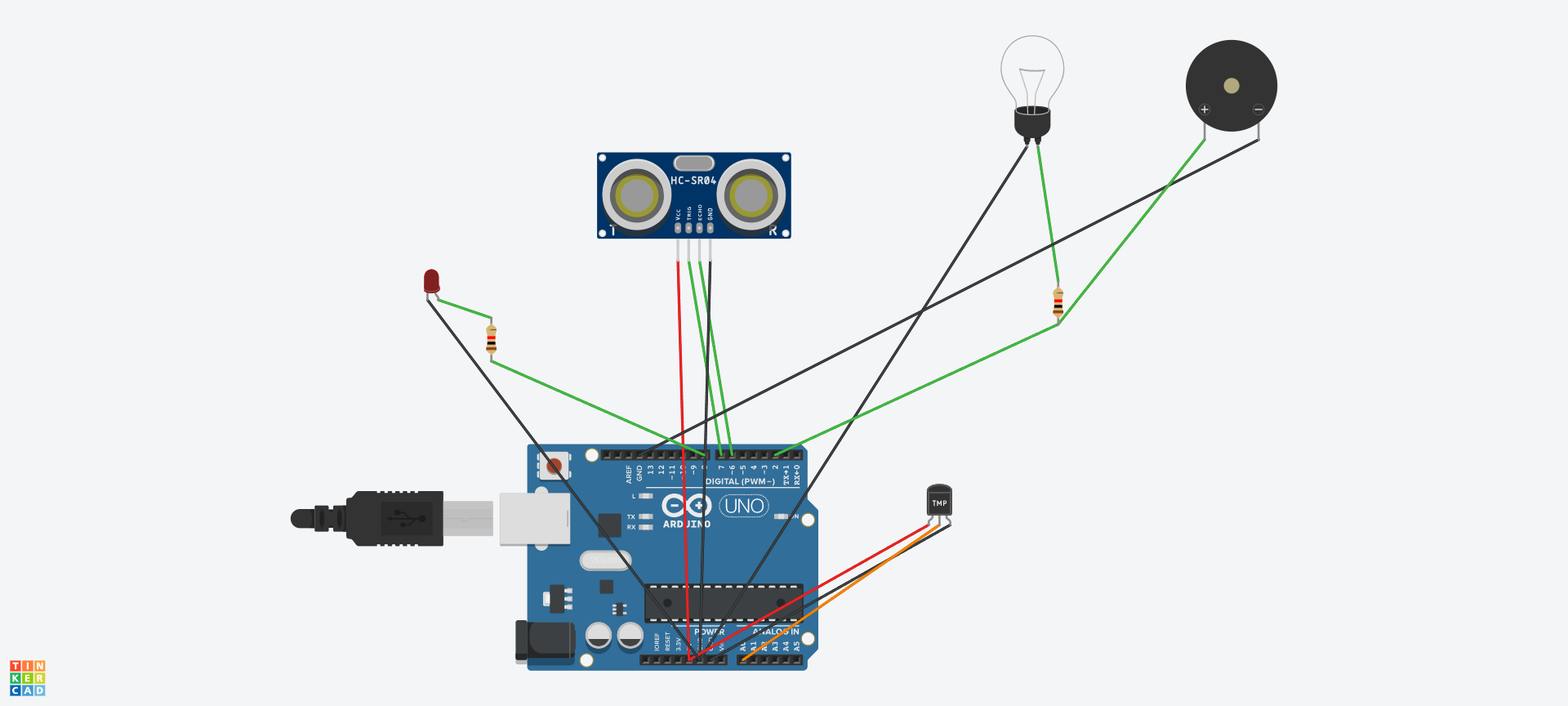
**ASSIGNMENT-1**

SMART HOME AUTOMATION

**Make a Smart Home in Tinkercad, using 2+ sensors, Led, Buzzer in single code and circuit.**



|  |
| --- |
| #include <EEPROM.h> |
|  |  | #include <IRremote.h> |
|  |  | #include <SoftwareSerial.h> |
|  |  | SoftwareSerial BT\_Serial(2, 3); // RX, TX |
|  |  |  |
|  |  | const int RECV\_PIN=A5; |
|  |  | IRrecv irrecv(RECV\_PIN); |
|  |  | decode\_results results; |
|  |  |  |
|  |  | #define touch1 A0 // Power Button Pin in |
|  |  | #define touch2 A1 // Light Button Pin in |
|  |  | #define touch3 A2 // Lamp Button Pin in |
|  |  | #define touch4 A3 // Socket Button Pin in |
|  |  | #define touch5 A4 // Fan Button Pin in |
|  |  |  |
|  |  | #define Relay1 4 // Load1 Pin Out |
|  |  | #define Relay2 5 // Load2 Pin Out |
|  |  | #define Relay3 6 // Load3 Pin Out |
|  |  | #define Relay4 7 // Load4 Pin Out |
|  |  |  |
|  |  | char bt\_data; // variable to receive data from the serial port |
|  |  |  |
|  |  | int load1, load2, load3, load4, power; |
|  |  | int Stop=0, timer=0; |
|  |  |  |
|  |  | void setup(){ |
|  |  | Serial.begin(9600); |
|  |  | BT\_Serial.begin(9600); |
|  |  |  |
|  |  | irrecv.enableIRIn(); // Start the receiver |
|  |  | irrecv.blink13(true); |
|  |  |  |
|  |  | pinMode(touch1, INPUT); |
|  |  | pinMode(touch2, INPUT); |
|  |  | pinMode(touch3, INPUT); |
|  |  | pinMode(touch4, INPUT); |
|  |  | pinMode(touch5, INPUT); |
|  |  |  |
|  |  | pinMode(Relay1, OUTPUT); digitalWrite(Relay1, 1); |
|  |  | pinMode(Relay2, OUTPUT); digitalWrite(Relay2, 1); |
|  |  | pinMode(Relay3, OUTPUT); digitalWrite(Relay3, 1); |
|  |  | pinMode(Relay4, OUTPUT); digitalWrite(Relay4, 1); |
|  |  |  |
|  |  | load1 = EEPROM.read(1); |
|  |  | load2 = EEPROM.read(2); |
|  |  | load3 = EEPROM.read(3); |
|  |  | load4 = EEPROM.read(4); |
|  |  |  |
|  |  | power = EEPROM.read(5); |
|  |  | delay(500); |
|  |  | } |
|  |  |  |
|  |  | void loop() { |
|  |  |  |
|  |  | if(digitalRead(touch1)==1 && Stop==0){Stop=1; |
|  |  | load1 = !load1; |
