BLE - FAQ

This is the FAQ (Frequently Asked Questions) list of TI's BLE (Bluetooth Low Energy) solution.

BLE Protocol Related Questions

How is the compatibility between BLE and the classic Bluetooth protocol?

BLE (Bluetooth Low Energy) is part of the Bluetooth Core Specification version 4.0 [1] from Bluetooth SIG which defines the the classic Bluetooth legacy protocols, the Wi-Fi based Bluetooth high speed, and also the Bluetooth Lew Energy (BLE). The BLE is defined as subset of Bluetooth v4.0 aiming low power, low latency, and low throughput applications with completely different new protocol which is not compatible with the other legacy Bluetooth protocols.

What is the difference between Single Mode and Dual Mode BLE devices?

The single-mode BLE device implements only the new BLE protocol while the dual-mode BLE device implementation is also integrated with the classic Bluetooth protocol.

BLE Hardware Questions

How can I access the CC254x GPIOs on a SmartRF05

Almost all CC2541 I/Os are available on jumper P1 and/or P10 on the SmartRF05 board:

I/O Mapping

CC2541	**SmartRF05**	
	Alt. 1	Alt. 2
P0.0	P10.5	P1.33
P0.1	P10.17	
P0.2		P1.7
P0.3		P1.5
P0.4		P1.3
P0.4		P1.1
P0.6	P10.3	
P0.7	P10.33	
P1.0	P10.21	
P1.1	P10.25	
P1.2	P10.7	P1.31
P1.3	P10.15	
P1.4	P10.29	P1.23
P1.5	P10.13	P1.31
P1.6	P10.11	P1.27
P1.7	P10.9	P1.25

P2.0		
P2.1		P1.21
P2.2		P1.19

Do I need a 32 kHz crystal to run BLE?

If you are implementing the POWER_SAVING (using the 1 uA PM2 sleep) in your CC254x design, an accurate 32 kHz clock is needed for the sleep timer to maintain the connection timing required by BLE. If you are mains powered, for example, and do not need to use the POWER_SAVING feature, the timing is maintained with the 32 MHz crystal and no 32 kHz crystal is needed.

Sleep crystal accuracy

According to the Bluetooth 4.0 Core Specifications, a peer device with up to 500 PPM frequency deviation on its sleep clock must be accepted. When selecting a 32.768kHz crystal for a custom board, therefore, you may choose one with up to 500 PPM spread in the production around the expected value.
In TI's BLE stacks, a default PPM value of 40 is used for peripheral- and 50 for central devices. If you select a more inaccurate crystal, this must be compensated in software.
To compensate for sleep clock inaccuracy you can use the command `HCI_EXT_SetSCACmd(<ppm_val>);` which must be executed during initialization of the device. For central devices, this value is used internally for compensation and also converted to a BLE Spec-defined ordinal which tells peripheral devices how much drift they can expect. For peripheral devices this value is used directly, as the central/initiating device always transmits first in a connection event. Read more about this command in the Vendor Specific HCI Guide included in the TI BLE stack.
The drawback of using a clock with a wider distribution is that the software compensation entails widening the RX-window. That is, the time the device spends listening before the 'real' beginning of a connection event in case either peer has drifted. This consumes more power.

Measuring power consumption

Power measurements over time can be carried out on e.g. a CC2540 KeyFob by following the instructions in our Application Note AN092 Measuring Bluetooth Low Energy Power Consumption [2].
The application note includes details on how to modify the board, an example measurement and an Excel spreadsheet template for inputting measured data and automatically compute average power consumption.

Some measurements

These measurements were carried out with a CC2541 with a TPS62730 DC-DC step-down converter. Battery life is based on an ideal 230 mAh battery.

- **SimpleBLEPeripheral -** ***Advertising*** Spreadsheet
  - Time awake: 3.9 ms: 1.28 ms processing after Rx/Tx
  - Average current draw for event only: 8.6 mA
  - Average current draw with 1s interval: 0.035 mA / ~260 days continuously
- **SimpleBLEPeripheral -** ***Empty connection event*** Spreadsheet
  - Time awake: 3.0 ms: 1.3 ms processing after Rx/Tx
  - Average current draw for event only: 8.9 mA
  - Average current draw with 1s interval: 0.028 mA / ~327 days continuously
- **SimpleBLEPeripheral -** ***Read request*** **to KeyFob, 1 characteristic** Spreadsheet
  - Time awake: 5.7 ms: 3.9 ms processing after Rx/Tx

