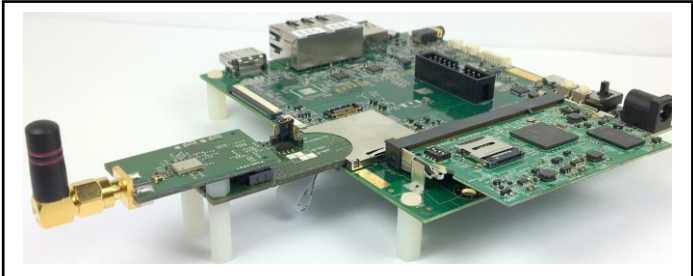


**Murata Wi-Fi/BT  
Solution for i.MX**

**Quick Start Guide  
(Linux)**



**Revision History**

Revision	Date	Author	Change Description
1.0	Sept 1, 2015	S Kerr, G Mohiuddin	Initial Release
1.1	Sept 6, 2015	S Kerr	Removed software compile/build dependency. User can bring up NXP platform by downloading necessary files before flashing bootable SD card. Refer to Linux User Guide on software build procedures.
2.0	Nov 7, 2015	S Kerr	Modified Murata Wi-Fi/BT EVK definition. This simplifies bring-up on NXP i.MX6 Platforms. Incorporated changes for i.MX6UL 3.14.38 GA BSP Release.
4.0	Feb 14, 2017	S Kerr	Renamed document to "Murata Wi-Fi/BT Solution for i.MX Quick Start Guide (Linux)". Incorporated changes for NXP Linux 4.1.15_2.0.0 GA BSP release. Modified NXP Linux 3.14.52_1.1.0 GA BSP release to build in bcmhdh WLAN driver, thereby matching 4.1.15_2.0.0 configuration. Added instructions for Murata binary patch release which addresses errata/features on both releases. Provided support for new i.MX 7Dual SDB, i.MX 6ULL EVK, and Murata Type 1CK. Expanded WLAN test verification section.

This page intentionally left blank.

# Table of Contents

REVISION HISTORY .....	1
TABLE OF CONTENTS .....	3
<b>1 INTRODUCTION .....</b>	<b>5</b>
1.1 Overview of Murata Wi-Fi/BT Hardware Solution for i.MX.....	5
1.2 Overview of Software Considerations.....	10
1.2.1 WLAN Driver Modifications on Linux 3.14.52_1.1.0.....	10
1.2.2 Out-Of-Band IRQ on i.MX 6UL/6ULL EVK's .....	10
1.2.3 Murata Support for Edge-Sensitive Interrupts in OOB IRQ Configuration .....	11
1.2.4 Murata SN8000 Support.....	11
1.2.5 Latest i.MX Hardware Support on Linux 4.1.15_2.0.0 Release .....	11
1.3 Acronyms .....	11
1.4 References.....	12
1.4.1 Murata Linux User Manual .....	12
1.4.2 Murata Hardware User Manual .....	12
1.4.3 Murata i.MX Linux Quick Start Binary Patches .....	12
<b>2 MURATA WI-FI/BT BRING-UP ON I.MX6 PLATFORMS .....</b>	<b>13</b>
2.1 Connecting to i.MX 6SoloX SDB.....	14
2.2 Connecting to i.MX 6SoloLite EVK .....	15
2.3 Connecting to i.MX 6Q/DL SDB/SDP or i.MX 6 QP SDB .....	16
2.3.1 Specific Hardware Considerations for i.MX 6Quad/DualLite SDB/SDP.....	16
2.3.2 Specific Hardware Considerations for i.MX 6QP SDB .....	16
2.3.3 Murata Wi-Fi/BT EVK Bring-Up on i.MX 6Q/DL SDB/SDP or i.MX 6 QP SDB .....	16
2.4 Connecting to i.MX 6UltraLite EVK or i.MX 6ULL EVK.....	18
<b>3 BRINGING UP WI-FI/BT ON I.MX 7DUAL SABRE DEVELOPMENT BOARD .....</b>	<b>19</b>
<b>4 PREPARING BOOTABLE SD CARD FOR I.MX6 WITH MURATA WI-FI/BT EVK .....</b>	<b>20</b>
4.1 Getting Signed Up to Access Support Resources .....	21
4.2 Downloading i.MX6/7 Image Files to Flash SD Card.....	21
4.2.1 Linux PC Steps to Extract “*.sdcard” File .....	22
4.2.2 Windows PC Steps to Extract “*.sdcard” File .....	22
4.3 Flashing SD Card.....	22
4.3.1 Linux PC Steps to Flash SD Card .....	22
4.3.2 Windows PC Steps to Flash SD Card.....	23
4.4 Murata Modifications to the Default NXP BSP Release .....	24
4.4.1 Murata Quick Start Binary Patch Modifications to Linux 3.14.52_1.1.0 Release.....	24
4.4.2 Murata Quick Start Binary Patch Modifications to Linux 4.1.15_2.0.0 Release.....	24
4.5 WLAN Firmware and NVRAM Update Considerations.....	25
4.6 Specific Kernel / DTB / NVRAM / Firmware files for each Configuration.....	25
<b>5 TEST/VERIFICATION OF WI-FI AND BLUETOOTH.....</b>	<b>28</b>
5.1 Wi-Fi Interface Test/Verification .....	30
5.1.1 Useful Environment Setup on NXP Linux .....	30
5.1.2 Bringing Up Wi-Fi Interface .....	30
5.1.3 STA/Client Mode: Scan for Visible Access Points .....	32
5.1.4 STA/Client Mode: Connecting to Unsecured Access Point or Wireless Router .....	36
5.1.5 STA/Client Mode: Connecting to Secured Access Point or Wireless Router (WPA2-PSK) .....	38
5.1.6 STA/Client Mode: Basic WLAN Connectivity Testing .....	42
5.1.7 Wi-Fi Direct Testing .....	43
5.1.8 Soft AP or Wi-Fi Hot Spot Testing.....	45
5.1.9 WLAN Manufacturing or RF Testing .....	46
5.2 Bluetooth Interface Test/Verification .....	50
5.2.1 i.MX 7Dual SDB.....	51

5.2.2	i.MX 6Quad(Plus)/DualLite SDB/SDP .....	51
5.2.3	i.MX 6SoloX SDB .....	52
5.2.4	i.MX 6SoloLite EVK .....	52
5.2.5	i.MX 6UltraLite EVK (Linux 3.14.52).....	52
5.2.6	i.MX 6UL/ULL EVK (Linux 4.1.15).....	52
<b>6</b>	<b>VERIFYING ADAPTER BOARDS .....</b>	<b>53</b>
6.1	Murata i.MX InterConnect V1 Adapter .....	53
6.2	Murata i.MX InterConnect V2 Adapter .....	54
<b>7</b>	<b>TECHNICAL SUPPORT CONTACT.....</b>	<b>56</b>
<b>8</b>	<b>ADDITIONAL USEFUL LINKS .....</b>	<b>57</b>

## LIST OF TABLES

Table 1:	Murata Wi-Fi/BT EVK (for i.MX6) Contents .....	6
Table 2:	Murata Wi-Fi/BT EVK's Supported .....	7
Table 3:	Murata i.MX InterConnect Adapter Selection.....	9
Table 4:	Acronyms used in Quick Start Guide .....	11
Table 5:	i.MX6/7 Platforms' SD Card Image Filenames .....	21
Table 6:	Specific Files for Each i.MX Platform/Linux Kernel Version/Murata EVK Configuration .....	27
Table 7:	Embedded Wi-Fi/Bluetooth Files .....	29
Table 8:	Murata Module to Firmware/NVRAM Mapping .....	32
Table 9:	GPIO and UART Settings for Bluetooth Tests.....	50
Table 10:	List of Support Resources.....	56
Table 11:	Additional Useful Links.....	57

## LIST OF FIGURES

Figure 1:	Murata IMX6 Interconnect Kit Interfaces .....	7
Figure 2:	i.MX 7Dual SDB Block Diagram .....	8
Figure 3:	Murata Type 1CK Interconnect Block Diagram .....	13
Figure 4:	Murata Type 1CK Interconnect to NXP i.MX 6SoloX .....	13
Figure 5:	i.MX 6SoloX SDB with V1 Adapter and Type ZP EVB .....	14
Figure 6:	i.MX 6SoloLite EVK with V1 Adapter and Type 1DX EVB.....	15
Figure 7:	i.MX 6Quad/DualLite SDB (Inverted) with V2 Adapter and Type ZP EVB .....	17
Figure 8:	i.MX 6UltraLite EVK with V2 Adapter and Type 1DX EVB .....	18
Figure 9:	i.MX 7Dual SDB with Murata Type ZP.....	19
Figure 10:	USB to SD Card Reader/Writer Adapter.....	22
Figure 11:	Murata i.MX InterConnect V1 Adapter – Top .....	53
Figure 12:	Murata i.MX InterConnect V1 Adapter – Bottom #1 .....	54
Figure 13:	Murata i.MX InterConnect V1 Adapter – Bottom #2 .....	54
Figure 14:	Murata i.MX InterConnect Adapter V2 Adapter – Top.....	55
Figure 15:	Murata i.MX InterConnect Adapter V2 Adapter - Bottom .....	55

# 1 Introduction

Murata has partnered with NXP Semiconductors N.V. and Cypress Semiconductor Corporation to offer a complete Wi-Fi and Bluetooth connectivity environment for building world class Internet-connected products. The Murata Connectivity Modules enable developers to minimize the development time and effort for connectivity function implementation. This guide provides details for getting started with Wi-Fi and Bluetooth on iMX6/7 platforms using [NXP Linux 3.14.52 1.1.0<sup>1</sup> and 4.1.15 2.0.0 GA BSPs](#). The NXP i.MX BSP releases support a baseline configuration which is tested against the Murata ZP module. This and other Murata documents<sup>2</sup> provide additional details necessary in supporting the additional Murata modules summarized in **Table 2**. These additional details include the following:

- Software changes necessary to get past functional issues on specific i.MX platforms with non-ZP EVK's.
- Software changes desired for optimization of the Murata Wi-Fi/BT (including ZP EVK) solution on i.MX platforms.
- Hardware changes are referenced for getting the Murata Wi-Fi/BT EVK fully functional on an i.MX platform or when certain optimizations are desirable.

## 1.1 Overview of Murata Wi-Fi/BT Hardware Solution for i.MX

The NXP Linux 3.14.52 release was the first official GA BSP that integrated support for the Murata Wi-Fi/BT solution. The hardware solution consisted of the Murata Wi-Fi/BT EVK for i.MX6 platforms. With the newer 4.1.15 kernel NXP has introduced the i.MX 7Dual SDB and the i.MX 6ULL EVK. **Table 5** shows the kernel version support for the different i.MX platforms.

For the NXP i.MX6 platforms, the only solution is the Murata Wi-Fi/BT EVK: contents are listed in **Table 1**. By contrast the i.MX 7Dual SDB has the Murata Type ZP module soldered down: also documented in this quick start guide. One notable NXP i.MX platform not documented in this Quick Start Guide is the WaRP7 which is a “community” supported platform – more information [here](#). The WaRP7 has the i.MX7 processor mated with a Murata Type 1DX module. **Table 1** shows the contents for the Murata Wi-Fi/BT EVK. The specific kit contents differ based on module. Some important notes:

- Type 1CK includes interposer board that connects Type 1CK module (finished product) to the V1/V2 Adapter. Type 1CK kit does not come with external antenna option: **chip antenna already on module**. The 1CK module has a built-in test connector for conducted testing<sup>3</sup>.
- SN8000CMK includes two EVB's: one with UFL connector, other with chip antenna option.
- All kits include V1/V2 adapters necessary for connecting different i.MX6 platforms.
- Type ZP/1BW kits include dual-band antenna; Type 1DX/1FX kits include single-band.


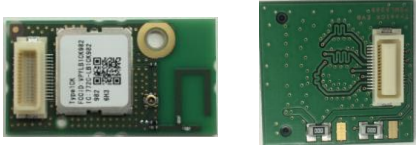

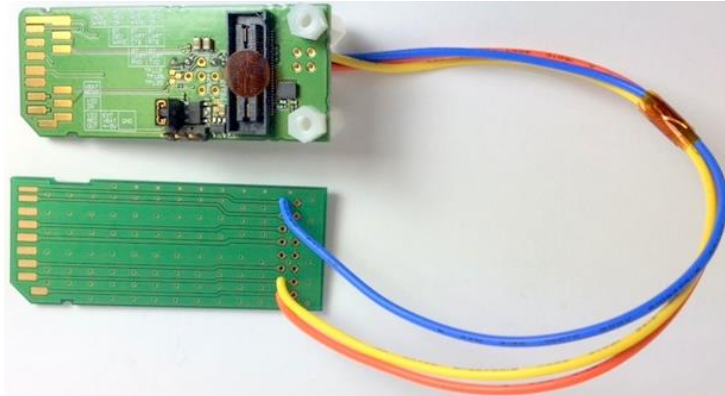
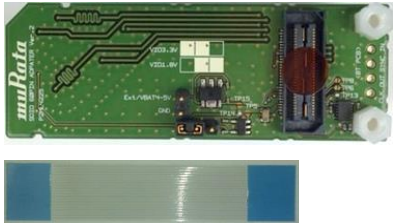


---

<sup>1</sup> The only “GA BSP” released for 3.14.52 is labelled “Linux 3.14.52” on NXP website. However the last kernel version released for 3.14 which Murata supports is “Linux 3.14.52\_1.1.0”.

<sup>2</sup> Some of the “other Murata documents” include the Linux User Manual and Hardware User Manual which are referenced throughout the Quick Start Guide.

<sup>3</sup> Relevant Murata part numbers are: 1CK test connector is MM8030-2610; test cable for conducted testing is MXHQ87WJ3000.

**Table 1: Murata Wi-Fi/BT EVK (for i.MX6) Contents**

Part Number	Picture of Contents	Description of Contents
1		<p>Type ZP/1BW/1DX/1FX Murata Wi-Fi/BT EVB. Type ZP is pictured. See <b>Table 2</b> for specific part numbers.</p>
		<p>Type 1CK EVB includes an additional interposer board which connects to V1/V2 Adapter. Type 1CK module (<a href="#">LBEE5ZZ1CK</a>) is FCC certified with built-in antenna and test connector.</p>
		<p>SN8000CMK includes two EVB's. One SN8000 configured with UFL connector (<a href="#">88-00153-02</a>), other SN8000 configured with chip antenna (<a href="#">88-00153-00</a>). SN8000 is FCC/CE certified.</p>
2		<p><b>Murata i.MX InterConnect V1:</b> SD pins (DAT0..7) provide both Wi-Fi SDIO and Bluetooth UART connection. Wired SD Card Extender connects control signals: WL_REG_ON, BT_REG_ON, and WL_HOST_WAKE.</p>
3		<p><b>Murata i.MX InterConnect V2:</b> SD pins provide Wi-Fi SDIO; ribbon cable connection provides Bluetooth UART and control signals: WL_REG_ON, BT_REG_ON, and WL_HOST_WAKE. Flexible 50 mm ribbon cable included.</p>
4		<p>Type ZP/1BW EVK's have 2.4/5.0 GHz Whip/Tilt SMA Antenna (for dual-band Wi-Fi) <b>OR</b> Type 1DX/1FX EVK's have 2.4 GHz Whip SMA Antenna (for single band Wi-Fi).</p>
		<p>SN8000CMK (SN8000 Carrier Module Kit) includes specific antennas which connect directly to UFL connector (2.4 GHz only).</p>

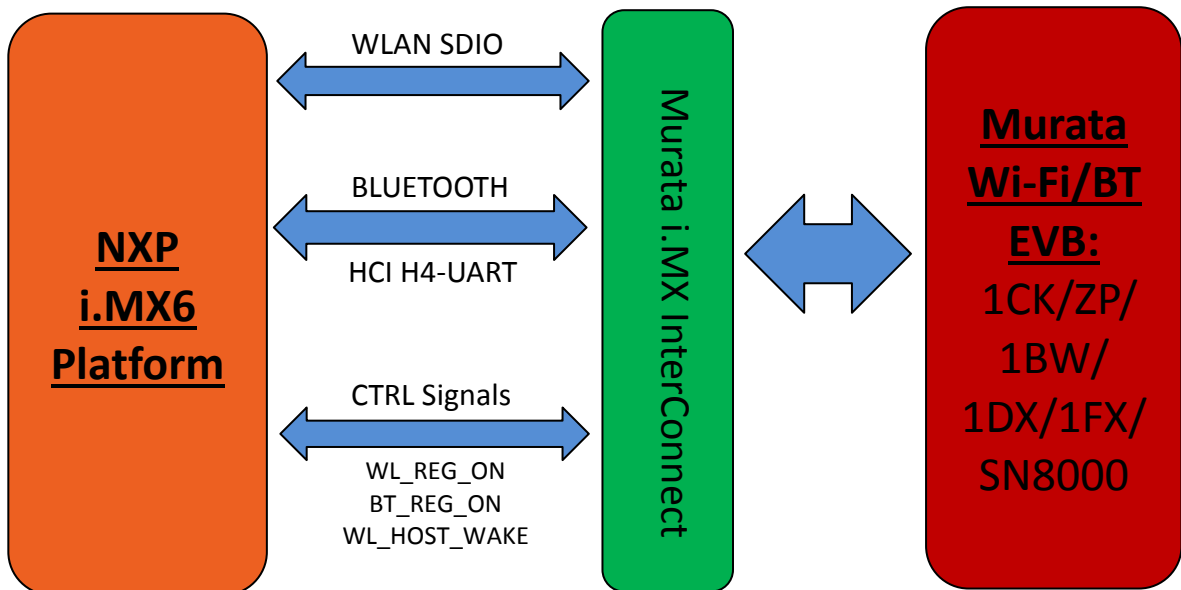
Murata Wi-Fi/BT EVK's supported on NXP i.MX6 Platforms are listed in **Table 2**. Six (6) different modules are available. If you are having difficulty obtaining the desired Murata EVK, contact [Murata](#) for additional support. Alternatively click on "Order part number" hyperlinks listed in **Table 2** to bring up the Murata module webpage. Now click on "purchase" tab, and scroll down to list currently available kits.

**Table 2: Murata Wi-Fi/BT EVK's Supported**

Part	Description	Order part number	Wi-Fi	Bluetooth
Type 1CK EVK+	802.11b/g/n/ac and BT EVK for i.MX6 FCC certified	<a href="#">LBEE5ZZ1CK-TEMP-DS-SD</a>	b/g/n/ac	Yes
Type ZP EVK+	802.11b/g/n/ac and BT EVK for i.MX6	<a href="#">LBEH5HMZPC-TEMP-DS-SD</a>	b/g/n/ac	Yes
Type 1BW EVK+	802.11a/b/g/n and BT EVK for i.MX6	<a href="#">LBEH5DU1BW-TEMP-DS-SD</a>	a/b/g/n	Yes
Type 1DX EVK+	802.11b/g/n and BT EVK for i.MX6 FCC "Reference" certified	<a href="#">LBEE5KL1DX-TEMP-DS-SD</a>	b/g/n	Yes
Type 1FX EVK+	802.11b/g/n EVK for i.MX6 FCC "Reference" certified	<a href="#">LBWA1KL1FX-TEMP-DS-SD</a>	b/g/n	No
SN8000CMK	802.11b/g/n EVK for i.MX6 FCC/CE certified, industrial	<a href="#">88-00153-90</a>	b/g/n	No

Connection Diagram for the Murata Interconnect kit is provided in **Figure 1**. Murata Wi-Fi/BT kit for i.MX6 enables this configuration by providing two custom-built Adapter boards. Two adapter boards are provided in each kit allowing the user to bring up the Wi-Fi/Bluetooth interfaces in the easiest manner possible.

