Why choose one single wireless protocol if you can have them all?

Nordic Semiconductor ASA

Larry Tsai

November 2018

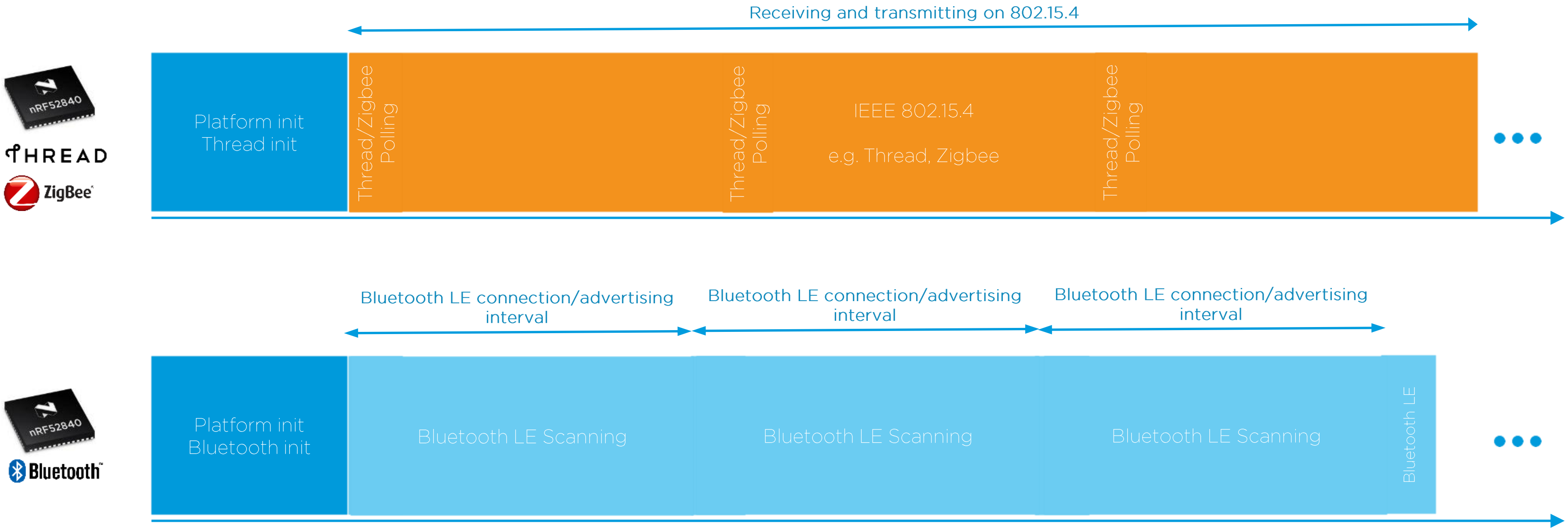
There are numerous of wireless standards

- There is no single protocol that covers all use cases
- Only selected protocols are supported by modern mobile phones and tablets
- Consumers don't understand protocol limitations
- Protocols speak different languages (both on application and physical layer)
- Let's imagine that we don't have to choose!

Why our embedded radios cannot implement them all?

- Radio transceivers dedicated for consumer electronics are based on hardware modulators, demodulators, front-end etc.
- How PHYs of protocols are different?
  - Frequencies
  - Modulation
  - Coding
  - Bitrate
  - Spread spectrum techniques
- How do we overcome these differences?

Option 1: use two physical radios



Is then using two radio technically flawless?

- Assumptions:

  - Both radios share the same frequency band

  - Both radios installed in the same equipment

- Affected:

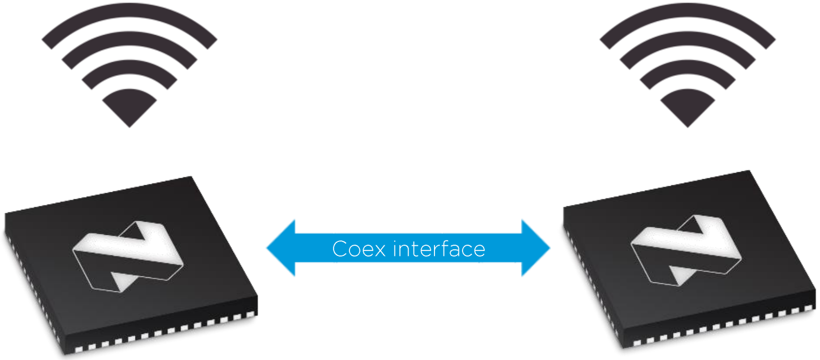
  Thread, Zigbee, Bluetooth LE, Wi-Fi transceivers etc.

