life.augmented

Welcome to
STM32WBA52 workshop

Hands-on #1
Build basic p2pServer application and connect

Workshop team

STM32

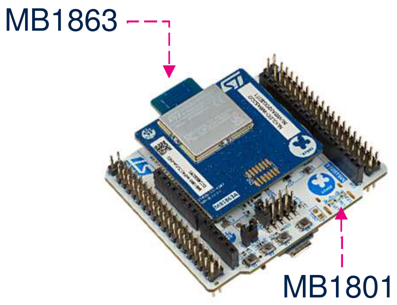
Prerequisites Refresh

SW prerequisites

- STM32CubeWBA MCU package (v1.1.0 or up)
- IDE: STM32CubeIDE (v1.13.0 or up)
- A serial terminal (e.g. TeraTerm)
- ST BLE ToolBox Smartphone application

HW prerequisites

- USB A to Micro-B Cable



ST BLE Toolbox

Download on the App Store

GET IT ON Google Play

Agenda

1 Hands-on presentation

2 Step 1: STM32CubeMX/STM32CubeIDE initialization for STM32WBA Nucleo board

3 Step2 : Advertising and BLE application configuration and explanation

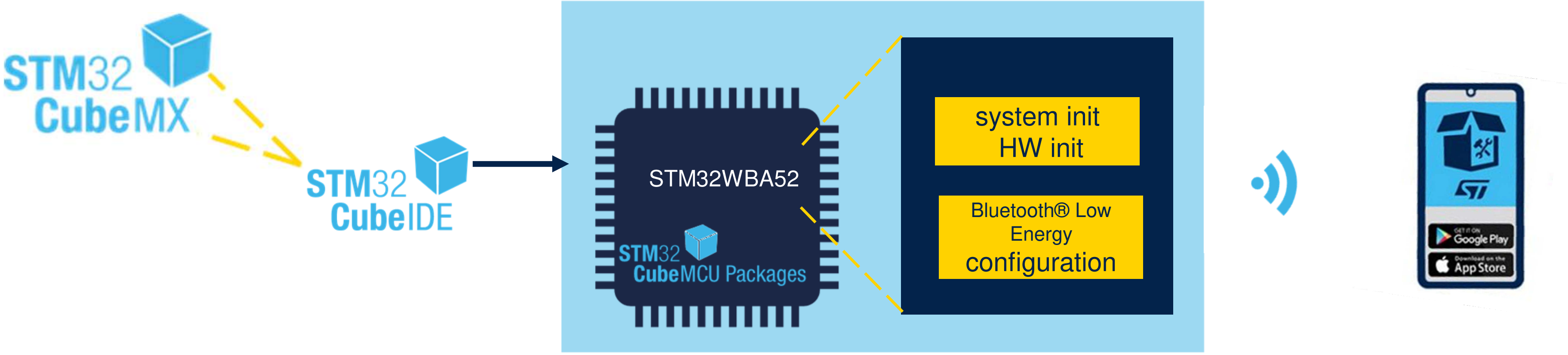
4 Step 3 : Code generation and user application code

5 “bonus track” : Adding logs

Hands-on presentation

Purpose

- The purpose is to start from WBA52 chipset level and build a basic server (p2pServer) application using STM32CubeMX/STM32CubeIDE
- In this first part, focus is to get device visible and connectable from my smartphone

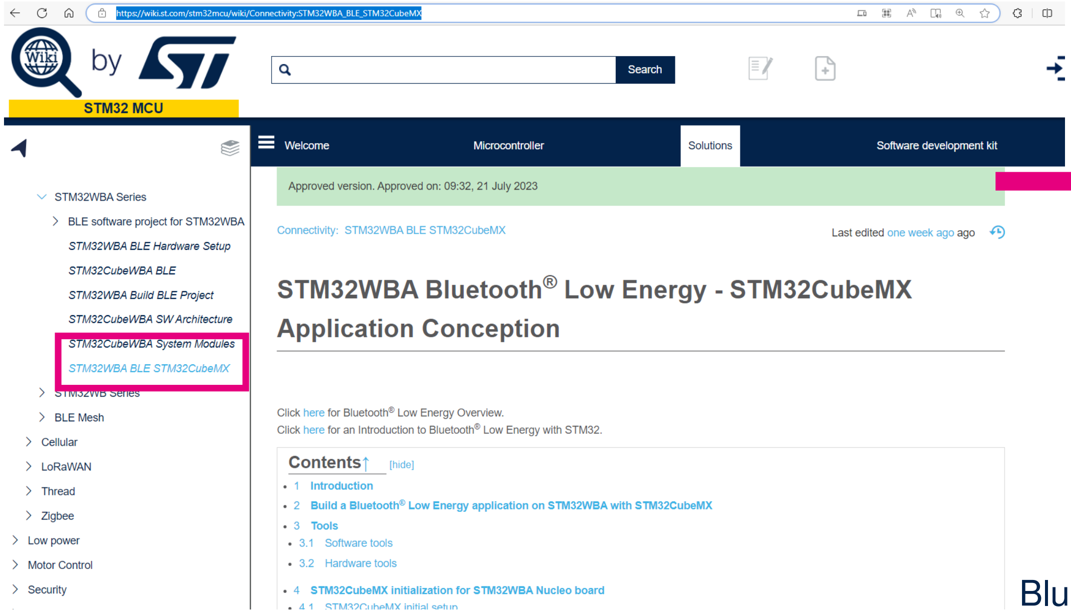


Bluetooth® Low Energy peripheral advertising

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

Source

Hands-on based on
https://wiki.st.com/stm32mcu/wiki/Connectivity:STM32WBA_BLE_STM32CubeMX



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

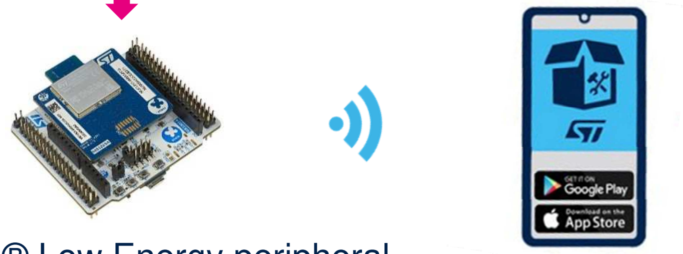
STM32WBA Bluetooth® Low Energy - STM32CubeMX Application Conception

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 Bluetooth® Low Energy peripheral application



Bluetooth® Low Energy peripheral exposing data to central device

Legenda

- Slides including following symbol are purely theoretical ones

- Source code for development is included inside blue boxes

```
HAL_Delay(500);
```

Step 1 : STM32CubeMX initialization for STM32WBA Nucleo board

life.augmented

STM32CubeMX capabilties

STM32CubeMX : “Standalone version” or “integrated version” into STM32CubeIDE 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)

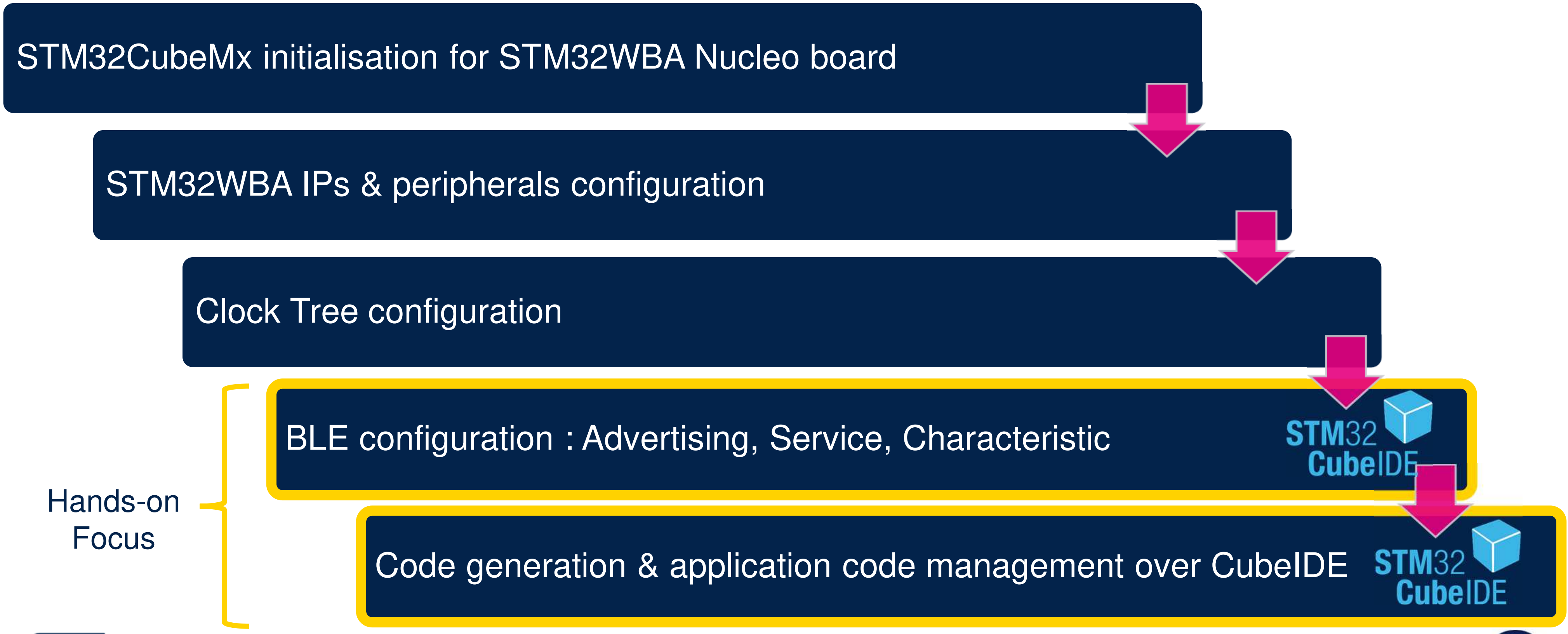
3 **Chipset level**
require to configure your HW (PCB) & your application

→ STM32WBA wiki page focus

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

STM32CubeMX can be standalone application but also part of STM32CubeIDE

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

Code generation & application code management over CubeIDE

Hands-on Focus

STM32CubeMx design from chispet level
Hands-on focus (1/2)

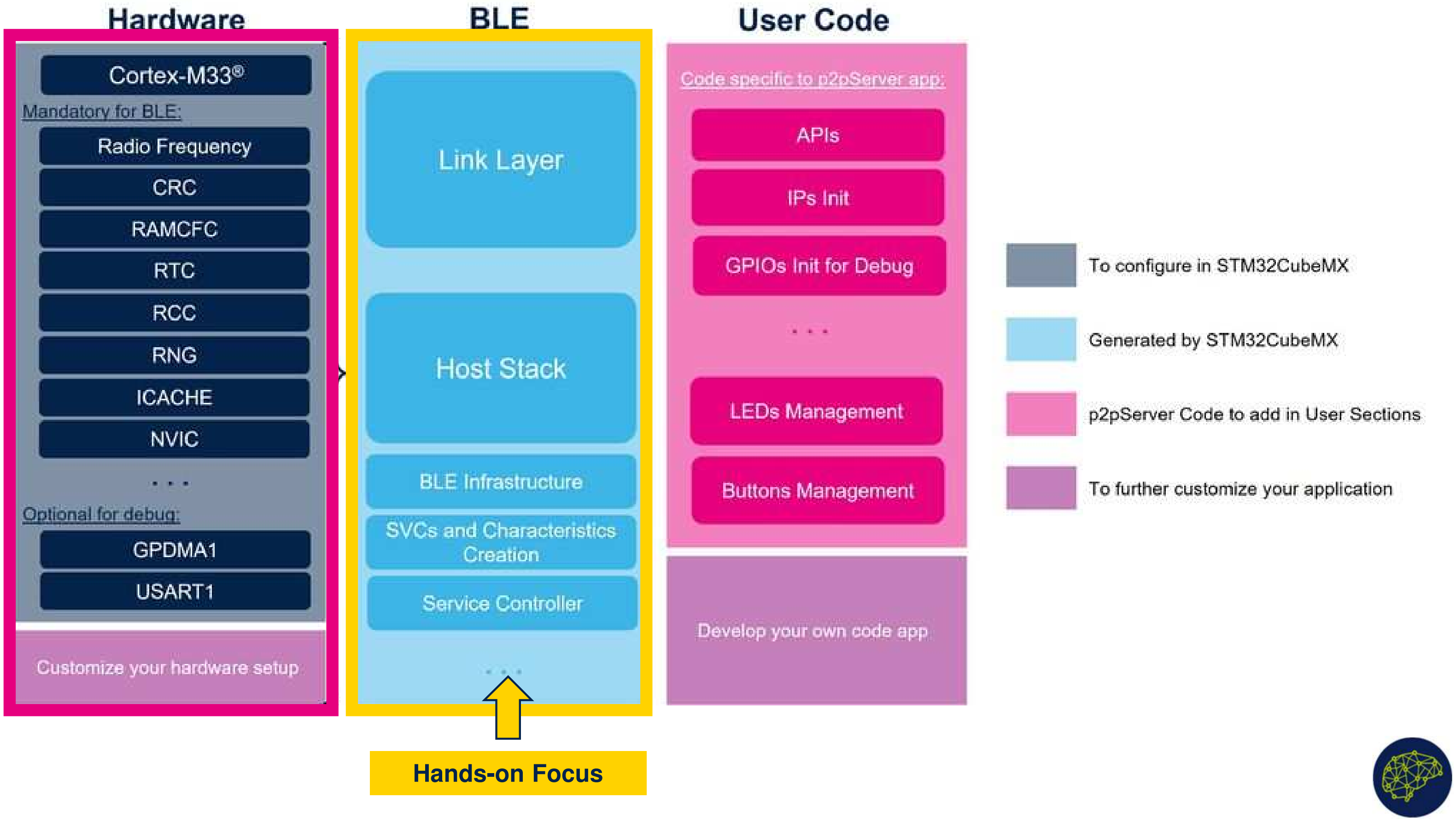
3 **Chipset level**
require to configure your HW (PCB) & your application

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



Hardware

Cortex-M33®

Mandatory for BLE:

- Radio Frequency
- CRC
- RAMCFC
- RTC
- RCC
- RNG
- ICACHE
- NVIC
- . . .