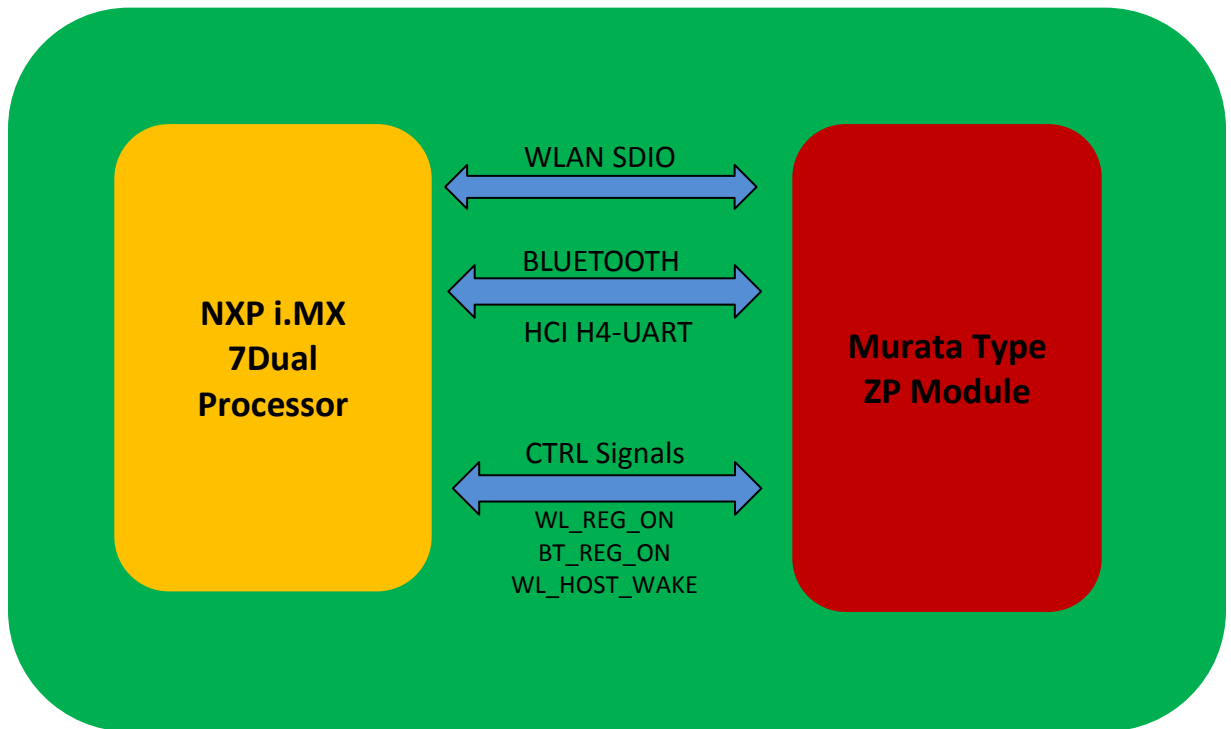
**Figure 1: Murata IMX6 Interconnect Kit Interfaces**



**Note: 3<sup>rd</sup> party NXP i.MX6 EVK's are \*not\* supported.**

In addition to the i.MX6 InterConnect Kit (Murata Wi-Fi/BT EVK), there is also support provided on the [i.MX 7Dual SDB](#). This platform has the Murata ZP module soldered down on the board. Both Wi-Fi and Bluetooth interfaces are supported on this platform. Unlike the InterConnect solution on i.MX6 platforms, there is no inherent “legacy” restriction on the i.MX7 platform which can limit SDIO throughput. The Murata ZP module supports a very high throughput (SDIO 3.0 mode – UHS) over SDIO bus. Refer to the NXP i.MX7 schematics for specifics on the Wi-Fi/BT interconnect: download package [here](#)<sup>4</sup>. **Figure 2** below shows a simplified block diagram for the i.MX 7Dual SDB. Although the UHS mode on SDIO bus is enabled on i.MX 7D SDB, the Wi-Fi throughput performance can still be optimized, refer to the [Murata Linux User Manual](#) for more details.

**Figure 2: i.MX 7Dual SDB Block Diagram**



**Table 3** lists the official NXP branded i.MX6/7 Reference Platforms and which Murata Adapter Version should be used.

⇒ **NOTE: Murata strongly recommends the recommended Adapter configuration. Even though an alternate adapter may work (with DTS file changes), that alternate configuration is not supported by NXP or Murata.**

<sup>4</sup> For Wi-Fi/BT schematics on i.MX 7Dual SDB, refer to page 13 of sch-28590\_i.mx7d\_saber\_rev\_2.pdf document.



As **Table 3** indicates, currently seven (7) NXP i.MX6 Platforms are supported. The i.MX 7Dual SDB (with onboard Murata Type ZP module) is listed here to highlight that it does not support the Murata V1/V2 InterConnect Adapter (i.e. for customers wanting to evaluate another Murata module other than Type ZP). The SD card slot on the i.MX7 platform is required for booting the platform<sup>5</sup>.

There are three listings for essentially a very similar platform: i.MX 6QuadPlus SDB, i.MX 6Quad/DualLite SDB, and i.MX 6Quad/DualLite SDP. The i.MX 6QuadPlus SDB has a modified schematic from the (essentially identical) i.MX 6Quad/DualLite SDB/SDP platforms. Although the Murata Wi-Fi/BT EVK is designed to be “plug ‘n play”, **rework is required** for the i.MX 6Quad/DualLite SDB/SDP platforms. Depending on the revision, rework **may be required** for the i.MX 6QuadPlus SDB. This rework connects the Bluetooth UART and Wi-Fi/BT control signals (WL\_REG\_ON, BT\_REG\_ON, and WL\_HOST\_WAKE). Refer to the [Hardware User Manual](#) for more details. Bring-up on all three platforms will be covered in **Section 2.3**.

Lastly the i.MX 6UltraLite EVK and i.MX 6ULL EVK share the same baseboard. As such the Murata Wi-Fi/BT EVK interconnect is identical. Both platforms will be covered in **Section 2.4**.

**Table 3: Murata i.MX InterConnect Adapter Selection**

NXP i.MX Platform	Adapter	Interrupt Configuration	Notes
<a href="#">i.MX 7Dual SDB</a>	N/A	OOB IRQ	i.MX 7Dual SDB integrates Murata Type ZP Module.
<a href="#">i.MX 6QuadPlus SDB</a>	<a href="#">V2</a>	OOB IRQ	Revision B of i.MX 6QP SDB populates the necessary resistors for connecting BT UART and Wi-Fi/BT control signals. If your board has revision earlier than B (i.e. A2) then you will have to populate the necessary resistors. Refer to <a href="#">Hardware User Manual</a> for rework instructions.
<a href="#">i.MX6Quad/DualLite SDB</a>	<a href="#">V2</a>	OOB IRQ	Refer to <a href="#">Hardware User Manual</a> for rework instructions.
<a href="#">i.MX 6Quad/DualLite SDP</a>			
<a href="#">i.MX 6SoloX SDB</a>	<a href="#">V1</a>	OOB IRQ	
<a href="#">i.MX 6SoloLite EVK</a>	<a href="#">V1</a>	OOB IRQ	NXP DTB file only configures Wi-Fi. Hardware supports Bluetooth but a modified DTB file is required – provided by Murata in Quick Start Binary Patch.
<a href="#">i.MX 6UltraLite EVK</a>	<a href="#">V2</a>	SDIO In-Band	Rework necessary to connect optional WL_HOST_WAKE for OOB IRQ. Refer to <a href="#">Hardware User Manual</a> for rework instructions. Regarding software mods for OOB IRQ, optional DTB binary is included in Murata Quick Start Binary patch.
<a href="#">i.MX 6ULL EVK</a>			

<sup>5</sup> The i.MX 7Dual SDB can be configured to flash u-boot and the kernel to the onboard QSPI-NOR. The root file system can then be NFS-mounted. However the steps at arriving at this configuration (to free up SD card slot) are complicated (including hardware rework). As such Murata does not support this configuration.

## 1.2 Overview of Software Considerations

There are some important software considerations when using the Murata Wi-Fi/BT solution on NXP i.MX6/7 platforms. We review the key topics in this section.

### 1.2.1 WLAN Driver Modifications on Linux 3.14.52\_1.1.0

With the latest Linux 4.1.15\_2.0.0 release, NXP has modified the WLAN driver (bcmhd) to be built into the kernel (not loadable module as previously done on 3.14.52\_1.1.0 release) and to support Out-Of-Band interrupts (OOB IRQ). The bcmhd driver changes have resulted in a fundamental shift in how the end user brings up the WLAN interface. To minimize confusion between necessary steps needed to run either release, Murata has provided a patched 3.14.52\_1.1.0 release which matches the bcmhd driver integration (built into kernel) on the Linux 4.1.15\_2.0.0 drop.

The modified 3.14.52\_1.1.0 release builds the WLAN driver (bcmhd) into the kernel and supports OOB IRQ interrupts. Additional critical WLAN patches are also included in the revised kernel (bcmhd driver integrated) release. Previously the Linux 3.14.52\_1.1.0 used a loadable bcmhd driver module (“bcmhd.ko”) and only supported SDIO in-band interrupts.

Due to the WLAN (“bcmhd”) driver changes (going from Linux 3.14.52 to 4.1.15\_2.0.0 releases), there is less flexibility on configuring OOB IRQ with the Linux 3.14.52\_1.1.0 release. As such a different kernel (“bcmhd” driver) needs to be used when running in either SDIO in-band or Out-Of-Band interrupt mode. For more information, refer to the [Murata i.MX L3.14.52\\_1.1.0 Quick Start Binary Patch](#).

### 1.2.2 Out-Of-Band IRQ on i.MX 6UL/6ULL EVK's

The default interrupt configuration for the Linux 4.1.15\_2.0.0 and (modified) Linux 3.14.52\_1.1.0 releases is to use OOB IRQ. However, there are two exceptions to this default interrupt configuration: the i.MX 6UL and i.MX 6ULL EVK's. SDIO in-band interrupts have been configured on these platforms due to the default i.MX hardware implementation (using modified kernel/bcmhd on 3.14.52\_1.1.0 and default NXP DTB file for 4.1.15\_2.0.0).

If the user prefers to run OOB IRQ on either of these two platforms, the necessary rework is quite straightforward: refer to the [Hardware User Manual](#) for specific instructions<sup>6</sup>. This document provides the necessary software steps to run either interrupt configuration. For the Linux 4.1.15\_2.0.0 release, a different DTB file is required for OOB IRQ. For Linux 3.14.52\_1.1.0, a different DTB file and kernel (bcmhd driver) is required.

**Note: For OOB IRQ configuration on i.MX6UL/ULL EVK, the second (of two) Ethernet ports is disabled due to hardware conflict (documented in Hardware User Manual steps).**

---

<sup>6</sup> Both the i.MX6UL and i.MX 6ULL EVK's share the same baseboard. It is this baseboard that requires the hardware modification to enable OOB IRQ: connect WL\_HOST\_WAKE. This is a straightforward modification – accomplished by moving one resistor.

### 1.2.3 Murata Support for Edge-Sensitive Interrupts in OOB IRQ Configuration

When running OOB IRQ's, the default configuration is for level-sensitive interrupts. However two of the Murata modules (Type 1BW and SN8000) generate edge-sensitive interrupts over WL\_HOST\_WAKE line. To support both OOB IRQ configurations, Murata provides a modified kernel (bcmhdh driver integrated) which supports edge-sensitive interrupts. For more information, refer to the [Murata i.MX Linux Quick Start Binary Patches](#).

### 1.2.4 Murata SN8000 Support

Both Linux 3.14.52\_1.1.0 and 4.1.15\_2.0.0 releases don't support SN8000 (Wi-Fi only) EVB on the i.MX 6UltraLite/ULL EVK's<sup>7</sup>. This can only be corrected by using Murata-modified i.MX 6UL/6ULL DTB files. For more information, refer to the [Murata i.MX Linux Quick Start Binary Patches](#).

### 1.2.5 Latest i.MX Hardware Support on Linux 4.1.15\_2.0.0 Release

The most recent i.MX6 and i.MX7 platforms are only supported on the 4.1.15\_2.0.0 release: i.MX 6ULL EVK and i.MX 7D SDB. Refer to **Table 6** for more specifics.

## 1.3 Acronyms

**Table 4: Acronyms used in Quick Start Guide**

Acronym	Meaning
API	Application Programming Interface
DTB	Device Tree Blob: Kernel reads in at boot time for configuration.
EVB	Evaluation Board (Murata module on custom PCB)
EVK	Evaluation Kit (includes EVB + Adapter)
FW	Firmware
GPIO	General Purpose Input/Output
NVRAM	Calibration file for WLAN.
PC	Personal Computer
SW	Software
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus

---

<sup>7</sup> The problem with SN8000 is due to the BT\_REG\_ON being connected through to the GPIO0 pin on the module which sets SPI/SDIO mode (SN8000 EVB schematic [here](#)). A high level on this line configures the SN8000 module for SPI mode which is not compatible with the i.MX interface and software. The default configuration on i.MX 6UL/6ULL EVK's is to drive BT\_REG\_ON high at kernel boot. Although a GPIO write can fix this situation on Linux 3.14.52\_1.1.0 release (bcmhdh module loaded later), the Linux 4.1.15\_2.0.0 release attempts to bring up bcmhdh driver at kernel boot (driver built-into kernel). End result is the default Linux 4.1.15\_2.0.0 fails on bringing up bcmhdh driver when SN8000 is connected on i.MX 6UL/ULL EVK's.

## 1.4 References

### 1.4.1 Murata Linux User Manual

Murata Wi-Fi/BT Solution for i.MX6 Linux User Manual 4.0, “Murata Wi-Fi & BT Solution for i.MX Linux User Manual 4.0.pdf”.

This manual describes all steps necessary to build the file system, kernel, DTB files, and WLAN “bcmhdh” driver necessary for supporting NXP i.MX6/7 Platforms and the Murata Wi-Fi/BT EVK.

The Murata Linux User Manual is available on “My Murata” support portal [here](#).

### 1.4.2 Murata Hardware User Manual

Murata Wi-Fi/BT Solution for i.MX Hardware User Manual 2.0, “Murata Wi-Fi & BT Solution for i.MX Hardware User Manual 2.0.pdf”.

This manual details the Murata Wi-Fi/BT EVK InterConnect Adapter hardware. All interface signals to the NXP i.MX6 Platforms are described. Specifics on interfacing each i.MX6 Platform to Murata Wi-Fi/BT EVK are provided. The Wi-Fi/BT interface on i.MX 7Dual SDB is also detailed.

The Murata Hardware User Manual is available on “My Murata” support portal [here](#).

### 1.4.3 Murata i.MX Linux Quick Start Binary Patches

Patches for both Linux 3.14.52\_1.1.0 and 4.1.15 are provided on the [NXP Murata i.MX Support Portal](#). These binary patches address specific errata, functionality issues, and necessary enhancements. Login credentials are required for this support portal. Refer to [Murata i.MX Landing Page](#) for specifics on accessing this portal.

#### 1.4.3.1 Murata i.MX L3.14.52\_1.1.0 Quick Start Binary Patch

This binary patch release includes specific fixes for the baseline NXP 3.14.52\_1.1.0 release. Refer to **Section 4.4.1** for more details.

Access [this link](#) to download the 3.14.52\_1.1.0 binary patch release.

#### 1.4.3.2 Murata i.MX L4.1.15\_2.0.0 Quick Start Binary Patch

This binary patch release includes specific fixes for the baseline NXP 4.1.15 release. Refer to **Section 4.4.2** for more details.

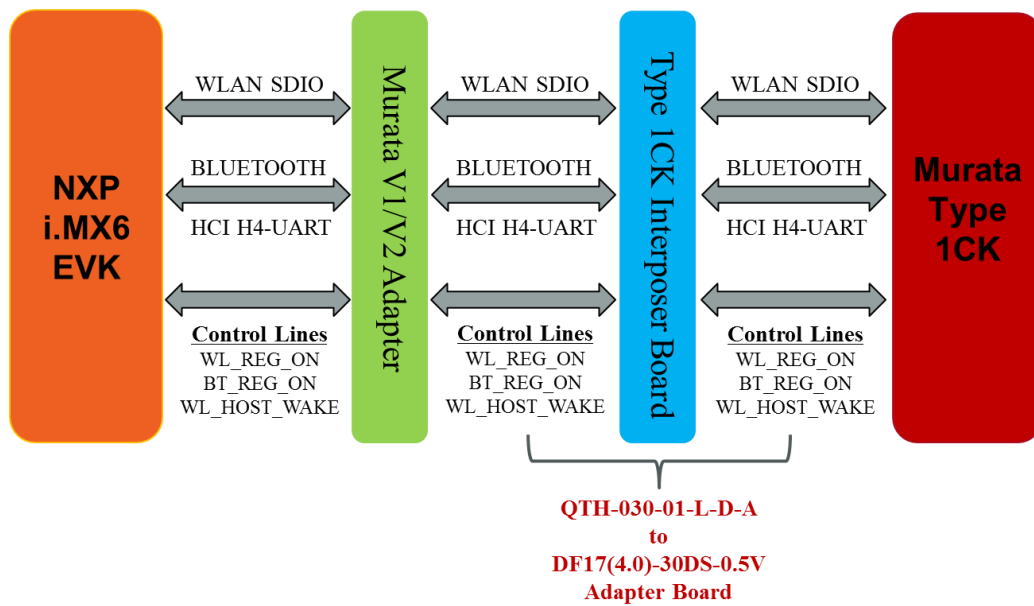
Access [this link](#) to download the 4.1.15\_2.0.0 binary patch release.

## 2 Murata Wi-Fi/BT Bring-Up on i.MX6 Platforms

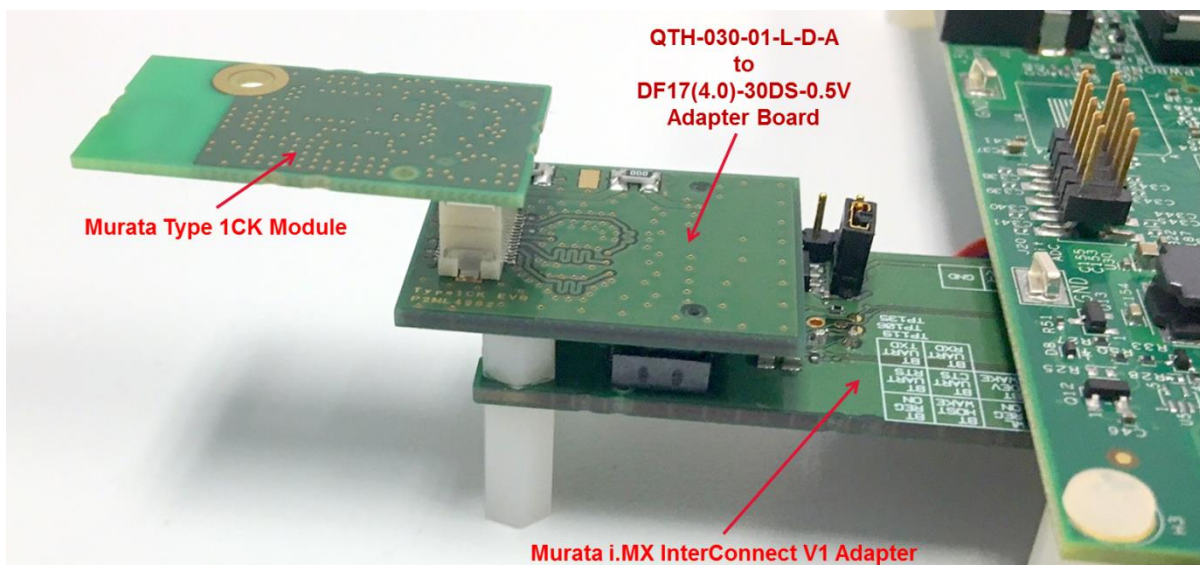
Any of the Murata Wi-Fi/BT EVK's listed in **Table 2** can be connected to the i.MX6 Platforms. The following sub-sections details steps for bringing up Wi-Fi/BT on the four major variants of the i.MX6 Platform. The specific steps described only vary slightly for the Murata Type ZP, 1BW, 1DX, 1FX, and SN8000 modules given the variances in EVB used and antenna.

Murata Type 1CK EVK provides an additional step with connecting the interposer board. **Figure 3** shows a block diagram of the Type 1CK interconnect. Refer to **Figure 4** for the **correct orientation** of Type 1CK module and interposer board.

**Figure 3: Murata Type 1CK Interconnect Block Diagram**



**Figure 4: Murata Type 1CK Interconnect to NXP i.MX 6SoloX**



## 2.1 Connecting to i.MX 6SoloX SDB

Referring to **Table 3**, the V1 Adapter is the correct adapter for this platform. No rework is required on i.MX Platform, Murata Wi-Fi/BT EVB is oriented right-side up, and it provides both Wi-Fi and BT functionality via SD3/SD2 slots: see **Figure 5** below.

