Application Note

CC256x Testing Guide

ABSTRACT

This application note is a testing guide for the CC256x devices.

Table of Contents

Trademarks

Bluetooth® is a registered trademark of Bluetooth SIG.
All trademarks are the property of their respective owners.

1 Device Power Up

1.1 Required Voltage Rail and Signal Sequence

Proper power up of the device:

1. *nSHUTDOWN* must be low.
2. Turn on power supplies:
   a. Turn on *VDD_IN* (3.3V, max range is 1.8V to 4.8V)
   b. Turn on *VDD_IO* (1.8V, max range is 1.62V to 1.92V)
   c. *VDD_IO* and *VDD_IN* must be stable before releasing *nSHUTDOWN*.
3. *Slow Clock* must be stable within 2ms of releasing *nSHUTDOWN*.
4. *Fast Clock* must be stable within 20ms of releasing *nSHUTDOWN*.
5. The CC256x device will indicate a complete proper power-up sequence by pulling *HCI_RTS* low.

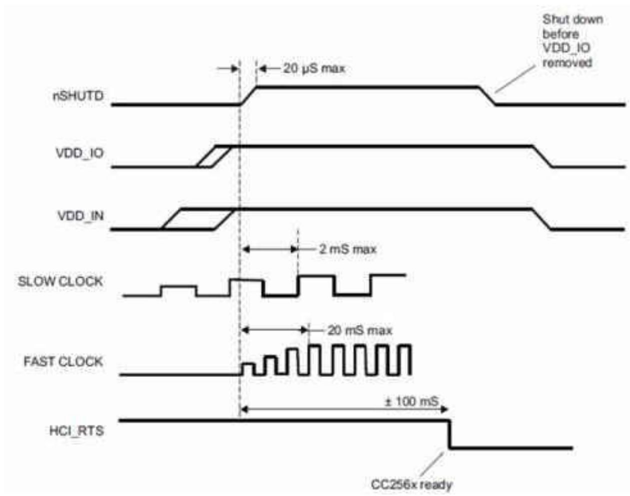

Figure 1-1. Proper Power Up Sequence

No signals are allowed on the I/O pins if *VDD_IO* power is not present. The only exceptions on this are the *SLOW_CLK*, *XTALP*, *XTALM*, and *AUD_xxx* pins that are fail-safe and can tolerate external voltages without *VDD_IO* and *VDD_IN* present.

1.2 nSHUTDOWN and HCI_RTS

The 4-wire UART connection (H4) has the following default settings:

- 115.2 kbps baudrate
- Hardware (HW) flow control
- UART 8N1 setting:
  – 8-bits
  – No parity bit
  – 1 stop bit

An example of UART HCI communication:

Outgoing command: **HCI_Read_Local_Version_Information**

Outgoing Hex dump: **0x01 0x01 0x10 0x00** where the first number indicates a command sent **(0x01)**, the second number **(0x01)** and third number **(0x10)** is the command opcode and fourth indicates parameter length

Incoming Hex dump: **0x04 0x0e 0x0c ......** where the first number **(0x04)** indicates an event received, the second number **(0x0e)** indicates the type of event, and the rest of the numbers is the event.

Note

It is not necessary to download the service pack to verify UART communication with the example above.

2 UART Communication

The 4-wire UART connection (H4) has the following default settings:

- 115.2 kbps baudrate
- Hardware (HW) flow control
- UART 8N1 setting:
  - 8-bits
  - No parity bit
  - 1 stop bit

An example of UART HCI communication:

- Outgoing command: **HCI_Read_Local_Version_Information**
- Outgoing Hex dump: **0x01 0x01 0x10 0x00** where the first number indicates a command sent **(0x01)**, the second number **(0x01)** and third number **(0x10)** is the command opcode and fourth indicates parameter length
- Incoming Hex dump: **0x04 0x0e 0x0c ......** where the first number **(0x04)** indicates an event received, the second number **(0x0e)** indicates the type of event, and the rest of the numbers is the event.

