

Developing Bluetooth® Low Energy Solution with TI SimpleLink™ Ultra Low Power Platform

Agenda






**CC2640 Wireless MCU
BLE-Stack™ 2.1
CC2650 SensorTag**

Tim Camise
*Applications Engineering Manager
Bluetooth-Smart
Low Power RF – San Diego*



TI Introduces *Next Generation Platform*

Improving the three key challenges for a *Bluetooth* Smart product:

 Easiest to design with	 Lowest Power	 Most Integrated
<ul style="list-style-type: none">• Qualified BT 4.1 Bluetooth Smart• Get-Started Documentation & Wiki• Dynamic Design Kits• Low-cost Tools 	<ul style="list-style-type: none">• ~6mA Radio peaks and 1uA Sleep• 65µA/MHz ARM Cortex M3• <10 uA avg. Current @ 1s Conn. Int• Sensor Controller Engine (SCE)	<ul style="list-style-type: none">• 4x4, 5x5, 7x7 QFN• On-Chip Flash• Single Ended Output• Integrated DCDC 
<i>Comprehensive Design Support</i>	<i>Multi-year operation on a coin cell</i>	<i>Complete Bluetooth Smart system on a finger-tip size</i>

"CC2640 puts Smart in Bluetooth SMART "

TI Bluetooth Smart Wireless MCU Portfolio

CC2640

Puts the SMART in Bluetooth Smart!

Easiest to design with

- Comprehensive design support: Complete SW stack, wiki guides, dynamic design kits, low-cost tools, & software starting points



Lowest power

- Multi-year on a coin cell: Cortex M3 MCU, radio, sleep current and unique Sensor Controller



Most integrated

- Complete solution on a finger tip: Single chip, flash-based 4x4mm QFN with only one crystal



CC2540

Add Bluetooth Smart to USB Solutions



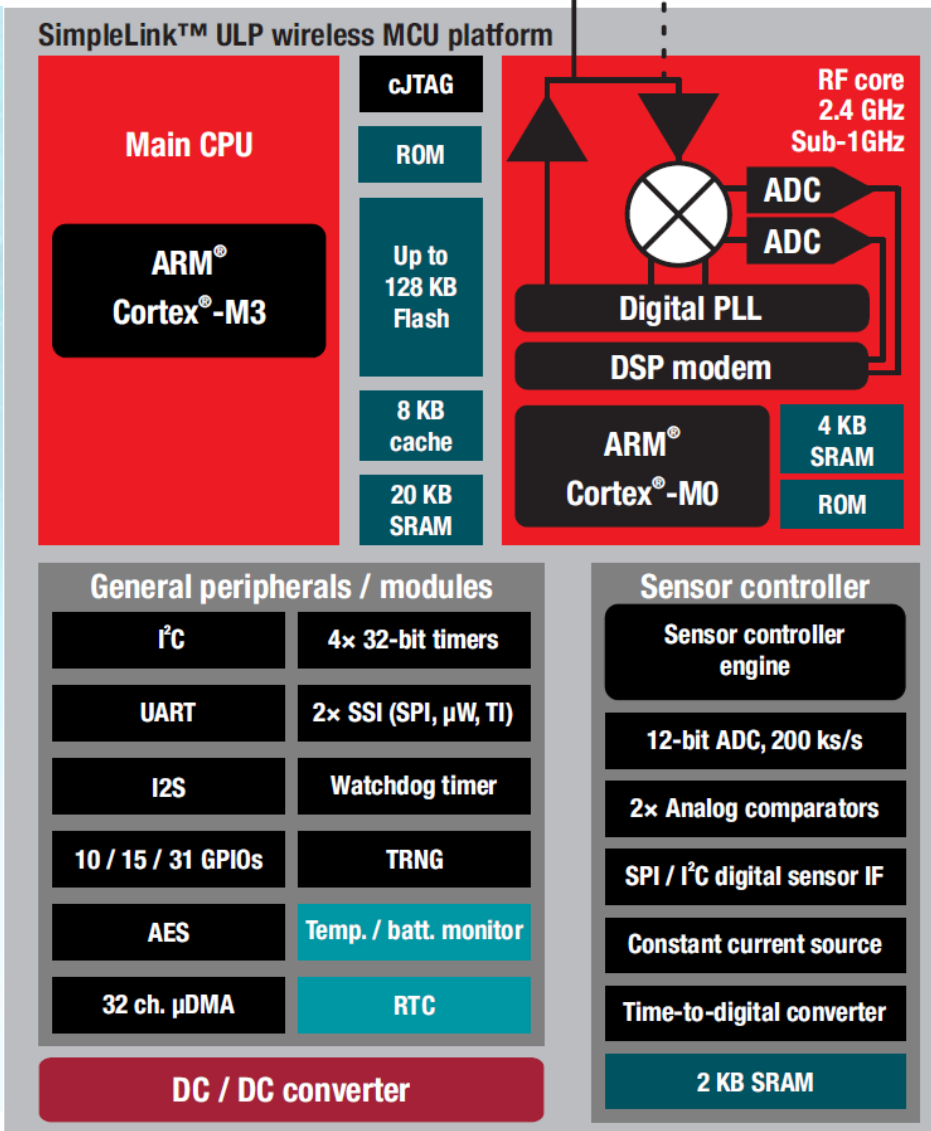
CC2540T

World's only 125°C graded Bluetooth Smart solution

CC2541Q1

Automotive Qualified

SimpleLink™ Bluetooth® Smart CC2640 wireless MCU



Quick Facts

Ultra-low Power Consumption

- 61 µA/MHz ARM Cortex M3
- 8.2 µA/MHz Sensor Controller
- 1 µA sleep with retention and RTC
- 5.9 mA RX (single-ended)
- 6.2 mA TX (single-ended)
- < 3uA while running 10 ADC samples/s

Key Features

- Autonomous sensor controller engine
- 4x4, 5x5, and 7x7 mm QFN
- 1.7 – 1.95 V or 1.8 – 3.8 V supply range
- 128 KB Flash + 8 KB Cache
- 20 KB RAM

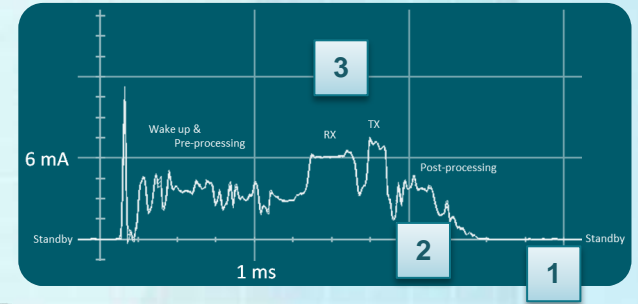
RF Key Features

- +5 dBm output power
- -97 dBm sensitivity
- Pin compatible and SW compatible across protocols and frequency bands

Power Consumption

Low average power consumption

1. When in Standby (with RTC and RAM retention)
2. When processing with MCU
3. When radio is in Receive or Transmit
4. When peripheral is polled for data

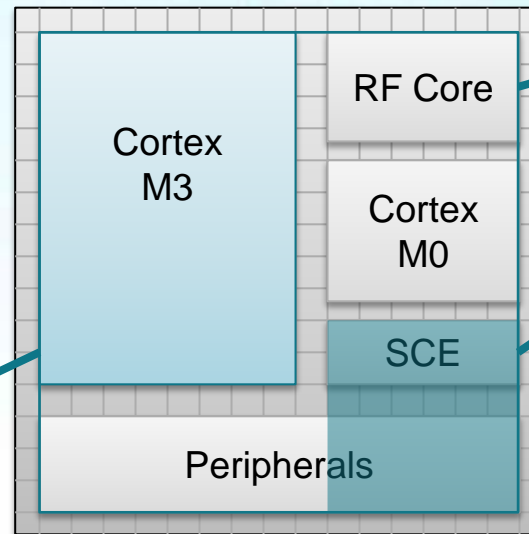


1 Standby Current

- **1 μ A** Standby with RTC and full retention
- **150 nA** in Shutdown

2 ARM Cortex M3

- Fast processing using less than **3 mA** @ 48MHz
- Less time used for stack and application processing and BLE connection events



3 Radio

- **~6mA** RX / TX current

4 Sensor Controller Engine

- Sensor controller and its peripherals can be powered while rest of system is power off.
- Run Sensor Reading with **< 5 μ A** current consumption

Application Note SWRA478: [Measuring Bluetooth Smart Power Consumption](#)

Radio Frontend *Flexibility*

Best Performance

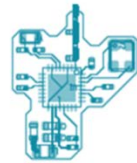
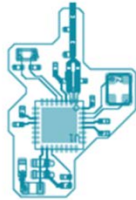
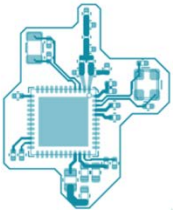
- Differential, external bias
- 5 dBm output power
- -97 dBm BLE sensitivity

Smallest Footprint

- Single ended, external bias
- 2 dBm output power
- -96 dBm BLE sensitivity

Other options are available. Internal biasing reduces BOM by one inductor at the cost of 1 dB sensitivity

NB! Differential External bias is not supported on 7x7

Evaluation Module		CC2650EM-4XS	CC2650EM-5XD	CC2650EM-7ID
CC2650 Package Type		4x4	5x5	7x7
Pitch [mm]		0.4	0.5	0.5
GPIOs		10	15	31
Design Example	RF Frontend Option	Single Ended External Bias	Differential External Bias	Differential Internal Bias
	Area [cm ²]	1.3	1.5	2.3
	Illustration			
	Crystals	2	2	2
	Capacitors	14	17	18
	Inductors	3	6	5
	Resistors	1	1	1
	Total	20	26	26

Bluetooth Smart *Reduce Time to Market*

Get Started

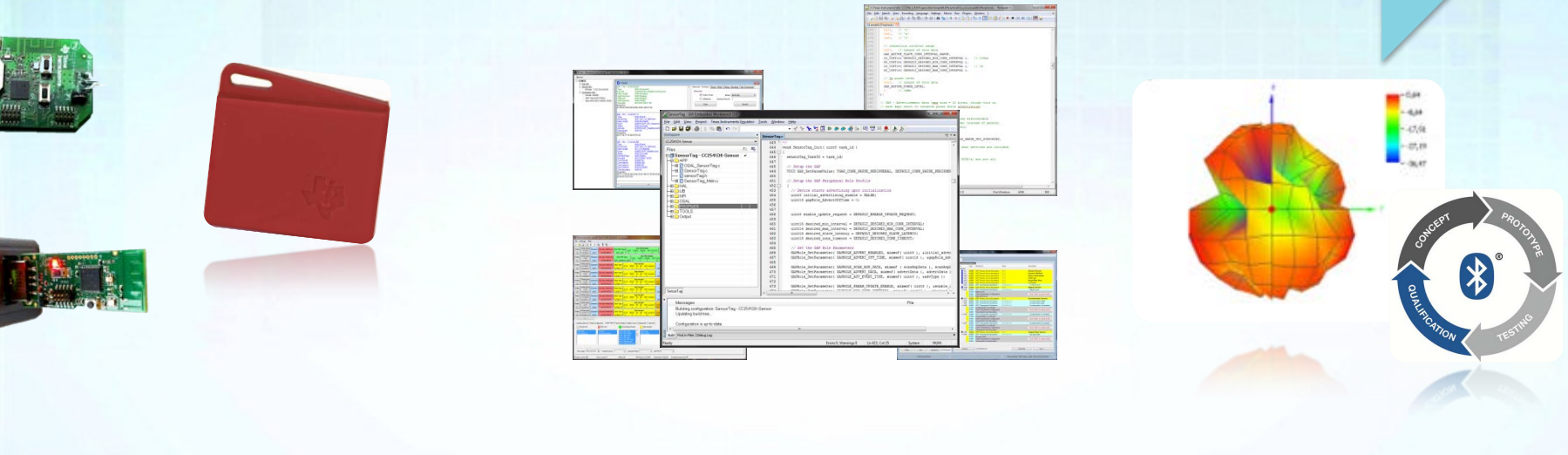
- Order Kit
- Download SDK
- Evaluate & Learn *Bluetooth* low energy

Develop

- Define Application
- Choose/Design Profiles
- Implement Prototype

Test & Release

- FCC/ETSI Certification
- *Bluetooth* Listing



Development Tools CC2650

SmartRF™ Tools

- SmartRF Studio
 - Radio performance evaluation, testing and configuration
- SmartRF Packet Sniffer
 - Capture over the air packets for RF link debugging and protocol analysis
- SmartRF Flash Programmer 2
 - Flash tool. Comes in command-line version for automated programming sequences.

