FTF 2016
TECHNOLOGY FORUM

# DESIGNING WITH NXP BLUETOOTH SMART SOLUTION

## FTF-MHW-N1988

ALI BUKHARI
MIN YANG
FTF-MHW-N1988
MAY 16, 2016

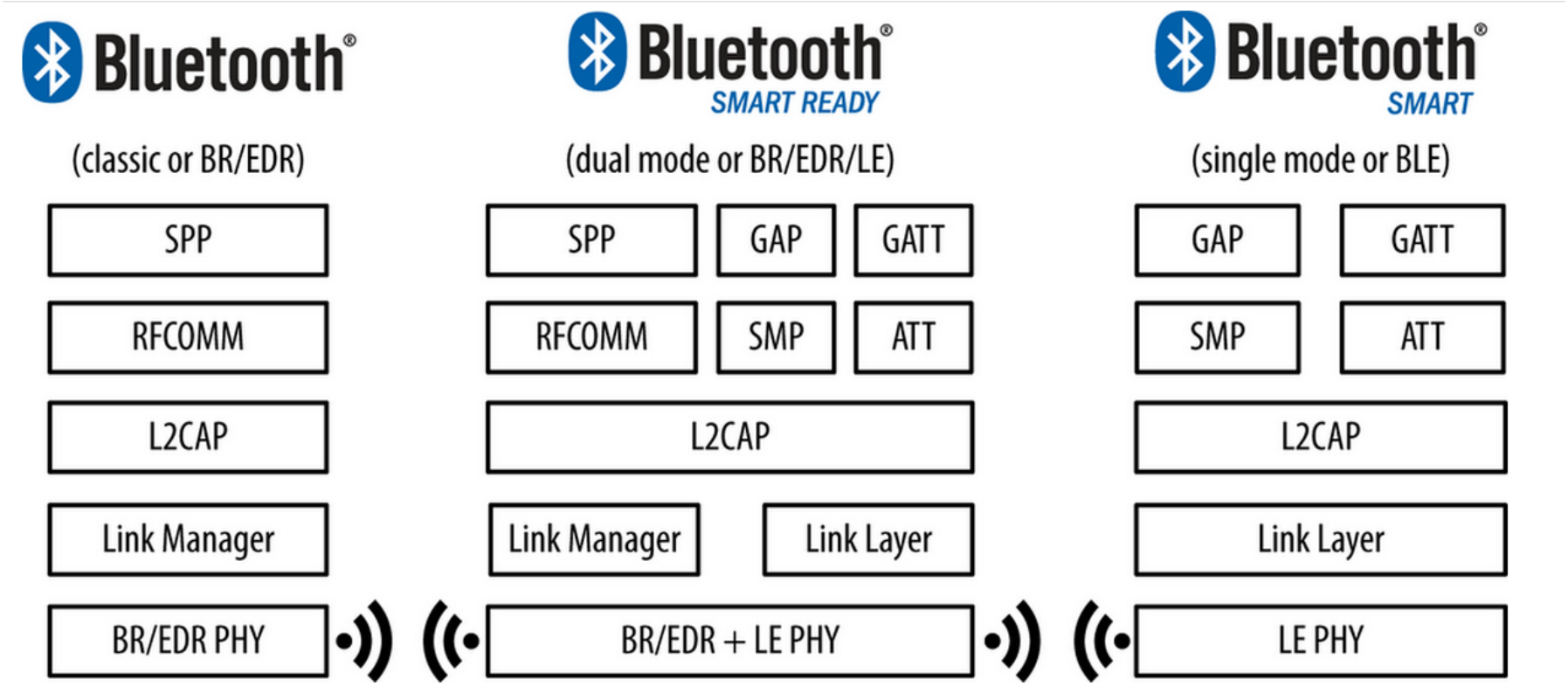
# AGENDA

- Introduction to BLE
- Introduction to QN902X Hardware
- Introduction to QN902X’s Evaluation Platform
- QN902X Code Review – Hands On
- Q&A

# Introduction to Bluetooth Low Energy

- Bluetooth Low Energy is a subset of the Bluetooth wireless technology, but is not backward compatible
- Perfect solution for transferring discrete data eg: Temp, Pressure, Mileage, Weight, Location, Control commands, etc.
- Sits in the 2.4GHz ISM band, hence license free
- Ultra low power consumption
- Frequency hopping helps in coexistence with  WiFi, Bluetooth Classic, ZigBee, etc.
- Extended battery life, makes it ideal for applications where its not feasible to have a charger, or portability is required
- Essential part of the ‘Internet of Things’

# Bluetooth Low Energy

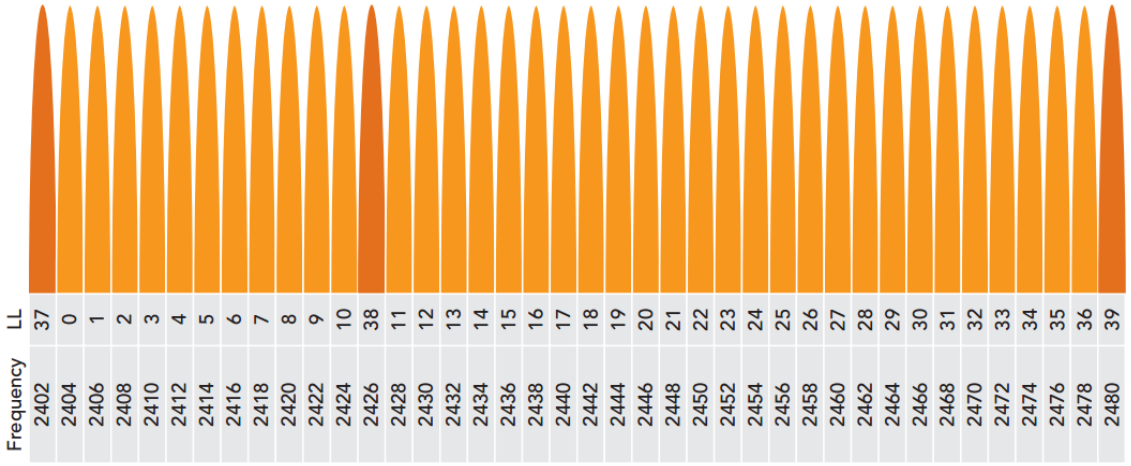



***The QN902X Bluetooth Low Energy platform from NXP is Single Mode BLE***

# Bluetooth Low Energy

# Bluetooth Low Energy

| Technical Specification | Classic Bluetooth | Bluetooth Low Energy |
|---|---|---|
| Frequency | 2400 to 2483.5 MHz | 2400 to 2483.5 MHz |
| Modulation Technique | Frequency Hopping | Frequency Hopping |
| Modulation Scheme | GFSK | GFSK |
| Modulation Index | 0.35 | 0.5 |
| Number of Channels | 79 | 40 |
| Channel Bandwidth | 1 MHz | 2 MHz |
| Nominal Data Rate | 1 - 3 Mbps | 1 Mbps |
| Application Throughput | 0.7 - 2.1 Mbps | < 0.3 Mbps |
| Nodes / Active Slaves | 7 | Unlimited |
| Security | 56 to 128 bit | 128-bit AES |
| Robustness | FHSS | FHSS |
| Voice | Capable | Not capable |




Advertisement Channels

Ch 37 : 2402MHz
Ch 38 : 2426MHz
Ch 39 : 2480MHz

Note: The channel numbers are in sequence but the frequencies are not adjacent.

SPECS FROM BLE 4.0

# BLE Data Throughput

The theoretical upper limit for BLE throughput is 1Mbps

**What's the Catch?**

- Bi-Directional Traffic
- Protocol Overhead
- CPU and Radio Limitations
- Artificial Software restrictions
- Uncontrolled devices (Phone, Tablet, Computer, etc)
- Connection interval (theoretically between 7.5ms to 4sec)
- Link Quality : Bit Error rate : Retransmission

- BLE Throughput = n * 20 B * 1/T { n = packets per interval, 20bytes per packet, T = Connection Interval }
- Max. packets per interval for iOS 8.3 is 6, Default connection interval is 30mSec
- BLE Throughput testing with iOS 8.3 = 6*20*(1/0.03) = 4000 bytes/sec = 32Kbits/sec

# Operating Range

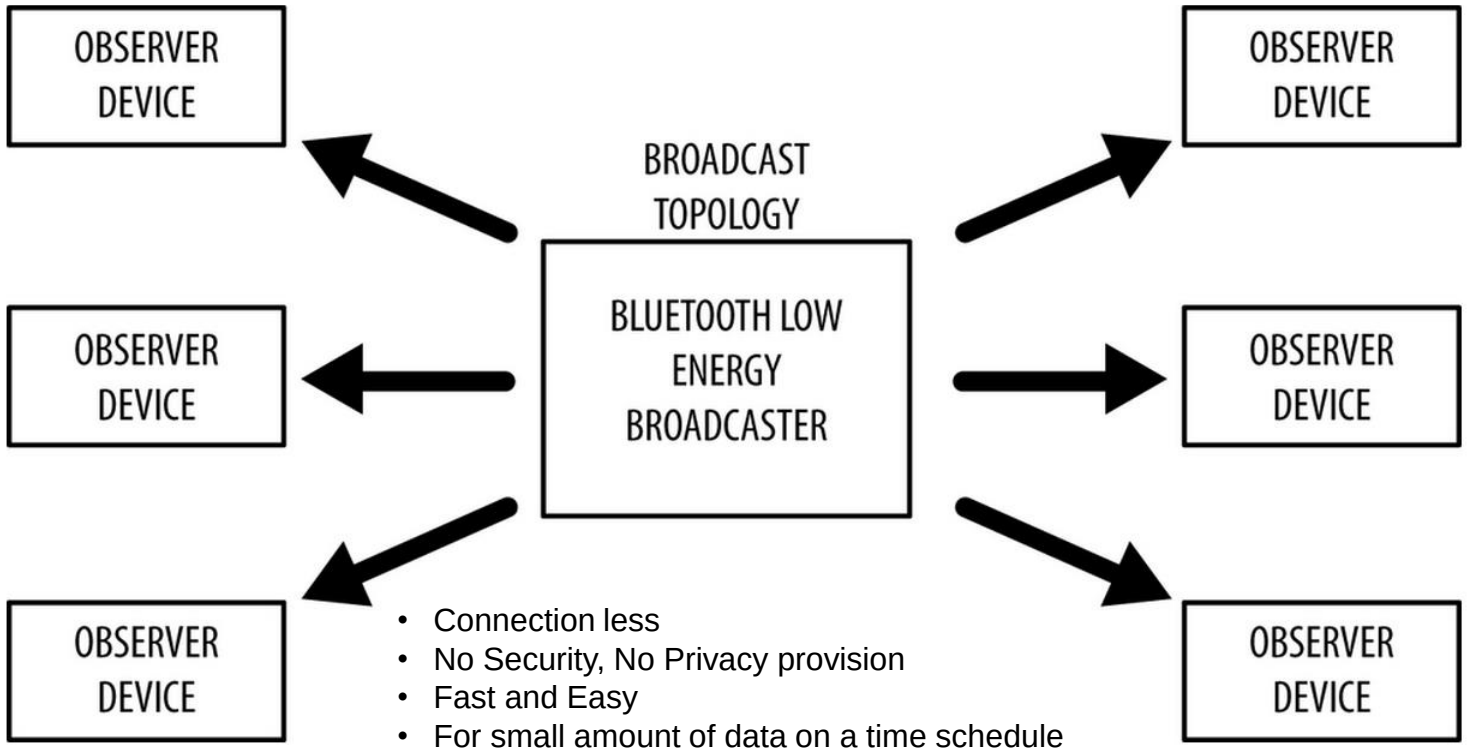
Actual covered range depends on various parameters

- Operational environment (indoor, outdoor, surroundings)
- Antenna design for Central and Peripheral(radiation pattern, orientation)
- RF Interference (proximity to Wifi, Zigbee, Office environment)
- Enclosure and Application scenario
- Final product placement and orientation

- NXP’s QN902x has the best in market RX sensitivity (upto -97dBm  with NON DCDC option)
- It has the market’s best Rx noise cancelling algorithm

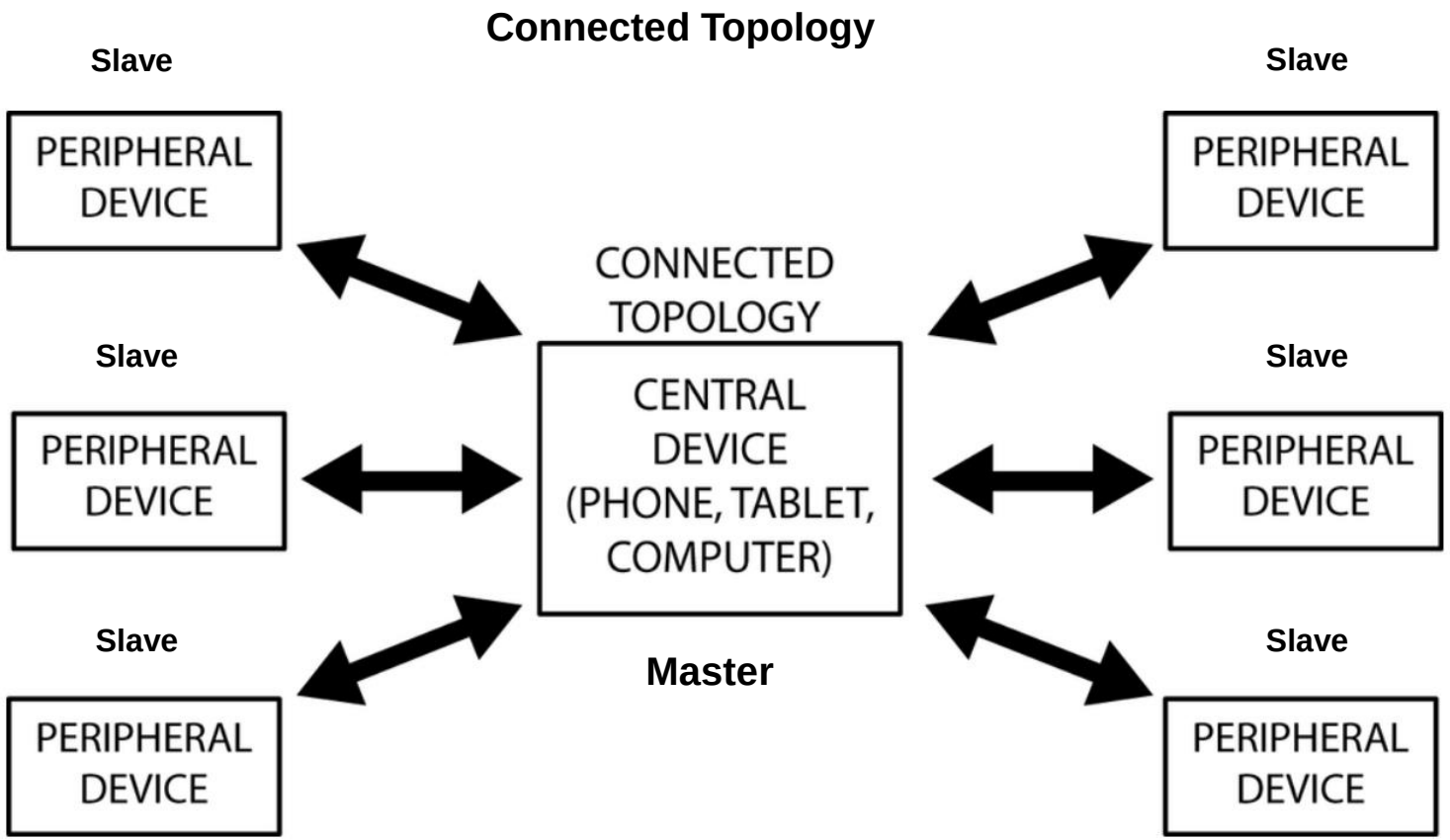
# Network Topology

- Can communicate in two ways: broadcasting or connections
- Subject to the guidelines established by the Generic Access Profile (GAP)
- Broadcast Topology

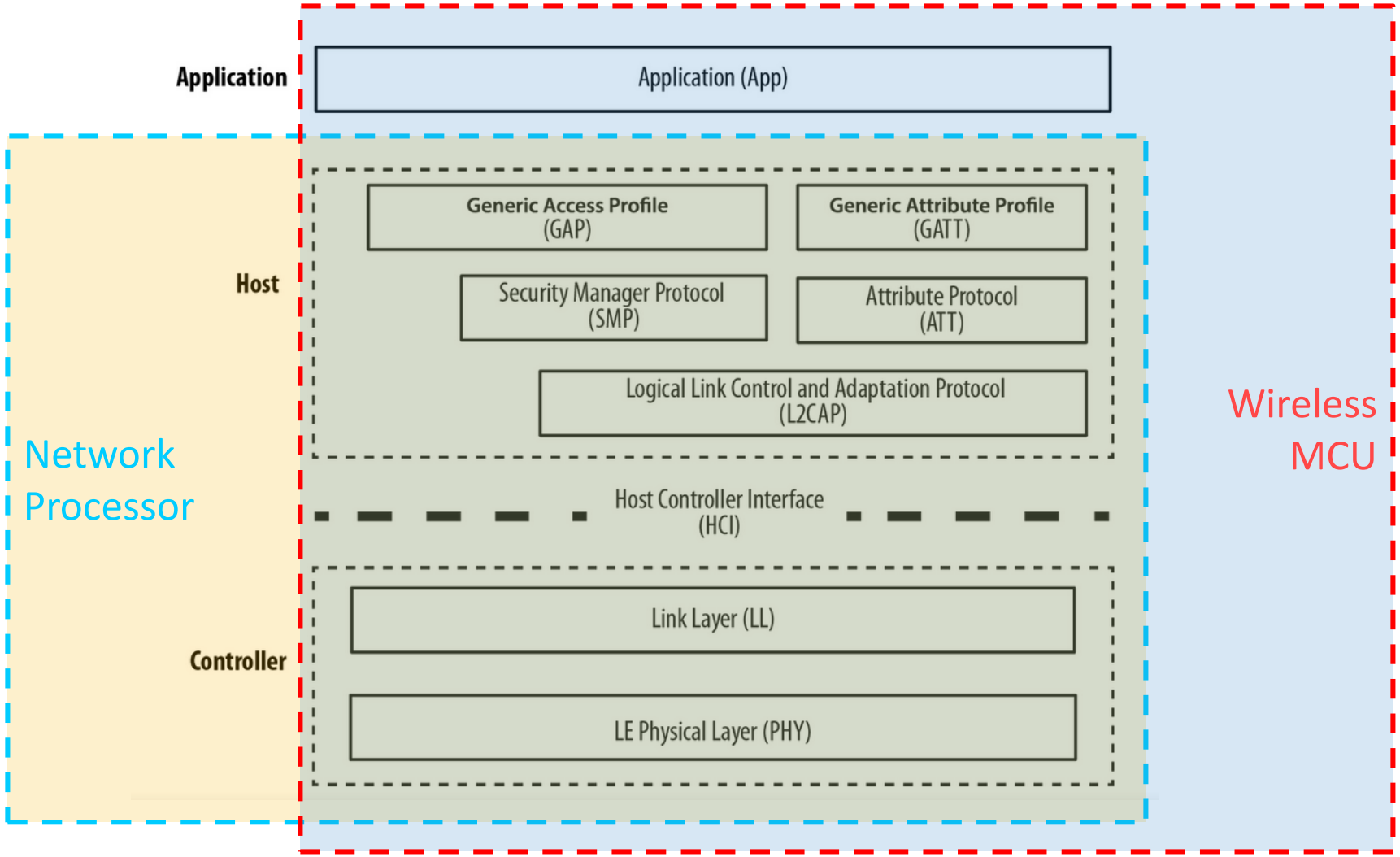


# Network Topology

- Connected
- Secured
- Configurable connection interval
- Can be used for sensor fusion
- Standard allows 8 connections to one Master



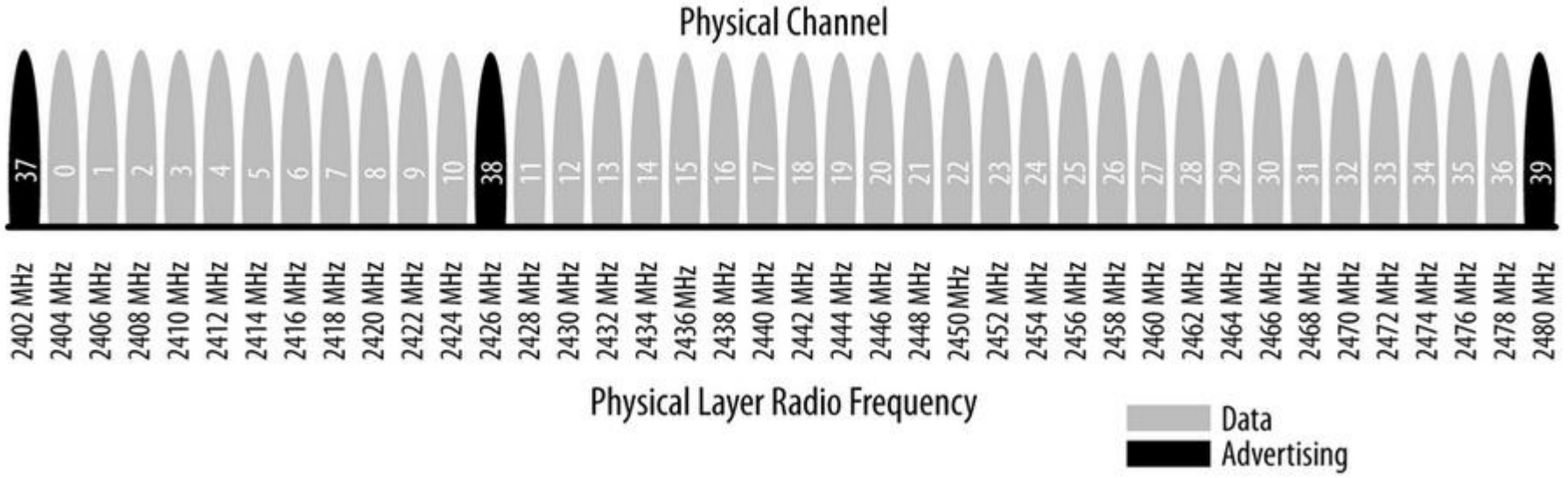
# LE Protocol Stack Architecture



# BLE Protocol Stack Architecture

## Physical Layer

- Frequency band: 2400~2483.5MHz
- RF channels: f = 2402+K*2 MHz, k=0,…,39
- Frequency hopping to combat interference and fading
- TX power: 0.01mW (-20dBm) ~ 10mW (+10dBm)
- Modulation: Gaussian Frequency Shift Keying (GFSK)
- Air data rate: 1Mbps (BLE 4.0 and 4.1)



# BLE Protocol Stack Architecture

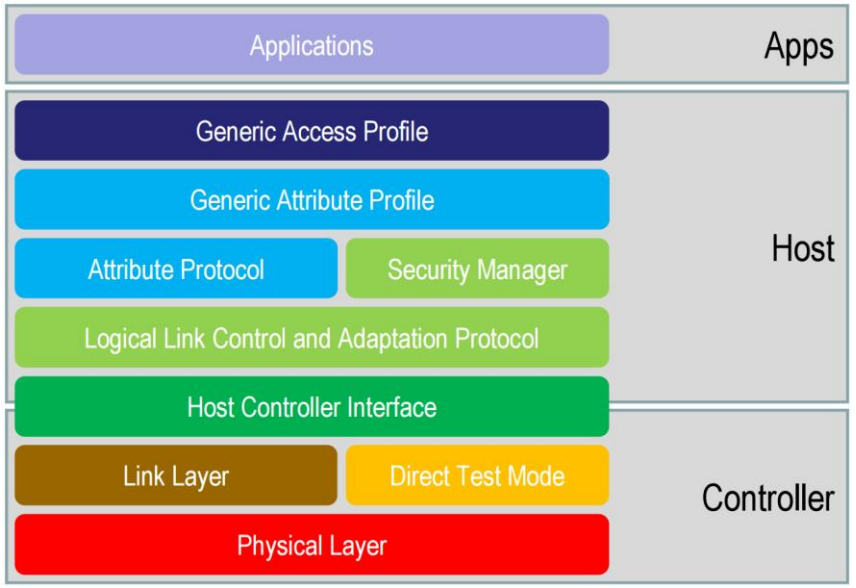
- **Protocols**
  - Layers that implement :
  - Packet Format
  - Routing
  - Multiplexing
  - Encoding / Decoding
  - Allowing data to flow effectively between peers

- **Profiles**: *Profiles essentially define how protocols should be used to achieve a particular goal, whether generic or specific*
  - Are Functionality:
  - Covering basic modes of operation (Generic Access Profile, Generic Attribute Profile)
  - Covering specific use cases (Proximity Profile, Glucose Profile)

- **Generic Profiles**: *Defined by specification and are fundamental to ensure interoperability*

- **Generic Access Profile (GAP)**
  - Defines roles, procedures, modes allowing device to broadcast, discover, establish and manage connection and negotiate security level
  - Top most BLE control layer and is mandatory for all devices

- **Generic Attribute Profile (GATT)**
  - Defines basic data model and procedure to allow devices to discover, read, write, and push data elements between them
  - Top most BLE data layer



# BLE Protocol Stack Architecture

SIG defined GATT based Profiles

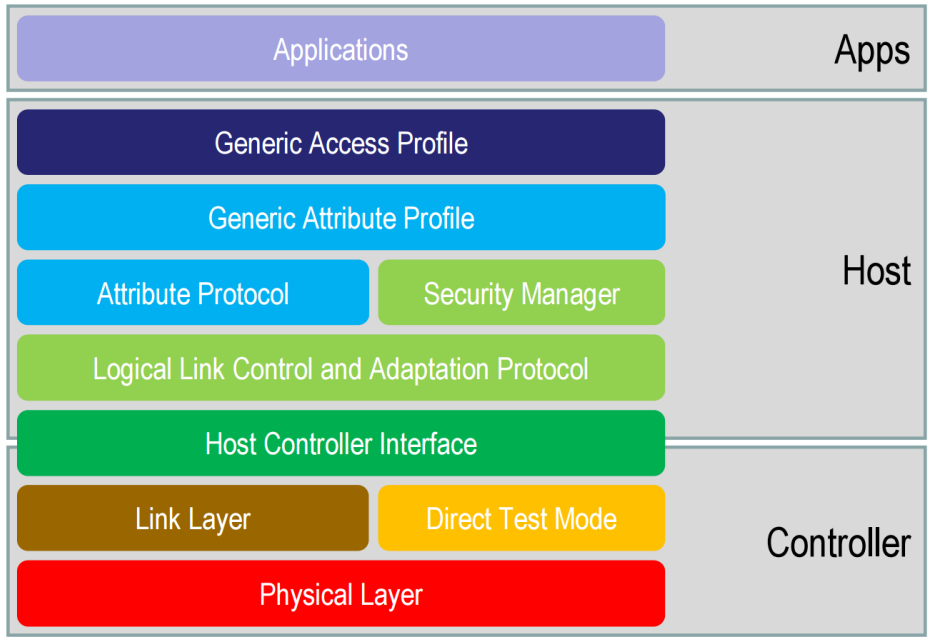
- Gives a predefined set of use case profiles based on GATT

