**INDEX**

**Embedded and IoT Technology**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr No** | **Practical** | **Date** | **Sign** |
| 1 | Design and implement basics embedded circuits  1. Automatic Alarm system- Alarm should get trigger by senor  2. Timer based buzzer  3. Sensor based Counting device |  |  |
| 2 | Demonstrate communication between two embedded devices using UART port. |  |  |
| 3 | Build an IoT system to send ticket before entering the bus. |  |  |
| 5 | Develop a IoT application which will record the movement and orientation of your phone and give the data back to the PC |  |  |
| 6 | Develop an IoT application that will raise an alarm whenever with going to rain outside based on the weather prediction data. |  |  |
| 7 | Deploy an IoT application which will alert you by beeping or vibrating your phone whenever you get someone call your name. |  |  |
| 8 | Develop an IoT application for monitoring water levels in tanks and automatically start the motor to fill the tank if the level goes below critical level. |  |  |
| 9 | Develop an IoT module to which measure the intensity of light and send the same to your PC/phone. |  |  |
| 10 | Develop an IoT application for Motion detection. |  |  |

**Practical 1**

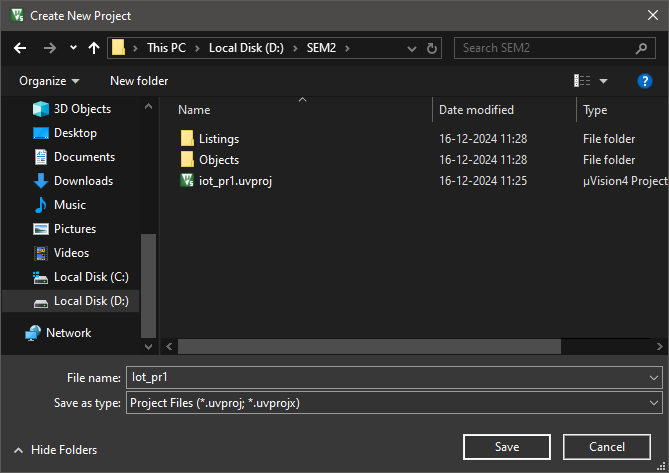
Aim: Design and implement basics embedded circuits

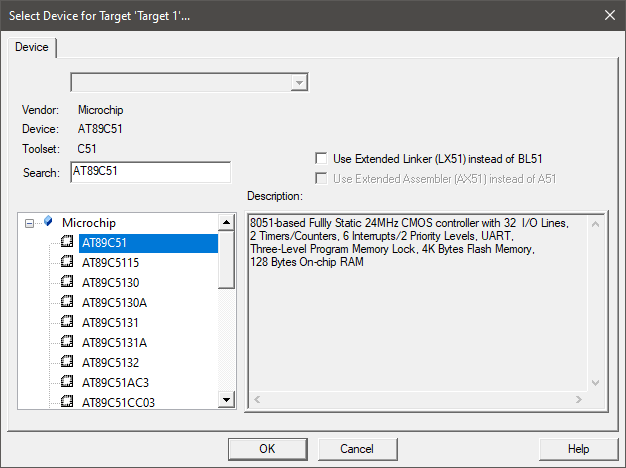
1. Automatic Alarm system- Alarm should get trigger by senor

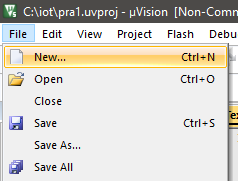
Keep Keil folder in C: & Proteus folder in D:

Keil:

Click on Project 🡪 New uVision Project

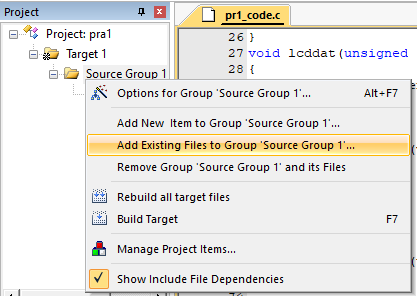




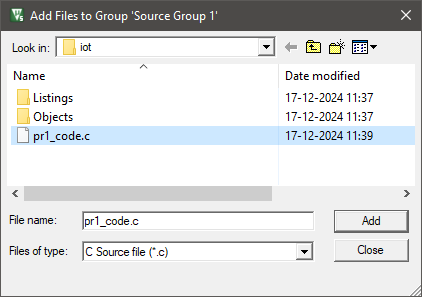


Type code & save it with .c extension

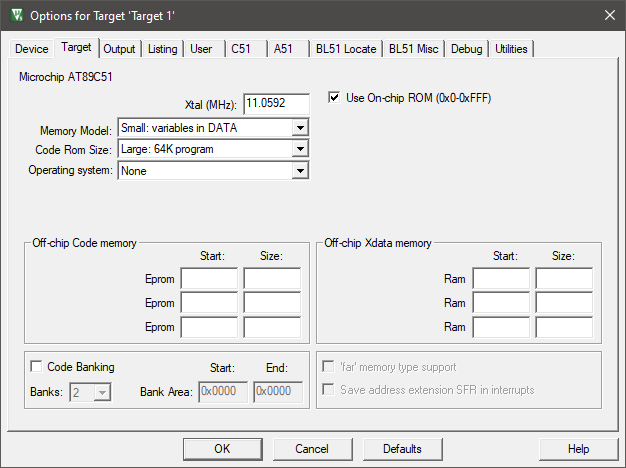
Go to Target 🡪 Source group1 🡪 right click 🡪 select add existing files

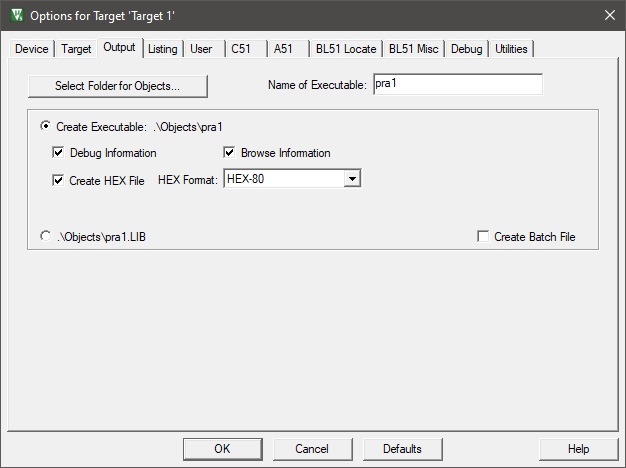


Select file & click Add

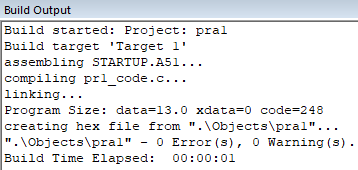


Right click on target 🡪 Option for target 🡪 Change frequency to 11.0592 MHz 🡪 Click on use on-chip ROM 🡪 Go to output label 🡪 Click on Create Hex file 🡪 Ok

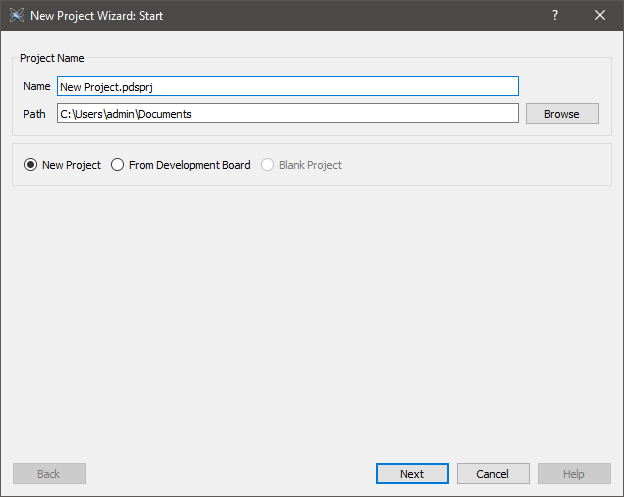


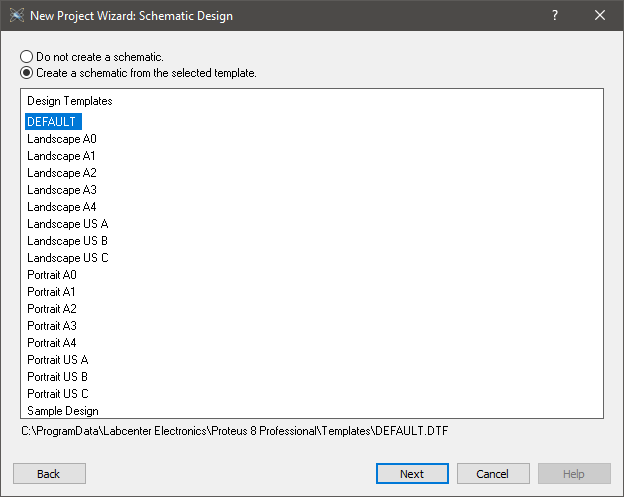


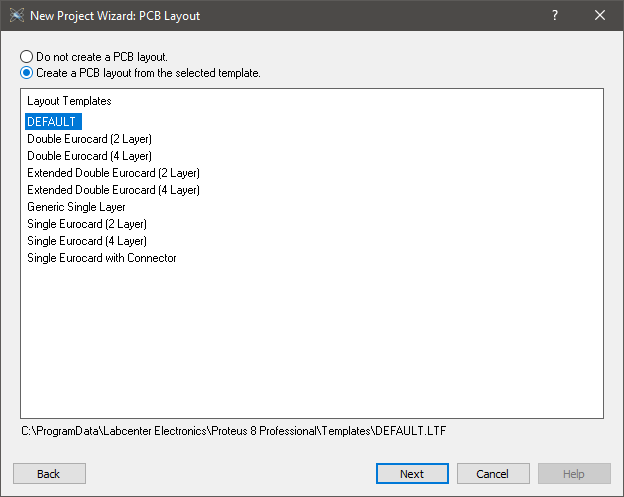
Click on Project 🡪 Build Target

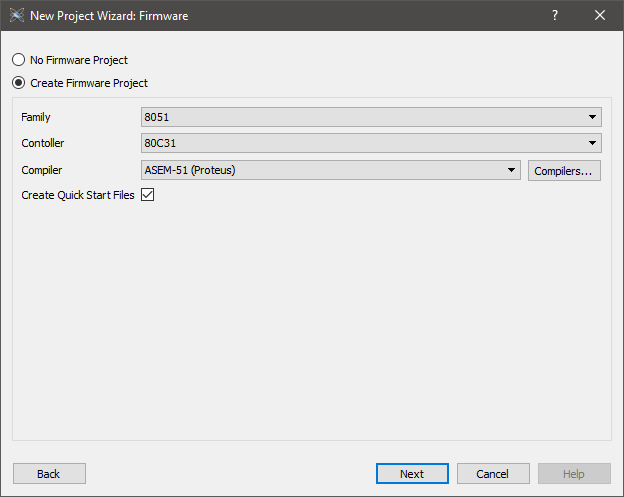


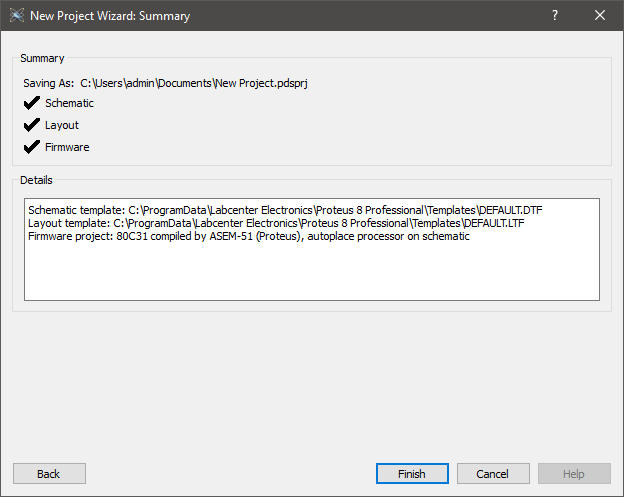
Proteus:





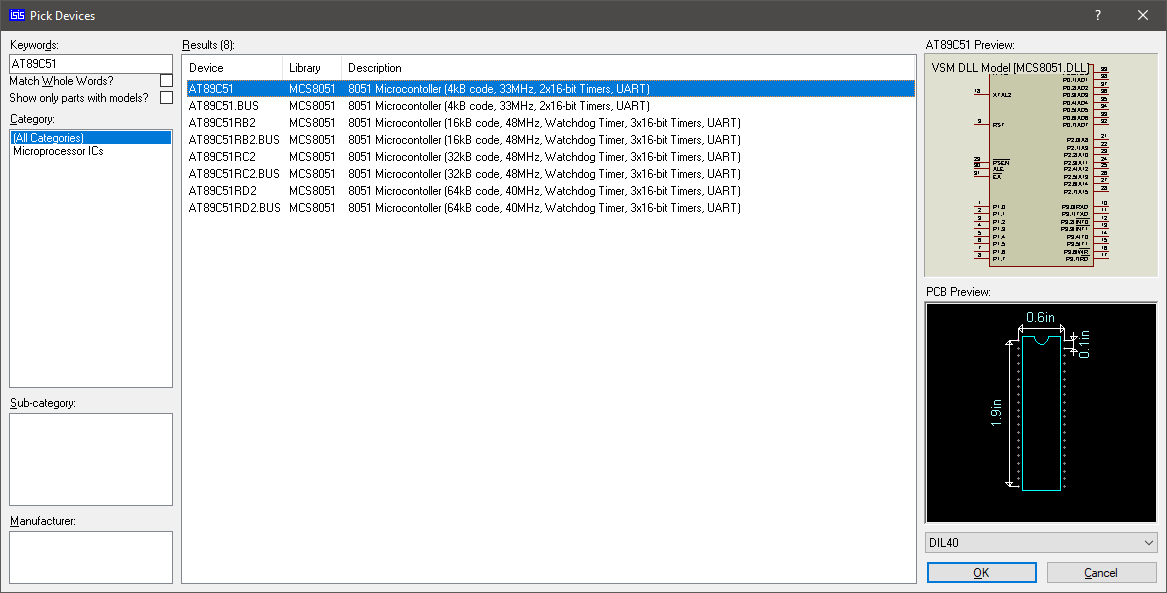


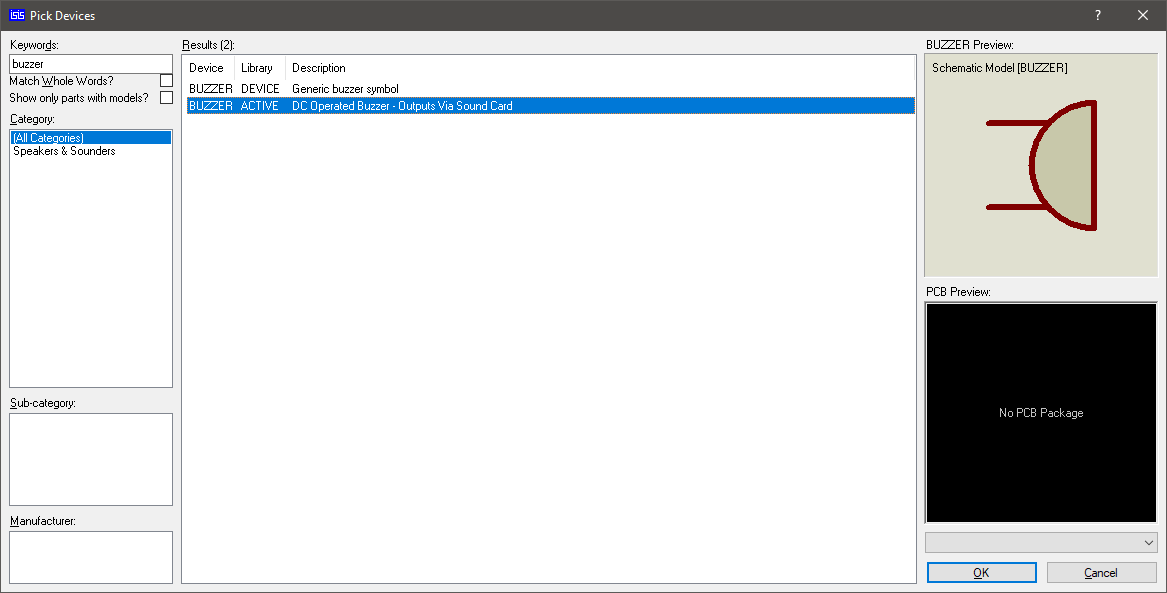


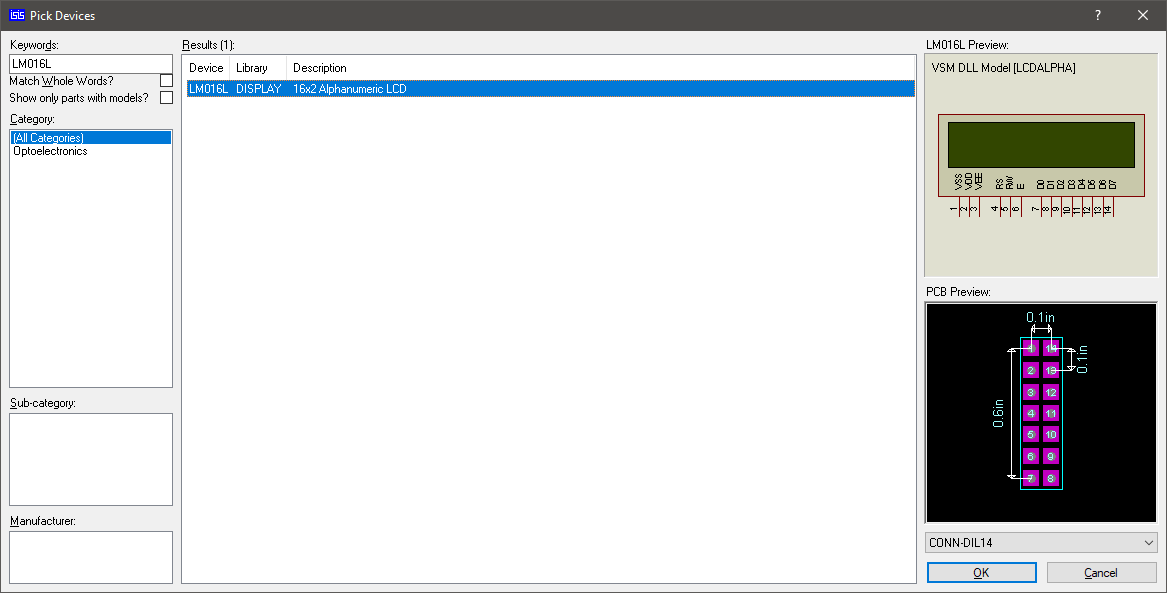


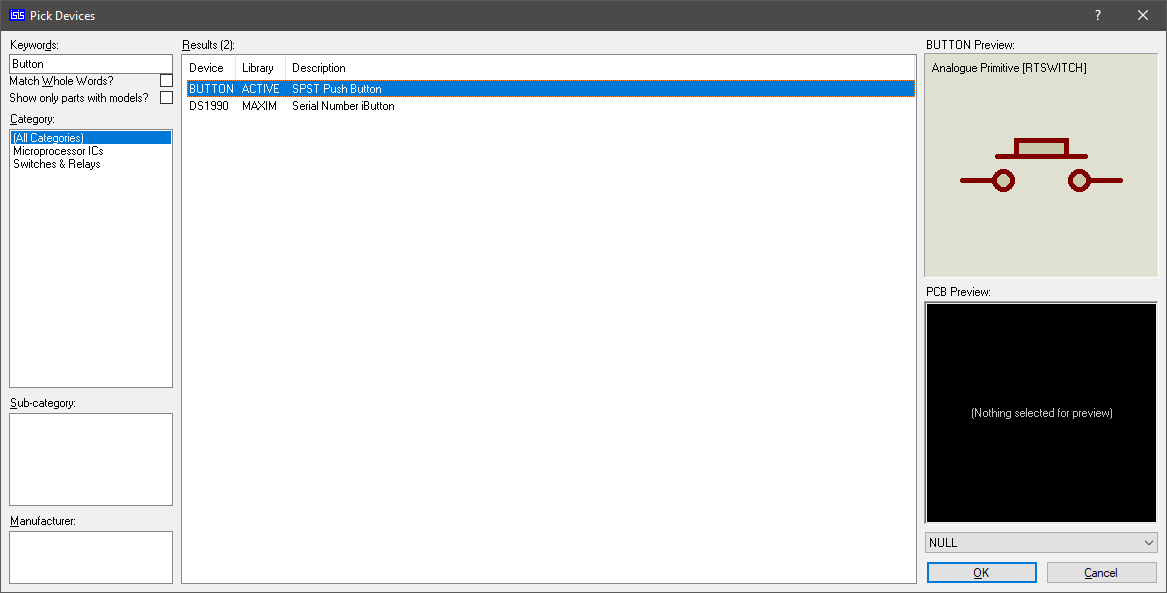
Click P 🡪 following window will open, add devices

* AT89C51
* Buzzer active DC Operated Buzzer Output Via Sound Card
* LCD: LM016L 16x2 Alphanumeric LCD
* Button (SPST Push button)

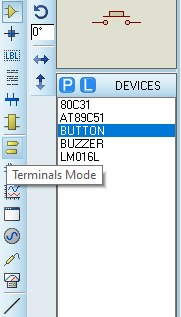




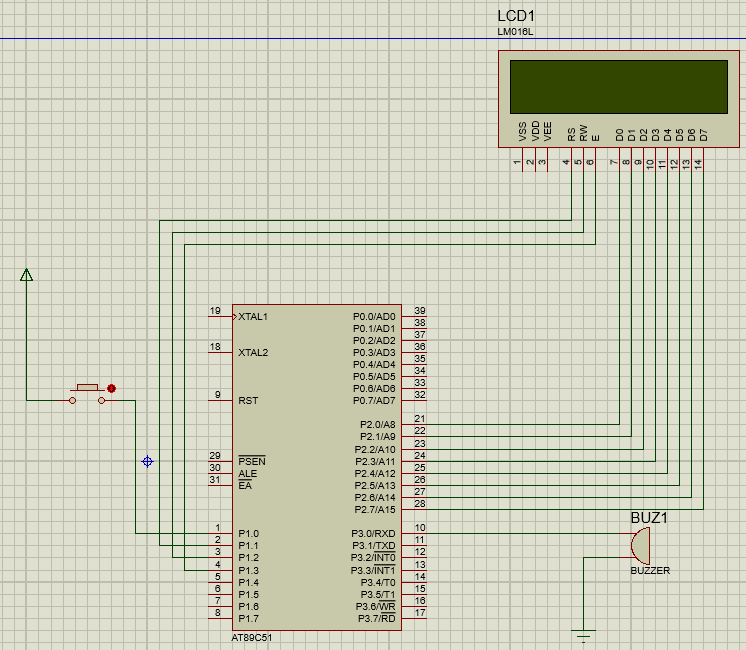




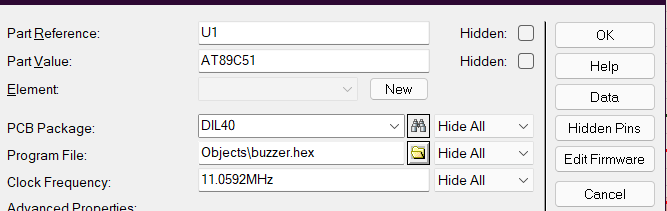
Go to Schematic Capture 🡪 Drag and drop all devices 🡪 click on Terminals mode 🡪 add power & ground



Design the Circuit

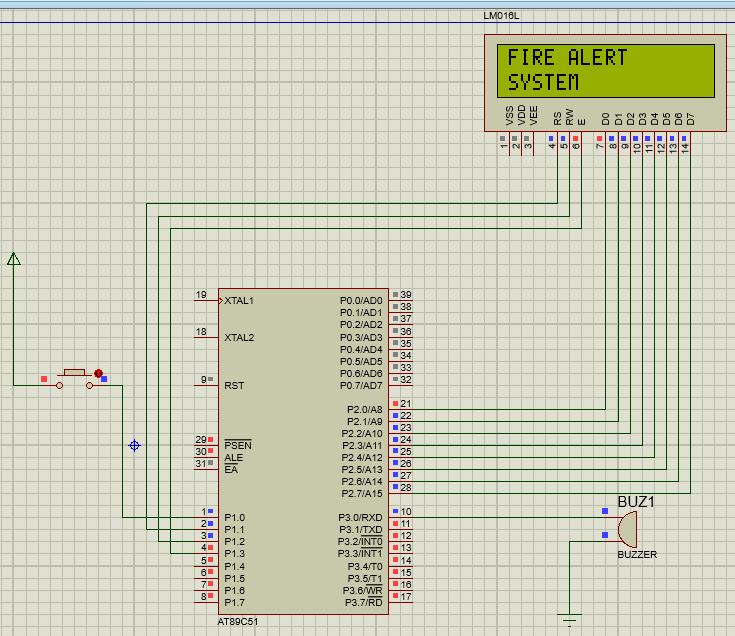


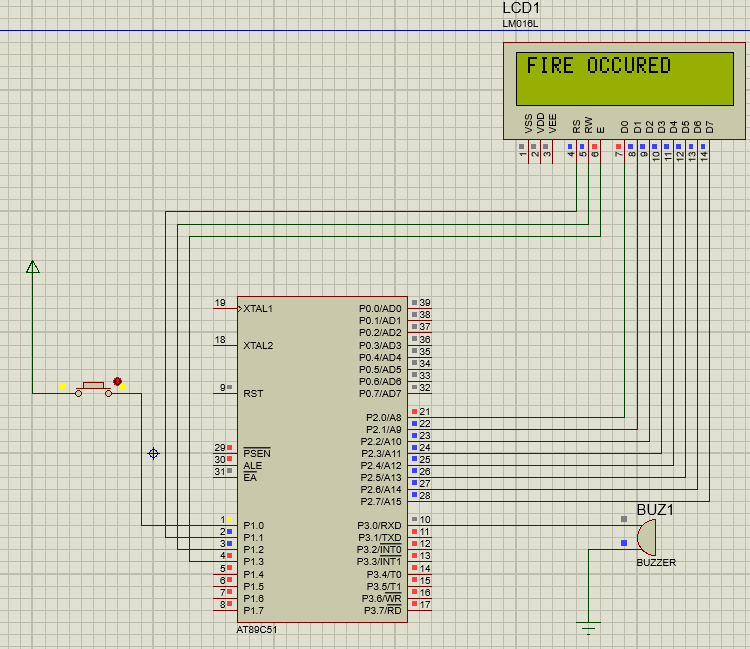
Double click on Micro controller 🡪 Select HEX file form program file 🡪 Change Clock frequency to 11.0592MHz 🡪 Click Ok



Output:

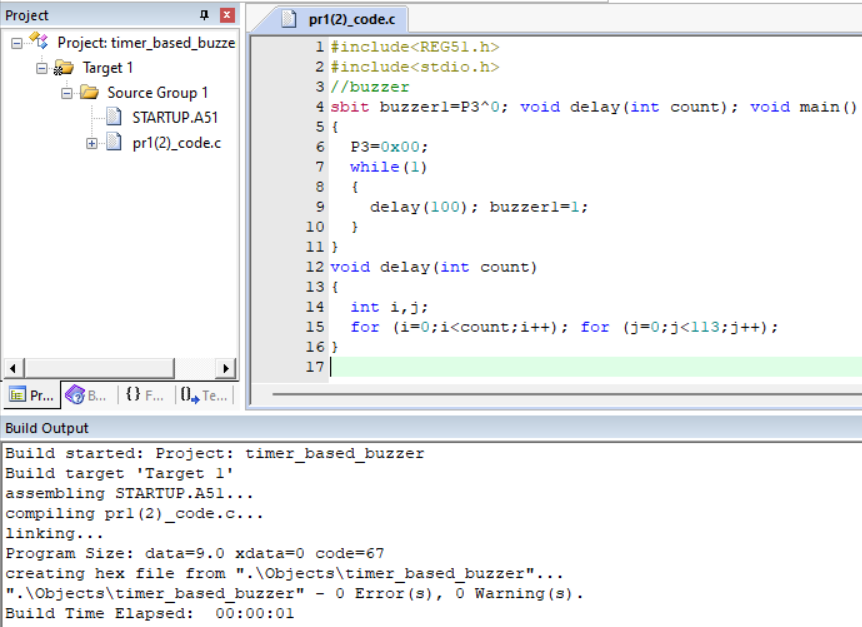
Start the simulation





2. Timer based buzzer

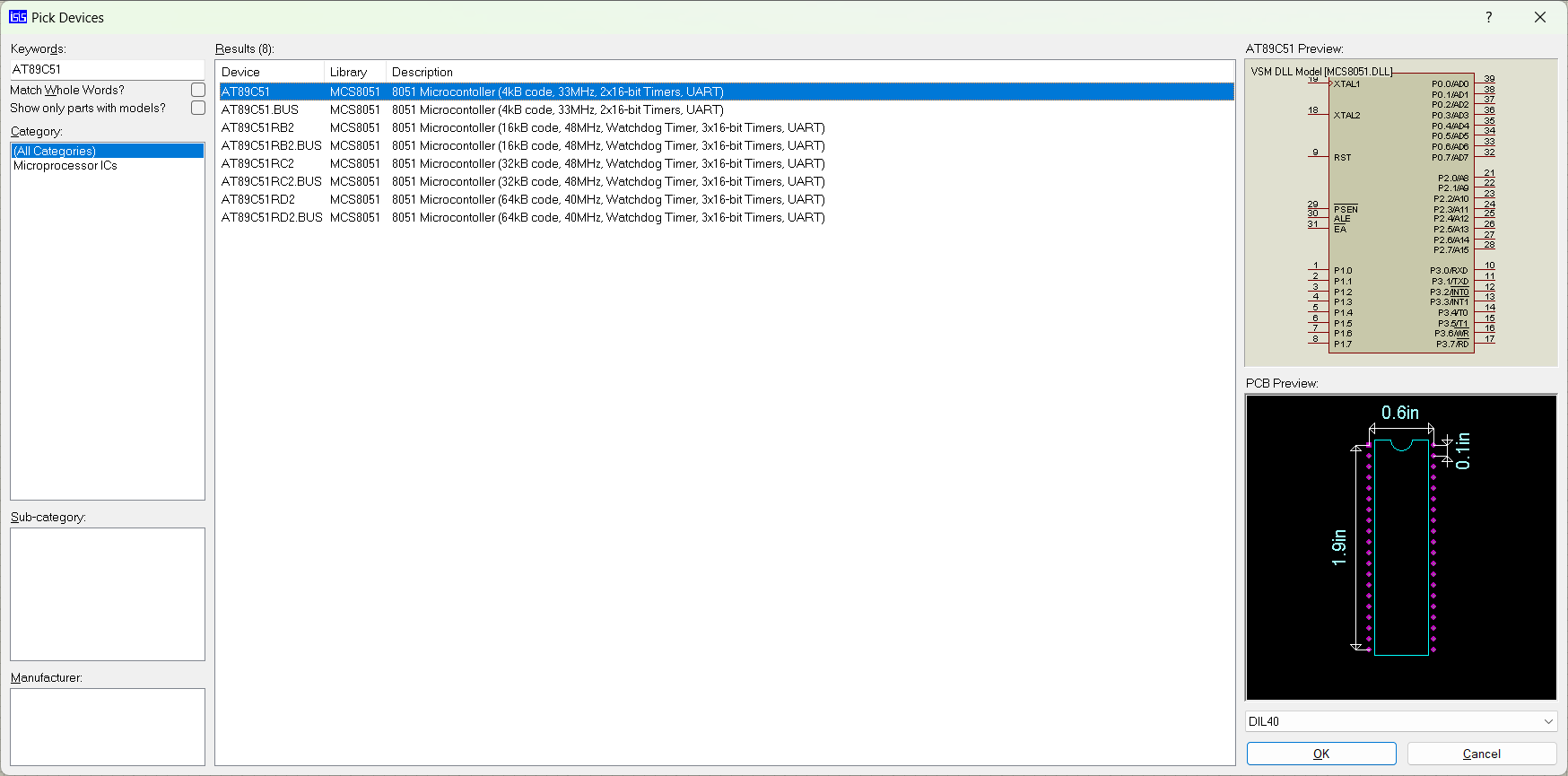
Keil:

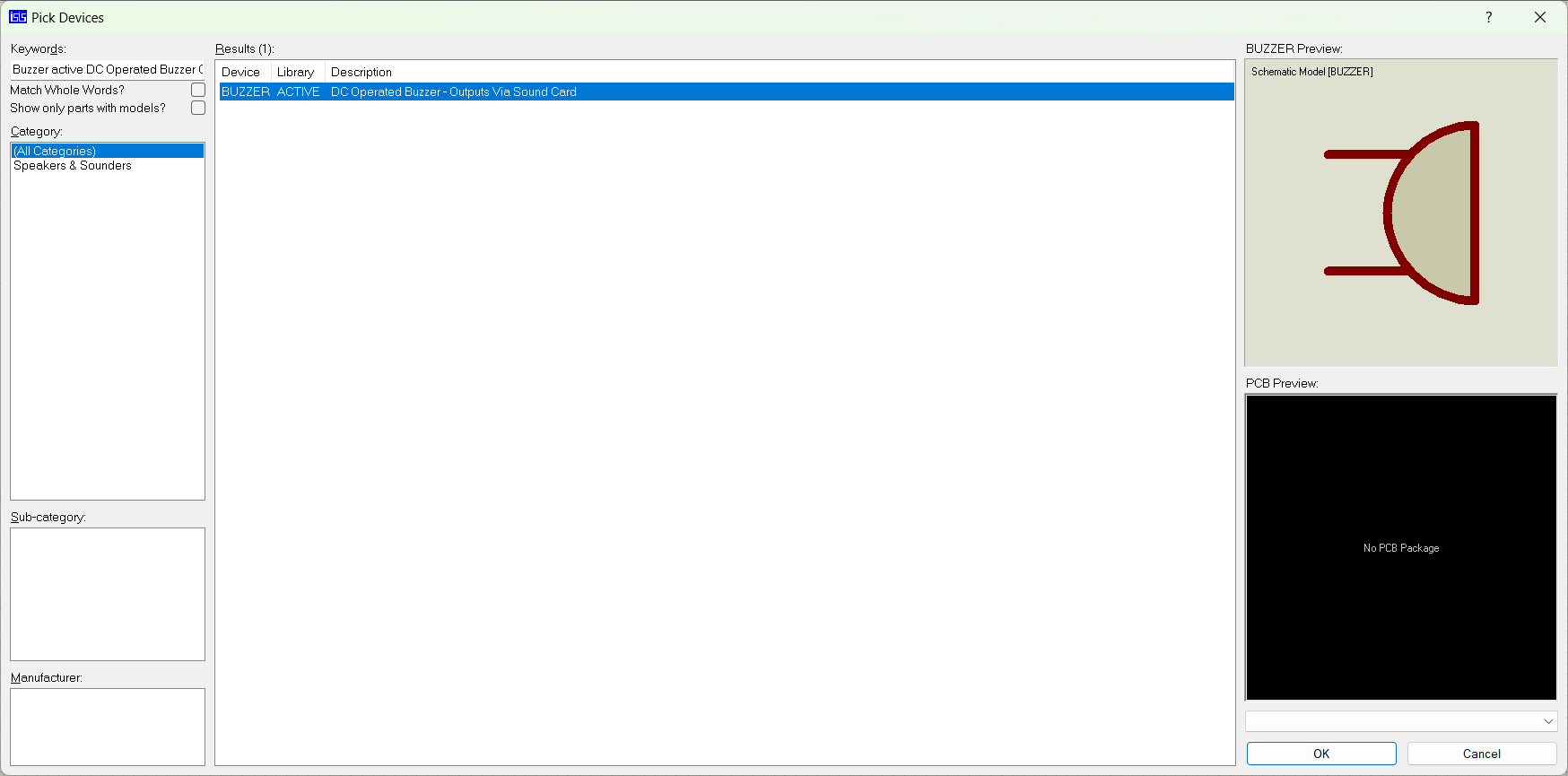


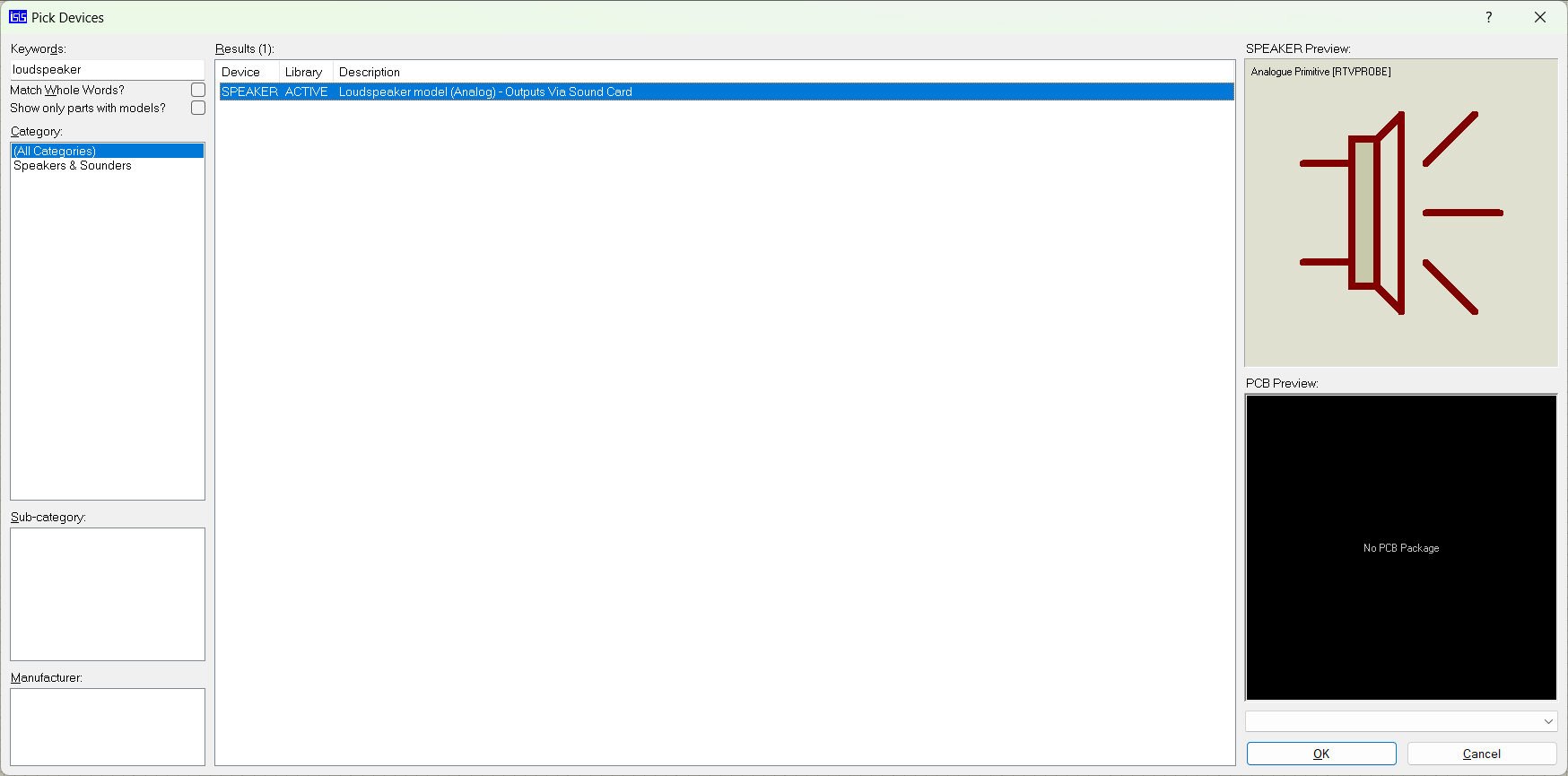
Proteus:

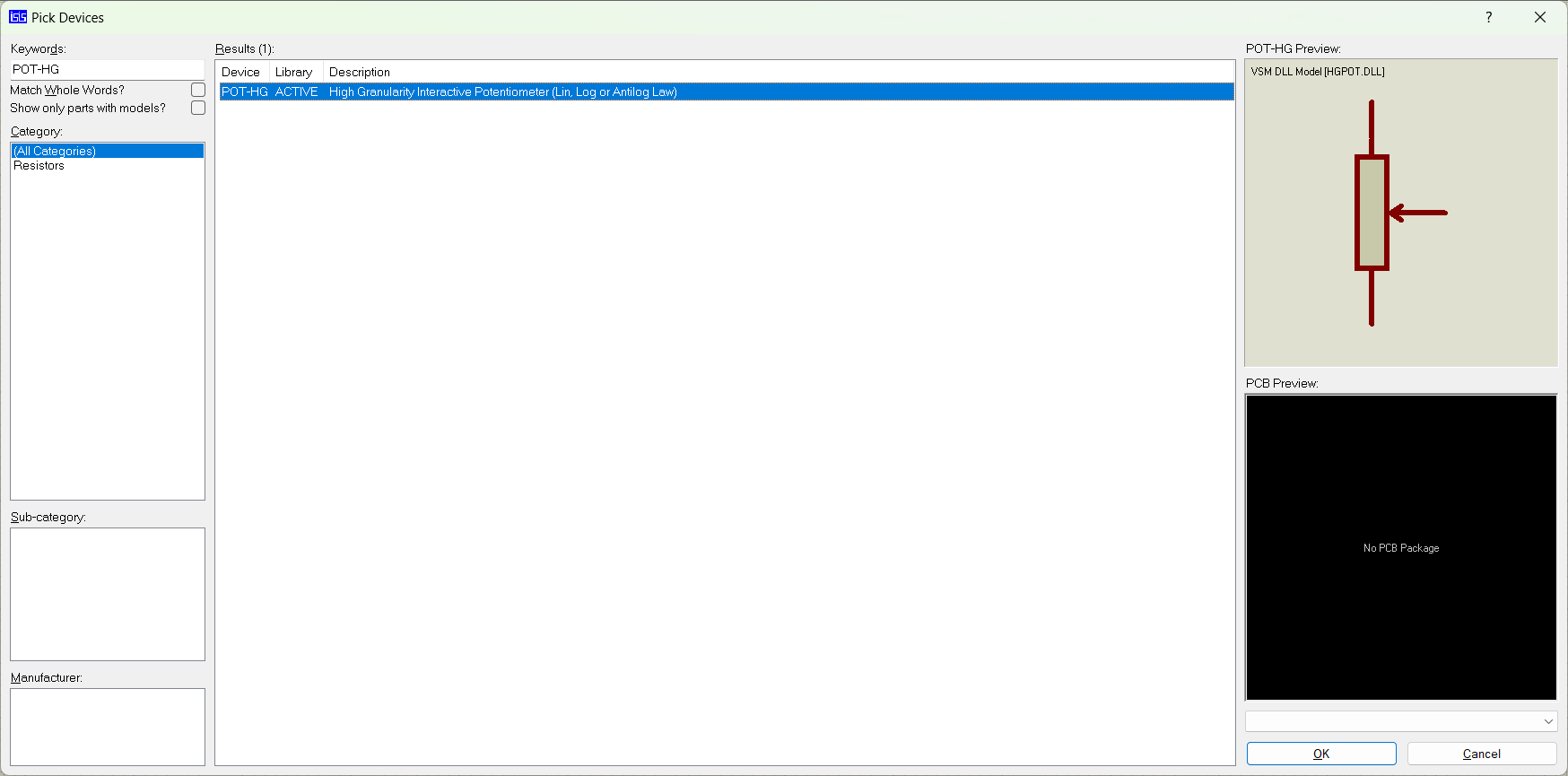
Create new project 🡪 add following devices

