION TO
COMPLETE
            TI = 0;                                               // CLEAR TRANSMISSION
INTERRUPT FLAG
}

void serial_ISR (void) interrupt 4
{
    //receive character
            char chr;
            if(RI==1)
            {
                        chr = SBUF;
                        RI = 0;
            }

    P0 = ~P0;    //Show the data has been updated

    switch(chr)
            {
     case '1':  Appliance1 = 1; SendByteSerially('k');  break;
     case '2':  Appliance2 = 1; SendByteSerially('k');  break;
             case '3':  Appliance3 = 1; SendByteSerially('k');  break;
             case '4':  Appliance4 = 1; SendByteSerially('k');  break;
             case '5':  Appliance5 = 1; SendByteSerially('k');  break;
             case '6':  Appliance6 = 1; SendByteSerially('k');  break;
             case '7':  Appliance7 = 1; SendByteSerially('k');  break;
             case '8':  Appliance8 = 1; SendByteSerially('k');  break;


             case 'a':  Appliance1 = 0; SendByteSerially('k');  break;
             case 'b':  Appliance2 = 0; SendByteSerially('k');  break;
             case 'c':  Appliance3 = 0; SendByteSerially('k');  break;
             case 'd':  Appliance4 = 0; SendByteSerially('k');  break;
             case 'e':  Appliance5 = 0; SendByteSerially('k');  break;
             case 'f':  Appliance6 = 0; SendByteSerially('k');  break;
             case 'g':  Appliance7 = 0; SendByteSerially('k');  break;
             case 'h':  Appliance8 = 0; SendByteSerially('k');  break;


            default: ;
            break;     //do nothing
            }

            RI = 0;
}
```

CIRCUIT DIAGRAM: SERIAL COMMUNICATION WITH 8051 MICROCONTROLLER



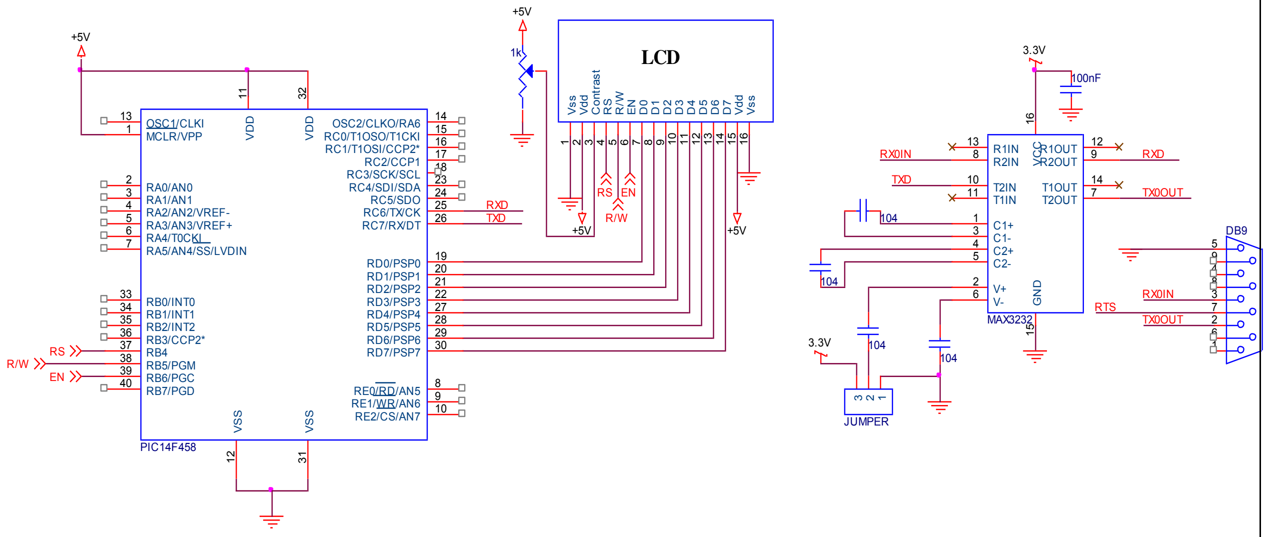
PROGRAM: SERIAL COMMUNICATION OF PIC18F458 WITH LCD

/* header file used in this program is already included in software MPLAB for pic*/

```
#define rs portb.RB4
#define rw portb.RB5
#define en portb.RB6
#define lcd portd

int rec;

void lcd_ini();
void lcd_display(unsigned int);
void cmd(unsigned char);
void lcd_str(unsigned char*);
void serial_ini()
 char serial_re();
void serial_tr(unsigned char x);

void main()
   {
      int i=0;
      TRISC.RC7=1;
      TRISC.RC6=0;
      TRISD=0X00;
      TRISB=0X00;
      serial_ini();
      lcd_ini();
      while(1)
        {
          i++;
```