- [1] Ensure no power is applied to i.MX 6SoloX SDB. Connect J16 micro-USB port to PC and start terminal emulator: “minicom” on Linux or “tera term” on Windows. Set port to 115200-N-8-1.
- [2] Check that VIO setting on Murata i.MX6 Interconnect V1 Adapter (Part #2 in **Table 1**) is set to 3.3V (VBAT\_SDIO). Refer to **Red Rectangle** for correct jumper setting in **Figure 11**.
- [3] Insert V1 Adapter board into SD3 slot and SD Card Extender into SD2 slot. Note the orientation as shown in **Figure 5**.
- [4] Connect antenna (Part #4) to the SMA connector of the Murata EVB (Part #1).
- [5] Now you can connect the EVB to the 60-pin Samtec connector on the V1 Adapter board.
- [6] Prepare SD card to boot platform per **Section 3**. Insert SD card, power on platform and interrupt at u-boot. Now type:  
=> `setenv fdt_file imx6sx-sdb-btwifi.dtb`

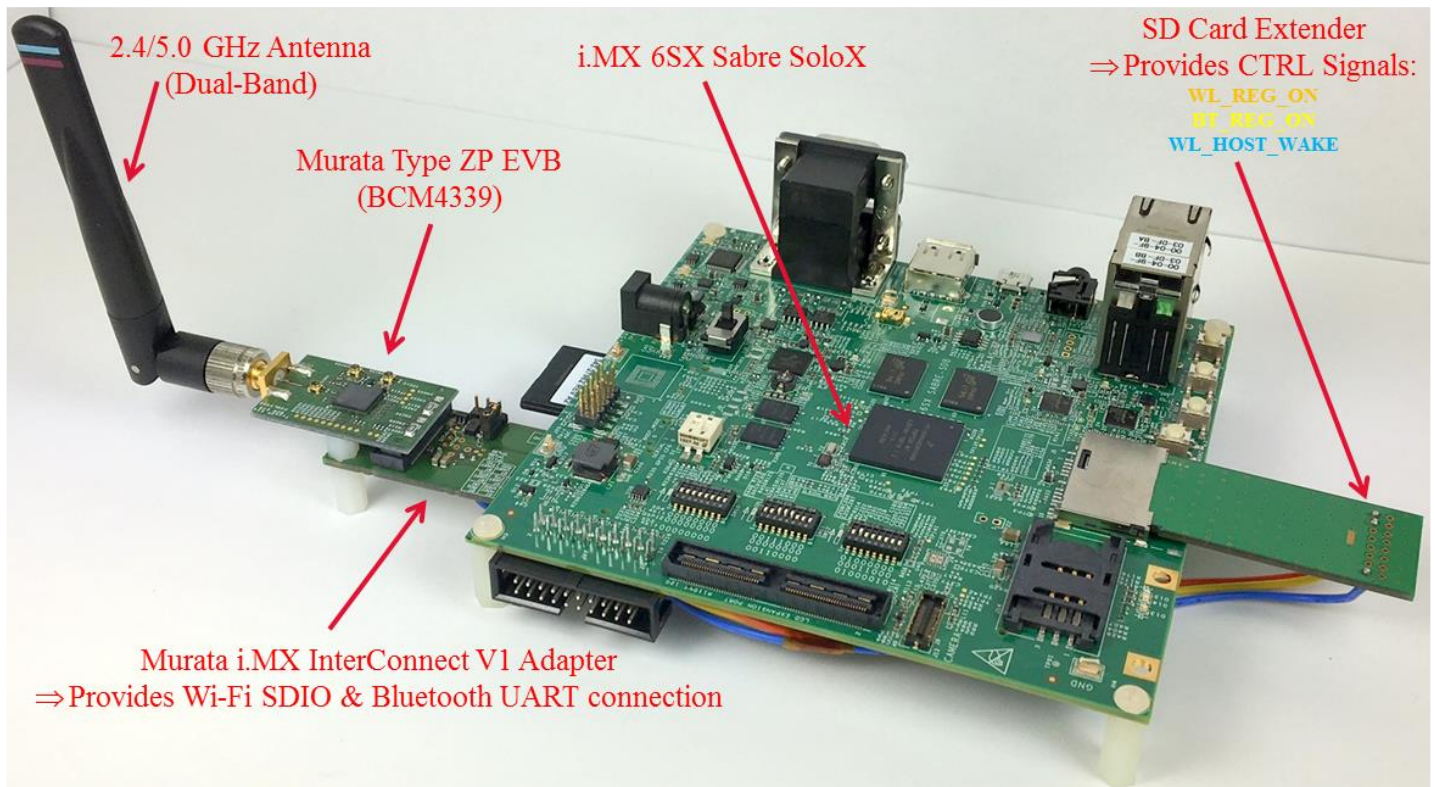
Now save the u-boot configuration and boot the platform:

=> `saveenv`

=> `boot` (causes platform to boot kernel)

- [7] Refer to **Section 5** to test/verify Wi-Fi and Bluetooth functionality.

**Figure 5: i.MX 6SoloX SDB with V1 Adapter and Type ZP EVB**



## 2.2 Connecting to i.MX 6SoloLite EVK

Referring to **Table 3**, V1 Adapter is the only solution that will work for this platform. Murata Wi-Fi/BT EVB is oriented right-side up, and it provides both Wi-Fi and BT functionality via SD1 slot with control signals connected from SD3 slot using SD Card Extender: see **Figure 6** below.

- [1] Ensure no power is applied to i.MX 6SL EVK. Connect J26 micro-USB port to PC and start terminal emulator: “minicom” on Linux or “tera term” on Windows. Set port to 115200-N-8-1.
- [2] Check that VIO setting on Murata i.MX6 Interconnect V1 Adapter (Part #2 in **Table 1**) is set to 3.3V (VBAT\_SDIO). Refer to **Red Rectangle** for correct jumper setting in **Figure 11**.
- [3] Insert V1 Adapter board into SD1 slot and SD Card Extender into SD3 slot. Note the orientation as shown in **Figure 6**.
- [4] Connect antenna (Part #4) to the SMA connector of the Murata EVB (Part #1).
- [5] Now you can connect the EVB to the 60-pin Samtec connector on the V1 Adapter board.
- [6] Prepare SD card to boot platform per **Section 3**. Insert SD card, power on platform and interrupt at u-boot. Now type:  
=> `setenv fdt_file imx6sl-evk-btwifi.dtb`

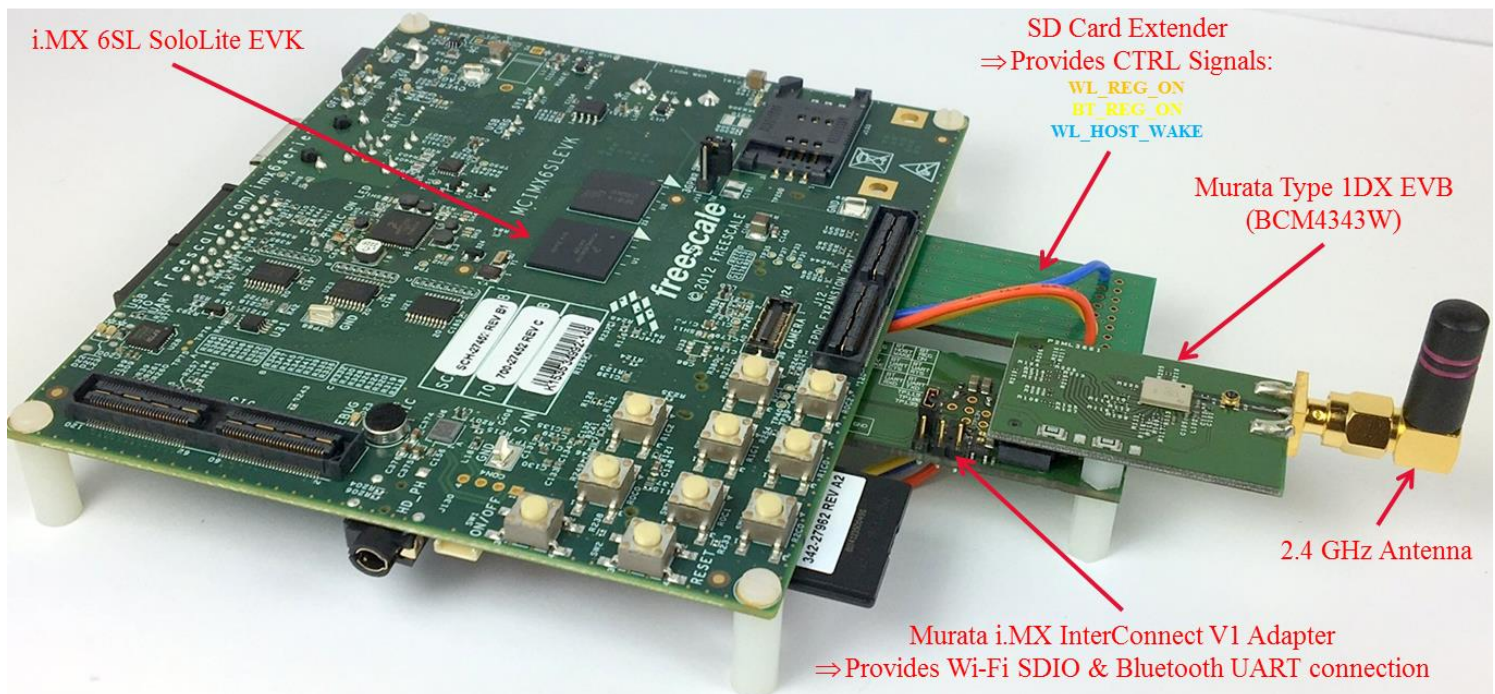
Now save the u-boot configuration and boot the platform:

=> `saveenv`

=> `boot` (causes platform to boot kernel)

- [7] Refer to **Section 5** to test/verify Wi-Fi and Bluetooth functionality.

**Figure 6: i.MX 6SoloLite EVK with V1 Adapter and Type 1DX EVB**



## 2.3 Connecting to i.MX 6Q/DL SDB/SDP or i.MX 6 QP SDB

The following section provides bring-up instructions for i.MX 6QuadPlus SDB, i.MX 6Quad/DualLite SDB, and i.MX 6Quad/DualLite SDP. The i.MX 6QuadPlus SDB has a modified schematic from the (essentially identical) i.MX 6Quad/DualLite SDB/SDP platforms.

### 2.3.1 Specific Hardware Considerations for i.MX 6Quad/DualLite SDB/SDP

Although the Murata Wi-Fi/BT EVK is designed to be “plug ‘n play”, **rework is required** for the i.MX 6Quad/DualLite SDB/SDP platforms. As shipped from the factory, the i.MX 6Q/DL SDB/SDP **do not** connect the J13 Bluetooth ribbon cable connector to the necessary UART and control signals. Refer to the [Hardware User Manual](#) for necessary rework. NXP also details the board rework in their schematic file (Bluetooth page). Page 15 of the NXP schematic (SPF-27516\_C3.pdf) correctly captures the necessary rework to be done.

#### **Repeated here:**

**NOTE:** To use J13, populate resistors R209 - R213 and depopulated the SPI NOR FLASH U14. Resistors R214 and R215 should not be populated because both UART outputs (TXDs) have been crossed together and both UART inputs (RXDs) have been crossed together. To make the UART work correctly, solder a jumper wire from R215 pad 1 to R214 pad 2 and from R215 pad 2 to R214 pad 1.

### 2.3.2 Specific Hardware Considerations for i.MX 6QP SDB

Depending on the revision, rework **may be required** for the i.MX 6QuadPlus SDB. This rework connects the Bluetooth UART and Wi-Fi/BT control signals (WL\_REG\_ON, BT\_REG\_ON, and WL\_HOST\_WAKE). Revision B of i.MX 6QP SDB populates the necessary resistors for connecting BT UART and Wi-Fi/BT control signals. If your board has revision earlier than B (i.e. A2) then you will have to populate the necessary resistors. Refer to the [Hardware User Manual](#) for necessary rework. For the Rev A2 board, page 15 of the NXP schematic (SPF-28857\_A2.pdf) correctly captures the necessary rework to be done. Note the much simpler rework on the i.MX 6QB SDB given that the no special “crossing” of TX and RX resistor pads is necessary.

#### **Repeated here:**

**NOTE:** To use J13, populate resistors R209 - R213 and depopulate the SPI NOR FLASH U14.

### 2.3.3 Murata Wi-Fi/BT EVK Bring-Up on i.MX 6Q/DL SDB/SDP or i.MX 6 QP SDB

**Note:** The following steps will only pass if NXP Platform has been correctly reworked The NXP i.MX6 platform has been inverted. This makes working with Wi-Fi/BT EVK much easier. The one drawback is Ethernet port access. To properly match Wi-Fi/BT EVK and i.MX6 platform heights, additional nylon standoffs are required.



- [1] Ensure no power is applied to i.MX 6QP SDB or i.MX 6Q/DL SDB/SDP. Connect J509 micro-USB port to PC and start terminal emulator: “minicom” on Linux or “tera term” on Windows. Set port to 115200-N-8-1.
- [2] Check that VIO setting on Murata i.MX6 Interconnect V2 Adapter (Part #3 in **Table 1**) is set to 3.3V (VBAT\_SDIO). Refer to **Red Rectangle** for correct jumper setting in **Figure 14**.
- [3] After connecting ribbon cable to both adapter and i.MX6 platform, insert Adapter board into SD2 slot. Note the orientation as shown in **Figure 7**.
- [4] Connect antenna (Part #4) to the SMA connector of the Murata EVB (Part #1).
- [5] Now you can connect the EVB to the 60-pin Samtec connector on the Adapter board.
- [8] Prepare SD card to boot platform per **Section 3**. Insert SD card, power on platform and interrupt at u-boot. Now type:  
=> `setenv fdt_file imx6q-sabresd-btwifi.dtb` (or `imx6dl-sabresd-btwifi.dtb` or `imx6qp-sabresd-btwifi.dtb`)

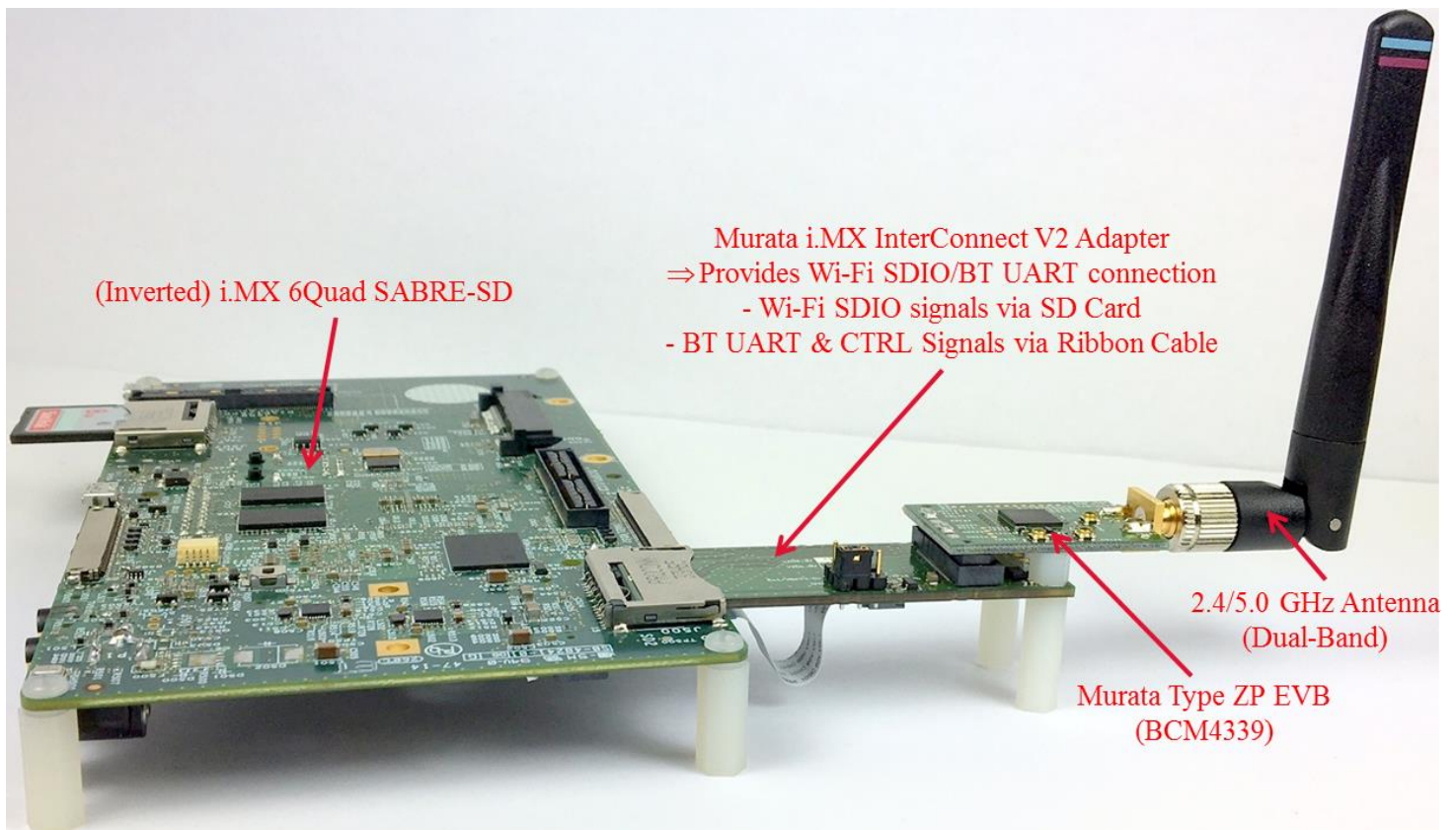
Now save the u-boot configuration and boot the platform:

=> `saveenv`

=> `boot` (causes platform to boot kernel)

- [6] Refer to **Section 5** to test/verify Wi-Fi and Bluetooth functionality.

**Figure 7: i.MX 6Quad/DualLite SDB (Inverted) with V2 Adapter and Type ZP EVB**



## 2.4 Connecting to i.MX 6UltraLite EVK or i.MX 6ULL EVK

As indicated earlier, the i.MX 6UL and i.MX 6ULL EVK's share the same baseboard. As such interconnect for the Murata Wi-Fi/BT EVK is identical on both platforms. No rework is required for V2 Adapter to provide both Wi-Fi (SDIO in-band interrupt signaling) and Bluetooth connectivity. Note that for out-of-band interrupt support, hardware rework and software modifications are necessary. For specifics refer to the [Hardware User Manual](#) and [Quick Start Binary Patches](#).

- [1] Ensure no power is applied to i.MX 6UltraLite or i.MX 6ULL EVK. Connect J1101 micro-USB port to PC and start terminal emulator: "minicom" on Linux or "tera term" on Windows. Set port to 115200-N-8-1.
- [2] Check that VIO setting on Murata i.MX6 Interconnect V2 Adapter (Part #3 in **Table 1**) is set to 3.3V (VBAT\_SDIO). Refer to **Red Rectangle** for correct jumper setting in **Figure 14**.
- [3] Connect antenna (Part #4) to the SMA connector of the Murata EVB (Part #1).
- [4] Connect the EVB to the 60-pin Samtec connector on the Adapter board.
- [5] Connect ribbon cable at both ends before inserting Murata EVK into SD1 slot. Note the orientation as shown in **Figure 8**. **Make sure that the adapter clicks in correctly.**
- [9] Prepare micro SD card to boot platform per **Section 3**. Insert SD card, power on platform and interrupt at u-boot. Now type (remove "<.>" string for default SDIO in-band interrupts; include "OOB\_IRQ" string for OOB IRQ configuration):  
`=> setenv fdt_file imx6ul-14x14-evk-btwifi<.OOB_IRQ>.dtb`  
(or `imx6ul-9x9-evk-btwifi<.OOB_IRQ>.dtb` or  
`imx6ull-14x14-evk-btwifi<.OOB_IRQ>.dtb` or `imx6ull-9x9-evk-btwifi<.OOB_IRQ>.dtb`)

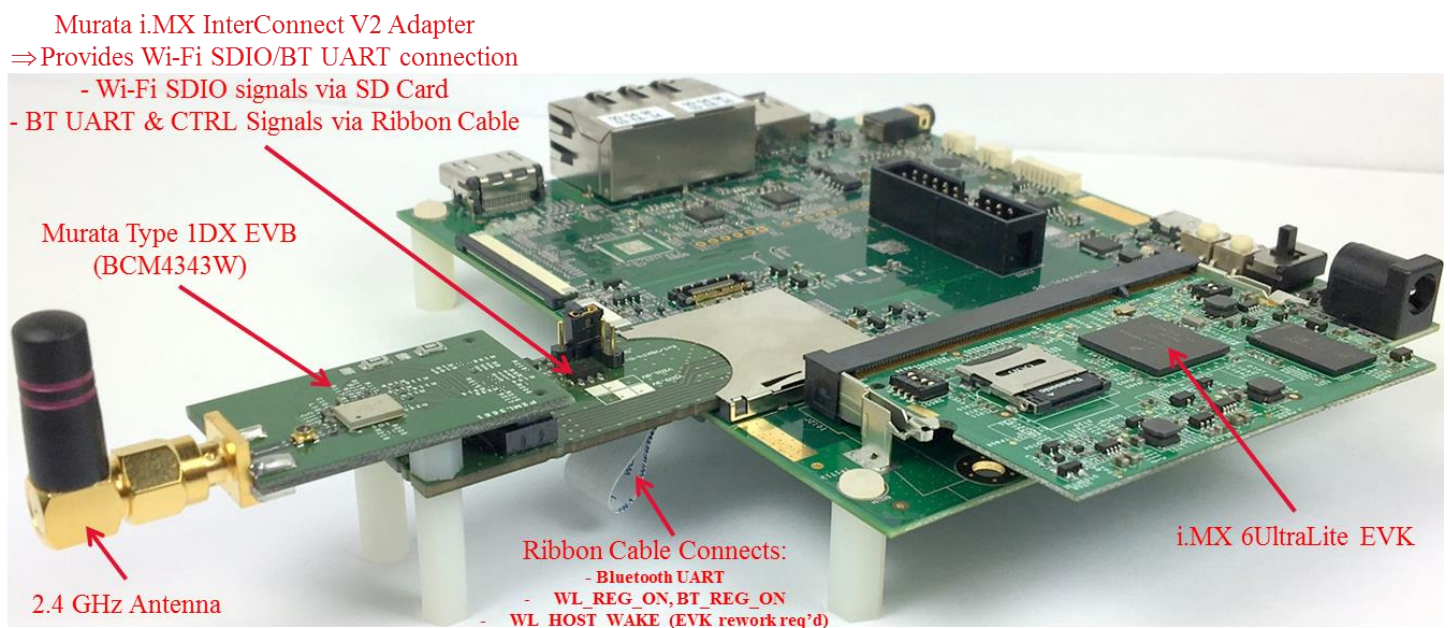
Now save the u-boot configuration and boot the platform:

`=> saveenv`

`=> boot` (causes platform to boot kernel)

- [6] Refer to **Section 5** to test/verify Wi-Fi and Bluetooth functionality.