Optional for debug:

- GPDMA1
- USART1

Customize your hardware setup

BLE

- Link Layer
- Host Stack
- BLE Infrastructure
- SVCs and Characteristics Creation
- Service Controller
- . . .

Hands-on Focus

User Code

Code specific to p2pServer app:

- APIs
- IPs Init
- GPIOs Init for Debug
- . . .
- LEDs Management
- Buttons Management

Develop your own code app

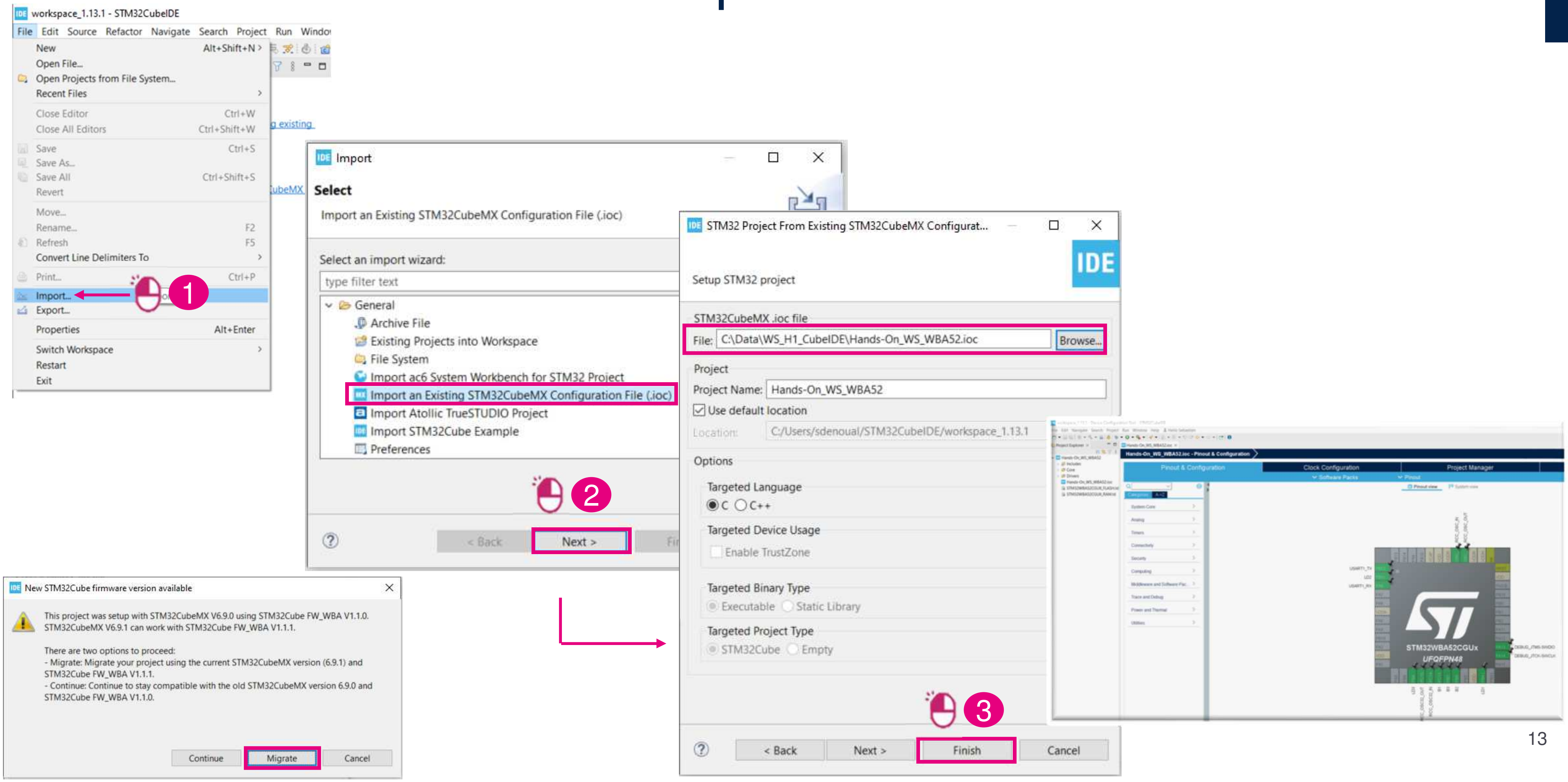
▨ To configure in STM32CubeMX

▨ Generated by STM32CubeMX

▨ p2pServer Code to add in User Sections

▨ To further customize your application

Open and Start STM32CubeIDE



workspace_1.13.1 - STM32CubeIDE
File Edit Source Refactor Navigate Search Project Run Windo
New Alt+Shift+N
Open File...
Open Projects from File System...
Recent Files
Close Editor Ctrl+W
Close All Editors Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename... F2
Refresh F5
Convert Line Delimiters To
Print... Ctrl+P
Import...
Export...
Properties Alt+Enter
Switch Workspace
Restart
Exit
1

Import

Select

Import an Existing STM32CubeMX Configuration File (.ioc)

Select an import wizard:

type filter text
General
Archive File
Existing Projects into Workspace
File System
Import ac6 System Workbench for STM32 Project
Import an Existing STM32CubeMX Configuration File (.ioc)
Import Atollic TrueSTUDIO Project
Import STM32Cube Example
Preferences
2
< Back
Next >

STM32 Project From Existing STM32CubeMX Configurat...

Setup STM32 project

STM32CubeMX .ioc file

File: C:\Data\WS_H1_CubeIDE\Hands-On_WS_WBA52.ioc

Project

Project Name: Hands-On_WS_WBA52

☑ Use default location

Location: C:/Users/sdenoual/STM32CubeIDE/workspace_1.13.1

Options

Targeted Language

● C ○ C++

Targeted Device Usage

□ Enable TrustZone

Targeted Binary Type

● Executable ○ Static Library

Targeted Project Type

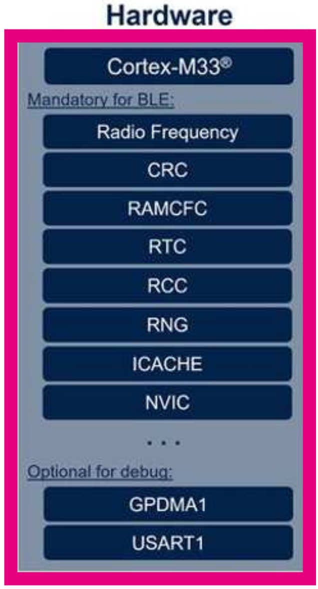
● STM32Cube ○ Empty

New STM32Cube firmware version available

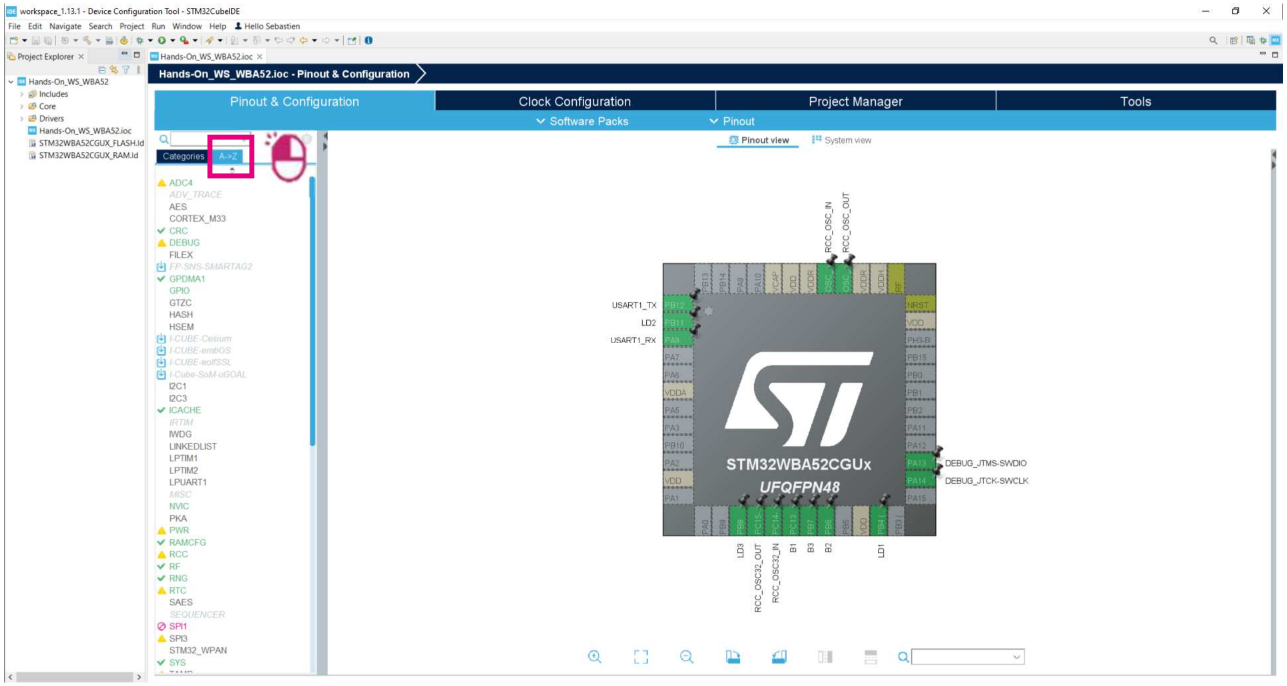
This project was setup with STM32CubeMX V6.9.0 using STM32Cube FW_WBA V1.1.0.
STM32CubeMX V6.9.1 can work with STM32Cube FW_WBA V1.1.1.

There are two options to proceed:
- Migrate: Migrate your project using the current STM32CubeMX version (6.9.1) and STM32Cube FW_WBA V1.1.1.
- Continue: Continue to stay compatible with the old STM32CubeMX version 6.9.0 and STM32Cube FW_WBA V1.1.0.

Peripherals in place to start BLE configuration !



Hands-On_WS_WBA52.ioc

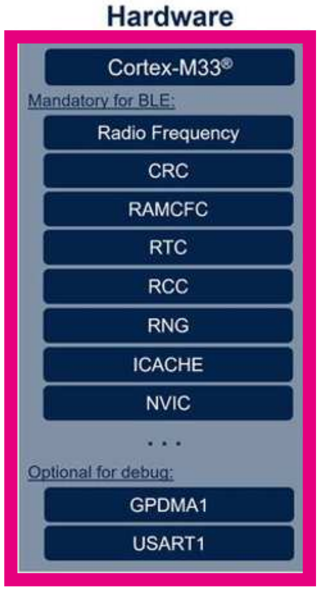


Hands-On_WS_WBA52.ioc

- 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



Hardware

Cortex-M33®

Mandatory for BLE:

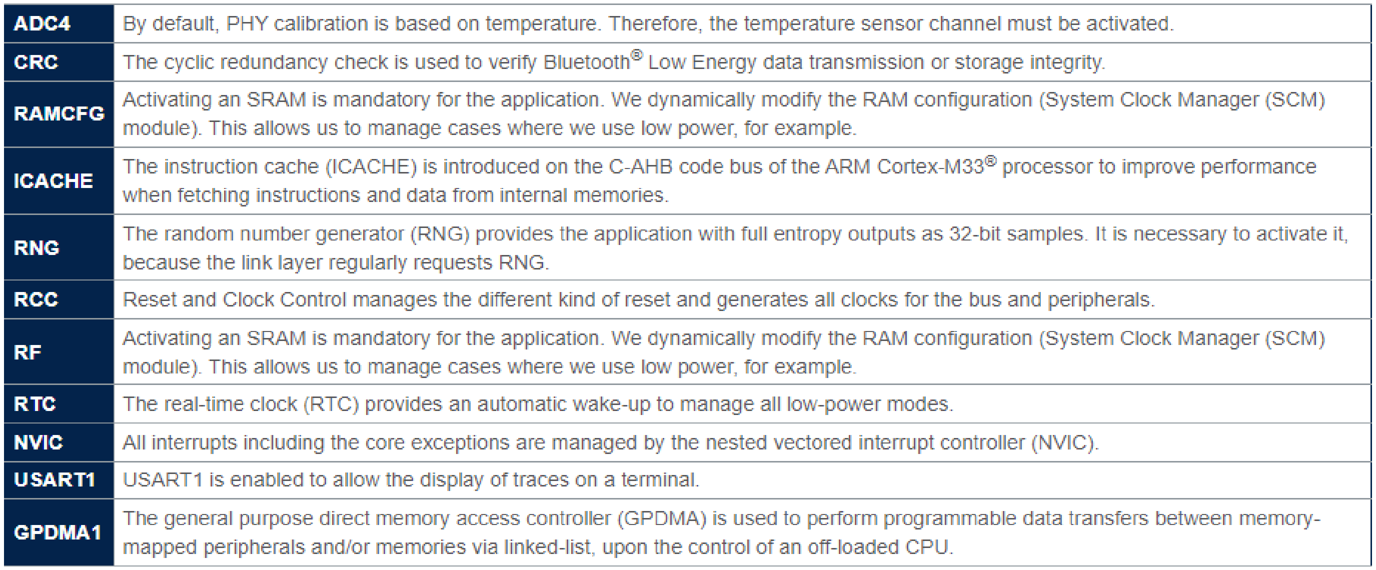
- Radio Frequency
- CRC
- RAMCFC
- RTC
- RCC
- RNG
- ICACHE
- NVIC
- . . .

