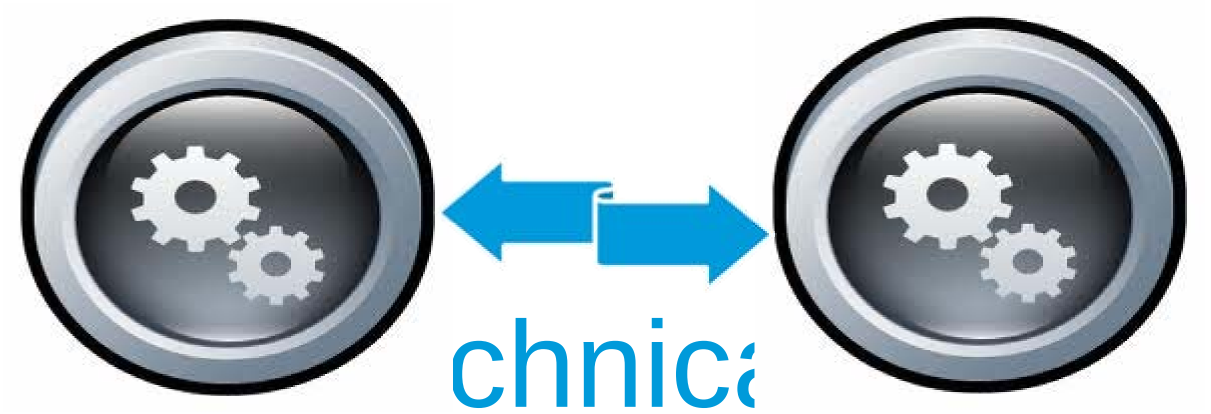
who,where, date



chnica

Training

Bluetooth Low Energy Agenda

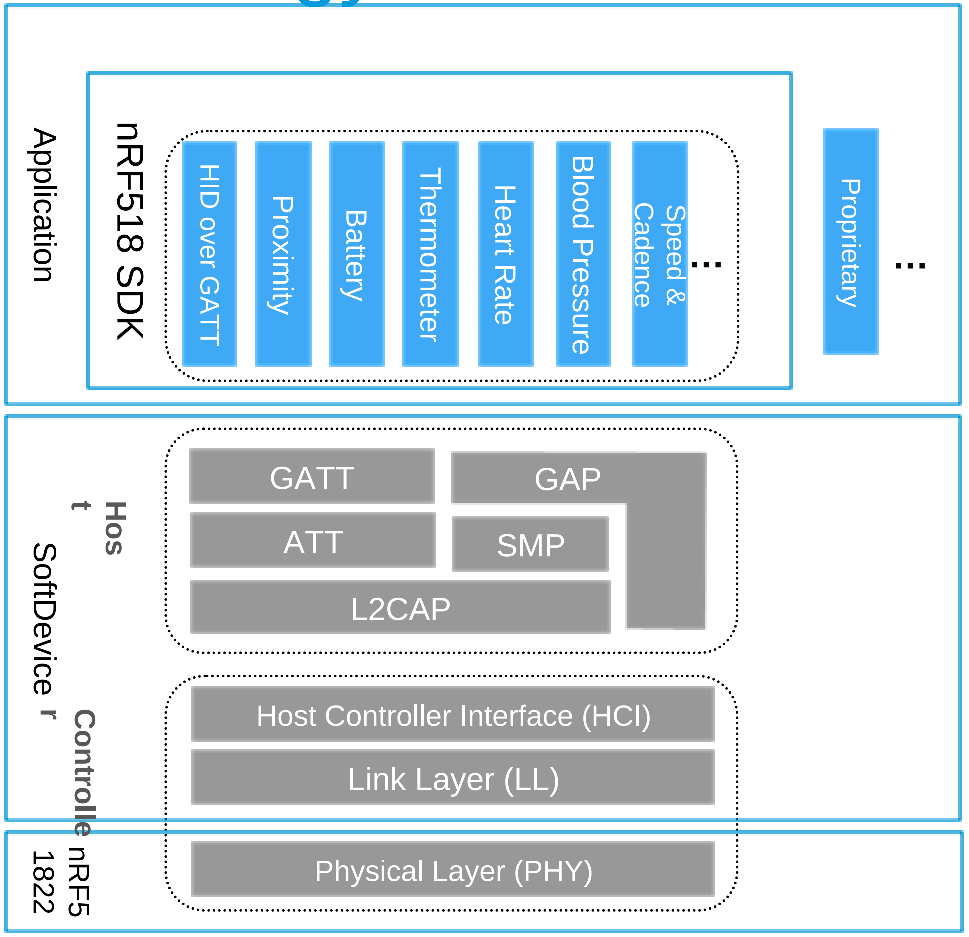
- Theory session 1:
  - Introduction.
  - Roles.
  - Advertising.
  - Scanning and connection.
- Practical session 1:
  - Changing advertising interval.
  - Adding Scan Response data.
- Theory session 2:
  - Connection parameters.
  - Service discovery.
  - Data transfer.
- Practical session 2:
  - Adding a characteristic.
- Theory session 3:
  - Security in BLE.
  - Bonding, methods.
- Practical session 3:
  - Adding security to the example.
- Theory session 4:
  - Bluetooth qualification.
- Practical session 4:
  - Bluetooth SIG web site.

Theory 1: Introduction and advertising

Introduction

- Bluetooth Low Energy is also called Bluetooth Smart.
  - Bluetooth Low Energy is not Bluetooth Classic.
- Defined by the Bluetooth 4.0 Core Specification
  - Huge document, freely available.
- Dual mode/single mode.
  - Dual mode: Can connect to Low Energy and Classic devices.
  - Single mode: Can only connect to other Low Energy devices.
- More flexible from a software perspective than Classic.

Bluetooth Low Energy overview



All BLE training: example based

- Look at all technology aspects based on an example:
  - Temperature fob.
- Should be connectable from some device, to read the temperature of the fob.

Roles

Peripheral

- Fob is peripheral.
- Slave of the link.
  - Can request specific timing, encryption.

Temperature fob

Central

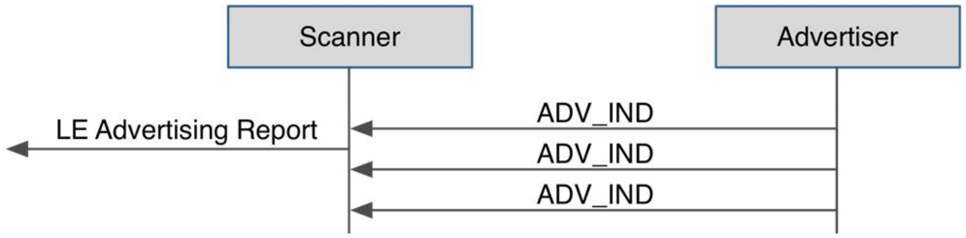
- Connected to some device.
  - This device is the central.
- Master of the link.
  - Controls everything: encryption, timing.

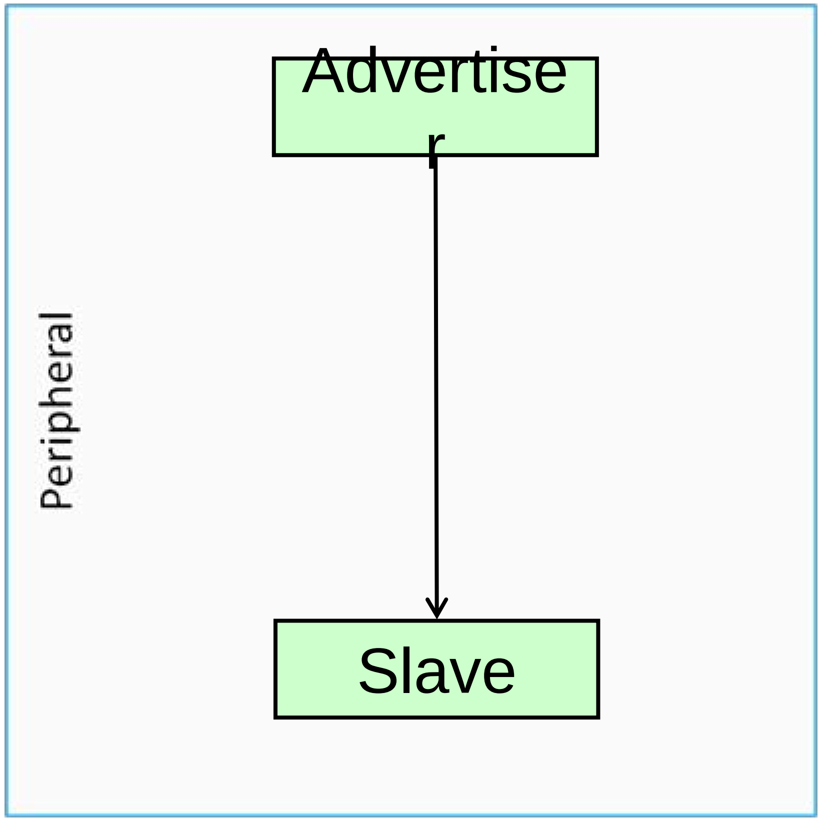
Phone

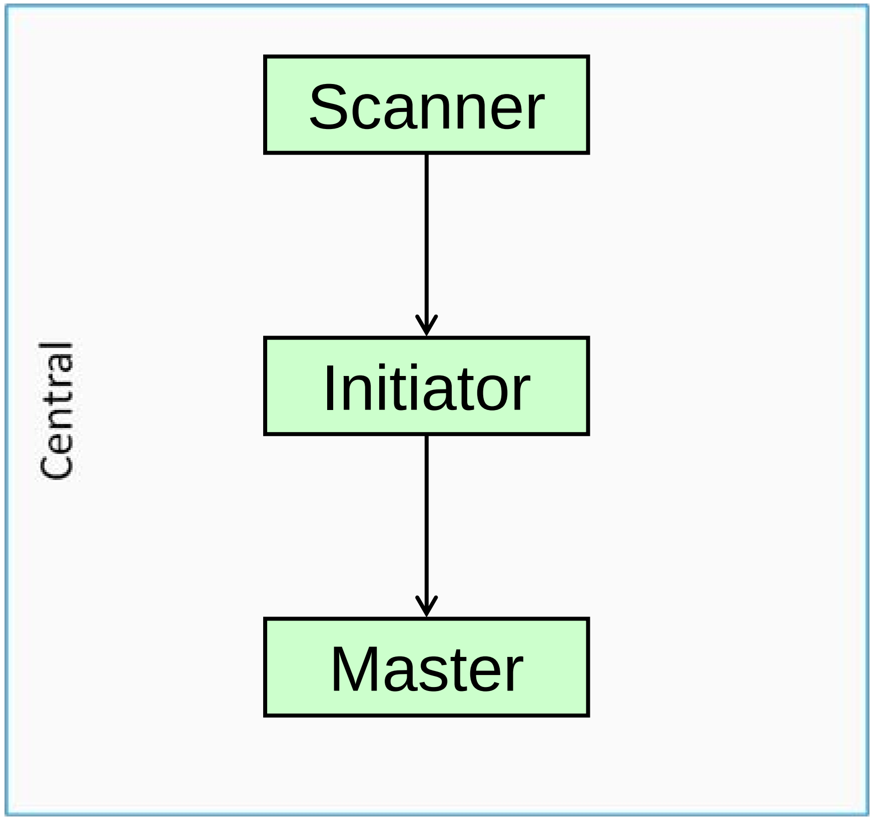
Device on the other end of the link is the peer device.

Find a device

- To transfer data, devices must be connected.
- Connection is initiated by the master, the phone.
- How can the master know the fob exists?
  - The phone scans.
  - The fob advertises.
- All the information about the fob that the master needs must be advertised.




Peripheral
Advertiser
Slave


Central
Scanner
Initiator
Master

Modes of advertisement and scanning

Peripheral

Advertising

- Limited Discoverable
  - Special mode for devices willing to be connectable for a limited period of time.
  - Can only be in this mode for 30 seconds.
- General Discoverable
  - Otherwise.
  - No timeout.

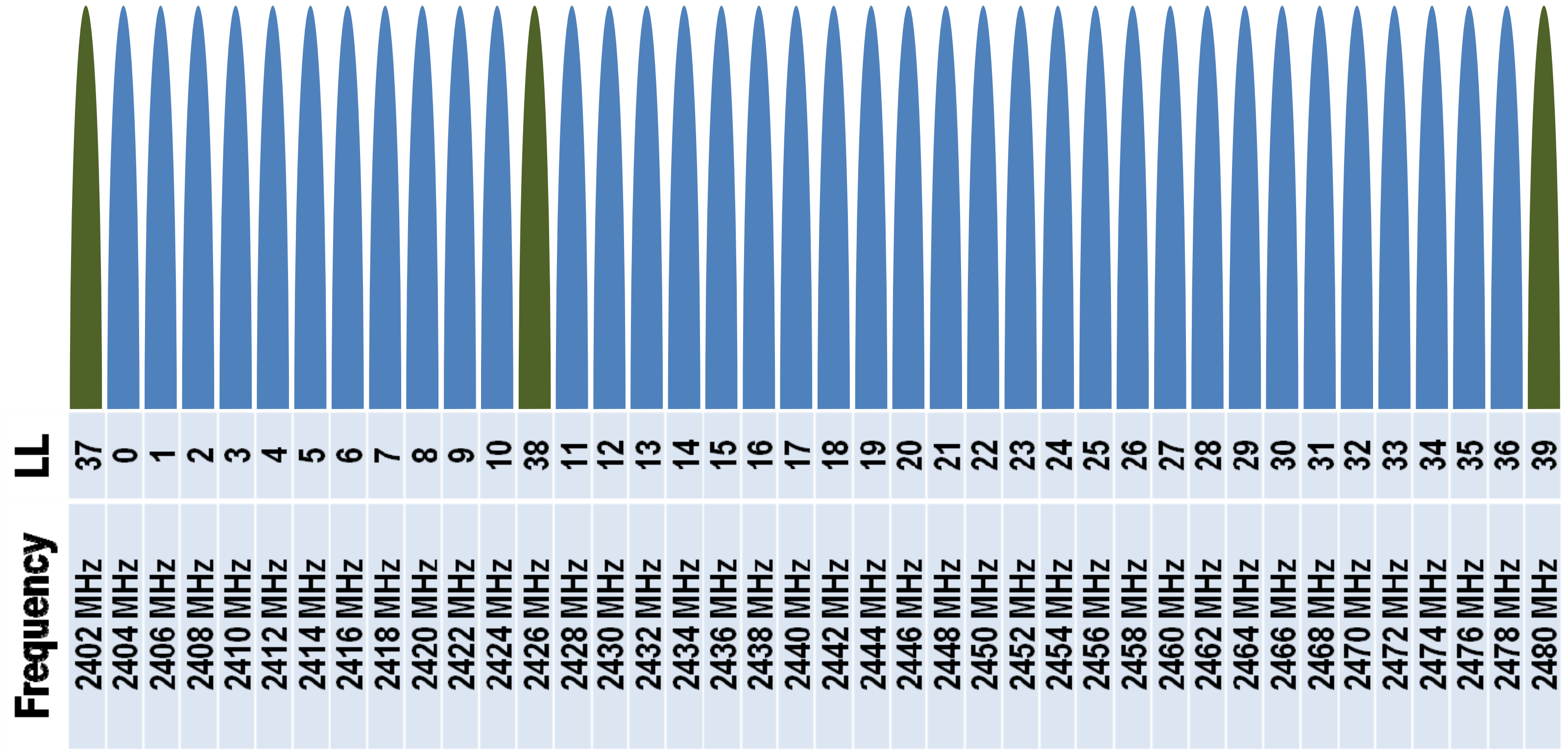
Central

Scanning

- Limited Discovery
  - to only discover devices which are in Limited Discoverable mode
- General Discovery
  - to discover all devices in Limited or General Discoverable modes

