QUALCOMM®
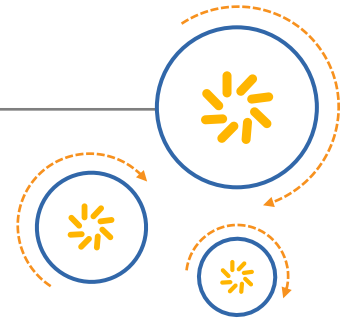
Qualcomm Technologies, Inc.

IoE Wireless System Release (I-WSR) QCA4010.TX.2.0 Hostless SDK for QCA4010/12

Release Notes

80-YA116-15 Rev. A

January 31, 2017

Qualcomm Technologies, Inc. ("QTI") and its affiliates reserve the right to make any updates, corrections and any other modifications to its documentation. The information provided in this document represents QTI's knowledge and belief as of the date this document is provided. QTI makes no representation or warranty as to the accuracy of such information, and QTI assumes no liability for any use of the information in this documentation. You should obtain the latest information before placing orders for any hardware, and you should verify that such information is current and complete. Information published by QTI regarding any third-party products does not constitute a license to use such products or a warranty or endorsement thereof. Use of such information may require a license from a third party under the intellectual property rights of such third party, or a license from QTI or its affiliates under the intellectual property rights of QTI or its affiliates.

All hardware, equipment, components or products are sold subject to QTI's (or such other QTI affiliated company that is designated by QTI) standard terms and conditions of sale, as applicable. Notwithstanding anything to the contrary in this documentation or otherwise: (i) you do not receive any rights, licenses, or immunities from suit under any patents of QUALCOMM Incorporated, QTI or their respective affiliates as a result of receiving this documentation (whether expressly, impliedly, by virtue of estoppel or exhaustion, or otherwise), (ii) without limitation, you shall not use or sell any wireless wide area network ("WWAN") baseband integrated circuit that you purchase or acquire from QTI or any product that incorporates any such WWAN baseband integrated circuit (whether alone or in combination with any other software or components) without a separate license or non-assertion covenant from QUALCOMM Incorporated in respect of or under all applicable patents, (iii) nothing in this document modifies or abrogates your obligations under any license or other agreement between you and QUALCOMM Incorporated, including without limitation any obligation to pay any royalties, and (iv) you will not contend that you have obtained any right, license, or immunity from suit with respect to any patents of QUALCOMM Incorporated, QTI or their respective affiliates under or as a result of receiving this documentation (whether expressly, impliedly, by virtue of estoppel or exhaustion, or otherwise).

Revision history

Revision	Date	Description
A	January 2017	Initial release

Contents

Tables

1 Introduction

This document provides details on the QCA4010.TX.2.0 release. The release version is QCA4010.TX.2.0 (r00059.1). All open source components and patches are available for download from publicly accessible servers.

Instructions for using this release are described in the Building and Running Demos chapter.

2 Features

This section shows the I-WSR (Internet of Everything Wireless System Release) features included in this release. This release (QCA4010.TX.2.0) contains the R5.0 QCA4010 firmware version. QCA4010.TX.2.0 and I-WSR R5.0 terms are used interchangeably below.

O   Supported on the RB01/RB02 platform, hostless operation

	QCA4010 TX.1.1	QCA4010 TX.2.0
Development tools		
Xtensa 10.0.3 (RE-2013.3)	O	O
ThreadX V5.6	O	O
Interfaces		
SPI master	O	O
UART	O	O
UART Power Save	O	O
I^2C (master)	O	O
I2C (slave)	O	O
I^2S (master)	O	O
I^2S (slave)	O	O
PWM	O	O
ADC	O	O
Manufacturing		
MAC address programming	O	O
OTP programming	O	O
Flash programming	O	O
USB-based thin mode (without MCU)	O	O
Self-initialization	O	O

Network stack		
TCP	O	O
UDP	O	O
BSD socket	O	O
SSL	O	O
HTTP server	O	O
HTTP client	O	O
SNTP	O	O
IPv4		
DHCP client	O	O
DHCP client release/renew	O	O
DHCP server	O	O
DNS server	O	O
DNS client	O	O
ICMP	O	O
IPv6		
Off mode	O	O
Router advertisement	O	O
DNS client	O	O
Applications, utilities, and demos		
Command line shell	O	O
OTA firmware upgrade	O	O
Ping	O	O
Fail-safe firmware upgrade	O	O
Qonstruct tunables	O	O
Demos		
Concurrent demo SCC (SoftAP +STA)	O	O
Concurrent demo MCC (SoftAP + STA)	O	O
Hostless UART	O	O
AllJoyn demo	O	O
UART AT Commands Demo	O	O
Dual UART Demo	O	O

