Murata Wi-Fi/BT Solution for i.MX

Linux User Manual

Revision History

Revision	Date	Author	Change Description
1.0	Sept 7, 2015	S Kerr	Initial Release
2.0	Nov 7, 2015	S Kerr	Changes for L3.14.38_6UL GA BSP Release and support modified Murata Wi-Fi/BT EVK Definition.
3.0	March 1, 2016	S Kerr	Incorporated changes for NXP Linux 3.14.52 GA BSP Release. Added support for hostapd and Broadcom firmware package.
4.0	Feb 14, 2017	S Kerr	Renamed document to “Murata Wi-Fi/BT Solution for i.MX Linux User Manual”. 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 bcmdhd WLAN driver, thereby matching 4.1.15_2.0.0 configuration. Added instructions for Murata source patch release which addresses errata/features on both releases. Added support for new i.MX 7Dual SDB, i.MX 6ULL EVK and Murata Type 1CK.

This page intentionally left blank.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

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 Linux User Manual provides details on building all necessary software for enabling Murata Wi-Fi/Bluetooth on i.MX6/7 platforms. The latest releases supported include both NXP Linux 3.14.52_1.1.0[1] and 4.1.15_2.0.0 GA BSPs. The manual also details optional software modifications, depending on how the customer would like to configure the platform. **Note that currently only NXP officially branded i.MX6/7 Platforms are supported[2]:**

- i.MX 7Dual SABRE Development Board
- i.MX 6QuadPlus SABRE Development Board
- i.MX 6Quad/DualLite SABRE Development Board
- i.MX 6Quad/DualLite SABRE Platform (for Smart Devices)
- i.MX 6SoloX SABRE Development Board
- i.MX 6SoloLite Evaluation Kit
- i.MX 6UltraLite Evaluation Kit
- i.MX 6ULL Evaluation Kit

A high level 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. Please refer to the Murata Quick Start Guide (Linux) and Murata Hardware User Manual for more details.

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 limits SDIO throughput. The Murata ZP module supports a very high throughput (SDIO 3.0 mode – UHS) over SDIO bus resulting in a much better performance. Please reference the NXP i.MX7 schematics for specifics on the Wi-Fi/BT interconnect: download package here[3]. **Figure 2** below shows a simplified block diagram for the i.MX 7Dual SDB. For more details on Wi-Fi throughput dependency on SDIO bus speed, please refer to the Murata Hardware User Manual.

[1] 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".
[2] A notable exclusion to this list is NXP's WaRP7 – link here. The WaRP7 software is supported by "community" software releases.
[3] For Wi-Fi/BT schematics on i.MX 7Dual SDB, refer to page 13 of sch-28590_i.mx7d_saber_rev_2.pdf document.

Figure 1: Murata i.MX6 Interconnect Kit Interfaces

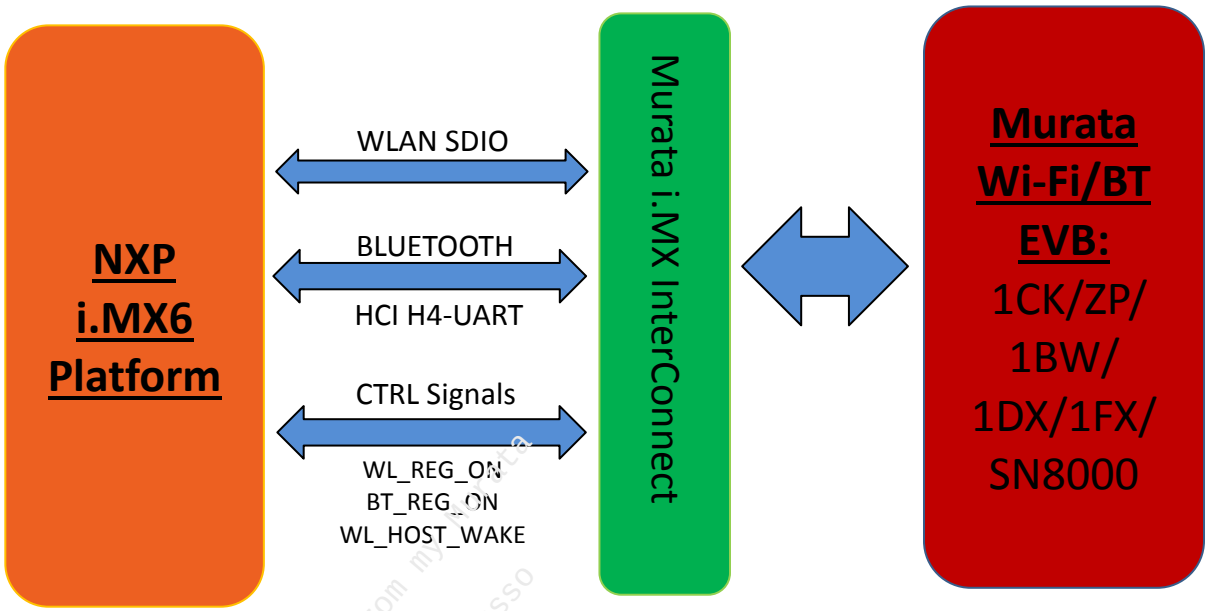
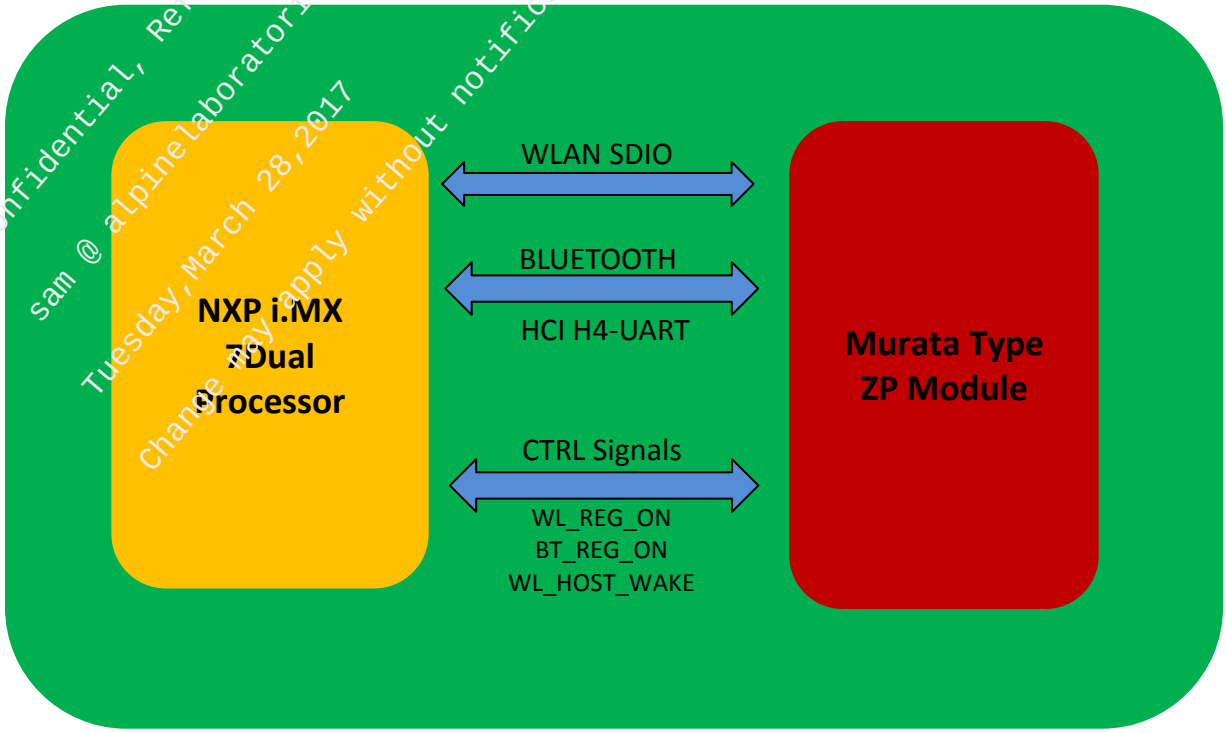


Figure 2: i.MX 7Dual SDB Block Diagram



1.1 Acronyms

Table 1: Acronyms used in Linux User Manual

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
PC	Personal Computer
SW	Software
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus

1.2 References

1.2.1 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 revised manual adds details on the i.MX 7Dual SDB platform which has an integrated Murata Type ZP module onboard.

The Murata Hardware User Manual is available on “My Murata” support portal here.