Example:

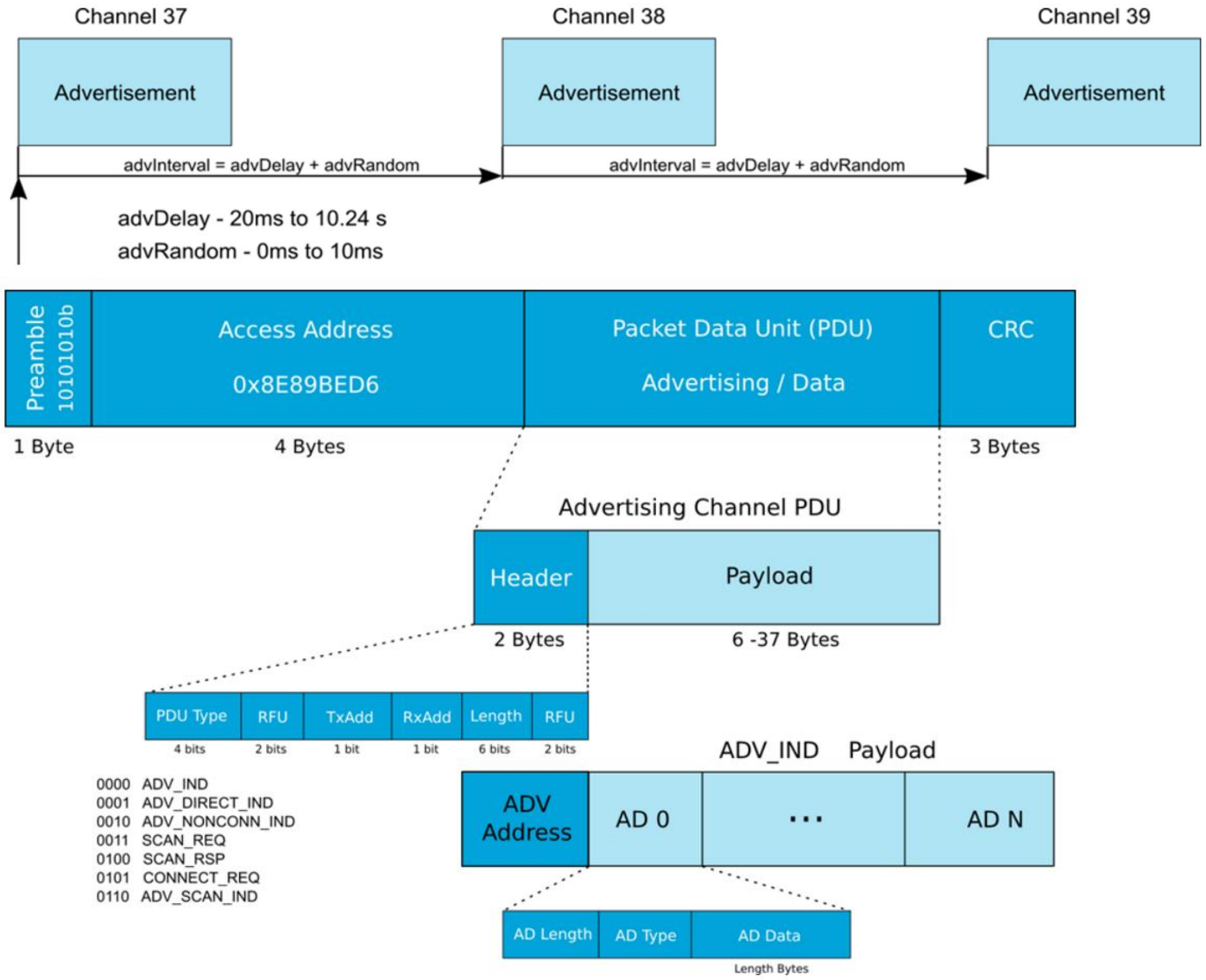
- Find Me Profile
- Proximity Profile
- HID over GATT Profile
- Glucose Profile
- Cycling Speed and Cadence Profile

Vendor Specific Profiles

- Users can define their own profiles, for cases not covered by SIG
- Such profiles can be kept private or can be published
- Example of published vendor profiles include Apple's iBeacon and Apple Notification Center Service (ANCS)



# Advertising



# Connection

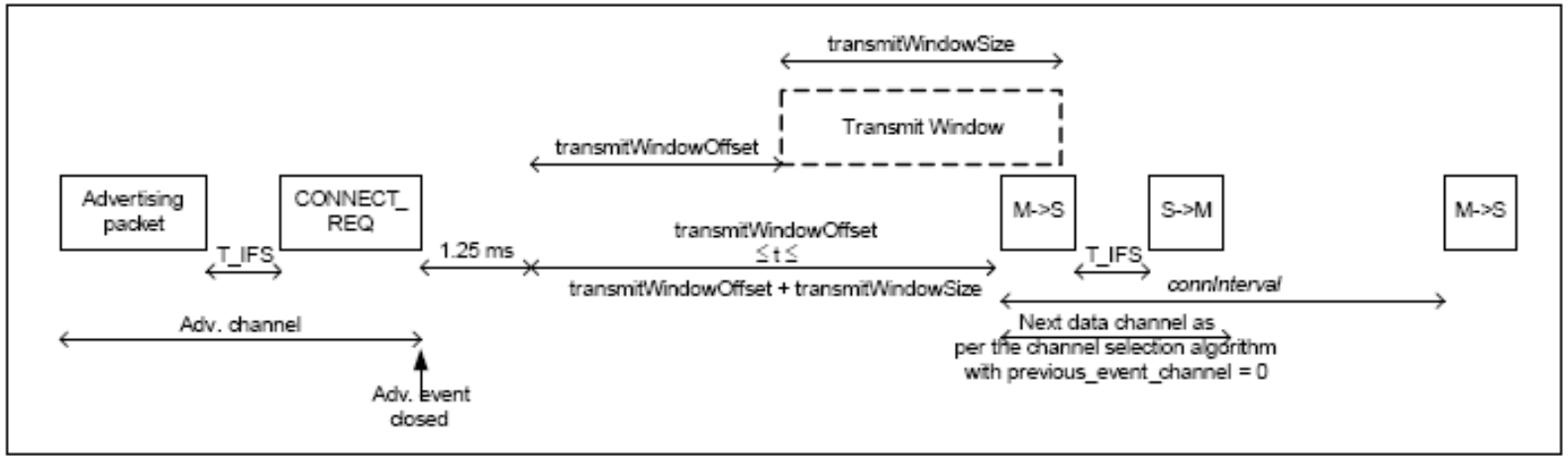



*Figure 4.11: Master's view on LL connection setup with a non-zero transmitWindowOffset*

1. connInterval: 7.5 ms to 4.0 s, = n * 1.25 ms,  n= 6~3200
2. Master decides default connection parameters to slave in CONNECT_REQ
3. Slave can request master to use appropriate parameters later
4. Master to decides what parameters to use finally

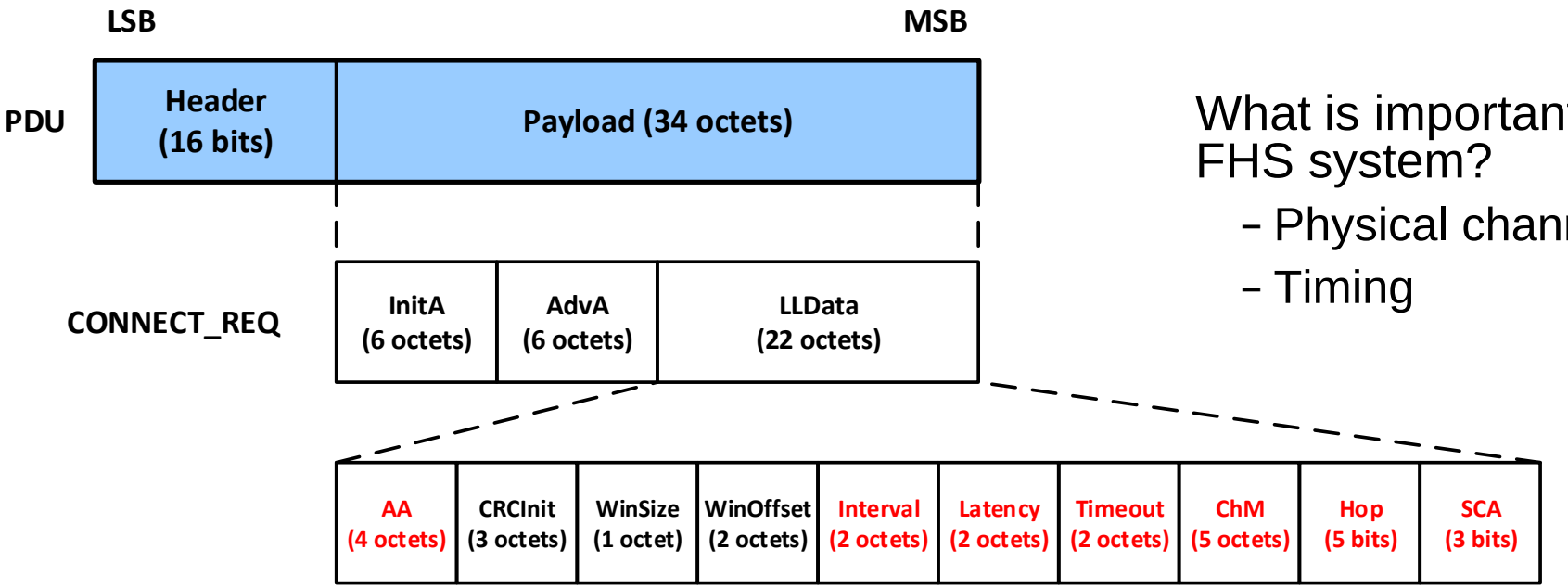
# LE Air Packet Format

| | LSB | | | MSB |
|---|---|---|---|---|
| **Packet Format** | **Preamble (1 octet)** | **Access Address (4 octets)** | **PDU (2 to n octets)** | **CRC (3 octets)** |
| **Advertising Channels** | 10101010b | 0x 8E89BED6 | Adv Channel PDU | CRC over PDU<br>Polynomial:<br>$x^{24} + x^{10} + x^9 + x^6 + x^4 + x^3 + x + 1$. |
| **Data Channels** | 10101010b (LSB of AA=0)<br>01010101b (LSB of AA=1) | A random 32-bit value, generated by Initiator, & sent in CONNECT_REQ | Data Channel PDU | |
| **Usage** | Freq sync<br>Timing sync<br>AGC training | For scanner to filter adv, or master/slave to find the specific link | | PDU integrity check |

**Max. length of PDU n:**

| | v4.0 | v4.1 | v4.2 |
|---|---|---|---|
| **Adv PDU** | 39 | 39 | 39 |
| **Data PDU** | 39 | 39 | 253 |

# CONNECT_REQ

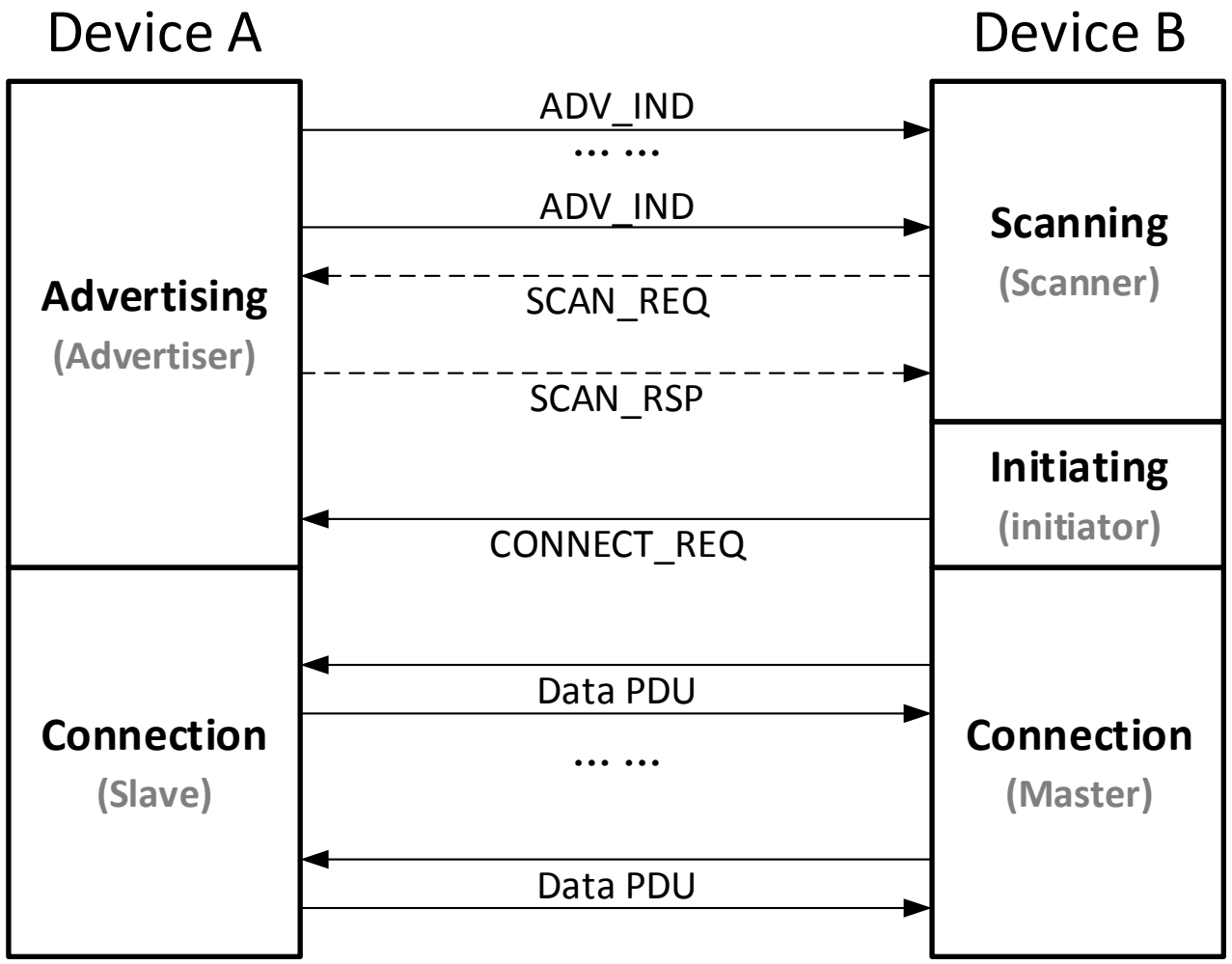



What is important to FHS system?

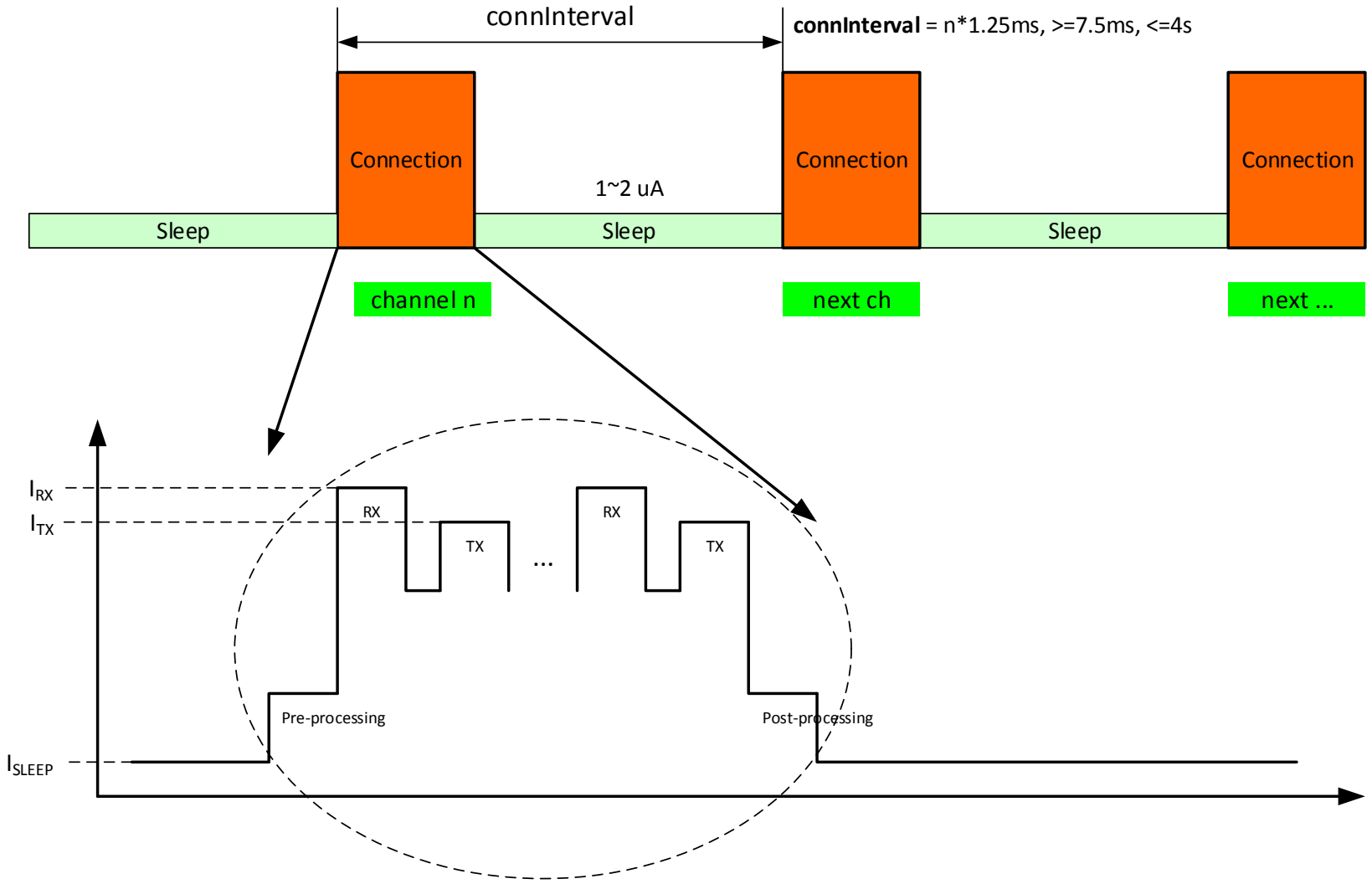
- Physical channels
- Timing

| | Name | Usage |
|---|---|---|
| Interval | Connection Interval | Timing sync between master and slave |
| Latency | Slave latency | |
| Timeout | Supervision timeout | |
| SCA | Sleep Clock Accuracy | Sleep clock accuracy of master to slave |
| ChM | Channel Map | Physical channels to be used for this link |
| Hop | Hop increment | Channel hopping increment |

# Message Sequence Chart



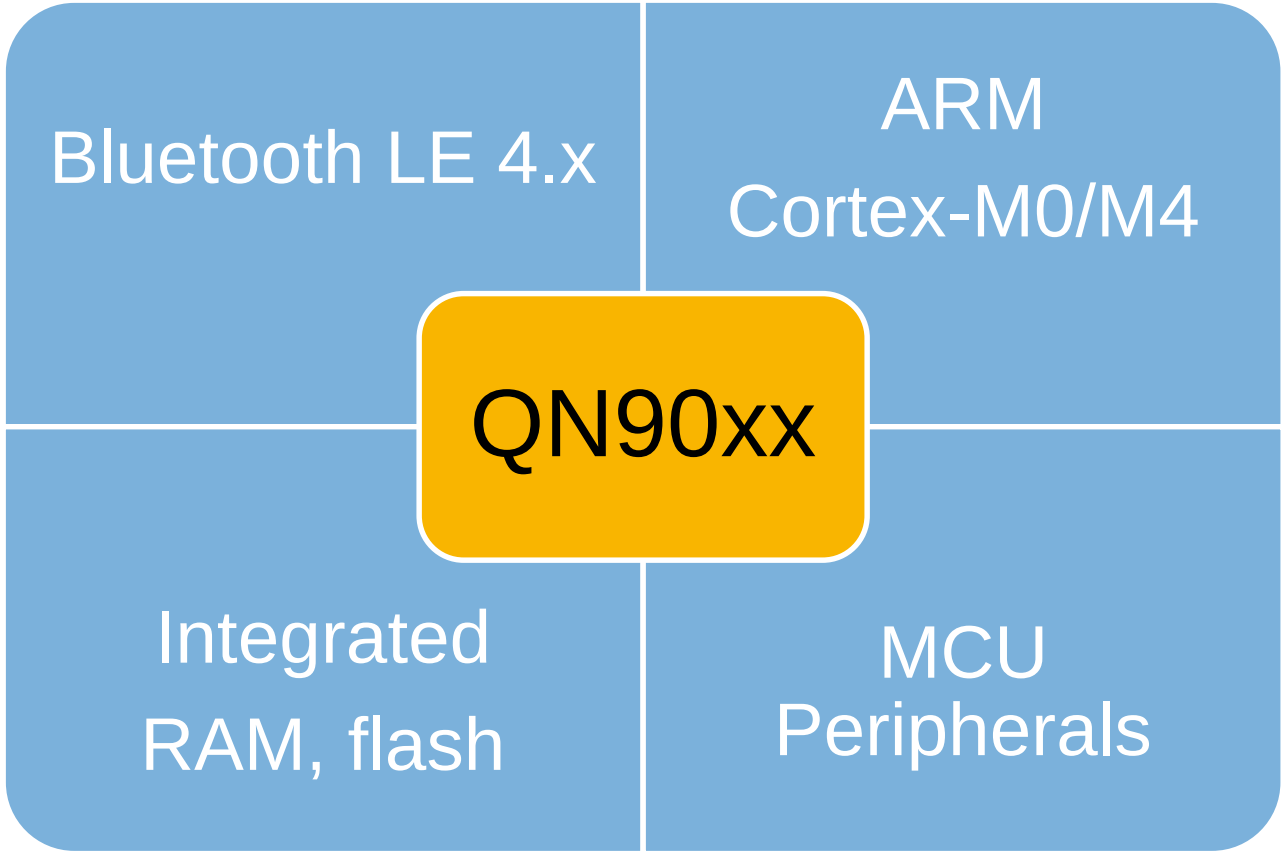
# Ultra Low Power Operation



# Design Resource

- Bluetooth SIG
  - Spec: https://www.bluetooth.org/en-us/specification/adopted-specifications
  - Listing: https://www.bluetooth.org/tpg/listings.cfm
- iOS Core Bluetooth Programming Guide
  - https://developer.apple.com/hardwaredrivers/BluetoothDesignGuidelines.pdf
  - https://developer.apple.com/library/ios/documentation/CoreBluetooth/Reference/CoreBluetooth_Framework/index.html
- Android BLE API
  - http://developer.android.com/guide/topics/connectivity/bluetooth-le.html
- Win 8 BLE API
  - https://msdn.microsoft.com/en-us/library/windows/hardware/ff536596(v=vs.85).aspx

# QN90xx – Ultra Low Power Wireless SoC Platform



# Product Roadmap


***QN908x***
**Wireless SoC with Cortex-M4F**
**Sensor Fusion**
**Bluetooth Smart (4.2)**


***QN902x***
**Wireless SoC with Cortex-M0**
**Bluetooth Smart (4.1)**

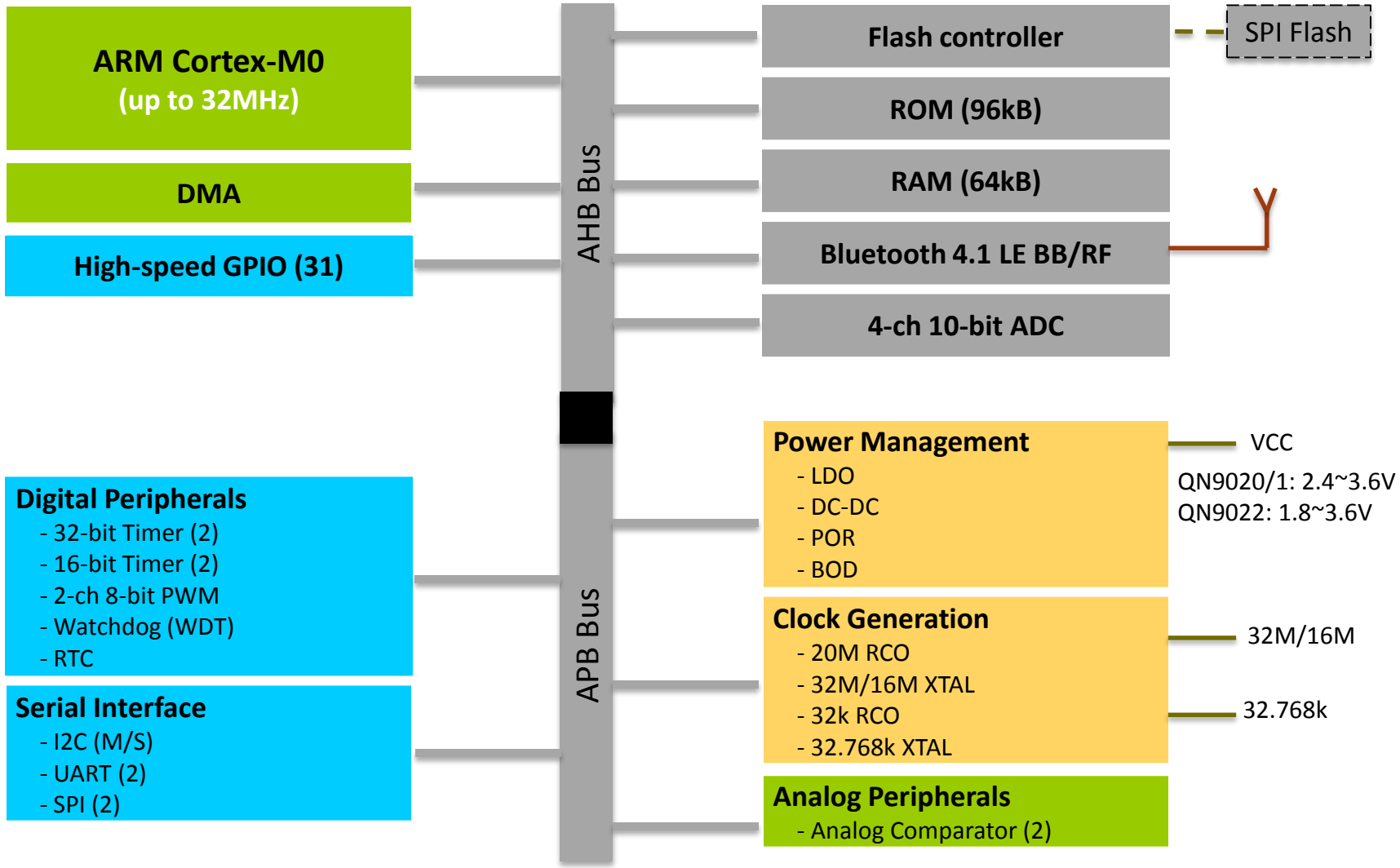

***QN903x***
**Optimized**
**Bluetooth Smart (5.0)**

| Q3 | Q4 | Q1 | Q2 | Q3 | Q4 | Q1 | Q2 | Q3 | Q4 | Q1 | Q2 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2014 | | 2015 | | | | 2016 | | | | 2017 | |

