life.augmented

STM32WB Video Series – Getting Started

#12 STM32CubeMX & STM32CubeIDE lab

T.O.M.A.S. Team

life.augmented

STM32WB Video Series – Getting Started

#12a STM32CubeMX & STM32CubeIDE lab Introduction

T.O.M.A.S. Team

Prerequisites

What we‘ll use

- STM32CubeMX V6.3.0 + download STM32CubeWB V1.12.1 from the package manager
- CubeIDE V1.7.0
- STM32CubeProgrammer V2.8.0
- Android or iOS phone with ST BLE Toolbox app V1.0.0
- 1x micro USB cable
- STM32WB Nucleo (P-NUCLEO-WB55)
- Install latest FUS and BLE Full Stack. From STM32CubeWB V1.12.1 Installation instruction are provided as a homework before the session
- Tera Term (or similar terminal app, only for optional task)

Goal

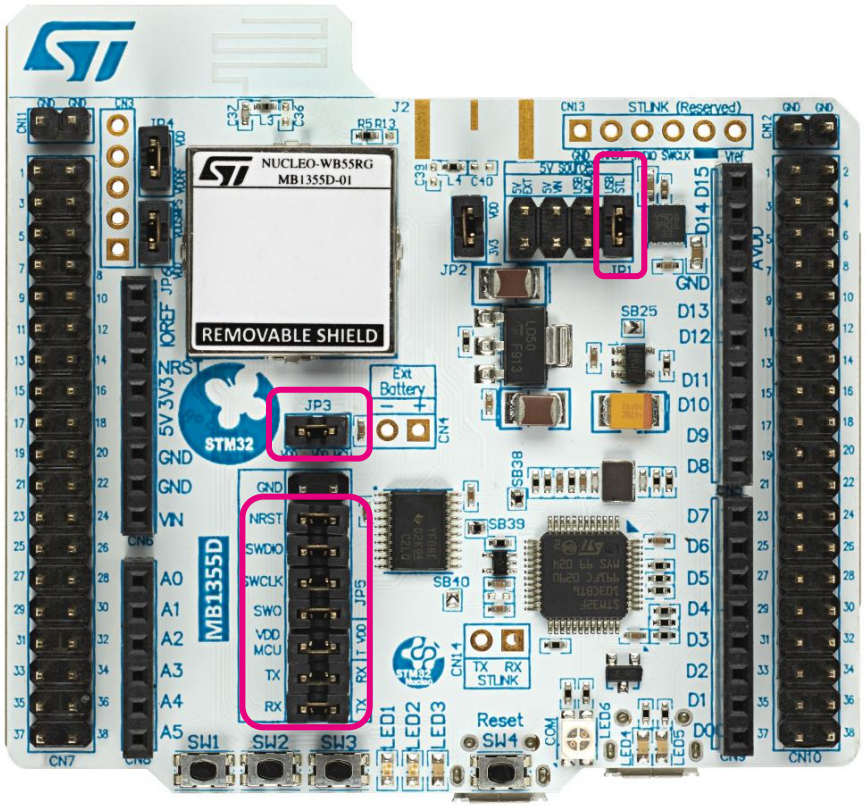
- What are we going to do?
- How are we going to do that?

ST BLE Toolbox

Download on the App Store

GET IT ON Google Play

GATT Client **GAP central**



GATT Server **GAP peripheral**

life.augmented

Goal in more details

- Create custom BLE Service with two characteristics
  - CHARACTERISTIC 1 has Write property
  - CHARACTERISTIC 2 has Notify property
- Writing to CHARACTERISTIC 1 (LED control) will toggle green LED
- Pressing SW1 button on Nucleo board will send some data to the phone (Button 1 notification)

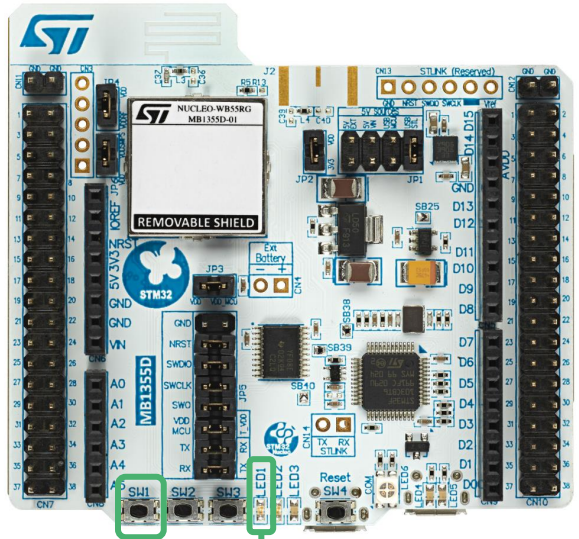
Write

Notification

Custom SERVICE

Custom CHARACTERISTIC 1
W
LED control

Custom CHARACTERISTIC 2
N
Button 1 state



life.augmented

Creating BLE template project in CubeMX

Reference part 1

- Open CubeMX
- Start new project for Nucleo WB55RG (Initialize all peripherals in default mode? NO)
- System Core -> RCC: Enable HSE and LSE – Crystal/Ceramic Resonator
- System Core -> HSEM - Check Activated
- System Core -> IPCC – Check activated and enable both interrupts
- Timers -> RTC – Activate Clock, Enable internal WakeUp, enable interrupt
- Connectivity -> RF – Activate RF1
- Middleware -> STM32_WPAN – Check BLE
  - BLE Application and Services
    - Disable Custom P2P Server
    - Enable Custom Template

Creating BLE template project in CubeMX
Reference part 2

- Middleware -> STM32_WPAN
  - BLE Advertising
    - Include AD_TYPE_COMPLETE_LOCAL_NAME Yes
  - BLE GATT
    - Number of services = 1
    - Service long name = “My_SVC”
    - Service short name = “My_SVC”
  - My_SVC
    - Characteristic long name = “My char”
    - Characteristic short name = “My char”
    - CHAR_PROP_BROADCAST NO
    - CHAR_PROP_READ YES
    - GATT_NOTIFY_READ_REQ_AND_WAIT_FOR_APPL_RESP No
- Clock configuration
  - Set RTC clock to LSE
  - Set RFWKP clock to LSE
- Project Manager: Generate code for CubeIDE or any other supported IDE
- Add some user application code (LED, Button handling)

Optional tasks

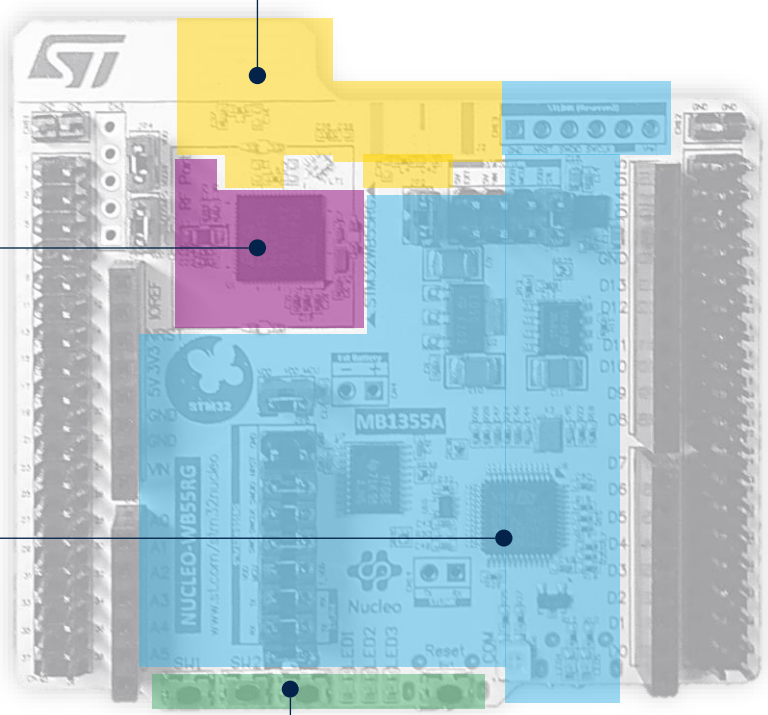
- For those of you who finish early you might follow the instructions at the end of this presentation
- Optional task 1: Parse data input written to the characteristic by phone
- Optional task 2: Enable application traces over USART and display them in terminal console

NUCLEO-WB55 in detail

RF area
- 2.4GHz PCB antenna
- impedance matching
- ext. LP filter (IPD)
- (space for SMA connector and RF switch)

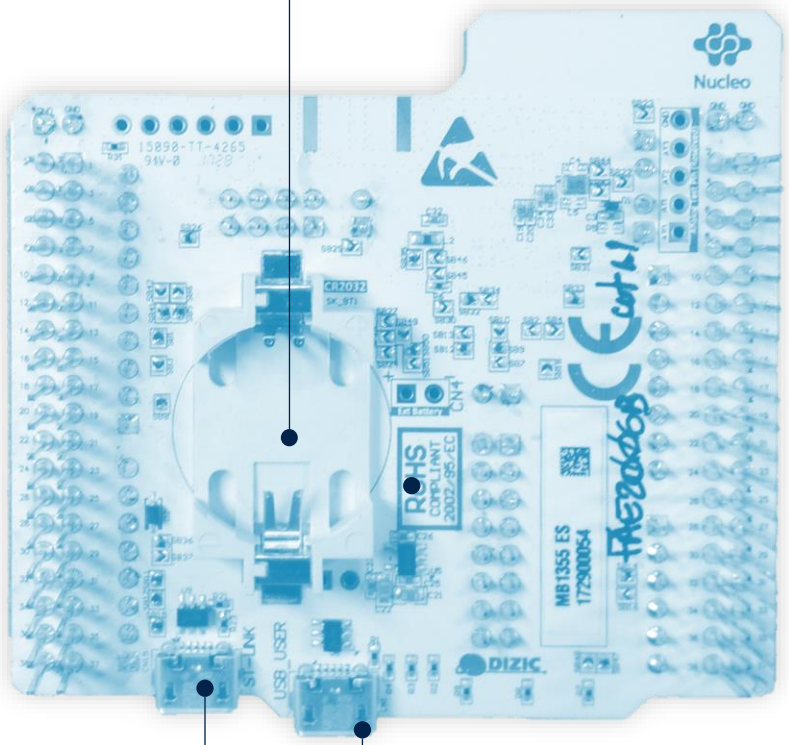
MCU area
- STM32WB55RC
- HSE & LSE crystals
- Decoupling
- SMPS ext. parts

ST-Link area
- ST-Link/V2-1
- SWD debugger
- Virtual COM Port
- USB MSC (.bin flashing)



UI Area
- 3x Button
- 3x LED
- Reset Button

Space for **CR2032** battery **socket**



ST-Link connector
(micro-USB Type B)

User USB FS Device
(micro-USB Type B)

