



life.augmented



Hands-on
Build basic **p2pServer**
application and connect





Agenda

1 Hands-on presentation

2 Step 1: STM32CubeMX initialization for STM32WBA Nucleo board

3 Step2 : Advertising and BLE application configuration and explanation

4 Step 3 : Code generation and user application code

5 Step 4 : Adding logs

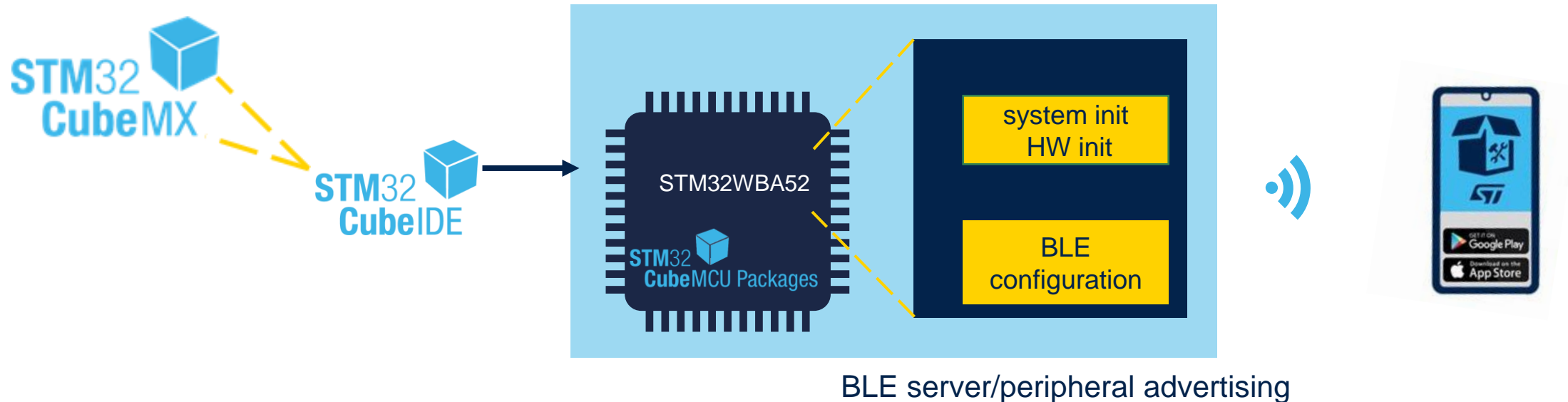
6 Step 5 : Adding service, profile and characteristic



life.augmented

Hands-On presentation

- The purpose is to start from WB5A52 chipset level and build a basic server (p2pServer) application using STM32CubeMX and associated STM32CubeIDE
- In this first part, focus is to get device **visible and connectable** from my smartphone



Unpack NUCLEO-WBA52, plug to laptop,
install your favorite ST BLE ToolBox App and Let's start !



Source

STM32 MCU

Hands-On based on

https://wiki.st.com/stm32mcu/wiki/Connectivity:STM32WBA_BLE_STM32CubeMX

Approved version. Approved on: 09:32, 21 July 2023

Connectivity: STM32WBA BLE STM32CubeMX

Last edited one week ago ago

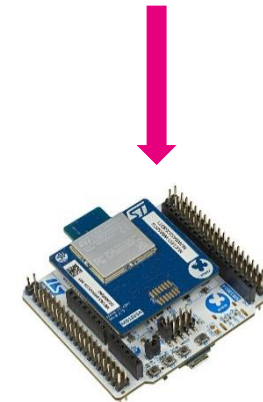
STM32WBA Bluetooth® Low Energy - STM32CubeMX Application Conception

Click [here](#) for Bluetooth® Low Energy Overview.
Click [here](#) for an Introduction to Bluetooth® Low Energy with STM32.

Contents [hide]

- 1 Introduction
- 2 Build a Bluetooth® Low Energy application on STM32WBA with STM32CubeMX
 - 3 Tools
 - 3.1 Software tools
 - 3.2 Hardware tools
- 4 STM32CubeMX initialization for STM32WBA Nucleo board
 - 4.1 STM32CubeMX initial setup

step by step guideline to build a BLE peripheral application



BLE server/peripheral
exposing data to central device

Step 1 : STM32CubeMX initialization for STM32WBA Nucleo board

STM32CubeMx capabilities



STM32CubeMx allow to start design within 3 options

1

Example application

complete application running over NUCLEO

2

Board level

all the hardware is already configured (NUCLEO_WBA52)

3

Chipset level

require to configure your HW (PCB) & your application

[STM32WBA wiki
page focus](#)

Hands-on focus. As customer let's build my own App



STM32CubeMx design from chispet level complete journey

STM32CubeMx initialisation for STM32WBA Nucleo board



STM32WBA IPs & peripherals configuration



Clock Tree configuration



BLE configuration : Advertising, Service, Characteristic



Hands-on
Focus

Code generation & application code management over CubeIDE





STM32CubeMx design from chipset level

Hands-on focus (1/2)

3

Chipset level

require to configure your HW (PCB) & your application

To ease Hands-on session use [Hands-on_WS_WBA52.ioc](#)
All HW IPs & required peripheral to use RF are already initialized : NVIC, RNG, RCC,...
Thanks to [Hands-on_WS_WBA52.ioc](#) let's focus on BLE application design



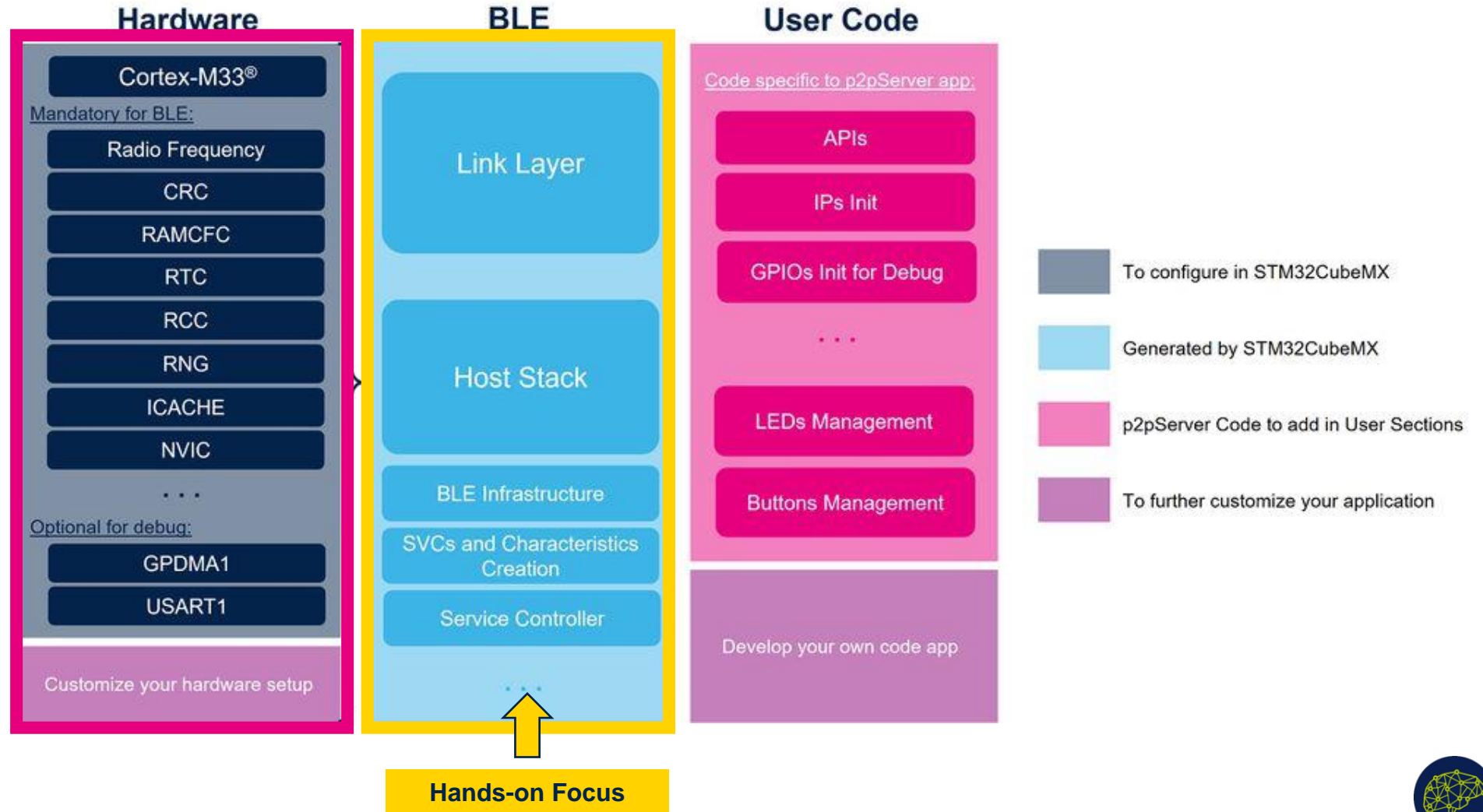
Copy Hands-on_WS_WBA52.ioc on your local repository :
example : C:\users\...\STM32WBA_WS\project



STM32CubeMx design from chispet level

Hands-on focus (2/2)

Hands-On_WS_WBA52.ioc





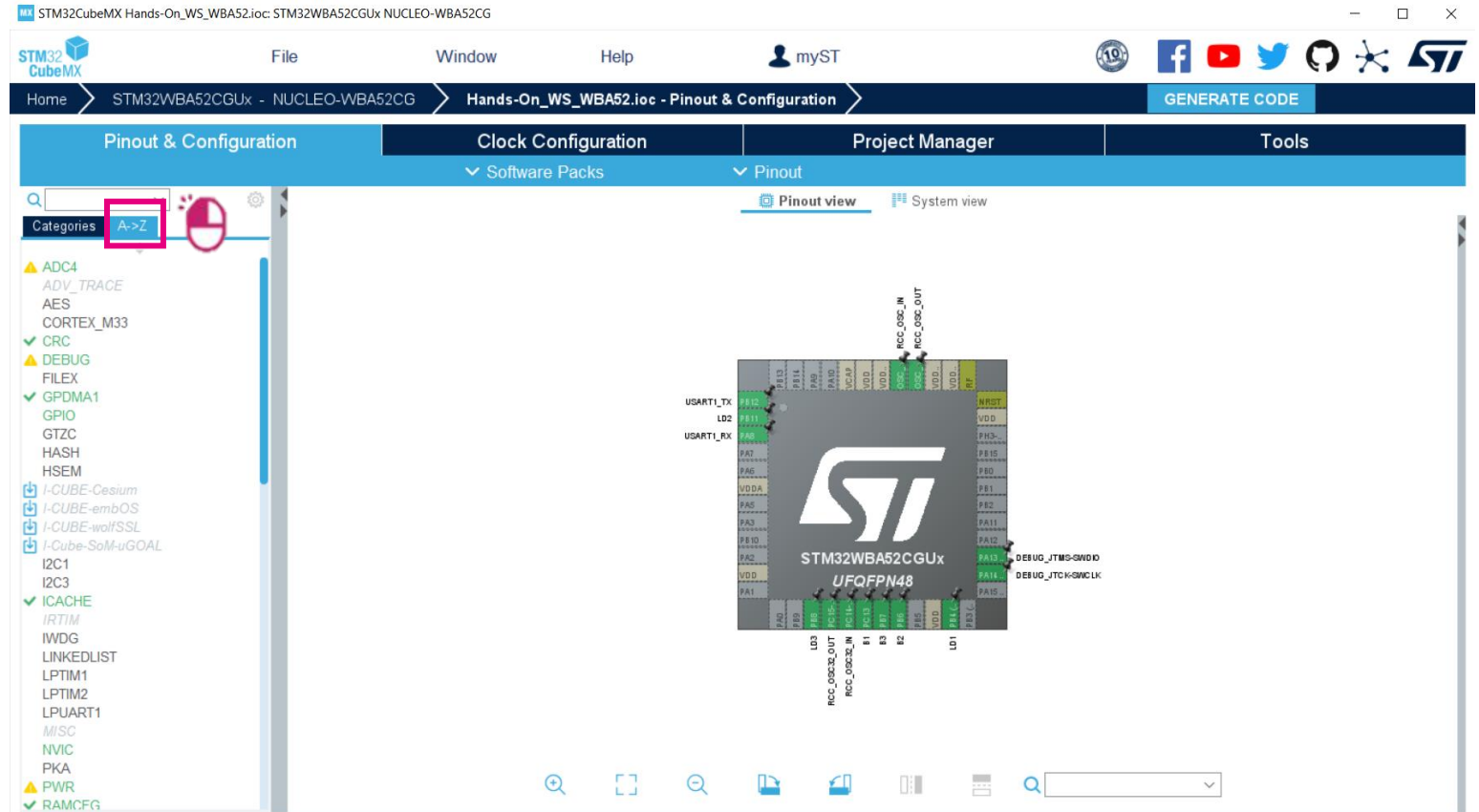
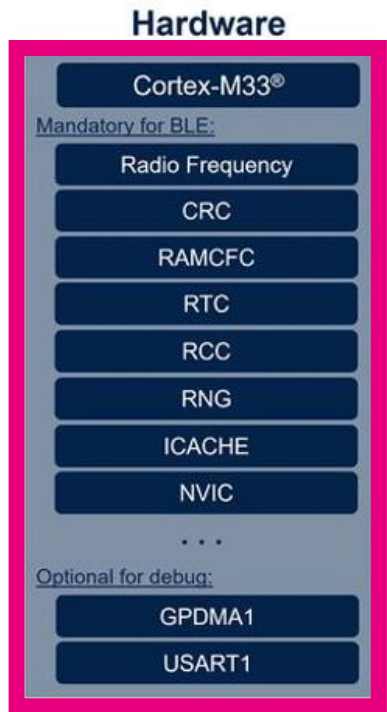
Start STM32CubeMX

CubeMx 6.9.0

The screenshot displays the STM32CubeMX 6.9.0 interface. The top menu bar includes 'File', 'Window', and 'Help'. The left sidebar shows 'Existing Projects' and 'Recent Opened Projects'. The main workspace is divided into 'New Project' and 'Manage software installations' sections. A 'Load Project' dialog box is open, showing the file 'Hands-On_WS_WBA52.ioc' selected. The 'File' menu is open, and the 'Load Project' option is highlighted. The 'Load Project' dialog box shows the file name 'Hands-On_WS_WBA52.ioc' and the file type 'STM32CubeMX project Files (.ioc)'. The 'Open' button is visible. The background shows the 'Pinout & Configuration' tab selected, displaying the pinout configuration for the STM32WBA52CGUx NUCLEO-WBA52CG.



Peripherals in place to start BLE configuration !



Hands-On_WS_WBA52.ioc

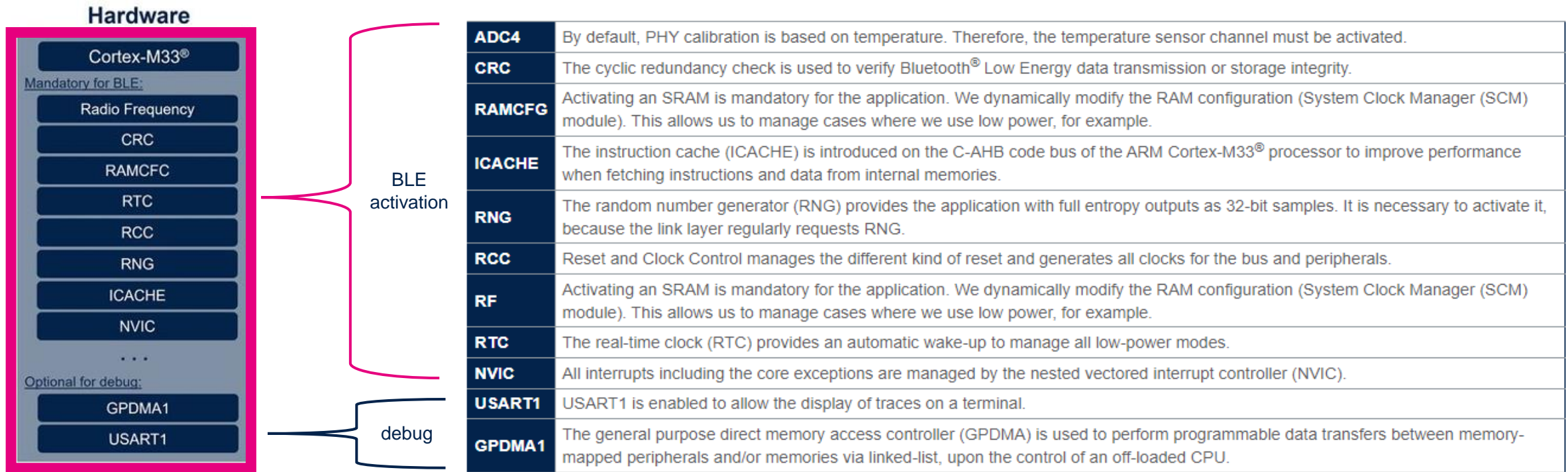
- HW configuraton
- enable STM32_WPAN (**BLE middleware activation**)

Peripherals in place to start BLE configuration !

