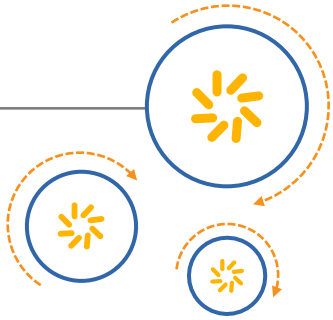
Qualcomm Technologies International, Ltd.



CSRmesh™ Sniffer

Application Note

80-CF535-1 Rev. AA

November 21, 2017

Revision history

Revision	Date	Description
1	OCT 2016	Initial release
AA	NOV 2017	Updated to use Agile DCN. Alternate DCN is CS-00335121-AN.

Contents

Figures

Tables

1 Introduction

This document describes the CSRmesh Sniffer on-chip application built using the Qualcomm® µEnergy™ SDK.

The application captures and decodes CSRmesh packets using different configurable parameters.

1.1 Application overview

The CSRmesh Sniffer application runs on an IOT Lighting Board or any CSR101x development kit. The application captures and prints the decoded mesh messages on UART in ASCII format. The application is delivered as a binary image and can be configured using the CS Key file.

1.1.1 Profiles supported

The CSRmesh Sniffer application implements the following CSRmesh custom profile to support the use cases described:

CSRmesh Control Profile

The CSRmesh Control Profile defines the behaviour when:

- A network of devices such as lights and switches needs to be created.
- Controlling the device after a network is created. For example, switching on/off, changing the intensity or the colour of a light.
- Reading the status of a device in the network, for example on/off state, colour or intensity of a light device.

Table 1-1 lists the roles defined by CSRmesh Control Profile.

Table 1-1 CSRmesh Control Profile roles

Role	Description
CSRmesh Bridge Device	Receives commands from the host and sends them over the CSRmesh network to associated devices. Receives responses from associated devices over the CSRmesh and forwards them to the host via a Bluetooth Smart connection.
CSRmesh Control Device	The CSRmesh Control Device provides the interface to create a network of devices and control the associated devices. The control commands are sent via a Bluetooth Smart connection to the CSRmesh devices.

The CSRmesh Bridge Device role is implemented on the CSRmesh Sniffer application. The CSRmesh Control Device is implemented on a Bluetooth Smart enabled phone or tablet.

1.1.2 Application topology

Table 1-2 and Table 1-3 list the topology that the CSRmesh Bridge uses.

Table 1-2 Application topology

Role	Mesh Control Service	GAP Service	GATT Service
GATT Role	GATT Server	GATT Server	GATT Server
GAP Role	Peripheral	Peripheral	Peripheral

Table 1-3 Role and responsibilities

Role	Responsibility
GATT Server	Accepts incoming commands and requests from a client and sends responses, indications and notifications to the client.
GAP Peripheral	Accepts connection from remote device and acts as a slave in the connection.

1.1.3 Services supported in GATT Server role

The application exposes the following services:

- CSR™ custom Mesh Control Service v 2.1
- GATT Service
- GAP Service

The Mesh Control Service is mandated by the CSRmesh Control Profile. The GATT and GAP Services are mandated by *Bluetooth Core Specification Version 4.1*.

Figure 1-1 shows the services that the GATT Server Role supports.

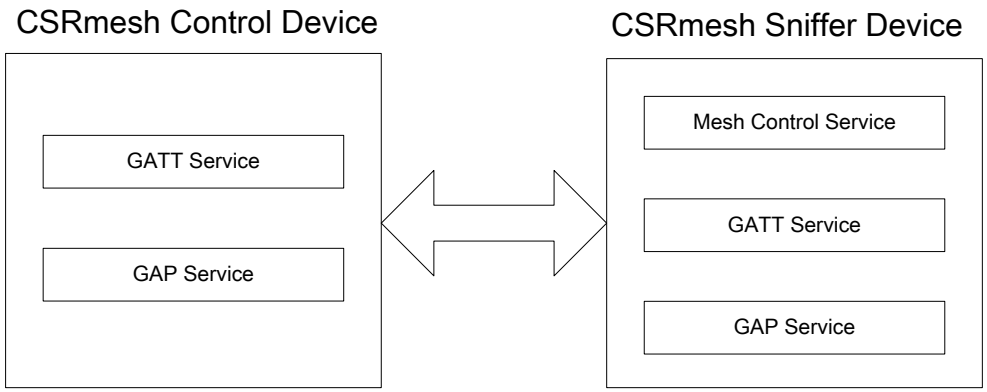


Figure 1-1 Primary services

For more information about GATT server and GAP peripheral, see *Bluetooth Core Specification Version 4.1*.