Optional for debug:

- GPDMA1
- USART1

BLE activation

debug

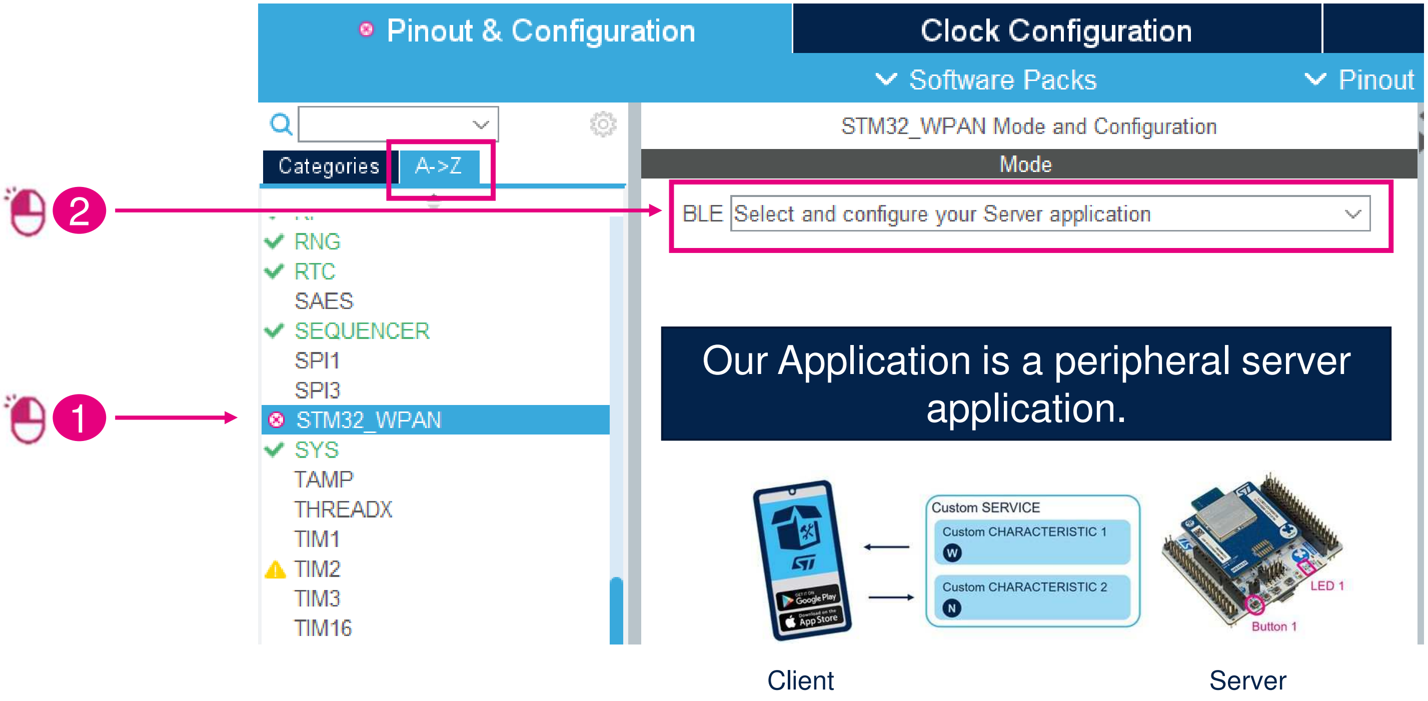


ADC4	By default, PHY calibration is based on temperature. Therefore, the temperature sensor channel must be activated.
CRC	The cyclic redundancy check is used to verify Bluetooth® Low Energy data transmission or storage integrity.
RAMCFG	Activating an SRAM is mandatory for the application. We dynamically modify the RAM configuration (System Clock Manager (SCM) module). This allows us to manage cases where we use low power, for example.
ICACHE	The instruction cache (ICACHE) is introduced on the C-AHB code bus of the ARM Cortex-M33® processor to improve performance when fetching instructions and data from internal memories.
RNG	The random number generator (RNG) provides the application with full entropy outputs as 32-bit samples. It is necessary to activate it, because the link layer regularly requests RNG.
RCC	Reset and Clock Control manages the different kind of reset and generates all clocks for the bus and peripherals.
RF	Activating an SRAM is mandatory for the application. We dynamically modify the RAM configuration (System Clock Manager (SCM) module). This allows us to manage cases where we use low power, for example.
RTC	The real-time clock (RTC) provides an automatic wake-up to manage all low-power modes.
NVIC	All interrupts including the core exceptions are managed by the nested vectored interrupt controller (NVIC).
USART1	USART1 is enabled to allow the display of traces on a terminal.
GPDMA1	The general purpose direct memory access controller (GPDMA) is used to perform programmable data transfers between memory-mapped peripherals and/or memories via linked-list, upon the control of an off-loaded CPU.

Step2 : Advertising and Bluetooth® Low Energy GAP/GATT custom application configuration

life.augmented

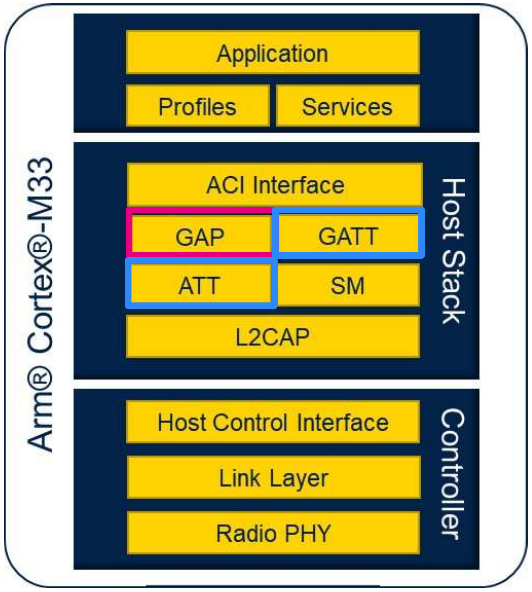
Enabling Bluetooth® Low Energy

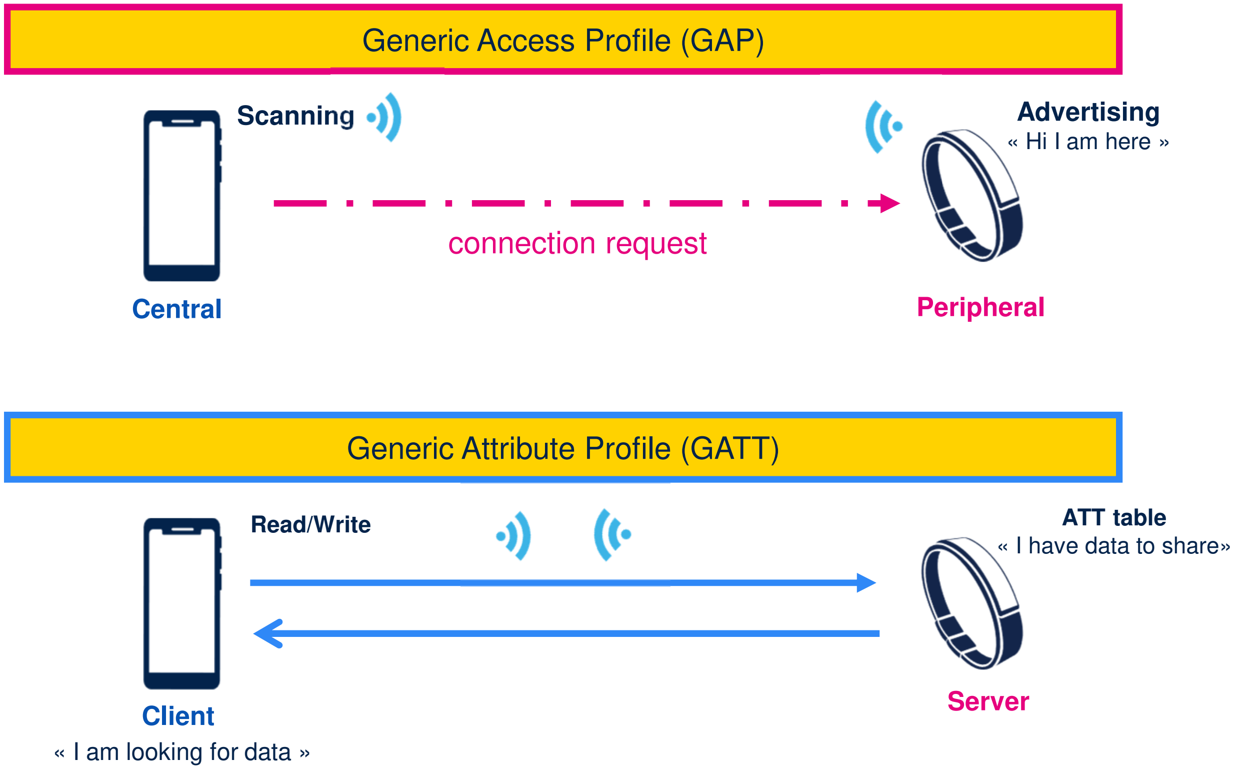


BLE Select and configure your Server application

Our Application is a peripheral server application.

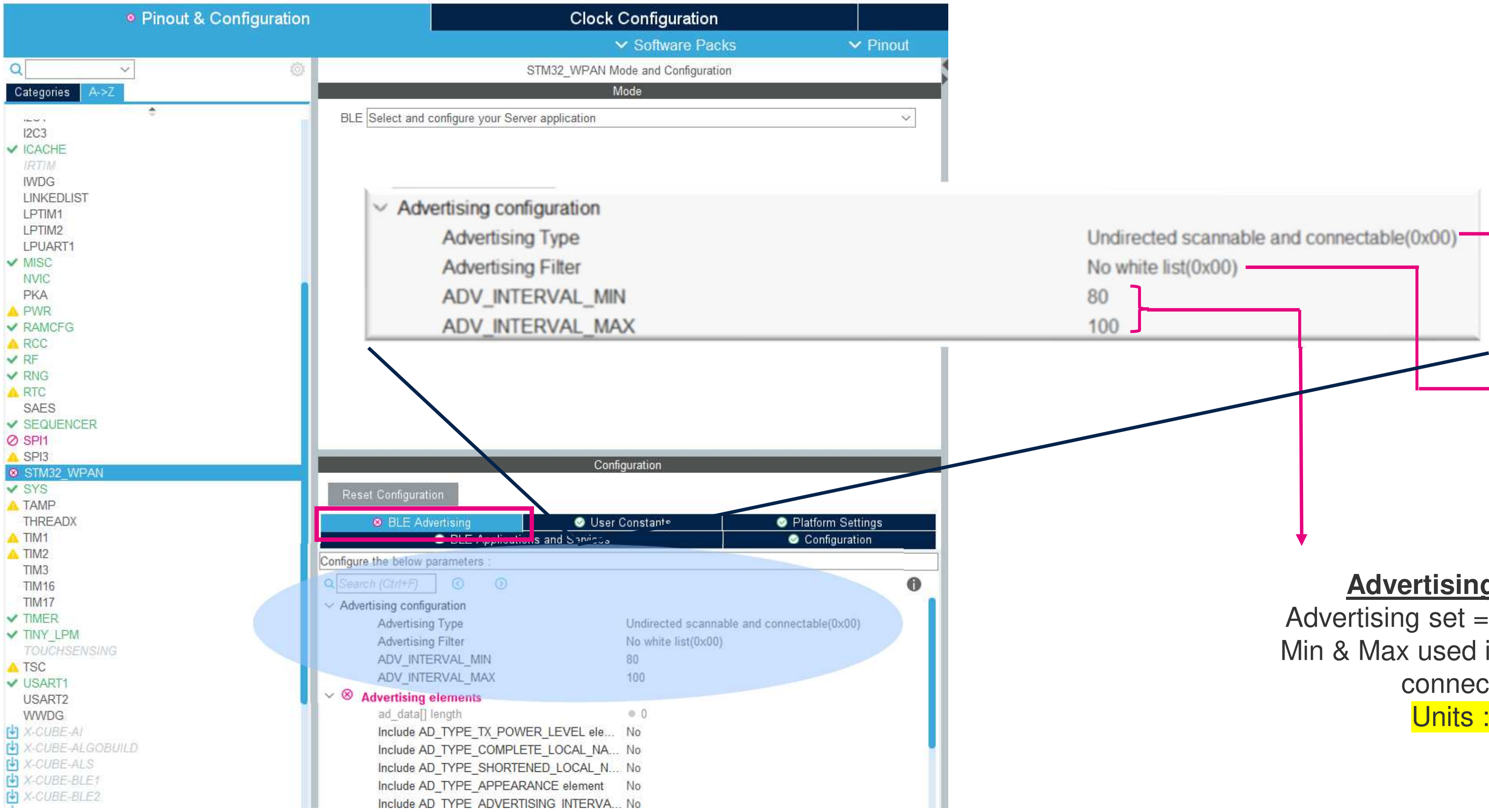
Bluetooth® Low Energy
Connection roles vs. Data roles