Advertising channels

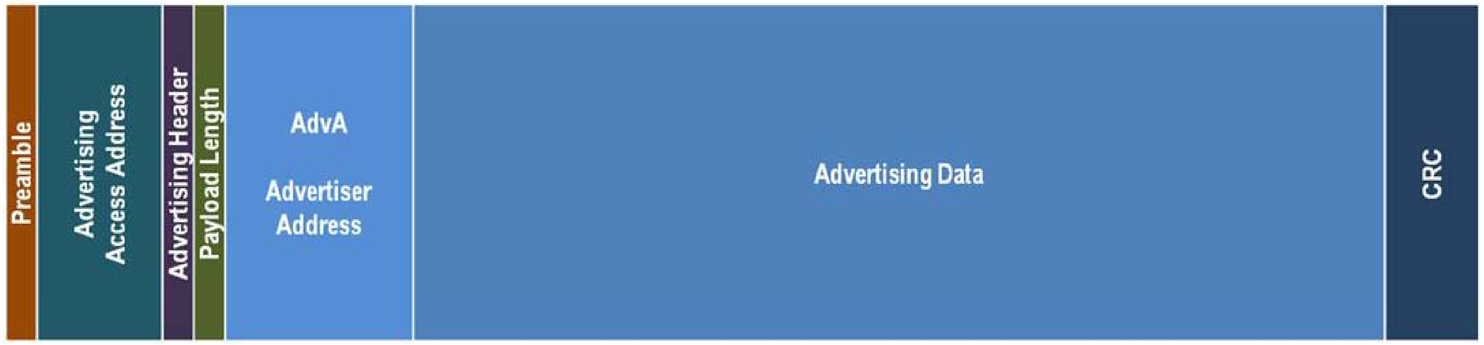
3 Advertising Channels and 37 Data Channels



LL	Frequency
37	2402 MHz
0	2404 MHz
1	2406 MHz
2	2408 MHz
3	2410 MHz
4	2412 MHz
5	2414 MHz
6	2416 MHz
7	2418 MHz
8	2420 MHz
9	2422 MHz
10	2424 MHz
38	2426 MHz
11	2428 MHz
12	2430 MHz
13	2432 MHz
14	2434 MHz
15	2436 MHz
16	2438 MHz
17	2440 MHz
18	2442 MHz
19	2444 MHz
20	2446 MHz
21	2448 MHz
22	2450 MHz
23	2452 MHz
24	2454 MHz
25	2456 MHz
26	2458 MHz
27	2460 MHz
28	2462 MHz
29	2464 MHz
30	2466 MHz
31	2468 MHz
32	2470 MHz
33	2472 MHz
34	2474 MHz
35	2476 MHz
36	2478 MHz
39	2480 MHz

Advertising data

- The advertisement can contain 31 bytes data.
- Core spec defines what is legal to put here.
  - Manufacturer specific data is also legal.
- Contains the advertisers address.



Addresses

- There are different types of addresses that could be used:
  - Can use a completely random one.
  - Can also use a registered one.
- If you use a random one, make sure that the first two bits are 1.
- Handled internally on the nRF51822, with the latest SDK.

	Public address		
LSB	24 company assigned bits.	24 company identification bits.	MSB

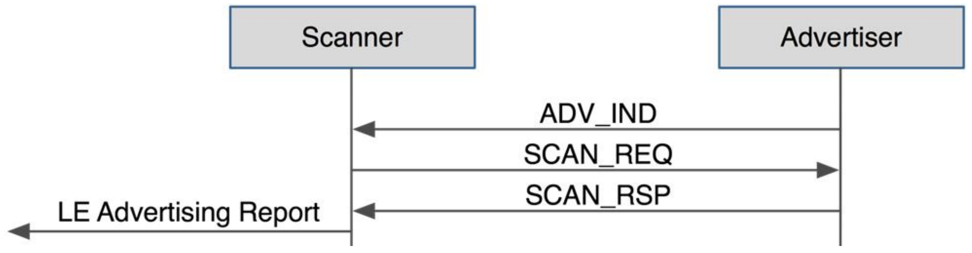
	Static random address			
LSB	Random 46 bits	1	1	MSB

More advertising data

- If you want to advertise more than 31 bytes, you need to use what is called a Scan Response.
- Up to an additional 31 bytes.

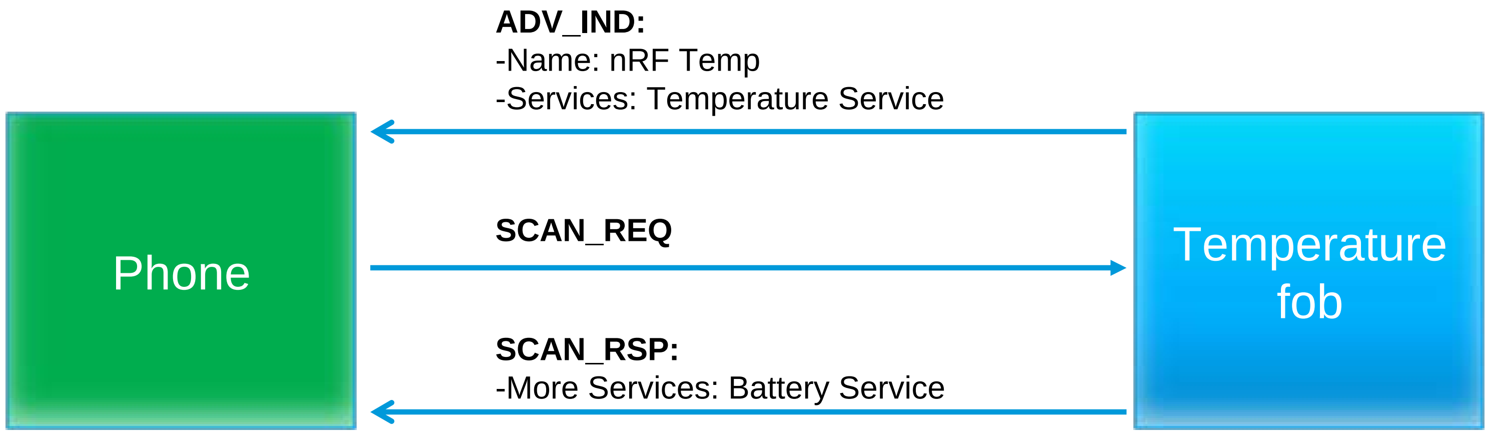
Note:

- A master does not have to send a SCAN_REQ.
  - Active scanning: Send SCAN_REQ.
  - Passive scanning: Don't.
- A peripheral can choose to not answer a SCAN_REQ.



Example advertising data: Temperature fob

- What does the master need to know?
  - What can I call this device, when presenting it to the user?
  - What kind of device is it?



Non-connectable advertising

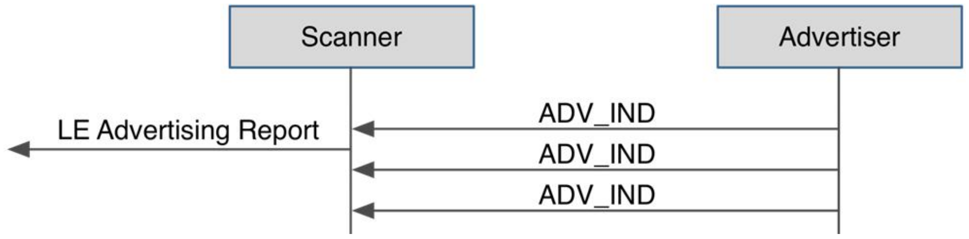
- It is possible to advertise without being connectable.
- Broadcast data.
- Again, 31 bytes.
  - If wanted, 31 bytes more using SCAN_RSP.
- A non-connectable ...
  - ... peripheral: Broadcaster.
  - ... central: Observer.

Advertising Event Type	Packet type used in this advertising event type	Allowable response packets for advertising event	
		SCAN_REQ	CONNECT_REQ
Connectable Undirected Event	ADV_IND	YES	YES
Connectable Directed Event	ADV_DIRECT_IND	NO	YES*
Non-connectable Undirected Event	ADV_NONCONN_IND	NO	NO
Discoverable Undirected Event	ADV_DISCOVER_IND	YES	NO

*only the correctly addressed initiator may respond

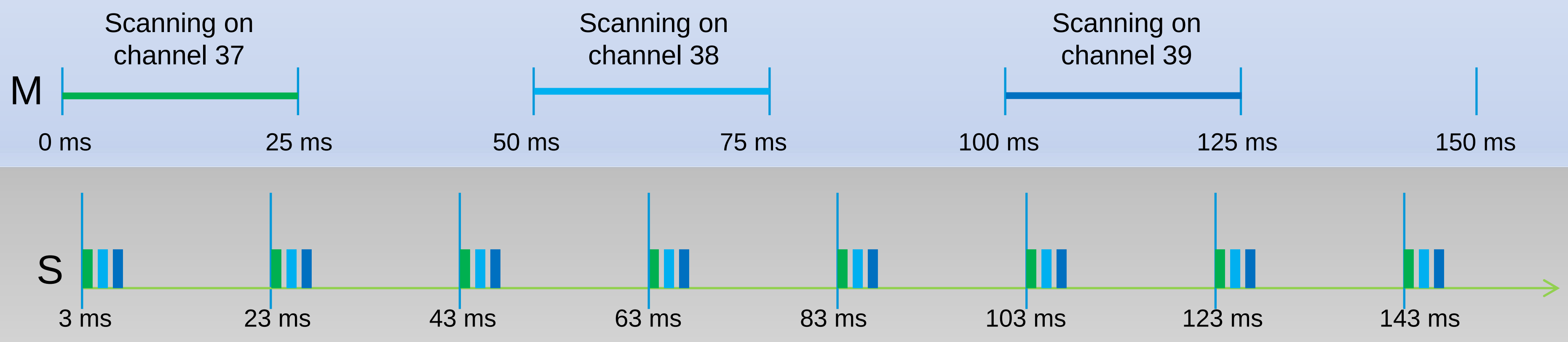
Advertisement timing

- How fast can the central discover the peripheral?
- Decided by:
  - Peripheral parameter:
    - Advertising interval, from 20 ms to 4 seconds.
  - Central parameters:
    - Scan window.
    - Scan interval.
- Affects current consumption heavily, especially on peripheral.



Advertising and scanning

Master scan interval = 50 ms
Master scan window = 25 ms



Advertising on **37**, **38** and **39**

Slave Advertising interval = 20 ms

From this we can see that the following advertising packets will be picked up by the scanning device: t=3 ch=37, t=23 ch=37, t=63 ch=38, t=103 ch=39, t=123 ch=39.

Practical session 1: Advertising

- Goal: Change advertising parameters and observe the changes using Master Control Panel.

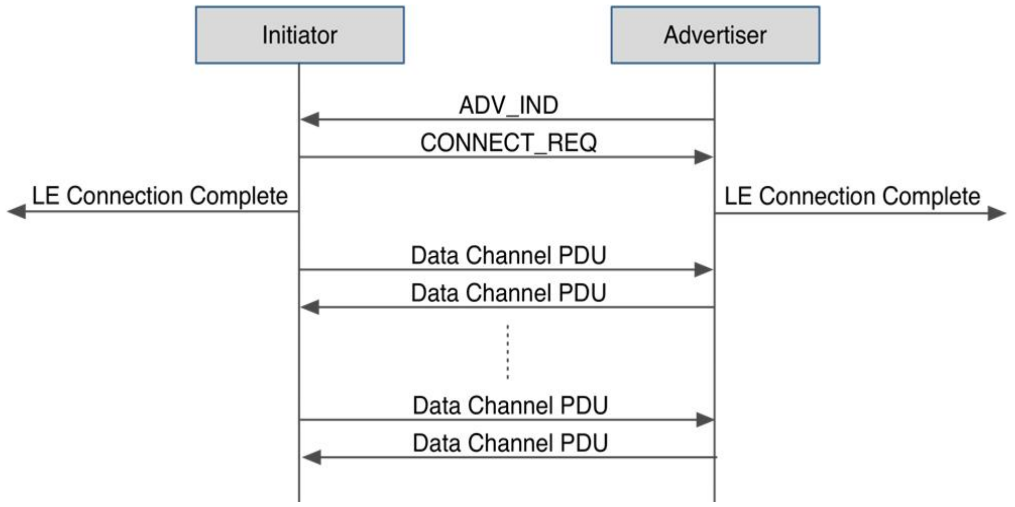
Steps:

1. Change name of device.
2. Observe new name in Master Control Panel.
3. Change the advertising interval.
4. Observe the change in discovery time.
   - Clear the discovered list by right-clicking in list and choose «Clear list».
5. Add scan response following the steps in the code.
6. Change the scan parameters in Master Control Panel to do «Active scanning».
   - Need to restart scanning to see change.

Theory 2: Connection and data transfer

Initiating a connection

- When a central has discovered a device, it can choose to connect to it.
- Done by sending a Connect Request.