**Figure 8: i.MX 6UltraLite EVK with V2 Adapter and Type 1DX EVB**

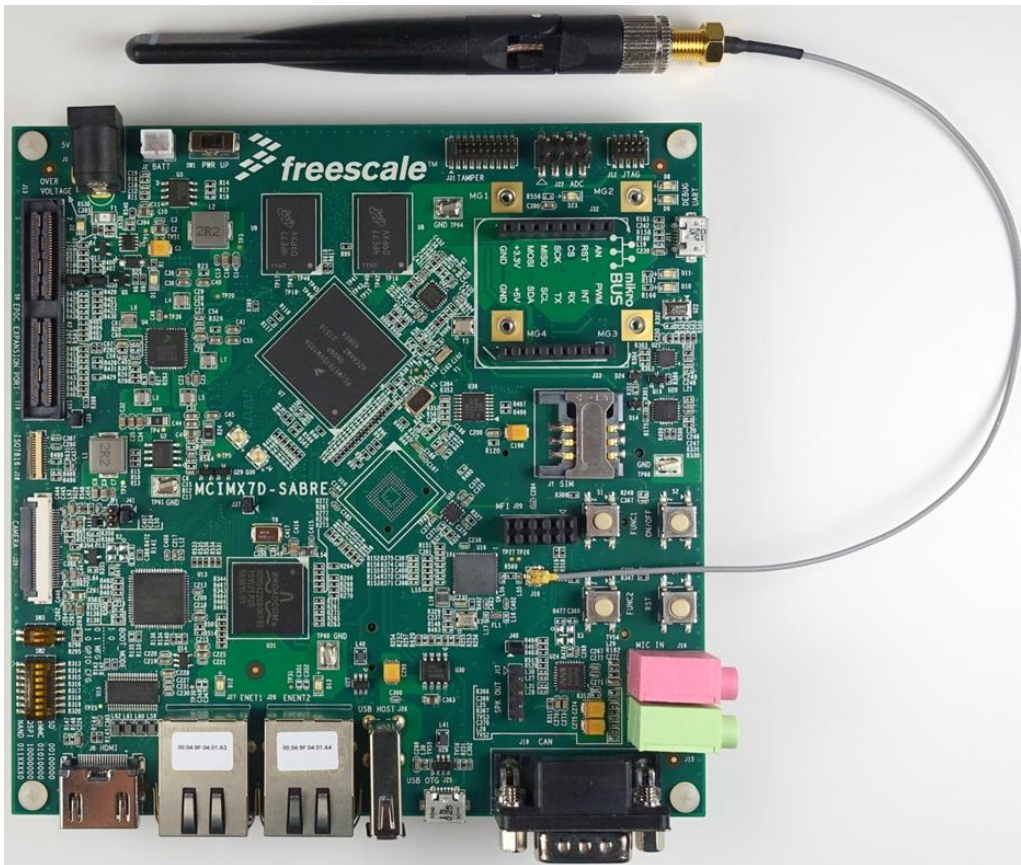


### 3 Bringing up Wi-Fi/BT on i.MX 7Dual SABRE Development Board

The NXP i.MX 7Dual SDB (see **Figure 9**) has the Murata Type ZP module soldered down on the board. As such there is no need to use the Murata InterConnect Kit. For customers wanting to evaluate another Murata module other than Type ZP, the SD card slot interface on the i.MX7D SDB is not recommended. The SD card slot is required for booting the platform<sup>8</sup>.

- [1] The i.MX 7Dual SABRE Development Board does not ship with an antenna. As such the WLAN sensitivity (RF reception) is attenuated by approximately 30 dBm. Murata strongly recommends that the user obtain the necessary antenna to operate Wi-Fi/BT properly<sup>9</sup>.
- [2] Connect J11 micro-USB port to PC and start terminal emulator: “minicom” on Linux or “tera term” on Windows. Set port to 115200-N-8-1.
- [3] Prepare SD card to boot platform per **Section 4**. U-boot has to be specific for i.MX 7D SDB.
- [4] Insert SD card and power on platform: no need to interrupt u-boot as the default DTB file integrates support for Wi-Fi and Bluetooth.
- [5] Refer to **Section 5** to test/verify Wi-Fi and Bluetooth functionality.

**Figure 9: i.MX 7Dual SDB with Murata Type ZP**



<sup>8</sup> The i.MX 7Dual SDB can be configured to flash u-boot and the kernel to the onboard QSPI-NOR. The root file system can then be NFS-mounted. However this configuration (to free up SD card slot) is very complicated and not supported in this document.

<sup>9</sup> Possible antenna/cable solution would be the following: [ANT-DB1-RAF-SMA](#) and [931-1188-ND](#).

## 4 Preparing Bootable SD Card for i.MX6 with Murata Wi-Fi/BT EVK

The current releases supported are [NXP Linux 3.14.52 1.1.0<sup>10</sup>](#) and [4.1.15 2.0.0 GA BSP's](#). Both of these GA releases from NXP provide verified/tested Wi-Fi/Bluetooth functionality on all major i.MX6/7 platforms. NXP provides pre-built images for each platform that includes all necessary components to get Wi-Fi/BT up and running. If you prefer to use your own customized build, then refer to the [Linux User Manual](#) to build the file system, kernel, and DTB files from source code.

With the Linux 3.14.52 1.1.0 and 4.1.15 2.0.0 releases, follow these steps:

- Sign up for access to necessary support resources. At minimum you need login access to NXP website and “My Murata” to prepare the (micro) SD card with correct image for booting i.MX platform with Murata Wi-Fi/BT EVK.
- Download the appropriate i.MX6/7 “Linux Binary Demo File” from NXP website: refer to **Table 5** for the correct hyperlink.
- After appropriate file extraction, flash (micro) SD card with “\*.sdcard” image file.
- Download the [Murata Quick Start Binary Patch file](#) for either Linux [3.14.52 1.1.0](#) or [4.1.15 2.0.0](#).
- Extract Murata Quick Start Binary Patch file and copy/replace files over to (micro) SD card. Note that the kernel selected (zImage with integrated bcmhd WLAN driver) and DTB<sup>11</sup> file depend on the Murata module and interrupt configuration. Refer to **Table 6** for more details on kernel/DTB file selection. **Note:** the binary patch files also contain changes to the default file system as well.
- If configuring WLAN for **automatic** bring-up after kernel boots, modify the “/etc/network/if-pre-up.d/wpa-suppllicant” file to configure the correct NVRAM and firmware files for your configuration. Refer to **Table 6** for specifics on filenames.
- Insert (micro) SD card into i.MX6/7 Platform.

When booting the i.MX6/7 Platform (already covered in **Sections 2 and 3**):

For i.MX6 Platform, connect Murata Wi-Fi/BT EVK to i.MX6 Platform and power up. Interrupt u-boot to change the DTB file (setenv fdt\_file):

- Refer to **Table 6: Specific Files for Each i.MX Platform/Linux Kernel Version/Murata EVK Configuration** for correct DTB file.
- For i.MX 7D SDB just power up the platform. Default DTB file supports onboard Murata Type ZP module.

---

<sup>10</sup> The only “GA BSP” released for 3.14.52 is labelled “Linux 3.14.52” on NXP website. However the last kernel version released for 3.14 which Murata supports is “Linux 3.14.52\_1.1.0”.

<sup>11</sup> Optional DTB files are only included for i.MX 6UL/ULL EVK's which have SDIO in-band interrupts configured by default. If desired, the user can rework the platform and use different kernel and DTB file (as specified in **Table 6**).

## 4.1 Getting Signed Up to Access Support Resources

To be able to correctly flash SD card, you need access to NXP website to download baseline “demo” images and the [NXP Murata i.MX Support Portal](#) to download Quick Start Patch. Refer to **Table 10** for a comprehensive list of support resources. It is important to have access to “My Murata”, NXP Website, NXP Portal, and Cypress Portal(s). You can access the main [Murata i.MX Landing Page](#) without any login credentials.

## 4.2 Downloading i.MX6/7 Image Files to Flash SD Card

**Table 5** provides direct download links to i.MX6/7 “Linux Binary Demo File” for each NXP Platform. The available SD card images are included in these downloads. All download links in **Table 5** originate on this [NXP webpage](#). Simply click on the hyperlink for the desired “SD Card Image File” and download the package. After downloading, extract the appropriate “\*.sdcard.bz2” file from the bundle/package. See **Sections 4.2.1** and **4.2.2** on steps to extract “\*.sdcard” file on Linux and Windows PC environment.

**NOTE:** For i.MX6UL/ULL EVK’s and i.MX7D SDB, there are “4.1.15\_2.0.1” patch links listed. The i.MX7D 4.1.15\_2.0.0 image will not boot the platform. However both 4.1.15\_2.0.0 and 4.1.15\_2.0.1 versions work on the i.MX6UL EVK.

**Table 5: i.MX6/7 Platforms’ SD Card Image Filenames**

i.MX6/7 Platform	Linux Version	Available SD Card Image Files
<a href="#">i.MX 6QuadPlus SDB</a> <a href="#">i.MX6Quad/DualLite SDB</a> <a href="#">i.MX 6Quad/DualLite SDP</a>	3.14.52_1.1.0	<a href="#">fsl-image-gui-x11-imx6qdsolo.rootfs.sdcard.bz2</a> <a href="#">fsl-image-qt5-x11-imx6qdsolo.rootfs.sdcard.bz2</a>
	4.1.15_2.0.0	<a href="#">fsl-image-validation-imx-x11-imx6qdsolo.sdcard.bz2</a>
<a href="#">i.MX 6SoloX SDB</a>	3.14.52_1.1.0	<a href="#">fsl-image-gui-x11-imx6sx_all.rootfs.sdcard.bz2</a> <a href="#">fsl-image-qt5-x11-imx6sx_all.rootfs.sdcard.bz2</a>
	4.1.15_2.0.0	<a href="#">fsl-image-validation-imx-x11-imx6sx_all.sdcard.bz2</a> <a href="#">fsl-image-qt5-validation-imx-x11-imx6sx_all.sdcard.bz2</a>
<a href="#">i.MX 6SoloLite EVK</a>	3.14.52_1.1.0	<a href="#">fsl-image-gui-x11-imx6slevk.rootfs.sdcard.bz2</a> <a href="#">fsl-image-qt5-x11-imx6slevk.rootfs.sdcard.bz2</a>
	4.1.15_2.0.0	<a href="#">fsl-image-validation-imx-x11-imx6slevk.sdcard.bz2</a> <a href="#">fsl-image-qt5-validation-imx-x11-imx6slevk.sdcard.bz2</a>
<a href="#">i.MX 6UltraLite EVK</a>	3.14.52_1.1.0	<a href="#">fsl-image-gui-x11-imx6ul.rootfs.sdcard.bz2</a>
	4.1.15_2.0.0	<a href="#">fsl-image-validation-imx-x11-imx6ul.sdcard.bz2</a>
	4.1.15.2.0.1	<a href="#">fsl-image-validation-imx-imx6ulevk.sdcard.bz2</a>
<a href="#">i.MX 6ULL EVK</a>	4.1.15_2.0.1	<a href="#">fsl-image-validation-imx6ull14x14evk.sdcard.bz2</a>
<a href="#">i.MX 7Dual SDB</a>	4.1.15_2.0.1	<a href="#">fsl-image-validation-imx-imx7dsabresd.sdcard.bz2</a>

## 4.2.1 Linux PC Steps to Extract “\*.sdcard” File

Before flashing (micro) SD card, there are two necessary operations after downloading the file from NXP website. Using Linux 4.1.15\_2.0.0 and i.MX 6UltraLite EVK as an example, download the Linux 4.1.15\_2.0.0 binary image for i.MX 6UL EVK [here](#), the following commands would extract the download package and decompress the “\*.sdcard.bz2” file:

```
$ tar -xvzf L4.1.15_2.0.0-ga_images_MX6ULEVK.tar.gz
$ cd L4.1.15_2.0.0-ga_images_MX6ULEVK
$ bzip2 -dk fsl-image-validation-imx-x11-imx6ul.sdcard.bz2
```

## 4.2.2 Windows PC Steps to Extract “\*.sdcard” File

On Windows PC various utilities can be used to unzip/extract this “\*.sdcard” file (i.e. WinZip, WinRAR, 7-Zip, etc.).

## 4.3 Flashing SD Card

### 4.3.1 Linux PC Steps to Flash SD Card

Insert (micro) SD card into host machine (PC). It is **imperative that the (micro) SD card comes up as “/dev/sdx” device**. If it does not then you may require a USB to SD card adapter as shown in **Figure 10**.

**Figure 10: USB to SD Card Reader/Writer Adapter**



Once the (micro) SD card has been inserted into the PC, run the “dmesg” command to find which “/dev/sdx” device was just enumerated:

```
$ dmesg
```

The enumeration log of the \*just\* inserted (micro) SD card should look like:

```
[285317.464075] usbcore: registered new interface driver usb-storage
[285318.472525] scsi 6:0:0:0: Direct-Access   Generic- USB3.0 CRW   -0 1.00 PQ: 0 ANSI: 4
[285318.473143] sd 6:0:0:0: Attached scsi generic sg2 type 0
[285319.263194] sd 6:0:0:0: [sdc] 15597568 512-byte logical blocks: (7.98 GB/7.43 GiB)
[285319.264368] sd 6:0:0:0: [sdc] Write Protect is off
[285319.264379] sd 6:0:0:0: [sdc] Mode Sense: 2f 00 00 00
[285319.265413] sd 6:0:0:0: [sdc] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
[285319.274779] sdc: sdc1 sdc2
```

Referencing this example log, the correct device for the (micro) SD card is “/dev/sdc”.

**NOTE: Before running next command, make sure you have selected the correct device. Otherwise you may wipe your hard drive! Substitute the correct (micro) SD device name for “/dev/sdx” in “dd” command line below.**

```
$ sudo dd if=<fsl image name>.sdcard of=/dev/sdx bs=1M && sync
```

Following previous Linux 4.1.15 2.0.0 i.MX 6UL EVK example and assuming “/dev/sdc” enumeration of micro SD card, the command is:

```
$ sudo dd if=./fsl-image-validation-imx-x11-imx6ul.sdcard.sdcard of=/dev/sdc bs=1M && sync
```

### 4.3.2 Windows PC Steps to Flash SD Card

Windows utilities such as “Win32 Disk Imager” or “NetBSD Disk Image Tool” can be used to flash the (micro) SD card. For example when using “Win32 Disk Imager”, follow these steps:

- After bringing up “Win32 Disk Imager” program<sup>12</sup>, click on the folder icon/button and navigate to the location of the desired “\*.sdcard” file. You need to change “\*.img” file type to “\*.\*” to select/open the “\*.sdcard” file.
- Select the “Device” button and select the drive letter corresponding to the (micro) SD card: formatting SD card may be necessary beforehand with Windows low level utilities<sup>13</sup>.
- Now click the “Write” button. A warning window will pop up that warns that the device being written to may be corrupted.
- Upon completion, a window with “Write Successful” should appear.
- Click “OK” on “Write Successful” window.
- Now click “Exit” on “Win32 Disk Imager” window.
- To be safe, you may elect to “eject” the SD card removable memory device before removing it.

**NOTE: The next steps (which require file modification on Linux “ext3” partition) require the use of a Linux PC.**

---

<sup>12</sup> “Win32 Disk Imager” is an open source tool that can be downloaded from websites such as “sourceforge.net”.

<sup>13</sup> Unlike Linux environment, Windows PC does not require use of “USB to SD Card Reader/Writer” adapter.

## 4.4 Murata Modifications to the Default NXP BSP Release

Murata provides a “Quick Start Binary Patch Release” for the NXP [3.14.52\\_1.1.0](#) and [4.1.15\\_2.0.0](#) releases. Description of the changes to default NXP i.MX image (just flashed to SD card) is provided for both kernels as follows.

**NOTE:** Prior to downloading the Quick Start Binary Patch for either kernel version, you will need access to the [NXP Murata i.MX Support Portal](#).

### 4.4.1 Murata Quick Start Binary Patch Modifications to Linux 3.14.52\_1.1.0 Release

- Reconfigure WLAN driver so bcmdhd is built into the kernel image – rather than being a loadable module (bcmdhd.ko). This configuration matches the newer Linux 4.1.15\_2.0.0.
- Add OOB IRQ support for all i.MX platforms. **Note:** the i.MX6UL EVK requires a hardware change to run optional OOB IRQ configuration.
- Fix SN8000CMK bring-up error on i.MX6UL EVK: BT\_REG\_ON line pulled low prior to SDIO initialization. This fix applies to all modules: i.e. default operation for all modules is that both WL\_REG\_ON and BT\_REG\_ON lines are pulled low (0V) during power-on-reset.
- Enable Bluetooth interface on i.MX 6SoloLite EVK.
- Correctly MUX WL\_HOST\_WAKE and BT\_REG\_ON lines on i.MX 6SoloLite EVK.
- Correctly MUX WL\_HOST\_WAKE line on i.MX 6SoloX SDB.
- Add specific edge-sensitive OOB IRQ support for Murata Type 1BW and SN8000 modules. Default OOB IRQ configuration on other modules is level-sensitive (also supported). **Note:** a different kernel is required to support edge versus level sensitive interrupts given that the bcmdhd driver is integrated into the kernel.
- Add patch for module initialization failure when reference clock (32 KHz) not asserted.
- Add patch for suspend/resume power save erratum.
- Patch bcmdhd driver to fix Wi-Fi Direct erratum.
- Modify WPA supplicant to enable Wi-Fi Direct (P2P) and provide associated binaries.
- Provide Hostapd supplicant & associated files to enable “Wi-Fi Hot Spot” or “Soft AP”.
- Provide NVRAM/calibration files for Murata 1CK module.
- Provide NVRAM/calibration file update to support OOB IRQ on SN8000.
- Provide modified “/etc/network/interfaces” file to automatically bring up Wi-Fi/WLAN (“wlan0”) interface when kernel boots (optional).
- Provide sample “etc/networks/if-pre-up.d/wpa-supPLICANT” file which pulls in necessary modifications for configuration NVRAM and firmware path.

### 4.4.2 Murata Quick Start Binary Patch Modifications to Linux 4.1.15\_2.0.0 Release

- Add optional OOB IRQ support for i.MX6UL/ULL EVK’s. **Note:** the i.MX6UL/ULL EVK’s require a hardware change to support OOB IRQ configuration.
- Support OOB IRQ configuration for Type 1BW and SN8000 modules on all i.MX platforms: this requires a modified kernel as built-in bcmdhd driver supports edge-sensitive interrupts.
- Enable Bluetooth interface on i.MX 6SoloLite EVK.
- Correctly MUX BT\_REG\_ON line on i.MX 6SoloLite EVK.
- Fix SN8000CMK bring-up error on i.MX6UL/ULL EVK’s: BT\_REG\_ON line pulled low prior to SDIO initialization. This fix applies to all modules: i.e. default operation for all modules is that both WL\_REG\_ON and BT\_REG\_ON lines are pulled low (0V) during power-on-reset.



- Correctly MUX WL\_REG\_ON and BT\_REG\_ON lines on i.MX 7Dual SDB.
- Add patch for module initialization failure when reference clock (32 KHz) not asserted.
- Add patch for suspend/resume power save erratum.
- Modify WPA supplicant to enable Wi-Fi Direct (P2P) and provide associated binaries.
- Provide Hostapd supplicant & associated files to enable “Wi-Fi Hot Spot” or “Soft AP”.
- Provide NVRAM/calibration files for Murata 1CK module.
- Provide NVRAM/calibration file update to support OOB IRQ on SN8000.
- Provide modified “/etc/network/interfaces” file to automatically bring up Wi-Fi/WLAN (“wlan0”) interface when kernel boots (optional).
- Provide sample “etc/networks/if-pre-up.d/wpa-supPLICANT” file which pulls in necessary modifications for configuration NVRAM and firmware path.
- Provide specific kernel (with bcmhdh.ko loadable module driver) to work around erratum (kernel crash) with running manufacturing test firmware. **Note:** this is only required for specific RF testing when using built-in Cypress “wl” tool.

## 4.5 WLAN Firmware and NVRAM Update Considerations

When updating (micro) SD card, it is recommended that the user check for latest firmware (fw\_bcmdhd.bin) and NVRAM (bcmhdh.cal) updates from both [My Murata i.MX Support Portal](#) (NVRAM updates) and [Cypress Murata i.MX Support Portal](#) (NVRAM and firmware updates). Refer to **Table 8: Murata Module to Firmware/NVRAM Mapping** for correct folder location and filenames.

## 4.6 Specific Kernel / DTB / NVRAM / Firmware files for each Configuration

To address existing errata, add necessary features, and make both Linux releases functionally equivalent; it was necessary to introduce additional kernels as shown in **Table 6**. For a given **i.MX Platform / Linux Version / Murata module / interrupt configuration**, there is a corresponding list detailing the following:

- NVRAM (calibration file used by WLAN).
- Firmware (binary downloaded to WLAN MAC core)
- Kernel (varies given different interrupt configurations)
- DTB (Device Tree Blob: target specific with specific exception for i.MX 6UL/ULL EVK's).

When copying over the [Murata Quick Start Binary Patch files](#) to the (micro) SD card, and prior to inserting the (micro) SD card to the i.MX platform, refer to **Table 6** and follow these steps:

- Select the correct Linux kernel and copy it to the “boot” folder, renaming it to “zImage”.
- Copy all DTB files from kernel-specific Binary patch to “boot” folder.
- Copy Murata Patch file system updates which includes:
  - Type 1CK and SN8000 NVRAM files.
  - WPA and Hostapd binary and configuration files.
  - Network configuration files for automatic WLAN bring-up during kernel boot.

**NOTE:** There are additional steps required when running manufacturing test mode on the Linux 4.1.15 kernel release. Refer to **Section 5.1.9.2** for more details.

**NOTE:** There are two different configurations for the WPA and Hostapd supplicant binary and configuration files: one is compiled for CortexA7 (i.MX6UL/ULL and i.MX7D), the other for CortexA9 (i.MX 6SX/SL/Quad(Plus)/DualLite). **Make sure to select** the correct Cortex A7 or A9 configuration based on the target processor.

