Creating Bluetooth®-based IoT Solutions with Zephyr™ OS

Johan Hedberg
Intel Corporation

Zephyr is a trademark of the Linux Foundation.
Bluetooth low energy technology

- Also known as BLE or Bluetooth Smart
- Introduced in 2010 with Bluetooth 4.0
- 2.4 GHz, slightly different radio modulation than Bluetooth Classic
- 100m range, 1Mbps bandwidth
- Years of battery life on a coin-cell battery
- Controllers come in single- & dual-mode variants
- Perfect for IoT use-cases
Zephyr OS Bluetooth Stack

- Bluetooth 4.2 compliant
- Almost complete low energy feature set
  - L2CAP Connection-oriented Channels
  - LE Secure Connection
- Bluetooth Classic (BR/EDR) on its way
- Host-controller separation through HCI
- Recently added controller implementation
Bluetooth host stack architecture

- GAP (Generic Access Profile)
  - Peripheral & Central
  - Observer & Broadcaster
- IPSP for IPv6 over Bluetooth LE
- Clean HCI driver abstraction
  - Standard physical transport drivers (e.g. UART)
  - Virtual driver for local Controller support
- Verified with multiple popular controllers
- Highly configurable
  - Features, buffer sizes/counts, etc.
Host stack runtime view

ACL Data -> rx_fiber
ACL Data -> HCI Event

HCI Event -> bt_hci_cmd_send()
HCI Event -> bt_conn_send()

rx_fiber -> bt_recv()

 HCI Event

cmd_tx_fiber -> bt_send()
conn_tx_fiber -> bt_send()
conn_tx_fiber -> bt_send()

HCI Driver
Network buffer integration

- Common network buffer API: net_buf
- Easy encoding & decoding
- Fragmentation
- (Near) zero-copy
- Compatible with kernel objects like FIFOs
- Cross-layer, e.g. to/from controller
- Cross-subsystem to/from IP stack
Configuring Bluetooth Host Features

- HCI driver
- Features
  - GAP/GATT roles
  - Security (pairing & signing)
- Define buffer sizes & counts
- Number of paired devices
- Number of connections (even 0)
- Debug options
Creating a Bluetooth LE application

- Initialize the stack
  - `bt_init()`
- Register GATT service database
  - `bt_gatt_register(services)`
- Advertise and let others connect
  - `bt_le_adv_start(parameters)`
- Notify of value changes
  - `bt_gatt_notify(parameters)`
- Many samples available
  - `samples/bluetooth/*`
Development tools

- QEMU* support
  - Integration with BlueZ on the Linux* host
  - HCI tracing
  - GDB
- Real devices
  - Bluetooth Monitor Protocol over console UART
  - Interleaved log messages & HCI data
  - Decoded using btmon from BlueZ
- Follow-up presentation coming about Zephyr OS-BlueZ joint usage

*Other names and brands may be claimed as the property of others.
LE Controller implementation

- Contributed by Nordic Semiconductor
- On its way to Zephyr 1.6
- LE Link Layer
- As many instances of connected LE roles as RAM & configuration permit
- nRF5x radios supported
- Ongoing radio abstraction work
- HCI for internal interface with host
Possible configuration options

**Controller-only**
- UART/SPI/USB
- Raw HCI API
- Controller

**Host-only**
- Bluetooth app
- Host stack
- HCI Driver

**Combined Host & Controller**
- Bluetooth app
- Host stack
- Controller
Possible configuration options - details

Controller-only
- Raw HCI API
  - UART, USB, SPI
- Arduino 101* (nRF51)
- Carbon (nRF51)

Host-only
- Bluetooth API
- HCI transport driver
  - UART, SPI
- Arduino 101 (Quark SE)
- Carbon (Cortex M4)
- QEMU*

Combined Host & Controller
- Bluetooth API
- Virtual HCI driver
- nRF52-based boards

*Other names and brands may be claimed as the property of others.
Bluetooth BR/EDR support

- Bluetooth Classic
- GAP, L2CAP, RFCOMM, SDP
- Hands-Free Profile (HFP)
- Advanced Audio Distribution Profile (A2DP)
- Audio/Video Remote Control Profile
Future

- Work on upcoming specifications
- Bluetooth 5.0
- Bluetooth Mesh
- LE Link Layer support for more radios
- Better net_buf integration for Link Layer
- Link Layer Privacy
Questions?