- Solutions:

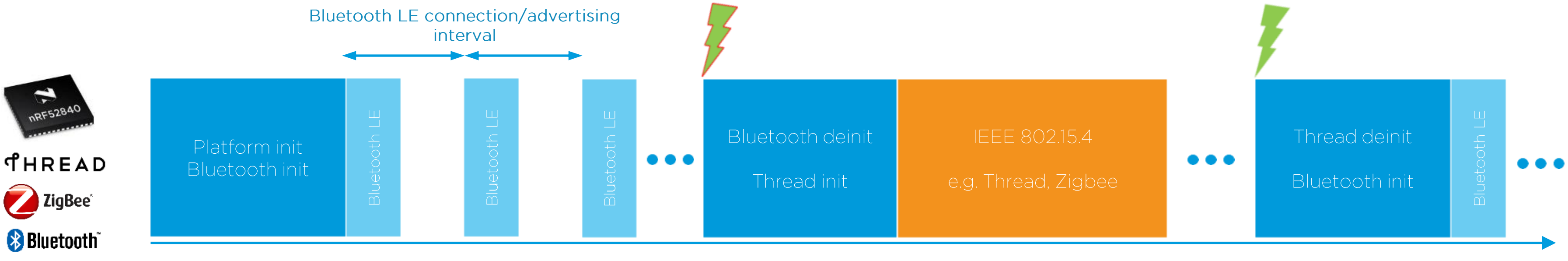
  - Spatial isolation

  - Frequency isolation

  - Collaborative coexistence



Option 2: Single SoC – switched multiprotocol



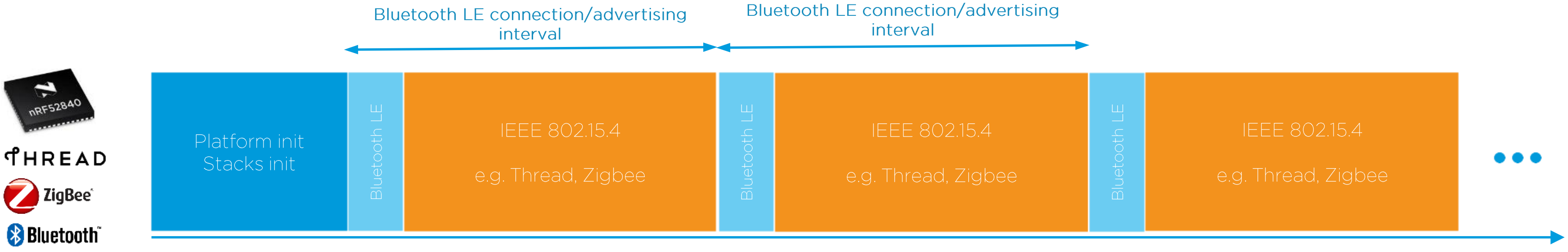
Pros (+) / Cons (-)

- (+) Protocols do not interfere with each other
- (+) No need for radio arbitration
- (-) Losing connections when switching protocols
- (-) Application must be aware of multi-protocol nature of the solution
- (-) Switching time may be noticeable by user

