CSI THOOTHUKUDI-NAZARETH DIOCESE

DR.G.U.POPE COLLEGE OF ENGINEERING

POPE NAGAR SAWYERPURAM-628 251
Thoothukudi District

Register Number :

Certified that this is the bonafide record of work done by Selvan/Selvi ……………………………………………………………………………… of ………. Semester ……….. branch for the lab ………………………………………………………… During the year…………………

Staff In-charge **H. O.D**

Submitted for the university practical Examination held on …………..

Internal Examiner **External Examiner**

Ex no: 1

Write addition and Multiplication operation using Embedded c.

Aim:

To write a basic and arithmetic programs to perform addition, multiplication of 8 bit numbers in Embedded C using Keil.

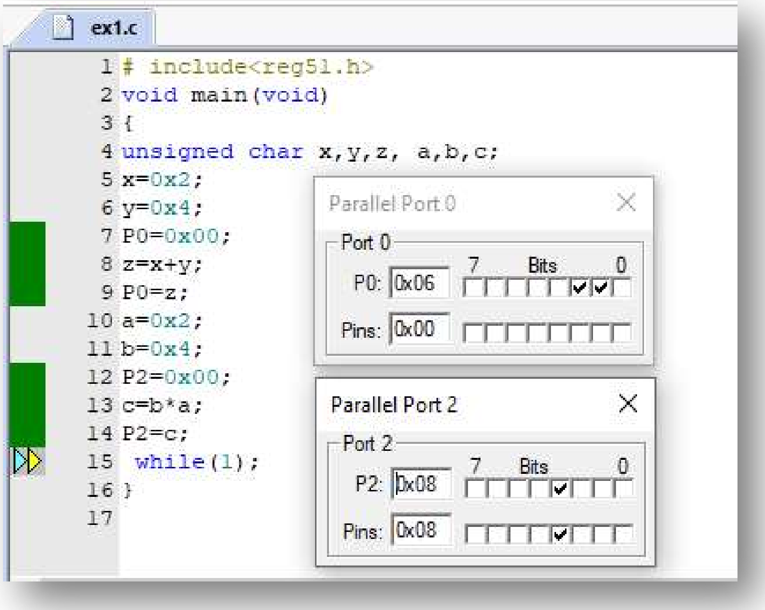
Algorithm:

Step 1: Assign two variables for getting the inputs
Step 2: Perform arithmetic operations
Step 3: Initialize the port value with 0x00
Step 4: Display the output using Ports of 8051.
Step 5: End the program

Program:

```
# include<reg51.h>
void main(void)
{
unsigned char x,y,z, a,b,c;
x=0x2;
y=0x4;
P0=0x00;
z=x+y;
P0=z;
a=0x2;
b=0x4;
P2=0x00;
c=b*a;
P2=c;
 while(1);
}
```

Output:



```
# include<reg51.h>
void main(void)
{
unsigned char x,y,z, a,b,c;
x=0x2;
y=0x4;
P0=0x00;
z=x+y;
P0=z;
a=0x2;
b=0x4;
P2=0x00;
c=b*a;
P2=c;
 while(1);
}
```

Parallel Port 0

Port 0

7 Bits 0

P0: 0x06 ☐☐☐☐☐☑☑☐

Pins: 0x00 ☐☐☐☐☐☐☐☐

Parallel Port 2

Port 2

7 Bits 0

P2: 0x08 ☐☐☐☐☑☐☐☐

Pins: 0x08 ☐☐☐☐☑☐☐☐

Result:

Thus the program was executed and the output was verified successfully.

Ex no: 2

Write Subtraction and Division operation using embedded c.

Aim:

To write a basic and arithmetic programs to perform subtraction, division of 8 bit numbers in Embedded C using Keil.

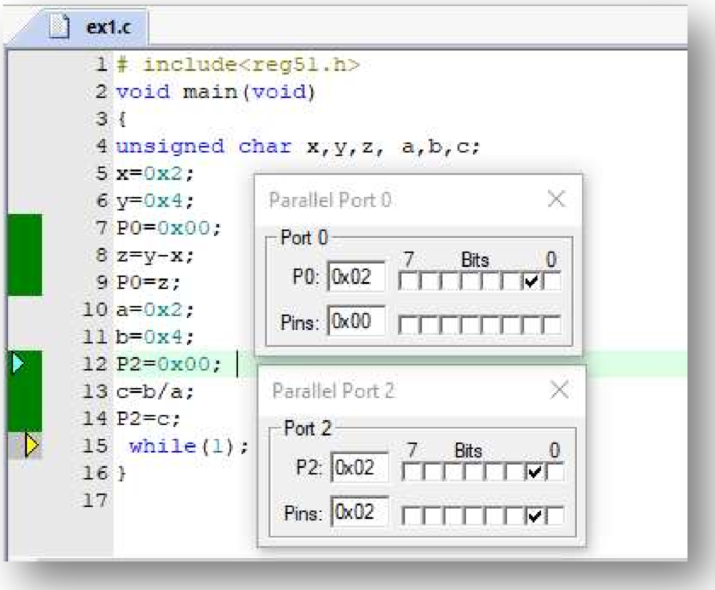
Algorithm:

Step 1: Assign two variables for getting the inputs
Step 2: Perform arithmetic operations
Step 3: Initialize the port value with 0x00
Step 4: Display the output using Ports of 8051.
Step 5: End the program

Program:

```
# include<reg51.h>
void main(void)
{
unsigned char x,y,z, a,b,c;
x=0x2;
y=0x4;
P0=0x00;
z=y-x;
P0=z;
a=0x2;
b=0x4;
P2=0x00;
c=b/a;
P2=c;
 while(1);
}
```

Output:



ex1.c

```
# include<reg51.h>
void main(void)
{
unsigned char x,y,z, a,b,c;
x=0x2;
y=0x4;
P0=0x00;
z=y-x;
P0=z;
a=0x2;
b=0x4;
P2=0x00;
c=b/a;
P2=c;
 while(1);
}
```

Parallel Port 0

Port 0

P0: 0x02 — Bits 7 … 0: □□□□□□☑□

Pins: 0x00 — □□□□□□□□

Parallel Port 2

Port 2

P2: 0x02 — Bits 7 … 0: □□□□□□☑□

Pins: 0x02 — □□□□□□☑□

Result: Thus the program was executed and the output was verified successfully.

Exno:3

Write a program to perform logical operations using embedded C.

Aim:

To write an embedded c Program to perform logical operations in 8051 microcontroller using Keil

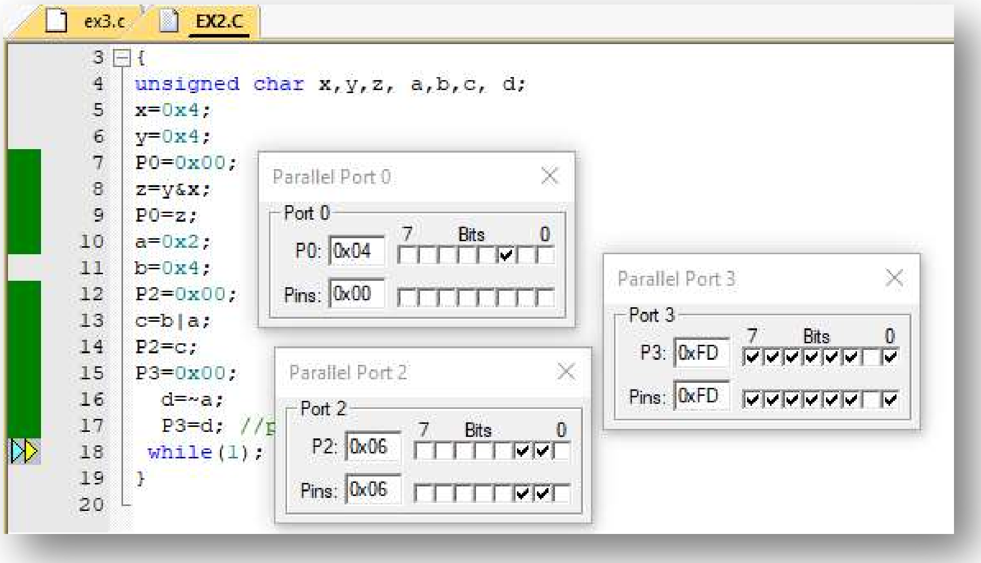
Algorithm:

Step 1: Initialize the starting address of the memory address.
Step 2: Load the port 0,2,3
Step 3: perform Logical operation such as AND,OR,NOT.
Step 4: End the program

Program:

```
# include<reg51.h>
void main(void)
{
unsigned char x,y,z, a,b,c, d;
x=0x4;
y=0x4;
P0=0x00;
z=y&x;
P0=z;
a=0x2;
b=0x4;
P2=0x00;
c=b|a;
P2=c;
P3=0x00;
      d=~a;
      P3=d; //port 3 is caps (P3)
 while(1);
}
```

Output:



```
{
unsigned char x,y,z, a,b,c, d;
x=0x4;
y=0x4;
P0=0x00;
z=y&x;
P0=z;
a=0x2;
b=0x4;
P2=0x00;
c=b|a;
P2=c;
P3=0x00;
  d=~a;
  P3=d; //p
 while(1);
}
```

Parallel Port 0
Port 0
P0: 0x04 (Bits 7…0: □ □ □ □ □ ☑ □ □)
Pins: 0x00 (Bits 7…0: □ □ □ □ □ □ □ □)

Parallel Port 2
Port 2
P2: 0x06 (Bits 7…0: □ □ □ □ □ ☑ ☑ □)
Pins: 0x06 (Bits 7…0: □ □ □ □ □ ☑ ☑ □)

Parallel Port 3
Port 3
P3: 0xFD (Bits 7…0: ☑ ☑ ☑ ☑ ☑ ☑ □ ☑)
Pins: 0xFD (Bits 7…0: ☑ ☑ ☑ ☑ ☑ ☑ □ ☑)

Result:

Thus the program was executed and the output was verified successfully

Exno: 4

Write a program to perform ALU operations using embedded C.

Aim:

To write an embedded c Program to perform ALU operations in 8051 microcontroller using Keil

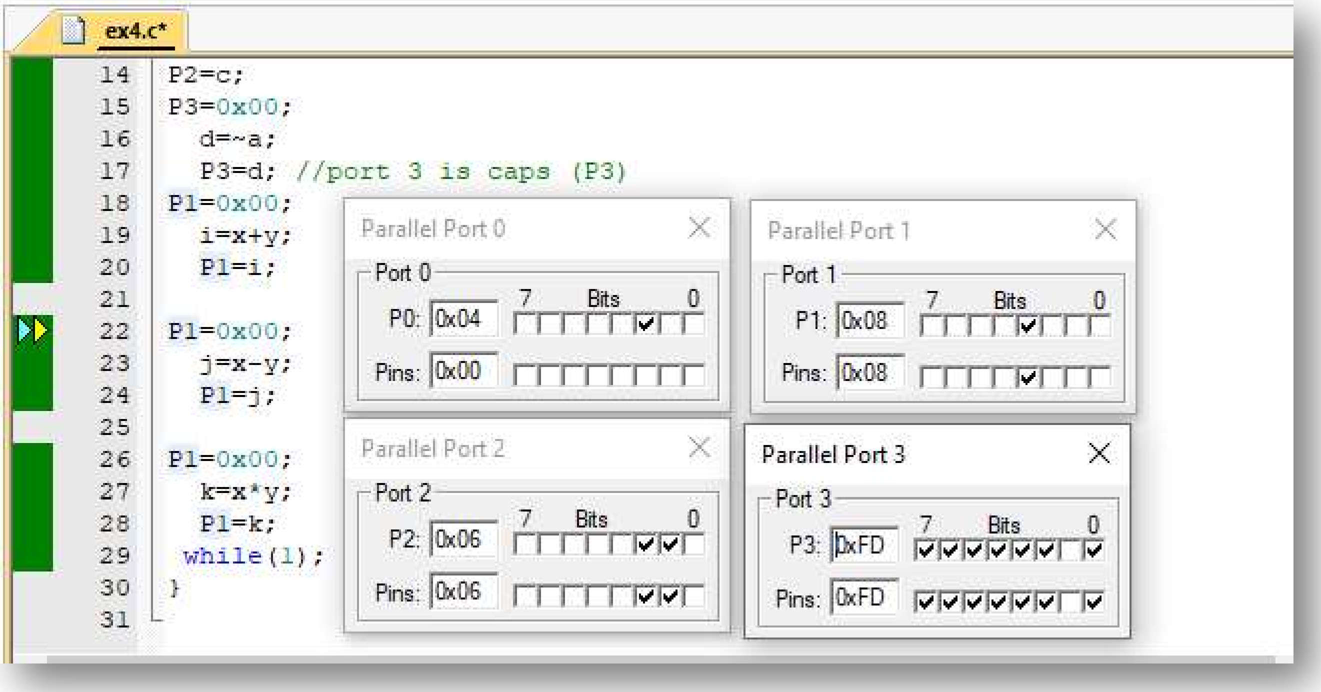
Algorithm:

Step 1: Initialize the starting address of the memory address.
Step 2: Load the port 0,1,2,3
Step 3: Perform addition, subtraction, multiplication, division with the help of mnemonics
Step 4: Also perform Logical operation such as AND,OR,NOT.
Step 5: End the program.

Program:

```
# include<reg51.h>
void main(void)
{
unsigned char x,y,z, a,b,c, d ,i,j,k;
x=0x4;
y=0x4;
P0=0x00;
z=y&x;
P0=z;

a=0x2;
b=0x4;
P2=0x00;
c=b|a;
P2=c;

P3=0x00;
      d=~a;
      P3=d; //port 3 is caps (P3)

P1=0x00;
      i=x+y;
      P1=i;

P1=0x00;
      j=x-y;
      P1=j;

P1=0x00;
      k=x*y;
      P1=k;


 while(1);
}
```

Output:



```
P2=c;
P3=0x00;
  d=~a;
  P3=d; //port 3 is caps (P3)
P1=0x00;
  i=x+y;
  P1=i;

P1=0x00;
  j=x-y;
  P1=j;

P1=0x00;
  k=x*y;
  P1=k;
 while(1);
}
```

Parallel Port 0

Port 0 — 7 Bits 0

P0: 0x04 □□□□□☑□□

Pins: 0x00 □□□□□□□□

Parallel Port 1

Port 1 — 7 Bits 0

P1: 0x08 □□□□☑□□□

Pins: 0x08 □□□□☑□□□

Parallel Port 2

Port 2 — 7 Bits 0

P2: 0x06 □□□□□☑☑□

Pins: 0x06 □□□□□☑☑□

Parallel Port 3

Port 3 — 7 Bits 0

P3: 0xFD ☑☑☑☑☑☑□☑

Pins: 0xFD ☑☑☑☑☑☑□☑

Result:

Thus the program was executed and the output was verified successfully.

Ex no: 5

Write a program to perform comparison operations using Embedded C.