Sensor Controller Studio

- Setup SCE tasks and code the behaviour script for them
- Generate driver source files for the CM3

Multiple software development tools supported

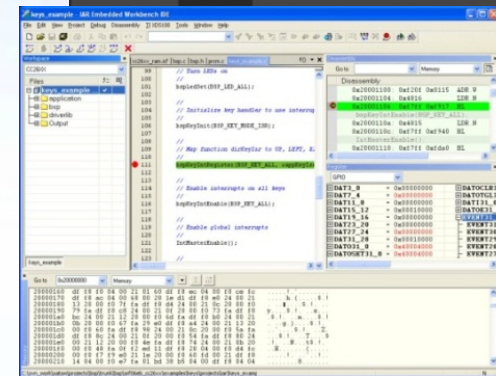
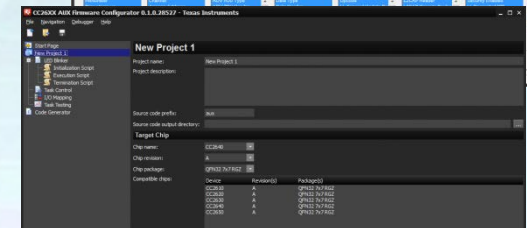
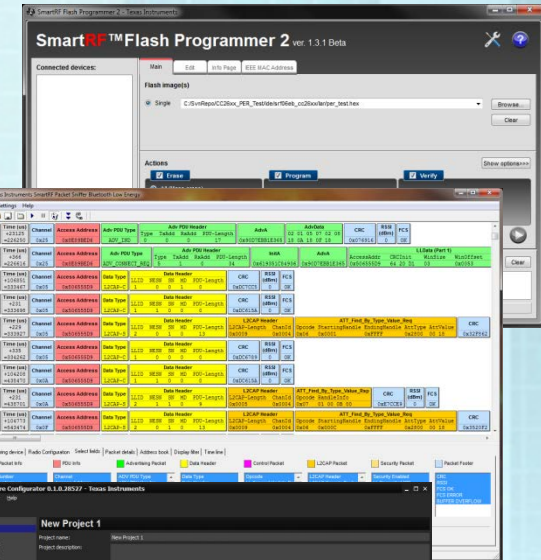
- IAR Embedded Workbench® for ARM
- Code Composer Studio



Bluetooth Developer Studio



- IDE developed by BT SIG to simplify BLE development
- TI Plugin for auto generating Custom & Adopted Profiles



Development Kits CC2650



CC2650DK
\$299



CC2650EMK
\$99



CC2650STK
\$29

Full feature development kit with embedded TI XDS emulator for development, debugging & power measurement.

EMK's for all CC2650 package types

SmartRF06 Features:

- Dot matrix LCD
- 4 LEDs
- 5 buttons
- Accelerometer
- Ambient Light Sensor
- UART backchannel
- I/O breakout headers

Application development & IoT demonstration platform powered by CR2032 Coin Cell Battery
Native sensor support for:

- 6-axis MEMS motion tracking
- Humidity (TI)
- IR temperature (TI)
- Light Sensor (TI)
- Microphone
- Pressure
- Reed Relay

Dev. Pack for custom functionality.

- Debugger, LCD etc.

Supported by accompanying iOS/Android apps

TI Designs *Bluetooth Smart*

Bluetooth® Smart



RS-485



Haptic Feedback



Gas Sensor



Keyfob



Light Harvesting



BLE Light



Postage Stamp



Long Range



Mini Broadcaster



Biometric Wheel



USB Dongle



SensorTag iBeacon



Heart Monitor



Body Composition



Optical Heart Rate

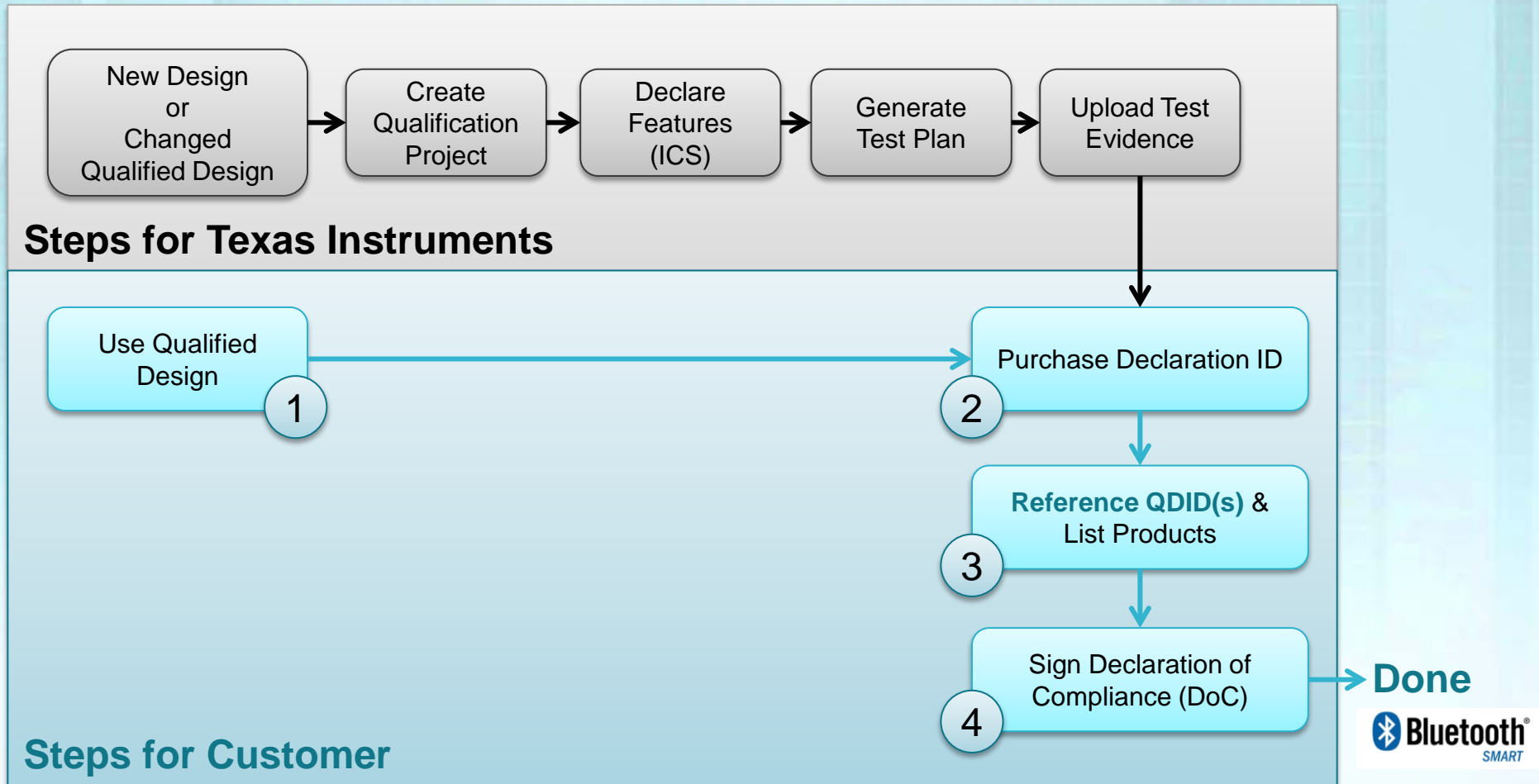


Weight Scale



Pulse Oximeter

Qualification *Listing* Process



See “Bluetooth Qualification” on TI BLE wiki: www.ti.com/ble-wiki

Reference *Qualified Design Listings*

Declaration ID: **D024979**

Qualified Design ID: **61713**

QDL Bluetooth® Qualified Design Listing

The Bluetooth SIG Hereby Recognizes

Texas Instruments Incorporated
Member Company

BLE2_x_CC26xx
Qualified Design Name

Declaration ID: D024979
Qualified Design ID: 61713

Reference

Specification Name: 4.1
Product Type: Component (Tested)
Model Number: BLE2_x
Listing Date: 18 February 2015
Hardware Version Number: CC26xx

Assessment Date: 22 November 2014
Software Version Number: v2.0

This certificate acknowledges the Bluetooth® Specifications declared by the member were achieved in accordance with the Bluetooth Qualification Process as specified within the Bluetooth Specifications and as required within the current PRD



QDL Bluetooth® Qualified Design Listing

The Bluetooth SIG Hereby Recognizes

Texas Instruments Incorporated
Member Company

SimpleLink BLE CC2640/CC2650
Qualified Design Name

Declaration ID: D024920
Qualified Design ID: 61691
Specification Name: 4.1
Product Type: Component (Tested)
Model Number: CC2650-71D
Listing Date: 18 February 2015
Hardware Version Number: 1.2.7 (2.0)

Assessment Date: 21 November 2014
Software Version Number: v2.0

This certificate acknowledges the Bluetooth® Specifications declared by the member were achieved in accordance with the Bluetooth Qualification Process as specified within the Bluetooth Specifications and as required within the current PRD

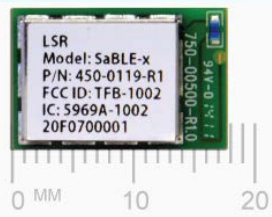


CC2640 *Module Partners*

Two modules already available, more to come..

Company	Part Number	TI Chip	Dimensions [mm] with Antenna	Temp Range [°C]	Certification
LSR	<u>SaBLE-x</u>	CC2640	11.6 x 17.9	-40°C to +85°C	FCC/ IC / CE / C-Tick / Giteki
Innocomm	BM10	CC2640	16.70 x 25.45 x 2.0	-40°C to +85°C	ETSI EN 300 328 and EN 300 440 Class 2(Europe) FCC CFR47 Part 15(US) ARIB STD-T66 (Japan)

SaBLE-x



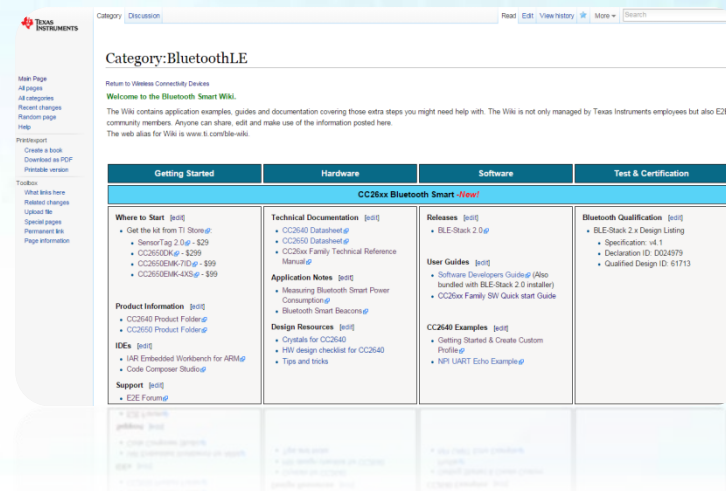
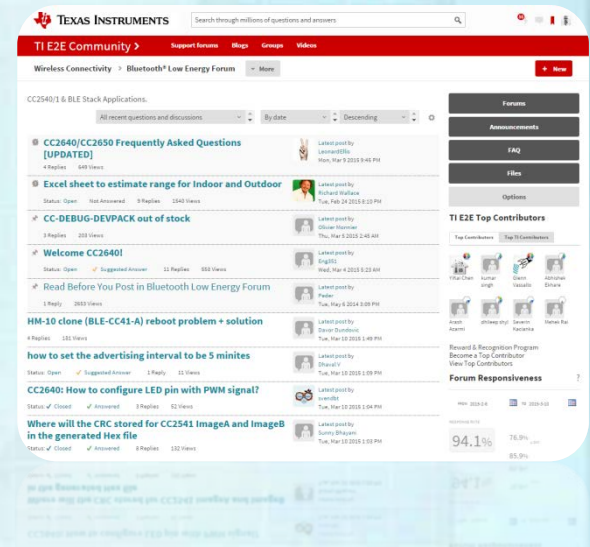
BM10



or use TI Reference Designs to create your own..

Bluetooth Smart Support

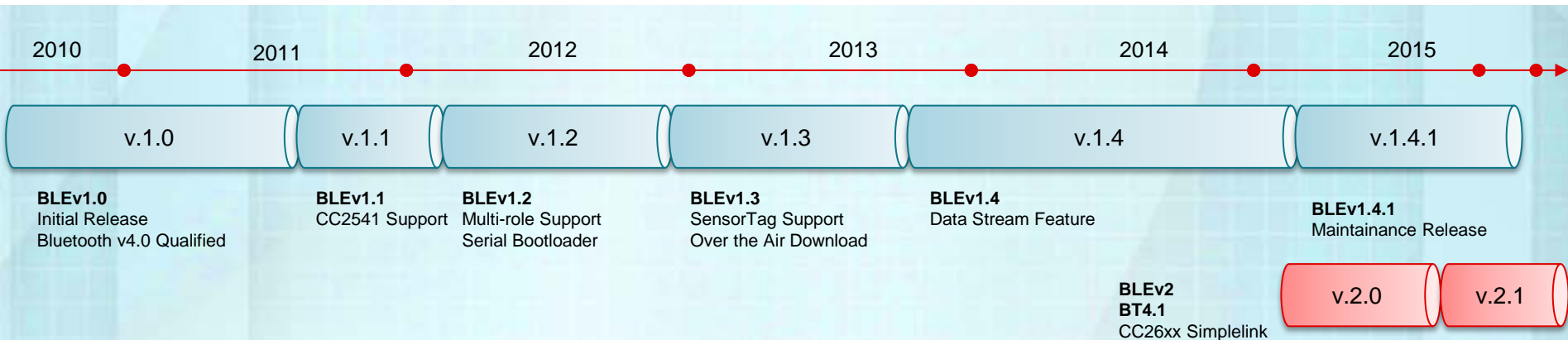
- Online E2E Support Community (www.ti.com/ble-forum)
 - Supervised by TI Software and Hardware Experts
- Guides included in the BLE installer (www.ti.com/ble-stack)
 - Software Developers Guide
 - Vendor Specific HCI Guide
- Online documentation (www.ti.com/ble-wiki)
 - Getting Started & User Guides
 - Code Examples
 - HW Design Resources
 - Training videos & guides
 - Walkthrough Guides



BLE-Stack™ 2.1 (July 2015)

Introduction

Software *BLE-Stack*™



- **Mature and Robust Software Package**
 - Released 2010, yearly updates with improvements and add-ons
 - Golden unit for Bluetooth low energy interoperability test
 - Fully BT Qualified Solution
- **Complete Power Optimized Bluetooth low energy Stack**
 - Central, Peripheral, Observer and/or Broadcaster Device Roles
 - GATT Client and Server
 - Security Manager with 128 bit AES Encryption and Decryption
 - Operating System for prioritized scheduling and task handling

Royalty Free Software Development Kit (SDK) Available at ti.com/ble-stack

Bluetooth Smart *Updated Specification*

BT4.1 *Adopted by BT SIG, December 2013*

Low Duty Cycle Directed Advertising

Lower-power / lower-latency implementation of directed advertisements is now allowed

32-Bit UUID Support

Allows more application data in advertisements (useful for Broadcast-mode applications)
Improves service discovery times (i.e. Faster initial pairing)

LE Ping

First step towards future enhanced security features in BLE (Core Spec Addendum)

L2CAP Connection Oriented Channels

Allows data to be sent without the overhead of ATT/GATT layers
Provides ~15% improvement in throughput compared to BT 4.0 using GATT (23-bytes per packets vs. 20-bytes per packet)

LE Link Layer Topology Enhancements

Allows for new BLE topologies such as simultaneous master/slave and slave connected to multiple masters

BT4.2 *Adopted by BT SIG, December 2014*

LE Secure Connections


Based on Diffie-Hellman Key Exchange, similar to the Secure Simple Pairing feature in classic Bluetooth
Closes the well-known security holes in BLE that exist in the BT 4.0/4.1 specs

LE Privacy 1.2

Keeps BLE devices from being tracked
Significant improvement over the confusing and little-used privacy features in BT 4.0/4.1 (which are being deprecated)

LE Data Length Extension

Increases data throughput by up to 2.5x by increasing the packet capacity by 10x



CC2640
capable of
supporting all
BT4.2
features!