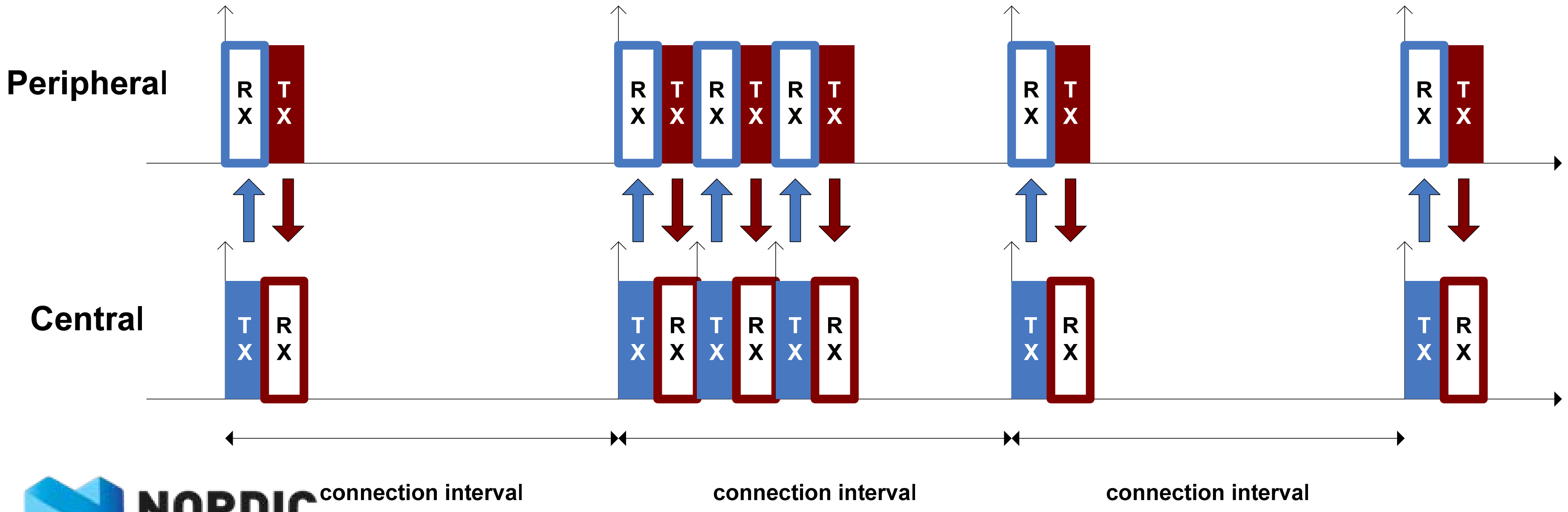


Connection parameters

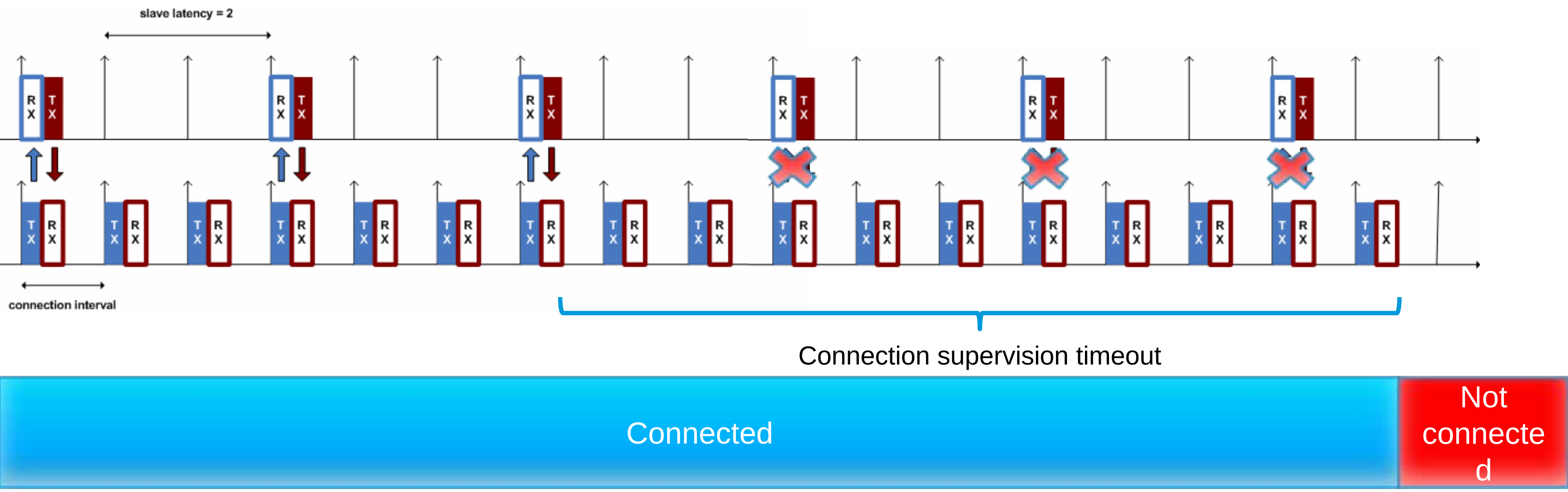
- After a connection is established, it has certain parameters:
  - Connection interval.
  - Slave latency.
  - Supervision timeout.
- The slave can request parameters, but the master might deny them.
  - If so, it’s up to the peripheral to decide what to do. Disconnect? Request others?
- Connection timing is affected by clock drift of both master and slave.
  - Is accounted for automatically.

Data transfer in a connection

- When in a connection, the master sends first, and the slave responds.
- Can do multiple transactions per connection event, which happens each connection interval.
- Connection interval can be from 7.5 ms to 4 seconds.



More on connection parameters



Take care choosing parameters

- Decides the data transfer capacity of the link.
  - Can transfer only a certain number of bytes per connection event.
- Decides the latency of data transmission.
  - For example: A short connection interval, but high slave latency gives short latency from slave to master, but long from master to slave.
- Have huge impact on the battery lifetime of the slave especially.
- Decides how rapidly a link is considered disconnected.

Apple connection parameters

- Can only do 4 data transactions per connection event.
- Lowest documented connection interval: 37.5 ms.

- Gives data rate of

$$\frac{1\ \text{sec}}{37.5\ \text{msec}} \times 20\ \text{bytes} \times 4\ \text{data transactions} = 2.133\ \text{kilobytes/sec}$$
$$= 17\ \text{kilobits/sec}$$

Windows 8 connection parameters

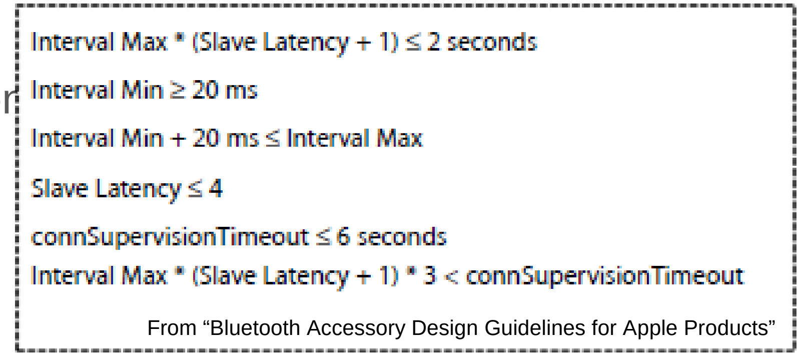
- Data transactions per connection event will be dependent on master dongle.
  - In example we'll use 6, just to show max data rate.
- Lowest documented connection interval: 7.5 ms.

- Gives data rate of

$$\frac{1\ \text{sec}}{7.5\ \text{msec}} \times 20\ \text{bytes} \times 6\ \text{data transactions} = 16\ \text{kilobytes/sec} = 128\ \text{kilobits/sec}$$

Example: Parameters for fob

- Temperature doesn't change rapidly. Battery lifetime is important.
  - Should therefore use long connection interval.
- Can add slave latency, to avoid transmitting if the temperature doesn't change.
- Want to connect to iPhone.
  - Should therefore follow Apple's recommendation
- Suggestion:
  - Connection interval min: 1 second
  - Connection Interval max: 2 seconds
  - Slave latency: 0.
  - Connection supervision timeout: 6 seconds.



Interval Max * (Slave Latency + 1) ≤ 2 seconds

Interval Min ≥ 20 ms

Interval Min + 20 ms ≤ Interval Max

Slave Latency ≤ 4

connSupervisionTimeout ≤ 6 seconds

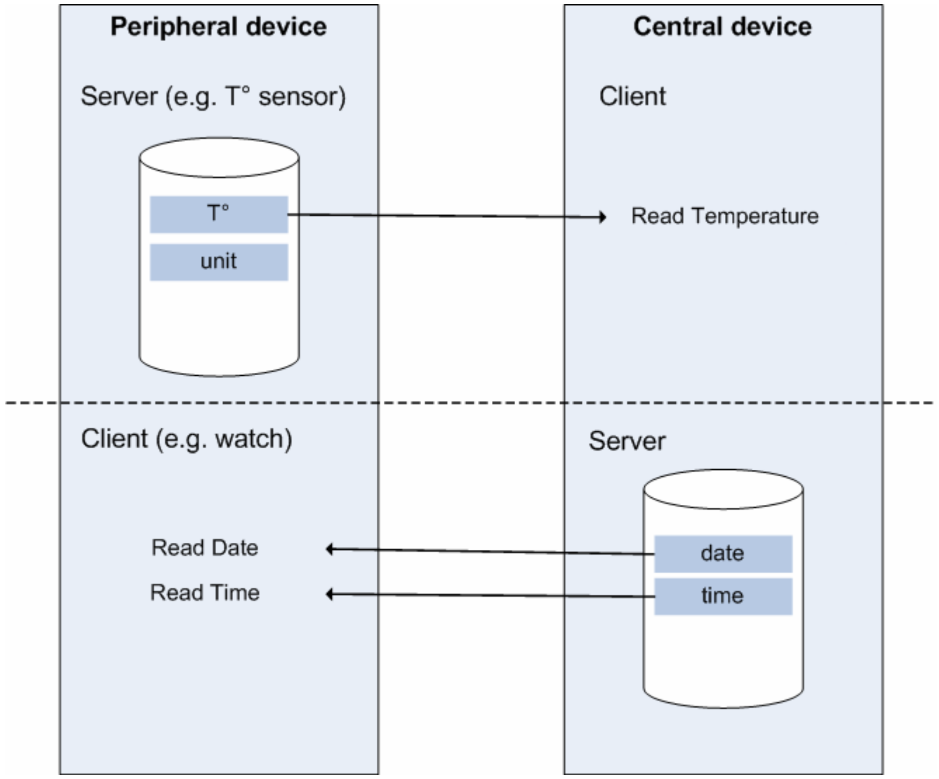
Interval Max * (Slave Latency + 1) * 3 < connSupervisionTimeout

From "Bluetooth Accessory Design Guidelines for Apple Products"

Temperature fob

Interacting with a device

- To transfer data, one must know what data can be transferred.
- When talking about data transfer, terminology is extended:
  - GATT Server.
  - GATT Client.
- Most often a master would be a Client, and the peripheral a Server.
  - A device can be both.
- How can a client know what data a server has?
  - Service discovery.
- A Service Discovery is used by a to get hold of the GATT table of the server.



GATT table

- Shows all the data available on the fob.
  - Each element is an attribute.
- Every attribute have a handle, a UUID and a value.
- A handle is a reference to a specific attribute on a device.
  - Handle is unique per device.
- A UUID tells what type of data this attribute is.

Handle	UUID	Value
0x0001	0x2800	0x1800
0x0002	0x2803	{0x0A, 0x0003, 0x2A00}
0x0003	0x2A00	"Example Device"
0x0010	0x2800	0x1801
0x0100	0x2800	0x180A
0x0101	0x2803	{0x02, 0x0102, 0x2A19}
0x0102	0x2A19	0x04
0x0200	0x2800	0x5AB20001-B355-4D8A-96EF-2963812DD0B8
0x0201	0x2803	{0x12, 0x0202, 0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8}
0x0202	0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8	0x028A
0x0203	0x2904	{0x0E, 0xFE, «Celsius», «Outside»}
0x0204	0x2901	"Outside Temperature"
0x0205	0x2902	0x0000
0x0210	0x2803	{0x12, 0x0211, 0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8}
0x0211	0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8	0x27
0x0212	0x2904	{0x04, 0x00, «Percent», «Outside»}
0x0213	0x2901	"Outside Relative Humidity"
0x0214	0x2902	0x0000

Example: Annotated GATT table for the fob

Handle	UUID (Type)	Value (Type)
0x0001	0x2800 (Service)	0x1800 (GAP Service)
0x0002	0x2803 (Characteristic)	{0x0A, 0x0003, 0x2A00}
0x0003	0x2A00 (Device Name)	“Example Device”
0x0010	0x2800 (Service)	0x1801 (GATT Service)
0x0100	0x2800 (Service)	0x180A (Battery State Service)
0x0101	0x2803 (Characteristic)	{0x02, 0x0102, 0x2A19}
0x0102	0x2A19 (Battery Level)	0x04
0x0200	0x2800 (Service)	0x5AB20001-B355-4D8A-96EF-2963812DD0B8 (Proprietary Thermometer Humidity Service)
0x0201	0x2803 (Characteristic)	{0x12, 0x0202, 0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8}
0x0202	0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8 (Proprietary Temperature Characteristic)	0x028A
0x0203	0x2904 (Characteristic Format)	{0x0E, 0xFE, «Celsius», «Outside»}
0x0204	0x2901 (Characteristic User Description)	“Outside Temperature”
0x0205	0x2902 (Client Characteristic Configuration Descriptor)	0x0000
0x0210	0x2803 (Characteristic)	{0x12, 0x0211, 0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8}
0x0211	0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8 (Proprietary Humidity Characteristic)	0x27
0x0212	0x2904 (Characteristic Format)	{0x04, 0x00, «Percent», «Outside»}
0x0213	0x2901 (Characteristic User Description)	“Outside Relative Humidity”
0x0214	0x2902 (Client Characteristic Configuration Descriptor)	0x0000

Service Definition

Characteristic Definition

Service Declaration

Characteristic Declaration

Characteristic Descriptor

UUIDs

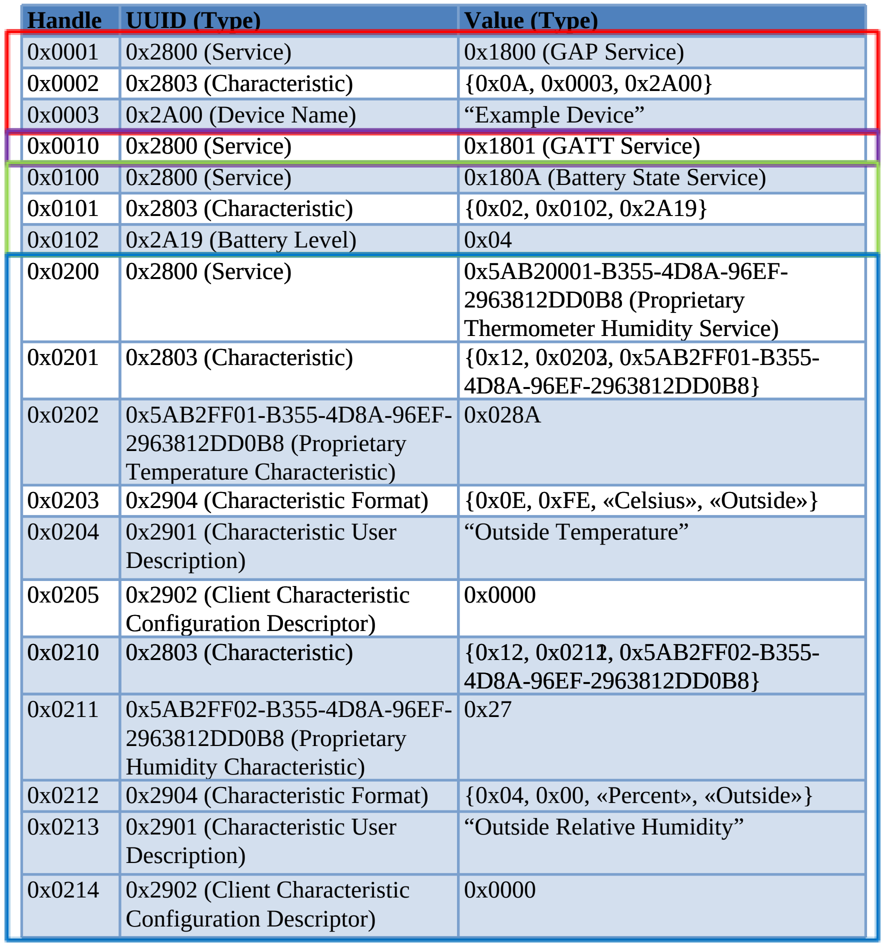
- 128-bit number, identifying attribute type.
- Consists of two parts:
  - Base.
  - Alias.
- Alias is a 16-bit part of the base.
- All Bluetooth SIG Services and Characteristics use the same base:
  - 0000xxxx-0000-1000-8000-00805F9B34FB
- Custom UUID: create custom, random base and use an incremeting alias.