```
            serial_re();
            lcd_display(rec);
            if(i==16)
                cmd(0xc0);
            if(i==32)
              {
                  cmd(0x01);
                  i=0;
                }
           }
    }


void lcd_display(unsigned int x)
  {
    lcd=x;
    rs=1;
    rw=0;
    en=1;
    delay_ms(100);
    en=0;
  }

void cmd(unsigned char m)
   {
     lcd=m;
     rs=0;
     rw=0;
     en=1;
     delay_ms(10);
     en=0;
  }

void lcd_ini()
   {
      cmd(0x38);
      cmd(0x0e);
      cmd(0x01);
      cmd(0x06);
      cmd(0x80);
   }

void lcd_str(unsigned char *str)
   {
      while(*str!='\0')
        {
            lcd_display(*str);
            str++;
        }
    }

void serial_ini()
   {
       RCSTA=0X90;
       SPBRG=15;
       TXSTA.TXEN=1;
    RCSTA.SPEN=1;
  }

char serial_re()
  {
    while(PIR1.RCIF==0);
    rec=RCREG;
    return rec;
  }

void serial_tr(unsigned char x)
  {
    TXREG=x;
    while(PIR1.TXIF==0);
  }
```

PROGRAM: SERIAL COMMUNICATION OF PIC18F458 WITH LCD



Inference:

1.
2.
3.
4.

Result

Thus, serial communication between PC and 8051, PIC18F458 is tested successfully.

****

Ex.No : 8
Date :

PC Interfacing of stepper motor

Aim

To actuate the unipolar stepper motor stepper motor using PC parallel port through MATLAB Codes.

Apparatus Required

Sl. No	Apparatus Name
1	Parallel port cable
2	IC-ULN2003 or ULN2803
3	Stepper motor
4	PC – Matlab

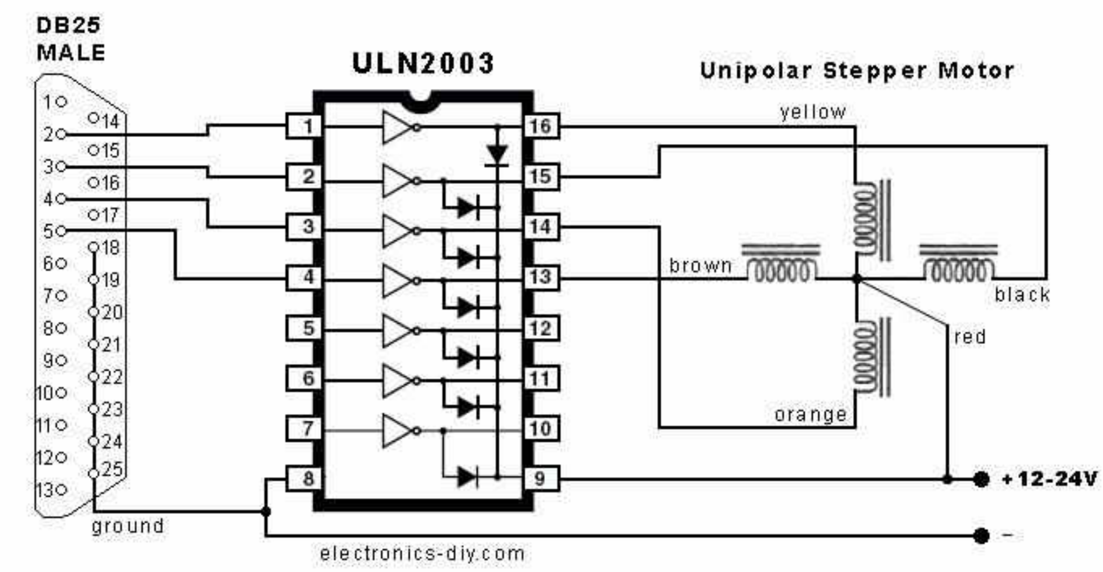
Procedure

1. Write the following code
2. “parport=digitalio(‘parallel’,LPT1’);
3. line=addline(parport,0:3,’out); in matlab to give input to parallel port.
4. Give the required connections from parallel port to IC chips.
5. The signal is given to stepper motors coil form IC
6. The power source of 5v is given to the IC, to switch the current among the 5 coils respectively.
7. Thus the stepper motor is actuated.

PROGRAM

