BlueNRG-1

BLE SOC

AMS

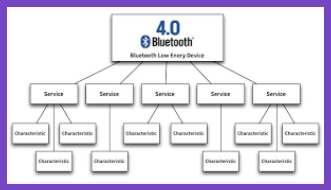
Application team EMEA

life.augmented

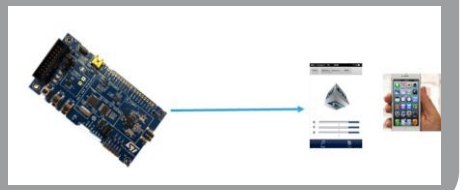
BlueNRG-1 BLE SOC

BlueNRG-1 and associated DK
(& promotion) package presentation

BLE concept demystification

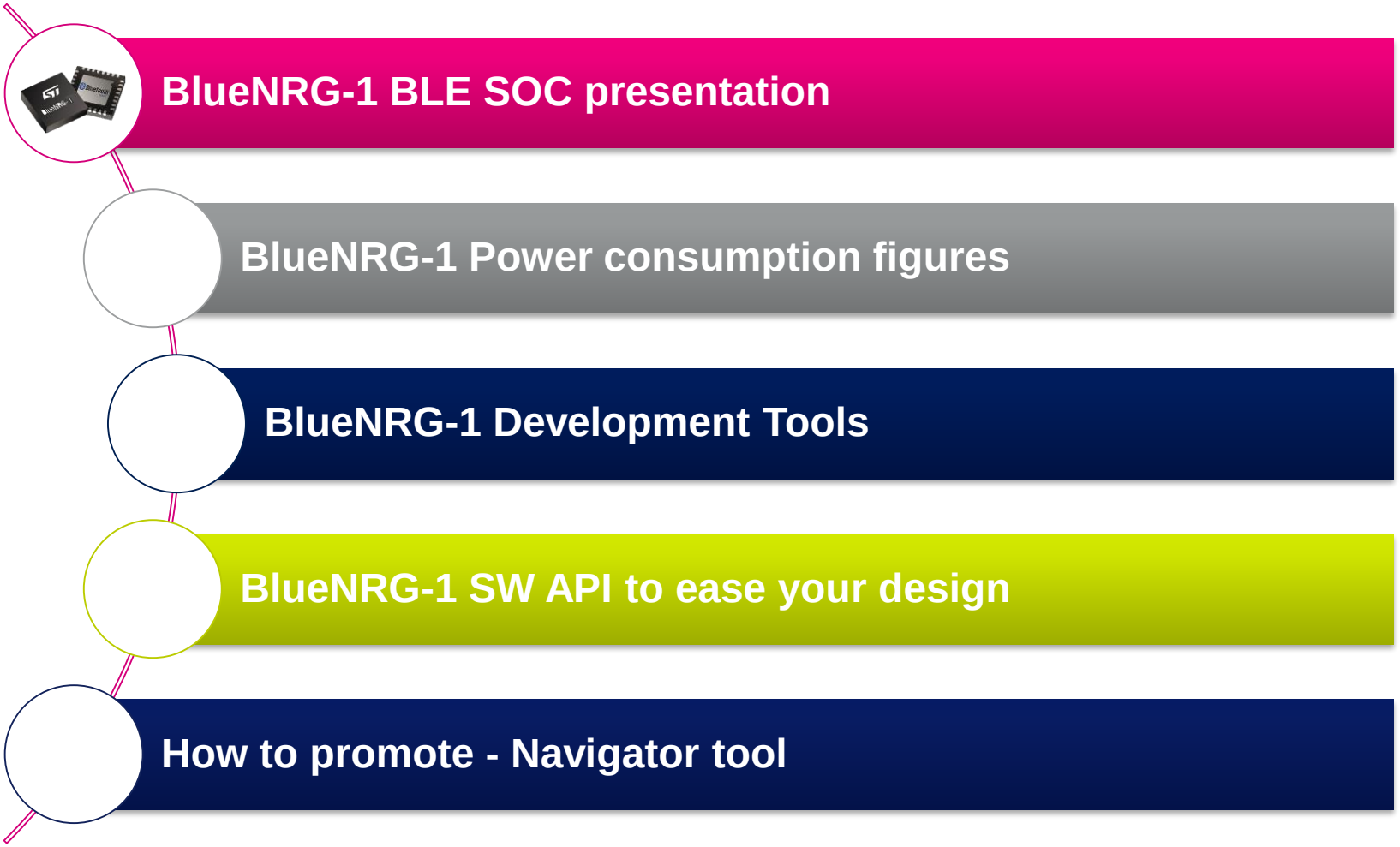


Enable BLE link over BlueNRG-1



life.augmented

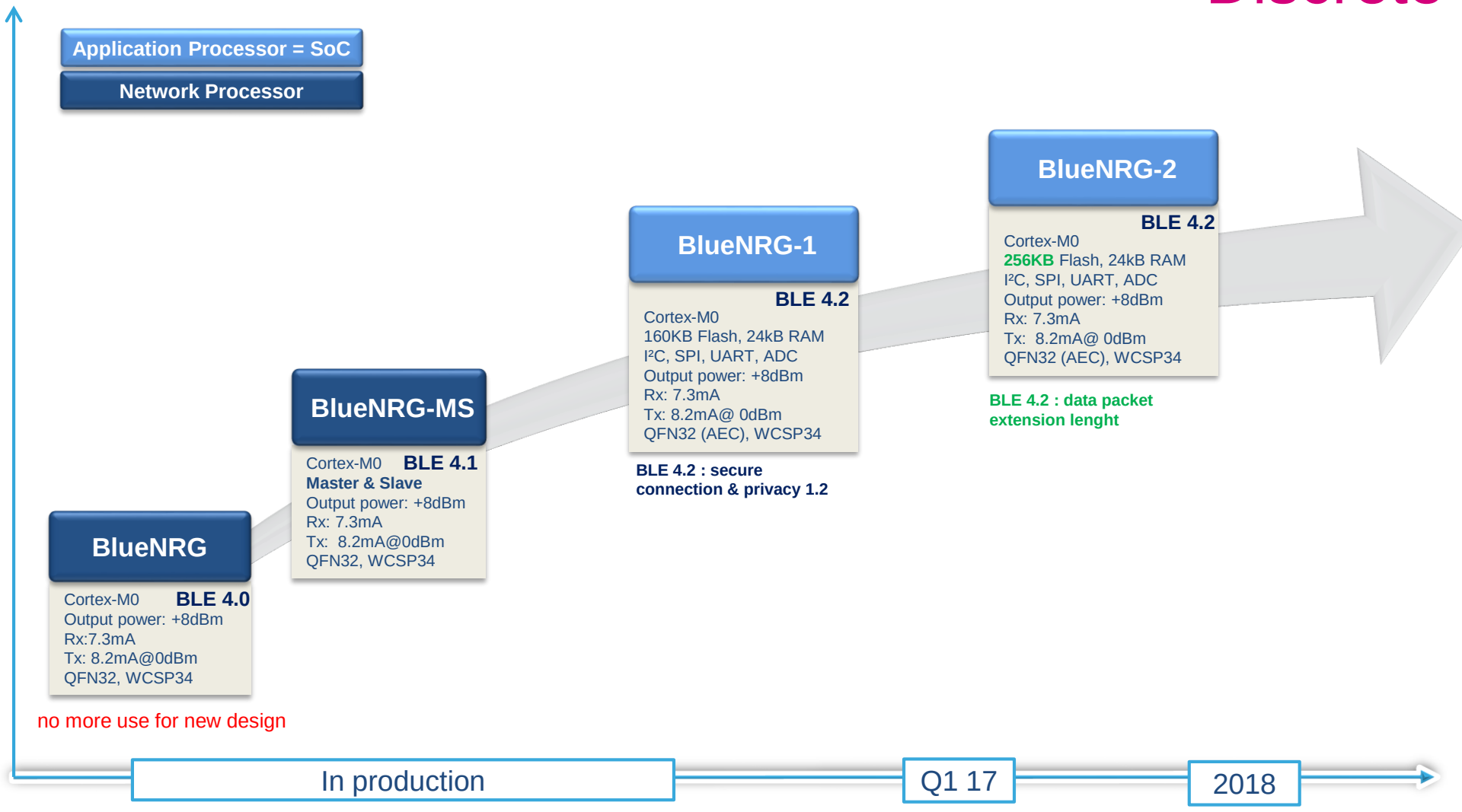
BlueNRG-1
BLE SOC presentation



life.augmented

Bluetooth® SMART offering Roadmap

Discrete



life.augmented

Bluetooth® SMART offering Roadmap

Module

Application Processor = SoC

Network Processor

BlueNRG

Cortex-M0 **BLE 4.0**
Output power: +8dBm
Rx:7.3mA
Tx: 8.2mA@0dBm
QFN32, WCSP34

no more use for new design

BlueNRG-MS

Cortex-M0 **BLE 4.1**
Master & Slave
Output power: +8dBm
Rx: 7.3mA
Tx: 8.2mA@0dBm
QFN32, WCSP34

BlueNRG-1

BLE 4.2
Cortex-M0
160KB Flash, 24kB RAM
I²C, SPI, UART, ADC
Output power: +8dBm
Rx: 7.3mA
Tx: 8.2mA@ 0dBm
QFN32 (AEC), WCSP34

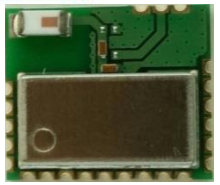
BLE 4.2 : secure connection & privacy 1.2