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

Advertising Configuration



BLE: Select and configure your Server application

Configure the below parameters :

Advertising configuration

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

Advertising Type
accept connection requests from any peer device

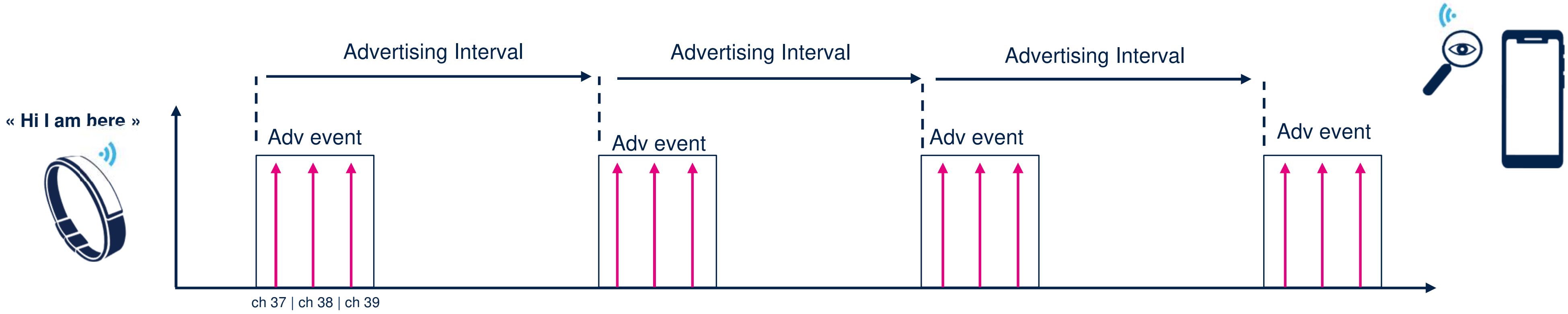
Advertising Filter
In general, used in case of Privacy.

Advertising Interval
Advertising set = (MIN+MAX)/2
Min & Max used in case of multi connections
Units : **ms**

For this session, let's keep default values at this stage

life.augmented

Advertising Configuration
Legacy Advertising Interval

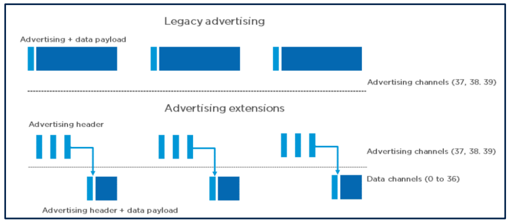


- The advertising interval value ranges all the way from **20** milliseconds up to **10.24** seconds in small increments of **625** microseconds.
- The advertising interval greatly impacts battery life and should be chosen carefully.

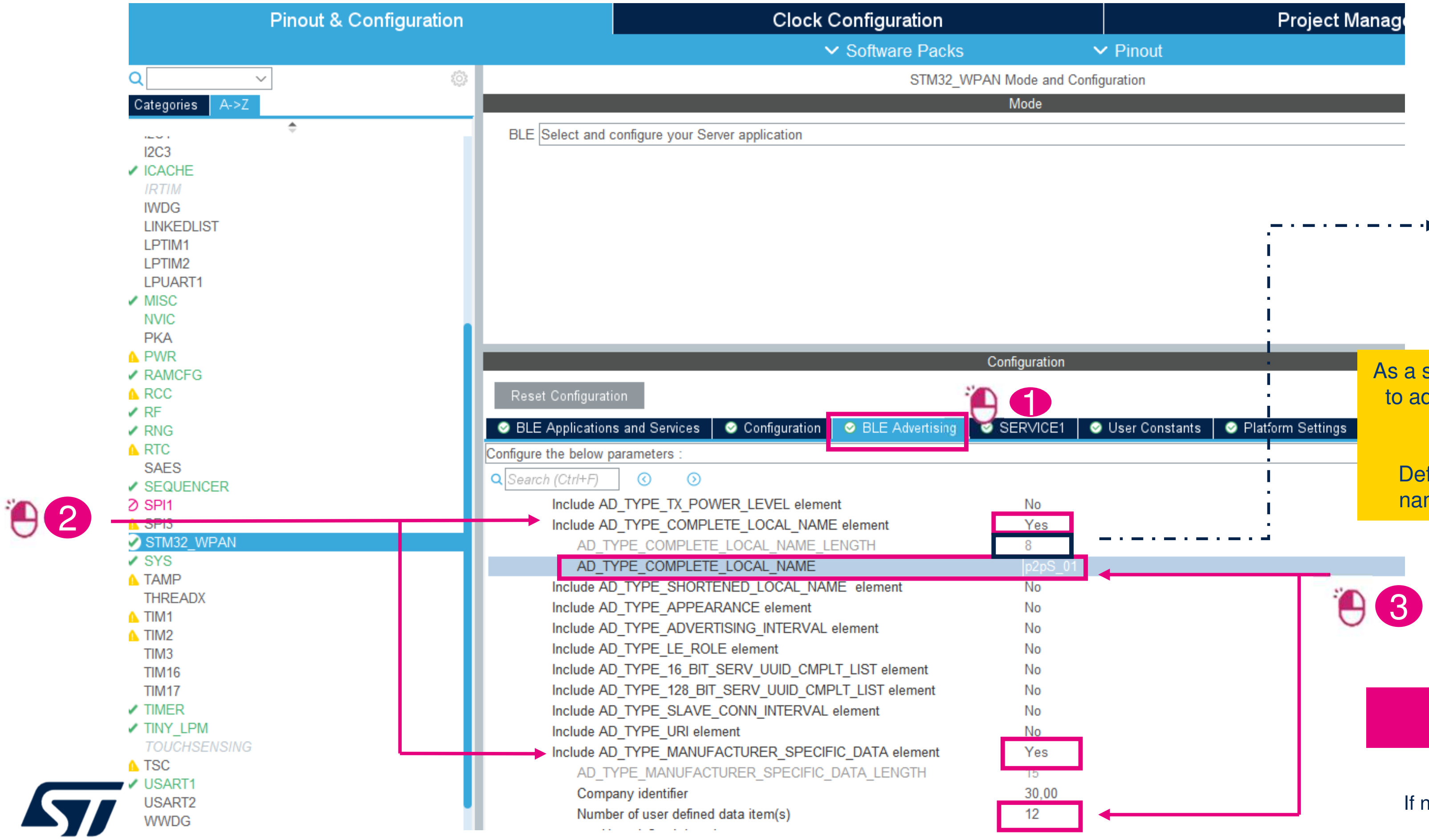
connectivity latency vs. power consumption efficiency

- 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)

WBA5x supporting advertising extension to increase your advertising data
Thanks to adv extension , Periodic advertising supported


Legacy advertising
Advertising + data payload
Advertising channels (37, 38. 39)
Advertising extensions
Advertising header
Advertising channels (37, 38. 39)
Data channels (0 to 36)
Advertising header + data payload

Advertising Elements
Local Name



- 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
- ✓ TIMER
- ✓ TINY_LPM
- TOUCHSENSING
- ⚠ TSC
- ✓ USART1
- USART2
- WWDG

STM32_WPAN Mode and Configuration

Mode

BLE: Select and configure your Server application

Configuration

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

Configure the below parameters :

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 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	15
Company identifier	30,00
Number of user defined data item(s)	12

Local Name length = Local name + 1

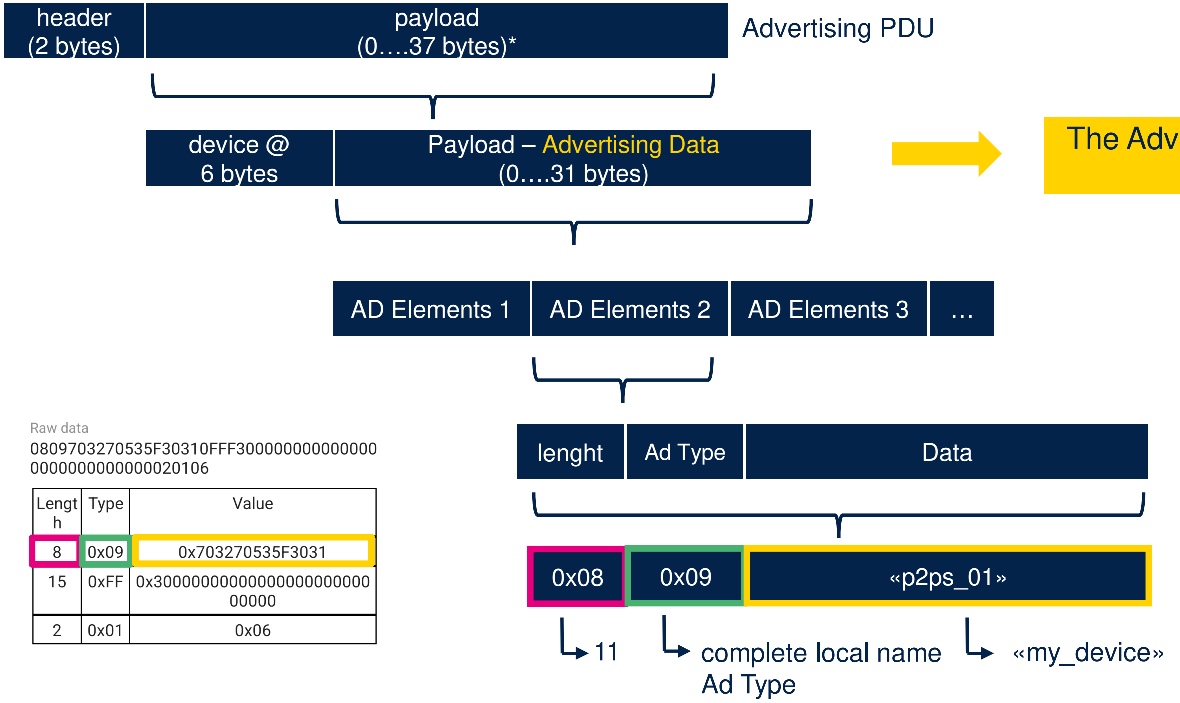
As a server, our application will have to advertise waiting for connection request from a client.