It is not necessary to download the service pack to verify UART communication with the example above.

3 Device Initialization Procedure

The service pack (SP) download procedure:

- Is done right after device power-up sequence.
- Must be done after every power cycle.
- Is done before any Bluetooth® RF testing.
- Must be done before any Bluetooth operations such as *Inquiry*, *Page* or connections

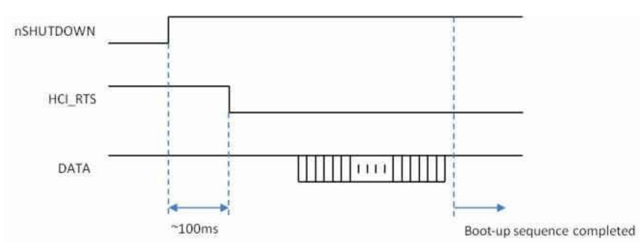

Figure 3-1. Device Initialization Sequence

4 Basic Bluetooth Operations

- Service pack must have been loaded.
- Examples of basic Bluetooth operations:
  - Inquiry: The CC256x device is looking for other Bluetooth devices that are nearby.
  - Inquiry Scan: The CC256x device is visible but not connectable (it has discoverable MAC address).
  - Page Scan: The CC256x device is connectable but not visible (can accept connections).

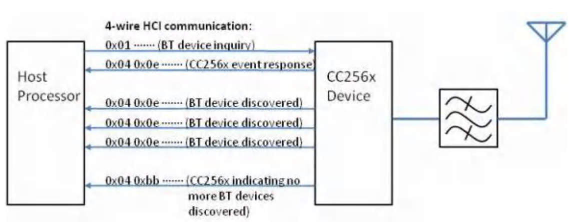

Figure 4-1. Example of Bluetooth Inquiry Operation

5 Bluetooth RF FCC Modes

The Bluetooth RF FCC modes are connection-less RF spectrum tests which uses a spectrum analyzer for verification.

Note

The CC256x device must be power cycled when switched between different FCC modes. Please also note that the initscript must be downloaded to the CC256x device after each power cycle before performing the next test.

5.1 Continuous TX

This is a non-packet continuous transmission with either CW (Continuous Wave), GFSK (BR), π/4-DQPSK (2-EDR) or 8DPSK (3-EDR).

The HCI commands to put the device in the constant TX mode (after service pack has been loaded) are:

- HCI_VS_DRPb_Tester_Con_TX 0xFD84, 0x1, 0, 0, 15, 0x00000000, 0x00000000
- HCI_VS_Write_Hardware_Register 0xFF01, 0x0019180c, 0x0101
- HCI_VS_DRPb_Enable_RF_Calibration 0xFD80, 0xFF, 0xFFFFFFFF, 0x01

The parameters for the *HCI_VS_DRPb_Tester_Con_TX* can be found.

Note

The second and third commands are for internal settings (*HCI_VS_Write_Hardware_Register* and *HCI_VS_DRPb_Enable_RF_Calibration*). The HCI_VS_Write_Hardware_Register 0xFF01, 0x0019180c, 0x0101 command is only required for the continuous TX test of modulated (GFSK, π/4-DQPSK or 8DPSK) signal. This command should be skipped when performing continuous TX test for CW.