2 Using the application

This section describes how the CSRmesh Sniffer application can be used with CSRmesh Control application to control devices network.

2.1 Demonstration kit

Table 2-1 lists the components that demonstrate the application.

Table 2-1 CSRmesh components

Component	Hardware	Application
Sniffer Device	CSRmesh Development PCB (DB-CSR1010-10185-1A) OR CSR1011 Development Board	CSRmesh Sniffer Application v 2.1

2.1.1 IOT Lighting Board

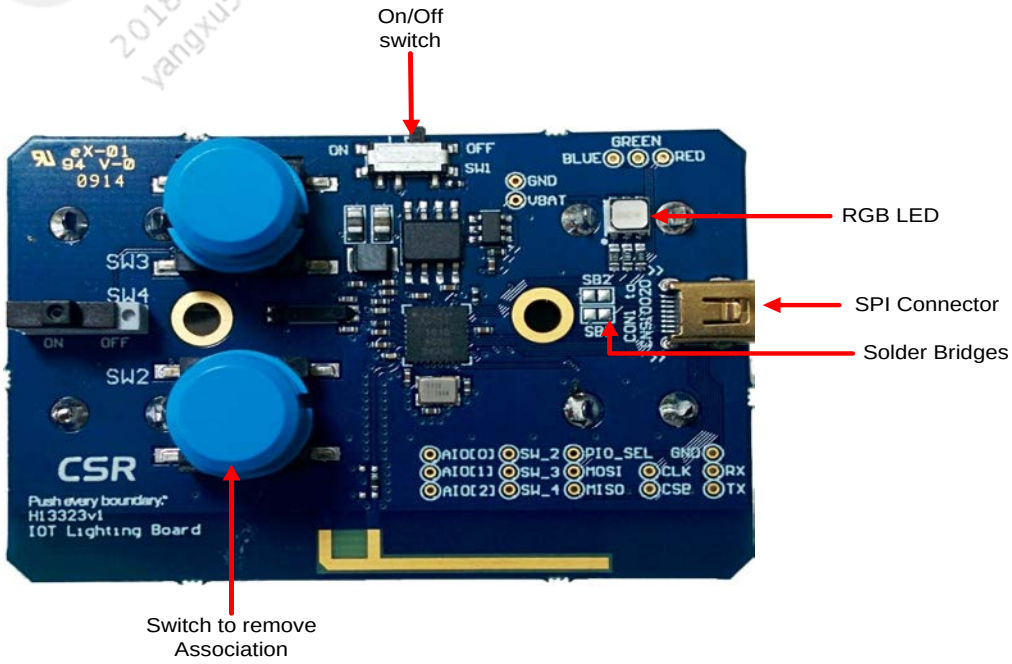


Figure 2-1 CSRmesh IOT Lighting Board

Ensure the development board is switched on using the Power Slider switch.

Figure 2-1 shows the switch in the **OFF** position.

NOTE: Shorting the solder bridges allows the UART to be brought out via the SPI connector. The PIOs connected to SW2 and SW3 are also mapped to UART Rx and Tx lines. Pressing any of these buttons will short the UART lines to ground and corrupt the data on the UART. It is recommended not to press these buttons during UART communication.

2.1.2 User interface

The application makes use of the buttons available on the CSRmesh IOT Lighting Board for the CSRmesh Sniffer application.

Table 2-2 CSRmesh IOT Lighting Board user interface

User Interface Component	Function
Switch SW1	Power slider switch allows the user to power on/off the board
Switch SW4	Unused
Buttons SW2, SW3	Unused
RGB LED	Blinks blue until it is not associated with any CSRmesh network Blinks yellow when the device association is in progress Turns off when the device is associated

NOTE: The association status display on the RGB LED is available only on the IOT Lighting Board. On any other board the application runs in the same way but the LED display is not available.

2.1.3 CSR1011 Development Board

Figure 2-2 shows a CSR1011 board with the switch set to the **BATT** position to receive power from the battery. Set the switch to receive power from the mini-USB connector.