|  |  | EEPROM.write(1, load1); |
|  |  | delay(100); |
|  |  | } |
|  |  |  |
|  |  | if(digitalRead(touch2)==1 && Stop==0){Stop=1; |
|  |  | load2 = !load2; |
|  |  | EEPROM.write(2, load2); |
|  |  | delay(100); |
|  |  | } |
|  |  |  |
|  |  | if(digitalRead(touch3)==1 && Stop==0){Stop=1; |
|  |  | load3 = !load3; |
|  |  | EEPROM.write(3, load3); |
|  |  | delay(100); |
|  |  | } |
|  |  |  |
|  |  | if(digitalRead(touch4)==1 && Stop==0){Stop=1; |
|  |  | load4 = !load4; |
|  |  | EEPROM.write(4, load4); |
|  |  | delay(100); |
|  |  | } |
|  |  |  |
|  |  | if(digitalRead(touch5)==1 && Stop==0){Stop=1; |
|  |  | power = !power; |
|  |  | EEPROM.write(5, power); |
|  |  | delay(100); |
|  |  | } |
|  |  |  |
|  |  | if(digitalRead(touch1)==0 && digitalRead(touch2)==0 && digitalRead(touch3)==0 && digitalRead(touch4)==0 && digitalRead(touch5)==0){Stop=0;} |
|  |  |  |
|  |  | if(irrecv.decode(&results)){ |
|  |  | Serial.println(results.value,HEX); |
|  |  |  |
|  |  | if(results.value==0xFFB04F){ // For Load1 On |
|  |  | load1 = 0; |
|  |  | EEPROM.write(1, load1); |
|  |  | } |
|  |  | else if(results.value==0xFF6897){ // For Load1 Off |
|  |  | load1 = 1; |
|  |  | EEPROM.write(1, load1); |
|  |  | } |
|  |  |  |
|  |  | else if(results.value==0xFF7A85){ // For Load2 On |
|  |  | load2 = 0; |
|  |  | EEPROM.write(2, load2); |
|  |  | } |
|  |  | else if(results.value==0xFF30CF){ // For Load2 Off |
|  |  | load2 = 1; |
|  |  | EEPROM.write(2, load2); |
|  |  | } |
|  |  |  |
|  |  | else if(results.value==0xFF5AA5){ // For Load3 On |
|  |  | load3 = 0; |
|  |  | EEPROM.write(3, load3); |
|  |  | } |
|  |  | else if(results.value==0xFF10EF){ // For Load3 Off |
|  |  | load3 = 1; |
|  |  | EEPROM.write(3, load3); |
|  |  | } |
|  |  |  |
|  |  | else if(results.value==0xFF52AD){ // For Load4 On |
|  |  | load4 = 0; |
|  |  | EEPROM.write(4, load4); |
|  |  | } |
|  |  | else if(results.value==0xFF42BD){ // For Load4 Off |
|  |  | load4 = 1; |
|  |  | EEPROM.write(4, load4); |
|  |  | } |
|  |  |  |
|  |  | else if(results.value==0xFFE21D){ // For Power On |