1.2.2 Murata Quick Start Guide (Linux)

Murata Wi-Fi/BT Solution for i.MX Quick Start Guide (Linux) 4.0, “Murata Wi-Fi & BT Solution for i.MX Quick Start Guide (Linux) 4.0.pdf”.

This Quick Start Guide details steps to get Murata Wi-Fi/BT EVK up and running quickly on i.MX6 platforms. It is specific to NXP i.MX Linux 3.14.52_1.1.0 & 4.1.15_2.0.0 GA BSP releases.

In addition the Quick Start manual details steps on exercising Murata Type ZP modules on the NXP i.MX 7Dual SDB.

The Murata Quick Start Guide is available on the “Murata i.MX Landing Page” here or from “My Murata” support portal here.

1.2.3 Murata i.MX Linux Source Code Patches

Patches for both Linux 3.14.52 and 4.1.15 are provided on the NXP Murata i.MX Support Portal. These source code patches address specific errata, functionality issues, and necessary enhancements. Refer to Murata i.MX Landing Page for specifics on accessing this portal.

1.2.3.1 Murata i.MX L3.14.52_1.1.0 Source Code Patch

This source code patch release includes specific fixes for the baseline NXP 3.14.52_1.1.0 release. Refer to **Section 4.3.1** for more details.

Access this link to download the 3.14.52_1.1.0 source code patch release.

1.2.3.2 Murata i.MX L4.1.15_2.0.0 Source Code Patch

This source code patch release includes specific fixes for the baseline NXP 4.1.15 release. Refer to **Section 4.3.2** for more details.

Access this link to download the 4.1.15_2.0.0 source code patch release.

1.2.4 NXP Yocto Project User's Guide

1.2.4.1 Kernel Version 3.14.52_1.1.0

Freescale Yocto Project User's Guide, Document Number: IMXLXYOCTOUG, Rev. 0, 12/2015; "Freescale_Yocto_Project_User's_Guide.pdf".

This document describes how to build an image for a NXP i.MX6 platform by using a Yocto Project build environment. It describes the NXP release layer and the NXP-specific usage. Download "Linux 3.14.52 BSP & Multimedia Codecs Documentation" package from here.

1.2.4.2 Kernel Version 4.1.15_2.0.0

i.MX Yocto Project User's Guide, Document Number: IMXLXYOCTOUG, Rev. 0, 10/2016; "i.MX_Yocto_Project_User's_Guide.pdf".

This document describes how to build an image for a NXP i.MX platform by using a Yocto Project build environment. It describes the NXP release layer and the NXP-specific usage. Download "Linux 4.1.15_2.0.0 BSP & Multimedia Codecs Documentation" package from here.

1.2.5 i.MX Linux User's Guide

1.2.5.1 Kernel Version 3.14.52_1.1.0

i.MX Linux User's Guide, Document Number: IMXLUG, Rev. L3.14.52_1.1.0p-ga, 12/2015; "i.MX_Linux_User's_Guide.pdf".

This document explains how to build and install the NXP Linux O/S BSP on the i.MX6 platform. It also covers special NXP features and how to use them. Download "Linux 3.14.52 BSP & Multimedia Codecs Documentation" package from here.

1.2.5.2 Kernel Version 4.1.15_2.0.0

i.MX Linux User's Guide, Document Number: IMXLUG, Rev. L4.1.15_2.0.0-ga, 10/2016; "i.MX_Linux_User's_Guide.pdf".

This document explains how to build and install the NXP Linux O/S BSP on the i.MX platform. It also covers special NXP features and how to use them. Download "Linux 4.1.15_2.0.0 BSP & Multimedia Codecs Documentation" package from here.

1.2.6 i.MX Linux Reference Manual

1.2.6.1 Kernel Version 3.14.52_1.1.0

i.MX Linux Reference Manual, Document Number: IMXLXRM, Rev. 0, 12/2015; "i.MX_Linux_Reference_Manual.pdf".

This document supports porting the i.MX Linux O/S BSP to customer-specific products. Intended audience should have a working knowledge of Linux O/S 3.0 kernel internals, driver models and i.MX processors. Download "Linux 3.14.52 BSP & Multimedia Codecs Documentation" package from here.

1.2.6.2 Kernel Version 4.1.15_2.0.0

i.MX Linux Reference Manual, Document Number: IMXLXRM, Rev. 0, 10/2016; "i.MX_Linux_Reference_Manual.pdf".

This document supports porting the i.MX Linux O/S BSP to customer-specific products. Intended audience should have a working knowledge of Linux O/S 4.0 kernel internals, driver models and i.MX processors. Download "Linux 4.1.15_2.0.0 BSP & Multimedia Codecs Documentation" package from here.

1.2.7 i.MX Linux Release Notes

1.2.7.1 Kernel Version 3.14.52_1.1.0

i.MX Linux Release Notes, Document Number: IMXLXRN, Rev. L3.14.52_1.1.0-ga, 12/2015; “i.MX_Linux_Release_Notes.pdf”.

This document contains important information about the package contents, supported features, known issues, and limitations in this release. Download “Linux 3.14.52 BSP & Multimedia Codecs Documentation” package from here.

1.2.7.2 Kernel Version 4.1.15_2.0.0

i.MX Linux Release Notes, Document Number: IMXLXRN, Rev. L4.1.15_2.0.0-ga, 10/2016; “i.MX_Linux_Release_Notes.pdf”.

This document contains important information about the package contents, supported features, known issues, and limitations in this release. Download “Linux 4.1.15_2.0.0 BSP & Multimedia Codecs Documentation” package from here.

2 Preparing Software for NXP i.MX with Murata Wi-Fi/BT EVK

First off it is important to underline that the NXP Linux 3.14.52_1.1.0 BSP only supports i.MX6 platforms. The subsequent Linux 4.1.15_2.0.0 release officially adds support for the i.MX 7Dual SABRE Development Board. Given the different kernel versions, the following sub-sections will detail steps for the 3.14.52_1.1.0 **or** 4.1.15_2.0.0 kernels.

2.1 Overview

The NXP Linux 3.14.52_1.1.0 and 4.1.15_2.0.0 GA BSPs integrates the necessary Cypress WLAN and Bluetooth drivers to support Murata's Wi-Fi/BT EVK's. Here are the high level steps necessary to arrive at bootable SD card for NXP's i.MX Platform:

- Download the appropriate branch of NXP Yocto Project Community BSP (3.14.52_1.1.0 or 4.1.15_2.0.0).
- Configure Yocto build for specific i.MX architecture and graphical interface.
- Optionally enable "hostapd" recipe (not included in default).
- Enable "Broadcom Firmware package" recipe in Yocto build.
- Build complete NXP Yocto project image for one of the following supported targets (not all of these targets are supported on 3.14.52_1.1.0 kernel; please refer to **Table 2**):
  - imx7dsabresd
  - imx6qpsabresd
  - imx6qsabresd
  - imx6dlsabresd
  - imx6sxsabresd
  - imx6slevk
  - imx6ulevk
  - imx6ullevk

- Flash (micro) SD card with *.sdcard (just built) image.
- Optionally enable Wi-Fi Direct (P2P) in WPA supplicant (requires WPA supplicant configuration change and re-compile).
- Setup environment for standalone Linux kernel. Checkout either 3.14.52_1.1.0 or 4.1.15_2.0.0 version. Apply Murata Linux patches from "My Murata". Compile/build Linux kernel, DTB files, and (manufacturing test[4]) bcmdhd driver files.
- Modifications to newly flashed (micro) SD card include:
  - Modified kernel, DTB files, and (for 4.1.15) MFGTEST bcmdhd driver file.
  - Type 1CK and SN8000 NVRAM files.
  - Optional modified WPA supplicant and Hostapd binaries.
  - Optional modified "hostapd.conf" and "udhcpd.conf" files for correct Wi-Fi hot spot (Soft AP) operation.
  - Optional modified network initialization files for automatic WLAN configuration.

[4] Erratum in Linux 4.1.15_2.0.0 release requires use of loadable bcmdhd.ko driver module.

2.2 Building Standard NXP Yocto Image & Flash to SD Card

The user should be very familiar with the NXP Yocto Project User's Guide. We are only emphasizing some important steps in this section.

2.2.1 Host PC Preparation

Host PC typically used has Ubuntu 14.04 or 12.04 installed with 50 GB free disk space. Please reference Section 3 of NXP Yocto Project User's Guide to setup Host PC, Host packages, and the repo utility.