  - Average current draw for event only: 7.5 mA
  - Average current draw with 1s interval: 0.044 mA / ~208 days continuously
- **SimpleBLEPeripheral - *Read response* from KeyFob, 1 characteristic, 1 byte**
  - Time awake: 2.8 ms: 1.3 ms processing after Rx/Tx
  - Average current draw for event only: 8.6 mA
  - Average current draw with 1s interval: 0.026 mA / ~352 days continuously
- **SimpleBLEPeripheral - *Write request* to KeyFob, 1 characteristic, 1 byte** Spreadsheet
  - Time awake: 5.9 ms: 4.1 ms processing after Rx/Tx
  - Average current draw for event only: 7.5 mA
  - Average current draw with 1s interval: 0.045 mA / ~200 days continuously
- **SimpleBLEPeripheral - *Write response* from KeyFob, 1 characteristic**
  - Time awake: 3.1 ms: 1.4 ms processing after Rx/Tx
  - Average current draw for event only: 8.8 mA
  - Average current draw with 1s interval: 0.028 mA / ~327 days continuously
- **SimpleBLEPeripheral - *Notification* from KeyFob, 1 byte** Spreadsheet
  - Time awake: 3.2 ms: 1.4 ms processing after Rx/Tx
  - Average current draw for event only: 8.8 mA
  - Average current draw with 1s interval: 0.029 mA / ~316 days continuously
- **SimpleBLEPeripheral - *Notification* from KeyFob, 60 bytes looped** Spreadsheet
  - Time awake: 8.8 ms: 5.8 ms processing after Rx/Tx
  - Average current draw for event only: 7.89 mA
  - Average current draw with 1s interval: 0.071 mA / ~135 days continuously

BLE Software Questions

Where can i find example codes for the BLE host (PC/Smartphone)?

Please refer: here [3].

Where can i find the example code for BLE SoC (CC2540/1) peripherals?

The BLE-Stack contains example code with HAL (Hardware Abstraction Layers) which can be used as the peripheral drivers. Since the CC254x shares the same peripherals as the CC253x (Zigbee SoC), example codes for CC253x peripherals should also apply to CC254x.

Others can be found as follows:

- CC2541: SWRC257 [4]

How many slave devices can be connected to a BLE master simultaneously?

The BLE standard does not really define the maximum number of devices (see here [5]), but using the BLE-Stack (starting from v1.1) the master shall support up to 3 simultaneous connections.

The following is taken from the Release Notes of BLE-Stack v1.1:

```
Version 1.1
July 13, 2011

Changes and Enhancements:
```

```
- The stack now supports up to 3 simultaneous connection as a central / master
  device, with a few constraints:

      - All connection intervals must be a multiple of the minimum connection
        interval (i.e. the minimum connection interval is the greatest common
        denominator of all connection intervals).

      - The minimum connection interval allowed is 25ms when using more than
        one connection.

      - When more than one connection is active, only one data packet per
        connection event will be allowed in each direction.

      - Scanning is not supported while in a connection. The consequences of
        this is that device discovery is not possible while in a connection.
        Therefore, to discover and connect to multiple devices, the device
        discovery must occur before the first connection is established.
```

Advertisement timeout

There's been a change in v1.3 of the stack to the max timeout for limited advertising, it's now 180 seconds instead of 30.72. This to follow Bluetooth spec addendum. See Thread #1 [6] and Thread #2 [7] for more information on how to change this.

How can I change the output Power on CC254x?

With the Vendor Specific HCI command **HCI_EXT_SetTxPowerCmd()**. Allowed paramters:

- HCI_EXT_TX_POWER_MINUS_23_DBM
- HCI_EXT_TX_POWER_MINUS_6_DBM
- HCI_EXT_TX_POWER_0_DBM
- HCI_EXT_TX_POWER_4_DBM (CC2540 Only)

Parameter update request

Sometimes you want to change the connection interval and slave latency from the peripheral side to conserve power, especially when connection to an iOS device. To do this, you can call the function

```
bStatus_t GAPRole_SendUpdateParam( uint16 minConnInterval, uint16 maxConnInterval, uint16 latency, uint16 connTimeout, uint8 handleFailure )
```

which is defined and documented in peripheral.c and will send an L2CAP request for parameter update.

Alternatively, see example in simpleBLEperipheral.c, change #define DEFAULT_ENABLE_UPDATE_REQUEST to TRUE near the top and fiddle with the desired parameters. This will cause peripheral.c to issue an update request on the GAP_LINK_ESTABLISHED_EVENT of its gapRole_ProcessGAPMsg(). See simpleBLEperipheral's task init to see how this GAPRole data is communicated to peripheral.c.

Be sure to follow Apple's Bluetooth design guidelines [8] when doing this, as there are limitations on acceptable parameters.

IAR 8.20 Linker error

For BLE-Stack v.1.3, TI recommends to use IAR 8051 EWB 8.10.4 or 8.11.4. For downloading the specific IAR 8051 EWB please refer here [9].

For a workaround for the SLEEP_CODE segment error, use this workaround [10].

Which IDE can be used for working with BLE-Stack

At the moment for BLE-Stack v1.3.x, the BLE-Stack can only be compiled using the IAR EWB 8051 [9] version 8.x. No other IDE is possible as part of BLE-Stack is released in binary library format which can only work with the IAR compiler.

We recommend using IAR 8.10.4, as that's what the library files have been compiled with and the stack has been tested with.

How big is the memory footprint of the BLE-Stack?

The answer varies depending on the stack configurations. To give some numbers, the following memory footprint is acquired from the current BLE-Stack examples (as is) compiled using IAR 8051 EWB v8.11.1:

Project Name	**Memory Footprint**
SimpleBLECentral-CC2540EM	118 379 bytes of CODE memory 29 bytes of DATA memory (+ 81 absolute) 7 246 bytes of XDATA memory 194 bytes of IDATA memory 8 bits of BIT memory 4 360 bytes of CONST memory
SimpleBLECentral-CC2541EM	121 188 bytes of CODE memory 29 bytes of DATA memory (+ 81 absolute) 7 246 bytes of XDATA memory 194 bytes of IDATA memory 8 bits of BIT memory 792 bytes of CONST memory
SimpleBLEObserver-CC2540EM	38 926 bytes of CODE memory 29 bytes of DATA memory (+ 81 absolute) 3 272 bytes of XDATA memory 194 bytes of IDATA memory 8 bits of BIT memory 3 847 bytes of CONST memory
SimpleBLEPeripheral-CC2541DK-MINI Keyfob	108 502 bytes of CODE memory 35 bytes of DATA memory (+ 72 absolute) 6 270 bytes of XDATA memory 194 bytes of IDATA memory 8 bits of BIT memory 4 145 bytes of CONST memory
SimpleBLEPeripheral-CC2540EM	110 648 bytes of CODE memory 35 bytes of DATA memory (+ 72 absolute) 6 270 bytes of XDATA memory 194 bytes of IDATA memory 8 bits of BIT memory 581 bytes of CONST memory

Heartrate-CC2540DK-MINI Keyfob Slave	108 110 bytes of CODE memory 35 bytes of DATA memory (+ 74 absolute) 6 113 bytes of XDATA memory 194 bytes of IDATA memory 8 bits of BIT memory 4 151 bytes of CONST memory

References

[1] http://www.bluetooth.org/Technical/Specifications/adopted.htm

[2] http://www.ti.com/general/docs/litabsmultiplefilelist.tsp?literatureNumber=swra347a

[3] http://processors.wiki.ti.com/index.php/Category:BluetoothLE#Application_Examples

[4] http://www.ti.com/litv/zip/swrc257

[5] http://en.wikipedia.org/wiki/Bluetooth_low_energy#Technical_Details

[6] http://e2e.ti.com/support/low_power_rf/f/538/p/236961/830730.aspx#830730

[7] http://e2e.ti.com/support/low_power_rf/f/538/p/85623/599426.aspx#599426

[8] https://developer.apple.com/hardwaredrivers/BluetoothDesignGuidelines.pdf

[9] http://www.iar.com/en/Products/Wireless-solutions/Tools-for-TI-wireless/

[10] http://e2e.ti.com/support/low_power_rf/f/538/t/235981.aspx#828065

Article Sources and Contributors

BLE - FAQ *Source*: http://processors.wiki.ti.com/index.php?oldid=153072 *Contributors*: A.normann, JLindh, Lhend, P.rand

License

matter take effect in the relevant jurisdiction in which the License terms are sought to be enforced according to the corresponding provisions of the implementation of those treaty provisions in the applicable national law. If the standard suite of rights granted under applicable copyright law includes additional rights not granted under this License, such additional rights are deemed to be included in the License; this License is not intended to restrict the license of any rights under applicable law.