BLE-Stack™ 2.1.0 *New Features*

- Simple Network Processor (SimpleNP)
 - Easily add BLE connectivity to an existing MCU system
 - Uses simplified API to manage CC2640 operating as network processor
 - Examples with MSP432 LP & CC2650DK
 - Works with TI's HomeKit implementation
- Off-Chip OAD
 - Allows FW update using off-chip flash memory
 - Example implemented with SPI Flash on CC2650 SensorTag
- TI-RTOS driver update
- Improvements to Flash memory configurations
 - More options to fine-tune for additional Application Flash memory
- Power Optimization
 - Updated DCDC re-charge algorithm improves standby power consumption

BLE-Stack™ 2.1.0 Highlights

- **Fully Qualified BT4.1 Protocol Stack**
 - All BT Core Specification features included

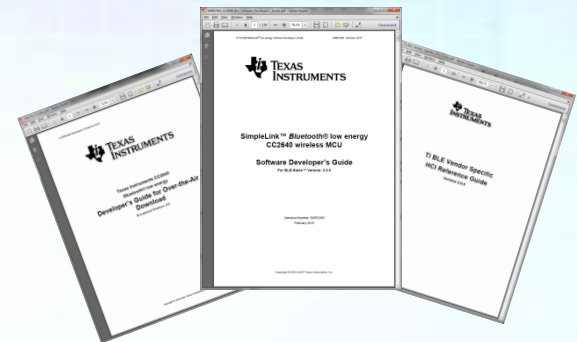


- **IDE Support**
 - IAR Embedded Workbench for ARM v7.40
 - Code Composer Studio (CCS) v6.1



- **TI-RTOS**
 - Simplified Management & Drivers
 - Pre-emptive Multi-threaded environment

- **Documentation**
 - Software Developers Guide
 - TI-RTOS Document Overview
 - CC2640 Simple Network Processor API Guide
 - CC2640 BLE OAD User's Guide
 - Vendor Specific HCI & SimpleNP API Guides



BLE-Stack™ 2.1.0 *Sample Applications*

- Get started easy with Sample Applications!

Example Application	Description
SimpleBLEPeripheral	Generic Peripheral using proprietary Profile example
SimpleBLECentral	Generic Central
SimpleBLEBroadcaster	Generic Broadcaster
SimpleBLEObserver	Generic Observer
SimpleNP / SimpleAP	Simple Network & Application Processor
SensorTag	SensorTag 2.0 Firmware
OAD Target	Example of OTA Firmware Update
HostTestApp	HCI Network Processor (Application via SPI/UART)
BloodPressure	Example using Blood Pressure Profile (BSP)
CyclingSensor	Example using Cycling Speed and Cadence Profile (CSCP)
GlucoseCollector	Example using Glucose Profile (GLP) as Collector
GlucoseSensor	Example using Glucose Profile (GLP) as Sensor
HeartRate	Example using Heart Rate Profile (HRP)
HIDEmuKbd	Example using HID over GATT Profile (HOGP)
ProximityTag	Example using Proximity Profile (PXP)
RunningSensor	Example using Running Speed and Cadence Profile (RSCP)
Thermometer	Example using Health Thermometer Profile (HTP)
TimeApp	Example using Time Profile (TIP)

New:
Quick and easy
option to add
BLE to any MCU
based system

Additional Services

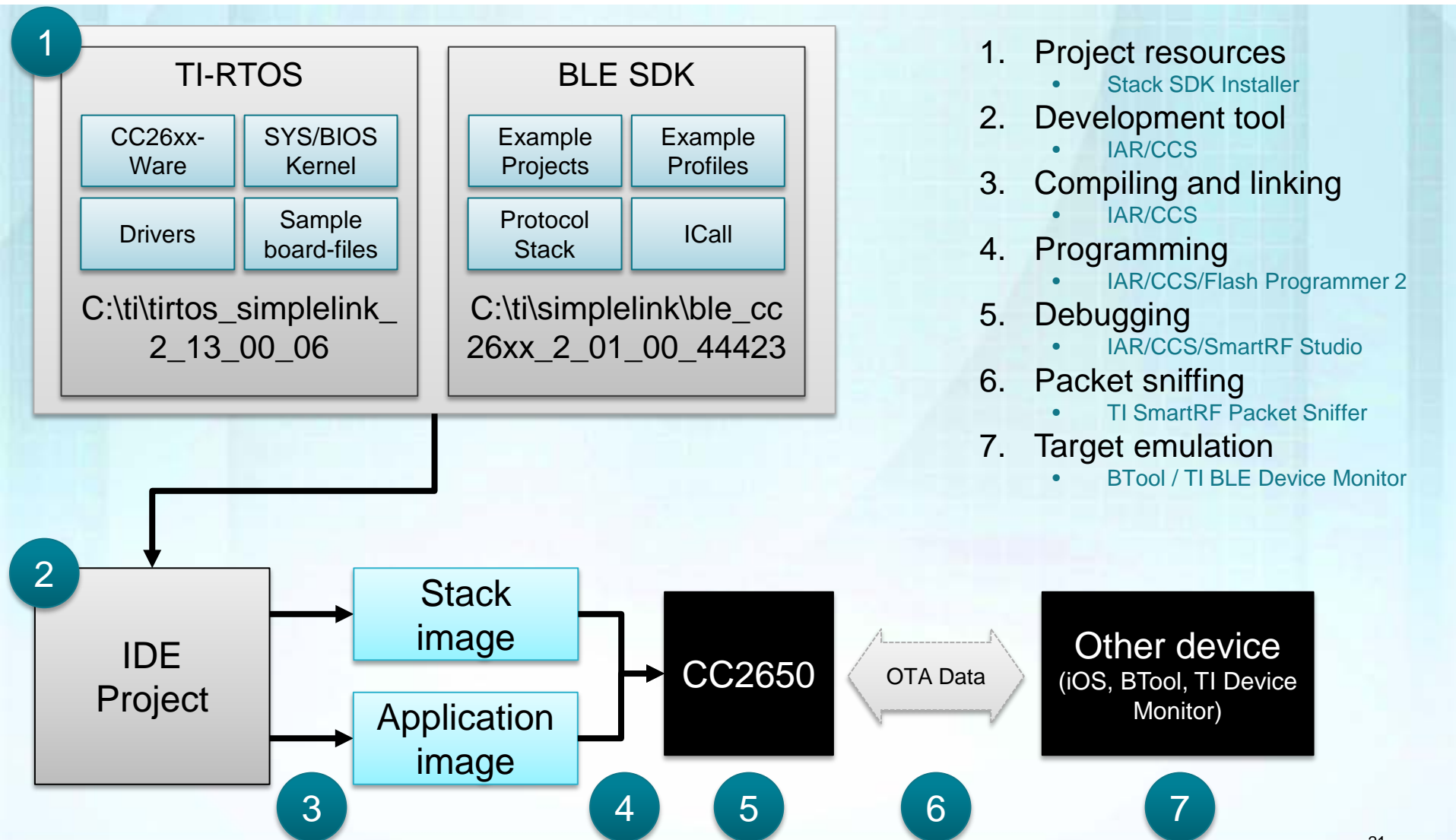
Proprietary

Accelerometer
Barometer
Gyrometer
Humidity
IR Temperature
Magnetometer
Movement
Optics
Connection Control
Simple Keys

BT SIG Adopted

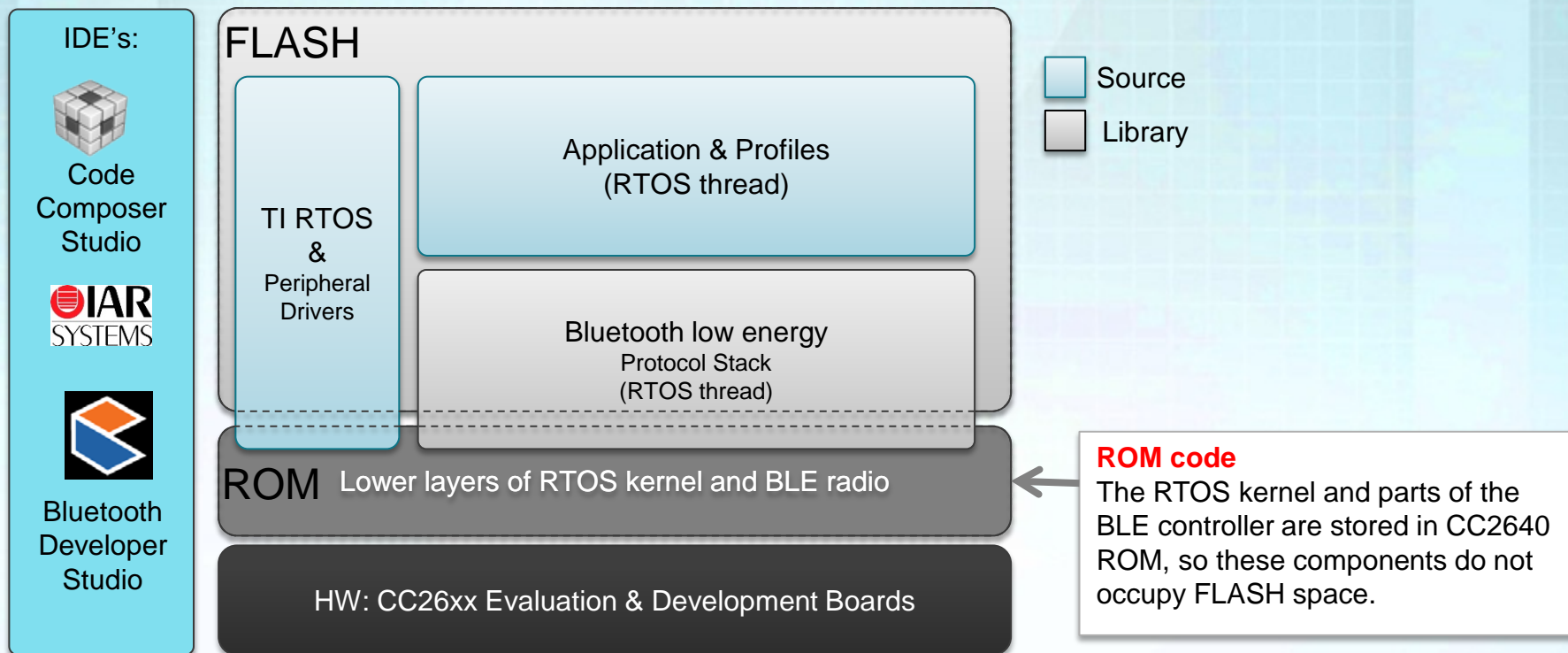
Find me
Alert Notification
Battery Status
Device Information

