
Bluetooth®
SPECIAL INTEREST GROUP

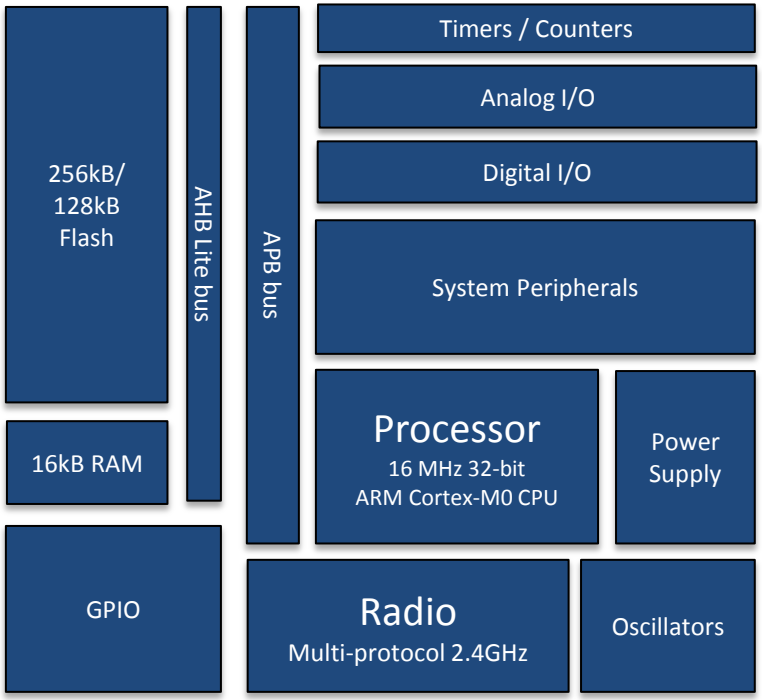
Bluetooth Innovation Training 2015

Developer Foundation, Ken Lam – Nordic Semiconductor

- Introduction to the SoftDevice (Nordic BTLE Stack)
- GAP (Generic Access Profile) and GATT (Generic Attribute Profile) Recap
- Using GAP and GATT on Nordic nRF51 series
- Using the Nordic nRF5x Plugin for Bluetooth Developer Studio
- Questions and Answers (10mins)

NORDIC SEMICONDUCTOR

nRF51 Series Device Feature Sets



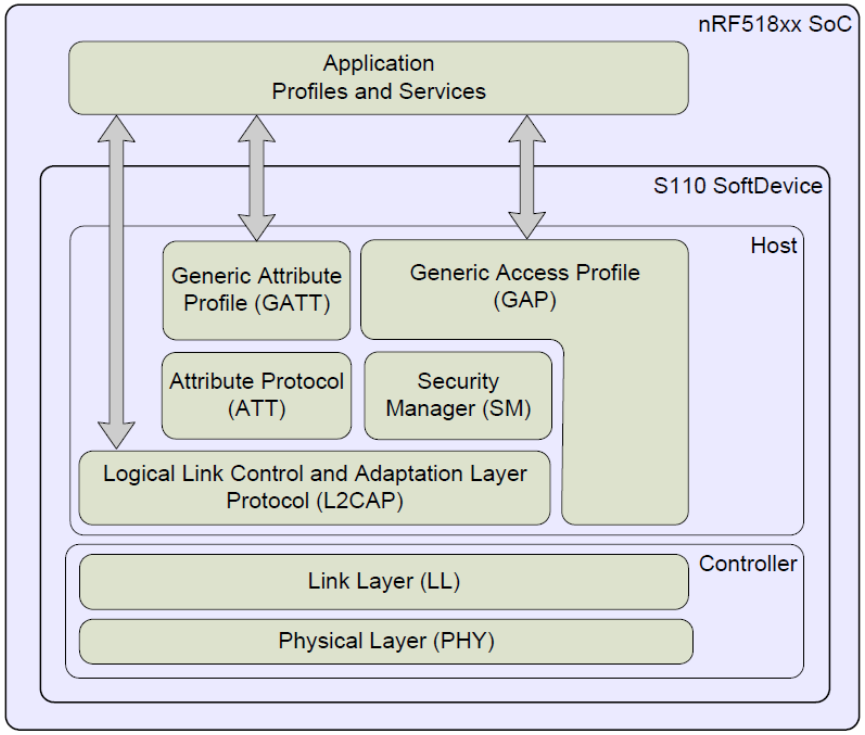
Memory	256kB Flash, 16kB RAM 128kB Flash, 16kB RAM 256kB Flash, 32kB RAM
Analog and digital I/O	2x SPI Master, SPI Slave, 2x 2-wire Master, (improved) UART, 10-bit ADC, Quadrature Demodulator , Wake-up comparator
System Peripherals	16-channel PPI, 128-bit AES ECB/CCM/AAR co-processor, RNG, Temp sensor, Watchdog
Oscillators	16/32 MHz XO, 32kHz XO 16 MHz RC, 32kHz ± 250ppm RC
Timers / Counters	2 × 16-bit Timers, 1 × 24-bit Timer , 2 × 24-bit RTC
Power Supply (Supply range)	LDO (1.8 to 3.6V), LDO bypass (1.75 to 1.95V) Buck DC/DC (2.1 to 3.6V)
Software stacks	S110, S120, S130 -*Bluetooth®* low energy solutions S210, S310 ANT+ compatible Gazell 2.4GHz RF (SDK) or customer proprietary
Package options (GPIO count)	6x6mm 48-pin QFN (31) 3.5x3.8 WCSP (32) (256KB/16KB variant only) 3.33x3.5 WCSP (32) (128KB/16KB variant only) 3.8x3.8 WCSP (32) (256KB/32KB variant only)

NORDIC SEMICONDUCTOR

Introduction to the nRF51 BLE Stack

How a BLE application works ?

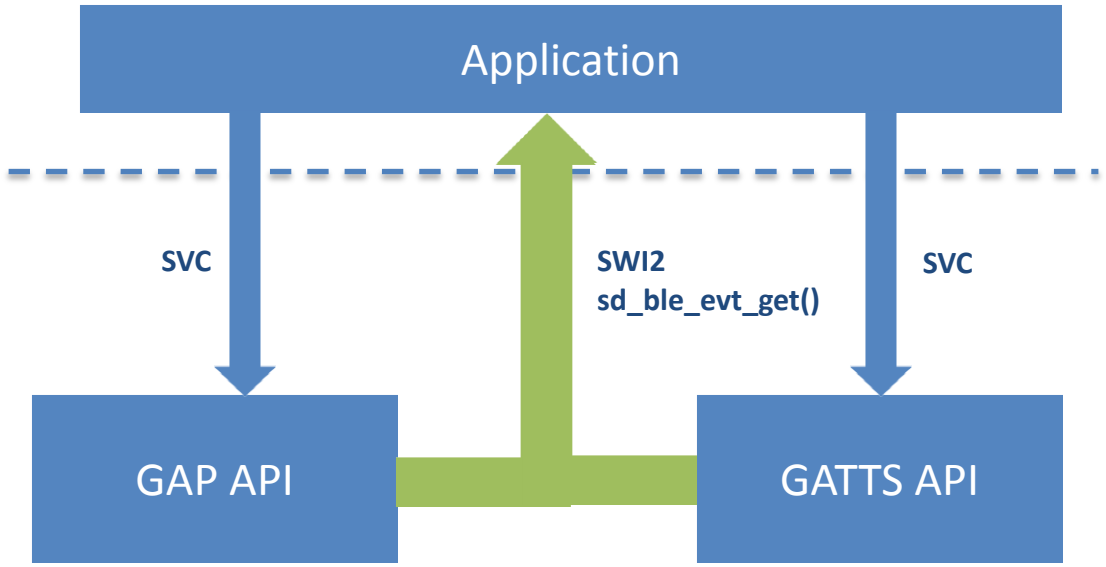
- SoftDevice is a protocol stack solution
  - that runs in a protected code area
  - Accompanying protected RAM area.
- SoftDevice is a precompiled and prelinked HEX file
  - Independent from the application
  - Can be programmed separately.



NORDIC SEMICONDUCTOR

Bluetooth® Low Energy API

- Generic Access Profile (GAP)
- Generic Attribute Profile Server
  - (GATTS)
- API calls as **S**uper**V**isor **C**alls
  - Switches Core to SV priority
  - Each SV Call numbered
- Events as **S**oft**W**are **I**nterrupts
  - Always through SWI2
  - Interrupt priority: Application Low
  - `sd_ble_evt_get()`
    - For all BLE events
    - From ISR or main context



NORDIC SEMICONDUCTOR

Bluetooth® Low Energy API

```
main()
{
    // Enable BLE event interrupt
    sd_nvic_EnableIRQ(SWI2_IRQn);

    sd_ble_gap_adv_start(adv_params);
}

void SWI2_IRQHandler(void)
{
    // Pull event from stack
    sd_ble_evt_get(m_evt_buffer, &evt_len);
    m_ble_evt_handler(m_evt_buffer);
}
```