```
clc;
clear all;
close all;
par=digital(‘parallel,’lpt1’);
line=addline(par,2:5,’out’);
for i=0:25
put value(par,1);
pause(0.5);
put value(par,2);
pause(0.5);
put value(par,4);
pause(0.5);
put value (par,8);
pause(0.5);
end
```

CIRCUIT DIAGRAM



Inference:

1.
2.
3.
4.

Result

Thus the stepper motor is successfully actuated using MATLAB from PC.

****

Ex.No : 9
Date :

Programming of ARM Processor for sensor interface

Aim

To interface a sensor with ARM processor

Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision Software
3	ARM trainer kit
4	LM35
5	Serial cable

Procedure

1. Create a new project in keil software.
2. Select the controller as ARM.
3. Open a new script and type the program.
4. Save the program as '.c' file.
5. Add this file to the source group folder in the target folder.
6. Build a target location for the program by clicking 'build target' option in project tab.
7. Now start executing the program by clicking 'start debug session' option in debug tab.
8. Check for the errors and warning and finally run the program
9. The output can be viewed from the project status window and the hex files also generated.
10. Embed the .hex file in trainer kit and RESET the controller.

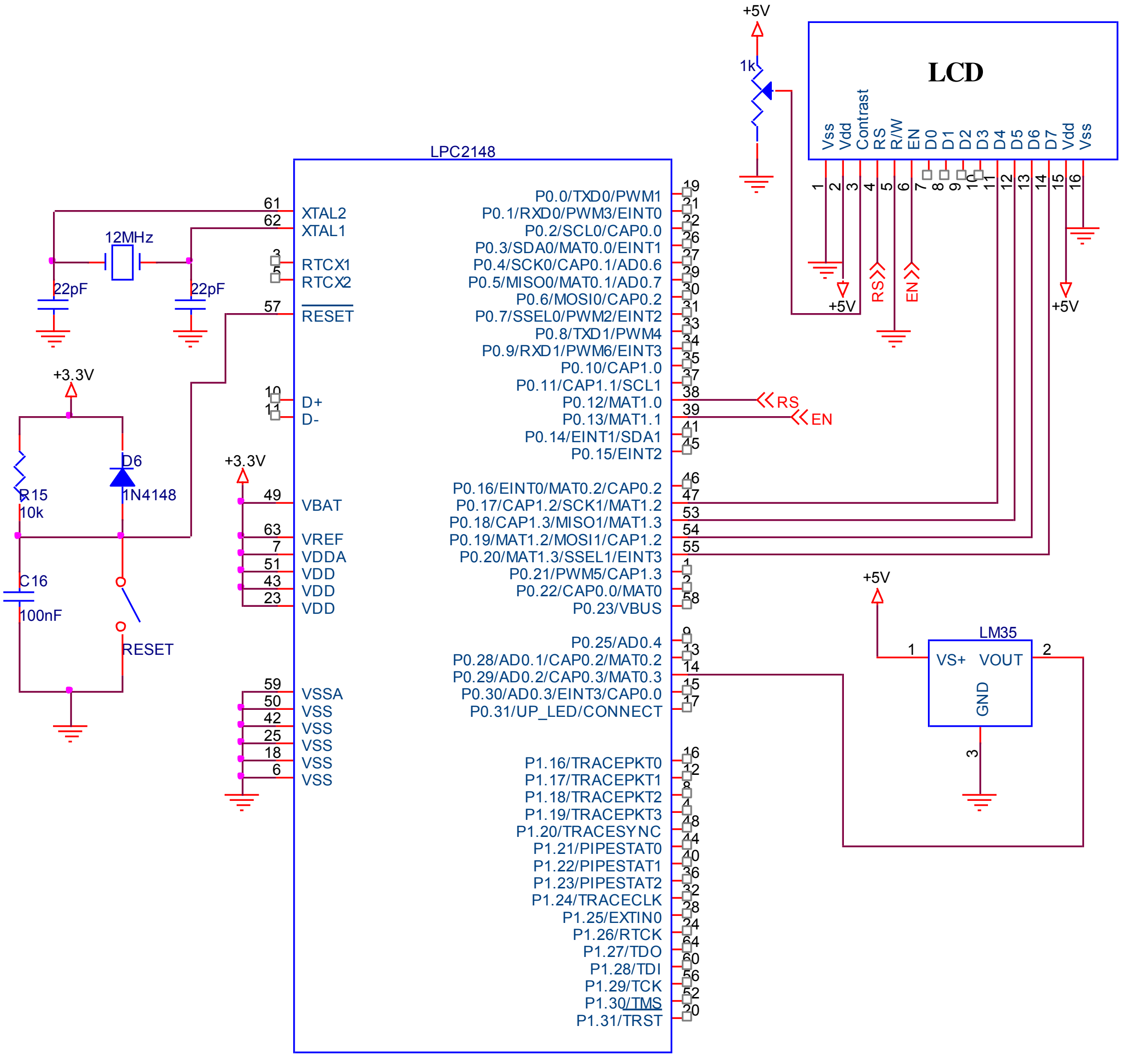
PROGRAM: TEMPERATURE SENSOR (LM35) USING ADC AND LPC2148 WITH ARM (LPC2148) CONTROLLER

```
#include<lpc2148.h>

#define LCD (0xff<<16)
#define RS (1<<13)
#define RW (1<<14)
#define EN (1<<15)

void delay_fv(unsigned int x,int y);
void lcd_display(unsigned int x);
void cmd(unsigned char m);
void lcd_ini();
void lcd_pos(int line, int pos);
void lcd_str(unsigned char *x);
void pll();
void adc_ini();
unsigned long int adc_data();

int main()
  {
                    unsigned long temp;
                    unsigned char first,second,third,fourth,fifth;
                    PINSEL0=0X00000000;
                    IO0DIR=0XFFFFFFFF;
                    pll();
                    adc_ini();
                    lcd_ini();
                    lcd_str("TEMP VALUE IS");
                    lcd_pos(2,6);
                    lcd_display('C');
                    while(1)
             {
                                              temp=adc_data();
                                              temp=temp*3300;
                                              temp=temp/1023;
                                              first=temp%10+'0';
                                              temp=temp/10;
                                              second=temp%10+'0';
                                              temp=temp/10;
                                              third=temp%10+'0';
                                              temp=temp/10;
                                              fourth=temp%10+'0';
                                              temp=temp/10;
                                              lcd_pos(2,0);
                                              lcd_display(fourth);
                                              lcd_display(third);
                                              lcd_display(second);
                                              lcd_display('.');
                                              lcd_display(first);