> **NOTE: It is very important to keep the repo utility up-to-date. Otherwise the "repo init" or "repo sync" commands may fail.**

2.2.2 Yocto Project Setup

First make sure that git is setup properly with the commands below:

```
$ git config --global user.name "Your Name"
$ git config --global user.email "Your Email"
$ git config --list
```

The NXP Yocto Project BSP Release directory contains a "sources" directory, which contains the recipes used to build, one or more build directories, and a set of scripts used to set up the environment. The recipes used to build the project come from both the community and NXP. The Yocto Project layers are downloaded to the sources directory. This sets up the recipes that are used to build the project. The following example shows how to download the NXP Yocto Project Community BSP recipe layers. Rather than use a specific folder name, we use the variable "bsp-dir" name to represent the base directory for Yocto Build. We will use example folder names for the two kernels supported in this document:

```
$ bsp-dir=/home/skerr/NXP/fsl-yocto-3.14.52_1.1.0_CGA      ← 3.14.52_1.1.0 kernel folder
```

OR:

```
$ bsp-dir=/home/skerr/NXP/fsl-yocto-4.1.15_2.0.0_CGA       ← 4.1.15_2.0.0 kernel folder
```

Now that "bsp-dir" variable is created, we can create the folder and switch to that directory:

```
$ mkdir /home/skerr/NXP     ← If the BSP directory is nested, need to create parent folder.
$ mkdir ${bsp-dir}
$ cd ${bsp-dir}
```

➔Now we are ready to download the Freescale Yocto Project Community BSP recipe layers.

For Linux 3.14.52_1.1.0 release, execute the following repo commands:

```
$ repo init -u git://git.freescale.com/imx/fsl-arm-yocto-bsp.git -b imx-3.14.52-1.1.0_ga
$ repo sync
```

Otherwise for Linux 4.1.15_2.0.0 release:

```
$ repo init -u git://git.freescale.com/imx/fsl-arm-yocto-bsp.git -b imx-4.1-krogoth -m imx-4.1.15-2.0.0.xml
$ repo sync
```

NOTE: The linux 4.1.15_2.0.0 Yocto build has introduced new syntax for pulling specific GA releases or including latest patches in the 4.1.15_2.X.X_ga release line. For more details reference the appropriate NXP Yocto Project User's Guide or this link.

Once "repo sync" is completed, the source code is checked out into the directory "${bsp-dir}/sources". You can perform repo synchronization, with the command repo sync, periodically to update to the latest code. If errors occur during repo initialization, try deleting the .repo directory and running the repo initialization command again.

2.2.3 Image Build Configuration

The following NXP i.MX6/7 targets are supported with Murata Wi-Fi/BT EVK. The additional columns in **Table 2** reference Kernel versions supported, the relevant DTB file, and the appropriate Murata Adapter (V1/V2) to use.

The i.MX6UL/ULL DTB files highlighted in RED are generated (later when building i.MX kernel stand-alone) only if the Murata i.MX 6UL/ULL OOB_IRQ patch is applied. For more details, refer to Sections **4.3.1.3** and **4.3.2.3**.

Table 2: i.MX6/7 Targets supported on Linux 3.14.52_1.1.0/4.1.15_2.0.0 GA Releases

Target	Linux Version	NXP i.MX DTB File	Adapter
imx7dsabresd	4.1.15_2.0.0	imx7d-sdb.dtb	Integrated[5]
imx6qpsabresd	3.14.52_1.1.0/4.1.15_2.0.0	imx6qp-sabresd-btwifi.dtb	V2
imx6qsabresd	3.14.52_1.1.0/4.1.15_2.0.0	imx6q-sabresd-btwifi.dtb	V2
imx6dlsabresd	3.14.52_1.1.0/4.1.15_2.0.0	imx6dl-sabresd-btwifi.dtb	V2
imx6sxsabresd	3.14.52_1.1.0/4.1.15_2.0.0	imx6sx-sdb-btwifi.dtb	V1
imx6slevk	3.14.52_1.1.0/4.1.15_2.0.0	imx6sl-evk-btwifi.dtb[6]	V1
imx6ulevk	3.14.52_1.1.0/4.1.15_2.0.0	imx6ul-14x14-evk-btwifi.dtb/ imx6ul-9x9-evk-btwifi.dtb[7]/ imx6ul-14x14-evk-btwifi.OOB_IRQ.dtb/ imx6ul-9x9-evk-btwifi.OOB_IRQ.dtb	V2
imx6ull14x14evk	4.1.15_2.0.0	imx6ull-14x14-evk-btwifi.dtb/ imx6ull-14x14-evk-btwifi.OOB_IRQ.dtb	V2
imx6ull9x9evk	4.1.15_2.0.0	imx6ull-9x9-evk-btwifi.dtb/ imx6ull-9x9-evk-btwifi.OOB_IRQ.dtb	V2

[5] i.MX 7Dual SDB has Murata Module ZP soldered down onto platform.
[6] Default NXP i.MX DTB file for SoloLite EVK *only* supports Wi-Fi and not Bluetooth. This is only a software limitation and can be corrected with a modified DTS file to enable BT UART. Murata maintains patch files to correct this for both 3.14.52/4.1.15 kernels on "My Murata" support website.
[7] Note there are two options for i.MX 6UltraLite EVK (DTB file) depending on size of CPU.

NOTE: The following steps for image build configuration are split for 3.14.52_1.1.0 and 4.1.15_2.0.0 kernels given the important differences.

2.2.3.1 Kernel Version 3.14.52_1.1.0 Specifics

<u>To setup build directory for Linux 3.14.52_1.1.0, the following command syntax is used:</u>

```
$ cd ${bsp-dir}
$ MACHINE=<machine name> source fsl-setup-release.sh -b <build dir> -e <backend>
```

<u>One example is to build x11 graphical back end for i.MX 6SoloX SDB:</u>

```
$ cd ${bsp-dir}
$ MACHINE=imx6sxsabresd source fsl-setup-release.sh -b build-imx6sxsabresd-x11 -e x11
```

NOTE: When specifying the build directory (-b <build dir>), <u>it is better to specify a unique folder name</u> (in this example “build-imx6sxsabresd-x11”). Otherwise the next time you configure a Yocto build from the same folder (i.e. invoke “fsl-setup-release.sh” script from “bsp-dir”, it will <u>delete</u> your previous build.

<u>Final expected output is (after entering space bar to page down and accept license agreement):</u>

```
Your build environment has been configured with:

MACHINE=imx6sxsabresd
SDKMACHINE=i686
DISTRO=fsl-imx-x11
EULA=1
BSPDIR=
BUILD_DIR=.
$ /home/skerr/NXP/fsl-yocto-3.14.52_1.1.0_CGA/build-imx6sxsabresd-x11$
```

2.2.3.2 Kernel Version 4.1.15_2.0.0 Specifics

For the 4.1.15_2.0.0 kernel builds, there is a new descriptor “DISTRO” added to the command line to configure the build. As such the syntax changes from the previous 3.14.52 kernel builds.

<u>To setup build directory for Linux 4.1.15_2.0.0, the following command syntax is used:</u>

```
$ cd ${bsp-dir}
$ DISTRO=<distro name> MACHINE=<machine name> source fsl-setup-release.sh -b <build dir>
```

<u>One example is to build x11 graphical back end for i.MX 6UltraLite EVK:</u>

```
$ cd ${bsp-dir}
$ DISTRO=fsl-imx-x11 MACHINE=imx6ulevk source fsl-setup-release.sh -b build-imx6ulevk-x11
```

NOTE: When specifying the build directory (-b <build dir>), it is better to specify a unique folder name (in this example "build-imx6ulevk-x11"). Otherwise the next time you configure a Yocto build from the same folder (i.e. invoke "fsl-setup-release.sh" script from "bsp-dir", it will delete your previous build.

Final expected output is (after entering space bar to page down and accept license agreement):

```
Your build environment has been configured with:

MACHINE=imx6ulevk
SDKMACHINE=i686
DISTRO=fsl-imx-x11
EULA=
BSPDIR=
BUILD_DIR=.
$ /home/skerr/NXP/fsl-yocto-4.1.15_2.0.0_CGA/build-imx6ulevk-x11$
```

2.2.4 The Importance of the Build Directory and Current Working Environment

Having run the "fsl-setup-release.sh" script, the current terminal window has the correct build environment setup (i.e. for running "bitbake command" → **can only be done from build directory**). In addition to the "bsp-dir" variable, we will also refer to the "build-dir" folder. Following the existing examples, we can set the "build-dir" variable as follows:

```
$ build-dir=${bsp-dir}/ build-imx6sxsabresd-x11        ← 3.14.52_1.1.0 kernel folder
OR:
$ build-dir=${bsp-dir}/ build-imx6ulevk-x11            ← 4.1.15_2.0.0 kernel folder
```

NOTE: If a new terminal window is opened or the machine is rebooted after a build directory is set up, the setup environment script should be used to set up the environment variables and run a build again. The full fsl-setup-release.sh is not needed. Just execute the following command:

```
$ cd ${bsp-dir}
$ source setup-environment ${build-dir}
```

2.2.5 Important Next Steps Prior to Kicking off Yocto Build

Once the "fsl-setup-release.sh" script completes, the current directory is changed to the build directory. The next step is to make any customizations to configuration to pull in any additional recipes (such as "hostapd") before building the image.