Events

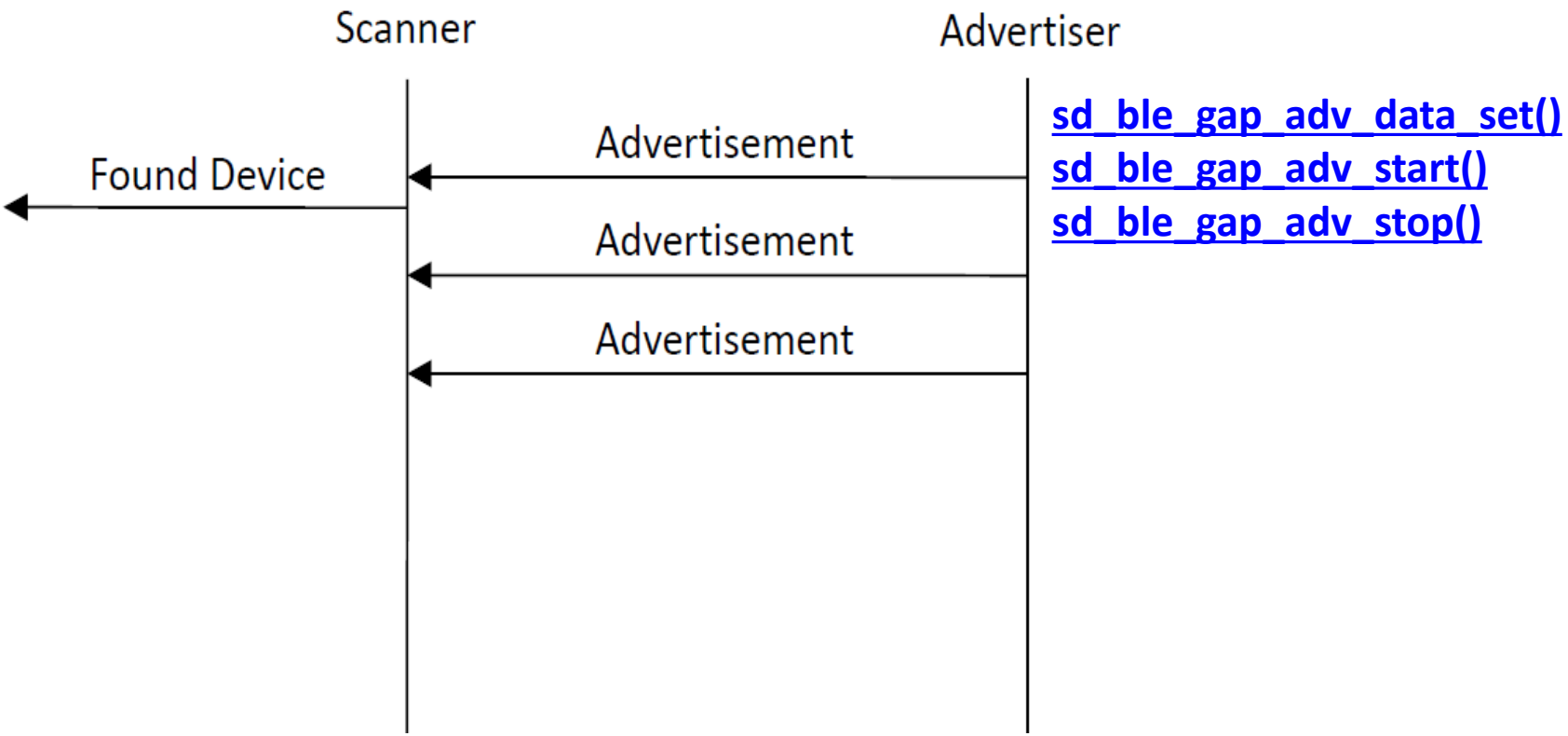
SVC

Events

NORDIC SEMICONDUCTOR

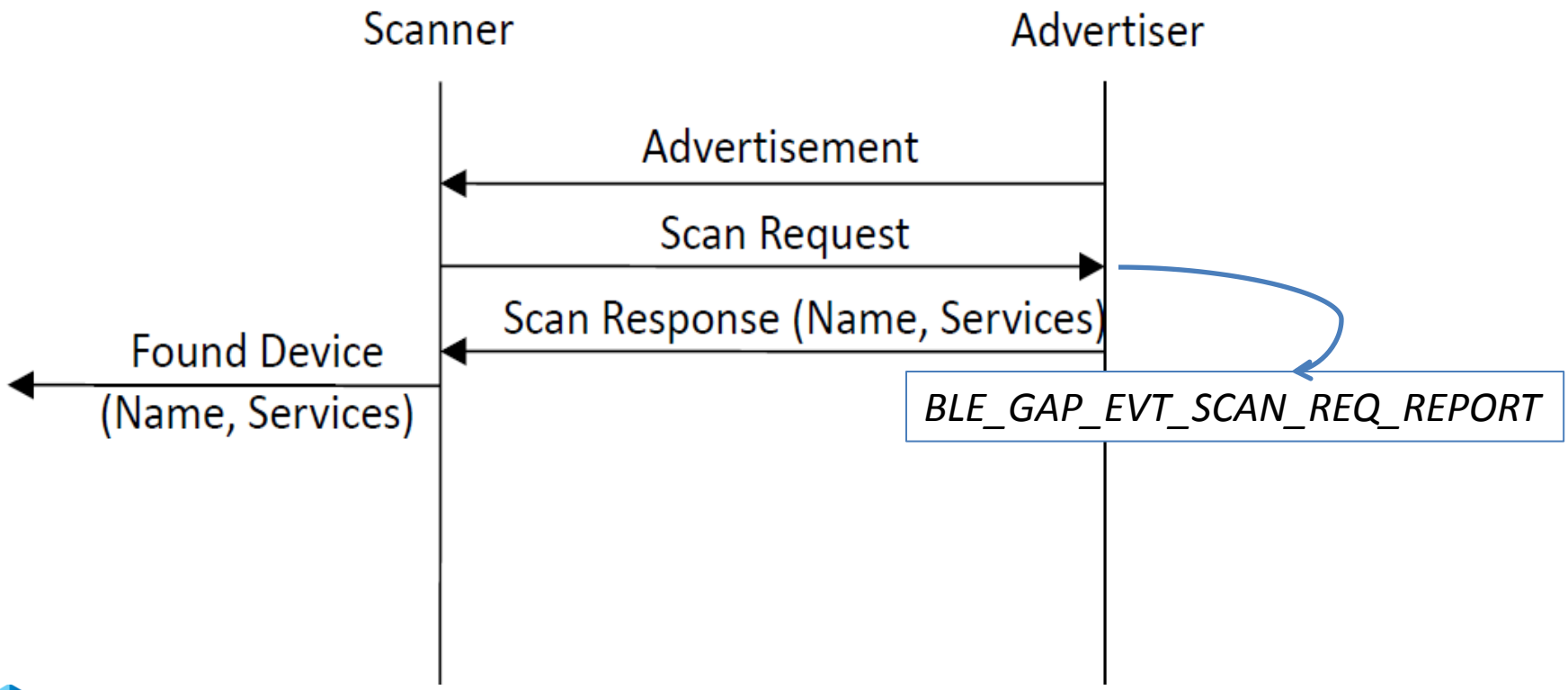
Generic Access Profile (GAP)
通用访问配置文件

Discovering Devices – Passive Scanning
发现设备 – 扫描广告包



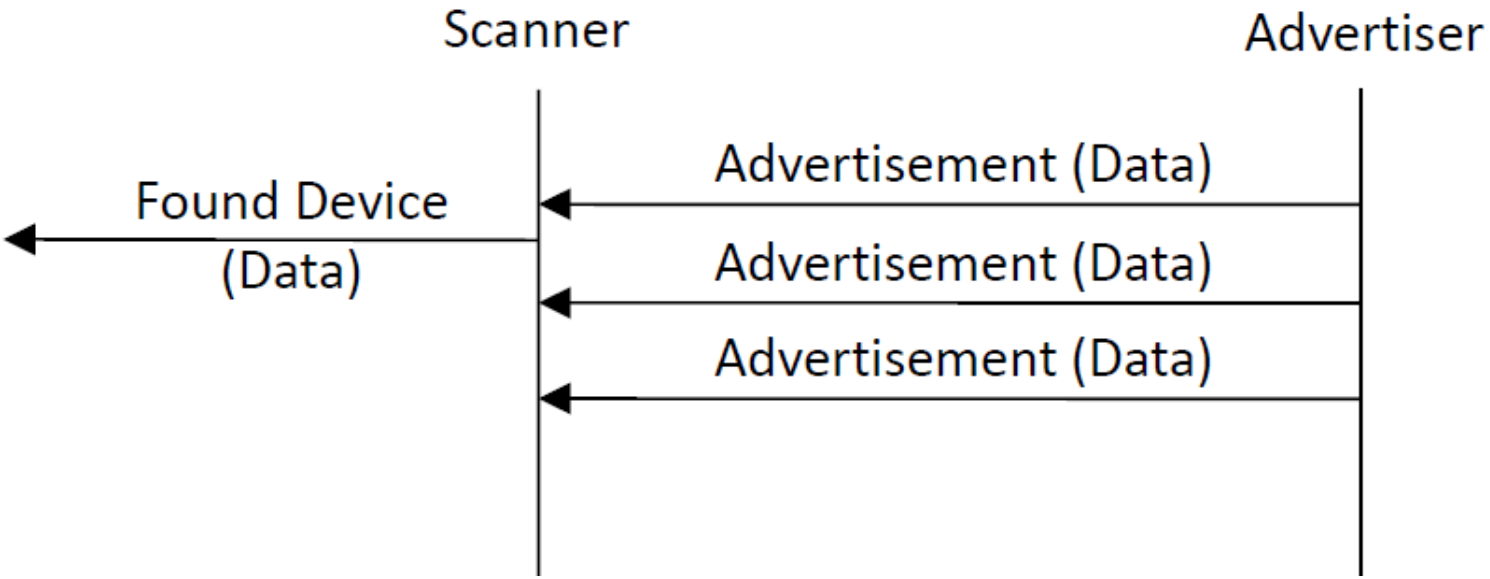
NORDIC SEMICONDUCTOR

Discovering Devices – Active Scanning
发现设备 – 扫描响应



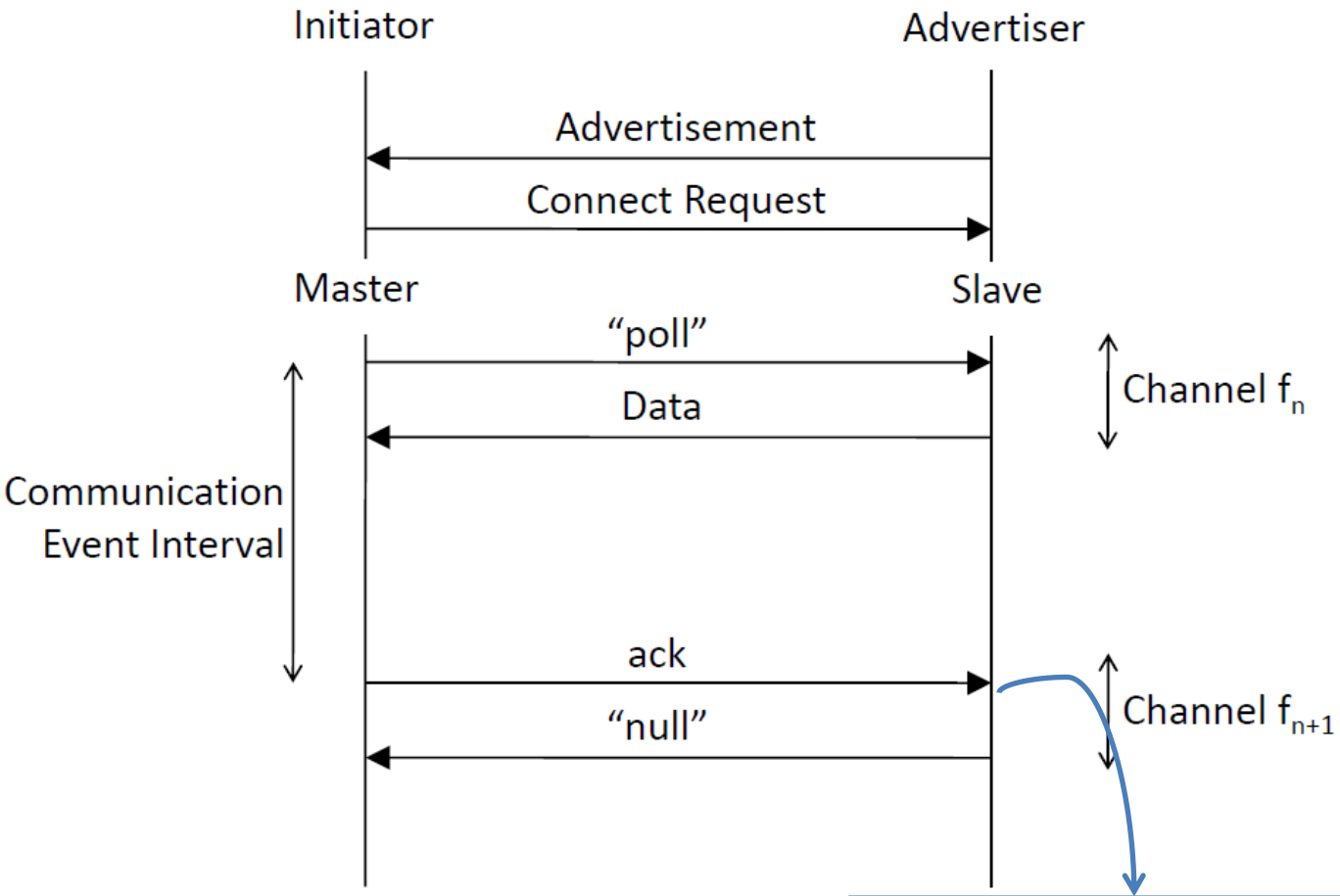
NORDIC SEMICONDUCTOR

Broadcasting Data (广播包)



