ESIOT LAB EXTERNAL

1. Write the following Assembly programs for 8051

a) Create a square wave of 50% duty cycle on bit 0 of port 1

b) Create a square wave of 66% duty cycle on bit 3 of port 1

NOTE: Use Software delays

1st Program:

```
L1: SETB P1.0
      ACALL DELAY

      CLR P1.0
      ACALL DELAY

      SJMP L1

DELAY: MOV R0, #0FFH
L2: DJNZ R0, L2
       RET

END
```

2nd Program:

```
L1: SETB P1.3
      ACALL DELAY
      ACALL DELAY

      CLR P1.3
      ACALL DELAY
      SJMP L1

DELAY: MOV R0, #0FFH
L2: DJNZ R0, L2
       RET

END
```

Duty Cycle is 50%: (High wave – 50%, Low wave – 50%)

Calling Delay Ratio is 1:1

Duty Cycle is 66%: (High wave – 66%, Low wave – 33%)

Calling Delay Ratio is 2:1

2. A switch is connected to pin p1.7. Write C and assembly language program (8051) to check the status of switch and perform the following: If switch status is 0, send 45H to P2 else 55H (use Keil simulator)

ASSEMBLY LANGUAGE PROGRAM:

```
SETB P1.7
L1: JB P1.7, L2

        MOV A, #45H
        MOV P2, A
        SJMP L3

        L2: MOV A, #55H
                MOV P2, A
                L3: SJMP L1
END
```

C LANGUAGE PROGRAM

```
#include <reg51.h>

sbit s = P1^7;
#define LED P2

void main() {
        while(1) {
                if (s == 00)
                        LED = 0x45;
                else
                        LED = 0x55;
        }
}
```

3. Write an assembly and C program (8051) for generating a square wave of 50% duty cycle on the P1.5 bit. Use timer0 in mode1 to generate the time delay. Show the calculations for the frequency of the square wave generated. (Assume clock frequency as 11.0592MHz)

PROGRAM:

```
#include <reg51.h>
sbit b = P1^5;
void main(){
        TMOD = 0x01;
        while(1){
                TH0 = 0xFF;
                TL0 = 0x1A;
                b = ~b;
                TR0 = 0x01;
                while(TF0 == 0);
                TR0 = 0x00;
                TF0 = 0x00;
        }
}
```

THEORY:

Square wave of frequency say 2 kHz at pin P1.5 using Timer 0 in Mode1
T = 1/f = 1/2kHz = 0.5ms
50% duty cycle means 1/2 for high and 1/2 for low => T = 250us
250us/1.085us = 230
maximum in mode0 is FFFFH or 65535 - 230 + 1 = 65306 or FF1AH => TL = 1A, TH = FF

4. Assuming that XTAL=11.0592MHz, write an assembly and C program (8051) to generate a square of 2KHz frequency on pin p1.5. Show relevant calculations.

ASSEMBLY LANGUAGE PROGRAM:

```
MOV TMOD, #01H
L1: MOV TL0, #1AH
    MOV TH0, #0FFH
    CPL P1.5
    SETB TR0
    TARGET: JNB TF0, TARGET
            CLR TR0
            CLR TF0
             SJMP L1
END
```

C LANGUAGE PROGRAM:

```
#include <reg51.h>
sbit b = P1^5;
void main(){
        TMOD = 0x01;
        while(1){
                TH0 = 0xFF;
                TL0 = 0x1A;
                b = ~b;
                TR0 = 0x01;
                while(TF0 == 0);
                TR0 = 0x00;
                TF0 = 0x00;
        }
}
```

5. **Assuming that XTAL =11.0592MHz, write an assembly and C program (8051) for generating a square wave of smallest frequency with timer0 in mode1. Show relevant calculations.**

CALCULATIONS:

Timer 0 + Mode – 1 counts up to 65,535 pulses

Number of clock pulses = (delay / 1.085us)

Time period = (2 * delay)

Frequency = (1 / Time period) = (1 / 2 * delay)

Delay = (1 / 2 * Frequency)

Number of clock pulses = (1 / (2 * Frequency * 1.085us))

Frequency = (1 / (2 * 1.085 * Number of clock pulses))

If we want smallest frequency then number of clock pulses is to be **maximized.**

Frequency = (1 / (2 * 1.085us * **65535**)) = 0.0070 KHz

ASSEMBLY LANGUAGE PROGRAM:

```
MOV TMOD, #01H
L1: MOV TH0, #00H
        MOV TL0, #00H
        CPL P1.1
        SETB TR0
        L2: JNB TF0, L2
        CLR TF0
        CLR TR0
        SJMP L1

END
```

C LANGUAGE PROGRAM:

```
#include <reg51.h>
sbit b = P1^1;

void main(void) {
        TMOD = 0x01;
        while(1) {
                TH0 = 0x00;
                TL0 = 0x00;
                b = ~b;
                TR0 = 0x01;
                while (TF0 == 0);
                TF0 = 0x00;
                TR0 = 0x00;
        }
}
```

6. Assuming that XTAL =11.0592MHz, write an assembly and C program (8051) for generating a square wave of largest frequency with timer0 in mode1. Show relevant calculations

CALCULATIONS:

Timer 0 + Mode – 1 counts up to 65,535 pulses

Number of clock pulses = (delay / 1.085us)

Time period = (2 * delay)

Frequency = (1 / Time period) = (1 / 2 * delay)

Delay = (1 / 2 * Frequency)

Number of clock pulses = (1 / (2 * Frequency * 1.085us))

Frequency = (1 / (2 * 1.085 * Number of clock pulses))

If we want smallest frequency then number of clock pulses is to be **minimized.**

Frequency = (1 / (2 * 1.085us * **1**)) = 460829.493 Hz

ASSEMBLY LANGUAGE PROGRAM:

```
MOV TMOD, #01H
L1: MOV TH0, #0FFH
        MOV TL0, #0FFH
        CPL P1.1
        SETB TR0
        L2: JNB TF0, L2
        CLR TF0
        CLR TR0
        SJMP L1

END
```

C LANGUAGE PROGRAM:

```
#include <reg51.h>
sbit b = P1^1;

void main(void) {
        TMOD = 0x01;
        while(1) {
                TH0 = 0xFF;
                TL0 = 0xFF;
                b = ~b;
                TR0 = 0x01;
                while (TF0 == 0);
                TF0 = 0x00;
                TR0 = 0x00;
        }
}
```

7. Assuming that XTAL=11.0592MHz, 8051 timer1 operated in mode2 and started with initial value of 5 H. write an assembly and C language program for generating a square on p2.0. Calculate the frequency of the square wave generated.

ASSEMBLY LANGUAGE PROGRAM:

```
MOV TMOD, #20H
MOV TH1, #5H
L1: CPL P2.0
        SETB TR1
        L2: JNB TF1, L2
        CLR TF1
        CLR TR1
        SJMP L1

END
```

C LANGUAGE PROGRAM:

```
#include <reg51.h>
sbit b = P2^0;

void main(void) {
        TMOD = 0x20;
        TH1 = 0x05;
        while(1) {
                b = ~b;
                TR1 = 0x01;
                while (TF1 == 0);
                TF1 = 0x00;
                TR1 = 0x00;
        }
}
```

Frequency Calculation:

Number of clock pulses counted are = (255 – 5) + 1(Roll over to 00H)
= 251

Number of clock pulses to be counted = (delay / 1.085us)
251 = (delay / 1.085us)
Delay = 272.335 us

Total time period = (2 * Delay) = (2 * 272.335) = 544.67us
Frequency = (1 / Total Time period) = (1 / 544.67us) = 1835Hz

8. Write an assembly and C program for the 8051 to transfer letters “YES” only 5 times. Write and explain Special Function Registers used for framing the program.

ASSEMBLY LANGUAGE PROGRAM

```
mov b, #05h
mov tmod, #20h
mov th1, #-3
mov scon, #50h

again: setb tr1
      mov A,#"Y"
      acall trans
      mov A,#"E"
      acall trans
      mov A,#"S"
      acall trans
      mov A,#" "
      acall trans
      clr tr1
      djnz b, again

trans: mov sbuf,A
here: jnb ti,here
   clr ti
        ret

end
```

C LANGUAGE PROGRAM

```
#include<reg51.h>
void SerTx(unsigned char);
void main(void)
{
        int i = 5;
        TMOD=0x20;
        TH1=0xFD;
        SCON=0x50;
        while(i >= 0)
        {
                TR1 = 1;
                 SerTx('Y');
                 SerTx('E');
                 SerTx('S');
                 SerTx(' ');
               TR1 = 0;

               i -= 1;
        }
}
void SerTx(unsigned char x)
{
        SBUF=x;
        while(TI==0);
        TI=0;
}
```

9. Write an assembly and C program for the 8051 to transfer letters “RANI” continuously. Write and explain Special Function Registers used for framing the program.