In production

Q2 17

11.5 mm x 13.5 mm

SPBTLE-RF



11.5 mm x 13.5 m

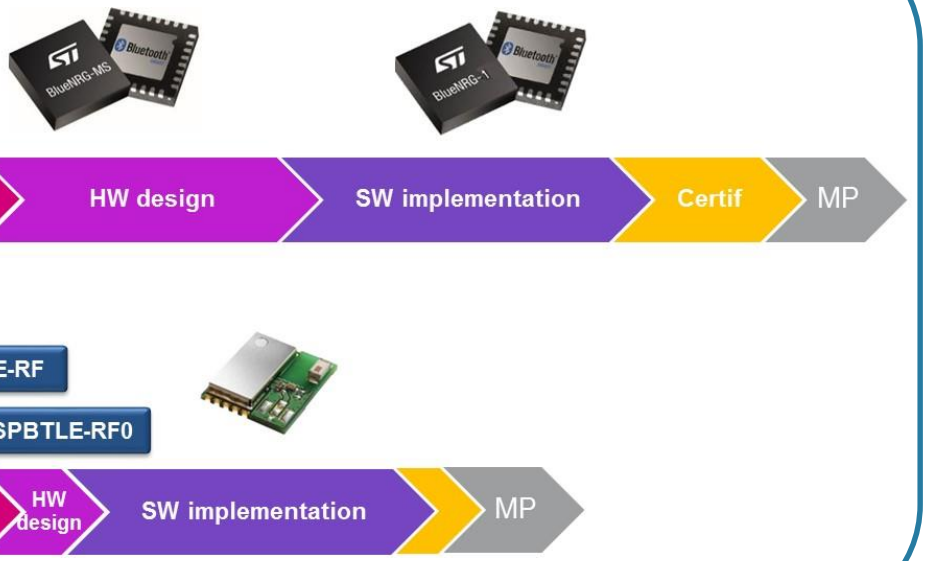
SPBTLE-1S

life.augmented

SPBTLE modules
BLE 4.1 RF Module

FCC, CE, IC BLE certified

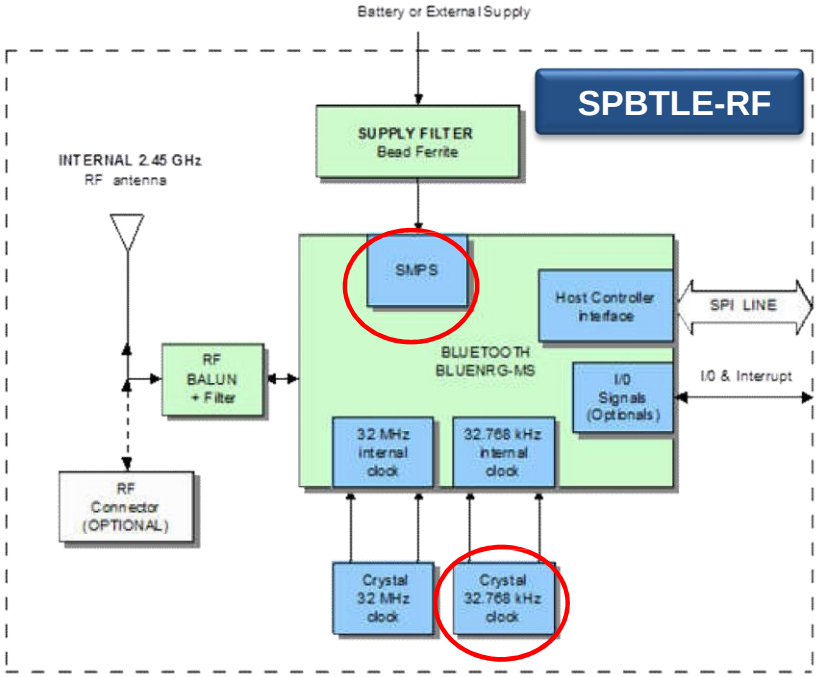
Modules designed for time to market

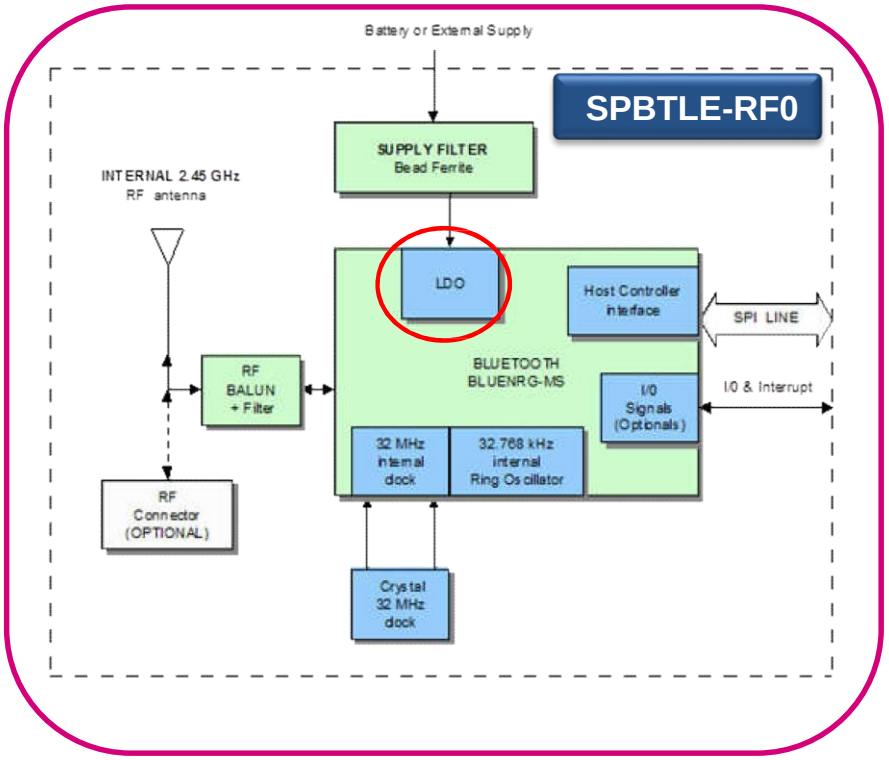


life.augmented

SPBTLE-RF0
Low cost release of SPBTLE-RF

SPBTLE-RF0 – Sub 3$ ST Bluetooth Smart module



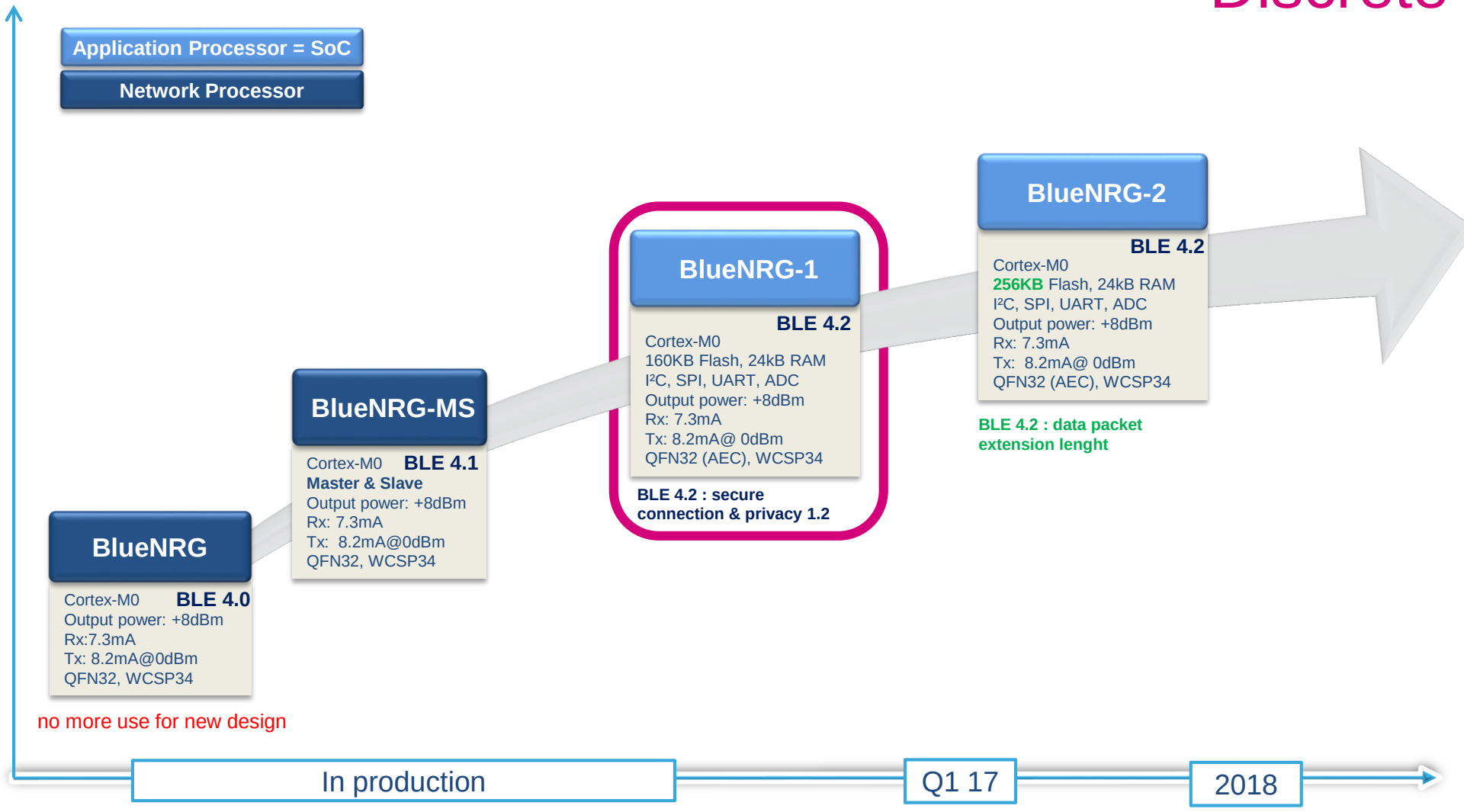


SPBTLE-RF0 Pin to Pin compatible SPBTLE-RF

life.augmented

Bluetooth® SMART offering Roadmap

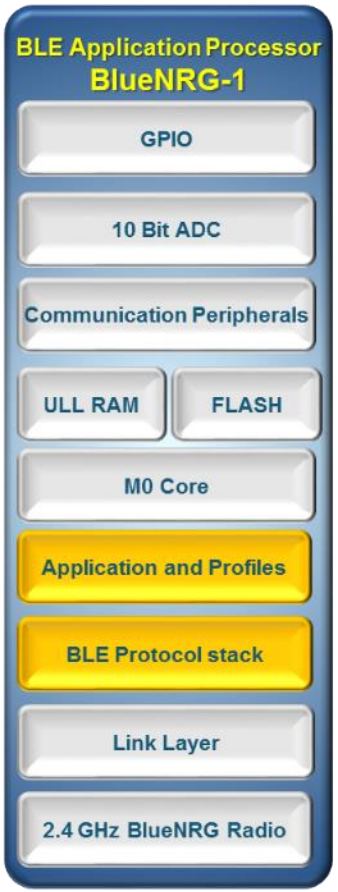
Discrete



life.augmented

BlueNRG-1 Application Processor

Highlights



Application and Profiles

- Application and BlueNRG profiles run in BlueNRG-1

BLE stack

Q2 2017

- 4.2 Bluetooth Low Energy compliancy
  - Privacy 1.2
  - Secure connection

Peripherals Drivers

- STM32 “like” drivers

life.augmented

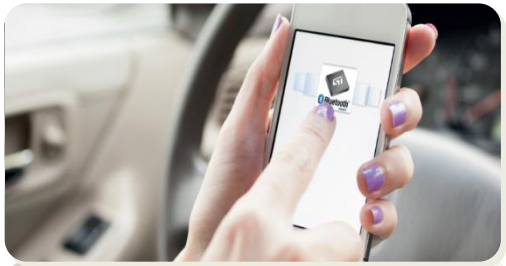
BlueNRG-1 Application Processor
Applications

BLE 4.2
Application Processor

- GPIO
- 10 Bit ADC
- Communication Peripherals
- ULL RAM
- Flash
- M0 Core
- Application and Profiles
- BLE Protocol stack
- Link Layer
- 2.4 GHz Radio

BlueNRG-1 capabilities to enable low to mid end smart connected applications

Beacon
Sensor tags
Remote Control



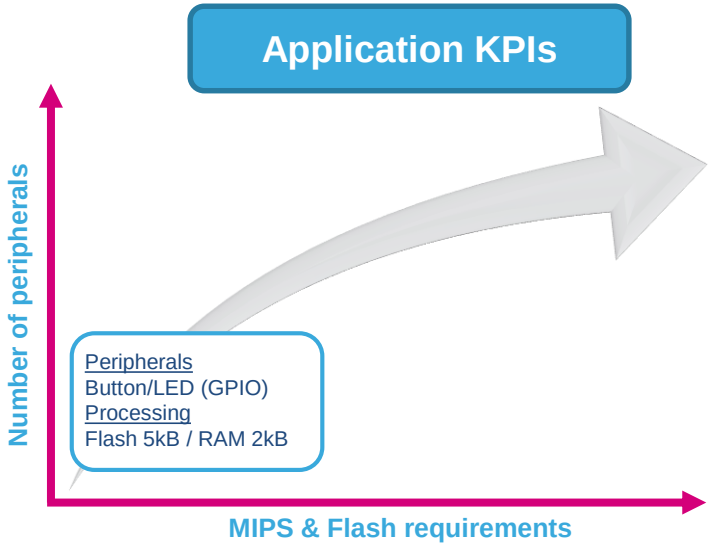
Automotive grade

life.augmented

BlueNRG-1 Application Processor
Applications

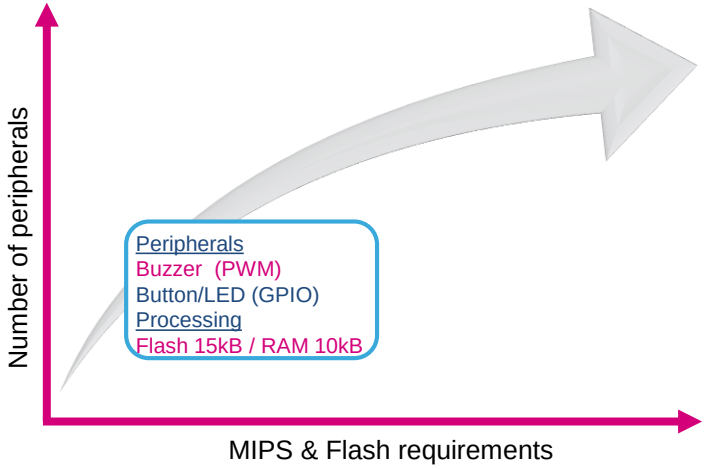
Beacon

- broadcasting data



Key Fob

- connected basic application
- localization

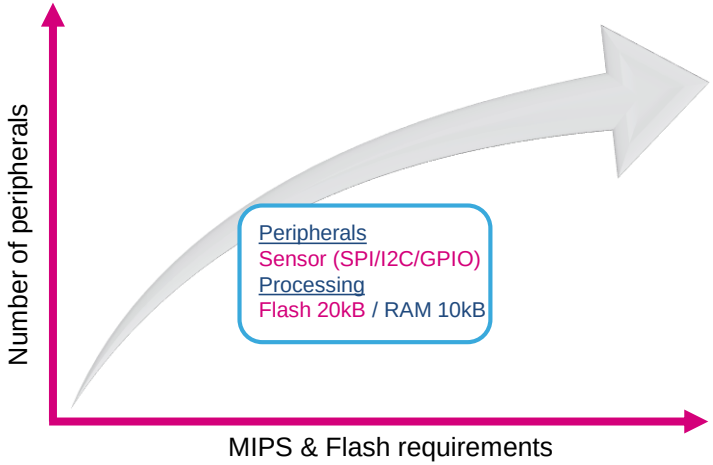


life.augmented

BlueNRG-1 Application Processor Applications

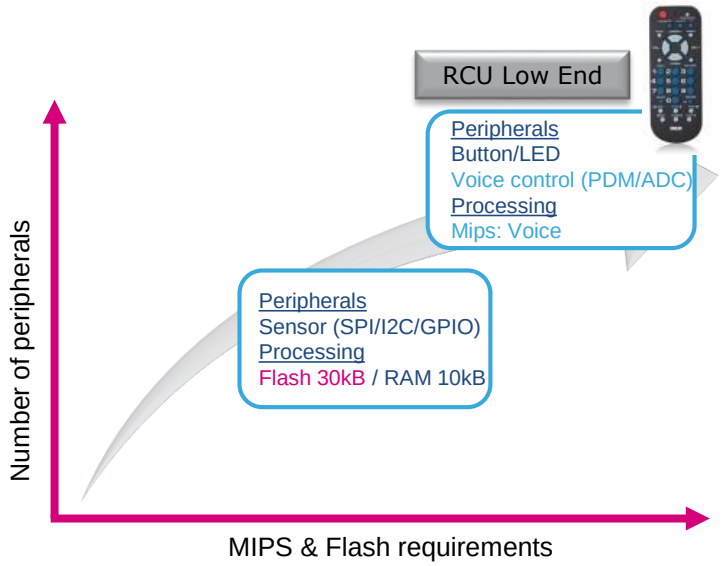
Sensor Tag

- Collect Sensor data



Appliance Remote Control

- Control remote device
- Device Configuration
- Device Application update



life.augmented

BlueNRG-1
Main Figures

BLE Application Processor
BlueNRG-1

- GPIO
- 10 Bit ADC
- Communication Peripherals
- ULL RAM
- Flash
- M0 Core
- Application and Profiles
- BLE Protocol stack
- Link Layer
- 2.4 GHz BlueNRG Radio

Superior Battery life (DCDC)
- RX 7.3mA
- TX 8.2mA @0dBm
- **Sleep 1µA**

Excellent RF perfs
- Best in Class Output Power **Level: +8dBm**
- Receiver sensitivity -88dBm

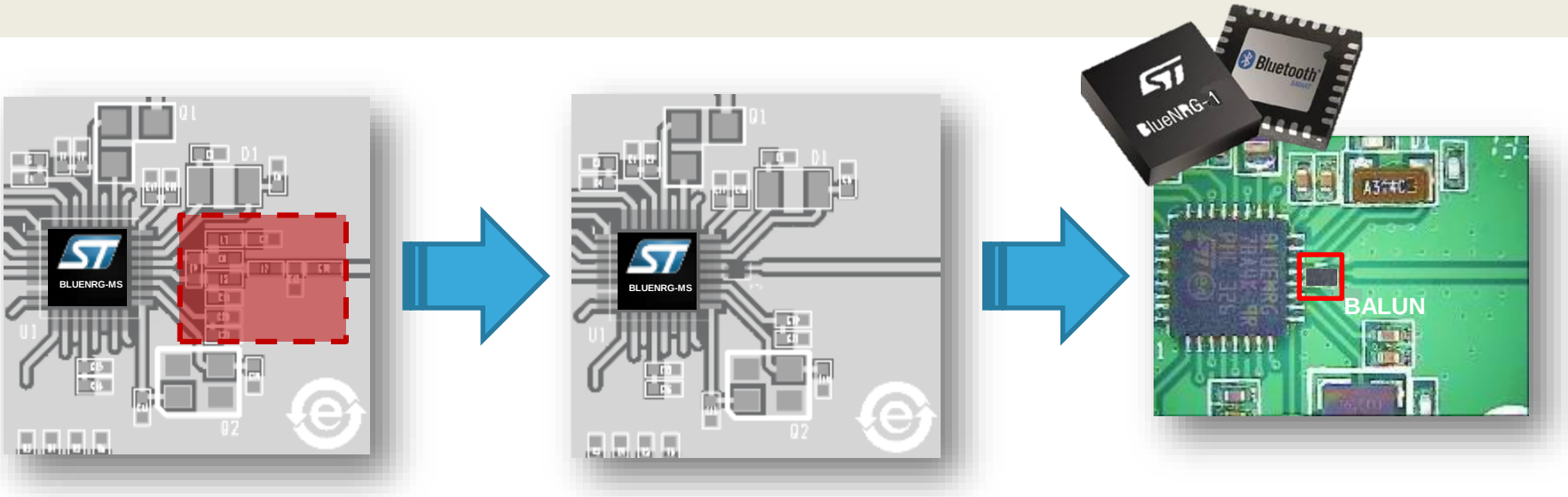
Two Package flavors
- Extended Temperature Range: up to 105^{0}C
- **WCSP34 2.65x2.65mm**
- **QFN32 5x5 mm (Automotive)**

life.augmented

BlueNRG family optimized footprint

RF balun and filtering

- The **BALF-NRG-01D3** is an ultra-miniature balun integrating a matching network and harmonics filter. The matching impedance is customized for ST's **BlueNRG** , **BlueNRG-MS** , **BlueNRG-1** transceiver (both QFN and WLCSP versions). It uses ST's IPD technology on a non-conductive glass substrate which optimizes RF performance.
- PACKAGE: flip-chip package 4 bumps, 1.2 mm² footprint.