NORDIC SEMICONDUCTOR

Initiating Connection (发起连接)



NORDIC SEMICONDUCTOR

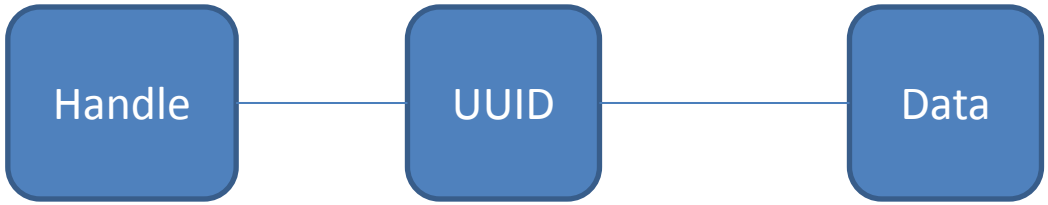
Free stuff you get

- Every piece of data that you send in a connection is **Acknowledged**
  - Acknowledgement is sent to your firmware
  - Acknowledgement is typically not send to your app on the phone
- CRC protection is present on every packet sent on the air so errors are detected
- In-order delivery
- Encryption on the link

Generic Attribute Profile(GATT)
通用属性配置文件

Relationships in GATT

唯一的序号 (index)



分配编号(Universal Unique Identifier)

Bluetooth Developer Studio

[sd_ble_gatts_service_add](#)
[sd_ble_gatts_characteristic_add](#)
[sd_ble_gatts_descriptor_add](#)

Name: Glucose Measurement

Type: org.bluetooth.characteristic.glucose_measurement

Assigned Number: 0x2A18

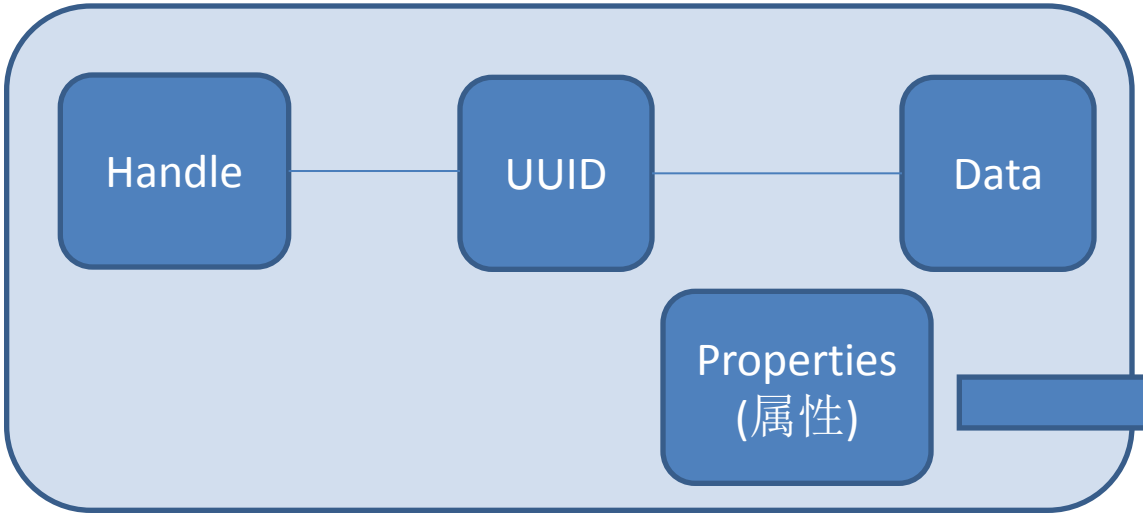
Summary:

The Glucose Measurement characteristic is a variable length structure containing a Flags field, a Sequence Number field, a Base Time field and, based upon the contents of the Flags field, may contain a Time Offset field, Glucose Concentration field, Type-Sample Location field and a Sensor Status Annunciation field. The maximum length of this structure if all Flag bits are set is 17 octets.

NORDIC SEMICONDUCTOR

Relationships in GATT

Characteristic (特征)



[sd_ble_gatts_characteristic_add]

Read
Write Request (Ack)
Write Without
Response (No Ack)
Notify (No Ack)
Indicate(Ack)

Overview	Properties

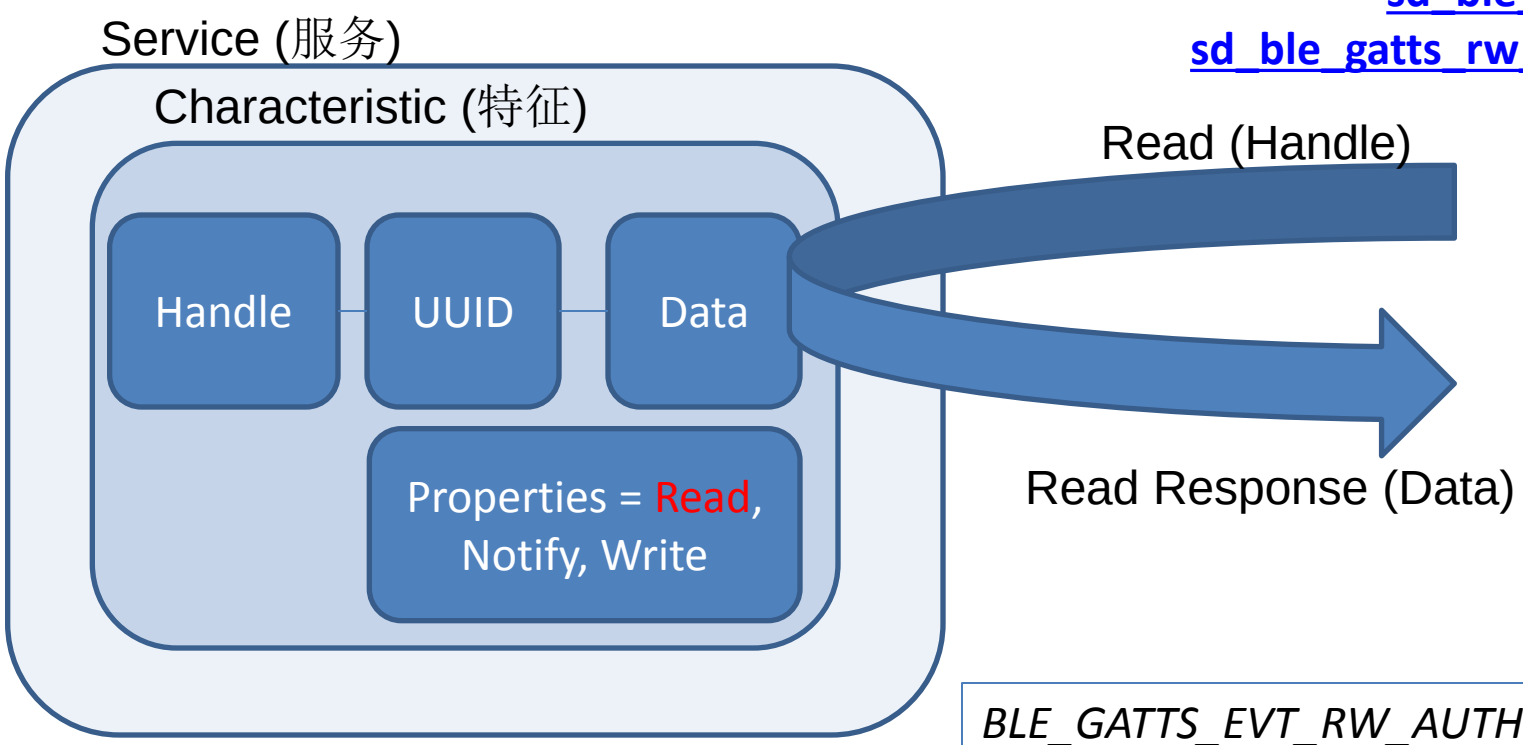
Name:
Blood Pressure Measurement
Description:
The BLOOD PRESSURE MEASUREMENT characteristic is used to send a Blood Pressure measurement.
Type:
org.bluetooth.characteristic.blood_pressure_measurement
Requirement:
Mandatory

Property	Requirement
Read	Excluded
Write	Excluded
WriteWithoutResponse	Excluded
SignedWrite	Excluded
Notify	Excluded
Indicate	Mandatory
WritableAuxiliaries	Excluded
Broadcast	Excluded