NOTE: Packages such as "hostapd" may be optional if you do not need "Wi-Fi hot spot" or "Soft AP" mode. However packages such as "Broadcom Firmware" are essential to arrive at a fully functional image which supports both Wi-Fi and Bluetooth.

2.2.5.1 Add Hostapd Support for Wi-Fi Hot Spot Functionality (Optional)

"hostapd" is used to provide a Wi-Fi hotspot or "Soft AP" functionality. Without "hostapd", Wi-Fi can only operate in client/STA mode or P2P/Wi-Fi Direct. As such, inclusion of this package is important. By default the "hostapd" recipe is not included in this Linux 3.14.52_1.1.0 or 4.1.15_2.0.0 Yocto projects. To verify correct syntax and version numbers of packages related to "hostapd", we can use the "bitake" command:

```
$ cd ${build-dir}
$ bitbake -s | grep hostap
```

For the Linux 3.14.52_1.1.0 Yocto package, the output should be:

```
hostap-conf                    :1.0-r15
hostap-utils                   :0.4.7-r4
hostapd                        :2.2-r0
```

Alternatively, for the Linux 4.1.15_2.0.0 Yocto package:

```
hostap-conf                    :1.0-r15
hostap-utils                   :0.4.7-r4
hostapd                        :2.5-r0
```

To include "hostapd" functionality for either kernel version, then we need to edit the <build dir>/conf/local.conf file and add the following string:

```
CORE_IMAGE_EXTRA_INSTALL += "hostap-conf hostap-utils hostapd"
```

2.2.5.2 Add Cypress Wi-Fi/Bluetooth Firmware and Murata NVRAM Files (Required)

NOTE: This recipe is still referred to as "Broadcom Firmware" on NXP website. It contains Cypress Wi-Fi/Bluetooth firmware binaries and an important Wi-Fi utility (WL tool) for multiple Murata modules (see Figure 1). In addition this recipe includes Murata NVRAM files which specifically "tune" the RF characteristics. Lastly a Cypress Bluetooth Stack implementation (BSA Server and Client) are included as well in the 4.1.15_2.0.0 recipe[8].

Before kicking off final build, we need to setup a recipe which will install the necessary Broadcom Wi-Fi/Bluetooth and Murata NVRAM files in the final image. We need to download the **"Broadcom Firmware package"** which is specific to the NXP Linux 3.14.52_1.1.0/4.1.15_2.0.0 BSP Release that is posted on the NXP i.MX6_SW webpage here (look under "i.MX 6 BSP Updates and Releases->Linux->Linux <Kernel Version>".

[8] BlueZ stack is already integrated into NXP i.MX release. However some customers may opt to use Cypress' BSA stack implementation. For more details, please reference the Murata Page on Cypress Community Website: link here.

All the necessary steps to configure the Yocto build (to pull in the firmware package) are already described in “README-bcmdhd” file (you’ll see this file after extraction of downloaded file). However to make this manual more comprehensive, the specific/required steps are included here.

2.2.5.2.1 Downloading “Broadcom Firmware Package”

Refer to **Table 3** below for specifics on where to download firmware package and the names assigned to these packages.

Table 3: “Broadcom Firmware package” download

Kernel Version	Download Link	Package Name	File Name
3.14.52_1.1.0	here	“Broadcom firmware package for i.MX Linux L3.14.52 BSP”	bcmdhd-3.14.52-1.1.0-ga.gz
4.1.15_2.0.0	here	“BCMDHD-L4.1.15_2.0.0”	bcmdhd-1.0.4.tar.gz

2.2.5.2.1.1 Kernel Version 3.14.52_1.1.0 Specifics

After downloading “Broadcom Firmware package” from main i.MX6 SW page, extract it:

```
$ tar -xvzf bcmdhd-3.14..52-1.1.0-ga.gz
```

Expected Output is:

```
bcmdhd-3.14.52-1.1.0-ga/
bcmdhd-3.14.52-1.1.0-ga/README-bcmdhd
bcmdhd-3.14.52-1.1.0-ga/SCR-bcmdhd.txt
bcmdhd-3.14.52-1.1.0-ga/packages/
bcmdhd-3.14.52-1.1.0-ga/packages/firmware-bcmdhd-1.0.2.bin
bcmdhd-3.14.52-1.1.0-ga/meta-fsl-bcmdhd/
bcmdhd-3.14.52-1.1.0-ga/meta-fsl-bcmdhd/conf/
bcmdhd-3.14.52-1.1.0-ga/meta-fsl-bcmdhd/conf/layer.conf
bcmdhd-3.14.52-1.1.0-ga/meta-fsl-bcmdhd/recipes-bsp/
bcmdhd-3.14.52-1.1.0-ga/meta-fsl-bcmdhd/recipes-bsp/firmware-bcmdhd/
bcmdhd-3.14.52-1.1.0-ga/meta-fsl-bcmdhd/recipes-bsp/firmware-bcmdhd/firmware-bcmdhd_1.0.2.bb
bcmdhd-3.14.52-1.1.0-ga/COPYING
```

Now copy the “meta-fsl-bcmdhd” folder to the current “sources” directory:

```
$ mkdir ${bsp-dir}/sources/meta-fsl-bcmdhd
$ cp -Rfp bcmdhd-3.14.52-1.1.0-ga/meta-fsl-bcmdhd/* ${bsp-dir}/sources/meta-fsl-bcmdhd/
```

Now add the following line to “${build-dir}/conf/bblayers.conf”:

```
BBLAYERS += " ${BSPDIR}/sources/meta-fsl-bcmdhd "
```

Now we need to place the “Broadcom firmware package” binary in the “downloads” folder. This folder does not currently exist given that we have not kicked off the build with “bitbake” command. So first we need to create this folder:

```
$ mkdir ${bsp-dir}/downloads
```

Next copy over the “firmware-bcmdhd-1.0.2.bin” package:

```
$ cp bcmdhd-3.14.52-1.1.0-ga/packages/firmware-bcmdhd-1.0.2.bin ${bsp-dir}/downloads/
```

Now create “.done” file in the “downloads” directory for firmware package:

```
$ touch ${bsp-dir}/downloads/firmware-bcmdhd-1.0.2.bin.done
```

Edit “${build-dir}/conf/local.conf” to add the following line:

```
INTERNAL_MIRROR = "http://localhost"
```

Last step is to invoke specific bitbake commands to include “firmware-bcmdhd” recipe in “packagegroup-base”:

```
$ cd ${build-dir}
$ bitbake linux-imx packagegroup-base -c cleansstate
$ bitbake linux-imx firmware-bcmdhd packagegroup-base
```

After both “bitbake” commands, final expected output should include “all succeeded” string:

```
NOTE: Tasks Summary: Attempted 6 tasks of which 0 didn't need to be rerun and all succeeded.
```

2.2.5.2.1.2 Kernel Version 4.1.15_2.0.0 Specifics

After downloading “Broadcom Firmware package” from main i.MX6 SW page, copy it to a specific temporary “firmware” folder and extract it:

```
$ mkdir bcmdhd-1.0.4-firmware
$ cp bcmdhd-1.0.4.tar.gz bcmdhd-1.0.4-firmware/
$ cd bcmdhd-1.0.4-firmware
$ tar -xvzf bcmdhd-1.0.4.tar.gz
```

Expected Output is:

```
./
./COPYING
./meta-fsl-bcmdhd/
./meta-fsl-bcmdhd/conf/
./meta-fsl-bcmdhd/conf/layer.conf
./meta-fsl-bcmdhd/recipes-bsp/
./meta-fsl-bcmdhd/recipes-bsp/firmware-bcmdhd/
```

```
./meta-fsl-bcmdhd/recipes-bsp/firmware-bcmdhd/firmware-bcmdhd_1.0.4.bb
./meta-fsl-bcmdhd/recipes-bsp/bsa-serverandclientapps/
./meta-fsl-bcmdhd/recipes-bsp/bsa-serverandclientapps/bsa-serverandclientapps_1.0.0.bb
./meta-fsl-bcmdhd/recipes-fsl/
./meta-fsl-bcmdhd/recipes-fsl/packagegroup/
./meta-fsl-bcmdhd/recipes-fsl/packagegroup/packagegroup-fsl-tools-testapps.bbappend
./SCR-bcmdhd.txt
./README-bcmdhd
./packages/
./packages/BSA-ServerAndClientApps-0107.00.16.00.bin
./packages/firmware-bcmdhd-1.0.4.bin
```