Aim:

To write an embedded c Program to perform comparison operations in 8051 microcontroller using Keil

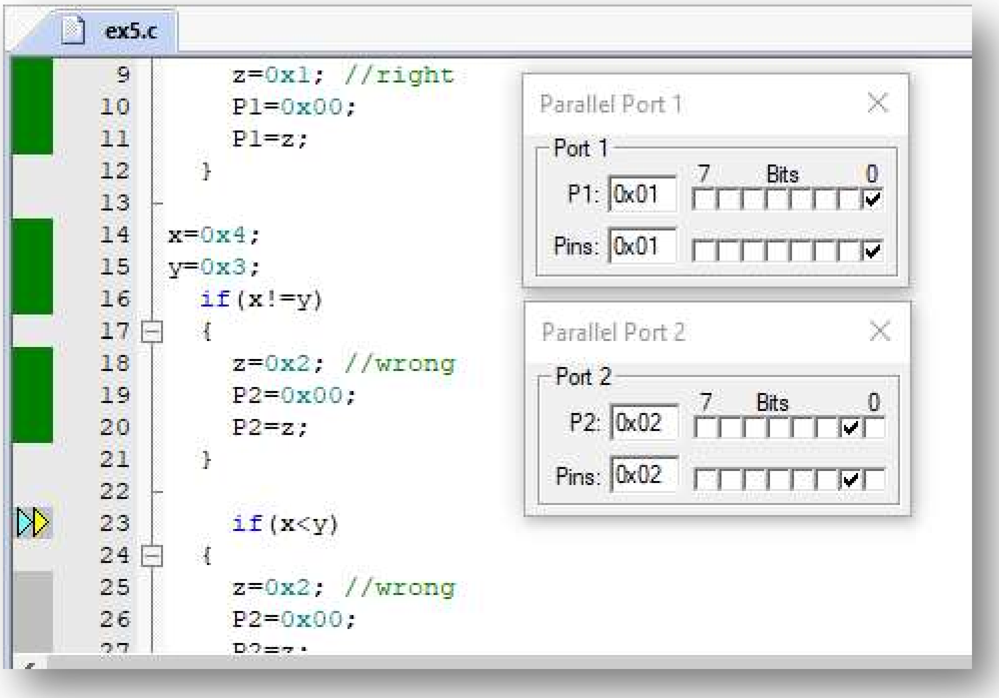
Algorithm:

Step 1: Initialize the starting address of the memory address.
Step 2: Load the registers a,b
Step 3: Perform comparison operation such as ==,!=,>=,<=,<,>.
Step 4: End the program.

Program:

```
# include<reg51.h>
void main(void)
{
unsigned char x,y,z;
x=0x4;
y=0x4;
      if(x==y)
      {
        z=0x1; //right
            P1=0x00;
            P1=z;
      }

x=0x4;
y=0x3;
      if(x!=y)
      {
        z=0x2; //wrong
            P2=0x00;
            P2=z;
      }

            if(x<y)
      {
        z=0x2; //wrong
            P2=0x00;
            P2=z;
      }

            if(x>y)
      {
        z=0x1; //right
            P1=0x00;
            P1=z;
      }

 while(1);
}
```

Output:



ex5.c

```
    z=0x1; //right
    P1=0x00;
    P1=z;
  }

x=0x4;
y=0x3;
  if(x!=y)
  {
    z=0x2; //wrong
    P2=0x00;
    P2=z;
  }

    if(x<y)
  {
    z=0x2; //wrong
    P2=0x00;
    P2=z;
```

Parallel Port 1

Port 1

P1: 0x01 7 Bits 0 □□□□□□□☑

Pins: 0x01 □□□□□□□☑

Parallel Port 2

Port 2

P2: 0x02 7 Bits 0 □□□□□□☑□

Pins: 0x02 □□□□□□☑□

Result:

Thus the program was executed and the output was verified successfully.

Ex no: 6

Write a program to transfer the data between two registers.

Aim:

To write an Assembly Program to perform data transfer operations in 8051 microcontroller using Keil.

Algorithm:

Step 1: Initialize the starting address.

Step 2: Load the registers A,B

Step 3:Initialize the R0

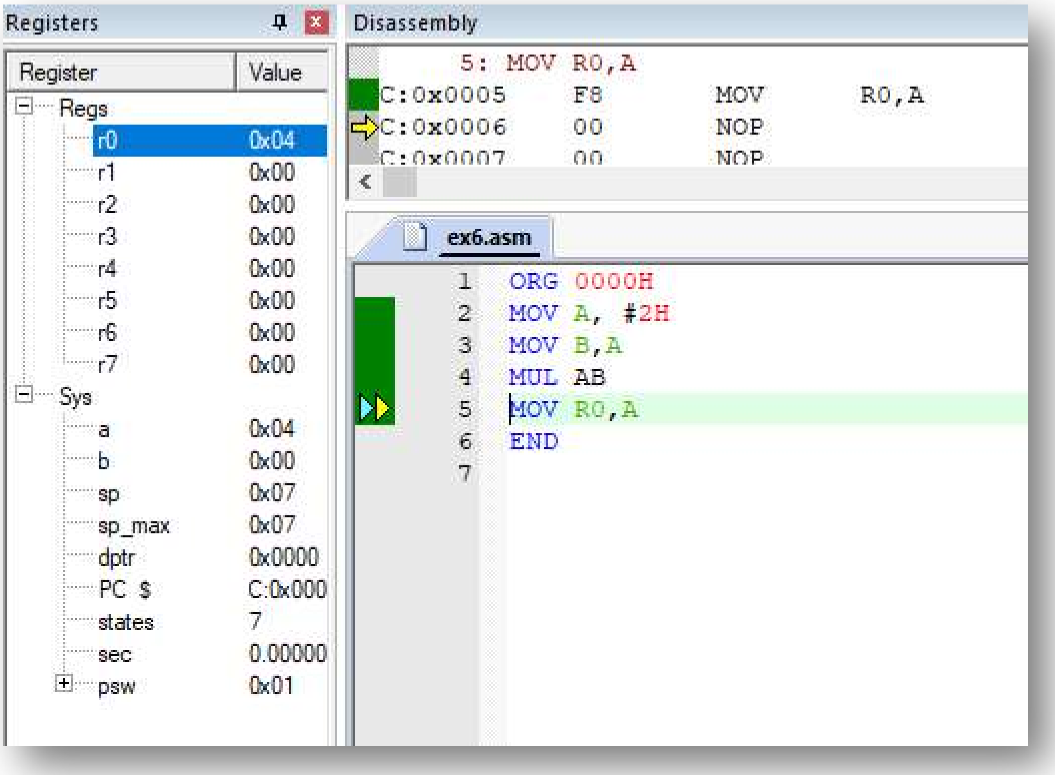
Step 4: MOV is used to move the data from A to R0 register.

Step 5: End the program.

Program:

```
ORG 0000H
MOV A, #2H
MOV B,A
MUL AB
MOV R0,A
END
```

Output:



Register	Value
Regs	
r0	0x04
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x04
b	0x00
sp	0x07
sp_max	0x07
dptr	0x0000
PC $	C:0x000
states	7
sec	0.00000
psw	0x01

```
     5: MOV R0,A
C:0x0005    F8       MOV      R0,A
C:0x0006    00       NOP
C:0x0007    00       NOP
```

```
ORG 0000H
MOV A, #2H
MOV B,A
MUL AB
MOV R0,A
END
```

Result:

Thus the program was executed and the output was verified successfully.

Exno: 7

Write a program to transfer data between memory and register.

Aim:

To write an Assembly Language Program to transfer the data between internal RAM to External memory of 8051 microcontroller using Keil.

Alogorithm:

Step 1: Initialize the starting address of the memory address.

