INSTITUTO TECNOLOGICO DE AGUASCALIENTES

Carrera: Tecnologías de la Información y Comunicación.

Materia: Tecnologías inalámbricas

Horario: 12:00 – 1:00

Docente: Ricardo Alejandro Rodríguez Jiménez

Nombre del Trabajo:

Fecha de Entrega: 19 de febrero de 2026

Alumno:

Luis Arturo Cruz Coria 23151197

Axel Johab Rodríguez Ortiz 23151212

Link Github: https://github.com/arturocruz-0503/ProyectoBLE

Investigación BLE vs Bluetooth.

Por un lado, tenemos al Bluetooth, este es una tecnología de comunicación inalámbrica de corto alcance que permite transmitir datos entre dispositivos sin necesidad de cables, utilizando la banda ISM de 2.4 GHz. Fue diseñada para reemplazar conexiones físicas como cables USB, audio o serial y se utiliza en audífonos, bocinas, teclados, etc.

Mientras que el BLE (Bluetooth Low Energy) es una versión optimizada de Bluetooth diseñada para consumir muy poca energía, ideal para dispositivos que funcionan con batería durante largos periodos, se utiliza en relojes inteligentes, dispositivos médicos, proyectos con ESP32, etc.

Característica	**Bluetooth Clásico**	**BLE**
Consumo de energía	Alto	Muy bajo
Velocidad de transmisión	Mayor (hasta ~3 Mbps)	Menor (1–2 Mbps aprox.)
Uso típico	Audio y archivos	Sensores y datos pequeños
Tiempo de conexión	Más lento	Muy rápido
Duración de batería	Menor	Mucho mayor
Comunicación continua	Sí	No siempre (envíos cortos y periódicos)

En pocas palabras, el bluetooth mantiene una conexión constante, es ideal para la transmisión continua, tiene un mayor consumo energético, pero tiene una mayor velocidad. El BLE envía pequeños paquetes de datos, permanece en modo reposo la mayor parte del tiempo, se activa solo cuando necesita enviar información y tiene una menor velocidad.

Componentes utilizados.

- Placa ESP32 con conexión Bluetooth.
- Celular Android.
- 3 LEDs (verde, amarillo y azul)

- Protoboard
- Jumpers

Objetivo del proyecto.

Diseñar e implementar un sistema de control inalámbrico de LEDs haciendo uso de la comunicación BLE entre un ESP32 y una aplicación móvil, a través de dicha aplicación móvil se controlarán LEDs de manera inalámbrica que estarán conectados a la placa ESP32

Descripción del proyecto.

El proyecto consiste en desarrollar un sistema de comunicación inalámbrica utilizando BLE entre una placa de desarrollo ESP32 (o alguna parecida) y una aplicación móvil creada en MIT App Inventor.

Mediante la aplicación creada con App Inventor se podrá realizar lo siguiente:

- Buscar dispositivos Bluetooth cercanos.
- Mostar una lista de los dispositivos detectados.
- Conectar la placa ESP32 a un dispositivo móvil usando la aplicación móvil creada en App Inventor.
- Enviar comandos (cadenas de texto) del dispositivo móvil al ESP32 con los cuales encenderá LEDs según el botón apretado.