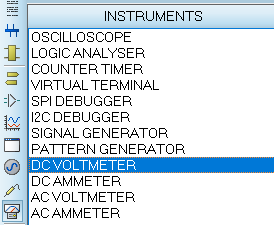
* AT89C51
* Buzzer active DC Operated Buzzer Output Via Sound Card
* Speaker- loudspeaker model(analog)- output via sound card
* Potentiometer- POT-HG
* Low-frequency low-power silicon NPN Transistor
* DC voltmeter from instruments

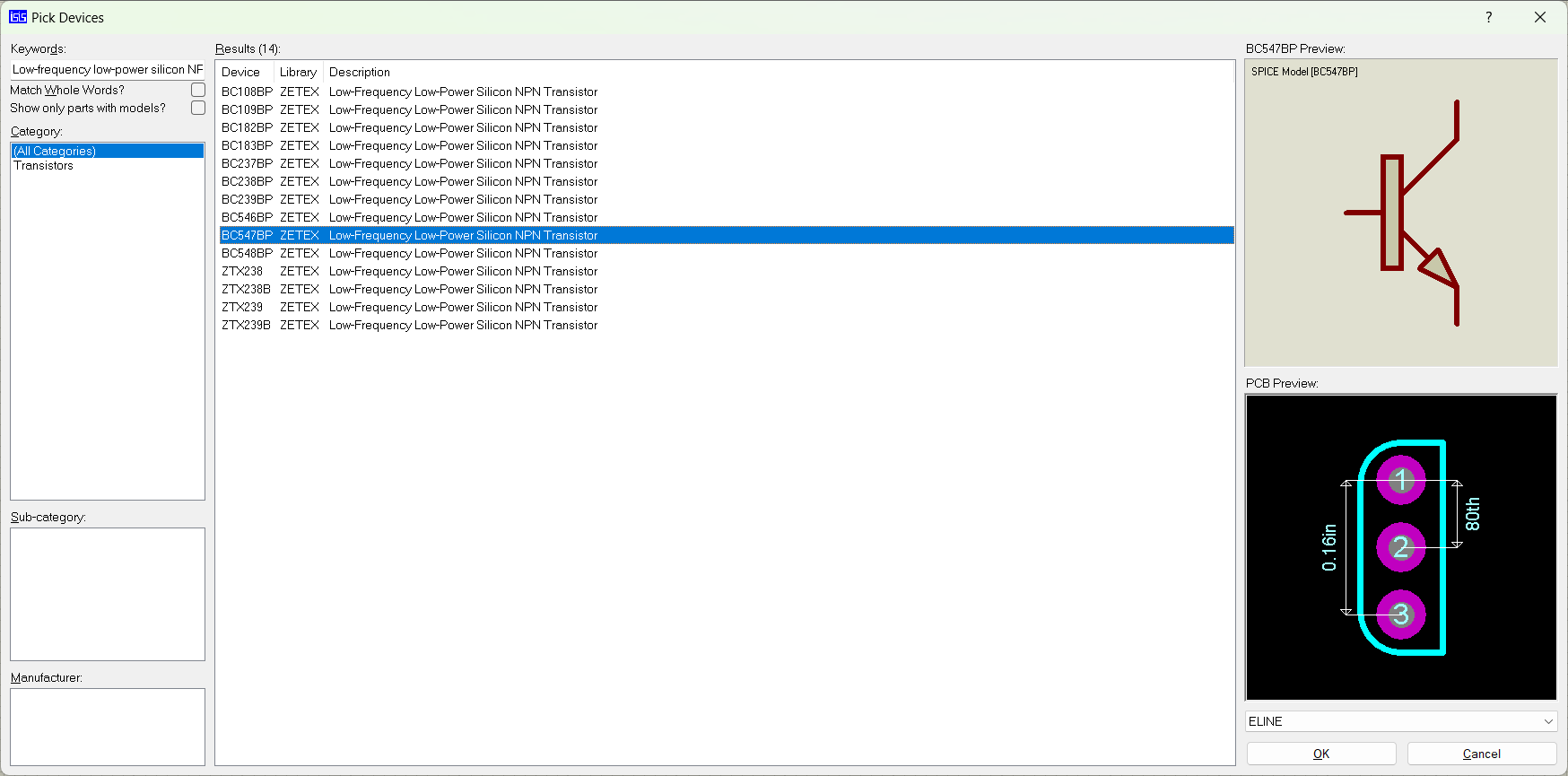




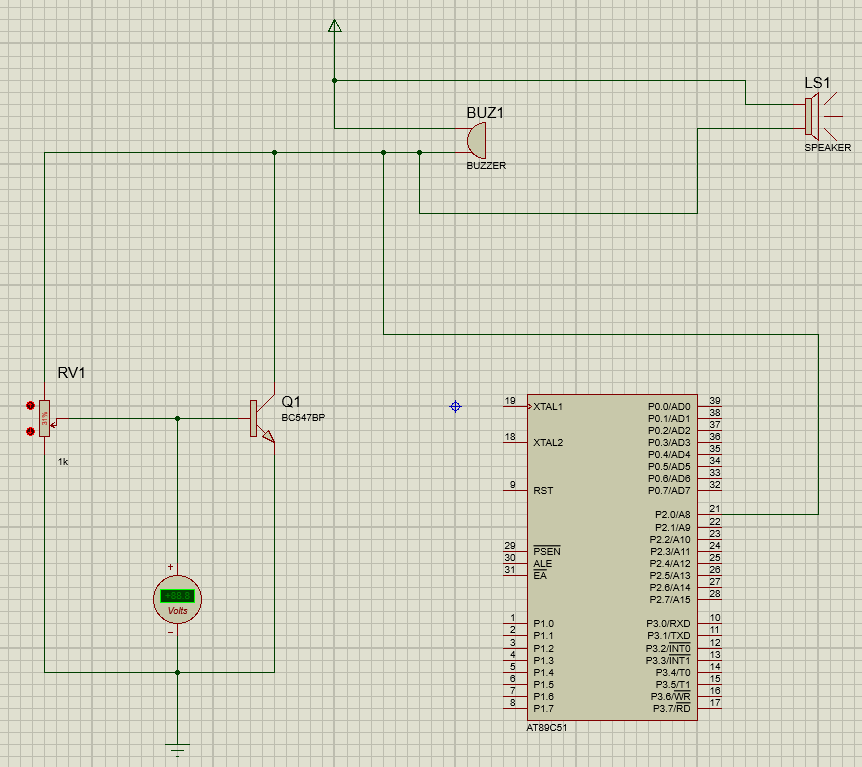




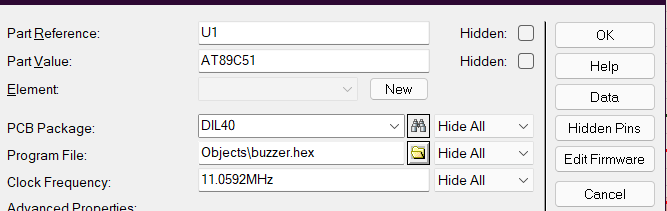




Design the Circuit



Double click on Micro controller 🡪 Select HEX file form program file 🡪 Change Clock frequency to 11.0592MHz 🡪 Click Ok



Output: Start the simulation

3. Sensor based Counting device

Keil:

#include<reg51.h>

sbit sen1=P1^0; //entry

sbit sen2=P1^1; //exit

sbit rs=P1^2;

sbit rw=P1^3;

sbit en=P1^4;

void lcdcmd(unsigned char);

void lcddat(unsigned char);

void delay();

void lcddis(unsigned char \*s,unsigned char r);

void lcdconv(unsigned char);

void main()

{

unsigned char x,y;

lcdcmd(0x38);

delay();

lcdcmd(0x01);

delay();

lcdcmd(0x10);

delay();

lcdcmd(0x0c);

delay();

lcddis("WELCOME",7);

lcdcmd(0xc0);

delay();

lcddis("VISITOR COUNTER",15);

delay();

lcdcmd(0x01);

delay();

while(1)

{

if(sen1==0)

{

lcdcmd(0x80);

delay();

lcddis("ENTRY:",6);

lcdcmd(0x87);

delay();

x=x+1;

lcdconv(x);

delay();

}

if(sen2==0)

{

lcdcmd(0xc0);

delay();

lcddis("EXIT:",5);

lcdcmd(0xc6);

delay();

y=y+1;

lcdconv(y);

delay();

}

}

}

void lcdcmd(unsigned char val)

{

P2=val;

rs=0;

rw=0;

en=1;

delay();

en=0;

}

void lcddat(unsigned char dat)

{

P2=dat;

rs=1;

rw=0;

en=1;

delay();

en=0;

}

void lcddis(unsigned char\*s,unsigned char r)

{

unsigned char w;

for(w=0;w<r;w++)

{

lcddat(s[w]);

delay();

}

}

void lcdconv(unsigned char num)

{

unsigned char p,n;

p=num/10;

n=num%10;

p=num+0x30;

n=n+0x30;

lcddat(p);

lcddat(n);

}

void delay()

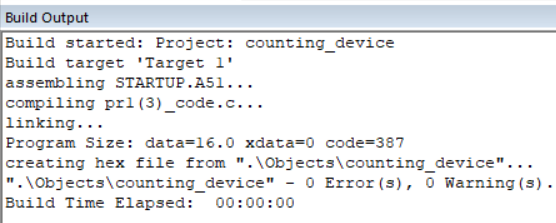
{

unsigned int k,l;

for(k=0;k<1000;k++);

for(l=0;l<100;l++);