Handle	UUID (Type)	Value (Type)
0x0001	0x2800 (Service)	0x1800 (GAP Service)
0x0002	0x2803 (Characteristic)	{0x0A, 0x0003, 0x2A00}
0x0003	0x2A00 (Device Name)	"Example Device"
0x0010	0x2800 (Service)	0x1801 (GATT Service)
0x0100	0x2800 (Service)	0x180A (Battery State Service)
0x0101	0x2803 (Characteristic)	{0x02, 0x0102, 0x2A19}
0x0102	0x2A19 (Battery Level)	0x04
0x0200	0x2800 (Service)	0x5AB20001-B355-4D8A-96EF-2963812DD0B8 (Proprietary Thermometer Humidity Service)
0x0201	0x2803 (Characteristic)	{0x12, 0x0202, 0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8}
0x0202	0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8 (Proprietary Temperature Characteristic)	0x028A
0x0203	0x2904 (Characteristic Format)	{0x0E, 0xFE, «Celsius», «Outside»}
0x0204	0x2901 (Characteristic User Description)	"Outside Temperature"
0x0205	0x2902 (Client Characteristic Configuration Descriptor)	0x0000
0x0210	0x2803 (Characteristic)	{0x12, 0x0211, 0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8}
0x0211	0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8 (Proprietary Humidity Characteristic)	0x27
0x0212	0x2904 (Characteristic Format)	{0x04, 0x00, «Percent», «Outside»}
0x0213	0x2901 (Characteristic User Description)	"Outside Relative Humidity"
0x0214	0x2902 (Client Characteristic Configuration Descriptor)	0x0000

NORDIC SEMICONDUCTOR

Services

- A Service is a grouping of associated characteristics.
- Can see four services:
  - GAP Service
  - GATT Service
  - Battery State Service
  - Thermometer Humidity Service
- Each service has zero or more characteristics.



Handle	UUID (Type)	Value (Type)
0x0001	0x2800 (Service)	0x1800 (GAP Service)
0x0002	0x2803 (Characteristic)	{0x0A, 0x0003, 0x2A00}
0x0003	0x2A00 (Device Name)	"Example Device"
0x0010	0x2800 (Service)	0x1801 (GATT Service)
0x0100	0x2800 (Service)	0x180A (Battery State Service)
0x0101	0x2803 (Characteristic)	{0x02, 0x0102, 0x2A19}
0x0102	0x2A19 (Battery Level)	0x04
0x0200	0x2800 (Service)	0x5AB20001-B355-4D8A-96EF-2963812DD0B8 (Proprietary Thermometer Humidity Service)
0x0201	0x2803 (Characteristic)	{0x12, 0x0202, 0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8}
0x0202	0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8 (Proprietary Temperature Characteristic)	0x028A
0x0203	0x2904 (Characteristic Format)	{0x0E, 0xFE, «Celsius», «Outside»}
0x0204	0x2901 (Characteristic User Description)	"Outside Temperature"
0x0205	0x2902 (Client Characteristic Configuration Descriptor)	0x0000
0x0210	0x2803 (Characteristic)	{0x12, 0x0211, 0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8}
0x0211	0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8 (Proprietary Humidity Characteristic)	0x27
0x0212	0x2904 (Characteristic Format)	{0x04, 0x00, «Percent», «Outside»}
0x0213	0x2901 (Characteristic User Description)	"Outside Relative Humidity"
0x0214	0x2902 (Client Characteristic Configuration Descriptor)	0x0000

Characteristics

- A Characteristic is the basic data entity.
- Two records in the GATT Table:
  - Definition.
  - Value.
- Definition consists of:
  - Properties.
  - Handle of value.
  - Type (UUID).
- Value is just the value.

Handle	UUID (Type)	Value (Type)
0x0001	0x2800 (Service)	0x1800 (GAP Service)
0x0002	0x2803 (Characteristic)	{0x0A, 0x0003, 0x2A00}
0x0003	0x2A00 (Device Name)	“Example Device”
0x0010	0x2800 (Service)	0x1801 (GATT Service)
0x0100	0x2800 (Service)	0x180A (Battery State Service)
0x0101	0x2803 (Characteristic)	{0x02, 0x0102, 0x2A19}
0x0102	0x2A19 (Battery Level)	0x04
0x0200	0x2800 (Service)	0x5AB20001-B355-4D8A-96EF-2963812DD0B8 (Proprietary Thermometer Humidity Service)
0x0201	0x2803 (Characteristic)	{0x12, 0x0202, 0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8}
0x0202	0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8 (Proprietary Temperature Characteristic)	0x028A
0x0203	0x2904 (Characteristic Format)	{0x0E, 0xFE, «Celsius», «Outside»}
0x0204	0x2901 (Characteristic User Description)	“Outside Temperature”
0x0205	0x2902 (Client Characteristic Configuration Descriptor)	0x0000
0x0210	0x2803 (Characteristic)	{0x12, 0x0211, 0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8}
0x0211	0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8 (Proprietary Humidity Characteristic)	0x27
0x0212	0x2904 (Characteristic Format)	{0x04, 0x00, «Percent», «Outside»}
0x0213	0x2901 (Characteristic User Description)	“Outside Relative Humidity”
0x0214	0x2902 (Client Characteristic Configuration Descriptor)	0x0000

Characteristic properties

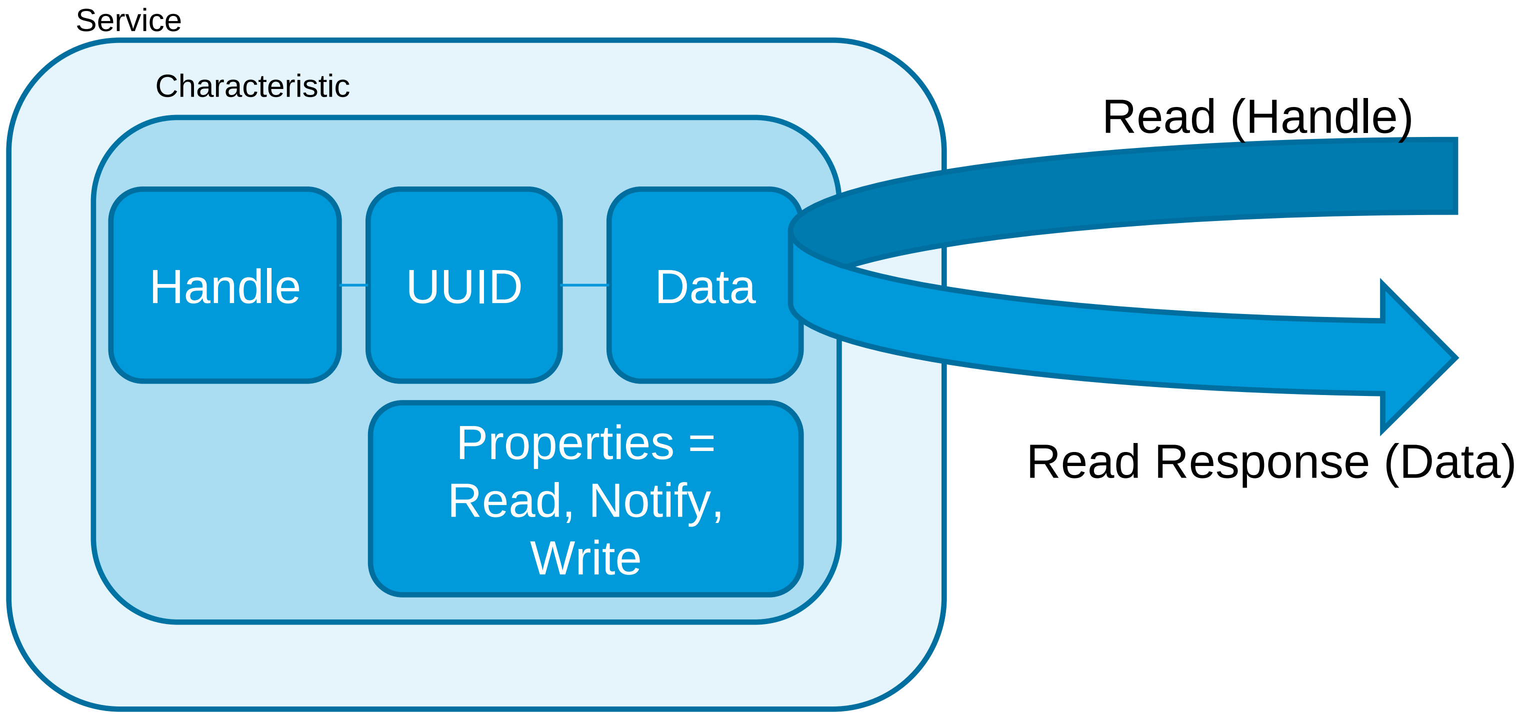
- The properties of a characteristic shows what can be done with it.
- Mostly self-explanatory.
  - Note that «Without Reponse» is still ACK-ed on the link layer.
- Notify:
  - Allows the server to notify the client on changes.
  - No application level ACK.
- Indication:
  - Same as notify, but with application level ACK.

Properties	Value
Broadcast	0x01
Read	0x02
Write Without Response	0x04
Write	0x08
Notify	0x10
Indicate	0x20
Authenticated Signed Writes	0x40
Extended Properties	0x80

NORDIC SEMICONDUCTOR

On-air operations

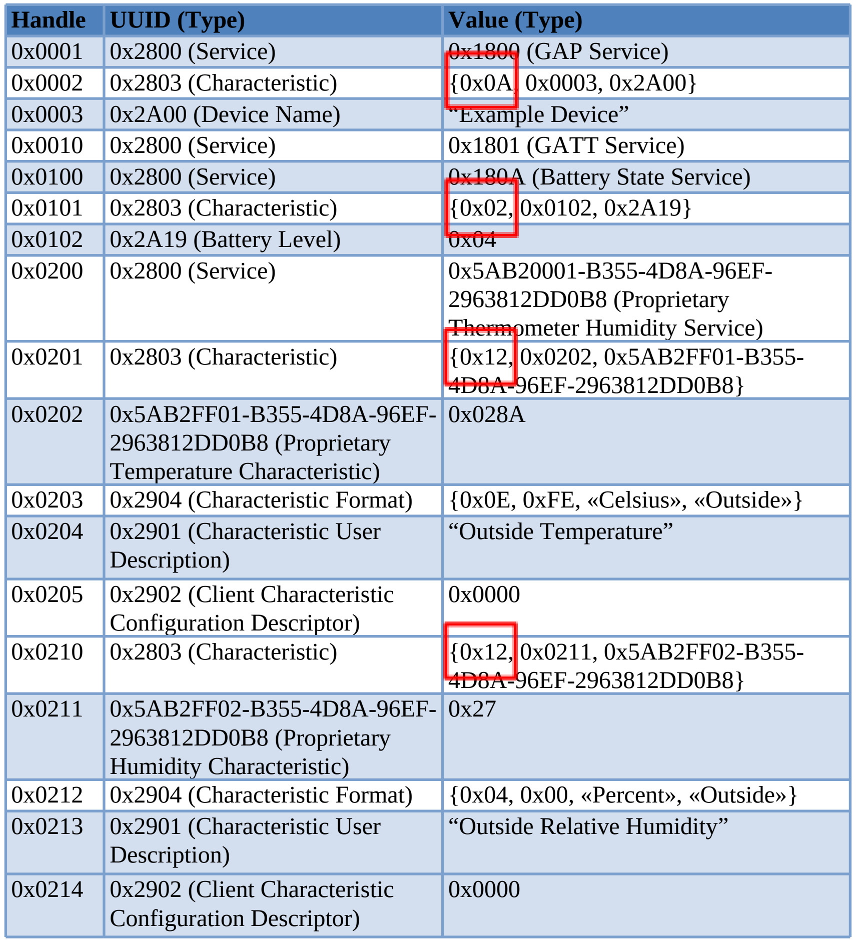
- Done by handle. Saves on-air size.



Example: Properties on the fob

- Each characteristic has properties:
  - 0x0A: Write and Read.
  - 0x02: Read.
  - 0x12: Notify and Read.

Properties	Value
Broadcast	0x01
Read	0x02
Write Without Response	0x04
Write	0x08
Notify	0x10
Indicate	0x20
Extended Properties	0x80



Handle	UUID (Type)	Value (Type)
0x0001	0x2800 (Service)	0x1800 (GAP Service)
0x0002	0x2803 (Characteristic)	{0x0A, 0x0003, 0x2A00}
0x0003	0x2A00 (Device Name)	"Example Device"
0x0010	0x2800 (Service)	0x1801 (GATT Service)
0x0100	0x2800 (Service)	0x180A (Battery State Service)
0x0101	0x2803 (Characteristic)	{0x02, 0x0102, 0x2A19}
0x0102	0x2A19 (Battery Level)	0x04
0x0200	0x2800 (Service)	0x5AB20001-B355-4D8A-96EF-2963812DD0B8 (Proprietary Thermometer Humidity Service)
0x0201	0x2803 (Characteristic)	{0x12, 0x0202, 0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8}
0x0202	0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8 (Proprietary Temperature Characteristic)	0x028A
0x0203	0x2904 (Characteristic Format)	{0x0E, 0xFE, «Celsius», «Outside»}
0x0204	0x2901 (Characteristic User Description)	"Outside Temperature"
0x0205	0x2902 (Client Characteristic Configuration Descriptor)	0x0000
0x0210	0x2803 (Characteristic)	{0x12, 0x0211, 0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8}
0x0211	0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8 (Proprietary Humidity Characteristic)	0x27
0x0212	0x2904 (Characteristic Format)	{0x04, 0x00, «Percent», «Outside»}
0x0213	0x2901 (Characteristic User Description)	"Outside Relative Humidity"
0x0214	0x2902 (Client Characteristic Configuration Descriptor)	0x0000

Characteristic descriptors

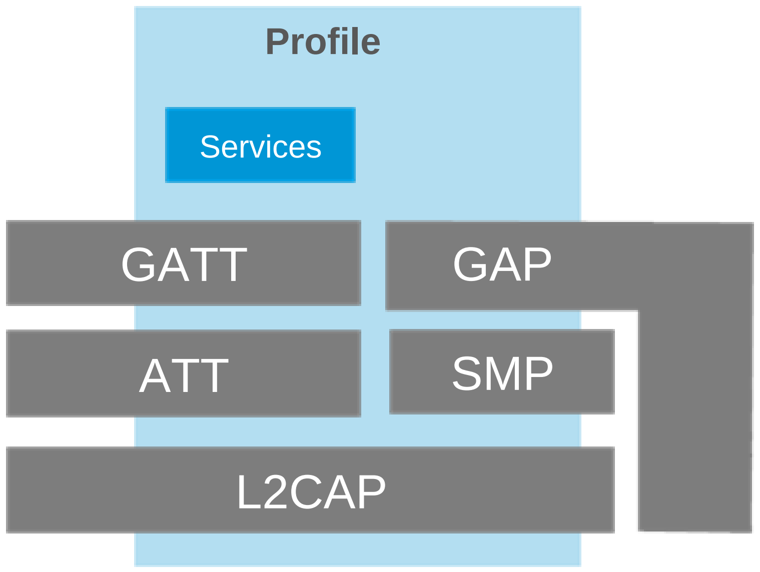
- A characteristic can have any number of descriptors.
- Adds metadata about the characteristic.
  - Description.
  - Format.
- A special one: CCCD.
- Needed when having Notify or Indicate.
  - Write to it to enable them.

