ESLab HW4

游祖鈞 B06209040

November 8, 2021

Reports with several issues and solutions presented. You may clone the project and configure per the instruction.
GitHub link: https://github.com/NOOMA-42/NTUEE-ESLab/tree/hw4
*Note: commit of different homework are in different branch.

1 general approach

Rasberry Pi 3 serves as central and Mbed as peripheral. Implement BLE notification upon LED1 on-off, read write LED2 and read the button value. We firstly made the change with Mbed BLE button service example by adding more class header files and register them to the GATT service. Later we tested our program with LightBlue to avoid a more complex situation to debug Python and Mbed at the same time. Based on the UUID, we wrote the corresponding code to receive the data in Python. To better test these services, I added the functionality selection by interupt during testing, and all the testing can be done and up to user to choose. Figure 1 and 2

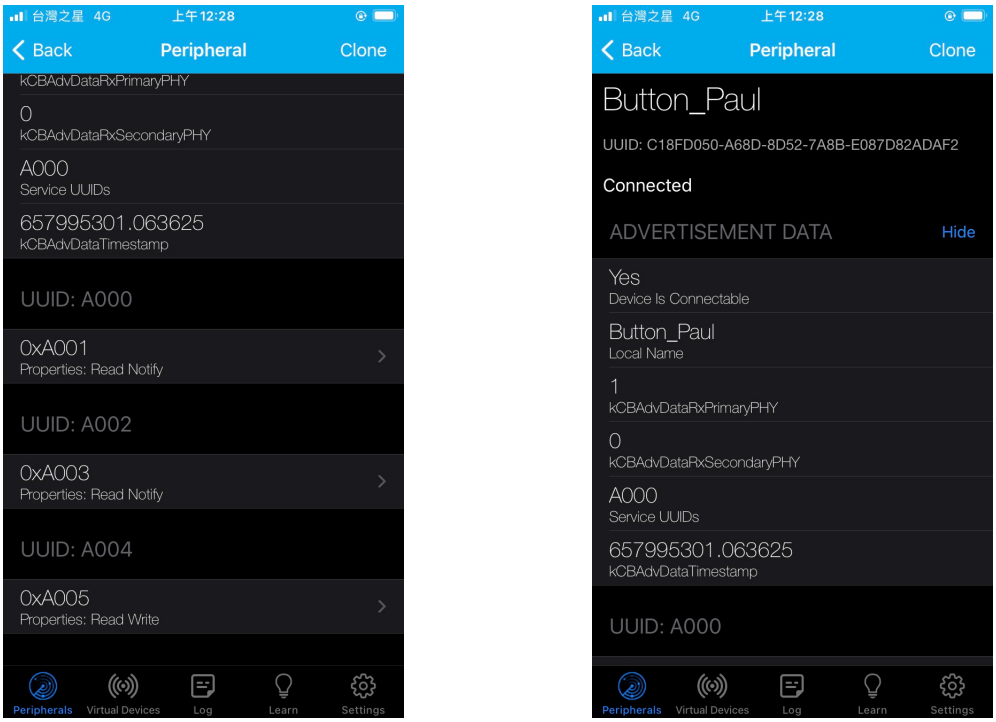
The service list is shown. we set the peripheral service UUID in Mbed and matched in in python.

1.1 (1) Write a Python or C/C++ BLE program (BLE Central) in a Linux host to communicate with STM32L4 IoT node as BLE Peripheral, controlling the LEDs and reading the Buttons.

We added LED header and register 2 instances: LED1 for a read-notify alive LED and LED2 for a read-write actuated LED2. The button waiting and notify sometimes shows I pressed sometimes and return the value. Without pressing it's still waiting. Figure 3 and 4

1.2 (2) The STM32L4 IoT node supports notifications or indications. Demo your BLE connections with the notification functionality.

This shows the alive LED, LED2 turning on and off. Figure 5



(a) (b)

Figure 1: Service Service List

```
class ButtonService {
public:
    const static uint16_t BUTTON_SERVICE_UUID              = 0xA000;
    const static uint16_t BUTTON_STATE_CHARACTERISTIC_UUID = 0xA001;
```

Figure 2: Mbed UUID

```
Getting LED2...

Test Service, Select 1 for LED2 toggle, 2 for Button Notification, 3 for LED1 notification, Ctrl+C to switch, -1 or press Ctrl+D for leaving
2
Waiting Button...
Waiting Button...
13

Test Button Notification
13

Test Button Notification
Waiting Button...
Waiting Button...
```

Figure 3: Button Notification

```
Getting LED2...
Test Service, Select 1 for LED2 toggle, 2 for Button Notification, 3 for LED1 notification, -1 or press Ctrl+D for leaving
1
Enter led state:
1
Enter led state:
```

Figure 4: Set LED1 state

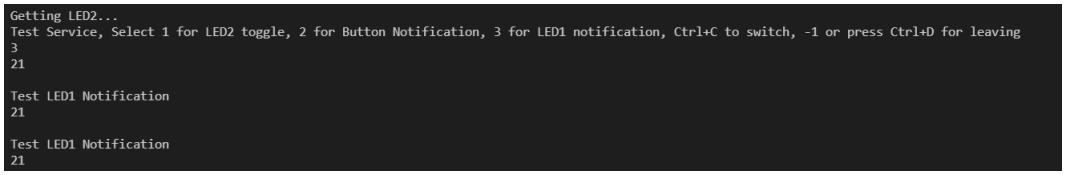


Figure 5: LED2 Notification

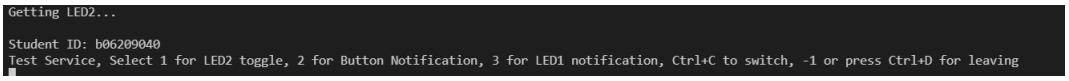


Figure 6: Student ID

1.3 (3) Add a service in STM32L4 IoT node so that you can provide your student id as the data value to be read by a BLE client (your Python/C/C++ program in a Linux host).

This shows the initial student ID value. Figure 6

1.4 (4) Note the following Mbed OS APIs used in the STM32 IoT Node Program. Explain the main purpose of the APIs.

1. Eventqueue: With this, we move events out of interrupt context (deferred execution of time consuming or non-ISR safe operations). We postpone the execution of a code sequence from an interrupt handler to a user context. Like blink, button state update. Therefore, the code can finish as fast as possible, to allow other interrupts to be handled. [?]

2. Interuptin: We register the interruptin to the a callback function. Whenever the button fall or rise, it's considered as a interrupt and trigger the corresponding reaction in the callback function, which is update button state here.

2 Miscellaneous Learning

2.1 SSH to Raspberry Pi, find Raspberry Pi IP on windows powershell

The IP will change if the board change even in the same WIFI Windows $arp\ -a$ command helps us find it.

2.2 Inline

This is mean to speed up the execution time. More of a compiler beneficial technique. It's quite similar to the loop unrolling. It inline unroll the function instead of jumping to other subroutine. Figure 7 [?]

```
/** print device address to the terminal */
inline void print_address(const Gap::Address_t &addr)
{
    printf("%02x:%02x:%02x:%02x:%02x:%02x\r\n",
           addr[5], addr[4], addr[3], addr[2], addr[1], addr[0]);
}
```

Figure 7: Inline

3 Potential Improvement and question

I'm curious about if there's a read write notify type of service.