NORDIC SEMICONDUCTOR

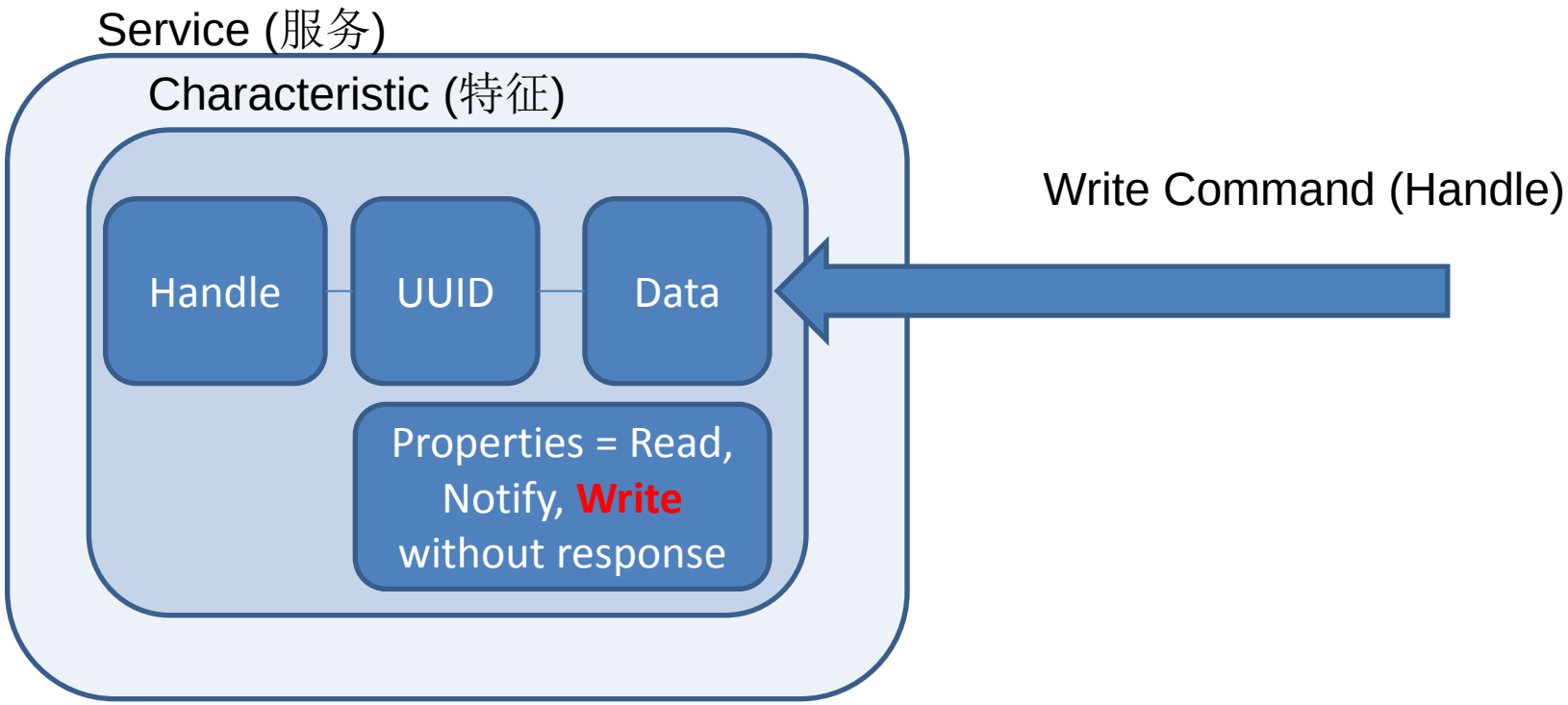
Operations in GATT

sd_ble_gatts_value_set
sd_ble_gatts_value_get
sd_ble_gatts_rw_authorize_reply



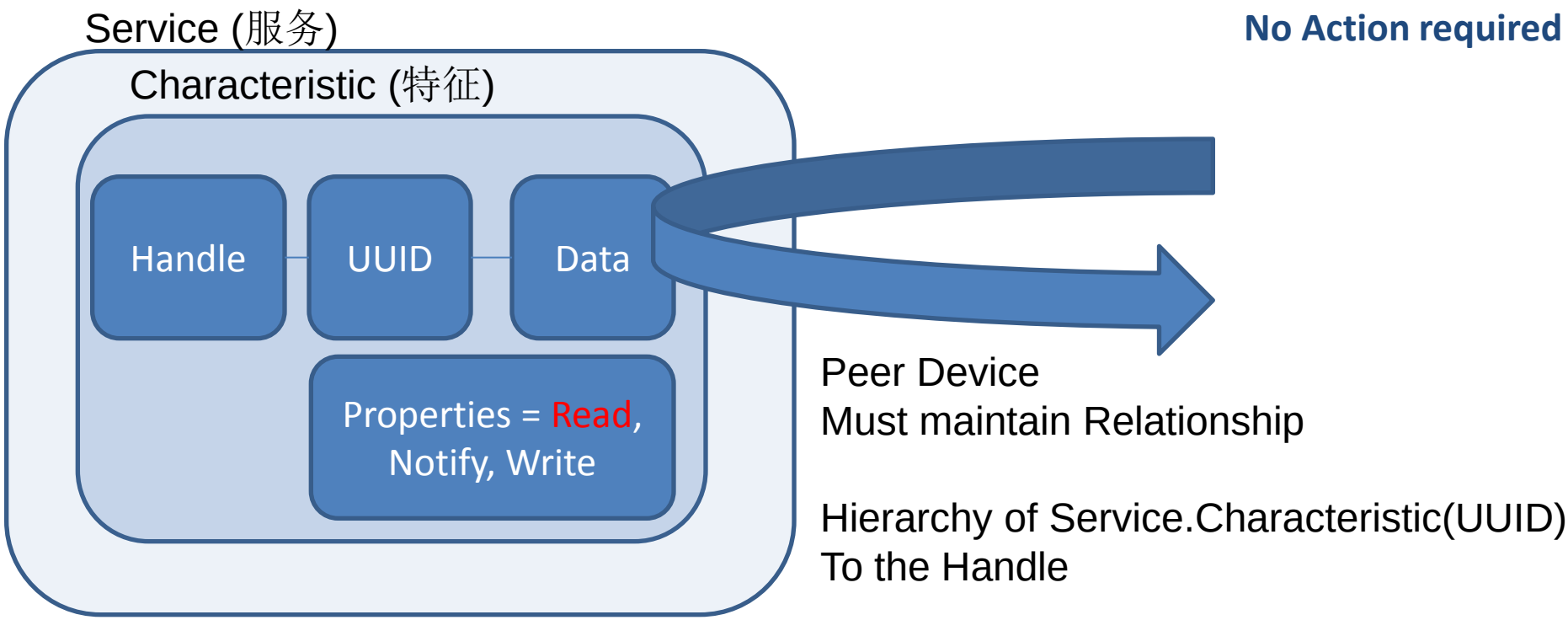
BLE_GATTS_EVT_RW_AUTHORIZE_REQUEST

Operations in GATT

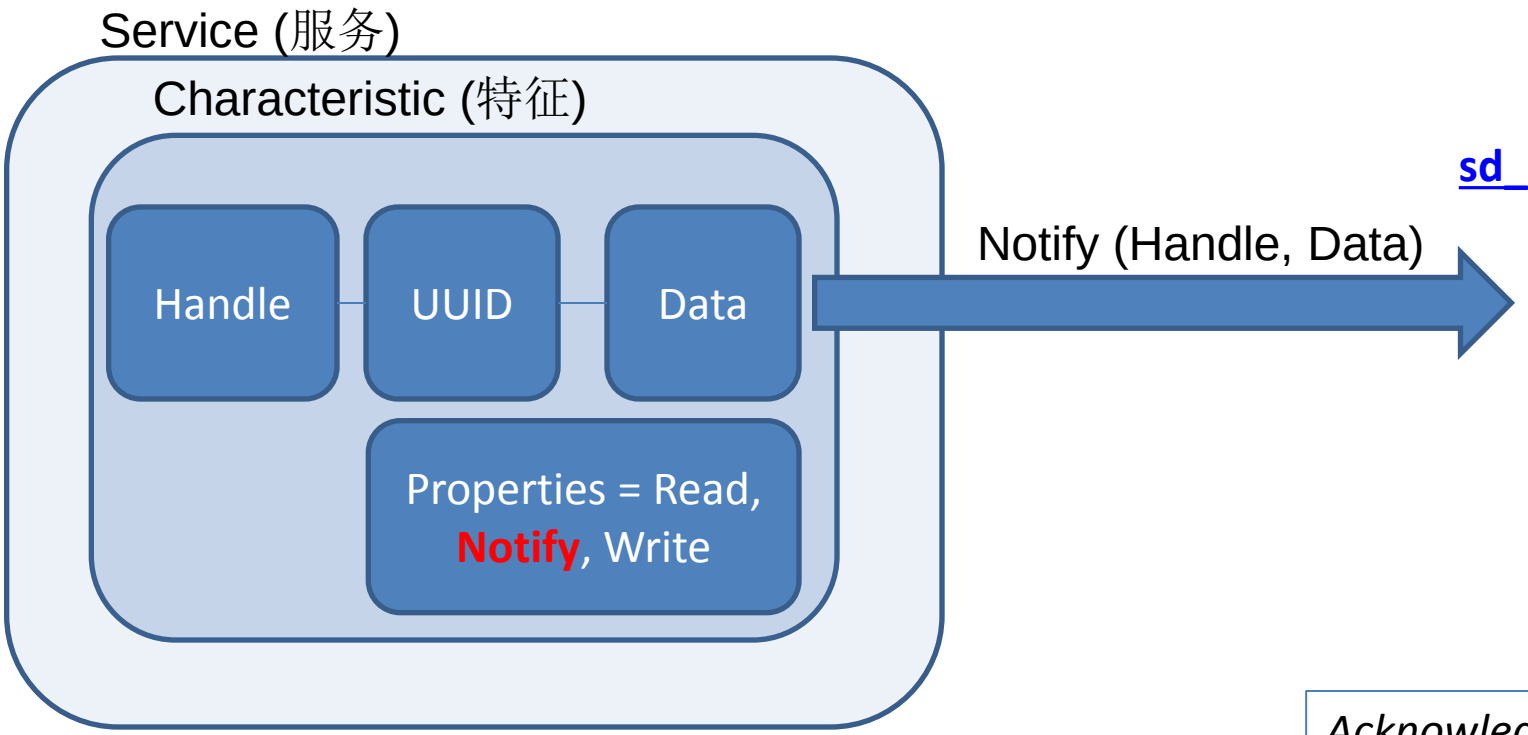


BLE_GATTS_EVT_WRITE

Service Discovery (发现服务)