<u>Now copy the “meta-fsl-bcmdhd” folder to the current “sources” directory:</u>

```
$ mkdir ${bsp-dir}/sources/meta-fsl-bcmdhd
$ cp -Rfp meta-fsl-bcmdhd/* ${bsp-dir}/sources/meta-fsl-bcmdhd/
```

<u>Now add the following line to “${build-dir}/conf/bblayers.conf”:</u>

```
BBLAYERS += " ${BSPDIR}/sources/meta-fsl-bcmdhd "
```

Now we need to place the Broadcom firmware and BSA Stack binaries in the “downloads” folder. This folder does not currently exist given that we have not kicked off the build with “bitbake” command. So <u>first we need to create the folder:</u>

```
$ mkdir ${bsp-dir}/downloads
```

<u>Next copy over the firmware and BSA Stack binaries.</u>

```
$ cp -Rfp packages/* ${bsp-dir}/downloads/
```

<u>Now create “.done” files in the “downloads” directory for the two files (necessary for Yocto build):</u>

```
$ touch ${bsp-dir}/downloads/firmware-bcmdhd-1.0.4.bin.done
$ touch ${bsp-dir}/downloads/BSA-ServerAndClientApps-0107.00.16.00.bin.done
```

<u>Edit “${build-dir}/conf/local.conf” to add the following line:</u>

```
    INTERNAL_MIRROR = "http://localhost"
```

<u>Last step is to invoke specific bitbake commands to include “firmware-bcmdhd” recipe in “packagegroup-base”:</u>

```
$ cd ${build-dir}
$ bitbake linux-imx packagegroup-base packagegroup-fsl-tools-testapps -c cleansstate
$ bitbake linux-imx firmware-bcmdhd packagegroup-base packagegroup-fsl-tools-testapps
```

After both “bitbake” commands, final expected output should include “all succeeded” string:

NOTE: Tasks Summary: Attempted 2969 tasks of which 1165 didn't need to be rerun and all succeeded.

2.2.6 Final Image Build

Now that we have made all of the configurations and (optional) customizations related to file system build, we can invoke “bitbake” command to build the image.

> **NOTE: Given that we need to modify default kernel, drivers, and DTB files for specific Murata modifications then it is necessary to install the stand-alone build environment. This only adds one (optional) step to the overall build process. We need to invoke “bitbake” with “populate_sdk” option:**

```
$ cd ${build-dir}
$ bitbake fsl-image-gui -c populate_sdk
```

Please reference **Section 3** in this manual for details on how to setup build environment (i.e. cross compiler) for building kernel, drivers, DTB files, etc.

Now to build the actual SD card image, “bitbake” needs to be invoked again (using fsl-image-gui graphics configuration as example):

```
$ cd ${build-dir}
$ bitbake fsl-image-gui
```

After both “bitbake” commands, final expected output should include “all succeeded” string:

NOTE: Tasks Summary: Attempted 7236 tasks of which 5421 didn't need to be rerun and all succeeded.

2.2.7 Flashing SD Card

Given we've already built the SD card image, we can now flash the (micro) SD card used for booting the i.MX platform. Insert the (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 3**.

Figure 3: 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.**

Following the examples for Linux 3.14.52_1.1.0 (SoloX) and 4.1.15_2.0.0 (UltraLite), the following commands for flashing the (micro) SD are…

For Linux 3.14.52_1.1.0:

$ sudo dd if=${build-dir}/tmp/deploy/images/imx6sxsabresd/fsl-image-gui-imx6sxsabresd.sdcard of=/dev/sdx bs=1M && sync

For Linux 4.1.15_2.0.0:

$ sudo dd if=${build-dir}/tmp/deploy/images/imx6ulevk/fsl-image-gui-imx6ulevk.sdcard of=/dev/sdx bs=1M && sync

⇨ **SD Card is now flashed with NXP i.MX image which supports Murata Wi-Fi/BT EVK.**

Note: Although we have finished “baseline” Wi-Fi/BT support, further changes to (micro) SD card are needed to incorporate specific Murata enhanced features, patches, support Wi-Fi Direct (P2P) and Wi-Fi hot spot (Soft AP).

2.2.8 Adding Wi-Fi Direct Capability (Optional)

Wi-Fi Direct (P2P) is not enabled by default in either Yocto build (for kernels 3.14.52_1.1.0 and 4.1.15_2.0.0). To enable it, we need to modify the WPA supplicant configuration and recompile the WPA supplicant binaries. The only file which needs modification is the WPA supplicant “defconfig” file.

NOTE: There are two different configurations for the WPA 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). Depending on the target processor selected, the path to compiled image files changes accordingly. See “cortex” string below: need to select either “a7” or “a9”.

For Linux 3.14.52_1.1.0, edit this file:

${build-dir}/tmp/work/<cortex…>/wpa-supplicant/2.3-r0/defconfig

Alternatively for Linux 4.1.15_2.0.0, edit this file:

${build-dir}/tmp/work/<cortex…>/wpa-supplicant/2.5-r0/defconfig

The relevant section is in “defconfig” is:

```
# P2P (Wi-Fi Direct)
# This can be used to enable P2P support in wpa_supplicant. See README-P2P for
# more information on P2P operations.
# CONFIG_P2P=y    ← Uncomment this line.
```

Uncomment the “CONFIG_P2P=y” line and reconfigure/rebuild the WPA supplicant with following steps:

```
$ cd ${build-dir}
$ bitbake -c configure wpa-supplicant -f
$ bitbake -c compile wpa-supplicant -f        ← Compile new WPA supplicant binaries.
$ bitbake -c install wpa-supplicant -f        ← Copy binary and configuration files to “image” subfolder.
```

NOTE: Only after we have flashed the SD card with the Yocto image, can we then copy over these new WPA supplicant binaries.

The relevant WPA supplicant binaries include the following:

- /usr/sbin/wpa_supplicant
- /usr/sbin/wpa_cli
- /usr/bin/wpa_passphrase

For Linux 3.14.52_1.1.0, these binaries and associated configuration files are located at:

```
${build-dir}/tmp/work/<cortex…>/wpa-supplicant/2.3-r0/image/
```

Alternatively for Linux 4.1.15_2.0.0, these binaries are located at:

```
${build-dir}/tmp/work/<cortex…>/wpa-supplicant/2.5-r0/image/
```

Note: After executing the “dd” command to flash the (micro) SD card, we need to remove it and re-insert it before modifying the file system.

For Kernel 3.14.52_1.1.0, we need to execute the following commands to replace the WPA supplicant binaries and configuration files:

```
$ sudo cp -Rfp ${build-dir}/tmp/work/<cortex…>/wpa-supplicant/2.3-r0/image/* /media/<root file system string>/
```

Alternatively for Kernel 4.1.15_2.0.0, we need to execute the following commands to replace the WPA supplicant binaries and configuration files:

```
$ sudo cp -Rfp ${build-dir}/tmp/work/<cortex…>/wpa-supplicant/2.5-r0/image/* /media/<root file system string>/
```

⇨ **Linux image is now ready to support Wi-Fi Direct (P2P).**

2.2.9 Configuring for Wi-Fi Hot Spot (Optional)

Some minor modifications are required to configure the Linux Yocto image to support Wi-Fi hot spot (Soft AP) mode. These changes will only work if we have already added support for Hostapd (detailed in Section 2.2.5.1).

Note: After executing the "dd" command to flash the (micro) SD card, we need to remove it and re-insert it before modifying the file system. You can ignore this comment if you've already followed Section 2.2.8 with changes to support Wi-Fi Direct.