Step 2: Load the registers R0,R1

Step 3:Initialize the external memory location

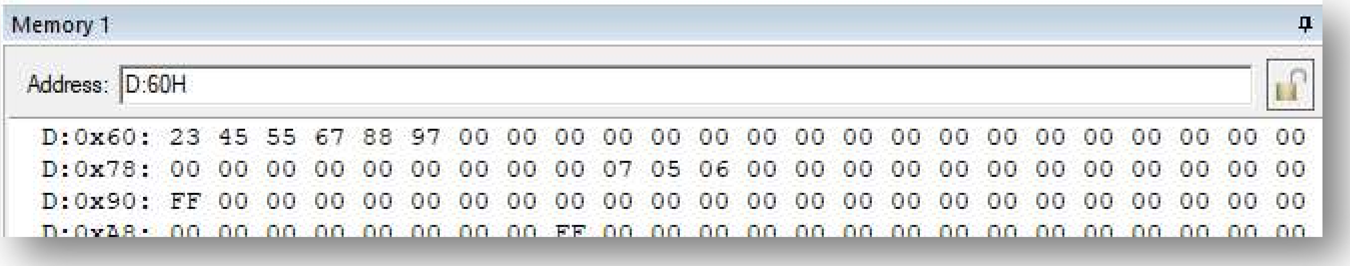
Step 4: MOVX is used to move the data from Internal memory to external memory and vice-versa.

Step 5: End the program

Program:(a)

```
//ram memory is 00h -7fh(128 bytes)
//rom memory is 0000h -0fffh(4kbytes)
//transfering data from rom -600h to into 605h into ram 60h to 65h

org 0000h
mov dptr,#600h //rom memory intial address
mov r0,#60h    //ram memory intial address
mov r7,#06     //count
back:movc a,@a+dptr
mov @r0, a
inc dptr
inc r0
clr a
djnz r7, back
org 600h
db 23h,45h,55h,67h,88h,97h
end
```

Output:(a)



Memory 1

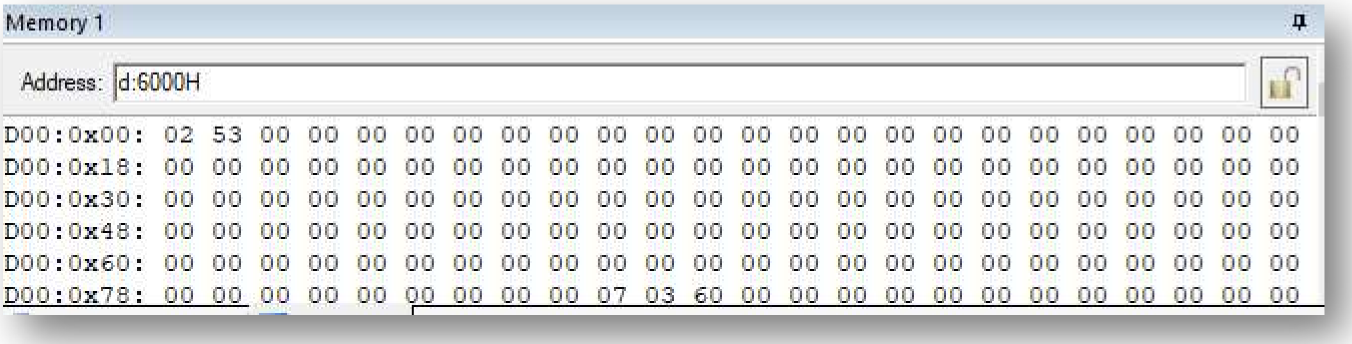
Address: D:60H

```
D:0x60: 23 45 55 67 88 97 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D:0x78: 00 00 00 00 00 00 00 00 00 07 05 06 00 00 00 00 00 00 00 00 00 00 00 00
D:0x90: FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D:0xA8: 00 00 00 00 00 00 00 00 FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Program:(b)

```
MOV   R0,#05H
MOV R1,#50H
MOV DPTR,#6000H
L1:MOV A,@R1
MOVX @DPTR,A
INC R1
INC DPL
DJNZ R0,L1
END
```

Output:(b)



Address: d:6000H

```
D00:0x00: 02 53 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D00:0x18: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D00:0x30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D00:0x48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D00:0x60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D00:0x78: 00 00 00 00 00 00 00 00 00 07 03 60 00 00 00 00 00 00 00 00 00 00 00 00
```

Result:

Thus the program was executed and the output was verified successfully

Exno: 8

Write a program to perform arithmetic operations in 8051 using simulator.

Aim:

Assembly Language Program to perform Arithmetic operations in 8051 microcontroller using Keil.

Algorithm:

Step 1: Initialize the starting address of the memory address.

Step 2: Load the registers a,b

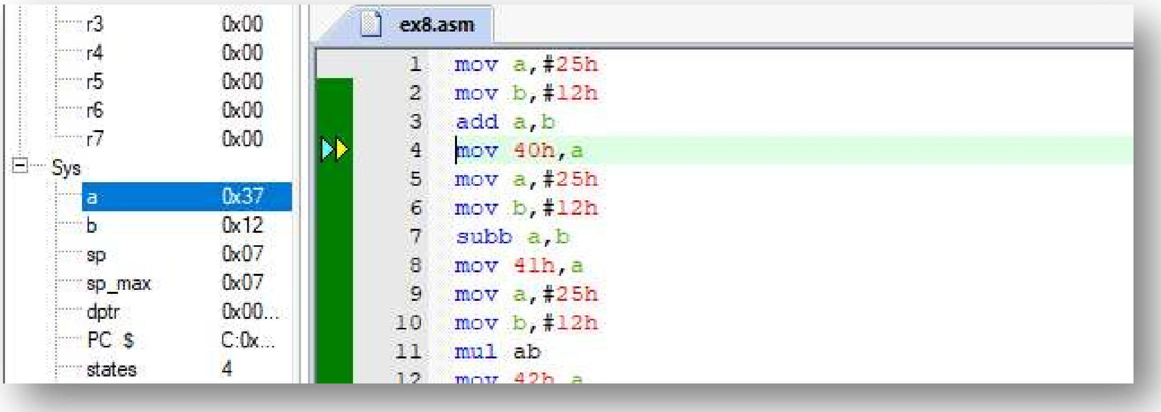
Step 3: Perform addition, subtraction, multiplication, division

Step 4: End the program

Program:

```
mov a,#25h
mov b,#12h
add a,b
mov 40h,a
mov a,#25h
mov b,#12h
subb a,b
mov 41h,a
mov a,#25h
mov b,#12h
mul ab
mov 42h,a
mov 43h,b
mov a,#25h
mov b,#12h
div ab
mov 44h,a
mov 45h,b
mov a,#25h
inc a
mov 46h,a
dec a
end
```

Output:



```
r3        0x00
r4        0x00
r5        0x00
r6        0x00
r7        0x00
Sys
a         0x37
b         0x12
sp        0x07
sp_max    0x07
dptr      0x00...
PC  $     C:0x...
states    4
```

ex8.asm