Wiki explanations

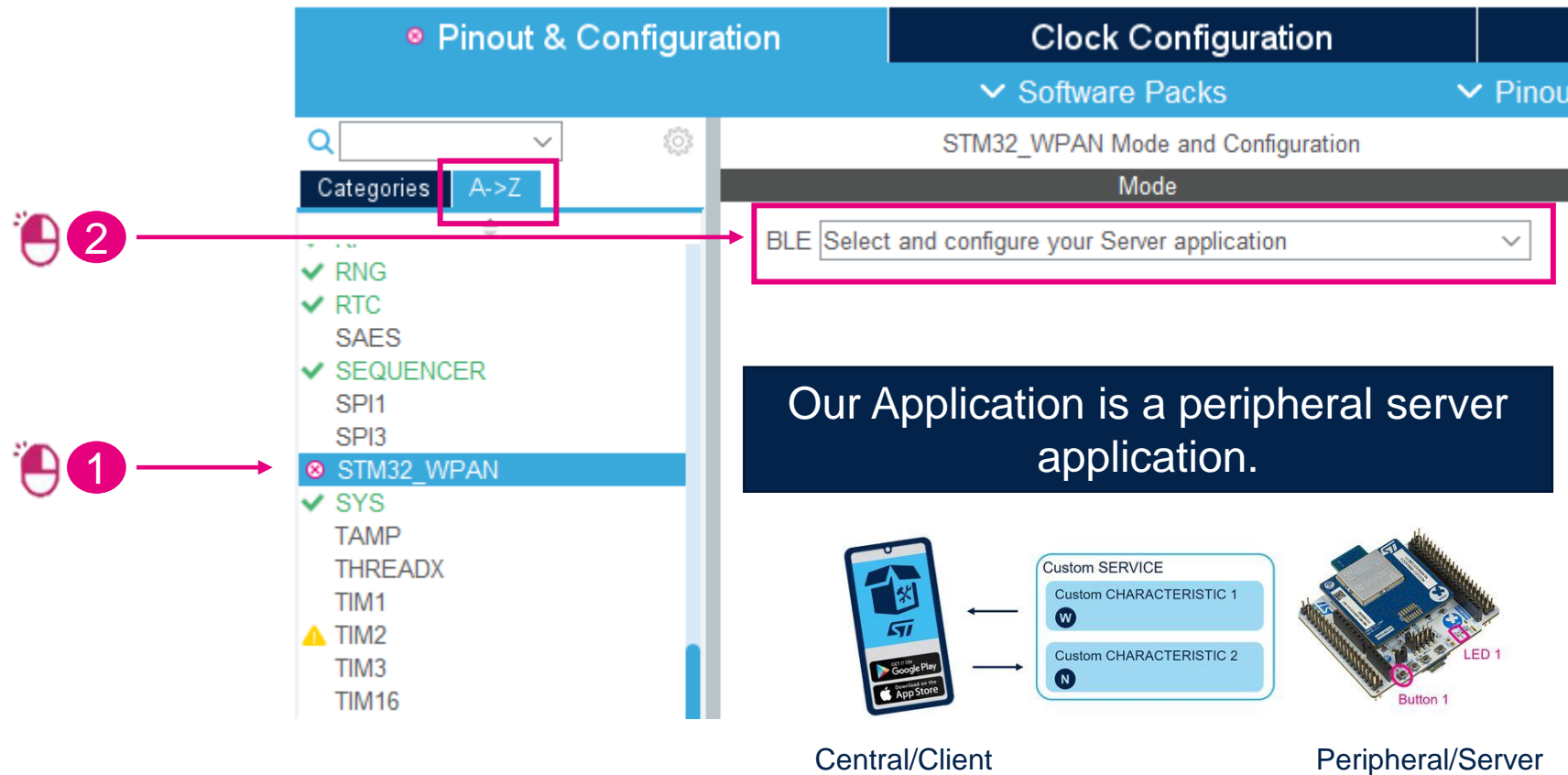


https://wiki.st.com/stm32mcu/wiki/Connectivity:STM32WBA_BLE_STM32CubeMX



Step2 : Advertising and BLE GAP/GATT custom application configuration

Enabling Bluetooth Low Energy



Pinout & Configuration

Clock Configuration

Software Packs

Pinout

STM32_WPAN Mode and Configuration

Mode

BLE Select and configure your Server application

Categories A->Z

RNG

RTC

SAES

SEQUENCER

SPI1

SPI3

STM32_WPAN

SYS

TAMP

THREADX

TIM1

TIM2

TIM3

TIM16

Our Application is a peripheral server application.

Central/Client

Peripheral/Server

Custom SERVICE

Custom CHARACTERISTIC 1

W

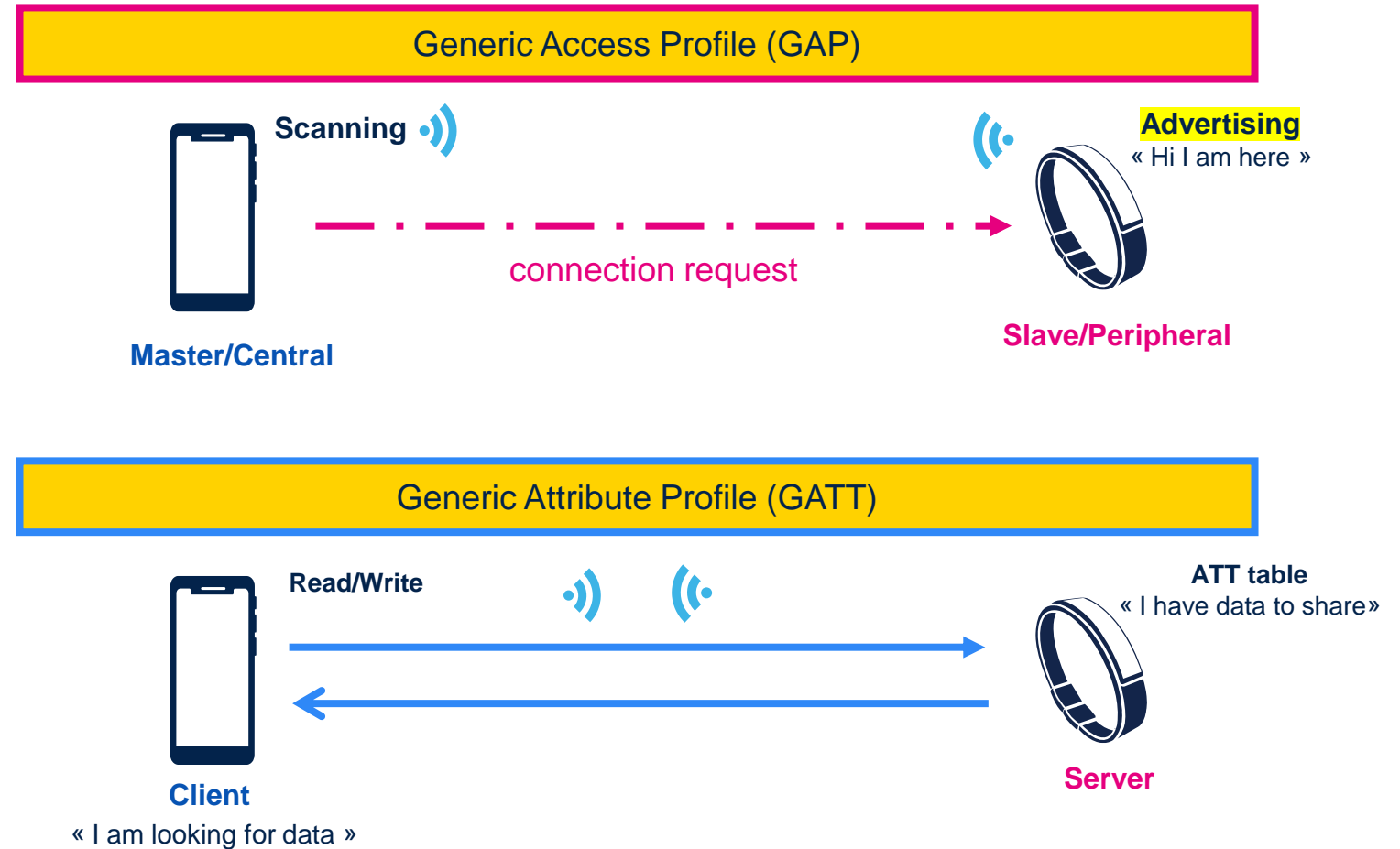
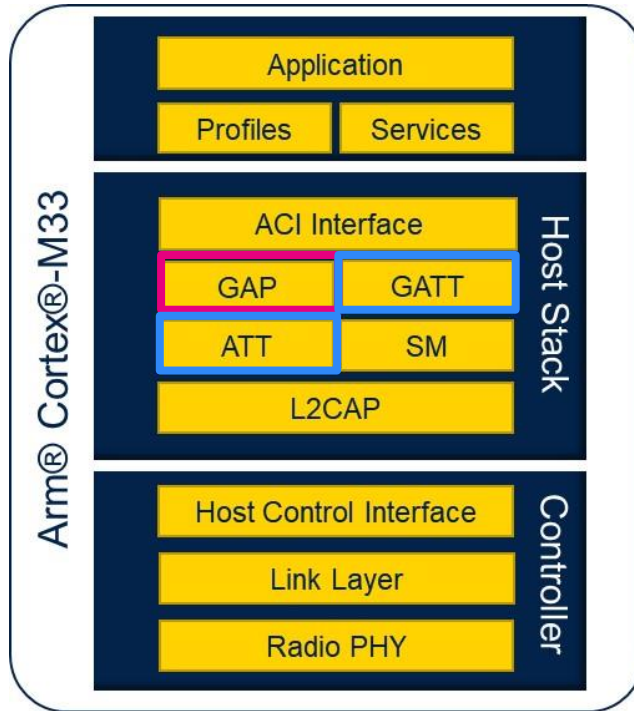
Custom CHARACTERISTIC 2

N

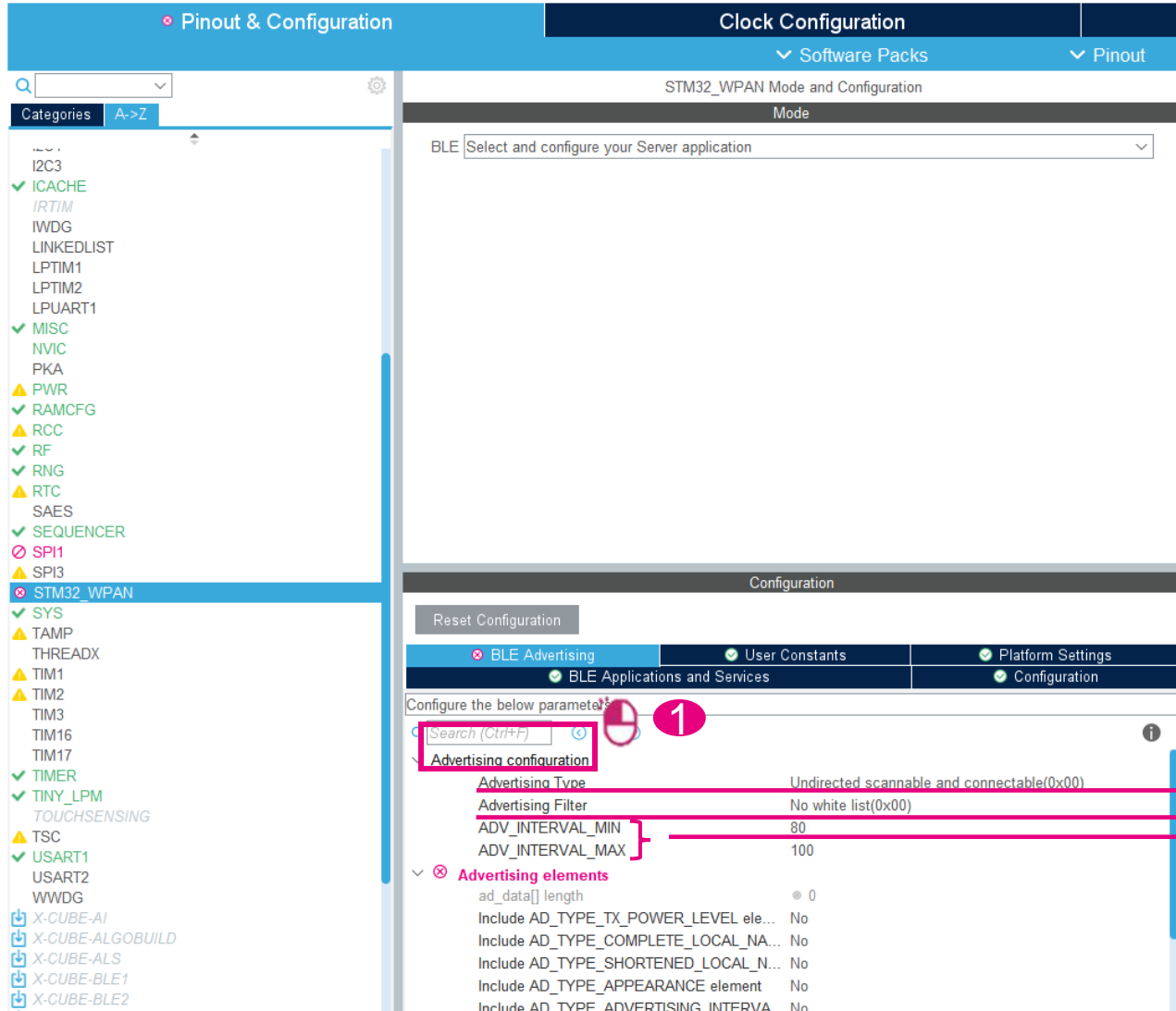
LED 1

Button 1

Bluetooth Low Energy Connection roles vs. Data roles



In the general run of things....
a Central is acting as GATT Client, a peripheral as a GATT server



The screenshot shows the STM32CubeMX Pinout & Configuration window. The left sidebar lists various components, with STM32_WPAN selected. The main area shows the configuration for STM32_WPAN Mode and Configuration. The BLE Advertising configuration is expanded, showing the following parameters:

Parameter	Value
Advertising Type	Undirected scannable and connectable(0x00)
Advertising Filter	No white list(0x00)
ADV_INTERVAL_MIN	80
ADV_INTERVAL_MAX	100
Advertising elements	
ad_data[] length	0
Include AD_TYPE_TX_POWER_LEVEL ele...	No
Include AD_TYPE_COMPLETE_LOCAL_NA...	No
Include AD_TYPE_SHORTENED_LOCAL_N...	No
Include AD_TYPE_APPEARANCE element	No
Include AD_TYPE_ADVERTISING INTERVA...	No

Annotations in the image include a red box around the 'Advertising configuration' section, a red circle with the number '1' next to the search icon, and three red arrows pointing from the configuration parameters to explanatory text on the right.

Advertising Type

accept connection requests from any peer device

Advertising Filter

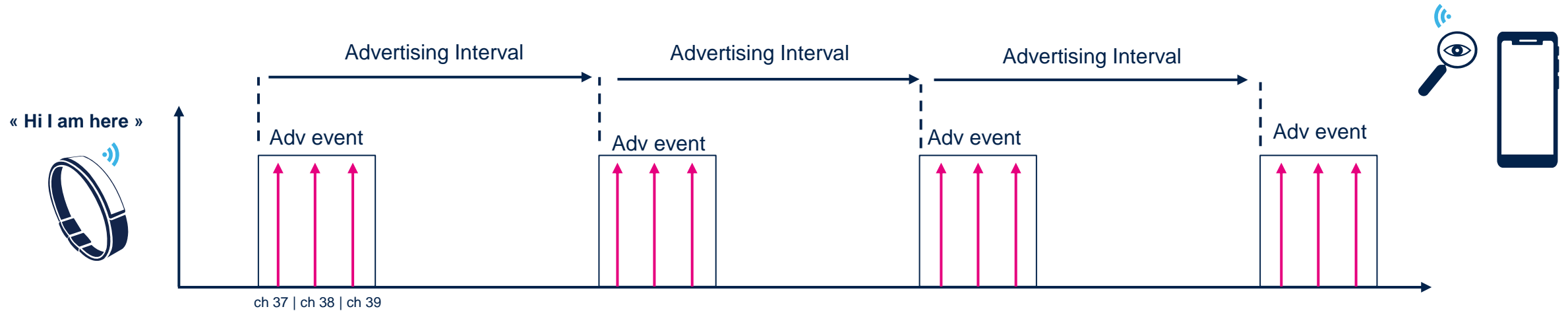
In general used in case of Privacy.