life.augmented

life.augmented

STM32WB Video Series – Getting Started

#12b STM32CubeMX & STM32CubeIDE lab
Receiving data from the phone

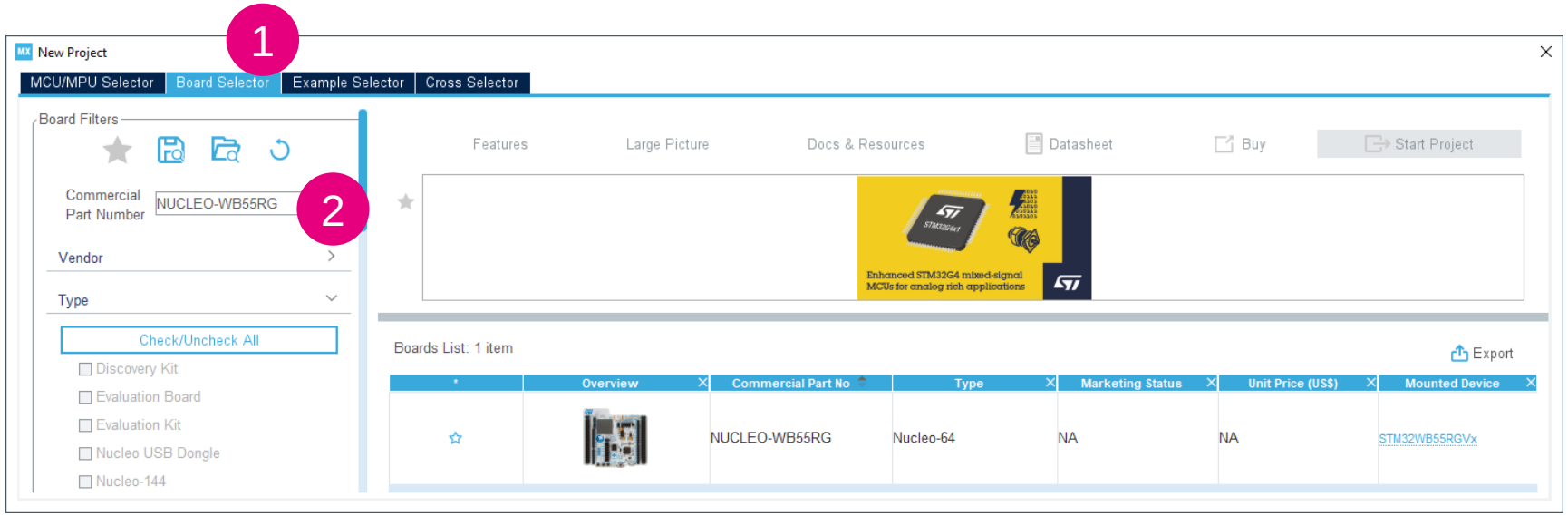
T.O.M.A.S. Team

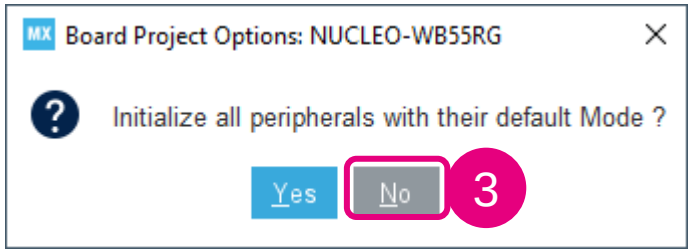
Hands-On

1. Connect USB cable between PC and ST-LINK (right connector from top view)
2. Open STM32CubeMX

Hands-On

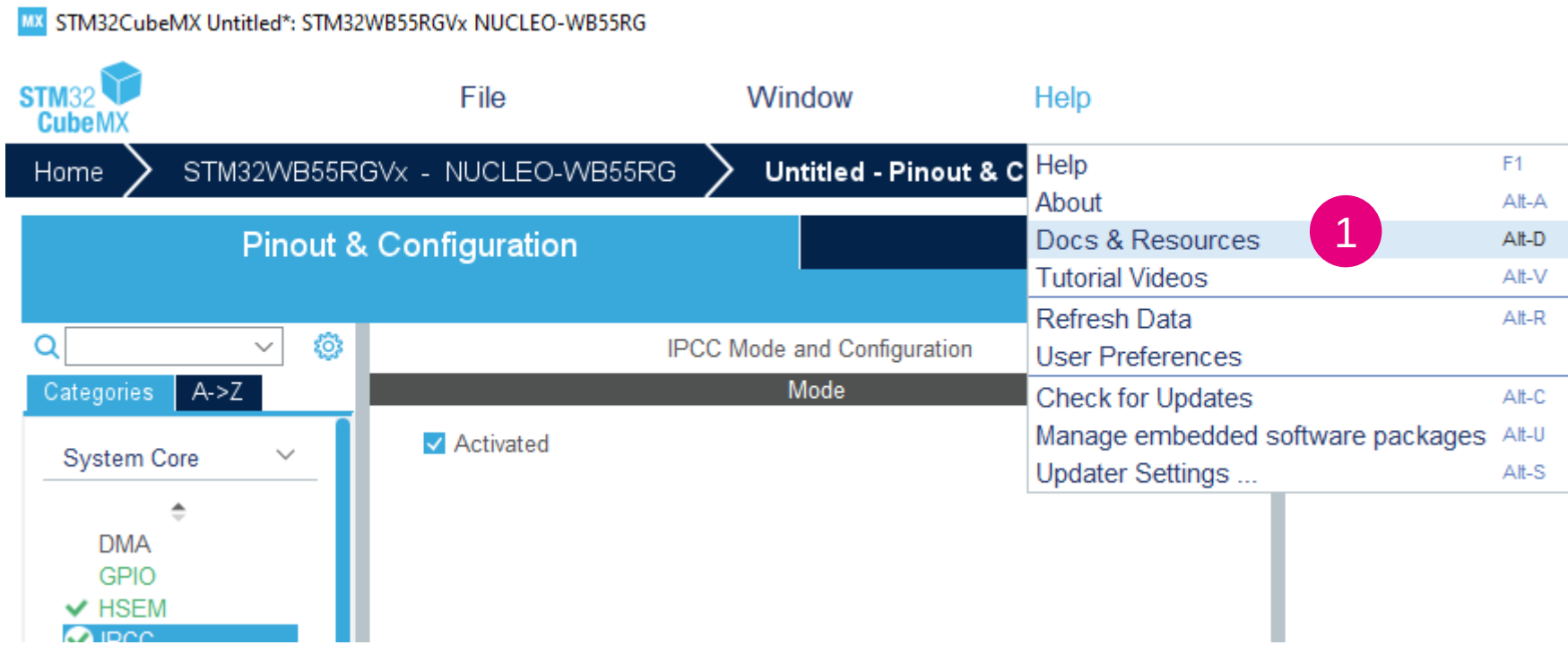
- Start new project
- Select Board selector
- Type “WB55RG” and select NUCLEO-WB5RG
- Start Project
- Initialize all peripherals with their default Mode? No
  This will only initialize GPIO for LEDs and buttons





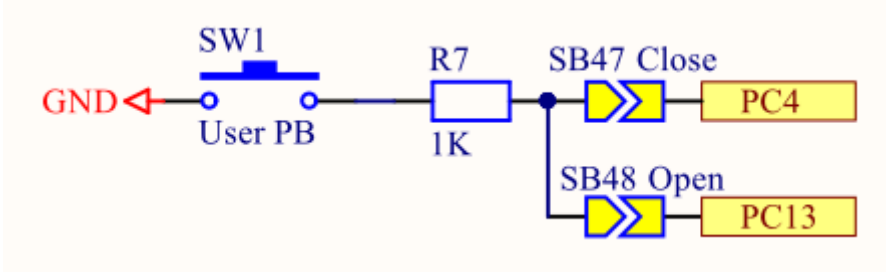
Hands-On

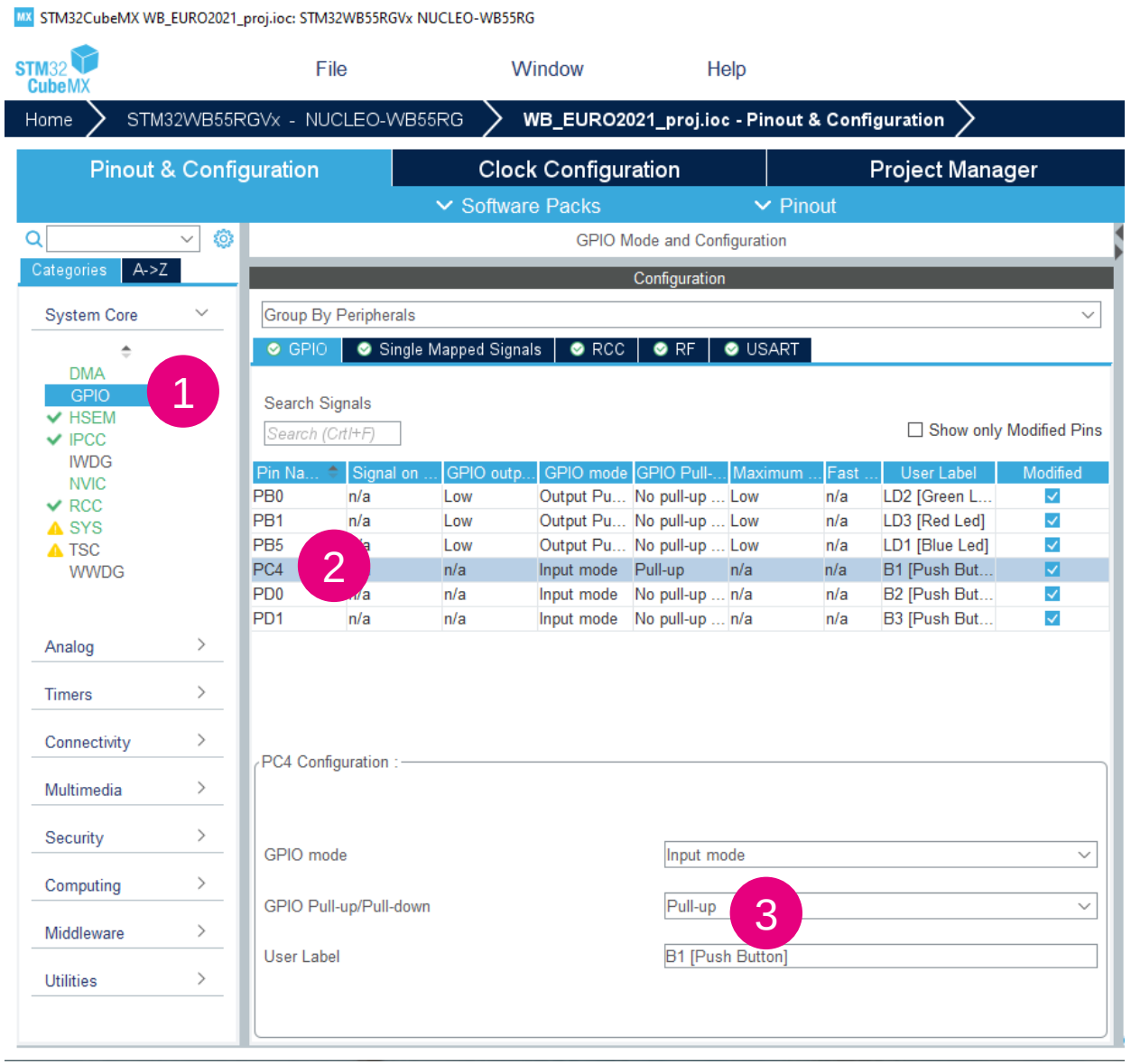
- To access reference manual for the board or for the MCU, datasheet, errata sheet or application notes you may go to Help -> Docs & Resources



Hands-On

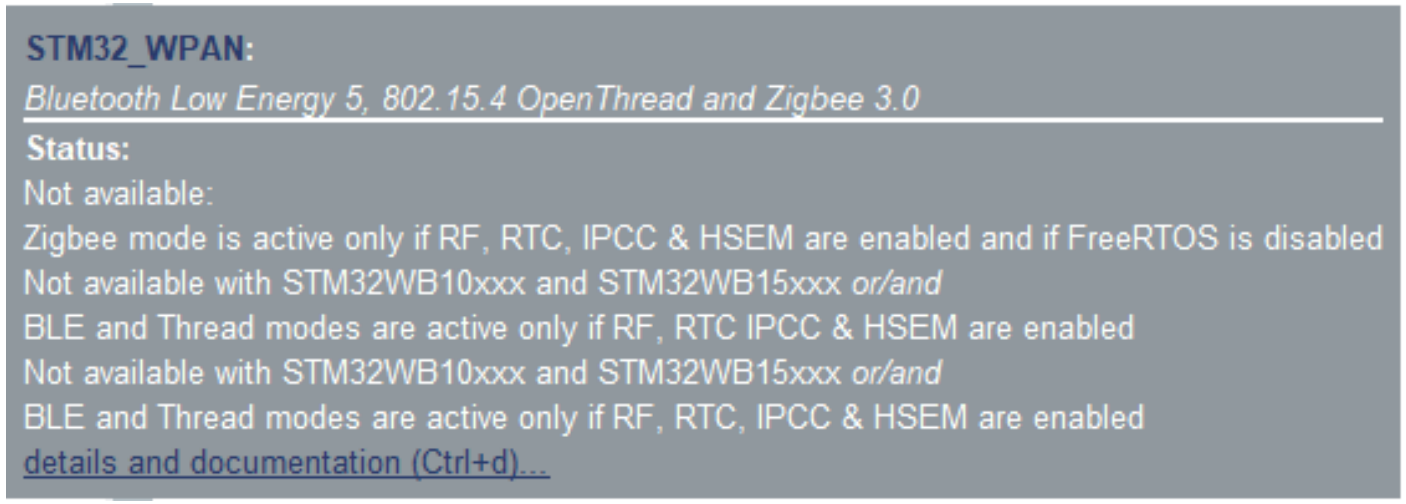
- Select GPIO under System Core tab
- Find PC4
- Enable internal pull-up

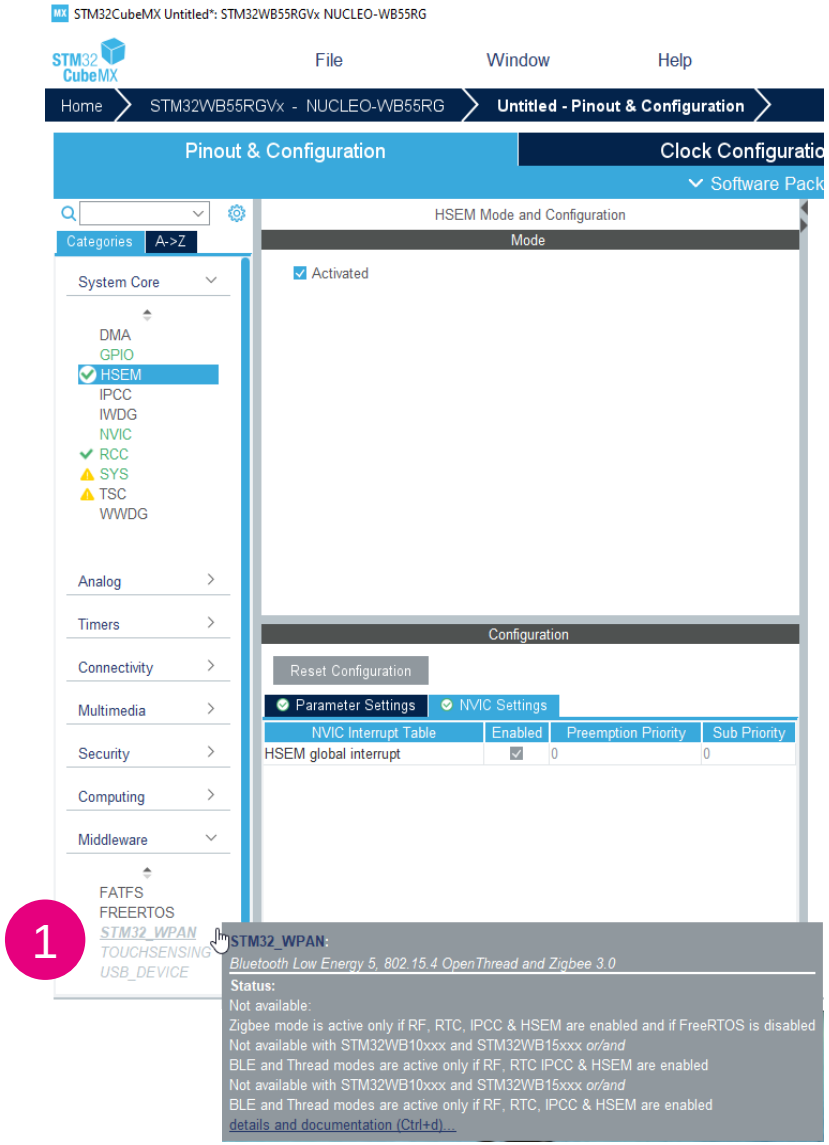




Hands-On

- Hover the cursor on STM32 WPAN Middleware. It will give you contextual help on what to do. We will do each step together