```
mov a,#25h
mov b,#12h
add a,b
mov 40h,a
mov a,#25h
mov b,#12h
subb a,b
mov 41h,a
mov a,#25h
mov b,#12h
mul ab
mov 42h,a
```

Result: Thus the program was executed and the output was verified successfully.

Ex no: 9

Write a program to compare two numbers in 8051 using simulator.

Aim:

To write an Assembly Language Program to find the largest number using keil.

Algorithm:

Step 1: Start the program with the memory address 0000H

Step 2: Initialize the memory pointer r0=20h and counter r1=05h

Step 3: Initialize the register r2=00h for higher byte result

Step 4: Clear accumulator and add two numbers.

Step 5: Compare the content of memory addressed by the register.

Step 6: If Carry = 0, go to step 8 or if Carry = 1 go to step 7.

Step 7: Move the content of memory addressed by the program.

Step 8: Decrement the count.

Step 9: Check for Zero with JNZ condition

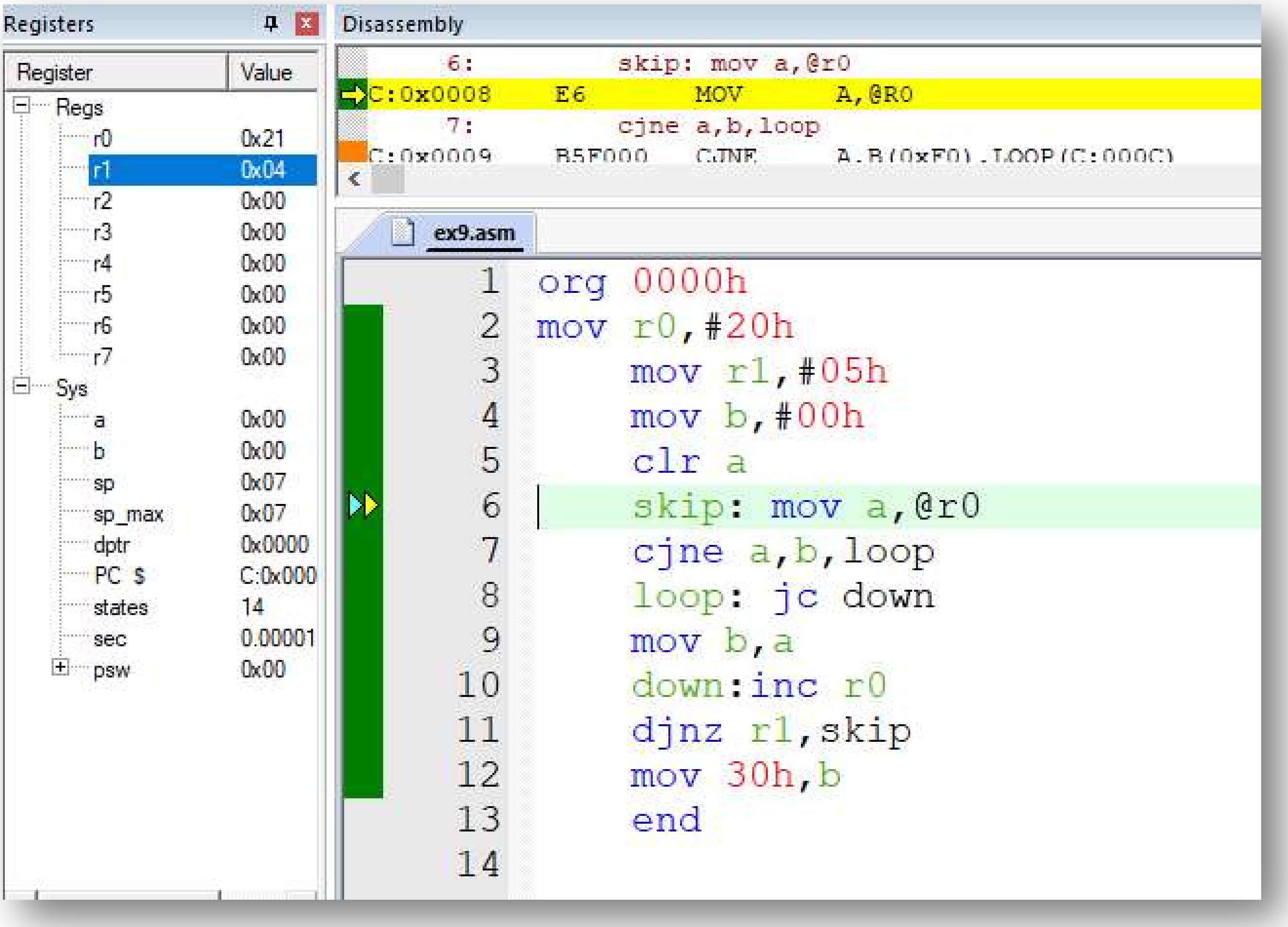
Step 10: Store the largest data in memory.

Step 11: End the program.

Program:

```
org 0000h
mov r0,#20h
      mov r1,#05h
      mov b,#00h
      clr a
      skip: mov a,@r0
      cjne a,b,loop
      loop: jc down
      mov b,a
      down:inc r0
      djnz r1,skip
      mov 30h,b
      end
```

Output:



Register	Value
Regs	
r0	0x21
r1	0x04
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x00
b	0x00
sp	0x07
sp_max	0x07
dptr	0x0000
PC $	C:0x000
states	14
sec	0.00001
psw	0x00

```
     6:          skip: mov a,@r0
C:0x0008    E6        MOV      A,@R0
     7:          cjne a,b,loop
C:0x0009    B5F000    CJNE     A,B(0xF0),LOOP(C:000C)
```

```
org 0000h
mov r0,#20h
    mov r1,#05h
    mov b,#00h
    clr a
    skip: mov a,@r0
    cjne a,b,loop
    loop: jc down
    mov b,a
    down:inc r0
    djnz r1,skip
    mov 30h,b
    end
```

Result:

Thus the program was executed and the output was verified successfully.

Exno: 10

Explain Arduino platform and programming with sample programs.

Aim:

To write the Arduino Program to blink single LED in the Proteus simulator

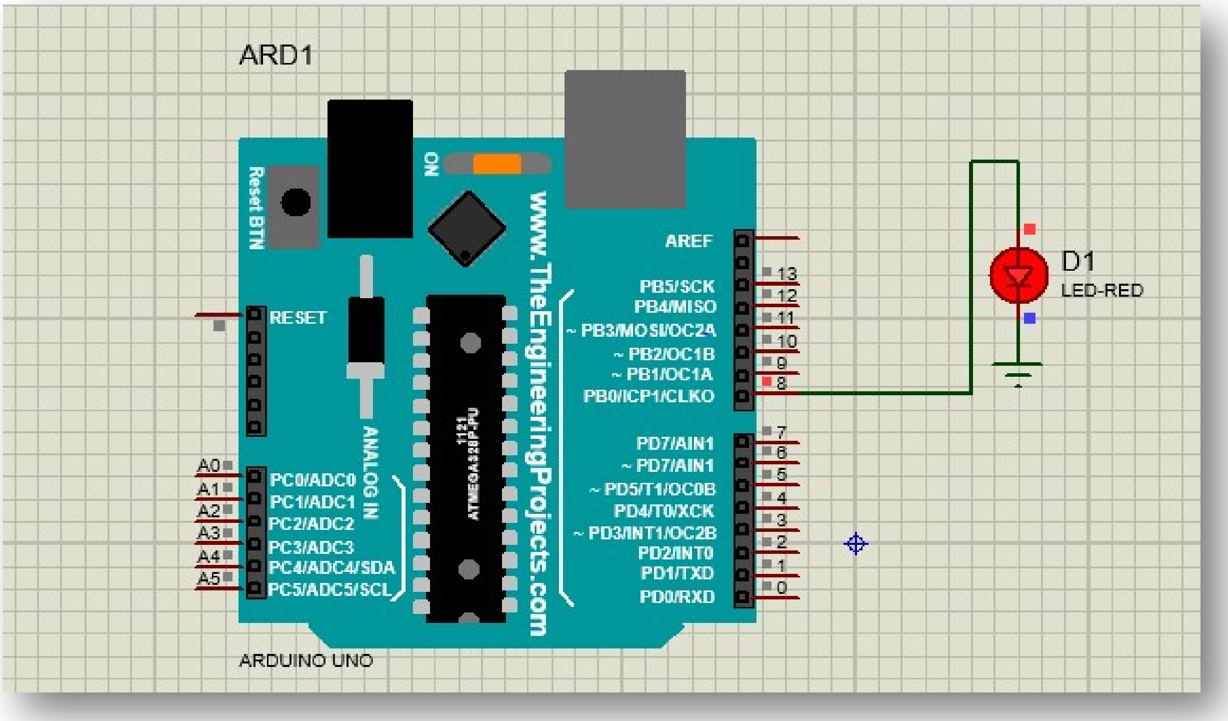
Algorithm:

Step 1: Open the Proteus simulator and make the circuit for blinking the single LED.
Step 2: Connect the ternminals according to the circuit
Step 3: Type and Execute the program
Step 4: View that the LED is blinked

Program:

```
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(8, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(8, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(8, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Output:



Result:

Thus the program was executed and the output was verified successfully.