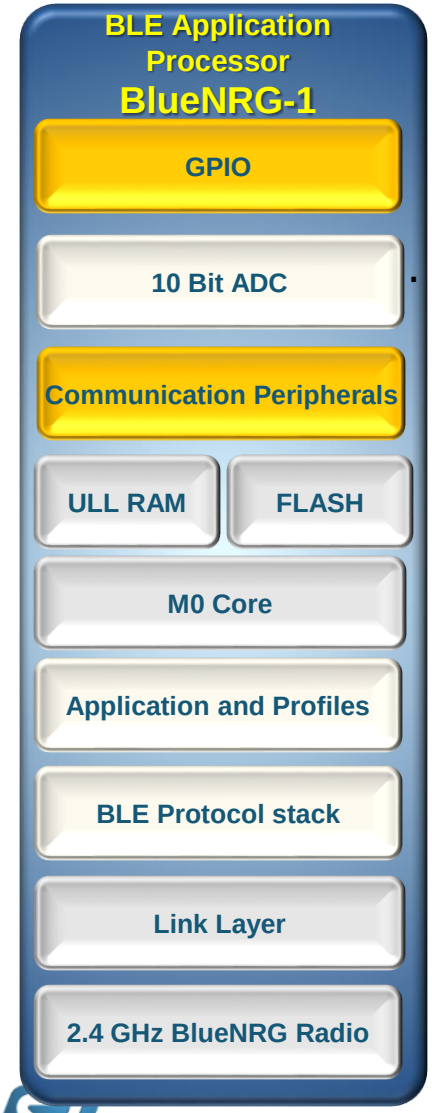


Footprint and **Cost** optimization

- From **9** to **1** SMD
- PCB real-estate savings: from **32mm²** to **1.2mm²**
- Optimized RF tuning antenna matching
- Simplified PCB layout and lower manufacturing costs

life.augmented

BlueNRG-1 Peripherals



GPIO

- 15 pin (QFN package)
- 14 pin (CSP package)
- Wake up function

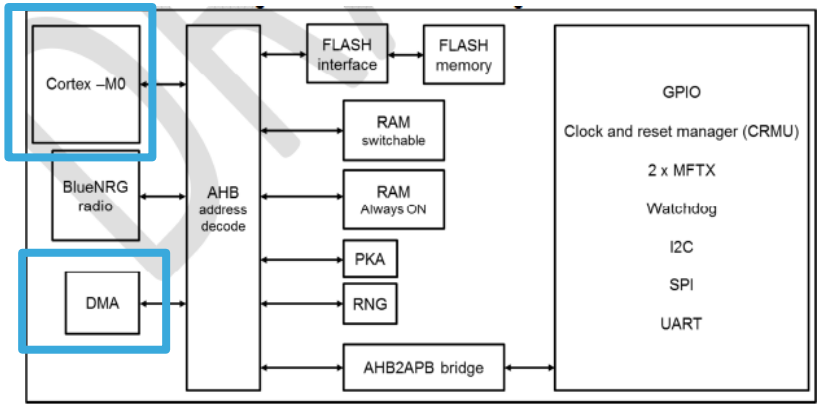
Communication Peripherals

- SPI: Master and Slave support
- I2C: Baud rate supported up to 400 kb/s
- UART: Programmable Baud Rate, support of HW flow control
- PDM streaming (audio MEMS interface)

BlueNRG-1 Core

BLE Application Processor
BlueNRG-1

- GPIO
- 10 Bit ADC
- Communication Peripherals
- ULL RAM
- FLASH
- M0 Core
- Application and Profiles
- BLE Protocol stack
- Link Layer
- 2.4 GHz BlueNRG Radio



M0 Core

- **32 Bit architecture, 32 MHz speed**
- ultra-low leakage retention state
- **SWD debug port**

TIMER

- **MFTX: Two multi function timers**

DMA

- **Data transfer without CPU intervention**

life.augmented

BlueNRG-1
Application Flash

BLE Application Processor
BlueNRG-1

- GPIO
- 10 Bit ADC
- Communication Peripherals
- ULL RAM
- FLASH
- M0 Core
- Application and Profiles
- BLE Protocol stack
- Link Layer
- 2.4 GHz BlueNRG Radio

ULL RAM

- One 12kB block always in retention
- One 12kB block switchable

160KB

My Application

ble_stack_ lib.a

BlueNRG-1 User Flash
160KB

~100KB for application

FLASH **Application and Profiles**

- 70kB available for application code (full feature BLE stack)
- 110kB available for application code (minimum feature BLE stack)

life.augmented

BlueNRG-1
BLE stack

BLE Network Processor
BlueNRG-1

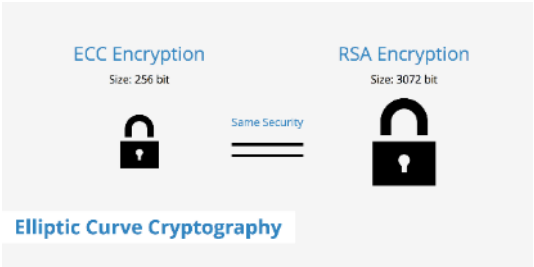
- GPIO
- 10 Bit ADC
- Communication Peripherals
- ULL RAM | FLASH
- M0 Core
- Application and Profiles
- **BLE Protocol stack**
- Link Layer
- 2.4 GHz BlueNRG Radio

Q1 2017

BLE 4.2 - Improved privacy and security

Advantages	Features	End User Examples
Industry-leading Privacy Keeps Bluetooth Smart devices from being tracked	**LE Privacy 1.2**	A Bluetooth Smart location tracker can only be followed by the owner or trusted group all while consuming less power
More Power Efficient Introduces refinements that help Bluetooth Smart devices save even more energy		
Highly Secure Features FIPS-compliant encryption ensuring confidential data stays that way	**LE Secure Connections**	A Bluetooth Smart lock or other smart home device provides industry standard security for added user confidence during device pairing

ECC Encryption
Elliptic-Curve-Cryptography



life.augmented

BlueNRG-1
BLE SOC presentation

- BlueNRG-1 BLE SOC presentation
-  BlueNRG-1 Power consumption figures
- BlueNRG-1 Development Tools
- BlueNRG-1 SW API to ease your design
- How to promote - Navigator tool

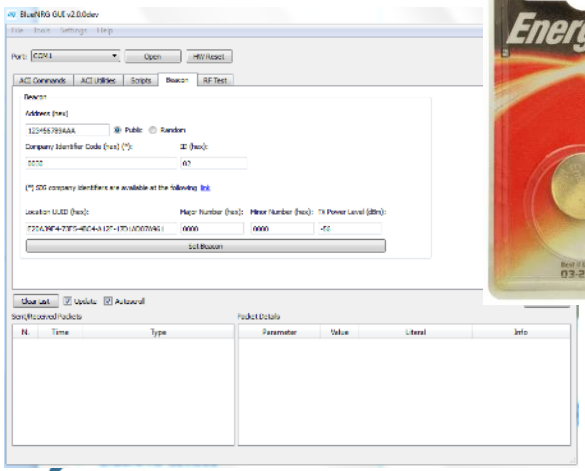
life.augmented

BlueNRG-1
Optimized for ultra-low-power

“Engineered to Advertise”

Ultra-low-power consumption in advertisement mode

16uA @ 1.28s



Prolonged battery life

life.augmented

BlueNRG-1

Low Power average power consumption

Beyond DS specs:
how to benchmark for real-case scenarios

From PEAK to AVERAGE power consumption

- RX **7.7mA**
- TX **8.3mA** @0dBm
- Sleep **0.9µA**
- Shut Down **2.5nA**

+

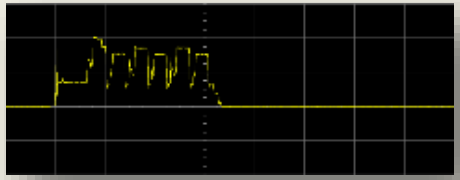
Ultra-fast **SLEEP** to **ACTIVE** transition time

=

Ultra-low average power consumption

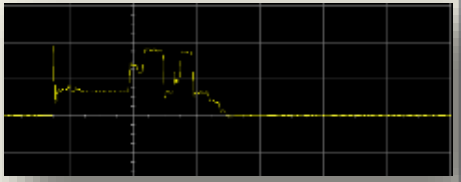
Advertisement Scenario
(with 15 byte payload)

Advertising Interval	Average Consumption SMPS ON	Average Consumption SMPS OFF
1.28s	16µA	26µA
500ms	37µA	64µA



Connection Scenario
(**without** payload)

Connection Interval	Average Consumption SMPS ON	Average Consumption SMPS OFF
1.28s	5.5µA	9.6µA
30ms	167µA	316µA



Connection Scenario
(**with** payload)

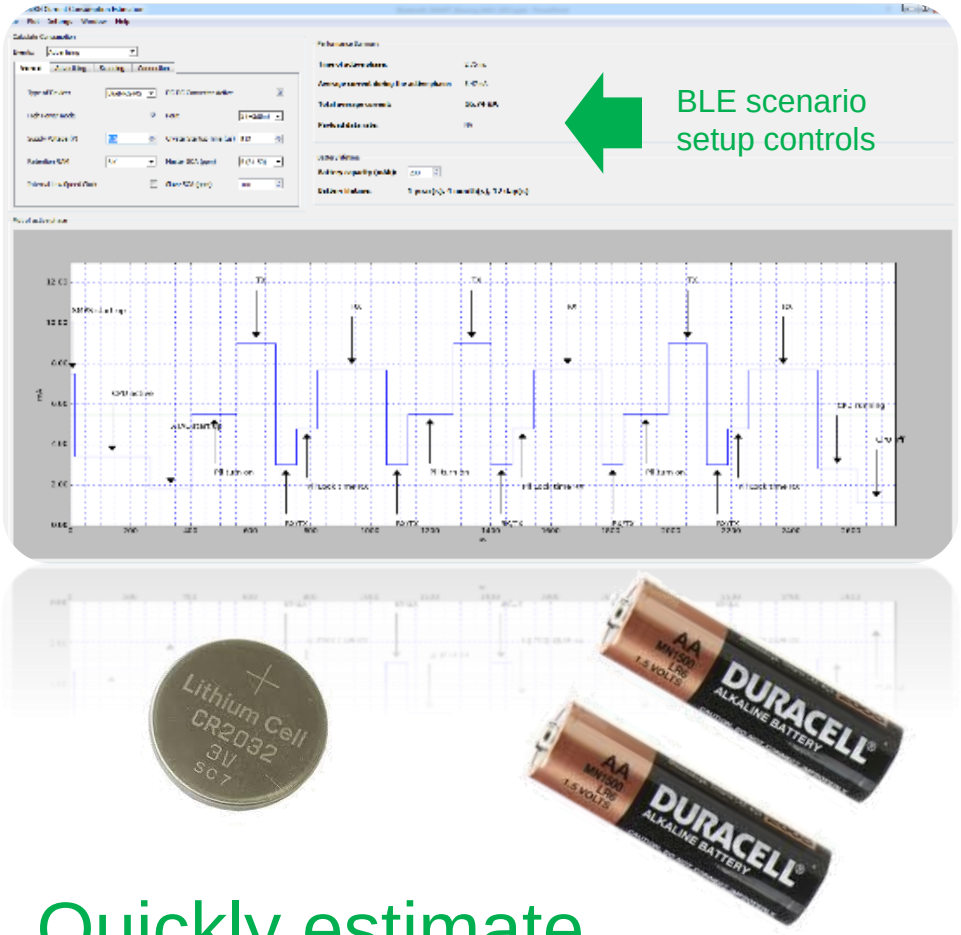
Connection Interval	Application Data	Average Consumption SMPS ON	Average Consumption SMPS OFF
1.28s	2 byte	6µA	10µA
30ms	20 byte	233µA	437µA

Measurement conditions: Vin=3.0V, Slave Mode, 32KHz XOSC - High Power Mode - Pout 2dBm

life.augmented

Test your real end-user scenario – beyond marketing numbers

BlueNRG-1
Power consumption Tool



Quickly estimate
the battery life

- ST provides a **Current Consumption Estimation Tool**
- It enables the user to estimate the average current consumption and the battery lifetime in the applicative cases
- The user can select:
  - General:
    - Supply voltage
    - TX output power
    - Master/Slave sleep clock accuracy
    - Retention RAM
  - Connection Advertising or Scanning Interval
  - Data length
  - DC-DC converter active or not

STSW-BNRG001
BlueNRG current consumption estimation tool

http://www.st.com/web/en/catalog/tools/PF260405

life.augmented

BlueNRG-1
BLE SOC presentation

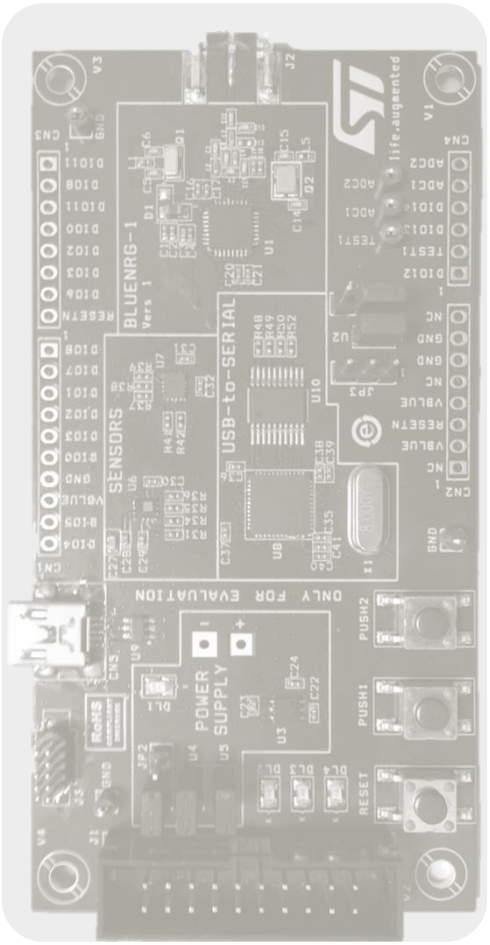
- BlueNRG-1 BLE SOC presentation
- BlueNRG-1 Power consumption figures
-  BlueNRG-1 Development Tools
- BlueNRG-1 SW API to ease your design
- How to promote - Navigator tool

life.augmented

BlueNRG-1 Development Tools
Full-featured DK

DK Resources

- High-level abstraction layer APIs
- Firmware documentation
- Pre-compiled HEX files (for rapid evaluation)
- Examples and templates, in source code
- Drivers for sensors (motion and environmental)
- Beacon Application for iOS / Android (source code)
- Multiple tool-chains supported (IAR, Keil, Atollic, GCC)
- Real-time debugging capabilities