# NXP BLE – Technical Specs

| | QN902x | QN908x |
|---|---|---|
| **Status** | MP | MP by Q3/16 |
| **Package** | QN9020: 6x6 HVQFN48<br>QN9021: 5x5 HVQFN32<br>QN9022: 5x5 HVQFN40 | QN9080: 6x6mm HVQFN48<br>QN9083: WLCSP |
| **Standard support** | BLE 4.1 | BLE 4.2 |
| **MCU** | Cortex-M0 | Cortex-M4 with FPU |
| **Signal processing** | No | HW accelerator - FSP |
| **Tx power** | -20~4dBm | -20~2dBm |
| **Rx sensitivity** | -95dBm | -95dBm |
| **NVM** | 128kB SPI flash, 96kB ROM | 512kB eflash, 256kB ROM |
| **RAM** | 64kB | 128kB |
| **Code execute** | In RAM | In flash or RAM |
| **Max. MCU speed** | 32MHz | 32MHz |
| **BLE Peak Power** | 9mA at 3V supply | 3.5mA at 3V supply |
| **Operation temp** | -40~85 C | -40~85 C |

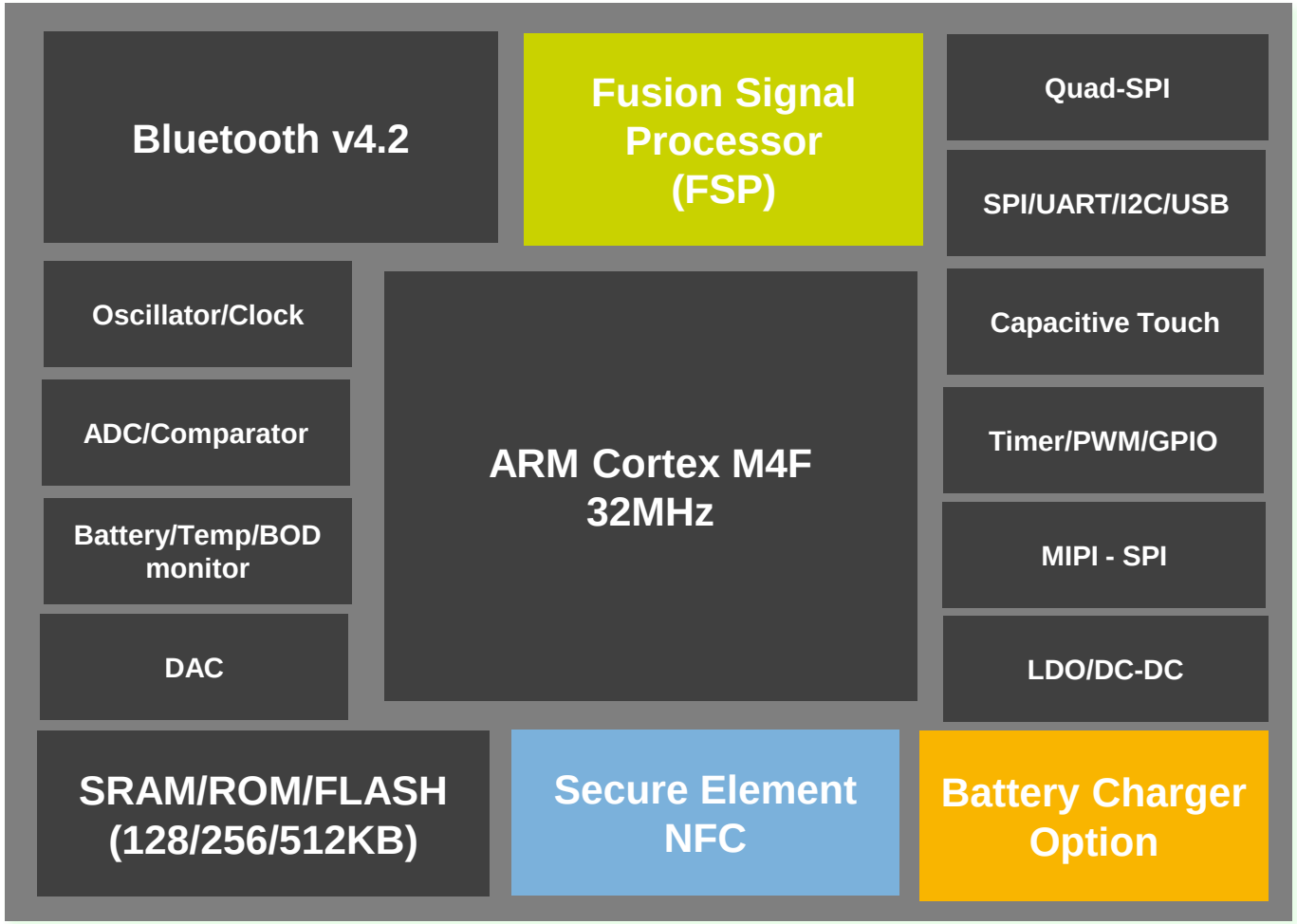
# QN902x System Architecture



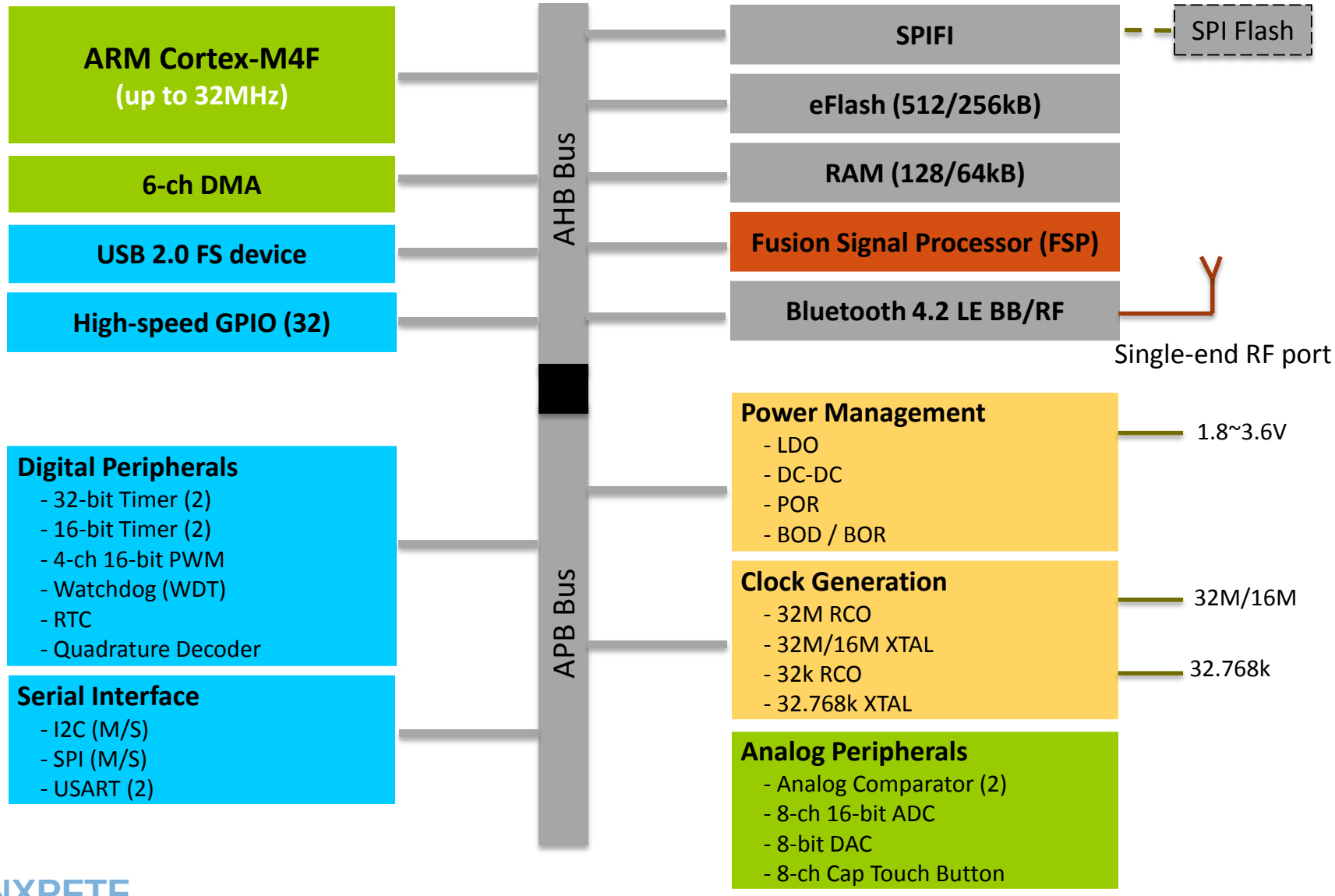
# RF Performance and Current Consumption

| | Condition | Result |
|---|---|---|
| RX sensitivity | High performance mode | -95dBm |
| | Low power mode | -93dBm |
| TX output power | | -20~4dBm |
| RX current | LDO mode at 3V supply (-95dBm sensitivity) | 13.6mA |
| | DC-DC mode at 3V supply (-93dBm sensitivity) | 9.25mA |
| TX current | 0dBm Tx power, LDO mode at 3V supply | 13.3mA |
| | -4dBm Tx power, LDO mode at 3V supply | 10.5mA |
| | 0dBm Tx power, DC-DC mode at 3V supply | 8.8mA |
| | -4dBm Tx power, DC-DC mode at 3V supply | 6.9mA |
| Deep Sleep Mode | Wakeup using external interrupt, clocks off, at 3V supply | 2uA |
| Sleep Mode | Wakeup using Timer, 32kHz sleep timer, at 3V supply | 3uA |
| Active current | 8MHz clock, execute in RAM | 1.35mA |

# QN908x – Ultra Low Power Wireless SoC Platform



# QN9080 System Architecture



# NXP QN9080 Introduction

## General System

- Single chip multi-mode wireless SoC optimized for wearable applications
- 6x6 QFN48 and 3.2x3.2 WLCSP package address different form factor options
- 10mW peak system power set the new industry standard, enables energy harvest in wearable applications
- FSP sensor hub co-processor improves by 5 ~ 10 times computation speed of sensor fusion and machine learning algorithm

# QN9080 Key Solution and Feature Differentiation

- Multilink concurrent master/slave roles with up to 8 simultaneous connections, enables true BLE mesh solutions for IoT and social network applications
- Fully integrated BLE v4.2 with unique high throughput scheme enables true BLE based voice/audio steaming solutions
- Secure and reliable OTA ensure the integrity and future proof solutions
- FSP low power computation or more advanced algorithm based wearable solution enables more accurate and reliable user experience
- Fully integrated A4WP application enables wireless charging solution
- Fully integrated AoA/AoD together with Beacon capability enables a high precision indoor location solution
- Integrated security engine and NFC connectivity enables mobile payment/wallet, convenient transportation and authentication solution

# NXP QN9080: Key Radio and Electrical Specification

| Items | Specs | Condition |
|---|---|---|
| Vcc (V) | 1.8 ~ 3.6 | |
| Peak current (mA) | 3.3 | Vcc = 3V, TX/RX +MCU active |
| TX Max Pout (dBm) | 2 | |
| RX sensitivity (dBm) | -94 | DC/DC on |
| CCI (dB) | 4 | |
| N+/-2 channel Interference (dB) | 40 | |
| MCU current (uA/MHz) | 45 | Program run from SRAM (DC/DC disabled), while (1), Vcc = 1.8V |
| | 65 | Program run from eFlash (DC/DC disabled), while (1), Vcc = 1.8V |
| Sleep current (uA) | 2 | 32XO + timer+ Retention memory |
| Deep sleep current (uA) | 1 | Retention memory |
| ADC ENOB | 16 | 128Hz sample, 8 channel |

# NXP QN9080 BT LE Benchmark

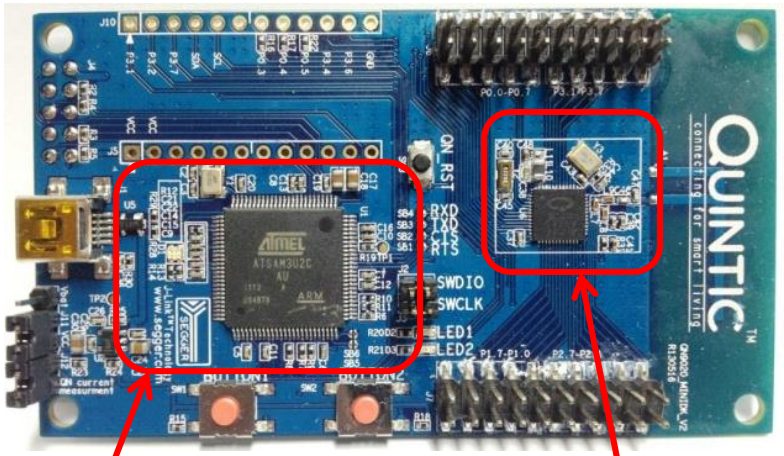
- Average current is critical to battery life, QN9080 BT LE link average current is at least 40% lower than the leading low power combination in the market
- For a simple BT LE device, QN9080 based solution will extend the battery life by more than 40%

| | QN9080 | DA14580 | DA14580 + EFM32 | DA14580 + STM32 |
|---|---|---|---|---|
| Peak current (mA) | 3.3 | 5.0 | 5.4 | 5.4 |
| Average current (uA) | 4.0 | 6.75 | 7.75 | 8.05 |