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

Local Name length must be < 11
CubeMx contraints

If not ST Toolbox potential crash

Advertising Elements
Advertising PDU



Raw data
0809703270535F30310FFF300000000000000
0000000000000020106

Length	Type	Value
8	0x09	0x703270535F3031
15	0xFF	0x3000000000000000000000000000
2	0x01	0x06

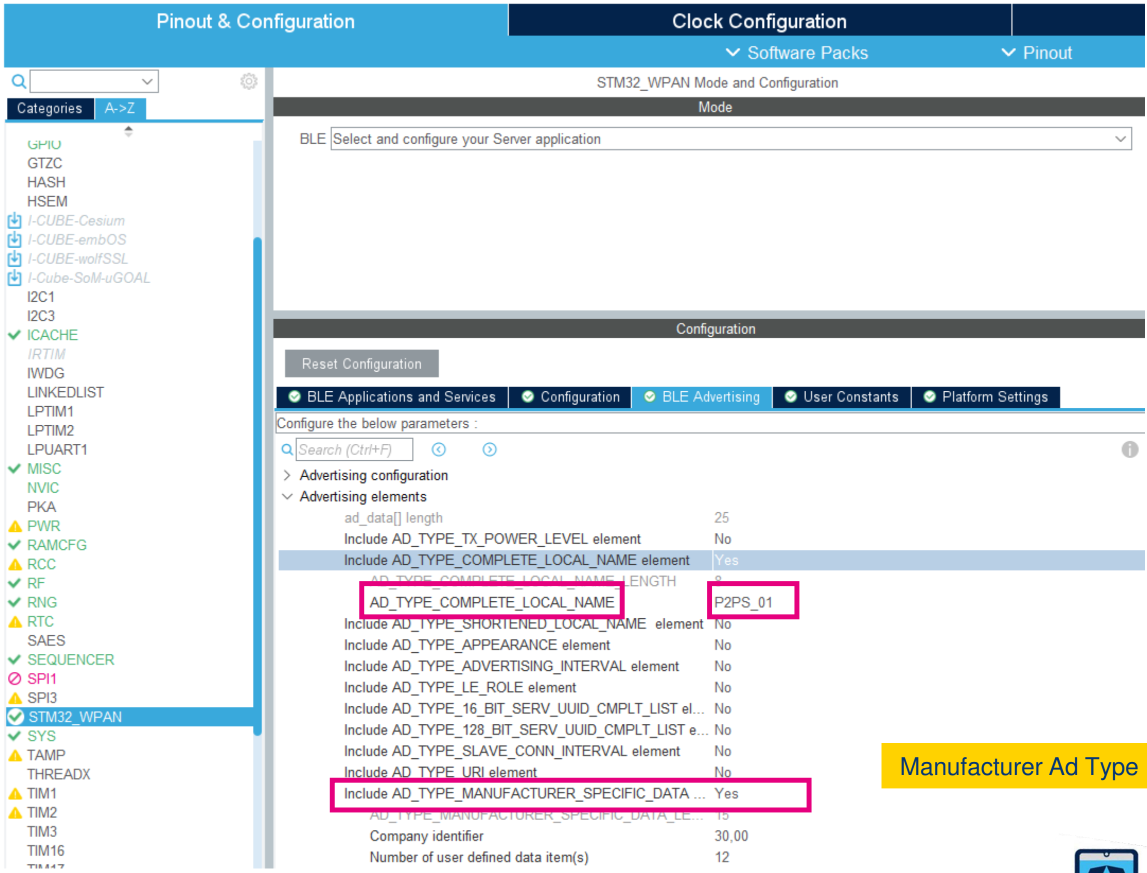
The Advertising Data consists of one or more Advertising Data elements
AD Element/Type are listed at Bluetooth SIG website

Most commonly used AD elements :
- 0x01 = Flags (**mandatory for connectable device**)
- 0x09 = Complete Local Name
- 0xFF = Manufacturer Data

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



STM32_WPAN Mode and Configuration

BLE: Select and configure your Server application

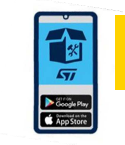
Configure the below parameters :

- 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 el...	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

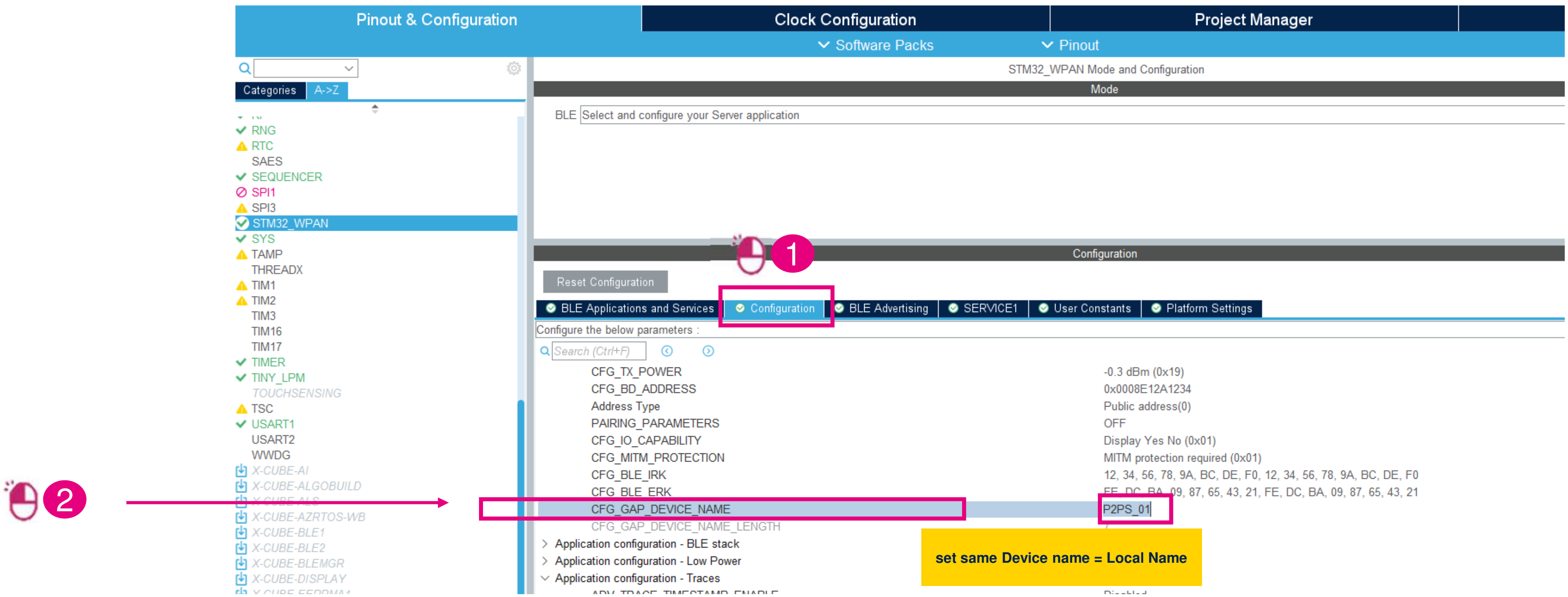
Manufacturer Ad Type , with company ID 0x30 (STMicroelectronics)

↓



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

Customize Device Name



BLE Select and configure your Server application

Configure the below parameters :

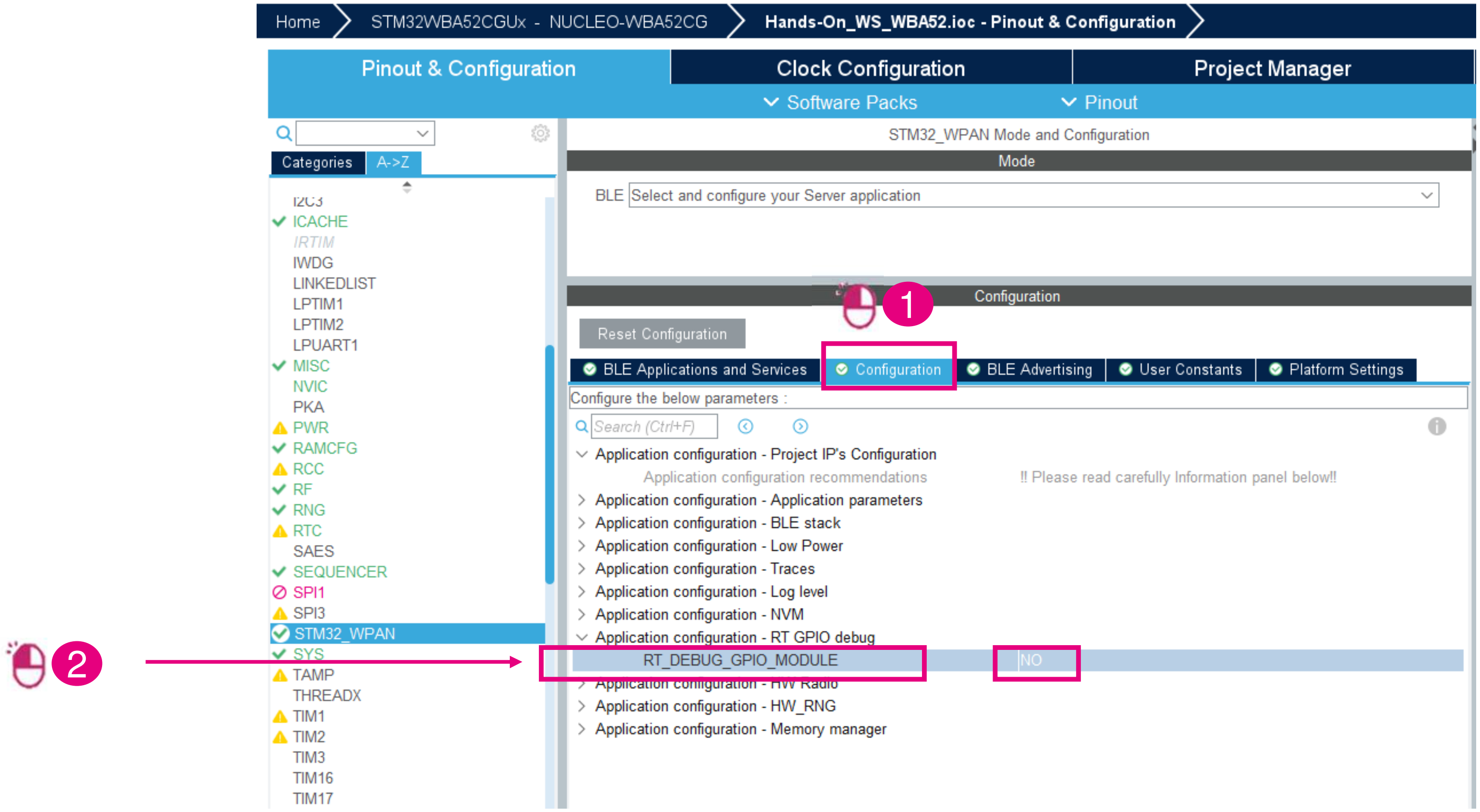
CFG_TX_POWER	-0.3 dBm (0x19)
CFG_BD_ADDRESS	0x0008E12A1234
Address Type	Public address(0)
PAIRING_PARAMETERS	OFF
CFG_IO_CAPABILITY	Display Yes No (0x01)
CFG_MITM_PROTECTION	MITM protection required (0x01)
CFG_BLE_IRK	12, 34, 56, 78, 9A, BC, DE, F0, 12, 34, 56, 78, 9A, BC, DE, F0
CFG_BLE_ERK	FE, DC, BA,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

set same Device name = Local Name

iOS displays Local Name (advertising data) prior to a 1st connexion.
After a 1st connexion iOS displays Device name (thanks to look up table : associates BLE MAC @ & Device Name)

Disable GPIO Debug capabilities



BLE Select and configure your Server application

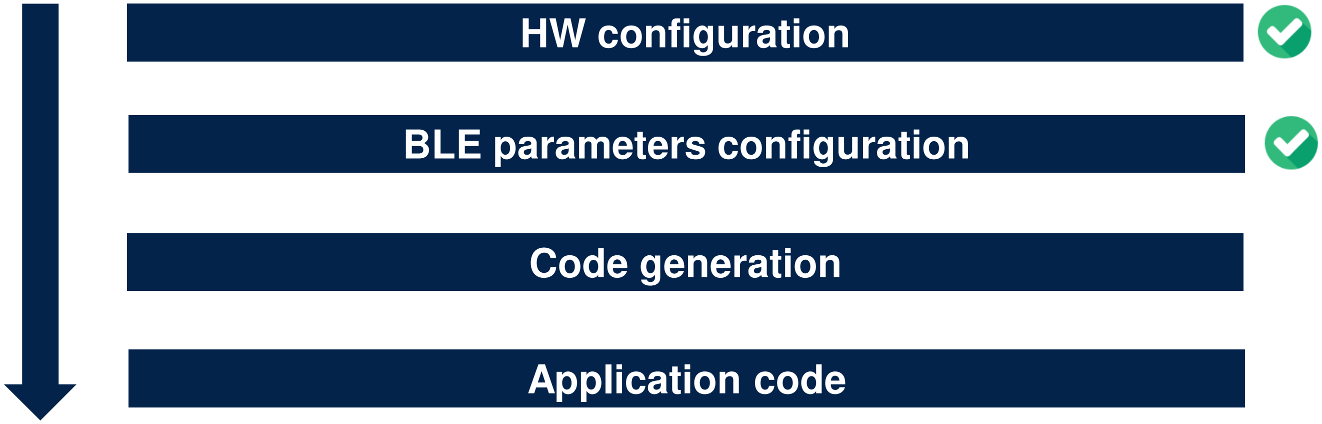
Configure the below parameters :

- Application configuration - Project IP's Configuration
  - Application configuration recommendations — !! Please read carefully Information panel below!!
- Application configuration - Application parameters
- Application configuration - BLE stack
- Application configuration - Low Power
- Application configuration - Traces
- Application configuration - Log level
- Application configuration - NVM
- Application configuration - RT GPIO debug
  - RT_DEBUG_GPIO_MODULE: NO
- Application configuration - HW Radio
- Application configuration - HW_RNG
- Application configuration - Memory manager

Why ?

Avoid adding sw line of code to enable debug capabilities in hands on code

Configuration completed
What’s next : code generation ?