Conexiones

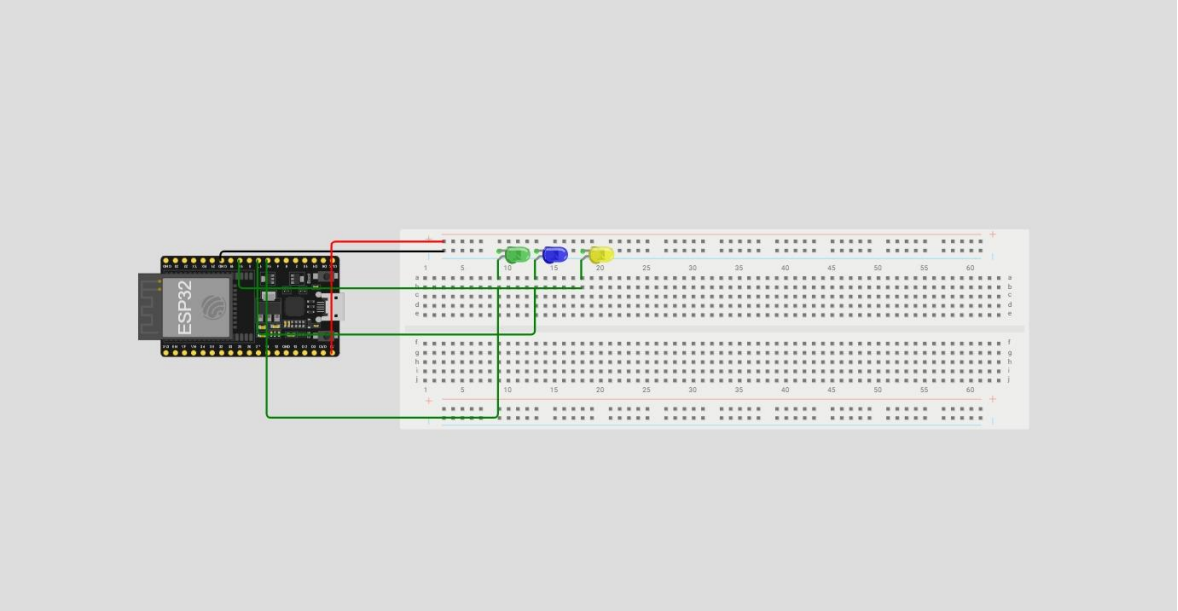
LED	GPIO ESP32	Descripción
Verde	16	Listo para conectar / Control LED 1
Azul	17	Bluetooth conectado / Control LED 2
Amarillo	18	Bluetooth desconectado / Control LED 3

Comandos

Comando	Acción	GPIO
A	Enciende LED Verde	16
a	Apaga LED Verde	16
B	Enciende LED Azul	17

b	Apaga LED Azul	17
C	Enciende LED Rojo	18

Diagrama



Código

```
#include <BLEDevice.h>

#include <BLEServer.h>

#include <BLEUtils.h>


// ID para identificar el dispsitivo y mandar las acciones desde mit app inventor //

#define SERVICE_UUID        "12345678-1234-1234-1234-1234567890ab"

#define CHARACTERISTIC_UUID "abcd1234-5678-1234-5678-abcdef123456"


// numero de pines de los leds //

#define LED1_PIN 16

#define LED2_PIN 17

#define LED3_PIN 18
```

```
bool dispositivoConectado = false;

void setAll(bool on) {
  digitalWrite(LED1_PIN, on ? HIGH : LOW);
  digitalWrite(LED2_PIN, on ? HIGH : LOW);
  digitalWrite(LED3_PIN, on ? HIGH : LOW);
}

class MyServerCallbacks : public BLEServerCallbacks {
  void onConnect(BLEServer* pServer) override {
    dispositivoConectado = true;
    Serial.println("Conectado");
  }
  void onDisconnect(BLEServer* pServer) override {
    dispositivoConectado = false;
    Serial.println("Desconectado");
    pServer->getAdvertising()->start(); // desconcetarse de el esp32
  }
};

class MyCallbacks : public BLECharacteristicCallbacks {

void onWrite(BLECharacteristic* pCharacteristic) override {
  String rx = pCharacteristic->getValue();   // ahora lo recibe como texto para que cuando en la aplicacion se ejecute un boton, el esp32 lo tome y pueda hacer la accion segun correponda con el boton que se presiono
```

```
  if (rx.length() == 0) return;

  char c = rx.charAt(0);                // tomamos el primer caracter o letra que da la aplicacion o el boton presionado
  Serial.print("Recibido: ");
  Serial.println(c);

  switch (c) {
    case 'A': digitalWrite(LED1_PIN, HIGH); break;
    case 'a': digitalWrite(LED1_PIN, LOW);  break;
    case 'B': digitalWrite(LED2_PIN, HIGH); break;
    case 'b': digitalWrite(LED2_PIN, LOW);  break;
    case 'C': digitalWrite(LED3_PIN, HIGH); break;
    case 'c': digitalWrite(LED3_PIN, LOW);  break;
    default:
      Serial.println("Comando no reconocido");
      break;
  }
}

};

void setup() {
  Serial.begin(115200);

  pinMode(LED1_PIN, OUTPUT);
```

```
  pinMode(LED2_PIN, OUTPUT);
  pinMode(LED3_PIN, OUTPUT);
  setAll(false);

  BLEDevice::init("ESP32 sin seven");
  BLEServer *pServer = BLEDevice::createServer();
  pServer->setCallbacks(new MyServerCallbacks());

  BLEService *pService = pServer->createService(SERVICE_UUID);

  BLECharacteristic *pCharacteristic = pService->createCharacteristic(
    CHARACTERISTIC_UUID,
    BLECharacteristic::PROPERTY_WRITE    |
    BLECharacteristic::PROPERTY_WRITE_NR  | // por si la app escribe sin respuesta
    BLECharacteristic::PROPERTY_READ
  );

  pCharacteristic->setCallbacks(new MyCallbacks());

  pService->start();
  pServer->getAdvertising()->start();

  Serial.println("ESP32 listo (BLE)");
}

void loop() {
```