# QN902X HARDWARE INTRODUCTION

# QN9020 Development Kit (available in HK Sample Store)

## MINI DK



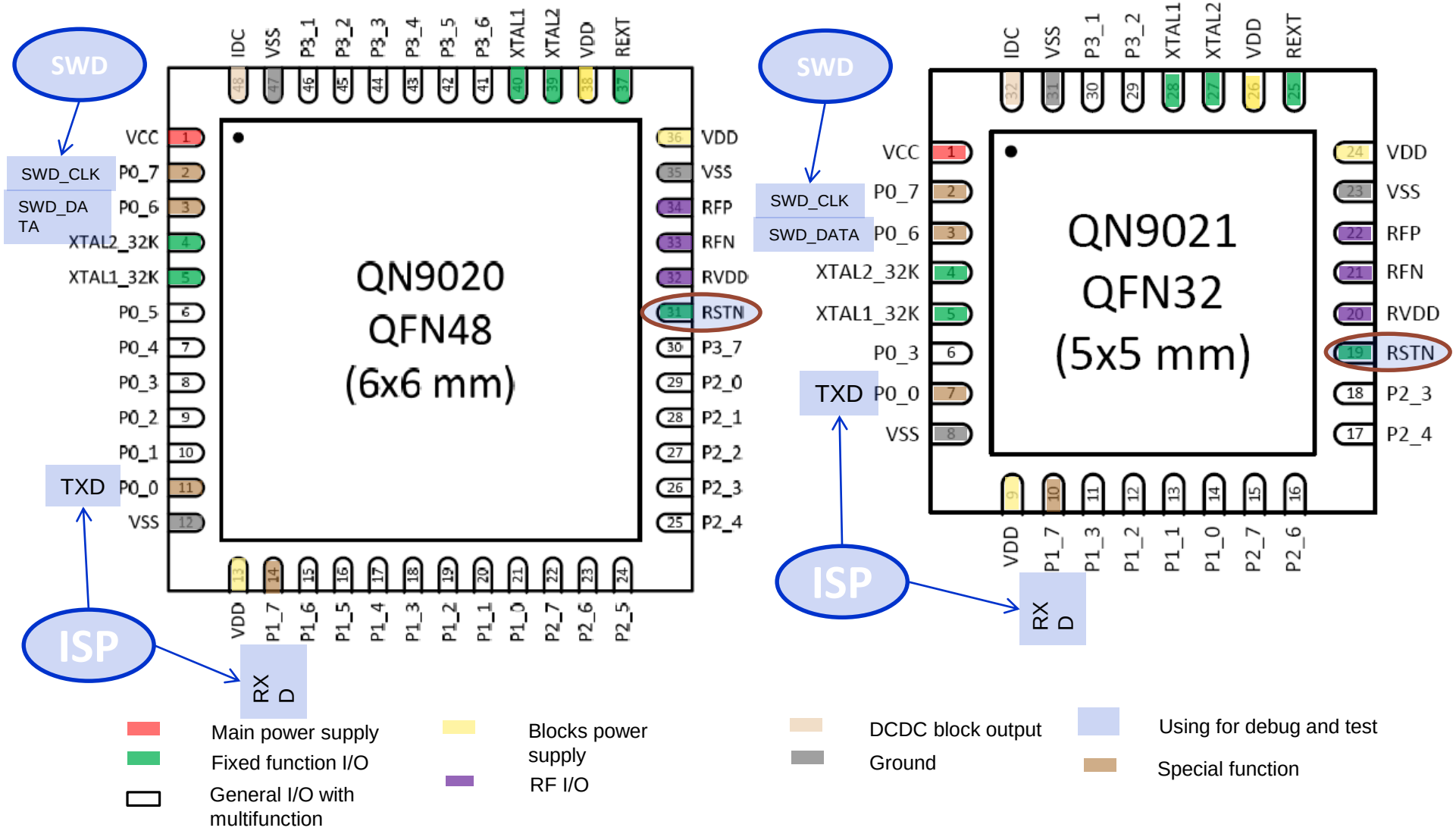

J-Link onboard debugger

QN9020 & external circuit

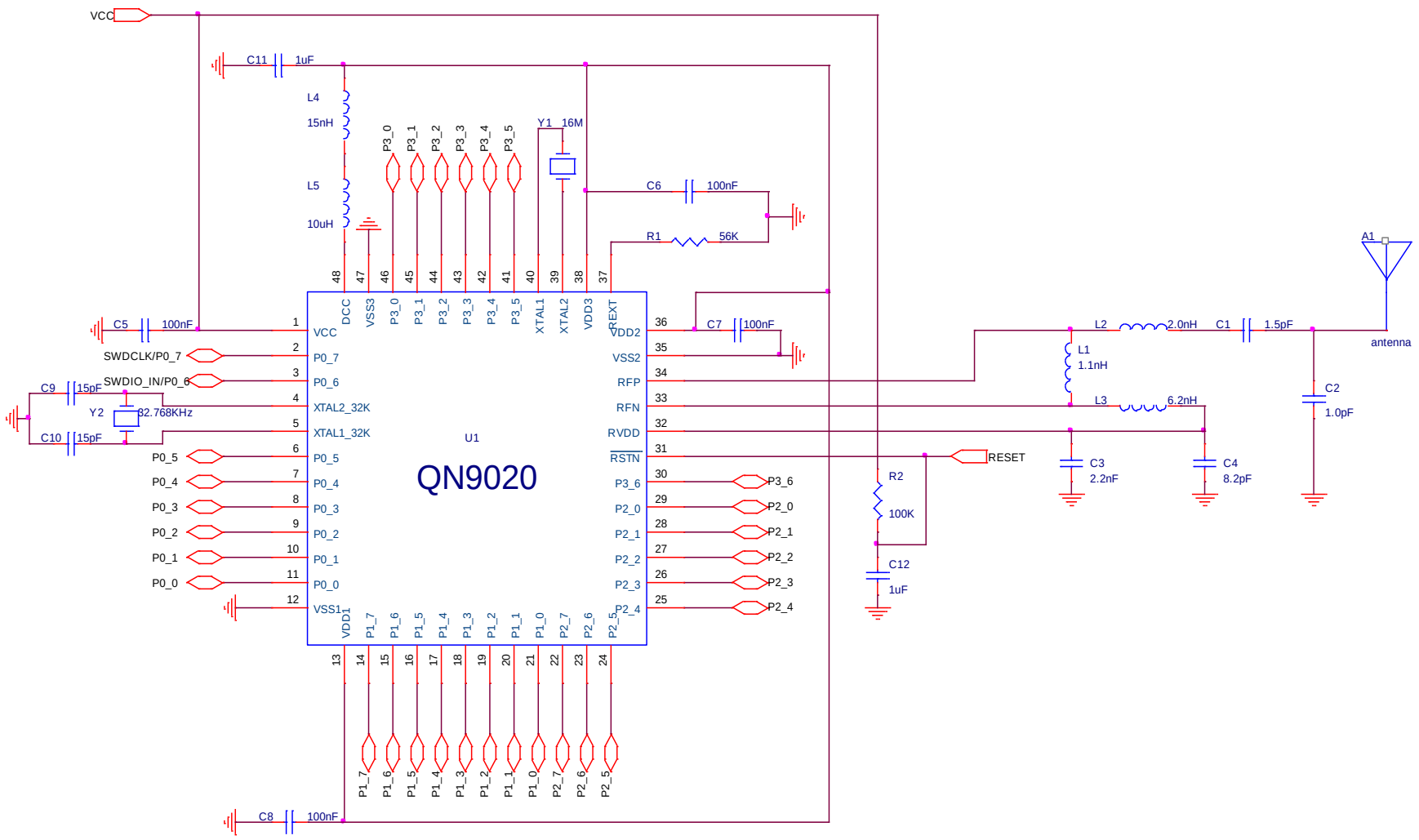
## BLE dongle

Act as a BLE peer device, controlled by QTool on PC for development

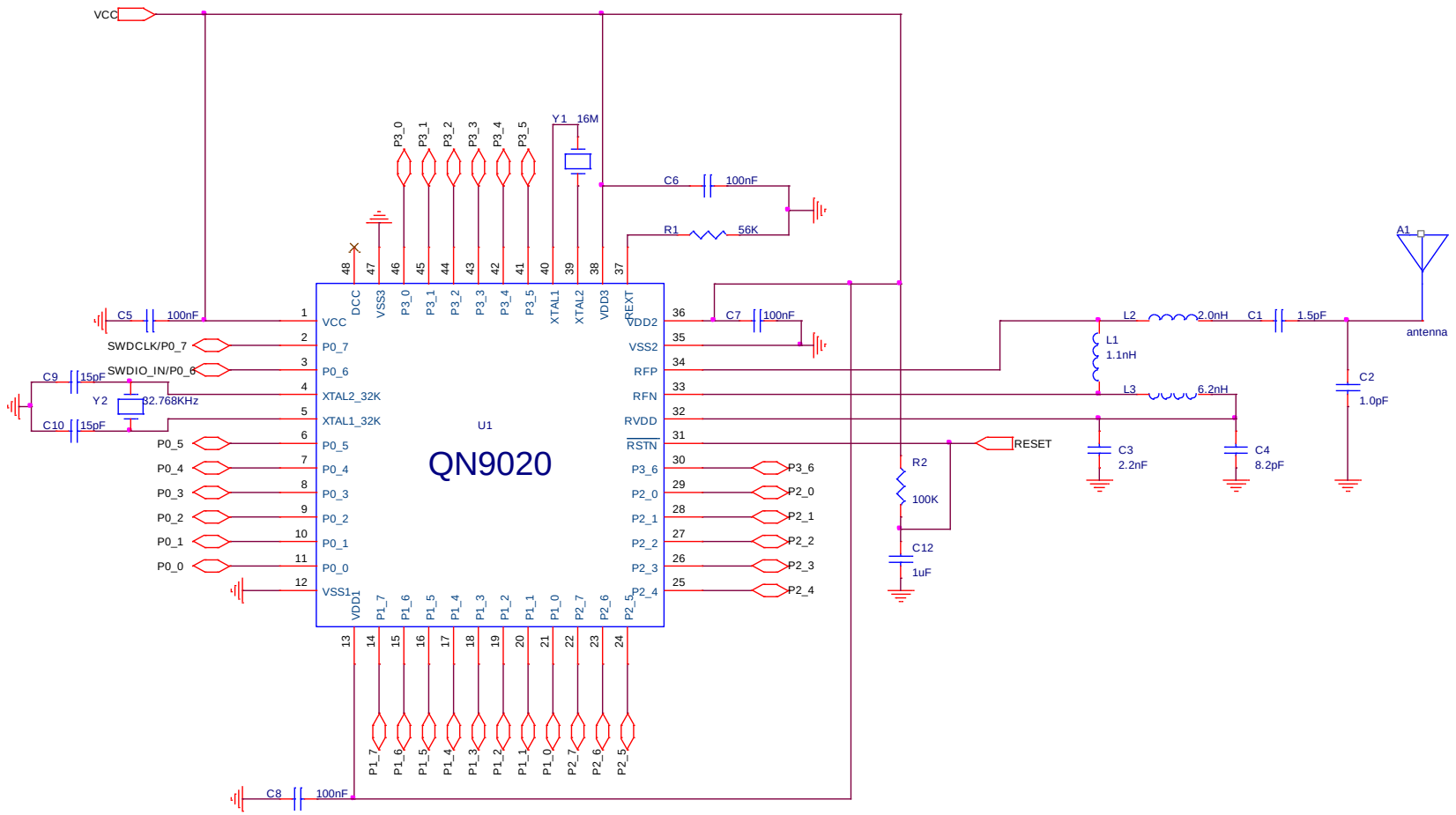
# QN902x Hardware Design



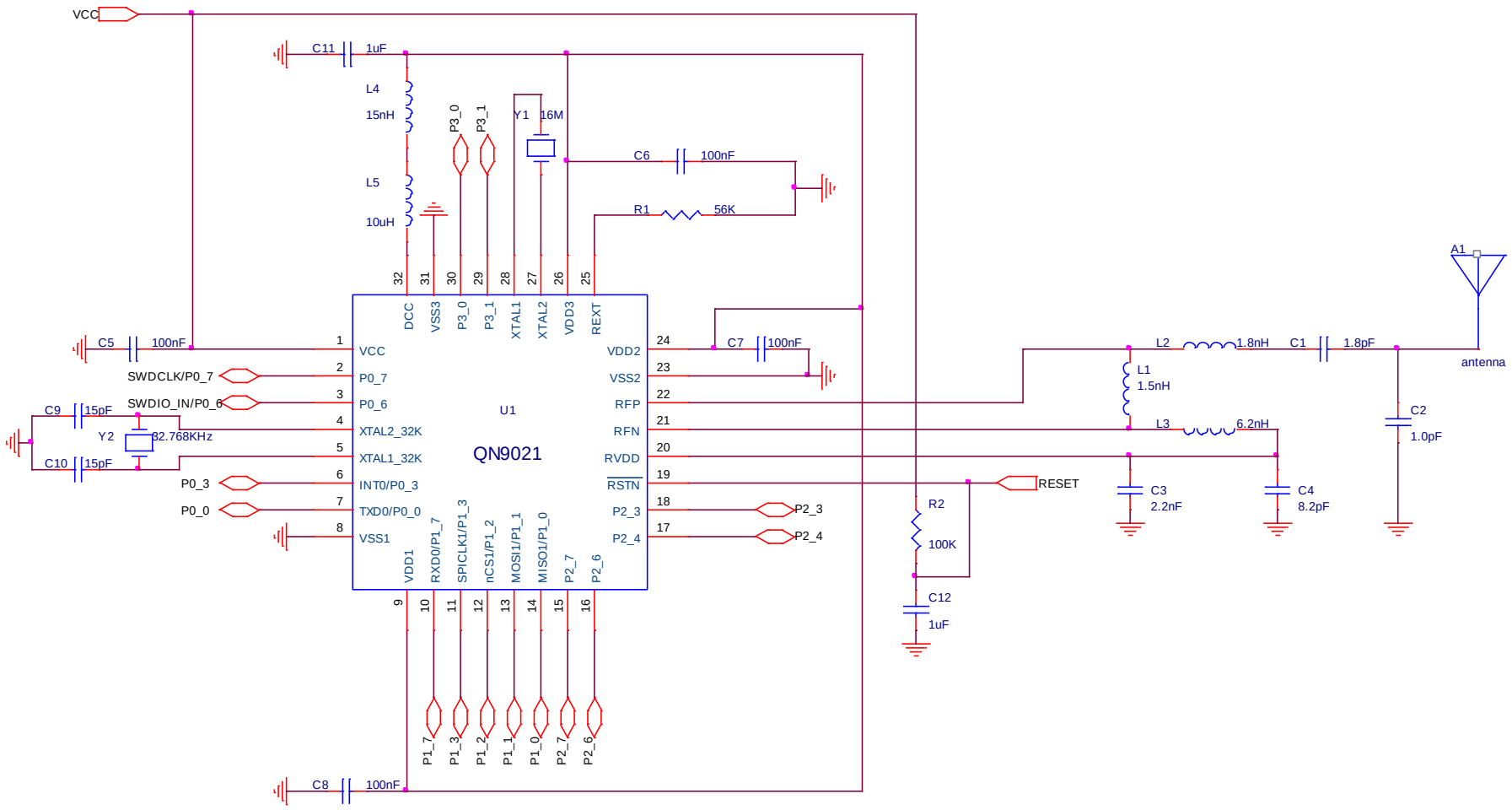
# QN902x Hardware Design



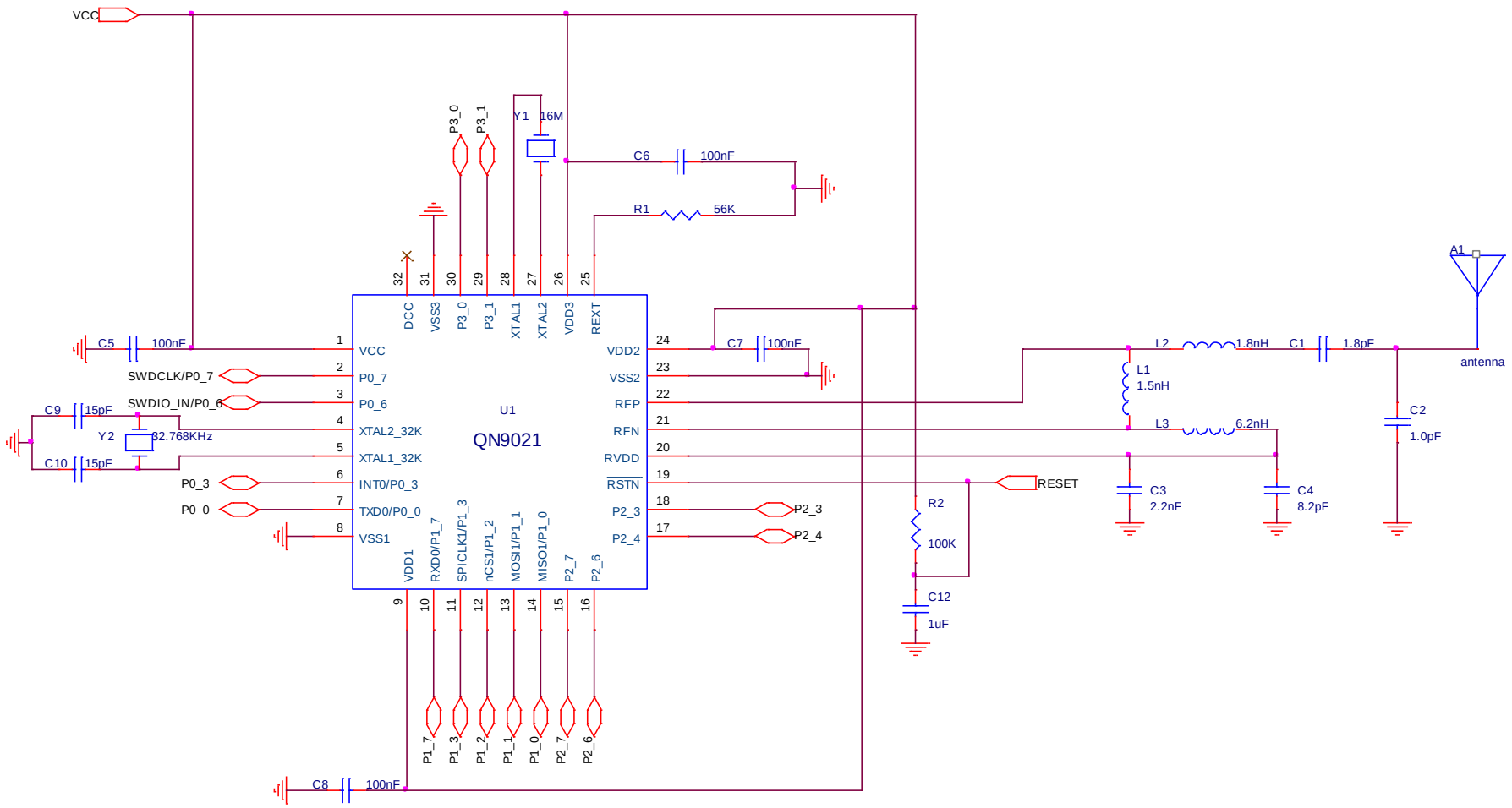
# QN902x Hardware Design



# QN902x Hardware Design

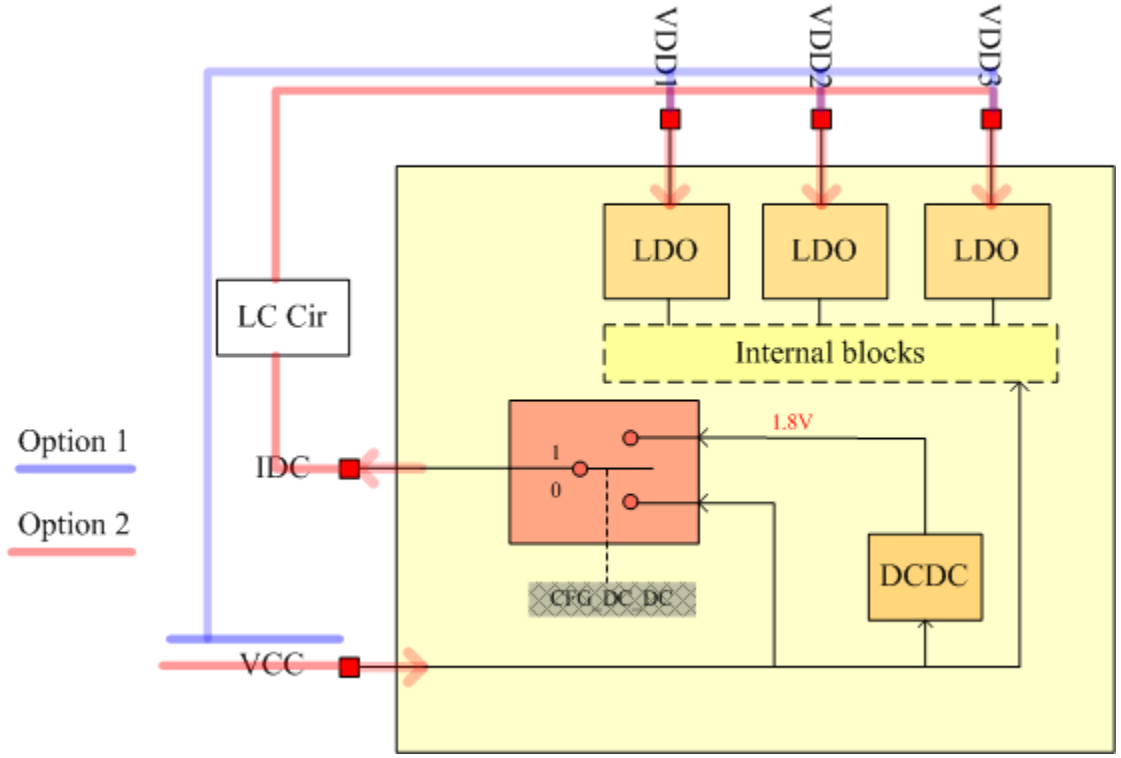


# QN902x Hardware Design



# QN902x Hardware Design

## Power connection selection




### Option 1:

**Using without internal DCDC block and SWITCH block**

In this option, all the power pins (VCC, VDD1, VDD2, VDD3) should connect to external power supply directly, this is (non DCDC mode).

### Option 2:

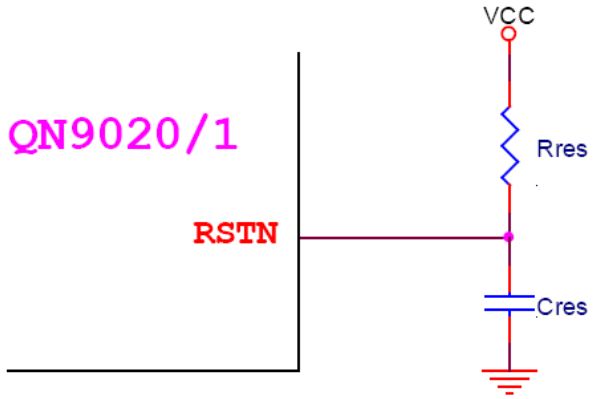
**Using with internal SWITCH block**

a. If in firmware you have defined “CFG_DC_DC” using internal DCDC, the VCC should connect to the external power supply directly, the VDD1, VDD2 and VDD3 can get the 1.8V power supply from IDC / DCC pin where you have a LC loop filter for the DCDC converter. (this configuration is called DCDC enabled)

b. If in firmware you haven’t defined “CFG_DC_DC” it will not use internal DCDC and the switch will connect the IDC pin to VCC pin directly. It will cause the same connection result as option 1 (no dcdc)

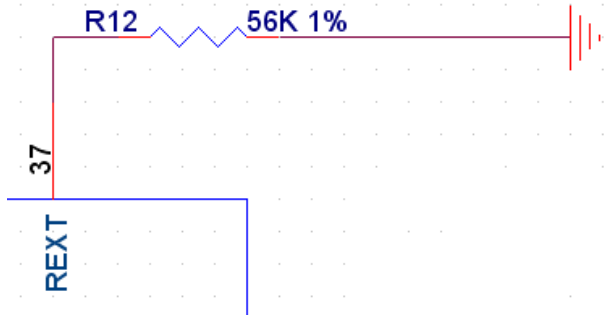
# QN902x Hardware Design

## RESET circuit




**RESET is active low, a RC circuit is proposed as shown in the figure**

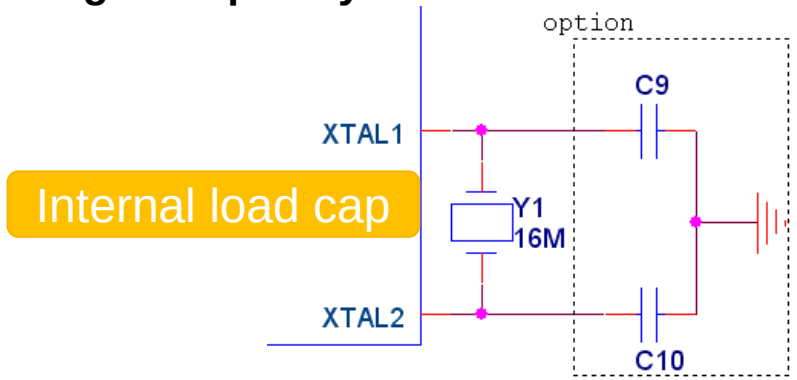
## External reference resister




**An accurate resistor is needed at pin Rext, for reference generation. The value of the resister is 56k ohm and it should have 1% precision**

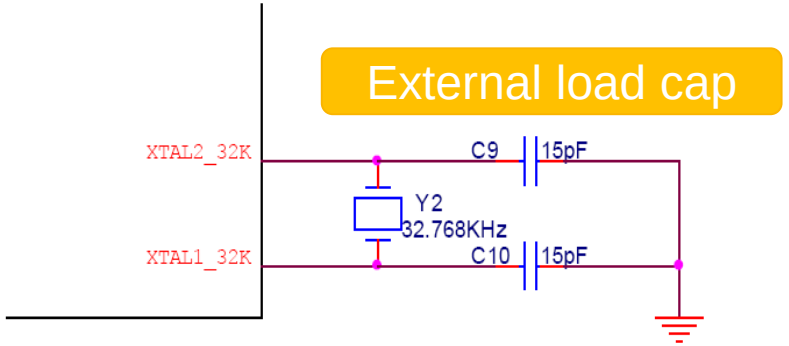
# QN902x Hardware Design

## Crystal Selection
## High Frequency Clock




| Frequency | Accuracy | Load capacitance | |
|---|---|---|---|
| 16/32 MHz | < ±50ppm | typical | maximum |
| | | 12pF | 15pF |

## Low frequency clock

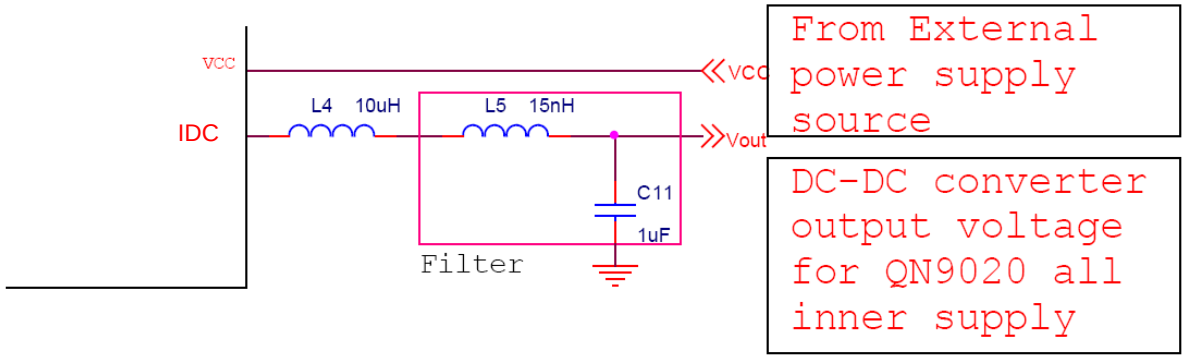



| Frequency | Accuracy |
|---|---|
| 32.678kHz | < ±100ppm |

Note: Crystal accuracy is proportional to the RF frequency accuracy. Using a +/-20ppm accuracy on the crystal is recommended, to get maximum RX response and connection stability

# QN902x Hardware Design

**DCDC block circuit**

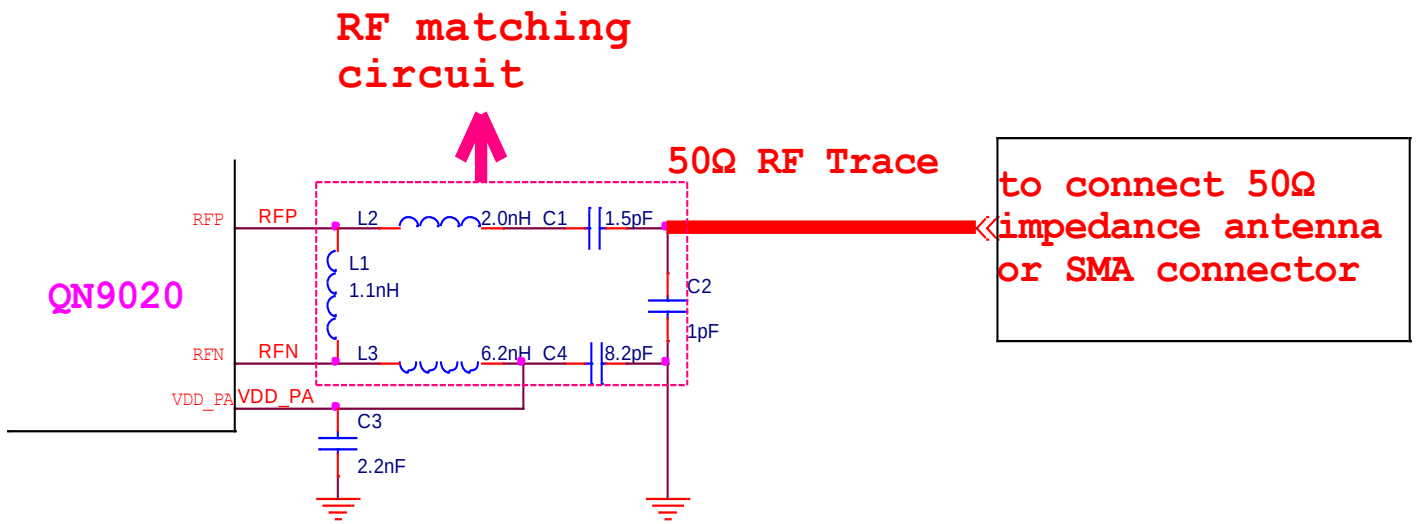



The IDC output should connect the 10uH inductor and the 1.5nH inductor in series for DCDC voltage converting and filtering. Also need a 1uF capacitor in parallel for decoupling.

DCDC converter creates noise and in order to reduce the noise, the components should be placed as close to the IC as physically possible.

# QN902x Hardware Design

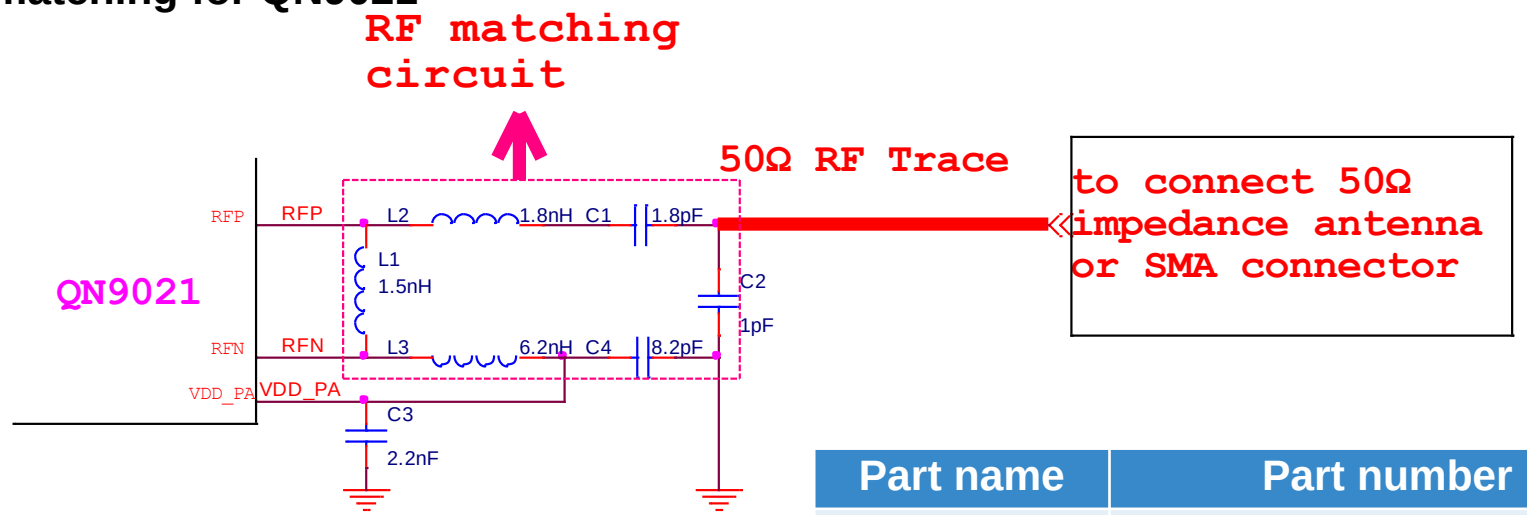
## RF matching for QN9020




| Part name | Part number | Value | Size |
|---|---|---|---|
| Inductor L1 | LQP15MN1N1B02 | 1.1nH | 0402 |
| Inductor L2 | LQP15MN2N0B02 | 2.0nH | 0402 |
| Inductor L3 | LQP15MN6N2B02 | 6.2nH | 0402 |
| Capacitor C1 | GRM1555C1H1R5CA01 | 1.5pF | 0402 |
| Capacitor C2 | GRM1555C1H1R0CA01 | 1.0pF | 0402 |
| Capacitor C4 | GRM1555C1H8R2DA01 | 8.2pF | 0402 |
| Capacitor C3 | GRM155R71H222KA01 | 2.2nF | 0402 |

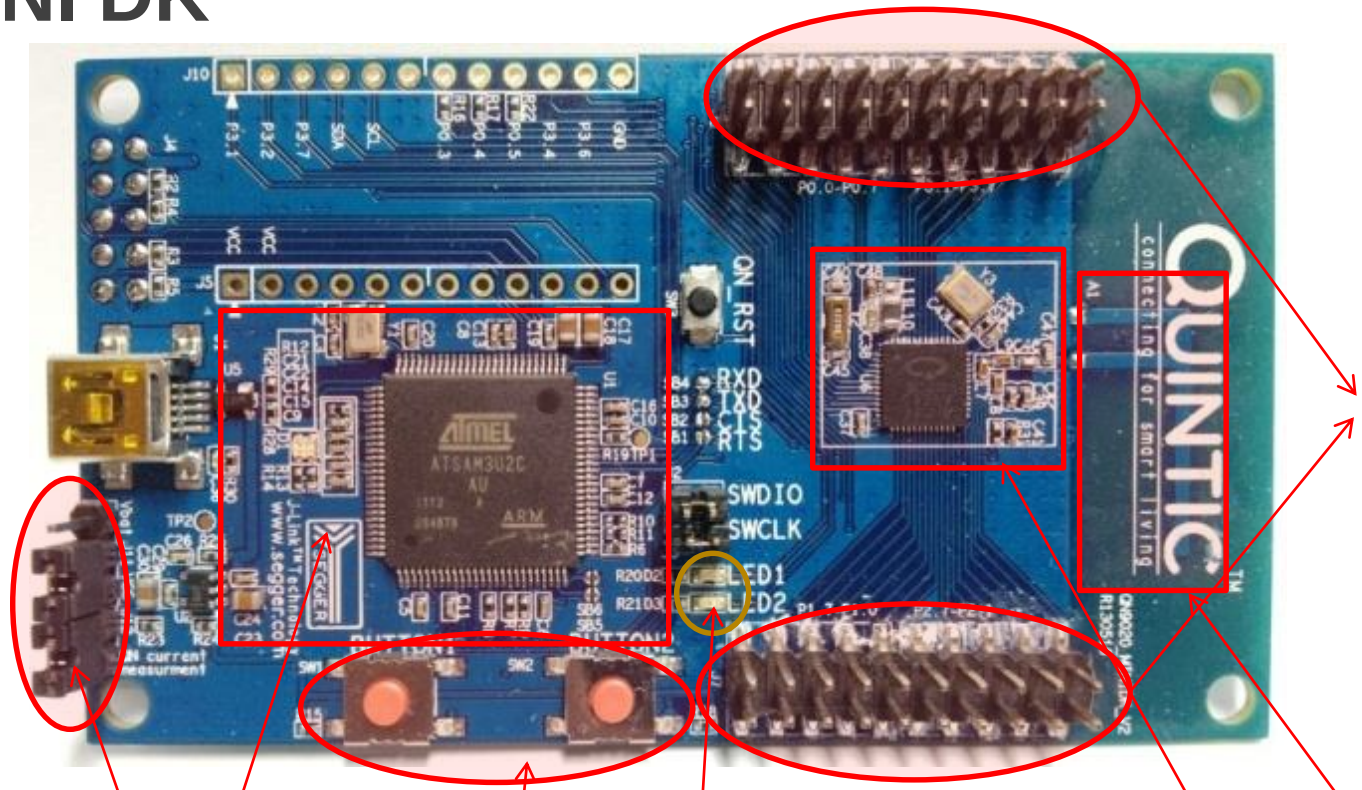
# QN902x Hardware Design

## RF matching for QN9021




| Part name | Part number | Value | Size |
|---|---|---|---|
| Inductor L1 | LQP15MN1N5B02 | 1.5nH | 0402 |
| Inductor L2 | LQP15MN1N8B02 | 1.8nH | 0402 |
| Inductor L3 | LQP15MN6N2B02 | 6.2nH | 0402 |
| Capacitor C1 | GRM1555C1H1R8CA01 | 1.8pF | 0402 |
| Capacitor C2 | GRM1555C1H1R0CA01 | 1.0pF | 0402 |
| Capacitor C4 | GRM1555C1H8R2DA01 | 8.2pF | 0402 |
| Capacitor C3 | GRM155R71H222KA01 | 2.2nF | 0402 |

# QN902x MINI DK




| | | | |
|---|---|---|---|
| GND | VCC | P2.0 | P2.1 |
| P3.7 | P3.6 | P2.2 | P2.3 |
| P3.5 | P3.4 | P2.4 | P2.5 |
| P3.3 | P3.2 | P2.6 | P2.7 |
| P3.1 | RESET | GND | VCC |
| GND | VCC | P1.0 | P1.1 |
| P0.7 | P0.6 | P1.2 | P1.3 |
| P0.5 | P0.4 | P1.4 | P1.5 |
| P0.3 | P0.2 | P1.6 | P1.7 |
| P0.1 | P0.0 | GND | VCC |