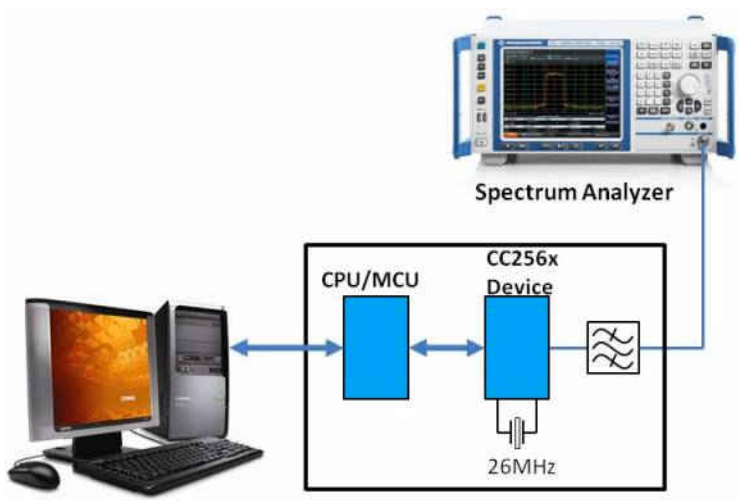

Figure 5-1. Typical Test Setup for Continuous TX Mode

Note

It is recommended to use the HCITester tool on the PC to send the HCI commands to the CC256x device. The HCITester can be downloaded as part of the Wireless Tools package.

5.2 Packet RX TX

To enable the continuous packet transmission/receive the following command needs to be sent after the loading of the service pack:

```
HCI_VS_DRPb_Tester_Packet_TX_RX 0xFD85, 0x03, 0, 0xFF, 0, 2, 0, 27, 15, 1, 0x01FF
```

The parameters for the *HCI_VS_DRPb_Tester_Packet_TX_RX* can be found.

Note

By default, the device uses all the 79 channels in frequency hopping mode. To reduce the number of channels, the following commands should be executed prior to *HCI_VS_DRPb_Tester_Packet_TX_RX*.

- Enable Local AFH Assessment

```
HCI_Write_AFH_Channel_Assessment_Mode 0x1
```

- Classify Channels to Reduce Hopping Scheme

```
Bad Channel = 1, Unknown Channel = 0
<ChannelMap> = B0B1B2B3B4B5B6B7B8B9
Bx = b7b6b5b4b3b2b1b0
Chx = bx
Freq = 2402 + Chx
e.g. <ChannelMap> = '000000E0FFFF01000000', Blocked Channels = 0 - 28 and 49 - 78, Used
Frequencies = 2,431MHz - 2,450MHz
HCI_Set_AFH_Host_Channel_Classification <ChannelMap>
```

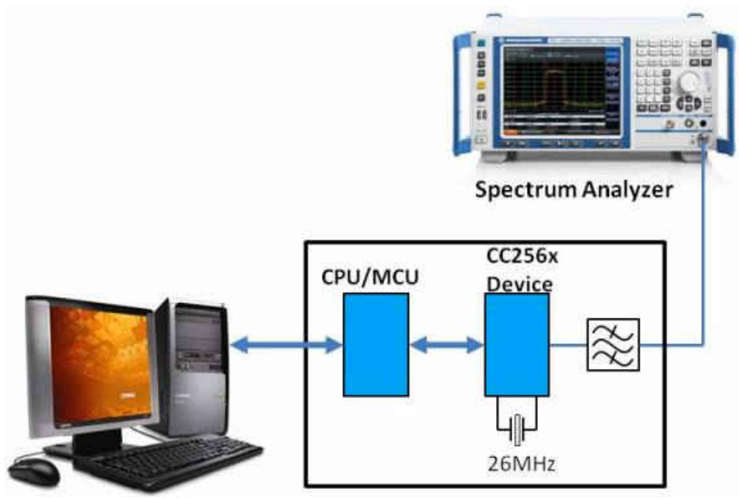

Figure 5-2. Typical Test Setup for Continuous TX Mode

5.3 Continuous RX

To turn on the receiver portion of the chip the following command needs to be sent after the loading of the service pack:

```
HCI_VS_DRPb_Tester_Con_RX 0xFD17, 0, 0x00
```

The parameters for the *HCI_VS_DRPb_Tester_Con_RX* are:

1. Op code (0xFD17)
2. Frequency index (**0-39**: f=2402+(2*i)MHz, **40-78**: f=2403+2(i-40)MHz)
3. Internal setting (0x00)

The setup would be the same as for *Continuous TX*

6 Bluetooth RF SIG Mode

The Bluetooth RF SIG mode is meant for connecting with a Bluetooth tester where the CC256x is controlled over the LMP (Link Management Protocol). The procedure for enabling the CC256x for Bluetooth SIG mode is:

1. Proper device power-up
2. Load the correct service pack
3. Load the DUT script (three HCI commands)

Once the device (DUT) is in test mode, the Bluetooth tester will, through the RF connection (LMP), take control of the DUT.