Key Usage / Applications

- Performing provisioning
- Performing OTA Upgrade
- Diagnostics

Option 3: Single SoC – dynamic multiprotocol



Bluetooth LE connection/advertising interval

Bluetooth LE connection/advertising interval

Platform init
Stacks init

Bluetooth LE

IEEE 802.15.4

e.g. Thread, Zigbee

Bluetooth LE

IEEE 802.15.4

e.g. Thread, Zigbee

Bluetooth LE

IEEE 802.15.4

e.g. Thread, Zigbee

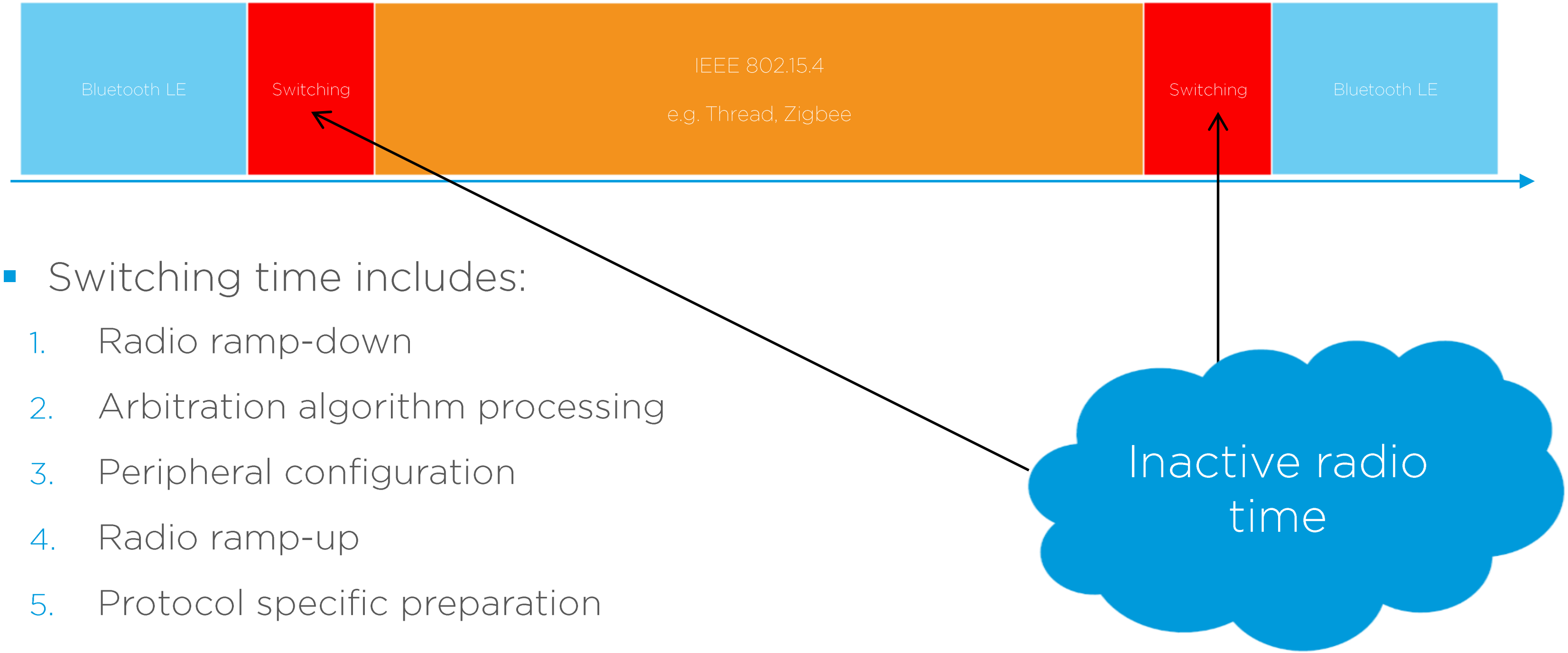
Pros (+) / Cons (-)

- (+) Lack of time-expensive initializations in between
- (+) Application developer unaware about multiple protocol being used
- (+) Connections are not ceased while switching
- (-) One protocol may decrease performance of another one (frames might be lost)