Jlink adapt

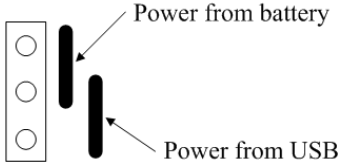
Embedded Antenna

LED

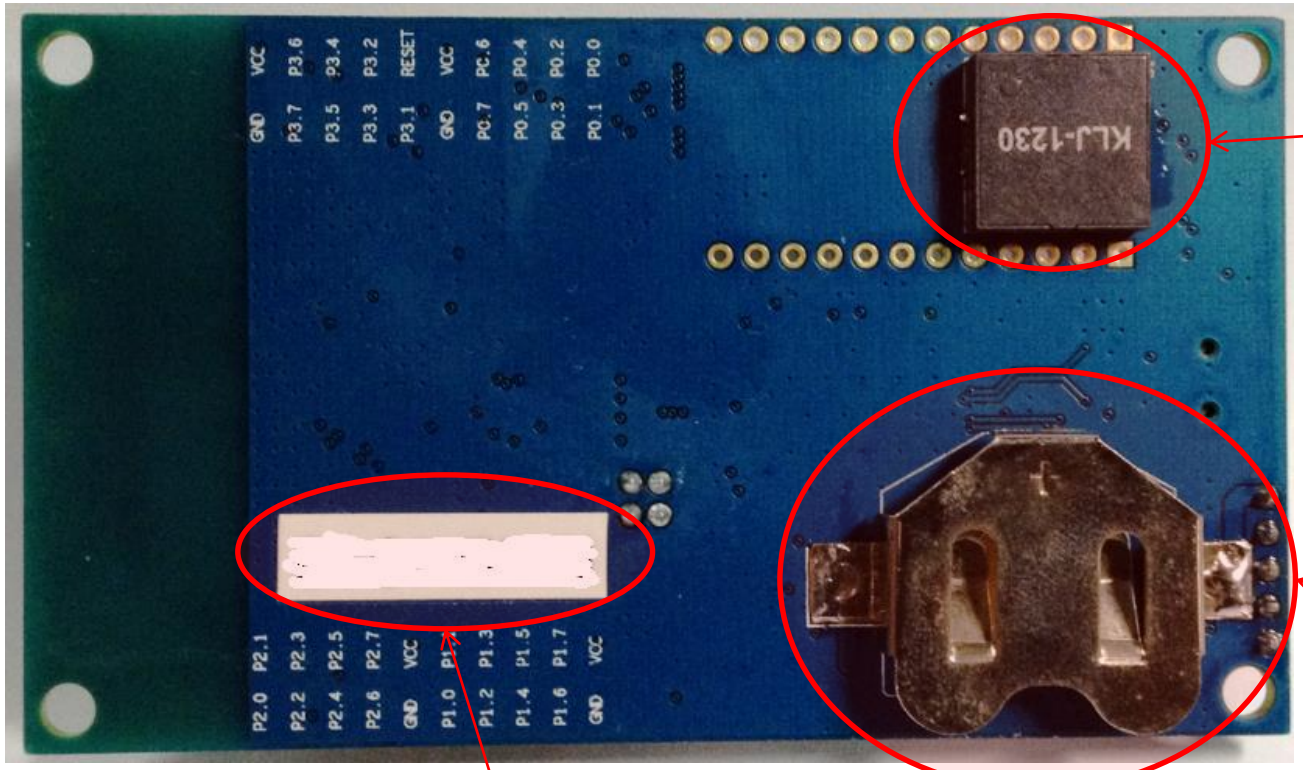
QN9020

button

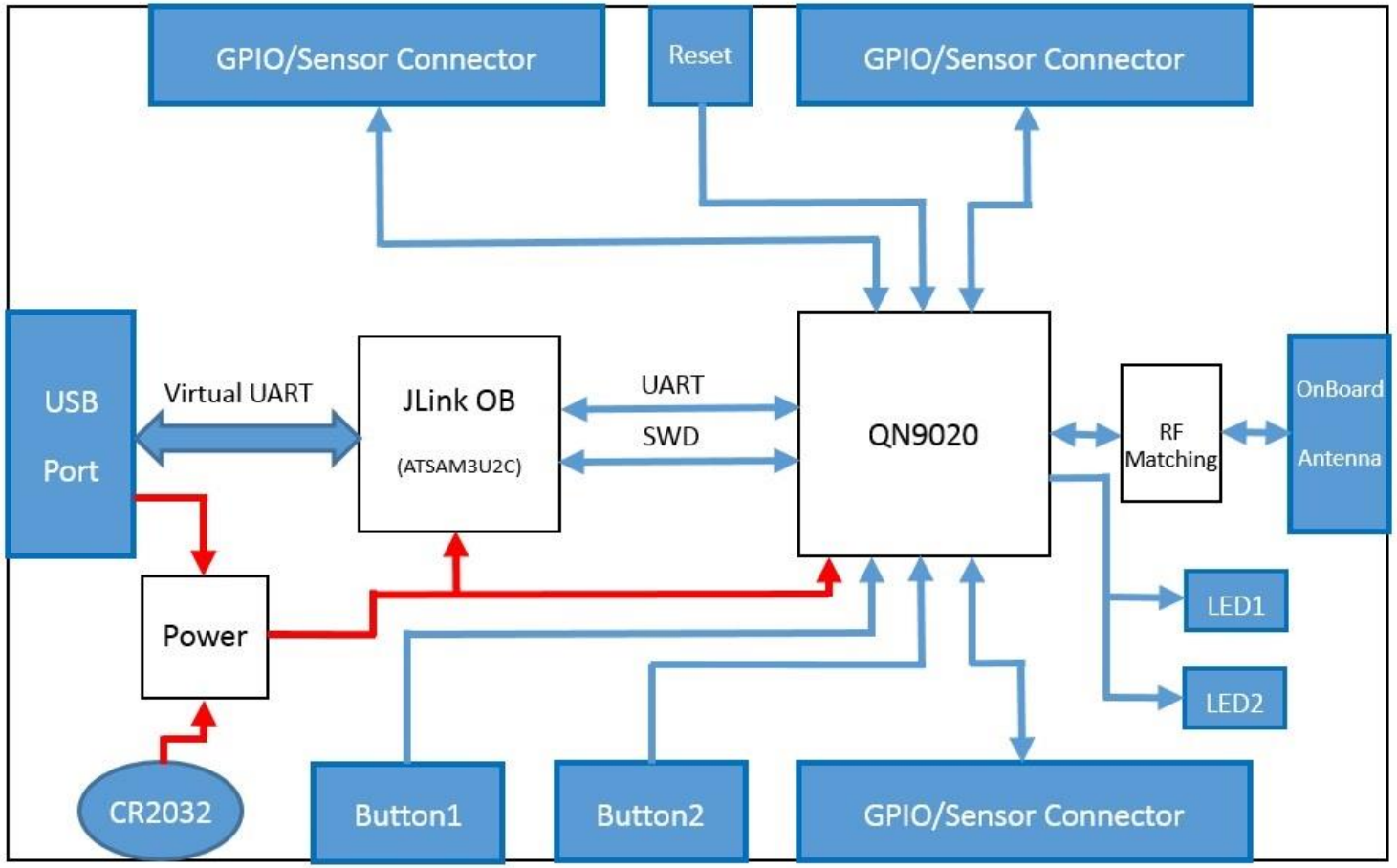
Power source selection



# QN902x MINI DK



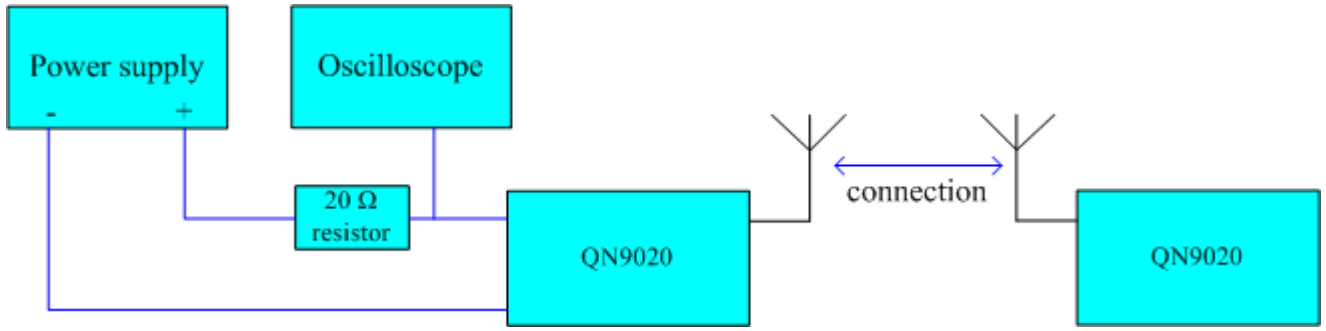
# QN902x Mini DK(Slave) -- Functional Diagram



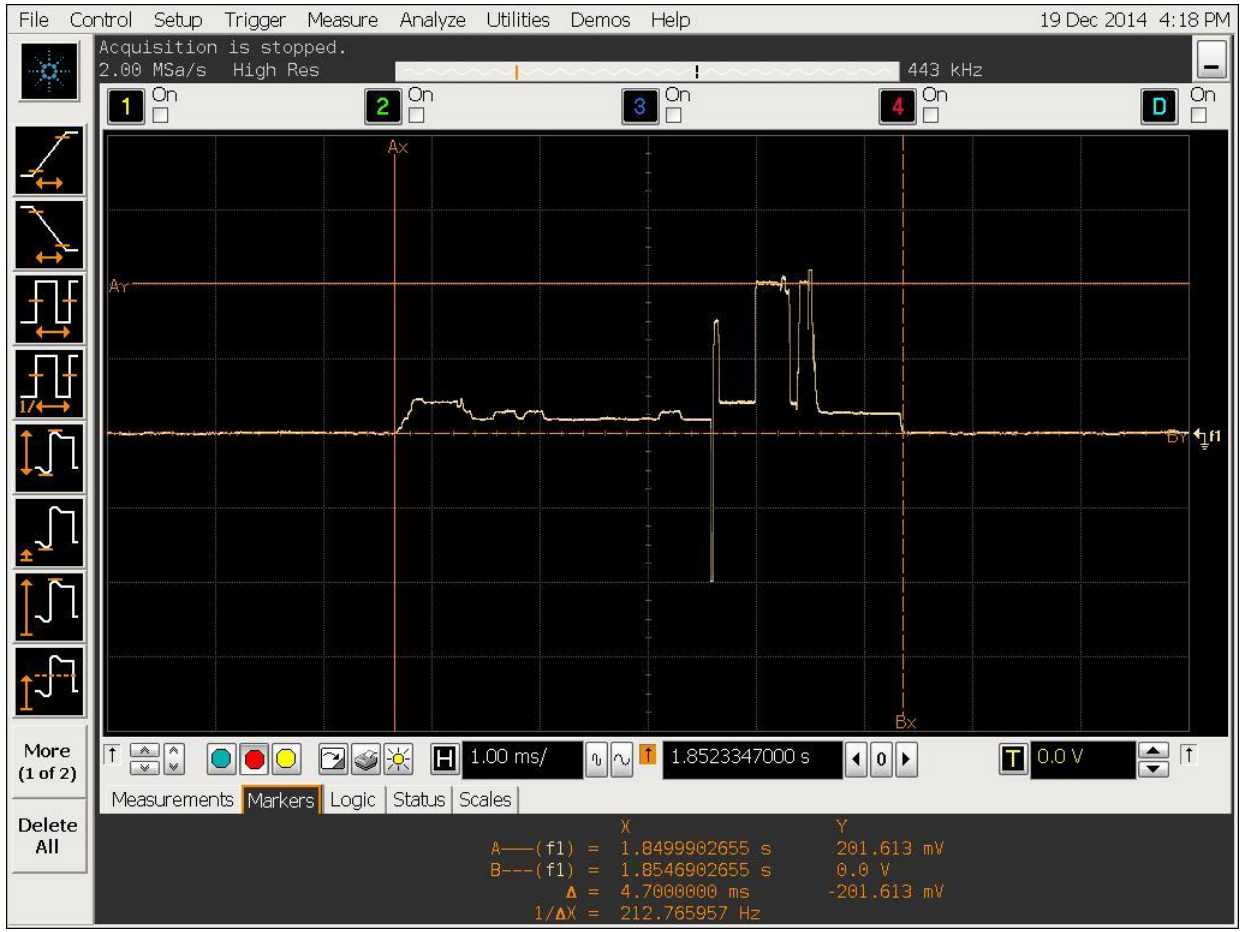
# QN902x Current Consumption Test

## Measurement Methods

- BLE event average current measured using Oscilloscope across 22.1 Ohm (Measured) series resistor
- Sleep current measured using Agilents DMM 34410A
- The Resistor value was measured using Fluke 175 True RMS Multimeter
- Following table shows the measured values and the respective calculations



# Test case 1.28sec - 23bytes (connected)



# Average Current Test Results

| Test Equipment | |
|---|---|
| Oscilloscope | MSO 9254A |
| Agilent DMM | 34410A |
| Fluke Multimeter | 175 True RMS |
| Quintic Motherboard | MB036 |
| Quintic EVB (QN9020) | QN152 |
| | QN153 |
| | QN170 |
| | QN175 |

***Measurement Methods***

*BLE event average current measured using Oscilloscope across 22.1 Ohm series resistor*

*Sleep current measured using Agilents DMM 34410A*

*The Resistor value was measured using Fluke 175 True RMS Multimeter*

*Following table shows the measured values and the respective calculations*

| | BLE Event Time (s) | BLE Event average current (mA) | Sleep Time (s) | Sleep Current (mA) | Total time Interval (s) | Average Current (mA) |
|---|---|---|---|---|---|---|
| ***Test Case 11.25msec (connected) 23 bytes*** | 0.00735 | 2 | 0.0039 | 0.0032 | 0.01125 | 1.307776 |
| ***Test case 1.28sec (connected)*** | 0.00432 | 2.08 | 1.27568 | 0.0032 | 1.28 | 0.0102092 |

***Average Current = [{BLE Event time * BLE Event Average Current} + {Sleep time * Sleep current}] / Total time Interval***

## QN9020 Current consumption assessment

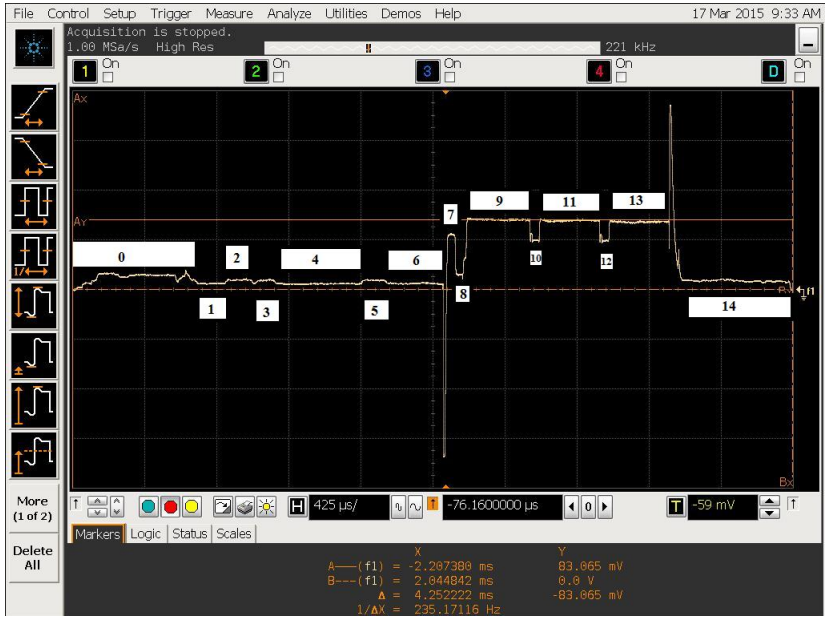
*Test Case iBeacon - 32 byte - Tx power 0dBm*

| State | Current (mA) |
|---|---|
| Sleep | 0.003 |
| Wakeup | 1.72 |
| MCU | 1.28 |
| Idle | 0.85 |
| LO | 6.9 |
| MCU + Ref PLL | 2.2 |
| Tx | 8.5 |
| Rx | 9.2 |

| | BLE EVENT STATES | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *0* | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *10* | *11* | *12* | *13* | *14* | |
| | *Wakeup* | *Idle* | *MCU* | *MCU* | *Idle* | *MCU* | *Idle* | *LO* | *MCU + Ref PLL* | *Tx* | *MCU + Ref PLL* | *Tx* | *MCU + Ref PLL* | *Tx* | *MCU* | *TOTAL (s)* |
| **Duration (msec)** | 0.699 | 0.151 | 0.151 | 0.137 | 0.496 | 0.151 | 0.331 | 0.068 | 0.057 | 0.373 | 0.057 | 0.355 | 0.057 | 0.359 | 0.709 | **0.00415** |
| **BLE event Avg I (mA)** | 3.273512263 | | | | | | | | | | | | | | | |

| | BLE Event time (s) | BLE Event Avg Current (mA) | Sleep time (s) | Sleep Current (mA) | Total time Interval (s) | Avg Current (mA) |
|---|---|---|---|---|---|---|
| ***Test Case 11.25msec*** | 0.00415 | 3.273512263 | 0.095845 | 0.003 | 0.1 | **0.138745882** |

| Test Equipment | |
|---|---|
| Oscilloscope | MSO 9254A |
| Agilent DMM | 34410A |
| Fluke Multimeter | 175 True RMS |
| Quintic Motherboard | MB66 |
| Quintic EVB (QN9020) | QN152 |

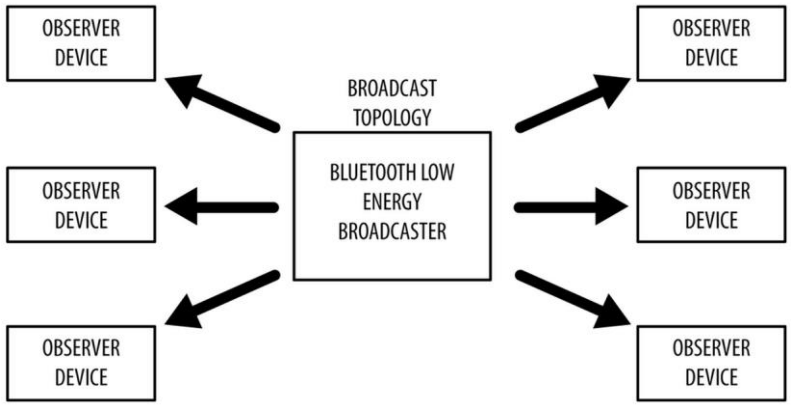


# INTRODUCTION TO QN902X SOFTWARE ARCHITECTURE

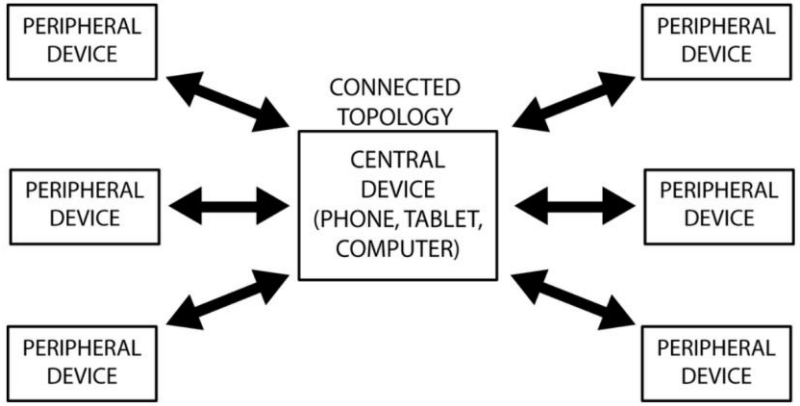
# BLE Network Topology

Can communicate in two ways: *broadcasting* or *connections*

**Broadcast Topology**

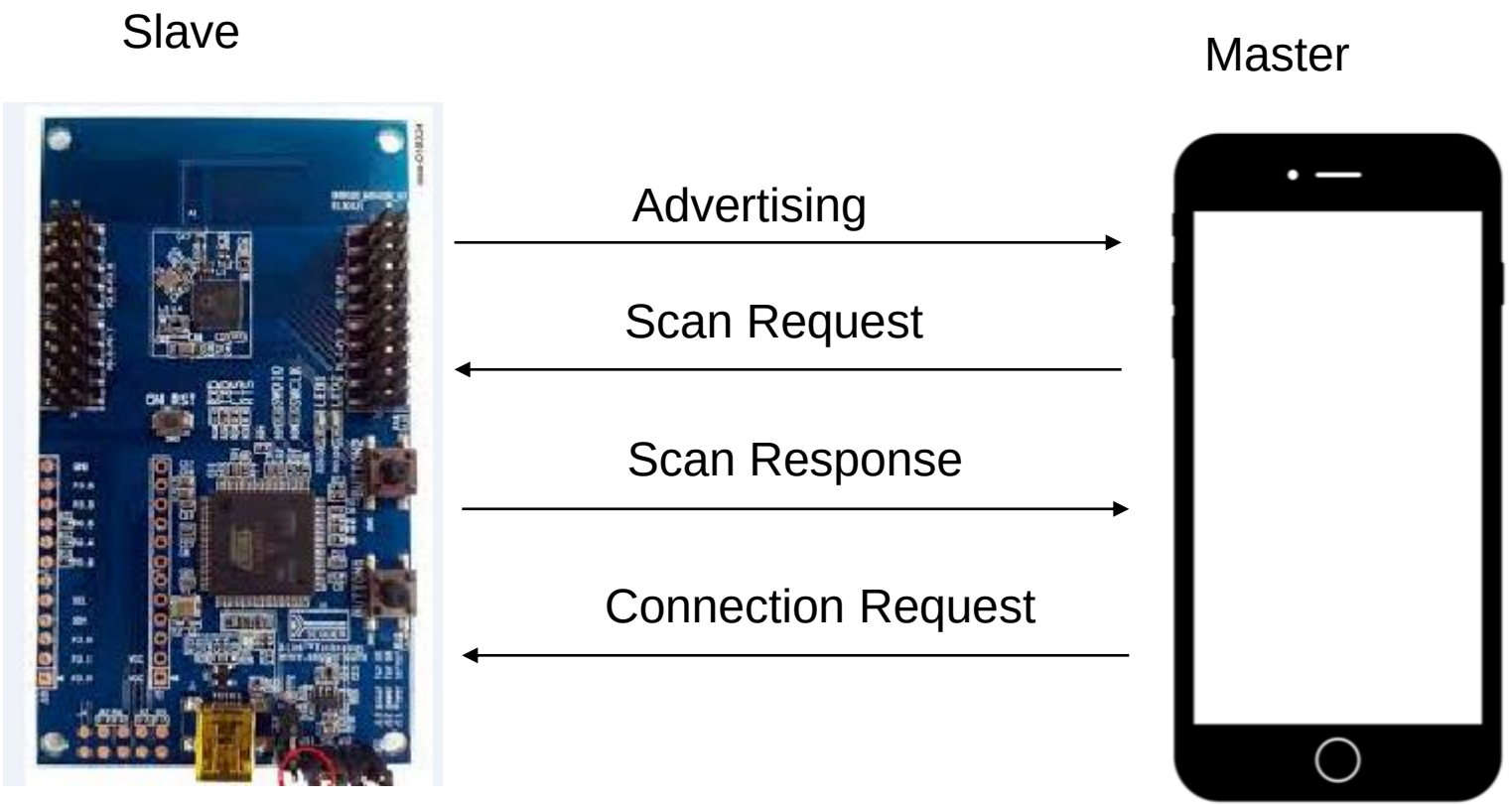



**Connected Topology**



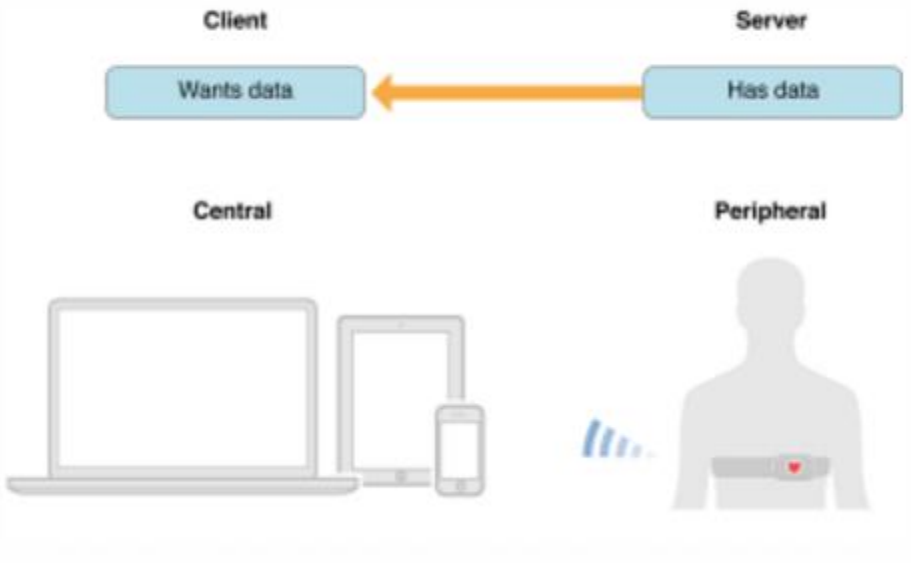
# Connected Topology : Roles in BLE

**Master** (or "**central**") devices scan for other devices. Usually, the master is the smartphone/tablet/PC.
**Slave** (or "**peripheral**") devices advertise and wait for connections.

