PSoC BLE Lab 02 Report

University of Western Ontario | ECE 9047

Jiayuan Hong | Tanushree Das | June 8th 2016

PSoC BLE Lab 01 Report

University of Western Ontario | ECE 9047

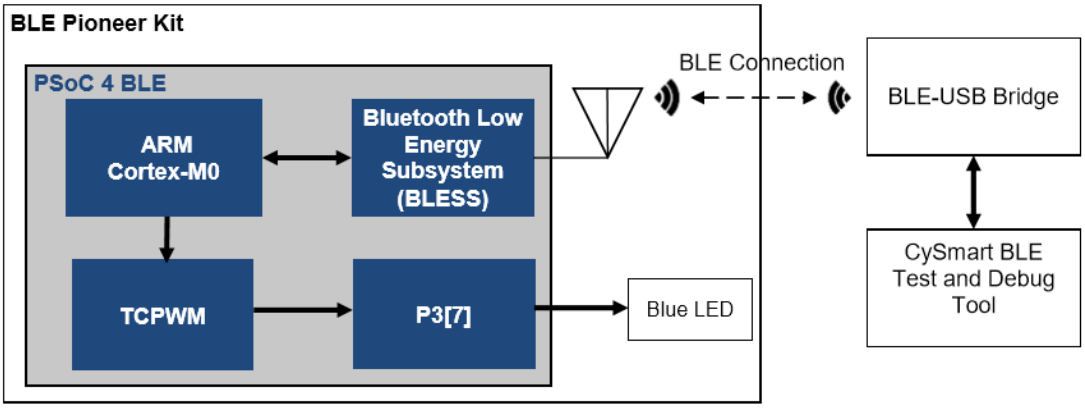
Objectives:

1. Learn how to apply the function of components.
2. Fulfill Immediate Alert Service (IAS) by use a standard BLE Find Me Profile.
3. Learn how to debug BLE designs by CySmart BLE Test and Debug Tool.

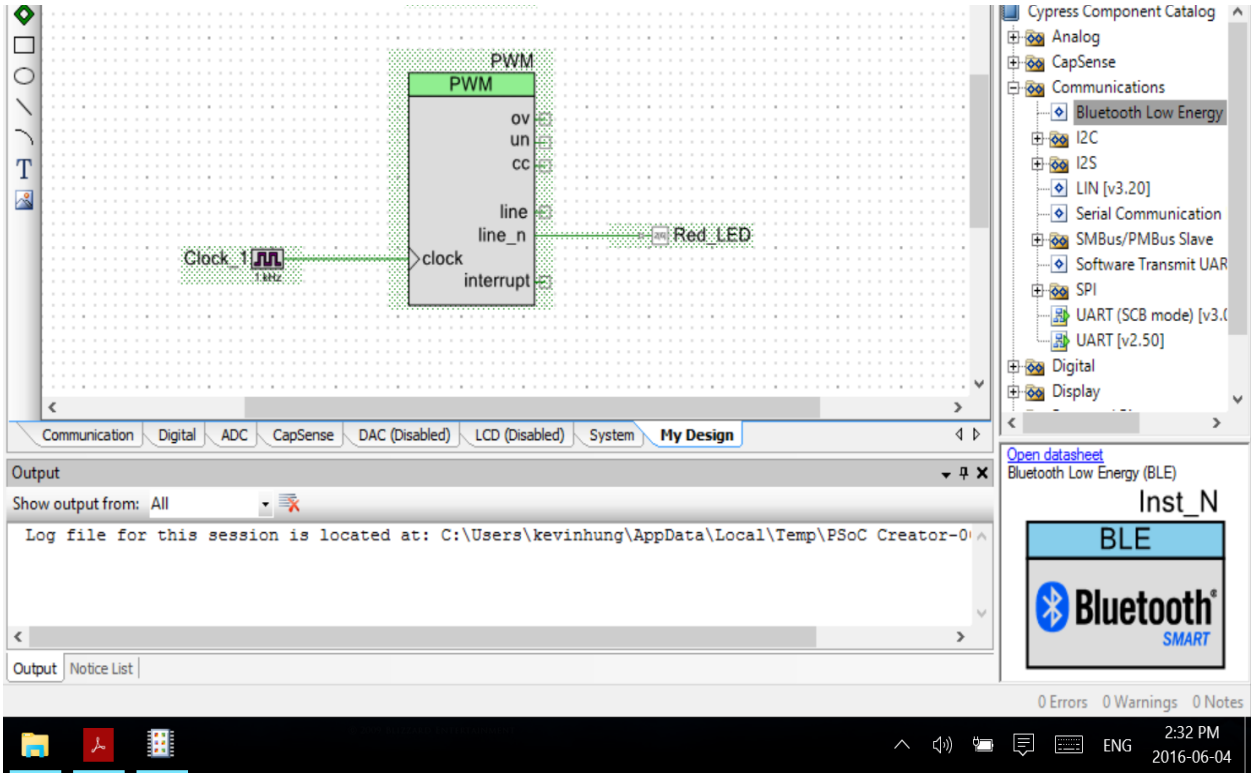
Method:

Lab 2 block diagram:

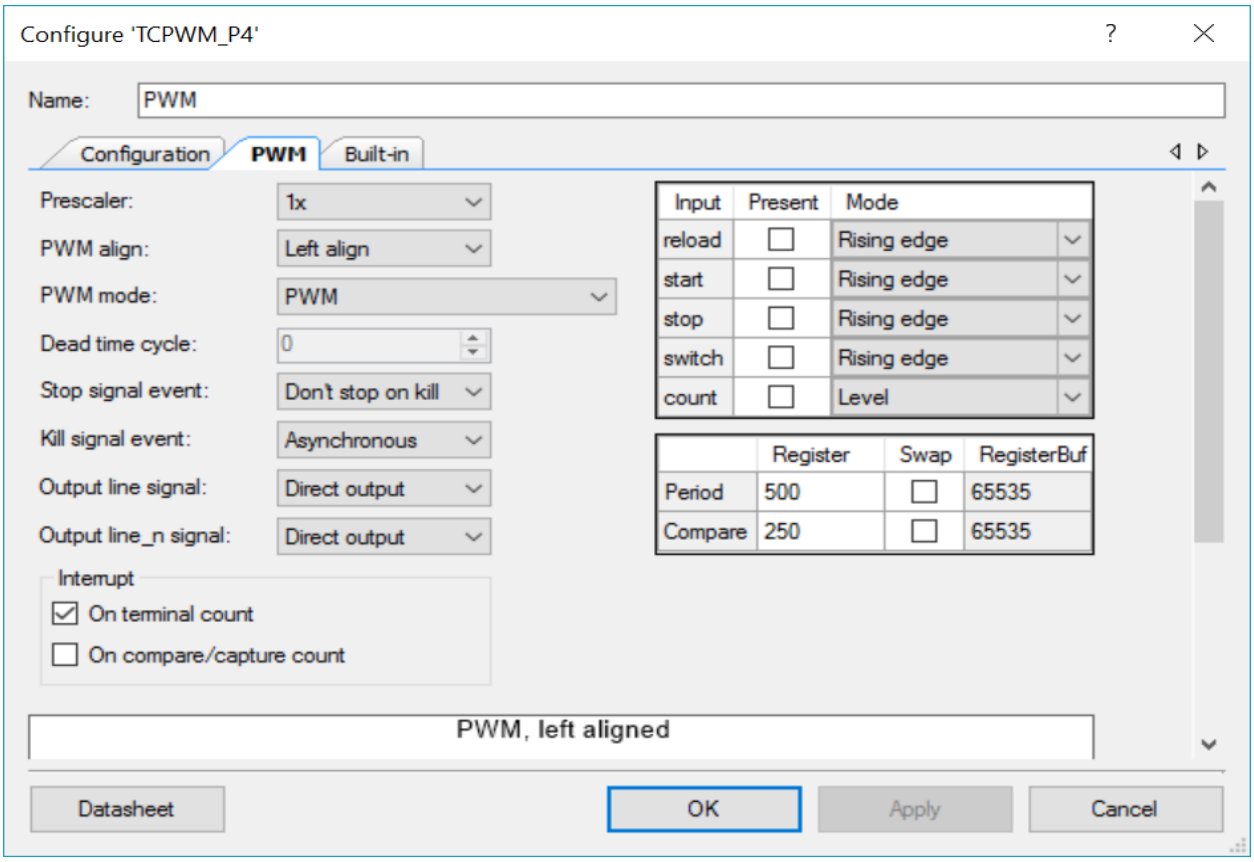
Figure #2: Lab 2 Block Diagram



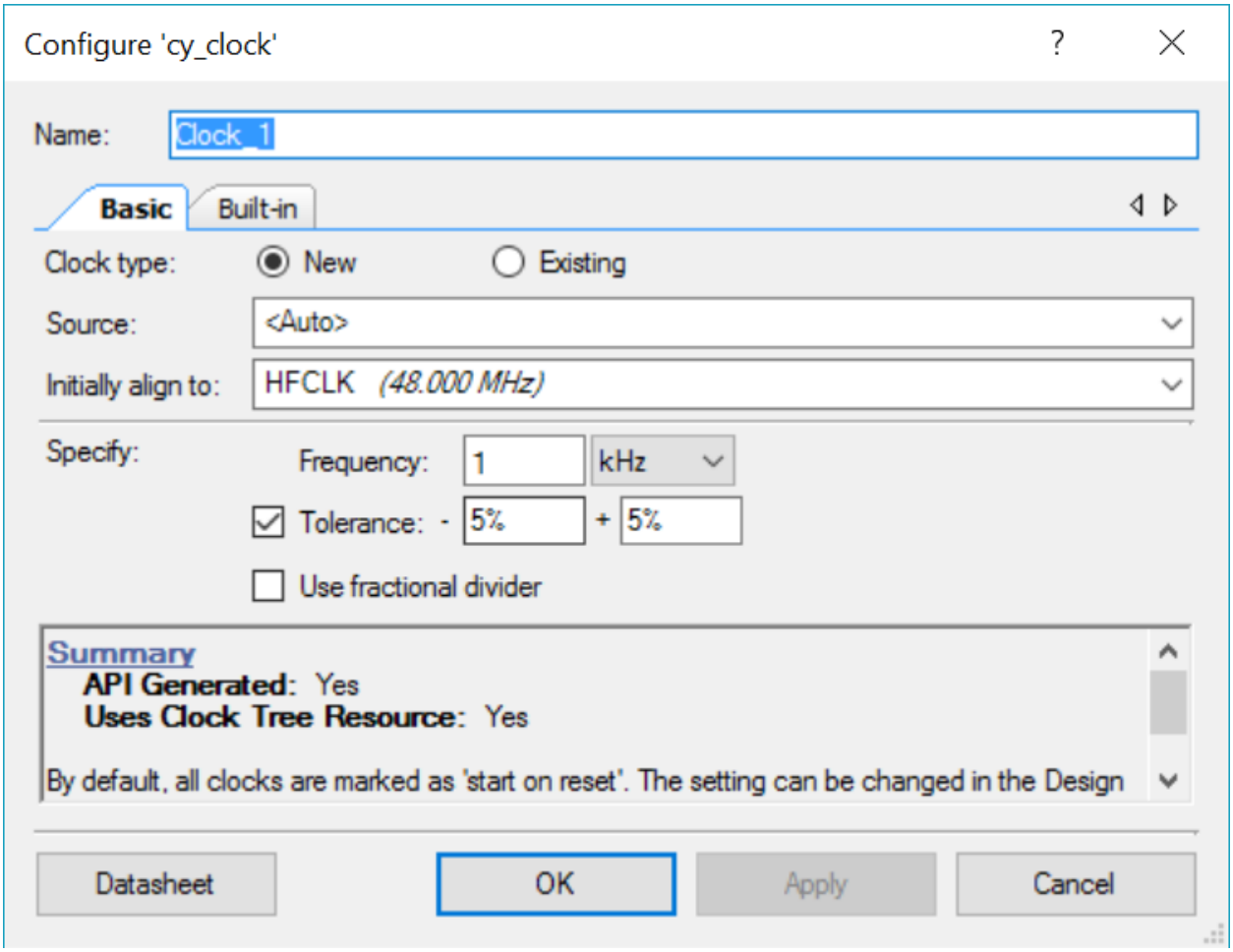
1. We create PSoc 4100/4200 BLE Design project and drag PWM, Clock and Pin components to Mydesign tab and connect the components as followed:



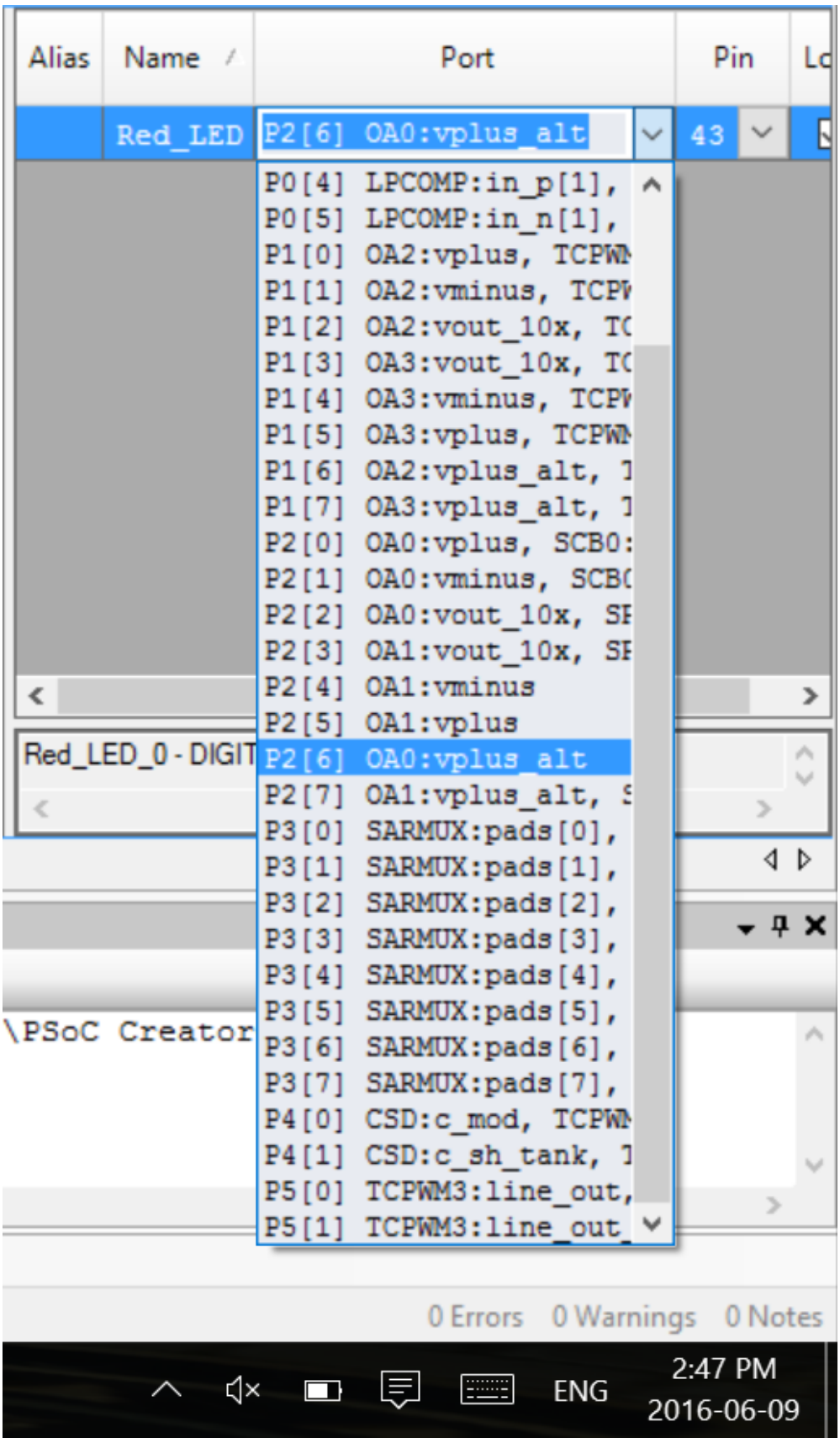
1) Configure PWM component:



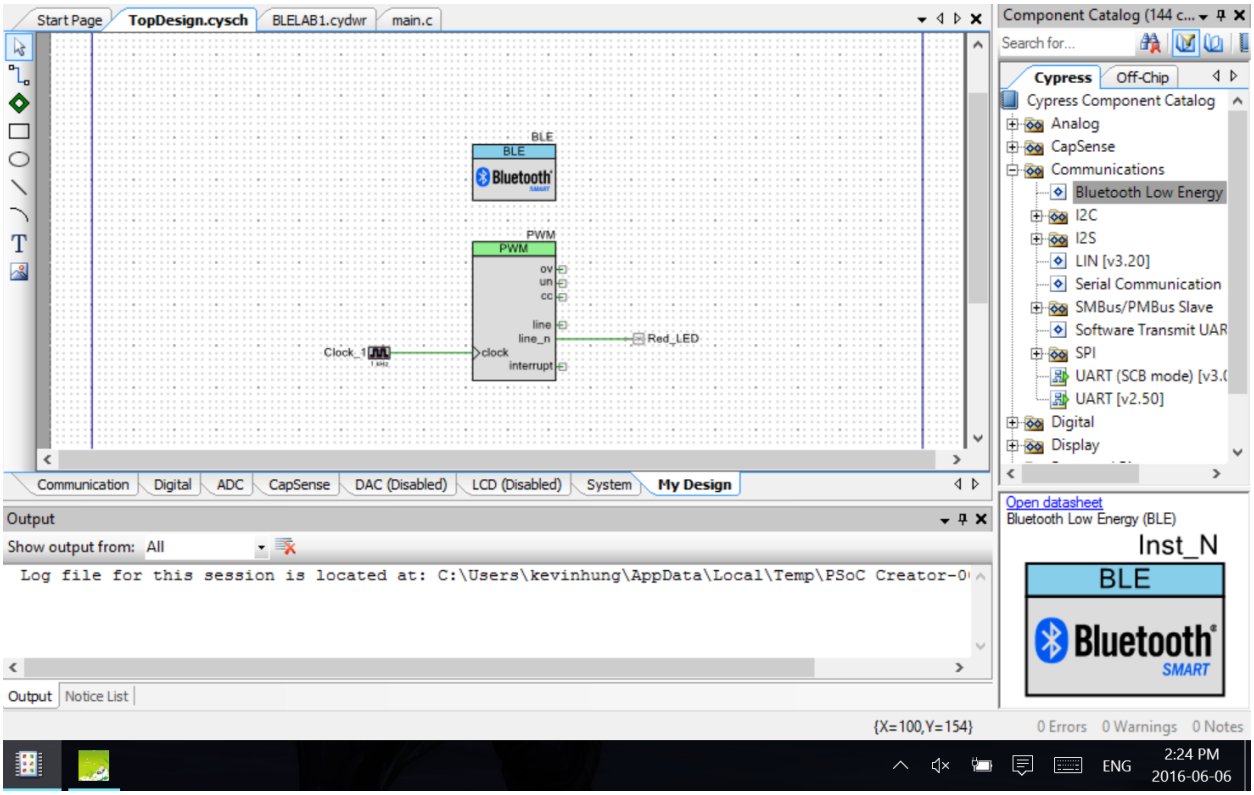
2) Configure clock's frequency to 1kHz:



3) Change Pin component's name to Red_LED and choose the port P2[6] for it:

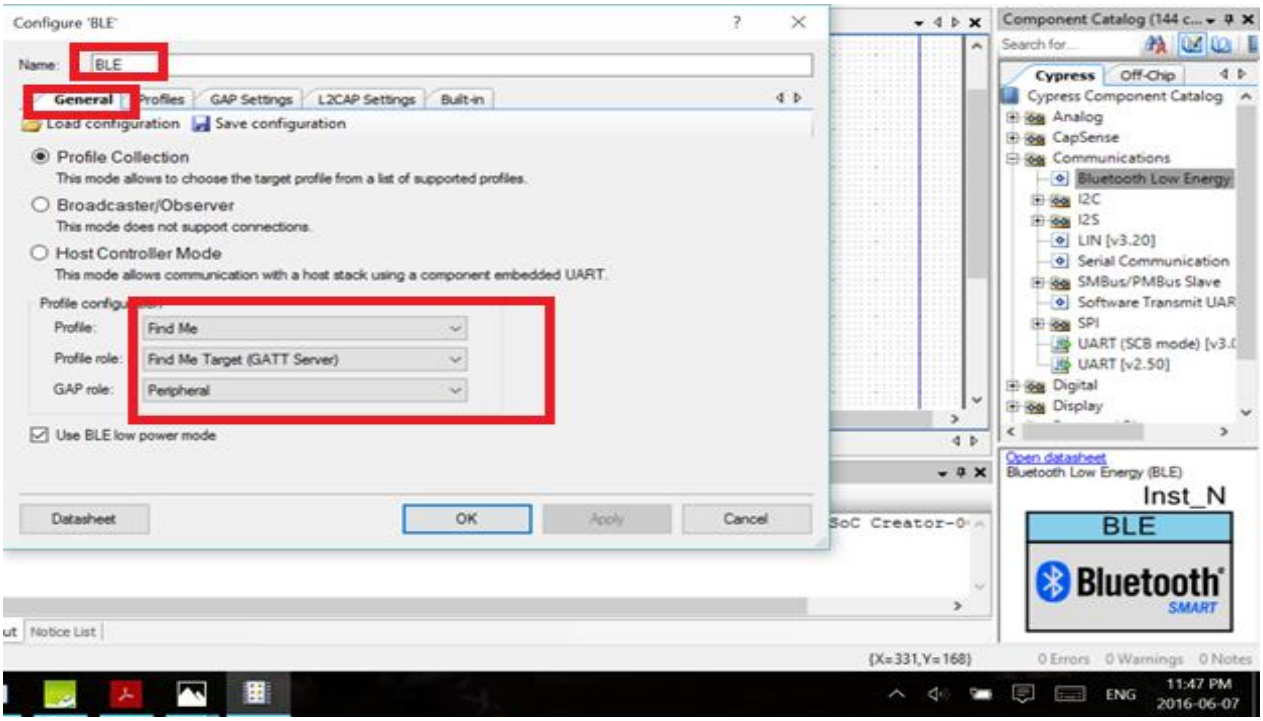

Alias
Name
Port
Pin
Red_LED
P2[6] OA0:vplus_alt
43
P0[4] LPCOMP:in_p[1],
P0[5] LPCOMP:in_n[1],
P1[0] OA2:vplus, TCPWM
P1[1] OA2:vminus, TCPW
P1[2] OA2:vout_10x, TC
P1[3] OA3:vout_10x, TC
P1[4] OA3:vminus, TCPW
P1[5] OA3:vplus, TCPWM
P1[6] OA2:vplus_alt, T
P1[7] OA3:vplus_alt, T
P2[0] OA0:vplus, SCB0:
P2[1] OA0:vminus, SCB0
P2[2] OA0:vout_10x, SP
P2[3] OA1:vout_10x, SP
P2[4] OA1:vminus
P2[5] OA1:vplus
P2[6] OA0:vplus_alt
P2[7] OA1:vplus_alt, S
P3[0] SARMUX:pads[0],
P3[1] SARMUX:pads[1],
P3[2] SARMUX:pads[2],
P3[3] SARMUX:pads[3],
P3[4] SARMUX:pads[4],
P3[5] SARMUX:pads[5],
P3[6] SARMUX:pads[6],
P3[7] SARMUX:pads[7],
P4[0] CSD:c_mod, TCPWM
P4[1] CSD:c_sh_tank, T
P5[0] TCPWM3:line_out,
P5[1] TCPWM3:line_out_
Red_LED_0 - DIGIT
\PSoC Creator
0 Errors
0 Warnings
0 Notes
ENG
2:47 PM
2016-06-09