Bluetooth Smart SDK



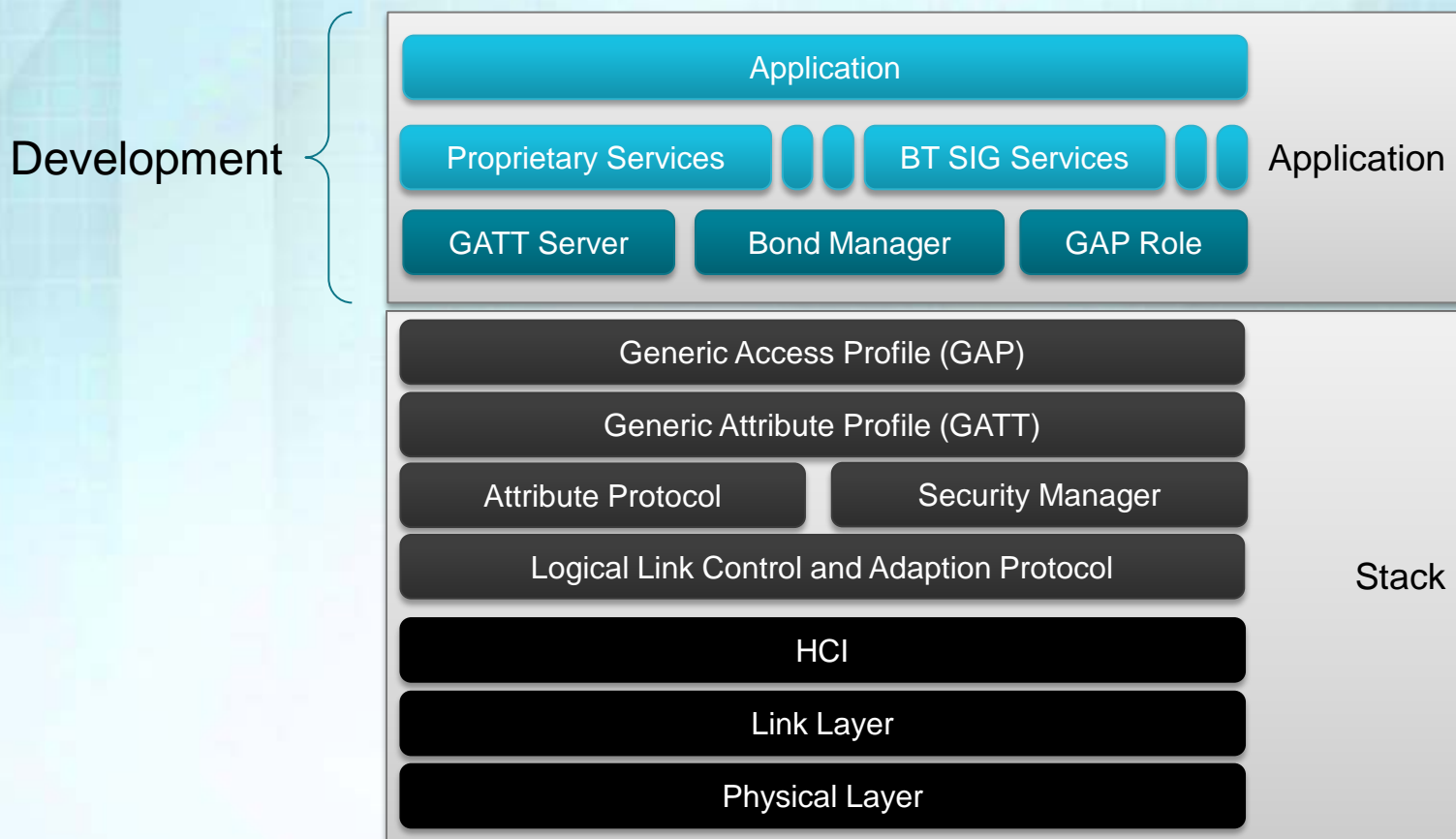
Software Platform CC2640

Royalty free from TI, ready for application development



Application and Profiles

- A **Profile** defines and describes the use of **Services** necessary to implement a given **Application**



Software tools CC2640

BTool

Run and test all possible *Bluetooth* low energy functionality controlled from the PC tool.

BLE Device Monitor

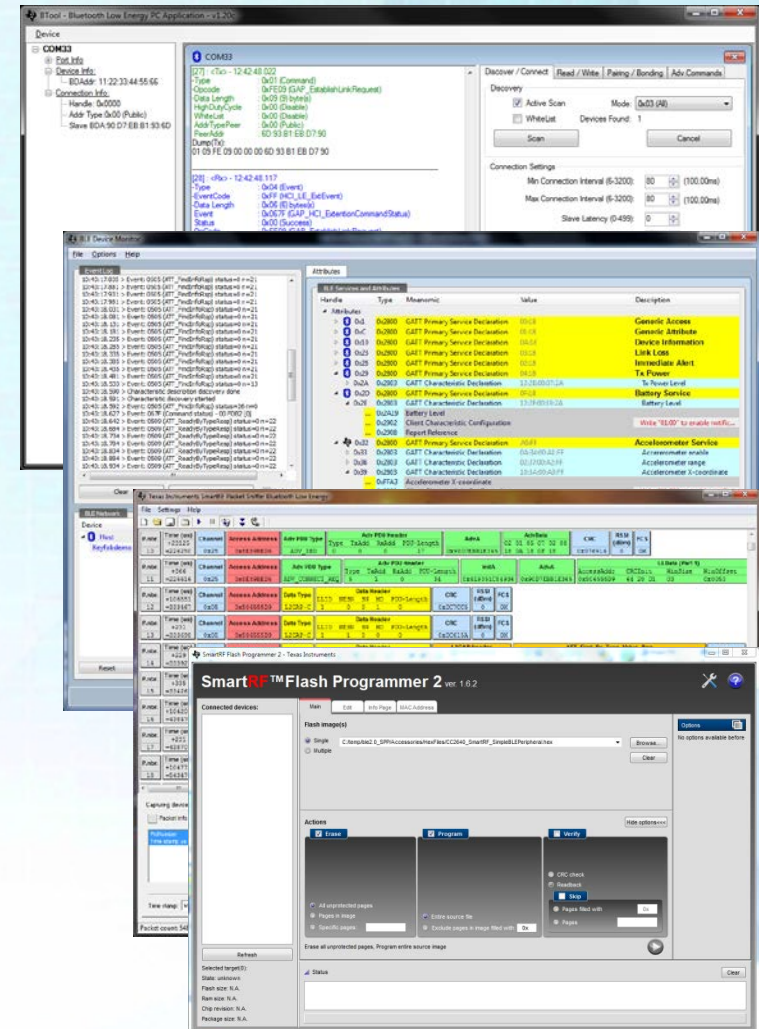
Provides an intuitive and graphical way to explore *Bluetooth* low energy Services and Characteristics.

SmartRF™ Protocol Packet Sniffer

Capture *Bluetooth* low energy communication live with full overview.

SmartRF™ Flash Programmer 2

Program devices and Read/write IEEE addresses



Software tools *Bluetooth* Developer Studio

IDE for BLE Profile & Service Definition

Use Adopted Profiles or define custom Services using 'drag & drop' GUI.

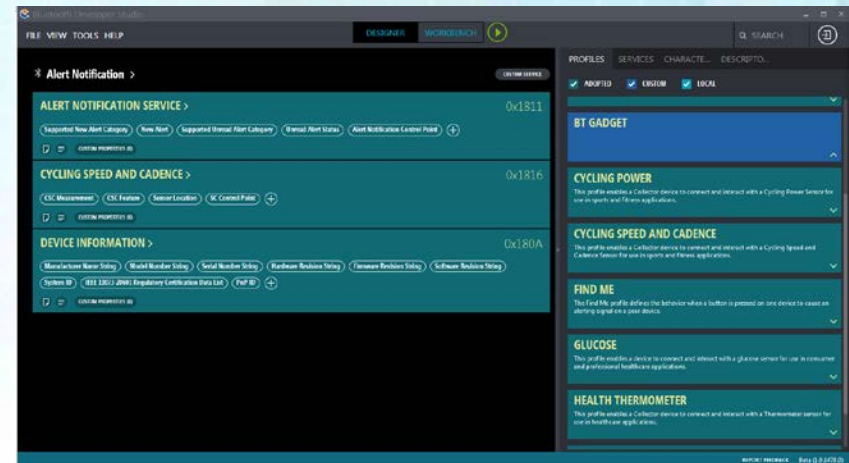


Plugin for TI BLE-Stack

Use TI's plugin to generate Profile code for BLE Stack V2.1. Application code to integrate BDS generated code is also generated.

Reduces BLE Development Time

Design & test your Profile / Service in BDS. Add code to your BLE project.



Currently in Limited Beta (Sept 2015)

Public Beta. Register at bluetooth.com

Training SimpleLink Academy

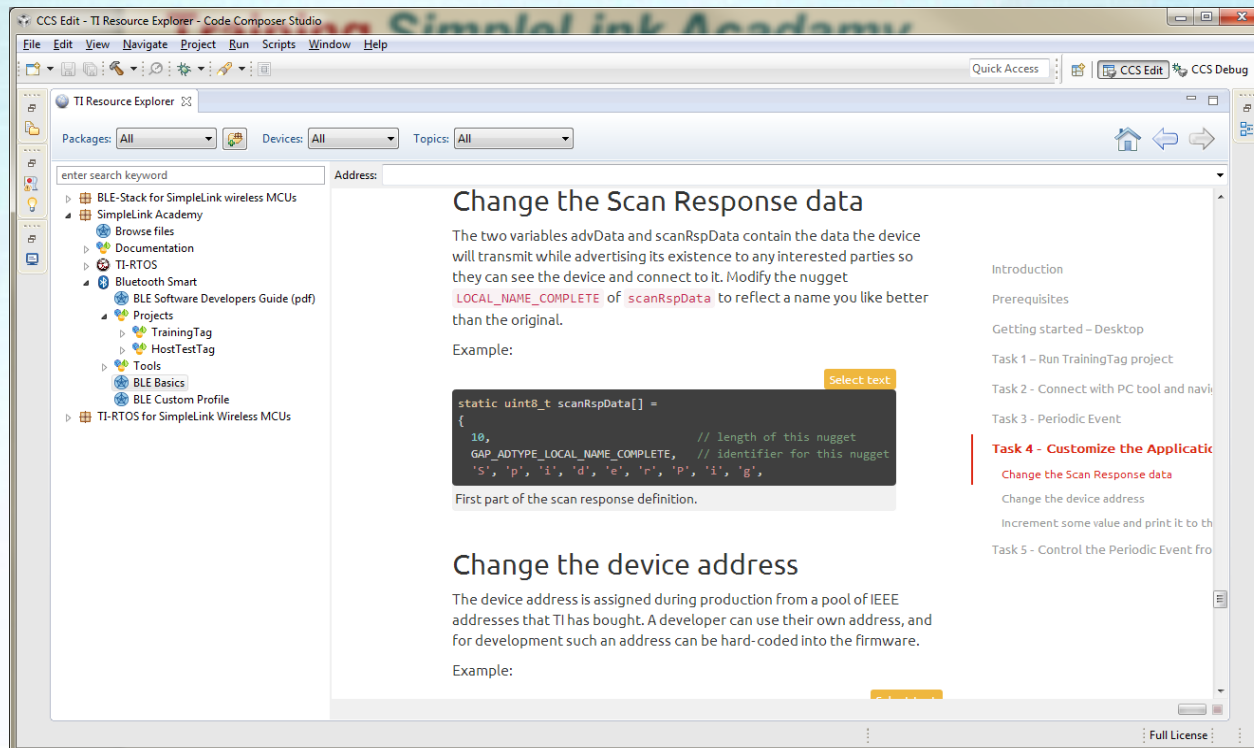
Self study, hands on program

- TI-RTOS Basics
- *Bluetooth* Smart Basics
- *Bluetooth* Smart Custom Profile
- Uses latest BLE-Stack V2.1

Based on CCS & SensorTag

Uses CC2650 SensorTag with DevPack Debugger and free CCS development environment.

Easiest way to learn BLE!



Software Walkthrough

BLEv2.1 SDK using CCS and IAR Embedded Workbench for ARM

BLEv2 *Directory*

Documentation

- CC2640 Software Developers Guide
- TI BLE Vendor Specific HCI Guide
- OAD & Network Processor User Guides

Accessories

- Boundary Tool
- Btool
- HexFiles
 - HostTestRelease
 - Simple Network Processor
 - SimpleBLEPeripheral
 - SimpleBLECentral
 - SensorTag

Projects/ble

Components

- applib
- ble
- bsp
- hal
- icall
- npi
- osal
- services
- ti-rtos

- BloodPressure
- common
- config
- CyclingSensor
- GlucoseCollector
- GlucoseSensor
- HeartRate
- HIDEmuKbd
- HostTest
- ICall
- Include
- KeyFob
- Libraries
- OADTarget
- Profiles
- ProximityTag
- RunningSensor
- SensorTag
- SimpleBLEBroadcaster
- SimpleBLECentral
- SimpleBLEObserver
- SimpleBLEPeripheral
- Thermometer
- TimeApp
- util

IAR
CCS

SimpleBLEPeripheral *Introduction*

- Example project for a BLE **Peripheral** Role Device
- What is a BLE Peripheral Device?
 - An advertiser that is connectable
 - Operates as a **slave** in a connection
 - *Example: Heart Rate Sensor*
- Implements a GATT Server
 - Makes it a Bluetooth SMART candidate
 - Both Bluetooth SIG Qualified and Proprietary Profiles
- Excellent for a base product when implementing a BLE device with a peripheral role purpose.



SimpleBLEPeripheral Walkthrough

- Project Workspace in CCS and IAR Embedded Workbench

The diagram illustrates the project workspace structure in CCS and IAR Embedded Workbench. It shows two main views: the Project Explorer on the left and the Workspace on the right.

Project Explorer (Left): Displays the project structure for the SimpleBLEPeripheral project. The project is divided into two main sections:

- SimpleBLEPeripheral [Active - FlashROM]:** This section contains the following folders:
 - Binaries
 - Includes
 - Application
 - Drivers
 - FlashROM
 - ICall
 - ICallBLE
 - Include
 - PROFILES
 - Startup
 - targetConfigs
 - TOOLS
- SimpleBLEPeripheralStack:** This section contains the following folders:
 - Binaries
 - Includes
 - FlashROM
 - HAL
 - ICallBLE
 - INCLUDE
 - LIB
 - OSAL
 - PROFILES
 - Startup
 - targetConfigs
 - TOOLS

Workspace (Right): Displays the project structure for the CC2650App - FlashROM project. The project is divided into two main sections:

- SimpleBLEPeripheral:** This section contains the following folders:
 - Application
 - Drivers
 - ICall
 - ICallBLE
 - PROFILES
 - Startup
 - TOOLS
 - Output
- CC2650Stack - FlashROM:** This section contains the following folders:
 - HAL
 - ICallBLE
 - INCLUDE
 - LIB
 - OSAL
 - PROFILES
 - Startup
 - TOOLS
 - Output

Annotations:

- A blue bracket on the left side of the Project Explorer groups the SimpleBLEPeripheral and SimpleBLEPeripheralStack sections under the label **Application Profiles RTOS**.
- A blue bracket on the right side of the Project Explorer groups the SimpleBLEPeripheralStack section under the label **BLE Stack**.
- A red arrow points from the **SimpleBLEPeripheralStack** section to the **CC2640App - FlashROM** dropdown menu in the Workspace, with the label **Switch between App/Stack**.