Exno: 11

Write a python programming to perform data transfer operation in arduino.

Aim:

To write the Arduino Program to perform data transfer operation in ESP32 using wokwi simulator

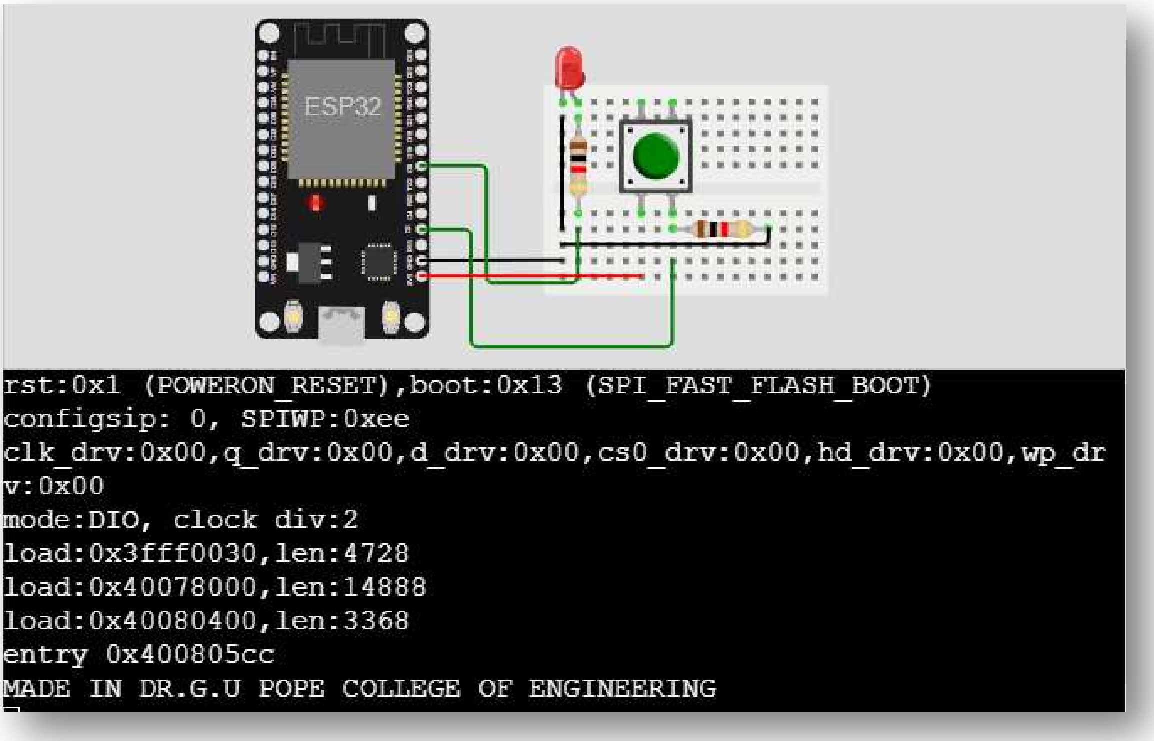
Algorithm:

Step 1: import libraries
Step 2: pin initialization
Step 3: print a data
Step 4: main loop for data transfer
Step 5: create Delay

Program:

```
from machine import Pin
from time import sleep

led=Pin(5,Pin.OUT)
pushbutton=Pin(2,Pin.IN)
print('MADE IN DR.G.U POPE COLLEGE OF ENGINEERING')
while True:

      led.value(pushbutton.value())
      sleep(0.5)
```

Output:



```
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_dr
v:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:4728
load:0x40078000,len:14888
load:0x40080400,len:3368
entry 0x400805cc
MADE IN DR.G.U POPE COLLEGE OF ENGINEERING
```

Result: Thus the program was executed and the output was verified successfully.

Exno: 12

Write a program to interfacing any one sensor with Raspberry PI.

Aim:

To write the python Program to interfacing any one sensor with Raspberry PI using wokwi simulator .

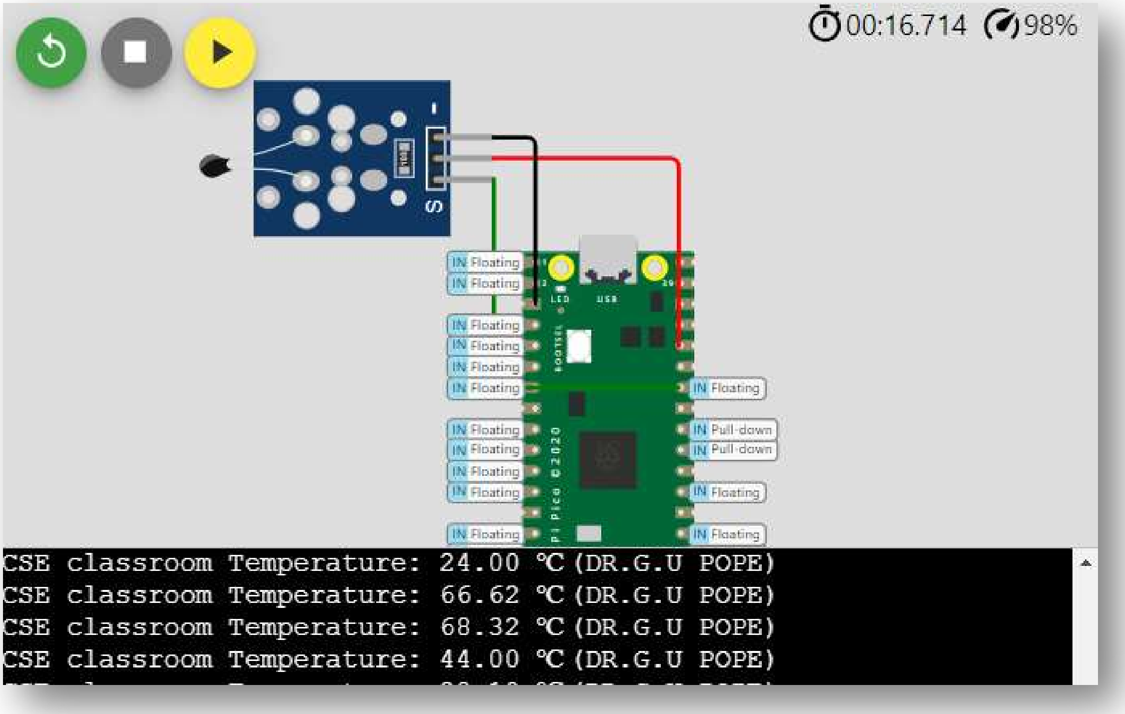
Algorithm:

```
Step 1: import libraries (machine, math, utime)
Step 2: constants declaration
Step 3: initialize the ADC on pin to read analog values
Step 4: temperature measurement loop
Step 5: create Delay
```

Program:

```python
import machine
import math
import utime
BETA = 3950
R1 = 10000
V_REF = 3.3
adc = machine.ADC(machine.Pin(28))
while True:
    analog_value = adc.read_u16()
    voltage = (analog_value / 65535) * V_REF
    resistance = R1 / ((V_REF / voltage) - 1)
    kelvin = 1 / (1 / 298.15 + 1 / BETA * math.log(resistance / 10000))
    celsius = kelvin - 273.15
    print("CSE classroom Temperature: {:.2f} ℃ (DR.G.U POPE)".format(celsius))
    utime.sleep(1)
```

Output:



```
CSE classroom Temperature: 24.00 ℃ (DR.G.U POPE)
CSE classroom Temperature: 66.62 ℃ (DR.G.U POPE)
CSE classroom Temperature: 68.32 ℃ (DR.G.U POPE)
CSE classroom Temperature: 44.00 ℃ (DR.G.U POPE)
```

Result: Thus the program was executed and the output was verified successfully.

Exno: 13

Design the smart city application based on IoT.