2. Drag Bluetooth Low Energy (BLE) component to “Mydesign” Schematic:

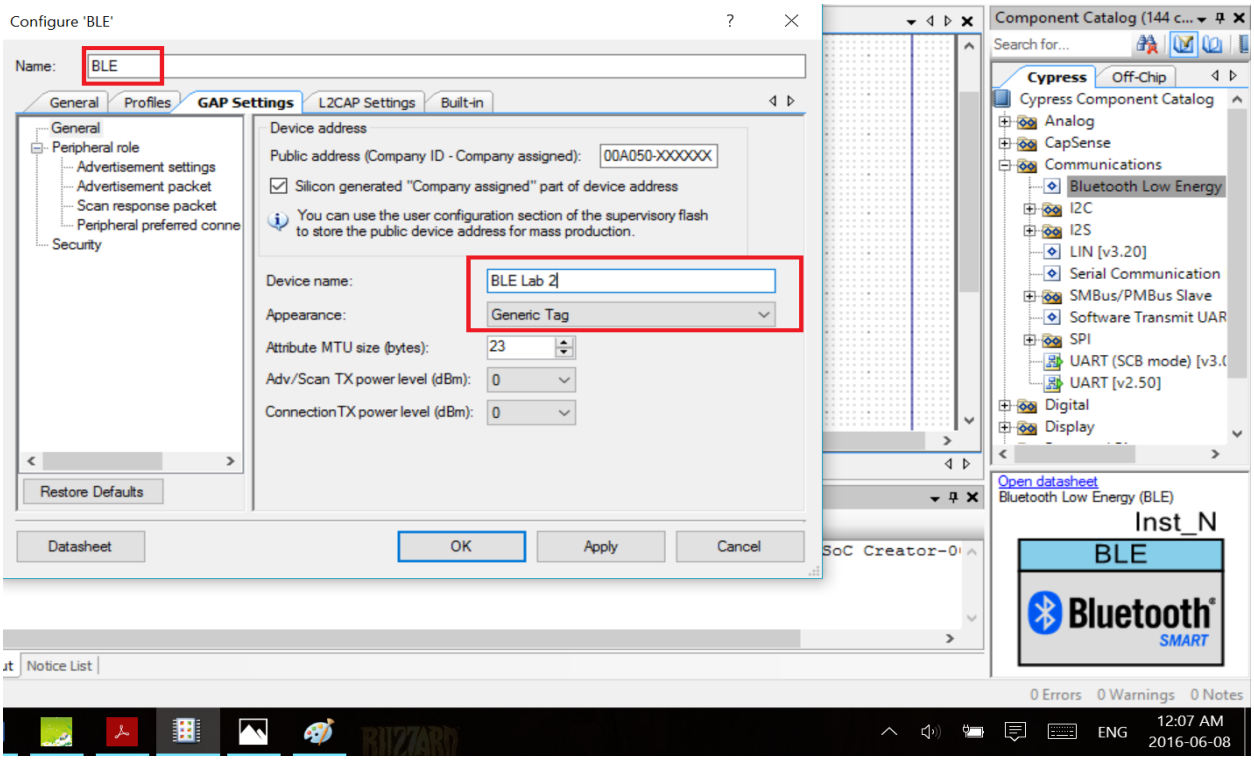


3. Go in to BLE component’s configuration.

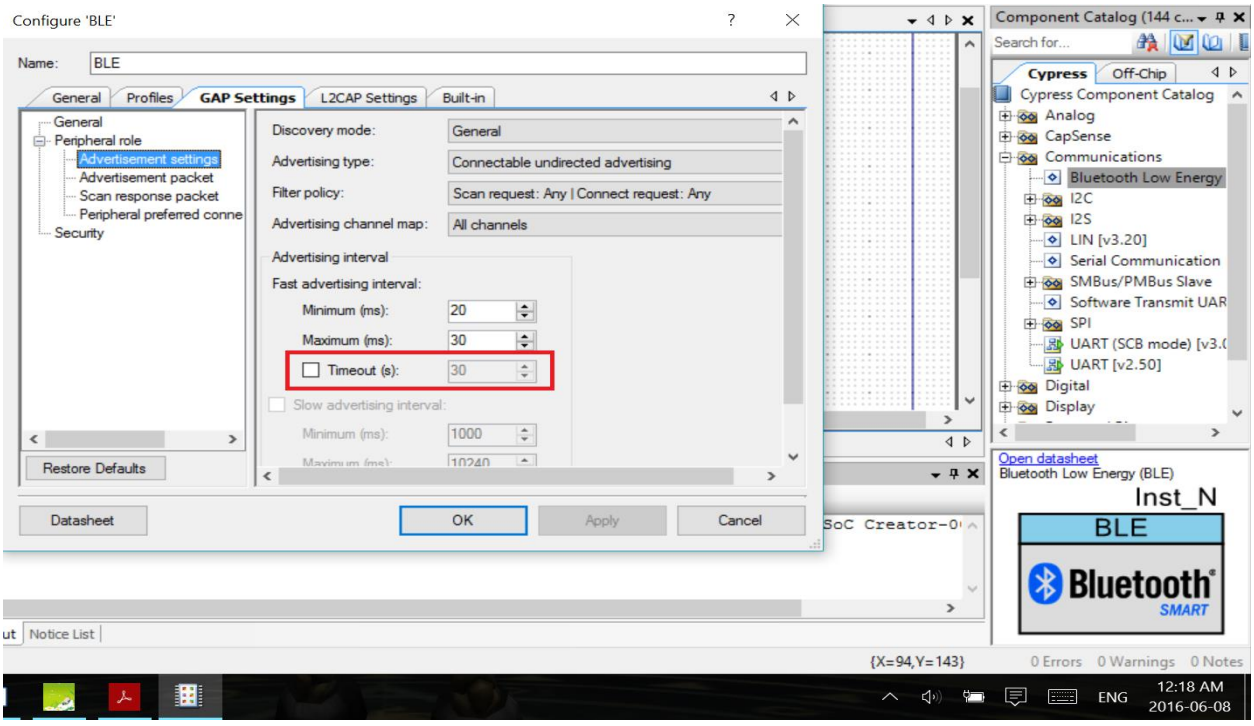
Change component’s name to BLE and choose profile to “Find Me”.



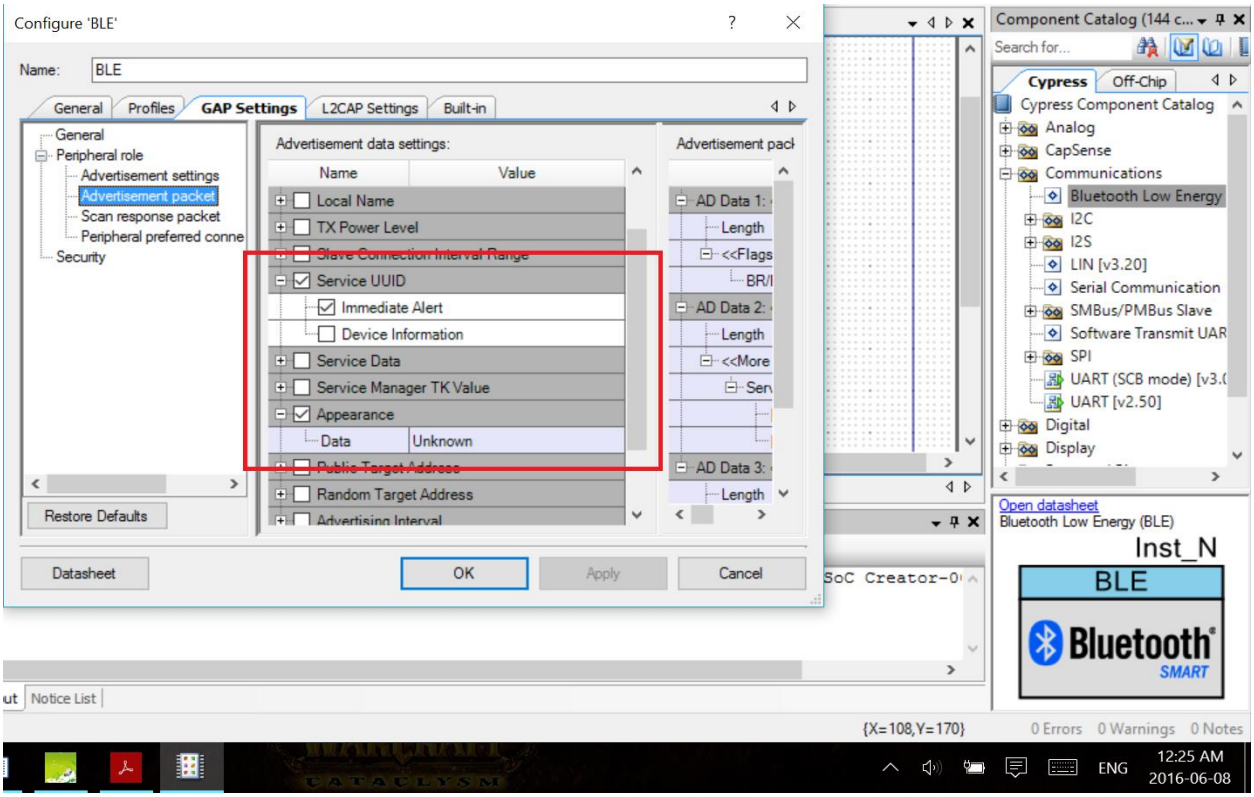
4. Go to GAP setting tab, change device name to BLE Lab 2 and appearance to Generic Tag:



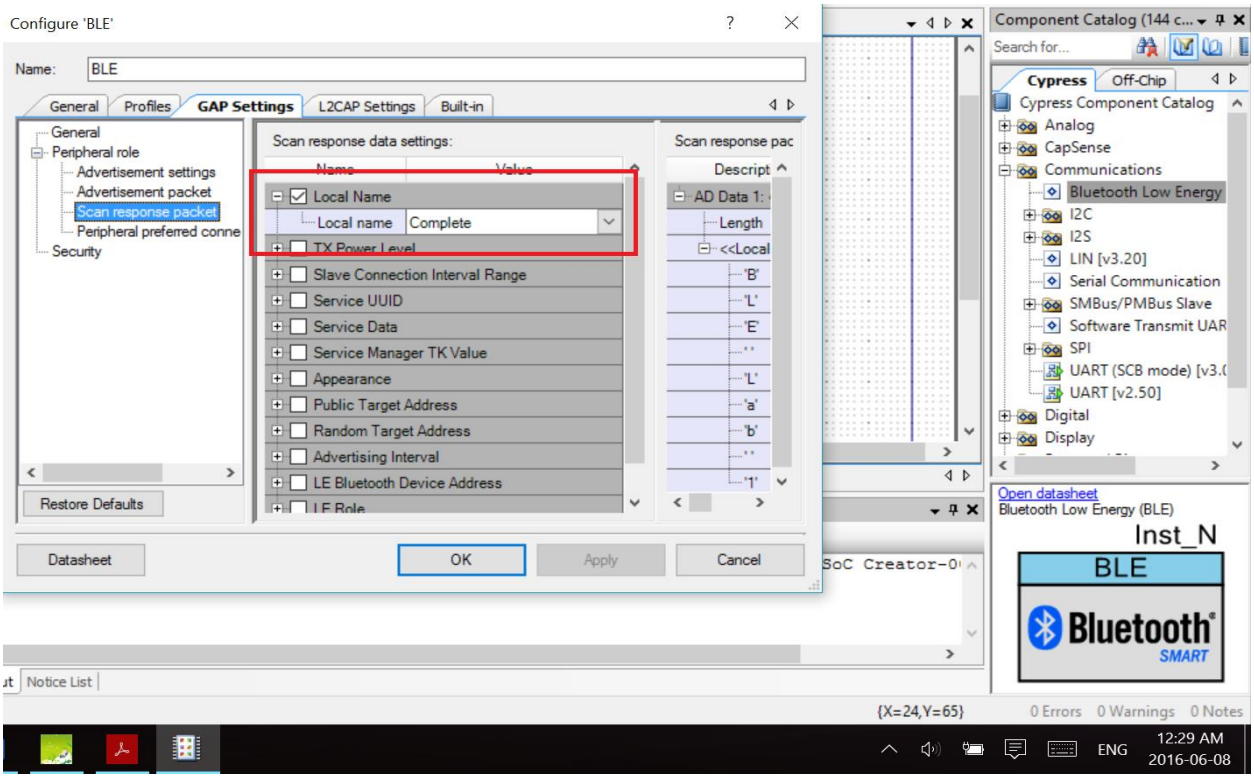
5. Configure Peripheral Role: Advertisement Setting: disable timeout function in fast advertising interval:



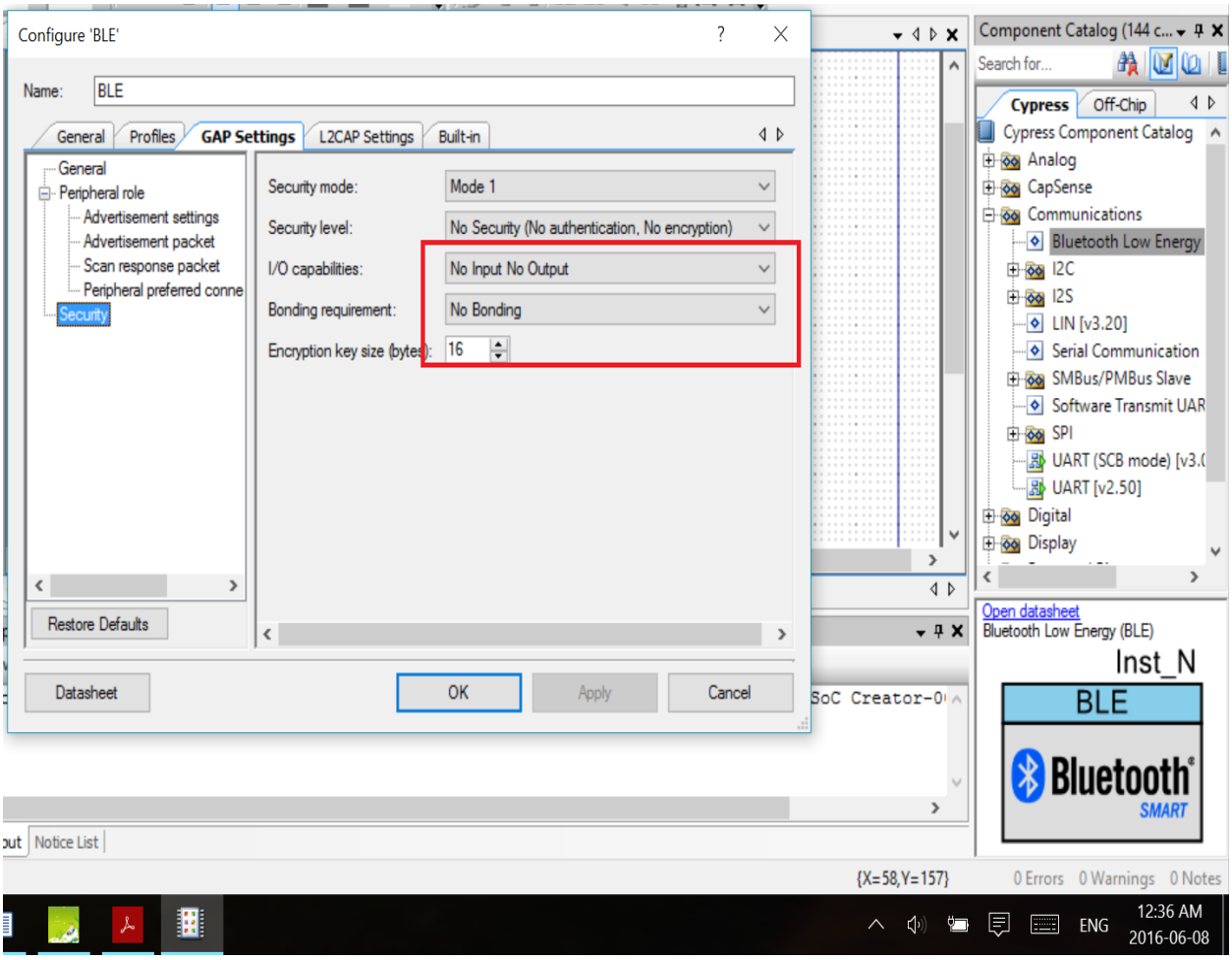
6. Advertisement packet: able Immediate alert and appearance:



7. Advertisement Scan Response Packet: use local name: complete



8. Configue Security: I/O Capabilities set to no input and no output; Bonding requirement set to No Bonding:



9. Import the main.c file from supporting file:

It defines different Alert level as shown:

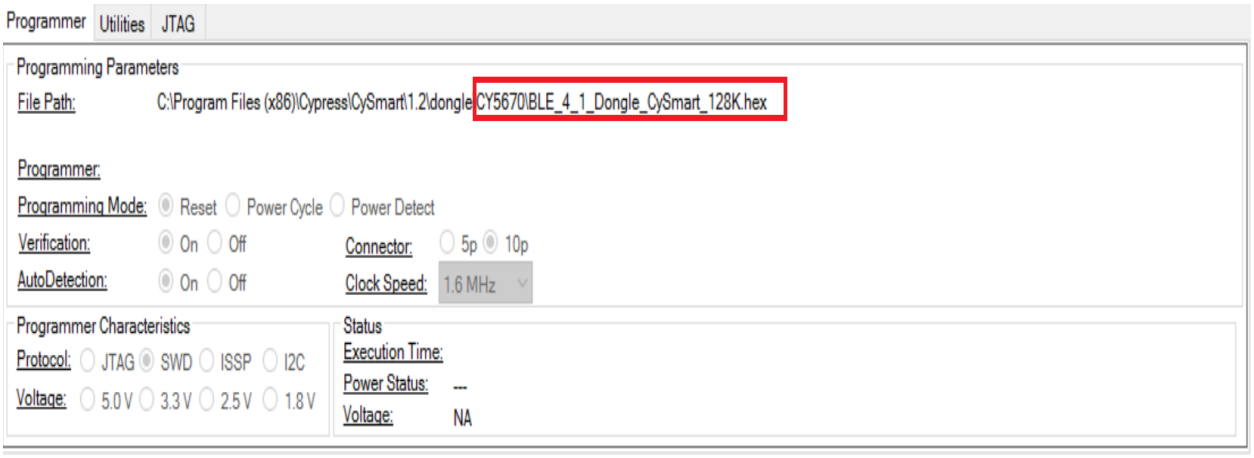
```
void HandleAlertLEDs(uint8 status)
{
    /* Update Alert LED status based on IAS Alert level characteristic. */
    switch(status)
    {
        case NO_ALERT:
            PWM_WriteCompare(NO_ALERT_COMPARE);
            break;

        case MILD_ALERT:
            PWM_WriteCompare(MILD_ALERT_COMPARE);
            break;

        case HIGH_ALERT:
            PWM_WriteCompare(HIGH_ALERT_COMPARE);
            break;
    }
}
```

Alert Level	PWM Duty Cycle	LED Status
NO_ALERT	0%	Always OFF
MILD_ALERT	50%	LED toggling at 2Hz
HIGH_ALERT	100%	Always ON