To further illustrate this procedure, here is a sample sequence when preparing micro SD card for **“i.MX 6SoloX (A9) / Kernel 4.1.15 / Murata Type 1BW”** configuration:

```
$ cd <Murata 4.1.15 Quick Start Extracted Folder>
$ sudo chown -R root:root *
$ sudo cp Kernel/zImage.OOB_IRQ_Edge /media/<i.MX Boot Folder Name>/zImage
$ sudo cp DTB/* /media/<i.MX Boot Folder Name>/
$ sudo cp -Rfp i.MX_4.1.15_Murata_Image_Update_CortexA9/* /media/<root file system string>/
```

Given “automatic” WLAN bring-up configuration, we need to edit “/etc/network/if-pre-up.d/wpa-supPLICANT” file (on SD card) so **bcmdhd driver parameters point to Type 1BW NVRAM and firmware:**

```
$ echo /lib/firmware/bcm/1BW_BCM43340/fw_bcmdhd.bin > /sys/module/bcmdhd/parameters/firmware_path
$ echo /lib/firmware/bcm/1BW_BCM43340/bcmdhd.1BW.OOB.cal > /sys/module/bcmdhd/parameters/nvram_path
```

➔ If we want to disable the automatic WLAN bring-up configuration, then we just have to edit the “/etc/network/interfaces” file (on SD card) and comment out the “auto wlan0” line.

Now the (micro) SD card preparation is complete and it can be inserted into the i.MX platform. The DTB file cannot be set without a serial console connection to the i.MX Platform. Typically “minicom” is used on a Linux PC whereas “tera term” is used on Windows environment. Regardless of the console emulator used, the correct port settings are 115200-N-8-1. Once the i.MX Platform is powered up, the bootloader/u-boot must be interrupted to change the “fdt\_file” parameter: i.e. hit the “carriage return” button repeatedly after powering up platform.

**Power on platform and interrupt at u-boot. Type in following commands to set DTB file:**

```
=> setenv fdt_file <“btwifi” DTB filename from Table 6>14
=> saveenv
=> boot (causes platform to boot kernel)
```

**NOTE:** Both i.MX 6UL/ULL EVK’s have twice as many entries in **Table 6** given the optional OOB IRQ configuration. The default configuration (no hardware change) for i.MX 6UL/ULL EVK’s is SDIO in-band interrupts. When configuring these platforms for OOB IRQ, the hardware change **necessitates disabling the second of two Ethernet ports.** Refer to the **[Murata Hardware User Manual](#)** for more details.

---

<sup>14</sup> Note that i.MX 7D SDB does not have “btwifi” string in DTB filename. This is because the board has integrated Wi-Fi/BT module.

**Table 6: Specific Files for Each i.MX Platform/Linux Kernel Version/Murata EVK Configuration**

<b>i.MX Platform</b>	<b>Linux Version</b>	<b>Murata Wi-Fi/BT EVK</b>	<b>NVRAM Firmware Kernel DTB</b>
<a href="#">i.MX 7Dual SDB</a>	4.1.15_2.0.0	ZP	/lib/firmware/bcm/ZP_BCM4339/bcmdhd.ZP.OOB.cal /lib/firmware/bcm/ZP_BCM4339/fw_bcmdhd.bin zImage.OOB_IRQ_Level imx7d-sdb.dtb
<a href="#">i.MX 6QuadPlus SDB</a>	3.14.52_1.1.0/ 4.1.15_2.0.0	1CK, ZP, 1DX,1FX	/lib/firmware/bcm/<Module>_<Chipset>/bcmdhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Level imx6qp-sabresd-btwifi.dtb
		1BW SN8000	/lib/firmware/bcm/<Module>_<Chipset>/bcmdhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Edge imx6qp-sabresd-btwifi.dtb
<a href="#">i.MX6Quad SDB/SDP</a>	3.14.52_1.1.0/ 4.1.15_2.0.0	1CK, ZP, 1DX,1FX	/lib/firmware/bcm/<Module>_<Chipset>/bcmdhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Level imx6q-sabresd-btwifi.dtb
		1BW SN8000	/lib/firmware/bcm/<Module>_<Chipset>/bcmdhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Edge imx6q-sabresd-btwifi.dtb
<a href="#">i.MX6DualLite SDB/SDP</a>	3.14.52_1.1.0/ 4.1.15_2.0.0	1CK, ZP, 1DX,1FX	/lib/firmware/bcm/<Module>_<Chipset>/bcmdhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Level imx6dl-sabresd-btwifi.dtb
		1BW SN8000	/lib/firmware/bcm/<Module>_<Chipset>/bcmdhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Edge imx6dl-sabresd-btwifi.dtb
<a href="#">i.MX 6SoloX SDB</a>	3.14.52_1.1.0/ 4.1.15_2.0.0	1CK, ZP, 1DX,1FX	/lib/firmware/bcm/<Module>_<Chipset>/bcmdhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Level imx6sx-sdb-btwifi.dtb
		1BW SN8000	/lib/firmware/bcm/<Module>_<Chipset>/bcmdhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Edge imx6sx-sdb-btwifi.dtb
<a href="#">i.MX 6SoloLite EVK</a>	3.14.52_1.1.0/ 4.1.15_2.0.0	1CK, ZP, 1DX,1FX	/lib/firmware/bcm/<Module>_<Chipset>/bcmdhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Level imx6sl-evk-btwifi.dtb
		1BW SN8000	/lib/firmware/bcm/<Module>_<Chipset>/bcmdhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Edge imx6sl-evk-btwifi.dtb

<a href="#">i.MX 6UL EVK</a>	3.14.52_1.1.0	1CK,ZP,1BW, 1DX,1FX, SN8000	/lib/firmware/bcm/<Module>_<Chipset>/bcmhd.<Module>.SDIO.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.SDIO_InBand imx6ul-14x14-evk-btwifi.dtb <b>or</b> imx6ul-9x9-evk-btwifi.dtb <sup>15</sup>
<a href="#">i.MX 6UL EVK</a>	4.1.15_2.0.0	1CK,ZP,1BW, 1DX,1FX, SN8000	/lib/firmware/bcm/<Module>_<Chipset>/bcmhd.<Module>.SDIO.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Level <sup>16</sup> imx6ul-14x14-evk-btwifi.dtb <b>or</b> imx6ul-9x9-evk-btwifi.dtb <sup>17</sup>
<a href="#">i.MX 6UL EVK</a> <b>OOB_IRQ*</b>	3.14.52_1.1.0/ 4.1.15_2.0.0	1CK, ZP, 1DX,1FX	/lib/firmware/bcm/<Module>_<Chipset>/bcmhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Level imx6ul-14x14-evk-btwifi.OOB_IRQ.dtb <b>or</b> imx6ul-9x9-evk-btwifi.OOB_IRQ.dtb
		1BW SN8000	/lib/firmware/bcm/<Module>_<Chipset>/bcmhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Edge imx6ul-14x14-evk-btwifi.OOB_IRQ.dtb <b>or</b> imx6ul-9x9-evk-btwifi.OOB_IRQ.dtb
<a href="#">i.MX 6ULL EVK</a>	4.1.15_2.0.0	1CK,ZP,1BW, 1DX,1FX, SN8000	/lib/firmware/bcm/<Module>_<Chipset>/bcmhd.<Module>.SDIO.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Level <sup>16</sup> imx6ull-14x14-evk-btwifi.dtb <b>or</b> imx6ull-9x9-evk-btwifi.dtb
<a href="#">i.MX 6ULL EVK</a> <b>OOB_IRQ*</b>	4.1.15_2.0.0	1CK, ZP, 1DX,1FX	/lib/firmware/bcm/<Module>_<Chipset>/bcmhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Level imx6ull-14x14-evk-btwifi.OOB_IRQ.dtb <b>or</b> imx6ull-9x9-evk- btwifi.OOB_IRQ.dtb
		1BW SN8000	/lib/firmware/bcm/<Module>_<Chipset>/bcmhd.<Module>.OOB.cal /lib/firmware/bcm/<Module>_<Chipset>/fw_bcmdhd.bin zImage.OOB_IRQ_Edge imx6ull-14x14-evk-btwifi.OOB_IRQ.dtb <b>or</b> imx6ull-9x9-evk- btwifi.OOB_IRQ.dtb

## 5 Test/Verification of Wi-Fi and Bluetooth

Now the kernel should be booting correctly on the platform with Murata Wi-Fi/BT EVK plugged in (i.e. already set correct DTB file when interrupting u-boot). Next steps are to verify Wi-Fi and Bluetooth functionality. The NXP i.MX (Murata modified) images include all the necessary files to support Wi-Fi and Bluetooth bring-up/testing/verification. The relevant folders and files are summarized in **Table 7: Embedded Wi-Fi/Bluetooth Files**.

<sup>15</sup> Note there are two options for i.MX 6UltraLite EVK (DTB file) depending on size of CPU.

<sup>16</sup> For 4.1.15 kernel release, DTB file configures OOB IRQ versus SDIO in-band interrupts. Either 4.1.15 kernel binary can be used in this SDIO in-band interrupt configuration.

<sup>17</sup> Note there are two options for i.MX 6UL/ULL EVK (DTB file) depending on size of CPU.

**Table 7: Embedded Wi-Fi/Bluetooth Files**

Filename or Folder	Details
/lib/modules/\$(uname -r)/kernel/drivers/net/wireless/bcmdhd/bcmdhd.ko	WLAN loadable module (bcmdhd.ko) driver. Only used as workaround for kernel crash erratum when running MFGTEST on Linux 4.1.15_2.0.0 release. "uname -r" will not work when user overwrites default i.MX image with Murata-compiled kernel.  <b>NOTE:</b> This driver is built into both the (modified) <b>Linux 3.14.52_1.1.0</b> and <b>4.1.15_2.0.0</b> kernels. The default bcmdhd driver configuration is for OOB interrupts on all platforms except i.MX 6UL/6ULL EVK.
/lib/firmware/bcm/ZP_BCM4339/	<b>fw_bcmdhd.bin:</b> Client/STA/P2P firmware. <b>fw_bcmdhd_apsta.bin:</b> SoftAP or Wi-Fi hot spot firmware. <b>fw_bcmdhd_mfgtest.bin:</b> Manufacturing test firmware. <b>bcmdhd.&lt;Module&gt;.OOB.cal:</b> NVRAM/calibration file for Out-Of-Band interrupt mode. <b>bcmdhd.&lt;Module&gt;.SDIO.cal:</b> NVRAM/calibration file for SDIO in-band interrupt mode.  <b>NOTES:</b> <ul style="list-style-type: none"> <li>1DX_BCM4343W folder used for both 1DX and 1DX.</li> <li>1CK NVRAM/calibration files are copied from Murata Quick Start Patch to ZP folder. Both ZP and 1CK use same firmware.</li> </ul>
/lib/firmware/bcm/1BW_BCM43340	
/lib/firmware/bcm/1DX_BCM4343W	
/lib/firmware/bcm/SN8000_BCM43362	
/etc/firmware/BCM4335C0.ZP.hcd	Bluetooth patch-file for 1CK & ZP. Location/filename are relevant for correct initialization (i.e. otherwise "hciattach" call fails).
/etc/firmware/BCM43341B0.1BW.hcd	Bluetooth patch-file for 1BW. Location/filename are relevant for correct initialization (i.e. otherwise "hciattach" call fails).
/etc/firmware/BCM43430A1.1DX.hcd	Bluetooth patch-file for 1DX. Location/filename are relevant for correct initialization (i.e. otherwise "hciattach" call fails).
/bin/wl	WL tool is used for Wi-Fi RF and manufacturing tests. Also useful tool for initial bring-up and debug.
/usr/sbin/iw	Linux "iw" executable.
/usr/sbin/wpa_supplicant	WPA supplicant executable.
/usr/sbin/wpa_cli	WPA CLI tool.
/usr/bin/wpa_passphrase	WPA Passphrase generator.
/etc/wpa_supplicant.conf	WPA supplicant configuration file.
/usr/sbin/hostapd	Hostapd executable – manages wireless link in Soft AP mode.
/usr/sbin/hostapd_cli	Hostapd CLI tool.
/etc/hostapd.conf	Hostapd configuration file.
/etc/udhcpd.conf	DHCP server configuration file.
/etc/network/interfaces	Modify this file to automatically bring up "wlan0" interface. Also assign static IP address if desired.
/etc/network/if-pre-up.d/wpa-suplicant	Modify this file to configure "/sys/module/bcmdhd/parameters/firmware_path" and "/sys/module/bcmdhd/parameters/nvram_path" to the desired firmware and NVRAM configuration.
/usr/bin/hciattach	"hciattach" binary – used for initializing Bluetooth UART connection.
/usr/bin/hciconfig	"hciconfig" binary – used for configuring Bluetooth interface.
/usr/bin/hcitool	"hcitool" binary – used for controlling Bluetooth interface.
/usr/bin/iperf	iPerf throughput test tool on Linux 3.14.52_1.1.0 release.
/usr/bin/iperf3	iPerf throughput test tool on Linux 4.1.15_2.0.0 release.

## 5.1 Wi-Fi Interface Test/Verification

### 5.1.1 Useful Environment Setup on NXP Linux

Once the kernel has booted, there are a couple of quick commands (sequence is important) which make the terminal console easier to work on:

```
$ stty rows 80 cols 132 ← set your favorite row and column width here  
$ export TERM=ansi ← invoke this command after “stty”
```

### 5.1.2 Bringing Up Wi-Fi Interface

Given the modifications to Linux 3.14.52\_1.1.0 (using Murata Quick Start Binary patches), the operation of WLAN on both Linux kernel releases is equivalent. The only exception is when using MFGTEST mode on Linux 4.1.15\_2.0.0 release (erratum requiring loadable bcmhdh.ko module).

**NOTE: If the correct DTB file is not set (i.e. fdt\_file u-boot parameter), then any attempt to bring up the WLAN interface will fail. Please double check **Table 6** to make sure you are using the correct DTB configuration.**

With the built-in bcmhdh driver, the WLAN driver is loaded as the kernel boots. As part of driver loading sequence, the WLAN device is probed over the SDIO bus. As an example, we will bring up Wi-Fi when using Type ZP (BCM4339) module using the “standard” STA/Client firmware. You should see a (very similar) log as the kernel boots:

```
dhd_module_init in  
Power-up adapter 'DHD generic adapter'  
wifi_platform_bus_enumerate device present 1  
mmc2: queuing unknown CIS tuple 0x80 (2 bytes)  
mmc2: queuing unknown CIS tuple 0x80 (7 bytes)  
mmc2: queuing unknown CIS tuple 0x80 (6 bytes)  
mmc2: queuing unknown CIS tuple 0x91 (3 bytes)  
mmc2: new high speed SDIO card at address 0001  
F1 signature OK, socitype:0x1 chip:0x4339 rev:0x1 pkg:0x0 ← Type ZP/BCM4339 Chip ID  
DHD: dongle ram size is set to 786432(orig 786432) at 0x180000  
wifi_platform_get_mac_addr  
CFG80211-ERROR) wl_setup_wiphy : Registering Vendor80211)  
wl_create_event_handler(): thread:wl_event_handler:91 started  
CFG80211-ERROR) wl_event_handler : tsk Enter, tsk = 0xa85e143c  
dhd_attach(): thread:dhd_watchdog_thread:92 started  
dhd_attach(): thread:dhd_dpc:93 started  
dhd_deferred_work_init: work queue initialized  
Dongle Host Driver, version 1.141.92 (r) ← bcmhdh driver version  
Compiled in drivers/net/wireless/bcmhdh  
Register interface [wlan0] MAC: 00:90:4c:11:22:33
```

```
CFG80211-ERROR) wl_event_handler : was terminated
wl_destroy_event_handler(): thread:wl_event_handler:91 terminated OK
dhd_prot_ioctl : bus is down. we have nothing to do
```

You can ignore the “CFG80211-ERROR” messages. Some of the key output strings are highlighted with **comments in red**. Also you can ignore the incorrect “wlan0” MAC address output as the kernel boots.

**NOTE: The next step will fail if the “firmware\_path” and “nvram\_path” parameters are not correctly initialized for the bcmhdhd driver. These parameters can be set manually or automatically by editing the “/etc/network/if-pre-up.d/wpa-supPLICant” file. Here are the manual steps:**

```
$ echo /lib/firmware/bcm/ZP_BCM4339/fw_bcmhdhd.bin > /sys/module/bcmhdhd/parameters/firmware_path
$ echo /lib/firmware/bcm/ZP_BCM4339/bcmhdhd.ZP.SDIO.cal > /sys/module/bcmhdhd/parameters/nvram_path
```

⇒ For other modules, refer to **Table 8: Murata Module to Firmware/NVRAM Mapping**.

Now invoke “ifconfig wlan0 up” command to initialize the “wlan0” interface (**not necessary if automatic WLAN bring-up has been configured** so that “wlan0” interface is brought up automatically after kernel boots). Example output shown below:

```
$ ifconfig wlan0 up
Dongle Host Driver, version 1.141.92 (r) ← bcmhdhd driver version
Compiled in drivers/net/wireless/bcmhdhd
wl_android_wifi_on in
mmc2: queuing unknown CIS tuple 0x80 (2 bytes)
mmc2: queuing unknown CIS tuple 0x80 (7 bytes)
mmc2: queuing unknown CIS tuple 0x80 (6 bytes)
F1 signature OK, socitype:0x1 chip:0x4339 rev:0x1 pkg:0x0 ← Type ZP/BCM4339 Chip ID
DHD: dongle ram size is set to 786432(orig 786432) at 0x180000
dhdswdio_write_vars: Download, Upload and compare of NVRAM succeeded. ← NVRAM loaded successfully
dhd_bus_init: enable 0x06, ready 0x06 (waited 0us)
wifi_platform_get_mac_addr
Firmware up: op_mode=0x0005, MAC=98:f1:70:63:94:ac ← Unique WLAN MAC Address (Murata OUI)
Firmware version = wl0: Feb 17 2016 12:06:31 version 6.37.39.36 (r614533) ← firmware version number
dhd_wlfc_init(): successfully enabled bdcv2 tlv signaling, 79
dhd_wlfc_init(): wlfc_mode=0x0, ret=-23
wl_create_event_handler(): thread:wl_event_handler:381 started
CFG80211-ERROR) wl_event_handler : tsk Enter, tsk = 0xa85e143c
```

Again, you can ignore the “CFG80211-ERROR” messages. Some of the key output strings are highlighted with **comments in red**. Note that the correct WLAN MAC address is output when interface is initialized. Now if we invoke “ifconfig wlan0” command, we should see similar output:

```
$ ifconfig wlan0
```

```
wlan0 Link encap:Ethernet HWaddr 98:f1:70:63:94:ac
inet6 addr: fe80::9af1:70ff:fe63:94ac/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:21 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:3754 (3.6 KiB)
```

**NOTE:** no IP address is assigned yet to the “wlan0” interface. That will be done later **Section 5.1.6**.

**Table 8: Murata Module to Firmware/NVRAM Mapping**

Module	Folder	<b>Firmware:</b> STA/Client/P2P Soft AP Manufacturing Test	<b>NVRAM:</b> OOB IRQ SDIO In-Band	Additional Notes
1CK	/lib/firmware/bcm/ ZP_BCM4339	fw_bcmdhd.bin fw_bcmdhd_apsta.bin <sup>18</sup> fw_bcmdhd_mfgtest.bin	bcmdhd.1CK.OOB.cal bcmdhd.1CK.SDIO.cal	1CK shares same firmware as ZP.
ZP	/lib/firmware/bcm/ ZP_BCM4339	fw_bcmdhd.bin fw_bcmdhd_apsta.bin <sup>18</sup> fw_bcmdhd_mfgtest.bin	bcmdhd.ZP.OOB.cal bcmdhd.ZP.SDIO.cal	
1BW	/lib/firmware/bcm/ 1BW_BCM43340	fw_bcmdhd.bin fw_bcmdhd_apsta.bin fw_bcmdhd_mfgtest.bin	bcmdhd.1BW.OOB.cal bcmdhd.1BW.SDIO.cal	OOB IRQ is edge-sensitive.
1DX/ 1FX	/lib/firmware/bcm/ 1DX_BCM4343W	fw_bcmdhd.bin fw_bcmdhd_apsta.bin fw_bcmdhd_mfgtest.bin	bcmdhd.1DX.OOB.cal bcmdhd.1DX.SDIO.cal	Same firmware and NVRAM used for both Type 1DX and 1FX.
SN8000	/lib/firmware/bcm/ SN8000_BCM43362	fw_bcmdhd.bin fw_bcmdhd_apsta.bin fw_bcmdhd_mfgtest.bin	bcmdhd.SN8000.OOB.cal bcmdhd.SN8000.SDIO.cal	OOB IRQ is edge-sensitive. Make sure to use updated NVRAM for OOB IRQ.

### 5.1.3 STA/Client Mode: Scan for Visible Access Points

In this section two different/simple methods for scanning are presented: one uses the Cypress “wl” tool; the other uses the Linux “iw” command. If you don’t see a list of SSID’s and there are broadcasting Access Points (Wireless Routers) in range, then something is wrong. Please check antenna connection, setup, etc. Also note that the strength of received signals is important to do connectivity testing (i.e. ping, iperf, etc.). Very attenuated signals will be in the high 80’s or 90’s (see “RSSI” value). If close to an Access Point, the returned RSSI value should be between -30 to -50 dBm for a properly configured setup.

<sup>18</sup> With 1CK or ZP module on Linux 4.1.15 release, the “fw\_bcmdhd\_apsta.bin” causes a kernel crash when loaded for Soft AP mode. Workaround is to use “fw\_bcmdhd.bin” instead (firmware version 6.37.39.36). Also check [Cypress Murata i.MX Support Portal](#) for firmware updates.