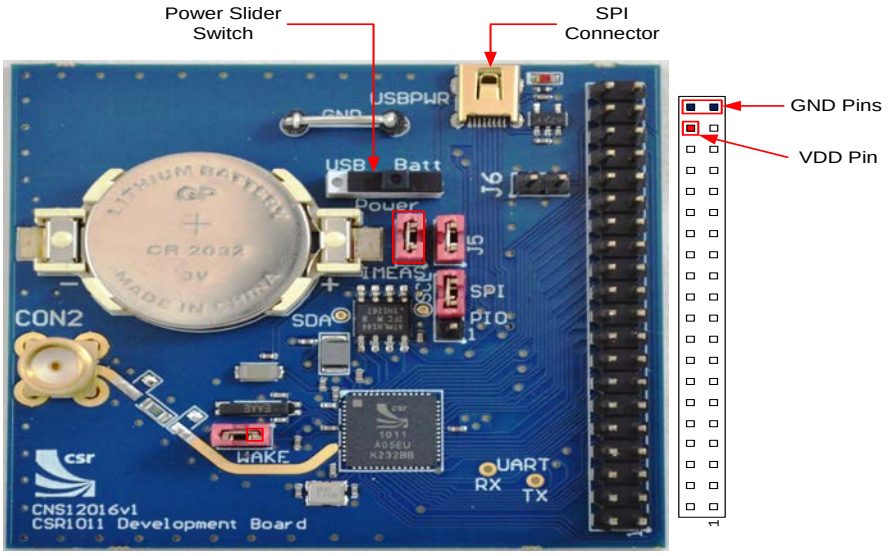


Figure 2-2 CSR1011 Development Board

2.2 Configuring the CSRmesh Sniffer parameters

Sniffer parameters can be configured by changing the CS user keys in the `csr_mesh_sniffer_csr101x_A05.keyr` file. The sniffer application sets the parameters based on the CS user key values.

Table 2-3 lists the parameters that can be configured on the sniffer.

Table 2-3 Configuring CSRmesh Sniffer parameters

CS User Key Index	Parameter	Default Value	Description
0	CSRmesh Sniffer Channel Mask	`0007`	Setting any of the bits from Bit-0 to Bit-2 enables scanning for advertisements in the corresponding channel: Bit – 0: Advertising channel-37 Bit – 1: Advertising channel-38 Bit – 2: Advertising channel-39 Bit – 3 to 15: Not used It is useful to have 3 sniffer devices, all configured to scan different channels. This helps achieve 100% scanning in all 3 channels.
1	CSRmesh Message Type Filter	`000F`	Setting any of the bits from Bit-0 to Bit-3 enables logging of the corresponding message type: Bit – 0: Print unknown network messages (MACFAIL) Bit – 1: Print MASP messages Bit – 2: Print MCP messages Bit – 3: Print Duplicate messages.(Retransmitted and relayed messages) Bit 4 to 15: Not used
2	CSRmesh Configuration Bitmask	`0001`	Bit-0 : CSRmesh Random Device UUID Enable 1 - Device generates a random UUID when it runs for the first time after flashing. The generated UUID is stored in the NVM and uses the same value upon further power cycles. 0 - Reads the UUID from NVM. See section 2.2.2 for programming a custom UUID. This can be enabled to avoid having the same UUID on multiple devices without explicitly programming an UUID on each device. Bits 1 to 15: Not used

The updated `.keyr` file can be merged with the sniffer image using the µEnergy SDK tools.

To merge the CS key file to the image file:

1. Open a command prompt.
2. Change directory to the folder where the sniffer image and the CS key file is placed.
3. Merge the updated CS key file with the application image.

```
\CSR_uEnergy_SDK-<version>\tools\bin\csconfigcmd merge
csr_mesh_sniffer_csr101x_A05.keyr -imagefile CSRMeshSniffer.img -
quiet
```

2.2.1 Flashing the application image

The µEnergy SDK downloads the CSRmesh Sniffer application on the development boards. See *µEnergy xIDE User Guide* for further information.

1. Before downloading the image to the device EEPROM, QTIL recommends that the EEPROM contents be reset to a known value, for example:

   ```
   \CSR_uEnergy_SDK-<version>\tools\bin\e2cmd -trans "SPITRANS=USB
   SPIPORT=2" fill 0xffff
   ```

   Replace the `trans` option with values appropriate to the SPI connection in use.

2. The image file may be downloaded to the device EEPROM over the SPI link as follows:

   ```
   \CSR_uEnergy_SDK-<version>\tools\bin\e2cmd -trans "SPITRANS=USB
   SPIPORT=2" download CSRMeshSniffer.img
   ```

   Replace the `trans` option with values appropriate to the SPI connection in use.

2.2.2 Setting a custom device UUID and Authorisation Code

To program the example UUID `0x0123456789ABCDEFFEDCBA9876543210` and Authorisation Code `0x0123456789ABCDEF` on the NVM:

1. Open a command prompt.
2. If base `NVM_START_ADDRESS` is defined in the `.keyr` file, the device UUID and Authorisation Code to the device can be programmed as below:

   ```
   <CSR_uEnergy_Tools path>\uEnergyProdTest.exe CSRMeshSniffer.img –k
   csr_mesh_sniffer_csr101x_A05.keyr –m1 0x04 0x3210 0x7654 0xBA98
   0xFEDC 0xCDEF 0x89AB 0x4567 0x0123 0xCDEF 0x89AB 0x4567 0x0123
   ```

   The first value following –m1 is the NVM offset from `NVM_START_ADDRESS`. The command takes the NVM offset as the byte address. Word offset 2 for device UUID is byte offset 4 (see Table 3-1).

3. If the `.keyr` file is not included in the command, the device UUID and Authorisation Code can be programmed as below:

   ```
   <CSR_uEnergy_Tools path>\uEnergyProdTest.exe CSRMeshSniffer.img –m1
   0xF804 0x3210 0x7654 0xBA980xFEDC 0xCDEF 0x89AB 0x4567 0x0123 0xCDEF
   0x89AB 0x4567 0x0123
   ```

   The first value following `–m1` is the NVM address, obtained by adding base address `0xF800` and word offset 2, for device UUID (see Table 3-1). The command takes the NVM address as the byte address. Word offset 2 is byte offset 4, so effective address is `0xF804`

2.3 Application behaviour

This section describes the application behaviour before and after it is associated.

2.3.1 Connectable advertisements

The application does connectable advertisements at approximately 1.25 ms intervals as long as it is not associated. The device can be used as a CSRmesh GATT Bridge when it is not associated.

2.3.2 Message filter and logging

The application logs all the received MASP messages before it is associated. The message filter is disabled as long as the device is not associated. Once the device is associated, it enables the configured scan channels and the message filter and starts printing the received messages.

2.3.3 Associating the device

The sniffer device needs to be associated with a CSRmesh network to decode the received messages from the CSRmesh devices in the network. The device can be associated either by connecting to the device or through a nearby bridge device.

The application sends CSRmesh device UUID message every 5 seconds for the CSRmesh control device to find the device and send an association request.

The device is listed on the **Device Association** page of the CSRmesh Control application as **Sniffer++**. Figure 2-3 shows the **Device Association** page on the CSRmesh Control application for Android.

Tap the device UUID with the name **Sniffer++** to start associating the sniffer. The sniffer device should start blinking yellow. For details about device association, see *CSRmesh Android Control Application Note* or *CSRmesh iOS Control Application Note.*

NOTE: If the device is associated when in connection, the device breaks the connection and starts logging the messages.