Handle	UUID (Type)	Value (Type)
0x0001	0x2800 (Service)	0x1800 (GAP Service)
0x0002	0x2803 (Characteristic)	{0x0A, 0x0003, 0x2A00}
0x0003	0x2A00 (Device Name)	“Example Device”
0x0010	0x2800 (Service)	0x1801 (GATT Service)
0x0100	0x2800 (Service)	0x180A (Battery State Service)
0x0101	0x2803 (Characteristic)	{0x02, 0x0102, 0x2A19}
0x0102	0x2A19 (Battery Level)	0x04
0x0200	0x2800 (Service)	0x5AB20001-B355-4D8A-96EF-2963812DD0B8 (Proprietary Thermometer Humidity Service)
0x0201	0x2803 (Characteristic)	{0x12, 0x0202, 0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8}
0x0202	0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8 (Proprietary Temperature Characteristic)	0x028A
0x0203	0x2904 (Characteristic Format)	{0x0E, 0xFE, «Celsius», «Outside»}
0x0204	0x2901 (Characteristic User Description)	“Outside Temperature”
0x0205	0x2902 (Client Characteristic Configuration Descriptor)	0x0000
0x0210	0x2803 (Characteristic)	{0x12, 0x0211, 0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8}
0x0211	0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8 (Proprietary Humidity Characteristic)	0x27
0x0212	0x2904 (Characteristic Format)	{0x04, 0x00, «Percent», «Outside»}
0x0213	0x2901 (Characteristic User Description)	“Outside Relative Humidity”
0x0214	0x2902 (Client Characteristic Configuration Descriptor)	0x0000

NORDIC SEMICONDUCTOR

Profiles

- A Profile is a higher-level collection of services.
  - Vertical through the stack.
- Most often defines
  - use case.
  - roles
  - security levels needed.
  - connection parameters.
- A device can support any number of profiles.
  - Possible memory limitations for the GATT table size.



Practical session 2: GATT

- Primary goal: Add a characteristic to an existing service.
- Secondary goal: Enable notifications for this added characteristic.

Steps:

1. Change device name.
2. Add characteristic for indoor temperature.
   - Hint: Look at how the outdoor is added.
3. Call the indoor initialization method.
4. Actually update the indoor temperature value.
   - After this, you can test it using Master Control Panel. You should be able to read both indoor and outdoor.
5. Add variable for connection handle.
6. Set connection handle when connected.
7. Handle the BLE_GATTS_EVT_SYS_ATTR_MISSING event.
8. Actually send the notification using ble_gatts_hvx().

Theory 3: Security

Security

- Security in BLE is about getting an encrypted link.
  - Avoids eavesdroppers.
- Uses AES-128 encryption.
- Protects the actual value of characteristics, not their definitions.
  - Can always connect.
  - Can always do service discovery.

Handle	UUID (Type)	Value (Type)
0x0001	0x2800 (Service)	0x1800 (GAP Service)
0x0002	0x2803 (Characteristic)	{0x0A, 0x0003, 0x2A00}
0x0003	0x2A00 (Device Name)	“Example Device”
0x0010	0x2800 (Service)	0x1801 (GATT Service)
0x0100	0x2800 (Service)	0x180A (Battery State Service)
0x0101	0x2803 (Characteristic)	{0x02, 0x0102, 0x2A19}
0x0102	0x2A19 (Battery Level)	0x04
0x0200	0x2800 (Service)	0x5AB20001-B355-4D8A-96EF-2963812DD0B8 (Proprietary Thermometer Humidity Service)
0x0201	0x2803 (Characteristic)	{0x12, 0x0202, 0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8}
0x0202	0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8 (Proprietary Temperature Characteristic)	0x028A
0x0203	0x2904 (Characteristic Format)	{0x0E, 0xFE, «Celsius», «Outside»}
0x0204	0x2901 (Characteristic User Description)	“Outside Temperature”
0x0205	0x2902 (Client Characteristic Configuration Descriptor)	0x0000
0x0210	0x2803 (Characteristic)	{0x12, 0x0211, 0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8}
0x0211	0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8 (Proprietary Humidity Characteristic)	0x27
0x0212	0x2904 (Characteristic Format)	{0x04, 0x00, «Percent», «Outside»}
0x0213	0x2901 (Characteristic User Description)	“Outside Relative Humidity”
0x0214	0x2902 (Client Characteristic Configuration Descriptor)	0x0000

🟦 Secured

NORDIC SEMICONDUCTOR

Security level

- Can define different security levels ...
  - ... for different characteristics.
  - ... for different operations.
- Means that you can allow unencrypted reads, but require encryption for writing.
- Can require encryption for temperature reading, but not for battery level.

Handle	UUID (Type)	Value (Type)
0x0001	0x2800 (Service)	0x1800 (GAP Service)
0x0002	0x2803 (Characteristic)	{0x0A, 0x0003, 0x2A00}
0x0003	0x2A00 (Device Name)	“Example Device”
0x0010	0x2800 (Service)	0x1801 (GATT Service)
0x0100	0x2800 (Service)	0x180A (Battery State Service)
0x0101	0x2803 (Characteristic)	{0x02, 0x0102, 0x2A19}
0x0102	0x2A19 (Battery Level)	0x04
0x0200	0x2800 (Service)	0x5AB20001-B355-4D8A-96EF-2963812DD0B8 (Proprietary Thermometer Humidity Service)
0x0201	0x2803 (Characteristic)	{0x12, 0x0202, 0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8}
0x0202	0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8 (Proprietary Temperature Characteristic)	0x028A
0x0203	0x2904 (Characteristic Format)	{0x0E, 0xFE, «Celsius», «Outside»}
0x0204	0x2901 (Characteristic User Description)	“Outside Temperature”
0x0205	0x2902 (Client Characteristic Configuration Descriptor)	0x0000
0x0210	0x2803 (Characteristic)	{0x12, 0x0211, 0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8}
0x0211	0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8 (Proprietary Humidity Characteristic)	0x27
0x0212	0x2904 (Characteristic Format)	{0x04, 0x00, «Percent», «Outside»}
0x0213	0x2901 (Characteristic User Description)	“Outside Relative Humidity”
0x0214	0x2902 (Client Characteristic Configuration Descriptor)	0x0000

🟦 Secured

What is bonding?

- Exchanging secrets.
  - Link only secure if bonding in secure location.
- Two devices should do it only once!
- Three methods:
  - Just Works.
  - Man-in-the-Middle-protection (MITM).
  - Out-of-Band (OOB).
- Which one can be used depends on capabilities of the devices.

- Just Works:
  - Does not need any user interaction.
  - Gives an unauthenticated link, i.e. you can’t know which devices are bonded.
- MITM:
  - Needs user to verify or enter passkey.
  - Gives authenticated link.
- OOB:
  - Uses some other way of transferring a seed to a key (for example NFC).
  - Gives authenticated link.

Example: Security mode for the fob

- I/O capabilites of the fob:
  - No display.
  - No keyboard.
  - No speaker.
  - No OOB capabilities.
- This means that we cannot do anything more than Just Works.

Temperature fob

Initiating bonding

- It is always the master that initiates the bonding process.
- Can be done for two reasons:
  - If a Client tries to read a secure characteristic over an unencrypted link, an error code will be returned.
    - Insufficient authentication.
  - A slave may send an encryption request, to make the master bond.
- Apple recommends to use Insufficient authentication.
- Windows 8 will always bond.

Handle	UUID (Type)	Value (Type)
0x0001	0x2800 (Service)	0x1800 (GAP Service)
0x0002	0x2803 (Characteristic)	{0x0A, 0x0003, 0x2A00}
0x0003	0x2A00 (Device Name)	“Example Device”
0x0010	0x2800 (Service)	0x1801 (GATT Service)
0x0100	0x2800 (Service)	0x180A (Battery State Service)
0x0101	0x2803 (Characteristic)	{0x02, 0x0102, 0x2A19}
0x0102	0x2A19 (Battery Level)	0x04
0x0200	0x2800 (Service)	0x5AB20001-B355-4D8A-96EF-2963812DD0B8 (Proprietary Thermometer Humidity Service)
0x0201	0x2803 (Characteristic)	{0x12, 0x0202, 0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8}
0x0202	0x5AB2FF01-B355-4D8A-96EF-2963812DD0B8 (Proprietary Temperature Characteristic)	0x028A
0x0203	0x2904 (Characteristic Format)	{0x0E, 0xFE, «Celsius», «Outside»}
0x0204	0x2901 (Characteristic User Description)	“Outside Temperature”
0x0205	0x2902 (Client Characteristic Configuration Descriptor)	0x0000
0x0210	0x2803 (Characteristic)	{0x12, 0x0211, 0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8}
0x0211	0x5AB2FF02-B355-4D8A-96EF-2963812DD0B8 (Proprietary Humidity Characteristic)	0x27
0x0212	0x2904 (Characteristic Format)	{0x04, 0x00, «Percent», «Outside»}
0x0213	0x2901 (Characteristic User Description)	“Outside Relative Humidity”
0x0214	0x2902 (Client Characteristic Configuration Descriptor)	0x0000