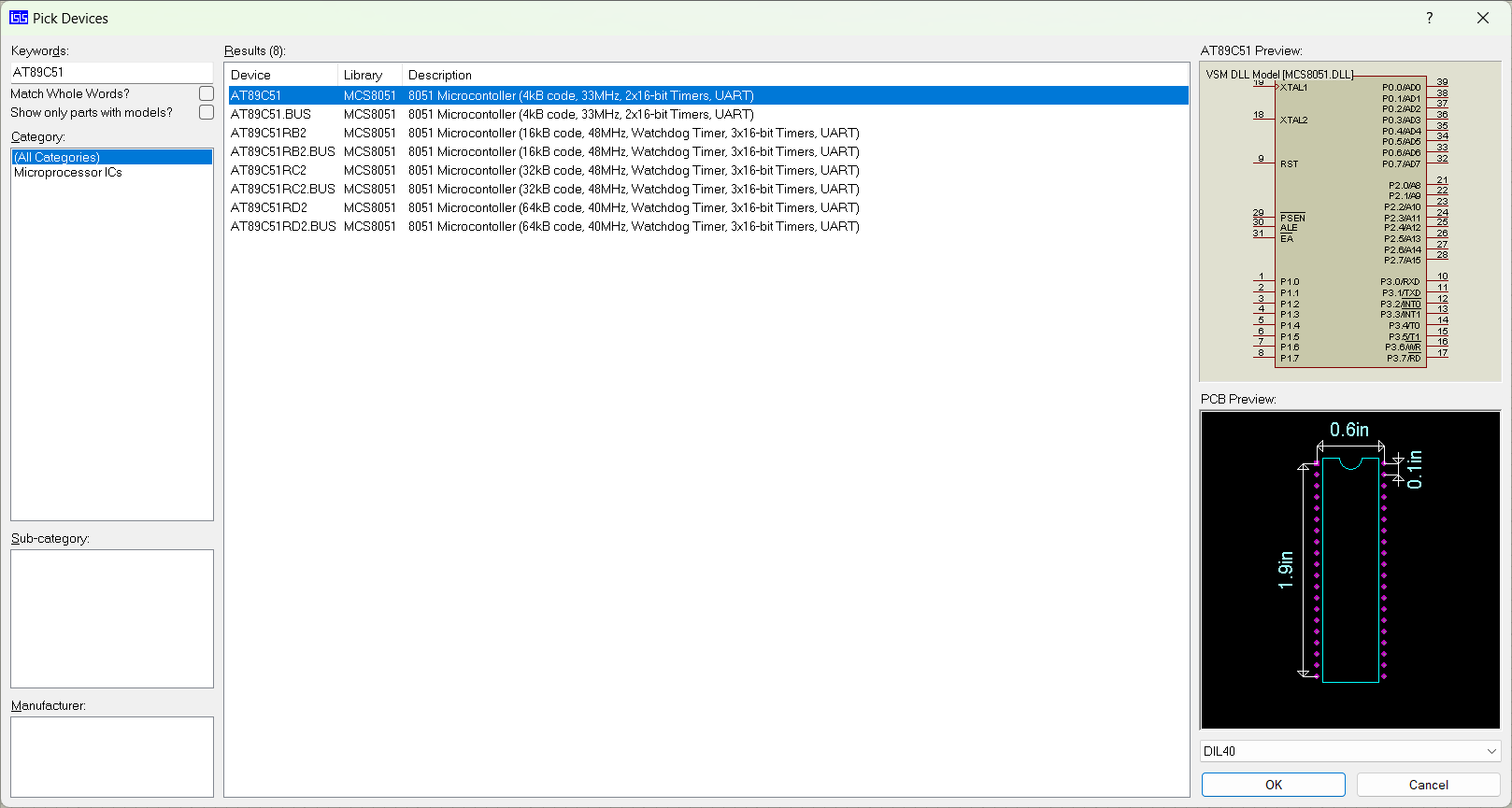
}

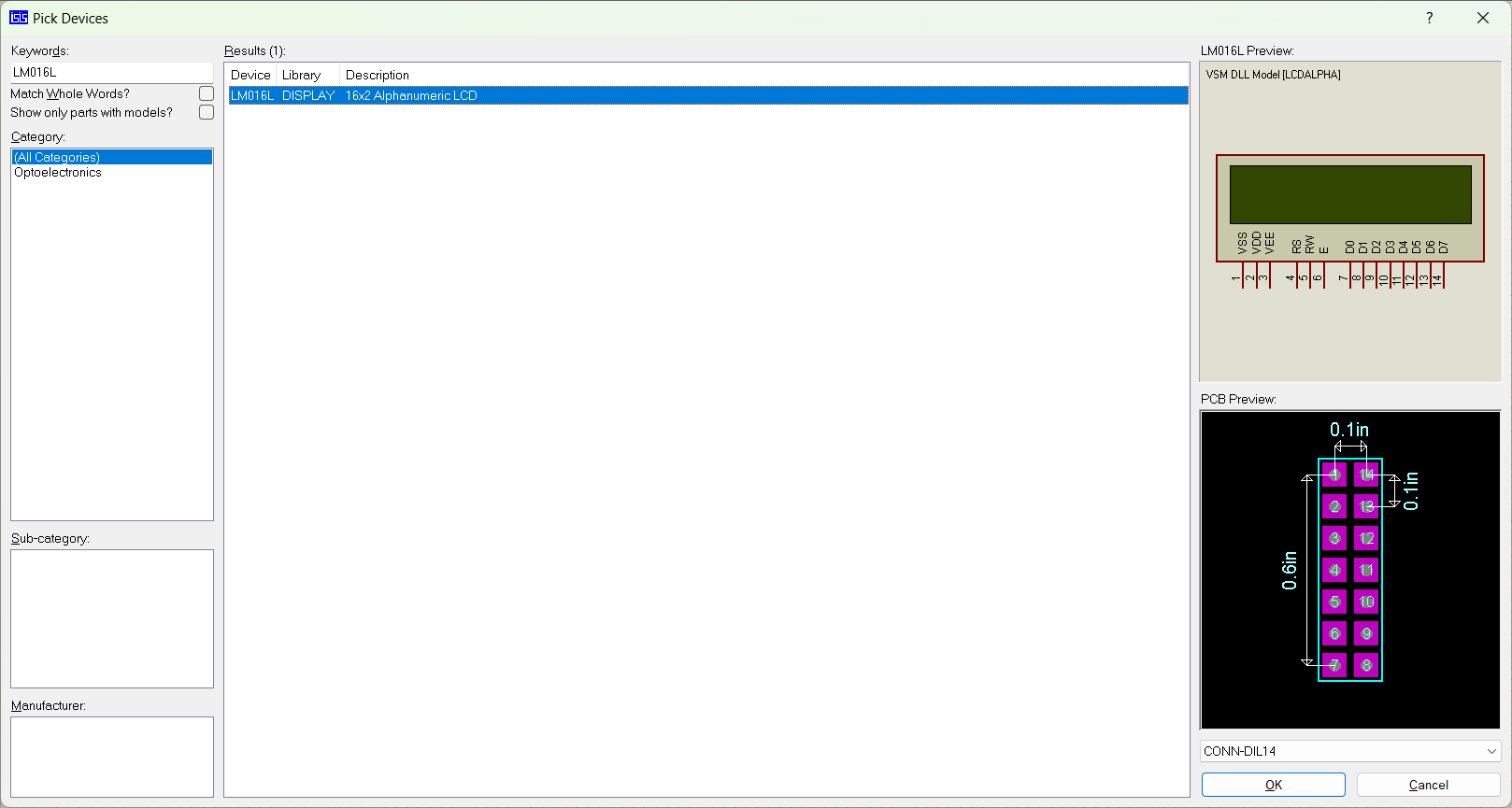


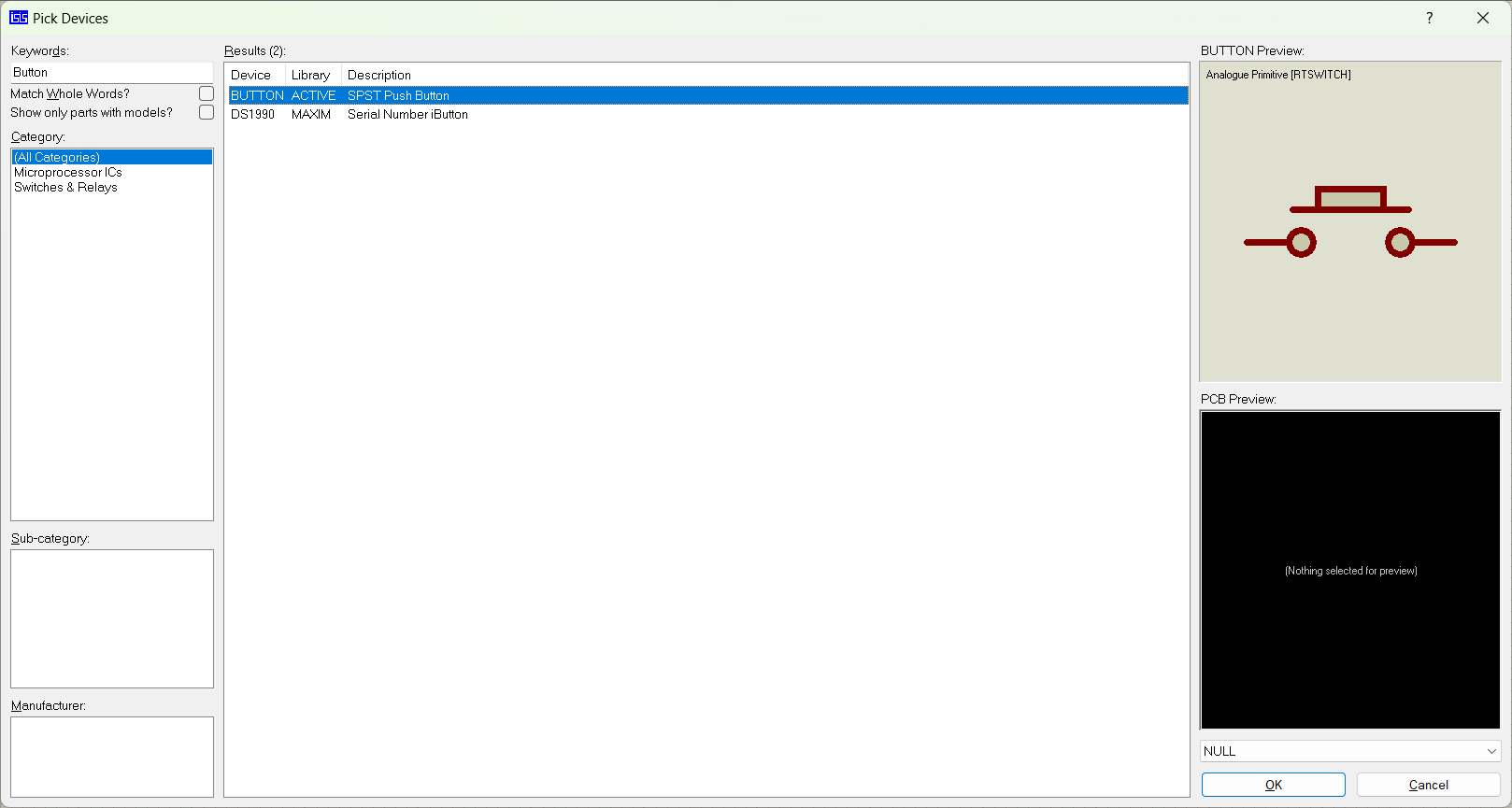
Proteus:

Create new project 🡪 add following devices

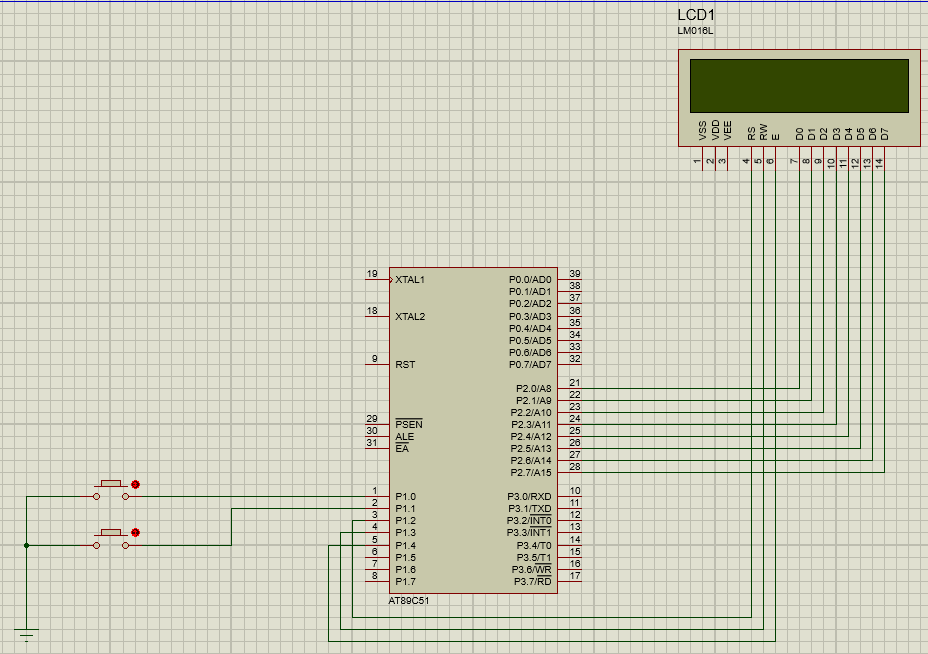
* AT8C51
* LCD: LM016L 16x2 Alphanumeric LCD
* Button (SPST Push button)



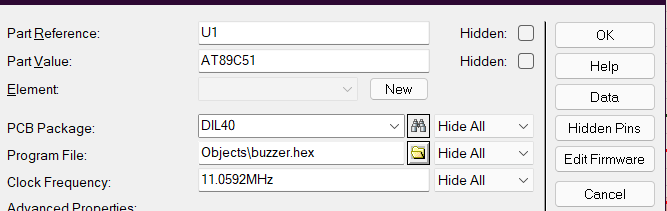




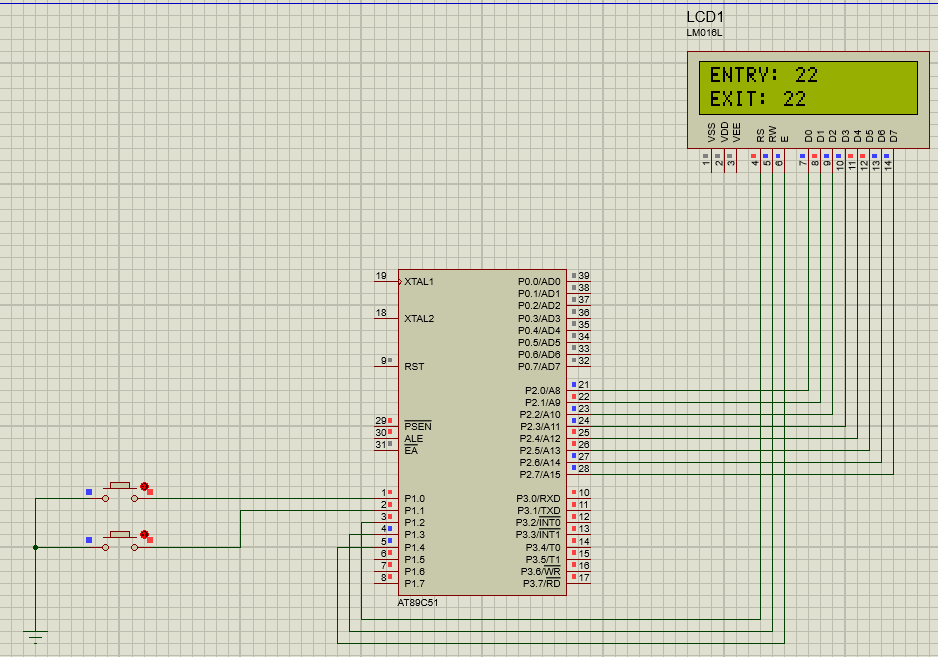
Design the Circuit



Double click on Micro controller 🡪 Select HEX file form program file 🡪 Change Clock frequency to 11.0592MHz 🡪 Click Ok



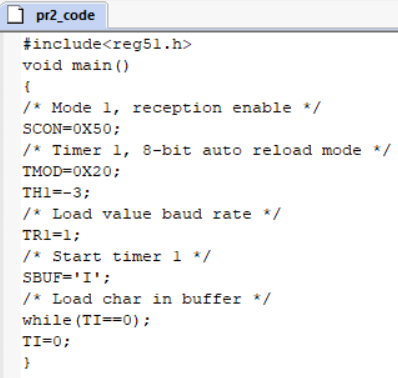
Output: Start the simulation

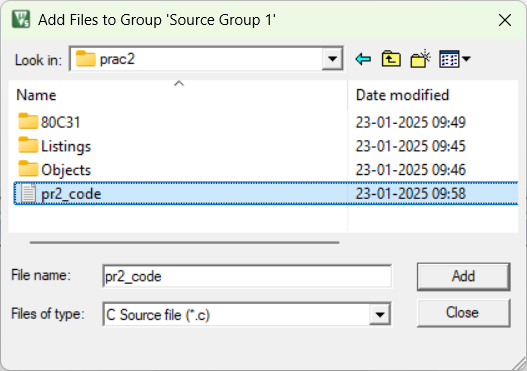


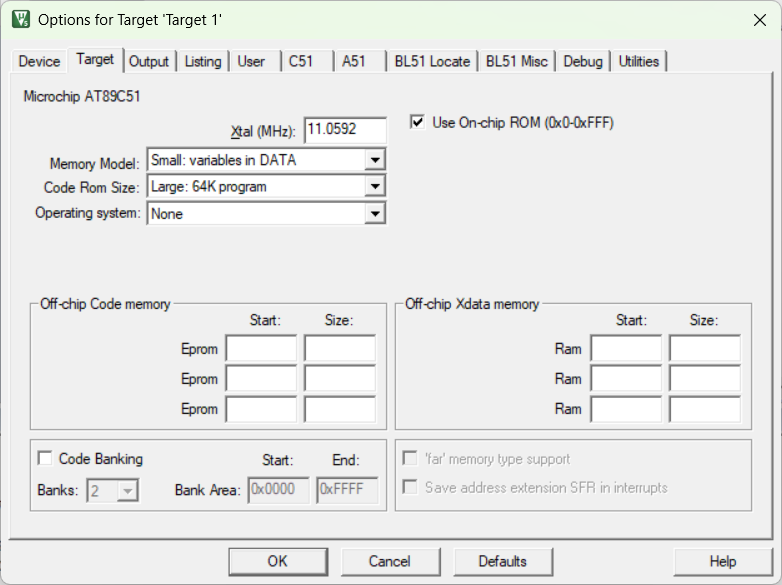
**Practical 2**

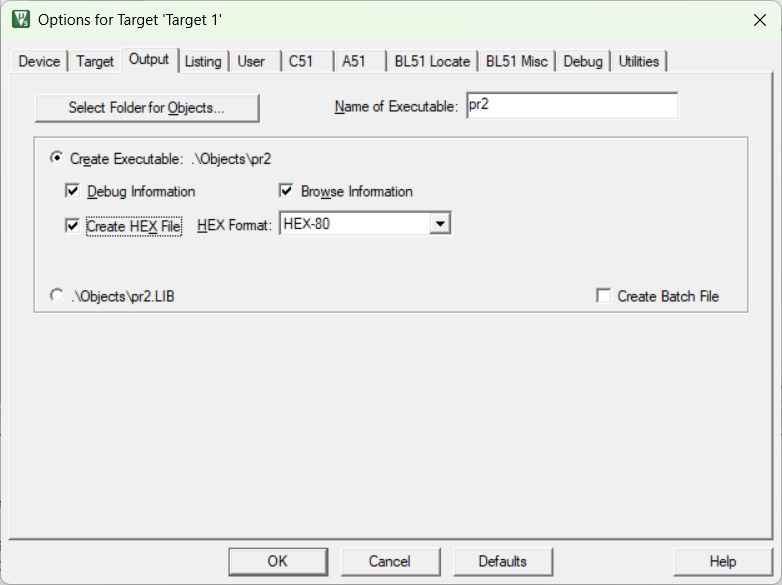
Aim: Demonstrate communication between two embedded devices using UART port.