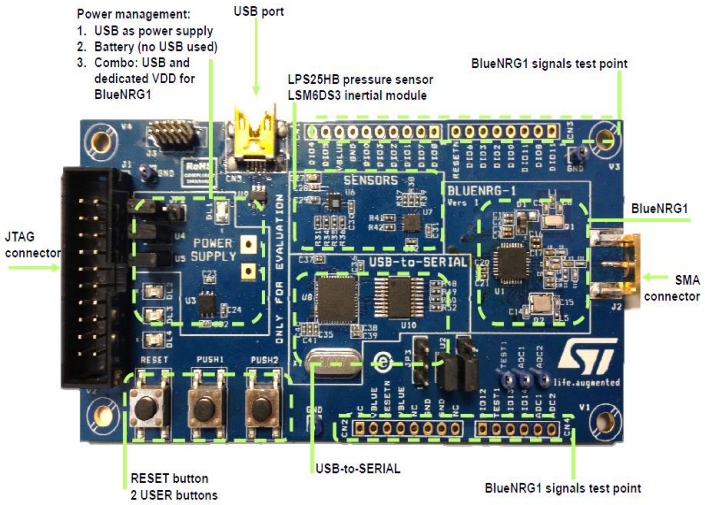
@ STSW-BLUENRG1-DK

life.augmented

BlueNRG-1 Development Tools
HW material

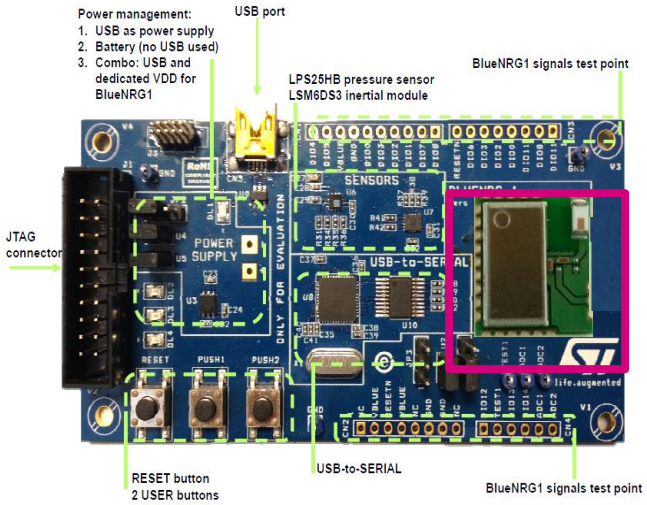
DK HW Resources

BlueNRG-1



STEVAL IDB007V1

SPBTLE-1S



STEVAL IDB007V1M

1 SW development kit

@ STSW-BLUENRG1-DK

life.augmented

BlueNRG-1 Development Tools
SW material

SW DK - BLE examples

```
void main(void)
{
  uint8_t ret;

  /* System Init */
  SystemInit();

  /* Identify BlueNRG-1 platform */
  SdkEvalIdentification();

  /* Init the UART peripheral */
  SdkEvalComUartInit(UART_BAUDRATE);

  /* BlueNRG-1 stack init */
  ret = BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params);
  if (ret != BLE_STATUS_SUCCESS) {
    printf("Error in BlueNRG_Stack_Initialization() 0x%02x\r\n", ret);
    while(1);
  }

  /* Init the BlueNRG-1 device */
  Device_Init();

  /* Start Beacon Non Connectable Mode*/
  Start_Beaconing();

  printf("BlueNRG-1 BLE Beacon Application (version: %s)\r\n", BLE_BEACON_VERSION_STRING);

  while(1)
  {
    /* BlueNRG-1 stack tick */
    BTLE_StackTick();

    /* Enable Power Save according the Advertising Interval */
    BlueNRG_Sleep(SLEEPMODE_WAKETIMER, 0, 0, 0);
  }
}
```

« SPP like »
Chat Demo

IAR SYSTEMS

KEIL™ Tools by ARM

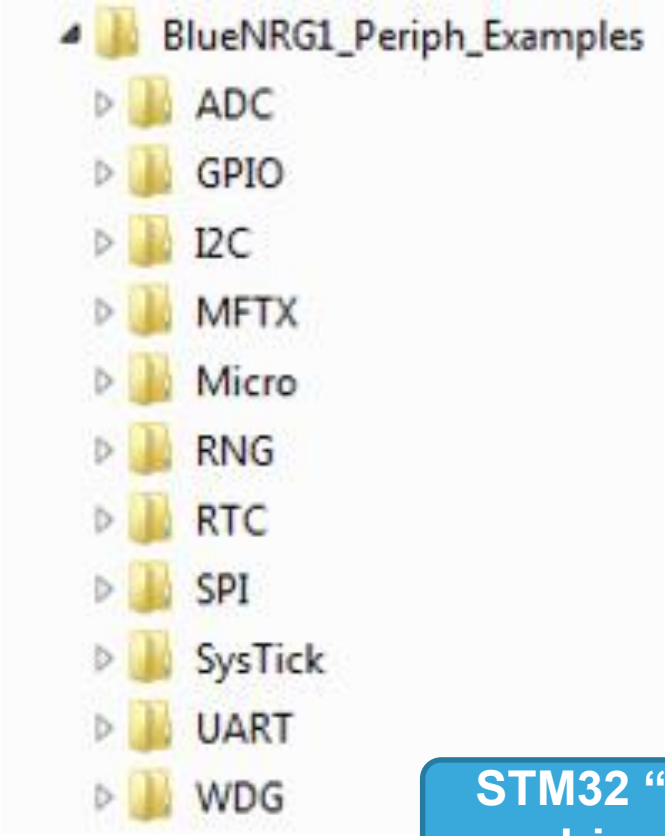
life.augmented

atollic

BlueNRG-1 Development Tools
SW material

SW DK - Peripherals examples

```
void main(void)
{
  uint8_t ret;

  /* System Init */
  SystemInit();

  /* Identify BlueNRG-1 platform */
  SdkEvalIdentification();

  /* Init the UART peripheral */
  SdkEvalComUartInit(UART_BAUDRATE);

  /* BlueNRG-1 stack init */
  ret = BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params);
  if (ret != BLE_STATUS_SUCCESS) {
    printf("Error in BlueNRG_Stack_Initialization() 0x%02x\r\n", ret);
    while(1);
  }

  /* Init the BlueNRG-1 device */
  Device_Init();

  /* Start Beacon Non Connectable Mode*/
  Start_Beaconing();

  printf("BlueNRG-1 BLE Beacon Application (version: %s)\r\n", BLE_BEACON_VERSION_STRING);

  while(1)
  {
    /* BlueNRG-1 stack tick */
    BTLE_StackTick();

    /* Enable Power Save according the Advertising Interval */
    BlueNRG_Sleep(SLEEPMODE_WAKETIMER, 0, 0, 0);
  }
}
```



STM32 “like” drivers

IAR SYSTEMS

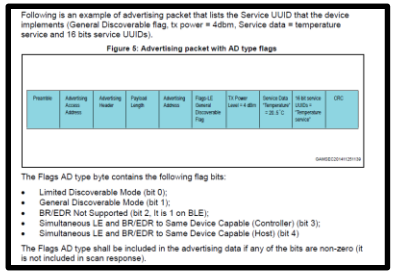
KEIL™ Tools by ARM

life.augmented

atollic

BlueNRG-1 Development Tools
SW material- Documentation

SW DK - Documentation



Following is an example of advertising packet that lists the Service UUID that the device implements (General Discoverable flag, tx power = 4dbm, Service data = temperature service and 16 bits service UUIDs).

Figure 5: Advertising packet with AD type flags

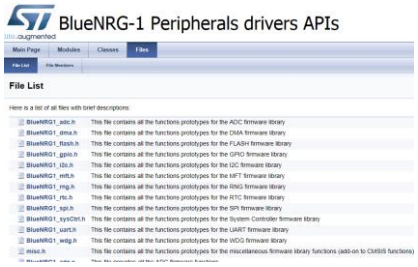
Preamble	Advertising Access Address	Advertising Header	Payload Length	Advertising Address	Flags-LE General Discoverable Flag	TX Power Level (+4 dBm)	Service Data "Temperature" = 25.5 °C	16 bit service UUIDs = "Temperature service"	CRC

The Flags AD type byte contains the following flag bits:

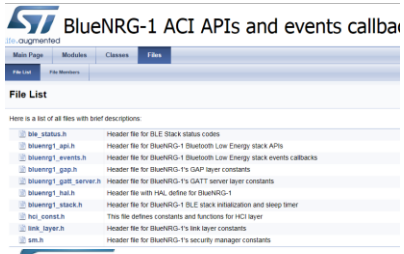
- Limited Discoverable Mode (bit 0);
- General Discoverable Mode (bit 1);
- BR/EDR Not Supported (bit 2. It is 1 on BLE);
- Simultaneous LE and BR/EDR to Same Device Capable (Controller) (bit 3);
- Simultaneous LE and BR/EDR to Same Device Capable (Host) (bit 4)

The Flags AD type shall be included in the advertising data if any of the bits are non-zero (it is not included in scan response).

PM0257 programming guide : BLE concepts & associated ST API



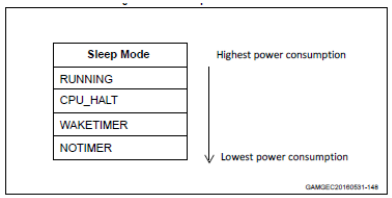
Doxygen documentation : BlueNRG-1 BLE and peripherals API



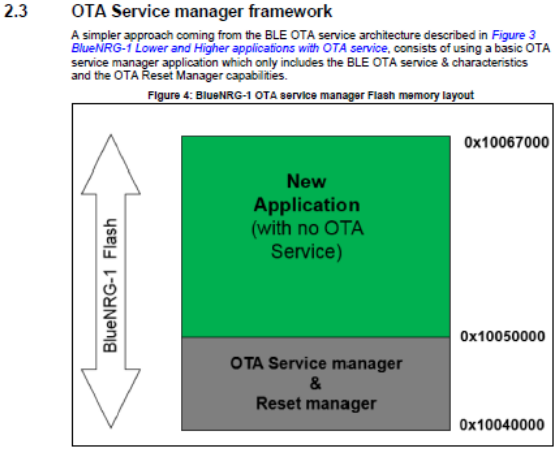
life.augmented

BlueNRG-1 Development Tools
SW material- Documentation

SW DK - Documentation



AN4820 : BlueNRG-1 Low Power modes



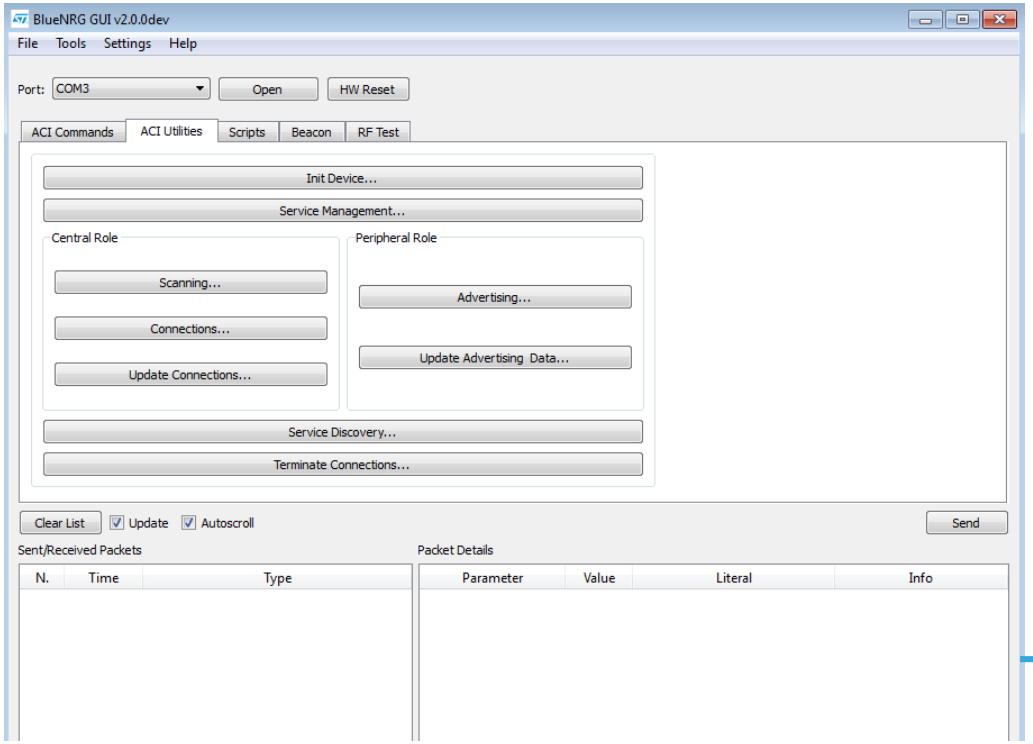
AN4869 : BlueNRG-1 Firmware Upgrade Over the Air

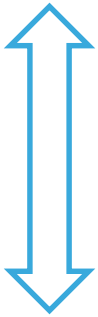
life.augmented

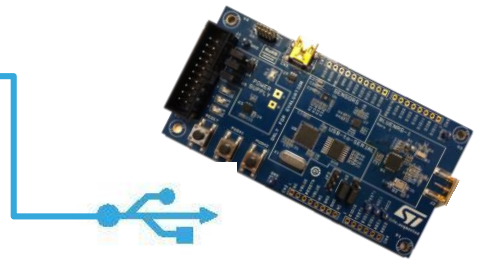
BlueNRG-1 Development Tools
Tool material

ST BLE GUI

@ STSW-BNRGUI







Comprehensive GUI to understand BLE concept and associated ST APIs

life.augmented

BlueNRG-1
BLE SOC presentation

- BlueNRG-1 BLE SOC presentation
- BlueNRG-1 Power consumption figures
- BlueNRG-1 Development Tools
-  BlueNRG-1 SW API to ease your design
- How to promote - Navigator tool

life.augmented

BlueNRG-1 Development Tools
SOC solution = SW leverage (1/3)

BlueNRG-1 User Flash – 160KB

0x10067FFF

My Application

ble_stack_ lib.a

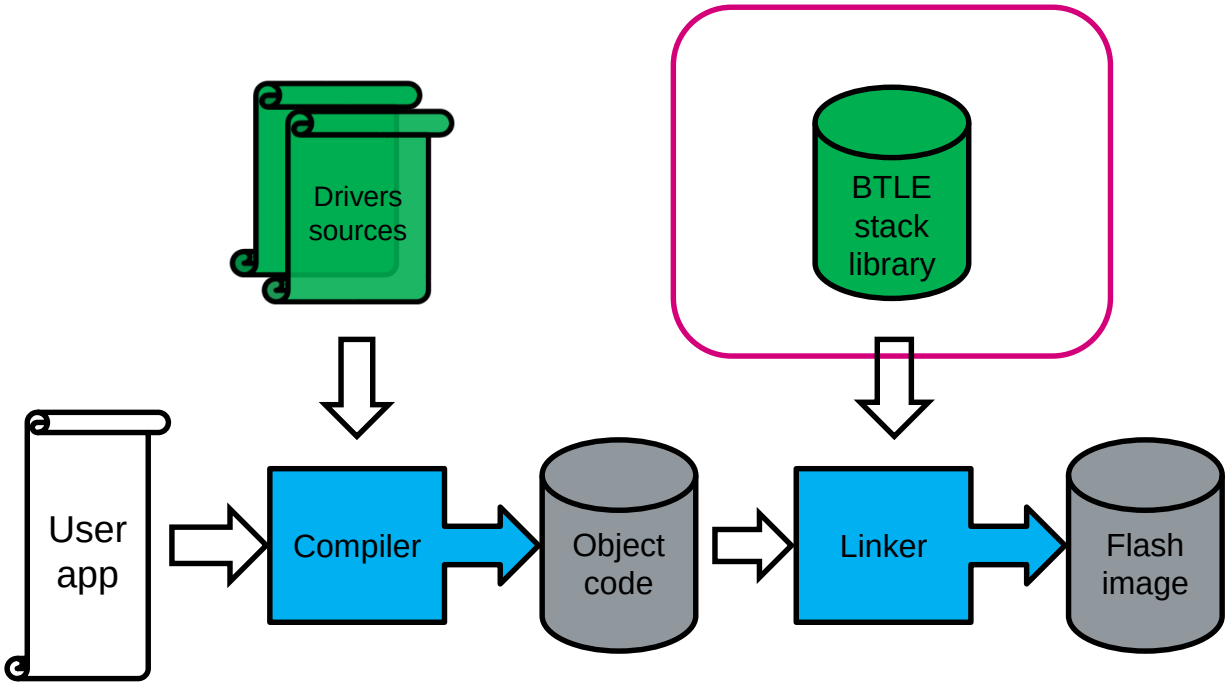
ST BLE Library

0x10040000

BlueNRG-1 User Flash 160KB

life.augmented

BlueNRG-1 Development Tools
SOC solution = SW leverage (2/3)