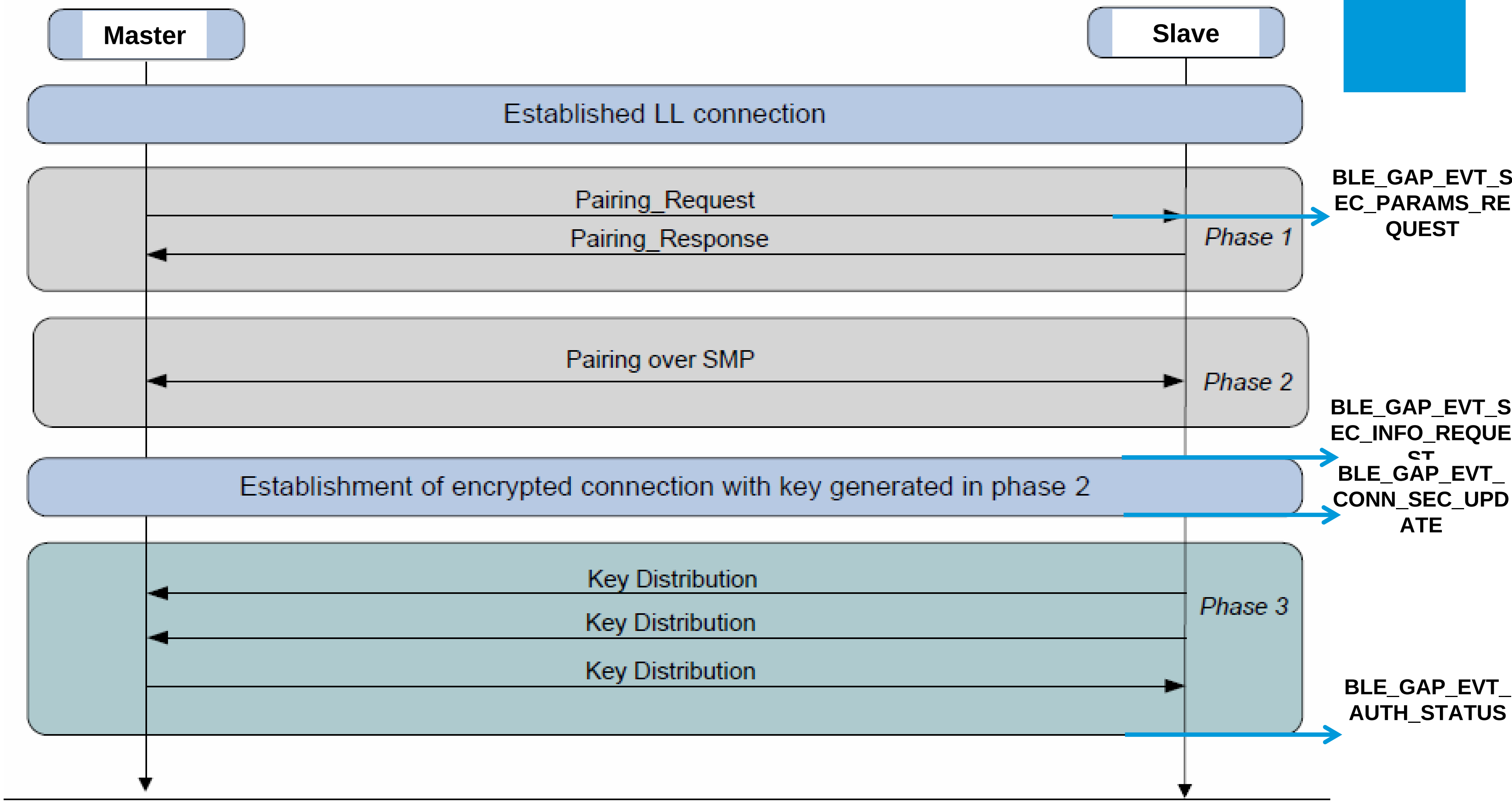
🟦 Secured

NORDIC SEMICONDUCTOR

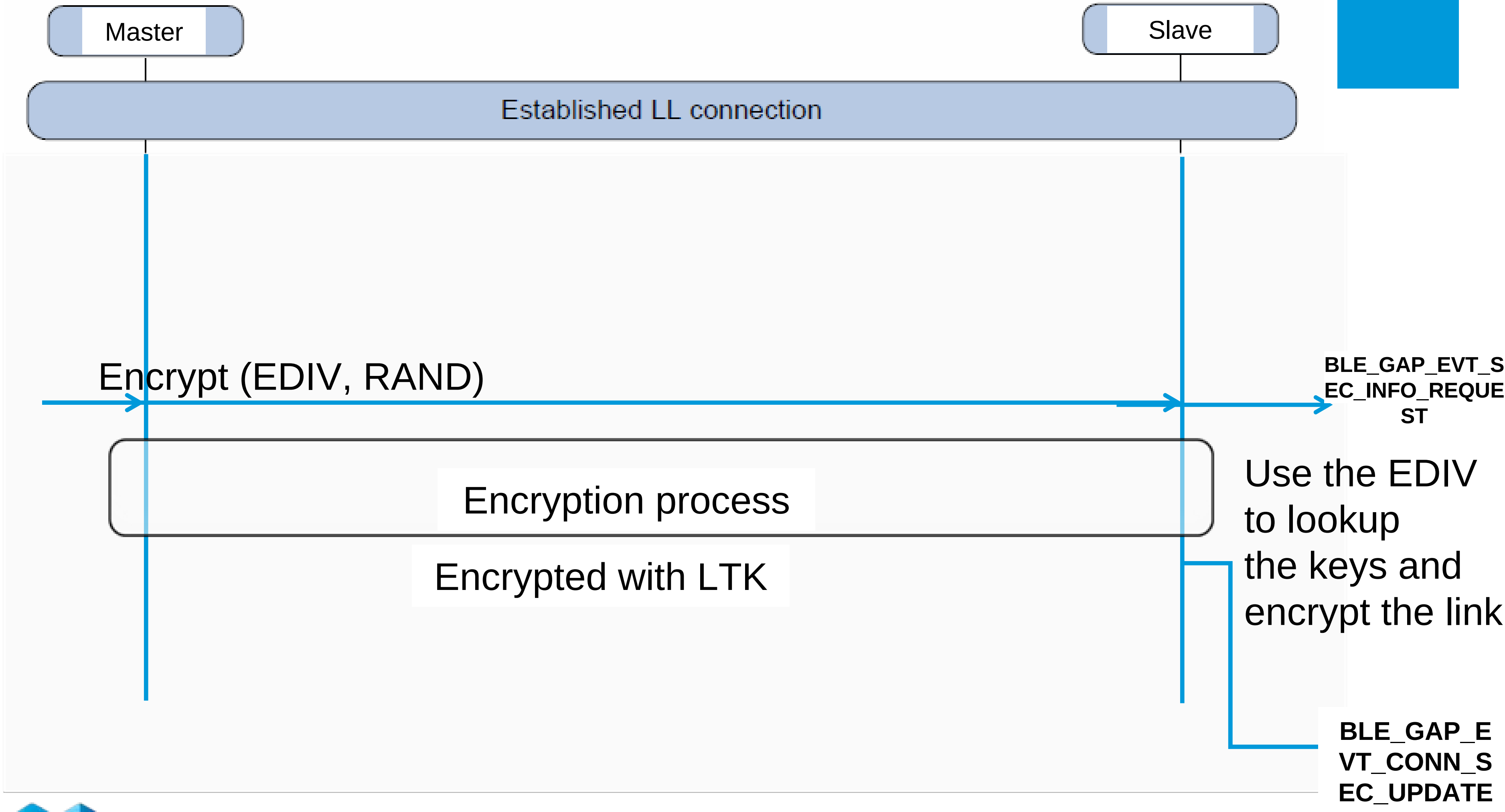
Keys in use

- Two keys:
  - Short-term key (STK).
  - Long-term key (LTK).
- The STK is used to secure the link over which the LTK is shared.
  - A passkey is used as seed for the STK.
- The LTK is used to encrypt the link on subsequent connects.
- Identified by a number called EDIV.
  - The slave needs to store the LTKs used for any master device, and the EDIV is the index attached to a certain key.
  - The master will tell the slave its EDIV value, to let the slave find its key.

Exchanging keys


Master
Slave
Established LL connection
Pairing_Request
Pairing_Response
Phase 1
BLE_GAP_EVT_SEC_PARAMS_REQUEST
Pairing over SMP
Phase 2
BLE_GAP_EVT_SEC_INFO_REQUEST
Establishment of encrypted connection with key generated in phase 2
BLE_GAP_EVT_CONN_SEC_UPDATE
Key Distribution
Key Distribution
Key Distribution
Phase 3
BLE_GAP_EVT_AUTH_STATUS

Subsequent connect


Master
Slave
Established LL connection
Encrypt (EDIV, RAND)
BLE_GAP_EVT_SEC_INFO_REQUEST
Encryption process
Encrypted with LTK
Use the EDIV to lookup the keys and encrypt the link
BLE_GAP_EVT_CONN_SEC_UPDATE

What happens if the LTK is lost?

- If the key is lost, encryption will fail.
- The nRF8001/nRF8002 disconnected if this happened.
- On nRF51, this behaviour is in the application space.
  - Can be chosen depending on what is wanted.

Practical session 3: Security

- Goal: Add security to our previous example, using the bond manager of the SDK.
- Does not include storing of bonds.
  - Look at ble_app_proximity for a more complete example.

Steps:

1. Add files to the project (ble_bondmngr.c, ble_flash.c).
2. Change device name.
3. Add variable for security parameters.
4. Initialize the bond manager and set security parameters.
5. Let the bond manager handle events.
6. Handle the BLE_GAP_EVT_SEC_PARAMS_ REQUEST event.
7. Remove handling of BLE_GATTS_EVT_SYS_ ATTR_MISSING.
8. Set the encryption mode for the write permission.

Theory 4: Qualification

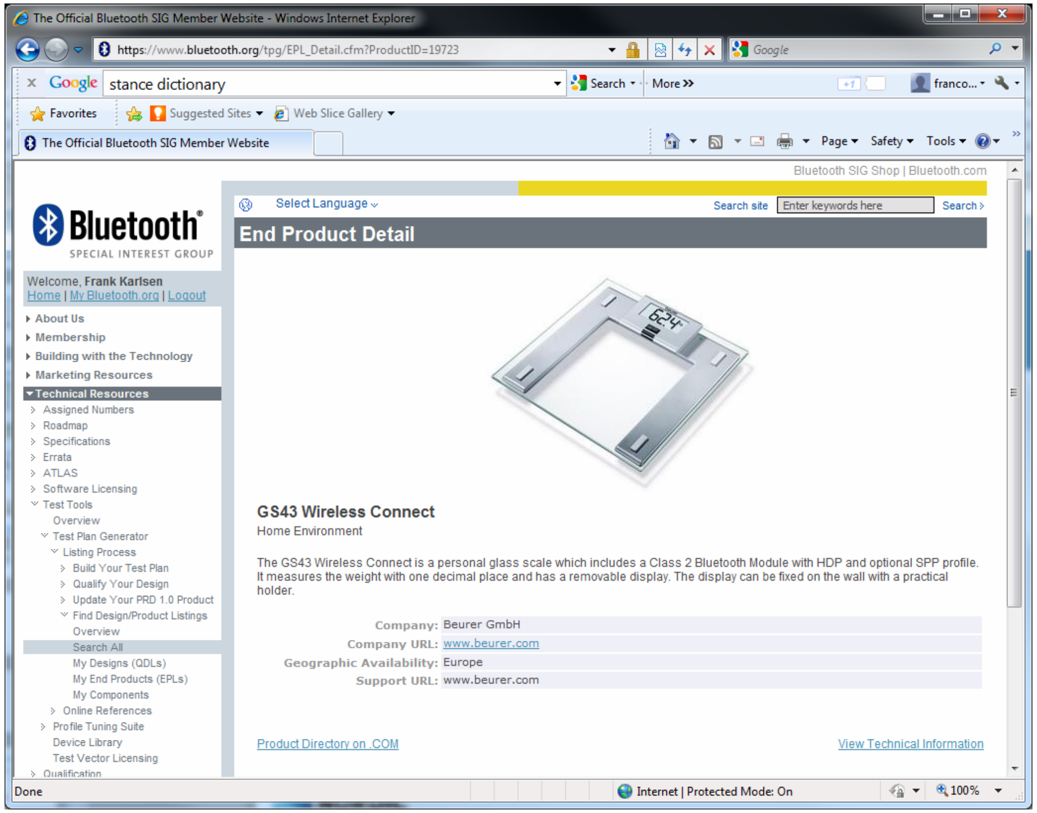
NORDIC
SEMICONDUCTOR

What do customers need to create (and understand)?

- EPL : End product listing
  - End products intended for sale to the consumer (Example)
  - All EPLs refer to one (or two) EP-QDLs
- EP-QDL : End product qualified design listing
  - Products/designs that are intended to be relisted by others (Example)
  - An EP-QDL may be required if the original EP-QDL needs modification
  - An EP-QDL can be used to create several EPLs
- BQTF - Bluetooth qualification test facility
  - A recognized test lab that can perform RF PHY testing and provide qualification assistance (for payment).
- BQE - Bluetooth qualification expert
  - A person associated with a BQTF that is accredited by the Bluetooth SIG to qualify products for customers (for payment)
- PCC – Product change checklist.
  - A guidance checklist aimed at identifying functionality that will require retesting in case of adjustments/modifications of the original EP-QDL.

End Product Listing (EPL)

- Generating a End product listing (i.e. an EPL) is free and does not require qualification/testing.
- An EPL always references a EP-QDL which is the base/original design. The qualification effort is linked to this EP-QDL.
- Generating a design that can be relisted as an EPL (i.e. an EP-QDL) requires a qualification fee and testing.



End Product Detail

GS43 Wireless Connect

Home Environment

The GS43 Wireless Connect is a personal glass scale which includes a Class 2 Bluetooth Module with HDP and optional SPP profile. It measures the weight with one decimal place and has a removable display. The display can be fixed on the wall with a practical holder.

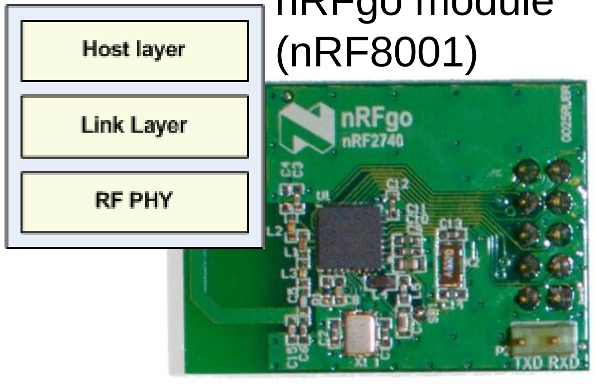
Company:	Beurer GmbH
Company URL:	www.beurer.com
Geographic Availability:	Europe
Support URL:	www.beurer.com

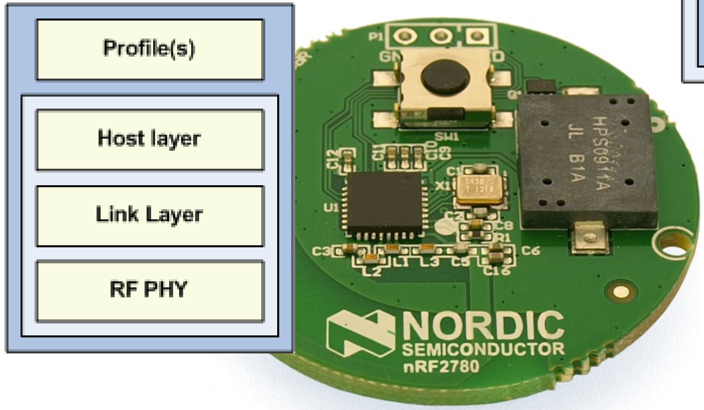
Product Directory on .COM

View Technical Information

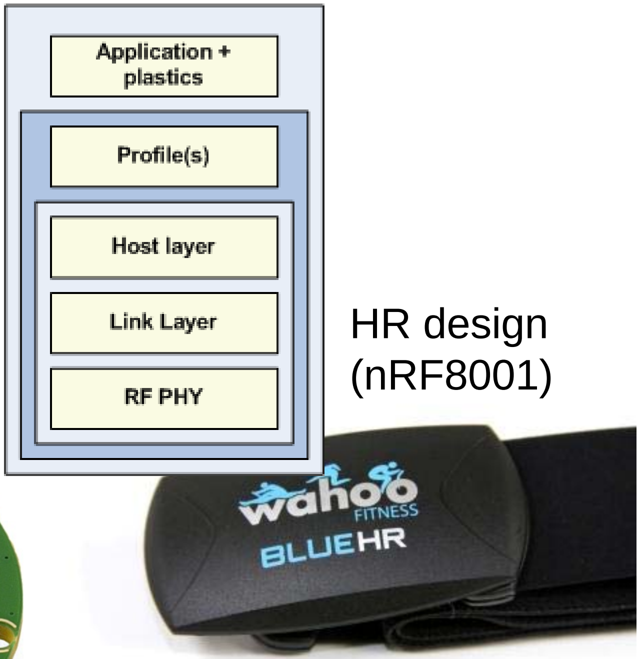
Real life EP examples

nRFgo module
(nRF8001)





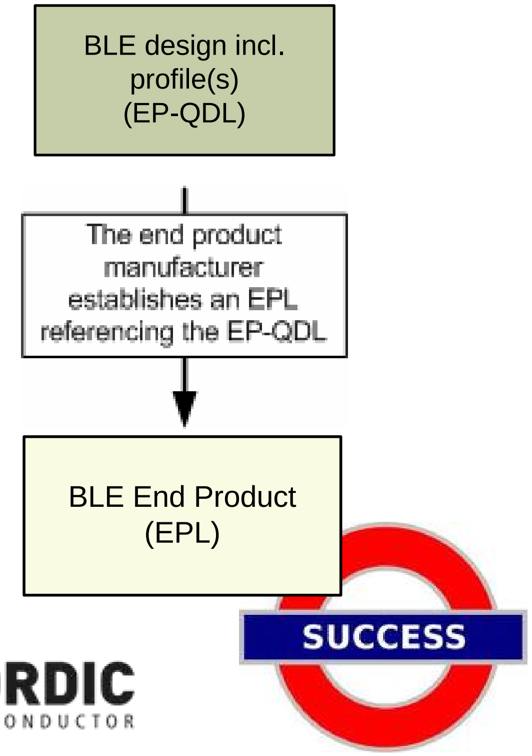
Proximity keyfob
(nRF8002)

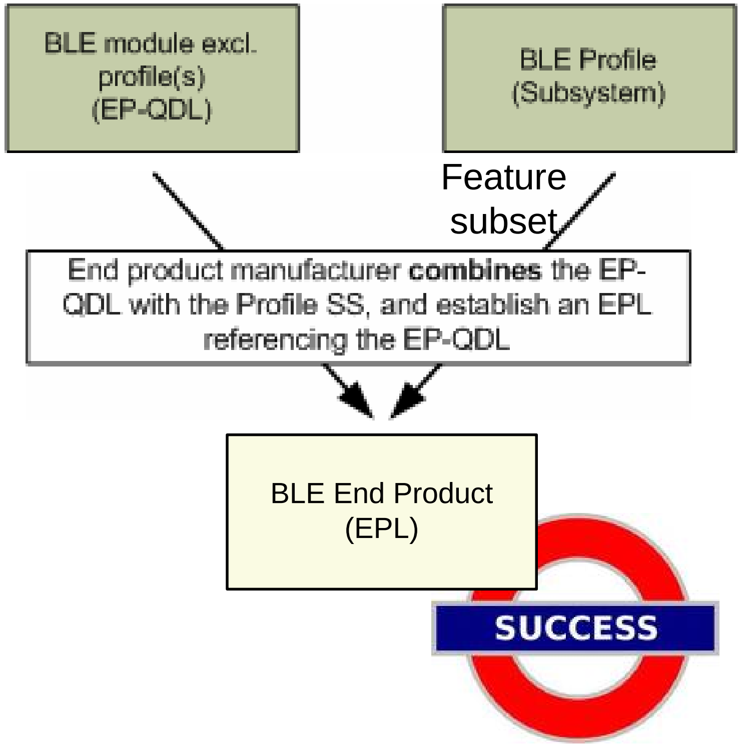


HR design
(nRF8001)