Edit "/media/<root file system string>/etc/hostapd.conf" to define the correct (nl80211) driver interface by adding the following line:

```
# Driver interface type (hostap/wired/none/nl80211/bsd);
# default: hostap). nl80211 is used with all Linux mac80211 drivers.
# Use driver=none if building hostapd as a standalone RADIUS server that does
# not control any wireless/wired driver.
# driver=hostap
driver=nl80211    ← Add this line
```

Once the Wi-Fi hot spot (Soft AP) is functional on the i.MX platform (using Murata Wi-Fi/BT solution), this modification will result in the following defaults (broadcast SSID of "test", 802.11g mode, and channel 1):

```
interface=wlan0
driver=nl80211
ssid=test
channel=1
hw_mode=g
```

The final change to Linux Yocto file system is to add "/etc/udhcpd.conf" file so that the i.MX platform will support a DHCP server. To better understand this, here is the standard sequence for WLAN association:

- Client/STA authenticates with Wi-Fi hot spot (Soft AP).
- Client/STA requests IP address (DHCP protocol).
- Wi-Fi hot spot (Soft AP) grants IP address ➔ this requires "udhcpd" configuration.

Create the "/media/<root file system string>/etc/udhcpd.conf" file with following contents:

```
start           192.168.1.100
end             192.168.1.150
interface       wlan0
option dns      192.168.1.1
option router   192.168.1.1
```

⇨ **Linux image is now ready to support Wi-Fi hot spot (Soft AP).**

3 Using Stand-Alone Build Environment

The Yocto project is a good tool for pulling in specific packages/recipes. However it is not convenient when it comes to modifying/customizing/testing kernel/drivers/DTB files. This is because the actual code compiled by the Yocto framework is treated as “temporary”. We have to use a stand-alone build environment to get into “code development” mode where source code can be modified “on the fly” and the kernel/driver/DTB file re-compiled multiple times.

Note: the stand-alone build environment is required for Murata enhancements, errata fixes, and customizations to both the Linux 3.14.52_1.1.0 and 4.1.15_2.0.0 releases.

3.1 Installing and Initializing Stand-Alone Build Environment

To build modified Kernel and DTB files, we need to setup stand-alone build environment. Following up from "bitbake -c populate_sdk" option in Section **2.2.6**.

3.1.1 Kernel Version 3.14.52_1.1.0 Specifics

Execute the following script to install the SDK:

```
$ sudo sh ${build-dir}/tmp/deploy/sdk/fsl-imx-x11-glibc-x86_64-fsl-image-gui-cortexa9hf-vfp-neon-toolchain-3.14.52-1.1.0.sh
```

This step might take a couple of minutes. Expected output is:

```
${build-dir}/tmp/deploy/sdk/fsl-imx-x11-glibc-x86_64-fsl-image-gui-cortexa9hf-vfp-neon-toolchain-3.14.52-1.1.0.sh: 72: read: Illegal option -e
You are about to install the SDK to "/opt/fsl-imx-x11/3.14.52-1.1.0". Proceed[Y/n]?Y
Extracting SDK...done
Setting it up...done
SDK has been successfully set up and is ready to be used.
```

Note: “fsl-image-gui” string in path above will change based on Yocto image name.

To setup toolchain environment (put this in your .bashrc file if you plan on doing a number of code modifications):

```
$ source /opt/fsl-imx-x11/3.14.52-1.1.0/environment-setup-cortexa9hf-vfp-neon-poky-linux-gnueabi
```

To confirm that your current working terminal is setup to compile, just test one of the environment variables such as CROSS_COMPILE:

```
$ printenv CROSS_COMPILE
arm-poky-linux-gnueabi-
```

3.1.2 Kernel Version 4.1.15_2.0.0 Specifics

Execute the following script to install the SDK:

```
$ sudo sh ${build-dir}/tmp/deploy/sdk/fsl-imx-x11-glibc-x86_64-fsl-image-gui-cortexa7hf-neon-toolchain-4.1.15-2.0.0.sh
```

This step might take a couple of minutes. Expected output is:

```
Freescale i.MX Release Distro SDK installer version 4.1.15-2.0.0
=================================================================
Enter target directory for SDK (default: /opt/fsl-imx-x11/4.1.15-2.0.0):
You are about to install the SDK to "/opt/fsl-imx-x11/4.1.15-2.0.0". Proceed[Y/n]? Y
Extracting SDK.....................................................................................................done
Setting it up...done
SDK has been successfully set up and is ready to be used.
Each time you wish to use the SDK in a new shell session, you need to source the environment setup script e.g.
 $ . /opt/fsl-imx-x11/4.1.15-2.0.0/environment-setup-cortexa7hf-neon-poky-linux-gnueabi
```

Note: "fsl-image-gui" string in path above will change based on Yocto image name.

To confirm that your current working terminal is setup to compile, just test one of the environment variables such as CROSS_COMPILE:

```
$ printenv CROSS_COMPILE
arm-poky-linux-gnueabi-
```

3.2 Configuring i.MX Linux Baseline Source Code

Get Linux Source Code:

```
$ cd <to desired folder for building kernel/drivers/dtb files>  (i.e. make this a permanent folder)
$ git clone git://git.freescale.com/imx/linux-imx.git linux-imx
```

Note: The new "linux-imx.git" repository supports both 3.14.52_1.1.0 and 4.1.15_2.0.0 releases.

Expected output is:

```
Cloning into 'linux-imx'...
remote: Counting objects: 4577786, done.
remote: Compressing objects: 100% (730367/730367), done.
Receiving objects: 100% (4577786/4577786), 986.87 MiB | 1.79 MiB/s, done.
remote: Total 4577786 (delta 3865112), reused 4526390 (delta 3813817)
Resolving deltas: 100% (3865112/3865112), done.
Checking connectivity... done.
```

Checking out files: 100% (38165/38165), done.

Change into kernel root folder (linux-imx) and select correct branch:

```
$ cd linux-imx
$ git branch -a
```

Expected output is shown below. Verify that your selected kernel is among them:

```
  remotes/origin/imx_3.10.17_1.0.0_ga
  remotes/origin/imx_3.10.53_1.1.0_ga
  remotes/origin/imx_3.14.28_1.0.0_ga
  remotes/origin/imx_3.14.52_1.1.0_ga   ← 3.14.52_1.1.0 Kernel
  remotes/origin/imx_4.1.15_1.0.0_ga
  remotes/origin/imx_4.1.15_2.0.0_ga    ← 4.1.15_2.0.0 Kernel
```

Switch to the desired Branch by invoking “git checkout” command (after setting “BRANCH” variable):

```
$ BRANCH=imx_3.14.52_1.1.0_ga
```

OR:

```
$ BRANCH=imx_4.1.15_2.0.0_ga
```

Now checkout the code:

```
$ git checkout -b ${BRANCH} origin/${BRANCH}
```

Expected output for 3.14.52_1.1.0 kernel is:

```
origin/${BRANCH}
Checking out files: 100% (42117/42117), done.
Branch imx_3.14.52_1.1.0_ga set up to track remote branch imx_3.14.52_1.1.0_ga from origin.
Switched to a new branch 'imx_3.14.52_1.1.0_ga'
```

Alternatively, expected output for 4.1.15_2.0.0 kernel is:

```
origin/${BRANCH}
Checking out files: 100% (50042/50042), done.
Branch imx_4.1.15_2.0.0_ga set up to track remote branch imx_4.1.15_2.0.0_ga from origin.
Switched to a new branch 'imx_4.1.15_2.0.0_ga'
```

⇨ **NOW we can build the modified kernel, modified drivers, and new DTB files: necessary to incorporate Murata customizations to NXP release(s).**

4 Murata Customizations to NXP Linux Releases (3.14.52 and 4.1.15)

Now that the stand-alone build environment is established, we can apply the necessary Murata patches to the baseline NXP Linux release. The patches applied to both 3.14.52 and 4.1.15 releases are outlined below. In terms of functionality, the most significant change is building the bcmdhd WLAN driver into the Linux 3.14.52 kernel and providing OOB IRQ support as well.

4.1 Murata Linux 3.14.52_1.1.0 Source Code Patch

- 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.

4.2 Murata Linux 4.1.15_2.0.0 Source Code Patch

- 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.
- Provide specific kernel (with bcmdhd.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.

4.3 Instructions to Apply Patch and Build Modified Kernel, DTB file(s), etc.

First download the Murata Linux (3.14.52_1.1.0 or 4.1.15_2.0.0) Source Code Patch file(s): link provided here. Having followed steps in **Section 3** to setup the stand-alone build environment and checkout the desired kernel, we now need to apply the Murata source code patches and kick off the build. Each patch release has its own README. It is important to note that the source code patches are not included in just one file. Rather, there are a few patch files included in each Murata release which can be optionally applied – according to what the end user needs.

4.3.1 Murata Source Patches for Linux 3.14.52_1.1.0 Release

The Murata patch files for Linux 3.14.52_1.1.0 are described in the following sub-sections.