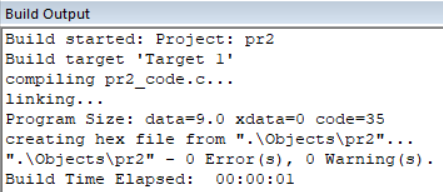
Keil:







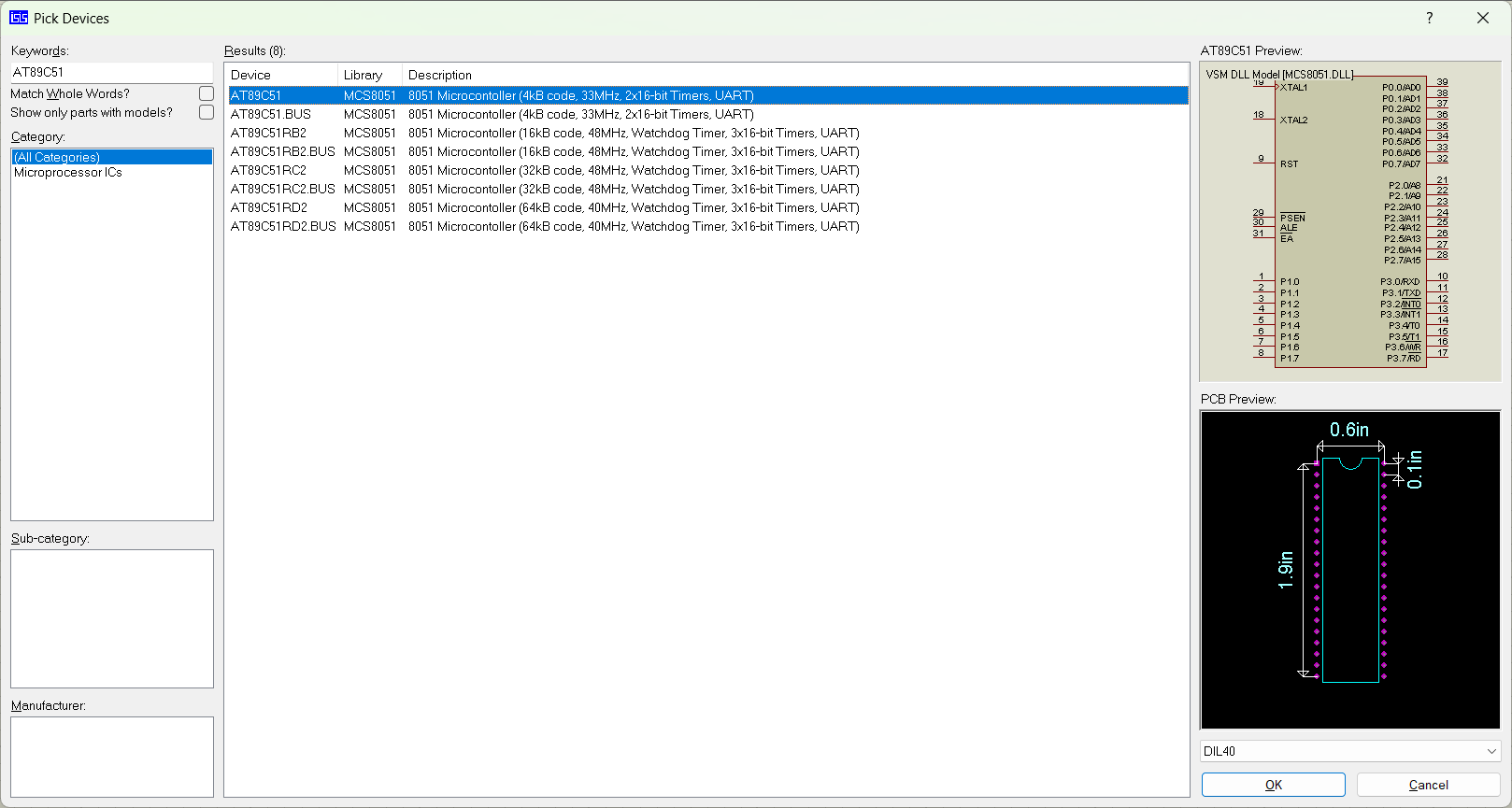


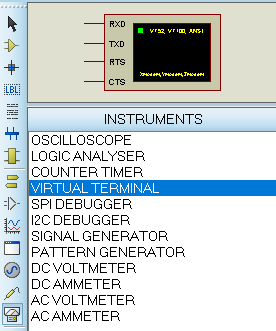


Proteus:

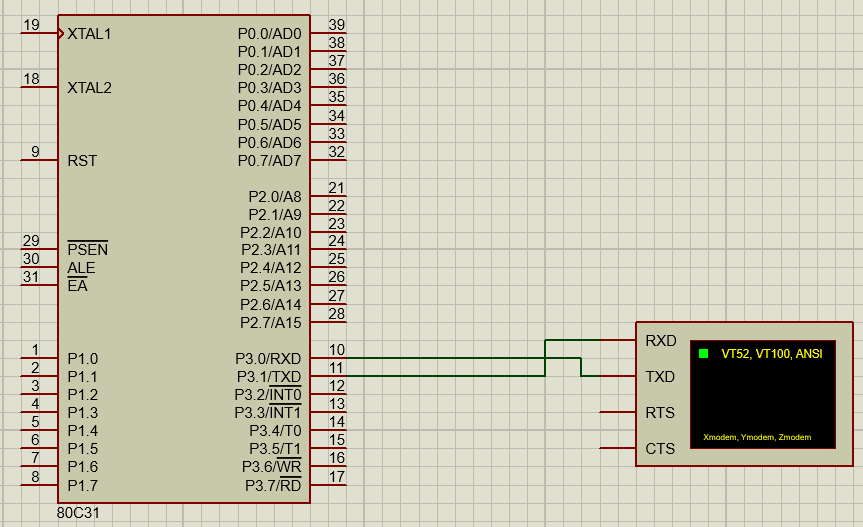
Create new project 🡪 add following devices

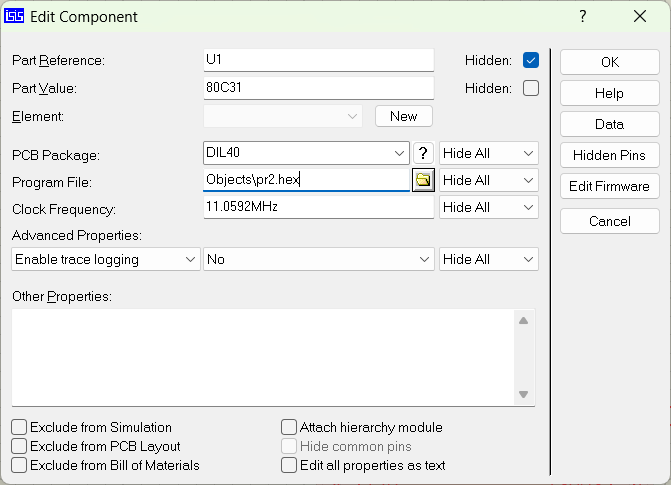
* AT8C51
* Virtual Terminal from instruments



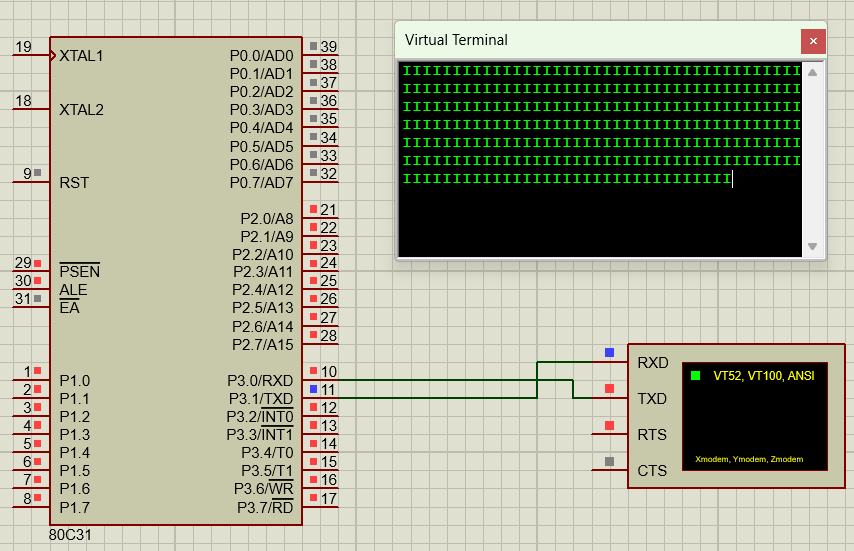


Design the Circuit





Output: Start the simulation



**Practical 3**

Aim: Build an IoT system to send ticket before entering the bus.

Arduino IDE:

A screenshot of a computer program

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

Select Arduino Uno Board

A black screen with white text

Description automatically generated

Proteus:

Create new project 🡪 add following devices

* SIMULINO UNO controller
* MOTOR- PWMSERVO -Animated PMW Controlled Servo motor
* Virtual Terminal from instruments.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

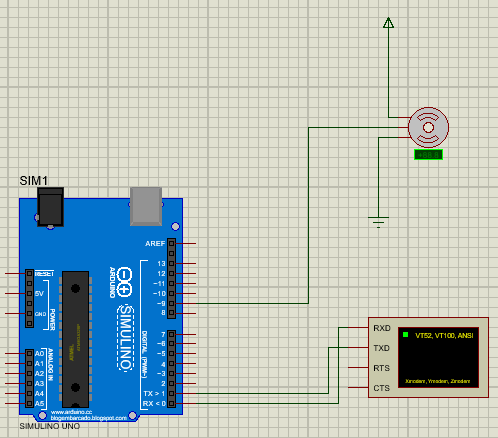
Description automatically generated

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

Design the circuit:



Output: Run the simulation

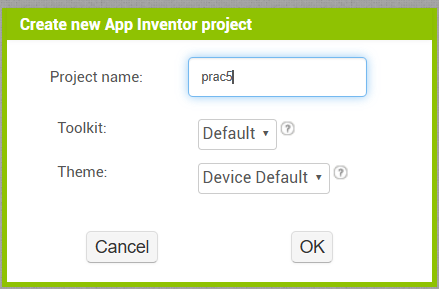
A blue circuit board with black wires and a black screen

Description automatically generated

**Practical 5**

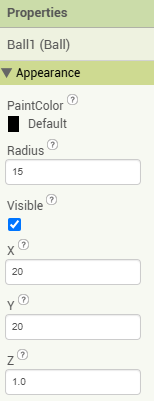
Aim: Develop a IoT application which will record the movement and orientation of your phone and give the data back to the PC.

Go to <https://appinventor.mit.edu/> 🡪 click on Get started 🡪 click on ‘online tool’ 🡪 create new project

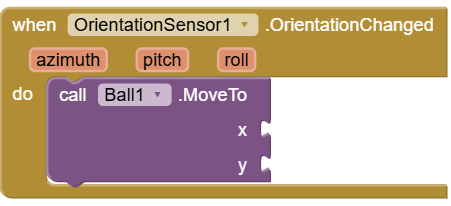


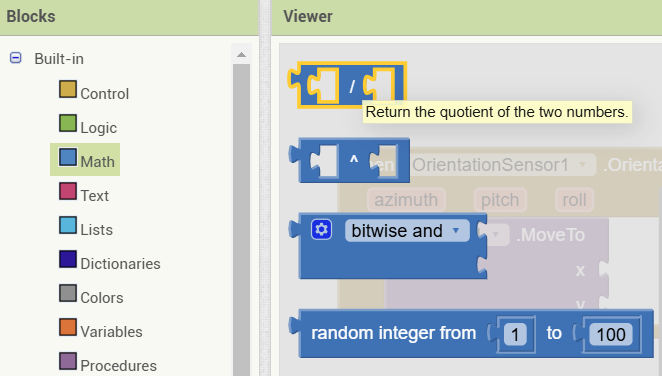
From palette 🡪 go to sensor 🡪 select orientation sensor 🡪 go to drawing and animation 🡪 select canvas 🡪 take a ball and put it in canvas

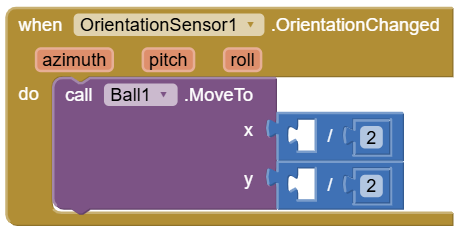
Set the properties

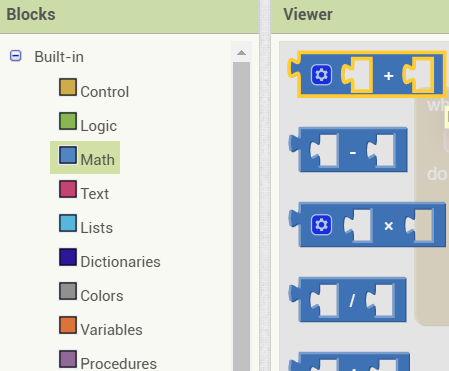
 

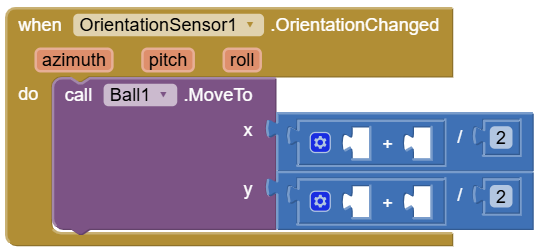
Click on Blocks 🡪 OrientationSensor1 🡪 click on **when** block 🡪 click on Ball1 🡪 select **call Ball1.Move to** block