Advertising Interval

Advertising set = (MIN+MAX)/2
Min & Max used in case of multi connections

Advertising Configuration

Advertising Interval



- The advertising interval value ranges all the way from **20** milliseconds up to **10.24** seconds in small increments of **625** microseconds.
- The advertising interval greatly impacts battery life and should be chosen carefully.

connectivity latency vs. power consumption efficiency

- The advertising event is the slot where peripheral will be able to push for advertising data "Hello I am here – this is my name"
- The advertising event is around **~3ms** considering legacy advertising (31 bytes)



Advertising Elements Local Name

2

Pinout & Configuration

Clock Configuration

Project Manager

Software Packs

Pinout

STM32_WPAN Mode and Configuration

Mode

BLE Select and configure your Server application

Configuration

Reset Configuration

BLE Applications and Services

Configuration

BLE Advertising

SERVICE1

User Constants

Platform Settings

Configure the below parameters :

Search (Ctrl+F)

Include AD_TYPE_TX_POWER_LEVEL element	No
Include AD_TYPE_COMPLETE_LOCAL_NAME element	Yes
AD_TYPE_COMPLETE_LOCAL_NAME_LENGTH	0
AD_TYPE_COMPLETE_LOCAL_NAME	p2pS_01
Include AD_TYPE_SHORTENED_LOCAL_NAME element	No
Include AD_TYPE_APPEARANCE element	No
Include AD_TYPE_ADVERTISING_INTERVAL element	No
Include AD_TYPE_LE_ROLE element	No
Include AD_TYPE_16_BIT_SERV_UUID_CMPLT_LIST element	No
Include AD_TYPE_128_BIT_SERV_UUID_CMPLT_LIST element	No
Include AD_TYPE_SLAVE_CONN_INTERVAL element	No
Include AD_TYPE_URI element	No
Include AD_TYPE_MANUFACTURER_SPECIFIC_DATA element	Yes
AD_TYPE_MANUFACTURER_SPECIFIC_DATA_LENGTH	13
Company identifier	30,00
Number of user defined data item(s)	12

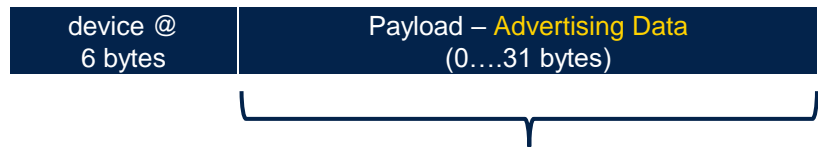
As a server, our application will have to advertise waiting for connection request from a client.

Define here your "custom" local name part of advertising frame.

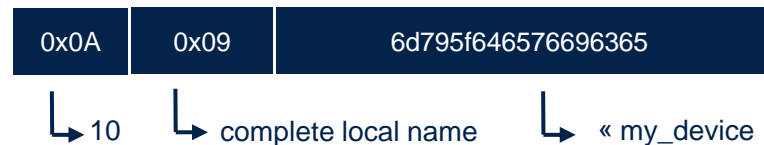
3

Advertising Elements

Advertising PDU



The Advertising Data consists of one or more Advertising Data elements
AD Element/Type are listed [at Bluetooth SIG website](#)



Most commonly used AD elements :

- 0x01 = Flags (**mandatory for connectable device**)
- 0x09 = Complete Local Name
- 0xFF = Manufacturer Data

added by
aci_gap_set_discoverable
command

You can push for what you want over the air ! All data need to be prefix using dedicated Ad Type

* up to 37 bytes in legacy advertising , up to 1650 in case of advertising extension (packet of 250 chained)

Advertising Elements Manufacturer Data

Pinout & Configuration

Search

Categories A-Z

- GPIO
- GTZC
- HASH
- HSEM
- I-CUBE-Cesium
- I-CUBE-embOS
- I-CUBE-wolfSSL
- I-Cube-SoM-uGOAL
- I2C1
- I2C3
- ✓ ICACHE
- IRTIM
- IWDG
- LINKEDLIST
- LPTIM1
- LPTIM2
- LPUART1
- ✓ MISC
- NVIC
- PKA
- ▲ PWR
- ✓ RAMCFG
- ▲ RCC
- ✓ RF
- ✓ RNG
- ▲ RTC
- SAES
- ✓ SEQUENCER
- ❌ SPI1
- ▲ SPI3
- ✓ STM32_WPAN
- ✓ SYS
- ▲ TAMP
- THREADX
- ▲ TIM1
- ▲ TIM2
- TIM3
- TIM16
- TIM17

Clock Configuration

Software Packs

Pinout

STM32_WPAN Mode and Configuration

Mode

BLE Select and configure your Server application

Configuration

Reset Configuration

BLE Applications and Services

Configuration

BLE Advertising

User Constants

Platform Settings

Configure the below parameters :

Search (Ctrl+F)

> Advertising configuration

Advertising elements

ad_data[] length	25
Include AD_TYPE_TX_POWER_LEVEL element	No
Include AD_TYPE_COMPLETE_LOCAL_NAME element	Yes
AD_TYPE_COMPLETE_LOCAL_NAME_LENGTH	8
AD_TYPE_COMPLETE_LOCAL_NAME	P2PS_01
Include AD_TYPE_SHORTENED_LOCAL_NAME element	No
Include AD_TYPE_APPEARANCE element	No
Include AD_TYPE_ADVERTISING_INTERVAL element	No
Include AD_TYPE_LE_ROLE element	No
Include AD_TYPE_16_BIT_SERV_UUID_CMPLT_LIST e...	No
Include AD_TYPE_128_BIT_SERV_UUID_CMPLT_LIST e...	No
Include AD_TYPE_SLAVE_CONN_INTERVAL element	No
Include AD_TYPE_URI element	No
Include AD_TYPE_MANUFACTURER_SPECIFIC_DATA ...	Yes
AD_TYPE_MANUFACTURER_SPECIFIC_DATA_LE...	15
Company identifier	30,00
Number of user defined data item(s)	12

Manufacturer Ad Type , with company ID 0x30 (STMicroelectronics)



Allow to detect device as
ST device and to connect
as P2P profile



Customize Device Name



Pinout & Configuration | Clock Configuration | Project Manager

▼ Software Packs | ▼ Pinout

STM32_WPAN Mode and Configuration

Mode

BLE Select and configure your Server application

Configuration

Reset Configuration

BLE Applications and Services | Configuration | BLE Advertising | SERVICE1 | User Constants | Platform Settings

Configure the below parameters :

Search (Ctrl+F)

> Application configuration - Project IP's Configuration

> Application configuration - Application parameters

CFG_TX_POWER	-0.3 dBm (0x19)
CFG_BD_ADDRESS	0x0008E12A1234
Address Type	Public address(0)
PAIRING_PARAMETERS	OFF
CFG_IO_CAPABILITY	Display Yes No (0x01)
CFG_MITM_PROTECTION	MITM protection required (0x01)
CFG_BLE_IRK	12, 34, 56, 78, 9A, BC, DE, F0, 12, 34, 56, 78, 9A, BC, DE, F0
CFG_BLE_ERK	FE, DC, BA, 09, 87, 65, 43, 21, FE, DC, BA, 09, 87, 65, 43, 21
CFG_GAP_DEVICE_NAME	p2pS_01
CFG_GAP_DEVICE_NAME_LENGTH	7

> Application configuration - BLE stack

> Application configuration - Low Power

> Application configuration - Traces

> Application configuration - Log level

> Application configuration - NVM

> Application configuration - RT GPIO debug

> Application configuration - HW Radio

> Application configuration - HW_RNG

> Application configuration - Memory manager

2

set same Device name = Local Name

Step 3 : Code generation and user application code

Home
STM32WBA52CGux - NUCLEO-WBA52CG
Hands-On_WS_WBA52.ioc - Project Manager
1
GENERATE CODE

Pinout & Configuration
Clock Configuration
Project Manager
Tools

2
Project
Code Generator
Advanced Settings

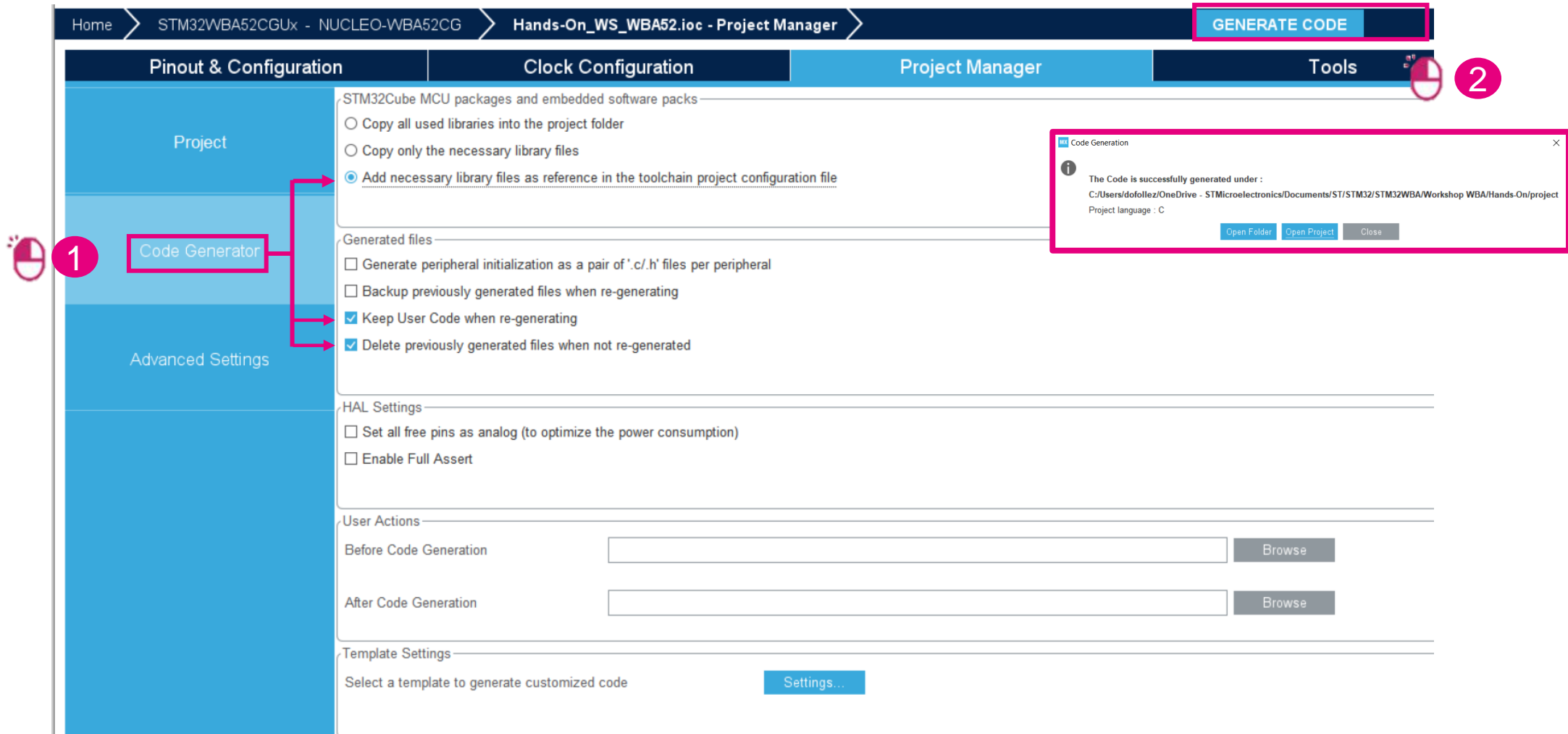
Project Settings
Project Name
Hands-On_WS_WBA52
Project Location
OneDrive - STMicroelectronics\Documents\ST\STM32\STM32WBA\Workshop WBA\Hands-On\project\
Browse
Application Structure
Advanced
Do not generate the main()
Toolchain Folder Location
ve - STMicroelectronics\Documents\ST\STM32\STM32WBA\Workshop WBA\Hands-On\project\Hands-On_WS_WBA52\
Toolchain / IDE
STM32CubeIDE
Generate Under Root

Linker Settings
Minimum Heap Size
0x200
Linker settings – Heap and CStack
Minimum Stack Size
0x400

Thread-safe Settings
CortexM33
Enable multi-threaded support
Thread-safe Locking Strategy
Default – Mapping suitable strategy depending on RTOS selection.

Mcu and Firmware Package
Mcu Reference
Be sure you're using STM32CubeWBA v1.1.0!!
Firmware Package Name and Version
STM32Cube FW_WBA V1.1.0
Use Default Firmware Location
Use default fw location
Firmware Relative Path
C:/Users/dofollez/STM32Cube/Repository/STM32Cube_FW_WBA_V1.1.0
Browse

Project configuration



The screenshot displays the STM32CubeMX Project Manager interface for the project "Hands-On_WS_WBA52.ioc". The interface is divided into several sections:

- Top Bar:** Includes "Home", "STM32WBA52CGux - NUCLEO-WBA52CG", "Hands-On_WS_WBA52.ioc - Project Manager", and a "GENERATE CODE" button (highlighted with a pink box).
- Left Sidebar:** Contains "Pinout & Configuration", "Clock Configuration", "Project Manager" (selected), and "Tools". A "Code Generator" button is highlighted with a pink box and labeled with a red circle "1".
- Main Content Area:**
 - Project:** Options for copying libraries:
 - ☐ Copy all used libraries into the project folder
 - ☐ Copy only the necessary library files
 - ☒ Add necessary library files as reference in the toolchain project configuration file
 - Generated files:**
 - ☐ Generate peripheral initialization as a pair of '.c/.h' files per peripheral
 - ☐ Backup previously generated files when re-generating
 - ☒ Keep User Code when re-generating
 - ☒ Delete previously generated files when not re-generated
 - HAL Settings:**
 - ☐ Set all free pins as analog (to optimize the power consumption)
 - ☐ Enable Full Assert
 - User Actions:**
 - Before Code Generation:
 - After Code Generation:
 - Template Settings:**
 - Select a template to generate customized code:
- Right Panel:** A "Code Generation" dialog box (highlighted with a pink box and labeled with a red circle "2") displays the message: "The Code is successfully generated under : C:/Users/dofollez/OneDrive - STMicroelectronics/Documents/ST/STM32/STM32WBA/Workshop WBA/Hands-On/project Project language : C". It includes buttons for "Open Folder", "Open Project", and "Close".

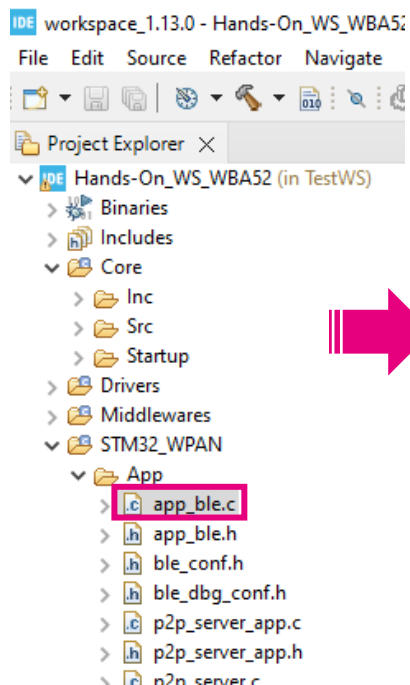


Here are our ADV data

```
178 /* Identity root key used to derive LTK and CSRK */
179 static const uint8_t a_BLE_CfgIrValue[16] = CFG_BLE_IRK;
180
181 /* Encryption root key used to derive LTK and CSRK */
182 static const uint8_t a_BLE_CfgErValue[16] = CFG_BLE_ERK;
183 static BleApplicationContext_t bleAppContext;
184
185 P2P_SERVER_APP_ConnHandleNotEvt_t P2P_SERVERHandleNotification;
186
187 static const char a_GapDeviceName[] = { 'P', 'e', 'e', 'r', ' ', 't', 'o', ' ', 'P', 'e', 'e', 'r', ' ', 'S', 'e', 'r', 'v', 'e', 'r' }; /* C
188
189 /* Advertising Data */
190 uint8_t a_AdvData[23] =
191 {
192     6, AD_TYPE_COMPLETE_LOCAL_NAME, 'c', 'i', 'r', 'o', '1', /* Complete name */
193     15, AD_TYPE_MANUFACTURER_SPECIFIC_DATA, 0x30, 0x00, 0x00 /* */, 0x00 /* */, 0x00 /* */, 0x00 /* */, 0x00 /* */, 0x00 /* */, 0x00 /* */, 0x00 /* */, 0x00 /* */, 0x00 /* */, 0x00 /* */, 0x00 /* */
194 }
195
196 uint64_t buffer_nvm[CFG_BLEPLAT_NVM_MAX_SIZE] = {0};
197
198 static AMM_VirtualMemoryCallbackFunction_t APP_BLE_ResumeFlowProcessCb;
199
200 /* Host stack init variables */
201 static uint32_t buffer[DIVC(BLE_DYN_ALLOC_SIZE, 4)];
202 static uint32_t gatt_buffer[DIVC(BLE_GATT_BUF_SIZE, 4)];
203 static BleStack_init_t pInitParams;
204
205 /* USER CODE BEGIN PV */
206
207 /* USER CODE END PV */
208
```