ASSEMBLY LANGUAGE PROGRAM

```
mov tmod, #20h
mov th1, #-3
mov scon, #50h

again: setb tr1
            mov A,#"R"
      acall trans
            mov A,#"A"
      acall trans
            mov A,#"N"
      acall trans
            mov A,#"I"
            acall trans
            mov A,#" "
            acall trans
            clr tr1
            sjmp again

trans: mov sbuf,A
here: jnb ti,here
     clr ti
           ret

end
```

C LANGUAGE PROGRAM

```
#include<reg51.h>
void SerTx(unsigned char);
void main(void)
{
        TMOD=0x20;
        TH1=0xFD;
        SCON=0x50;
        while(1)
        {
                TR1 = 1;
                 SerTx('R');
                 SerTx('A');
                 SerTx('N');
                 SerTx('I');
                 SerTx(' ');
               TR1 = 0;
        }
}
void SerTx(unsigned char x)
{
        SBUF=x;
        while(TI==0);
        TI=0;
}
```

10. Write an assembly and C program for the 8051 to transfer letters “IT DEPT” only one time. Write and explain Special Function Registers used for framing the program.

ASSEMBLY LANGUAGE PROGRAM

```
mov b, #01h
mov tmod, #20h
mov th1, #-3
mov scon, #50h

again: setb tr1
      mov A,#"I"
      acall trans
      mov A,#"T"
      acall trans
      mov A,#" "
      acall trans
      mov A,#"D"
      acall trans
      mov A,#"E"
      acall trans
      mov A,#"P"
      acall trans
      mov A,#"T"
      acall trans
      clr tr1
      djnz b, again

trans: mov sbuf,A
here: jnb ti,here
   clr ti
        ret

  end
```

C LANGUAGE PROGRAM

```
#include<reg51.h>
void SerTx(unsigned char);
void main(void)
{
        int i = 1;
        TMOD=0x20;
        TH1=0xFD;
        SCON=0x50;
        while(i >= 0)
        {
                TR1 = 1;
                 SerTx('I');
                 SerTx('T');
                 SerTx(' ');
                 SerTx('D');
                 SerTx('E');
                 SerTx('P');
                 SerTx('T');

                TR1 = 0;

                i -= 1;
        }
}
void SerTx(unsigned char x)
{
        SBUF=x;
        while(TI==0);
        TI=0;
}
```

11. Write an assembly and C program for the 8051for generating square wave that has high portion 25 micro seconds and low portion 15 Micro seconds using Timer 0 with mode of operation of your choice and also show relevant calculations.

ASSEMBLY LANGUAGE PROGRAM:

```
MOV TMOD, #01H
L1: SETB TR0
    ACALL DELAY1
    CLR TR0

    CPL P3.3
    SETB TR0
    ACALL DELAY2
    CLR TR0

    SJMP L1

DELAY1:  MOV TH0, #0FFH
            MOV TL0, #0E9H
            L2: JNB TF0, L2
                  CLR TF0
                  RET

DELAY2:  MOV TH0, #0FFH
            MOV TL0, #0F3H
            L3: JNB TF0, L3
                  CLR TF0
                  RET

END
```

C LANGUAGE PROGRAM:

```
#include <reg51.h>
sbit b = P3^3;

void delayH();
void delayL();

void main(){
        TMOD = 0x01;
        while(1) {
                b = 1;
                TR0 = 1;
                delayH();
                TR0 = 0;

                b = 0;
                TR0 = 1;
                delayL();
                TR0 = 0;
        }
}

void delayH() {
        TH0 = 0xFF;
        TL0 = 0xE9;
        while(TF0 == 0);
        TF0 = 0x00;
}

void delayL() {
        TH0 = 0xFF;
        TL0 = 0xF3;
        while(TF0 == 0);
        TF0 = 0x00;
}
```

12. **Write an assembly program for the 8051 for generating a square wave of 50Hz at pin p1.2 use timer 0 in mode2 for delays and simultaneously transfer data from p2 to p0. (HINT: operate timer in interrupt mode). Show relevant calculations for delay.**

PROGRAM:

```
ORG 0000H
LJMP MAIN
ORG 000BH
CPL P1.2
RETI
ORG 0030H
MAIN:    MOV TMOD, 02H
         MOV TH0, #-18177
         MOV P2, #0FFH
         MOV IE, #82H
         SETB TR0
         HERE: MOV A, P2
                 MOV P0, A
        SJMP HERE
END
```

13. **Write an assembly and C program for rotating stepper motor 270 degrees clock wise and 180 degrees anti-clock wise .Show relevant calculations**

ASSEMBLY LANGUAGE PROGRAM:

```
L1:  MOV R2, #150
     MOV A, #11 H
H2:  MOV P1, A
     MOV R3, #0FFh
H1:  DJNZ R3, H1
     RR A
     DJNZ R2, H2
     MOV R2, # 100
H4: MOV P1, A
     MOV R3, #0FFh
H5:  DJNZ R3, H5
     RL A
     DJNZ R2, H4
SJMP L1
```

C LANGUAGE PROGRAM:

```
#include<reg51.h>
void delay(unsigned int d);
int main(void)
{
        unsignedint i;
        while(1)
        {
                for(i = 0; i < 47; i++)
                {
                        P0=0x01;
                        delay(10);
                        P0=0x02;
                        delay(10);
                        P0=0x04;
                        delay(10);
                        P0=0x08;
                        delay(10);
                 }
                P0 = 0x01;
                delay(10);
                P0 = 0x02;
                delay(10);
                for(i=0;i<25;i++)
                {
                        P0=0x01;
                        delay(10);
                        P0=0x08;
                        delay(10);
                        P0=0x04;
                        delay(10);
                        P0=0x02;
                        delay(10);
                }
        }
}
void delay(unsigned int d)
{
        unsignedinti,j;
        for(i=0;i<d;i++)
        for(j=0;j<101;j++);
}
```

14. Write an assembly and C program for rotating stepper motor 90 degrees clock wise and 360 degrees anti-clock wise continuously. Show relevant calculations.

ASSEMBLY LANGUAGE PROGRAM:

```
L1:  MOV R2, #50
     MOV A, #11 H
H2:  MOV P1, A
     MOV R3, #0FFh
H1:  DJNZ R3, H1
     RR A
     DJNZ R2, H2
     MOV R2, # 200
H4: MOV P1, A
     MOV R3, #0FFh
H5:  DJNZ R3, H5
     RL A
     DJNZ R2, H4
SJMP L1
```

C LANGUAGE PROGRAM

```
#include<reg51.h>
void delay(unsigned int d);
int main(void)
{
        unsignedint i;
        while(1)
        {
                for(i = 0; i < 12; i++)
                {
                        P0=0x01;
                        delay(10);
                        P0=0x02;
                        delay(10);
                        P0=0x04;
                        delay(10);
                        P0=0x08;
                        delay(10);
                 }
                P0 = 0x01;
                delay(10);
                P0 = 0x02;
                delay(10);
                for(i=0;i<50;i++)
                {
                        P0=0x01;
                        delay(10);
                        P0=0x08;
                        delay(10);
                        P0=0x04;
                        delay(10);
                        P0=0x02;
                        delay(10);
                }
        }
}
void delay(unsigned int d)
{
        unsignedinti,j;
        for(i=0;i<d;i++)
        for(j=0;j<101;j++);
}
```

15. Write an assembly and C program for generating a triangular wave using DAC on two ports.

ASSEMBLY LANGUAGE PROGRAM:

```
L1:   MOV A, #00H
      MOV R0, #0FFH
L2:   MOV P1, A
      MOV P2, A
      INC A
      DJNZ R0, L2
      MOV R0, #0FFH
L3:   MOV P1, A
      MOV P2, A
      DEC A
      DJNZ R0, L3
      SJMP L1
```

C LANGUAGE PROGRAM:

```
#include <reg51.h>
void main(void) {
        unsigned int x;
        while(1) {
                for (x = 0; x < 255; x++) {
                        P1 = x;
                        P2 = x;
                }

                for (x = 255; x > 0; x--) {
                        P1 = x;
                        P2 = x;
                }
        }
}
```

16. Write an assembly and C program for generating square wave on port and triangular wave on other port using DAC

ASSEMBLY LANGUAGE PROGRAM:

```
HERE: MOV A, #00H
          MOV R0, #0FFH
          L1: MOV P0, A
          L2: MOV P1, A
                   INC A
                   DJNZ R0, L2
          L3: MOV P0, A
          L4: MOV P1, A
                   DEC A
                   DJNZ R0, L4

          SJMP HERE
END
```