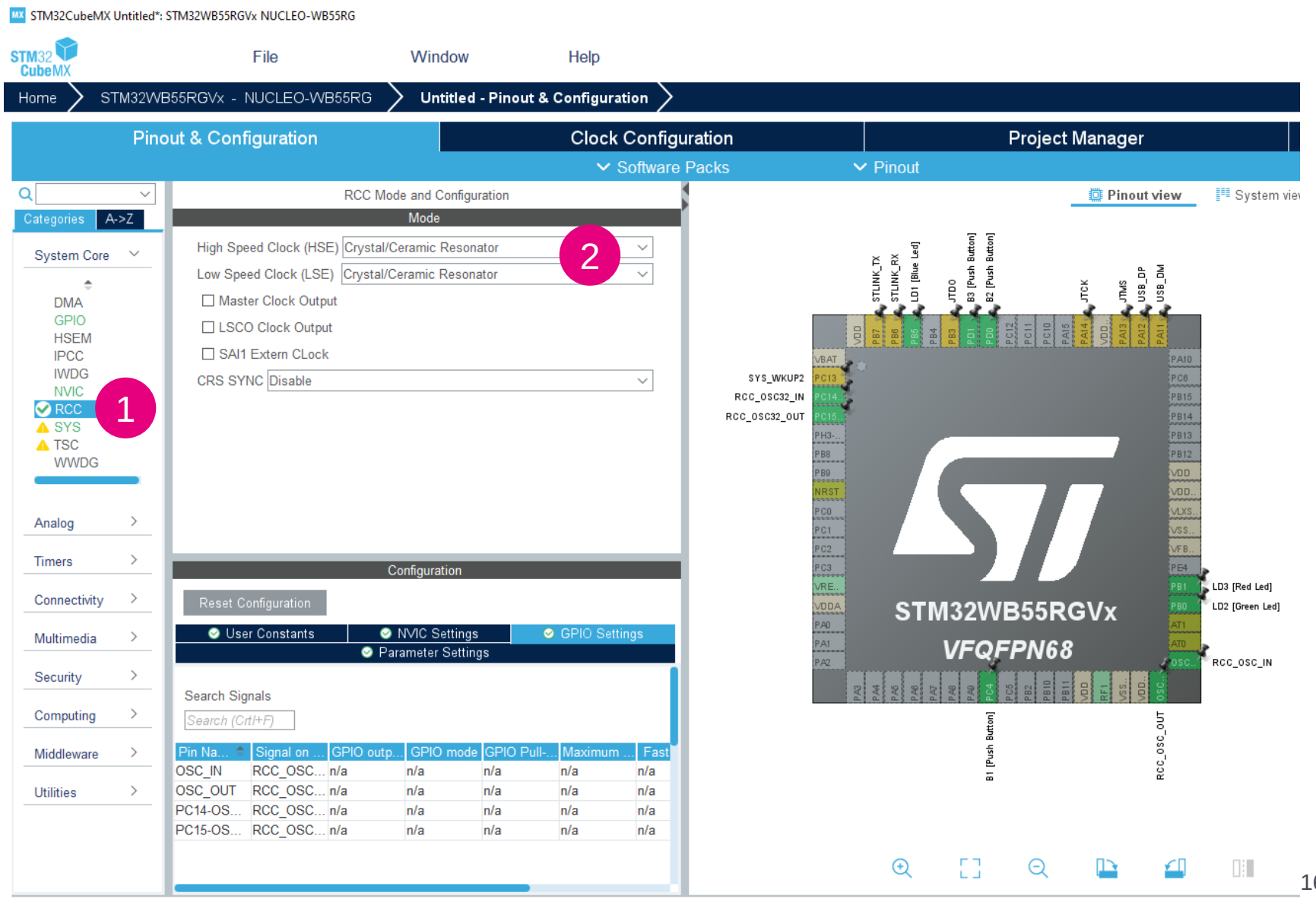




life.augmented

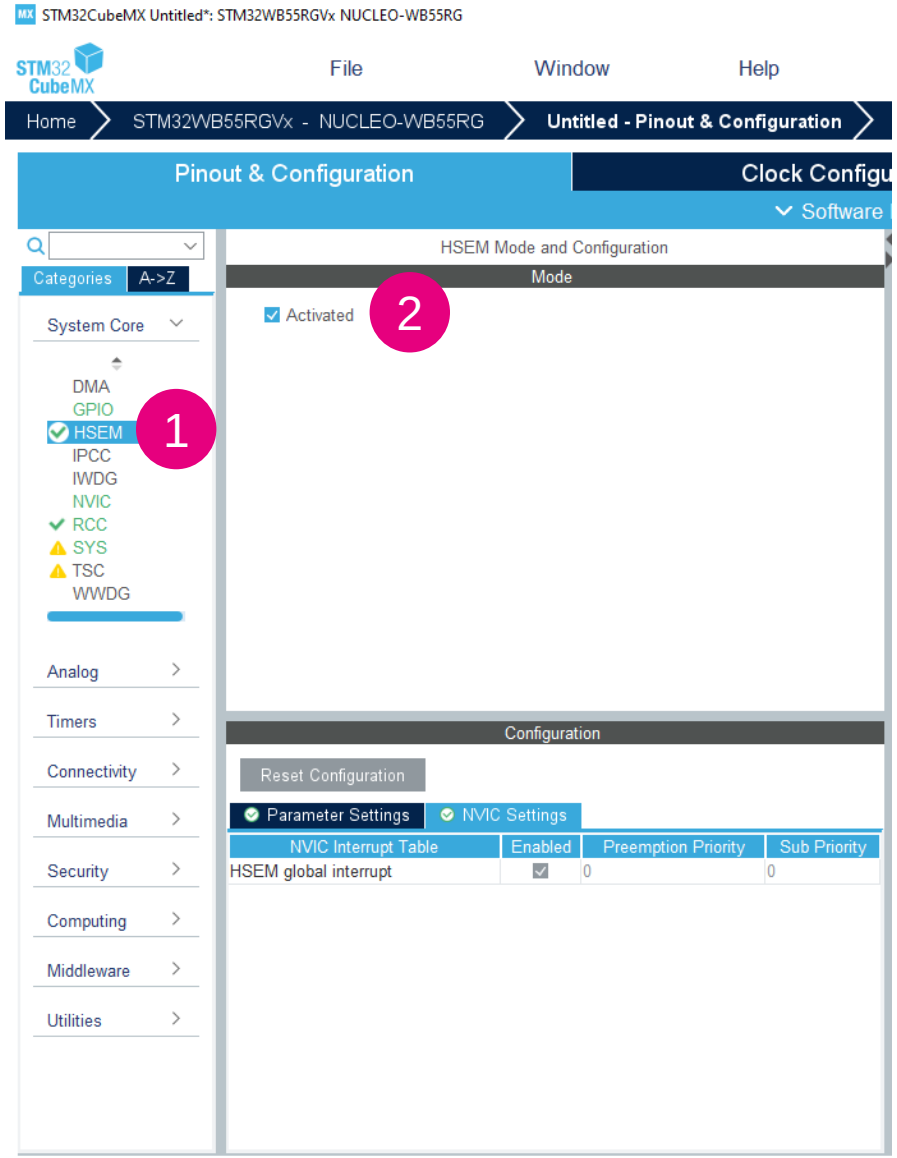
Hands-On

- Select RCC under System Core tab
- Enable HSE (High speed external crystal 32 MHz) and LSE (Low speed external crystal) 32768Hz
- HSE 32 MHz is directly used by the radio PHY.
- LSE is used as a low-speed clock by radio PHY. It is used to time events such as connection or advertising interval



Hands-On

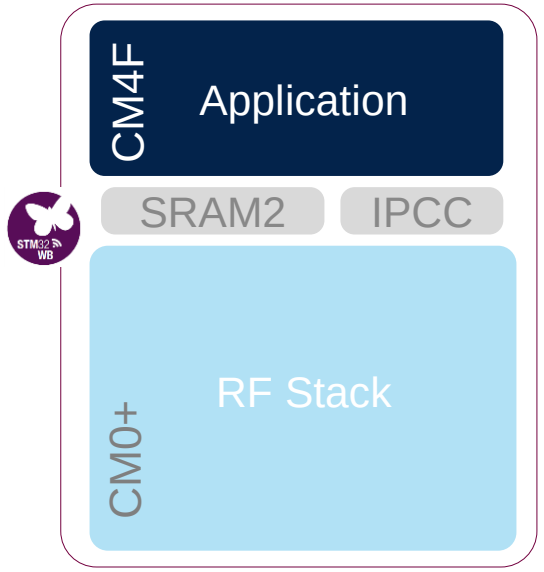
- Select HSEM (Hardware semaphores) and tick activate
- HSEM provides synchronization between CM0+ and CM4 when using shared resources (ex. Clock tree registers, RNG etc.)

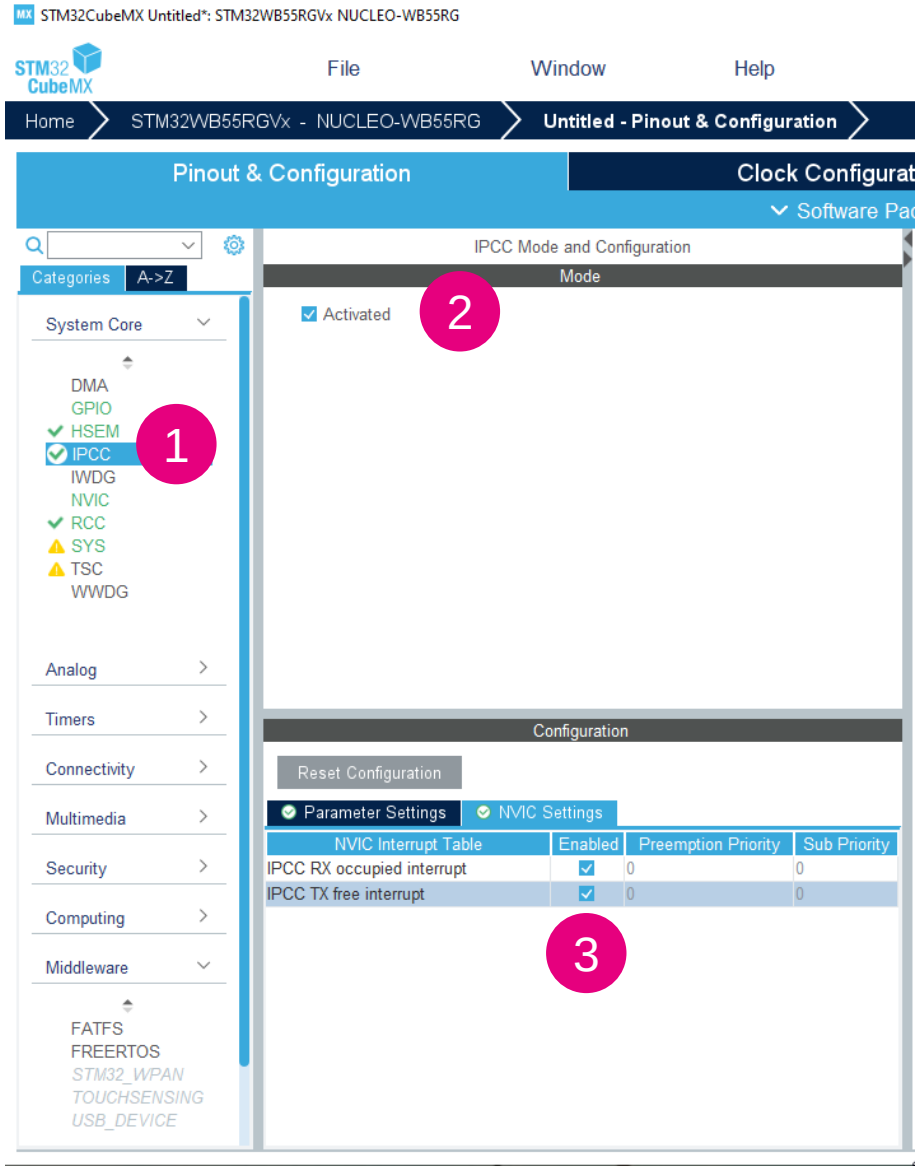


Hands-On

- Select IPCC (Inter-processor communication controller)
- Tick activate
- Enable both interrupts in NVIC

- IPCC provides asynchronous messaging mechanism between CM4 and CM0+
- Part of SRAM2 is shared