Open Project

Add application code to move to discoverable



#1

: Set device discoverable at init :

In app_ble.c > function APP_BLE_Init()

$(ADV_MIN + ADV_MAX) / 2$

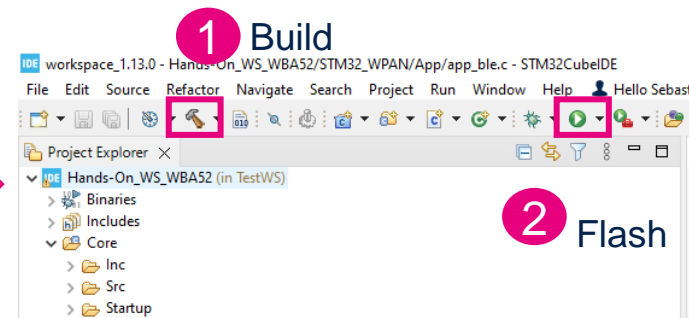
```
/* USER CODE BEGIN APP_BLE_Init_2 */
APP_BLE_Procedure_Gap_Peripheral(PROC_GAP_PERIPH_ADVERTISE_START_FAST);
/* USER CODE END APP_BLE_Init_2 */
```

#2 : Set device discoverable at disconnection :

In app_ble.c > SVCCTL_App_Notification -
HCI_DISCONNECTION_COMPLETE_EVT_CODE

```
/* USER CODE BEGIN EVT_DISCONN_COMPLETE */
APP_BLE_Procedure_Gap_Peripheral(PROC_GAP_PERIPH_ADVERTISE_START_FAST);
/* USER CODE BEGIN EVT_DISCONN_COMPLETE */
```

At disconnection, stack is not moving back to advertising, this is an application decision



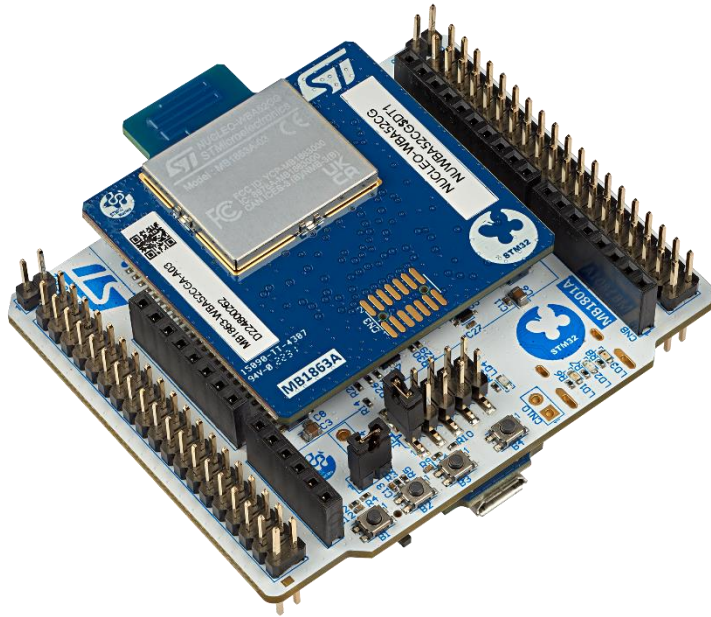
Open Project

Add application code to move to discoverable

Build& Flash



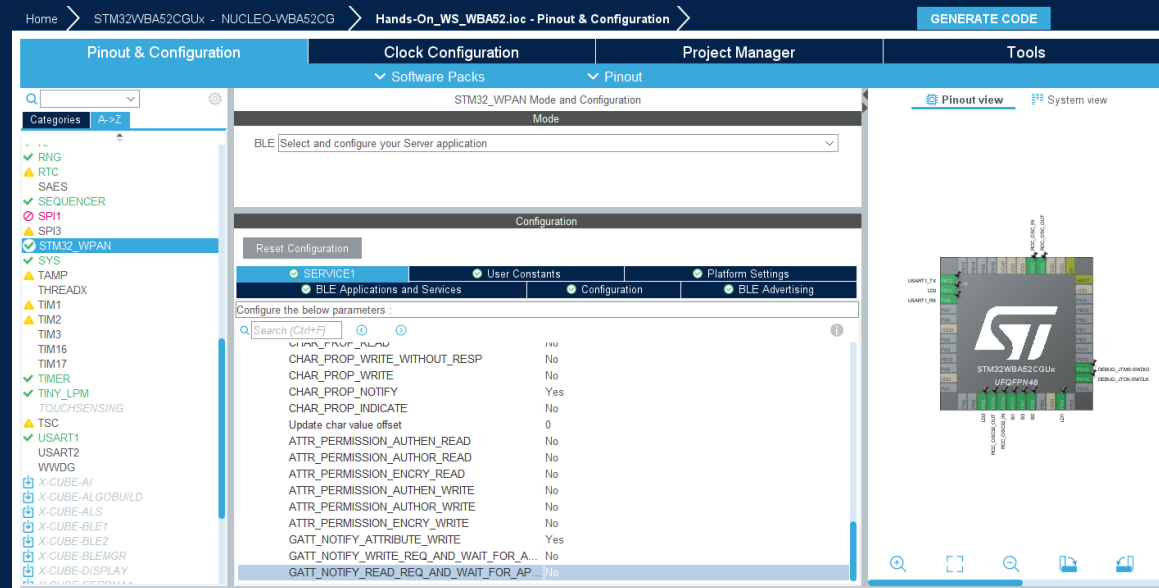
Open your App and Connect



Device should be visible and connectable

Add debug capabilities

Move back to CubeMx



Application configuration

Traces & logs

Pinout & Configuration

Search

Categories A->Z

I-Cube-SoM-uGOAL

I2C1

I2C3

✓ ICACHE

IRTIM

IWDG

LINKEDLIST

LPTIM1

LPTIM2

LPUART1

✓ MISC

NVIC

PKA

▲ PWR

✓ RAMCFG

▲ RCC

✓ RF

✓ RNG

▲ RTC

SAES

✓ SEQUENCER

⊗ SPI1

▲ SPI3

✓ STM32_WPAN

✓ SYS

▲ TAMP

THREADX

▲ TIM1

▲ TIM2

TIM3

TIM16

TIM17

Clock Configuration

Software Packs

Pinout

STM32_WPAN Mode and Configuration

Mode

BLE Select and configure your Server application

Configuration

Reset Configuration

✓ BLE Applications and Services

✓ Configuration

✓ BLE Advertising

✓ SERVICE1

✓ User Constants

✓ Platform Settings

Configure the below parameters :

Search (Ctrl+F)

> Application configuration - Project IP's Configuration

> Application configuration - Application parameters

> Application configuration - BLE stack

> Application configuration - Low Power

▼ Application configuration - Traces

ADV_TRACE_TIMESTAMP_ENABLE

CFG_DEBUG_APP_TRACE

CFG_DEBUG_TRACE_LIGHT

CFG_DEBUG_TRACE_FULL

DBG_TRACE_USE_CIRCULAR_QUEUE

DBG_TRACE_MSG_QUEUE_SIZE

MAX_DBG_TRACE_MSG_SIZE

> Application configuration - Log level

> Application configuration - NVM

> Application configuration - RT GPIO debug

> Application configuration - HW Radio

> Application configuration - HW RNG

> Application configuration - Memory manager

1

Disabled

Enabled

Disabled

Enabled

Enabled

4096

1024

Let's enable FULL trace at application level


Application configuration

Trace & Logs: configure log level

☒ BLE Applications and Services
 ☒ Configuration
 ☒ BLE Advertising
 ☒ SERVICE1
 ☒ User Constants
 ☐ Platform Settings

Configure the below parameters :

- > Application configuration - Project IP's Configuration
- > Application configuration - Application parameters
- > Application configuration - BLE stack
- > Application configuration - Low Power
- > Application configuration - Traces
- ☒ Application configuration - Log level



LOG_LEVEL_INFO

0
- > Application configuration - NVM
- > Application configuration - RT GPIO debug
- > Application configuration - HW Radio
- > Application configuration - HW_RNG
- > Application configuration - Memory manager

APPLI_CONFIG_LOG_LEVEL
 APPLI_PRINT_FILE_FUNC_LINE



Platform Settings

Trace & Logs: BSP settings

STM32_WPAN Mode and Configuration

Mode

BLE

Configuration

☒ BLE Applications and Services ☒ Configuration ☒ BLE Advertising ☒ SERVICE1 ☒ User Constants ☒ Platform Settings

Platform proposal

BSP

Name	IPs or Components	Found Solutions	BSP API
Serial Link for Traces	<input type="text" value="USART:Asynchronous"/>	<input type="text" value="USART1"/>	Unknown





Project configuration

Advanced settings

Home > STM32WBA52CGUx - NUCLEO-WBA52CG > Hands-On_WS_WBA52.ioc - Project Manager > **GENERATE CODE**

Project

Code Generator

Advanced Settings

Driver Selector

Search (Ctrl+F)

RCC
GPIO
> GPDMA
PWR
RAMCFG
RTC
> USART
> ADC
CRC
RNG

HAL
HAL
HAL
HAL
HAL
HAL
HAL
HAL
HAL
HAL
HAL

Generated Function Calls

Generate Code	Rank	Function Name	Peripheral Instance Name	<input type="checkbox"/> Do Not Generate Function Call	<input type="checkbox"/> Visibility (Static)
<input checked="" type="checkbox"/>	1	SystemClock_Config	RCC	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	2	MX_GPIO_Init	GPIO	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	3	MX_GPDMA1_Init	GPDMA1	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	4	SystemPower_Config	PWR	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	5	MX_RAMCFG_Init	RAMCFG	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	6	MX_RTC_Init	RTC	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	7	MX_USART1_UART_Init	USART1	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	8	MX_ADC4_Init	ADC4	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	9	MX_CRC_Init	CRC	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	10	MX_RNG_Init	RNG	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	11	MX_ICACHE_Init	ICACHE	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	12	APPE_Init	STM32_WPAN	<input type="checkbox"/>	<input type="checkbox"/>

Register CallBack

Search (Ctrl+F)

ADC
COMP
CRYP
HASH
I2C
IWDG
IRDA
LPTIM
PKA
RAMCFG
RNG
RTC
SAI
SMARTCARD
SMBUS
SPI
TIM
TSC
UART
USART
WWDG

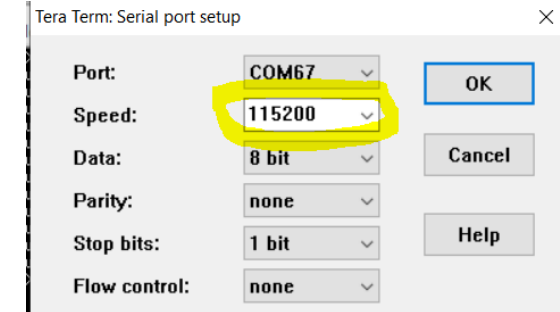
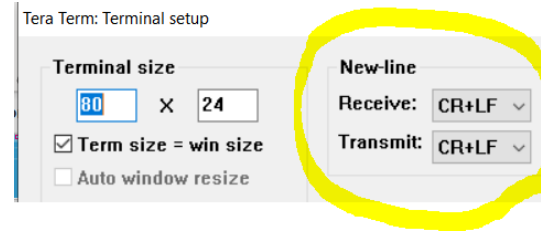
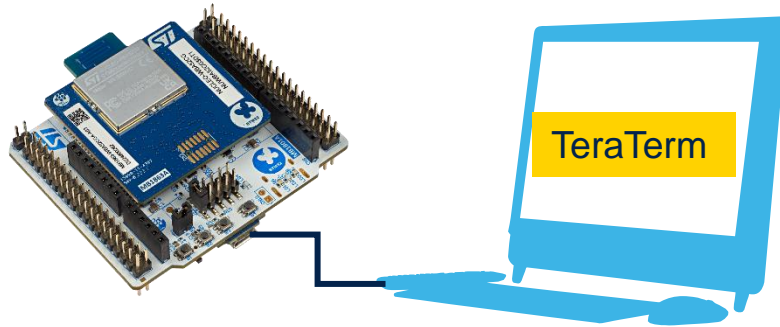
DISABLE
DISABLE
DISABLE
DISABLE
DISABLE
DISABLE
DISABLE
DISABLE
DISABLE
DISABLE
DISABLE
DISABLE
DISABLE
DISABLE
DISABLE
DISABLE
ENABLE
DISABLE
DISABLE

Regenerate Code

Open back existing project – refresh sources – build and flash



Open your App and Connect



1

reset device



```
COM67 - Tera Term VT
File Edit Setup Control Window Help
Success: aci_hal_write_config_data command - CONFIG_DATA_PUBADDR_OFFSET
Public Bluetooth Address: 00:80:e1:2a:19:82
Success: aci_hal_write_config_data command - CONFIG_DATA_IR_OFFSET
Success: aci_hal_write_config_data command - CONFIG_DATA_ER_OFFSET
Success: aci_hal_set_tx_power_level command
Success: aci_gatt_init command
Success: aci_gap_init command
Success: aci_gatt_update_char_value - Device Name
Success: aci_gatt_update_char_value - Appearance
Success: hci_le_set_default_phy command
Success: aci_gap_set_io_capability command
Success: aci_gap_set_authentication_requirement command
==> End Ble_Hci_Gap_Gatt_Init function

Services and Characteristics creation
Success: aci_gatt_add_service command: P2P_Server
Success: aci_gatt_add_char command : LED_C
Success: aci_gatt_add_char command : SWITHC_C
End of Services and Characteristics creation
==> aci_gap_set_discoverable - Success
==> Success: Start Advertising
```

2

Connect

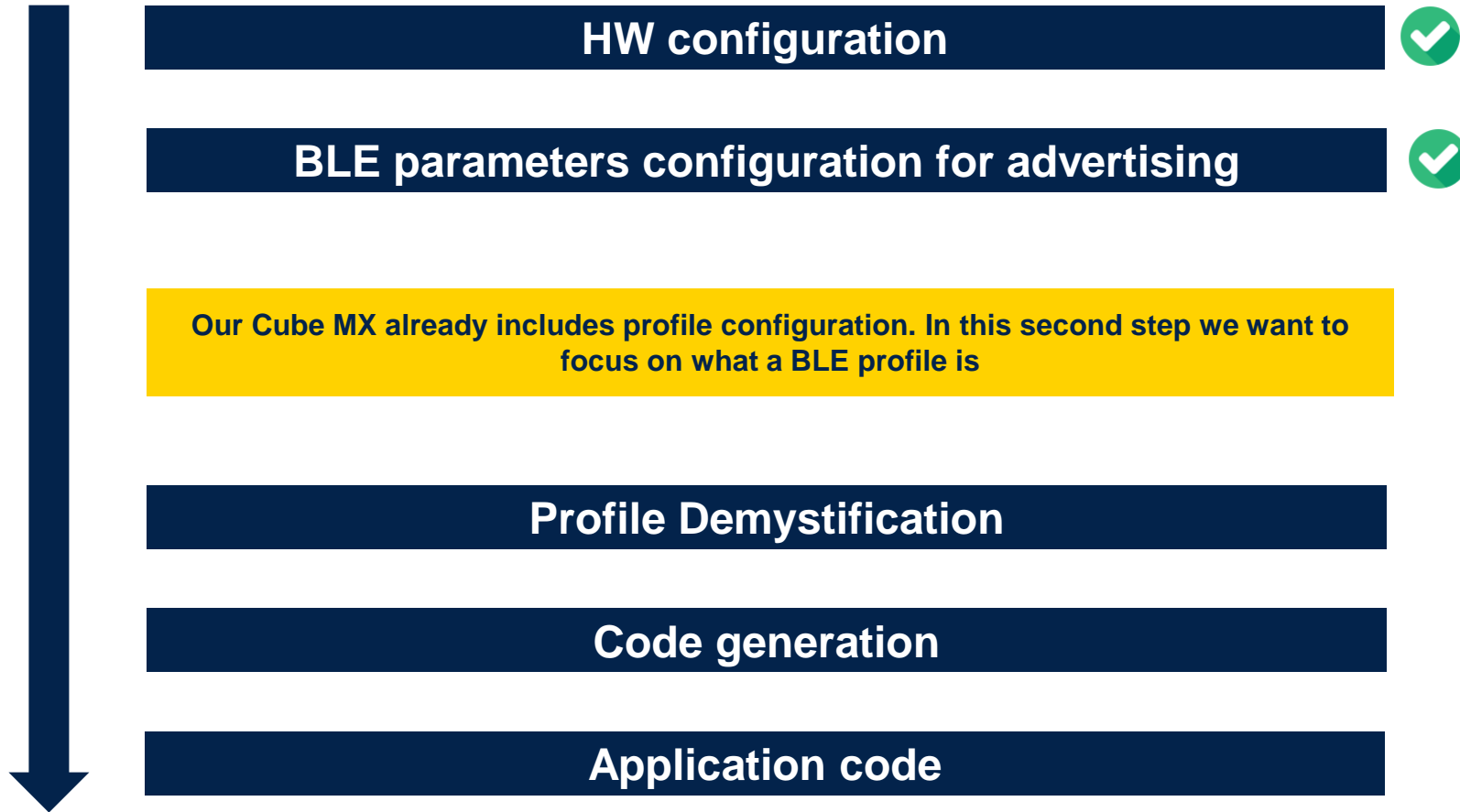


```
COM67 - Tera Term VT
File Edit Setup Control Window Help
>>= HCI_LE_CONNECTION_COMPLETE SUBEVT_CODE - Connection handle: 0x0001
- Connection established with 0:77:1c:a8:d6:d9:5a
- Connection Interval: ms
- Connection latency: 0
- Supervision Timeout: 720 ms
```

Hands On #2 : BLE Profiles

Configuration completed

What's next : code generation ?