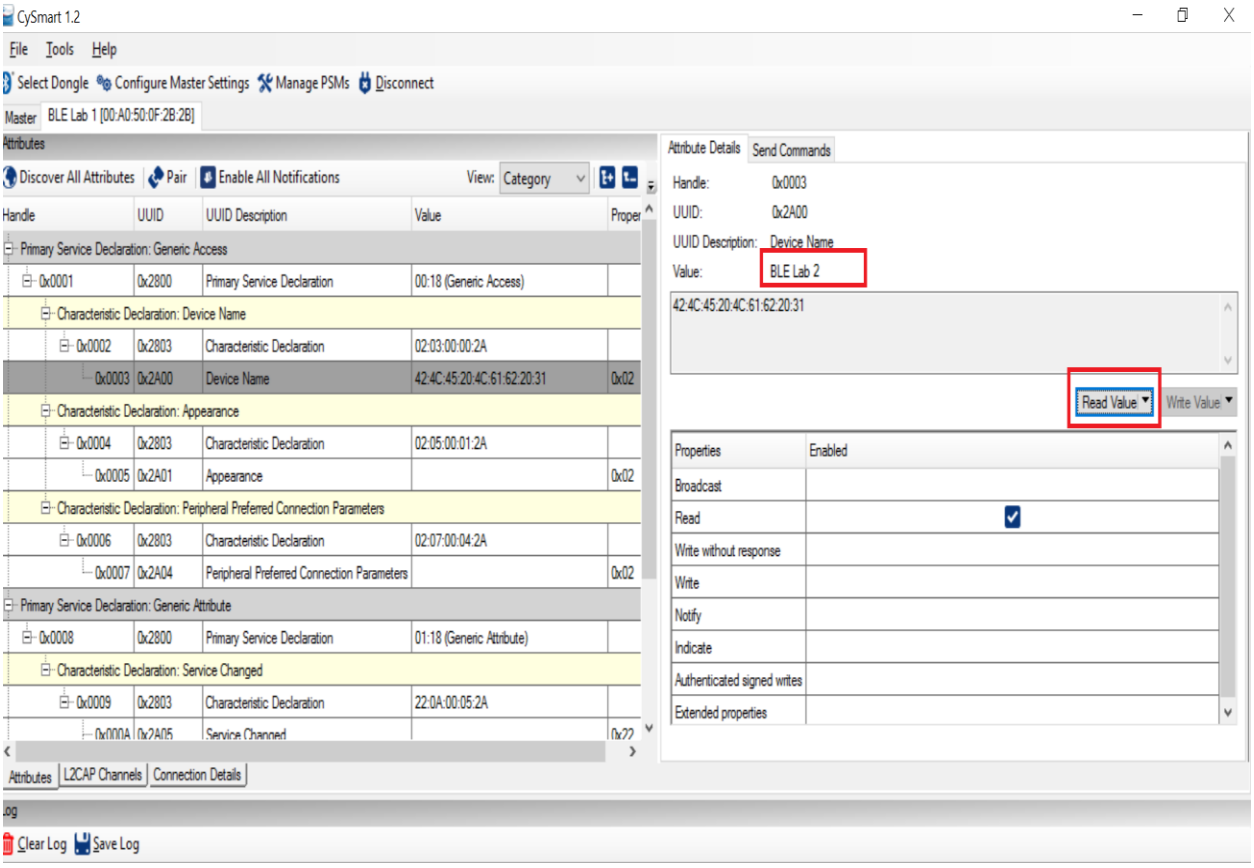
10. Click Ok, generate the application and then program it in to the kit.

Test:

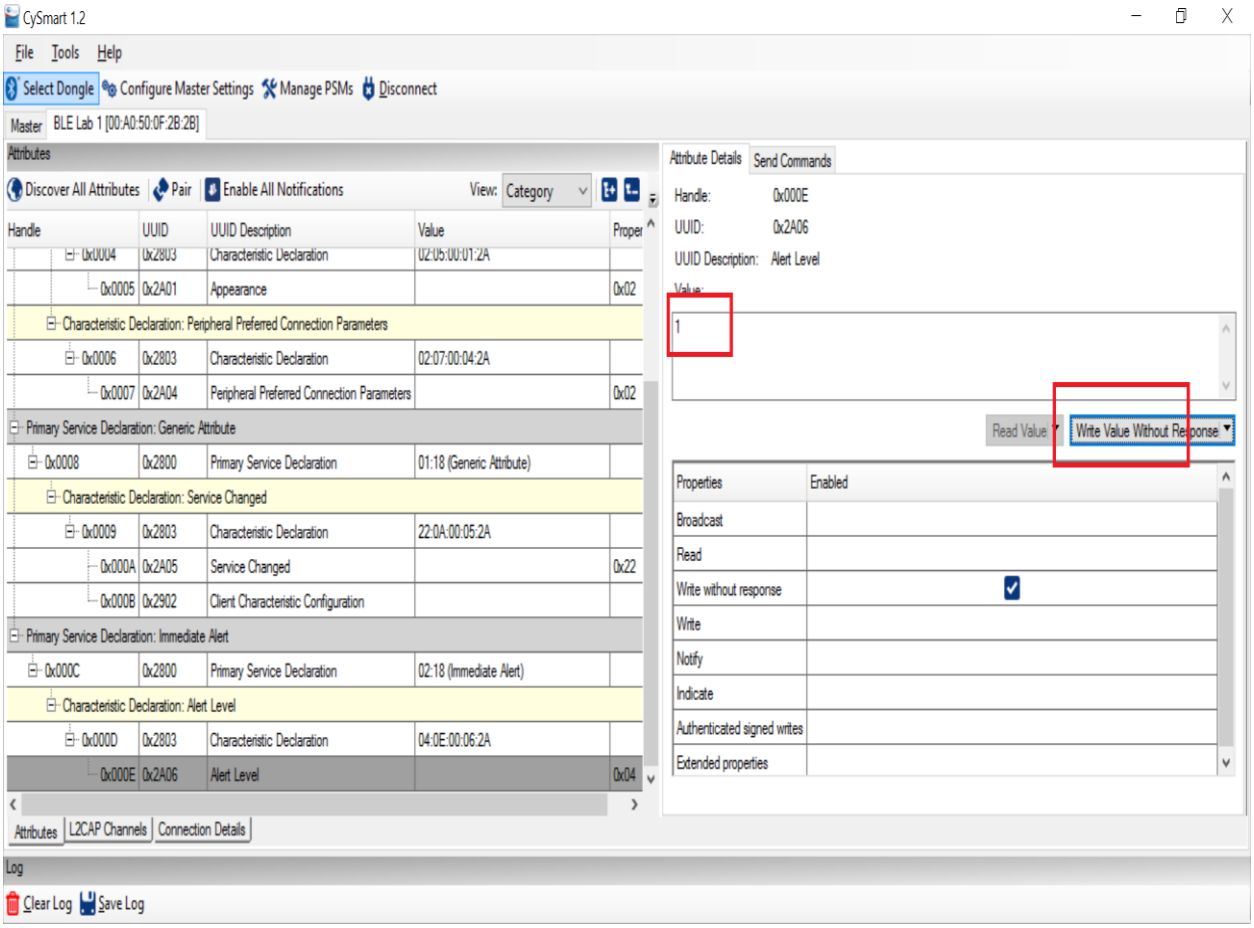
1. Use PSoc Programmer to renew the firmware of the BLE Dongle.



2. Open Cysmart to connect the BLE Dongle, and Scan Kit through Bluetooth. Read the value:



3. Click Write Value to change the alert level:



4. The LED on the kit flashed at 2Hz.

Conclusion:

In BLE Lab 2, we kept going on experiencing from Lab 1, implementing Immediate Alert Service (IAS) and using BLE Dongle to communicate with BLE kit. Through setting a different Alert Level, the BLE kit can have different status that have different LED flashing types.