Key Applications

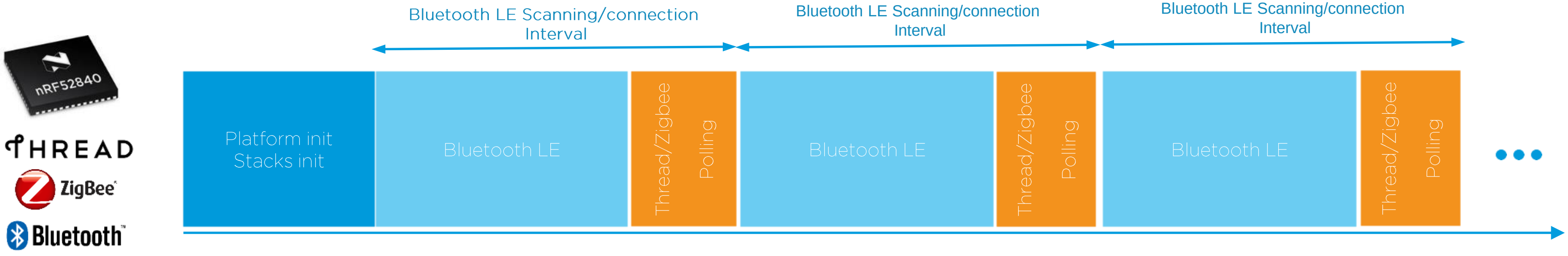
- Lighting control systems
- Sensors
- Remotes
- Asset tracking
- Leveraging Bluetooth support in smartphones to control Thread/Zigbee networks

Switching time

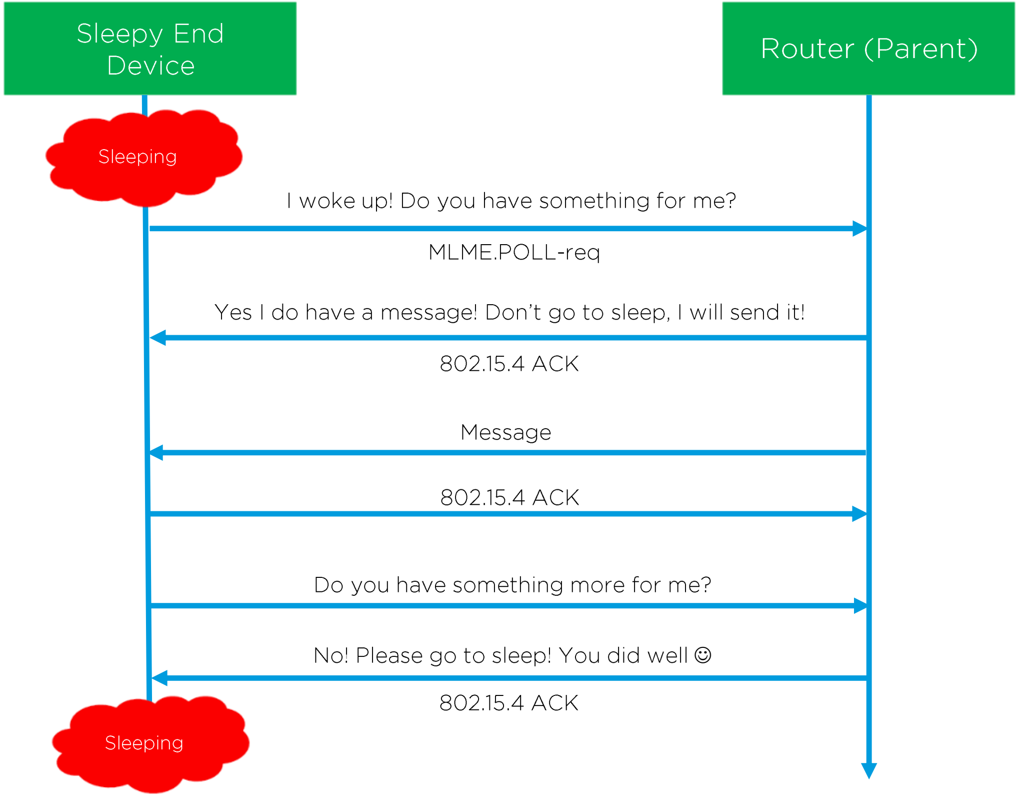


- Switching time includes:
  1. Radio ramp-down
  2. Arbitration algorithm processing
  3. Peripheral configuration
  4. Radio ramp-up
  5. Protocol specific preparation

Make use of Sleepy End Device role



Let’s limit the role to Sleepy End Device



I woke up! Do you have something for me?

MLME.POLL-req

Yes I do have a message! Don’t go to sleep, I will send it!