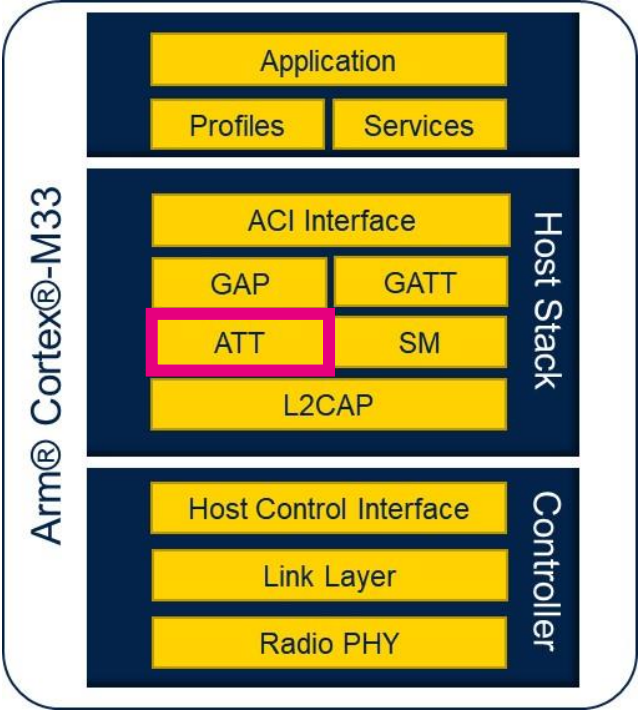


Profile Theory



What is a Bluetooth Low Energy Profile

Attribute Protocol (ATT)



Define Client - Server architecture



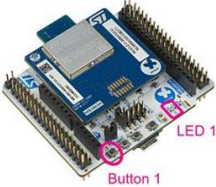
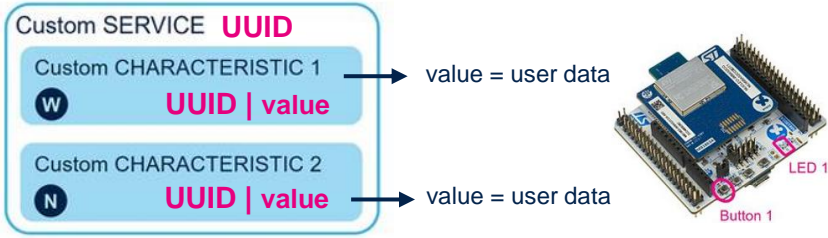
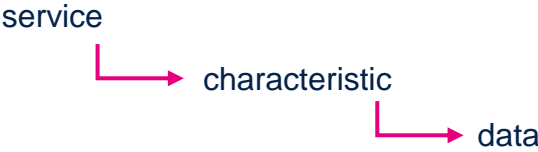
« I am looking for data »
Client



« I have data to share » (sensors, raw data...**what you want !**)
Server

Logical data structure – « How to access data » : Attribute table

user data is accessible trough attribute



What is a Bluetooth Low Energy Profile

A profile is a collection of data (**attributes**) exposes by device trough associated Service and Characteristic

Profile

Service **UUID**

Characteristic

R **UUID**

Service **UUID**

Characteristic 1

W **UUID**

Characteristic 2

R **N** **UUID**

- All attributes have a type which is identified by a UUID (**U**niversally **U**nique **I**dentifier)
- Characteristic can take 3 types of properties: **READ**, **WRITE**, **NOTIFY**
- Profile can be defined by **Bluetooth® SIG**
 - ↳ **UUID : 16 bits**
Service Heart Rate **0x180D**
Characteristic Heart Rate Measurement **0x2A37**
- Profile can be a **custom** (**proprietary**) profile
 - ↳ **UUID : 128 bits**
Service P2P **0000FE40-cc7a-482a-984a-7f2ed5b3e58f**
Characteristic LED **0000FE41-cc7a-482a-984a-7f2ed5b3e58f**

BLE standard profile vs. proprietary profile

Standard Heart Rate Profile

Heart Rate Service 0x180D

**Heart Rate Measurement
Characteristic** 0x2A37

**Body sensor Location
Characteristic** 0x2A38

Proprietary ST P2P Server Profile

P2P Service 0x0000FE40CC7A482A984A7F2ED5B3E58F

My_LED_Char 0x0000FE41CC7A482A984A7F2ED5B3E58F

SWITCH_C 0x0000FE42CC7A482A984A7F2ED5B3E58F

Define by the **SIG**, define the role, requirements, behavior and the structure of Attribute Table of each entity (central & peripheral)

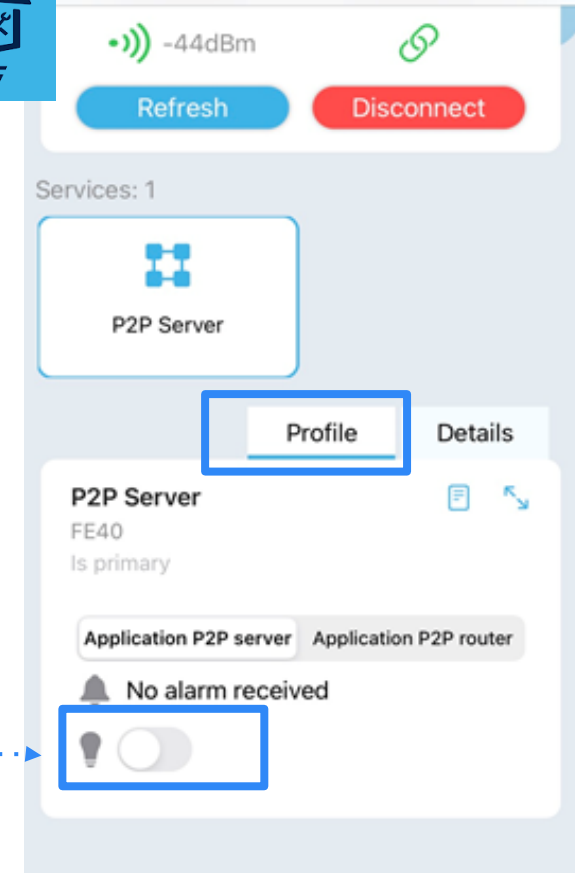
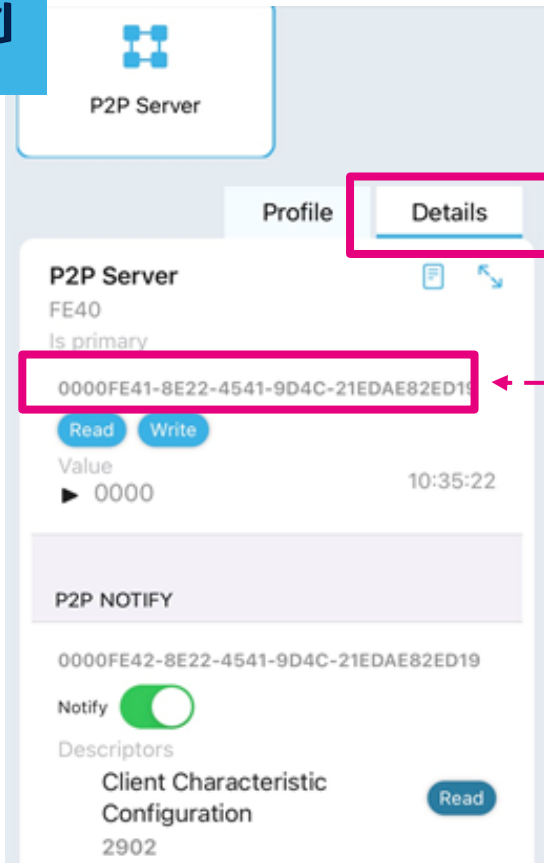
→ Any standard smartphone App will be able to communicate

Define your own behavior using your own Attribute Table based in 128 bits UUID

→  Only your own App will be able to communicate



Proprietary profile ST Toolbox App

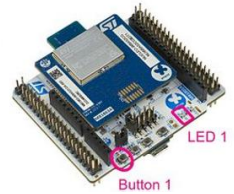


ST ToolBox App knows that **My_LED_Char proprietary UUID** is defined to toggle led .
As consequence App displays nice **toggle button**



Data exchanges

what is the magic behind



#1

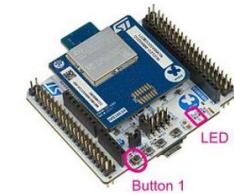
At profile initialization and entry point (**handler @**) will be created in RAM to expose data to client



WBA52 - Attribute Table (RAM)		
@ Serv service UUID	@ Char W R N char UUID	@ Char +1 user data

#2

As soon as connected client will discover server attribute table (**handler**) and will be able to access (**write/read**) data



#3

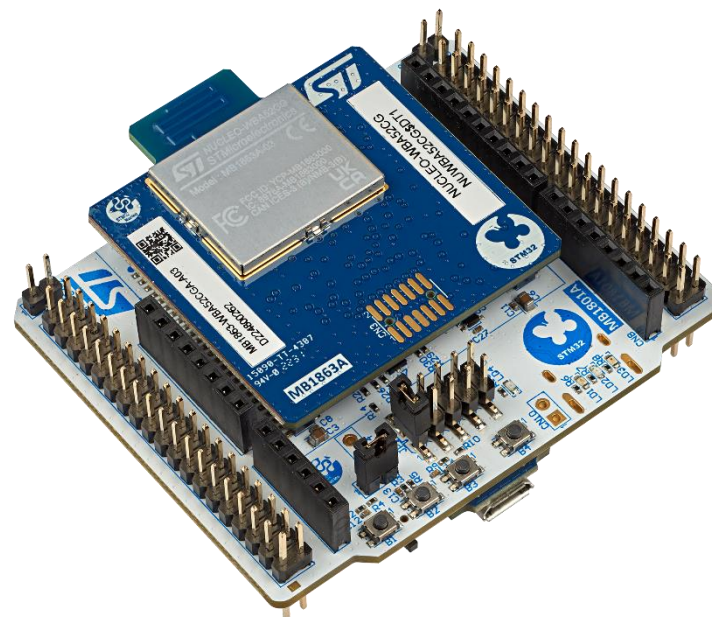
Application will update data (ie : push button), client will receive **notification** with data updates



@ Serv	@ Char W R N	@ Char +1 update data

BLE write, read, notify procedures using the right attribute handler make data exchange possible

Coming back to our example



ST BLE Toolbox



GATT Client

GAP central

GATT Server

GAP peripheral

How a profile is done in CubeMX?

Pinout & Configuration

Clock Configuration

Software Packs

Pinout

STM32_WPAN Mode and Configuration

Mode

BLE Select and configure your Server application

Configuration

Reset Configuration

BLE Advertising

SERVICE1

User Constants

Platform Settings

BLE Applications and Services

Configuration

Configure the below parameters :

Search (Ctrl+F)

1

Server Mode

Number of services

1

BLE Services Configuration

Peripheral Role

1

Central Role

0

BLE_CFG_SVC_MAX_NBR_CB

7

BLE_CFG_CLT_MAX_NBR_CB

0

Categories

A->Z

GPIO

GTZC

HASH

HSEM

I-CUBE-Cesium

I-CUBE-embOS

I-CUBE-wolfSSL

I-Cube-SoM-uGOAL

I2C1

I2C3

ICACHE

IRTIM

IWDG

LINKEDLIST

LPTIM1

LPTIM2

LPUART1

MISC

NVIC

PKA

PWR

RAMCFG

RCC

RF

RNG

RTC

SAES

SEQUENCER

SPI1

SPI3

STM32_WPAN

SVC

Profile Creation

Configure my P2P Service

Configuration

Reset Configuration

BLE Applications and Services Configuration BLE Advertising **SERVICE1** User Constants Platform Settings

Configure the below parameters :

Search (Ctrl+F)

Service

- UUID type
- UUID 128 input type
- UUID
- Service long name
- Service short name
- Type
- * Service max attributes record(s)
- Number of characteristics
- > Characteristic1
- > Characteristic2

128 bits UUID(0x02)
reduced
FE 40
P2P_Server
P2P_Server
Primary Service(0x01)
5
2

2

Service Long Name	My_P2P_Server	
Service Short Name	My_P2P	
UUID Type	128 bits	
UUID	0xFE40	
Characteristic Long Name	My_LED_Char	My_Switch_Char
Characteristic Short Name	LED_C	SWITCH_C
UUID Type	128 bits	128 bits
UUID	0xFE41	0xFE42
Char Properties	Read + Write w/o response	Notify
Char Permissions	None	None
Char GATT Events	GATT_NOTIFY_ATTRIBUTE_WRITE	GATT_NOTIFY_ATTRIBUTE_WRITE

service & characteristic naming used to name function at code generation

UUID : FE 40

The application code will append 112 bits (based on UUID generator) to have a complete **128 bits UUID**



Profile Creation

Configure 1st Characteristic

Configuration

Reset Configuration

BLE Applications and Services Configuration BLE Advertising SERVICE1 User Constants Platform Settings

Configure the below parameters :

Search (Ctrl+F)

> Service

Characteristic1

UUID type
128 bits UUID(0x02)
reduced
FE 41
My_LED_Char
LED_C
2
Variable
0x10

CHAR_PROP_BROADCAST No

CHAR_PROP_READ Yes

CHAR_PROP_WRITE_WITHOUT_RESP Yes

CHAR_PROP_WRITE No

CHAR_PROP_NOTIFY No

CHAR_PROP_INDICATE No

ATTR_PERMISSION_AUTHEN_READ No

ATTR_PERMISSION_AUTHOR_READ No

ATTR_PERMISSION_ENCRY_READ No

ATTR_PERMISSION_AUTHEN_WRITE No

ATTR_PERMISSION_AUTHOR_WRITE No

ATTR_PERMISSION_ENCRY_WRITE No

GATT_NOTIFY_ATTRIBUTE_WRITE Yes

GATT_NOTIFY_WRITE_REQ_AND_WAIT_FOR_APPL_RESP No

GATT_NOTIFY_READ_REQ_AND_WAIT_FOR_APPL_RESP No

> Characteristic2

UUID : FE 41

Application code will complete to have a complete **128 bits UUID**

Properties

Data (**2 bytes**) can be read and write.