4.3.1.1 0001-Murata-patch-for-imx-linux-3.14.52_1.1.0_ga.patch

- 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.
- 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 patch for Murata Type 1DX initialization failure.
- Add patch for suspend/resume power save erratum.
- Patch bcmdhd driver to fix Wi-Fi Direct erratum.

4.3.1.2 0002-Enable-OOB-INTR-in-bcmdhd.patch

Add OOB IRQ support for all i.MX platforms except i.MX6UL EVK. Unlike the bcmdhd driver included in 4.1.15_2.0.0 kernel, the 3.14.52_1.1.0 WLAN driver requires changes to the bcmdhd Makefile to switch between OOB IRQ and SDIO in-band interrupts.

Change default NVRAM and firmware path/filename: point to OOB IRQ version in “lib/firmware/bcm/ZP_BCM4339” folder. To support other modules, we just need to change the path/filenames. Refer to **Table 4: Murata Module to Firmware/NVRAM Mapping** for other modules.

Note: When switching WLAN configuration between SDIO in-band and OOB IRQ, the correct NVRAM file <u>must</u> be used. Otherwise the bcmdhd driver initialization will fail.

4.3.1.3 0003-Add-OOB-IRQ-supprt-on-imx6ul.patch

Add OOB IRQ support for i.MX 6UL EVK. A hardware change **is required** to run optional OOB IRQ configuration – refer to the Hardware User Manual for specifics. This patch creates two new DTB files which must be used when configuring for OOB IRQ operation:

- imx6ul-14x14-evk-btwifi.OOB_IRQ.dtb
- imx6ul-9x9-evk-btwifi.OOB_IRQ.dtb

Note: To enable OOB IRQ configuration on i.MX 6UL EVK, it is necessary to disable the second of two Ethernet ports ("fec2" in DTS file). More details in the Hardware User Manual.

4.3.1.4 0004-Enable-OOB_IRQ-edge-support.patch

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.

Table 4: Murata Module to Firmware/NVRAM Mapping

Module	Folder	Firmware: STA/Client/P2P Soft AP Manufacturing Test	NVRAM: OOB IRQ SDIO In-Band	Additional Notes
1CK	/lib/firmware/bcm/ ZP_BCM4339	fw_bcmdhd.bin fw_bcmdhd_apsta.bin[9] 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[9] 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 configuration.

[9] 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.

4.3.2 Murata Source Patches for Linux 4.1.15_2.0.0 Release

The Murata patch files for Linux 4.1.15_2.0.0 are described in the following sub-sections.

4.3.2.1 0001-Murata-patch-for-imx-linux-4.1.15_2.0.0_ga.patch

- Enable Bluetooth interface on i.MX 6SoloLite EVK.
- Correctly MUX BT_REG_ON line on i.MX 6SoloLite EVK.
- Correctly MUX WL_REG_ON and BT_REG_ON lines on i.MX 7Dual SDB.
- Add patch for Murata Type 1DX initialization failure.
- Add patch for suspend/resume power save erratum.

4.3.2.2 0002-SN8000-fix-on-imx6ul.patch

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.

4.3.2.3 0003-Add-OOB-IRQ-support-on-imx6ul.patch

Add optional OOB IRQ support for i.MX6UL/ULL EVK's. A hardware change **<u>is required</u>** to run optional OOB IRQ configuration – refer to the [Hardware User Manual] for specifics. This patch creates four new DTB files which must be used <u>when configuring for OOB IRQ operation:</u>

- imx6ul-14x14-evk-btwifi.OOB_IRQ.dtb
- imx6ul-9x9-evk-btwifi.OOB_IRQ.dtb
- imx6ull-14x14-evk-btwifi.OOB_IRQ.dtb
- imx6ull-9x9-evk-btwifi.OOB_IRQ.dtb

Note: To enable OOB IRQ configuration on i.MX 6UL/6ULL EVK's, it is necessary to disable the second of two Ethernet ports ("fec2" in DTS file). More details in the Hardware User Manual.

4.3.2.4 0004-Enable-OOB_IRQ-edge-support.patch

Support OOB IRQ configuration for Murata Type 1BW and SN8000 modules on all i.MX platforms: this requires a modified kernel as built-in bcmdhd driver is modified to support edge-sensitive interrupts.

4.3.2.5 0001-build-bcmdhd-driver-for-mfgtest.patch

Provide specific kernel (with separate bcmdhd.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.

4.3.3 Applying Murata Source Patches to NXP Linux Baseline Release

The Murata patch releases for both Linux 3.14.52_1.1.0 and 4.1.15_2.0.0 have to be applied in a specific order. That order is clearly notated by the leading number of the patch file.

For either kernel release the patch application sequence is the same:

- Follow **Section 3** to configure stand-alone build environment and extract the NXP Linux kernel source code.
- Download the Murata i.MX Linux Source Code Patch.
- Extract the patch files and copy to kernel root folder.
- Use “patch -p 1 < <patch file>” to apply each patch file to the Linux Kernel.
- If “/arch/arm/configs/imx_v7_defconfig” does not have correct NVRAM/firmware path/filenames setup, then edit this file and change accordingly – refer to **Table 4** (search for CONFIG_BCMDHD_NVRAM_PATH and CONFIG_BCMDHD_FW_PATH).
- Follow steps in **Section 4.3.4** to build kernel, DTB files, and optional MFGTEST bcmdhd driver (Kernel 4.1.15_2.0.0 only).

4.3.3.1 Linux 3.14.52_1.1.0 Patch Application

Referring to **Table 5**, we can see which patches need to be applied for a specific desired configuration.

Note: Due to included errata fixes and feature enhancements, Murata strongly recommends applying the “0001” patch.

Table 5: Linux 3.14.52_1.1.0 Patch to Configuration Mapping

Patch Numbers Applied	OOB IRQ	SDIO In-Band	Built-In BCMDHD	i.MX6UL OOB IRQ	Edge OOB IRQ
0001		x	x		
0001,0002	x		x		
0001,0002,0003	x		x	x	
0001,0002,0003,0004	x		x	x	x

⇨ One important consideration when compiling DTB files for i.MX6UL EVK. If the SDIO in-band interrupt configuration is desired, then make sure to **NOT** apply “0003-Add-OOB-IRQ-supprt-on-imx6ul” patch as this will disable the second of two Ethernet ports.

4.3.3.2 Linux 4.1.15_2.0.0 Patch Application

Referring to **Table 6**, we can see which patches need to be applied for a specific desired configuration. The last patch which provides support for MFGTEST **should only be used** for RF testing. This patch provides a workaround to a kernel crash: seen when loading the manufacturing test firmware with normal bcmdhd driver configuration. The kernel and bcmdhd driver are re-compiled

so that the bcmdhd driver is a separate loadable module. The bcmdhd driver is loaded to run with SDIO in-band interrupts.

Note: Due to included errata fixes and feature enhancements, Murata strongly recommends applying the "0001" and "0002" patches.

Table 6: Linux 4.1.15_2.0.0 Patch to Configuration Mapping

Patch Numbers Applied	SN8000 on i.MX6UL/ULL	i.MX6UL/ULL OOB IRQ	Edge OOB IRQ	MFGTEST Support
0001				
0001,0002	x			
0001,0002,0003	x	x		
0001,0002,0003,0004	x	x	x	
0001,0002,0003,0004, MFGTEST 0001	x			x

- ⇨ One important consideration when compiling DTB files for i.MX6UL/ULL EVK. If the SDIO in-band interrupt configuration is desired, then make sure to **NOT** apply "0003-Add-OOB-IRQ-support-on-imx6ul" patch as this will disable the second of two Ethernet ports **AND** configure the DTB file for OOB IRQ configuration.

4.3.4 Building Patched Linux Kernel, DTB files, and (optional) Drivers

With the stand-alone build environment setup and Murata patches applied, we can build the kernel, DTB files, and optional MFGTEST bcmdhd driver for Linux 4.1.15_2.0.0.