### 5.1.3.1 Using “wl” Tool

The Cypress “wl” tool is a powerful tool which allows the user to configure any number of radio characteristics. It is most commonly used for RF testing/evaluation. However it is also convenient to do initial bring-up testing. The “wl” tool is integrated into the NXP image and provides a quick way to check/verify functionality. For more information on “wl” tool please reference the following documents:

- [“WL Tool Instructions”](#) on [“My Murata->Common Documents”](#)
- [“WL Tool for Embedded 802.11 Systems”](#) on [Cypress Linux Support Portal](#).

Now that Wi-Fi interface is up and running (having loaded the default “fw\_bcmdhd.bin” – STA/Client mode firmware), let’s do some basic testing to verify functionality. The “wl scan” command will initiate an active probe of all visible SSID’s. The “wl scanresults” will return a list of visible SSID’s.

**NOTE: "CFG80211-ERROR) wl\_notify\_scan\_status : scan is not ready" message is to be expected after invoking “wl scan”. If using a script, the delay in command sequence between “wl scan” and successive “wl scanresults” command is necessary. Otherwise “wl scanresults” returns nothing. Only after “CFG80211-ERROR” message logged to console, does “wl scanresults” report correct list of SSID’s.**

In following example, one active AP has SSID of “Murata\_5G”. Here are expected results:

```
$ wl scan
CFG80211-ERROR) wl_notify_scan_status : scan is not ready
$ wl scanresults
SSID: "Murata_5G"
Mode: Managed  RSSI: -41 dBm  SNR: 0 dB   noise: 0 dBm  Flags: RSSI on-channel  Channel: 136u
BSSID: 00:10:18:A9:75:22   Capability: ESS
Supported Rates: [ 6(b) 9 12(b) 18 24(b) 36 48 54 ]
VHT Capable:
  Chanspec: 5GHz channel 134 40MHz (0xd986)
  Primary channel: 136
  HT Capabilities: 40Mhz SGI20 SGI40
  Supported MCS : [ 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 32 ]
  VHT Capabilities:
  Supported VHT (tx) Rates:
    NSS: 1 MCS: 0-9
    NSS: 2 MCS: 0-9
    NSS: 3 MCS: 0-9
  Supported VHT (rx) Rates:
    NSS: 1 MCS: 0-9
    NSS: 2 MCS: 0-9
    NSS: 3 MCS: 0-9
WPS: V2.0 Configured
..... etc. ← More SSID listings follow here.
```

### 5.1.3.2 Using “iw” Linux Command

“iw” is the default Linux command line tool for controlling a WLAN interface. It provides an alternative to “wl” tool. One useful link to learn more about “iw” is on the “Linux Wireless wiki” [here](#). In following example of listing WLAN devices and performing a scan, one active AP has SSID of “Murata\_5G”. Here are expected results:

```
$ iw dev ← list available WLAN devices
```

```
phy#0
```

```
Interface wlan0
  ifindex 7
  wdev 0x1
  addr 90:b6:86:13:0d:b8
  type managed
```

```
$ iw dev wlan0 scan ← perform scan on “wlan0” interface
```

```
BSS 00:10:18:a9:75:22(on wlan0)
  TSF: 453912899134 usec (5d, 06:05:12)
  freq: 5680
  beacon interval: 100 TUs
  capability: ESS (0x0001)
  signal: -42.00 dBm
  last seen: 10 ms ago
  Information elements from Probe Response frame:
  SSID: Murata_5G
  Supported rates: 6.0* 9.0 12.0* 18.0 24.0* 36.0 48.0 54.0
  BSS Load:
    * station count: 1
    * channel utilisation: 1/255
    * available admission capacity: 0 [*32us]
  HT capabilities:
    Capabilities: 0x9ef
    RX LDPC
    HT20/HT40
    SM Power Save disabled
    RX HT20 SGI
    RX HT40 SGI
    TX STBC
    RX STBC 1-stream
    Max AMSDU length: 7935 bytes
    No DSSS/CCK HT40
    Maximum RX AMPDU length 65535 bytes (exponent: 0x003)
    Minimum RX AMPDU time spacing: 4 usec (0x05)
    HT RX MCS rate indexes supported: 0-23, 32
    HT TX MCS rate indexes are undefined
```

HT operation:

- \* primary channel: 136
- \* secondary channel offset: below
- \* STA channel width: any
- \* RIFS: 1
- \* HT protection: no
- \* non-GF present: 1
- \* OBSS non-GF present: 0
- \* dual beacon: 0
- \* dual CTS protection: 0
- \* STBC beacon: 0
- \* L-SIG TXOP Prot: 0
- \* PCO active: 0
- \* PCO phase: 0

Extended capabilities: BSS Transition, 6

VHT capabilities:

VHT Capabilities (0x0f8259b2):

Max MPDU length: 11454

Supported Channel Width: neither 160 nor 80+80

RX LDPC

short GI (80 MHz)

TX STBC

SU Beamformer

SU Beamformee

VHT RX MCS set:

1 streams: MCS 0-9

2 streams: MCS 0-9

3 streams: MCS 0-9

4 streams: not supported

5 streams: not supported

6 streams: not supported

7 streams: not supported

8 streams: not supported

VHT RX highest supported: 0 Mbps

VHT TX MCS set:

1 streams: MCS 0-9

2 streams: MCS 0-9

3 streams: MCS 0-9

4 streams: not supported

5 streams: not supported

6 streams: not supported

7 streams: not supported

8 streams: not supported

VHT TX highest supported: 0 Mbps

VHT operation:

- \* channel width: 0 (20 or 40 MHz)
- \* center freq segment 1: 134
- \* center freq segment 2: 0
- \* VHT basic MCS set: 0x0000

WPS: \* Version: 1.0

- \* Wi-Fi Protected Setup State: 2 (Configured)
- \* Response Type: 3 (AP)
- \* UUID: 5d3cf72c-8c5e-5ebd-afbe-bd3612bf6530
- \* Manufacturer: Broadcom
- \* Model: Broadcom
- \* Model Number: 123456
- \* Serial Number: 0465
- \* Primary Device Type: 6-0050f204-1
- \* Device name: BroadcomAP
- \* Config methods: Display
- \* RF Bands: 0x2
- \* Unknown TLV (0x1049, 6 bytes): 00 37 2a 00 01 20

WMM: \* Parameter version 1

- \* u-APSD
- \* BE: CW 15-1023, AIFSN 3
- \* BK: CW 15-1023, AIFSN 7
- \* VI: CW 7-15, AIFSN 2, TXOP 3008 usec
- \* VO: CW 3-7, AIFSN 2, TXOP 1504 usec

..... etc. [← More SSID listings follow here.](#)

## 5.1.4 STA/Client Mode: Connecting to Unsecured Access Point or Wireless Router

**NOTE:** In the following test sequences of using “wl” tool or “iw” command, the WLAN interface is not assigned an IP address. That is done later in **Section 5.1.6** where connectivity testing is performed.

### 5.1.4.1 Using “wl” Tool

To verify connectivity quickly, associating to an unsecured Access Point is a quick test. Here is the syntax for the “wl join” command:

```
$ wl join
```

```
join Join a specified network SSID.
```

```
Usage: join <ssid> [key <0-3>:xxxxx] [imode bss|ibss] [amode  
open|shared|openshared|wpa|wpapsk|wpa2|wpa2psk|wpanone] [options]
```

```
Options:
```

- b MAC, --bssid=MAC BSSID (xx:xx:xx:xx:xx:xx) to scan and join
- c CL, --chanspecs=CL chanspecs (comma or space separated list)  
prescanned uses channel and bssid list from scanresults
- p, -passive: force passive assoc scan (useful for P2P)

Following example with “Murata\_5G” SSID, now invoke “wl join”:

```
$ wl join Murata_5G ← connect to “Murata_5G” Access Point
wl_bss_connect_done succeeded with 00:10:18:a9:75:22
wl_bss_connect_done succeeded with 00:10:18:a9:75:22
```

Check status of connection with “wl assoc” command:

```
$ wl assoc ← check association status
SSID: "Murata_5G"
Mode: Managed  RSSI: -38 dBm  SNR: 0 dB   noise: -92 dBm  Flags: RSSI on-channel  Channel: 149/80
BSSID: 00:10:18:A9:75:22   Capability: ESS
Supported Rates: [ 6(b) 9 12(b) 18 24(b) 36 48 54 ]
VHT Capable:
  Chanspec: 5GHz channel 155 80MHz (0xe09b)
  Primary channel: 149
  HT Capabilities: 40Mhz SGI20 SGI40
  Supported MCS : [ 0 1 2 3 4 5 6 7 ]
  VHT Capabilities:
  Supported VHT (tx) Rates:
    NSS: 1 MCS: 0-9
    NSS: 2 MCS: 0-9
    NSS: 3 MCS: 0-9
  Supported VHT (rx) Rates:
    NSS: 1 MCS: 0-9
    NSS: 2 MCS: 0-9
    NSS: 3 MCS: 0-9
WPS: V2.0 Configured
```

#### **5.1.4.2 Using “iw” Linux Command**

Following example with “Murata\_5G” SSID, now invoke “iw” connect command:

```
$ iw dev wlan0 connect Murata_5G
CFG80211-ERROR) wl_cfg80211_connect : Connecting withff:ff:ff:ff:ff:ff channel (0) ssid "Murata_5G", len (9)

wl_bss_connect_done succeeded with 00:10:18:a9:75:22
wl_bss_connect_done succeeded with 00:10:18:a9:75:22
```

Check status of connection with “iw” link command:

```
$ iw dev wlan0 link
Connected to 00:10:18:a9:75:22 (on wlan0)
  SSID: Murata_5G
  freq: 5745
  signal: -38 dBm
  tx bitrate: 390.0 MBit/s
```

## 5.1.5 STA/Client Mode: Connecting to Secured Access Point or Wireless Router (WPA2-PSK)

In this section, we will cover two different approaches to accomplish STA/Client association to a WPA2-PSK secured Access Point. One is using the embedded “wpa\_cli” tool, the other is configuring the “/etc/wpa\_supplicant.conf” file.

### Important Dependencies:

- WLAN device must be configured correctly (refer to **Section 5.1.2**).
- WPA supplicant must be up and running.

#### 5.1.5.1 Using “wpa\_cli” Command

“wpa\_cli” can only be invoked once the “wlan0” interface is configured and the WPA supplicant is running. The user can follow this command sequence to establish a secure client connection to an Access Point with WPA2-PSK authentication.

Prior to running “wpa\_cli”, you might like to back up default/previous “/etc/wpa\_supplicant.conf” file in case you overwrite it:

```
$ cp /etc/wpa_supplicant.conf /etc/wpa_supplicant.conf.bak
```

Now invoke “wpa\_cli” which brings up the tool in interactive mode:

```
$ wpa_cli -i wlan0
wpa_cli v2.5
Copyright (c) 2004-2015, Jouni Malinen <j@w1.fi> and contributors
```

This software may be distributed under the terms of the BSD license.  
See README for more details.

Interactive mode

Following previous example, we configure the “Murata\_5G” AP with WPA2-PSK security and associate to it using “wpa\_cli” tool with following commands:

```
> remove_network all ← tear down any existing network connections
OK
> status ← check current status
wpa_state=INACTIVE
ip_address=192.168.1.101
address=90:b6:86:13:0d:b8
uuid=236d8266-6abb-58ef-8605-8a8b896c4f3f
> scan ← initiate a scan
OK
<3>CTRL-EVENT-SCAN-STARTED
<3>CTRL-EVENT-SCAN-RESULTS
```

<3>WPS-AP-AVAILABLE

<3>CTRL-EVENT-NETWORK-NOT-FOUND

> scan\_results ← list results of scan

bssid / frequency / signal level / flags / ssid

00:10:18:a9:75:22 5660 -38 [WPA2-PSK-CCMP][WPS][ESS] Murata\_5G

00:90:4c:0d:c1:35 2462 -39 [WPA2-PSK-CCMP][WPS][ESS] Murata\_2G

...

> add\_network ← add a network. This returns integer value which is then used for setting parameters.

0

> set\_network 0 ssid "Murata\_5G" ← set SSID to "Murata\_5G"

OK

> set\_network 0 psk "your\_passphrase" ← set WPA passphrase

OK

> enable 0 ← enable network connection. If ssid and passphrase set correctly, connection will be established.

OK

<3>CTRL-EVENT-SCAN-STARTED

<3>CTRL-EVENT-SCAN-RESULTS

<3>WPS-AP-AVAILABLE

<3>Trying to associate with SSID 'Murata\_5G'

CFG80211-ERROR) wl\_cfg80211\_connect : Connecting with 00:10:18:a9:75:22 channel (132) ssid "Murata\_5G",  
le)

wl\_bss\_connect\_done succeeded with 00:10:18:a9:75:22 ← connection established

<3>Associated with 00:10:18:a9:75:22

<3>WPA: Key negotiation completed with 00:10:18:a9:75:22 [PTK=CCMP GTK=CCMP]

<3>CTRL-EVENT-CONNECTED - Connection to 00:10:18:a9:75:22 completed [id=0 iwlan\_bss\_connect\_done  
succeeded 2

d\_str=]

> status ← now verify that connection is established

bssid=00:10:18:a9:75:22

freq=5660

ssid=Murata\_5G

id=0

mode=station

pairwise\_cipher=CCMP

group\_cipher=CCMP

key\_mgmt=WPA2-PSK

wpa\_state=COMPLETED

ip\_address=192.168.1.101

address=90:b6:86:13:0d:b8

uuid=236d8266-6abb-58ef-8605-8a8b896c4f3f

>

> save\_config ← save current configuration: this overwrites "/etc/wpa\_supplicant.conf" file!

OK

> quit ← exit wpa\_cli interactive mode

Now let's check contents of "/etc/wpa\_supplicant.conf" file:

```
$ more /etc/wpa_supplicant.conf
```

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1
```

```
network={
    scan_ssid=1
    ssid="Murata_5G"
    psk="your_passphrase"
}
```

**NOTE: Using "save\_config" command in "wpa\_cli" interactive mode allows us to easily generate the "/etc/wpa\_supplicant.conf" file for a specific/desired configuration.**

### **5.1.5.2 Using "wpa\_supplicant.conf" file**

Another approach to establishing a WPA2-PSK secure connection is to properly configure the "/etc/wpa\_supplicant.conf" file and let the wpa\_supplicant establish the connection. The default contents of "/etc/wpa\_supplicant.conf" file are:

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1
```

```
network={
    key_mgmt=NONE
}
```

With the default configuration, the WPA supplicant will establish a connection with a random Access Point that has no authentication scheme enabled (i.e. "open").

Using "Murata\_5G" SSID example, the relevant/modified contents of the "/etc/wpa\_supplicant.conf" file (already shown in **Section 5.1.5.1**) is:

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1
```

```
network={
    scan_ssid=1
    ssid="Murata_5G"
    psk="your_passphrase"
}
```



To establish a secured WPA2-PSK connection by only modifying “/etc/wpa\_supplicant.conf” file, we need to follow these steps:

- Modify “/etc/wpa\_supplicant.conf” file to configure desired connection.
- Kill WPA supplicant process and re-start it.
- Re-started WPA supplicant will read in modified configuration file and associate to AP (Wireless Router) accordingly.

To kill the WPA supplicant process and re-start it:

```
$ killall wpa_supplicant
$ wpa_supplicant -i wlan0 -D nl80211 -c /etc/wpa_supplicant.conf -B
```

Expected output for these commands follows (driver taken down then brought up again):

```
$ killall wpa_supplicant ← take down WPA supplicant
CFG80211-ERROR) wl_cfg80211_disconnect : Reason 3
CFG80211-ERROR) wl_is_linkdown : Link down Reason : WLC_E_LINK
link down if wlan0 may call cfg80211_disconnected. event : 16, reason=2 from 00:10:18:a9:75:22
root@imx6ulevk:~# CFG80211-ERROR) wl_is_linkdown : Link down Reason : WLC_E_DEAUTH
CFG80211-ERROR) wl_event_handler : was terminated
wl_destroy_event_handler(): thread:wl_event_handler:20c terminated OK
wl_android_wifi_off in
dhd_wlfc_deinit():3277, maintain HOST RXRERORDER flag in tvl
dhd_wlfc_deinit():3291 successfully disabled bdcv2 tlv signaling, 64

$ wpa_supplicant -i wlan0 -D nl80211 -c /etc/wpa_supplicant.conf -B ← re-start the WPA supplicant
Successfully initialized wpa_supplicant
rfkill: Cannot open RFKILL control device

Dongle Host Driver, version 1.141.92 (r)
Compiled in drivers/net/wireless/bcmdhd
wl_android_wifi_on in
mmc0: queuing unknown CIS tuple 0x80 (2 bytes)
mmc0: queuing unknown CIS tuple 0x80 (7 bytes)
mmc0: queuing unknown CIS tuple 0x80 (6 bytes)
F1 signature OK, socitype:0x1 chip:0x4339 rev:0x1 pkg:0x0
DHD: dongle ram size is set to 786432(orig 786432) at 0x180000
dhdsdio_write_vars: Download, Upload and compare of NVRAM succeeded.
dhd_bus_init: enable 0x06, ready 0x06 (waited 0us)
wifi_platform_get_mac_addr
Firmware up: op_mode=0x0005, MAC=90:b6:86:13:0d:b8
Firmware version = wl0: Feb 17 2016 12:06:31 version 6.37.39.36 (r614533)
dhd_wlfc_init(): successfully enabled bdcv2 tlv signaling, 79
dhd_wlfc_init(): wlfc_mode=0x0, ret=-23
wl_create_event_handler(): thread:wl_event_handler:376 started
CFG80211-ERROR) wl_event_handler : tsk Enter, tsk = 0x8850143c
```

```
root@imx6ulevk:~# CFG80211-ERROR) wl_cfg80211_connect : Connecting with 00:10:18:a9:75:22 channel (132) ssid "Murata_5G", len (9)
```

```
wl_bss_connect_done succeeded with 00:10:18:a9:75:22  
wl_bss_connect_done succeeded with 00:10:18:a9:75:22
```

### 5.1.6 STA/Client Mode: Basic WLAN Connectivity Testing

Prior to running connectivity tests, we need to assign an IP address to the “wlan0” interface. If the subnet address is known, one option is to use manual “ifconfig” command to assign an IP address to “wlan0”. Here is an example “ifconfig” command assuming subnet of 192.168.1.255:

```
$ ifconfig wlan0 192.168.1.111 netmask 255.255.255.0
```

Alternatively, we can use the DHCP client to obtain an address (assuming wireless network associated to has a DHCP server):

```
$ udhcpc -i wlan0 ← Command to invoke DHCP client and obtain IP address.  
udhcpc (v1.23.1) started  
Sending discover...  
Sending select for 192.168.1.100...  
Lease of 192.168.1.100 obtained, lease time 86400  
/etc/udhcpc.d/50default: Adding DNS 192.168.1.1
```

The most basic connectivity test is to use the “ping” command. In this example, we assume the wireless router (associated to) has an IP address of 192.168.1.1:

```
$ ping 192.168.1.1  
PING 192.168.1.1 (192.168.1.1): 56 data bytes  
64 bytes from 192.168.1.1: seq=0 ttl=64 time=12.686 ms  
64 bytes from 192.168.1.1: seq=1 ttl=64 time=10.227 ms  
64 bytes from 192.168.1.1: seq=2 ttl=64 time=10.053 ms  
64 bytes from 192.168.1.1: seq=3 ttl=64 time=10.341 ms  
64 bytes from 192.168.1.1: seq=4 ttl=64 time=12.364 ms  
^C ← Enter <CTRL-C> to terminate ping session  
--- 192.168.1.1 ping statistics ---  
5 packets transmitted, 5 packets received, 0% packet loss ← Indicates that no packets were dropped  
round-trip min/avg/max = 10.053/11.134/12.686 ms
```

If we want to do more sophisticated connectivity tests, the “iperf” tool is available on Linux 3.14.52\_1.1.0 release (version 2.0.5). The executable is renamed to “iperf3” on Linux 4.1.15\_2.0.0 release (version 3.1). To run throughput performance tests with “iperf” you need at least one client and one server. Typically the user will install the “iperf” utility on a Windows or Linux PC which is wired to the associated wireless router. For more information on the “iperf” tool refer to [this link](#).

## 5.1.7 Wi-Fi Direct Testing

In this section we use the “wpa\_cli” tool to configure the i.MX6/Murata Wi-Fi platform as a P2P Group Owner (P2P GO). Another device (i.e. another i.MX platform or smartphone) is required to act as the P2P Client. Together the P2P GO and Client devices establish a P2P Group.

### Important Dependencies:

- WLAN device must be configured correctly (refer to **Section 5.1.2**).
- WPA supplicant must be up and running.
- “wpa\_supplicant” and “wpa\_cli” must be updated to support Wi-Fi Direct (refer to **Section 4.4** and **Table 7**).
- “bcmhdh” driver must be patched (on Linux 3.14.52\_1.1.0 release) to fix P2P erratum.

Prior to running “wpa\_cli”, you might like to back up default/previous “/etc/wpa\_supplicant.conf” file in case you overwrite it:

```
$ cp /etc/wpa_supplicant.conf /etc/wpa_supplicant.conf.bak
```

Now we can invoke “wpa\_cli” tool to configure the P2P interface:

```
$ wpa_cli -i wlan0
```

```
> remove_network all ← Let's remove any network association
```

```
CFG80211-ERROR) wl_cfg80211_disconnect : Reason 3
```

```
CFG80211-ERROR) wl_is_linkdown : Link down Reason : WLC_E_LINK  
link down if wlan0 may call cfg80211_disconnected. event : 16, reason=2 from e8:fc:af:f6:a2:cd  
OK  
<3>CTRL-EVENT-DISCONNECTED bssid=e8:fc:af:f6:a2:cd reason=3 locally_generated=1
```