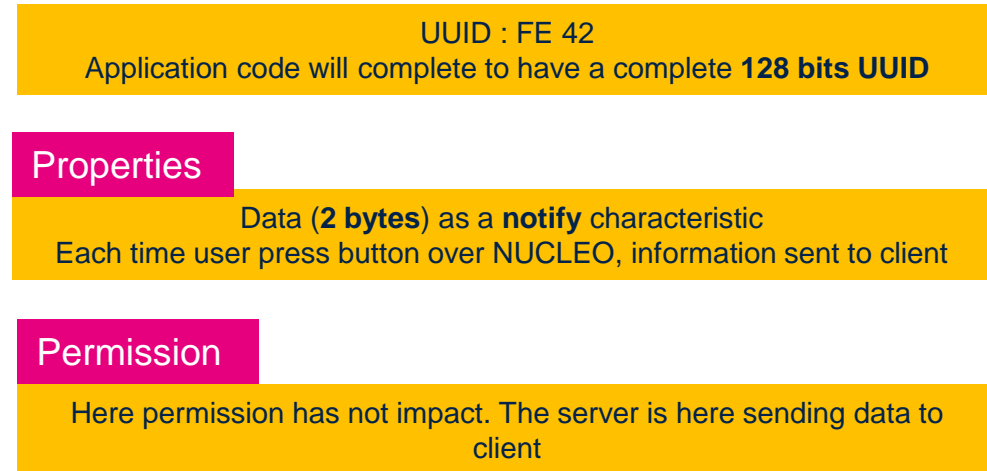
The purpose of characteristic 1 is to write data in order to control LED

Permission

Thanks to **notify write**, application is informed that attribute has been modified and can accordingly process expected use case

	Characteristic 1	Characteristic 2
UUID type	128 bits UUID (0x02)	128 bits UUID (0x02)
UUID 128 Input type	Reduced	Reduced
UUID	FE 41	FE 42
Characteristic long name	My_LED_Char	My_Switch_Char
Characteristic Short Name	LED_C	SWITCH_C
Value length	2	2
Length characteristic	Variable	Variable
Encryption key size	0x10	0x10
Char Properties	READ WRITE_WITHOUT_RESP	NOTIFY
GATT events	GATT_NOTIFY_ATTRIBUTE_WRITE	GATT_NOTIFY_ATTRIBUTE_WRITE

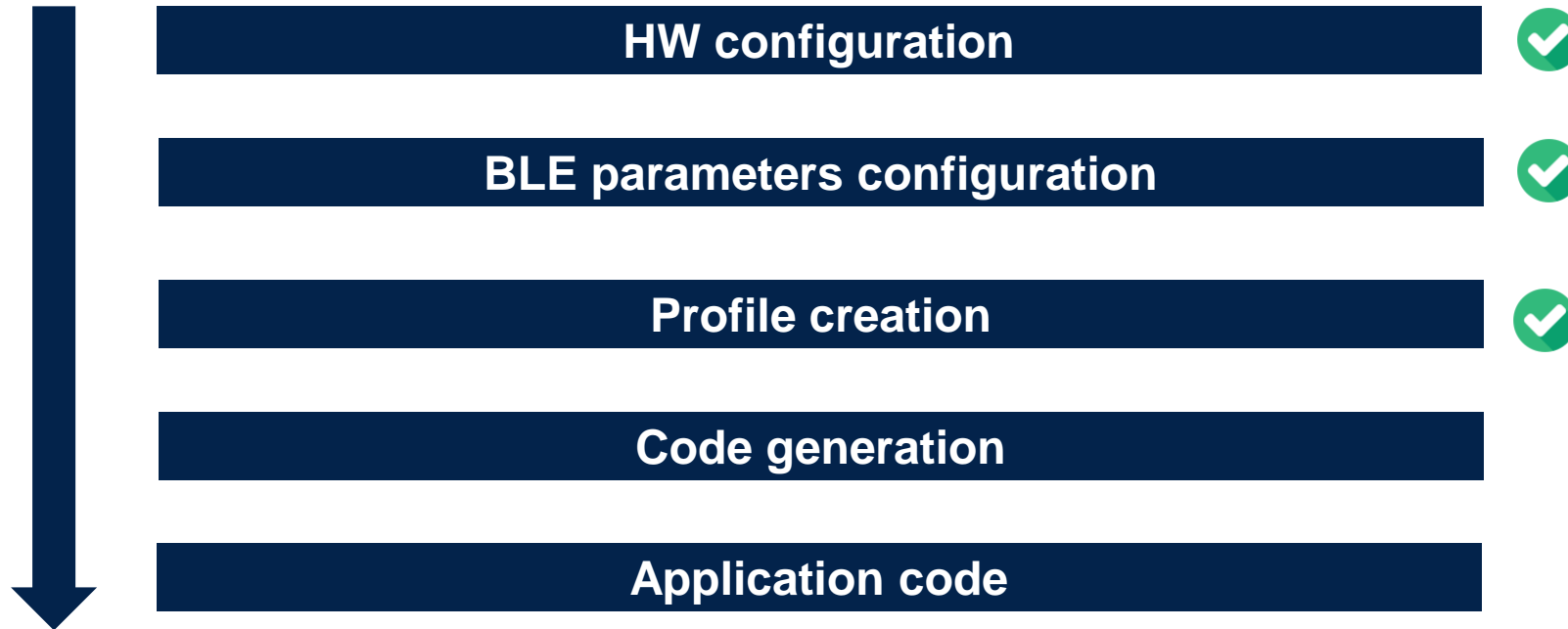






Configuration completed

What's next - Yes code generation



Step 2 : Code generation and user application code

Home
STM32WBA52CGUx - NUCLEO-WBA52CG
Hands-On_WS_WBA52.ioc - Project Manager
1
GENERATE CODE

Pinout & Configuration
Clock Configuration
Project Manager
Tools

2
Project
Code Generator
Advanced Settings

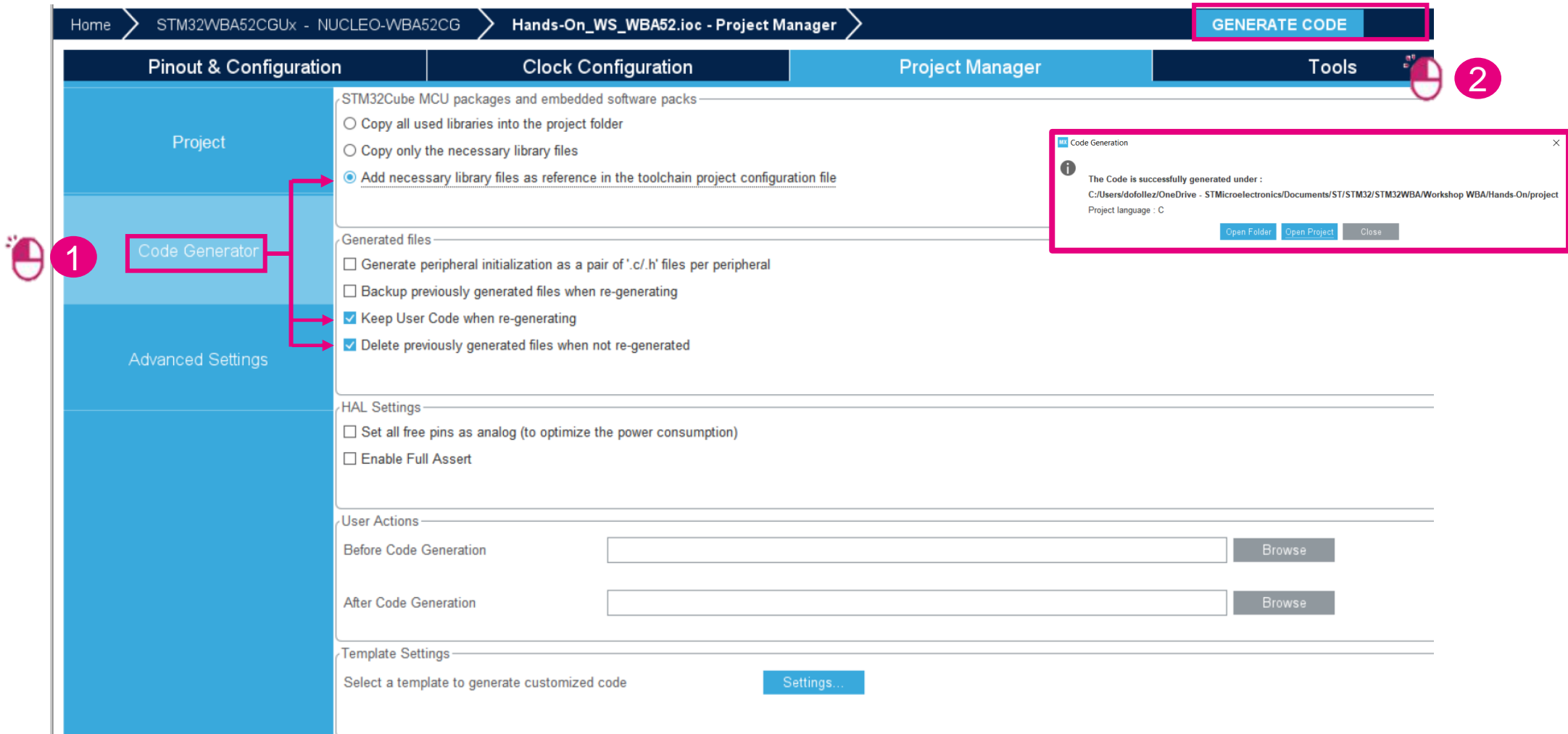
Project Settings
Project Name
Hands-On_WS_WBA52
Project Location
OneDrive - STMicroelectronics\Documents\ST\STM32\STM32WBA\Workshop WBA\Hands-On\project\
Browse
Application Structure
Advanced
Do not generate the main()
Toolchain Folder Location
ve - STMicroelectronics\Documents\ST\STM32\STM32WBA\Workshop WBA\Hands-On\project\Hands-On_WS_WBA52\
Toolchain / IDE
STM32CubeIDE
Generate Under Root

Linker Settings
Minimum Heap Size
0x200
Minimum Stack Size
0x400
Linker settings – Heap and CStack

Thread-safe Settings
CortexM33
Enable multi-threaded support
Thread-safe Locking Strategy
Default – Mapping suitable strategy depending on RTOS selection.

Mcu and Firmware Package
Mcu Reference
STM32WBA52CGUx
Firmware Package Name and Version
STM32Cube FW_WBA V1.1.0
Use Default Firmware Location
Use default fw location
Firmware Relative Path
C:/Users/dofollez/STM32Cube/Repository/STM32Cube_FW_WBA_V1.1.0
Browse

Project configuration



The screenshot displays the STM32CubeMX Project Manager interface for the project "Hands-On_WS_WBA52.ioc". The interface is divided into four main tabs: Pinout & Configuration, Clock Configuration, Project Manager, and Tools. The Project Manager tab is currently active.

On the left sidebar, the "Code Generator" option is highlighted with a red circle and a red arrow pointing to the "Project" section. The "Advanced Settings" section is also visible.

The main content area shows the "Project" configuration options:

- STM32Cube MCU packages and embedded software packs:
 - ☐ Copy all used libraries into the project folder
 - ☐ Copy only the necessary library files
 - ☒ Add necessary library files as reference in the toolchain project configuration file
- Generated files:
 - ☐ Generate peripheral initialization as a pair of '.c/.h' files per peripheral
 - ☐ Backup previously generated files when re-generating
 - ☒ Keep User Code when re-generating
 - ☒ Delete previously generated files when not re-generated
- HAL Settings:
 - ☐ Set all free pins as analog (to optimize the power consumption)
 - ☐ Enable Full Assert
- User Actions:
 - Before Code Generation:
 - After Code Generation:
- Template Settings:
 - Select a template to generate customized code:

A red circle with the number "2" is placed near the "GENERATE CODE" button in the top right corner. A red circle with the number "1" is placed near the "Code Generator" button in the left sidebar.

A "Code Generation" dialog box is open on the right side, displaying the following message:

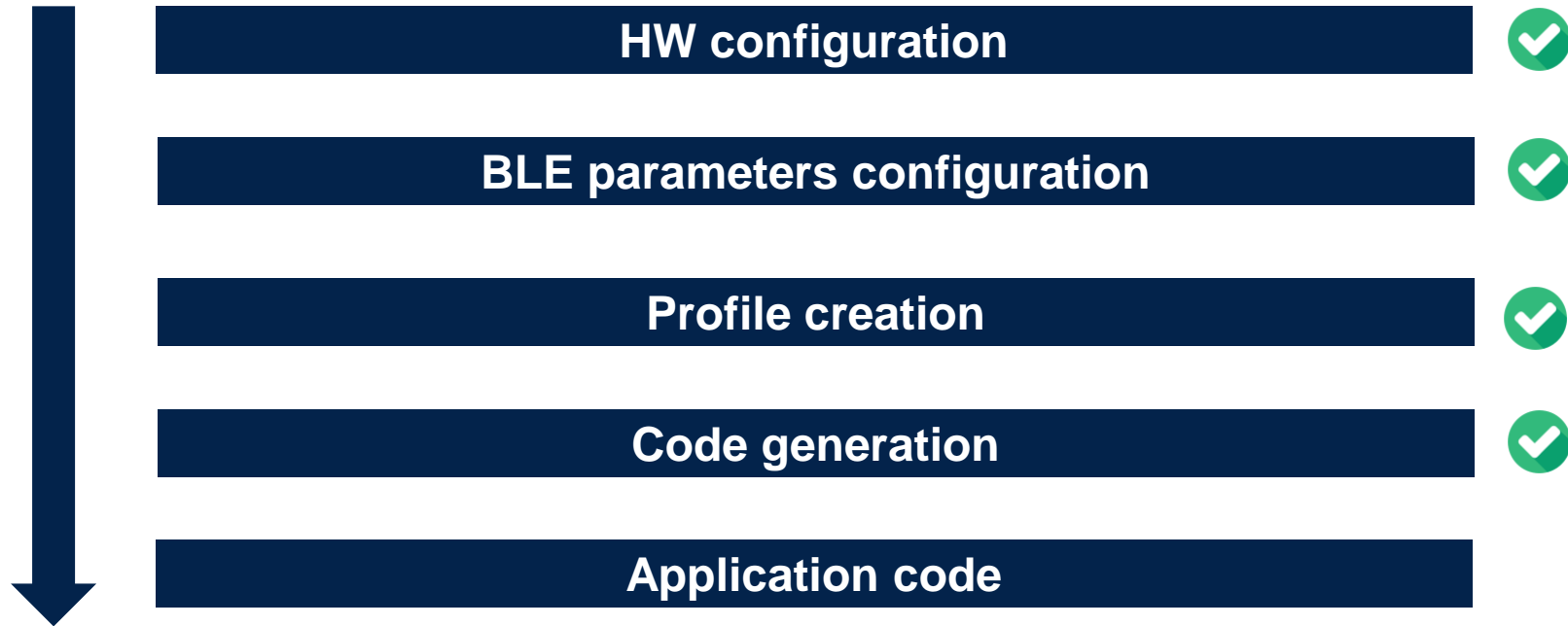
The Code is successfully generated under :
 C:/Users/dofollez/OneDrive - STMicroelectronics/Documents/ST/STM32/STM32WBA/Workshop WBA/Hands-On/project
 Project language : C

Buttons:



Configuration completed

What's next - Yes code generation



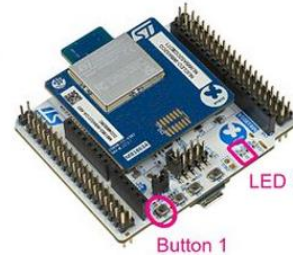


Add application code

Toggle LED from client



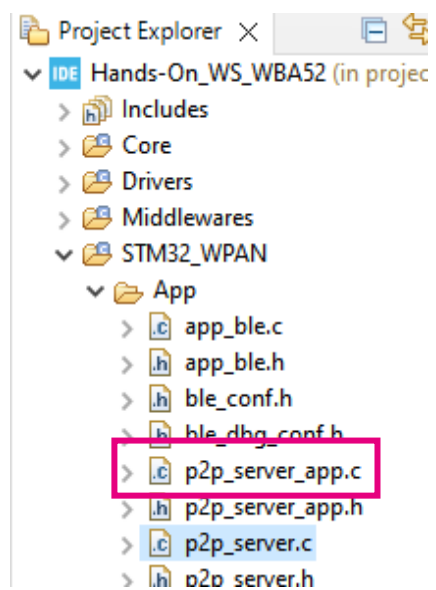
write to My_LED_Char (FE 41)



	Characteristic 1	Characteristic 2
UUID type	128 bits UUID (0x02)	128 bits UUID (0x02)
UUID 128 Input type	Reduced	Reduced
UUID	FE 41	FE 42
Characteristic long name	My_LED_Char	My_Switch_Char
Characteristic Short Name	LED_C	SWITCH_C
Value length	2	2
Length characteristic	Variable	Variable
Encryption key size	0x10	0x10
Char Properties	READ WRITE WRITE_WITHOUT_RESP	NOTIFY
GATT events	GATT_NOTIFY_ATTRIBUTE_WRITE	GATT_NOTIFY_ATTRIBUTE_WRITE



write client procedure triggers an
ACI_GATT_ATTRIBUTE_MODIFIED_VSEVT_CODE
at server application level



in p2p_server_app.c / function P2P_SERVER_Notification()

```
/* USER CODE BEGIN Service1Char1_WRITE_NO_RESP_EV */  
HAL_GPIO_TogglePin(GPIOB, LD2_Pin|LD3_Pin|LD1_Pin);  
/* USER CODE END Service1Char1_WRITE_NO_RESP_EV */
```

How to add a task in sequencer ?