SimpleBLEPeripheral CC2640App Startup

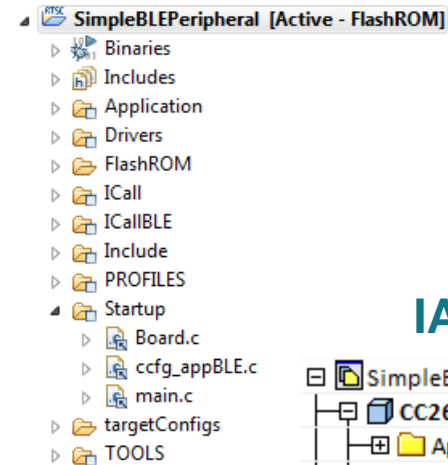
Pre-configuration

- bleUserConfig.h
 - RF Frontend Option
 - (PM Startup Margin)
 - Maximum Connections

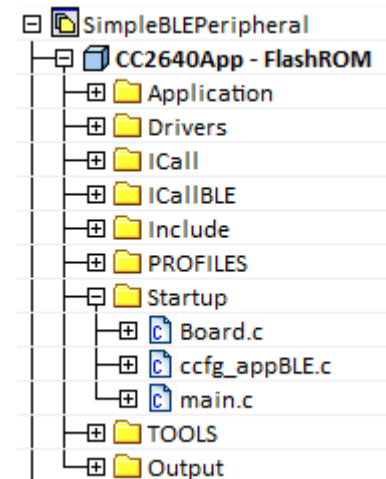
Startup

- Board.c
- ccfg_appBLE.c
 - Customer Configuration (**CCFG**).
 - DCDC/Clock options etc.
 - Designed for CC2650EM.
 - May require customization
- Main.c

CCS



IAR

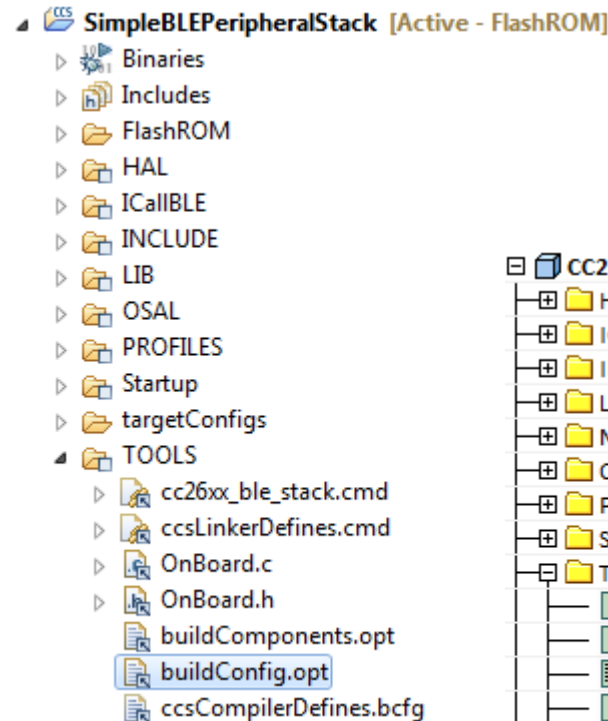


SimpleBLEPeripheral CC2640Stack

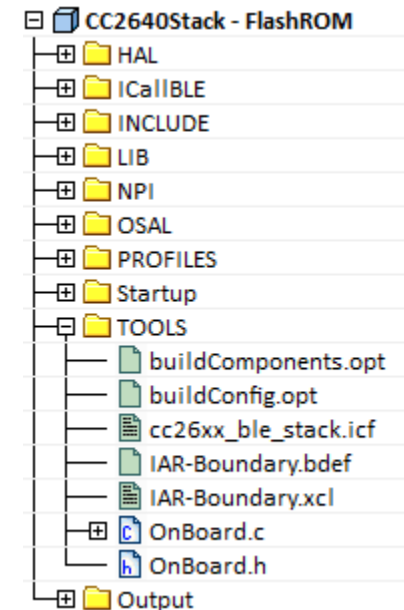
Pre-configuration

- buildConfig.opt
 - GAP Role Support
 - Privacy Features
 - L2CAP COC
 - PTM
 - BT4.1 Features
- Critical for memory optimization

CCS



IAR



SimpleBLEPeripheral *Main*

- Code start-up in main.c

```
Void main()
{
    PIN_init(BoardGpioInitTable);

    VIMSConfigure(VIMS_BASE, TRUE, TRUE);
    VIMSModeSet(VIMS_BASE, VIMS_MODE_ENABLED);

#ifdef POWER_SAVING
    Power_setConstraint(Power_SB_DISALLOW);
    Power_setConstraint(Power_IDLE_PD_DISALLOW);
#endif

    ICall_init();

    ICall_createRemoteTasks();

    GAPRole_createTask();

    SimpleBLEPeripheral_createTask();

    BIOS_start();
}
```

} Initiate Board Configuration

} Enable Cache

} Enable / Disable Power Management
Done with pre-processor POWER_SAVING

→ Initiate the ICall module

→ Create Tasks for BLE Stack (Priority 5)

→ Create Tasks for GAP Role (Priority 3)

→ Create Tasks for Application (Priority 1)

→ Call extern *BIOS_start*

- Enables interrupts
- Starts SYS/BIOS
- Starts the Task handler

SimpleBLEPeripheral *Main*

- SimpleBLEPeripheral_createTask

```
void SimpleBLEPeripheral_createTask(void)
{
    Task_Params taskParams;

    Task_Params_init(&taskParams);
    taskParams.stack = sbpTaskStack;
    taskParams.stackSize = SBP_TASK_STACK_SIZE;
    taskParams.priority = SPB_TASK_PRIORITY;

    Task_construct(&sbpTask, SimpleBLEPeripheral_taskFxn, &taskParams, NULL);
}
```

} Configure Application Task

SimpleBLEPeripheral *Initialization*

```
void SimpleBLEPeripheral_taskFxn(UArg a0, UArg a1)
{
    SimpleBLEPeripheral_init();

    for (;;) { /*Application loop*/ }
}
```

→ Initialize application and profile

- SimpleBLEPeripheral_init

```
static void SimpleBLEPeripheral_init(void)
{
    ICall_registerApp(&selfEntity, &sem)

    uint8_t initialAdvertEnable = TRUE;
    GAPRole_SetParameter(GAPROLE_ADVERT_ENABLED,
        sizeof(uint8_t), &initialAdvertEnable);
    //..
    //Setup the Bond Manager...
    //Register Services and their callbacks...
    //..
```

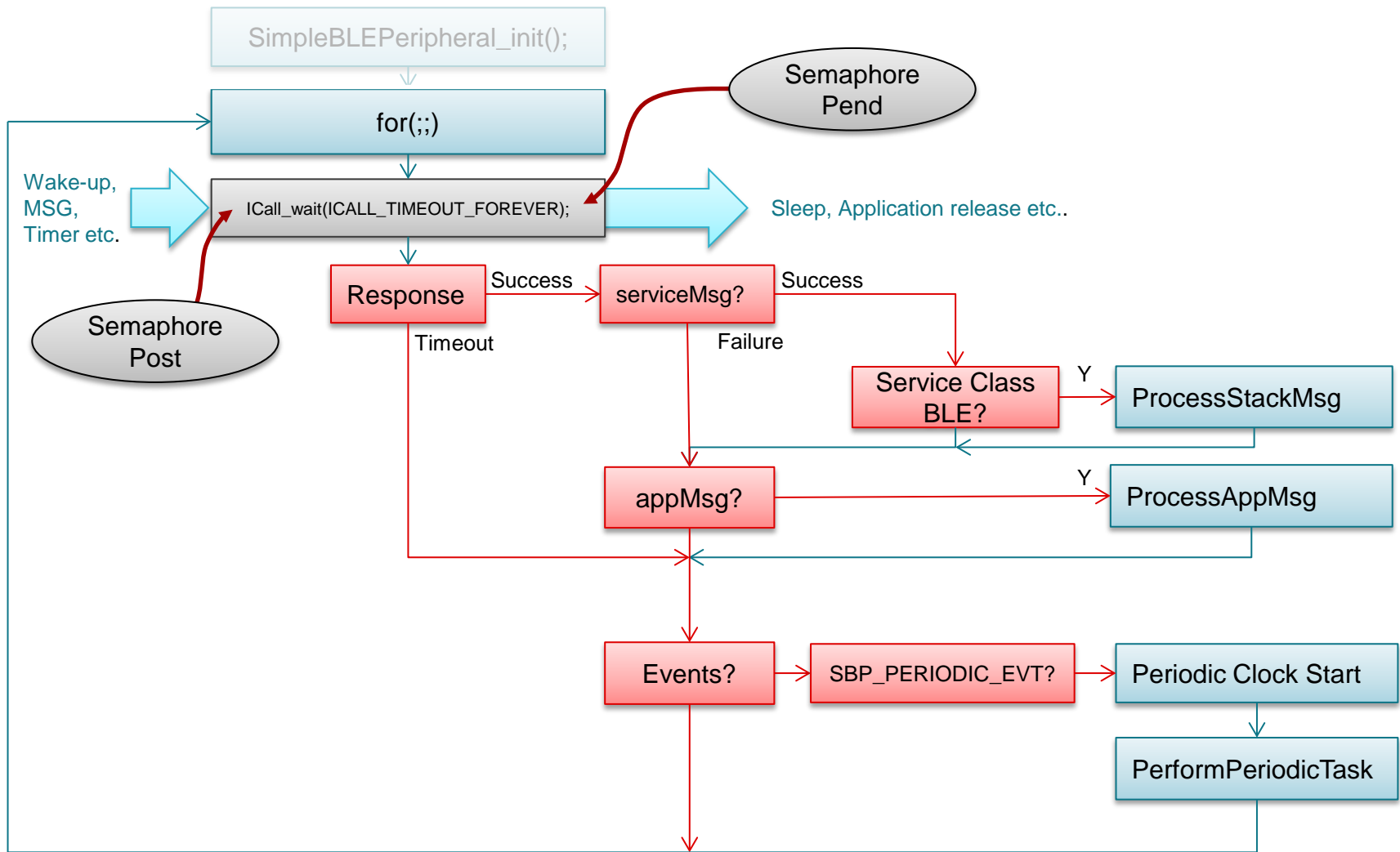
→ ICall registration of the application

→ Setup GAP Parameters, Ex. Advertise enable

GAPRole_StartDevice(&SimpleBLEPeripheral_gapRoleCBs) → Start Device, GAP_DeviceInit is called

GAPBondMgr_Register(&simpleBLEPeripheral_BondMgrCBs) → Start bond manager

SimpleBLEPeripheral *Application Loop*



SimpleBLEPeripheral *Event Handler*

- Events

- **SimpleBLEPeripheral_processStateChangeEvt()**
 - Contains actions for when GAP State has been changed, example
 - Device have started to advertise
 - A connection has been made
- **SimpleBLEPeripheral_processCharValueChangeEvt()**
 - Contains actions for when a service characteristic has been changed.
 - This function is specifically made for the SimpleGATTProfile.

- Timer Event

- **SimpleBLEPeripheral_performPeriodicTask()**
 - Is run when the *timeOut* value equals the user defined value or if an actual timeout have occurred.