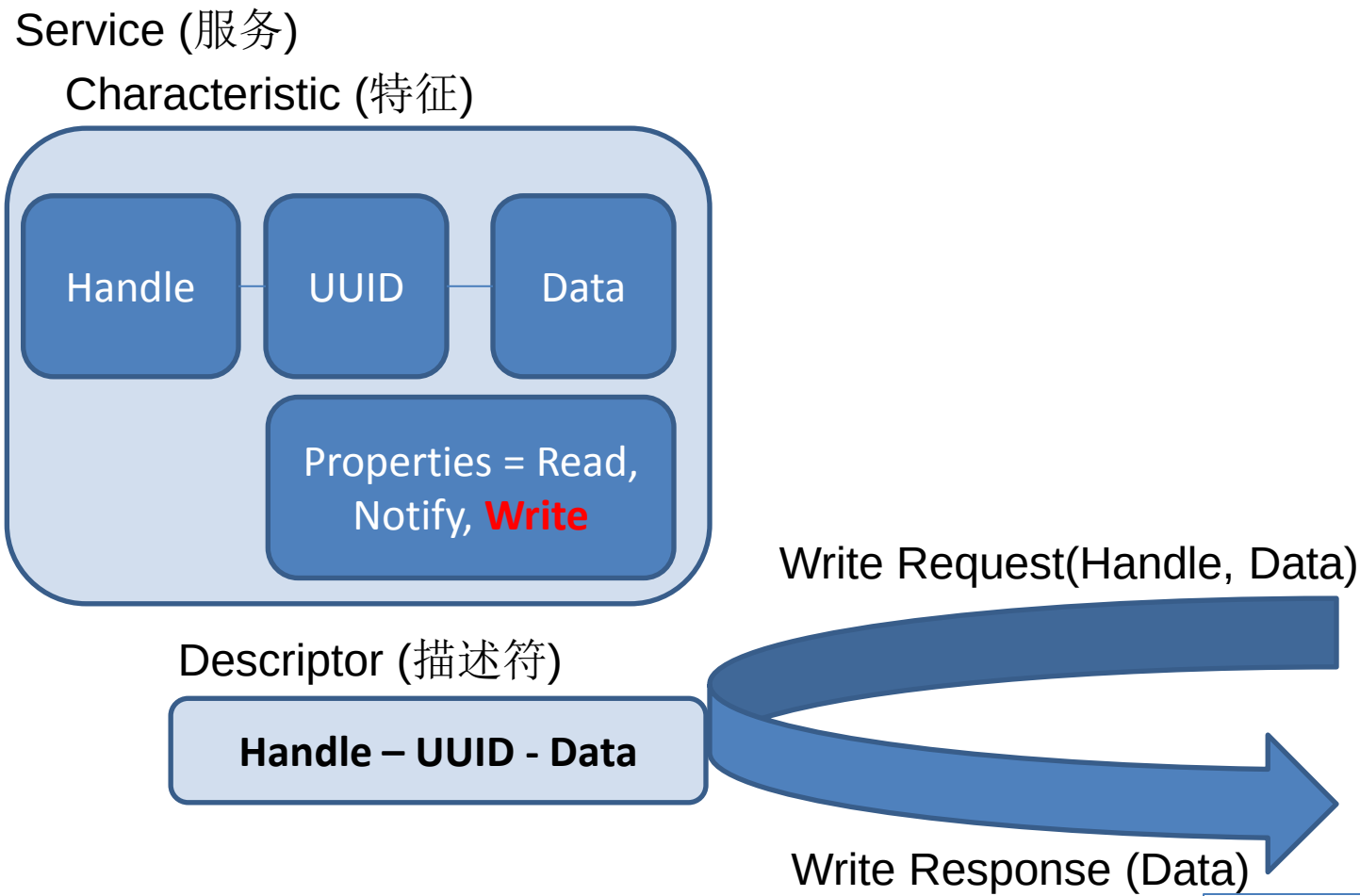


Operations in GATT



Acknowledgement Event for Notify
BLE_EVT_TX_COMPLETE

Subscribing to Notify/Indicate



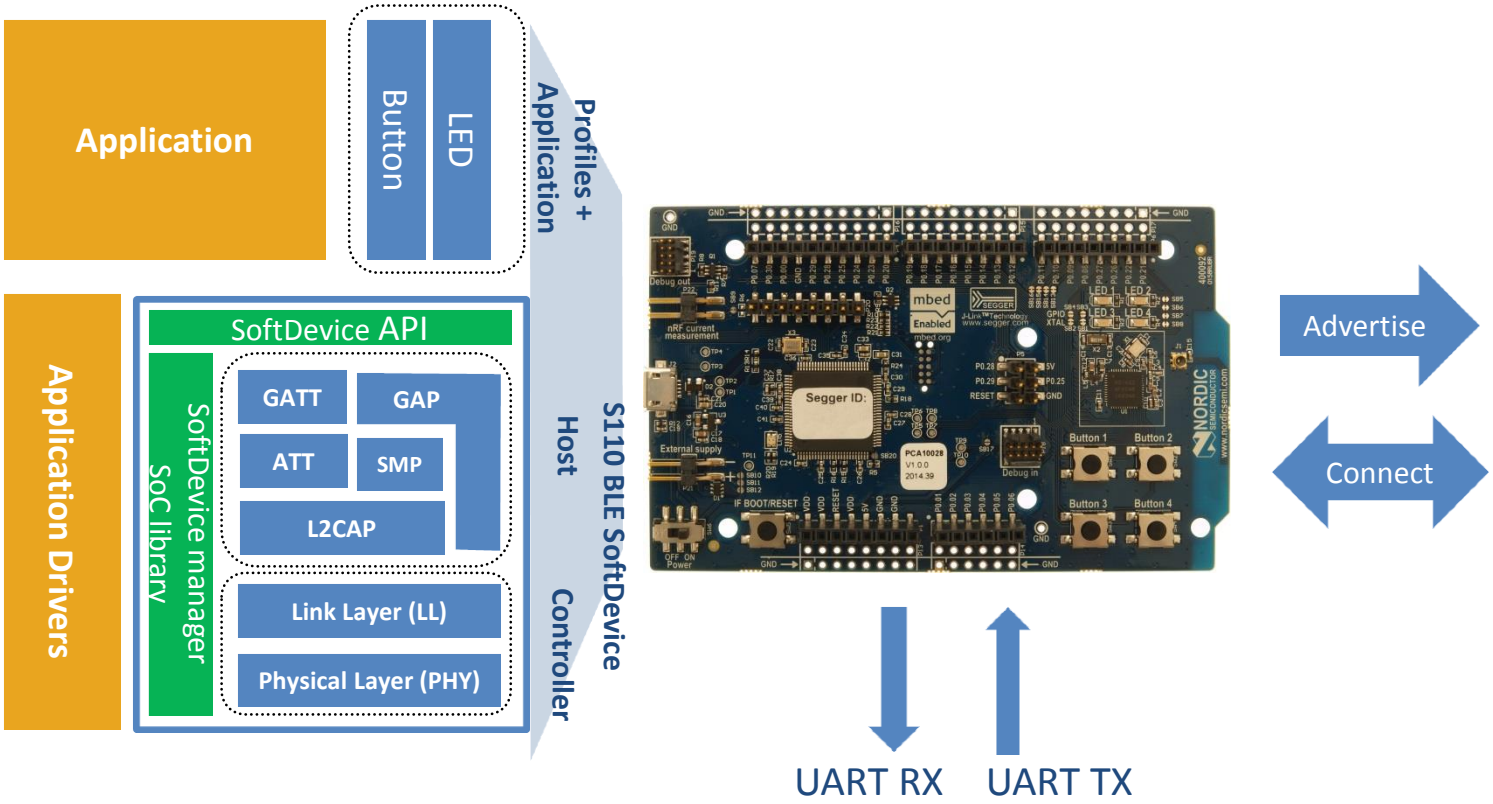
NORDIC SEMICONDUCTOR

Client Configuration Characteristic Descriptor

Permissions and Properties

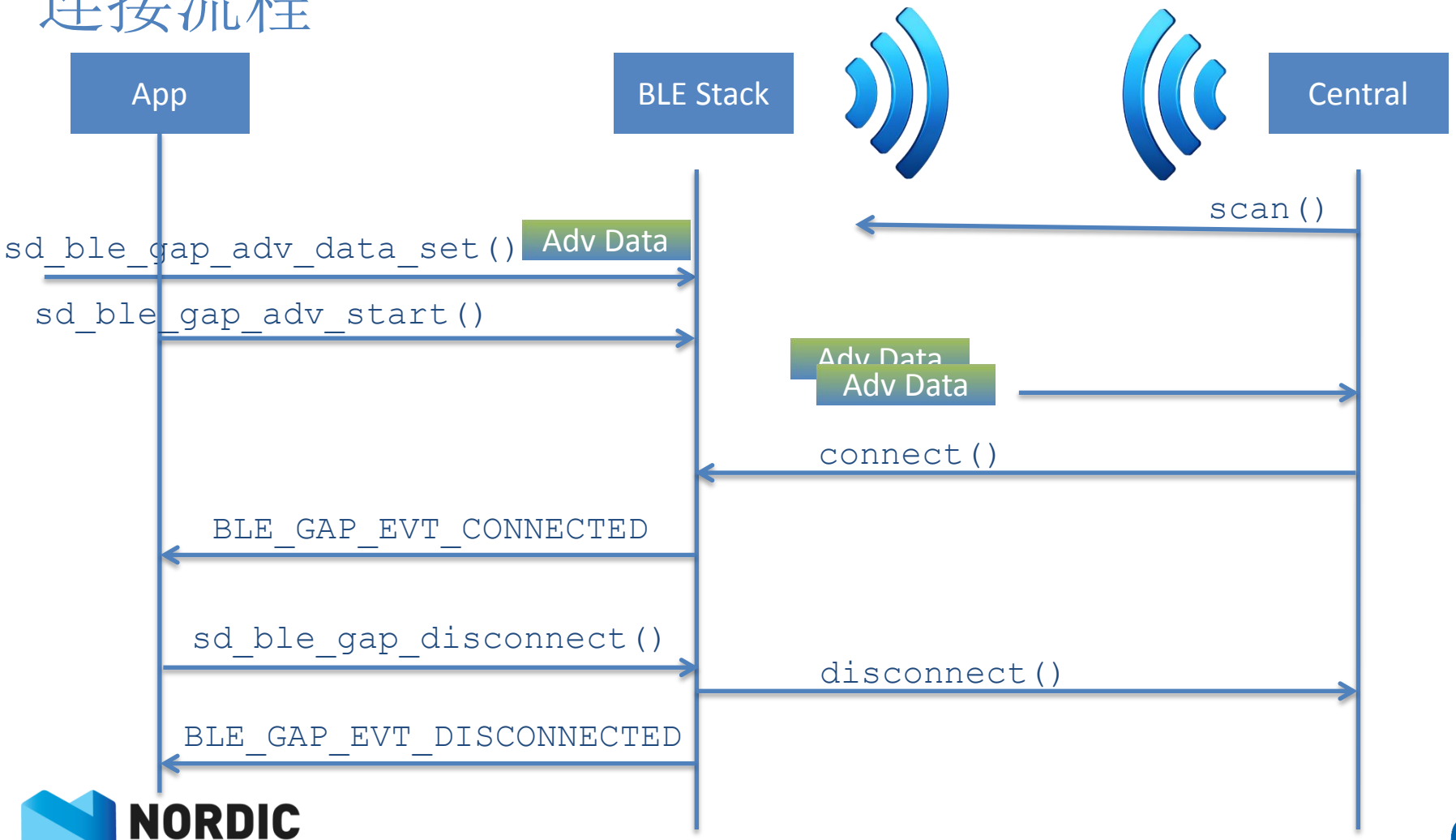
- Properties are visible over the air
- Permissions are available only locally
- Properties and Permissions must match
  - A Characteristic Read Property must have a Read Permissions
  - Permissions also include permissions to access the Characteristic when the link is Encrypted or Open