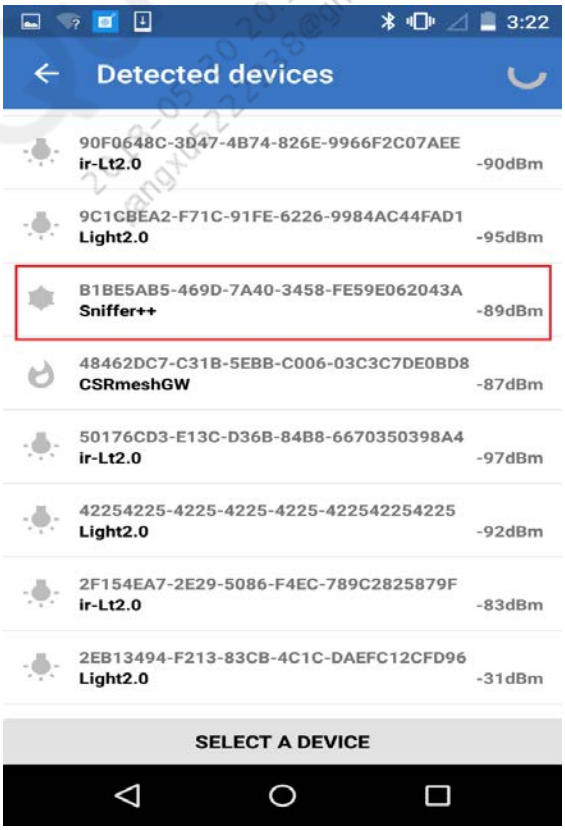


Figure 2-3 Device Association page on Control application

2.4 CSRmesh Control application

The CSRmesh Control application runs on Android version 4.3 or higher or an iOS device. It communicates with the CSRmesh devices by connecting to one of the devices that support the custom CSRmesh Control Profile. This application is required for:

- Setting up a network by associating devices
- Configuring and grouping the network devices

NOTE: For details about using the CSRmesh Control Application on an Android device, see *CSRmesh Android Control Application Note*. For details about using the CSRmesh Control Application on an iOS device, see *CSRmesh iOS Control Application Note.*

3 NVM memory map

The application stores the parameters listed in Table 3-1 in the NVM to prevent loss in case of a power off or a chip panic.

Table 3-1 Application NVM memory map

Entity Name	Type	Size of Entity (Words)	NVM Offset (Words)
Sanity Word	`uint16`	1	0
Application NVM Version	`uint16`	1	1
CSRmesh Device UUID	`uint16 array`	8	2
CSRmesh Device Authorisation Code	`uint16 array`	4	10
CSRmesh Network Key	`uint16 array`	8	15
CSRmesh Device ID	`uint16`	1	25
CSRmesh Message Sequence Number	`uint32`	2	26
CSRmesh Device eTag	`uint16 array`	4	28
CSRmesh Association State	`boolean`	1	32
GAP Device Name Length	`uint16`	1	33
GAP Device Name	`uint8 array`	20	34

A CSRmesh application GATT database

A.1 GATT Service characteristics

Table A-1 GATT Service characteristics

Characteristic Name	Database Handle	Access Permissions	Managed By	Security Permissions	Value
Service Changed	0x0003	Indicate	Application	Security Mode 1 and Security Level 1	Service Changed handle value
Service Changed Client Characteristic Configuration Descriptor	0x0004	Read, write	Application	Security Mode 1 and Security Level 1	Current client configuration for Service Changed characteristic

A.2 GAP Service characteristics

Table A-2 GAP Service characteristics

Characteristic Name	Database Handle	Access Permissions	Managed By	Security Permissions	Value
Device Name	0x0007	Read, write	Application	Security Mode 1 and Security Level 1	Device name Default value : CSRmesh
Appearance	0x0009	Read	Firmware	Security Mode 1 and Security Level 1	Unknown: 0x0000
Peripheral Preferred Connection Parameters	0x000b	Read	Firmware	Security Mode 1 and Security Level 1	Connection interval - Min 90 ms - Max 120 ms Slave latency - 0 Connection timeout - 6 s

For more information on GAP service and security permissions, see *Bluetooth Core Specification Version 4.1*.

A.3 Mesh Control Service characteristics

Table A-3 Mesh Control Service characteristics

Characteristic Name	Database Handle	Access Permissions	Managed By	Security Permissions	Value
Network Key	0x0018	Write	Application	Security Mode 1 and Security Level 1	0
Device UUID	0x001a	Read	Application	Security Mode 1 and Security Level 1	22e4-b12c-5042-11e3-9618-ce3f-5508-acd9
Device ID	0x001c	Read, write	Application	Security Mode 1 and Security Level 1	0x8001
MTL Continuation Control Point	0x001e	Write	Application	Security Mode 1 and Security Level 1	Dynamic
MTL Continuation Control Point Client Characteristic Configuration	0x001f	Read, write	Application	Security Mode 1 and Security Level 1	Current client configuration for MTL Continuation Control Point characteristic
MTL Complete Control Point	0x0021	Write, notify	Application	Security Mode 1 and Security Level 1	Dynamic
MTL Complete Control Point Client Characteristic Configuration	0x0022	Read, write	Application	Security Mode 1 and Security Level 1	Current client configuration for MTL Complete Control Point characteristic
MTL TTL	0x0024	Read, write	Application	Security Mode 1 and Security Level 1	50
MESH Appearance	0x0026	Read, write	Application	Security Mode 1 and Security Level 1	0

B Sniffer logs

This section describes the different CSRmesh messages printed on UART for analysis and debugging purpose.

B.1 Message log format

Table B-1 CSRmesh Sniffer message logging format

```
+<time delta>: <RSSI>:<BD address>: [(D)] HLM LEN: <length>: MAC: <mac> TTL: <ttl> ::
<message type>: <message>
```

Table B-2 Sniffer message log fields

Message Log	Description
Time delta	Time elapsed since the last message is received in microseconds.
RSSI	Received signal strength of the message in dBm
BD address	Bluetooth device address used to advertise the message
(D)	Indicates that the message is duplicate of a received message
length	Length of the higher layer message
mac	Message Authentication Code
TTL	Time-to-live value of the message
message type	Type of the message: MCP: Mesh Control Protocol message MASP: Mesh Association Protocol message MACFAIL: Unknown network message. The MAC cannot be verified.
message	Can be one of MCP, MASP or MACFAIL: `MCP:OPC:` <Operation Code Name>: Hexadecimal string of the message parameters `MASP: OPC:` <Operation Code Name>: Hexadecimal string of the message parameters `MACFAIL: PAYLOAD:` Hexadecimal string of the message

B.2 Application start-up logs

The messages shown in Table B-3 are displayed when the device is powered on and the sniffer is not associated.

Table B-3 Start-up logs pre-association

```
CSRmesh Sniffer Application
Sniffer is NOT Associated
Sniffer BD Address : 00:02:5b:03:18:57

Sniffer Device UUID : cd85 24f6 25b4 1a5b 3a80 1c41 001d 0162
```

The logs shown in Table B-4 are printed when the device is powered on and the sniffer is associated.

Table B-4 Start-up logs post association

```
Sniffer is Associated

Listening On Channel(s) : 37 38 39

Setting Message Log Filter:
Unknown (MAC_FAIL): ON
MASP              : ON
MCP               : ON
Duplicates        : ON
```

B.3 Logs during association

Table B-5 shows an example of the messages displayed during the association of the sniffer.