C LANGUAGE PROGRAM:

```
#include <reg51.h>
void main(void) {
         unsigned int x;
         while(1) {
                  P0 = 0;
                  for (x = 0; x < 255; x++) {
                           P1 = x;
                  }

                  P0 = 1;
                  for (x = 255; x > 0; x--) {
                           P1 = x;
                  }
         }
}
```

17. Write a C program to display your roll number using seven segment display interfaces

PROGRAM:

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

void main(void) {
        int d, b, s, i, j, k;
        int port[4] = {0xC0, 0xFC, 0x89, 0xA1};

        while(1) {
                for (d = 0; d < 1; d++) {
                        i = 0;
                        for (b = 0; b < 4; b++) {
                                k = port[i++];
                                for (j = 0; j < 8; j++) {
                                        s = k;
                                        s &= 0x80;

                                        if (s == 0x00)
                                                P1 = 0x00;
                                        else
                                                P1 = 0x01;

                                        P2 = 0x01;
                                        P2 = 0x00;

                                        s = k;
                                        s = s << 1;
                                        k = s;
                                }
                        }
                }
        }
}
```

18. **Write a program to display “Your name and Roll no” on LCD display interface.**

Hex Code	**Command to LCD Instruction Register**
0F	LCD ON, cursor ON
01	Clear display screen
02	Return home
04	Decrement cursor (shift cursor to left)
06	Increment cursor (shift cursor to right)
05	Shift display right
07	Shift display left
0E	Display ON, cursor blinking
80	Force cursor to beginning of first line
C0	Force cursor to beginning of second line
38	2 lines and 5×7 matrix
83	Cursor line 1 position 3
3C	Activate second line
08	Display OFF, cursor OFF
C1	Jump to second line, position 1
OC	Display ON, cursor OFF
C2	Jump to second line, position 2

PROGRAM:

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

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

sfr ldata = 0xA0;
sbit rs = P3^7;
sbit rw = P3^6;
sbit en = P3^5;

void main() {
        char* name = "Rishik";
        char* rno = "1602-19-737-156";
        int l1 = strlen(name), l2 = strlen(rno), i;

        lcdcmd(0x38);
        delay(500);
        lcdcmd(0x0e);
        delay(500);
        lcdcmd(0x01);
        delay(500);
        lcdcmd(0x06);
        delay(500);
        lcdcmd(0x85);
        delay(500);

        i = 0;
        for (; i < l1; i++) {
                lcddata(name[i]);
                delay(500);
        }

        lcdcmd(0xC2);
        delay(500);

        i = 0;
        for (; i < l2; i++) {
                lcddata(rno[i]);
                delay(500);
        }
}

void lcdcmd(unsigned char x) {
        ldata = x;
        rs = 0;
        rw = 0;
        en = 1;
        delay(500);
        en = 0;
```

```
	return;
}

void lcddata(unsigned char x) {
	ldata = x;
	rs = 1;
	rw = 0;
	en = 1;
	delay(500);
	en = 0;
	return;
}

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

19. Write a program to interface ADC (analog to digital converter) with 8051 using proteus software.

PROGRAMS:

1. MYLIB.H:

```
#include<at89x52.h>
sfr ldata=0xa0;
sbit rs=P3^7;
sbit rw=P3^6;
sbit en=P3^5;

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

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

void lcddata(unsigned char value)
{
            ldata=value;
            rs=1;
            rw=0;
            en=1;
            msdelay(1);
            en=0;
}

void lcdprint(unsigned char *msg)
{
   while(*msg)
   {
      lcddata(*msg);
      msg++;
   }
}
void lcd_init8bit()
{
   lcdcmd(0x38);
   lcdcmd(0x0e);
   lcdcmd(0x01);
   lcdcmd(0x06);
}
```

2. ADC.C:

```
#include<MYLIB.h>
#define input_port P1

sbit ADDA=P0^0;
sbit ADDB=P0^1;
sbit ADDC=P0^2;
sbit ale=P0^3;
sbit sc=P0^4;
sbit eoc=P0^5;
sbit oe=P0^6;

unsigned char number;

void convert(unsigned char value)
{
        unsigned char x,d1,d2,d3;
        x=value/10;
        d1=value%10;
        d2=x%10;
        d3=x/10;
        lcddata(d3+48);
        msdelay(10);
        lcddata(d2+48);
        msdelay(10);
        lcddata(d1+48);
        msdelay(10);
}

void main()
{
         number=0;
         eoc=1;
         ale=0;
         oe=0;
         sc=0;
         ADDC=0;
         ADDB=0;
         ADDA=0;

        lcd_init8bit();
        lcdcmd(0x83);
        lcdprint(" ADC 0808  ");
        lcdcmd(0xc1);
        lcdprint(" Interfacing   ");
        msdelay(500);
        lcdcmd(1);

        while(1) {
                ADDC=0;
                ADDB=0;
                ADDA=0;
                ale=1;
```

```
          sc=1;
          msdelay(1);
          ale=0;
          sc=0;

      while(eoc==0);
      oe=1;
      number=input_port;
      msdelay(1);
      oe=0;
      lcdcmd(0x80);
      lcdprint("var res =  ");

      convert(number);
     }
}
```

20. Write a program to interface decimal keyboard with 8051 using proteus software.

PROGRAMS:

1. MYLIB.H:

```
#include<at89x52.h>
sfr ldata=0xa0;
sbit rs=P3^7;
sbit rw=P3^6;
sbit en=P3^5;

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

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

void lcddata(unsigned char value)
{
            ldata=value;
            rs=1;
            rw=0;
            en=1;
            msdelay(1);
            en=0;
}

void lcdprint(unsigned char *msg)
{
   while(*msg)
   {
      lcddata(*msg);
      msg++;
   }
}
void lcd_init8bit()
{
   lcdcmd(0x38);
   lcdcmd(0x0e);
   lcdcmd(0x01);
   lcdcmd(0x06);
}
```

2. Key.C:

```
#include<at89x52.h>
#include "MYLIB.h"
unsigned char keyScan(void);
sbit R1=P1^0;
sbit R2=P1^1;
sbit R3=P1^2;
sbit R4=P1^3;
sbit C1=P1^4;
sbit C2=P1^5;
sbit C3=P1^6;

void main (void) {
        unsigned char key;
        while(1) {
                key=keyScan();
                P2=key;
                msdelay(100);
        }
}

unsigned char keyScan(void)
{
        R1=R2=R3=R4=0;
        C1=C2=C3=1;
        while(1) {
                if((C1==0)||(C2==0)||(C3==0)) {
                        R1=0;R4=R2=R3=1;
                        if (C1==0)
                                return 1;
                        if (C2==0)
                                return 2;
                        if (C3==0)
                                return 3;

                        R2=0;R4=R1=R3=1;
                        if (C1==0)
                                return 4;
                        if (C2==0)
                                return 5;
                        if (C3==0)
                                return 6;

                        R3=0;R4=R2=R1=1;
                        if (C1==0)
                                return 7;
                        if (C2==0)
                                return 8;
                        if (C3==0)
                                return 9;
```

```
                              R4=0;R1=R2=R3=1;
                              if (C1==0)
                                        return 10;
                              if (C2==0)
                                        return 0;
                              if (C3==0)
                                        return 12;

                    }
          }
}
```

21. Write a program to initiate multiple processes using VxWorks tasking routines.

PROGRAM:

```
#include "VxWorks.h"
#include "stdio.h"
#define ITERATIONS 10
void print(void);
Spawn_ten()
{
        int i, taskId;
        for(i=0;i<ITERATIONS;i++)
                taskId=taskSpawn("tprint",90,0x100,2000,
        print,0,0,0,0,0,0,0,0,0,0);
}
void print(void)
{
        printf("hello, I am task %d\n",taskIdSelf());
}
```

22. Write a program to demonstrate the use of VxWorks semaphores

PROGRAM:

```
#include"vxworks.h"
#include"tasklib.h"
#include"semLib.h"
#include"stdio.h"
void taskOne(void);
void taskTwo(void);
#define ITER 10
SEM_ID semBinary;
itnt global=0;
void binary(void)
{
        int taskIdOne, taskIdTwo;
        semBinary=semBCreate(SEM_Q_FIFO,SEM_FULL);
        semTake(semBinary,WAIT_FOREVER);
        taskIdOne=taskSpawn("t1",90,0x100,2000,(FUNCPTR)taskOne,0,0,0,0,0,0,0,0,0,0);
        taskIdTwo=taskSpawn("t2",90,0x100,2000,(FUNCPTR)taskTwo,0,0,0,0,0,0,0,0,0,0);
}
void taskOne(void)
{
        int i;
        for(i=0;i<ITER;i++)
        {
                semTake(semBinary,WAIT_FOREVER);
                printf("I am taskOne and global=%d.........\n",++global);
                semGive(semBinary);
        }
}
void taskTwo(void)
{
        int i;
        semGive(semBinary);
        for(i=0;i<ITER;i++)
        {
                semTake(semBinary,WAIT_FOREVER);
                printf("I am taskTwo and global=%d..\n",--global);
                semGive(semBinary);
        }
}
```

23. Write a program for creating a Queue and to transfer message from task2 to task1 in VxWorks

PROGRAM:

```
#include"VxWorks.h"
#include"msgQLib.h"
#include "stdio.h"
void taskOne(void);
void taskTwo(void);
#define MAX_MESSAGES 100
#define MAX_MESSAGE_LENGTH 50
MSG_Q_ID mesgQueueId;
void message(void)
{
        int taskIdOne, taskIdTwo ;
        if ((mesgQueueId=msgQCreate(MAX_MESSAGES,MAX_MESSAGE_LENGTH,MSG_Q_FIFO))==NULL)
                printf("msgQCreate is failed \n");
        if ((taskIdOne=taskSpawn("t1",90,0x100,2000,(FUNCPTR)taskOne,0,0,0,0,0,0,0,0,0,0))==ERROR)
                printf("taskSpawn mesg  taskOne failed \n");
        if ((taskIdTwo=taskSpawn("t2",90,0x100,2000,(FUNCPTR)taskTwo,0,0,0,0,0,0,0,0,0,0))==ERROR)
                printf("taskSpawntaskTwo failed \n");
}

void taskTwo(void)
{
        char message[]="received mesg from taskTwo";
        if((msgQSend(mesgQueueId,message,MAX_MESSAGE_LENGTH,WAIT_FOREVER,MSG_PRI_NORMAL)==ERRO
        R))
                printf("msgQSend in taskTwo failed \n");
}

void taskOne(void)
{
        char msgBuf[MAX_MESSAGE_LENGTH];
        if(msgQReceive(mesgQueueId,msgBuf,MAX_MESSAGE_LENGTH,WAIT_FOREVER)==ERROR)
                printf("\n msgQReceive in taskOne failed");
        else
                printf("\n %s",msgBuf);
        msgQDelete(mesgQueueId);
}
```

24. Write a program for creating a Queue and to transfer message from task1 to task2 in VxWorks

PROGRAM:

```
#include"VxWorks.h"
#include"msgQLib.h"
#include "stdio.h"
void taskOne(void);
void taskTwo(void);
#define MAX_MESSAGES 100
#define MAX_MESSAGE_LENGTH 50
MSG_Q_ID mesgQueueId;
void message(void)
{
        int taskIdOne, taskIdTwo ;
        if ((mesgQueueId=msgQCreate(MAX_MESSAGES,MAX_MESSAGE_LENGTH,MSG_Q_FIFO))==NULL)
                printf("msgQCreate is failed \n");
        if ((taskIdOne=taskSpawn("t1",90,0x100,2000,(FUNCPTR)taskOne,0,0,0,0,0,0,0,0,0,0))==ERROR)
                printf("taskSpawn mesg  taskOne failed \n");
        if ((taskIdTwo=taskSpawn("t2",90,0x100,2000,(FUNCPTR)taskTwo,0,0,0,0,0,0,0,0,0,0))==ERROR)
        printf("taskSpawntaskTwo failed \n");
}

void taskOne(void)
{
char message[]="received mesg from taskOne";
if((msgQSend(mesgQueueId,message,MAX_MESSAGE_LENGTH,WAIT_FOREVER,MSG_PRI_NORMAL)==ERROR))
        printf("msgQSend in taskOne failed \n");
}

void taskTwo(void)
{
        char msgBuf[MAX_MESSAGE_LENGTH];
        if(msgQReceive(mesgQueueId,msgBuf,MAX_MESSAGE_LENGTH,WAIT_FOREVER)==ERROR)
                printf("\n msgQReceive in taskTwo failed");
        else
                printf("\n %s",msgBuf);
        msgQDelete(mesgQueueId);
}
```

25. Write a python program for blinking of LEDs which are interfaced through MCP23017 and connected to Raspberry Pi.

- **RBG LED BLINKING**
- **RRR LED BLINKING**
- **BBB LED BLINKING**
- **GGG LED BLINKING**

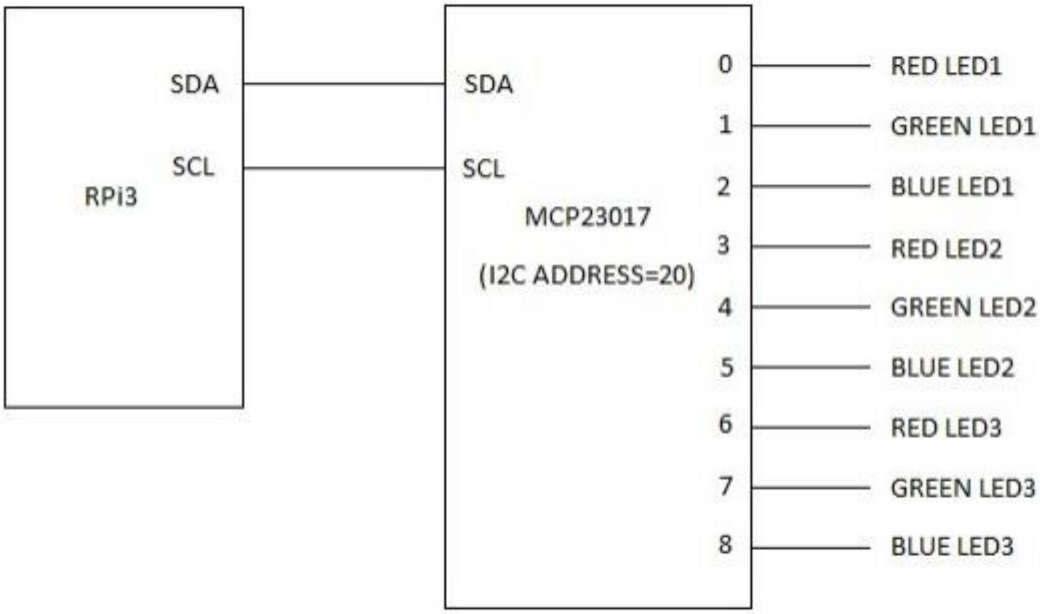


Figure: Connection diagram of RGB LED

RBG LED BLINKING:

```
import time
import sys

sys.path.append(‘/home/pi/Adafruit-Raspberry-Pi-Python-Code-legacy/Adafruit_MCP230xx’)
from Adafruit_MCP230XX import Adafruit_MCP230XX

mcp = Adafruit_MCP230XX(busnum = 1, address = 0x20, num_gpios = 16)

mcp.config(0, mcp.OUTPUT)
mcp.config(1, mcp.OUTPUT)
mcp.config(2, mcp.OUTPUT)
mcp.config(3, mcp.OUTPUT)
mcp.config(4, mcp.OUTPUT)
mcp.config(5, mcp.OUTPUT)
mcp.config(6, mcp.OUTPUT)
mcp.config(7, mcp.OUTPUT)
mcp.config(8, mcp.OUTPUT)

try:
        while True:
                mcp.output(0, 1)
                mcp.output(5, 1)
                mcp.output(7, 1)
                time.sleep(1)
                mcp.output(0, 0)
                mcp.output(5, 0)
                mcp.output(7, 0)
                time.sleep(1)
except KeyboardInterrupt:
        mcp.output(0, 0)
        mcp.output(1, 0)
        mcp.output(2, 0)
        mcp.output(3, 0)
        mcp.output(4, 0)
        mcp.output(5, 0)
        mcp.output(6, 0)
        mcp.output(7, 0)
        mcp.output(8, 0)
```

RRR LED BLINKING:

```
import time
import sys

sys.path.append(‘/home/pi/Adafruit-Raspberry-Pi-Python-Code-legacy/Adafruit_MCP230xx’)
from Adafruit_MCP230XX import Adafruit_MCP230XX

mcp = Adafruit_MCP230XX(busnum = 1, address = 0x20, num_gpios = 16)

mcp.config(0, mcp.OUTPUT)
mcp.config(1, mcp.OUTPUT)
mcp.config(2, mcp.OUTPUT)
mcp.config(3, mcp.OUTPUT)
mcp.config(4, mcp.OUTPUT)
mcp.config(5, mcp.OUTPUT)
mcp.config(6, mcp.OUTPUT)
mcp.config(7, mcp.OUTPUT)
mcp.config(8, mcp.OUTPUT)

try:
        while True:
                mcp.output(0, 1)
                mcp.output(3, 1)
                mcp.output(6, 1)
                time.sleep(1)
                mcp.output(0, 0)
                mcp.output(3, 0)
                mcp.output(6, 0)
                time.sleep(1)
except KeyboardInterrupt:
        mcp.output(0, 0)
        mcp.output(1, 0)
        mcp.output(2, 0)
        mcp.output(3, 0)
        mcp.output(4, 0)
        mcp.output(5, 0)
        mcp.output(6, 0)
        mcp.output(7, 0)
        mcp.output(8, 0)
```

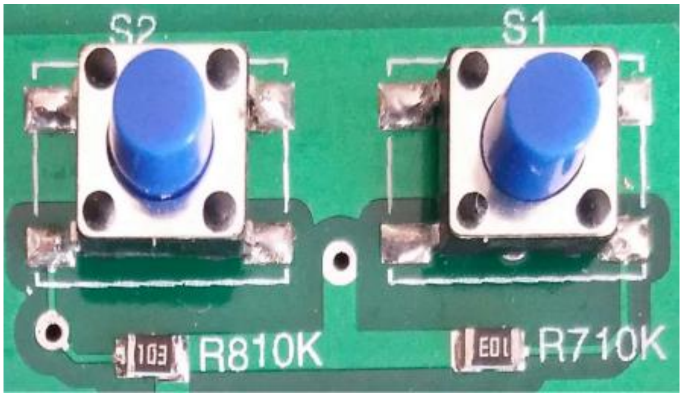
BBB LED BLINKING:

```
import time
import sys

sys.path.append(‘/home/pi/Adafruit-Raspberry-Pi-Python-Code-legacy/Adafruit_MCP230xx’)
from Adafruit_MCP230XX import Adafruit_MCP230XX

mcp = Adafruit_MCP230XX(busnum = 1, address = 0x20, num_gpios = 16)

mcp.config(0, mcp.OUTPUT)
mcp.config(1, mcp.OUTPUT)
mcp.config(2, mcp.OUTPUT)
mcp.config(3, mcp.OUTPUT)
mcp.config(4, mcp.OUTPUT)
mcp.config(5, mcp.OUTPUT)
mcp.config(6, mcp.OUTPUT)
mcp.config(7, mcp.OUTPUT)
mcp.config(8, mcp.OUTPUT)

try:
        while True:
                mcp.output(2, 1)
                mcp.output(5, 1)
                mcp.output(8, 1)
                time.sleep(1)
                mcp.output(2, 0)
                mcp.output(5, 0)
                mcp.output(8, 0)
                time.sleep(1)
except KeyboardInterrupt:
        mcp.output(0, 0)
        mcp.output(1, 0)
        mcp.output(2, 0)
        mcp.output(3, 0)
        mcp.output(4, 0)
        mcp.output(5, 0)
        mcp.output(6, 0)
        mcp.output(7, 0)
        mcp.output(8, 0)
```

GGG LED BLINKING:

```
import time
import sys

sys.path.append(‘/home/pi/Adafruit-Raspberry-Pi-Python-Code-legacy/Adafruit_MCP230xx’)
from Adafruit_MCP230XX import Adafruit_MCP230XX

mcp = Adafruit_MCP230XX(busnum = 1, address = 0x20, num_gpios = 16)

mcp.config(0, mcp.OUTPUT)
mcp.config(1, mcp.OUTPUT)
mcp.config(2, mcp.OUTPUT)
mcp.config(3, mcp.OUTPUT)
mcp.config(4, mcp.OUTPUT)
mcp.config(5, mcp.OUTPUT)
mcp.config(6, mcp.OUTPUT)
mcp.config(7, mcp.OUTPUT)
mcp.config(8, mcp.OUTPUT)

try:
        while True:
                mcp.output(1, 1)
                mcp.output(4, 1)
                mcp.output(7, 1)
                time.sleep(1)
                mcp.output(1, 0)
                mcp.output(4, 0)
                mcp.output(7, 0)
                time.sleep(1)
except KeyboardInterrupt:
        mcp.output(0, 0)
        mcp.output(1, 0)
        mcp.output(2, 0)
        mcp.output(3, 0)
        mcp.output(4, 0)
        mcp.output(5, 0)
        mcp.output(6, 0)
        mcp.output(7, 0)
        mcp.output(8, 0)
```

26. Write a python program for controlling LEDs with switches, connected to Rasberrypi3



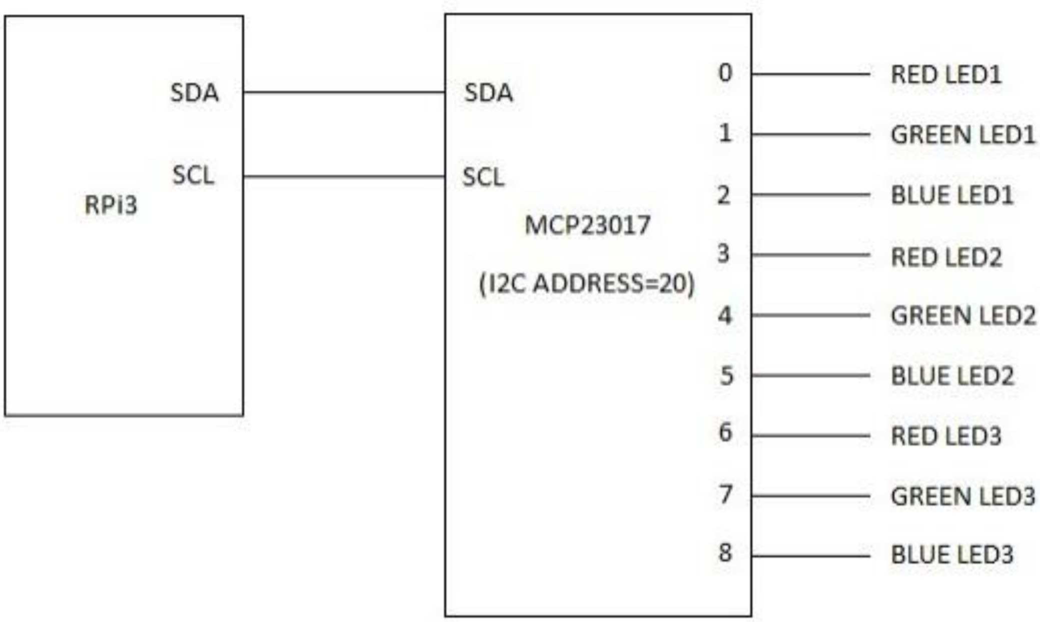


Figure: Connection diagram of RGB LED

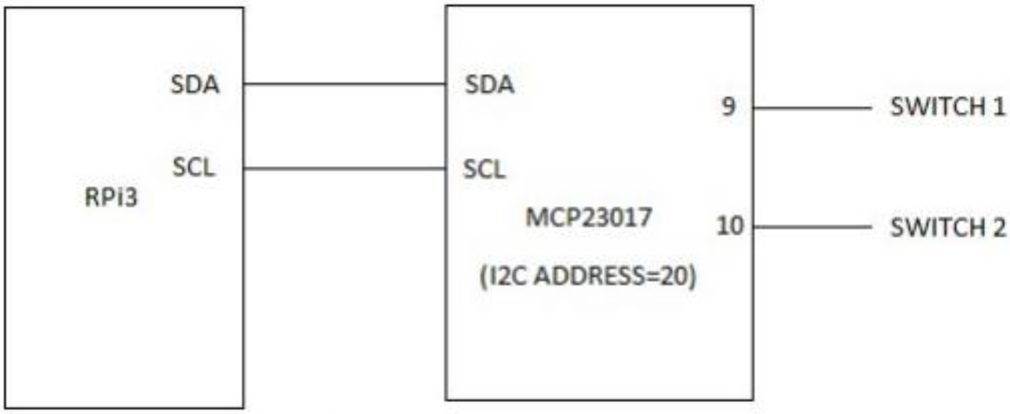


Figure: Connection diagram of SWITCH

PROGRAM:

```
import time
import sys

sys.path.append('/home/pi/Adafruit-Raspberry-Pi-Python-Code-legacy/Adafruit_MCP230xx')
from Adafruit_MCP230XX import Adafruit_MCP230XX

mcp = Adafruit_MCP230XX(busnum = 1, address = 0x20, num_gpios = 16)

mcp.config(0, mcp.OUTPUT)
mcp.config(1, mcp.OUTPUT)
mcp.config(2, mcp.OUTPUT)
mcp.config(3, mcp.OUTPUT)
mcp.config(4, mcp.OUTPUT)
mcp.config(5, mcp.OUTPUT)
mcp.config(6, mcp.OUTPUT)
mcp.config(7, mcp.OUTPUT)
mcp.config(8, mcp.OUTPUT)

mcp.config(9, mcp.INPUT)
mcp.pullup(9, 1)
mcp.config(10, mcp.INPUT)
mcp.pullup(10, 1)

mcp.config(11, mcp.OUTPUT)

try:
        while True:
                print('Switch: 9', mcp.input(9))
                print('Swtich: 10', mcp.input(10))

                x = mcp.input(9)
                y = mcp.input(10)

                if ((x == 512) and (y == 1024)):
                        mcp.output(2, 1)
                        time.sleep(1)
                        mcp.output(0, 0)
                        mcp.output(1, 0)
                        mcp.output(11, 0)
                        time.sleep(1)

                elif ((x == 0) and (y == 1024)):
                        mcp.output(4, 1)
                        time.sleep(1)
                        mcp.output(3, 0)
                        mcp.output(5, 0)
                        mcp.output(11, 0)
                        time.sleep(1)
                elif ((x == 512) and (y == 0)):
                        mcp.output(6, 1)
                        time.sleep(1)
                        mcp.output(7, 0)
                        mcp.output(8, 0)
                        mcp.output(11, 0)
                        time.sleep(1)

                else:
                        mcp.output(11, 1)
                        time.sleep(1)
                        mcp.output(2, 0)
                        mcp.output(4, 0)
                        mcp.output(6, 0)
                        time.sleep(1)

except KeyboardInterrupt:
        mcp.output(0, 0)
        mcp.output(1, 0)
        mcp.output(2, 0)
        mcp.output(3, 0)
        mcp.output(4, 0)
        mcp.output(5, 0)
        mcp.output(6, 0)
        mcp.output(7, 0)
        mcp.output(8, 0)
        mcp.output(9, 0)
        mcp.output(10, 0)
        mcp.output(11, 0)
```

27 a). Write a python program for RELAY CONTROL connected to Raspberry Pi.

b). Write a python program for implement BUZZER control that is interfaced through MCP23017 and connected to Raspberry Pi.

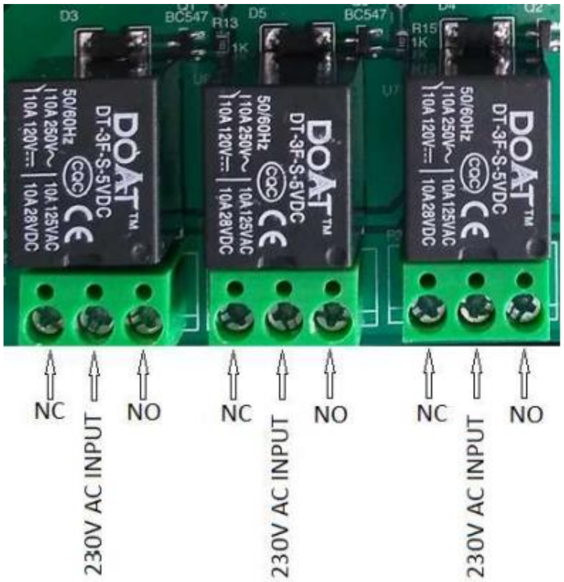


Figure: RELAY Image in ETS-IoT Trainer Kit

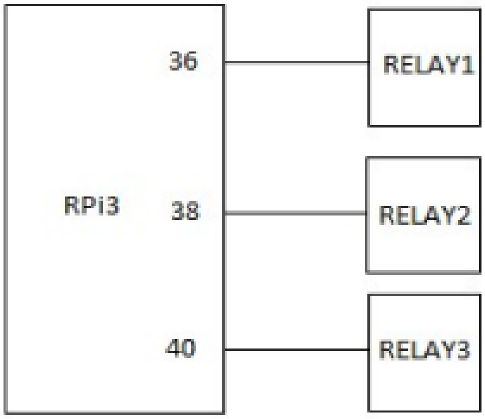


Figure: Connection diagram of RELAY

Figure: BUZZER Image in ETS-IoT Trainer Kit

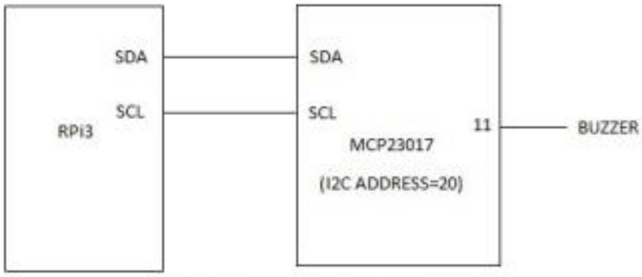


Figure: Connection diagram of BUZZER

REALY CONTROL:

```
import time
import RPi.GPIO as GPIO

GPIO.setwarnings(false)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(36, GPIO.OUT)

try:
    while True:
        GPIO.output(36, 1)
        print(‘Relay ON’)
        time.sleep(1)

        GPIO.output(36, 0)
        print(‘Relay OFF’)
        time.sleep(1)
except KeyboardInterrupt:
    GPIO.cleanup()
```

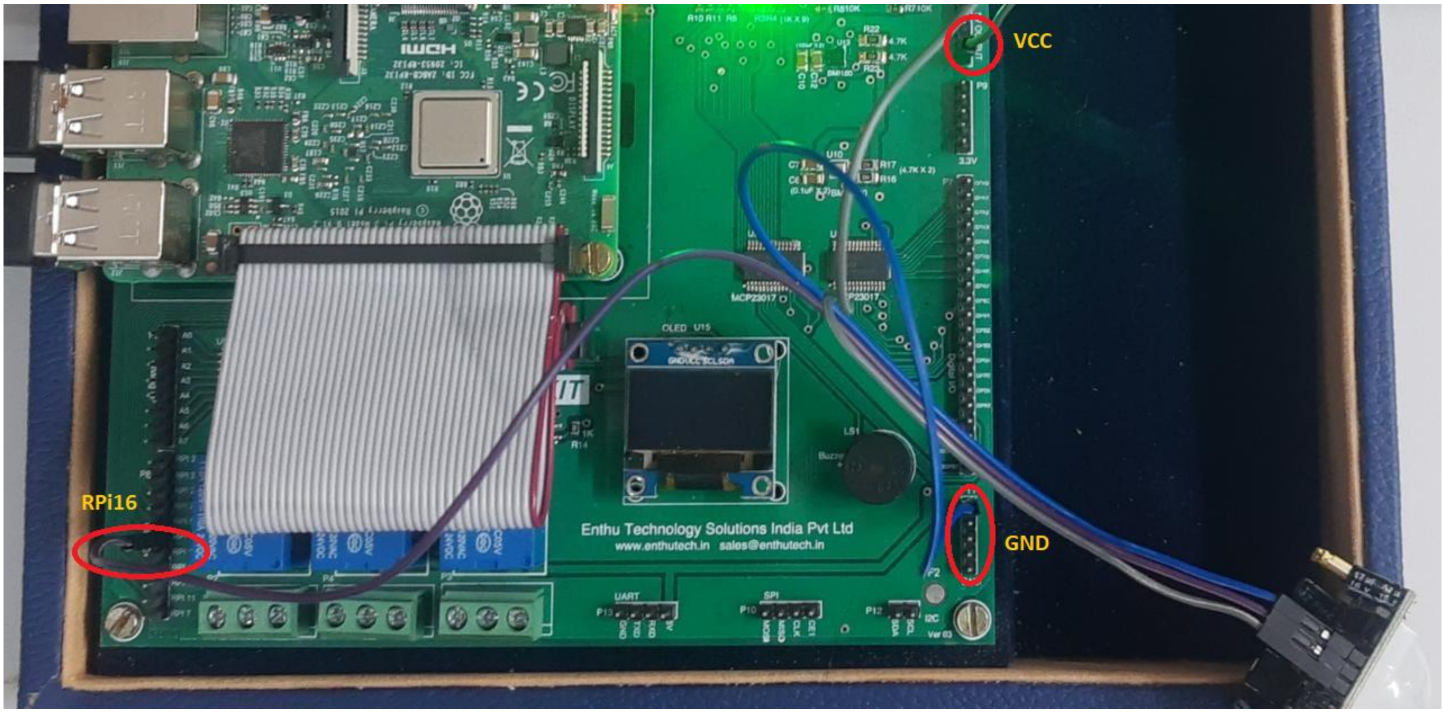
BUZZER CONTROL:

```
import time
import sys

sys.path.append(‘/home/pi/Adafruit-Raspberry-Pi-Python-Code-legacy/Adafruit_MCP230xx’)
from Adafruit_MCP230XX import Adafruit_MCP230XX

mcp = Adafruit_MCP230XX(busnum = 1, address = 0x20, num_gpios = 16)

mcp.config(11, mcp.OUTPUT)

try:
    while True:
        mcp.output(11, 1)
        print(‘Buzzer ON’)
        time.sleep(2)

        mcp.output(11, 0)
        print(‘Buzzer OFF’)
        time.sleep(2)
except KeyboardInterrupt:
    mcp.output(11, 0)
```

28) Write a program to interface PIR sensor to Raspberry Pi.

VCC	5V
GND	GND
OUTPUT	RPi16

CONNECTION DIAGRAM


VCC
RPi16
GND
Enthu Technology Solutions India Pvt Ltd
www.enthutech.in sales@enthutech.in

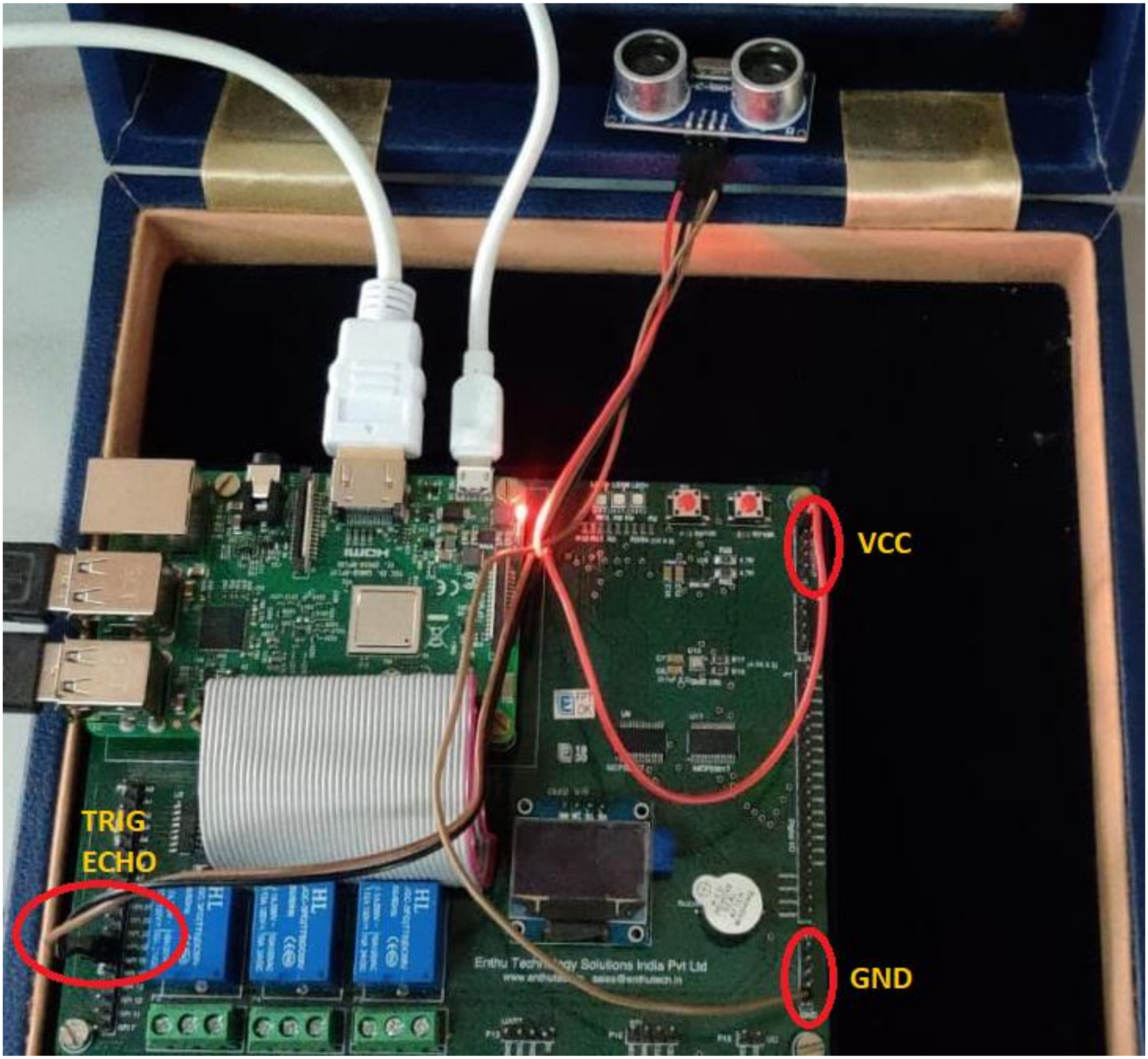
PROGRAM:

```
import time
import RPi.GPIO as GPIO

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(16, GPIO.IN)

try:
        while True:
                i = GPIO.input(16)
                if (i == 1):
                        print('Person DETECTED')
                        time.sleep(1)
                elif (i == 0):
                        print('Person NOT DETECTED')
                        time.sleep(1)
except KeyboardInterrupt:
        GPIO.cleanup()
```

29) Write a program to interface Ultrasonic sensor to Raspberry Pi. Calculate the distance of the object.

VCC	5V
GND	GND
TRIGGER	RPi16
ECHO	RPi18

CONNECTION DIAGRAM



PROGRAM:

```
import time
import RPi.GPIO as GPIO

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)

GPIO_TRIGGER = 16
GPIO_ECHO = 18

GPIO.setup(GPIO_TRIGGER, GPIO.OUT)
GPIO.setup(GPIO_ECHO, GPIO.IN)

GPIO.output(GPIO_TRIGGER, False)
time.sleep(0.5)

try:
    while True:
        GPIO.output(GPIO_TRIGGER, True)
        time.sleep(1)
        GPIO.output(GPIO_TRIGGER, False)

        while (GPIO.input(GPIO_ECHO) == 0):
            start = time.time()
        while (GPIO.input(GPIO_ECHO) == 1):
            stop = time.time()

        elapsed = stop – start
        distance = (elapsed * 34300)
        distance = distance / 2

        print(‘Distance: %.1f’, distance)
        time.sleep(1)

except KeyboardInterrupt:
    GPIO.cleanup()
```

30) Write a program to rotate the Servo Motor 0 degrees, 90 degrees, 180 degrees and 270 degrees continuously with Raspberry Pi

VCC	5V
GND	GND
OUTPUT	RPi22

PROGRAM:

```
import time
import RPi.GPIO as GPIO

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(22, GPIO.OUT)

p = GPIO.PWM(22, 50)
p.start(5)

try:
        while True:
                p.ChangeDutyCycle(2.5)
                time.sleep(1)
                p.ChangeDutyCycle(7.5)
                time.sleep(1)
                p.ChangeDutyCycle(12.5)
                time.sleep(1)

except KeyboardInterrupt:
        p.stop()
        GPIO.cleanup()
```

31) Write a python program for controlling LED’s connected to rasberrypi with mobile after establishing connection between Pi and mobile through RasberryPi.

PROGRAM:

```
import bluetooth
import time

import sys
sys.path.append(‘/home/pi/Adafruit-Raspberry-Pi-Python-Code-legacy/Adafruit_MCP230xx’)

from Adafruit_MCP230XX import Adafruit_MCP230XX
mcp = Adafruit_MCP230XX(busnum = 1, address = 0x20, num_gpios = 16)
mcp.config(0, mcp.OUTPUT)

server_sock = bluetooth.BluetoothSocket(bluetooth.RFCOMM)
port = 1
server_sock.bind((“”, port))
server_sock.listen(1)

client_sock, address = server_sock.accept()
print(‘Accepted Connection from: ‘, address)

while True:
        data = client_sock.recv(1024)
        print(‘Received: [%s]’, data)
        time.sleep(1)

        if (data == ‘on’):
                mcp.output(0, 1)
                time.sleep(1)
        elif (data == ‘off’):
                mcp.output(0, 0)
                time.sleep(1)

client_sock.close()
server_sock.close()
```

32. Write a python program for controlling servo motor connected to rasberrypi with mobile after establishing connection between Pi and mobile through RasberryPi.

PROGRAM:

```
import bluetooth
import time

import RPi.GPIO as GPIO

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(22, GPIO.OUT)

p = GPIO.PWM(22, 50)
p.start(5)

server_sock = bluetooth.BluetoothSocket(bluetooth.RFCOMM)
port = 1
server_sock.bind((“”, port))
server_sock.listen(1)

client_sock, address = server_sock.accept()
print(‘Accepted Connection from: ‘, address)

while True:
        data = client_sock.recv(1024)
        print(‘Received: [%s]’, data)
        time.sleep(1)

        if (data == ‘0’):
                p.ChangeDutyCycle(2.5)
        elif (data == ‘90’):
                p.ChangeDutyCycle(7.5)
        elif (data == ‘180’):
                p.ChangeDutyCycle(12.5)

client_sock.close()
server_sock.close()

p.stop()
GPIO.cleanup()
```

NOTE:

- In case of any servo motor program, the motor **doesn’t rotate** 0, 90 (or) 180 degrees it **moves** to the position of 0, 90 and 180 degrees.
- For example, if you are currently at 90 degrees position and put a message as 180 on your phone then it will move to 180 degrees (**rotates 90 degrees only)**. If you want a complete 180 degrees rotation move your motor to 0 degrees position by putting a message as 0 then put a message as 180.

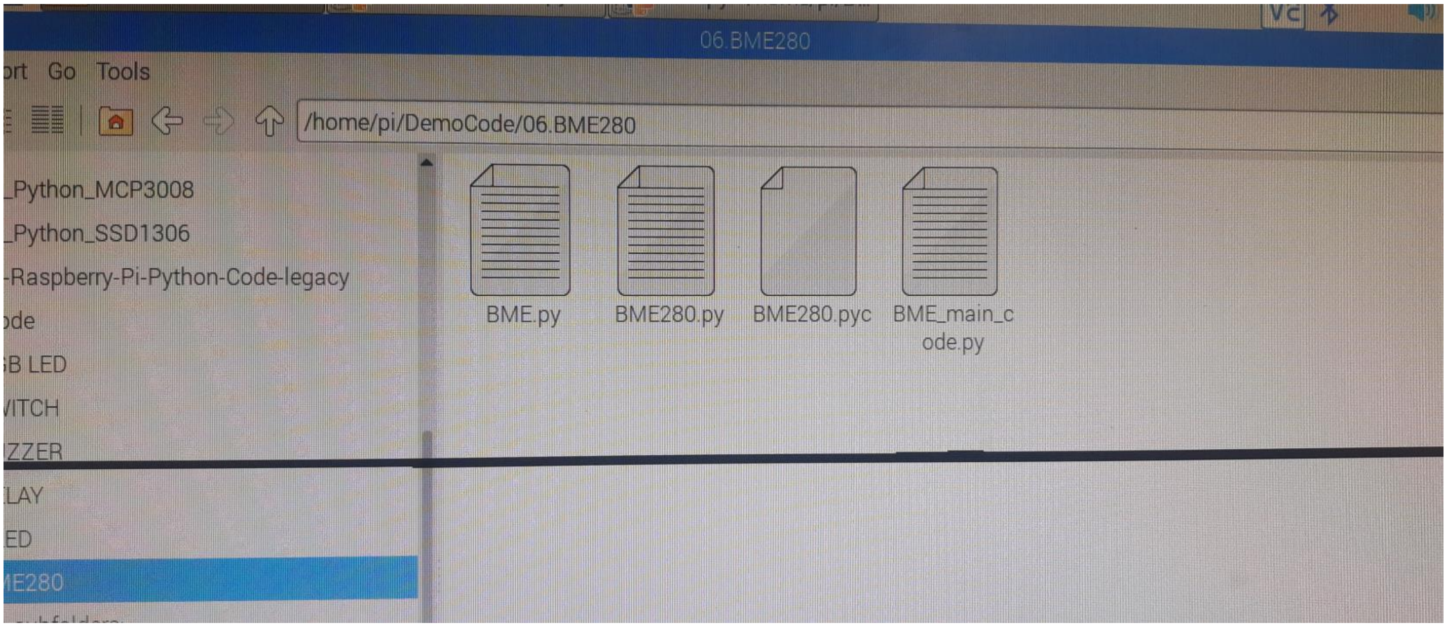
33 Write a python program for controlling buzzer connected to rasberrypi with mobile after establishing connection between Pi and mobile through RaspberryPi.

PROGRAM:

```
import bluetooth
import time

import sys
sys.path.append(‘/home/pi/Adafruit-Raspberry-Pi-Python-Code-legacy/Adafruit_MCP230xx’)

from Adafruit_MCP230XX import Adafruit_MCP230XX
mcp = Adafruit_MCP230XX(busnum = 1, address = 0x20, num_gpios = 16)
mcp.config(11, mcp.OUTPUT)

server_sock = bluetooth.BluetoothSocket(bluetooth.RFCOMM)
port = 1
server_sock.bind((“”, port))
server_sock.listen(1)

client_sock, address = server_sock.accept()
print(‘Accepted Connection from: ‘, address)

while True:
        data = client_sock.recv(1024)
        print(‘Received: [%s]’, data)
        time.sleep(1)

        if (data == ‘on’):
                mcp.output(11, 1)
                time.sleep(1)
        elif (data == ‘off’):
                mcp.output(11, 0)
                time.sleep(1)

client_sock.close()
server_sock.close()
```

34. Write a python program to switch on the buzzer when temperature increases beyond certain threshold using BME280 temperature sensor and Buzzer connected to RaspberryPi.

Note:



In case of BME280 program, create the code inside the folder of **06.BME280** in this case.

Do not change any codes of BME280.py and BME280.pyc which have some library functions that we will use in our program.

Create a file with name **BME.py** and staring writing your code in it.

There is **no** specific external sensor for BME280, it is a **built-in** sensor inside the RaspberryPi BOARD.

PROGRAM:

```
import time

import sys
sys.path.append(‘/home/pi/Onboard_sensor’)
import BME280 as BME

sys.path.append(‘/home/pi/Adafruit-Raspberry-Pi-Python-Code-legacy/Adafruit_MCP230xx’)

from Adafruit_MCP230XX import Adafruit_MCP230XX
mcp = Adafruit_MCP230XX(busnum = 1, address = 0x20, num_gpios = 16)
mcp.config(11, mcp.OUTPUT)

while True:
    (chip_id, chip_version) = BME.readBME2380ID()
    print(‘Chip ID: ’, chip_id)
    print(‘Chip Version: ’, chip_version)

    temperature, pressure, humidity = BME.readBME280All()
    print(‘Temperature: ‘, temperature, “C”)

    if (temperature >= 50):
        mcp.output(11, 1)
        time.sleep(1)
    else:
        mcp.output(11, 0)
        time.sleep(1)
```

36. Write the following Assembly programs for 8051

a) Create a square wave of 50% duty cycle on bit 0 of port 1

b) Create a square wave of 66% duty cycle on bit 3 of port 1

NOTE: use hard ware delays

<u>50% DUTY CYCLE</u>

```
L1:  SETB P1.3
     ACALL DELAY
     CLR P1.3
     ACALL DELAY
               SJMP L1
DELAY:
              MOV TMOD, #20H
              MOV TH1, #1AH
              SETB TR1
              HERE: JNB TF1, HERE
              CLR TR1
              CLR TF1
              RET

END
```

<u>66% DUTY CYCLE</u>

```
L1:  SETB P1.3
     ACALL DELAY
     ACALL DELAY
     CLR P1.3
     ACALL DELAY
               SJMP L1
DELAY:
              MOV TMOD, #20H
              MOV TH1, #1AH
              SETB TR1
              HERE: JNB TF1, HERE
              CLR TR1
              CLR TF1
              RET

END
```

37. Write a python program to rotate the servo motor 180 degrees from 0 degrees when intruder detected through PIR sensor

PROGRAM:

```
import time
import RPi.GPIO as GPIO

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(16, GPIO.IN)
GPIO.setup(22, GPIO.OUT)

p = GPIO.PWM(22, 50)
p.start(5)

try:
    while True:
        i = mcp.input(16)
        if (i == 1):
            print(‘Person Detected’)
            p.ChangeDutyCycle(12.5)
            time.sleep(1)
            p.ChangeDutyCycle(2.5)
            time.sleep(1)
        elif (i == 0):
            print(‘Person NOT Detected’)
            time.sleep(1)

except KeyboardInterrupt:
    p.stop()
    GPIO.cleanup()
```

38. Write a python program to rotate   the servo motor 270 degrees from 0 degrees when intruder detected through ultrasonic sensor at certain distance.

PROGRAM:

```
import time
import RPi.GPIO as GPIO

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(22, GPIO.OUT)

p = GPIO.PWM(22, 50)
p.start(5)

GPIO_TRIGGER = 16
GPIO_ECHO = 18

GPIO.setup(GPIO_TRIGGER, GPIO.OUT)
GPIO.setup(GPIO_ECHO, GPIO.IN)

GPIO.output(GPIO_TRIGGER, False)
time.sleep(0.5)

try:
        while True:
                GPIO.output(GPIO_TRIGGER, True)
                time.sleep(1)
                GPIO.output(GPIO_TRIGGER, False)

                while (GPIO.input(GPIO_ECHO) == 0):
                        start = time.time()
                while (GPIO.input(GPIO_ECHO) == 1):
                        stop = time.time()

                elapsed = stop – start
                distance = (elapsed * 34300)
                distance = distance / 2

                if (distance < 50):
                        p.ChangeDutyCycle(17.5)
                        time.sleep(1)
                        p.ChangeDutyCycle(2.5)
                        time.sleep(1)


except KeyboardInterrupt:
        p.stop()
        GPIO.cleanup()
```