HW configuration ✓

BLE parameters configuration ✓

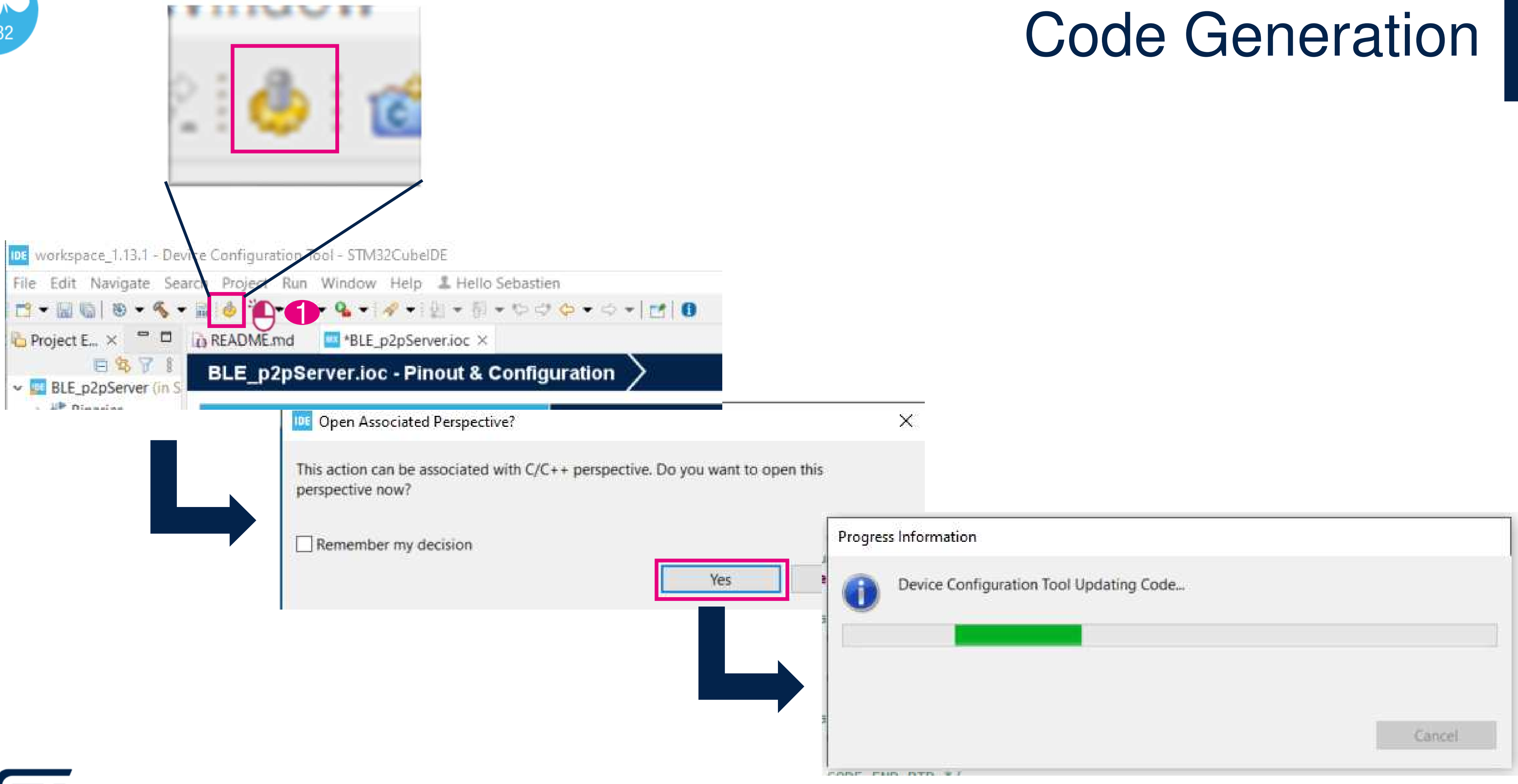
Code generation

Application code

Step 3 : Code generation and user application code

life.augmented

Code Generation



workspace_1.13.1 - Device Configuration Tool - STM32CubeIDE

Open Associated Perspective?

This action can be associated with C/C++ perspective. Do you want to open this perspective now?

□ Remember my decision

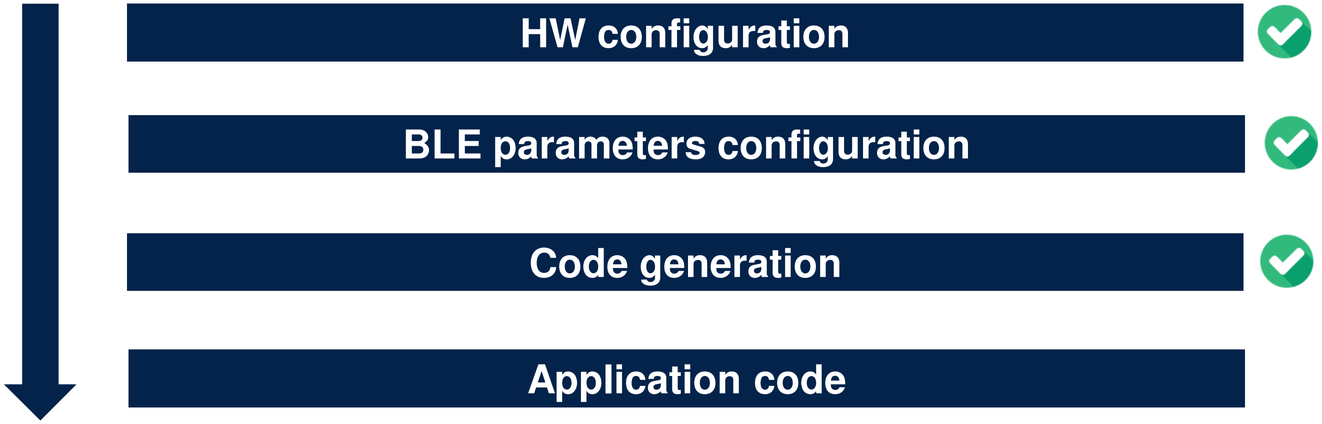
Yes

Progress Information

Device Configuration Tool Updating Code...

Cancel

Configuration completed
What’s next : code generation ?



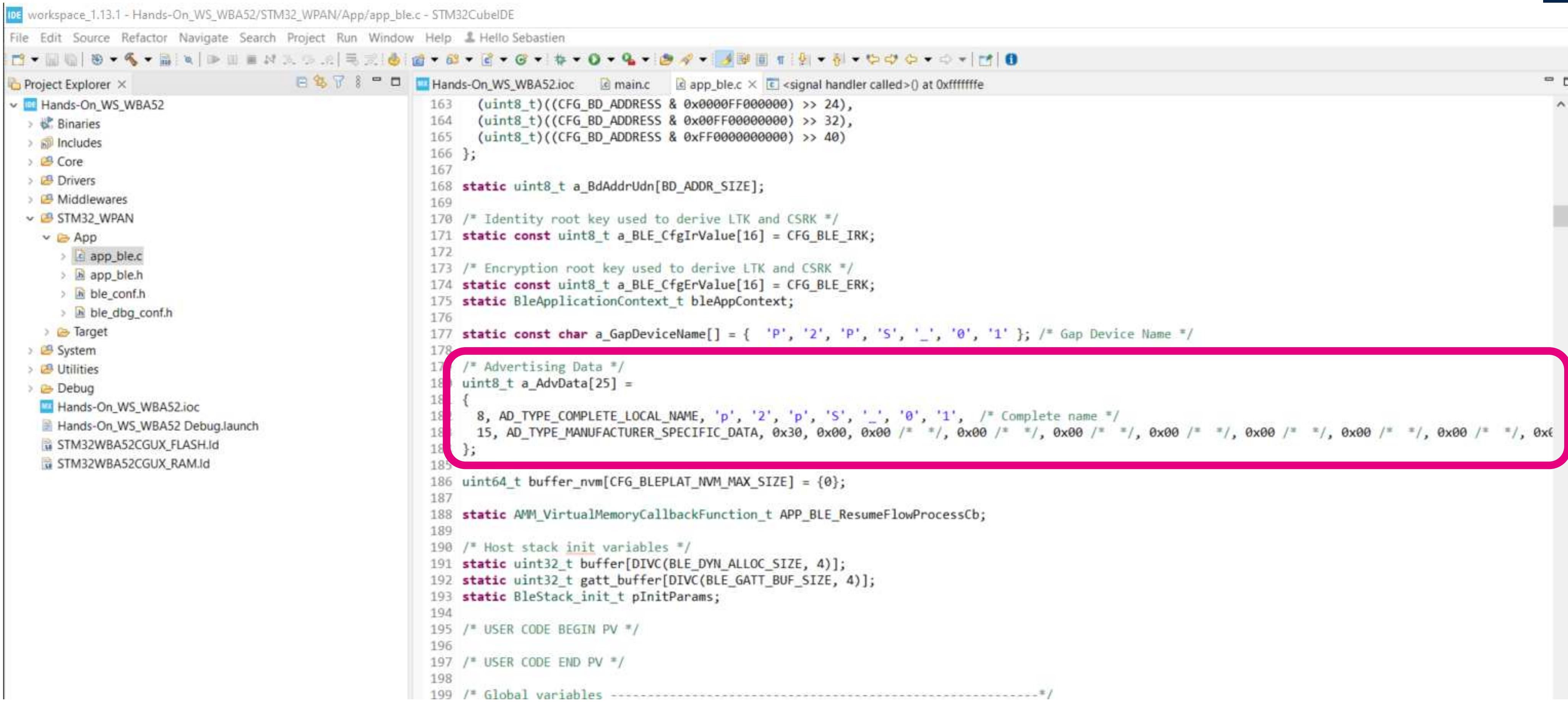
HW configuration ✓

BLE parameters configuration ✓

Code generation ✓

Application code

Here are our ADV data

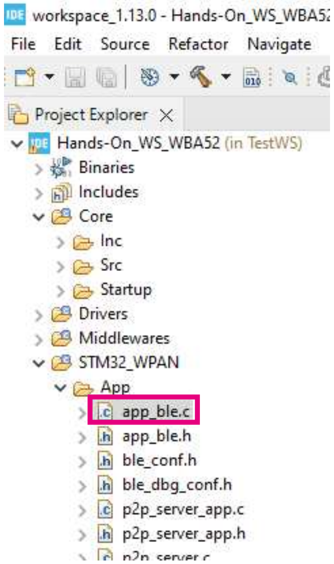


```c
  (uint8_t)((CFG_BD_ADDRESS & 0x0000FF000000) >> 24),
  (uint8_t)((CFG_BD_ADDRESS & 0x00FF00000000) >> 32),
  (uint8_t)((CFG_BD_ADDRESS & 0xFF0000000000) >> 40)
};

static uint8_t a_BdAddrUdn[BD_ADDR_SIZE];

/* Identity root key used to derive LTK and CSRK */
static const uint8_t a_BLE_CfgIrValue[16] = CFG_BLE_IRK;

/* Encryption root key used to derive LTK and CSRK */
static const uint8_t a_BLE_CfgErValue[16] = CFG_BLE_ERK;
static BleApplicationContext_t bleAppContext;

static const char a_GapDeviceName[] = {  'P', '2', 'P', 'S', '_', '0', '1' }; /* Gap Device Name */

/* Advertising Data */
uint8_t a_AdvData[25] =
{
  8, AD_TYPE_COMPLETE_LOCAL_NAME, 'p', '2', 'p', 'S', '_', '0', '1',  /* Complete name */
  15, AD_TYPE_MANUFACTURER_SPECIFIC_DATA, 0x30, 0x00, 0x00 /*  */, 0x00 /*  */, 0x00 /*  */, 0x00 /*  */, 0x00 /*  */, 0x00 /*  */, 0x00 /*  */, 0x
};

uint64_t buffer_nvm[CFG_BLEPLAT_NVM_MAX_SIZE] = {0};

static AMM_VirtualMemoryCallbackFunction_t APP_BLE_ResumeFlowProcessCb;

/* Host stack init variables */
static uint32_t buffer[DIVC(BLE_DYN_ALLOC_SIZE, 4)];
static uint32_t gatt_buffer[DIVC(BLE_GATT_BUF_SIZE, 4)];
static BleStack_init_t pInitParams;

/* USER CODE BEGIN PV */

/* USER CODE END PV */

/* Global variables ----------------------------------------------------------*/
```

life.augmented

Open Project
Add application code to move to discoverable (1/2)

Set device discoverable at init :
In app_ble.c > function APP_BLE_Init()



```
/* USER CODE BEGIN APP_BLE_Init_2 */
tBleStatus status;
 status = aci_gap_set_discoverable(ADV_TYPE, ADV_INTERVAL_MIN,ADV_INTERVAL_MAX,
                                   CFG_BD_ADDRESS_TYPE,
                                   ADV_FILTER,
                                   0, 0, 0, 0, 0, 0);
 if (status != BLE_STATUS_SUCCESS) {
           return;
 }

 status = aci_gap_delete_ad_type(AD_TYPE_TX_POWER_LEVEL);
 if (status != BLE_STATUS_SUCCESS) {
           return;
 }

 status = aci_gap_update_adv_data(sizeof(a_AdvData), (uint8_t*) a_AdvData);
 if (status != BLE_STATUS_SUCCESS) {
           return;
 }
/* USER CODE END APP_BLE_Init_2 */
```