// no es necesario hacer nada aqui ya que el funcionamiento se hace afuera del loop porque desde ahi se puede hacer

}

Conclusiones

Luis Arturo Cruz Coria.

Gracias a la realización de esta práctica, profundicé mi conocimiento sobre cómo utilizar microcontroladores (en este caso la placa ESP32), así como mis conocimientos sobre la conexión de componentes electrónicos. De la misma manera, se comprendió de manera clara la diferencia entre Bluetooth y BLE, así como en qué casos es mejor usar alguno de los dos. En este proyecto, también, se implementaron nuestros conocimientos de programación para realizar la conexión entre la placa y un celular Android. Para agregar, este proyecto abrió nuestras posibilidades de cómo implementar proyectos en situaciones de la vida real para el IoT.

Axel Johab Rodríguez Ortiz

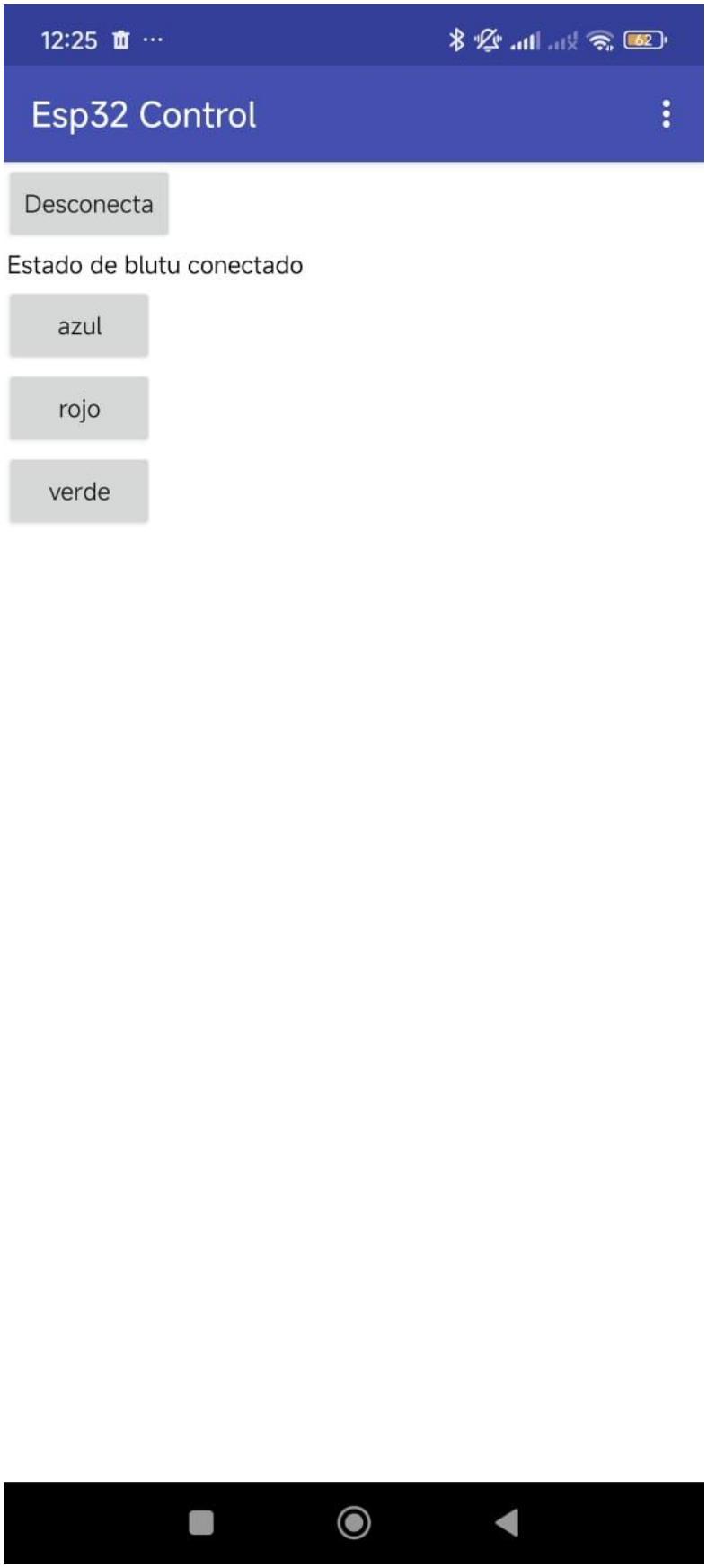
Este proyecto permitió al equipo fortalecer los conocimientos en torno al desarrollo de sistemas embebidos y la integración de tecnologías inalámbricas en entornos de bajo consumo. Se comprobó que, a través de plataformas de desarrollo accesibles como el ESP32 y protocolos de comunicación como Wi-Fi o BLE, es posible construir soluciones IoT escalables y funcionales sin