UART over BLE Application
蓝牙4.0模拟串口应用



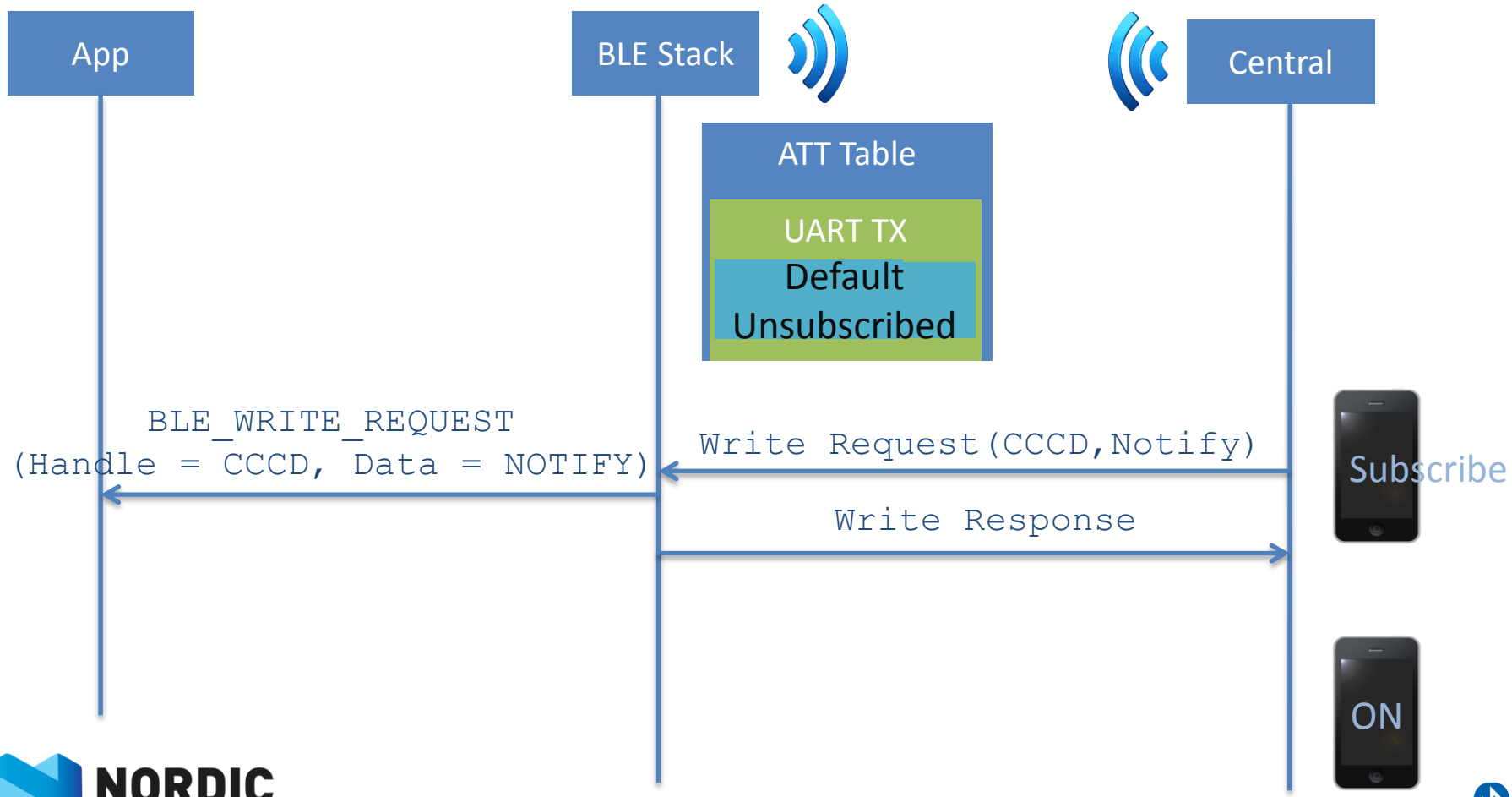
NORDIC SEMICONDUCTOR

BLE API: Connection Sequence
连接流程



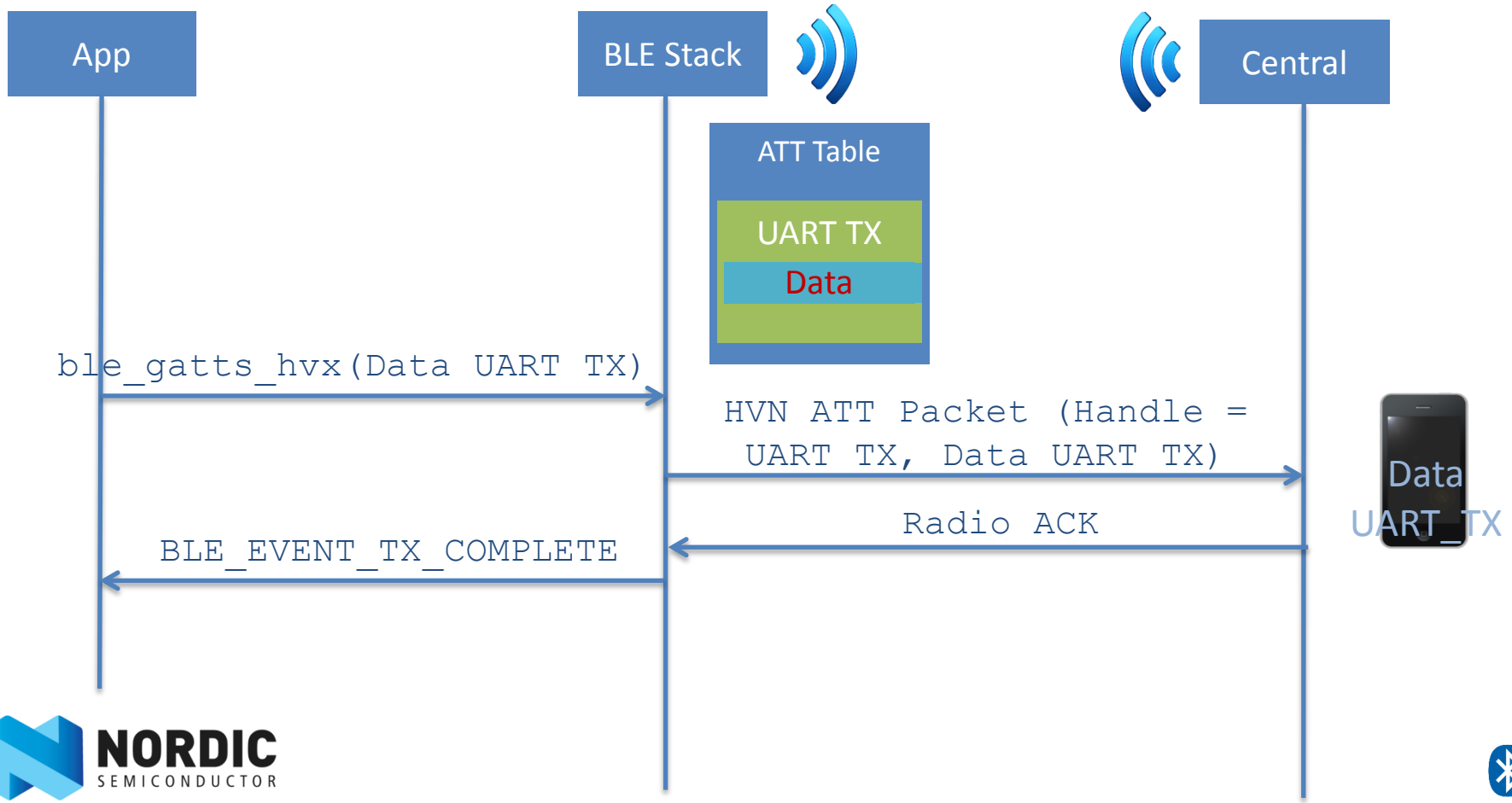
NORDIC SEMICONDUCTOR

BLE API: Handle Value Notification Master to Slave



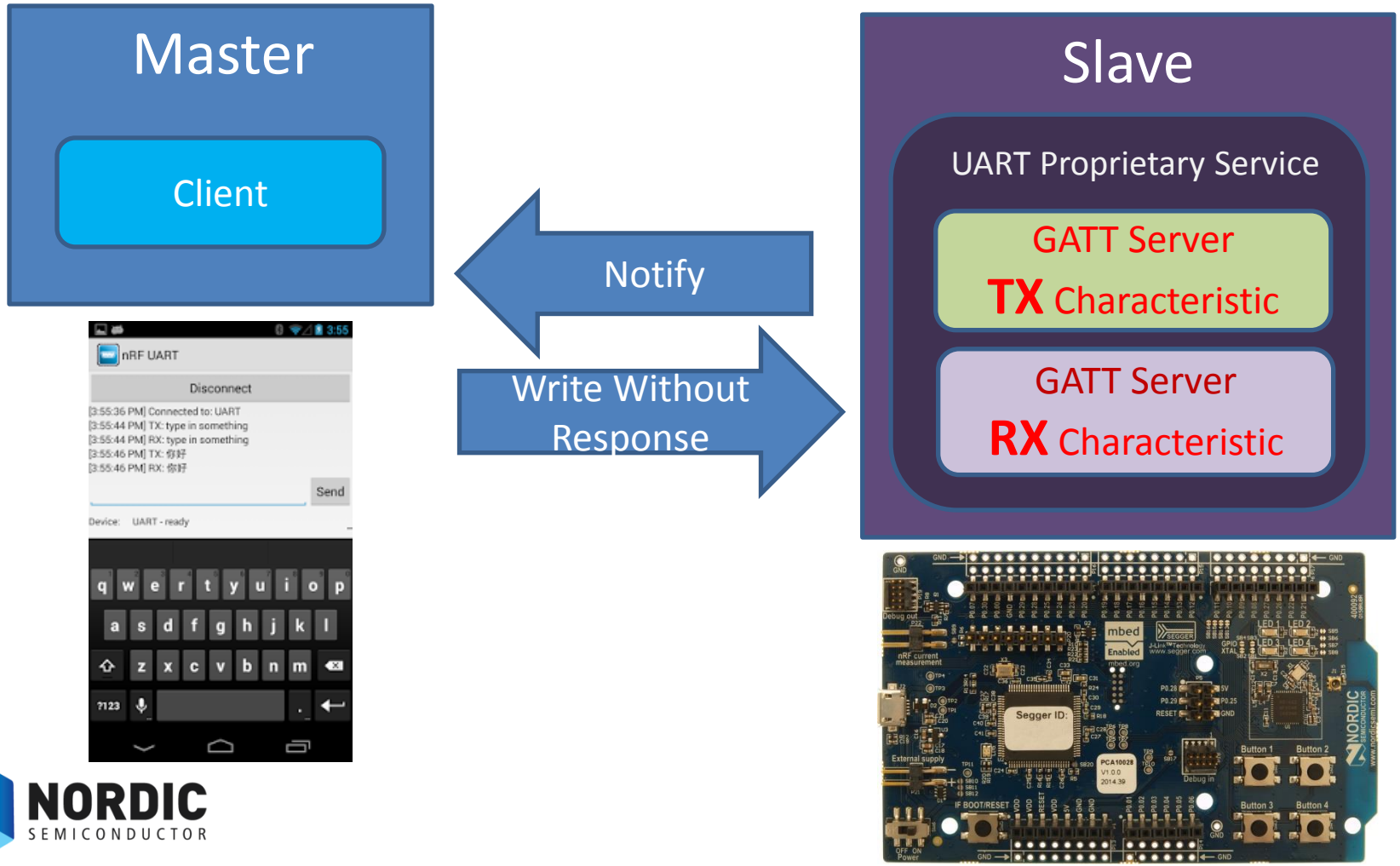
NORDIC SEMICONDUCTOR

BLE API: Handle Value Notification Slave to Master



NORDIC SEMICONDUCTOR

UART over BLE Application
蓝牙4.0模拟串口应用



NORDIC SEMICONDUCTOR

Building the Data Structure

- UART TX characteristic
  - Properties:
    - Read
    - Notification
  - Permission:
    - Read
- UART RX characteristic
  - Properties:
    - Read
    - Write without Response
  - Permission:
    - Read
    - Write

Standard versus Custom Services and Characteristics

- A UUID is a 128 bit number that is globally unique. The *Bluetooth Core Specification* separates between a base UUID and an alias
- Base UUID:
  - Bluetooth SIG base UUID:
    - 0x00001234-0000-1000-8000-00805F9B34FB
  - All Bluetooth SIG attribute will have UUID:
    - 0x0000xxxx-0000-1000-8000-00805F9B34FB
  - For UART over BLE example, an base UUID is generated:
    - 0x0000xxxx-1212-EFDE-1523-785FEABCD123

Standard versus Custom Services and Characteristics

- Alias is the 16 bit that are not defined by the Base UUID.
  - For example Battery Service UUID is 0x180F , Battery Level char is 0x2A19, etc
  - For UART over BLE example
    - Service: 0x0001
    - UART TX characteristic: 0x0002
    - UART RX characteristic 0x0003

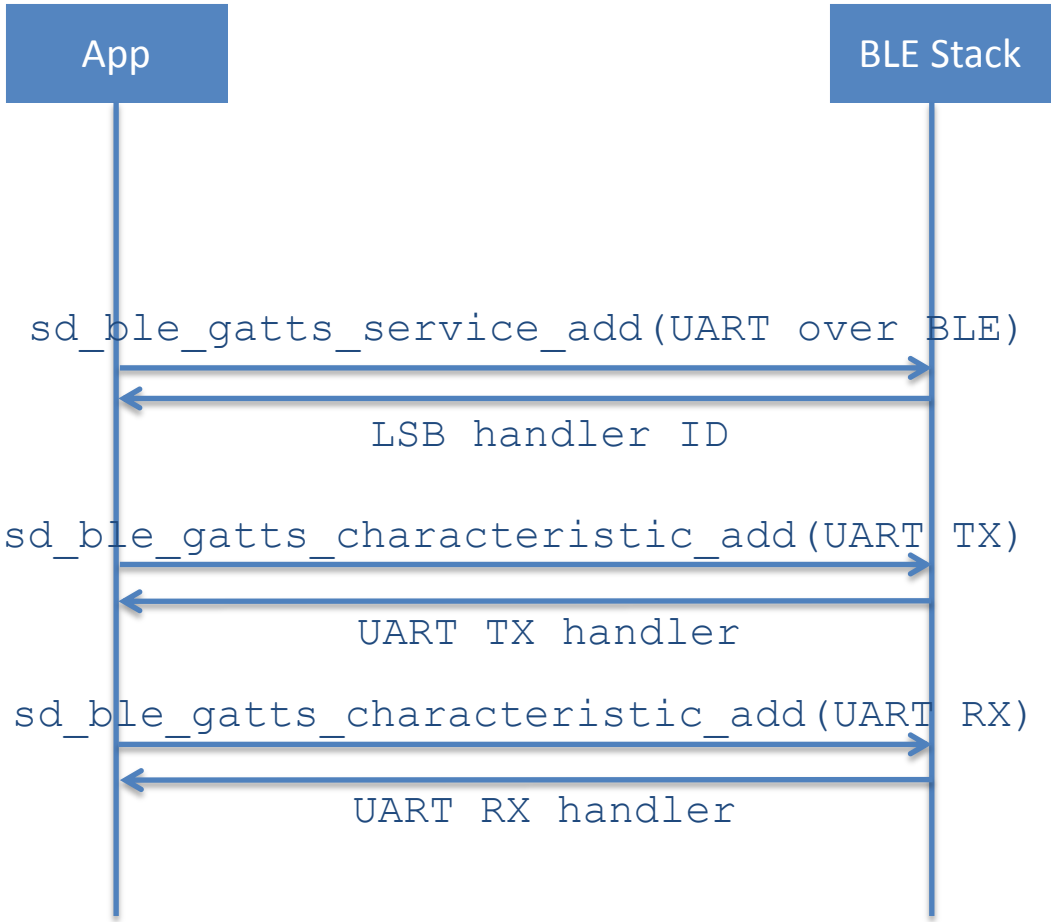
Description	UUID	Properties
UART Service	0x0000**0001**-1212-EFDE-1523-785FEABCD123	
TX Characteristic	0x0000**0002**-1212-EFDE-1523-785FEABCD123	Read, Notify
RX Characteristic	0x0000**0003**-1212-EFDE-1523-785FEABCD123	Write

NORDIC SEMICONDUCTOR

BLE API: Service Population (初始化服务)

NORDIC SEMICONDUCTOR

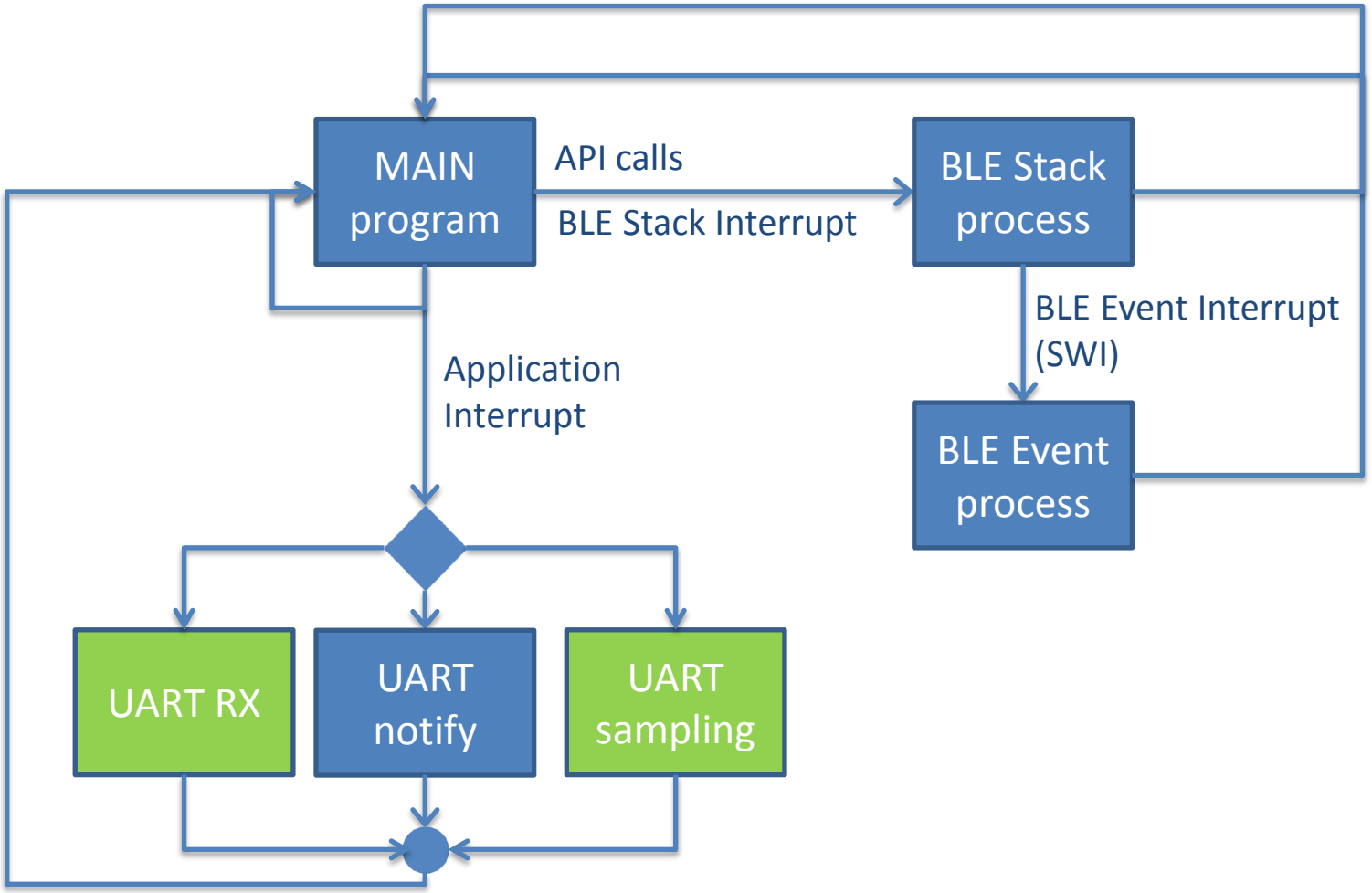
BLE API: Service Population (初始化服务)



Handle	UUID	Value
0x0001	*SERVICE*	*UOB*
0x0002	*CHAR*	*UART TX*
0x0003	*UART TX*	data
0x0004	*CHAR*	*UART RX*
0x0005	*UART RX*	*data*