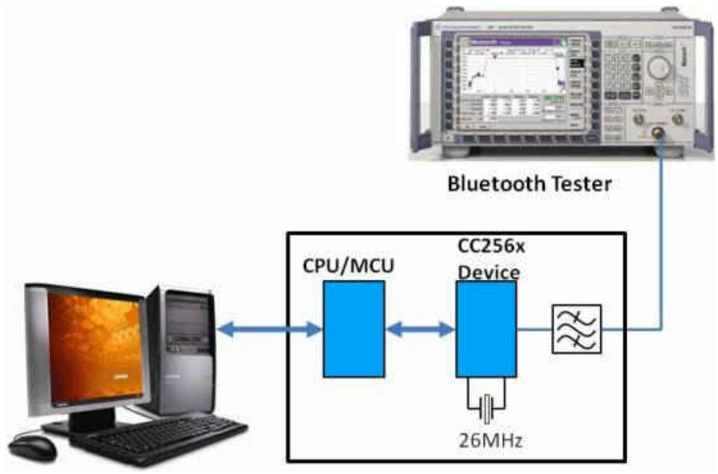

Figure 6-1. CC256x Bluetooth RF SIG Mode Test Setup

The HCI commands to put the device in test RF SIG mode are:

- HCI_Set_Event_Filter 0x02, 0x00, 0x03
- HCI_Write_Scan_Enable 0x03
- HCI_Enable_Device_Under_Test_Mode

This script will make the device visible and connectable (*HCI_Set_Event_Filter*), auto-accept all connections (*HCI_Write_Scan_Enable*) and put the DUT in test mode (*HCI_Enable_Device_Under_Test_Mode*). Once the proper sequence has been completed (first loading the Bluetooth service pack and then the DUT script), then the Bluetooth tester will take control of the device through the RF link (LMP).

7 Bluetooth Low Energy Testing

This section provides information on how to configure the CC2564 device for the Bluetooth Low Energy (BLE) SIG RF PHY Testing.

The DUT is the CC2564 dual-mode Bluetooth device (possibly interfaced with host MPU/CPU). The Upper Tester is the PC which communicates with the DUT by sending HCI commands using UART 8-N-1 serial protocol. The DUT is also connected to the lower tester which is the Bluetooth tester (such as CBT Tester) via a U.FL/SMA connector.