SimpleBLEPeripheral *SimpleGATT Service*

- BTool Provides a over the clear overview of the GATT Server
- The last Service is the SimpleGATTService

0x0000	0x001F	0x2800	GATT Primary Service Declaration			
0x0000	0x0020	0x2803	GATT Characteristic Declaration	0A:21:00:F1:FF		
0x0000	0x0021	0xFFF1	Simple Profile Char 1	01		Rd Wr 0x0A
0x0000	0x0022	0x2901	Characteristic User Description	Characteristic 1		
0x0000	0x0023	0x2803	GATT Characteristic Declaration	02:24:00:F2:FF		
0x0000	0x0024	0xFFF2	Simple Profile Char 2	02		Rd 0x02
0x0000	0x0025	0x2901	Characteristic User Description			
0x0000	0x0026	0x2803	GATT Characteristic Declaration	08:27:00:F3:FF		
0x0000	0x0027	0xFFF3	Simple Profile Char 3			Wr 0x08
0x0000	0x0028	0x2901	Characteristic User Description	Characteristic 3		
0x0000	0x0029	0x2803	GATT Characteristic Declaration	10:2A:00:F4:FF		
0x0000	0x002A	0xFFF4	Simple Profile Char 4			Nfy 0x10
0x0000	0x002B	0x2902	Client Characteristic Configuration			
0x0000	0x002C	0x2901	Characteristic User Description	Characteristic 4		
0x0000	0x002D	0x2803	GATT Characteristic Declaration	02:2E:00:F5:FF		
0x0000	0x002E	0xFFF5	Simple Profile Char 5			Rd 0x02
0x0000	0x002F	0x2901	Characteristic User Description	Characteristic 5		

SimpleBLEPeripheral *Software Timers*

- Example «Periodic Task»
- Initialization

```
static Clock_Struct periodicClock;
```



Clock instances for internal periodic events.

```
Util_constructClock(&periodicClock,  
                   SimpleBLEPeripheral_clockHandler,  
                   SBP_PERIODIC_EVT_PERIOD,  
                   0,  
                   false,  
                   SBP_PERIODIC_EVT);
```



Callback when timer runs out



Value 1000 => 1 second



0 means one-shot timer



Do not start immediately



Event flag to set when timer runs out

- Start Timer

```
Util_startClock(&periodicClock);
```

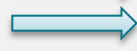
- SimpleBLEPeripheral_clockHandler

```
events |= arg;
```

```
Semaphore_post(sem);
```

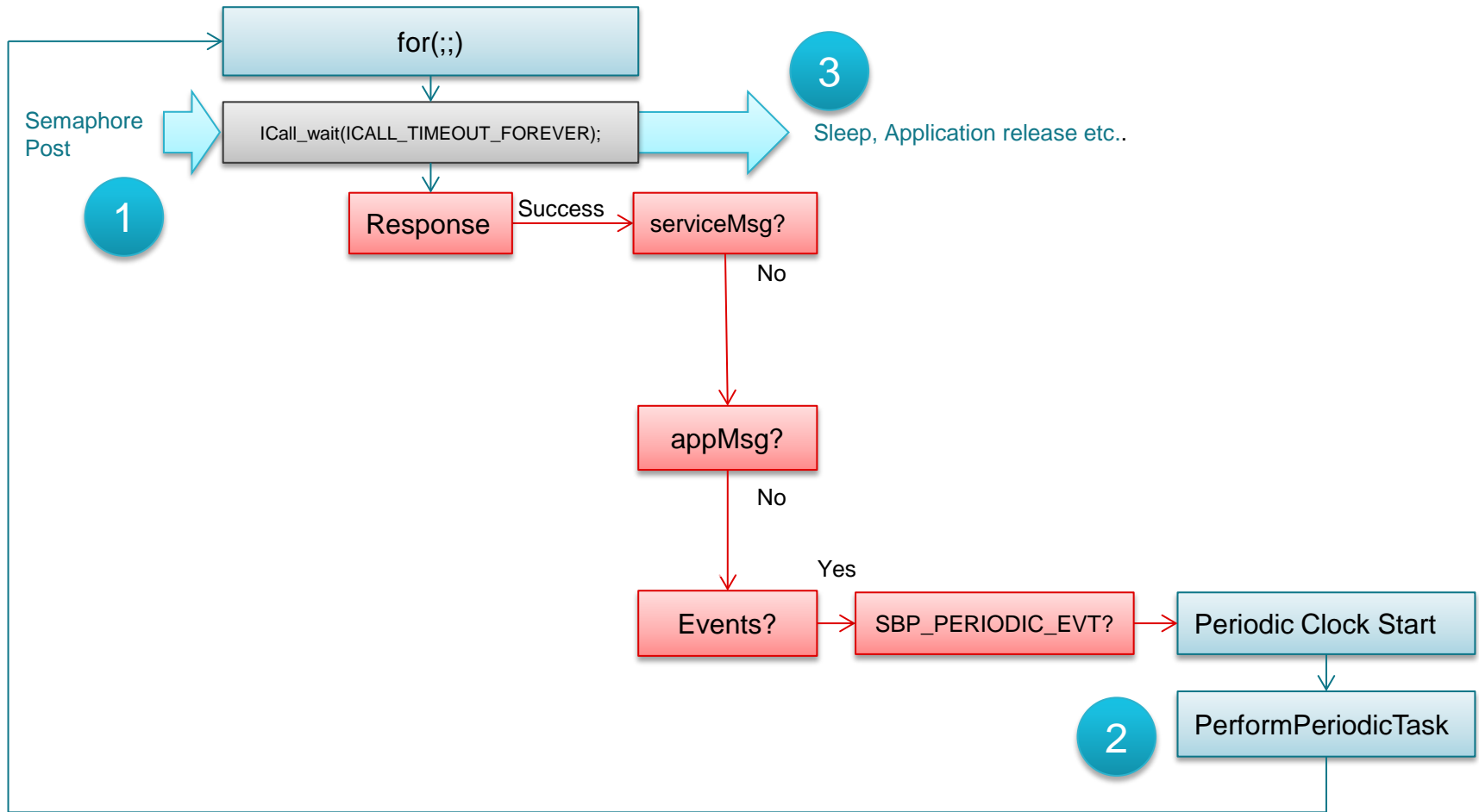


Store the event (SBP_PERIODIC_EVT)



Wake up the application

SimpleBLEPeripheral *Periodic Event*



CC2650 SensorTag

Introduction



CC2650 SensorTag *IoT Made Easy*

- \$29 Complete IoT development kit
- Access Sensor data in the cloud in 3 minutes
- 9 Low Power Sensors
 - 1 year battery life
- Expandable with DevPacks
 - Lowest cost \$15 debugger
 - Watch/Display
 - LED/Audio
 - Interchangeable between SensorTags
 - Build your own!
- Complete designs at www.ti.com/TIDesign
 - Including 3D files
 - Print your own SensorTag



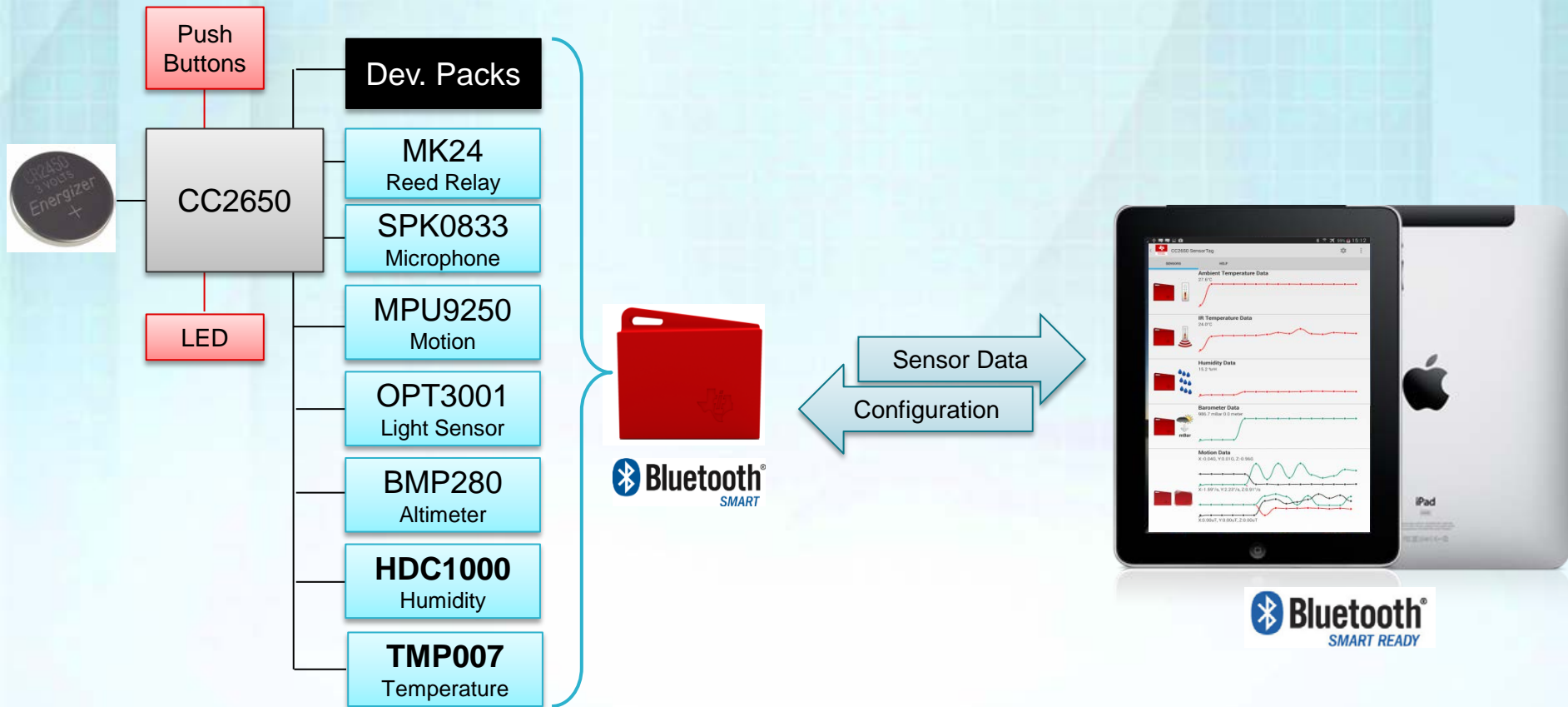
CC2650STK Overview

Peripheral Device

- Slave + Server (has data)

Central Device

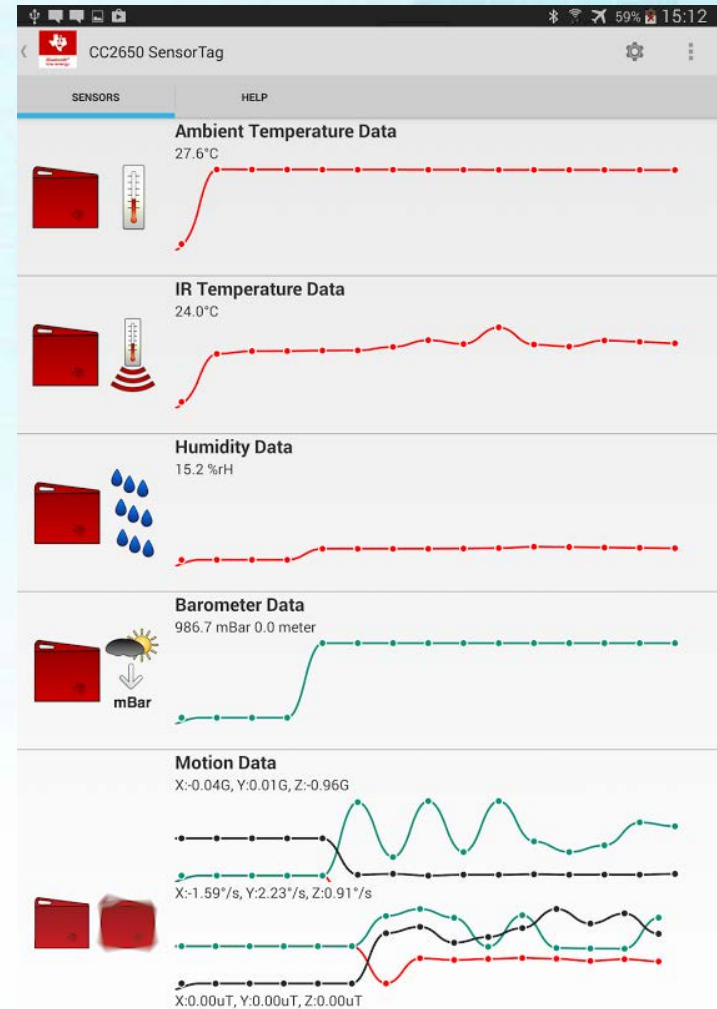
- Master + Client (wants data)



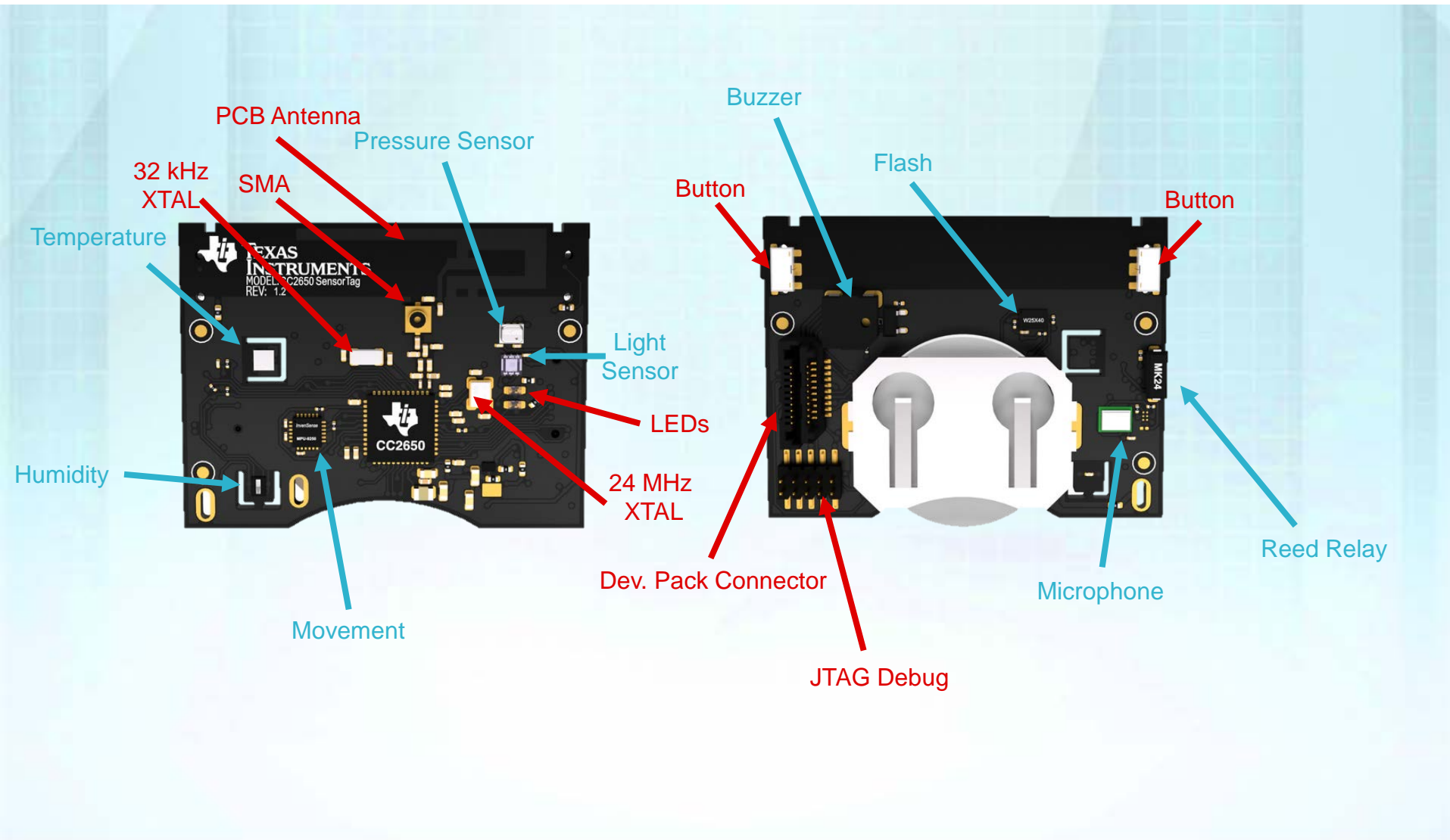
CC2650STK *Smartphone Applications*

TI SensorTag

- iOS (Apple App Store)
- Android (Google Play)