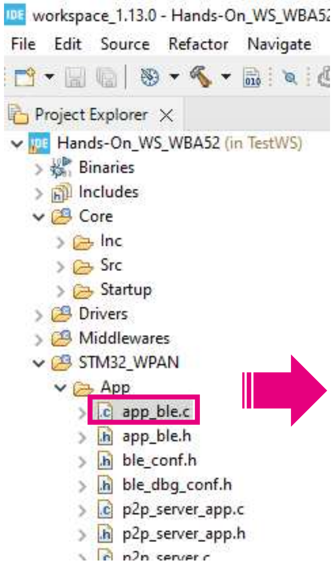
To accommodate the Advertising payload, remove the Tx power Adv Type set by stack

Search for “APP_BLE_Init_2”

Open Project > Add application code to move to discoverable > Build& Flash

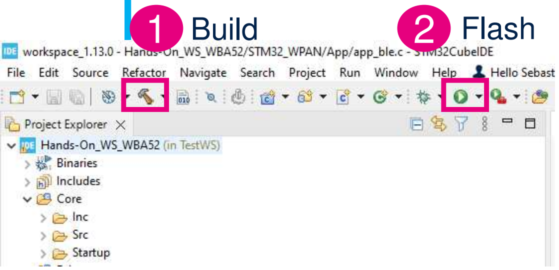
Please refer to cheatsheet for copy/paste

Open Project
Add application code to move to discoverable (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 */
    tBleStatus status;
    status = aci_gap_set_discoverable(ADV_TYPE, ADV_INTERVAL_MIN,ADV_INTERVAL_MAX,
                                      CFG_BD_ADDRESS_TYPE,
                                      ADV_FILTER,
                                      0, 0, 0, 0, 0, 0);
    if (status != BLE_STATUS_SUCCESS) {
            return;
    }
/* USER CODE END EVT_DISCONN_COMPLETE */
```



1 Build
2 Flash
workspace_1.13.0 - Hands-On_WS_WBA52/STM32_WPAN/App/app_ble.c - STM32CubeIDE
File Edit Source Refactor Navigate Search Project Run Window Help Hello Sebast
Project Explorer
Hands-On_WS_WBA52 (in TestWS)
Binaries
Includes
Core
Inc
Src
Startup

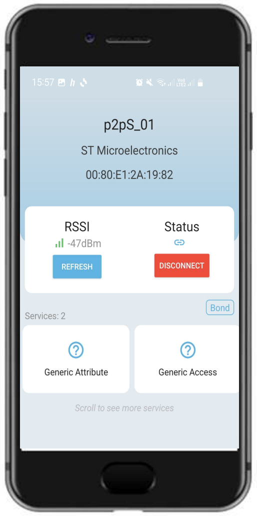
Search for “EVT_DISCONN_COMPLETE ”

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

Please refer to cheatsheet for copy/paste

Open your “ST BLE Toolbox” App and Connect



15:57

p2pS_01

ST Microelectronics

00:80:E1:2A:19:82

RSSI -47dBm REFRESH

Status DISCONNECT

Bond

Services: 2

Generic Attribute

Generic Access

Scroll to see more services

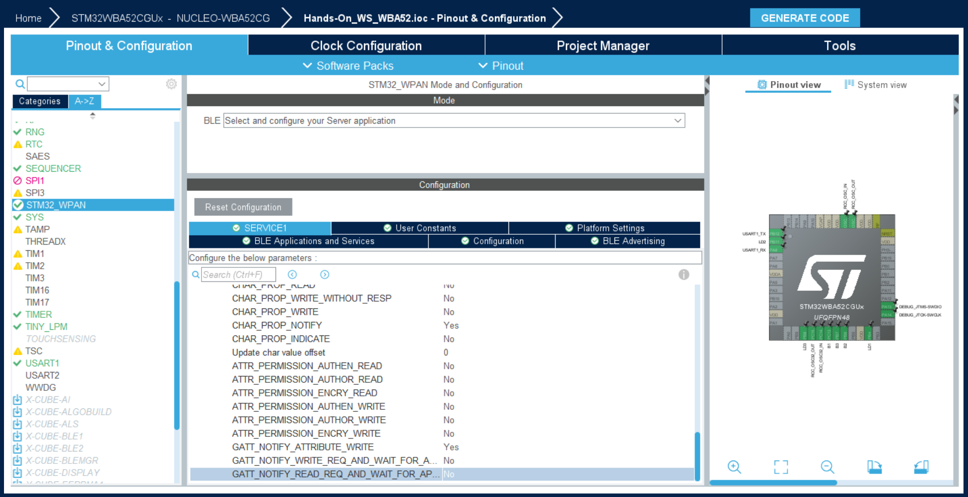
Download on the App Store

GET IT ON Google Play

Device should be visible and connectable

Bonus : Add debug capabilities

Move back to STM32CubeIDE/STM32CubeMX

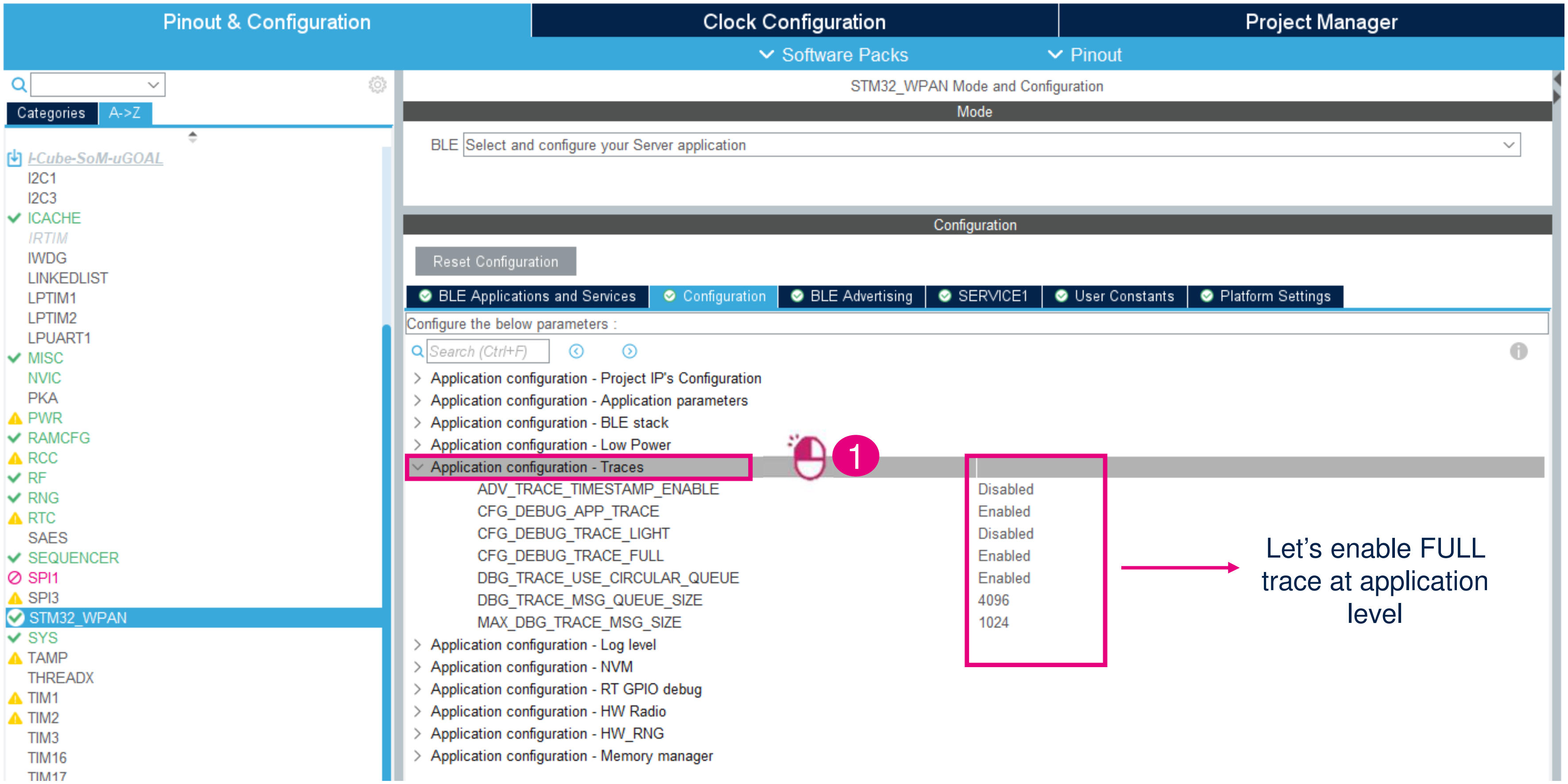


BLE Select and configure your Server application

Configure the below parameters :

CHAR_PROP_WRITE_WITHOUT_RESP	No
CHAR_PROP_WRITE	No
CHAR_PROP_NOTIFY	Yes
CHAR_PROP_INDICATE	No
Update char value offset	0
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_A...	No
GATT_NOTIFY_READ_REQ_AND_WAIT_FOR_AP...	No

Application configuration
Traces & logs



BLE Select and configure your Server application

Configure the below parameters :

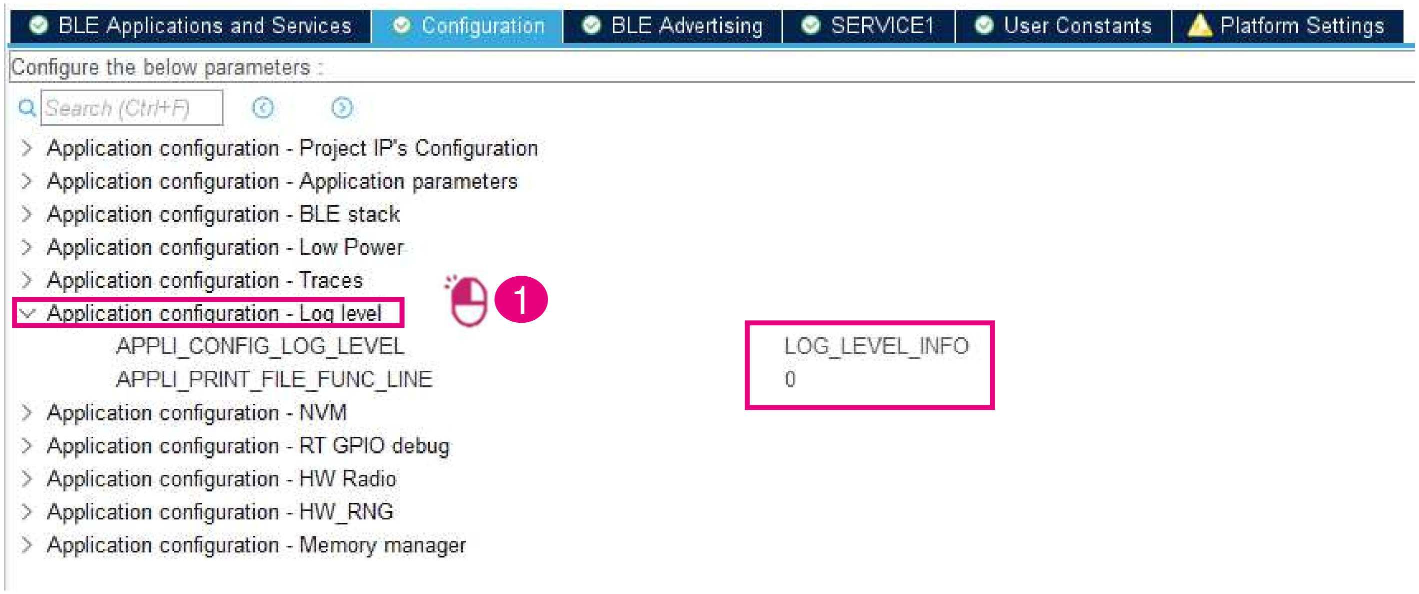
- Application configuration - Project IP's Configuration
- Application configuration - Application parameters
- Application configuration - BLE stack
- Application configuration - Low Power
- Application configuration - Traces

ADV_TRACE_TIMESTAMP_ENABLE	Disabled
CFG_DEBUG_APP_TRACE	Enabled
CFG_DEBUG_TRACE_LIGHT	Disabled
CFG_DEBUG_TRACE_FULL	Enabled
DBG_TRACE_USE_CIRCULAR_QUEUE	Enabled
DBG_TRACE_MSG_QUEUE_SIZE	4096
MAX_DBG_TRACE_MSG_SIZE	1024

- Application configuration - Log level
- Application configuration - NVM
- Application configuration - RT GPIO debug
- Application configuration - HW Radio
- Application configuration - HW_RNG
- Application configuration - Memory manager

Let’s enable FULL trace at application level

Application configuration
Trace & Logs: configure log level



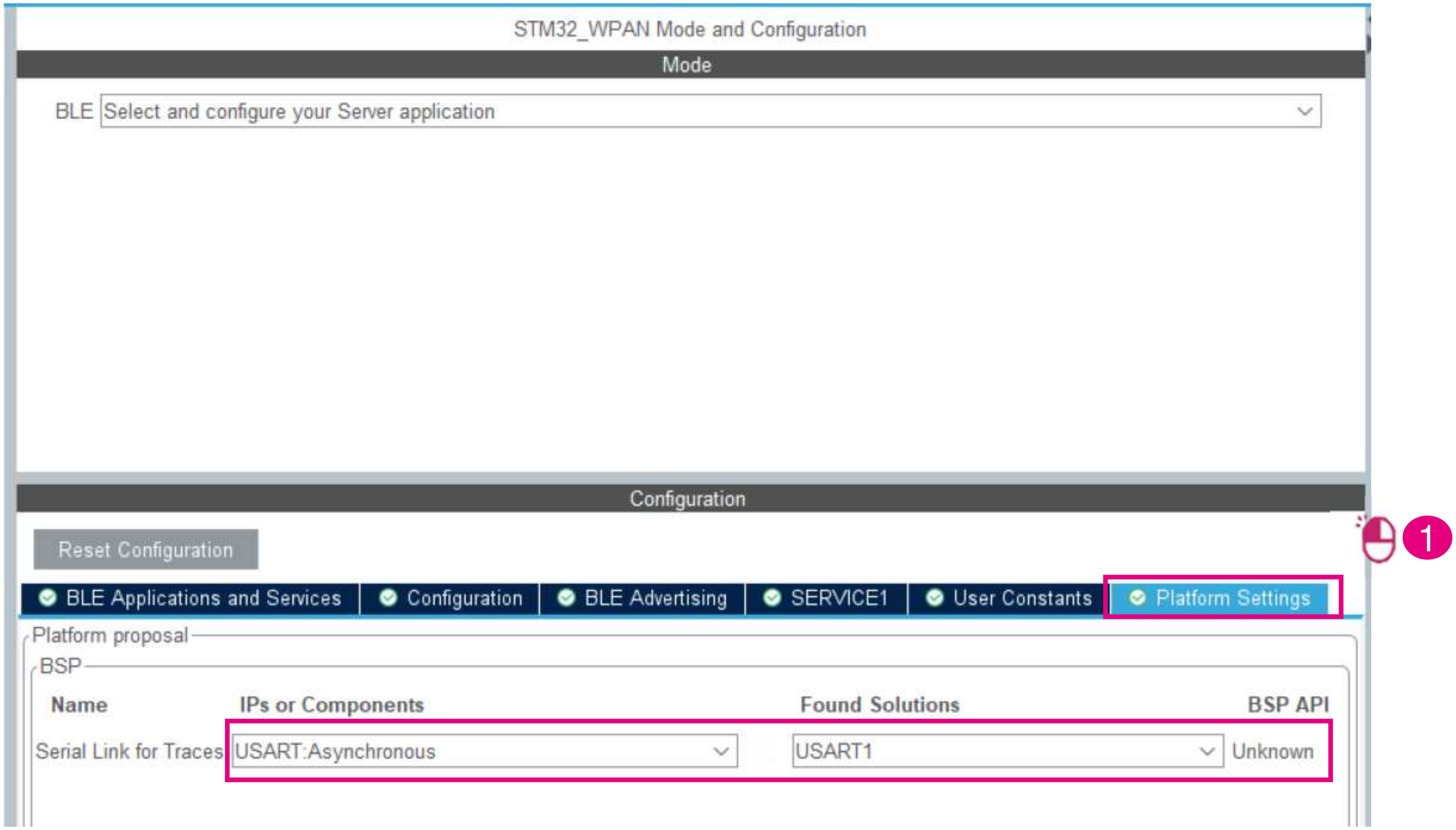
Configure the below parameters :

- 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

APPLI_CONFIG_LOG_LEVEL	LOG_LEVEL_INFO
APPLI_PRINT_FILE_FUNC_LINE	0

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

Platform Settings
Trace & Logs: BSP settings



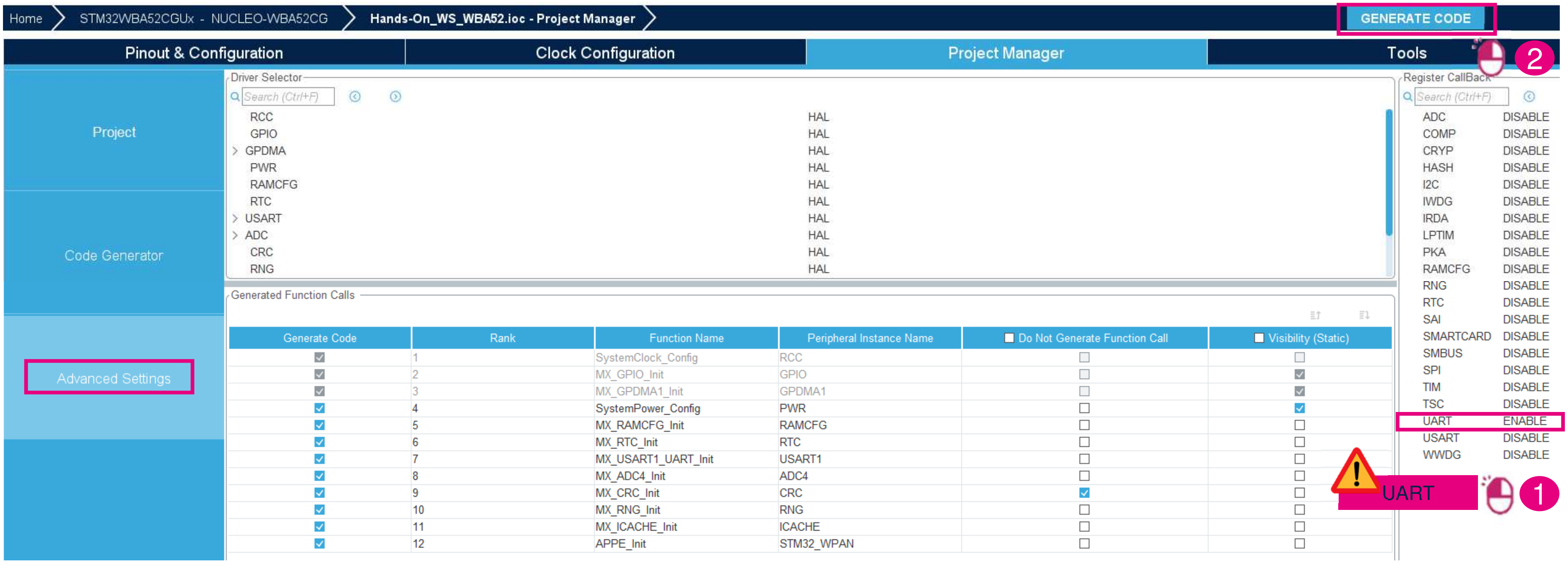
BLE: Select and configure your Server application

Platform proposal

BSP

Name	IPs or Components	Found Solutions	BSP API
Serial Link for Traces	USART:Asynchronous	USART1	Unknown

Project configuration Advanced settings



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

Pinout & Configuration | Clock Configuration | Project Manager | Tools

Driver Selector

Search (Ctrl+F)

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

Generated Function Calls

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

Register CallBack

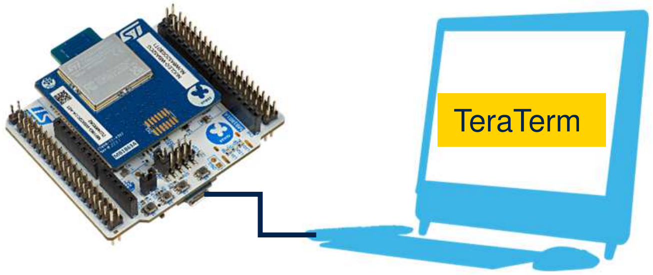
Search (Ctrl+F)

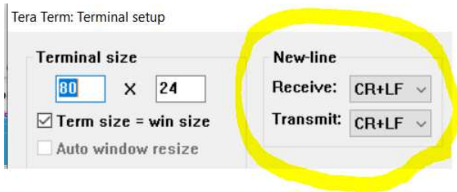
ADC	DISABLE
COMP	DISABLE
CRYP	DISABLE
HASH	DISABLE
I2C	DISABLE
IWDG	DISABLE
IRDA	DISABLE
LPTIM	DISABLE
PKA	DISABLE
RAMCFG	DISABLE
RNG	DISABLE
RTC	DISABLE
SAI	DISABLE
SMARTCARD	DISABLE
SMBUS	DISABLE
SPI	DISABLE
TIM	DISABLE
TSC	DISABLE
UART	ENABLE
USART	DISABLE
WWDG	DISABLE

⚠ UART

Regenerate Code
Open back existing project – refresh sources – build and flash

Open your App and Connect





Tera Term: Terminal setup

Terminal size: 80 X 24

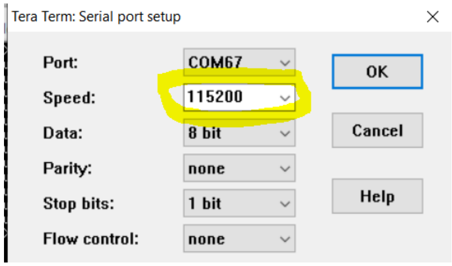
☑ Term size = win size

□ Auto window resize

New-line

Receive: CR+LF

Transmit: CR+LF



Tera Term: Serial port setup

Port: COM67

Speed: 115200

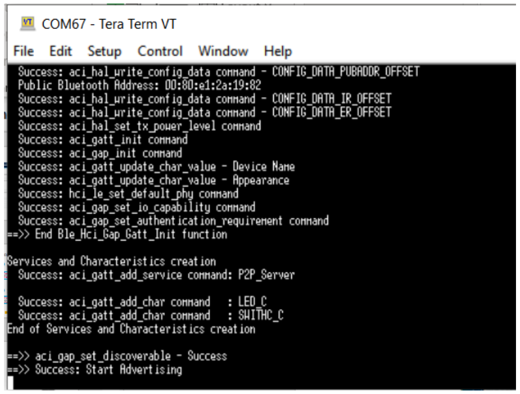
Data: 8 bit

Parity: none

Stop bits: 1 bit

Flow control: none

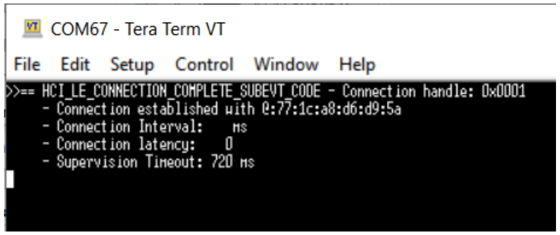
1 reset device



COM67 - Tera Term VT