How to create an EPL

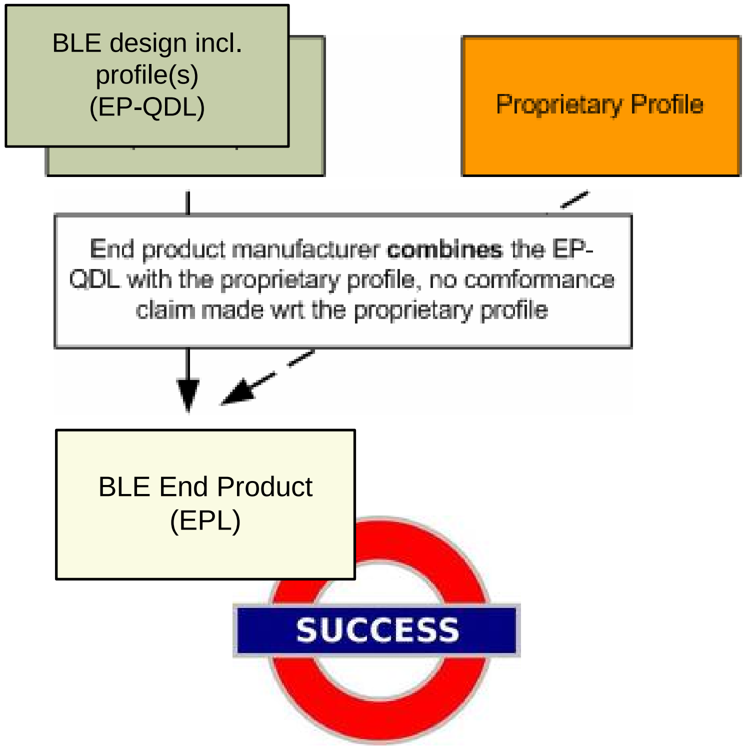
- The easiest way: Relisting SoC based product (or an OEM design)
- Moderate/high complexity: Combining an EP-QDL with a Profile Subsystem


BLE design incl. profile(s) (EP-QDL)
The end product manufacturer establishes an EPL referencing the EP-QDL
BLE End Product (EPL)
SUCCESS


BLE module excl. profile(s) (EP-QDL)
BLE Profile (Subsystem)
Feature subset
End product manufacturer **combines** the EP-QDL with the Profile SS, and establish an EPL referencing the EP-QDL
BLE End Product (EPL)
SUCCESS

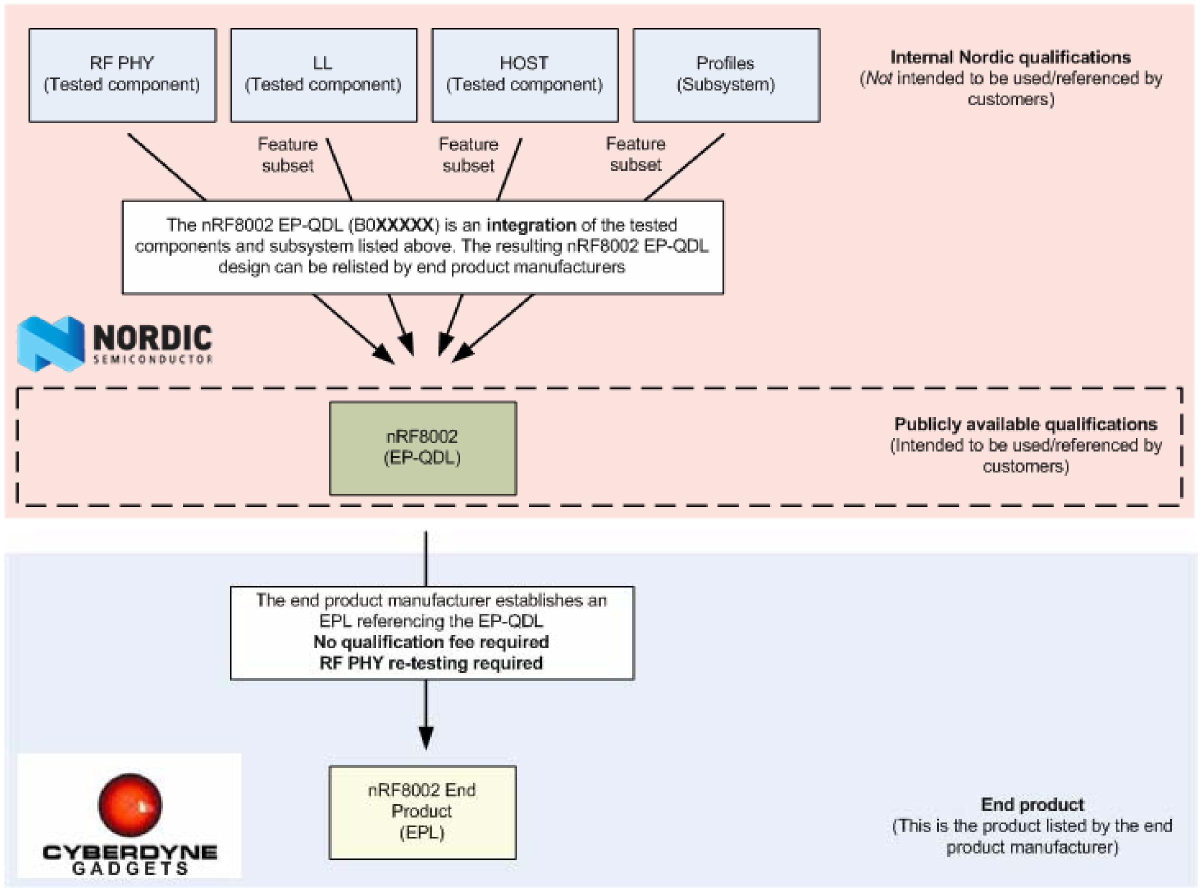
How to create an EPL

- Moderate/high complexity: Adding a proprietary profile
  - EPM creates an APP that renders support from phone vendors uneccessary to get to market, the cost is interoperability
  - Proprietary UUID encouraged but not a hard requirement
  - Note that an proprietary APP is not covered by the Bluetooth Patent and license agreement
  - May be combined with qualified profiles


BLE design incl. profile(s) (EP-QDL)
Proprietary Profile
End product manufacturer **combines** the EP-QDL with the proprietary profile, no comformance claim made wrt the proprietary profile
BLE End Product (EPL)
SUCCESS

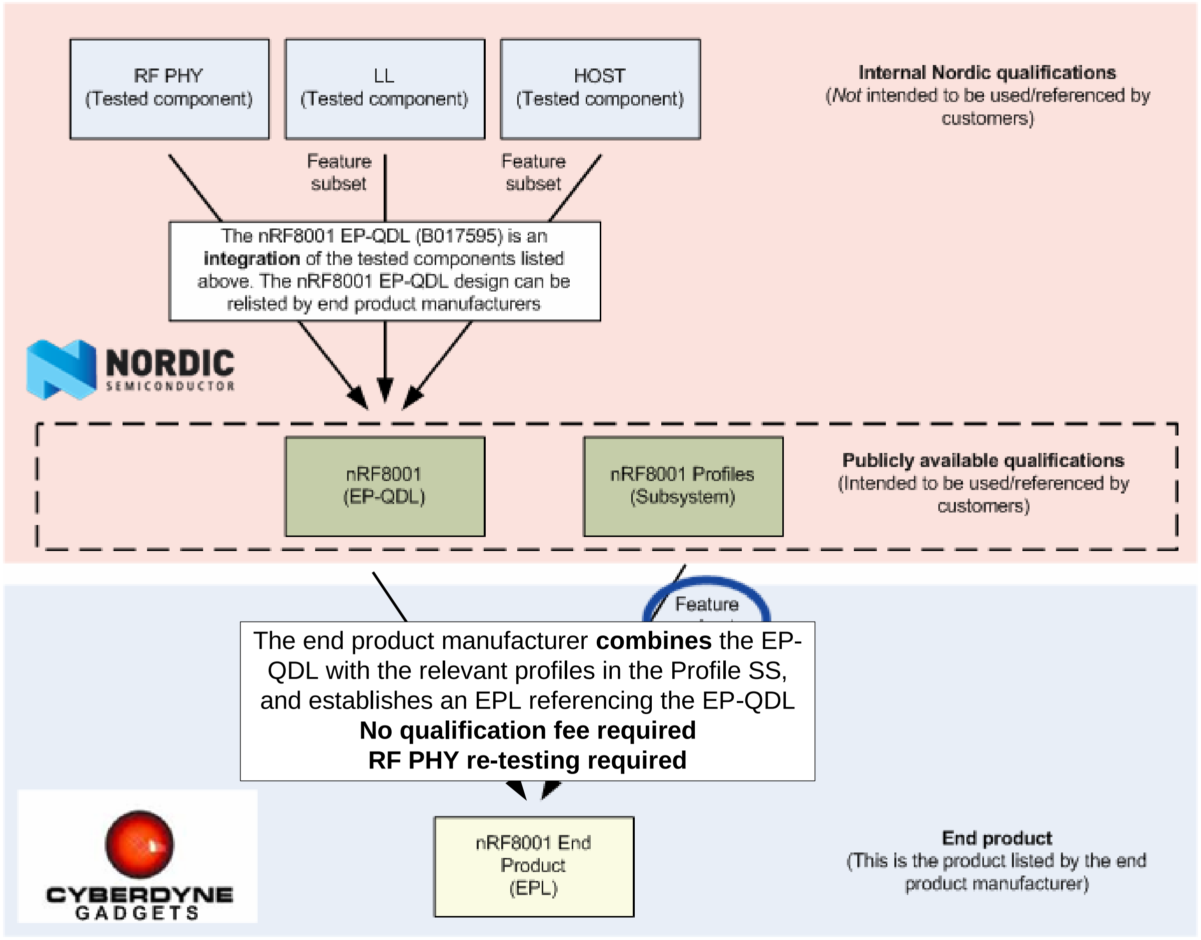
Real life example : nRF8002 keyfob

- Nordic creates the base design (the nRF8002 Development kit)
- Based on this, an EPM creates his keyfob design
- The EPM references our nRF8002 EP-QDL and creates an EPL
- **RF PHY re-testing required**


RF PHY (Tested component)
LL (Tested component)
HOST (Tested component)
Profiles (Subsystem)
Internal Nordic qualifications (Not intended to be used/referenced by customers)
Feature subset
Feature subset
Feature subset
The nRF8002 EP-QDL (B0XXXXX) is an integration of the tested components and subsystem listed above. The resulting nRF8002 EP-QDL design can be relisted by end product manufacturers
NORDIC SEMICONDUCTOR
nRF8002 (EP-QDL)
Publicly available qualifications (Intended to be used/referenced by customers)
The end product manufacturer establishes an EPL referencing the EP-QDL
No qualification fee required
RF PHY re-testing required
CYBERDYNE GADGETS
nRF8002 End Product (EPL)
End product (This is the product listed by the end product manufacturer)

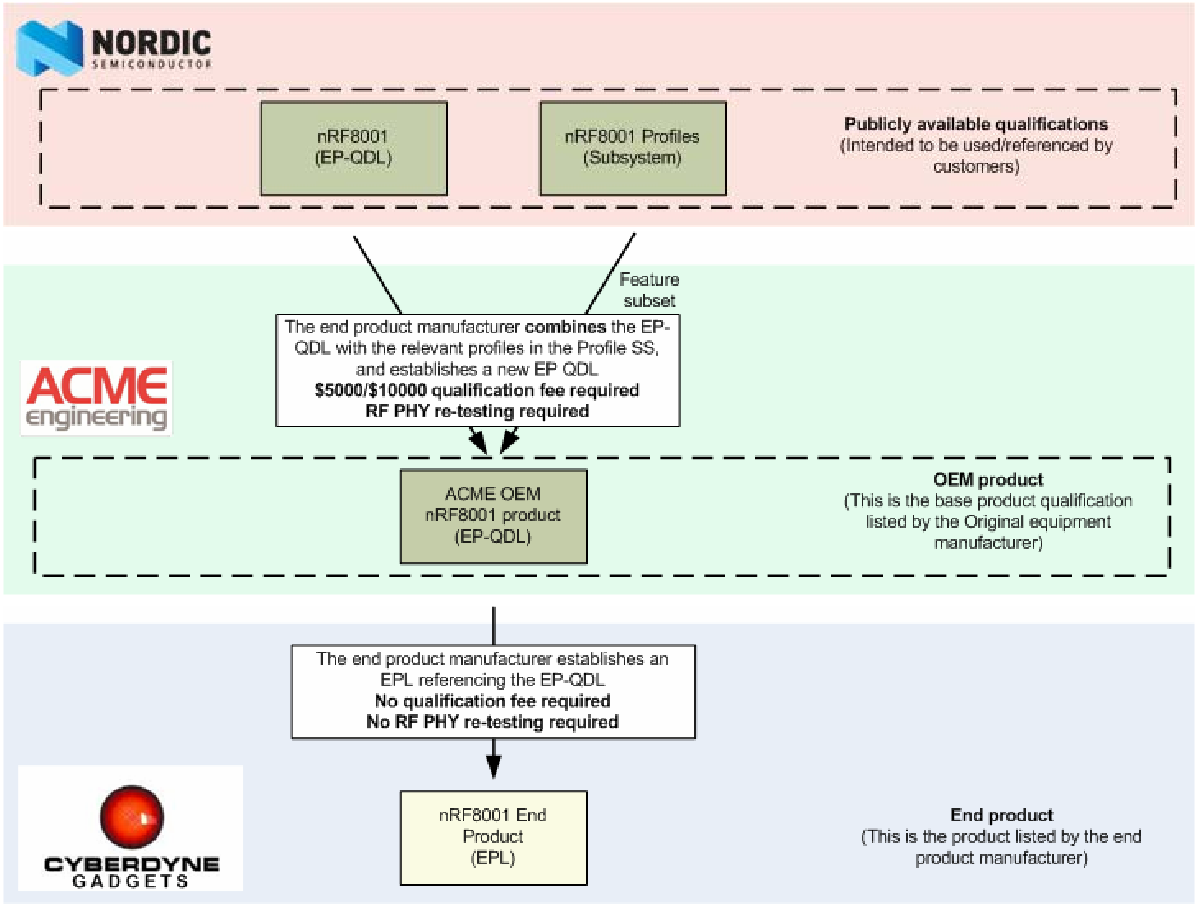
Real life example : nRF8001 + profiles

- The EPM creates a subset of our profile SS, selecting the profile(s) needed
- The EPM then combines the subset with the nRF8001 EP-QDL creating the EPL
- **RF PHY re-testing required by the EPM**
- **Due care needs to be taken when porting the profile code to another MCU (ref. App note on topic)**


RF PHY (Tested component)
LL (Tested component)
HOST (Tested component)
Internal Nordic qualifications (Not intended to be used/referenced by customers)
Feature subset
Feature subset
The nRF8001 EP-QDL (B017595) is an integration of the tested components listed above. The nRF8001 EP-QDL design can be relisted by end product manufacturers
NORDIC SEMICONDUCTOR
nRF8001 (EP-QDL)
nRF8001 Profiles (Subsystem)
Publicly available qualifications (Intended to be used/referenced by customers)
Feature
The end product manufacturer combines the EP-QDL with the relevant profiles in the Profile SS, and establishes an EPL referencing the EP-QDL
No qualification fee required
RF PHY re-testing required
CYBERDYNE GADGETS
nRF8001 End Product (EPL)
End product (This is the product listed by the end product manufacturer)

Real life example : nRF8001 OEM scenario

- OEM establishes new EP-QDL
- Qualification fee, assessment and test costs are taken care of by OEM
- EPMs generates new EPL by referencing the OEM EP-QDL
- **No RF PHY testing or qualifcation costs required for EPM**
- Can also be done for nRF8002 in order to spare EPM of test costs


NORDIC SEMICONDUCTOR
nRF8001 (EP-QDL)
nRF8001 Profiles (Subsystem)
Publicly available qualifications (Intended to be used/referenced by customers)
Feature subset
The end product manufacturer combines the EP-QDL with the relevant profiles in the Profile SS, and establishes a new EP QDL
$5000/$10000 qualification fee required
RF PHY re-testing required
ACME engineering
ACME OEM nRF8001 product (EP-QDL)
OEM product (This is the base product qualification listed by the Original equipment manufacturer)
The end product manufacturer establishes an EPL referencing the EP-QDL
No qualification fee required
No RF PHY re-testing required
CYBERDYNE GADGETS
nRF8001 End Product (EPL)
End product (This is the product listed by the end product manufacturer)

Assessment and re-testing

- A design that have been qualified by company A can later be relisted by company B
  - Company A bears the burden of qualification costs, company B piggybacks the qualification at no cost
- This is under the assumption that **the design is not changed**:
  - No extra features can be added
  - PCB/BoM modifications will require re-testing

- Any modification (however miniscule) to a design will need to be assessed:
  - I.e. «Has my modification invalidated any part of the qualification?»
  - If so, the corresponding qualification tests will need to be re-run

- **The responsibility for maintaining qualification integrity always rests on the relisting company.**
  - Even so, substandard products (i.e. modified and not re-tested) referring to a Nordic qualification may have a negative effect on Nordics reputation if these cause IOP problems or bad user experience.