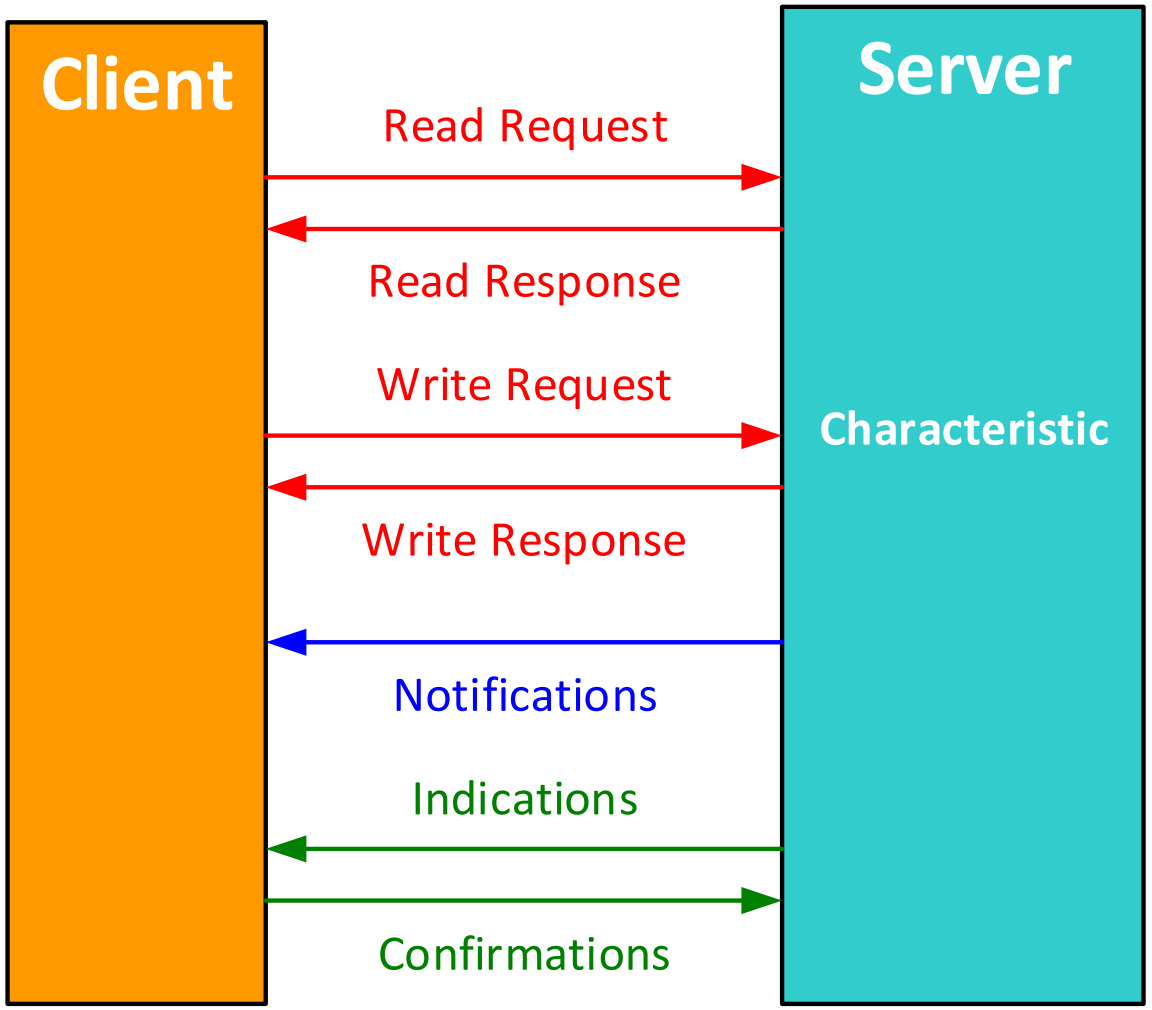


# Connected Topology : Roles in BLE

**Client** devices access remote resources over a BLE link using the GATT protocol. Usually, the master is also the client.
**Server** devices have a local database and access control methods, and provide resources to the remote client. Usually, the slave is also the server

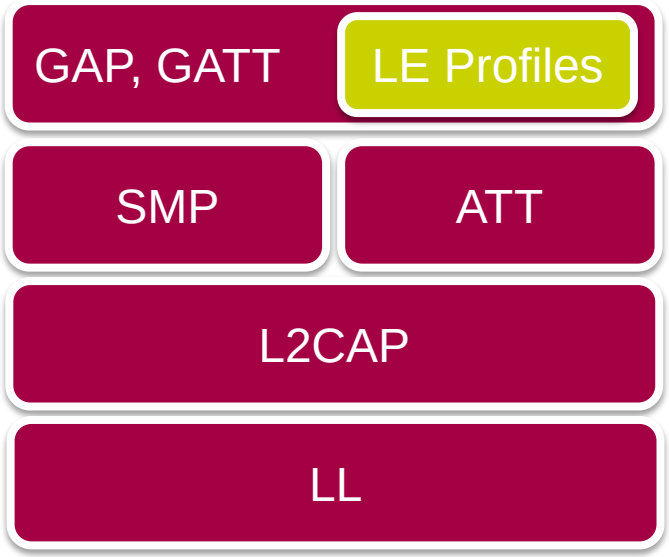


# Connected Topology : Access Characteristics

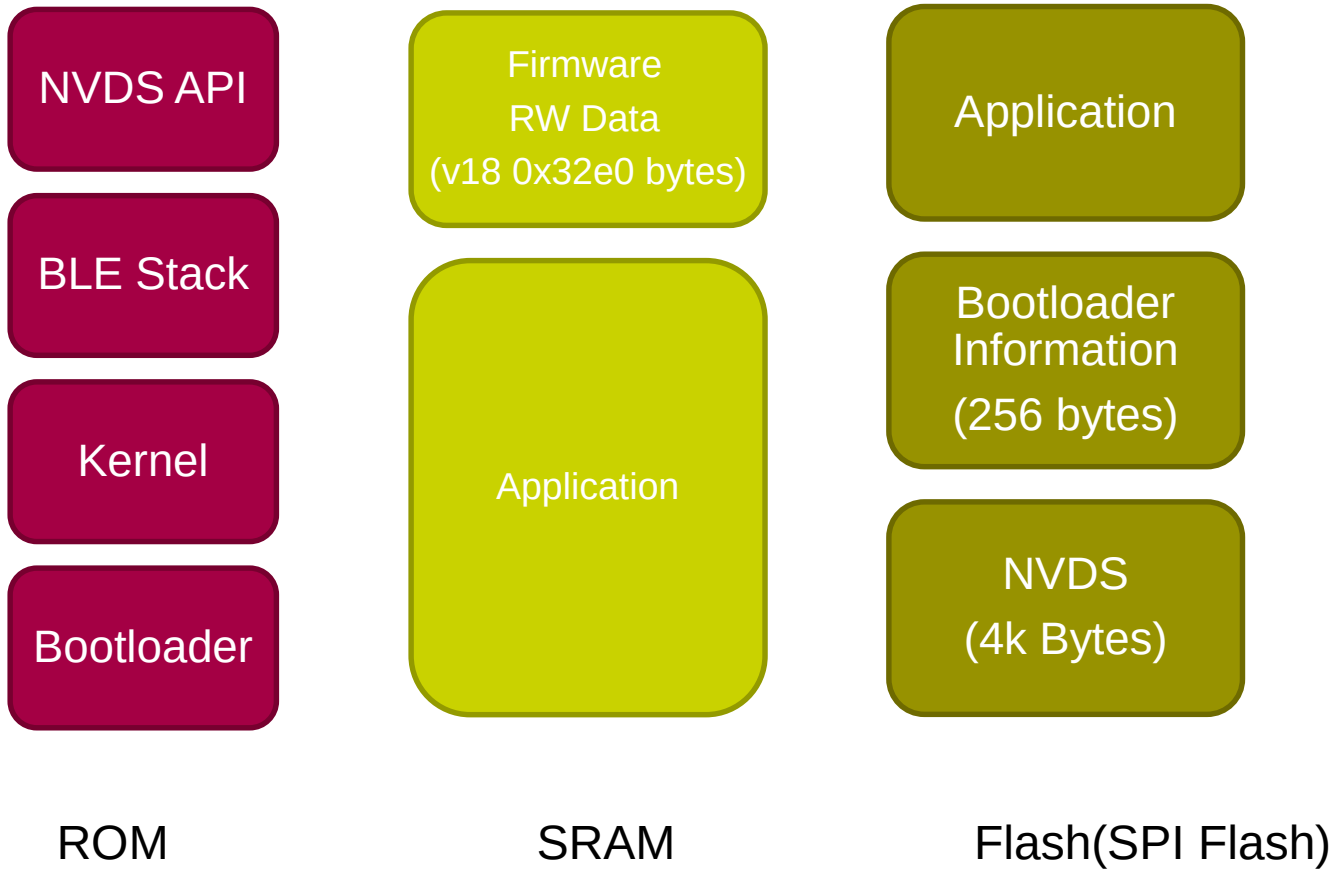


# BLE Stack – QN902x Features

- Supports all roles
- Up to 8 connections (master mode)
- No Rx buffer limitation in one event (High throughput).
- Source code for LE profiles
- Bluetooth qualified
- Proven interoperability
- No qualification costs



# QN902X Architecture – Program Mapping



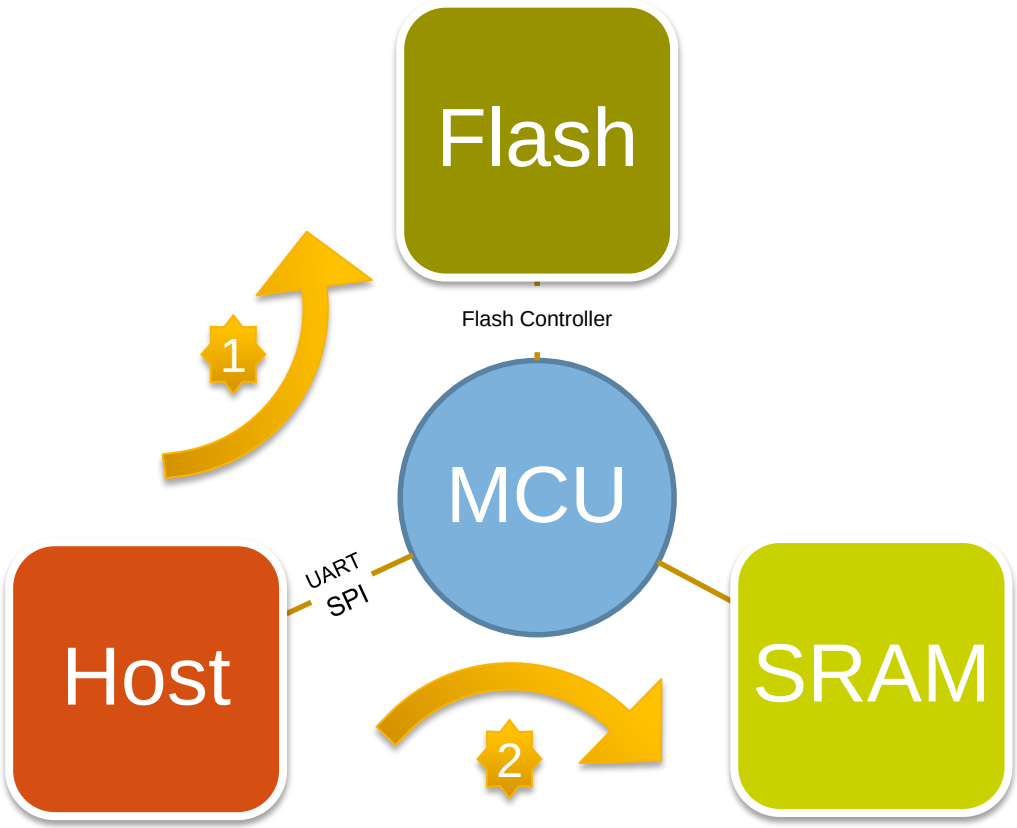
# Test Support - Bootloader

A. System Reset.

B. BL finds Connection Command from UART or SPI.

C. BL follows Host commands.
Host can download APP to Flash(1) or SRAM(2).

A. Host starts APP.



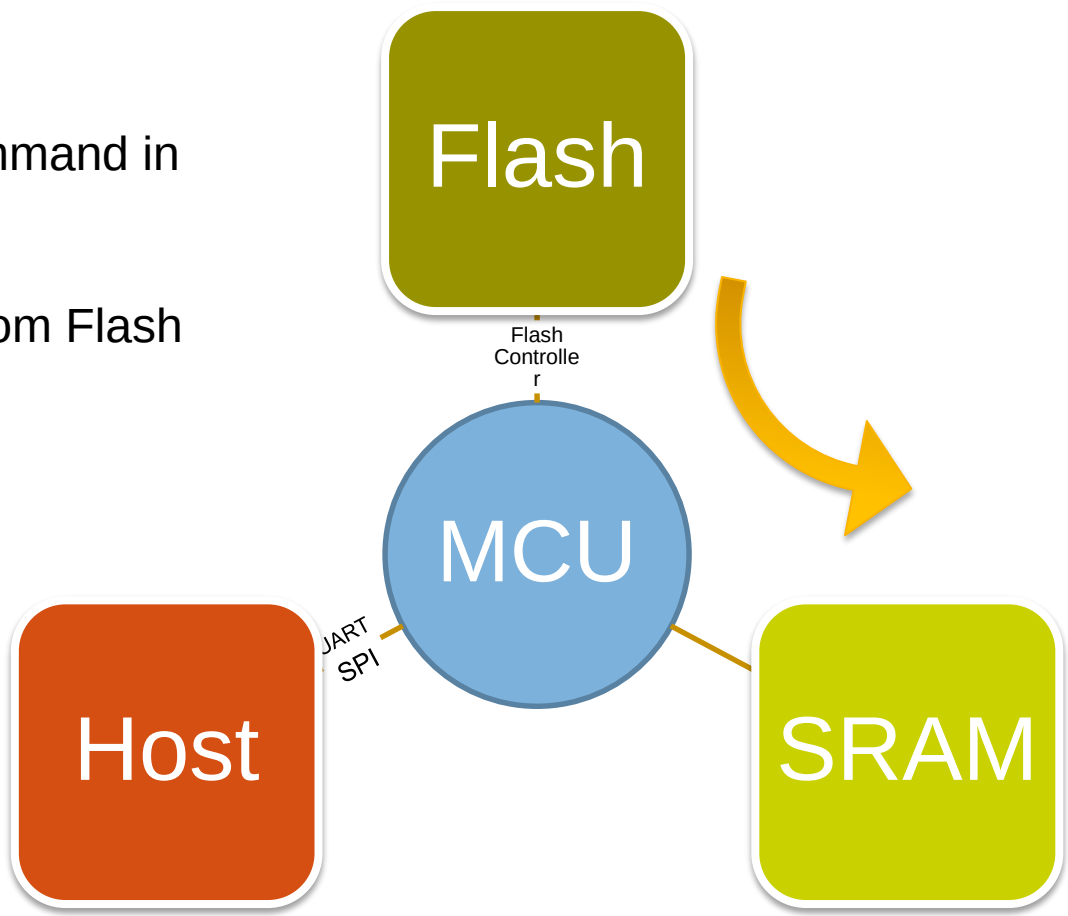
# Test Support - Bootloader

A. System reset.

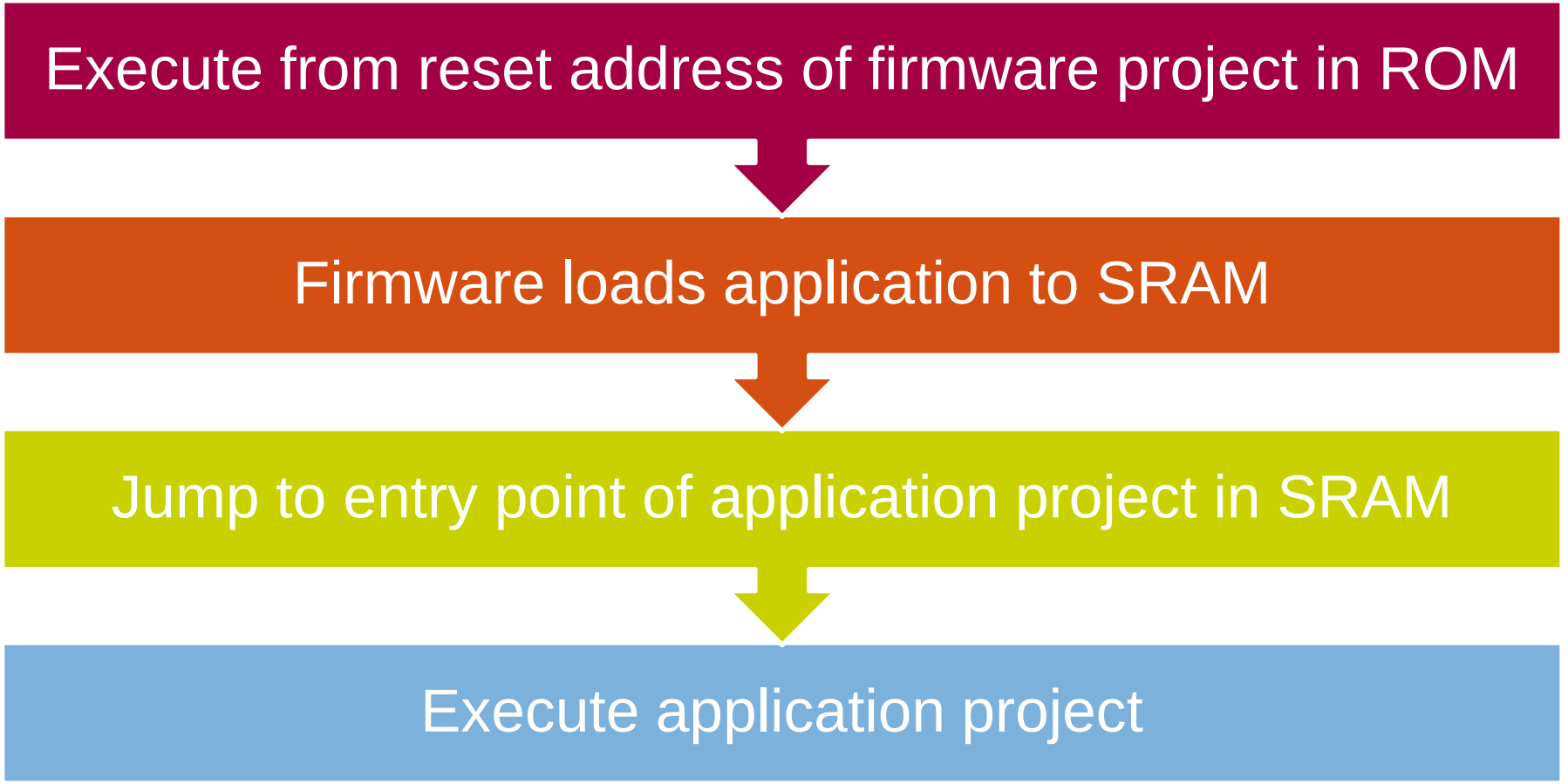
B. BL does not find Connection Command in 200ms

C. BL loads APP from Flash to SRAM

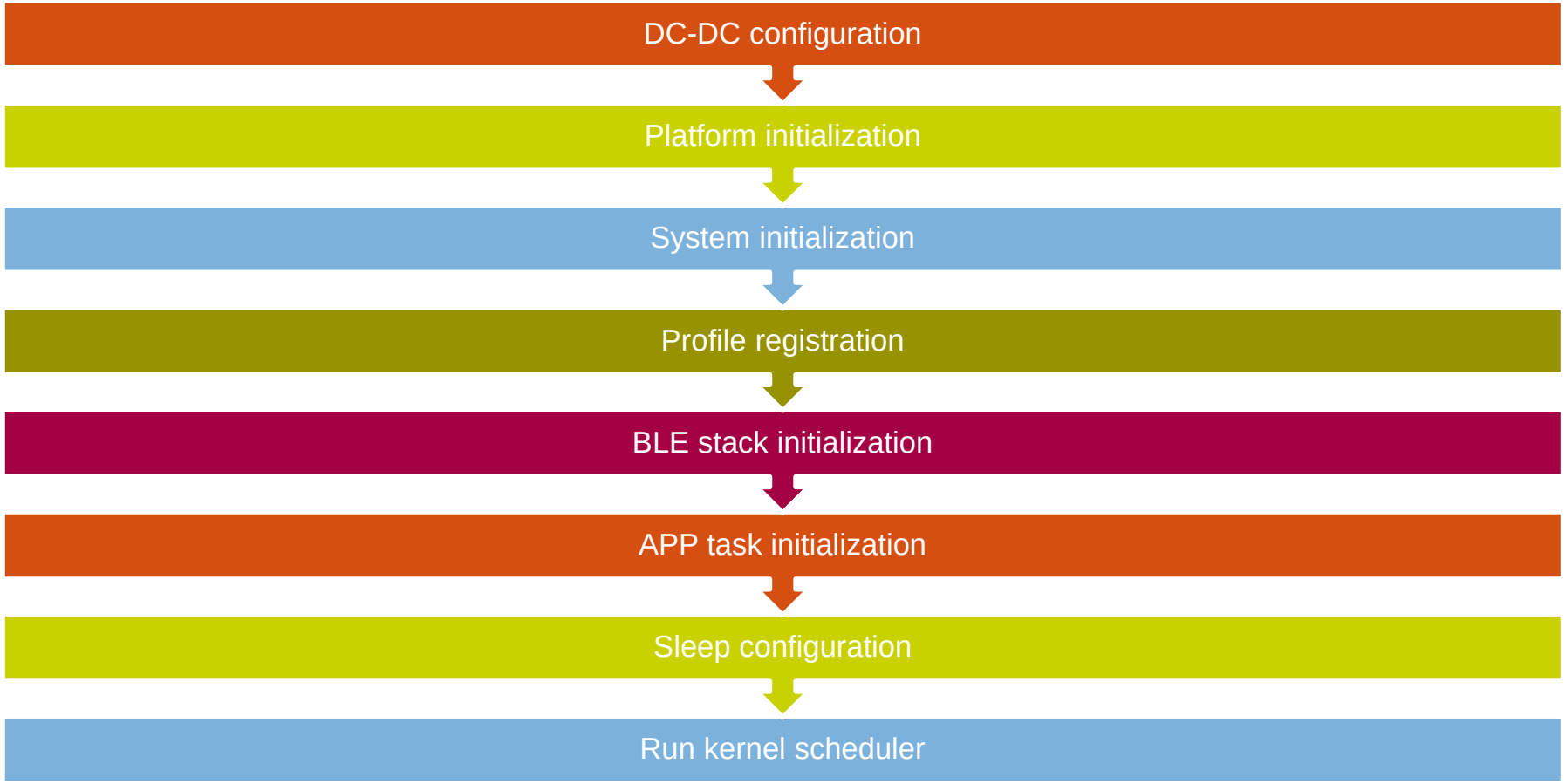
D. Jump to APP



# Application Development – Execution Flow1



# Application Development – Execution Flow2



# Power Mode

- Power mode:

  Deep Sleep / Idle / Sleep / Deep Sleep

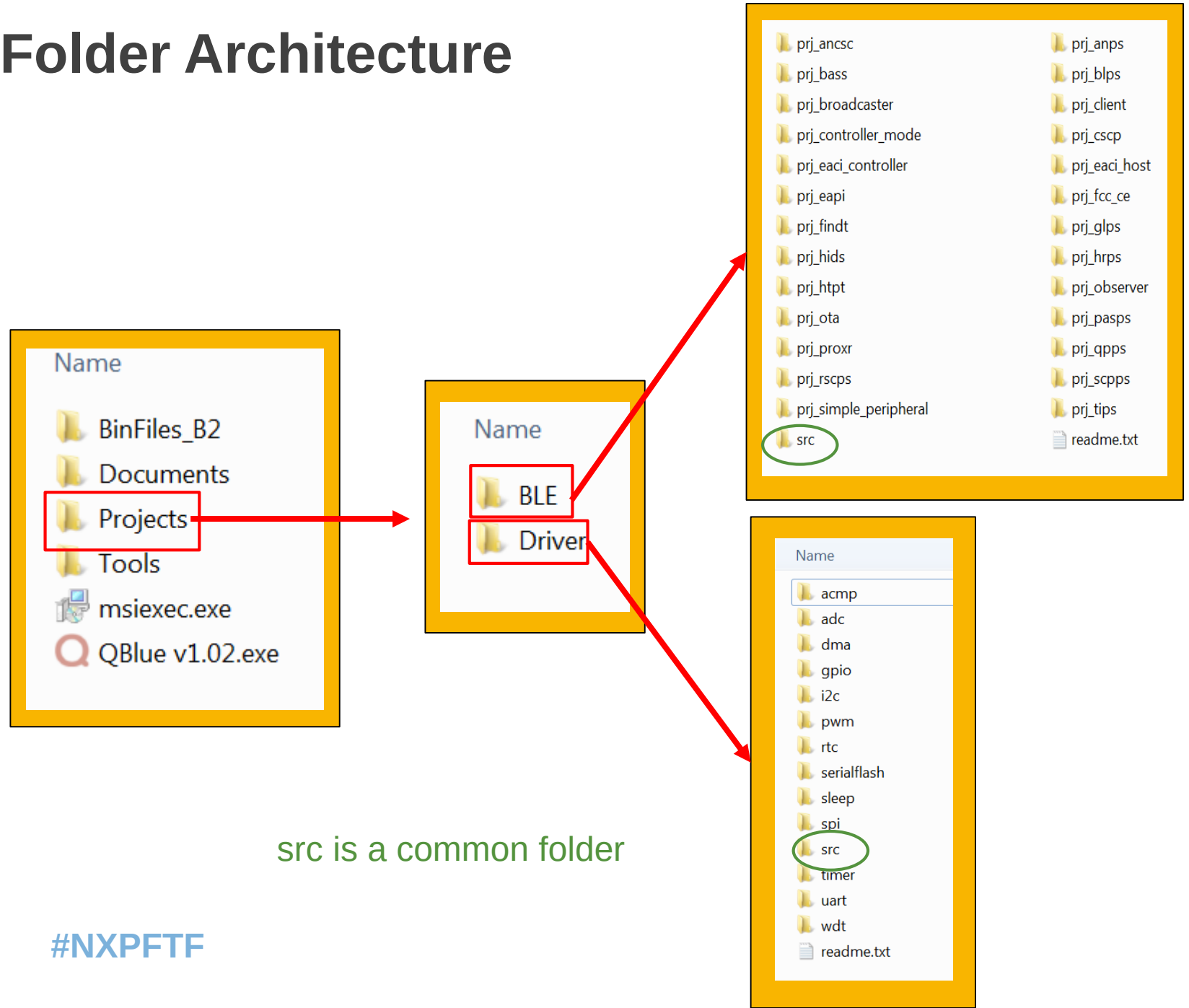
- Active > Idle > Sleep > Deep Sleep

| USR / BLE | Active (MCU) | Idle (CLOCK OFF) | Sleep (POWR DOWN) | Deep Sleep |
|---|---|---|---|---|
| Active | Active | Active | Active | Active |
| Idle | Active | Idle | Idle | Idle |
| Sleep | Active | Idle | Sleep | Deep Sleep |

Macro in usr_config.h

- ➢The macro CFG_DEEP_SLEEP defines the initial value of user's power mode setting.
- ➢If the macro CFG_BLE_SLEEP is not defined, ble_sleep() will not enter into Sleep Mode. So the system will not enter into Sleep Mode.