atollic

IAR SYSTEMS

KEIL Tools by ARM

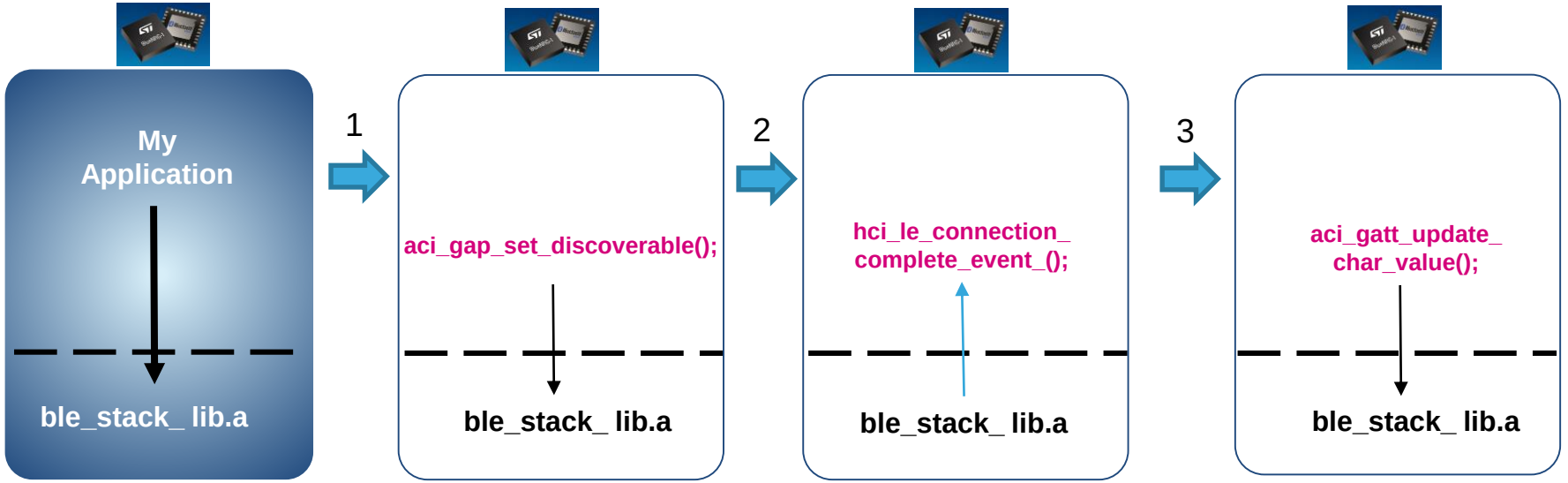
Provided by ST

Provided by third party

life.augmented

BlueNRG-1 Development Tools
SOC solution = SW leverage (3/3)

High-level abstraction layer APIs



BlueNRG-1 adversting

BlueNRG-1 connected event

BlueNRG-1 sending data

scan

connect request

notification data

life.augmented

BlueNRG-1

BlueNRG-1 DK to ease your SW design

SW architecture takeaways

```
void main(void)
{
  uint8_t ret;

  /* System Init */
  SystemInit();

  /* Identify BlueNRG-1 platform */
  SdkEvalIdentification();

  /* Init the UART peripheral */
  SdkEvalComUartInit(UART_BAUDRATE);

  /* BlueNRG-1 stack init */
  ret = BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params);
  if (ret != BLE_STATUS_SUCCESS) {
    printf("Error in BlueNRG_Stack_Initialization() 0x%02x\r\n", ret);
    while(1);
  }

  /* Init the BlueNRG-1 device */
  Device_Init();

  /* Start Beacon Non Connectable Mode*/
  Start_Beaconing();

  printf("BlueNRG-1 BLE Beacon Application (version: %s)\r\n", BLE_BEACON_VERSION_STRING);

  while(1)
  {
    /* BlueNRG-1 stack tick */
    BTLE_StackTick();

    /* Enable Power Save according the Advertising Interval */
    BlueNRG_Sleep(SLEEPMODE_WAKETIMER, 0, 0, 0);
  }
}
```

life.augmented

1 peripheral and stack on same core
=
simply application design

2 peripherals interface = “STM32 like”

4 stack and application over same flash
=
simplify FW upgrade procedure
&
Forget IFR process (if you know…)

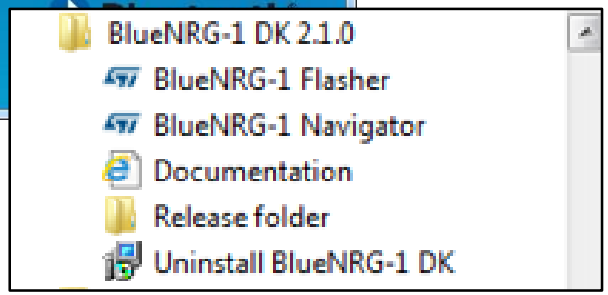
BlueNRG-1
BLE SOC presentation

- BlueNRG-1 BLE SOC presentation
- BlueNRG-1 Power consumption figures
- BlueNRG-1 Development Tools
- BlueNRG-1 SW API to ease your design
-  How to promote - Navigator tool

life.augmented

BlueNRG-1

BlueNRG-1 Navigator (1/3)

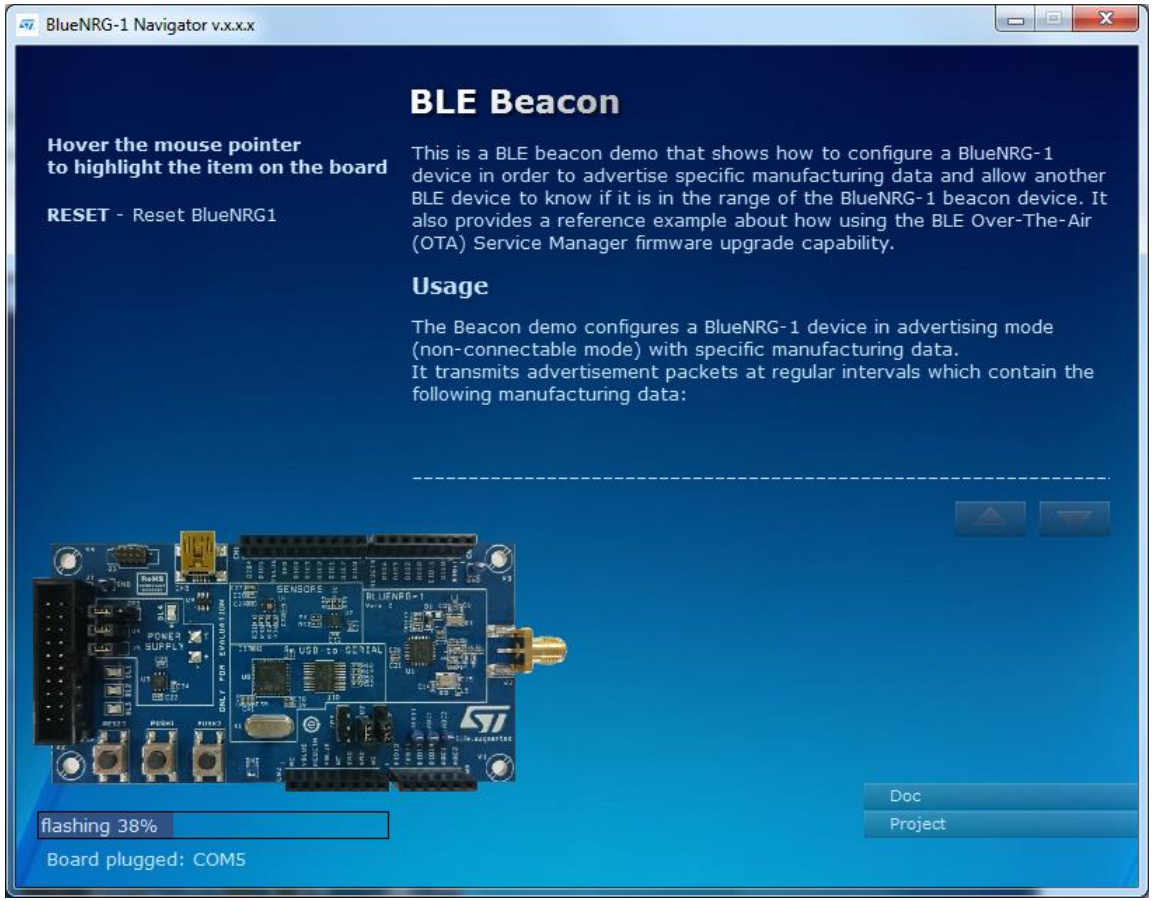


BlueNRG-1

BlueNRG-1 Navigator (2/3)

download and run the selected application prebuilt binary image into the BlueNRG-1 platform

access to the demo description, board configuration and to the source code



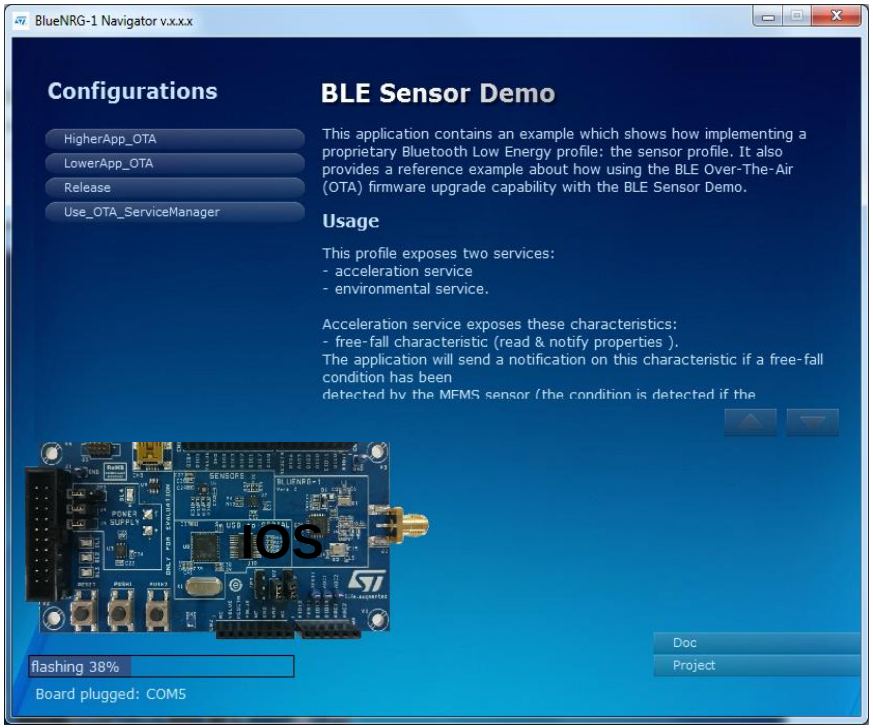
life.augmented

BlueNRG-1

BlueNRG-1 Navigator (3/3)

Sensor Demo

1



Discoverable as BlueNRG

2

BlueNRG App

BlueNRG App
≠
Blue MS (BlueMicrosystem over Nucleo)

life.augmented

BlueNRG-1

BlueNRG-1 Navigator (3/3)

Remote Control Demo

1

Discoverable as Node

2

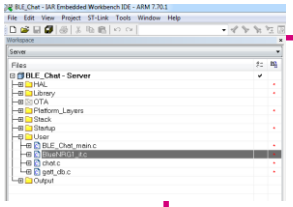
Free debug App on google and apple store

From smartphone write 0x04 , 0x05 ..to change LED status

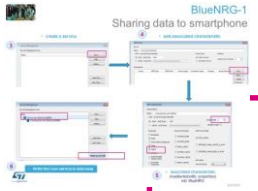
life.augmented

BlueNRG-1 Takeways

Optimized for ultra-low-power
Beacon mode : 16uA @ 1.28s



BlueNRG-1 Powerful Development Kit
Navigator : Promotion Tool
ST GUI : Comprehensive BLE concepts
BlueNRG-1 various code examples



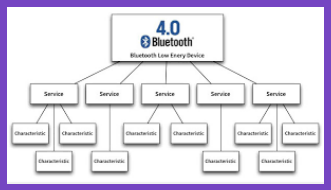
BlueNRG-1 on the web
Dedicated and specific documentation on demand
@ rf-support-emea@st.com

life.augmented

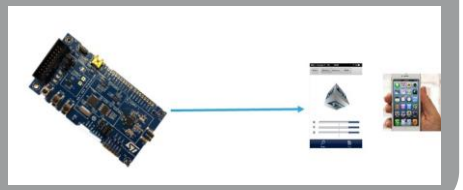
BlueNRG-1 BLE SOC

BlueNRG-1 and associated development (& promotion) package presentation

BLE concept demystification



Enable BLE link over BlueNRG-1



life.augmented

BLE Concepts
Definitions

server

client

- **2 entities in a BLE communication**
  1. The server : exposing data (temperature, position, raw data, what you want !)
  2. The client : connecting to server and looking for data

- **A BLE application is based on an application profile**
  - standard : glucose meter, Heart Rate Monitor, Find me
  - Proprietary : sensor profile, chat profile , my custom profile !