802.15.4 ACK

Message

802.15.4 ACK

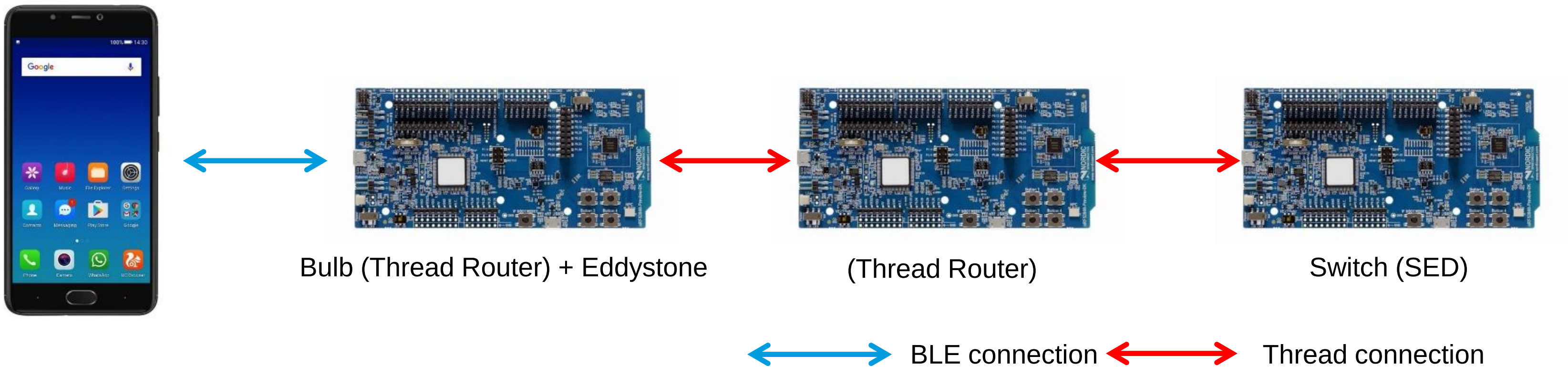
Do you have something more for me?

No! Please go to sleep! You did well ☺

802.15.4 ACK

Thread/BLE Multiprotocol Use Cases Eddystone Beacon

- Thread Router + BLE Eddystone beacon
- Functions as a Thread dimmable light
- In addition broadcasts an Eddystone beacon



Bulb (Thread Router) + Eddystone

(Thread Router)

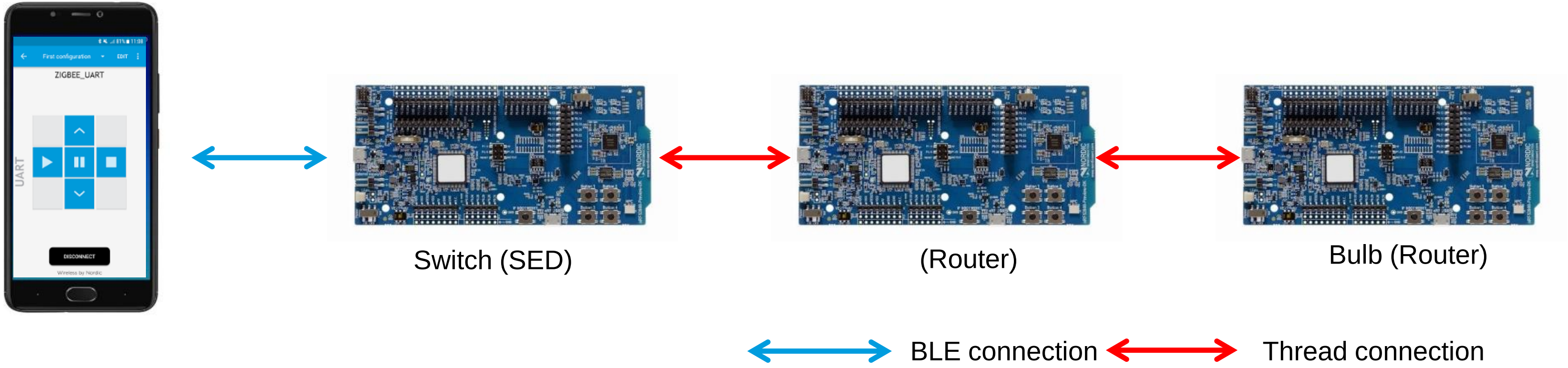
Switch (SED)

BLE connection

Thread connection

Thread/BLE Multiprotocol Examples (Nordic UART Service)