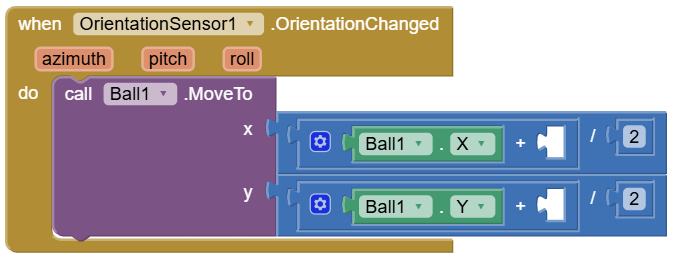


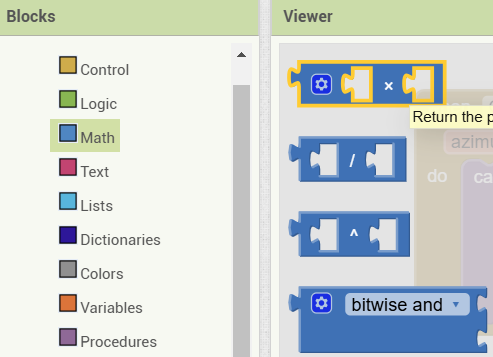


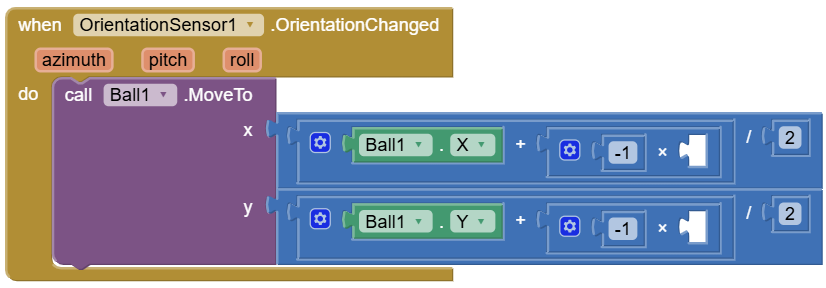


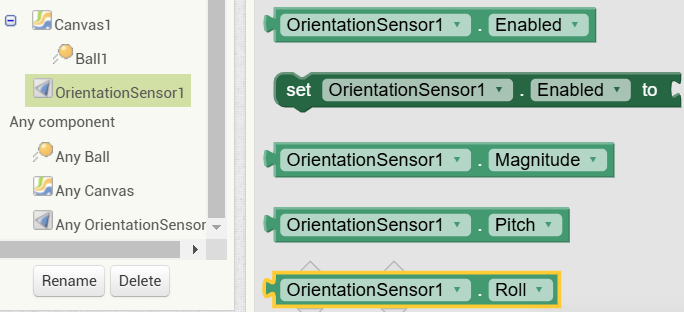


Click on Ball1 🡪 select **Ball1 X** and **Ball1 Y** block

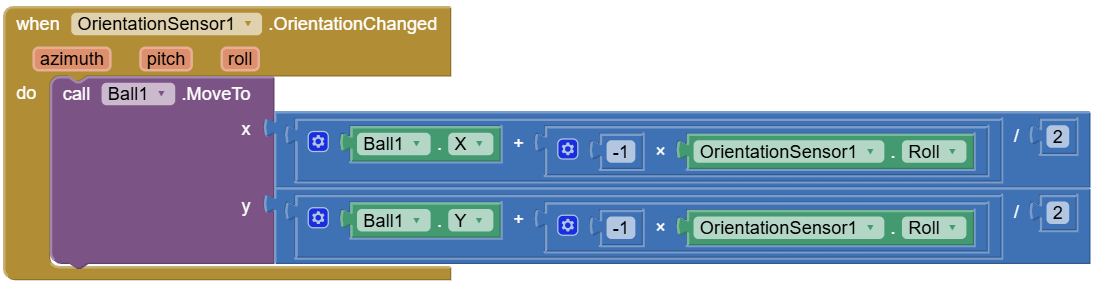




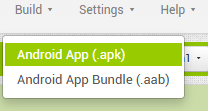




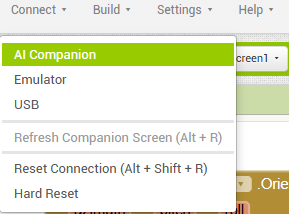
Final block diagram



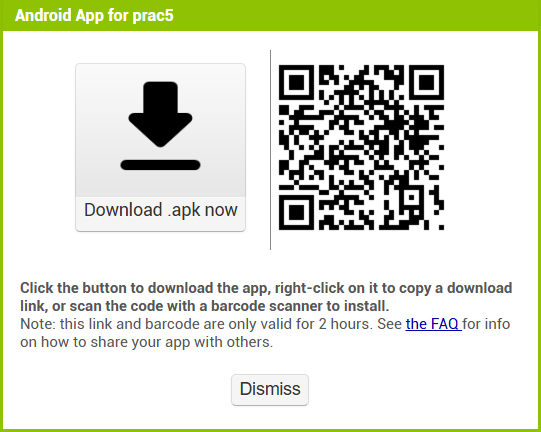
Save your project 🡪 Click on Build 🡪 Android App (.apk)



Click on connect 🡪 AI companion

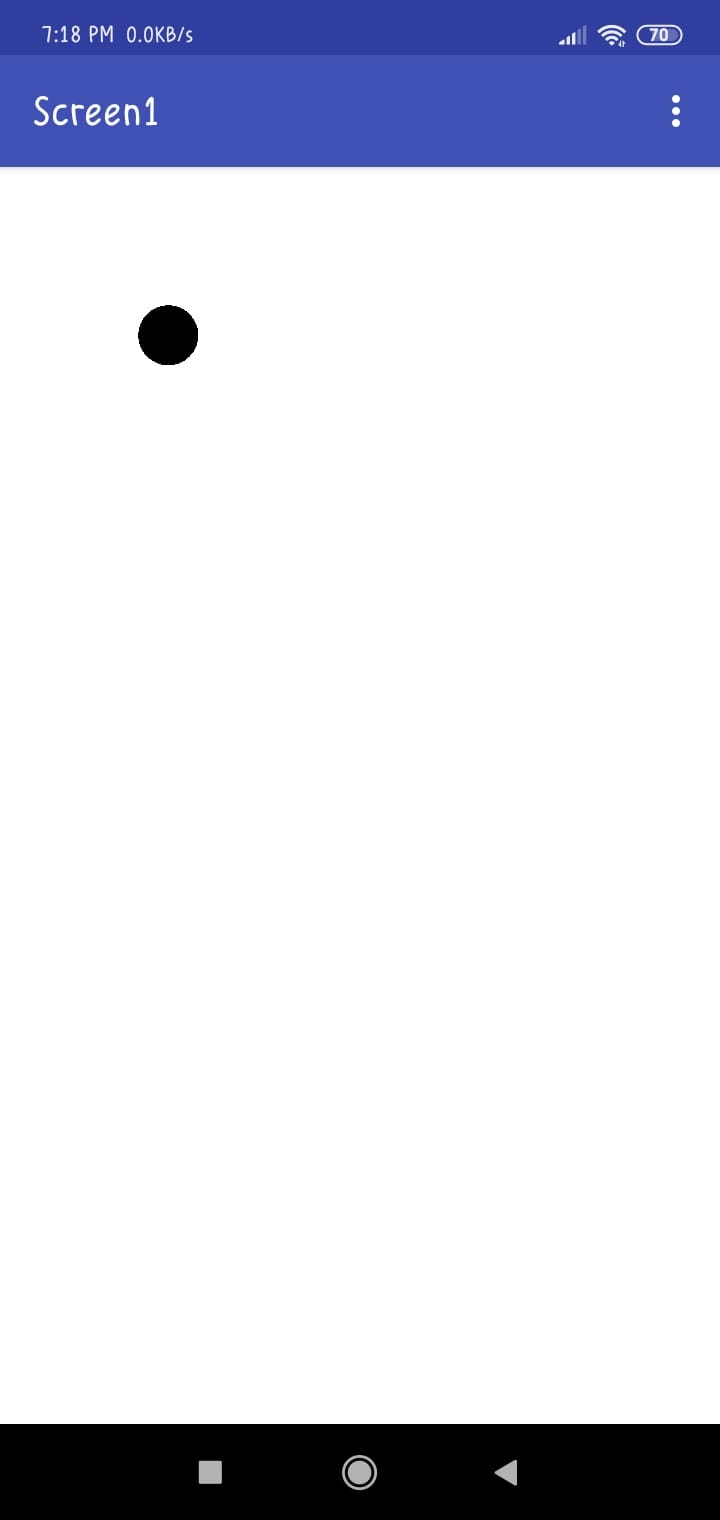


Open MIT App inventor on mobile 🡪 scan this QR code on mobile



Download prac5 apk and open it to see the output

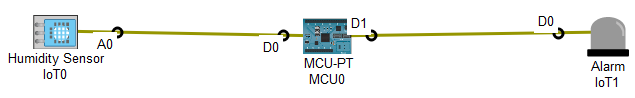




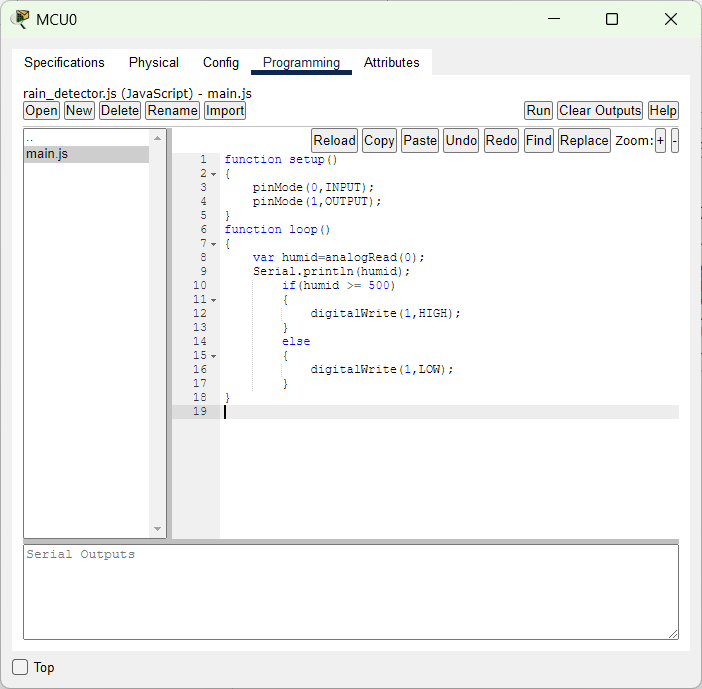
**Practical 6**

Aim: Develop an IoT application that will raise an alarm whenever with going to rain outside based on the weather prediction data.

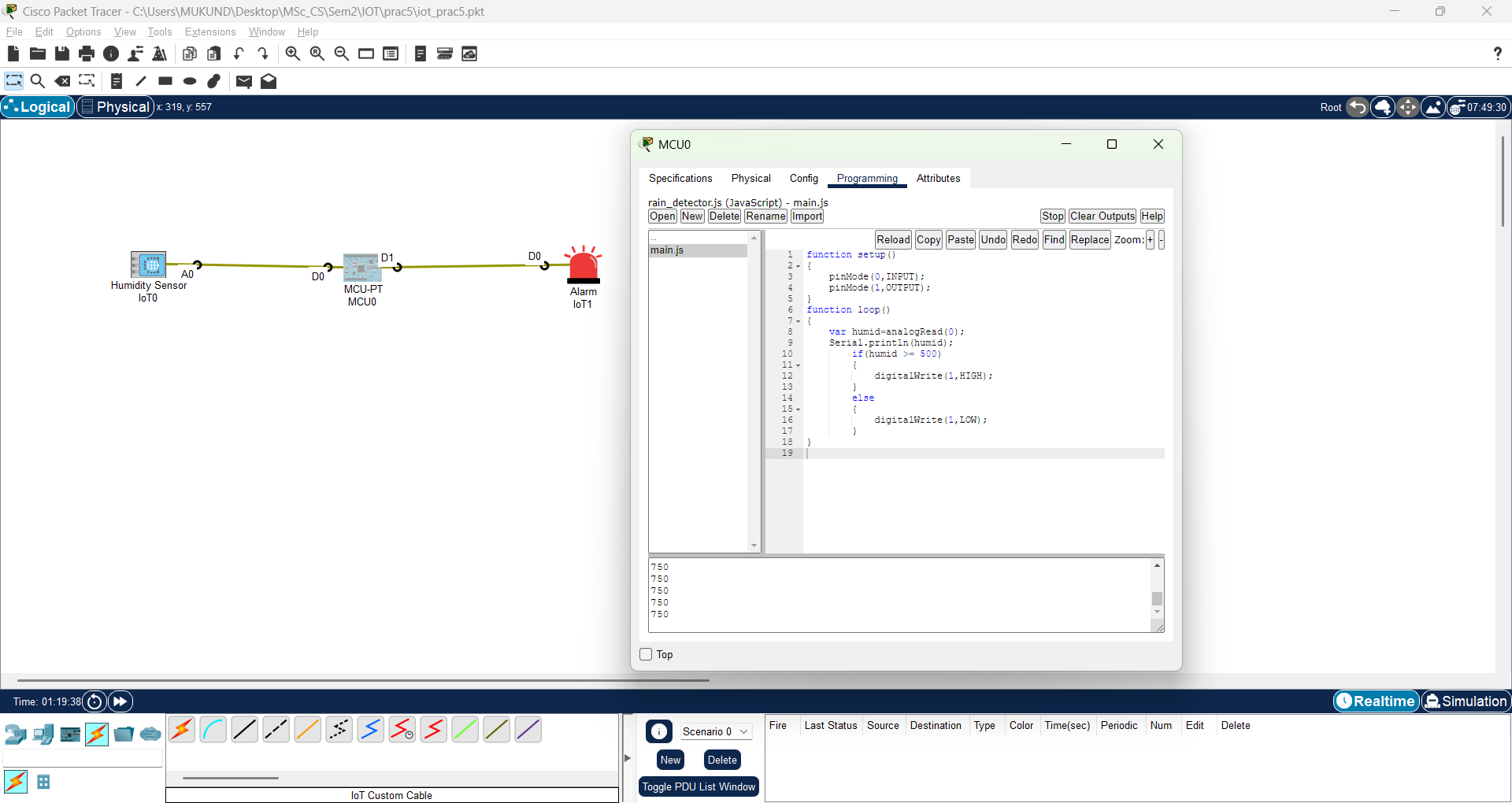
Design the circuit in cisco packet tracer



Click on MCU-PT 🡪 Programming 🡪 click on New 🡪 Create a new file as rain\_detector.js 🡪 Write following program



Click on Run 🡪 see the output



Here, we can see when humidity level is greater than 500 i.e., when it is raining outside the Alarm is turned ON.