Setting up i.MX configuration (patched with Murata modifications). Expected output is:

```
$ make imx_v7_defconfig
...
  HOSTCC  scripts/basic/fixdep
  HOSTCC  scripts/kconfig/conf.o
  SHIPPED scripts/kconfig/zconf.tab.c
  SHIPPED scripts/kconfig/zconf.lex.c
  SHIPPED scripts/kconfig/zconf.hash.c
  HOSTCC  scripts/kconfig/zconf.tab.o
  HOSTLD  scripts/kconfig/conf
#
# configuration written to .config
#
```

Now to build kernel, dtb files, and driver modules. Expected output is:

```
$ make zImage dtbs modules          ← insert "-j 8" or whatever matches your PC architecture
....
  LD [M]  lib/crc7.ko
  LD [M]  lib/libcrc32c.ko
  LD [M]  sound/core/snd-hwdep.ko
  LD [M]  sound/core/snd-rawmidi.ko
  LD [M]  sound/usb/snd-usbmidi-lib.ko
  LD [M]  sound/usb/snd-usb-audio.ko
  AS      arch/arm/boot/compressed/piggy.lzo.o
  LD      arch/arm/boot/compressed/vmlinux
  OBJCOPY arch/arm/boot/zImage
  Kernel: arch/arm/boot/zImage is ready
```

With the MFGTEST patch applied for Linux 4.1.15_2.0.0 kernel, we will generate the bcmdhd.ko driver module. Verify by invoking "find" command with expected output:

```
$ find . -name bcmdhd.ko
./drivers/net/wireless/bcmdhd/bcmdhd.ko
```

4.3.5 Final Modifications to (micro) SD card with Murata Customizations

We now have to copy the compiled kernel, DTB file(s)[10], and optional bcmdhd driver to the (micro) SD card. With mounted (micro) SD card, execute the following commands:

```
$ sudo cp arch/arm/boot/zImage /media/<i.MX Boot Folder Name>/
$ sudo cp arch/arm/boot/dts/<i.MX DTB file>  /media/<i.MX Boot Folder Name>/
```

If configuring Linux 4.1.15_2.0.0 release for MFGTEST, then copy over "bcmdhd.ko" after creating the necessary subfolders:

```
$ 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 drivers/net/wireless/bcmdhd/bcmdhd.ko /media/<root file system string>/lib/modules/<4.1.15…>/kernel/drivers/net/wireless/bcmdhd/
```

[10] Reference **Table 2: i.MX6/7 Targets supported on Linux 3.14.52_1.1.0/4.1.15_2.0.0 GA Releases** for DTB filenames. For a given i.MX Reference Platform typically only one DTB file is used. The only scenario when two DTB files are copied over is if we want to test both SDIO in-band and OOB IRQ configuration on i.MX6UL/ULL EVK's.

To add/update Type 1CK and SN8000 firmware files, use the Murata i.MX Linux Source Code Patches. With the patch already extracted, execute the following commands:

```
$ cd <Murata i.MX Linux Source Code Patch extracted folder>
$ sudo chown -R root:root *
$ sudo cp -Rfp NVRAM/* /media/<root file system string>/
```

If we want to be able to change module configuration on the fly (without recompiling kernel) then we need to be able to change the NVRAM and firmware files being loaded. One way of accomplishing this is to edit the "/etc/network/if-pre-up.d/wpa-supplicant" file and insert the following strings (example below shows ZP configuration with OOB IRQ):

```
echo /lib/firmware/bcm/ZP_BCM4339/fw_bcmdhd.bin > /sys/module/bcmdhd/parameters/firmware_path
echo /lib/firmware/bcm/ZP_BCM4339/bcmdhd.ZP.SDIO.cal > /sys/module/bcmdhd/parameters/nvram_path
```

Note: The "/sys/module/bcmdhd/parameters/*" are not persistent between kernel reboots.

With the bcmdhd driver built into the kernel, we can configure the system to automatically bring up the WLAN interface when the kernel boots. This is accomplished by editing the "/etc/network/interfaces" file and adding/changing the following:

```
# Wireless interfaces
    auto wlan0                  ← new entry to configure WLAN during kernel boot.
    iface wlan0 inet dhcp
    wireless_mode managed
    wireless_essid any
    wpa-driver nl80211          ← Modify this entry.
    wpa-conf /etc/wpa_supplicant.conf
```

⇨ **Changes are now complete to (micro) SD card: excluding firmware/NVRAM updates.**

4.4 WLAN Firmware and NVRAM Update Considerations

Prior to completing changes to (micro) SD card, it is recommended that the user check for latest firmware (fw_bcmdhd.bin) and NVRAM (bcmdhd.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 4: Murata Module to Firmware/NVRAM Mapping** for correct folder location and filenames.

5 Special Wi-Fi Driver Features

The Cypress bcmdhd driver has a number of features that require additional consideration. These features include:

- Out-Of-Band IRQ (OOB) versus SDIO in-band interrupts. Murata patches provided to support OOB IRQ configuration.
- Handling Edge versus Level sensitive interrupts: particularly for Murata Type 1BW (BCM43340) and SN8000 (BCM43362). Patch provided for OOB IRQ edge-sensitive interrupt.
- 1.8V versus 3.3V VIO signaling.
- Throughput Optimization for 802.11ac Wi-Fi.

5.1 Out-Of-Band (OOB) versus SDIO In-Band Interrupts

Just a quick backgrounder before we get into more software details on these two different types of interrupts:

- SDIO in-band interrupts use a time multiplexing scheme (sharing the SDIO_DATA1 line) to signal the Host CPU that the WLAN core has a pending interrupt status. This scheme is more convenient (for hardware design) given that a dedicated IRQ line is not required. However the key drawback is that we cannot optimize power consumption and high speed throughput.

- Out-Of-Band (OOB) makes use of the dedicated WL_HOST_WAKE line. This interrupt configuration allows the WLAN core to wake the Host CPU when the Host is suspended (i.e. suspend/resume sequence). This configuration allows for lower overall system power consumption, especially in situations where the WLAN core is in IEEE power-save mode and can filter specific packets that are intended for the Host CPU.

- Out-Of-Band (OOB) interrupts also allows for system performance enhancement. This is especially applicable for higher throughput 802.11ac applications (SDIO bus runs at 200 MHz in SDR104 mode). The OOB IRQ reduces one of the system bottlenecks (SDIO in-band interrupt latency), thereby allowing a higher overall throughput of Wi-Fi data.

5.2 Handling Edge versus Level Sensitive Interrupts

Two Murata modules require additional consideration when it comes to OOB interrupts: Type 1BW (BCM43340) and SN8000 (BCM43362). These chipsets generate edge sensitive interrupts versus level sensitive. As a result, the Host GPIO monitoring OOB interrupts must be configured specifically for edge sensitive interrupts. Murata patch has been provided to support this interrupt configuration.

5.3 1.8V versus 3.3V VIO Signaling

The Murata Wi-Fi/BT EVK is configured for standard 3.3V VIO signaling on all the i.MX6 reference platforms. By contrast, the i.MX 7Dual SDB is configured for 1.8V VIO signaling. Note that the VIO level has to be the same for both the Wi-Fi SDIO interface and the Bluetooth UART interface.

For SDIO 2.0 compliant devices, the 3.3V VIO interface is adequate. However for SDIO 3.0 devices such as Type ZP & 1CK modules, the 1.8V VIO signaling is necessary to achieve SDR104 mode if high throughput is required (i.e. WLAN TCP performance greater than 100 Mbps). In ideal conditions, the Murata Type ZP & 1CK modules can support TCP performance up to 300 Mbps TX/RX.

The patches provided by Murata only support 3.3V VIO signaling.

5.4 Throughput Optimization for 802.11ac Wi-Fi

There are specific tuning parameters in the bcmdhd driver to optimize WLAN throughput. These parameters are used to reduce any bottlenecks over the SDIO bus. They are only applicable for the 802.11ac modules such as Type ZP and 1CK. The remaining 802.11n modules don't require additional optimization – as their SDIO clock speed MAXes out at 50 MHz. If the bcmdhd driver's Makefile is examined, there is a "CONFIG_BCM4339" which pulls in these specific tuning parameters.

The patches provided by Murata do not currently support this optimization.

6 Technical Support Resources

Table 7 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 email alias. All website/email address are hyperlinked in the "Support Site" column below.

Table 7: List of Support Resources

Support Site	Notes
Murata i.MX Landing Page	**No** login credentials required. This is an excellent starting point to understand all hardware/software configurations supported. Quick Start Guides and Murata Module Datasheets provided.
My Murata i.MX Support Portal	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 this guide.
NXP Website	Register on NXP.com so you can download necessary demo/validation images, documentation, schematics, etc.
NXP Murata i.MX Support Portal	Login credentials required. NXP's support forum for this i.MX Wi-Fi/BT solution. Both NXP and Murata Team support this portal. **Note:** same username/password used for NXP Website (above). Once registered on the NXP Community, email your username to imxfaq@murata.com to join this group.
Cypress Murata i.MX Support Portal	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 imxfaq@murata.com to join this group.
Cypress Linux Support Portal	Login credentials required. Includes support forum, chipset datasheets, application notes, original bcmdhd driver code releases, etc. Cypress Team supports this forum.
Murata i.MX FAQ Email	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.