```
>> status ← Check status now
```

```
wpa_state=INACTIVE  
p2p_device_address=92:b6:86:13:0d:b8  
address=90:b6:86:13:0d:b8  
uuid=236d8266-6abb-58ef-8605-8a8b896c4f3f
```

```
> p2p_group_add ← Add P2P Group  
Register interface [p2p-wlan0-0] MAC: 92:b6:86:13:8d:b8
```

```
CFG80211-ERROR) wl_cfg80211_add_virtual_iface : virtual interface(p2p-wlan0-0) is created net attach done  
CFG80211-ERROR) dhd_cfg80211_set_p2p_info : Set : op_mode=0x0045  
OK  
> CFG80211-ERROR) wl_cfg80211_del_station : Disconnect STA : ff:ff:ff:ff:ff:ff scb_val.val 3  
CFG80211-ERROR) wl_cfg80211_set_channel : netdev_ifidx(8), chan_type(1) target channel(11)
```

```
<3>P2P-GROUP-STARTED p2p-wlan0-0 GO ssid="DIRECT-6G" freq=2462 passphrase="UVBy8n5J"  
go_dev_addr=92:b6:86:13:0d:b8  
CFG80211-ERROR) wl_notify_rx_mgmt_frame : WLC_GET_BSSID error -17
```

```
> quit      ← Quit "wpa_cli" tool
```

After running "p2p\_group add" command, the following are set:

- P2P virtual interface (see results of "ifconfig" command below)
- P2P SSID, with selected channel and secure passphrase needed by other P2P client to associate.

To verify new virtual P2P interface, we just invoke "ifconfig" command:

```
$ ifconfig
```

```
p2p-wlan0-0 Link encap:Ethernet HWaddr 92:b6:86:13:8d:b8      ← new P2P interface  
inet6 addr: fe80::90b6:86ff:fe13:8db8/64 Scope:Link  
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1  
RX packets:16 errors:0 dropped:0 overruns:0 frame:0  
TX packets:23 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:1000  
RX bytes:3590 (3.5 KiB) TX bytes:3863 (3.7 KiB)
```

```
wlan0 Link encap:Ethernet HWaddr 90:b6:86:13:0d:b8      ← existing "wlan0" interface  
inet6 addr: fe80::92b6:86ff:fe13:db8/64 Scope:Link  
UP BROADCAST MULTICAST MTU:1500 Metric:1  
RX packets:184 errors:0 dropped:0 overruns:0 frame:0  
TX packets:26 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:1000  
RX bytes:29495 (28.8 KiB) TX bytes:4009 (3.9 KiB)
```

To test connectivity, we can assign manual IP address to P2P interface:

```
$ ifconfig p2p-wlan0-0 192.168.2.1 netmask 255.255.255.0
```

Now connect a P2P Client such as smartphone (or similar) and force same IP address with same subnet address. We can now ping from either interface.

## 5.1.8 Soft AP or Wi-Fi Hot Spot Testing

In this section we use the “hostapd” supplicant to configure the i.MX6/Murata Wi-Fi platform as a “Soft AP” or “Wi-Fi hot spot”. Both unsecured and secured configurations are presented.

### Important Dependencies:

- “hostapd”, “hostapd.conf”, and “udhcpd.conf” files must be updated to support Wi-Fi hot spot configuration (refer to **Section 4.4** and **Table 7**).
- The correct WLAN firmware must be loaded for this mode of operation<sup>19</sup>.

First off, we need to kill the WPA supplicant and make sure we are using the correct firmware (“fw\_bcmdhd\_apsta.bin”). Type following commands (example 1DX on SoloX):

```
$ killall wpa_supplicant
$ echo /lib/firmware/bcm/1DX_BCM4343W/fw_bcmdhd_apsta.bin > /sys/module/bcmdhd/parameters/firmware_path
$ echo /lib/firmware/bcm/1DX_BCM4343W/bcmdhd.1DX.OOB.cal > /sys/module/bcmdhd/parameters/nvram_path
```

Using the default settings in “hostapd.conf” file, the configuration is setup for no authentication. We can start up the SoftAP with following commands:

```
$ ifconfig wlan0 192.168.1.1 up
$ udhcpd -S -I 192.168.1.1 /etc/udhcpd.conf
$ hostapd -B /etc/hostapd.conf
```

After invoking the “hostapd” command, the following log is expected:

```
$ hostapd -B /etc/hostapd.conf
Configuration file: /etc/hostapd.conf
rfkill: Cannot open RFKILL control device
CFG80211-ERROR) wl_cfg80211_del_station : Disconnect STA : ff:ff:ff:ff:ff:ff scb_val.val 3
Using interface wlan0 with hwaddr 90:b6:86:13:0d:b8 and ssid "test"
CFG80211-ERROR) wl_cfg80211_set_channel : netdev_ifidx(7), chan_type(1) target channel(1)
CFG80211-ERROR) wl_cfg80211_parse_ies : No WPSIE in beacon
CFG80211-ERROR) wl_cfg80211_parse_ies : No WPSIE in beacon
_dhd_wlfc_mac_entry_update():1649, entry(32)
wlan0: interface state UNINITIALIZED->ENABLED
wlan0: AP-ENABLED
```

We can now associate from another client device and ping the “wlan0” interface (in this example 192.168.1.1).

---

<sup>19</sup> With 1CK or ZP module and Linux 4.1.15 release, the “fw\_bcmdhd\_apsta.bin” causes a kernel crash when loaded for Soft AP mode. Workaround is to use “fw\_bcmdhd.bin” instead (firmware version 6.37.39.36). Also check [Cypress Murata i.MX Support Portal](#) for firmware updates.

For authenticating in secure fashion, we only have to change the “/etc/hostapd.conf” file to uncomment/configure the “wpa” and “wpa\_passphrase” lines correctly:

“#wpa=1” → “wpa=1”

“#wpa\_passphrase=secret passphrase” → “wpa\_passphrase=password123”

## 5.1.9 WLAN Manufacturing or RF Testing

There is specific firmware (“fw\_bcmdhd\_mfgtest.bin”) that is included for manufacturing test mode. This mode allows the end user to:

- Run specific RF tests to qualify the module during hardware evaluation phase. Typically this is done with Murata Wi-Fi/BT EVK.
- Validate RF performance on hardware prototypes and final shipping product.
- Perform RF validation tests on manufacturing line.

To run this specific test mode, the Cypress’ “wl” tool needs to be used (present on default NXP image at “/bin/wl”). For more information on using “wl” tool, you can reference the “My Murata” and Cypress support portals.

**NOTE: the WPA supplicant cannot run during manufacturing test mode. It will cause a kernel crash. Therefore the first critical step before running the bcmdhd driver in this mode is to kill the WPA supplicant process.**

There are some important differences regarding procedural steps for running manufacturing test mode on either Linux release. The following sub-sections explain those differences in detail.

➔ Refer to **Section 4.5** to ensure you have the latest NVRAM and firmware updates **prior** to running RF qualification tests on Murata Wi-Fi/BT modules.

### 5.1.9.1 Running Manufacturing Test Mode on Linux 3.14.52\_1.1.0 Release

For the 3.14.52\_1.1.0 release no modifications are necessary to run the manufacturing test mode. Once the kernel has booted, suggested sequence of steps (using ZP with kernel/bcmdhd configured for SDIO in-band interrupts) are shown below.

First of make sure to kill the WPA supplicant process with expected output:

```
$ killall wpa_supplicant
CFG80211-ERROR) wl_cfg80211_disconnect : root@imx6sx_all:~# Reason 3
CFG80211-ERROR) wl_is_linkdown : Link down Reason : WLC_E_LINK
link down if wlan0 may call cfg80211_disconnected. event : 16, reason=2 from 00:10:18:a9:75:22
cfg80211: Calling CRDA to update world regulatory domain
dhd_set_mcast_list_handler: interface info not available/down
wl_android_wifi_off in
dhd_wlfc_deinit():3274, maintain HOST RXRERODER flag in tvl
dhd_wlfc_deinit():3288 successfully disabled bdcv2 tlv signaling, 64
```

In case the WPA supplicant is not running, let's use "ifconfig" command to ensure interface is down:

```
$ ifconfig wlan0 down
wl_android_wifi_off in
dhd_wlfc_deinit():3274, maintain HOST RXRERORDER flag in tvl
dhd_wlfc_deinit():3288 successfully disabled bdcv2 tlv signaling, 64
```

Next let's setup correct NVRAM and firmware path/filenames before bringing up "wlan0" interface:

```
$ echo /lib/firmware/bcm/ZP_BCM4339/fw_bcmdhd_mfgtest.bin > /sys/module/bcmdhd/parameters/firmware_path
$ echo /lib/firmware/bcm/ZP_BCM4339/bcmdhd.ZP.OOB.cal > /sys/module/bcmdhd/parameters/nvram_path
$ ifconfig wlan0 up
```

After invoking "ifconfig" command, expected output is:

```
Dongle Host Driver, version 1.141.72 (r)
Compiled from
wl_android_wifi_on in
mmc2: queuing unknown CIS tuple 0x80 (2 bytes)
mmc2: queuing unknown CIS tuple 0x80 (7 bytes)
mmc2: queuing unknown CIS tuple 0x80 (6 bytes)
F1 signature OK, socitype:0x1 chip:0x4339 rev:0x1 pkg:0x0
DHD: dongle ram size is set to 786432(orig 786432) at 0x180000
dhdsdio_write_vars: Download, Upload and compare of NVRAM succeeded.
dhd_bus_init: enable 0x06, ready 0x06 (waited 0us)
dhd_preinit_ioctls : Set IOCTL response time for Manufacturing Firmware
wifi_platform_get_mac_addr
Firmware up: op_mode=0x0200, MAC=98:f1:70:63:94:ac
Firmware version = wl0: Jul 16 2015 22:50:52 version 6.37.39.16 (r550328 WLTEST) ← MFGTEST mode
dhd_wlfc_init(): successfully enabled bdcv2 tlv signaling, 79
dhd_wlfc_init(): wlfc_mode=0x0, ret=-23
```

Successfully loading of MFGTEST firmware can also be verified with following "wl" command:

```
$ wl ver
1.107 RC5.0
wl0: Jul 16 2015 22:50:52 version 6.37.39.16 (r550328 WLTEST) ← MFGTEST firmware
```

⇒ **We can now run RF testing using "wl" command.**

### 5.1.9.2 Running Manufacturing Test Mode on Linux 4.1.15\_2.0.0 Release

Unlike the 3.14.52\_1.1.0 release, running the manufacturing test mode on Linux 4.1.15\_2.0.0 release requires special configuration and steps. The underlying reason for this is an erratum: kernel crash when using bcmdhd built-in driver on 4.1.15 release. We use a workaround to avoid the kernel crash which consists of building the WLAN (bcmdhd) driver as a separate loadable module. As such we need to use a specific manufacturing test kernel and “bcmdhd.ko” module: both provided in the [Murata i.MX L4.1.15 2.0.0 Quick Start Binary Patch](#).

Prior to running manufacturing test mode, we need to make sure the correct kernel and “bcmdhd.ko” loadable module are copied over to the (micro) SD card. With the (micro) SD card mounted and the Quick Start Binary Patch extracted, execute the following commands (note that we are renaming the files):

```
$ cd <Murata 4.1.15 Quick Start Extracted Folder>
$ sudo mkdir /media/<root file system string>/lib/modules/<4.1.15...>/kernel/drivers/net/wireless
$ sudo mkdir /media/<root file system string>/lib/modules/<4.1.15...>/kernel/drivers/net/wireless/bcmdhd
$ sudo cp MFGTEST/bcmdhd.mfgtest.ko /media/<root file system
string>/lib/modules/<4.1.15...>/kernel/drivers/net/wireless/bcmdhd/bcmdhd.ko
$ sudo cp MFGTEST/zImage.mfgtest /media/<i.MX Boot Folder Name>/zImage
```

**NOTE: When running manufacturing test mode on Linux 4.1.15 release, all i.MX platforms run with SDIO in-band interrupts configured. This approach is taken to simplify support given the MFGTEST mode does not require OOB IRQ and i.MX 6UL/ULL EVK’s only support SDIO in-band interrupts in default hardware configuration.**

After connecting modified (micro) SD card and booting i.MX platform, we can execute the following commands to bring up WLAN interface in manufacturing test mode (example shown uses Type ZP module).

First of make sure to kill the WPA supplicant process:

```
$ killall wpa_supplicant
```

On the 4.1.15 release, the WPA supplicant is automatically brought up when WLAN interface is initialized. **To avoid a kernel crash, we need to rename the “/usr/bin/wpa\_supplicant” binary:**

```
$ mv /usr/sbin/wpa_supplicant /usr/sbin/wpa_supplicant2
```

Now we can bring up the WLAN interface in manufacturing test mode:

```
insmod /lib/modules/<4.1.15...>/kernel/drivers/net/wireless/bcmdhd/bcmdhd.ko
nvram_path=/lib/firmware/bcm/ZP_BCM4339/bcmdhd.ZP.SDIO.cal
firmware_path=/lib/firmware/bcm/ZP_BCM4339/fw_bcmdhd_mfgtest.bin dhd_oob_disable=1
```

**NOTE: One of the new/key parameters when invoking “insmod” command is “dhd\_oob\_disable=1” which forces SDIO in-band interrupt configuration.**



After invoking “insmod” command, expected output is:

```
dhd_module_init in
Power-up adapter 'DHD generic adapter'
wifi_platform_bus_enumerate device present 1
mmc0: queuing unknown CIS tuple 0x80 (2 bytes)
mmc0: queuing unknown CIS tuple 0x80 (7 bytes)
mmc0: queuing unknown CIS tuple 0x80 (6 bytes)
mmc0: queuing unknown CIS tuple 0x91 (3 bytes)
mmc0: new high speed SDIO card at address 0001
F1 signature OK, socitype:0x1 chip:0x4339 rev:0x1 pkg:0x0
DHD: dongle ram size is set to 786432(orig 786432) at 0x180000
wifi_platform_get_mac_addr
CFG80211-ERROR) wl_setup_wiphy : Registering Vendor80211)
wl_create_event_handler(): thread:wl_event_handler:334 started
CFG80211-ERROR) wl_event_handler : tsk Enter, tsk = 0x8910143c
dhd_attach(): thread:dhd_watchdog_thread:335 started
dhd_attach(): thread:dhd_dpc:336 started
dhd_deferred_work_init: work queue initialized
dhdsdio_htclk: HT Avail request NO ERROR: retried 0 times.
dhdsdio_write_vars: Download, Upload and compare of NVRAM succeeded.
dhdsdio_htclk: HT Avail request NO ERROR: retried 0 times.
dhd_bus_init: enable 0x06, ready 0x06 (waited 0us)
dhd_preinit_ioctls : Set IOCTL response time for Manufacturing Firmware
wifi_platform_get_mac_addr
Firmware up: op_mode=0x0200, MAC=90:b6:86:13:0d:b8
Firmware version = wl0: Feb 17 2016 12:11:33 version 6.37.39.36 (r614533 WLTEST) ← MFGTEST firmware
dhd_wlfc_init(): successfully enabled bdcv2 tlv signaling, 79
dhd_wlfc_init(): wlfc_mode=0x0, ret=-23
Dongle Host Driver, version 1.141.92 (r)
Compiled in drivers/net/wireless/bcmdhd
Register interface [wlan0] MAC: 90:b6:86:13:0d:b8
```

Successfully loading of MFGTEST firmware can also be verified with following “wl” command:

```
$ wl ver
1.107 RC5.0
wl0: Feb 17 2016 12:11:33 version 6.37.39.36 (r614533 WLTEST) ← MFGTEST firmware
```

⇒ **We can now run RF testing using “wl” command.**

**NOTE: To reconfigure file system for normal operation (not manufacturing test mode) which requires WPA supplicant, you will have to revert WPA supplicant binary to its default name of “/usr/sbin/wpa\_supplicant”.**

## 5.2 Bluetooth Interface Test/Verification

For Murata modules supporting Bluetooth, we can verify the HCI UART connection by invoking “hciattach”, bringing up the interface with “hciconfig” and then invoking “hcidtool scan” to see what Bluetooth devices are visible. The Bluetooth test commands vary depending on interconnect: which GPIO toggles the BT\_REG\_ON signal, and the UART port connection. It is necessary to reset the Bluetooth core before trying to initialize it (i.e. download Bluetooth “patch file” firmware – located in “/etc/firmware” folder). See **Table 9** below: hyperlinks to platform-specific subsections are provided so user can copy/paste/invoke entire command sequence from the console.

**NOTE:** the i.MX 6UL EVK has different GPIO’s for BT\_REG\_ON depending on which Linux version is being used – 3.14.52\_1.1.0 or 4.1.15.

**Table 9: GPIO and UART Settings for Bluetooth Tests**

i.MX6 Platform	KERNEL VERSION	GPIO/UART Configuration	Notes
i.MX 7Dual SDB	3.14.52/ 4.1.15	<a href="#">GPIO119; UART6</a>	
i.MX 6Q(P)/DL SDB/SDP	3.14.52/ 4.1.15	<a href="#">GPIO2; UART5</a>	
i.MX 6SoloX SDB	3.14.52/ 4.1.15	<a href="#">GPIO171; UART3</a>	
i.MX 6SoloLite EVK	3.14.52/ 4.1.15	<a href="#">GPIO145; UART4</a>	Hardware provides BT connectivity but software does not support it. Needs modified DTB file.
i.MX 6UL EVK	3.14.52	<a href="#">GPIO252; UART2</a>	GPIO252 does not allow its direction to be set. Always output.
i.MX 6UL/ULL EVK	4.1.15	<a href="#">GPIO508; UART2</a>	GPIO508 does not allow its direction to be set. Always output.

When controlling the “GPIO register/BT\_REG\_ON signal”, we drive BT\_REG\_ON low (resets BT core) and then drive it high again (takes BT core out of reset). Test command sequence is as follows:

```
echo [GPIO #] > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio[GPIO #]/direction ← SKIP this step for i.MX 6UL/ULL EVK
echo 0 > /sys/class/gpio/gpio[GPIO #]/value
sleep 0.1
echo 1 > /sys/class/gpio/gpio[GPIO#]/value
hciattach /dev/ttymx[UART# -1] bcm43xx 3000000 flow -t 20
hciconfig hci0 up
hcidtool scan
```

To clarify the syntax, here is the command sequence for i.MX 6Quad SDB (sequence sets UART rate to 3Mbaud - default rate for Cypress chipset):

```
echo 2 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio2/direction
echo 0 > /sys/class/gpio/gpio2/value
sleep 0.1
echo 1 > /sys/class/gpio/gpio2/value
hciattach /dev/ttymx4 bcm43xx 3000000 flow -t 20
hciconfig hci0 up
hcitool scan
```

Here is example output with Type ZP module (not including GPIO setup):

```
$ hciattach /dev/ttymx4 bcm43xx 3000000 flow -t 20
bcm43xx_init
Set Controller UART speed to 3000000 bit/s
Flash firmware /etc/firmware/BCM4335C0.ZP.hcd
Set Controller UART speed to 3000000 bit/s
Device setup complete
$ hciconfig hci0 up
$ hcitool scan
Scanning ...
    78:F7:BE:72:07:E6    SDK's GS4
```

### 5.2.1 i.MX 7Dual SDB

```
echo 119 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio119/direction
echo 0 > /sys/class/gpio/gpio119/value
sleep 0.1
echo 1 > /sys/class/gpio/gpio119/value
hciattach /dev/ttymx5 bcm43xx 3000000 flow -t 20
hciconfig hci0 up
hcitool scan
```

### 5.2.2 i.MX 6Quad(Plus)/DualLite SDB/SDP

```
echo 2 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio2/direction
echo 0 > /sys/class/gpio/gpio2/value
sleep 0.1
echo 1 > /sys/class/gpio/gpio2/value
hciattach /dev/ttymx4 bcm43xx 3000000 flow -t 20
hciconfig hci0 up
hcitool scan
```

### 5.2.3 i.MX 6SoloX SDB

```
echo 171 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio171/direction
echo 0 > /sys/class/gpio/gpio171/value
sleep 0.1
echo 1 > /sys/class/gpio/gpio171/value
hciattach /dev/ttymx2 bcm43xx 3000000 flow -t 20
hcid config hci0 up
hcid tool scan
```

### 5.2.4 i.MX 6SoloLite EVK

```
echo 145 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio145/direction
echo 0 > /sys/class/gpio/gpio145/value
sleep 0.1
echo 1 > /sys/class/gpio/gpio145/value
hciattach /dev/ttymx3 bcm43xx 3000000 flow -t 20
hcid config hci0 up
hcid tool scan
```

### 5.2.5 i.MX 6UltraLite EVK (Linux 3.14.52)