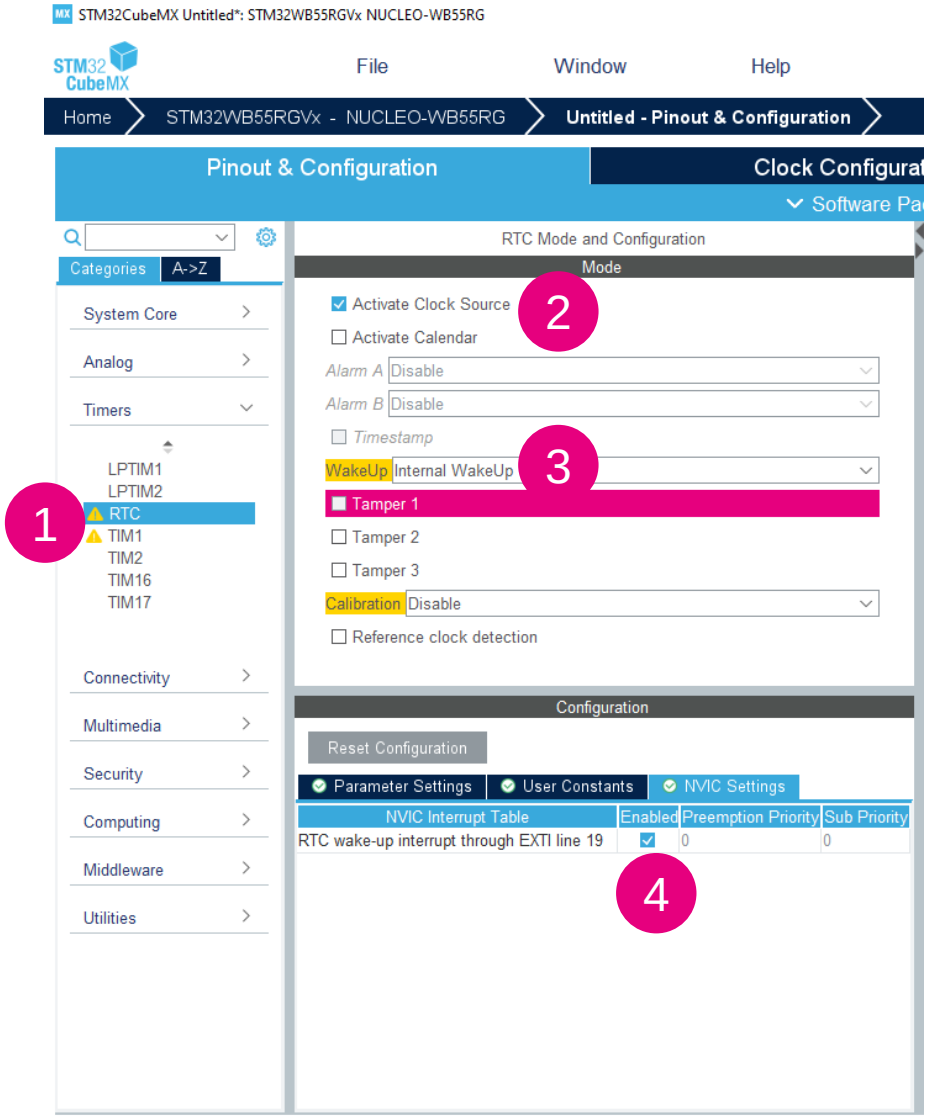


life.augmented

Hands-On

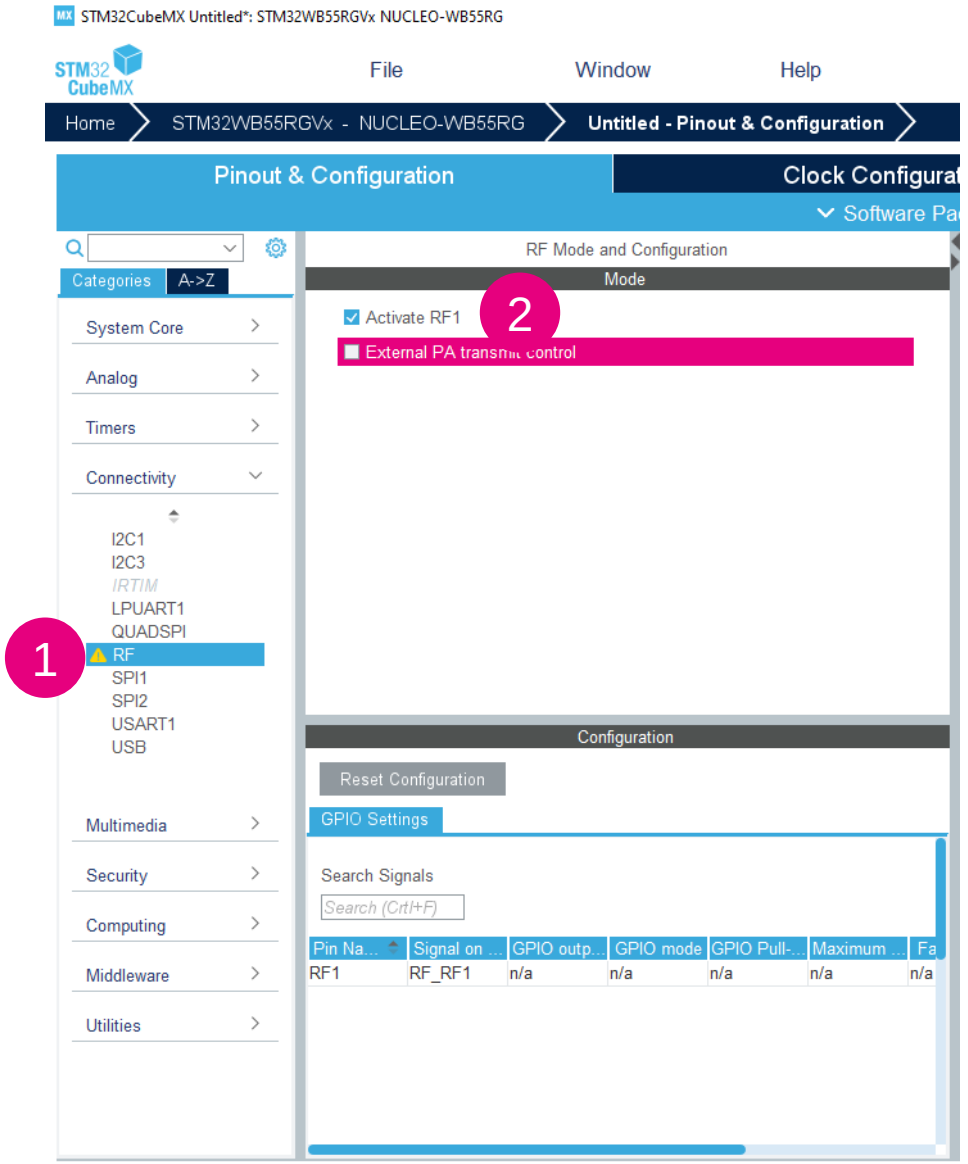
- Select RTC under Timers tab
- Tick Activate Clock Source
- Enable Internal WakeUp
- Enable interrupt in NVIC

- Required due to Virtual timer server FW component. (we will not use it in this hands on)



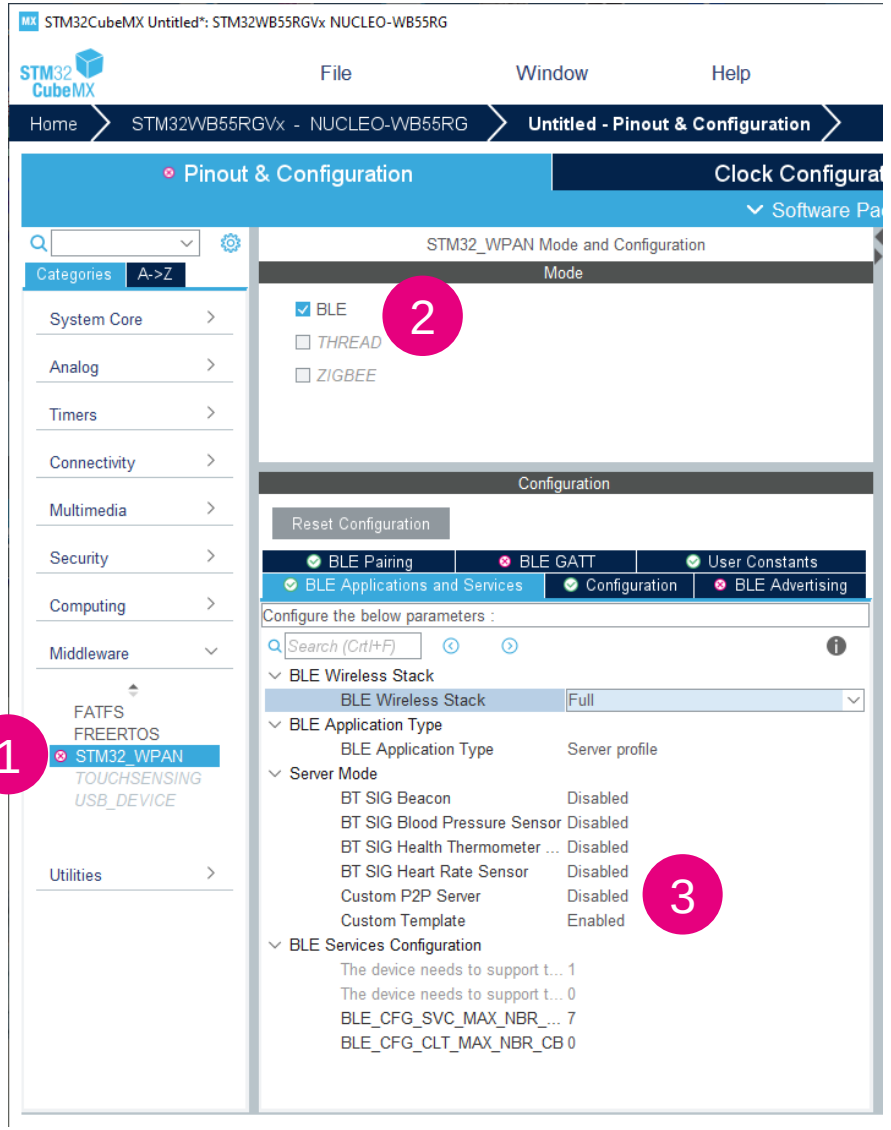
Hands-On

- Select RF under Connectivity tab
- Tick Activate RF1



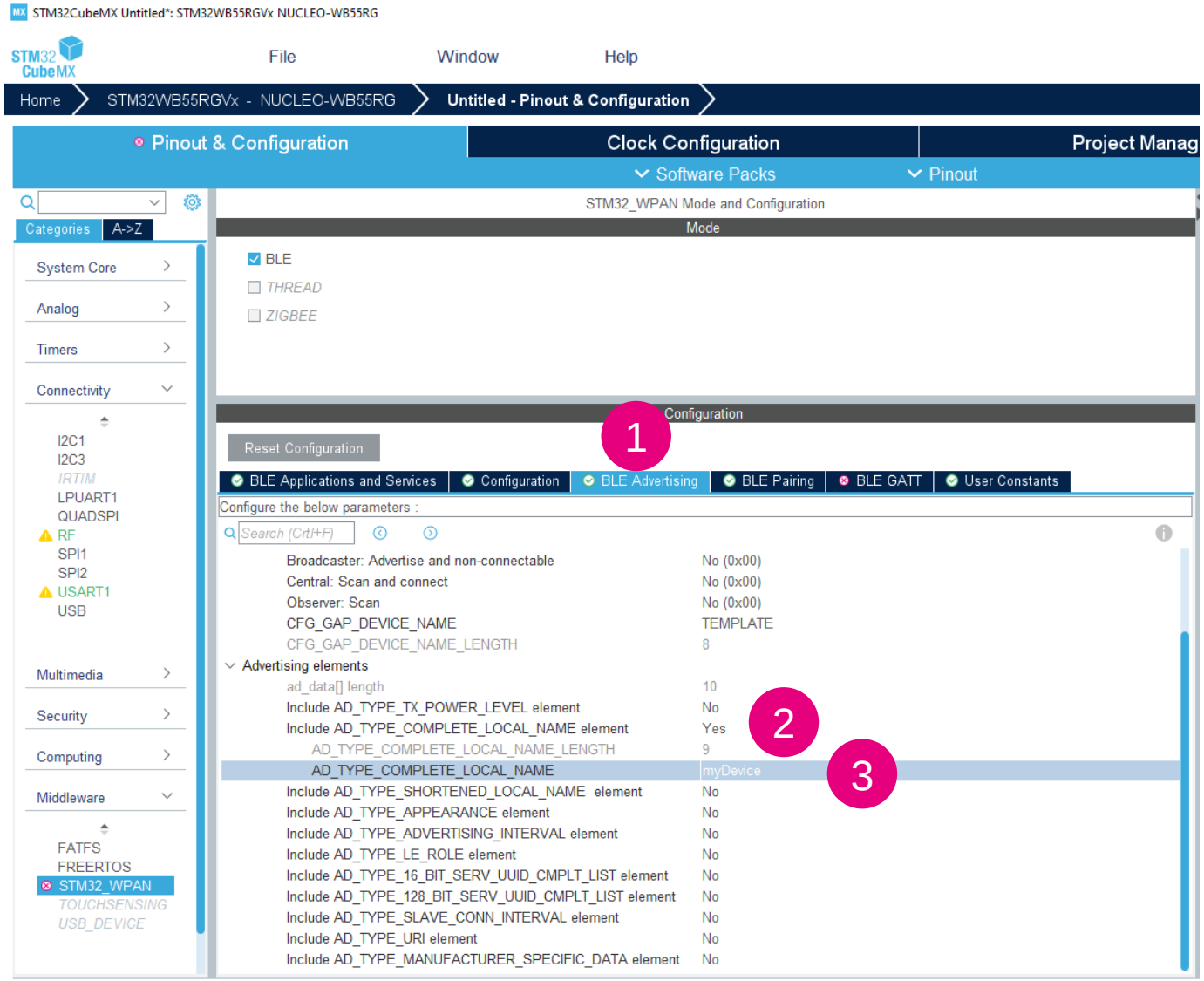
Hands-On

- Select STM32 WPAN Middleware under Middleware tab
- Tick BLE
- Disable Custom P2P Server
- Enable Custom Template



Hands-On

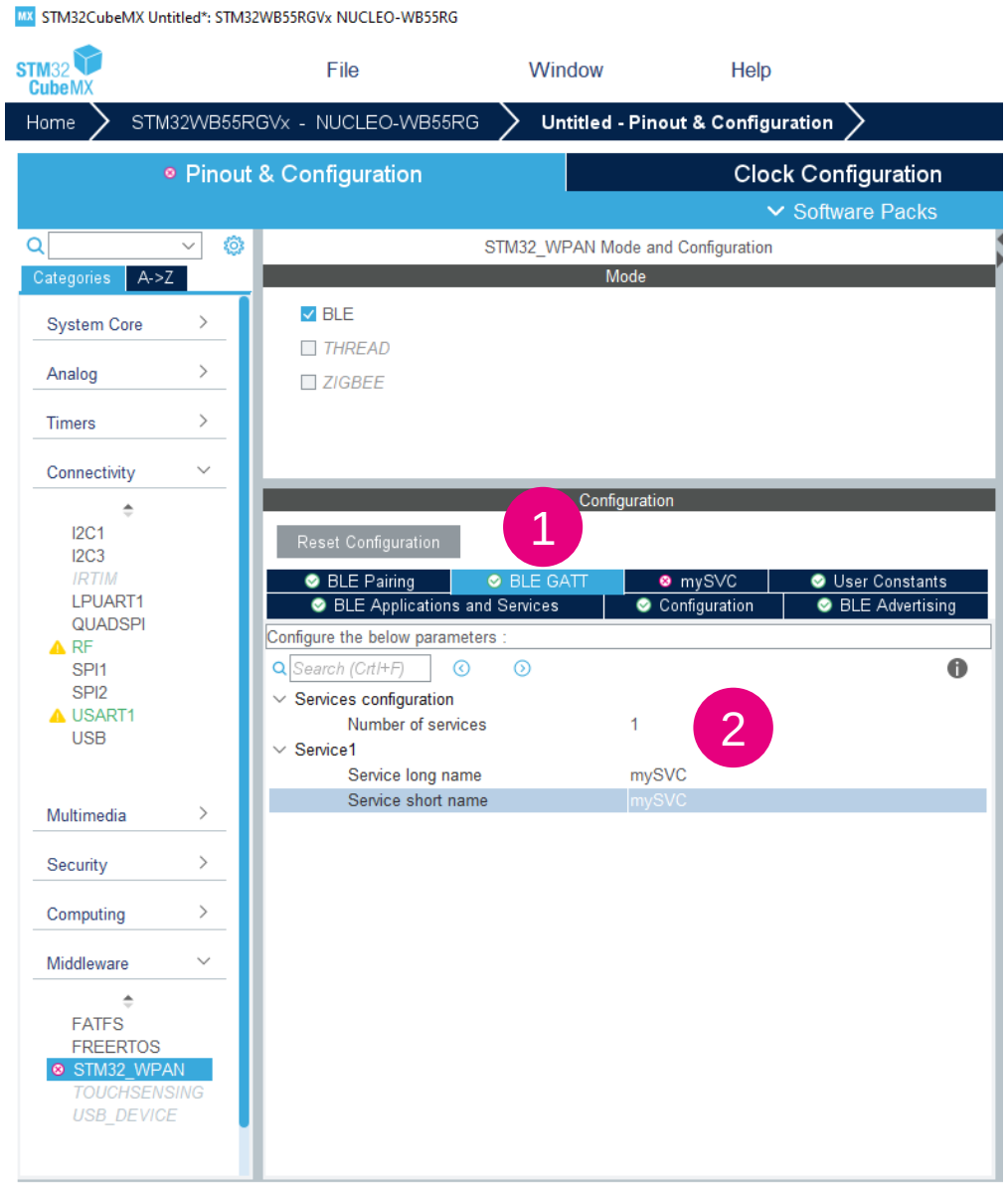
- Select BLE Advertising tab
- Include
  AD_TYPE_COMPLETE_LOCAL_NAME
  Yes
- Name your device uniquely