```
  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
```

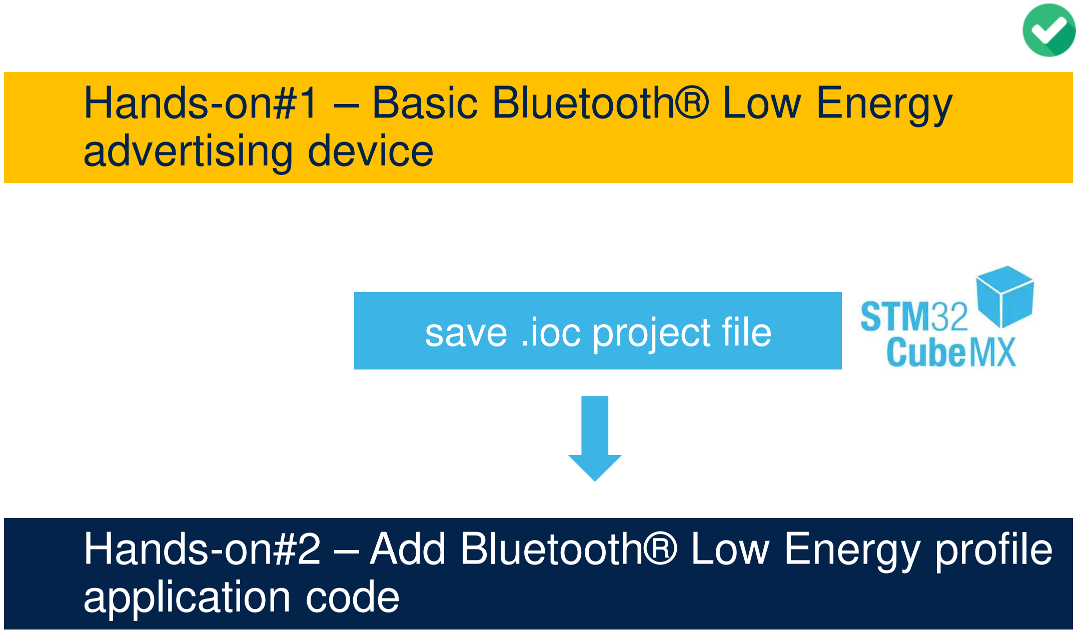
2 Connect



COM67 - Tera Term VT

```
>>== HCI_LE_CONNECTION_COMPLETE_SUBEVT_CODE - Connection handle: 0x0001
     - Connection established with @:77:1c:a8:d6:d9:5a
     - Connection Interval:    ms
     - Connection latency:    0
     - Supervision Timeout: 720 ms
```

Takeaways
What’s next



✓

Hands-on#1 – Basic Bluetooth® Low Energy advertising device

Inherit of STM32 ecosystem and build a Bluetooth® Low Energy advertising device application in few steps

save .ioc project file

Hands-on#2 – Add Bluetooth® Low Energy profile application code

Extend existing application code to enable proprietary profile (P2P_Server)

Thank you

life.augmented