requerir infraestructuras complejas. El sistema implementado cumplió satisfactoriamente con los objetivos establecidos, respondiendo de manera eficiente a las instrucciones enviadas desde la interfaz de usuario, lo que valida tanto el diseño de la arquitectura de software como la correcta configuración del hardware utilizado para la adquisición, procesamiento y transmisión de datos.

Evidencias

Capturas


5:45
0,23 kB/s
29
Esp32 Control
buscar blue
Desconecta
estado de blutu: desconctado
azul
ApagarAzul
rojo
ApagarRojo
verde
ApgarVerde


12:31
Esp32 Control
buscar blue
7F:80:0E:12:42:5F null -59
20:C7:A3:DC:C1:C5 null -59
2C:06:CA:71:1E:FD null -61
0C:20:DD:93:57:E2 null -61
03:D8:46:BB:D6:EC null -61
25:22:6E:47:D4:BB null -62
59:3C:AB:D9:B6:36 null -62
77:82:CD:EC:53:1D null -64
A4:F0:0F:74:A2:4A ESP sin seven -64
7D:50:8E:C9:12:13 null -65
33:37:FE:C0:90:5D null -66
4B:71:26:DB:E6:B1 null -67
F1:01:6A:1A:8D:C0 null -67
0E:B6:76:D4:25:60 null -67
41:42:67:CA:4A:EA Smartwatch -68
80:B5:4E:D7:EA:DD ESP32_S3_BLE -68
28:40:EB:A5:E1:9F null -68
1E:F7:5B:04:A5:69 null -69


12:25
Esp32 Control
Desconecta
Estado de blutu conectado
azul
rojo
verde

Video demostrativo

https://youtube.com/shorts/qsudXpdoDys?feature=share