life.augmented

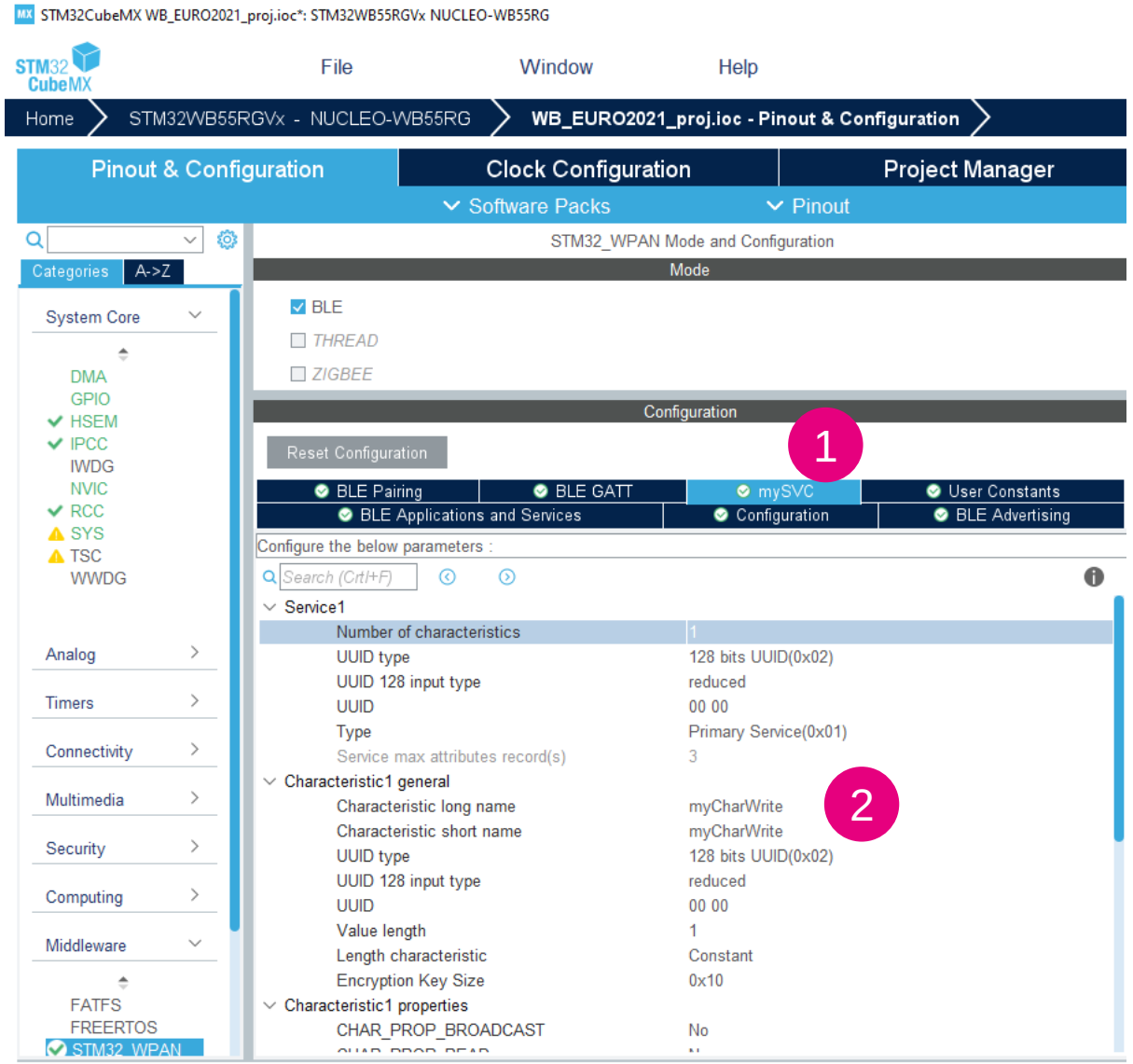
Hands-On

- Select BLE GATT tab
- Set number of services to 1
- Name your BLE service, both long and short name



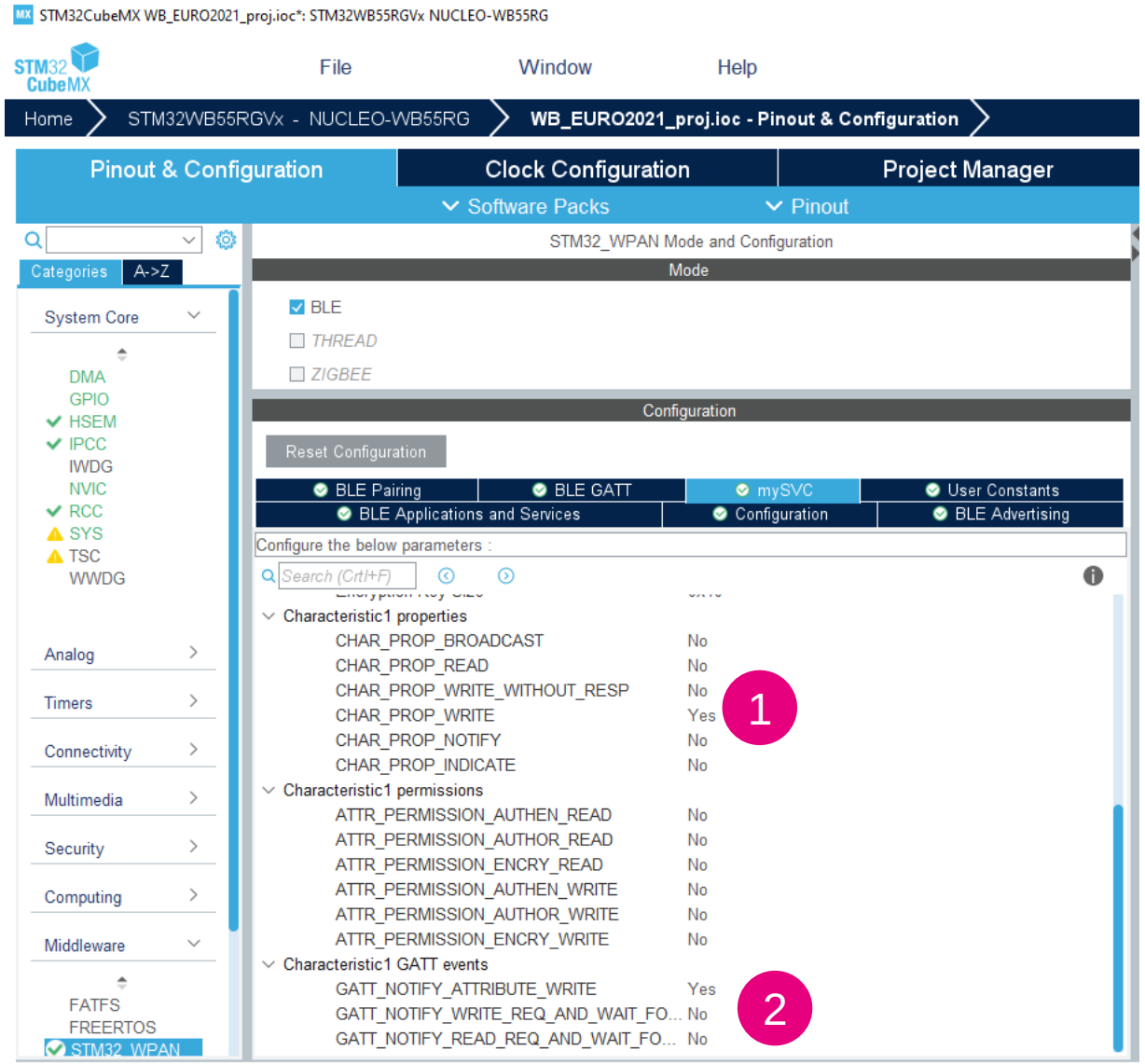
Hands-On

- Select ‘mySVC’ tab. The exact name depends on your previous selection
- Name you characteristic ‘myCharWrite‘ as both long and short name
- Notice UUID - universally unique identifier



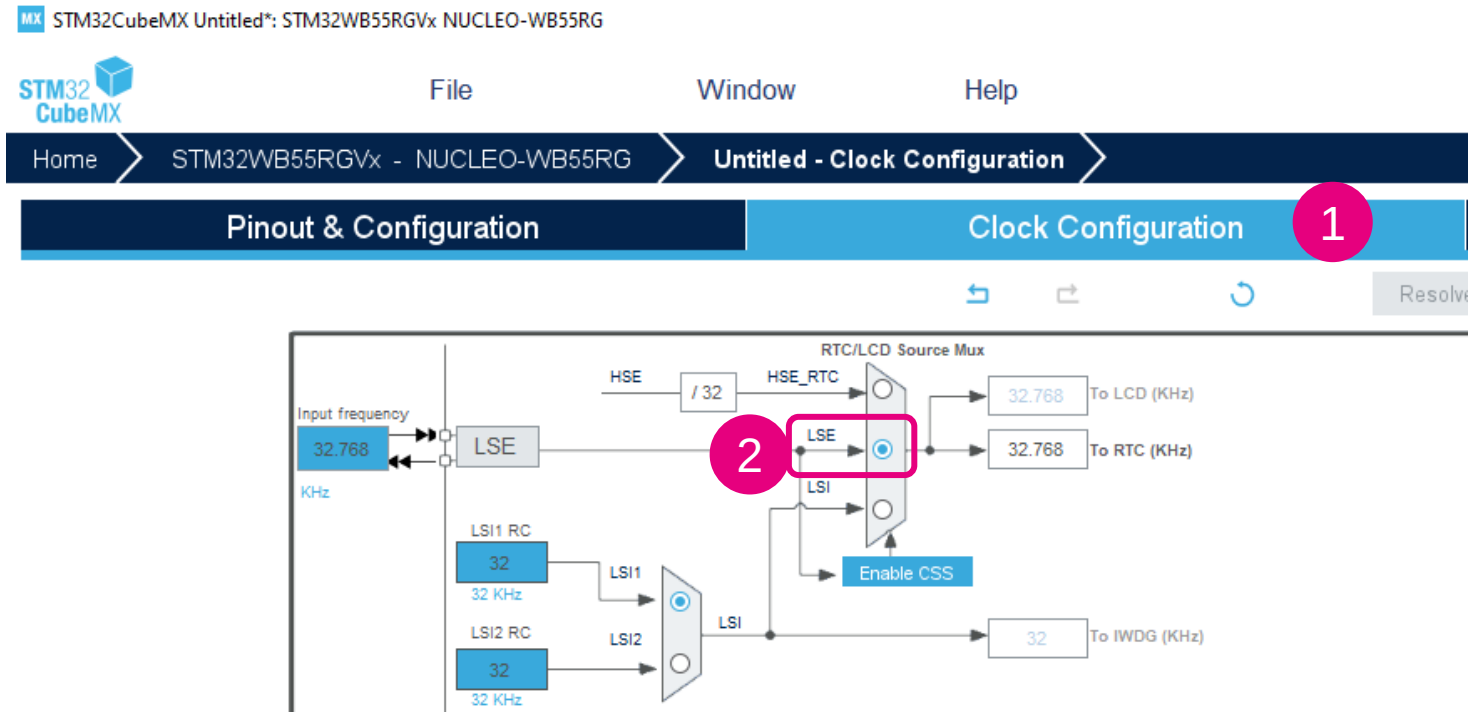
Hands-On

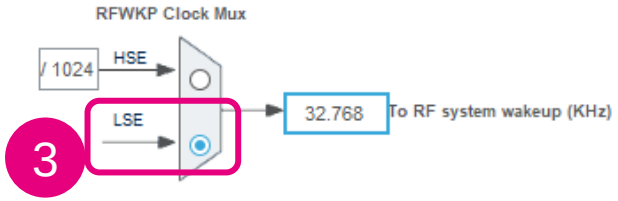
- Enable CHAR_PROP_WRITE property
- Disable all GATT events except GATT_NOTIFY_ATTRIBUTE_WRITE