             }
  }

void delay_fv(unsigned int x,int y)
```

```
 {
                                        unsigned int i,j;
                                        for(i=0;i<x;i++)
                                        for(j=0;j<y;j++);
 }
void lcd_display(unsigned int x)
  {
                                        IO0CLR|=(RS|RW|EN|LCD);
                                        IO0SET|=(x<<16);
                                        IO0SET|=RS;
                                        IO0CLR|=RW;
                                        IO0SET|=EN;
                                        delay_fv(100,200);
                                        IO0CLR|=EN;
                                        delay_fv(10,10);
  }

void cmd(unsigned char m)
  {
                                        IO0CLR|=(RS|RW|EN|LCD);
                                        IO0SET|=(m<<16);
                                        IO0CLR|=RS;
                                        IO0CLR|=RW;
                                        IO0SET|=EN;
                                        delay_fv(100,10);
                                        IO0CLR|=EN;
                                        delay_fv(100,10);
  }

void lcd_ini()
 {
                                       cmd(0X38);
                                       cmd(0X0e);
                                       cmd(0X06);
                                       cmd(0X01);
                                       cmd(0X80);
 }

void lcd_pos(int line, int pos)
 {
                 if(line==1)
                               cmd(0x80+pos);
                 else if(line==2)
                               cmd(0xc0+pos);
 }

void lcd_str(unsigned char *x)
 {
                 while(*x!='\0')
                   {
                                  lcd_display(*x);
                                  x++;
                   }
 }
void pll()
 {
```

```
                /*PLL IS CONFIGURED TO GET 60HZ pCLK*/
                        PLLCFG=0X24;                    // SET PSEL=2 AND MSEL=5
                        PLLCON=0X01;                    //PLL IS ACTIVE BUT NOT YET
CONNECT
                        PLLFEED=0XAA;                   //FEED SEQUENCE
                        PLLFEED=0X55;                   //FEED SEQUENCE
                        while((PLLSTAT & 0X400)==0);   //WAIT FOR FEED SEQUENCE
TO BE INSERTED
                        PLLCON=0X03;                    // PLL HAS BEEN ACTIVE AND
BEING CONNECTRD
                        VPBDIV=0X00;                    // SET PCLK 1/4th of FCCLK
                        PLLFEED=0XAA;                   //FEED SEQUENCE
                        PLLFEED=0X55;                   //FEED SEQUENCE
 }

void adc_ini()
 {
                AD0CR = 1<<21;          //A/D is Operational
                AD0CR = 0<<21;          //A/D is in Power Down Mode
                PINSEL1 = 0x01000000;//P0.28 is Configured as Analog to Digital Converter
Pin AD0.1
                AD0CR = 0x00200802;  //CLKDIV=4,Channel-0.1
Selected,BURST=0,EDGE=0
                /*PDN=0
                A/D Clock = PCLK /(CLKDIV+1);*/

 }

unsigned long int adc_data()
  {
            unsigned long rec;
                                AD0CR |= (1<<24);
                //Start Conversion
                                while(!(AD0GDR & 0x80000000));
                                /*Wait untill the DONE bits Sets*/
                                rec = AD0GDR;
                                AD0CR &= ~0x01000000;      //Stops the A/D Conversion

                                rec = rec >> 6;  // data is present after 6 bit
                                rec = rec & 0x3FF;    //Clearing all other Bits
                         return (rec);
  }
```

CIRCUIT DIAGRAM: TEMPERATURE SENSOR (LM35) USING ADC AND LPC2148 WITH ARM (LPC2148) CONTROLLER



Inference:

1.
2.
3.
4.

Result

Thus LM35 is interfaced with ARM processor successfully.

****

Ex.No : 10

Date :

Programming of ARM Processor for display interface

Aim

To interface LCD (16X2) with ARM processor.

Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision Software
3	ARM trainer kit
4	Serial cable

Procedure

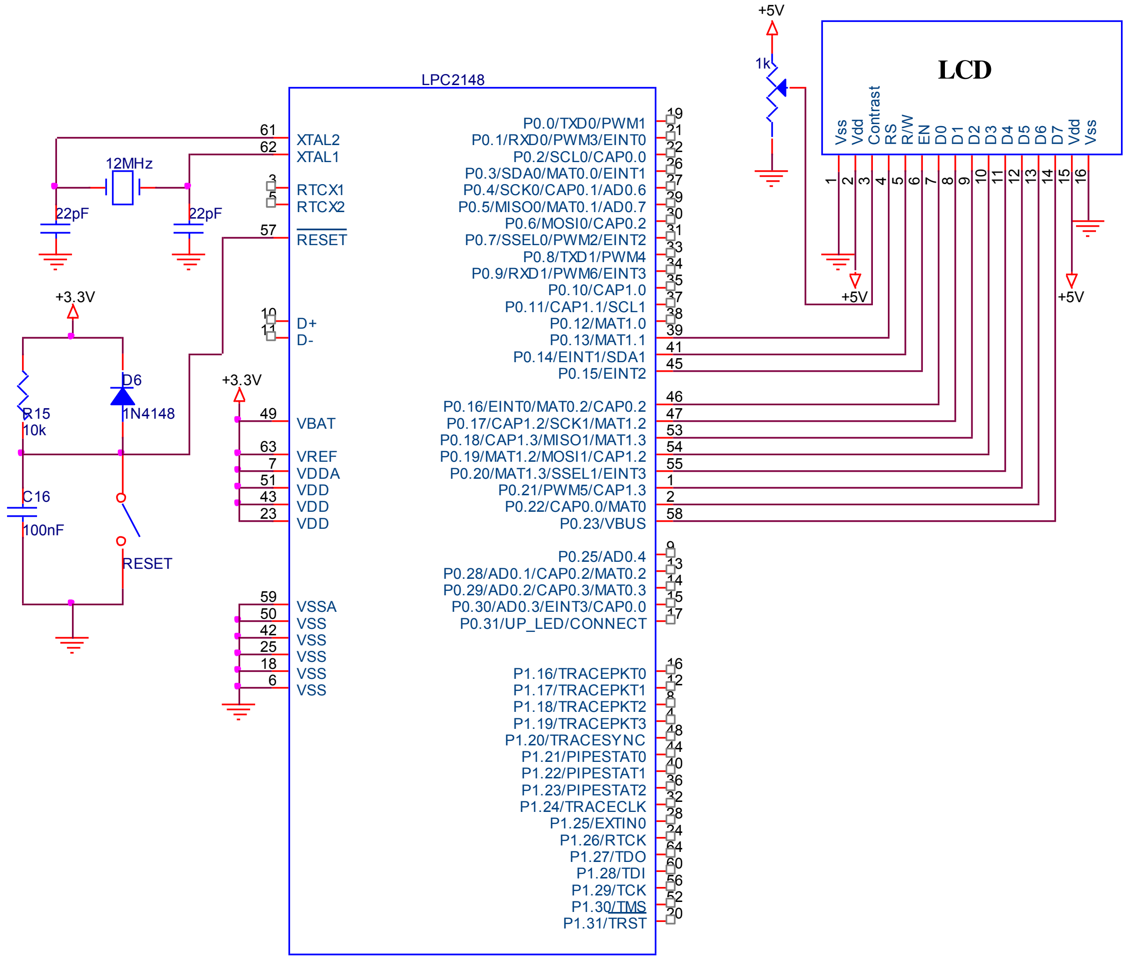
1. Create a new project in keil software.
2. Select the controller as ARM.
3. Open a new script and type the program.
4. Save the program as '.c' file.
5. Add this file to the source group folder in the target folder.
6. Build a target location for the program by clicking 'build target' option in project tab.
7. Now start executing the program by clicking 'start debug session' option in debug tab.
8. Check for the errors and warning and finally run the program
9. The output can be viewed from the project status window and the hex files also generated. Embed the .hex file in trainer kit and RESET the controller.

PROGRAM: LCD DISPLAY USING ARM7 (LPC2148)