- **A profile is a basic collection of data exposed by the device**
  - service : **a basic UUID** (0x1808 = Glucose meter Service)
  - characteristic : **basic UUID** and associated data exposed (MEMS,ect…)

BLE Concepts
Profile Definition

server

client

- **A profile is a basic collection of data exposed by the device**
  - service : a basic UUID (0x1808 = Glucose meter Service)
  - characteristic : basic UUID and associated data you are willing to expose (MEMS)

Service → Characteristic → User Data

0x1808 → 0x2A18 → Glucose measure

Glucose meter Profile

life.augmented

BLE Concepts
Profile Definition

server

Application Profile

Service

characteristic

Health thermometer profile

- GAP service (UUID: 1800)
- Health thermometer service (UUID: 1809)
- Device information service (UUID:180A)

Health thermometer service (UUID: 1809)

- Temperature measurement (UUID: 2A1C) Mandatory
- Temperature type (UUID: 2A1D) Optional
- Intermediate temperature (UUID: 2A1E) Optional
- Measurement interval (UUID: 2A21) Optional

Temperature measurement

Handle : 1
UUID: 2803 (declaration)
Data:
Data UUID: 2A1D
Data Handle: 2
Data Properties: Indicate

Handle : 2
UUID: 2A1C (attribute value)
Data:
Flags: <uint8>
Measurement value:<float>
Time stamp: <date_time>
Temperature type: <temperature_type>

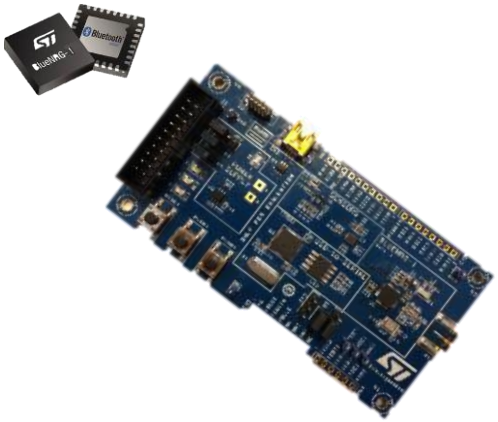
Handle: 3
UUID: 2903
Client characteristics configuration
Properties: Read, write

- Standard services & characteristics specification & UUID assignation available:
  - https://developer.bluetooth.org/gatt/services/Pages/ServicesHome.aspx
  - https://developer.bluetooth.org/gatt/characteristics/Pages/CharacteristicsHome.aspx

life.augmented

BLE Concepts
Attribut Table

Sensor demo Application Profile
service = Sensor Service
characteristic = Sensor characterisic



server

client

BlueNRG-1
Attribut Table

life.augmented

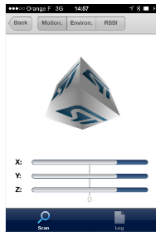
BLE Concepts
Attribut Table

server

1 The sensor proprietary profile at initialisation will add in BlueNRG-1RAM an entry (attribut table) to expose application data (MEMS)

BlueNRG-1 Attribut table

@1	@2	@3
Sensor Service UUID	Sensor Char UUID	MEMS Data



client

2 As soon as connected client will be able to access (write/read) to attribute table thanks to BLE specification

3 Application MEMS update will update the attribut table with the new characteristic value

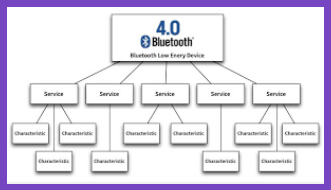
		MEMS Data

life.augmented

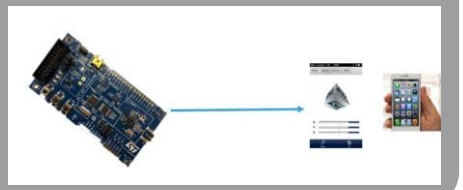
BlueNRG-1 BLE SOC

BlueNRG-1 and associated development (& promotion) package presentation

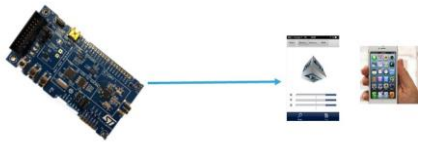
BLE concept demystification



Enable BLE link over BlueNRG-1



life.augmented

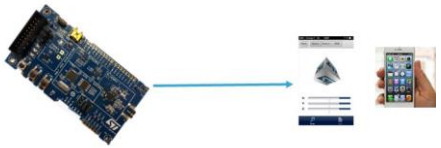


Enable BLE link over BlueNRG-1

BlueNRG-1 GUI Hands On

SW code implementation

life.augmented



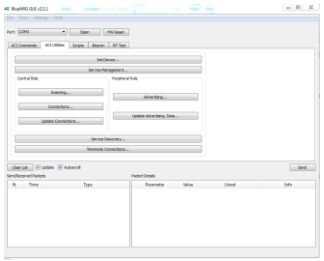
BlueNRG-1 – GUI Hands On
Prerequisites (1/2)

BlueNRG-1 SW development Kit : **STSW-BLUENRG1 DK**

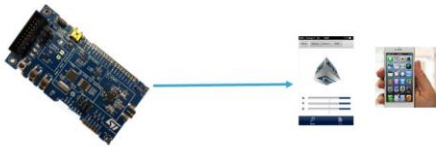
- http://www.st.com/content/st_com/en/products/embedded-software/evaluation-tool-software/stsw-bluenrg1-dk.html

BlueNRG-1 BLE GUI : **STSW-BNRGUI**

- http://www.st.com/content/st_com/en/products/embedded-software/wireless-connectivity-software/wireless-connectivity-software/stsw-bnrgui.html



life.augmented



BlueNRG-1 – GUI Hands On
Prerequisites (2/2)

Free BLE debug application

- Required to perform GUI basic hands on to enable connection

LightBlue
Apple Store

IOS

BLE Scanner
Google store

Android

Sensor debug application

- Required to execute and perform the sensor demo part of BlueNRG-1 DK binaries

Keyword: BlueNRG
Apple Store & Google store

life.augmented

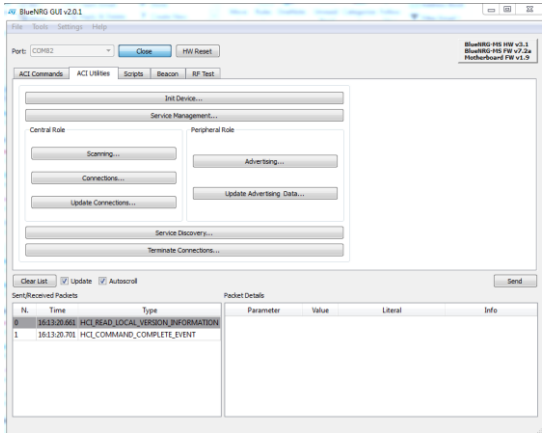
BlueNRG-1
ST BLE GUI - Hands On

1. from Navigator load DTM 16Mhz UART application on BlueNRG-1

@ STSW-BLUENRG1-DK

2. open ST BlueNRG-1 GUI

@ STSW-BNRGUI

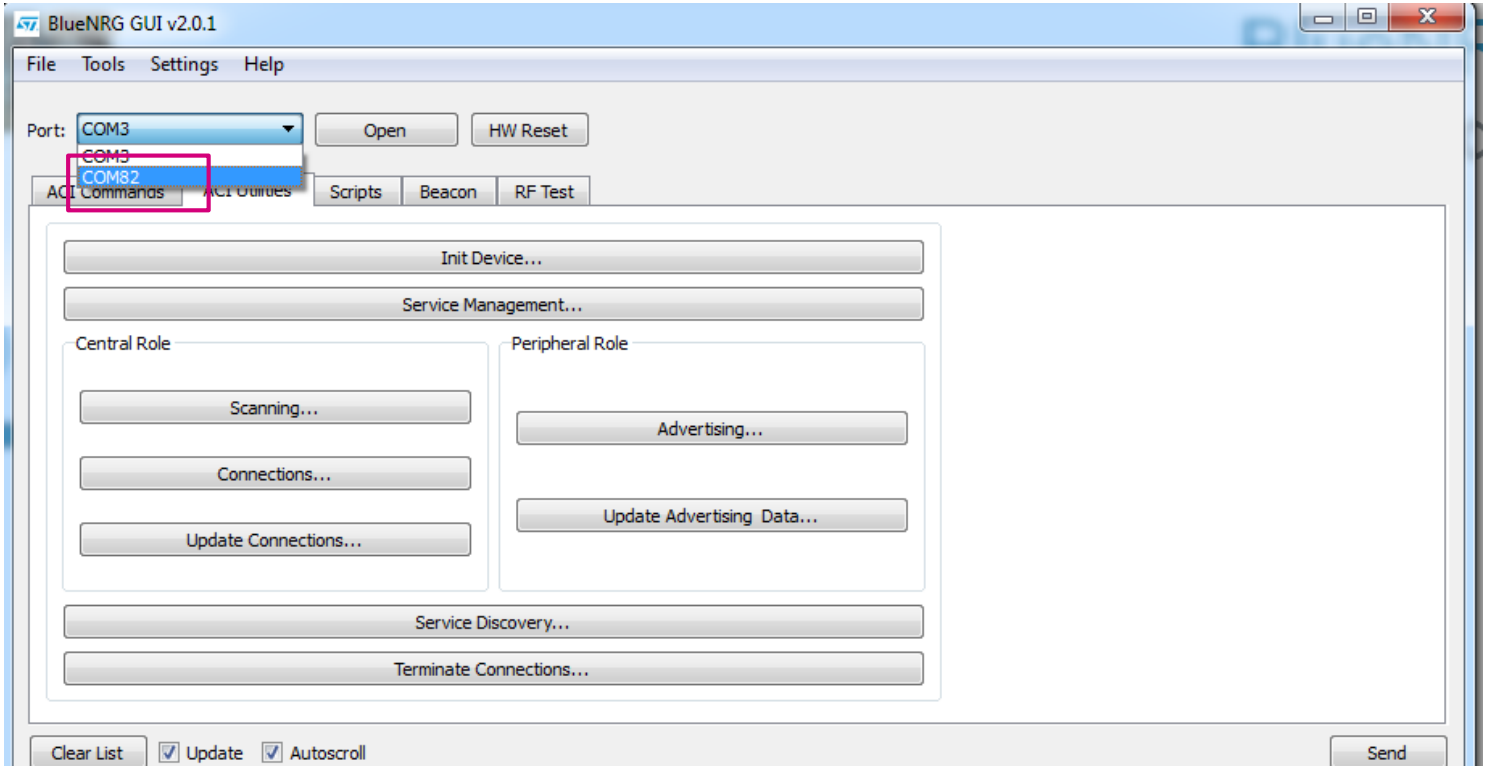


life.augmented

BlueNRG-1

ST BLE GUI - Hands On

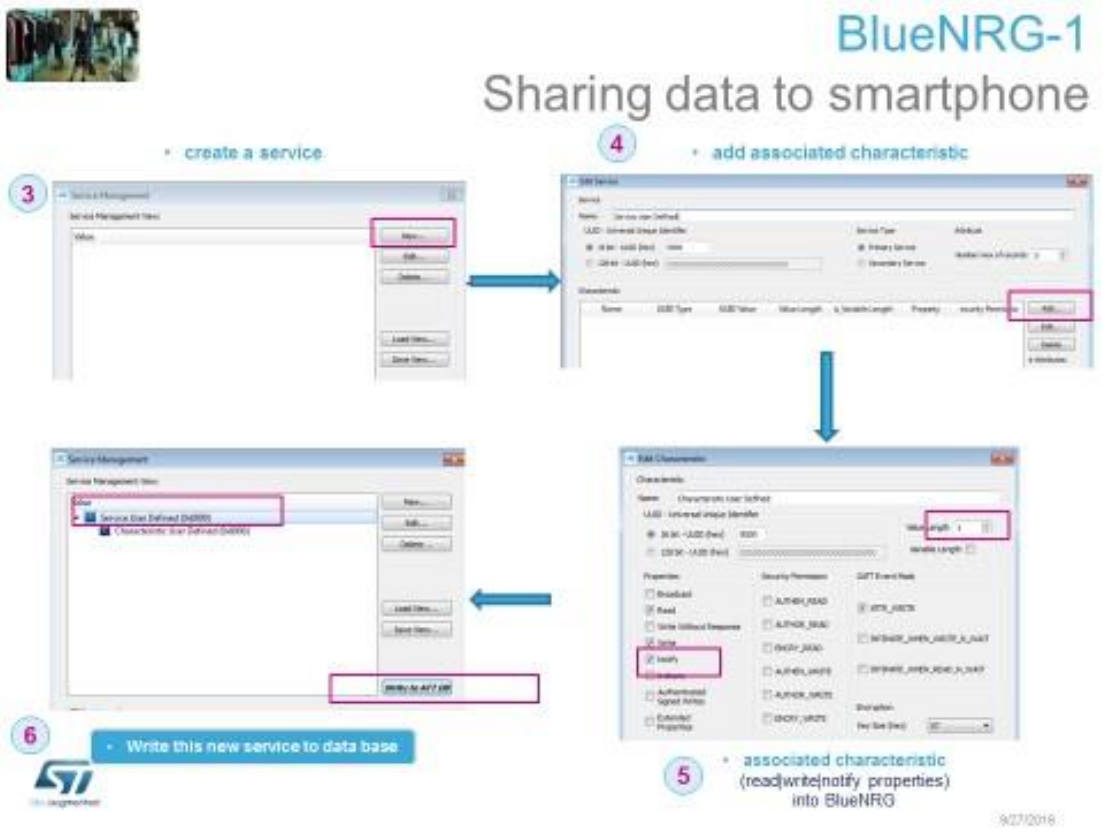
3



GUI now controlling BlueNRG-1 SOC

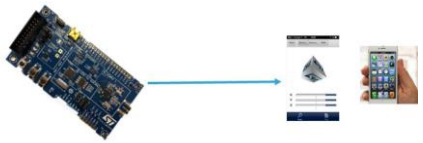
life.augmented

BlueNRG-1
ST BLE GUI - Hands On



@ BlueNRG-1 Hands On IDB007V1 V3.0.pdf

life.augmented



Enable BLE link over BlueNRG-1

BlueNRG-1 GUI Hands On

SW code implementation

BlueNRG-1
Application structure

```
/* System configuration */
SystemInit();

/* BlueNRG-1 stack init*/
BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params
);

while(1) {

/* BLE state machine */
BTLE_StackTick();

/* Application state machine */
APP_Tick();

/* Power Save management */
BlueNRG_Sleep(SLEEPMODE_WAKETIMER, 0, 0, 0);

}/* while (1) */
```

Classical embedded system structure

- Two state machines for BLE stack and Application

- Advanced power management controlled by BLE stack and Applications

life.augmented

BlueNRG-1
Application structure

```
/* System configuration */
SystemInit();

/* BlueNRG-1 stack init*/
BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params
);

while(1) {

/* BLE state machine */
BTLE_StackTick();

/* Application state machine */
APP_Tick();

/* Power Save management */
BlueNRG_Sleep(SLEEPMODE_WAKETIMER, 0, 0, 0);

}/* while (1) */
```

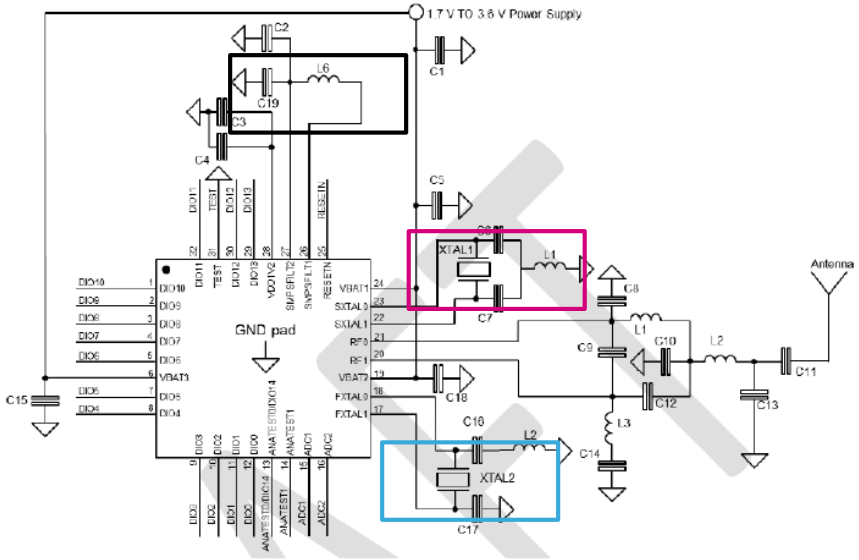
life.augmented

BlueNRG-1

System Init

```
/* System configuration */
SystemInit();
```

BlueNRG-1 device configuration parameters: Project Preprocessor options



HS_SPEED_XTAL

- HS_SPEED_XTAL_16MHZ /* High Speed crystal 16 MHz (Default) */
- HS_SPEED_XTAL_32MHZ /* High Speed crystal 32 MHz */
- HS_SPEED_XTAL_INTERNAL_RO /* High Speed Internal RO. Not useful when radio operations are needed or in any case when accurate ref clock is needed */

LS_SOURCE

- LS_SOURCE_INTERNAL_RO /* Low Speed Internal RO */
- LS_SOURCE_EXTERNAL_32KHZ /* Low Speed External 32 KHz (Default) */

SMPS_INDUCTOR

- SMPS_INDUCTOR_10uH /* SMPS Inductor 10 uH(Default) */
- SMPS_INDUCTOR_4_7uH /* SMPS Inductor 4.7 uH*/
- SMPS_INDUCTOR_NONE /* SMPS Inductor None */