Hands-On

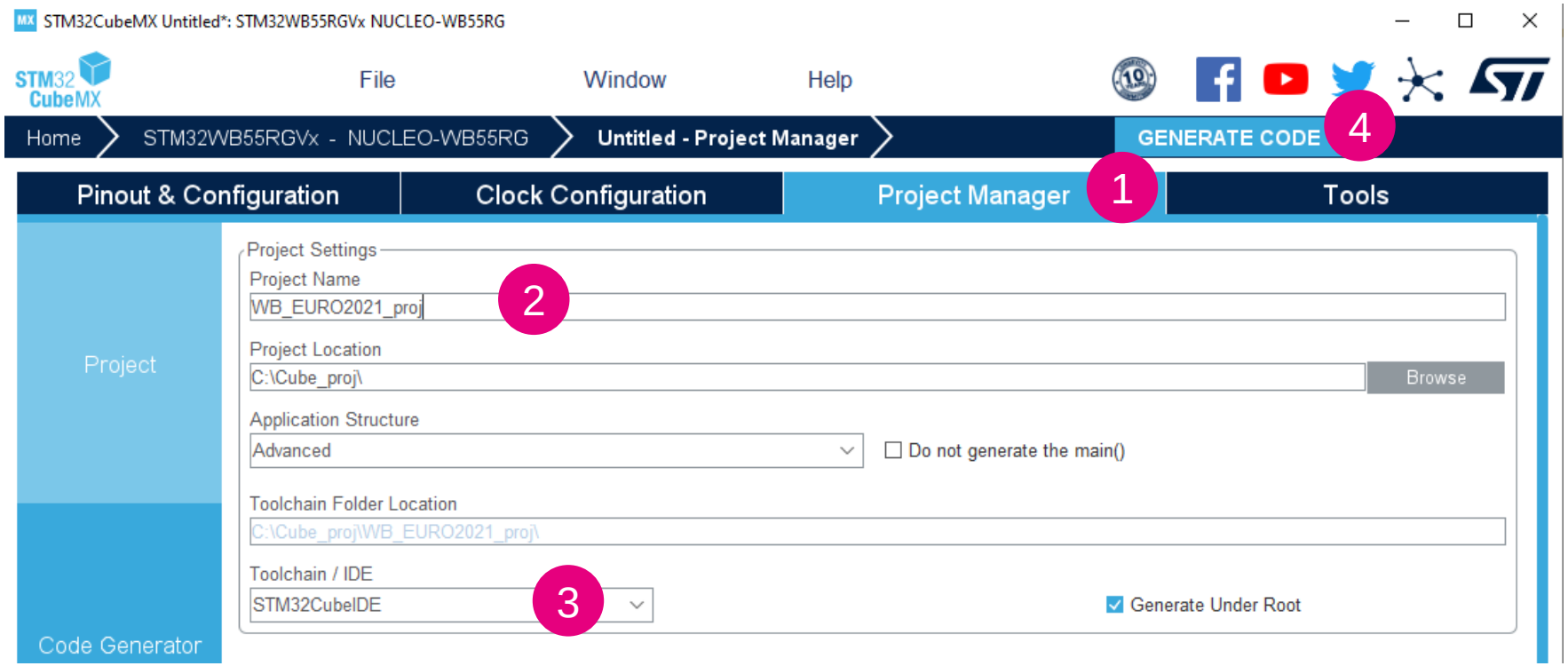
- Go to Clock configuration tab
- Set RTC clock to LSE
- Set RFWKP Clock to LSE



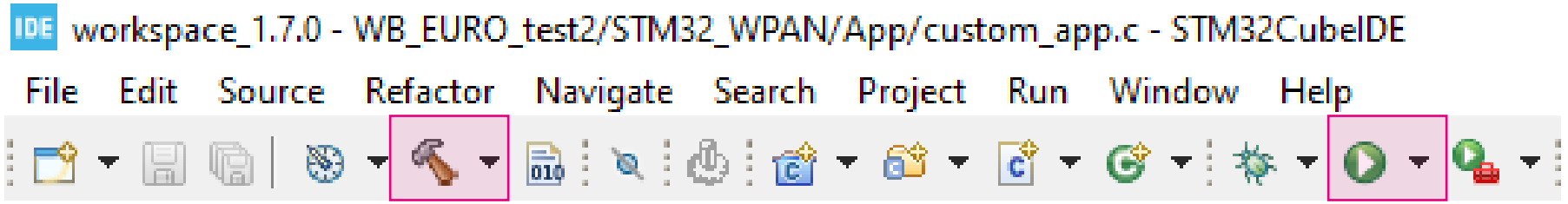


Hands-On

- Got to Project Manager
- Name your project
- Select Toolchain STMCubeIDE
- Generate code



Compile and run



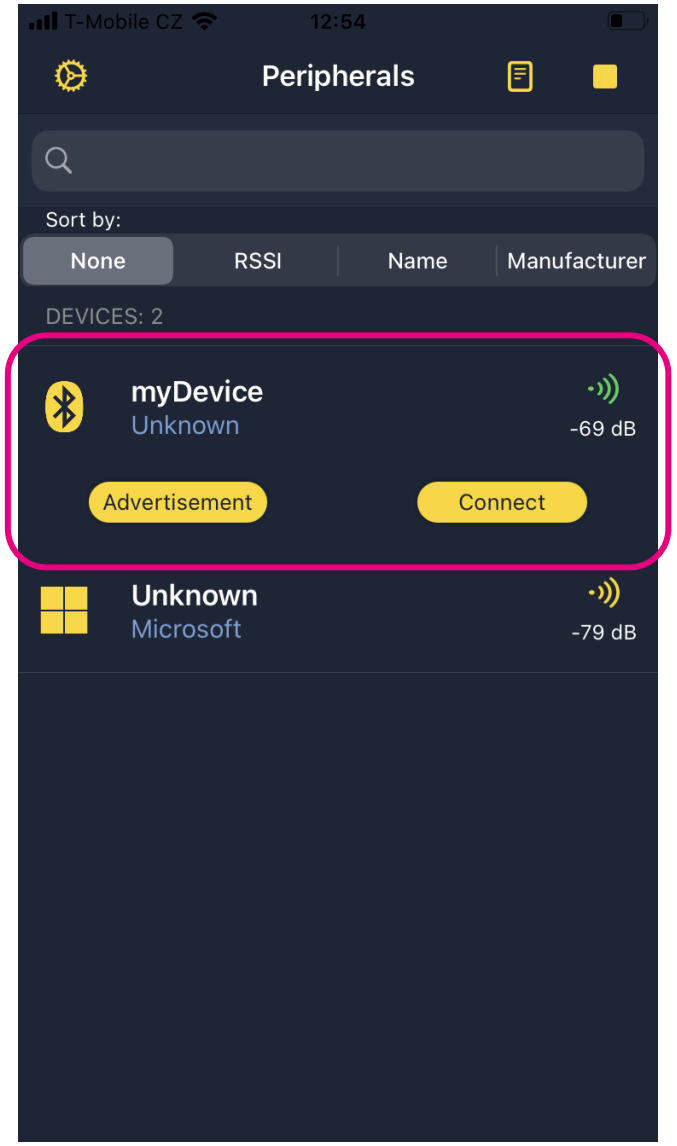
Build

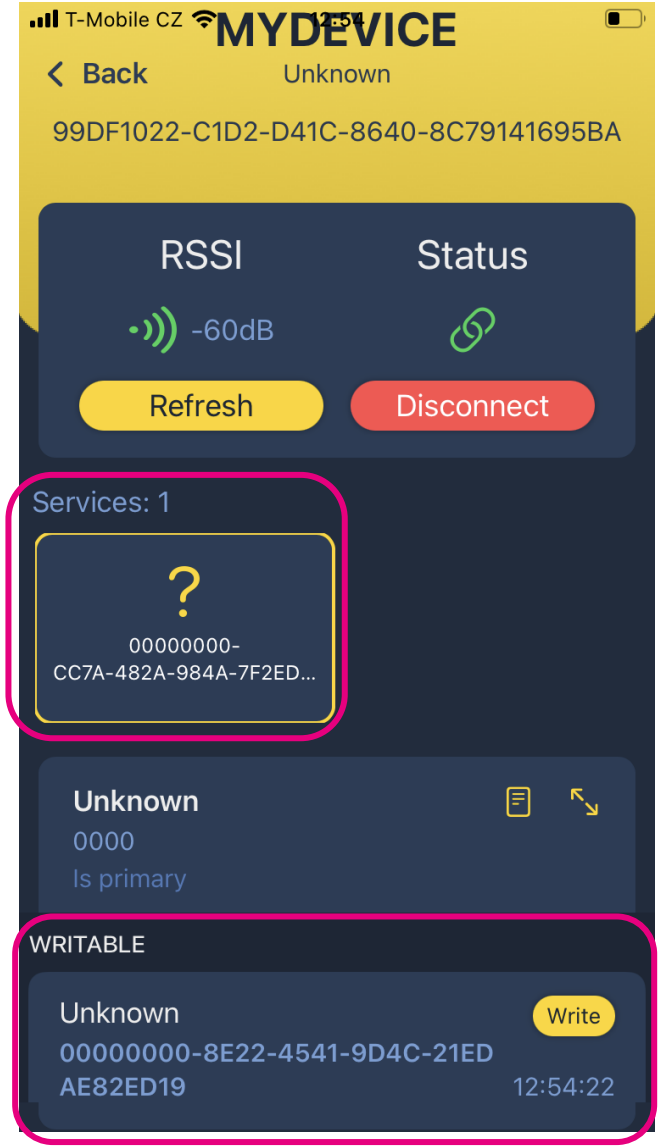
Run

Hands-On

- Open ST BLE Toolbox on your phone

- Connect to your device



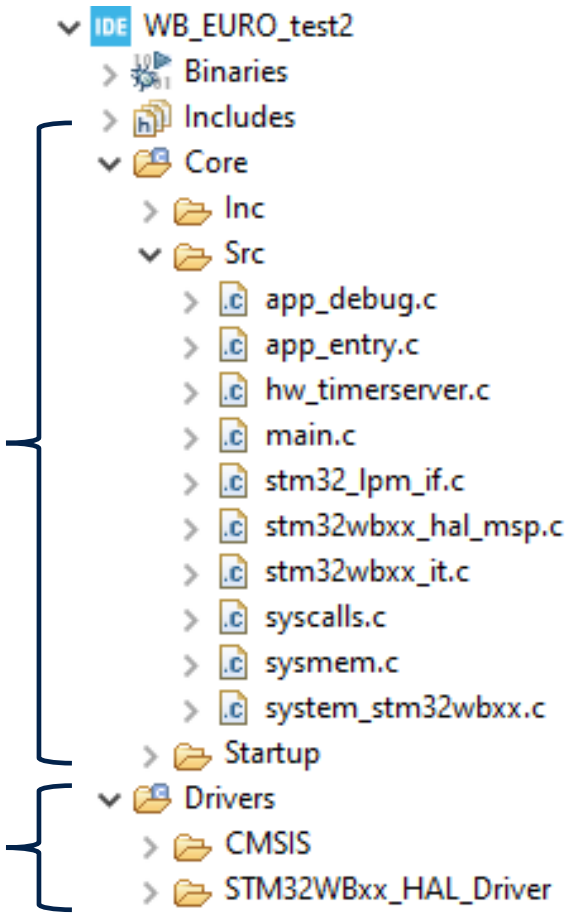


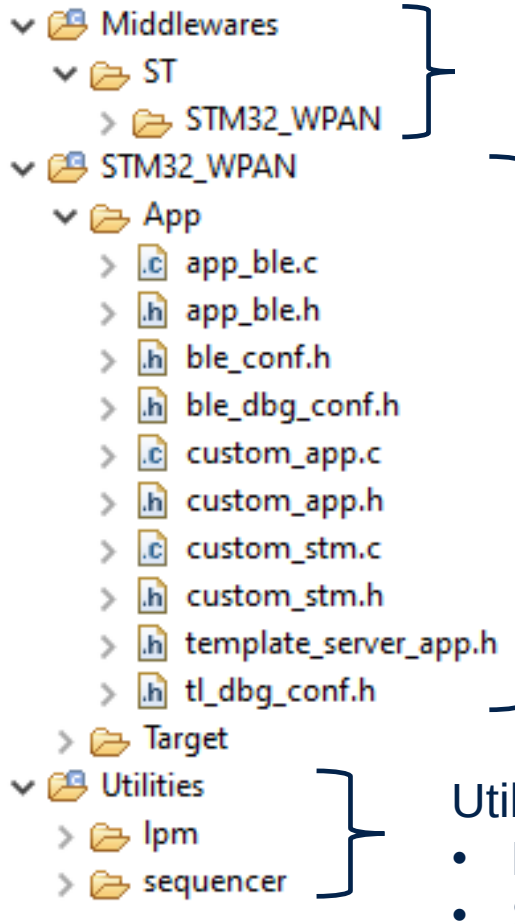
life.augmented

Project structure

Application and system initialization + Interrupt service routines

CMSIS and HAL Drivers





BLE Middleware

BLE application files. Configuration, BLE service definition. Event handlers etc.