- Light switch + NUS UART service
- Functions as a Thread dimmable light
- In addition can be controlled using mobile phone via BLE



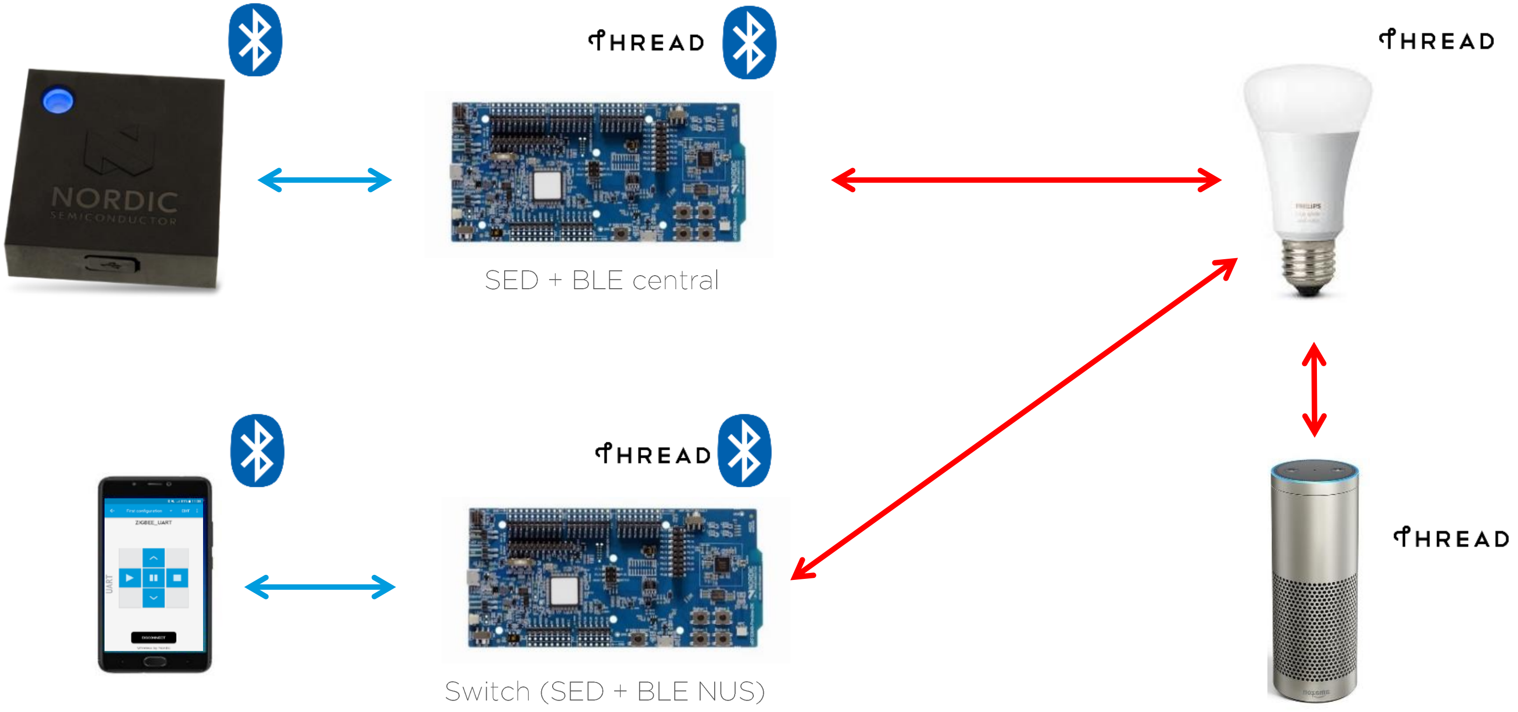
Switch (SED)

(Router)

Bulb (Router)