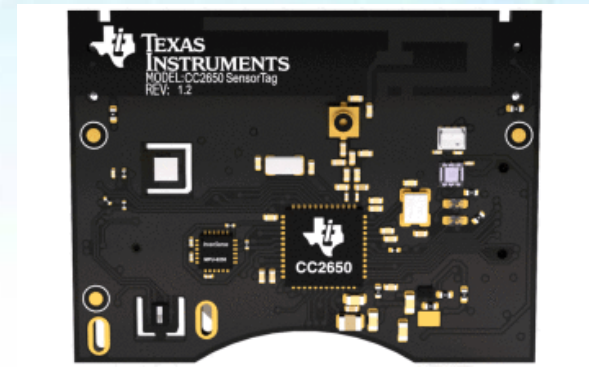
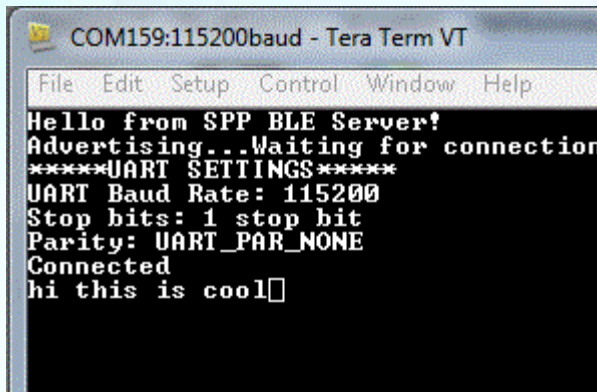


CC2650STK Rev 1.2 *Teardown*



SensorTag Example: *UART to BLE Bridge*

- Complete TI Reference Design implementing a UART to BLE Bridge on the SensorTag platform
- Software included!
- Serial interface via DevPack Debugger USB or build your own RS-232 DevPack with included design files
- Many applications: cable replacement, industrial, etc.
- <http://www.ti.com/tool/TIDC-SPPBLE-SW-RD>



CC2650STK Schedule

February 25th

Protocol Support

- Bluetooth Smart
- ZigBee End Device

iOS/Android App

- Sensor Interaction
- IBM Cloud (Quickstart) – iOS only
- Source Code
 - iOS (Limited)
 - Android (Full, Github)

Device Monitor

- Production Test
- GPIO control

Dev. Pack

- Debugger (\$15)
 - XDS110
 - GPIO Breakout
 - Grove Connector Pads



April

Marketing Campaign

- Launch Page (ti.com/sensorTag)
- Bluetooth Smart SensorTag
- WiFi SensorTag (CC3200)

iOS/Android App

- OAD

Dev. Pack

- LCD (~\$20)
- LED/Audio (>\$20)



Collateral

- Cadence PCB Designs
- 3D Files (.step/.easm, eDrawings)

Dev. Platforms

- IAR Embedded Workbench
- CCS Cloud
- CCS

Sensor Controller Studio

- I2C with Light Sensor example

Future

Enhanced IoT Support

- Energia
- More Cloud Partners
- Node Red

Protocol Support

- 6LowPAN

Use External Flash

- Sensor data storage
- Boot Between protocols

Community Support

- Dev. Packs
- Sensor functions
 - Pedometer Etc.

Microphone SW Support

- RTOS PDM Driver
- Audio transfer

CC2650STK Support

www.ti.com/sensortag - Design Resources under «Teardown» tab

TEXAS INSTRUMENTS Everything Search Login / register


Products Applications & designs Tools & software Support & community Sample & buy About TI Cart English myTI

TI Home > Wireless Connectivity > Sensortag

IoT made easy

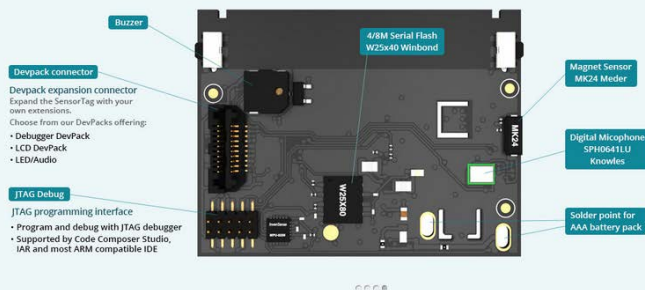
By 2020 there will be 50 billion IoT devices - Get yours today!

The SensorTag allows quick and easy prototyping of IoT devices. It simply works - Connect your sensor solution to the Cloud in 3 minutes. Get started with Wi-Fi, Bluetooth smart, or 6LoWPAN development for only \$29.



The SensorTag Story Getting Started **Teardown** DevPacks Apps and Partners Community & Buzz

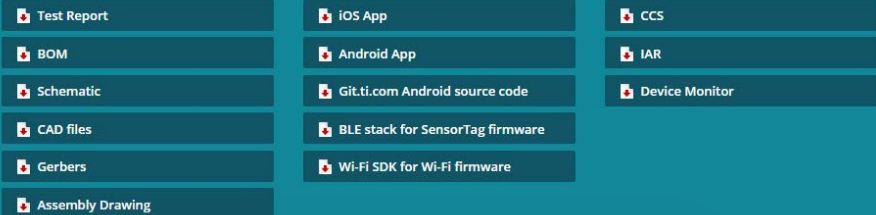
Teardown



SensorTag for:



Bluetooth Smart SensorTag



Compare SensorTags

	Wi-Fi	Bluetooth Smart	6LoWPAN
Price	\$29	\$29	\$29
Battery type	AAA	Coin Cell	Coin Cell
Connects to Internet	Wi-Fi	Smartphone	Low cost gateway
Beacon support	✓	✓	✓
DevPack support	✓	✓	✓
Mesh network	✓	✓	✓
Range	100m/300ft	50m/150ft	100m/300ft (extended with mesh network)
Max number of devices	64	8	200
Battery lifetime*	1 year (15 second report interval)	1 year (1 second report interval)	1 year (1 second report interval)
User Interface	App, Web	App	App, Web

CC2640: Reduce your time to market

Easy to design with: Do your final prototype within 10 days (checkout the wiki)

Get Started

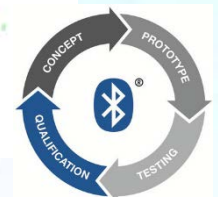
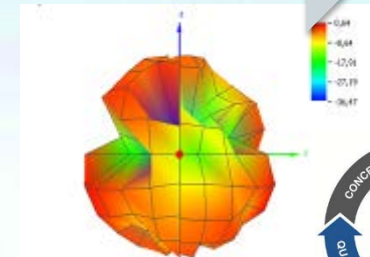
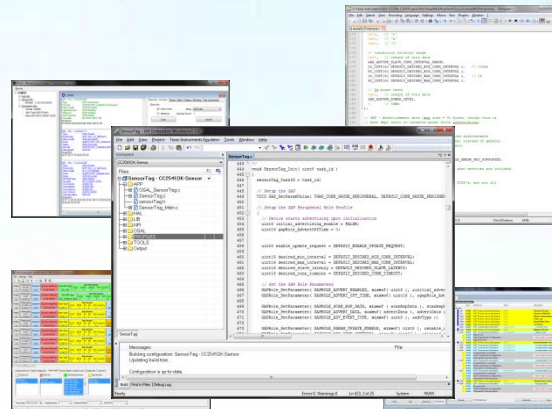
- Order Kit
- Download SDK
- Evaluate & Learn *Bluetooth* low energy

Develop

- Define Application
- Choose/Design Profiles
- Implement Prototype

Test & Release

- FCC/ETSI Certification
- *Bluetooth* Listing



Extensive Online Knowledge Base & E2E Support – ti.com/ble-forum

Next Steps

Free Samples: <http://www.ti.com/product/CC2640/samplebuy>

Just Choose the package:

CC2640F128RSMR

https://store.ti.com/AddToCart_TI.aspx?f=CC2640F128RSMR

CC2640F128RHBR

https://store.ti.com/AddToCart_TI.aspx?f=CC2640F128RHBR

CC2640F128RGZR

https://store.ti.com/AddToCart_TI.aspx?f=CC2640F128RGZR

Suggested Development Boards

\$29 SensorTag (CC2650STK) with \$15 DevPack Debugger (CC-DEVPACK-DEBUG)

<https://store.ti.com/cc2650stk.aspx> with below

<https://store.ti.com/cc-devpack-debug.aspx>

\$299 CC2650DK

<https://store.ti.com/cc2650dk.aspx>

New SaBLE-x™ Bluetooth® Smart Module featuring the TI CC2640

Delivering all the RF performance and power benefits of the TI SimpleLink™ CC2640:

- 5 dBm output power, -96 dBm sensitivity

LSR's Developer Tool Suite provides simplification and efficiency for the developer:

- **Serial-to-BLE** serial command set API library
- **PC-as-Host** test and scripting tools

Unmatched breadth of certification coverage:

- FCC/ IC / CE / Giteki / C-Tick (pending)
- Multiple certified antenna options
 - On-board Trace, Dipole, FlexPIFA™, and FlexNotch™
 - In-house certification of additional antennas at little or no cost to customer

The “LSR Difference” in RF Modules:

- Established module manufacturer with over 30 years in wireless product development
- US-based technical support staffed by LSR design engineers
- Offer flexible business models to best meet customer need:
 - Off the shelf module, custom module, integrated designs, or license design files



Small footprint:
11.6 mm x 17.9 mm

- Pricing: \$6.99 MSRP at 10k Qty
- In full production
- Dev Kits and Samples available now

Backup

What is Bluetooth low energy

What is *Bluetooth* low energy

- Introduced in *Bluetooth* 4.0

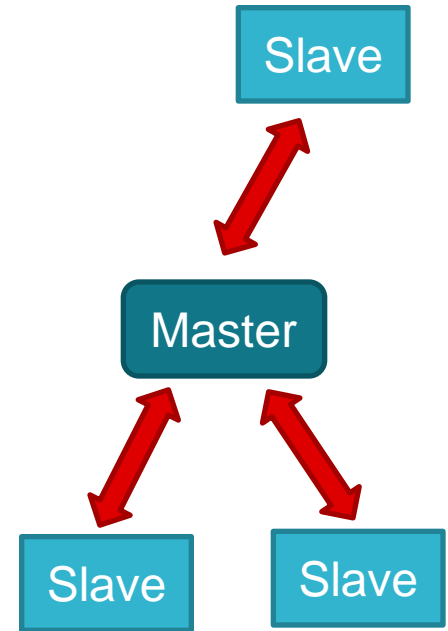


- Wireless Personal Area Network technology

- Target Applications:

- Low Power
- Low Latency
- Low Throughput

- Devices supporting only *Bluetooth* low energy branded as “*Bluetooth* Smart”.



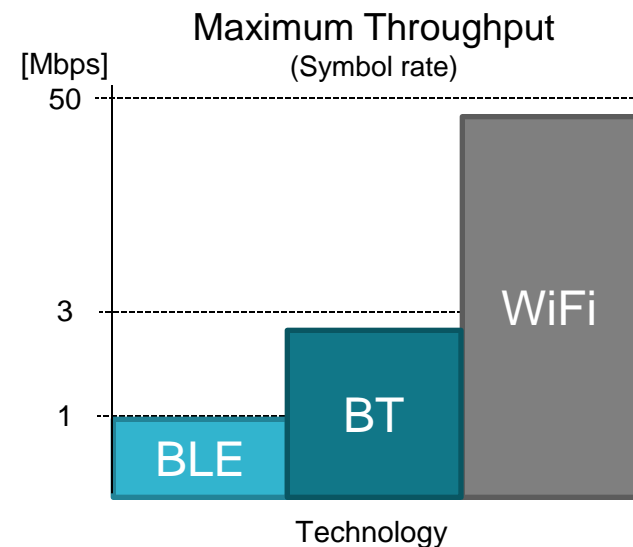
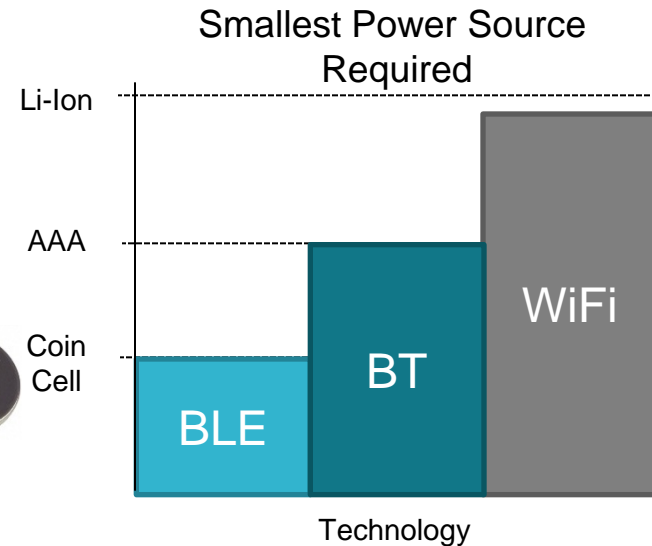
Bluetooth low energy vs WiFi/BT

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

1

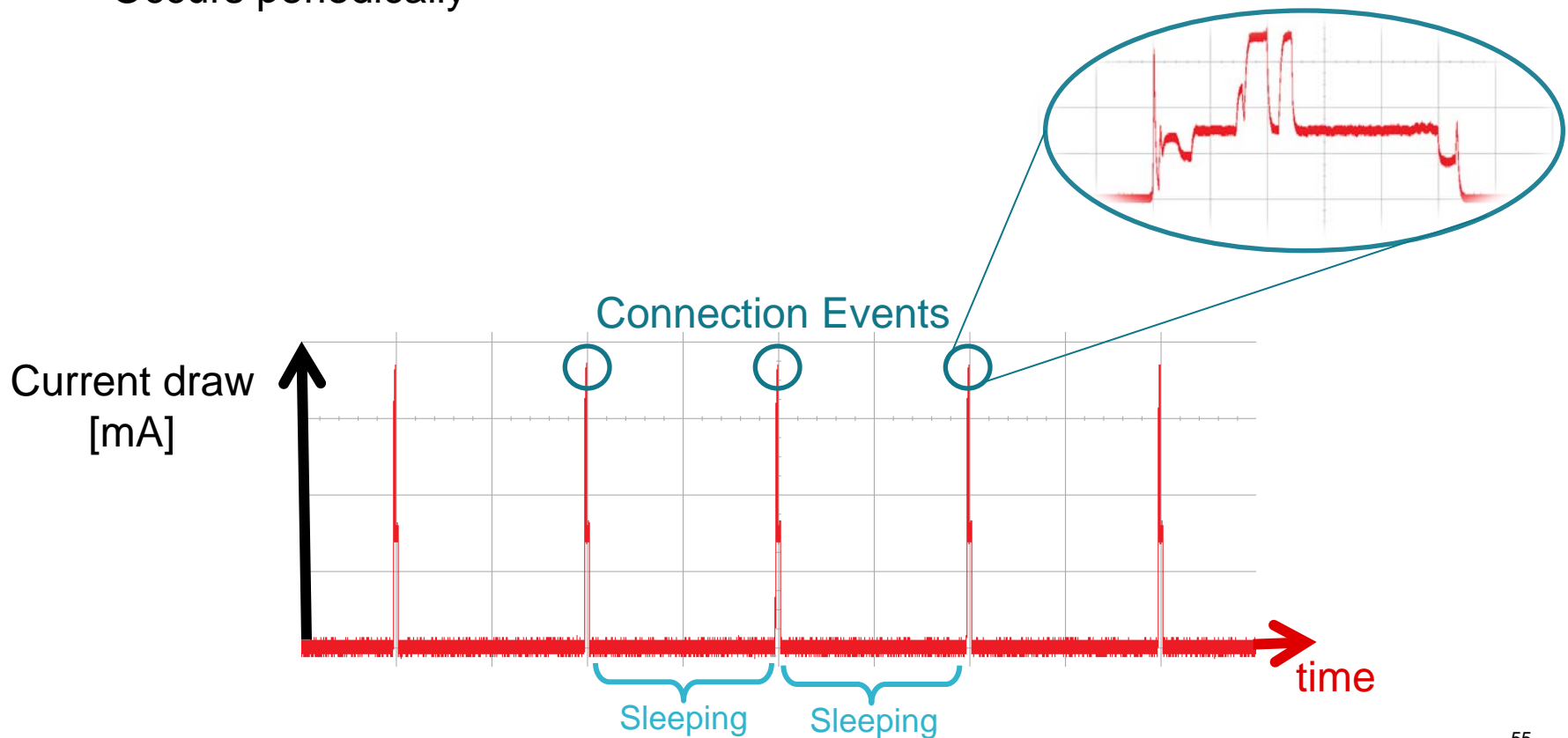
6

11



Connection Events

- All communication between two connected devices occur during **Connection Events**
 - Occur even when one (or both) sides have no data to send
 - Occurs periodically



Bluetooth low energy Device Roles

- A *Bluetooth* low energy device can operate in four profile roles:

- **Peripheral**

- An advertiser that is connectable
- Operates as a **slave** in a connection
- Example: *Heart Rate Sensor*

- **Central**

- Scans for advertisements and initiates connections
- Operates as a **master** in connections.
- Example: *Smartphone*

- **Broadcaster**

- An advertiser that is non-connectable
- Example: *Temperature Sensor*

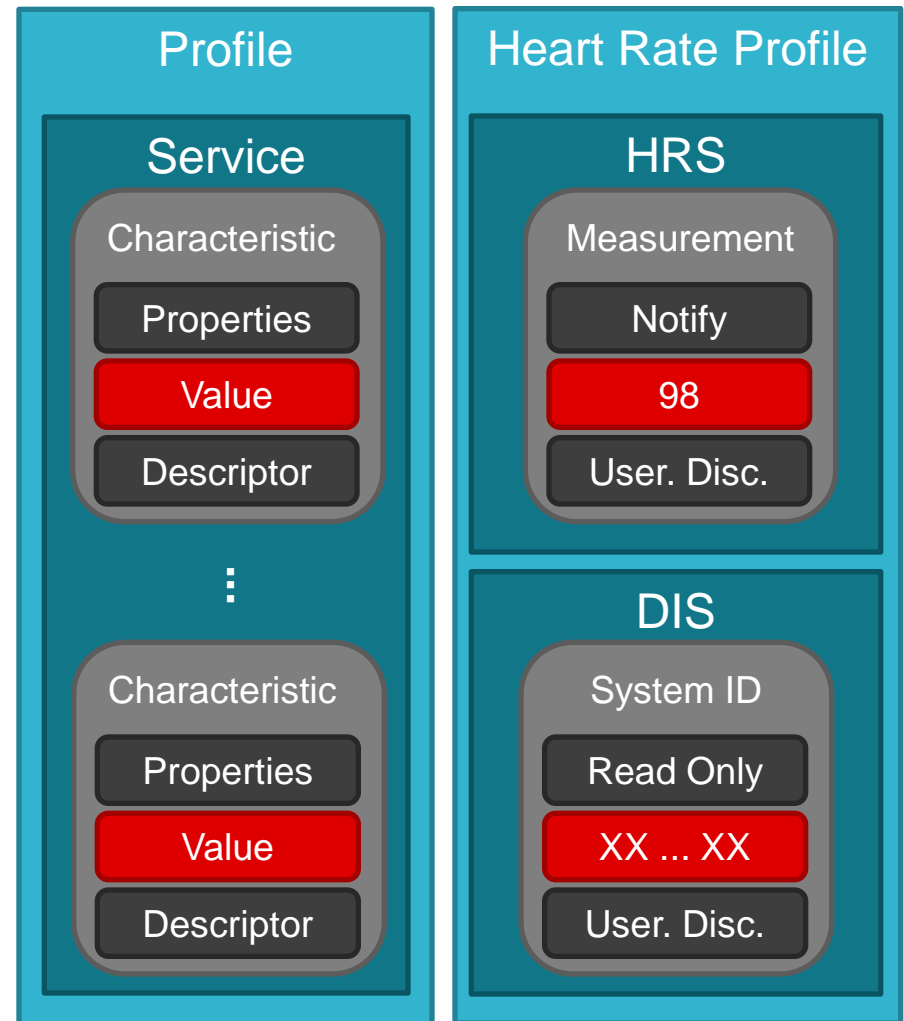
- **Observer**

- Scans for advertisements, but cannot initiate connections
- Example: *Temperature Display*



Profiles and Services

- A **Profile** defines and describes the use of **Services** necessary to implement a given **Application**
- Bluetooth SIG adopted examples:
 - Heart Rate Profile
 - Heart Rate Service (**HRS**)
 - Device Information Service (**DIS**)
 - Proximity Profile
 - Link Loss Service
 - Immediate Alert Service
 - Find Me Profile
 - Immediate Alert Service

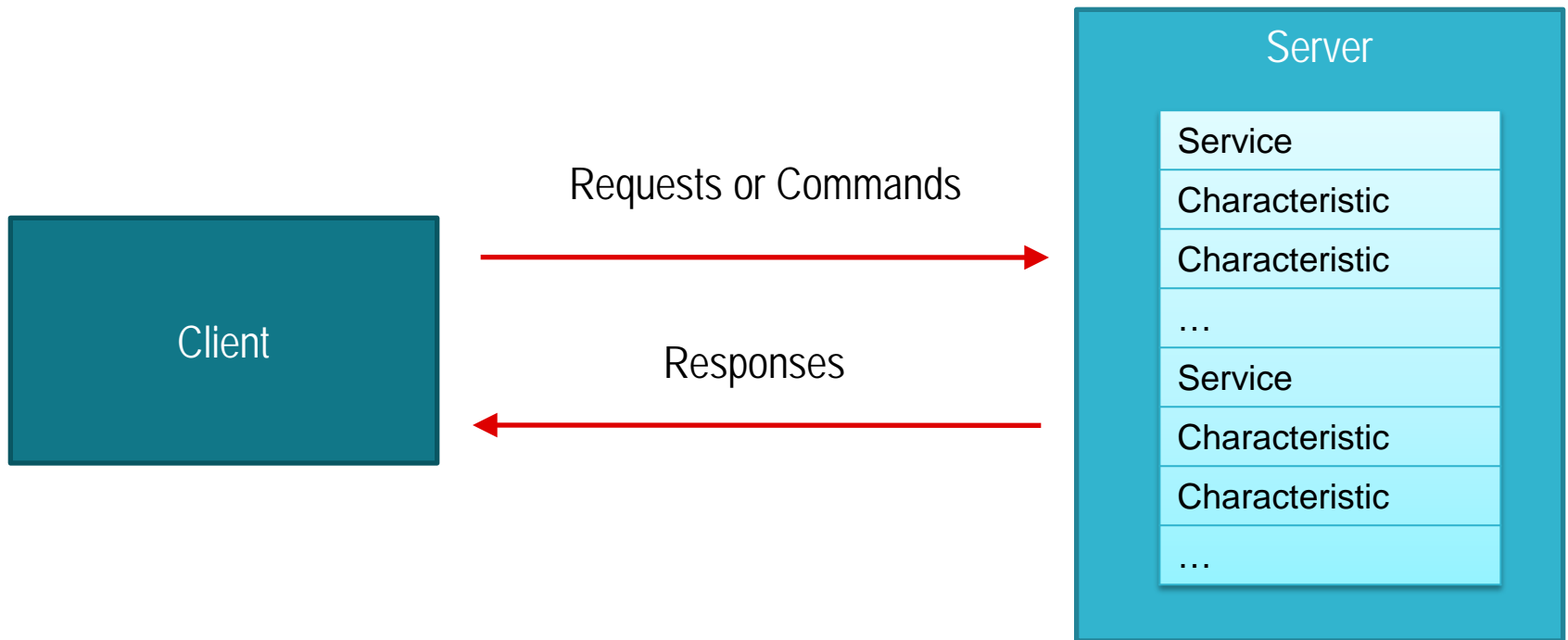


GATT = Generic Attribute Profile

57

GATT Architecture

- GATT specifies the structure in which **data** is stored and exchanged.
- Servers have **data**, which are exposed using characteristics.
- Clients want to use this **data**



Hands-on



Hands-on *Requirements*

Hardware

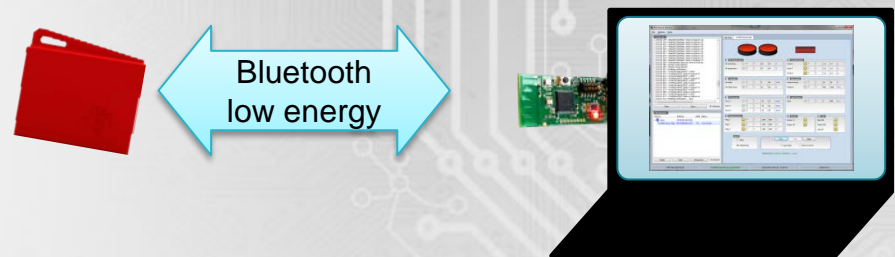
- CC2540 USB Dongle (pre-programmed with HostTest)
- SensorTag 2.0
- *iPhone (Optional)*

Software

- Device Monitor: <http://www.ti.com/lit/zip/swrc258>
- *TI SensorTag iOS App (Optional)*

System Setup 1: Connecting SensorTag to Device Monitor

- Production Test
- GATT Explorer
- OAD Update



System Setup 2: Connecting SensorTag to Smartphone

- Sensor Monitoring
- GATT Explorer
- OAD Update
- IBM Cloud Service



Hands-on *Optional Task*

Hardware

- CC2540 USB Dongle (pre-programmed with HostTest)
- SensorTag 2.0
- *Dev pack*
- *USB Cable*

Software

- CCS v6.1 or IAR Embedded Workbench
- Device Monitor: <http://www.ti.com/lit/zip/swrc258>

Steps:

- 1) Connect the Devpack Debugger (XDS110).
- 2) Connect USB between devpack and computer.
- 2) Download sensortag firmware (Stack+App) .
- 3) Change local name (Hint: scanRspData).
- 4) Add a Characteristic to an existing service.
- 5) Verify the new characteristic in Device Monitor.
- 6) Have som fun! ☺