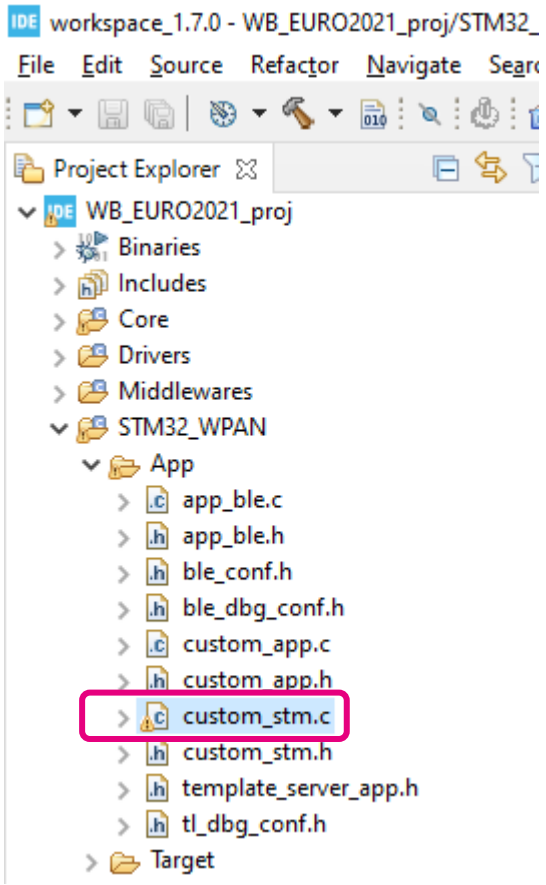
Utilities

- Low power manager
- Sequencer

life.augmented

Hands-On

- Toggle Green LED when BLE Stack reports that characteristic was written by the phone
- Open `custom_stm.c` and add highlighted code
- Compile and run

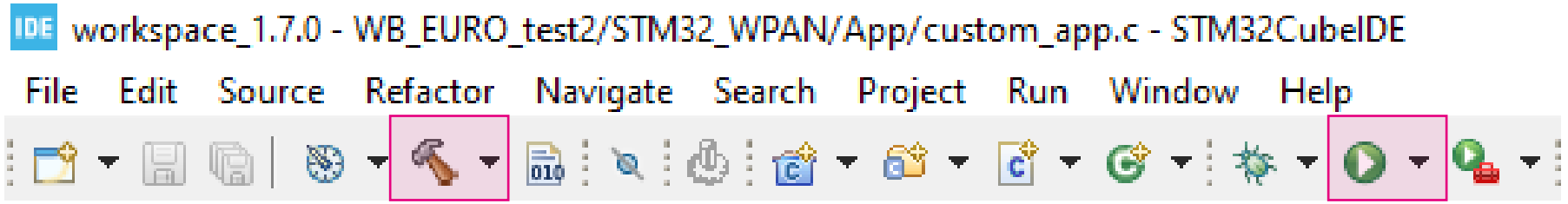


custom_stm.c

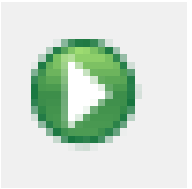
```
26 /* USER CODE BEGIN Includes */
27 #include "main.h"
28 /* USER CODE END Includes */
…

144           else if(attribute_modified->Attr_Handle == (CustomContext.CustomMycharwriteHdle + CHARACTERISTIC_VALUE_ATTRIBUTE_OFFSET))
145           {
146             return_value = SVCCTL_EvtAckFlowEnable;
147             /* USER CODE BEGIN CUSTOM_STM_Service_1_Char_1_ACI_GATT_ATTRIBUTE_MODIFIED_VSEVT_CODE */
148             HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin);
```

life.augmented

Compile and run

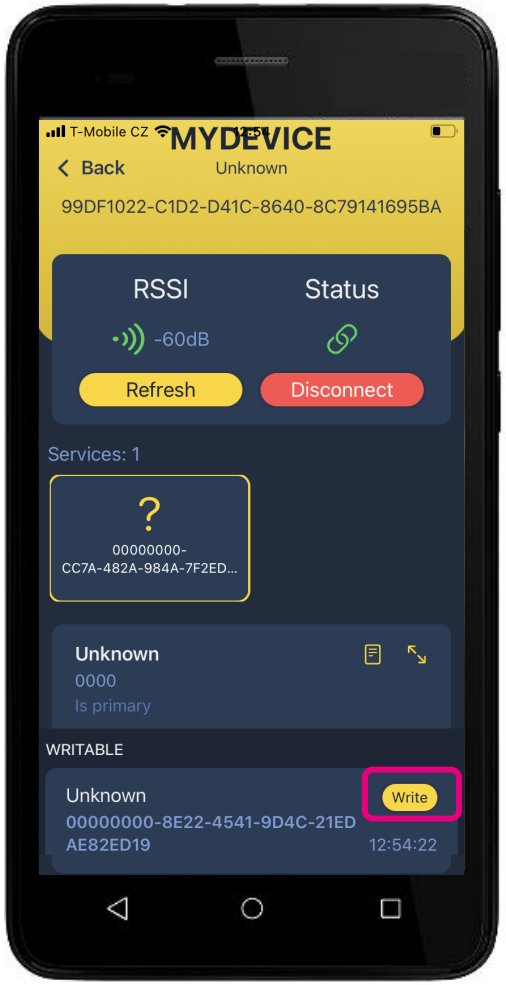


Build



Run

Hands-On

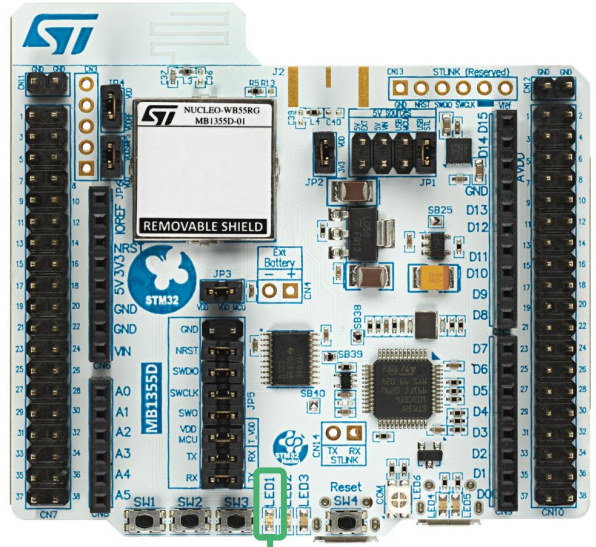


- Write a single arbitrary byte to the characteristic.
- Green LED will toggle
- Parsing the input value is shown as optional hands-on at the end of this presentation

Write

Custom SERVICE

Custom CHARACTERISTIC 1
LED control
W



Green LED

life.augmented

life.augmented

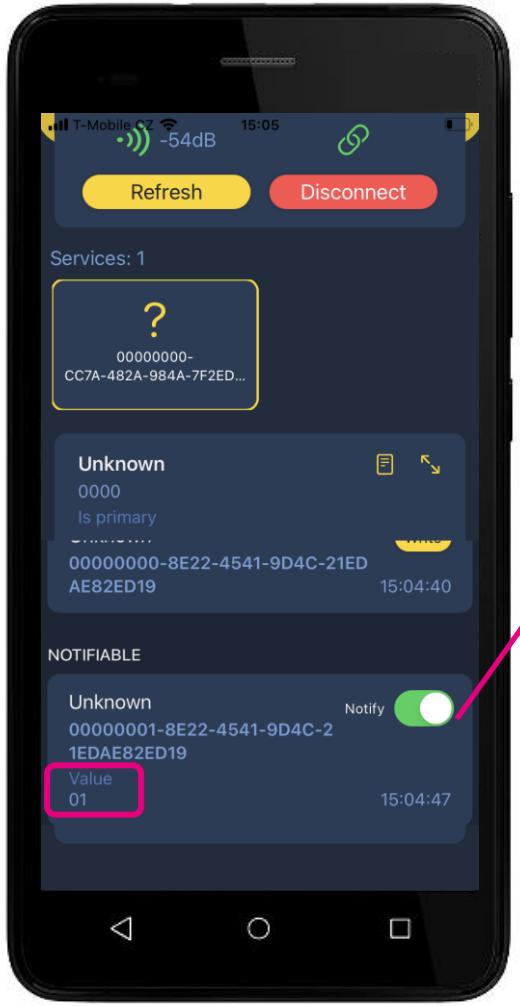
STM32WB Video Series – Getting Started

#12c STM32CubeMX & STM32CubeIDE lab
Sending notifications to the phone

T.O.M.A.S. Team

Hands-On part 2

Add Button 1 notification characteristic



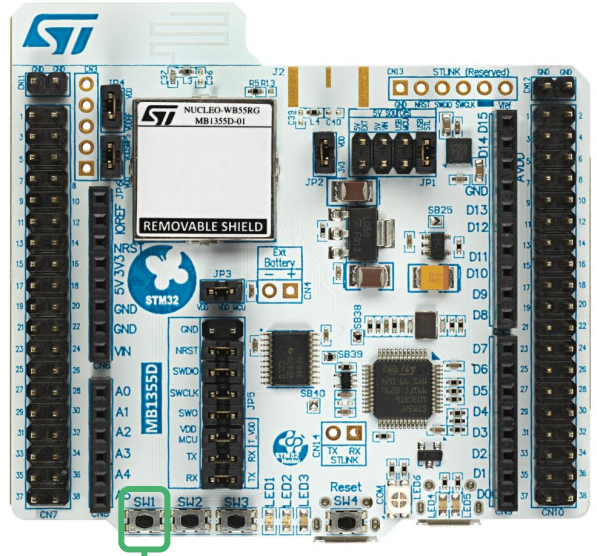
Enable notification
On Button 1 press notification will be send to the phone

Notification

Custom SERVICE

Custom CHARACTERISTIC 1
W LED control

Custom CHARACTERISTIC 2
N Button 1 notification

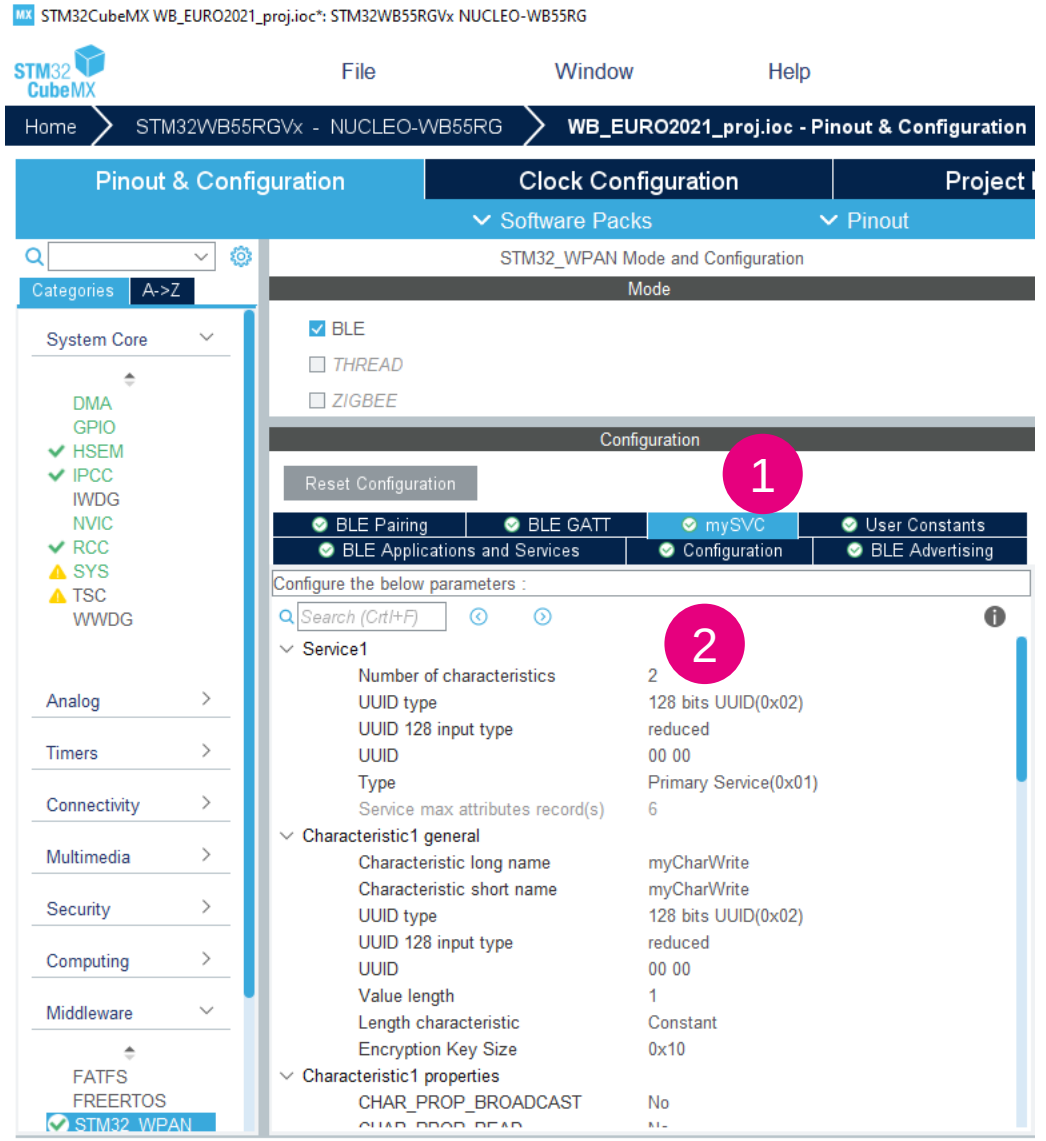


Button 1

life.augmented

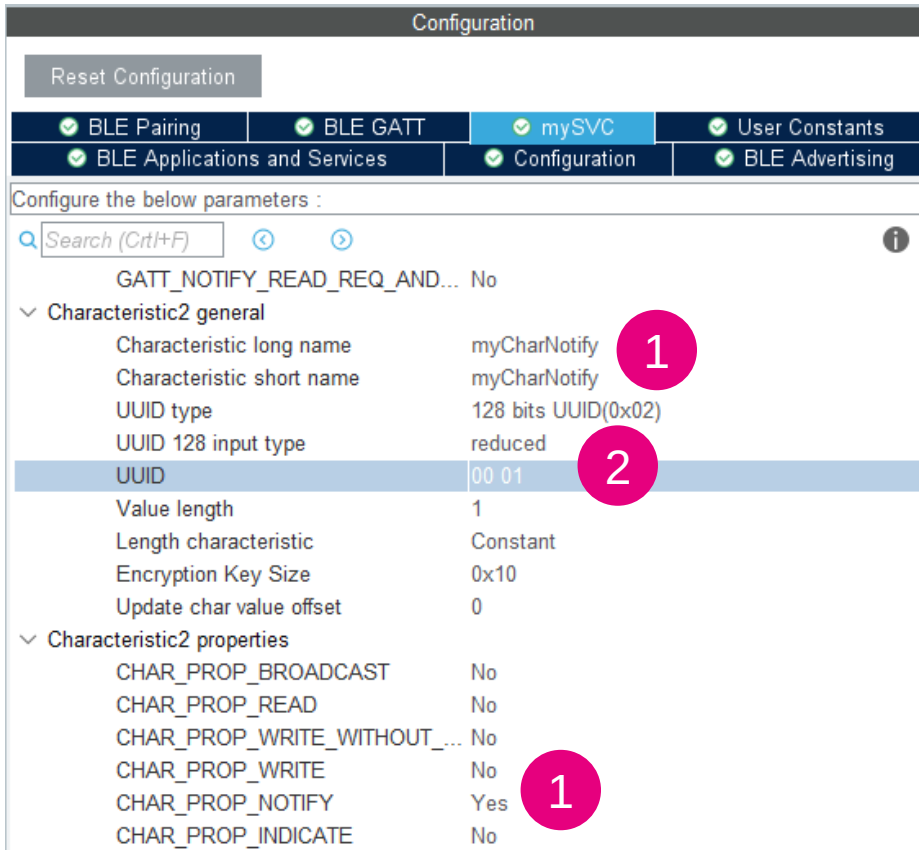
Hands-On

- Go back to CubeMX
- Select ‘mySVC’ tab. The exact name depends on your previous selection
- Add a second Characteristic.
  Set Number of characteristic to 2