NORDIC SEMICONDUCTOR

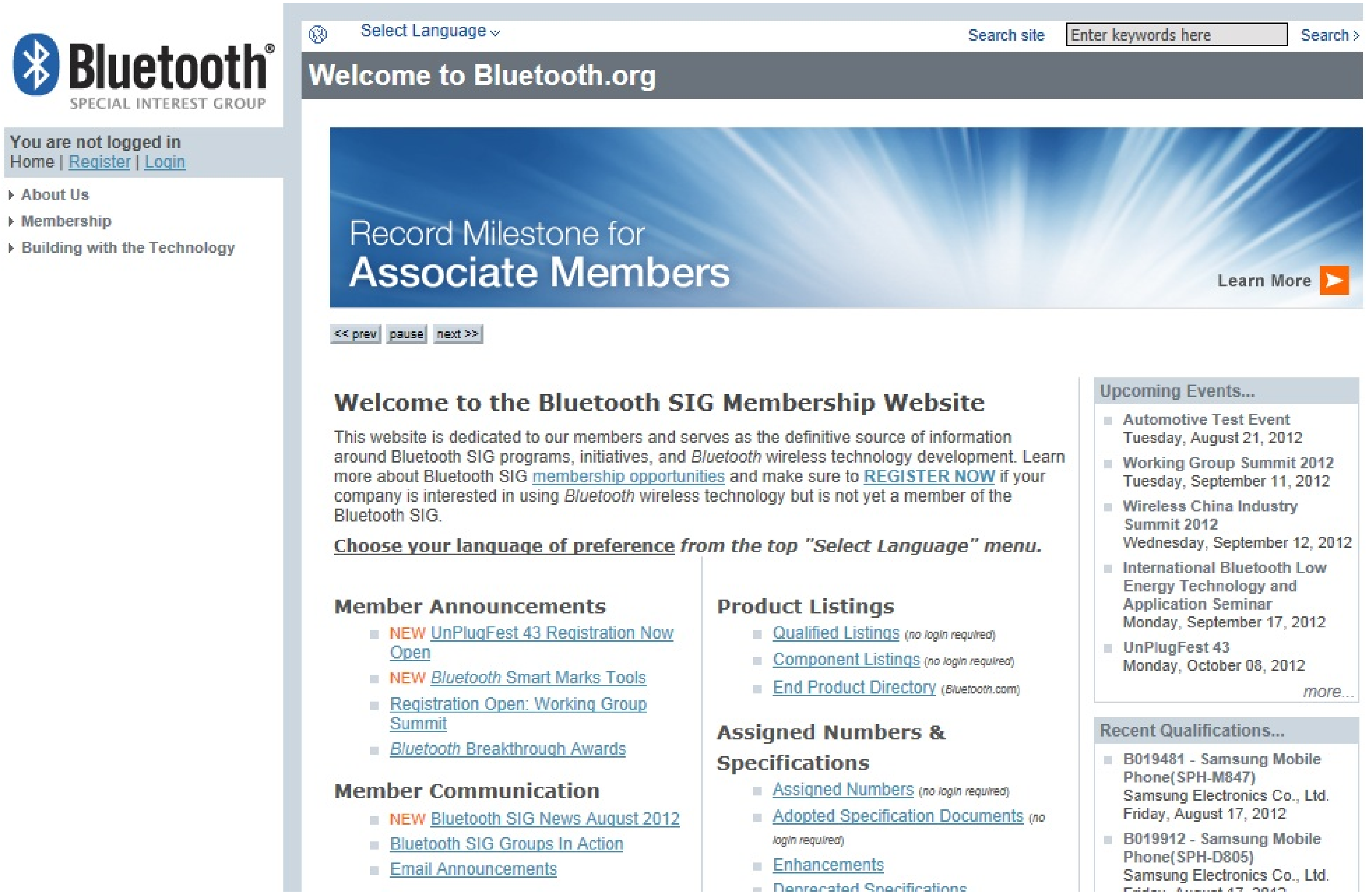
Cost structure

EPL relisting	free
EP-QDL listing (full)	\$5k/\$10k[1]
EP-QDL listing (enhancement)	\$2k/\$4k[1] (40%)
RF PHY testing	2.5k€[2]
BQE assistance	1.8k€[2]

- Top tip: 7layers offer cost competitive pricing and predictable cost based knowledge to Nordic BLE products calculator (reference)

- 1) Depending on membership level
- 2) 7Layers pricing

Bluetooth SIG website
www.bluetooth.org



You are not logged in
Home | Register | Login

Welcome to Bluetooth.org

Welcome to the Bluetooth SIG Membership Website

This website is dedicated to our members and serves as the definitive source of information around Bluetooth SIG programs, initiatives, and *Bluetooth* wireless technology development. Learn more about Bluetooth SIG membership opportunities and make sure to **REGISTER NOW** if your company is interested in using *Bluetooth* wireless technology but is not yet a member of the Bluetooth SIG.

Choose your language of preference *from the top "Select Language" menu.*

Member Announcements
- NEW UnPlugFest 43 Registration Now Open
- NEW *Bluetooth* Smart Marks Tools
- Registration Open: Working Group Summit
- *Bluetooth* Breakthrough Awards

Member Communication
- NEW Bluetooth SIG News August 2012
- Bluetooth SIG Groups In Action
- Email Announcements

Product Listings
- Qualified Listings *(no login required)*
- Component Listings *(no login required)*
- End Product Directory *(Bluetooth.com)*

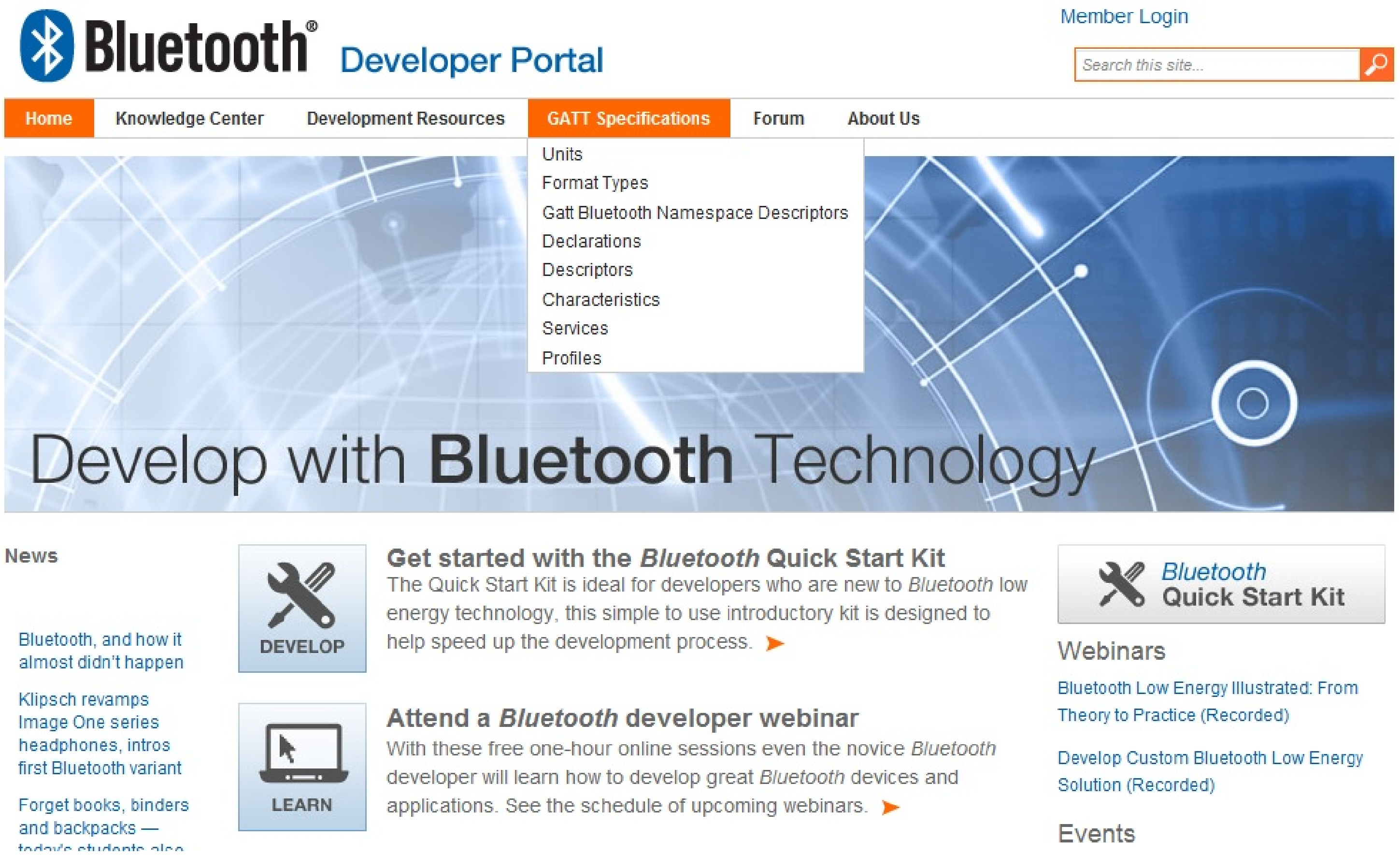
Assigned Numbers & Specifications
- Assigned Numbers *(no login required)*
- Adopted Specification Documents *(no login required)*
- Enhancements

Upcoming Events...
- Automotive Test Event
  Tuesday, August 21, 2012
- Working Group Summit 2012
  Tuesday, September 11, 2012
- Wireless China Industry Summit 2012
  Wednesday, September 12, 2012
- International Bluetooth Low Energy Technology and Application Seminar
  Monday, September 17, 2012
- UnPlugFest 43
  Monday, October 08, 2012

more...

Recent Qualifications...
- B019481 - Samsung Mobile Phone(SPH-M847)
  Samsung Electronics Co., Ltd.
  Friday, August 17, 2012
- B019912 - Samsung Mobile Phone(SPH-D805)
  Samsung Electronics Co., Ltd.

Bluetooth SIG developer website developer.bluetooth.org



Bluetooth® Developer Portal

Member Login

Search this site...

Home | Knowledge Center | Development Resources | GATT Specifications | Forum | About Us

- Units
- Format Types
- Gatt Bluetooth Namespace Descriptors
- Declarations
- Descriptors
- Characteristics
- Services
- Profiles

Develop with **Bluetooth** Technology

News

Bluetooth, and how it almost didn't happen

Klipsch revamps Image One series headphones, intros first Bluetooth variant

Forget books, binders and backpacks — today's students also

DEVELOP

Get started with the *Bluetooth* Quick Start Kit

The Quick Start Kit is ideal for developers who are new to *Bluetooth* low energy technology, this simple to use introductory kit is designed to help speed up the development process. ➤

LEARN

Attend a *Bluetooth* developer webinar

With these free one-hour online sessions even the novice *Bluetooth* developer will learn how to develop great *Bluetooth* devices and applications. See the schedule of upcoming webinars. ➤

Bluetooth Quick Start Kit

Webinars

Bluetooth Low Energy Illustrated: From Theory to Practice (Recorded)

Develop Custom Bluetooth Low Energy Solution (Recorded)

Events

NORDIC SEMICONDUCTOR

Practical Session 5 (30 min)

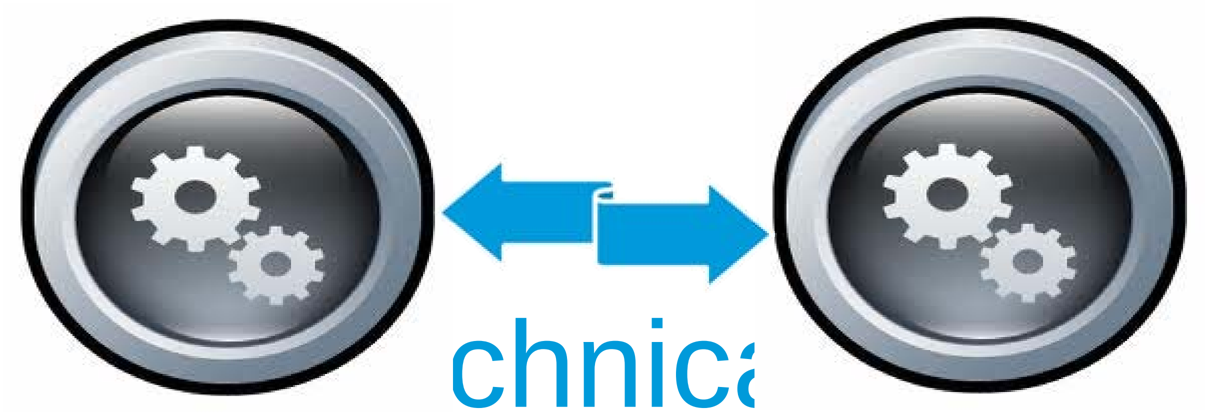
- Navigate the Bluetooth SIG and BT SIG developer websites (30 min)
  - Locate a specific specification, associated UUIDs

Quiz (45 min)

- Multiple-choice questionnaire

Product Management

Part 1



chnica Training

NORDIC SEMICONDUCTOR