|  |  | power = 0; |
|  |  | EEPROM.write(5, power); |
|  |  | } |
|  |  | else if(results.value==0xFFA25D){ // For Power Off |
|  |  | power = 1; |
|  |  | EEPROM.write(5, power); |
|  |  | } |
|  |  |  |
|  |  | irrecv.resume(); // Receive the next value |
|  |  | delay(100); |
|  |  | } |
|  |  |  |
|  |  | if(BT\_Serial.available()>0){bt\_data = BT\_Serial.read();} |
|  |  |  |
|  |  | if(bt\_data == 'A'){load1=0;EEPROM.write(1, load1);} |
|  |  | if(bt\_data == 'a'){load1=1;EEPROM.write(1, load1);} |
|  |  |  |
|  |  | if(bt\_data == 'B'){load2=0;EEPROM.write(2, load2);} |
|  |  | if(bt\_data == 'b'){load2=1;EEPROM.write(2, load2);} |
|  |  |  |
|  |  | if(bt\_data == 'C'){load3=0;EEPROM.write(3, load3);} |
|  |  | if(bt\_data == 'c'){load3=1;EEPROM.write(3, load3);} |
|  |  |  |
|  |  | if(bt\_data == 'D'){load4=0;EEPROM.write(4, load4);} |
|  |  | if(bt\_data == 'd'){load4=1;EEPROM.write(4, load4);} |
|  |  |  |
|  |  | if(bt\_data == 'E'){power=0;EEPROM.write(5, power);} |
|  |  | if(bt\_data == 'e'){power=1;EEPROM.write(5, power);} |
|  |  |  |
|  |  | bt\_data = '0'; |
|  |  |  |
|  |  | if(power==1){ |
|  |  | digitalWrite(Relay1, 1); |
|  |  | digitalWrite(Relay2, 1); |
|  |  | digitalWrite(Relay3, 1); |
|  |  | digitalWrite(Relay4, 1); |
|  |  | }else{ |
|  |  | digitalWrite(Relay1, load1); |
|  |  | digitalWrite(Relay2, load2); |
|  |  | digitalWrite(Relay3, load3); |
|  |  | digitalWrite(Relay4, load4); |
|  |  | } |
|  |  |  |
|  |  | timer = timer+1; |
|  |  | if(timer>9){ timer=0; |
|  |  | BT\_Serial.print(power); //send distance to MIT App |
|  |  | BT\_Serial.print(";"); |
|  |  | BT\_Serial.print(load1); //send distance to MIT App |
|  |  | BT\_Serial.print(";"); |
|  |  | BT\_Serial.print(load2); //send distance to MIT App |
|  |  | BT\_Serial.print(";"); |
|  |  | BT\_Serial.print(load3); //send distance to MIT App |
|  |  | BT\_Serial.print(";"); |
|  |  | BT\_Serial.print(load4); //send distance to MIT App |
|  |  | BT\_Serial.println(";"); |
|  |  | } |
|  |  |  |
|  |  | delay(50); |
|  |  | } |