```
#include<lpc2148.h>

#define LCD (0xff<<16)
#define RS (1<<13)
#define RW (1<<14)
#define EN (1<<15)

void delay_fv(unsigned int x,int y);
void lcd_display(unsigned int x);
void cmd(unsigned char m);
void lcd_ini();

int main()
  {
     PINSEL0=0X00000000;
     IO0DIR=0XFFFFFFFF;
     lcd_ini();
     while(1)
       {
          lcd_ini();
          lcd_display(' ');
          lcd_display('W');
          delay_fv(1000,400);
          lcd_display('E');
          delay_fv(1000,400);
          lcd_display('L');
          delay_fv(1000,400);
          lcd_display('C');
          delay_fv(1000,400);
          lcd_display('O');
          delay_fv(1000,400);
          lcd_display('M');
          delay_fv(1000,400);
          lcd_display('E');
          delay_fv(1000,400);
          lcd_display(' ');
          delay_fv(1000,400);
          lcd_display('T');
          delay_fv(1000,400);
          lcd_display('O');
          delay_fv(1000,400);
          cmd(0x0c0);
          lcd_display('M');
          delay_fv(1000,400);
          lcd_display('E');
          delay_fv(1000,400);
          lcd_display('C');
          delay_fv(1000,400);
          lcd_display('H');
          delay_fv(1000,400);
          lcd_display('A');
          delay_fv(1000,400);
```

```
                lcd_display('T');
                delay_fv(1000,400);
                lcd_display('R');
                delay_fv(1000,400);
                lcd_display('O');
                delay_fv(1000,400);
                lcd_display('N');
                delay_fv(1000,400);
                lcd_display('I');
                delay_fv(1000,400);
                lcd_display('C');
                delay_fv(1000,400);
                lcd_display('S');
                delay_fv(1000,400);
           }
    }

void delay_fv(unsigned int x,int y)
  {
      unsigned int i,j;
      for(i=0;i<x;i++)
      for(j=0;j<y;j++);
  }

void lcd_display(unsigned int x)
  {
     IO0CLR|=(RS|RW|EN|LCD);
     IO0SET|=(x<<16);
     IO0SET|=RS;
     IO0CLR|=RW;
     IO0SET|=EN;
     delay_fv(100,10);
     IO0CLR|=EN;
     delay_fv(10,10);
  }