Table B-5 Logging during association

```
Association Started

+50022: -69dBm:3e7d6a0267d3: HLM LEN: 12 MAC: 8038d3a522ef7571 TTL: 7c :: MASP: OPC:
MASP_DEVICE_IDENTIFICATION PARAM: cf000ae83d9812fc3e0884187a15be792a
+16960: -87dBm:00025b021253: HLM LEN: 0f MAC: cc1db5243e579e13 TTL: c8 :: MASP: OPC:
MASP_ASSOCIATION_REQUEST PARAM: 46916b350001000000000000000001
+59394: -69dBm:2b4f4129f137: HLM LEN: 12 MAC: 0718d1b00b60dffd TTL: 08 :: MASP: OPC:
MASP_DEVICE_IDENTIFICATION PARAM: 53121a376167b4affab10020bf40efef80
+311573: -81dBm:707c18000566: HLM LEN: 07 MAC: 7140fdac2dcb064e TTL: 78 :: MASP:
OPC: MASP_ASSOCIATION_RESPONSE PARAM: 46916b350102
+316568: -81dBm:13fdd625b3d1: HLM LEN: 0f MAC: 075790f359c5b7a6 TTL: fe :: MASP:
OPC: MASP_PUBLIC_KEY_REQUEST PARAM: 46916b35001624c426c1d891dc00
+127864: -81dBm:1d099f0cd78a: HLM LEN: 0f MAC: f4fb5aa6c7dbb163 TTL: fe :: MASP:
OPC: MASP_PUBLIC_KEY_REQUEST PARAM: 46916b350181eb3b331746ed9c00
+267400: -69dBm:12017b804a6e: HLM LEN: 0f MAC: ccce1250df022eb0 TTL: fd :: MASP:
OPC: MASP_PUBLIC_KEY_REQUEST PARAM: 46916b3502906ea9e43fc0e80100
+159768: -75dBm:3b981dc2ceb1: HLM LEN: 0f MAC: 04f9772169caf922 TTL: fd :: MASP:
OPC: MASP_PUBLIC_KEY_REQUEST PARAM: 46916b350323a8dc054199b3bf00
+564362: -81dBm:707c18000292: HLM LEN: 0f MAC: 3027e16640b400ba TTL: 7a :: MASP:
OPC: MASP_PUBLIC_KEY_RESPONSE PARAM: 46916b350209d9e805083aad7b07
+188379: -81dBm:707c18000292: HLM LEN: 0f MAC: c34732203324b31b TTL: 7a :: MASP:
OPC: MASP_PUBLIC_KEY_RESPONSE PARAM: 46916b350347181e0a139a5c2208
+66948: -87dBm:00025b0447f3: HLM LEN: 0f MAC: cc1db5243e579e13 TTL: a8 :: MASP: OPC:
MASP_ASSOCIATION_REQUEST PARAM: 46916b350001000000000000000001
+137091: -81dBm:0ae7cef12b9f: HLM LEN: 0f MAC: 04f9772169caf922 TTL: f4 :: MASP:
OPC: MASP_PUBLIC_KEY_REQUEST PARAM: 46916b350323a8dc054199b3bf00
+16064: -81dBm:3a5f1ed18411: HLM LEN: 0f MAC: 2cc5c98581e7fed7 TTL: fd :: MASP: OPC:
MASP_PUBLIC_KEY_REQUEST PARAM: 46916b350323a8dc054199b3bf01
+107981: -81dBm:3640af2f41f5: HLM LEN: 0f MAC: 86c5836c2e534185 TTL: fe :: MASP:
OPC: MASP_PUBLIC_KEY_REQUEST PARAM: 46916b3504fae67b604caeb4cc00
+101093: -75dBm:129e179210ed: HLM LEN: 0f MAC: 394ac2e2a10ff2a5 TTL: fd :: MASP:
OPC: MASP_PUBLIC_KEY_REQUEST PARAM: 46916b35056ff6ac73ea6b3cb500
+81975: -81dBm:17e493fda10d: HLM LEN: 0f MAC: 3027e16640b400ba TTL: 72 :: MASP: OPC:
MASP_PUBLIC_KEY_RESPONSE PARAM: 46916b350209d9e805083aad7b07
+51748: -48dBm:1a9cea1c3a7e: HLM LEN: 0f MAC: c34732203324b31b TTL: 72 :: MASP: OPC:
MASP_PUBLIC_KEY_RESPONSE PARAM: 46916b350347181e0a139a5c2208
+280876: -93dBm:00025b021253: HLM LEN: 0f MAC: 04f9772169caf922 TTL: f2 :: MASP:
OPC: MASP_PUBLIC_KEY_REQUEST PARAM: 46916b350323a8dc054199b3bf00
+45373: -81dBm:3a144b028268: HLM LEN: 0e MAC: 2ef9d76955f27733 TTL: fe :: MASP: OPC:
MASP_CONFIRMATION_REQUEST PARAM: 46916b356bb2765dbbb5ec1600
+199401: -78dBm:13f89825edaa: HLM LEN: 0e MAC: 15c046d810c9a536 TTL: fe :: MASP:
OPC: MASP_RANDOM_REQUEST PARAM: 46916b3541b4116679069d8000
+210517: -81dBm:24408e0eb6c6: HLM LEN: 08 MAC: 4a4558455ee064f8 TTL: fe :: MASP:
OPC: MASP_ID_DISTRIBUTION PARAM: 46916b352db200
+3369315: -81dBm:3142a731ee2f: HLM LEN: 12 MAC: 7de0a8faf4b1c9ce TTL: 7c :: MASP:
OPC: MASP_DEVICE_IDENTIFICATION PARAM: a1065f08e6cc4a093e48e8a3ff042b1800
+130282: -81dBm:06be2fc36cf3: HLM LEN: 0f MAC: bab7aedda25dc4a2 TTL: fd :: MASP:
OPC: MASP_KEY_DISTRIBUTION PARAM: 46916b35004e4cbeeada8de2ef00
+70931: -81dBm:35ab84a2d21f: HLM LEN: 0f MAC: d3080b1d10ddc254 TTL: fd :: MASP: OPC:
MASP_KEY_DISTRIBUTION PARAM: 46916b35019981bd9f1191c31e00
Association Complete
```

B.4 Logs post association

Table B-6 gives an example of the sniffer message logs post association.

Table B-6 Logs post association

```
Listening On Channel(s) : 37 38 39

Setting Message Log Filter:
Unknown (MAC_FAIL): ON
MASP              : ON
MCP               : ON
Duplicates        : ON

+9190: -75dBm:2eec4ec20411:(D) HLM LEN: 12 MAC: 9ed119727cdc4fe4 TTL: 09 :: MASP:
OPC: MASP_DEVICE_APPEARANCE PARAM: 252c734f6010004e6172656e6472610087
+8367: -51dBm:1bac892a0bac: HLM LEN: 0f MAC: a03a84825018b1e3 TTL: 31 :: MCP: SEQ:
00162c79 SRC: 8002 DST: 0000 OPC: MCP_WATCHDOG_MESSAGE PARAM: 0063863aff52a8
+5147: -87dBm:707c180005e8:(D) HLM LEN: 12 MAC: 50f81945762b1a41 TTL: 03 :: MASP:
OPC: MASP_DEVICE_APPEARANCE PARAM: c27ca47e60100063686574616e0000000b
+8682: -87dBm:0d1159334646:(D) HLM LEN: 12 MAC: 23604af5e7c305d4 TTL: 7a :: MASP:
OPC: MASP_DEVICE_IDENTIFICATION PARAM: 4c011e18c2881e1f2a191d88e3dd4e1808
+11414: -75dBm:051d3d875341: HLM LEN: 10 MAC: 73fed08243cbef41 TTL: 30 :: MACFAIL:
PAYLOAD: c5da6d058065d9912fcc0d20e407c7c5
+9124: -48dBm:08e7ec3f39c1:(D) HLM LEN: 10 MAC: 5fdd72e63bc7481e TTL: 30 :: MACFAIL:
PAYLOAD: c6da6d058060352107e3f8074df99ba9
+9289: -81dBm:24549714a62d:(D) HLM LEN: 10 MAC: 298280a7e79e0b7d TTL: 2e :: MACFAIL:
PAYLOAD: 12f47a048007af256fff71d8af499ac0
```

Document references

Document	Reference
Bluetooth Core Specification Version 4.1	www.bluetooth.org
µEnergy xIDE User Guide	CS-212742-UG
CSRmesh Android Controller Application Note	CS-347470-AN
CSRmesh iOS Controller Application Note	CS-347471-AN
CSRmesh Mobile Application User Guide	CS-347469-UG
CSRmesh 2.1 API Guide	www.csrsupport.com
GATT Database Generator	CS-219225-UG
CSR µEnergy Bluetooth USB Dongle Driver Installation User Guide	CS-315781-UG
Service Characteristics And Descriptions	developer.bluetooth.org

Terms and definitions

AC	Authorisation Code
API	Application Programmer's Interface
BLE	Bluetooth Low Energy (now known as Bluetooth Smart)
Qualcomm® BlueCore™	Group term for CSR's range of Bluetooth wireless technology chips
Bluetooth	Set of technologies providing audio and data transfer over short-range radio connections
CS	Configuration Store
CSR	Cambridge Silicon Radio
CSRmesh	A CSR protocol that enables peer-to-peer-like networking of Bluetooth Smart devices
e.g.	*exempli gratia*, for example
EEPROM	Electrically Erasable Programmable Read Only Memory
etc.	*et cetera*, and the rest, and so forth
GAP	Generic Access Profile
GATT	Generic Attribute Profile
i.e.	*Id est*, that is
I^2C	Inter-Integrated Circuit
IOT	Internet of Things
IRK	Identity Resolving Key
LED	Light Emitting Diode
LM	Link Manager
MCP	Mesh Control Protocol
MASP	Mesh Association Protocol
MTL	Message Transport Layer
NVM	Non Volatile Memory
OTA	Over The Air
PCB	Printed Circuit Board
PIO	Programmable Input Output
PWM	Pulse Width Modulation
QR-Code	Quick Response Code
QTI	Qualcomm® Technologies Inc.
QTIL	Qualcomm Technologies International Ltd
Rx	Receiver
SDK	Software Development Kit
SMP	Security Manager Protocol
SPI	Serial Peripheral Interface
Tx	Transmit
UART	Universal Asynchronous Receiver Transmitter
USB	Universal Serial Bus
UUID	Universally Unique Identifier