# QN902x Folder Architecture



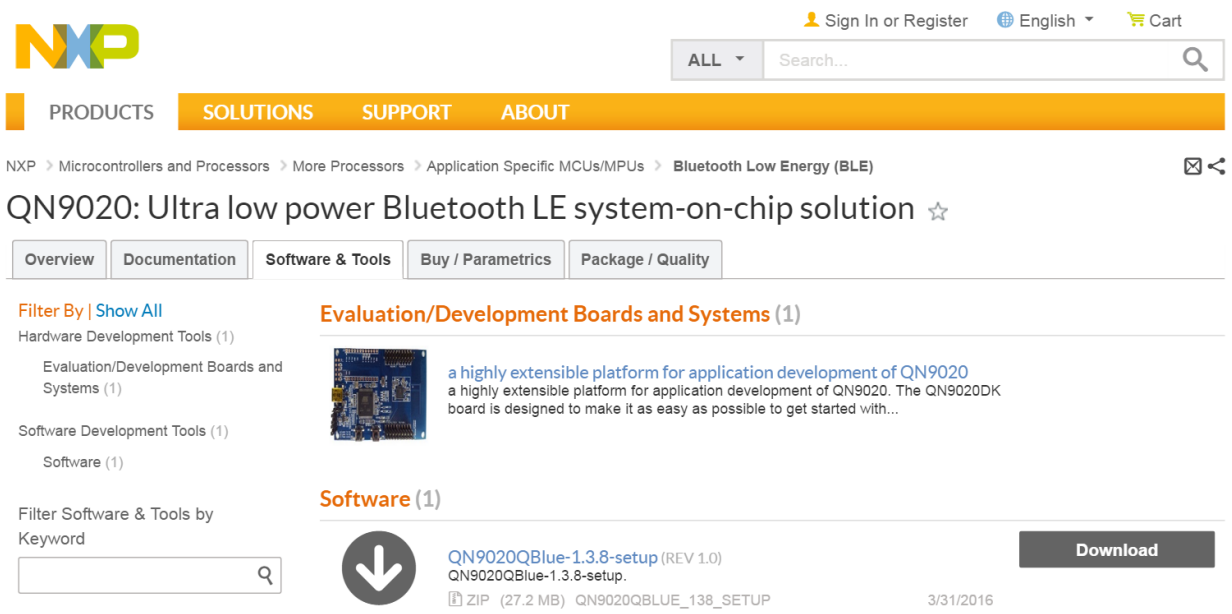

src is a common folder

# SDK Installation

1. Get SDK package from NXP website
2. Install IDE (Keil MDK-ARM or IAR)
3. Install SDK package
4. Install J-Link SW by Segger
5. Install VCP driver for BLE dongle

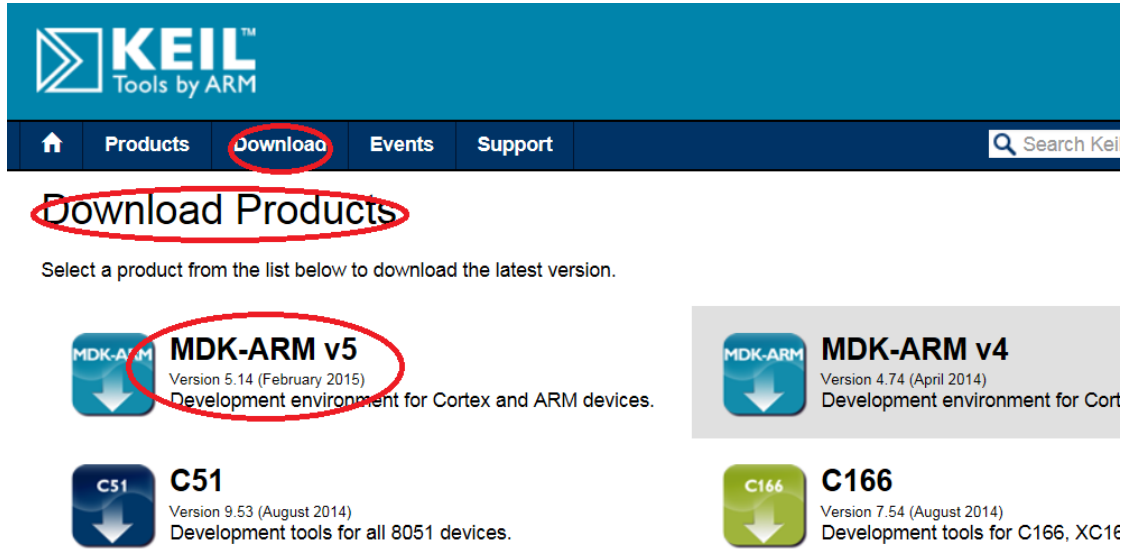
# Get SDK package from NXP website

- Website address: http://www.nxp.com/products/microcontrollers-and-processors/more-processors/application-specific-mcus-mpus/bluetooth-low-energy-ble/ultra-low-power-bluetooth-le-system-on-chip-solution:QN9020?fpsp=1&tab=Design_Tools_Tab



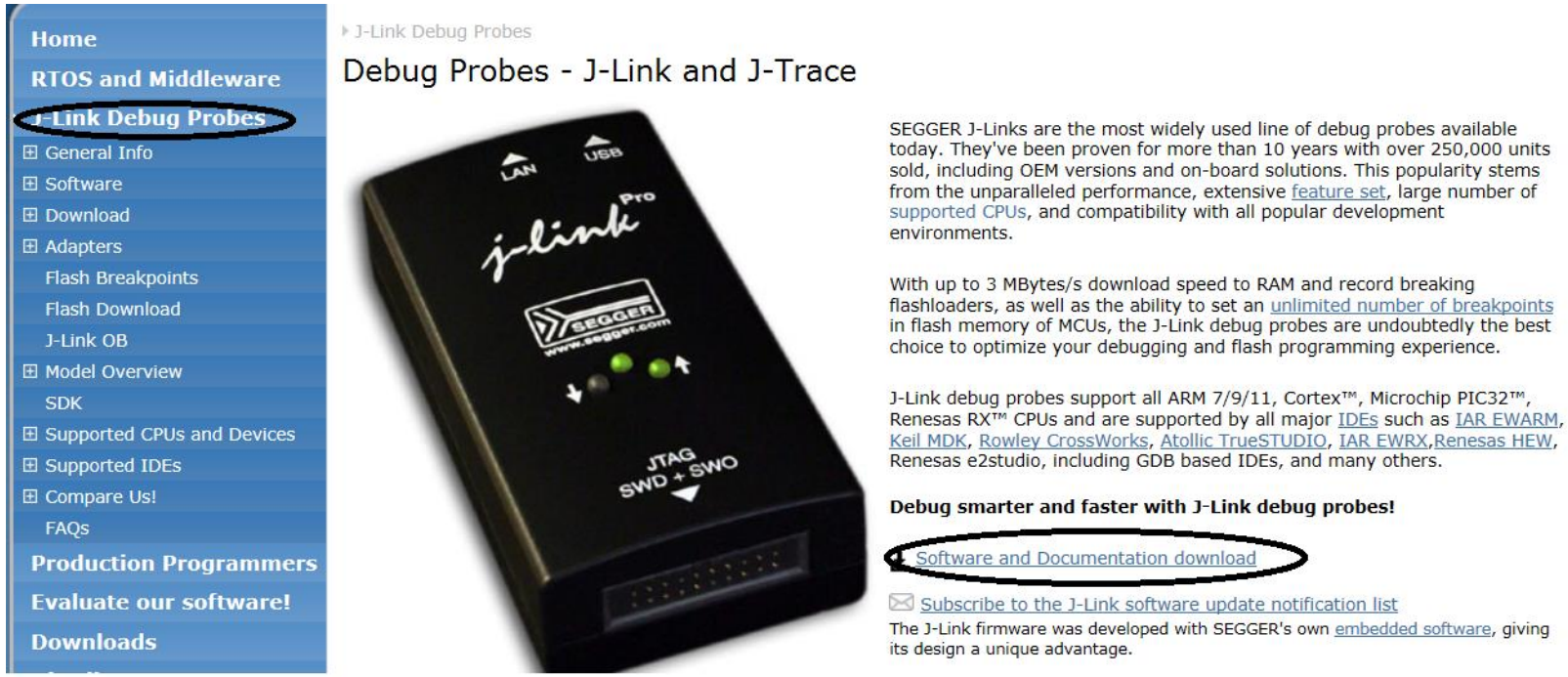
# Install Keil MDK-ARM

- Download Keil MDK-ARM at https://www.keil.com/download/product/
- Version 4.5 or newer is recommended



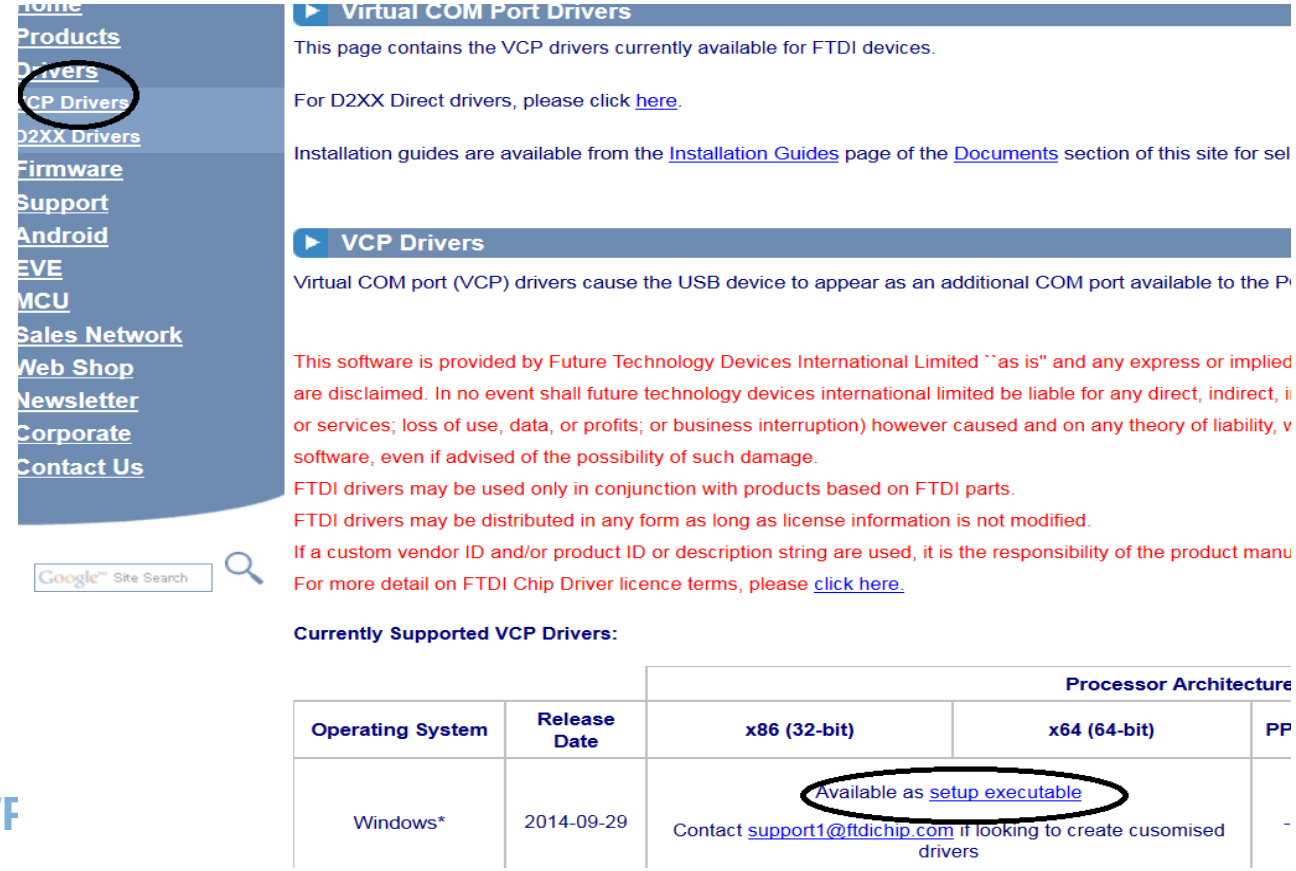
# Install J-Link SW

- Download Setup_JLink_Vxxx.exe at www.segger.com
- USB driver for J-Link will be installed automatically and appear in Windows Device manager as “Jlink CDC UART Port (COMx)”.
- Version 4.6.6 or newer is recommended.



# Install VCP Driver for BLE dongle

- Download VCP driver from http://www.ftdichip.com
- AN-104 to guide installation is available at http://www.ftdichip.com/Support/Documents/AppNotes.htm



Products
Drivers
VCP Drivers
D2XX Drivers
Firmware
Support
Android
EVE
MCU
Sales Network
Web Shop
Newsletter
Corporate
Contact Us

Google™ Site Search

## Virtual COM Port Drivers

This page contains the VCP drivers currently available for FTDI devices.

For D2XX Direct drivers, please click here.

Installation guides are available from the Installation Guides page of the Documents section of this site for sel

## VCP Drivers

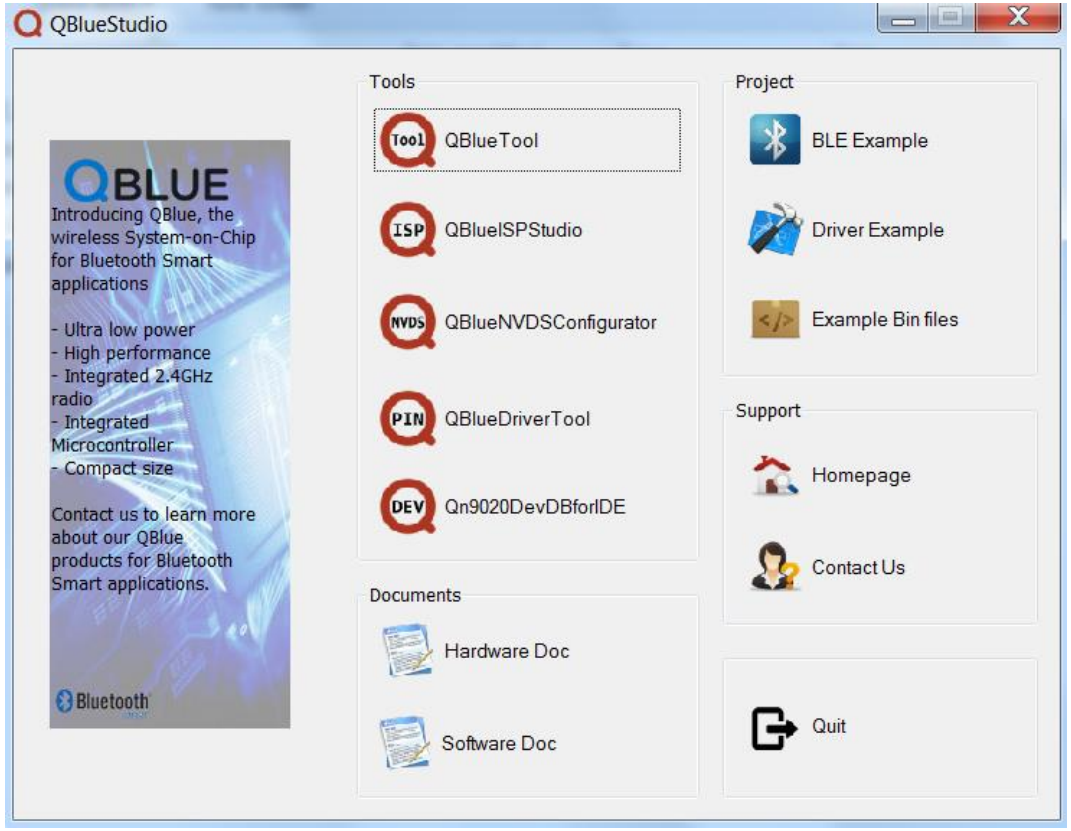
Virtual COM port (VCP) drivers cause the USB device to appear as an additional COM port available to the P

This software is provided by Future Technology Devices International Limited ``as is" and any express or implied
are disclaimed. In no event shall future technology devices international limited be liable for any direct, indirect, i
or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, v
software, even if advised of the possibility of such damage.
FTDI drivers may be used only in conjunction with products based on FTDI parts.
FTDI drivers may be distributed in any form as long as license information is not modified.
If a custom vendor ID and/or product ID or description string are used, it is the responsibility of the product manu
For more detail on FTDI Chip Driver licence terms, please click here.

**Currently Supported VCP Drivers:**

| | | Processor Architecture | | |
|---|---|---|---|---|
| **Operating System** | **Release Date** | **x86 (32-bit)** | **x64 (64-bit)** | **PP** |
| Windows* | 2014-09-29 | Available as setup executable<br>Contact support1@ftdichip.com if looking to create cusomised drivers | | - |

# Launch QBlueStudio




| Tools | Usage |
|---|---|
| **QTool** | Work with BLE dongle, to act as a BLE peer device |
| **QBlueISPStudio** | Download bin file to QN902x via UART0 port |
| **QBlueNVDSConfigurator** | Manage configuration info in NVDS area in flash |
| **QBlueDriverTool** | Generate source code for GPIO and pin mux configuration |
| **Qn9020DevDBforIDE** | Device database for IDE (Keil and IAR) |

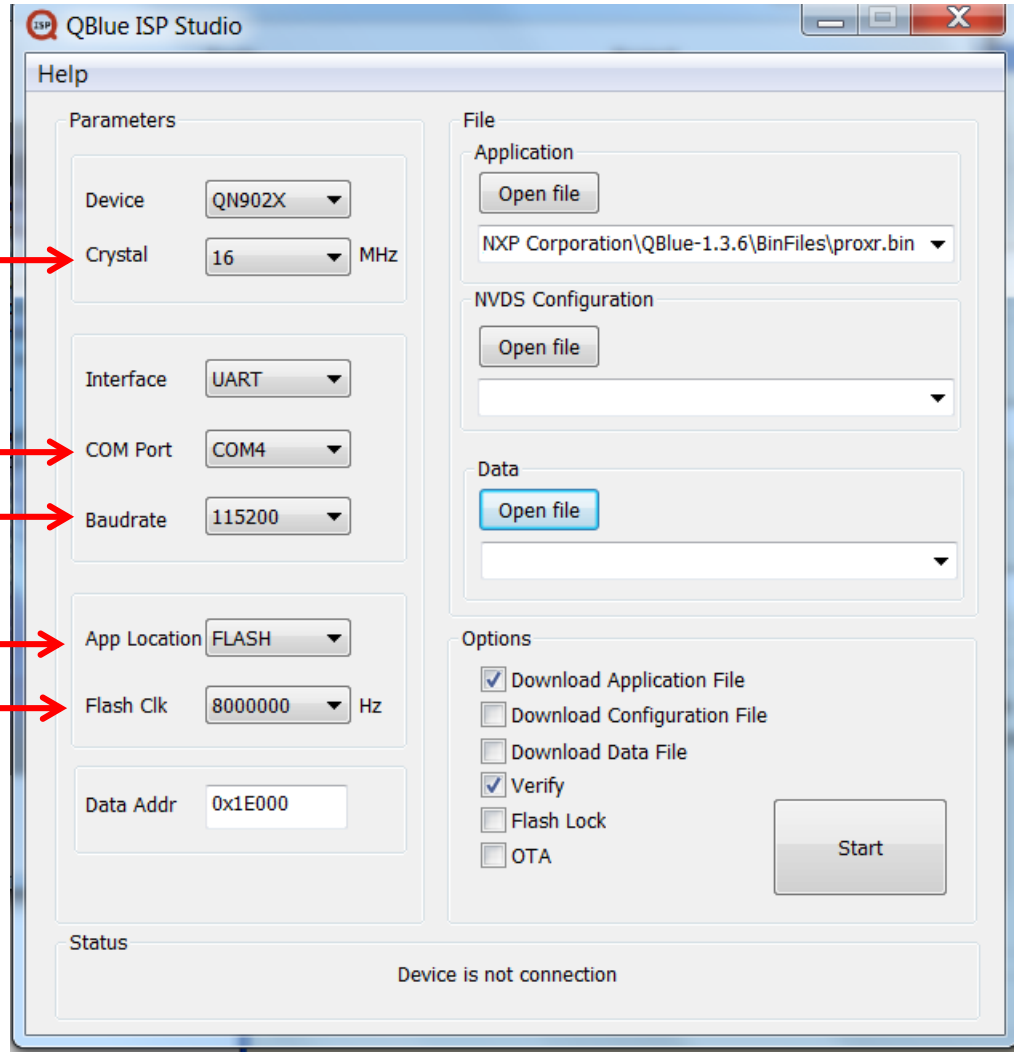
# Firmware Downloading

- Tool: QBlue ISP Studio

16M or 32M crystal

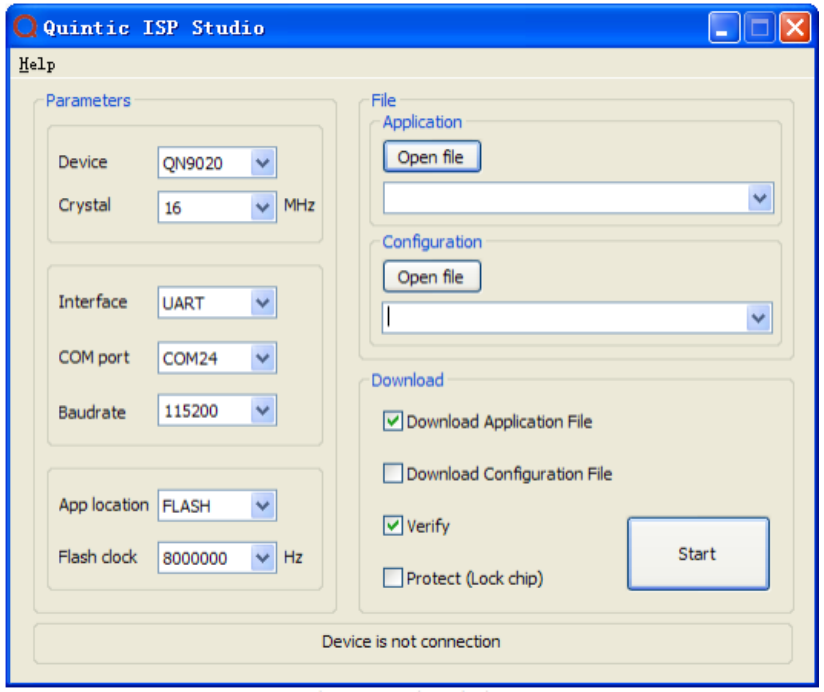
Select the right COM port

UART baud rate



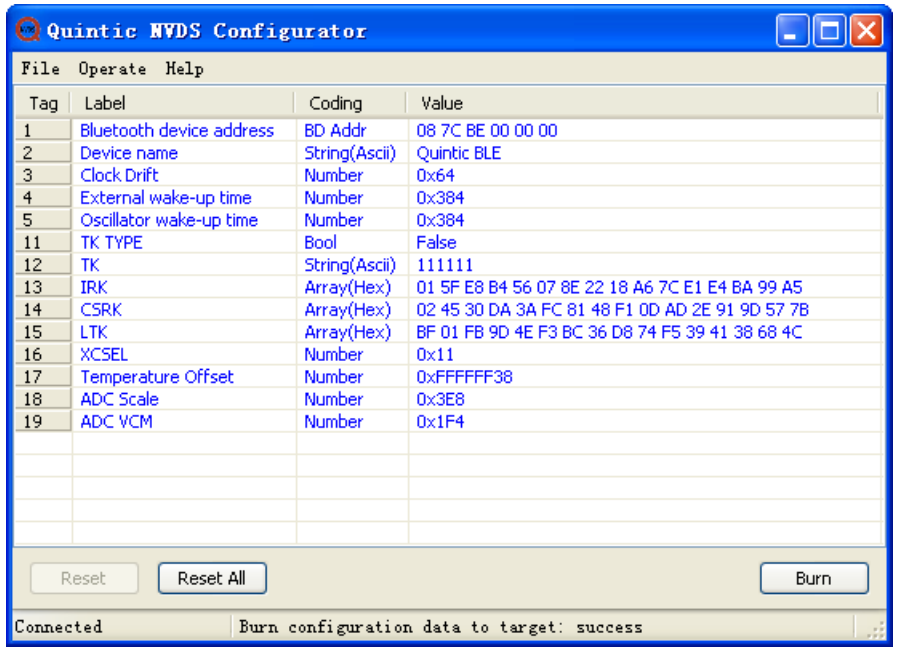
# SDK – Software Tools – QnISPStudio

- QnISP Studio provides a way to download application binary file and NVDS configuration file into Flash through boot protocol via UART or SPI interface



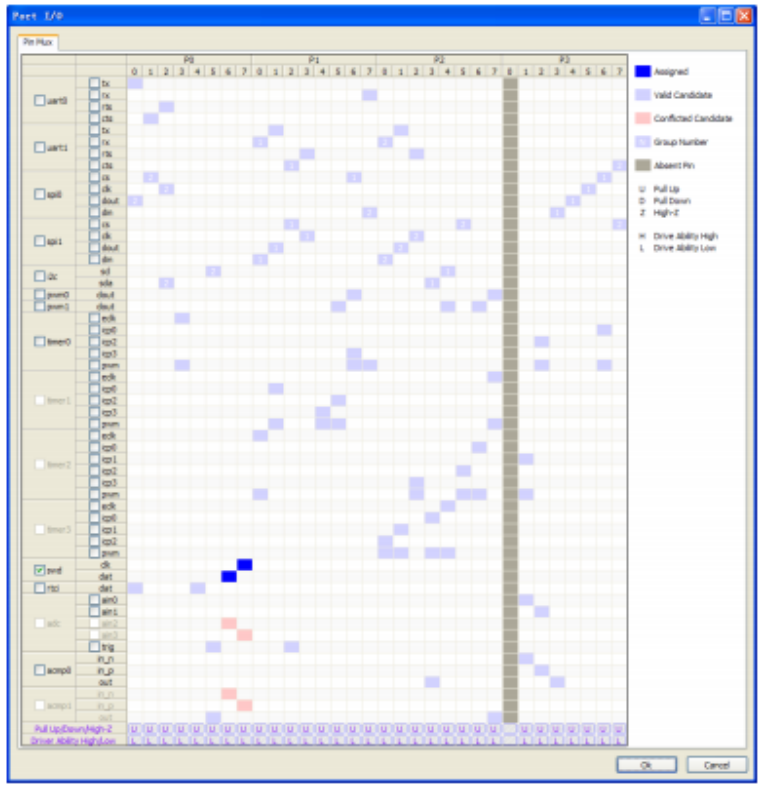
# SDK – Software Tools – QnNVDSConfigurator

- NXP NVDS Configurator supports editing, creating and programming NVDS configuration file.
- Keep the factory setting of Temperature Offset, ADC Scale and ADC VCM.



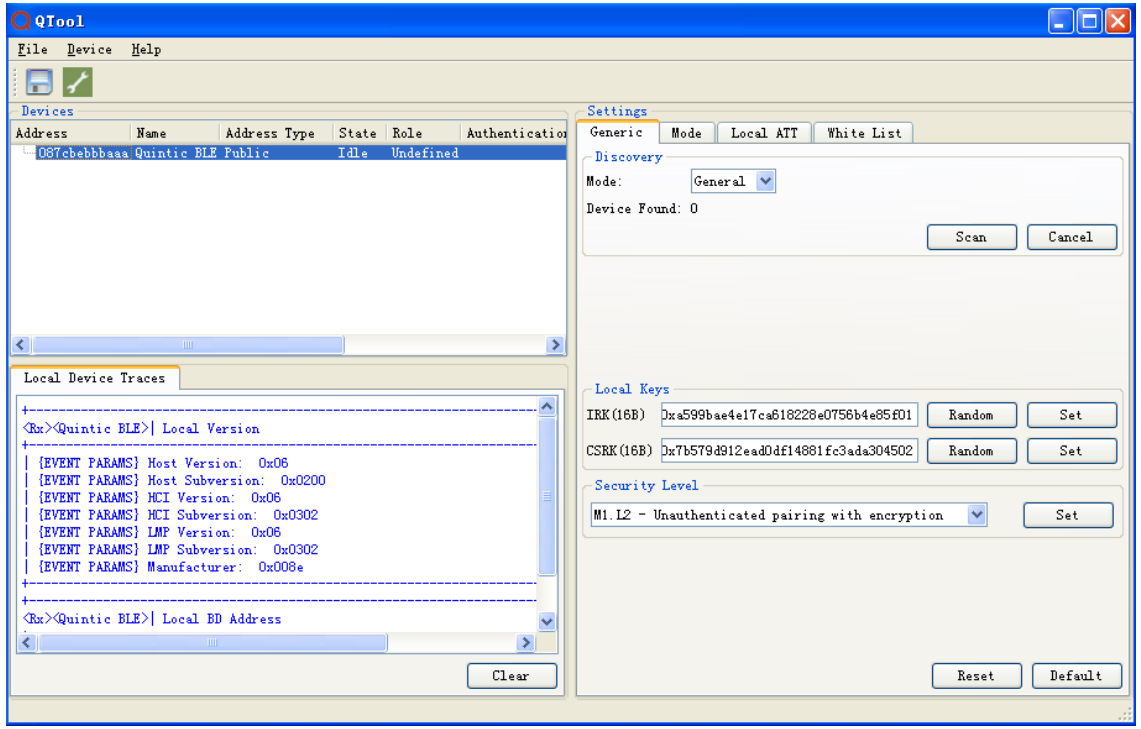
# SDK – Software Tools – QnDriverTools

- QnDriver Tool is used to generate QN9020 pin-mux setting



# SDK – Software Tools – QTool

- QTool communicates with NXP BLE device through UART. NXP BLE device works on network processor mode using NXP ACI interface.