Aim:

To write the Program create smart parking management system using wokwi simulator .

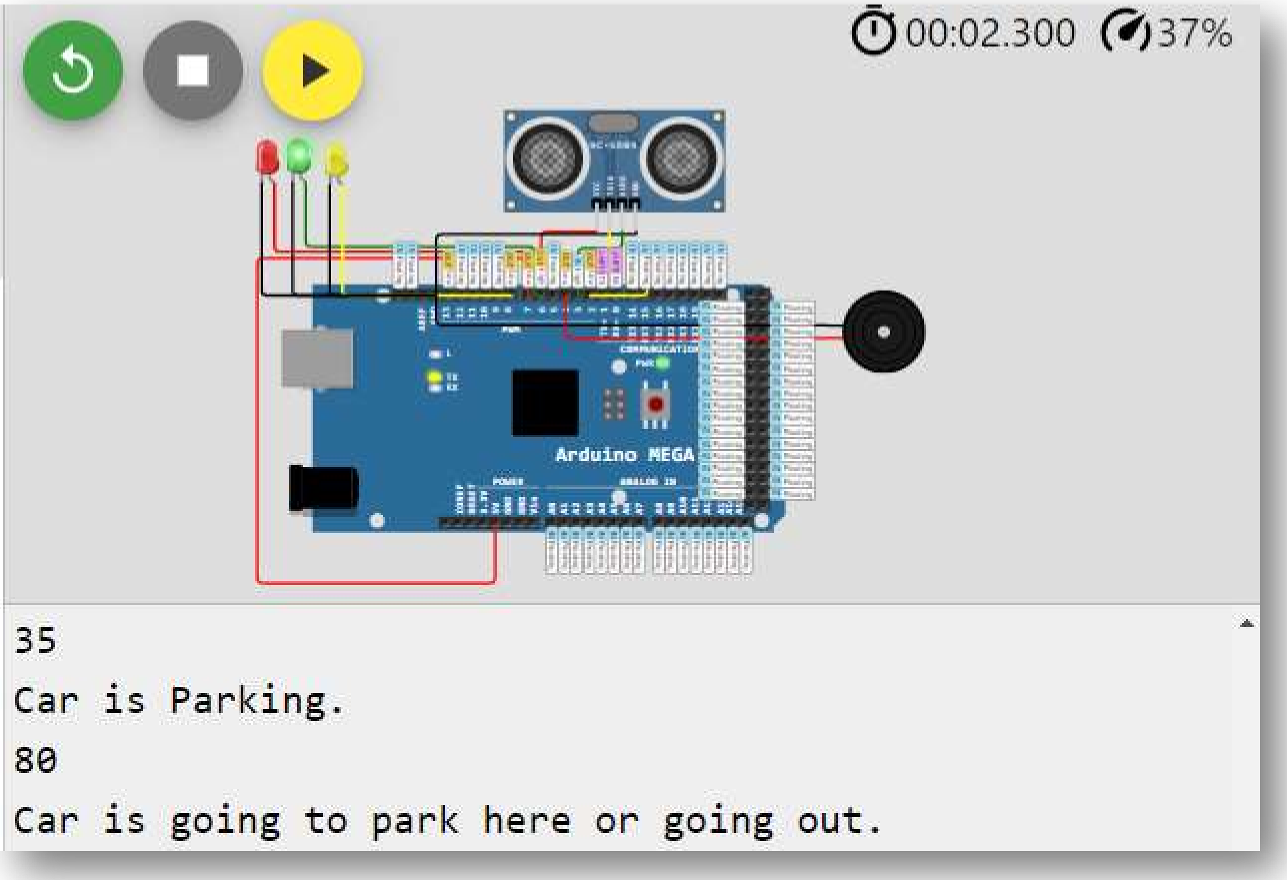
Algorithm:

1. **Setup:**
    - Initialize serial communication with a baud rate of 115200.
    - Set pin modes:
    - trigPin as OUTPUT for triggering the ultrasonic sensor.
    - echoPin as INPUT for receiving echo signals from the ultrasonic sensor.
    - LEDPin, LED_EMPTY, LED_FULL, LED_PENDING, and BUZZER as OUTPUT for controlling LEDs and buzzer.
2. **Main Loop:**
    - Continuously:
    - Call the ULT() function to measure the distance using the ultrasonic sensor.
    - Print the measured distance over the serial port for debugging purposes.
    - Based on the measured distance:
    - If the distance is greater than or equal to 200 cm, indicate an empty parking space by turning on the green LED (LED_EMPTY) and turning off other LEDs (LED_PENDING and LED_FULL).
    - If the distance is less than 200 cm but greater than or equal to 50 cm, indicate a pending parking action by turning on the yellow LED (LED_PENDING) and briefly activating the buzzer.
    - If the distance is less than 50 cm, indicate an occupied parking space by turning on the red LED (LED_FULL).
3. **ULT Function:**
    - Trigger the ultrasonic sensor by setting trigPin LOW for 2 microseconds, then HIGH for 10 microseconds, and then LOW again.
    - Measure the duration of the echo pulse using pulseIn() function, which captures the time elapsed between sending and receiving the ultrasonic pulse.
    - Calculate the distance in centimeters based on the duration of the echo pulse, considering the speed of sound.

Program:

```
#define echoPin 3 // Echo Pin
#define trigPin 2 // Trigger Pin
#define LEDPin 13 // Onboard LED
int LED_EMPTY = 6;
int LED_FULL = 7;
int LED_PENDING = 8;
int BUZZER = 4;
void ULT(void);
int maximumRange = 200; // Maximum range needed
int minimumRange = 0; // Minimum range needed
long duration, distance; // Duration used to calculate distance

void setup() {
  Serial.begin (115200);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(LEDPin, OUTPUT); // Use LED indicator (if required)
  pinMode(LED_EMPTY, OUTPUT);
  pinMode(LED_FULL, OUTPUT);
  pinMode(LED_PENDING, OUTPUT);
  pinMode(BUZZER, OUTPUT);
}

void loop() {
  ULT();
  Serial.println(distance);  //show distance

  /*vacant, out green light*/
  if(distance >= 200){
    digitalWrite(LED_EMPTY,1);
    digitalWrite(LED_PENDING,0);
    digitalWrite(LED_FULL,0);
    Serial.println("Empty Space.");
  }

  /*someone is parking, out yelow light*/
  else if(distance < 200 && distance >= 50){
    digitalWrite(LED_EMPTY,0);
    digitalWrite(LED_PENDING,1);
    digitalWrite(LED_FULL,0);
    tone(BUZZER, 800);
    delay(100);
    digitalWrite(LED_EMPTY,0);
    digitalWrite(LED_PENDING,0);
    digitalWrite(LED_FULL,0);
    noTone(BUZZER);
    delay(500);
    Serial.println("Car is going to park here or going out.");
  }

  /*occupied, out red light*/
    else{
    digitalWrite(LED_EMPTY,0);
    digitalWrite(LED_PENDING,0);
    digitalWrite(LED_FULL,1);
    Serial.println("Car is Parking.");
  }
}

void ULT(){
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  //Calculate the distance (in cm) based on the speed of sound.
  distance = duration / 58.2;
}
```

Output:



```
35
Car is Parking.
80
Car is going to park here or going out.
```

Result:

Thus the program was executed and the output was verified successfully.

Exno: 14

Write a program to setup cloud platform to log the data.

Aim:

To write the python Program to interfacing sensor with setup cloud platform to log the data.