↔ BLE connection ↔ Thread connection

nRF5 SDK for Thread Demo



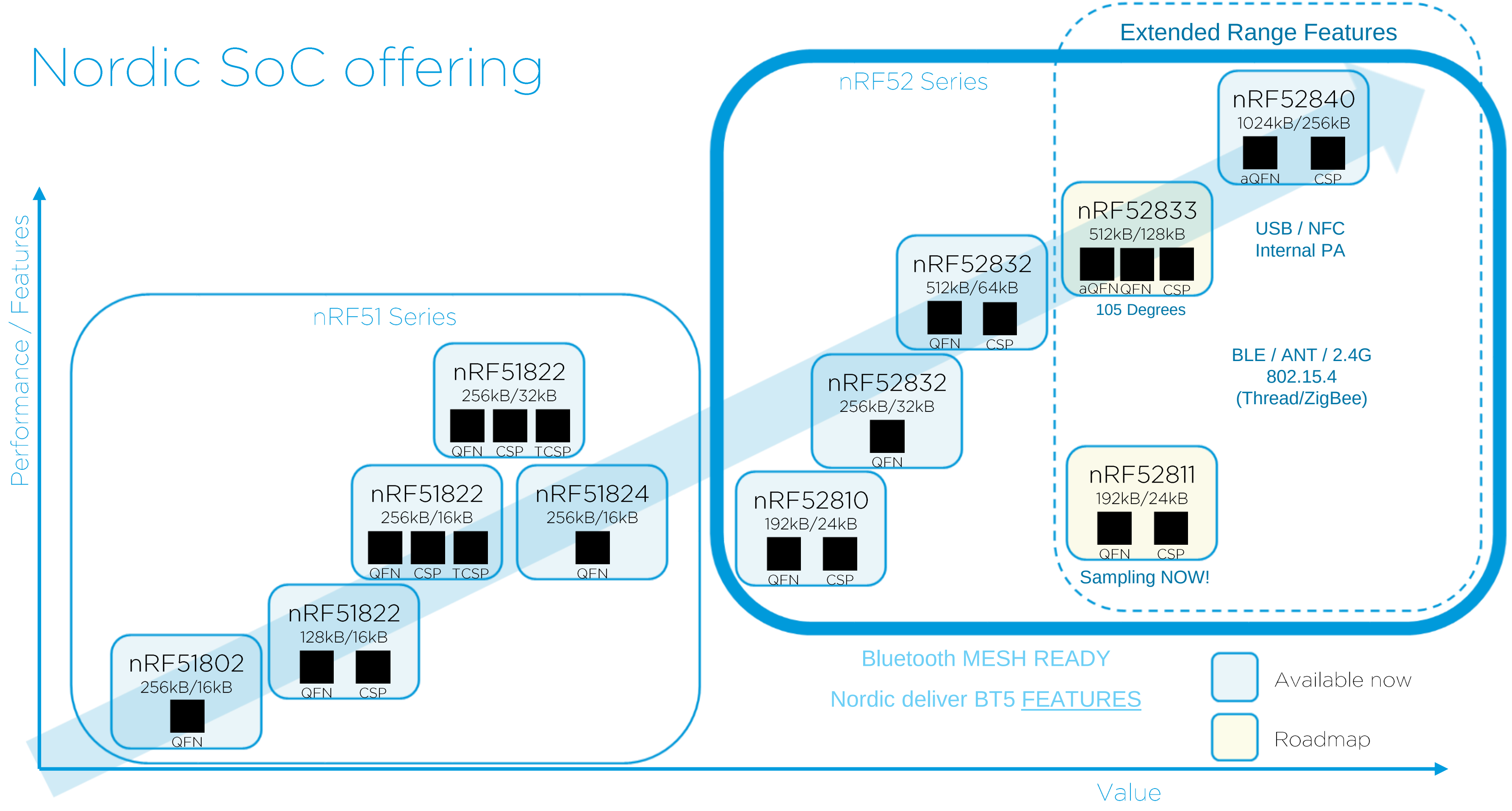
BLE Device

THREAD Thread Device

⟷ BLE connection

⟷ Thread connection

Nordic SoC offering



Bluetooth MESH READY

Nordic deliver BT5 FEATURES

Thank you for your attention!

- Visit www.nordicsemi.com
- Try out our SDKs!
  - Thread & BLE
  - Zigbee & BLE
  - ANT & BLE
  - BLE Mesh & BLE
  - All above + NFC

NORDIC
SEMICONDUCTOR
Smarter Things

NORDIC DEVELOPER ZONE
ask questions, share info, and be inspired!

Thank you for your attention!