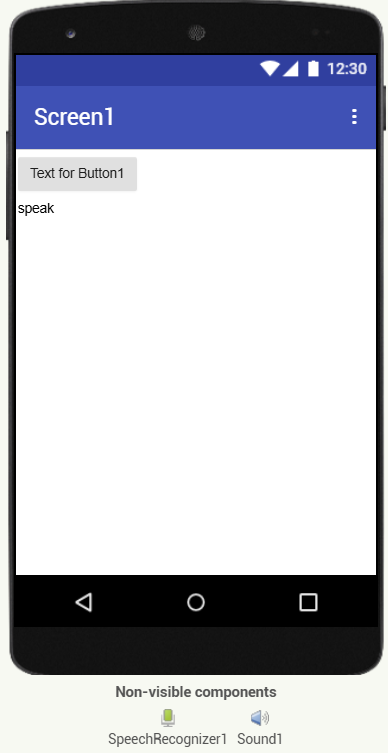
**Practical 7**

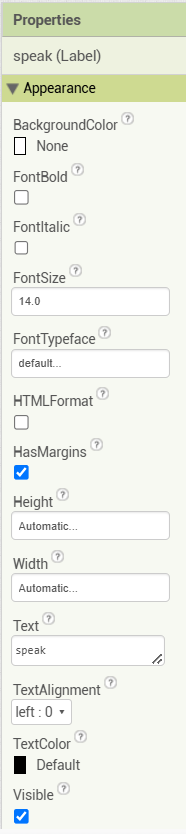
Aim: Deploy an IoT application which will alert you by beeping or vibrating your phone whenever you get someone call your name.

Go to MIT App Inventor 🡪 Projects 🡪 Start new project 🡪 give project name 🡪 click OK

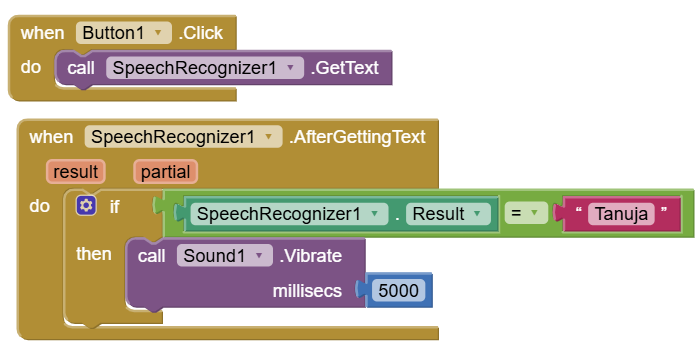
Go to User Interface 🡪 select Button 🡪 then select Label

Go to Media 🡪 select Speech Recognizer 🡪 then select Sound





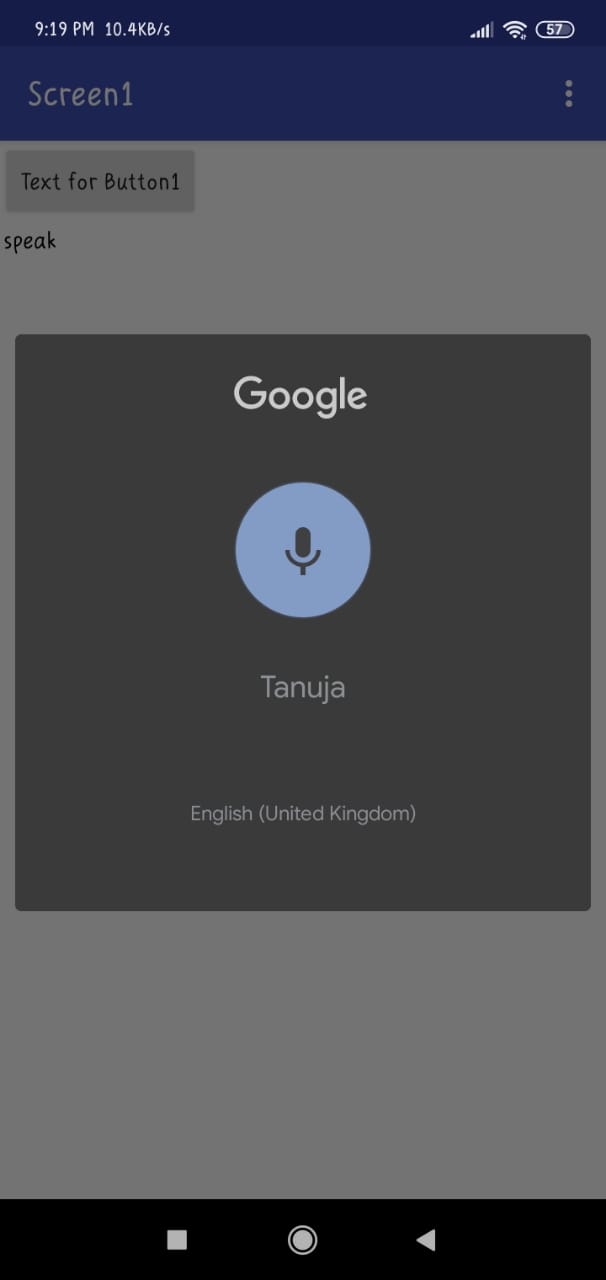
Draw the block diagram as follows



Click on Projects 🡪 save project

Click on Build 🡪 Android app (.apk) 🡪 then scan the code and get output 🡪 click on button 🡪 As you speak your name your phone gets vibrate





**Practical 8**

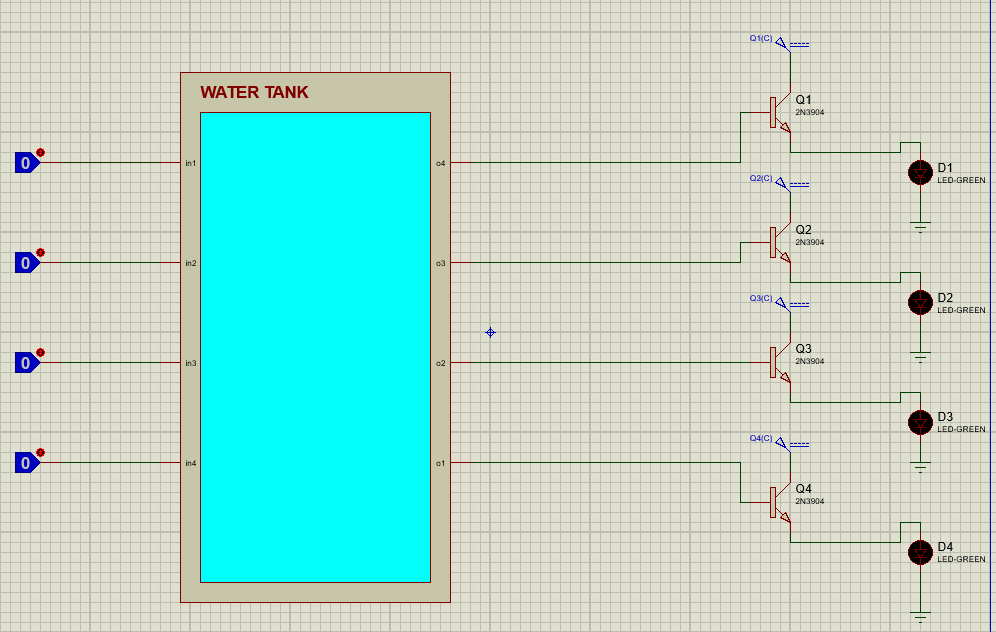
Aim: Develop an IoT application for monitoring water levels in tanks and automatically start the motor to fill the tank if the level goes below critical level.

Proteus:

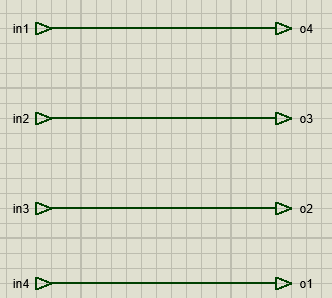
Create new project 🡪 add following devices

* 2N3904 – 4 placed, Silicon NPN Low Power High Frequency Bipolar Transistor
* LED GREEN – 4 placed, Animated LED model (Green)
* LOGICTOGGLE – 4 placed, Logic State Source (Momentary Action)

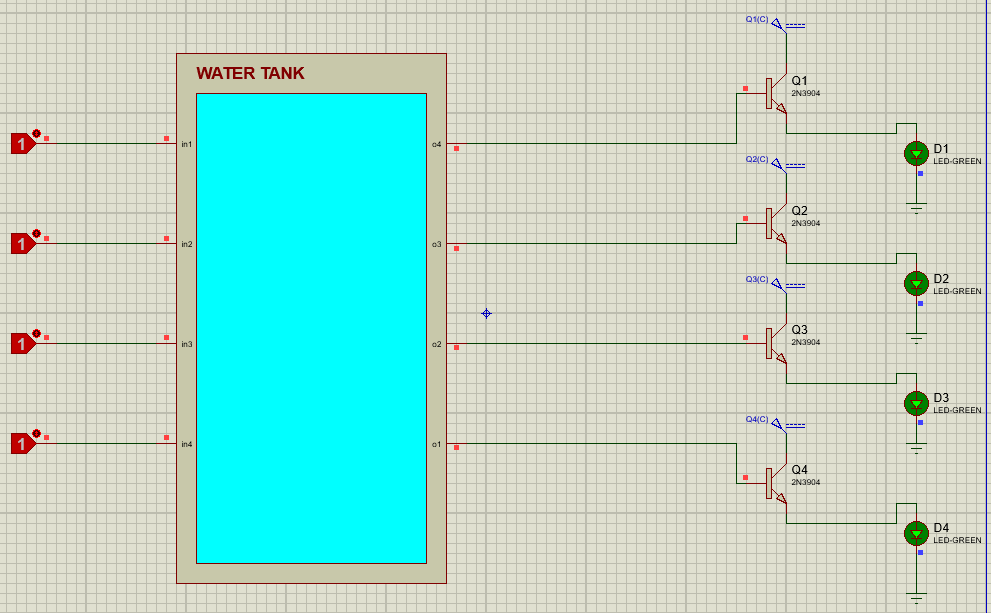
Design the circuit:



Right Click on Water Tank 🡪 go to child sheet 🡪 design the logic:



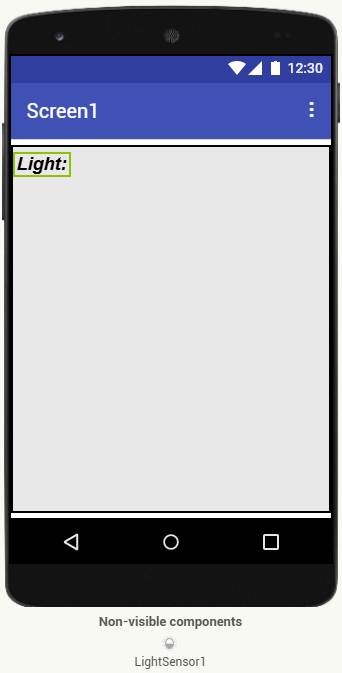
Output: Start the simulation



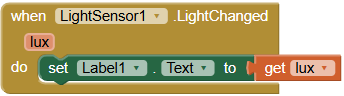
**Practical 9**

Aim: Develop an IoT module to which measure the intensity of light and send the same to your PC/phone.

MIT App inventor



Block diagram



Output:

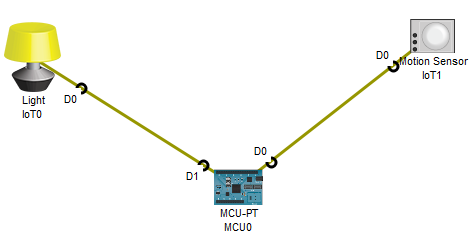




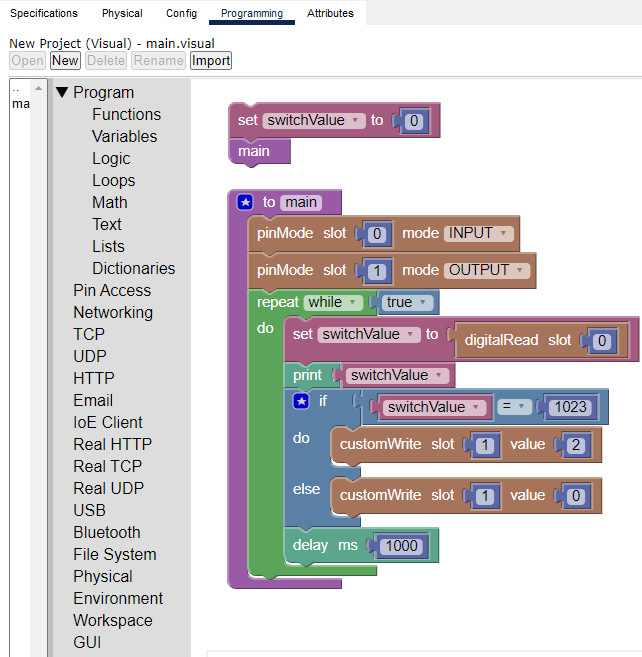
**Practical 10**

Aim: Develop an IoT application for Motion detection.

Open Cisco Packet Tracer 🡪 design the circuit 🡪 take components Motion sensor, MCU-PT and Light 🡪 D0 pin of MCU connect with D0 pin of Motion sensor, D1 pin of MCU connect with D0 pin of light using IOT custom cable.



Click on MCU-PT 🡪 click on Programming 🡪 New 🡪 give name as “motion\_detection.visual” 🡪 click Create



Click on run button and run the program 🡪 Click Alt and move mouse on the motion sensor and view the output of Light.

Output:

