Introduction to Bluetooth LE

Stephen Okay
steve@inveneo.org
March 25th, 2015
Abdus Salam Int’l Center for Theoretical Physics
Trieste, Italy
A little background...

- My professional background in Systems & Networking programming and administration
- Involved with Inveneo & ICTP in ICT-related work for 10 years
- Living & working in Silicon Valley since 1997
- Largely self-taught “Technology enthusiast” (Which is a polite way of saying I’m just a big GEEK)
Bluetooth LE

- AKA Bluetooth 4.0
- AKA “Bluetooth Low Energy”, BLE, BTLE
- As opposed to “Classic Bluetooth” or “Bluetooth 2.0” or “Bluetooth BDA*”

*Battery’s Dead Again*
Why?

• Why do we need another set of wireless protocols, standards, widgets, etc.?
• What’s wrong with Bluetooth 1.x/2.x?
• Haven’t we been here before?
Let’s take a little hike back in time...
Personal/Body-Area Networks
“The future is gonna be AWESOME!!”
Then reality intruded

- Bluetooth 1.x/2.x modules were expensive ($15-35/unit in quantity in 2001, never really got much cheaper)

- Market timing/application
  - Most vendors & manufacturers ended up trying to compete w/ WiFi.
  - Devices were power-hungry, Battery technologies not mature
  - Competing & incomplete implementations of the Bluetooth spec
    - Pairing & security issues
    - Interference w/ other wireless tech. like WiFi
“But wait, there’s just one more thing...”
Technology moves with time....

- Classic Bluetooth: 1994(spec) 1999(devices)
- Battery Technology:
  - Li-Ion Sony 1991
  - LiPoly Bellcore 1996
  - “Smart Dust” sensor motes (1998)
....but also with *money*

- First-gen iPhone sold ~5 million units in the first year (2007-2008)*
- iPhone 4s first to support BLE in 2011
  - 4 million BLE-enabled phones sold in first 4 days**
  - Simplified chipset made BT/BLE commercially viable for Apple to use in the 4s


SO...

Bluetooth LE is as much about being in the right place & the right time in the market as it is about being a useful technology.
Bluetooth “Classic” Design

- “Cable replacement” technology
- RFCOMM
  - Serial emulation & modem/Dialup profile
    - PPP-over-Bluetooth AKA wireless TCP/IP
  - A2DP - Audio transmission (Headset/headphones)
  - Wireless data-syncing
- All very much designed to solve 1990s era problems.
- Sensors?—-yeah, we’ll get to that…
BLE Design

- Designed to be a low-power, low-bandwidth, wireless technology for sensors
- Spec was originally written separate of Bluetooth, then later rolled into Bluetooth 4.0
- Not meant to be another “data pipe”
- “Ask a specific question, get a specific answer”
BLE Design (cont’d)

- Specialization yields
  - A less complex protocol stack
  - Smaller packets
  - Asynchronous data transfer
  - Less energy spent running radio hardware, etc.
BLE in hardware

System-on-a-chip (SOC):

- Application
- Host
- Radio

Dual IC:

- Application
- Host
- Radio
- UART or HCI
- Dual IC

Connectivity Device:

- Application
- Host
- Radio
- custom interconnect
How low can you go?

- BLE specification defines a connection duration of 7.25ms to 4 seconds.
- 10ms transmit window
- Data packets nominally 20 bytes, up to 39 bytes in payload.
- This results in an effective throughput of 10KB/s
- 0.01-0.5W peak power consumption
  < 15mA peak current
How low can you go?

- Power consumption:
- BLE Specification:
  0.01-0.5W peak power consumption
  < 15mA peak current

- Studies by Microsoft Research* and Univ. Michigan** looked at power draw during sleep, scanning, connection negotiation, data transmission, etc.

* http://research.microsoft.com/pubs/192688/IWS%202013%20wireless%20power%20consumption.pdf
** http://www.eecs.umich.edu/courses/eecs589/papers/06215496.pdf
Comparing power consumption between BLE, ZigBee and ANT

<table>
<thead>
<tr>
<th></th>
<th>BLE</th>
<th>ZigBee</th>
<th>ANT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time of one connection</td>
<td>1150 ms ±260 ms</td>
<td>250 ms ±9.1 ms</td>
<td>930 ms ±230 ms</td>
</tr>
<tr>
<td>Sleep current</td>
<td>0.78 uA</td>
<td>4.18 uA</td>
<td>3.1 uA</td>
</tr>
<tr>
<td>Awake current</td>
<td>4.5 mA</td>
<td>9.3 mA</td>
<td>2.9 mA</td>
</tr>
<tr>
<td>Min current (at 120 sec interval)</td>
<td>10.1 uA</td>
<td>15.7 uA</td>
<td>28.2 uA</td>
</tr>
<tr>
<td>Optimal sleep interval</td>
<td>10.0 s</td>
<td>14.3 s</td>
<td>15.3 s</td>
</tr>
</tbody>
</table>

*SD: standard deviation

“Power Consumption Analysis of Bluetooth Low Energy, ZigBee and ANT” - Microsoft Research
http://research.microsoft.com/pubs/192688/IWS%202013%20wireless%20power%20consumption.pdf
BLE still inherits much from Classic Bluetooth

• Roles/Topologies:
  • Support for Role-switching between Central & Peripheral, multiple simultaneous connections

• Security
  • Pairing
  • Bonding
  • Similar Encryption Algorithms
    • ….similar bugs
BLE in the Bluetooth 4.0 stack

- **BLE (single mode or BLE)**
  - GAP
  - GATT
  - SMP
  - ATT
  - L2CAP
  - Link Layer
  - LE PHY

- **Bluetooth Smart Ready (dual mode or BR/EDR/LE)**
  - SPP
  - GAP
  - GATT
  - SMP
  - ATT
  - L2CAP
  - Link Layer
  - BR/EDR + LE PHY

- **Classic or BR/EDR**
  - SPP
  - RFCOMM
  - L2CAP
  - Link Manager
  - BR/EDR PHY
Some BLE chipsets & vendors*

<table>
<thead>
<tr>
<th>Vendor</th>
<th>CC2540/CC2541</th>
<th>CC256x</th>
<th>nRF51822</th>
<th>DA14580</th>
<th>PSOC 4 BLE / PROC BLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Texas Instruments</td>
<td>Texas Instruments</td>
<td>Texas Instruments</td>
<td>Nordic Semiconductor</td>
<td>Dialog Semiconductor</td>
<td>Cypress Semiconductor</td>
</tr>
<tr>
<td>Mode</td>
<td>Single Mode v4.0</td>
<td>Dual Mode Classic + BLE/ANT</td>
<td>Single Mode v4.1 / ANT</td>
<td>Single Mode BLE v4.1</td>
<td>Single Mode BLE v4.1</td>
</tr>
<tr>
<td>Integrated Processor</td>
<td>8051</td>
<td>No - External</td>
<td>Cortex-M0</td>
<td>Cortex-M0</td>
<td>Cortex-M0</td>
</tr>
<tr>
<td>Flash</td>
<td>128kB/256kB</td>
<td>None</td>
<td>128kB / 256kB</td>
<td>32kB OTP</td>
<td>128kB</td>
</tr>
<tr>
<td>RAM</td>
<td>8kB</td>
<td>None</td>
<td>16kB / 32kB</td>
<td>42kB + 8kB</td>
<td>16kB</td>
</tr>
<tr>
<td>Current Consumption (RX/TX)</td>
<td>17.9mA / 18.2mA to 14.7mA / 14.3mA</td>
<td>-</td>
<td>9.7mA / 8mA</td>
<td>4.9mA / 4.9mA</td>
<td>15.6mA / 16.4mA</td>
</tr>
<tr>
<td>Chip Size</td>
<td>6mmx6mm QFN-40</td>
<td>8mmx8mm QFN</td>
<td>6x6mm QFN 3.5mmx3.8mm WLCSP</td>
<td>2.5mmx2.5mm CSP 6mmx6mm QFN48</td>
<td>7mmx7mm QFN 3.9mmx3.5mm WLCSP</td>
</tr>
</tbody>
</table>

BLE Chipsets

Nordic nRF 51882

PAN 1721

BlueGIGA 112

Broadcom WICed Dev Board
Platforms and Devices

- Estimote iBeacon
- TI CC2540
- Nordic nRF Developer Kit
- Adafruit “BLE Friend”
- Seeduino BLE Shield
How does this work?

scan

conn req.

advertisement

conn. response

GAP req.

GATT Service: “health thermometer"

ATT req:

“read:temperature(sublingual)”

Char. req: “current temp in C”

“write:C or F”

“38.3 C”
How does this work? (cont’d)

Wireless Comm. + Sensors + (optional) Co-processor
Mind the GAP (and GATT)

• GAP - General Access Profile
  • Defines Roles, Protocols, etc. for discovery, connection & security between BLE devices

• GATT Generic ATTRibute Profile
  • Defines Roles, Attributes, Permissions, etc. for the actual exchange of data between connected devices
GAP Device Roles

- Broadcast - Beacons, sends all data in beacon frame.
- Central - Master/Coordinator nodes
- Observer - Passive link-layer receiver
- Peripheral - Link-layer slave device, beacons for Central nodes and connectivity
GATT Profiles

• Defines/Controls data transfer between connected devices

• Applicable after discovery/connection has been established.

• Describes Services and Characteristics available on connectable devices with UUIDs
GATT Profiles

image courtesy of ktownsend@adafruit.com - CC-BY-SA
GATT Example

Blood Glucose Measurement

UUID 0x1808
org.bluetooth.service.glucose

UUID 0x2A08
org.bluetooth.characteristic.glucose_measurement

Measurement Timestamp - org.bluetooth.characteristic.date_time
Glucose Concentration - units of kg/L 16-bit SFLOAT
Glucose Concentration - units of mol 16-bit SFLOAT
Sample Location - bitfield (0-finger, 1-earlobe, 2- internal/IV....)
UUIDs

• “Universally Unique ID”

• 128-bit (16-byte) value used to represent a (very likely) unique identifier for Services and Characteristics

• Originating from ITU-T Rec. X.667(ISO/IEC 9834-8:2005)

• Used in BLE to identify: Devices, Services and Profiles
“If generated according to one of the mechanisms defined in Rec. ITU-T X.667 | ISO/IEC 9834-8, a UUID is either guaranteed to be different from all other UUIDs generated before 3603 A.D ….The UUID generation algorithm specified in this standard supports very high allocation rates: 10 million per second per machine if necessary, so UUIDs can also be used as transaction IDs…”

–ITU UUID Website
Defining Custom UUIDs

- Check again to make sure that there isn’t an existing UUID set that defines what you want.

- Visit the ITU UUID Generator site: http://www.itu.int/en/ITU-T/asn1/Pages/UUID/uuids.aspx

- Include in the libraries/headers of your application source code.
public class SampleGattAttributes {
    private static HashMap<String, String> attributes = new HashMap<>();
    public static String HEART_RATE_MEASUREMENT = "00002a37-0000-1000-8000-00805f9b34fb";
    public static String CLIENT_CHARACTERISTIC_CONFIG = "00002902-0000-1000-8000-00805f9b34fb";
    public static String TINY_BLE_SENSOR_MEASUREMENT = "195ae58a-437a-489b-b0cd-b7c9c394bae4";
    public static String TINY_BLE_SENSOR_READ_CHARACTERISTIC = "21819ab0-c937-4188-b0db-b9621e1696cd";
    public static String TINY_BLE_SENSOR_WRITE_CHARACTERISTIC = "5fc569a0-74a9-4fa4-b8b7-8354c86e45a4";

    static {
        // Sample Services.
        attributes.put("0000180d-0000-1000-8000-00805f9b34fb", "Heart Rate Service");
        attributes.put("0000180a-0000-1000-8000-00805f9b34fb", "Device Information Service");

        // Sample Characteristics.
        attributes.put(HEART_RATE_MEASUREMENT, "Heart Rate Measurement");
        attributes.put("00002a29-0000-1000-8000-00805f9b34fb", "Manufacturer Name String");
    }

    static {
        attributes.put(TINY_BLE_SENSOR_MEASUREMENT, "tinyBLE Sensor Service");
        // Sample Characteristics.
        attributes.put(TINY_BLE_SENSOR_READ_CHARACTERISTIC, "tinyBLE Sensor Read Characteristic");
        attributes.put(TINY_BLE_SENSOR_WRITE_CHARACTERISTIC, "tinyBLE Sensor Write Characteristic");
        attributes.put("00002a29-0000-1000-8000-00805f9b34fb", "Manufacturer Name String");
    }
#include "st_util.h"

#define BAROMETER_SERV_UUID 0xAA40  // F000AA40-0451-4000-B000-000000000000
#define BAROMETER_DATA_UUID 0xAA41
#define BAROMETER_CONF_UUID 0xAA42
#define BAROMETER_CALI_UUID 0xAA43
#define BAROMETER_PERI_UUID 0xAA44

#define BAROMETER_SERVICE 0x00000010
#define BAROMETER_DATA_LEN 4
#define BAROMETER_CALI_LEN 16
BLE and the IOT

• Not immediately IOT-ready

• Some research has been done to run IPV6 over BLE *

• Generally, BLE Central Role device with Internet connectivity is required.
  
  • Laptop, Raspberry Pi, Beaglebone, Mobile Phone, etc.

Platform Support

- Platform/OS
  - iOS5 and up (iPhone 4s or later)
  - Android 4.3 and up (Galaxy S4, etc.)
  - Apple OS X 10.6+
  - Windows 8
  - GNU/Linux BlueZ 4.93+
  - Arduino: Various shields & peripherals
Developer Support for BLE chipsets

- Mostly closed-source, depends on chipset & vendor
  - TI CC2540 : OSAL via Keil/IAR compilers
  - iBeacons: Obj. C using Apple Xcode
  - BlueGIGA:BGScript on Linux and Windows
  - Nordic nrf51822: C/C++ w/ GCC
  - Arduino IDE
  - Android:Android Studio (Eclipse support limited)
Looking forward...
## Projected sales of Bluetooth devices 2015-2018

millions of units shipped

<table>
<thead>
<tr>
<th>BLUETOOTH UNITS IN PHONES &amp; OTHERS</th>
<th>2015</th>
<th>2016</th>
<th>2017</th>
<th>2018</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bluetooth Technology Total</td>
<td>3,500</td>
<td>4000</td>
<td>4300</td>
<td>4600</td>
</tr>
<tr>
<td>Bluetooth Technology In Mobile Phones</td>
<td>1,800</td>
<td>1925</td>
<td>2025</td>
<td>2100</td>
</tr>
<tr>
<td>Bluetooth Technology In Other Applications</td>
<td>1,700</td>
<td>2,075</td>
<td>2,275</td>
<td>2,500</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BLUETOOTH UNITS BY VERSION</th>
<th>2015</th>
<th>2016</th>
<th>2017</th>
<th>2018</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bluetooth Classic</td>
<td>900</td>
<td>750</td>
<td>650</td>
<td>500</td>
</tr>
<tr>
<td>Bluetooth 4.0 (BLE)</td>
<td>700</td>
<td>1050</td>
<td>1100</td>
<td>1200</td>
</tr>
<tr>
<td>Bluetooth Dual Mode (BTDM)</td>
<td>1900</td>
<td>2200</td>
<td>2550</td>
<td>2900</td>
</tr>
<tr>
<td>BTLE+BTDM</td>
<td>2600</td>
<td>3250</td>
<td>3650</td>
<td>4100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PERCENTAGE SHARES</th>
<th>2015</th>
<th>2016</th>
<th>2017</th>
<th>2018</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classic Percentage</td>
<td>26%</td>
<td>19%</td>
<td>15%</td>
<td>11%</td>
</tr>
<tr>
<td>BTLE Percentage</td>
<td>20%</td>
<td>26%</td>
<td>26%</td>
<td>26%</td>
</tr>
<tr>
<td>BTDM Percentage</td>
<td>54%</td>
<td>55%</td>
<td>59%</td>
<td>63%</td>
</tr>
<tr>
<td>BTLE+BTDM Percentage</td>
<td>74%</td>
<td>81%</td>
<td>85%</td>
<td>89%</td>
</tr>
</tbody>
</table>

Market data provided courtesy of Cratus Technology, Inc. [www.cratustech.com](http://www.cratustech.com)
BLUETOOTH SHIPMENTS BY TECHNOLOGY [MU]

Market data provided courtesy of Cratus Technology, Inc. www.cratustech.com
Bluetooth Technology In Mobile Phones & Other Applications

Market data provided courtesy of Cratus Technology, Inc. www.cratustech.com
Links/Resources

• Documentation

• Bluetooth SIG BLE Site: https://developer.bluetooth.org/TechnologyOverview/Pages/BLE.aspx

• O’Reilly & Associates  “Getting Started with Bluetooth Low Energy” (Book)

• Adafruit Learning Center - Intro to BLE: https://learn.adafruit.com/introduction-to-bluetooth-low-energy/introduction
BLE Starter kits

- TI CC2540 Sensor Tag $40
BLE Starter kits

- SeeedStudio BLE kits & modules
BLE Diagnostic tools

Adafruit Bluefruit
Dedicated BLE Sniffer

Nordic 51822 chipset

Ubertooth One
BT/BLE/802.15.4 sniffer

TI CC259x
2.4Ghz chipset
Questions ?
Contact me

• Stephen Okay
  • steve@inveneo.org
  • espressobot@gmail.com