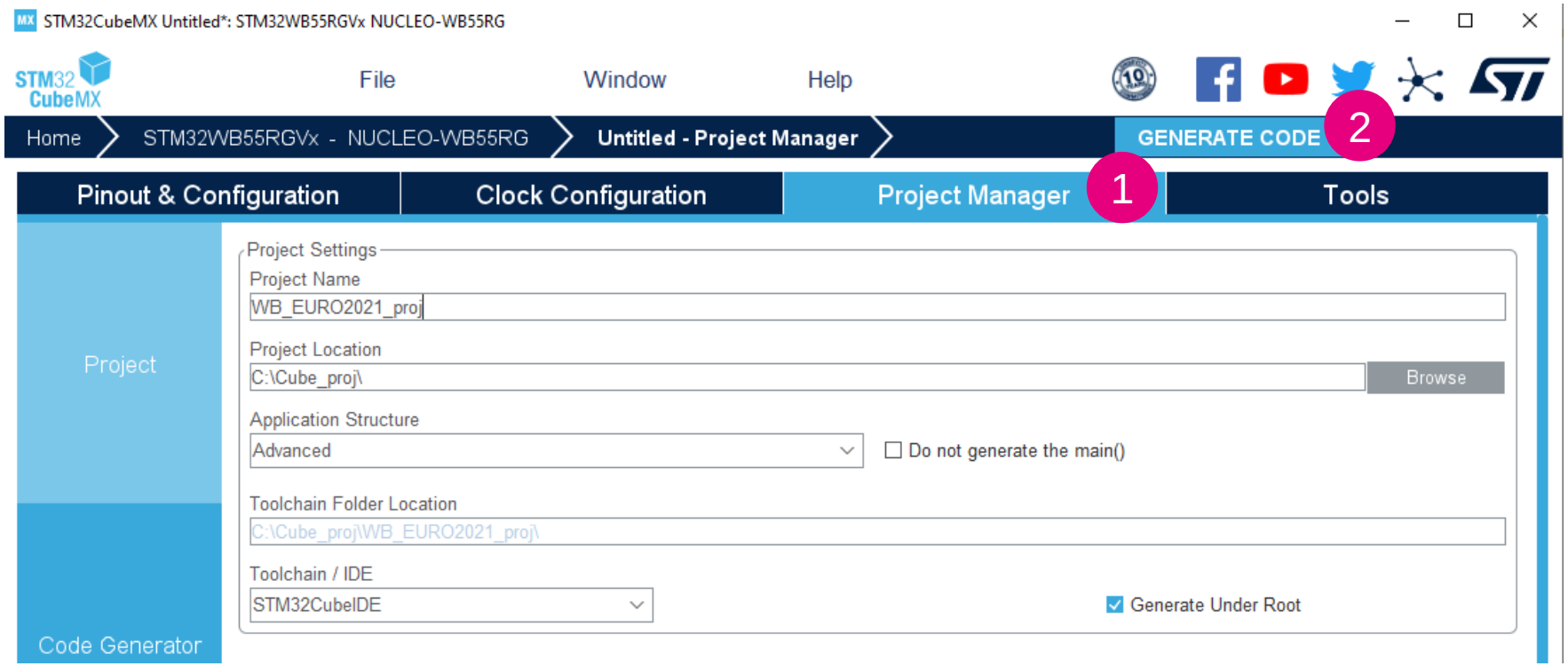
Hands-On

- Name the second characteristic ‘myCharNotify’ as both long and short name
- Increment the UUID to 00 01
- Enable CHAR_PROP_NOTIFY property



Hands-On

- Got to Project Manager
- Regenerate code



Hands-On

- Now we want to call `aci_gatt_update_char_value(…)` API on button press. This will send a CMD to CM0+ passing the value as an argument
- This cannot be done from interrupt context, because it is not safe (ACI interface is not not reentrant).
- We need to create a task that runs in the background.

Hands-On

app_conf.h 1

```
543 /* USER CODE BEGIN Defines */
544 void myTask(void);
545 /* USER CODE END Defines */
…
558 /**< Add in that list all tasks that may send a ACI/HCI command */
559 typedef enum
560 {
561     CFG_TASK_ADV_CANCEL_ID,
562 #if (L2CAP_REQUEST_NEW_CONN_PARAM != 0 )
563     CFG_TASK_CONN_UPDATE_REG_ID,
564 #endif
565     CFG_TASK_HCI_ASYNCH_EVT_ID,
566 /* USER CODE BEGIN CFG_Task_Id_With_HCI_Cmd_t */
567           CFG_TASK_MY_TASK,
568 /* USER CODE END CFG_Task_Id_With_HCI_Cmd_t */
```

app_ble.c 2

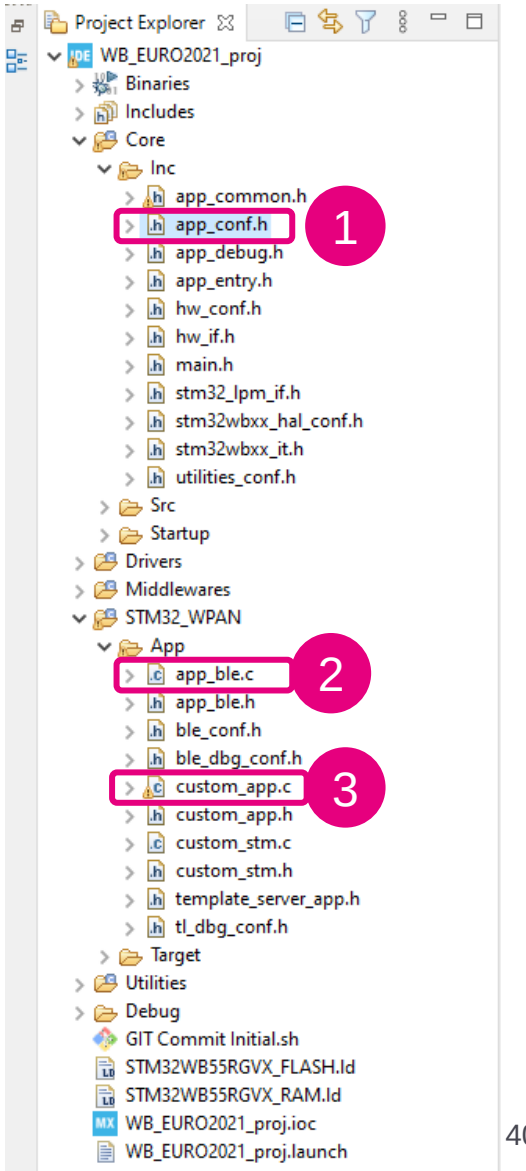
```
253 /* Functions Definition ------------------------------------*/
254 void APP_BLE_Init( void )
255 {
256 /* USER CODE BEGIN APP_BLE_Init_1 */
257    UTIL_SEQ_RegTask( 1<<CFG_TASK_MY_TASK, UTIL_SEQ_RFU, myTask);
258    UTIL_SEQ_SetTask(1 << CFG_TASK_MY_TASK, CFG_SCH_PRIO_0);
```

custom_app.c 3

```
85 /* USER CODE BEGIN PFP */
86 void myTask(void)
87 {
88   if(!HAL_GPIO_ReadPin(B1_GPIO_Port, B1_Pin))
89   {
90           UpdateCharData[0] ^= 0x1;
91           Custom_Mycharnotify_Update_Char();
92   }
93   UTIL_SEQ_SetTask(1 << CFG_TASK_MY_TASK, CFG_SCH_PRIO_0);
94 }
95 /* USER CODE END PFP */
```

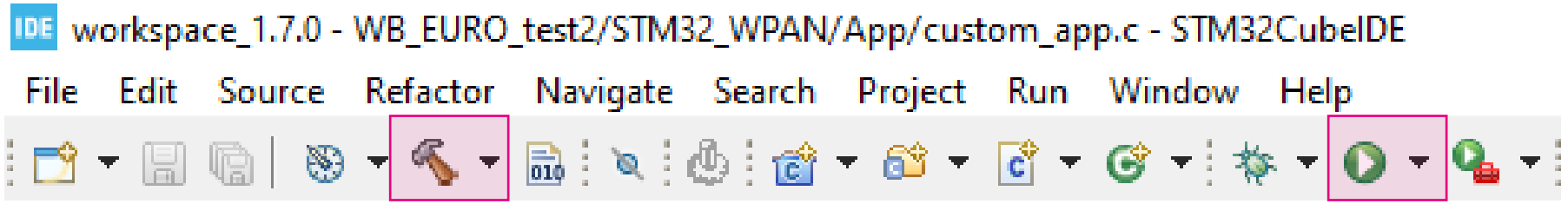
Right click to open finished file and copy paste, CTRL + A, CTRL + C/V

Careful!!! The function name depends on your naming. You might need to modify

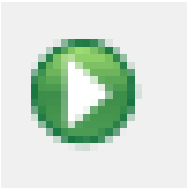


life.augmented

Compile and run



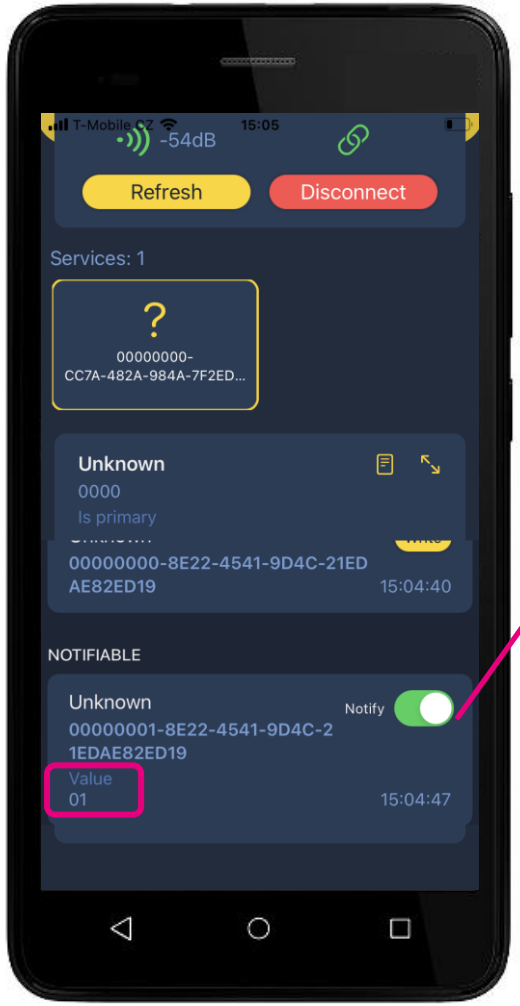
Build



Run

Hands-On part 2

Add Button 1 notification characteristic



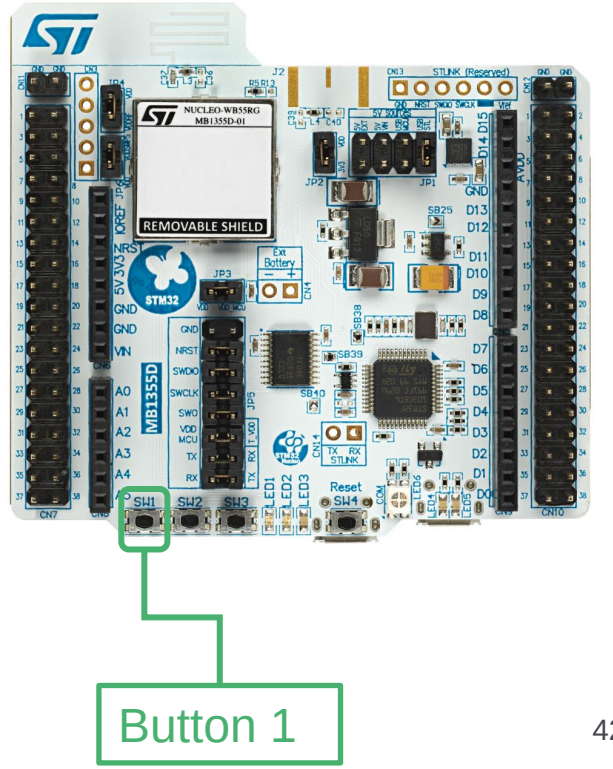
Enable notification
On Button 1 press notification will be send to the phone

Notification

Custom SERVICE

Custom CHARACTERISTIC 1
W
LED control

Custom CHARACTERISTIC 2
N
Button 1 notification



life.augmented

What we learned?

- How to create BLE project from scratch in CubeMX
- Which peripherals are required by BLE Middleware
- How to define Custom BLE Service and how to add characteristics
- How to process events generated by BLE Stack
- How to call BLE Stack API safely

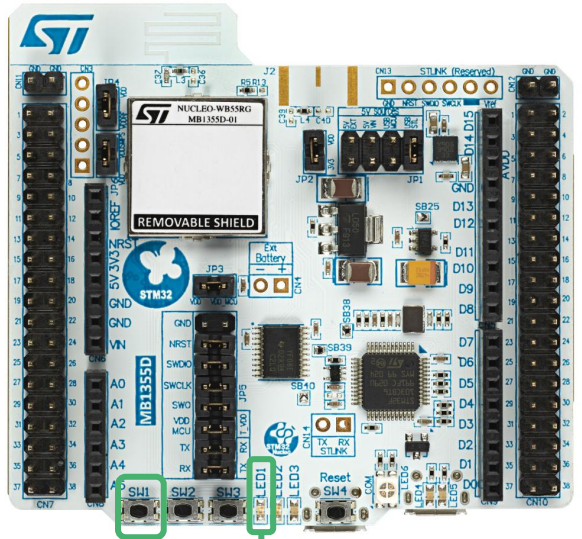
Write

Notification

Custom SERVICE

Custom CHARACTERISTIC 1
W
LED control

Custom CHARACTERISTIC 2
N
Button 1 state



Button 1

Green LED

life.augmented

Thank you

life.augmented