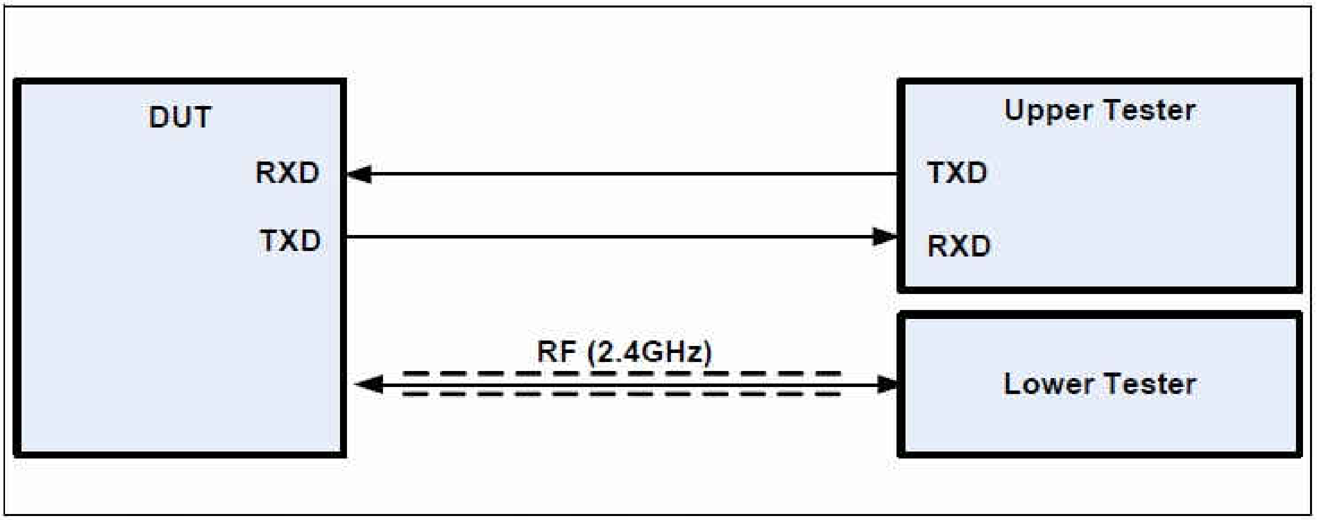

Figure 1.2: RF PHY test setup for Direct Test Mode (UART control)

Figure 7-1. Bluetooth SIG Core V4.0

7.1 Transmitter Test

The Bluetooth device will transmit to the Bluetooth tester which will analyze the power spectrum of the received data.

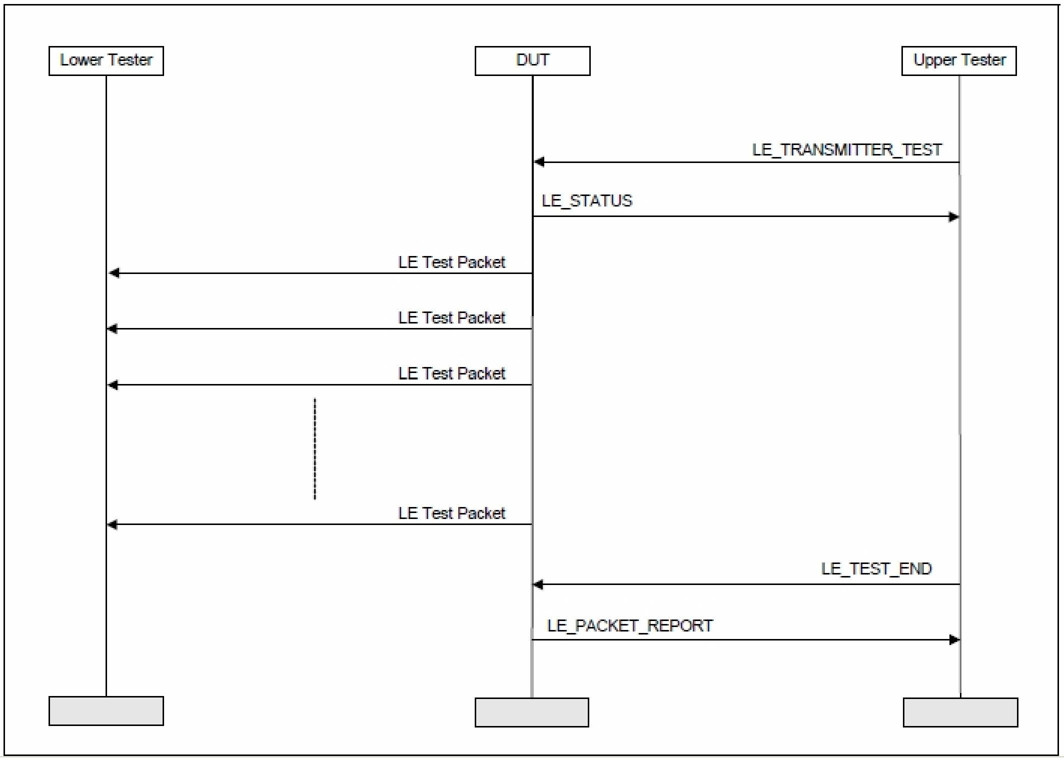

Figure 7-2. Transmitter Test, Source: Code V4.0

1. Initialize CC2564 device
   Load Core BT Service Pack (BT SP)to the CC2564 device Download Core BT SP
2. Enable BLE mode

   Note

   If the CC256x Service Pack (SP) version contains a BLE add-on SP, then the *HCI_VS_LE_Enable 0xFD5B* command is embedded inside the BLE add-on SP and must not be sent again.

   If the CC256x Service Pack (SP) version contains a BLE add-on SP:
   Load BLE Add-On SP Download BLE Add-On SP

   Otherwise,

   Execute the "HCI_VS_LE_Enable 0xFD5B, 1, 1" command
3. Set the BLE Test Parameters

   Note

   Power Level 1 is used for BLE. For more information on this command, refer to the CC256x_VS_HCI_Commands wiki

   HCI_VS_Set_LE_Test_Mode_Parameters 0xFD77, 0x01, 0x0, Packets to transmit, Access Code, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00000000

Table 7-1. BLE Test Parameters

Command Parameter	Size (bytes)	Value	Description
Packets to transmit	2	0x0000 N	*Unlimited* - Continuous TX *N* number of packets to transmit (1)
Access code	4	0xXXXXXXXX	An access code to sync and transmit*Default = 0x71764129 (TestMode AC)*

(1) If the *Packets to transmit* is a specific value (i.e. Non-continuous), then the *HCI_BLE_Transmitter_Test 0x201E* must be sent twice. For more information, refer to the following E2E post

4. Start TX test
   HCI_BLE_Transmitter_Test 0x201E, TX_Channel , Data_Length, Payload_Type

Table 7-2. Start TX Test

Command Parameter	Size (bytes)	Value	Description
TX_Channel	1	0 - 39	TX Frequency: 2402 + 2*k,where k is the channel (max val is 39)
Data_Length	1	0 - 37 (0x025)	Packet payload length up to 37 (0x25) bytes
Payload_Type	1	0 - 7	0 - PRBS9 1 - FOFO 2 - ZOZO 3 - PRBS15 4 - All Ones 5 - All Zeros 6 - OFOF 7 - OZOZ

5. Take measurements
   Look at the Bluetooth tester results.
6. End BLE TX Test
   HCI_BLE_Test_End 0x201F

Note

To re-start transmission, go back to steps 3 (optional) and 4.

7.2 Receiver Test

The Bluetooth device will receive data from CBT and PC will calculate PER and BER from received data.

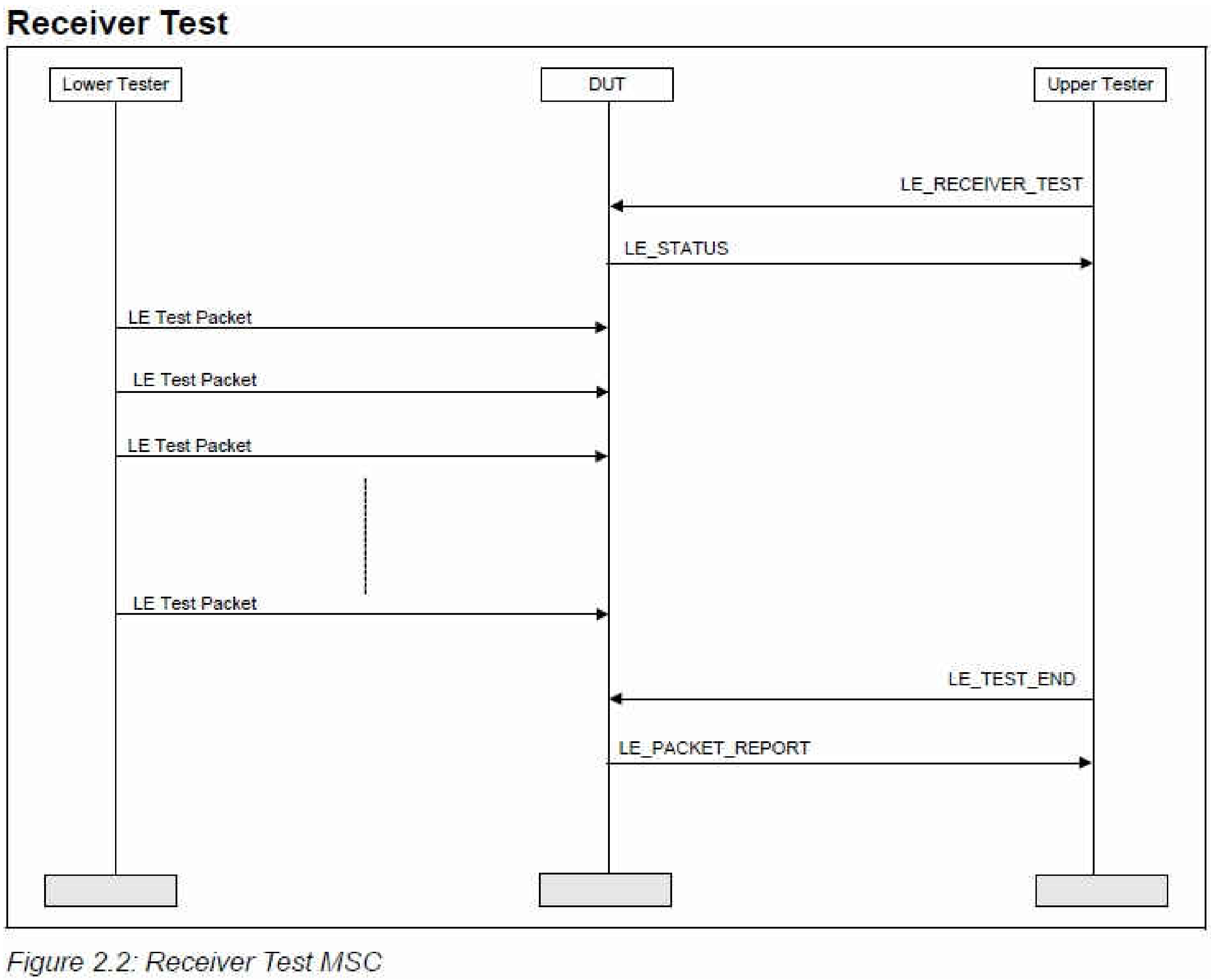

Figure 7-3. Source: Code V4.0

1. Enable BLE mode:
   HCI_VS_LE_Enable 0xFD5B, 1, 1
2. Set the type of test to run
   PER_nBER = 1 (PER=1 or BER=0)
3. Set here the number of packets the tester transmits for PER measurements
   Tx_Packets = 1500
4. Set RX Frequency Channel 2402 + 2*k, where k is the channel (max val is 39)
   RX_Channel = 0 (same as k)
5. Set the RX Mode According to the options below (Default is "1"):

   # 0 - Normal mode, Open loop & no power saving

   # 1 - Wide window mode, Close loop & no power saving

   # 2 - Continuous RX, Open loop before SYNC

   # 3 - Wide window & power saving mode, Close loop & power saving
   RX_Mode = 1
6. Set the RX Access Code to Sync on (Default is 0x71764129)
   RX_AC = 0x71764129

