Ex1:led blink

```
void setup()
{
pinMode(3, OUTPUT); pinMode(7,
OUTPUT); pinMode(11, OUTPUT);
Serial.begin(9600);
}
void loop()
{
  digitalWrite(3, HIGH); delay(1000);
  digitalWrite(3, LOW); delay(1000);
  digitalWrite(7, HIGH); delay(1000);
  digitalWrite(7, LOW); delay(1000);
  digitalWrite(11, HIGH); delay(1000);
  digitalWrite(11, LOW); delay(1000);
}
```

Ex 2:analog sensor

```
#define SENSOR_PIN A0 void setup(){
Serial.begin(9600);
}
void loop(){
  int analogValue = analogRead(SENSOR_PIN);
 float voltage = analogValue * (5.0/1023.0);
float temperature = voltage * 100.0;
  Serial.print("Analog value:");
  Serial.print(analogValue);
  Serial.print(" | Temperature (C) : ");
  Serial.println(temperature); delay(1000);
}
```

Ex3:servo motor

```
#include <Servo.h> Servo
myservo;
int potpin = 0; int val;
void setup() { myservo.attach(9);
}
void loop() {
  val = analogRead(potpin);
  val = map(val, 0, 1023, 0, 180);
   myservo.write(val);
  delay(15);
}
```

Ex4:UART

```
Transmitter void setup()
{
  Serial.begin(9600);
}
void loop() {
  Serial.println("Hello from Arduino 1!");
   delay(1000);
}
Receiver:
String receivedData = "";
void setup() {
  Serial.begin(9600);
 Serial.println("Arduino 2 Ready to Receive...");
}
void loop() {
  if (Serial.available() > 0) {
receivedData = Serial.readStringUntil('\n');
 Serial.print("Received: ");
Serial.println(receivedData);  }}
```

EX-5: HTTP req

```
#include <WiFi.h> #include
<HTTPClient.h>
const char* ssid= "Wokwi-GUEST";
const char* password = "";
const char* serverName = "http://httpbin.org/get";
void setup() { Serial.begin(115200);
  delay(1000);
 Serial.println("Connecting to WiFi...");
WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
   delay(500);
     Serial.print(".");
  }
  Serial.println("\nConnected to WiFi");
if (WiFi.status() == WL_CONNECTED) {
 HTTPClient http; http.begin(serverName);
 int httpResponseCode = http.GET();
if (httpResponseCode > 0) {
Serial.print("HTTP Response code: ");
Serial.println(httpResponseCode); String payload =
http.getString(); Serial.println("Response:");
Serial.println(payload);
} else {
Serial.print("Error code: ");
Serial.println(httpResponseCode);
     }
     http.end(); // Free resources
  } }
```

EX-6 :interfacing PC

```
#include <Adafruit_LiquidCrystal.h>
#define trigPin 2
#define echoPin 3 long
duration;
int distance;
Adafruit_LiquidCrystal lcd_1(0);
 void setup() {
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
Serial.begin(9600); lcd_1.begin(16,2);
lcd_1.print("Sensor Value :");
}
void loop() {
digitalWrite(trigPin, LOW);
delayMicroseconds(5);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distance = duration * 0.034 / 2;
 lcd_1.setCursor(0,1);
 lcd_1.print(distance);
   cd_1.print("cm");
   delay(50);
}
```

Exp7: multitask

```
#include <Arduino.h>
 int sensorValue = 0;
void TaskReadSensor(void *pvParameters)
{ while (1) {
sensorValue = analogRead(34);
 vTaskDelay(500 / portTICK_PERIOD_MS);
   } }
void TaskDisplay(void *pvParameters) {
while (1) {
Serial.print("Potentiometer Value: ");
Serial.println(sensorValue);
vTaskDelay(1000 / portTICK_PERIOD_MS);
   } }
void setup() { Serial.begin(115200);
   1 xTaskCreate(
      TaskReadSensor,
       "ReadSensor",
       1000,
      NULL,
      NULL);
   xTaskCreate( TaskDispl
   ay, "Display",   1000,
      NULL, 1,
      NULL);
}
```

Exp 8: Bluetooth

```
#include <BLEDevice.h>
#include <BLEServer.h>
#include <BLEUtils.h>
bool connected = false;
class MyCallbacks :
public BLEServerCallbacks {
  void onConnect(BLEServer *pServer)
{ connected = true; }
  void onDisconnect(BLEServer *pServer)
 { connected = false; }   };
void setup() {
  Serial.begin(115200);
  BLEDevice::init("ESP32-BLE");
  BLEServer *server = BLEDevice:
:createServer();
  server->setCallbacks(new MyCallbacks());
  BLEService *service = server->
  createService("1234");
  BLECharacteristic *charac =
service->createCharacteristic("5678",
    BLECharacteristic::PROPERTY_READ |
     BLECharacteristic::PROPERTY_WRITE);
  charac->setValue("Hello from ESP32");
  service->start();
  server->getAdvertising()->start();
  Serial.println("BLE advertising started...");  }
void loop() {
  Serial.println(connected ? "Device connected" :
"Waiting for device...");
  delay(2000);
    }
```