void cmd(unsigned char m)
  {
     IO0CLR|=(RS|RW|EN|LCD);
     IO0SET|=(m<<16);
     IO0CLR|=RS;
     IO0CLR|=RW;
     IO0SET|=EN;
     delay_fv(100,100);
     IO0CLR|=EN;
     delay_fv(100,10);
  }
 void lcd_ini()
   {
      cmd(0X38);
      cmd(0X0e);
      cmd(0X06);
      cmd(0X01);
      cmd(0X80);
  }
```

CIRCUIT DIAGRAM: LCD DISPLAY USING ARM7 (LPC2148)



Inference:

1.
2.
3.
4.

Result

Thus LCD (16X2) is interfaced with ARM processor successfully.

****

Ex.No : 11
Date :

Stepper motor and Servo motor control using ARM processor

Aim

To control stepper motor and servo motor using ARM processor

Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision Software
3	ARM trainer kit
4	Serial cable
5	Stepper motor (5V)
6	Gear motor (5V)

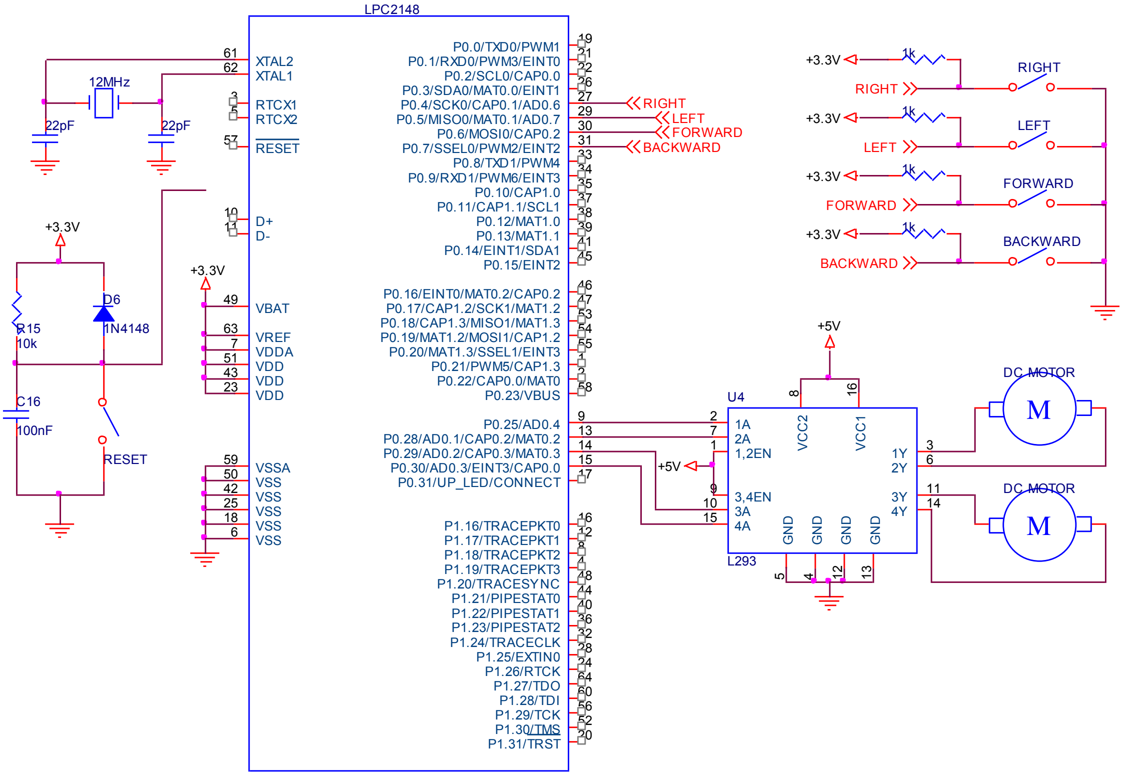
Procedure

1. Create a new project in keil software.
2. Select the controller as ARM.
3. Open a new script and type the program.
4. Save the program as '.c' file.
5. Add this file to the source group folder in the target folder.
6. Build a target location for the program by clicking 'build target' option in project tab.
7. Now start executing the program by clicking 'start debug session' option in debug tab.
8. Check for the errors and warning and finally run the program.
9. The output can be viewed from the project status window and the hex files also generated. Embed the .hex file in trainer kit and RESET the controller.

PROGRAM: PROGRAM CONTROL GEAR MOTOR USING ARM7 (LPC2148)

```
#include<lpc2148.h>

#define left (IO0PIN&(1<<4))
#define right (IO0PIN&(1<<5))
#define forward (IO0PIN&(1<<6))
#define backward (IO0PIN&(1<<7))

int main()
  {
      PINSEL0=0X00000000;
      PINSEL1=0X00000000;
      IO0DIR=0XFFFFFF0F;
      IO0CLR|=(1<<25)|(1<<26)|(1<<27)|(1<<28);
      while(1)
        {
            if(left==0)
              {
                  IO0CLR|=(1<<25)|(1<<26)|(1<<27)|(1<<28);
                  IO0SET|=(1<<25);

               }
            else if(right==0)
              {
                  IO0CLR|=(1<<25)|(1<<26)|(1<<27)|(1<<28);
                  IO0SET|=(1<<27);
              }
            else if(forward==0)
              {
                  IO0CLR|=(1<<25)|(1<<26)|(1<<27)|(1<<28);
                  IO0SET|=(1<<25)|(1<<27);
              }
            else if(backward==0)
              {
                  IO0CLR|=(1<<25)|(1<<26)|(1<<27)|(1<<28);
                  IO0SET|=(1<<26)|(1<<28);
              }
        }
  }
```

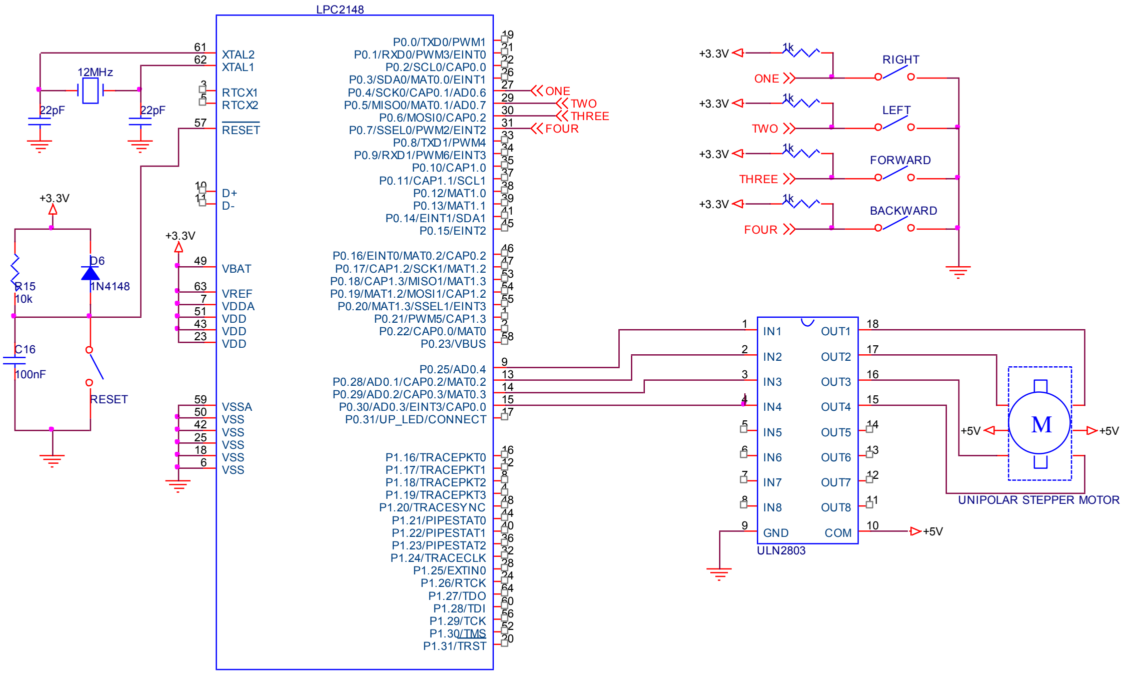
CIRCUIT DIAGRAM: PROGRAM CONTROL GEAR MOTOR USING ARM7 (LPC2148)



PROGRAM: PROGRAM TO CONTROL STEPPER MOTOR USING SWITCH CONNECTED WITH ARM7 (LPC2148)

```
#include<lpc2148.h>

#define one (IO0PIN&(1<<4))
#define two (IO0PIN&(1<<5))
#define three (IO0PIN&(1<<6))
#define four (IO0PIN&(1<<7))

int main()
  {
     PINSEL0=0X00000000;
     PINSEL1=0X00000000;
     IO0DIR=0XFFFFFF0F;
     IO0CLR|=(1<<25)|(1<<26)|(1<<27)|(1<<28);
     while(1)
      {
         if(one==0)
          {
             IO0CLR|=(1<<25)|(1<<26)|(1<<27)|(1<<28);
             IO0SET|=(1<<25);

          }
         else if(two==0)
          {
             IO0CLR|=(1<<25)|(1<<26)|(1<<27)|(1<<28);
             IO0SET|=(1<<26);
          }
         else if(three==0)
          {
             IO0CLR|=(1<<25)|(1<<26)|(1<<27)|(1<<28);
             IO0SET|=(1<<27);
          }
         else if(four==0)
          {
             IO0CLR|=(1<<25)|(1<<26)|(1<<27)|(1<<28);
             IO0SET|=(1<<28);
          }
      }
 }
```

CIRCUIT DIAGRAM: PROGRAM TO CONTROL STEPPER MOTOR USING SWITCH CONNECTED WITH ARM7 (LPC2148)



Inference:

1.
2.
3.
4.

Result

Thus stepper motor and servo motor are interfaced with ARM processor successfully.

****

Ex.No : 12
Date :

Serial communication of ARM processor with computation platform

Aim

Write program to test serial communication between PC and ARM processor.

Apparatus Required

Sl. No	Apparatus Name
1	Personal computer
2	Keil µVision Software
3	ARM trainer kit
4	Serial cable

Procedure

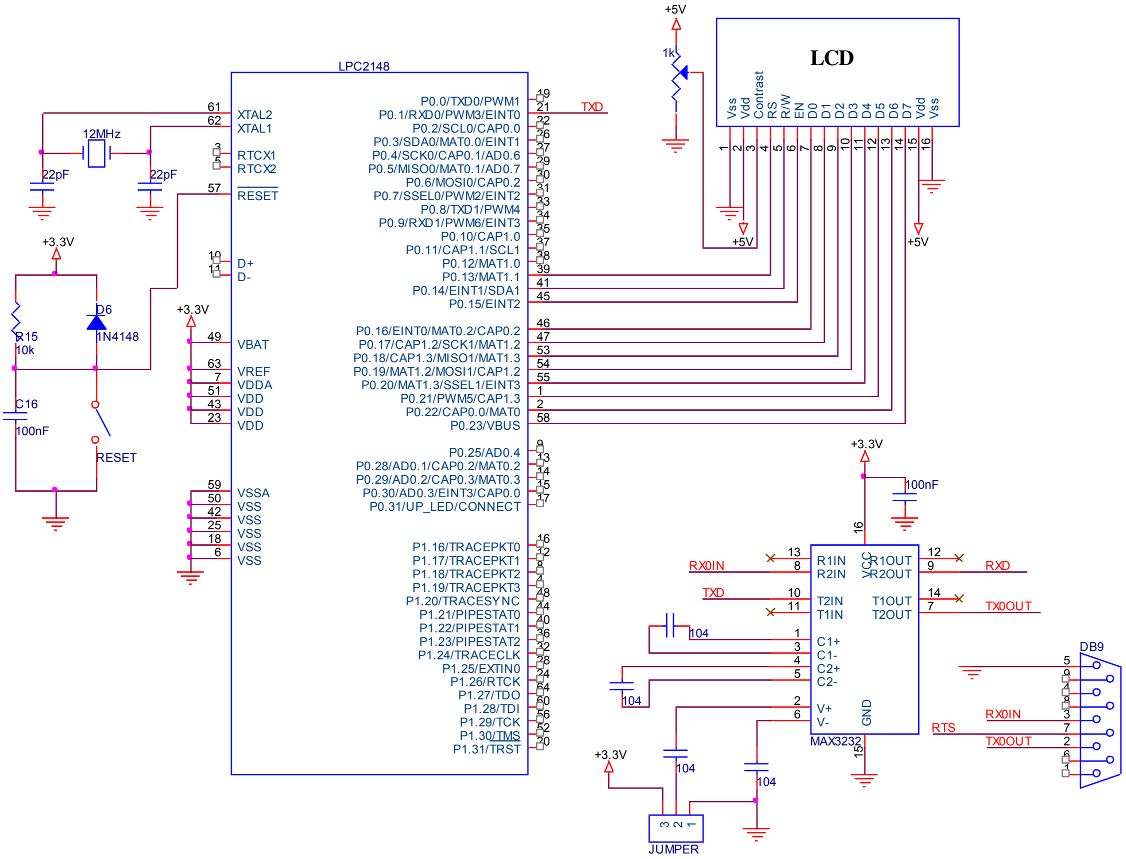
1. Create a new project in keil software.
2. Select the controller as ARM.
3. Open a new script and type the program.
4. Save the program as '.c' file.
5. Add this file to the source group folder in the target folder.
6. Build a target location for the program by clicking 'build target' option in project tab.
7. Now start executing the program by clicking 'start debug session' option in debug tab.
8. Check for the errors and warning and finally run the program.
9. The output can be viewed from the project status window and the hex files also generated. Embed the .hex file in trainer kit and RESET the controller.

PROGRAM: DATA SEND FROM ARM7 (LPC2148) THROUGH RS232 PROTOCOL AND DISPLAY DATA ON PC

```
#include<lpc2148.h>

unsigned char rec;

void pll();

void serial_ini();
void serial_transmit(unsigned char x);

 void delay(int x);

int main()
  {
        PINSEL0 = 0x00000005;
        IO0DIR=0XFFFFFFFF;
        serial_ini();
        pll();
        while(1)
          {
                    serial_transmit('W');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('W');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('W');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('.');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('F');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('I');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('R');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('M');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('C');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('O');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('D');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('E');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('S');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('.');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('C');  //  call transmit function
                    delay(100);           // calling of delay function
                    serial_transmit('O');  //  call transmit function
```

```
                        delay(100);            // calling of delay function
                        serial_transmit('M');  //  call transmit function
                        delay(100);            // calling of delay function
                       serial_transmit(0x0d);  //  call transmit function
                       delay(400);
                       delay(400);

           }
   }

void pll()
   {
       /*PLL IS CONFIGURED TO GET 60HZ pCLK*/
       PLLCFG=0X24;           // SET PSEL=2 AND MSEL=5
       PLLCON=0X01;           //PLL IS ACTIVE BUT NOT YET CONNECT
       PLLFEED=0XAA;          //FEED SEQUENCE
       PLLFEED=0X55;          //FEED SEQUENCE
       while((PLLSTAT & 0X400)==0);   //WAIT FOR FEED SEQUENCE TO BE INSERTED
       PLLCON=0X03;           // PLL HAS BEEN ACTIVE AND BEING CONNECTRD
       VPBDIV=0X00;           // SET PCLK SAME AS FCCLK
       PLLFEED=0XAA;          //FEED SEQUENCE
       PLLFEED=0X55;          //FEED SEQUENCE
   }

/* SERIAL INITILIZATION*/

void serial_ini()
   {

       U0LCR =0x83;
       U0DLM=0X00;
       U0DLL=0X5e;
       U0FDR=0X52;
       U0LCR =0x03;
   }

void serial_transmit(unsigned char x)
   {
       U0THR =x;                    // LOAD DATA IN U0THR REGISTER
       while ((U0LSR & 0x40)==0);  // WAIT FOR DATA TRANSMISSION
       U0LSR|=0X40;
   }

 void delay(int x)
   {
        int i,j;
        x=x*10;
        for(i=0;i<x;i++)
        for(j=0;j<350;j++);
   }
```

CIRCUIT DIAGRAM: DATA SEND FROM ARM7 (LPC2148) THROUGH RS232 PROTOCOL AND DISPLAY DATA ON PC



Inference:

1.
2.
3.
4.

Result

Thus serial communication is established between PC and ARM processor successfully.

****