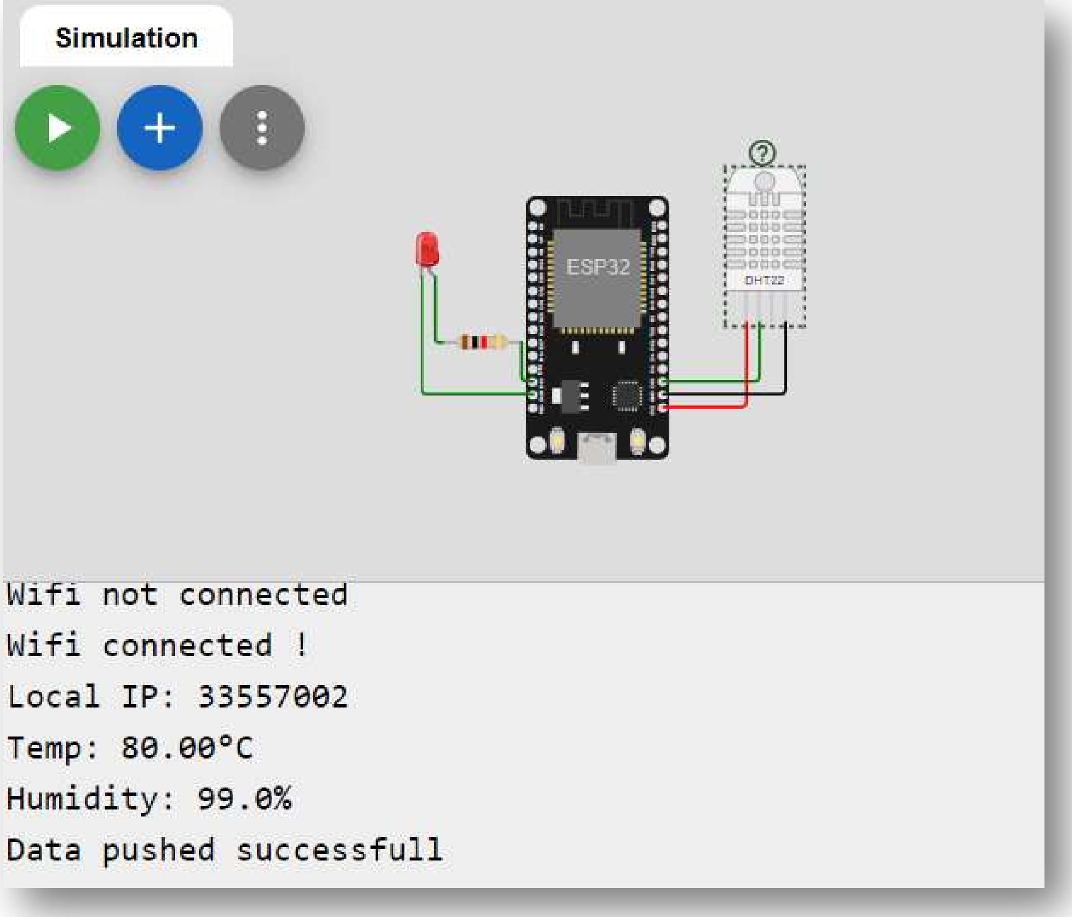
Algorithm:

- Include necessary libraries: WiFi library (<WiFi.h>), DHT sensor library ("DHTesp.h"), and ThingSpeak library ("ThingSpeak.h").
- Define constants for DHT sensor pin
- Define the ThingSpeak server address (server).
- Setup Function:
- Loop Function:
- Print success or error message based on the result of writing data to ThingSpeak.
- Delay execution for 10 seconds.

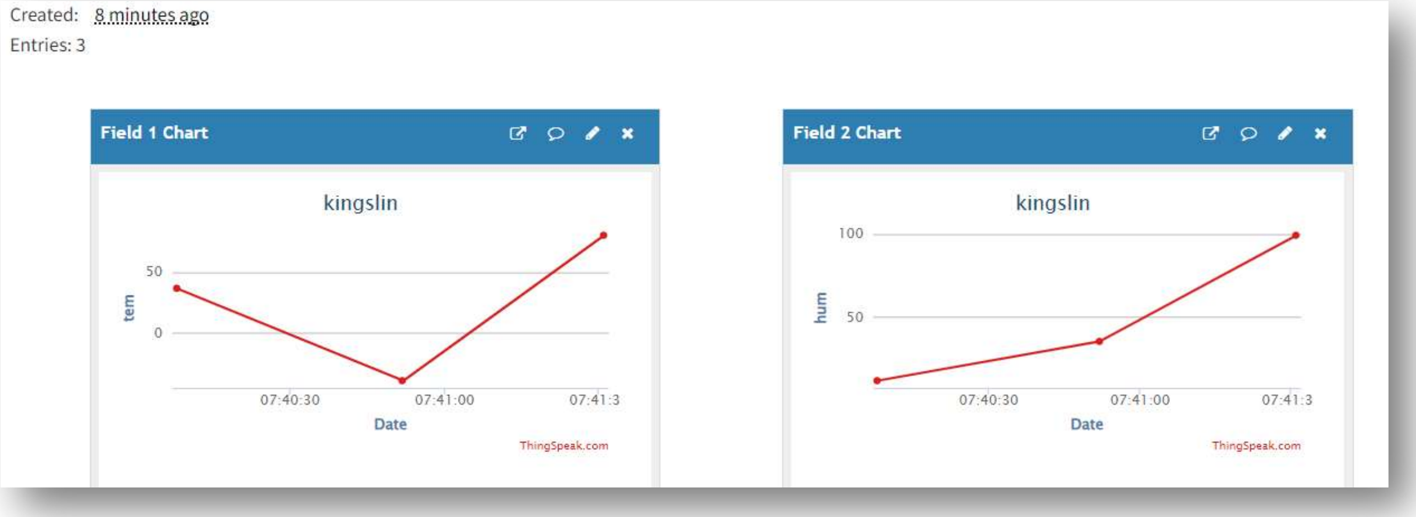
Program:

```
/**
   ESP32 + DHT22 Example for Wokwi

*/
#include <WiFi.h>
#include "DHTesp.h"
#include "ThingSpeak.h"

const int DHT_PIN = 15;
const int LED_PIN = 13;
const char* WIFI_NAME = "Wokwi-GUEST";
const char* WIFI_PASSWORD = "";
const int myChannelNumber = 2505973;//kingslin ID
const char* myApiKey = "MZ07TU3DVNOD6WVL";// kingslin KEY
const char* server = "api.thingspeak.com";

DHTesp dhtSensor;
WiFiClient client;

void setup() {
  Serial.begin(115200);
  dhtSensor.setup(DHT_PIN, DHTesp::DHT22);
  pinMode(LED_PIN, OUTPUT);
  WiFi.begin(WIFI_NAME, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED){
    delay(1000);
    Serial.println("Wifi not connected");
  }
  Serial.println("Wifi connected !");
  Serial.println("Local IP: " + String(WiFi.localIP()));
  WiFi.mode(WIFI_STA);
  ThingSpeak.begin(client);
}

void loop() {
  TempAndHumidity  data = dhtSensor.getTempAndHumidity();
  ThingSpeak.setField(1,data.temperature);
  ThingSpeak.setField(2,data.humidity);
  if (data.temperature > 35 || data.temperature < 12 || data.humidity > 70 ||
data.humidity < 40) {
    digitalWrite(LED_PIN, HIGH);
  }else{
    digitalWrite(LED_PIN, LOW);
  }

  int x = ThingSpeak.writeFields(myChannelNumber,myApiKey);

  Serial.println("Temp: " + String(data.temperature, 2) + "°C");
  Serial.println("Humidity: " + String(data.humidity, 1) + "%");

  if(x == 200){
    Serial.println("Data pushed successfull");
  }else{
    Serial.println("Push error" + String(x));
  }
  Serial.println("---");

  delay(10000);
}
```

Output:



```
Wifi not connected
Wifi connected !
Local IP: 33557002
Temp: 80.00°C
Humidity: 99.0%
Data pushed successfull
```



Created: 8 minutes ago

Entries: 3

Result:

Thus the program was executed and also data uploaded cloud, the output was verified successfully.