7. Set payload type

   # 0 - PRBS 9

   # 1 - FOFO

   # 2 - ZOZO

   # 3 - PRBS 15

   # 4 - All Ones

   # 5 - All Zeros

   # 6 - OFOF

   # 7 - OZOZ
   PayloadType = 0x00
8. Set length of packet fields in Bytes
   Header_Length = 2 Payload_Length = 37 CRC_Length = 3
9. Set CRC for PRBS9 with 37 Bytes length
   CRC_Value = 0x00E221E8
10. When traces are enabled than the Cortex is not able receive the next packet
    En_Traces = 0
11. Set the amount of bad bits/packet to identify FA (in percent) and set Enable_BER

```
FA_THR_inPer = 7
Total_Bits_in_packet = 8 * (Header_Length + Payload_Length + CRC_Length)
FA_THR_inBits = round((FA_THR_inPer/100) * Total_Bits_in_packet)

if (PER_nBER == 1) then
    Enable_BER = 0
else
    Enable_BER = 1
endif
```

12. Clear Sync Counter
    HCI_VS_Write_Hardware_Register 0xFF01, 0x0019324E, 0x0
    Outgoing Dump: 01 01 ff 06 4e 32 19 00 00 00
    Incoming Event: 04 0e 04 01 01 ff 00 (Command Complete Event)
13. Update Parameters

```
HCI_VS_Set_LE_Test_Mode_Parameters 0xFD77, 0x01, RX_Mode, 0, RX_AC, Enable_BER, PayloadType,
Payload_Length, FA_THR_inBits, En_Traces, CRC_Value
HCI_VS_Set_LE_Test_Mode_Parameters 0xFD77, 0x01, x01, 0x0000, 0x71764129(RX_AC),
0x00(Enable_BER), 0x00(PayloadType), 0x25(Payload_Length), 0x18(FA_THR_inBits), 0x00(En_Traces),
0x00E221E8 (CRC)
```

14. Start RX Scan
    HCI_BLE_Receiver_Test 0x201d, RX_Channel

    Pause here and send data packets from the CBT Tester. Continue with scripts after sending all the packets.
15. If testing for PER:
    if (PER_nBER == 1) then
    HCI_BLE_Test_End 0x201f

    Returns number of packets recieved, save into &Rx_Packets.
16. Read Phy - Sync Counter Value
    HCI_VS_Read_Hardware_Register 0xFF00, 0x0019324E
    Outgoing Dump: 01 00 ff 04 4e 32 19 00
    Incoming Event: 04 0e 06 01 00 ff 00 0a 00 (Command Complete Event, Value: 0x000A)

    Returns Sync Counter Value, save into &Sync_Counter.

17. Calculate PER = 100*(Tx_Packets - Rx_Packets)/Tx_Packets
18. For BER:
    else
    HCI_BLE_Test_End 0x201f

    Returns number of packets recieved, save into &Rx_Packets.
19. Read BER Results
    Hci_VS_LE_Read_Ber_Test_Results 0xFDAE

    Vendor-Specific LE_Read_Ber_Test_Results Command Complete parameters:
20. Read Phy - Sync Counter Value
    HCI_VS_Read_Hardware_Register 0xFF00, 0x0019324E
    Outgoing Dump: 01 00 ff 04 4e 32 19 00
    Incoming Event: 04 0e 06 01 00 ff 00 0a 00 (Command Complete Event, Value: 0x000A)

    Returns Sync Counter Value, save into &Sync_Counter.
21. Calculate Actual Syncs
    Actual_Syncs = BER_Syncs - FA_Events
22. Calculate BER:

```
# Bit Counters
Total_Bad_Bits = Htype_Bad_Bits + Hlength_Bad_Bits + Ppart1_Bad_Bits + Ppart2_Bad_Bits +
Ppart3_Bad_Bits + Ppart4_Bad_Bits + CRC_Bad_Bits
Dropped_Packets = Actual_Syncs - Total_Good_Packets
Detected_Packets = Actual_Syncs
Total_Received_Bits = Total_Bits_in_packet * Detected_Packets

if(Total_Received_Bits == 0) then
BER = 100
else
BER = 100*(Total_Bad_Bits / Total_Received_Bits)
endif
```

8 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

DATE	REVISION	NOTES
September 2022	*	Initial Release