BlueNRG-1
Application structure

```
/* System configuration */
SystemInit();

/* BlueNRG-1 stack init*/
BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params
);

while(1) {

/* BLE state machine */
BTLE_StackTick();

/* Application state machine */
APP_Tick();

/* Power Save management */
BlueNRG_Sleep(SLEEPMODE_WAKETIMER, 0, 0, 0);

}/* while (1) */
```

life.augmented

BlueNRG-1

BlueNRG_Stack_Init_params (1/4)

```
/* BlueNRG-1 stack init*/
BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params);
```

On file *SensorDemo*_config.h:

```
•const BlueNRG_Stack_Initialization_tBlueNRG_Stack_Init_params= {
(uint8_t*)stacklib_flash_data,
FLASH_SEC_DB_SIZE

FLASH_SERVER_DB_SIZE

(uint8_t*)stacklib_stored_device_id_data,

(uint8_t*)dyn_alloc_a,

NUM_GATT_ATTRIBUTES,

NUM_GATT_SERVICES,

ATT_VALUE_ARRAY_SIZE

NUM_LINKS,

CONFIG_TABLE};
```

BlueNRG-1

BlueNRG_Stack_Init_params (2/4)

```
/* BlueNRG-1 stack init*/
BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params);
```

```
FLASH_SEC_DB_SIZE, // Flash Security DB Size: 1024bytes (FLASH_SEC_DB_SIZE)

FLASH_SERVER_DB_SIZE,// Flash Server DB Size: 1024bytes (FLASH_SERVER_DB_SIZE)

(uint8_t*)stacklib_stored_device_id_data,

(uint8_t*)dyn_alloc_a,

NUM_GATT_ATTRIBUTES,

NUM_GATT_SERVICES,

ATT_VALUE_ARRAY_SIZE, // ATT_VALUE_ARRAY_SIZE (1344)

NUM_LINKS,

CONFIG_TABLE};
```

life.augmented

BlueNRG-1

BlueNRG_Stack_Init_params (3/4)

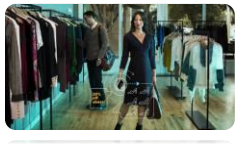
```
/* BlueNRG-1 stack init*/
BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params);
```

NUM_GATT_ATTRIBUTES

Number of attributes = **9 + 15** (NUM_GATT_ATTRIBUTES)

- The BlueNRG-1 stack uses **9 default attributes**
- The sensor demo application needs of attributes:
- Free fall characteristic = **3 attributes (declaration, value, client characteristic configuration descriptor)**
- Accelerometer characteristic = **3 attributes (declaration, value, client characteristic configuration descriptor**)
- Temperature characteristic = **3 attributes (declaration, value, characteristic format descriptor)**
- Pressure characteristic = **3 attributes (declaration, value, characteristic format descriptor)**
- Humidity characteristic = **3 attributes (declaration, value, characteristic format descriptor)**

life.augmented



BlueNRG-1

BlueNRG_Stack_Init_params (4/4)

```
/* BlueNRG-1 stack init*/
BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params);
```

NUM_GATT_SERVICES

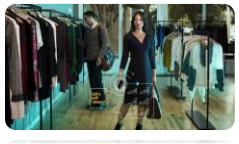
Number of services = **2 + 2 (NUM_GATT_SERVICES)**

- The BlueNRG-1 stack uses two default services GATT and GAP
- The sensor demo application needs of two services: accelerometer and environmental sensor

BlueNRG-1
Application structure

```
/* System configuration */
SystemInit();

/* BlueNRG-1 stack init*/
BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params
);

while(1) {

/* BLE state machine */
BTLE_StackTick();

/* Application state machine */
APP_Tick();

/* Power Save management */
BlueNRG_Sleep(SLEEPMODE_WAKETIMER, 0, 0, 0);

}/* while (1) */
```

life.augmented



BlueNRG-1

BTLE_StackTick()

```
/* BLE state machine */
BTLE_StackTick();
```

If there are BLE stack activities ongoing, user application is requested to call it:

- Timers state machine
- Link layer TX/RX state machine
- GAP procedures state machine
- GATT procedures state machine
- Security manager state machine
- Perform crystal calibration of low speed clock (internal or external).

BlueNRG-1 BLE Stack events callbacks

BlueNRG-1
Application structure

```
/* System configuration */
SystemInit();

/* BlueNRG-1 stack init*/
BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params
);

while(1) {

/* BLE state machine */
BTLE_StackTick();

/* Application state machine */
APP_Tick();

/* Power Save management */
BlueNRG_Sleep(SLEEPMODE_WAKETIMER, 0, 0, 0);

}/* while (1) */
```

life.augmented

BlueNRG-1
Application structure

```
/* System configuration */
SystemInit();

/* BlueNRG-1 stack init*/
BlueNRG_Stack_Initialization(&BlueNRG_Stack_Init_params
);

while(1) {

/* BLE state machine */
BTLE_StackTick();

/* Application state machine */
APP_Tick();

/* Power Save management */
BlueNRG_Sleep(SLEEPMODE_WAKETIMER, 0, 0, 0);

}/* while (1) */
```

life.augmented

BlueNRG-1

Sleep Mode management

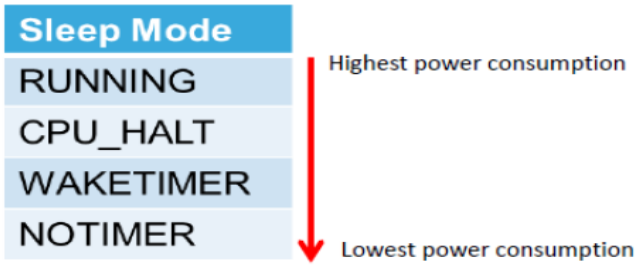
SW Low Power API

Sleep Mode		
RUNNING	2mA	
CPU_HALT	950nA + peripheral	→ CPU_HALT
WAKETIMER	950nA	→ SLEEP MODE
NOTIMER	400nA	→ STANDBYE

BlueNRG-1 HW low power modes

@ Please refer to AN4820

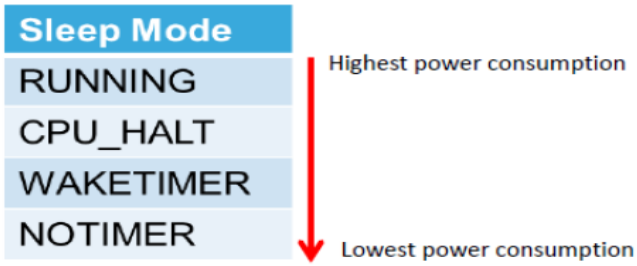
life.augmented



CPU_HALT

- In this mode only the CPU is stopped.
- All peripherals continue to operate and can wake up the CPU.
- The wakeup criteria are external interrupts or events (timers, RF timers).
- This is the lowest power save mode

950nA + peripheral



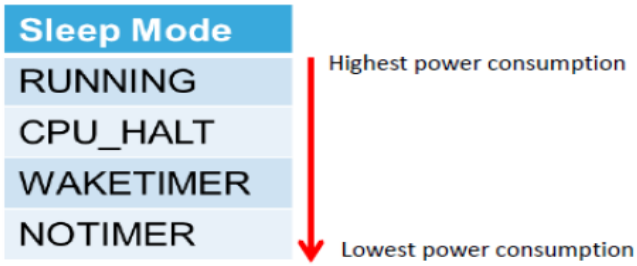
Sleep Mode

- In this mode the CPU is stopped and all the peripherals are disabled.
- Only the low speed oscillator block and the external wakeup sources block are running.
- The wakeup source will be:
  - Wakeup timer
  - IO9, IO10, IO11, IO12 and IO13

RF timer (Adv or Con events)
+ 4 Virtual Timers (application)
managed **by stack**

- When a wakeup is triggered by a previous listed source, the system reverts to the run mode with all the peripherals on.

950nA



Standby Mode

- In this mode the CPU is stopped and all the peripherals are disabled.
- The only wakeup sources are:
  - IO9, IO10, IO11, IO12 and IO13
- This mode is the highest power save mode.

400nA

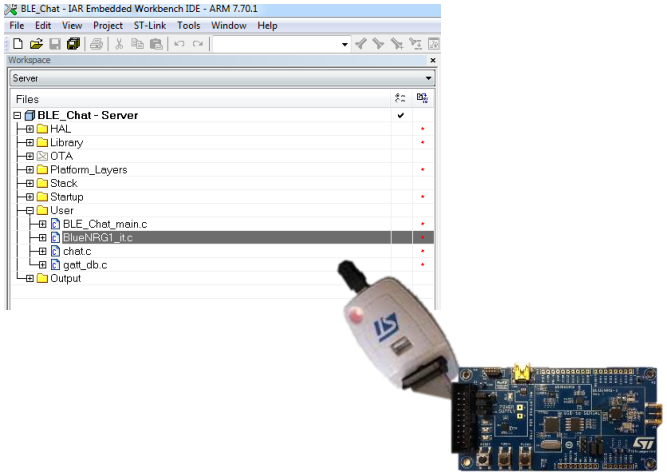
BlueNRG-1

Sleep Mode management vs. debug constraints

Sleep Mode
RUNNING
CPU_HALT
WAKETIMER
NOTIMER

Highest power consumption

Lowest power consumption



SOC enabling RF and application
brings real time constraints and so debug capabilities are limited

To ease debug we recommend to disable sleep
// BlueNRG_Sleep(SLEEPMODE_WAKETIMER, 0, 0, 0);

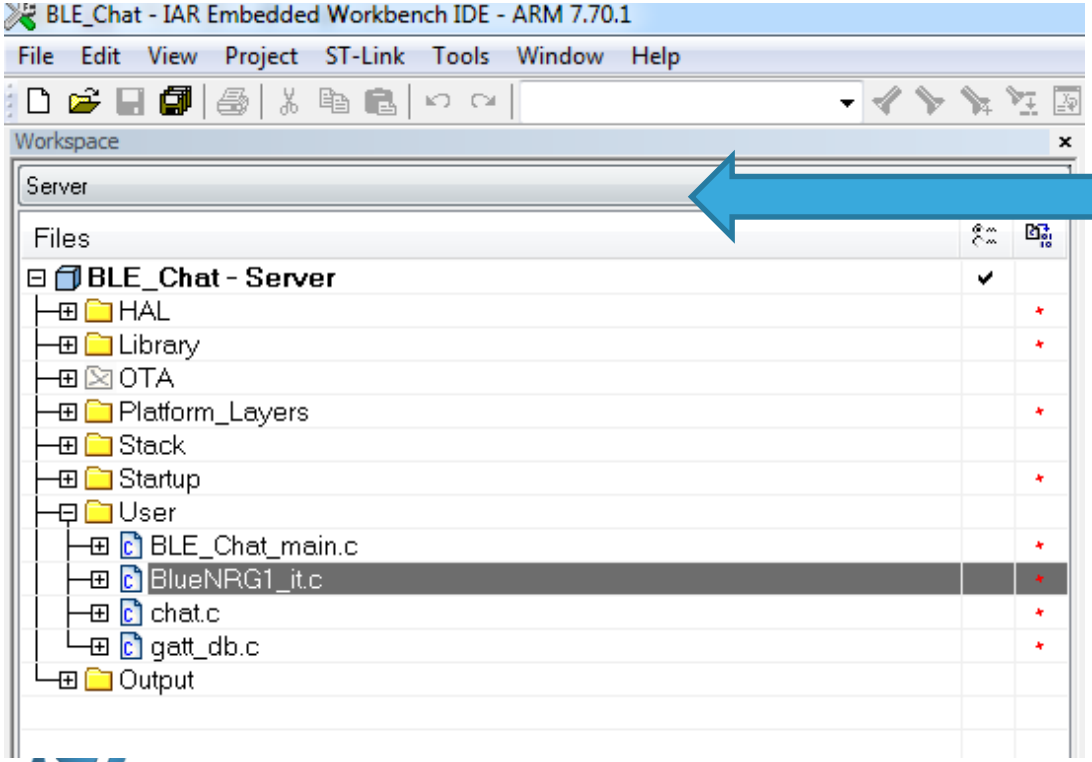
life.augmented

BlueNRG-1

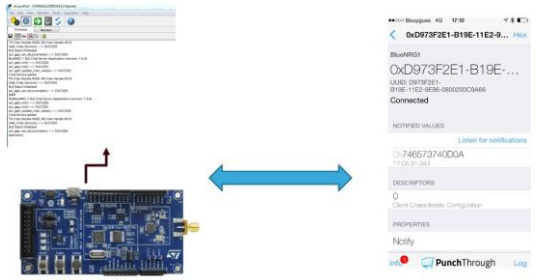
Enable SPP like connection- Chat Demo

The Chat demo described in BlueNRG-1 User Manual section 7 UM2071

1 Open Chat application from code example



Select Server workspace. The server is the device exposing service and characteristic (TX and RX)



life.augmented

Enable SPP like connection- Chat Demo

2 modify device name and BLE MAC @

In file chat.c

```
uint8_t CHAT_DeviceInit(void)
{
  uint8_t ret;
  uint16_t service_handle;
  uint16_t dev_name_char_handle;
  uint16_t appearance_char_handle;
  uint8_t name[] = {'B', 'l', 'u', 'e', 'N', 'R', 'G', '0  '}; //  from 0 to 10 //

#if SERVER
  uint8_t role = GAP_PERIPHERAL_ROLE;
  uint8_t bdaddr[] = {0xaa, 0x00, 0x00, 0xE1, 0x80, 0x00}; //  from 0 to 10 //
#else
…
…

ret = aci_gap_init(role, 0x00, 0x08, &service_handle,  &dev_name_char_handle, &appearance_char_handle);
// change device name lenght from 0x07 to 0x08 //
```

Enable SPP like connection- Chat Demo

3 modify local name

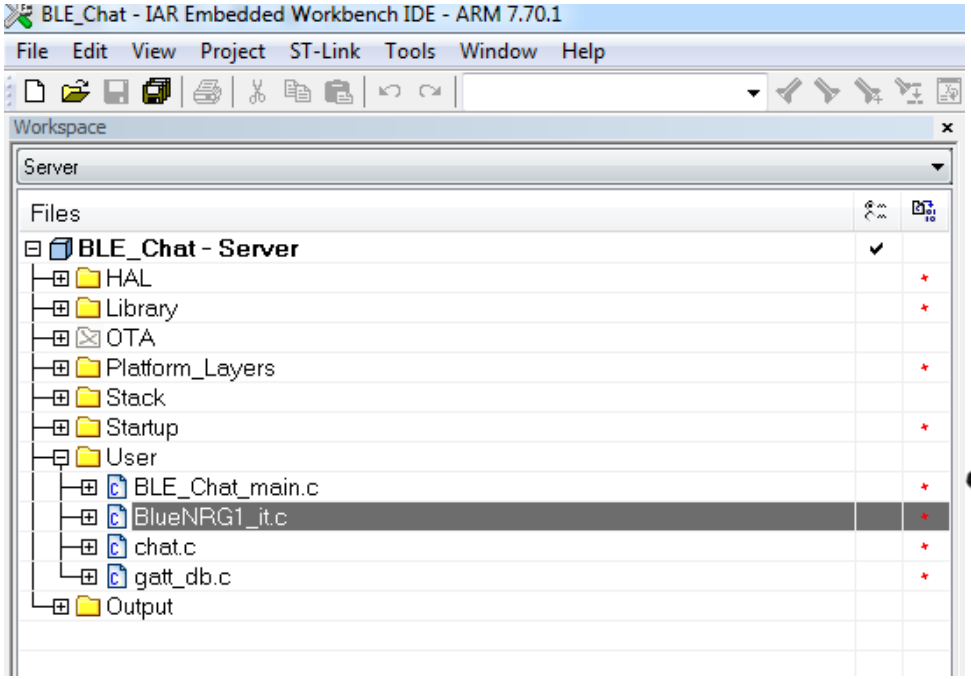
In file chat.c

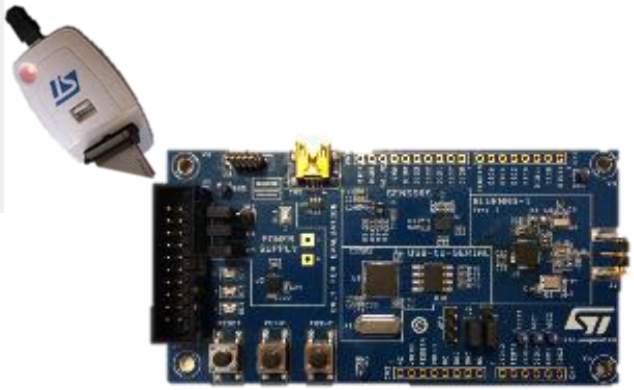
```
void Make_Connection(void)

  uint8_t local_name[] = {AD_TYPE_COMPLETE_LOCAL_NAME,'B','l','u','e','N','R','G','1','_','C','h','a','t','0'};


  ret = aci_gap_set_discoverable(ADV_IND, 0, 0, PUBLIC_ADDR, NO_WHITE_LIST_USE,
                                 sizeof(local_name), local_name, 0, NULL, 0, 0);
```