#1 Define a **TaskID** for your « new task » :

In app_conf.h
define a new ID in enum CFG_Task_Id_t
(USER code section)

```
/**
 * These are the lists of task id registered to the sequencer
 * Each task id shall be in the range [0:31]
 */
typedef enum
{
    CFG_TASK_HCI_ASYNC_EVT_ID,
    CFG_TASK_LINK_LAYER,
    CFG_TASK_LINK_LAYER_TEMP_MEAS,
    CFG_TASK_BLE_HOST,
    CFG_TASK_BPKA,
    CFG_TASK_HW_RNG,
    CFG_TASK_AMM_BCKGND,
    CFG_TASK_FLASH_MANAGER_BCKGND,
    CFG_TASK_BLE_TIMER_BCKGND,
    /* USER CODE BEGIN CFG_Task_Id_t */
    TASK_BUTTON_1,
    /* USER CODE END CFG_Task_Id_t */
    CFG_TASK_NBR /* Shall be LAST in the list */
} CFG_Task_Id_t;
```

#2 **UTIL_SEQ_RegTask()** to register your task in the sequencer

```
UTIL_SEQ_RegTask(1U << TASK_BUTTON_1, UTIL_SEQ_RFU, APPE_Button1Action);
```

It associates a callback to your Task.
To be done only Once

#3 **UTIL_SEQ_SetTask()** to notify the sequencer shall execute the registered task

```
UTIL_SEQ_SetTask(1U << TASK_BUTTON_1, CFG_SEQ_PRIO_0);
```

It notify the sequencer that the task must be triggered.
It will generate a call to registered function
(here : APPE_Button1Action())



Add application code

Raise an alarm from device to Smartphone(1/3)



press button

notify peer device trough SWITCH_C (FE 42)



	Characteristic 1	Characteristic 2
UUID type	128 bits UUID (0x02)	128 bits UUID (0x02)
UUID 128 Input type	Reduced	Reduced
UUID	FE 41	FE 42
Characteristic long name	My_LED_Char	My_Switch_Char
Characteristic Short Name	LED_C	SWITCH_C
Value length	2	2
Length characteristic	Variable	Variable
Encryption key size	0x10	0x10
Char Properties	READ WRITE_WITHOUT_RESP	NOTIFY
GATT events	GATT_NOTIFY_ATTRIBUTE_WRITE	GATT_NOTIFY_ATTRIBUTE_WRITE

On press button use notify procedure use to push data to client

#1 need to define specific task for button press

In app_conf.h

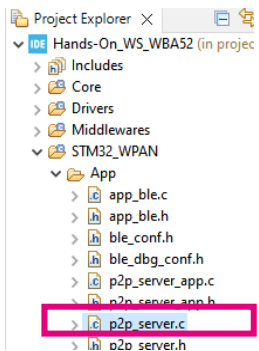
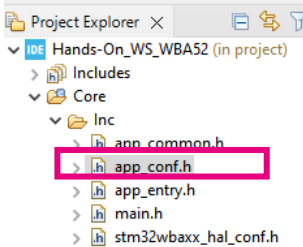
```
/* USER CODE BEGIN CFG_Task_Id_t */
TASK_BUTTON_1,
/* USER CODE END CFG_Task_Id_t*/
```

#2 register a « button task »

in p2p_server_app.c / function P2P_SERVER_APP_Init

```
/* USER CODE BEGIN Service1_APP_Init */
UTIL_SEQ_RegTask( 1U << TASK_BUTTON_1, UTIL_SEQ_RFU, P2P_SERVER_Switch_c_SendNotification);
/* USER CODE END Service1_APP_Init */
```

Function generated by CubeMx as per as Characteristic Short Name





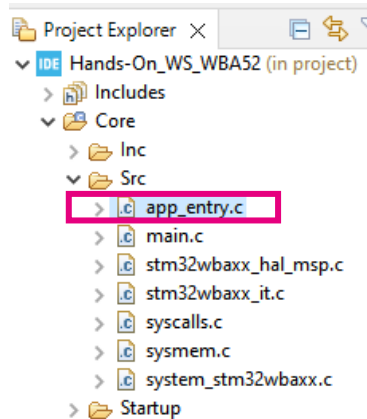
Add application code

Raise a notification from device to Smartphone(2/3)



press button

notify peer device trough SWITCH_C (FE 42)



#3 Manage Button1 interrupt : implement IRQ callback

In app_entry.c / function HAL_GPIO_EXTI_Rising_Callback

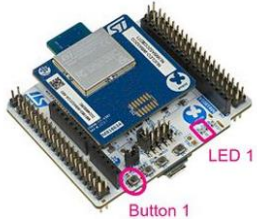
```
/* USER CODE BEGIN FD_WRAP_FUNCTIONS */
void HAL_GPIO_EXTI_Rising_Callback(uint16_t GPIO_Pin)
{
    if (GPIO_Pin == B1_Pin)
    {
        UTIL_SEQ_SetTask(1U << TASK_BUTTON_1, CFG_SEQ_PRIO_0);
    }

    return;
} /* USER CODE END FD_WRAP_FUNCTIONS */
```



Add application code

Raise a notification from device to Smartphone(3/3)



notify peer device trough SWITCH_C (FE 42)



	Characteristic 1	Characteristic 2
UUID type	128 bits UUID (0x02)	128 bits UUID (0x02)
UUID 128 Input type	Reduced	Reduced
UUID	FE 41	FE 42
Characteristic long name	My_LED_Char	My_Switch_Char
Characteristic Short Name	LED_C	SWITCH_C
Value length	2	2
Length characteristic	Variable	Variable
Encryption key size	0x10	0x10
Char Properties	READ WRITE_WITHOUT_RESP	NOTIFY
GATT events	GATT_NOTIFY_ATTRIBUTE_WRITE	GATT_NOTIFY_ATTRIBUTE_WRITE

#4 Manage BLE notification procedure

In p2p_server_app.c/ function P2P_SERVER_Switch_c_SendNotification

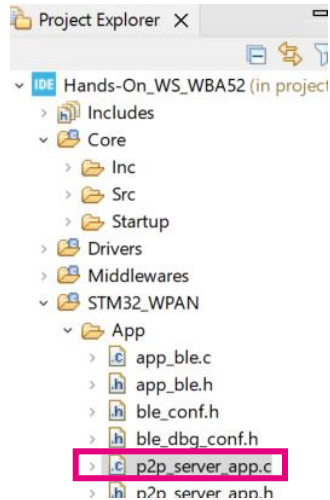
```
/* USER CODE BEGIN Service1Char2_NS_1 */
a_P2P_SERVER_UpdateCharData[0] = 0x01; /* Device Led selection */
a_P2P_SERVER_UpdateCharData[1] = 0x00;
/* Update notification data length */
p2p_server_notification_data.Length = (p2p_server_notification_data.Length) + 2;

notification_on_off = SWITCH_c_NOTIFICATION_ON;

/* USER CODE END Service1Char2_NS_1 */
```

Peer to Peer Service - SWITCH Characteristic		
Byte Index	0	1
Name	Button Selection	Status
Value	0x01: button 1	0x00 or 0x01

STM32WBA Bluetooth® LE – Peer 2 Peer Applications - stm32mcu



P2P_SERVER_UpdateValue

aci_gatt_update_char_value

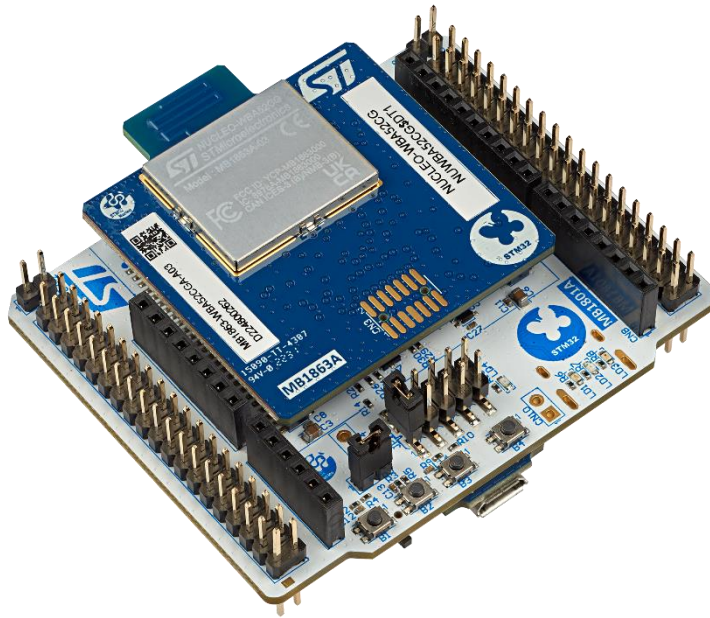
BLE stack API



life.augmented



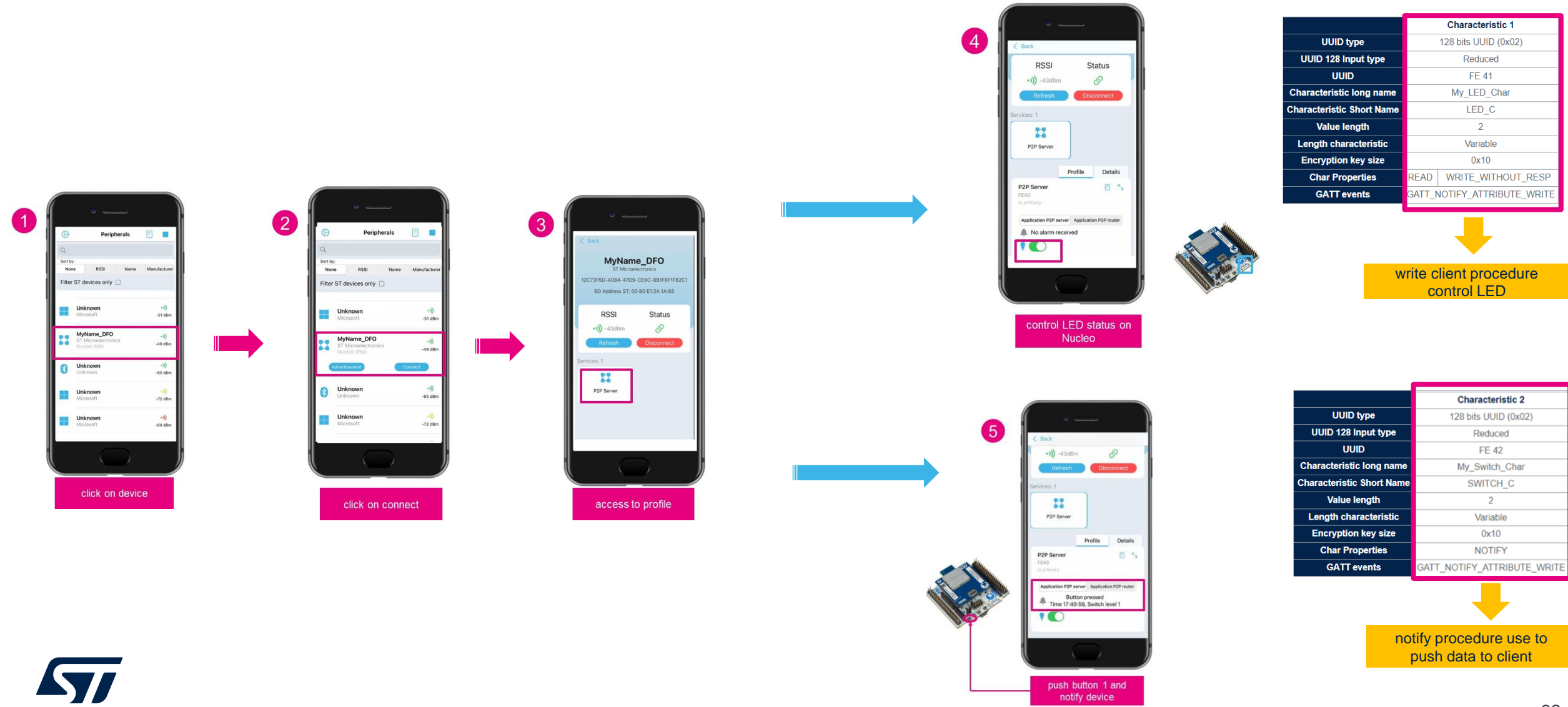
Open your App and Connect



ST BLE Toolbox

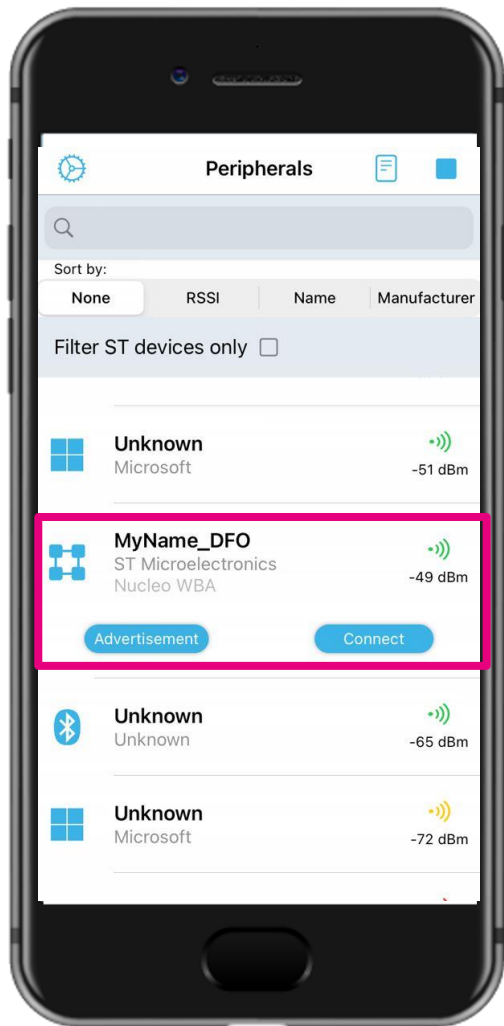


Open your App and Connect (1/2)

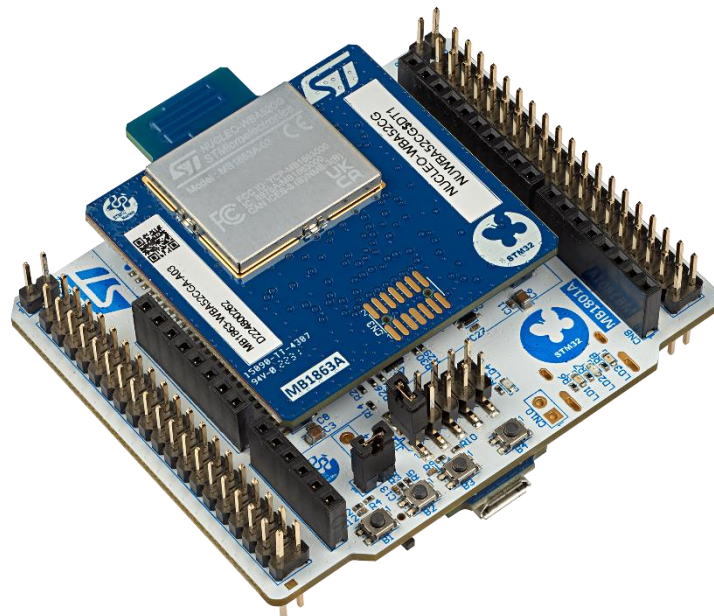


STBLE Toolbox (Connection)

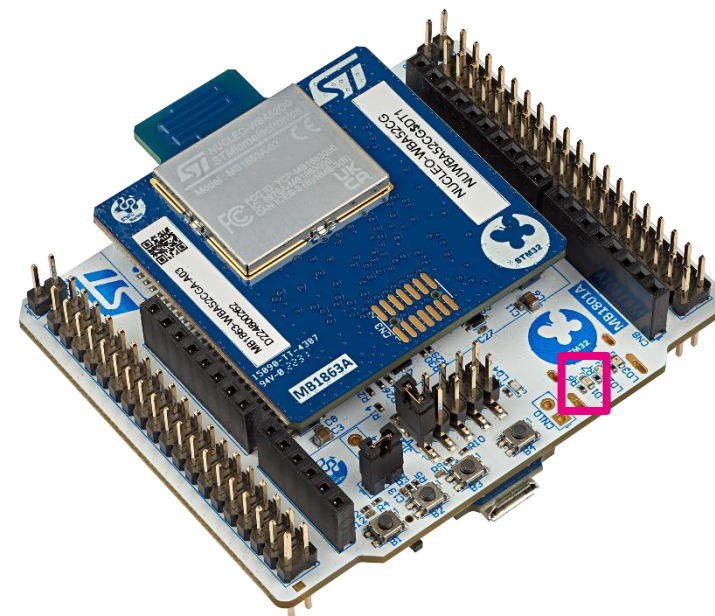
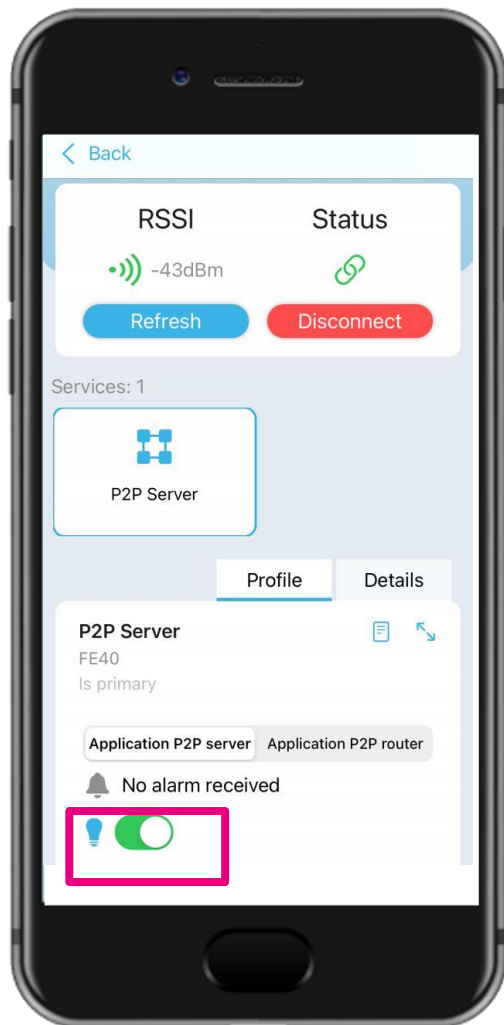
1