NORDIC SEMICONDUCTOR

Application Block Diagram

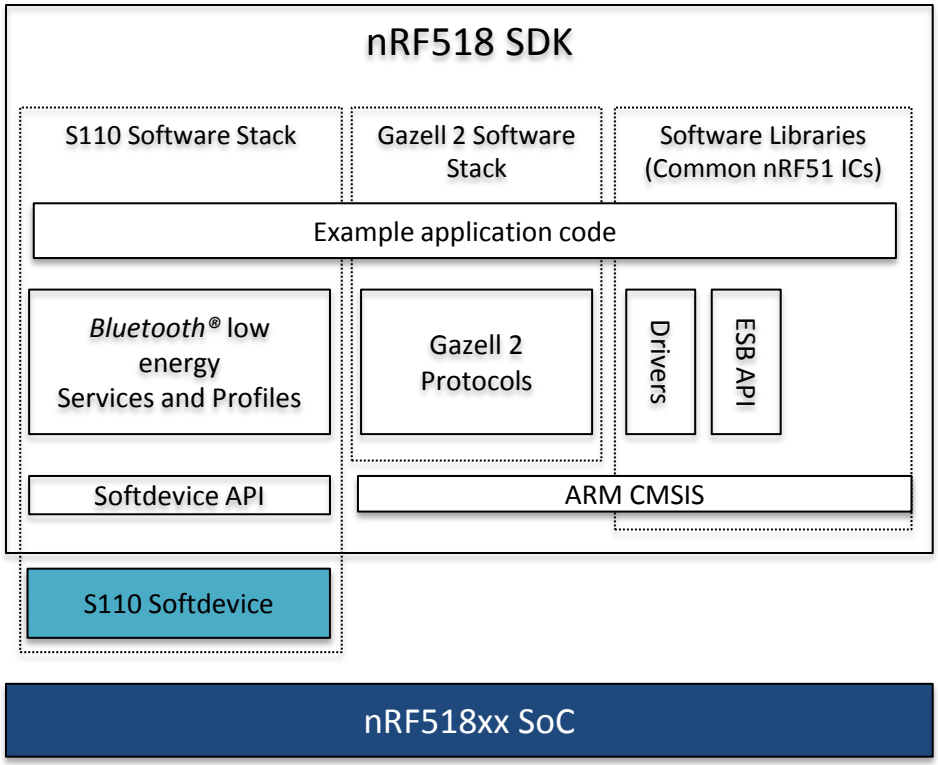


NORDIC SEMICONDUCTOR

Demo and code walkthrough

nRF51 SDK and Tool Chain

- Common SDK for nRF518xx ICs
- Software stacks
  - S1x0 *Bluetooth®* low energy
  - Gazell™ 2 Proprietary 2.4GHz RF
- Software Libraries
  - Common for all nRF51 ICs
  - ARM CMSIS (-CORE, -SVD, -DSP)
  - Enhanced ShockBurst™ API
- Keil™ MDK-ARM tool chain
  - IDE, Compiler, Programming, debug and simulation
  - Free -Lite edition up to 32kB code
- GCC/Eclipse on Windows/Mac
- IAR support



NORDIC SEMICONDUCTOR

nRF51 Tools and Software

Keil uvision IDE

Go all the way to production
with the *free* keil IDE
(32K code
size) for
windows or gcc/makefile
based toolchain
IAR and other IDEs
are also supported

nRF Tools

nrfjprog

Flash the softdevice/application/bootloader
Read, write locations in the flash and registers
Erase and manage the flash
Halt, run the CPU
Python API and other languages
Segger jlink debugger supported

mergehex

Merge the softdevice/application and bootloader
to a single hex file for production programming

nRFgo Studio

Flash the softdevice/application/bootloader
to the nRF51 DK/Dongle
Visual Editor and
power calculator for BTLE for the nRF8001 series

nRF51

nRF Sniffer and API

View, BTLE packets
Debug and learn about BTLE
Python API
Windows only from Nordic
OS X available from 3rd party

Master control Panel and API

Emulate a BTLE phone/PC
Graphical interface and API
Write test scripts and programs on the PC
API in IronPython and C#
Windows only

BLE Driver for Windows – OS X - Linux

Serialized Interface to the Softdevices from the PC
and other application controllers
Write test scripts and programs on the PC
API in C, Python and Other languages
Windows, OS X, linux

NORDIC SEMICONDUCTOR

Download the tools and software to be used with the nRF51 DK
http://www.nordicsemi.com/eng/Products/nRF51-DK#Downloads

Resources Available to Nordic Community

ask questions, share info, and be inspired!

http://devzone.nordicsemi.com/

Nordic Semiconductor
Nordic Semiconductor's official GitHub account.

https://github.com/NordicSemiconductor

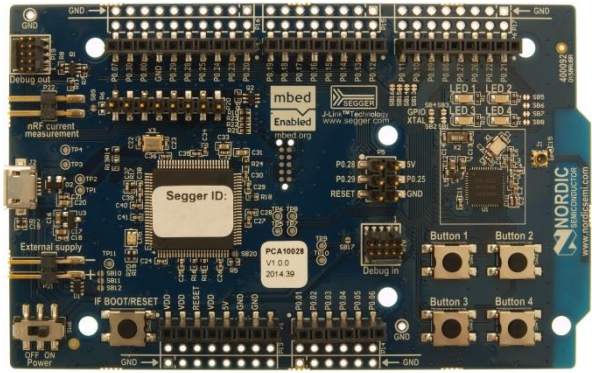
http://www.nordicsemi.com/Products/nRFready-Demo-APPS

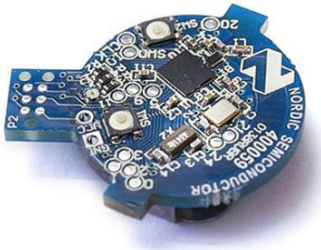
http://www.nordicsemi.com/Products/3rd-Party-Bluetooth-Smart-Modules

Get started – Hardware

http://tinyurl.com/nRF51-Develop

- **nRF51-DK**
  - Arduino UNO form factor to accept Arduino shields
  - BTLE Central/Peripheral/IoT SDK/Sample Mesh
  - Built-in Debugger and for custom boards
  - Free Keil Toolchain usable till production within 32KB Application Space
  - GCC support using makefiles
  - Flexible radio platform with support for BTLE, ANT and Proprietary 2.4GHz with concurrent operation





nRF51822-Beacon

- Small Form factor ref design
- BTLE Central/Peripheral capable
- Testing with many nodes

nRF51-Dongle

- BTLE Sniffer, Emulator for phone/Laptop
- BTLE Central/Peripheral/IoT SDK/Mesh
- Built-in Debugger

NORDIC SEMICONDUCTOR

Making it easier with Bluetooth Developer Studio

Bluetooth Developer Studio

- Benefits
  - Use it to understand the Attribute creation code
  - Can directly use generated code
- Tradeoffs
  - One-off generation of code
  - Code will not be in sync with BDS XML once edited

Bluetooth Developer Studio

- The generated code and the interfaces to the Nordic Infrastructure

Bluetooth Developer Studio

- Download the 9.0.x nRF5x SDK modified for BDS
- Generate the files in BDS
- Copy all the generated files to the experimental_bluetoothds_template folder
- Add the generated .c files to the bluetoothds_template project for keil/IAR
- Build the project
- Flash the softdevice using the nRFgo studio
- Flash the project from Keil/IAR

Key parts of the plugin generated code

Init
Create the BTLE Services and Characteristics

Send BTLE Events to Services

Call back to process Service Specific Events

NORDIC SEMICONDUCTOR

What is in the generated code

- services_init for the Service
  - Creates Service, Characteristic and Descriptors
- On Event handling for the Service
  - Feed the BTLE Events to the Service
  - Service will process the required Events
- Callback for the Service
  - Service will send Events to the applicaton for processing through this callback

main.c in bluetoothds_template

```
/**@brief Function for application main entry. */
int main(void)
{
    uint32_t err_code;

    ble_stack_init();
    device_manager_init(erase_bonds);
    gap_params_init();
    advertising_init();
    services_init();  /* The entry point bluetooth_init() called */
    conn_params_init();

    err_code = ble_advertising_start(BLE_ADV_MODE_FAST);

    // Enter main loop.
    for (;;)
    {
        power_manage();
    }
}
```

NORDIC SEMICONDUCTOR

Event handling

```
/**@brief Function for dispatching a BLE stack event to all modules
with a BLE stack event handler.
 *
 * @details This function is called from the BLE Stack event interrupt
handler after a BLE stack
 *          event has been received.
 *
 * @param[in] p_ble_evt  Bluetooth stack event.
 */
static void ble_evt_dispatch(ble_evt_t * p_ble_evt)
{
    dm_ble_evt_handler(p_ble_evt);
    ble_conn_params_on_ble_evt(p_ble_evt);
    bsp_btn_ble_on_ble_evt(p_ble_evt);
    ble_advertising_on_ble_evt(p_ble_evt);
    on_ble_evt(p_ble_evt);

/* Added for Sending Events to BDS generated Services */
    services_if_on_ble_evt(p_ble_evt);
}
```

NORDIC SEMICONDUCTOR

Generated code from the Nordic nRF51 plugin

- `uint32_t ble_txrx_service_new_service_init()`
- `void ble_txrx_service_new_service_on_ble_evt()`

```
  case BLE_GATTS_EVT_WRITE:
       on_write(p_txrx_service_new_service,
                &p_ble_evt->evt.gatts_evt.params.write);
```

- `uint32_t ble_txrx_service_new_service_tx_characteristic_set()`
- `uint32_t ble_txrx_service_new_service_tx_characteristic_send()`
- `static void on_txrx_service_new_service()` – Callback for Service Specific Events

Callback for handling events from the Service

```
static void on_scps_evt(ble_scps_t * p_scps, ble_scps_evt_t * p_evt)
{
        switch (p_evt->evt_type)
        {
                case BLE_SCPS_SCAN_INTERVAL_WINDOW_EVT_WRITE:
                        break;
                case BLE_SCPS_SCAN_REFRESH_EVT_NOTIFICATION_ENABLED:
                        break;
                case BLE_SCPS_SCAN_REFRESH_EVT_NOTIFICATION_DISABLED:
                        break;
                case BLE_SCPS_SCAN_REFRESH_EVT_CCCD_WRITE:
                        break;
```

Application to handle the events by adding the code needed to handle behavior.

Summary

- What have you learnt so far

Next Steps

- Generated code can only get you so far (baby step)
  - To build a product you need more
- **Required reading (nRF5x SDK and DK)**
  - nRF51-DK User Guide (link)
  - Getting started on the nRF51 SDK (link)
  - Creating Bluetooth Low Energy Applications Using nRF51 - Application Note (link)

NORDIC SEMICONDUCTOR

Nordic Semi Advantages

- Low Power
  - Hardware support
- Flash based for Firmware updates
- Easy to Use BTLE API
- Extensive Developer Support
  - Nordic Infocenter
  - Nordic Developer Zone
  - Nordic GitHub
  - Development-aid applications from Nordic

NORDIC SEMICONDUCTOR

Need help ?

http://devzone.nordicsemi.com/

Nordic Semiconductor
Nordic Semiconductor's official GitHub account.

https://github.com/NordicSemiconductor

Nordic Semiconductor
Support Portal

http://www.nordicsemi.com/eng/Support/Contact-Support-Team

NORDIC SEMICONDUCTOR