```
echo 252 > /sys/class/gpio/export
echo 0 > /sys/class/gpio/gpio252/value
sleep 0.1
echo 1 > /sys/class/gpio/gpio252/value
hciattach /dev/ttymx1 bcm43xx 3000000 flow -t 20
hcid config hci0 up
hcid tool scan
```

### 5.2.6 i.MX 6UL/ULL EVK (Linux 4.1.15)

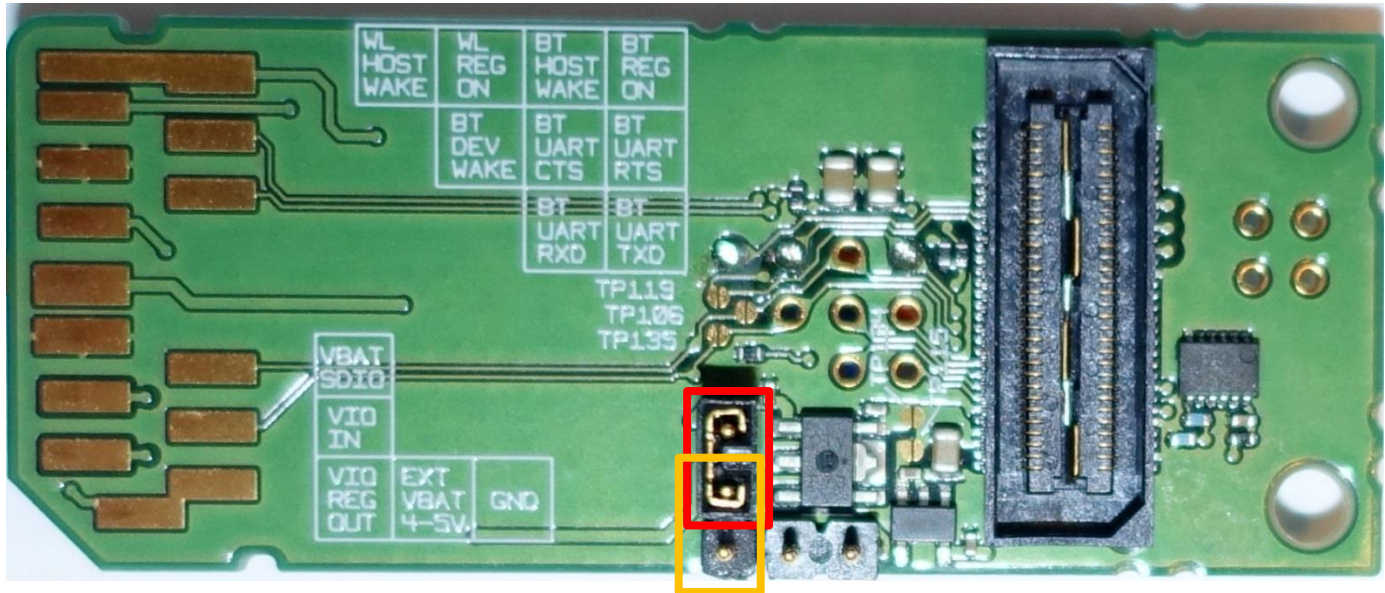
```
echo 508 > /sys/class/gpio/export
echo 0 > /sys/class/gpio/gpio508/value
sleep 0.1
echo 1 > /sys/class/gpio/gpio508/value
hciattach /dev/ttymx1 bcm43xx 3000000 flow -t 20
hcid config hci0 up
hcid tool scan
```

## 6 Verifying Adapter Boards

To ensure correct functioning of Murata Wi-Fi/BT EVK, it is of key importance to check the adapter configuration based upon jumper settings and short pads open/closed.

### 6.1 Murata i.MX InterConnect V1 Adapter

Figure 11: Murata i.MX InterConnect V1 Adapter – Top



Refer to **Figure 11** for default configuration on V1 Adapter board. VIO jumper should be in **RED** position – set for VBAT\_SDIO which is approximately 3.3V (i.e. VIO = VBAT). The adapter allows for 1.8V VIO operation (**ORANGE** position) but this Quick Start Guide currently does not support that option. A future revision will support both 1.8V and 3.3V VIO signaling on platforms/configurations that permit it. Note that there are **no closed** (soldered) short-pads on V1 Adapter from top view. Of course you will see the soldered connections for WL\_HOST\_WAKE, WL\_REG\_ON, and BT\_REG\_ON. Two figures are presented for the bottom side of the Murata i.MX InterConnect V1 Adapter. This is done to give unobstructed views of all the short pads. Refer to **Figure 12: Murata i.MX InterConnect V1 Adapter – Bottom #1** and **Figure 13: Murata i.MX InterConnect V1 Adapter – Bottom #2** for default configuration on V1 Adapter Board. Compare your adapter with the short pads (open versus closed). There should be a one-to-one mapping. The short pad selections on bottom of adapter board connect Bluetooth UART through to SD\_DAT4..7 pins (TP109, TP122, TP116, and TP130 closed/soldered). TP133 short pad is closed/soldered to connect VBAT\_SDIO (voltage supply from SD VDD Pin #4) to VBAT\_IN which powers the Murata Wi-Fi/BT EVB. The other power supply option is to use an external power supply: short TP134 (with TP133 open) and connect external supply to TP131 and TP132 (marked “EXT VBAT 4-5V” and “GND” on silkscreen – see **Figure 11**). The V1 Adapter is pre-wired to SD Card Extender. This is done to provide “plug ‘n play” interoperability with NXP i.MX 6SoloX SABRE-SD and i.MX 6SoloLite EVK. The connected signals are BT\_REG\_ON (yellow), WL\_REG\_ON (orange), and WL\_HOST\_WAKE (blue). This allows direct mapping to i.MX6 GPIO’s to these control signals. For additional specific information on default configuration, refer to the [Hardware User Manual](#).

Figure 12: Murata i.MX InterConnect V1 Adapter – Bottom #1

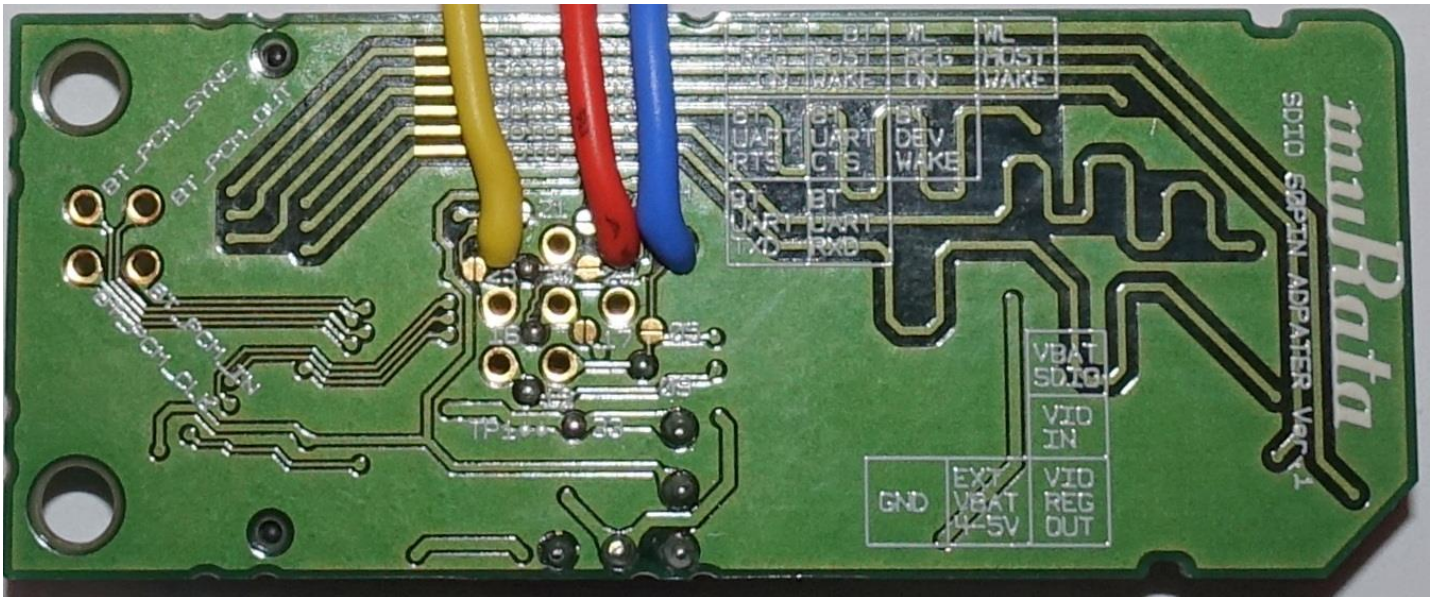
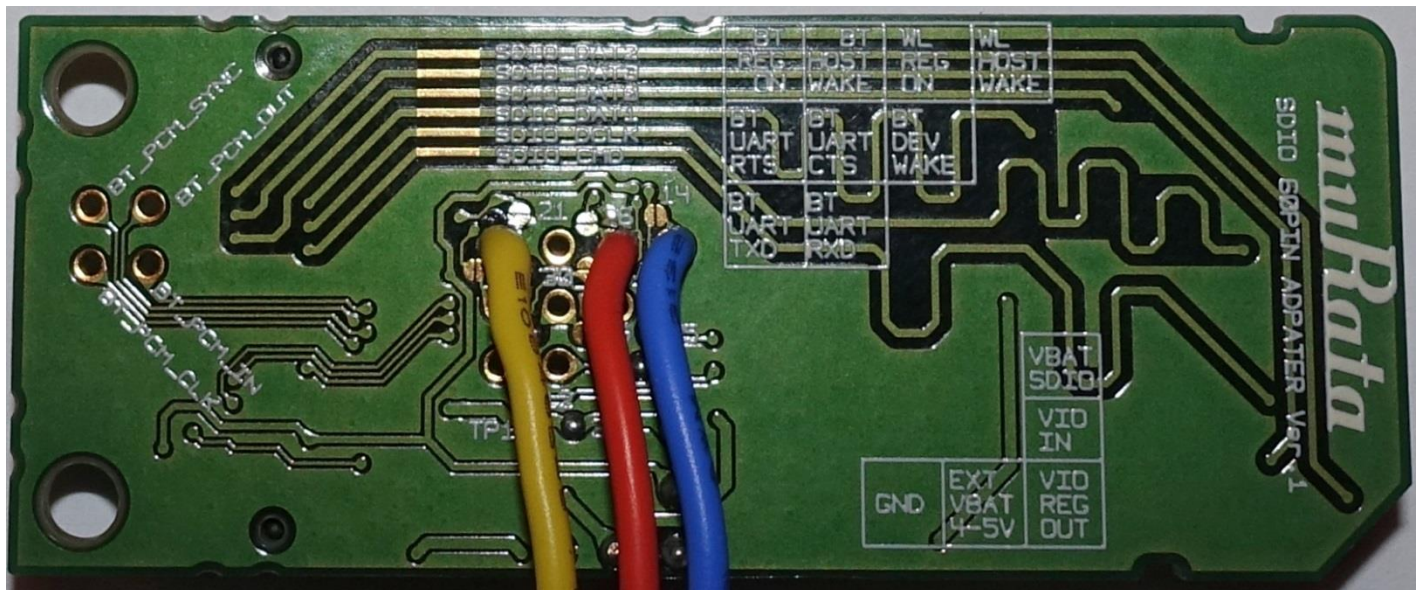


Figure 13: Murata i.MX InterConnect V1 Adapter – Bottom #2

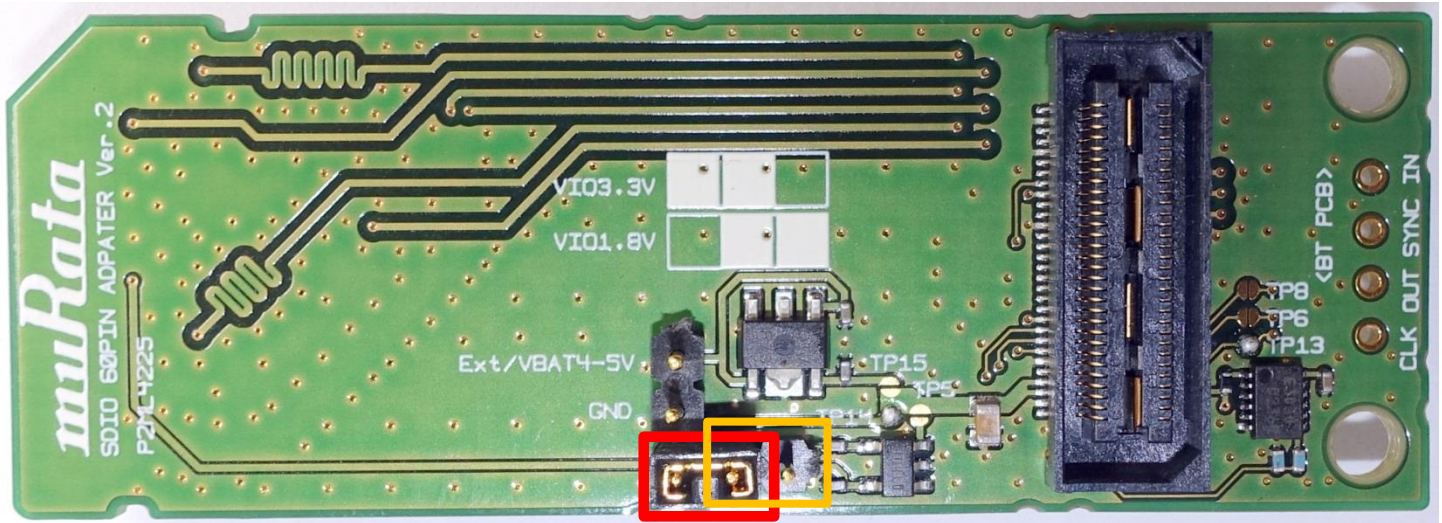


## 6.2 Murata i.MX InterConnect V2 Adapter

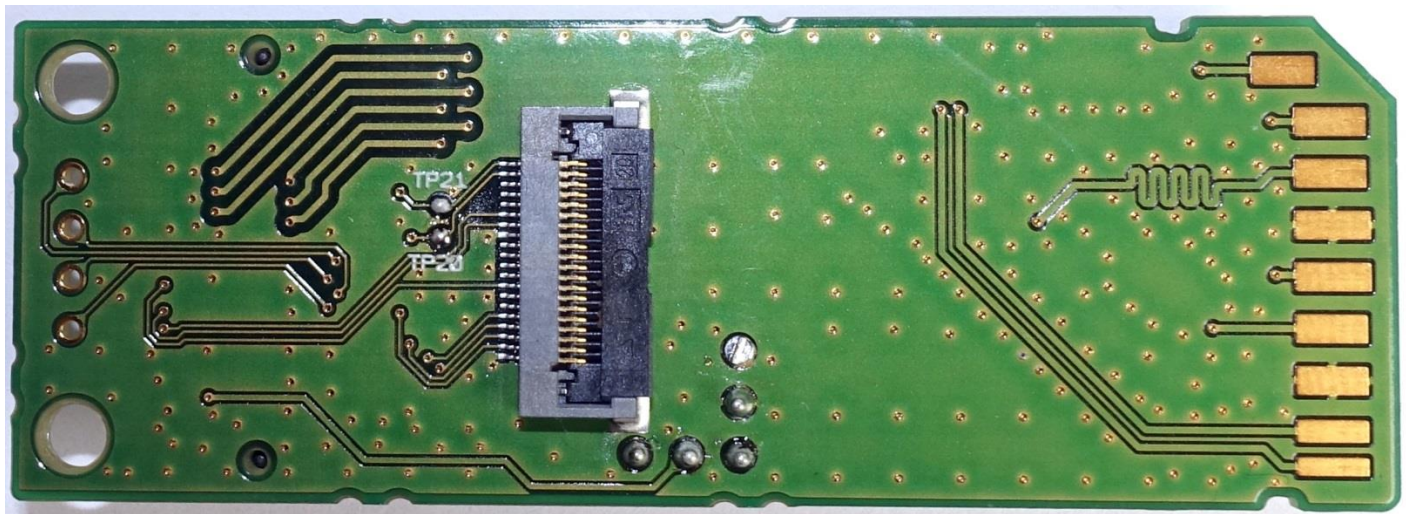
To ensure correct functioning of Murata Wi-Fi/BT EVK, it is of key importance to check the adapter configuration based upon jumper settings and short pads open/closed. V2 Adapter is much simpler than V1. However it is still important to check all connections to make sure they match defaults. Refer to **Figure 14: Murata i.MX InterConnect Adapter V2 Adapter – Top**. VIO jumper should be in **RED** position – set for VBAT\_SDIO which is approximately 3.3V (i.e. VIO = VBAT). The adapter allows for 1.8V VIO operation (**ORANGE** position) but this Quick Start Guide currently does not support that option. A future revision will support both 1.8V and 3.3V VIO signaling on platforms/configurations that permit it. Only two short pads are close on top: TP14 and TP13. TP14

short pad is closed/soldered to connect VBAT\_SDIO (voltage supply from SD VDD Pin #4) to VBAT\_IN which powers the Murata Wi-Fi/BT EVB. The other power supply option is to use an external power supply: short TP15 (with TP14 open) and connect external supply to TP11 and TP12 (marked “Ext/VBAT4-5V” and “GND” on silkscreen - see **Figure 14**). TP13 connects the BT\_REG\_ON control signal.

**Figure 14: Murata i.MX InterConnect Adapter V2 Adapter – Top**



**Figure 15: Murata i.MX InterConnect Adapter V2 Adapter - Bottom**



Refer to **Figure 15: Murata i.MX InterConnect Adapter V2 Adapter - Bottom**. Both TP20 and TP21 short pads are closed to connect WL\_REG\_ON and WL\_HOST\_WAKE respectively. WL\_HOST\_WAKE is an optional out-of-band interrupt signal that is not defined in this Quick Start document. It will be documented (and optionally enabled) in future version. For additional specific information on default configuration, refer to the [Hardware User Manual](#).

## 7 Technical Support Contact

**Table 10** below lists all the support resources available for the NXP/Cypress/Murata i.MX Wi-Fi/BT solution. There are two dedicated Murata websites (main landing page and i.MX support portal) in addition to a dedicated [imxfaq@murata.com](mailto:imxfaq@murata.com) email alias. All website/email addresses are hyperlinked in the “Support Site” column below.

**Table 10: List of Support Resources**

Support Site	Notes
<a href="#">Murata i.MX Landing Page</a>	<b>No</b> login credentials required. This is an excellent starting point to understand all hardware/software configurations supported. Quick Start Guides and Murata Module Datasheets provided.
<a href="#">My Murata i.MX Support Portal</a>	Login credentials required. More detailed Murata documentation provided: such as Linux User Manual, Hardware User Manual, RF Regulatory Test Manual, etc. For registering refer to <a href="#">this guide</a> .
<a href="#">NXP Website</a>	Register on NXP.com so you can download necessary demo/validation images, documentation, schematics, etc.
<a href="#">NXP Murata i.MX Support Portal</a>	Login credentials required. NXP’s support forum for this i.MX Wi-Fi/BT solution. Both NXP and Murata Team support this portal. <b>Note:</b> same username/password used for NXP Website (above). Once registered on the NXP Community, email your username to <a href="mailto:imxfaq@murata.com">imxfaq@murata.com</a> to join this group.
<a href="#">Cypress Murata i.MX Support Portal</a>	Login credentials required. Cypress’ dedicated support forum for NXP/Murata collaboration on i.MX Wi-Fi/BT Solution. Both Cypress and Murata Team support this portal. Once registered on the Cypress Community, please email your username to <a href="mailto:imxfaq@murata.com">imxfaq@murata.com</a> to join this group.
<a href="#">Cypress Linux Support Portal</a>	Login credentials required. Includes support forum, chipset datasheets, application notes, original bcmhd driver code releases, etc. Cypress Team supports this forum.
<a href="#">Murata i.MX FAQ Email</a>	i.MX FAQ email. Supported by Murata Team. Typically used to support issues accessing support sites, this email address is used for questions not addressed on support forums.



## 8 Additional Useful Links

In addition to **Table 10** listings of support resources, **Table 11** provides some useful links.

**Table 11: Additional Useful Links**

Link	Notes
<a href="#">“iw” Command Line</a>	“iw” is default Linux command to configure WLAN interface.
<a href="#">iPerf Performance Test Tool</a>	“iPerf” test tool is built into NXP Linux BSP image.

## X-ON Electronics

Largest Supplier of Electrical and Electronic Components

*Click to view similar products for [Multiprotocol Development Tools](#) category:*

*Click to view products by [Murata](#) manufacturer:*

Other Similar products are found below :

[CYW94343WWCD1\\_EVB](#) [MIKROE-2439](#) [XKC-M5T-W](#) [ATWINC3400-XPRO](#) [DEMOBOARD-T7024PGM](#) [2636](#) [Gpy](#) [STEVAL-FKI001V1](#) [8265.NGWMG.DTX1](#) [TEL0111](#) [SiPy 22 dBm](#) [SiPy 14 dBm](#) [ATWINC3400-XSTK](#) [RE-WFKIT-9260NVP](#) [2542](#) [irpi01-868](#) [irpi01-915](#) [BCM94343WWCD1\\_EVB](#) [INP3010](#) [INP3011](#) [ISM43340-L77-EVB](#) [ISMART43362-E](#) [ISP4520-AS-DK](#) [MIKROE-3542](#) [nRF9160-DK](#) [QPQ1906EVB-01](#) [102010129](#) [102991023](#) [107990093](#) [113990254](#) [SIMSA868C-Cloud-DKL](#) [SIMSA868-Cloud-DKL](#) [SIMSA915-Cloud-DKL](#) [SIMSA-DKL](#) [SKY66423-11EK2](#) [SKY66423-11EK1](#) [TEL0097](#) [80-000535](#) [DFR0505](#) [XKC-V1T-U](#) [FiPy](#) [453-00010-K1](#) [453-00011-K1](#) [DVK-BTM431](#) [DVK-RM186-SM-01](#) [XPC270300EK](#) [MTDOT-BOX-G-868-B](#) [MTDOT-BOX-G-915-B](#) [LBEH5DU1BW-TEMP-DS-SD](#) [113030023](#)