2



STBLE Toolbox (LED)

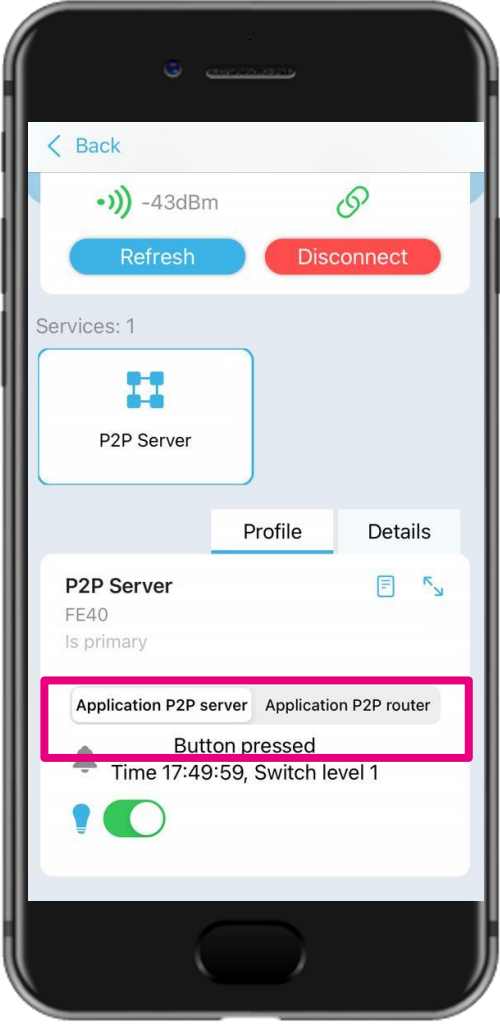


control LED status on
Nucleo



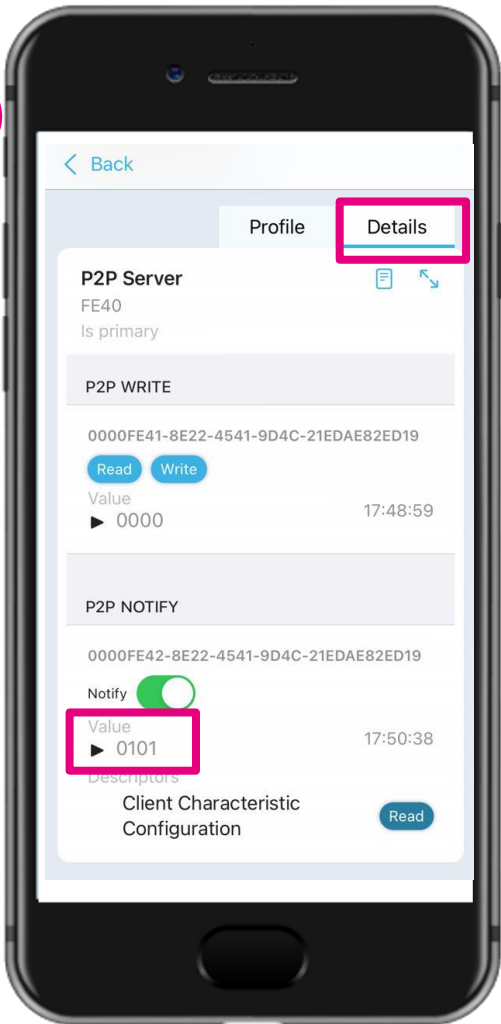
STBLE Toolbox (Push Button)

1

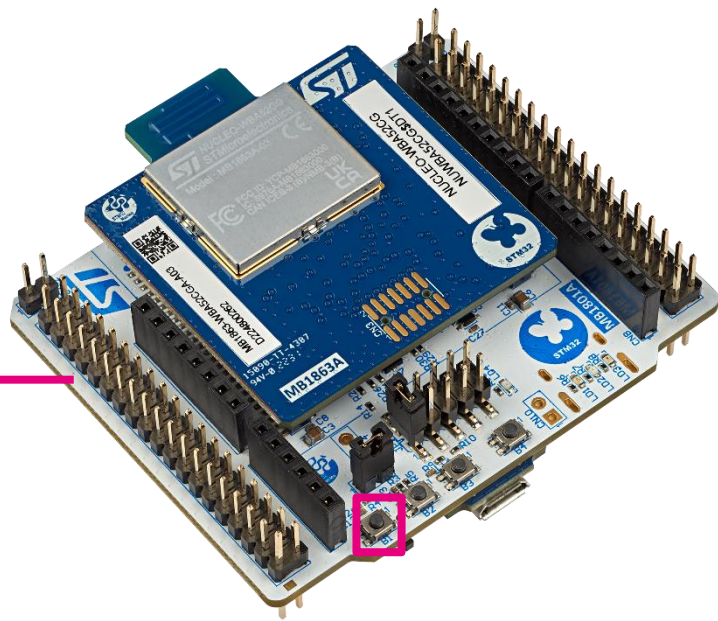
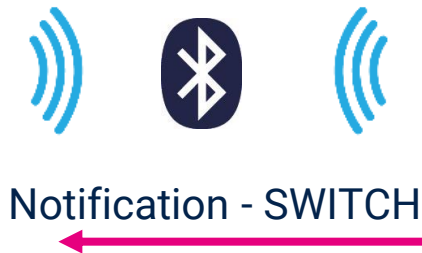


push button 1 and notify device

2



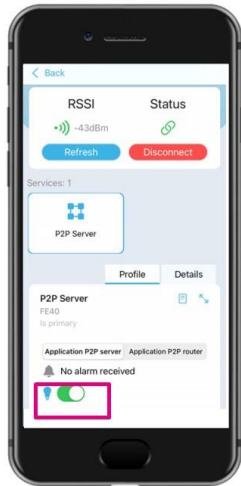
click on details to see bytes sent/received





Open your App and Connect call stack

4



control LED status on Nucleo

3

2

1

Add break point line 111



BLE write procedure initiated by client

ACI_GATT_ATTRIBUTE_MODIFIED event received at application level

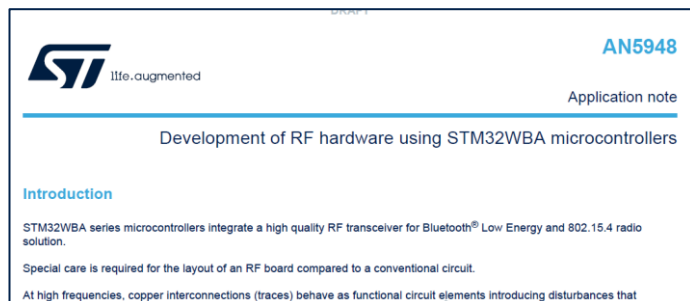
```
97 /* USER CODE END Service1_Notification_1 */
98 switch(p_Notification->EvtOpcode)
99 {
100 /* USER CODE BEGIN Service1_Notification_Service1_EvtOpcode */
101 /* USER CODE END Service1_Notification_Service1_EvtOpcode */
102
103 case P2P_SERVER_LED_C_READ_EVT:
104 /* USER CODE BEGIN Service1Char1_READ_EVT */
105 /* USER CODE END Service1Char1_READ_EVT */
106 break;
107
108 case P2P_SERVER_LED_C_WRITE_NO_RESP_EVT:
109 /* USER CODE BEGIN Service1Char1_WRITE_NO_RESP_EVT */
110 HAL_GPIO_TogglePin(GPIOB, LD2_Pin|LD3_Pin|LD1_Pin);
111 /* USER CODE END Service1Char1_WRITE_NO_RESP_EVT */
112 break;
113
114 case P2P_SERVER_SWITCH_C_NOTIFY_ENABLED_EVT:
115 /* USER CODE BEGIN Service1Char2_NOTIFY_ENABLED_EVT */
116 /* USER CODE END Service1Char2_NOTIFY_ENABLED_EVT */
117 break;
118
119 case P2P_SERVER_SWITCH_C_NOTIFY_DISABLED_EVT:
120 /* USER CODE BEGIN Service1Char2_NOTIFY_DISABLED_EVT */
121 /* USER CODE END Service1Char2_NOTIFY_DISABLED_EVT */
122 break;
123
124 /* USER CODE END Service1_Notification_1 */
125 }
```

```
97 /* USER CODE END Service1_Notification_1 */
98 switch(p_Notification->EvtOpcode)
99 {
100 /* USER CODE BEGIN Service1_Notification_Service1_EvtOpcode */
101 /* USER CODE END Service1_Notification_Service1_EvtOpcode */
102
103 case P2P_SERVER_LED_C_READ_EVT:
104 /* USER CODE BEGIN Service1Char1_READ_EVT */
105 /* USER CODE END Service1Char1_READ_EVT */
106 break;
107
108 case P2P_SERVER_LED_C_WRITE_NO_RESP_EVT:
109 /* USER CODE BEGIN Service1Char1_WRITE_NO_RESP_EVT */
110 HAL_GPIO_TogglePin(GPIOB, LD2_Pin|LD3_Pin|LD1_Pin);
111 /* USER CODE END Service1Char1_WRITE_NO_RESP_EVT */
112 break;
113
114 case P2P_SERVER_SWITCH_C_NOTIFY_ENABLED_EVT:
115 /* USER CODE BEGIN Service1Char2_NOTIFY_ENABLED_EVT */
116 /* USER CODE END Service1Char2_NOTIFY_ENABLED_EVT */
117 break;
118
119 case P2P_SERVER_SWITCH_C_NOTIFY_DISABLED_EVT:
120 /* USER CODE BEGIN Service1Char2_NOTIFY_DISABLED_EVT */
121 /* USER CODE END Service1Char2_NOTIFY_DISABLED_EVT */
122 break;
123
124 /* USER CODE END Service1_Notification_1 */
125 }
```


HW project development & certification

A complete set of documentation

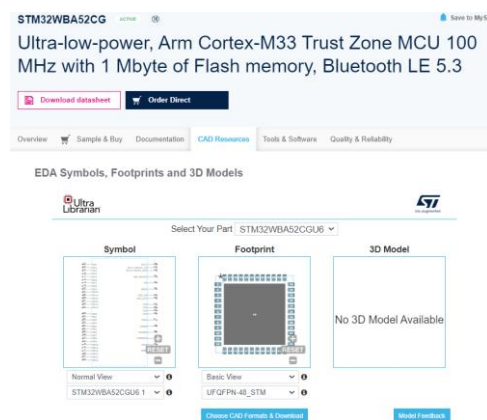
AN5948 : Development of RF Hardware using STM32WBA



COMING SOON on st.com!

- ➡ RF basis generalities
- ➡ Schematics & components selection guidelines
- ➡ STM32WBA5x layout checklist & guidelines

STM32WBA5x CAD resources on st.com



- ➡ Download STM32WBA5x symbol
- ➡ Download STM32WBA5x footprint

HW design with STM32WBA5x : key points

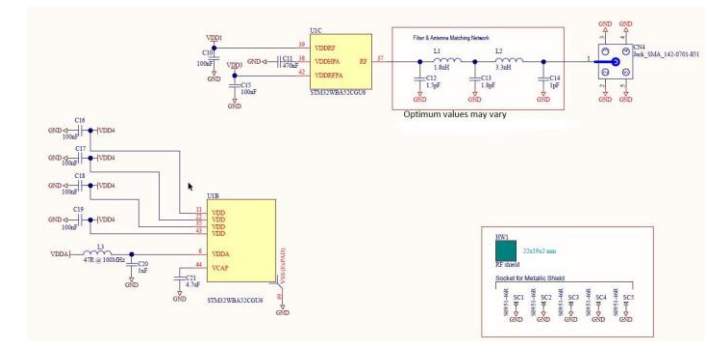
- HW design should be initiated based on documentation shown on previous slides
- Next slides are highlighting the key points you must pay attention when designing schematics and layout:

➤ HSE 32MHz Xtal requirements:

- STM32WBA5x includes internal programmable capacitances to trim the crystal frequency
- Select XTAL with 8pF load capacitors
- Recommended part (or equivalent): NX1612SA-32MHZ-EXS00A-CS09166

➤ LSE or LSI selection. LSE 32kHz Xtal requirements if used:

- LSE mandatory for accurate RTC calendar application.
- BOM optimized (save 32kHz Xtal cost)

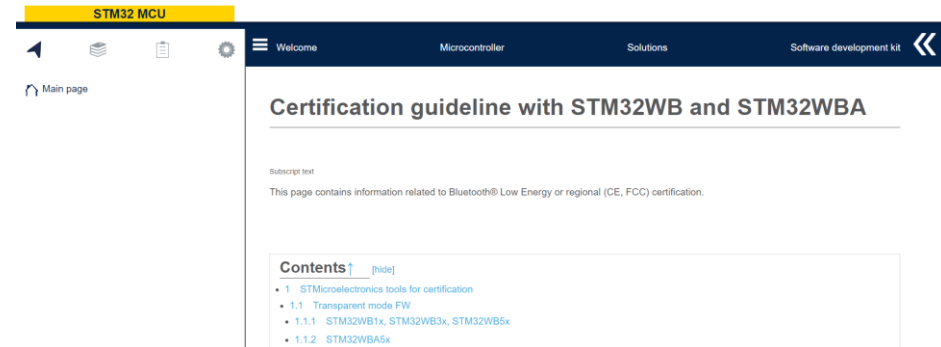


HW design with STM32WBA5x : key points

- RF matching & filtering:
 - Integrated balun so single ended RF matching.
 - Very limited number of discrete components for STM32WBAx matching and filtering.
- Power management. SMPS implementation for STM32WBA55 use case:
 - STM32WBA55 embeds a SMPS that can be used to improve power efficiency.
- Main layout recommendation:
 - Refer our various reference kits layout (Gerbers and Altium files available).
 - 4 layers stack-up recommended but 2 layers is possible.

STM32WBA5x : a certified solution

- STM32WBA5x is compliant in regards of regional (CE, FCC etc.) and Bluetooth requirements.
- We are providing complete set of documentation, FW and tools to certify your product.



Full set of tools and documentation

- Certification guideline on wiki ([Certification guideline with STM32WB and STM32WBA - stm32mcu](#))
- Transparent mode FW available in [STM32CubeWBA](#) MCU Package
- [STM32CubeMonRF](#) PC tool

Bluetooth certification

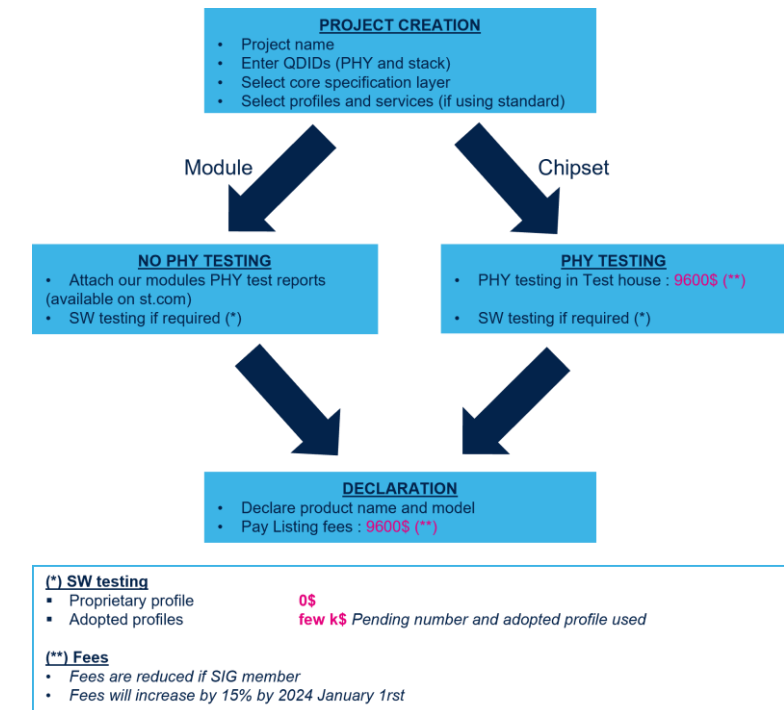
- STM32WBA5x is having reference QDIDs (components and stack) at Bluetooth SIG.
- Customer musty performed PHY testing and declare its product at Bluetooth SIG.

PHY QDID

Package	Part number	Cut version	RF PHY QDID
QFN48	STM32WBA52 (BLE5.4)	1.x	197135 (TCRL 2022-2)

Stack QDID

Features	Host Stack version	QDID
4.0 HCI Low Energy LL with extended advertising – ATT – GAP – GATT – L2CAP with Enhanced Connected Oriented Channel -SMP BLE 5.3	STM32Cube_WBA_BLE_HCI_STACK STM32Cube_WBA_BLE_FULL_STACK	198195 (TCRL 2022-1)



Refer wiki [Certification guideline](https://www.st.com/en/development-tools/certification-guideline.html) on st.com

Thank you

© STMicroelectronics - All rights reserved.

The STMicroelectronics corporate logo is a registered trademark of the STMicroelectronics group of companies. All other names are the property of their respective owners.



life.augmented