**باسمه تعالی**

****

**امنیت در اینترنت اشیاء**

**دکتر احمدی**

**تمرین اول**

**پیاده سازی BLE با 32E**

**پوریا دادخواه**

**401201381**

در این تمرین قصد داریم با پیاده‌سازی یک سرور gatt ble روی esp32 و اضافه کردن قابلیت هندل کردن رویداد مرتبط با LED، برد خود را با گوشی مدیریت کرده و چراغ LED را روشن کنیم.

به این منظور ابتدا برای نصب ESP-IDF از Espressif استفاده کردیم و مراحل build، flash و monitor کد را از طریق command line خود esp-idf مطابق دستورالعمل [این لینک](https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/windows-setup.html" \l "get-started-windows-first-steps) انجام ‌می‌دهیم. ( از محیط vscode نیز صرفا به عنوان ادیتور کد خود استفاده کردیم.)

برای دستیابی به هدف تمرین فعلی، ساختار اصلی کد که پیاده‌سازی gatt sever بر روی esp می‌باشد را از مثال موجود در خود پوشه espressif استفاده کردیم که یک دمو از یک سرور را پیاده‌سازی کرده ولی event handler خاصی برای اقدامات پس از اتصال فراهم نکرده است. پس ما باید ساختار این کد را به گونه‌ای تغییر دهیم که attribute های مرتبط با led را به درستی شناسایی و تنظیم کرده که event handler متناسب داشته باشد.

کد اصلی تمرین با نام GATT\_LED می‌باشد که تغییر یافته کد gatts\_demo از example های espressif بوده و سایر هدر‌های مورد استفاده در کد را نیز در پوشه تمرین قرار داده‌ایم که برای اجرای مجدد کد مشکلی ایجاد نشود. (اما فایل build پروژه را حذف کردیم که حجم فایل ارسالی بیش از اندازه نشود.)

در ادامه توضیح کلی از کد GATT\_LED می‌دهیم و سپس به طور جزیی اختلافات اصلی و تغییراتی که نسبت به فایل demo دارد را بیان می‌کنیم. ( هم‌چنین با کامنت‌های داخل کد نیز شفافیت تغییرات ریز را سعی کردیم براورده کنیم)

نتیجه تست خروجی را نیز در قالب ویدیو در فایل تمرین ضمیمه کرده‌ایم.

کد GATT\_LED:

این کد یک سرور Bluetooth Low Energy (BLE) را بر روی یک ESP32 با استفاده از چارچوب ESP-IDF تنظیم می‌کند که بخش‌های مهم آن به صورت زیر هستند:

1. فایل‌ها و ماکروهای هدر:

- شامل فایل‌های هدر استاندارد و ESP-IDF.

- تعریف ماکروهایی مانند UUID سرویس، UUID ویژگی و نام دستگاه.

2. متغیرها و آرایه‌های سراسری:

- تعریف آرایه‌ها و متغیرهای استاتیک برای ذخیره اطلاعات مربوط به BLE مانند UUIDها، داده‌های تبلیغاتی و هندل‌های سرویس.

3. پیکربندی تبلیغات BLE

- پیکربندی داده‌های تبلیغات BLE، شامل نام دستگاه، ظاهر و UUID سرویس‌ها.

- تنظیم پارامترهای تبلیغات مانند فاصله زمانی و نوع.

4. ساختار پروفایل GATT

- تعریف یک ساختار برای نگهداری اطلاعات پروفایل GATT، شامل هندل‌های سرویس و ویژگی.

5. توابع Callback

- پیاده‌سازی توابع بازخوانی برای کنترل رویدادهای مختلف BLE مانند ثبت نام، خواندن، نوشتن، اتصال و قطع اتصال.

6. GATT Profile Event Handler

- پیاده‌سازی event handler برای رویدادهای پروفایل GATT، مدیریت اقدامات بر اساس رویدادهای مختلف.

7. تنظیم نوشتن و اجرای دستور نوشتن:

- تعریف توابع ( (example\_write\_event\_env و (example\_exec\_write\_event\_env) برای کنترل رویدادهای نوشتن و اجرای دستور نوشتن.