# Code Review

- QPP : BLE raw data transfer proprietary profile
  - QPPS - prj_qpps
    - Source code : qpps.h, qpps.c, qpps_task.h, qpps_task.c
      app_qpps.h, app_qpps.c, app_qpps_task.h, app_qpps_task.c
  - QPPC - prj_client
    - Source code : qppc.h, qppc.c, qppc_task.h, qppc_task.c
      app_qppc.h, app_qppc.c, app_qppc_task.h, app_qppc_task.c
  - Configurations
    - Service UUID

      *#define QPP_SVC_PRIVATE_UUID*
      *"\xFB\x34\x9B\x5F\x80\x00\x00\x80\x00\x10\x00\x00\xE9\xFE\x00\x00"*
    - Number of notification characteristic (QPPS → QPPC)
      *#define QPPS_NOTIFY_NUM 5*

# NXP SOLUTIONS WITH QN902X

# LPC54102 and BLE - Solution:

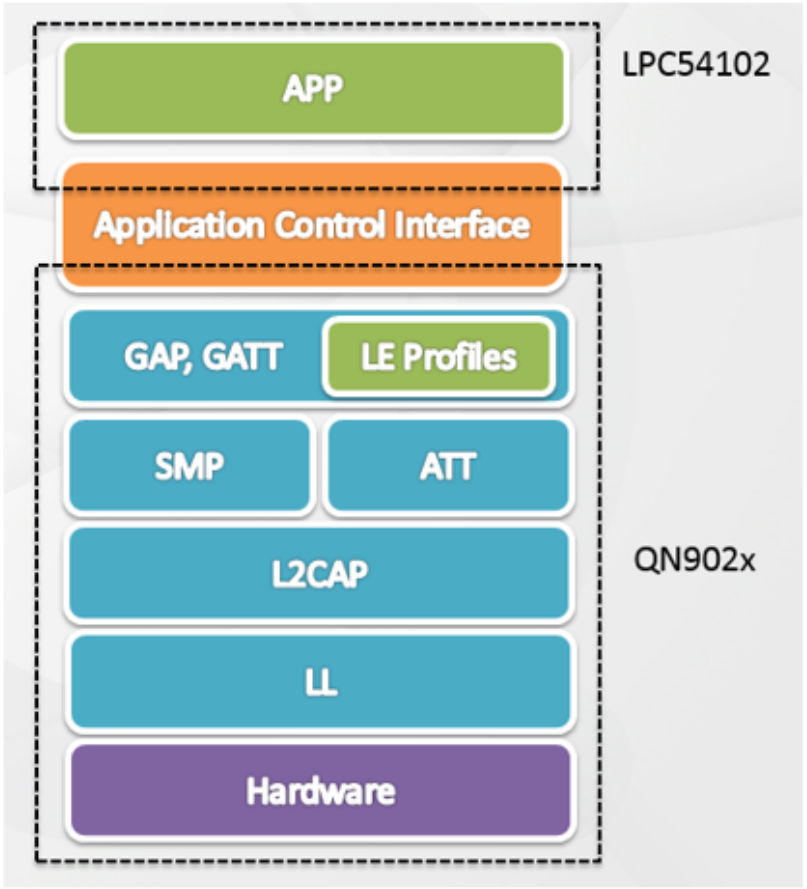
- 1. Niobe SIP introduction
- 2. Module information
- 3. Hardware setting
- 4. Software programming
- 5. Qpps demo
- 6. OTA demo
- 7. Schematics

# Niobe SIP introduction

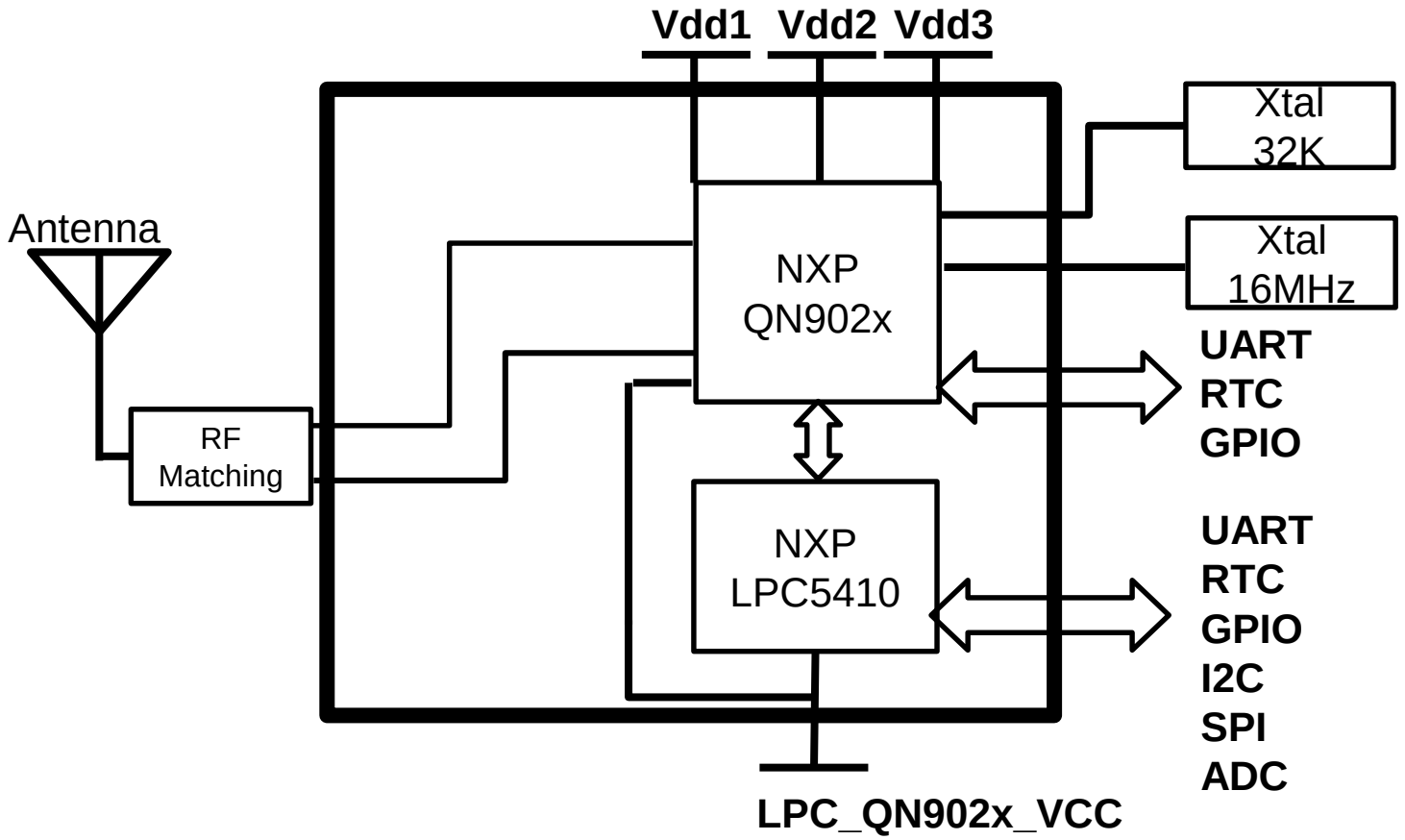
- Niobe SIP components :  LPC54102 MCU + QN902x BLE in which user application code is developed on the LPC54102 MCU and the QN902x is programmed as a BLE radio.
- A multichip module: Totally 48 pin LGA package. The standalone QN902x part has a serial flash stacked on it. For the MCM application the serial flash was removed, the flash in the LPC54102 will be used for application code and data storage.
- Interface: The interface between LPC and QN902x is UART

# Niobe SIP introduction

- Application runs on LPC54102
- BLE stack runs on QN902x
- EACI(Easy ACI) is the communication interface

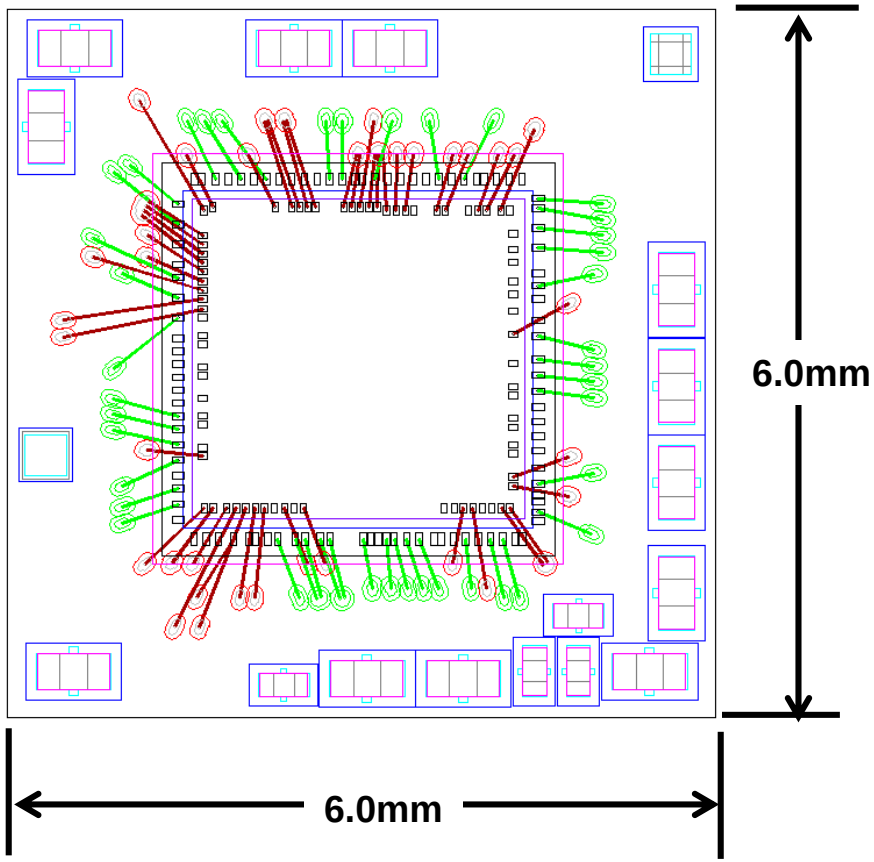


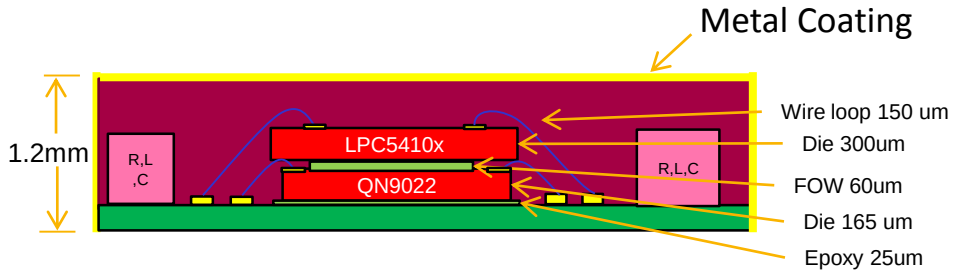
# Module information - Function Diagram




81

# Module information - Dimension

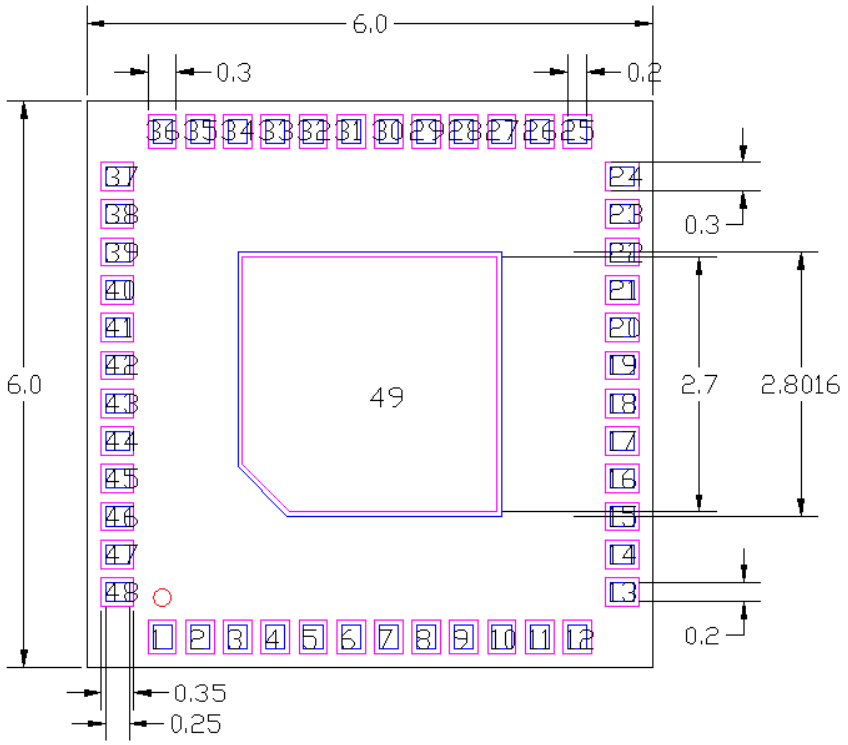



| | Wafer Size (8" or 12") | Wafer tec. (Low-K 65nm or normal) | Min. grinding thickness |
|---|---|---|---|
| Niobe(LPC5410) | 12″ | Low-K 90nm | 300um |
| MC3610 | 8″ | Normal | 550um |
| QN9020 | 8″ | Normal | 165um |

- Body Size 6.0x6.0mm
- Thickness: 1.2mm max depend on the final grind thickness =300um of LPC MCU
- Substrate : 4L BT
- Key Process : SMT, Stack die, EMI Coating
- Foot Print : LGA
- Component :
  QN9022 die, LPC5410x die,
  RLC : 12 pieces (0201) , 4 pieces (01005)

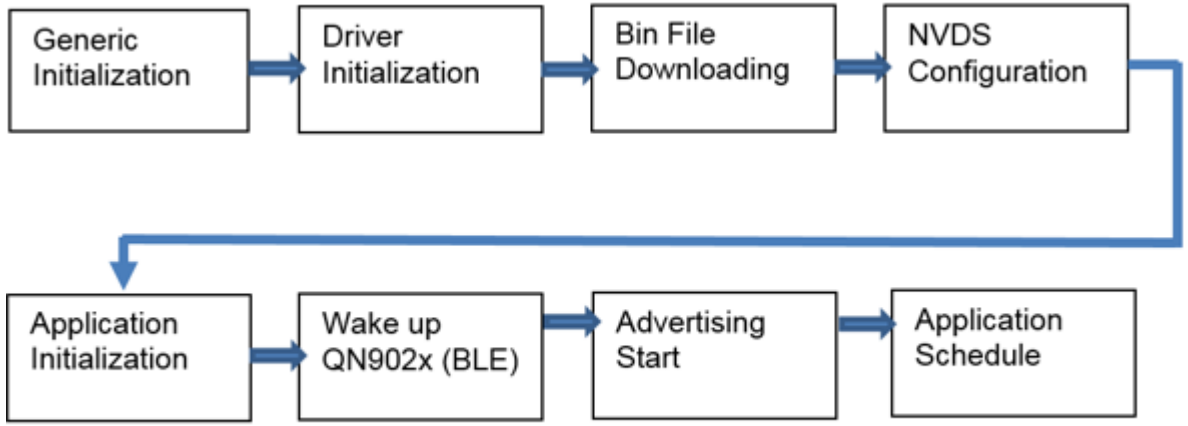
# Module information - Pinout Footprint




**Module Top View**

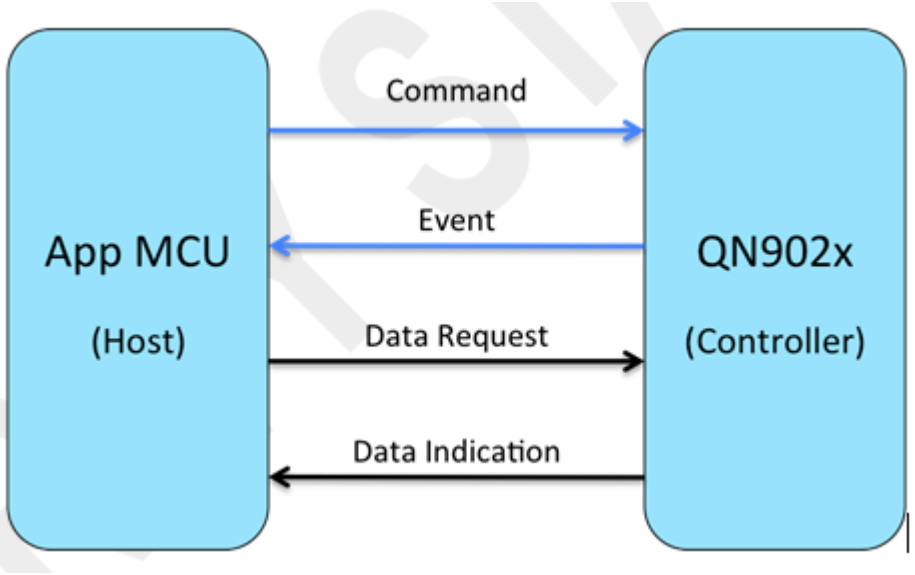
| Module | Size(mm) |
|---|---|
| Pad Size | 0.3x0.35 |
| Solder Mask Opening | 0.2x0.25 |
| Pitch | 0.4 |

# Software programming – Flow chart



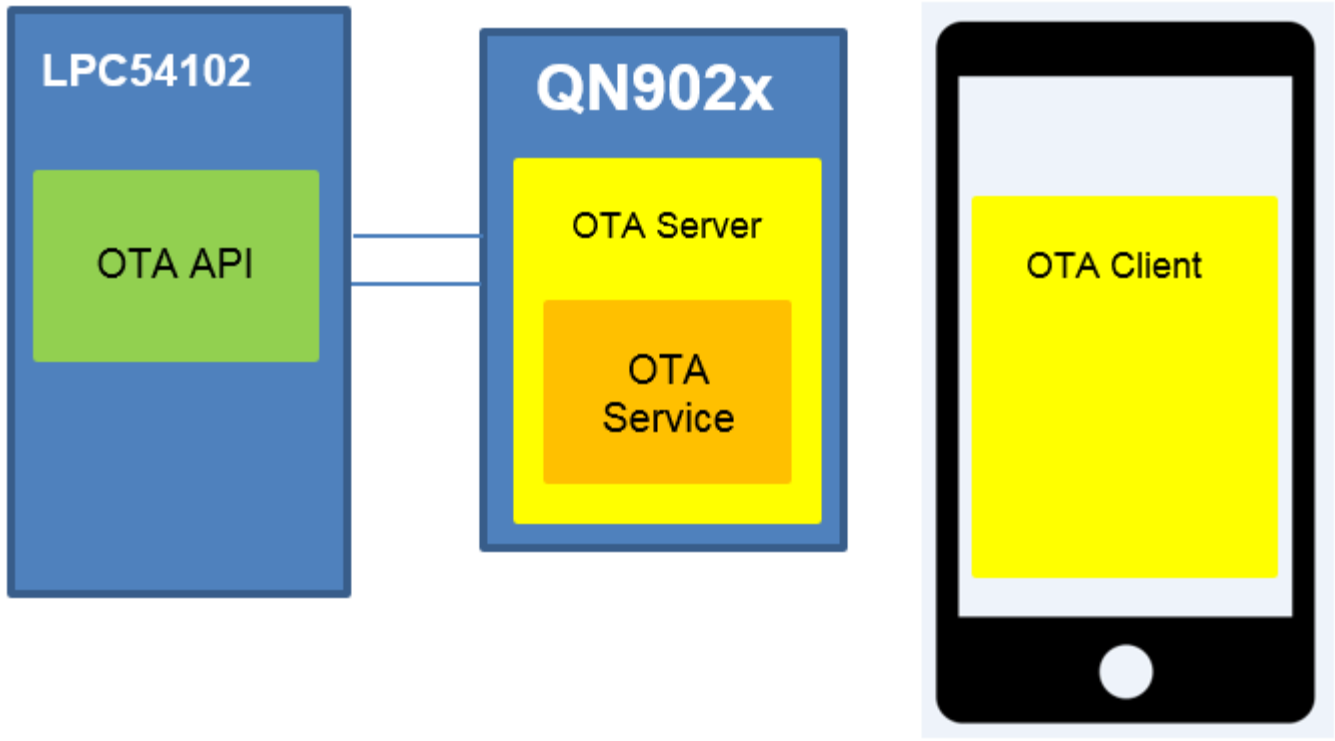
# Software programming – EACI UART

- EACI packet format: a message format to communicate between host unit and controller unit. The message can be divided into 4 kinds of types: command, event, data request and data indication.



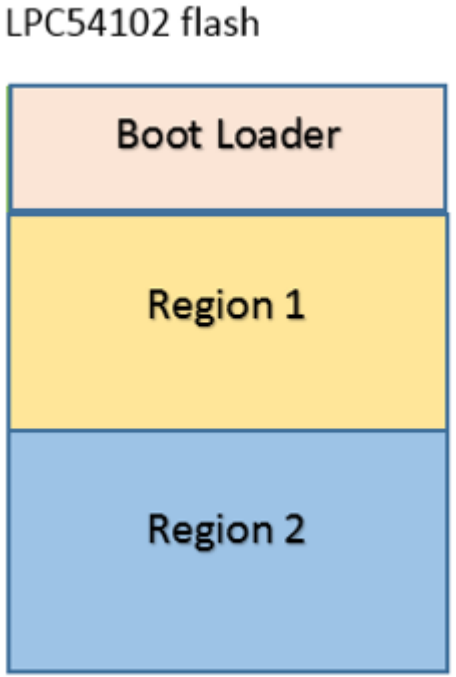
# OTA – Over the Air update

- The OTA profile is used to upgrade the firmware of LPC54102 via BLE over the air.



# OTA demo

1. For the very first time, the user needs to download the boot loader code into address 0x00 and download LPC54102_Host code to Region1.
2. When updating firmware from phone APPs, the new firmware will be stored into Image2 region.
3. Boot loader will erase original image and copy the new firmware to region 2
4. Boot loader will jump to image1 again and run the newly updated firmware.



# NFC READER WITH QN902X

# NFC Reader Library product features

## Characteristics

Supporting CLRC663 NFC front end.

Written in C programming language.

Modular multi-layer approach, enabling flexibility and scalability.

Intended to be used on microcontrollers.

Type A reader/writer application example included.

Built-in support for NXP QN902x Bluetooth microcontroller.

## Supported protocols

**Reader/Writer mode**
ISO/IEC 14443-3A (Type A)
MIFARE 1K, 4K

**Application Layer**
MIFARE Ultralight
MIFARE Classic

## Host Interface support

Built-in support for I2C.

Customizable to use any host interface.

# Memory footprint

## Type A reader/writer

| Component mandatory | Flash usage [Byte] | RAM usage [Byte] |
|---|---|---|
| HAL CLRC663 | 15677 | 108 |
| ISO 14443-3a | 2220 | 24 |
| Tools | 2412 | 221 |
| OSAL | 932 | 68 |
| **Total** | **21241** | **421** |
| | | |
| **Component optional** | **Flash usage [Byte]** | **RAM usage [Byte]** |
| ISO 14443-4 | 3896 | 20 |
| ISO 14443-4A | 1408 | 20 |
| PAL MIFARE | 1456 | 12 |
| AL MIFARE Classic | 1528 | 12 |
| AL MIFARE Ultralight | 1412 | 20 |
| Discovery Loop | 9012 | 292 |
| Keystore RC663 | 458 | 8 |
| AL MIFARE Desfire | 4352 | 104 |
| **Total** | **23522** | **488** |

## Type B/F reader/writer

| Component mandatory | Flash usage [Byte] | RAM usage [Byte] |
|---|---|---|
| HAL CLRC663 | 15677 | 108 |
| ISO 14443-3B | 3104 | 44 |
| Tools | 2412 | 221 |
| PAL Felica | 1952 | 32 |
| OSAL | 932 | 68 |
| Total | 24077 | 473 |
| | | |
| **Component optional** | **Flash usage [Byte]** | **RAM usage [Byte]** |
| ISO 14443-4 | 3896 | 20 |
| ISO 14443-4A | 1408 | 20 |
| AL Felica | 1012 | 12 |
| Discovery Loop | 9012 | 292 |
| Keystore RC663 | 458 | 8 |
| **Total** | **15786** | **352** |

# Adding NFC Library to BLE Keil project

- Copy “NxpNfcRdLib” folder to the project “src” folder
- Copy “Stub” folder to the project “src” folder
- Configure GPIO and interface settings in “system.c”
- Set up GPIO mapping in file “phhwConfig.h”
  - NFC reader I2C slave address : READER_CHIP_ADDR
  - NFC reader interface : I2C_USED / SPI_USED
  - IRQ Pin Number : INTERRUPT_PIN
  - I2C address configuration : PIN_AD0 and PIN_AD1
- Adding interrupt handler entry in function “gpio_interrupt_callback”

```
case INTERRUPT_PIN:
        PN512_IRQHandler();
break;
```

# Using NFC Library

- Include related header files
  - *#include "phhalHw.h"*
  - *#include "phhalHw_Rc663_Reg.h"*
  - *#include "phhwConfig.h"*
  - *#include "phpalI14443p3a.h"*
- Declare the NFC library parameters

```
phhalHw_Rc663_DataParams_t      sHalReader;
uint8_t                         bHalBufferTx[256];
uint8_t                         bHalBufferRx[256];
/* BAL variables*/
phbalReg_Stub_DataParams_t      sBalReader;
/*OSAL variables*/
phOsal_Stub_DataParams_t        sOsal;
/* others */
phpalI14443p3a_Sw_DataParams_t  I14443p3a;
void        *pHal;
```

  - Initialize and activate the card

# Use case - Polling & Interrupt

- Polling

Query TAG info

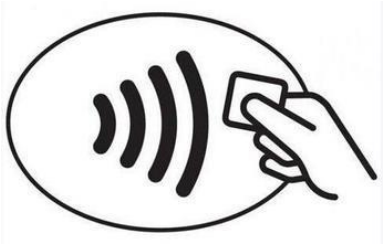
Reply TAG info

APP

NFC + BLE

- Interrupt

Notify TAG info



# Hands on

- How to change device name

  A: Use NVDS tool

- How to Auto-advertise on power on

  A: In app_config.h, enable the definition of below line.

  // #define QN_DEMO_AUTO 1

# Hands on

- How to enable/disable Sleep mode

  A: In usr_config.h, enable or disable the following line.

  /// BLE Sleep

  #define CFG_BLE_SLEEP

- How to enable/disable Deep sleep mode

  A: In usr_config.h, enable or disable the following line.

  /// Deep sleep support

  #define CFG_DEEP_SLEEP

- How to send and receive data using QPP (where can it be hard coded)

  A: The place to receive data: app_qpps_data_ind_handler

  The function app_test_send_data will be called to send data. Users can modify val[] array to try sending different data.

# Q&A

SECURE CONNECTIONS
FOR A SMARTER WORLD

# ATTRIBUTION STATEMENT