BlueNRG-1

Enable SPP like connection- Chat Demo

4 Build and download



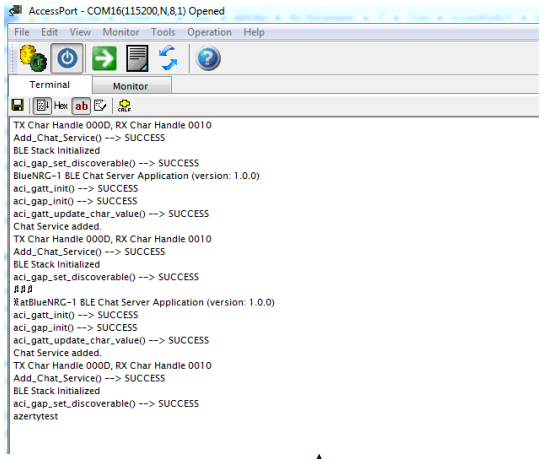


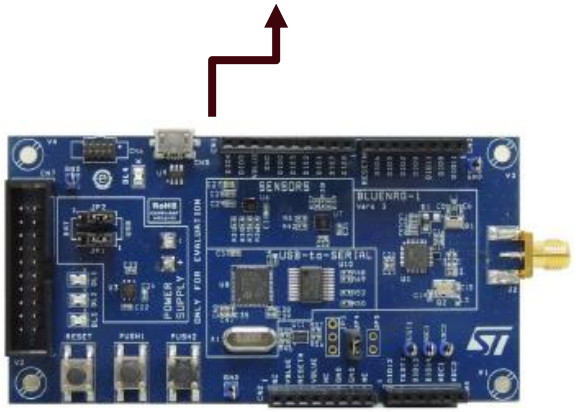
life.augmented

BlueNRG-1

Enable SPP like connection- Chat Demo

5



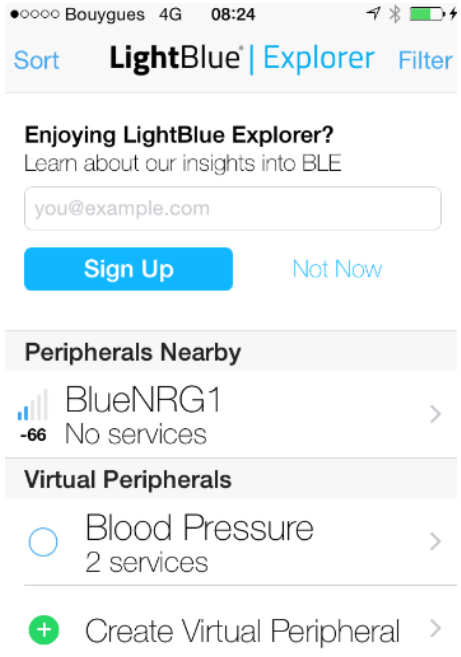


Server image

AccessPort
115 000

+

Reset the board



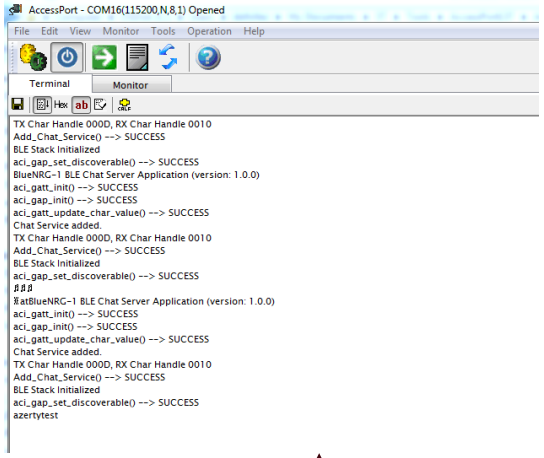
Client

life.augmented

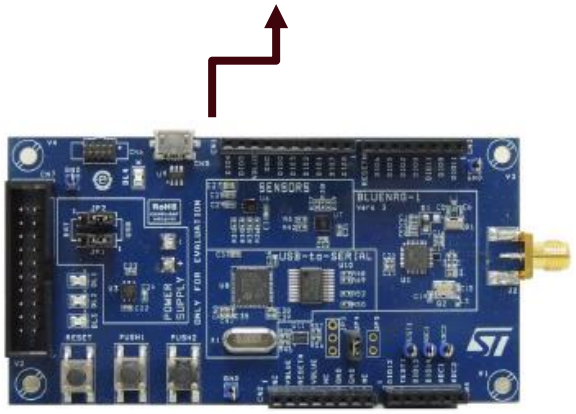
BlueNRG-1

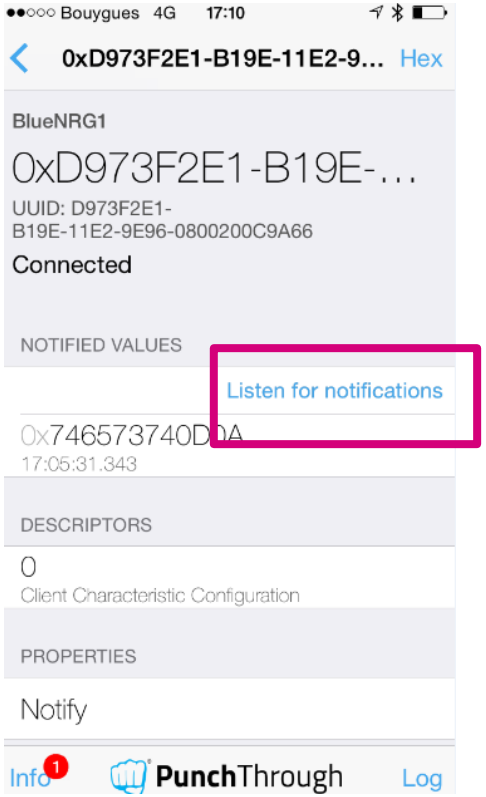
Enable SPP like connection- Chat Demo

6



Connect and enable
Listen for notification





Server image

Client

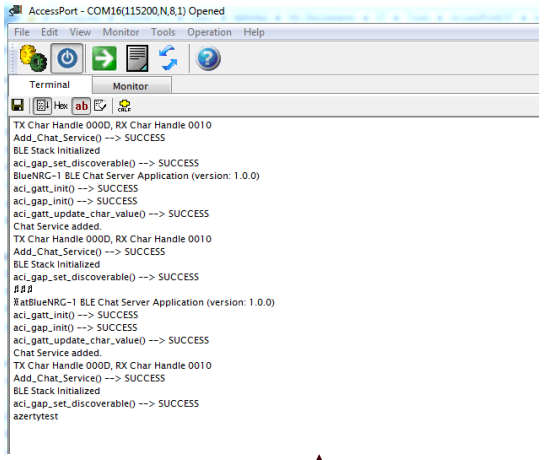
life.augmented

BlueNRG-1

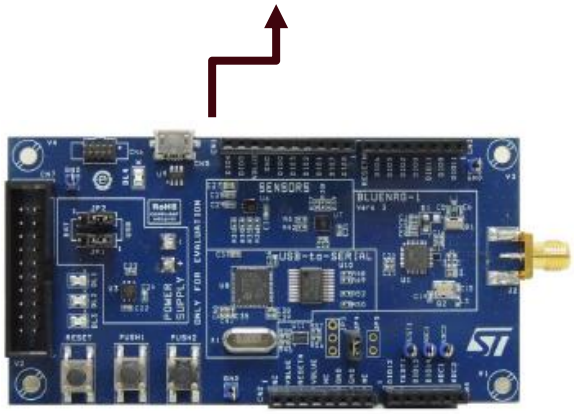
Enable SPP like connection

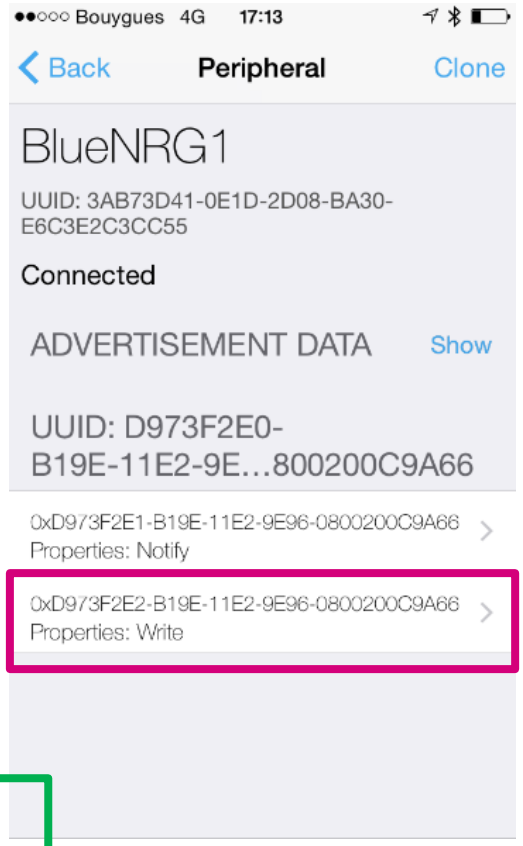
The Chat demo described in BlueNRG-1 User Manual section 7 UM2071

7



Write data from smartphone to BlueNRG-1





Attribute_Modified_CB
Event pushed to Application

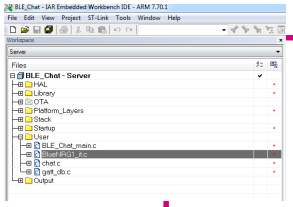
Server image

Client

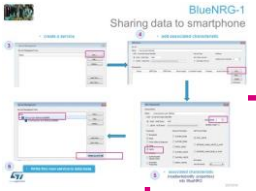
life.augmented

BlueNRG-1 Takeways

Optimized for ultra-low-power
Beacon mode : 16uA @ 1.28s

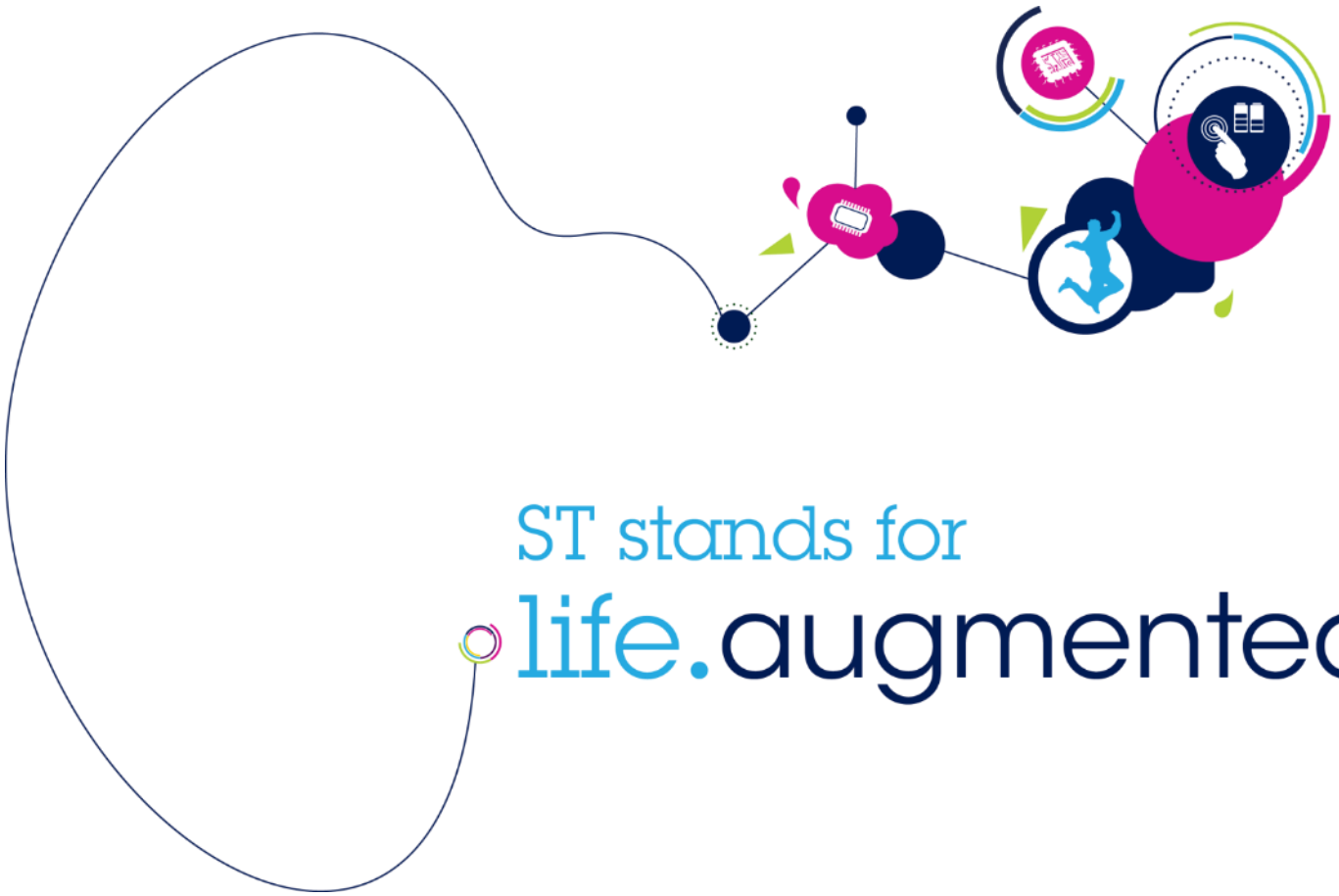


BlueNRG-1 Powerful Development Kit
Navigator : Promotion Tool
ST GUI : Comprehensive BLE concepts
BlueNRG-1 various code examples



BlueNRG-1 on the web
Dedicated and specific documentation on demand
@ rf-support-emea@st.com

life.augmented



ST stands for

life.augmented