8. آماده‌سازی سرور GATT

- آماده‌سازی سرور GATT، تنظیم نام دستگاه و پیکربندی داده‌های تبلیغاتی.

- ثبت توابع بازخوانی برای رویدادهای )GAP پروفایل دسترسی عمومی) و GATT

9. کنترل رویداد GATT

- پیاده‌سازی توابع برای کنترل رویدادهای مختلف GATT مانند خواندن، نوشتن، اتصال و قطع اتصال.

10. تابع main:

- آماده‌سازی ) NVS ذخیره‌سازی غیر‌اختیاری) و کنترلر بلوتوث.

- پیکربندی یک پین GPIO برای کنترل یک LED.

- ثبت توابع بازخوانی و آماده‌سازی GATT و GAP BLE

پس از بررسی بخش‌های مهم این کد که اکثرا از gatt\_demo هم الگو برداری شده بودند اکنون به تغییرات اصلی کد شخصی‌سازی شده خود می‌پردازیم:

البته! بیایید به تفاوت‌های مفصل بین کد کنترل کننده LED BLE و کد ساده سرور GATT بپردازیم:

1. UUIDهای سرویس و ویژگی:

- کد ساده سرور GATT

- از UUIDهای عمومی (0000180a-0000-1000-8000-00805f9b34fb برای سرویس، 00002a29-0000-1000-8000-00805f9b34fb برای ویژگی) استفاده کرد.

- کد کنترل کننده LED BLE

- UUIDهای خاصی (GATTS\_SERVICE\_UUID، `GATTS\_CHAR\_UUID`، `GATTS\_DESCR\_UUID`) برای سرویس، ویژگی و توضیحات تعریف کرد.

2. پیکربندی تبلیغات:

- کد ساده سرور GATT

- داده‌های تبلیغاتی ساده بدون UUIDهای خاص.

-کد کنترل کننده LED BLE

- پیکربندی تبلیغات پیشرفته‌تر با استفاده از داده‌های خام یا UUIDهای خاص برای تبلیغ سرویس.

3. کنترل مقدار ویژگی:

- کد ساده سرور GATT

- متغیر ساده (gatts\_demo\_char\_value) برای ذخیره مقدار.

- تمرکز بر خواندن مقدار آن.

- کد کنترل کننده LED BLE

- از ویژگی خاصی (LED\_Status\_Att) برای ذخیره وضعیت LED استفاده می‌کند.

- رویدادهای نوشتن به این ویژگی را برای کنترل LED اداره می‌کند.

- بر اساس مقدار دریافتی، بین حالت‌های روشن و خاموش LED تفاوت قائل می‌شود.

4. اداره رویداد:

- کد ساده سرور GATT

- اداره رویداد ساده‌تر (gatts\_profile\_event\_handler) که عمدتاً با رویدادهای ثبت نام و خواندن سر و کار دارد.

- کد کنترل کننده LED BLE

- اداره رویداد پیچیده‌تر gatts\_profile\_LED\_event\_handler))

- به رویدادهای مختلف GATT مانند خواندن، نوشتن، اتصال و غیره پاسخ می‌دهد.

5. اداره توضیحات:

- کد ساده سرور GATT

- توضیحات را شامل نکرده است.

- کد کنترل کننده LED BLE

- یک توضیحات به ویژگی افزوده می‌کند که امکان اطلاعیه‌ها و نشانی‌ها را فراهم می‌کند.

6. اداره شروع و متوقف کردن تبلیغات:

- کد ساده سرور GATT

- اداره صریح رویدادهای شروع و متوقف کردن تبلیغات را شامل نمی‌شود.

- کد کنترل کننده LED BLE

- به اداره رویدادهای شروع و متوقف کردن تبلیغات (ESP\_GAP\_BLE\_ADV\_START\_COMPLETE\_EVTو (ESP\_GAP\_BLE\_ADV\_STOP\_COMPLETE\_EVT اداره می‌کند.

7. اداره MTU

- کد ساده سرور GATT

- اداره صریح MTU را نداشت.

- کد کنترل کننده LED BLE

- اداره رویدادهای حداکثر انتقال واحد (ESP\_GATTS\_MTU\_EVT) را شامل می‌شود.

- اندازه MTU محلی را به 500 تنظیم می‌کنیم.