Wi-Fi features		
2.4 GHz HT20 STA	O	O
STBC, LDPC, MLD, MRC, Short GI	O	O
AMPDU	O	O
AMSDU Rx	O	O
Regulatory	O	O
Antenna diversity		
Dynamic Rx diversity (per packet)	O	O
Static Rx diversity/Tx diversity	O	O
Low power		
Store-recall	O	O
Raw mode	O	O
Low power listen	O	O
Green Tx	O	O
Rate reporting	O	O
Early Beacon Termination	O	O
Security		
Open authentication	O	O
WEP	O	O
WPA/WPA2 PSK	O	O
WPS/WPS 2.0	O	O
SoftAP mode		
Soft AP	O	O
Hidden SSID	O	O
Soft AP: Multiple STAs	O	O
Standards		
802.11b	O	O
802.11d	O	O
802.11g	O	O
802.11h (radar)	O	O
802.11n	O	O
Certifications		

Wi-Fi 802.11n STA	O	O
WPS2.0	O	O
Security		
Secure Boot	O	O
Security Enhancement	O	O
Others		
SSDP	O	O
ezXML parser	O	O
JSON parser	O	O
Integrated and verified with Google Weave Beta version		O
Chip support		
QCA4010	O	O
QCA4012	O	O

3 Supported Hardware

3.1 Development kit

The QCA4010.TX.2.0 hostless SDK is supported on RB01/RB02 platform.

Table 3-2 Boards and modules for RB01/RB02 platform

Hardware	Part number	Description
RB01	20-YA162-H1	The RB01 platform provides Wi-Fi connectivity and system interfaces using a modular approach. The RB01 main board accepts a Wi-Fi module and implements system level connectivity. The RB01 board includes high-speed UART, SDIO and SPI connectors to enable communication with external MCUs.
RB02	20-YA163-H1	The RB02 module uses the QCA4010 1 x 1 single-band 802.11b/g/n SoC. The RB02 module supports 2.4 GHz operation only with integrated balun solution. The 20-YA163-H1 module includes a QCA4010 Rev. 1.1 chip.

4 Limitations

4.1 QCA4010.TX.1.1 CS limitations

No limitations were found in the 4.1 CS release.

4.2 QCA4010.TX.2.0 CS limitations

The HTTP message body size is limited to 1782 bytes.

5 Known Issues

5.1 QCA4010.TX.1.1 known issues

- **IP Stack:** When DUT functions as a softAP, it can impact AllJoyn connection on receiving many DNS packets.

- **OTA:** If AP aggregation is enabled, DUT can fail in OTA upgrade with FTP error ERR1003. FTP OTA upgrade can hang up the DUT console when there is no Wi-Fi connection.

- **PWM:** PWM Voltage is about 1.6 V when duty cycle=0.01% and 99% with f=132kHz in port0.

- **ADC:** ADC APP init and connect fails during ADC recycle-test.

- **AllJoyn:** AllJoyn dash_board can send command when the IP address has still not been received from DHCP.

- **MTBF:** DUT occasionally hangs up after receiving TCP or UDP traffic with or without epc info (PC value at crash point). When epc info is reported, error report will show: "epc1=0x90293e, epc2=0x90fd90"

- **Secure boot**: Partition 1 cannot boot up if OTA upgrade image and secure boot are enabled. The factory image version updates OTP version. No value check for security version, working image with version 18 is stored as version 16 in OTP.

- **Stress**: Assertion CRASH occurs on the DUT when trying to bring up SoftAP during the SnS test process.
  - Single device (STA Mode) occasionally has disconnect issue.
  - Single device (SoftAP mode) has disconnection issue.
  - Dual devices (SoftAP+STA), SoftAP of DUT has many disconnection issues.

5.2 QCA4010.TX.1.1 fixed issues/new features

- **Multiple UART**: When HS-UART(UART1,2) stress test is run, traffic frame loss rate is around 8% at 115200 and 9600 baudrates.

- **Networking**: IPv6 is disabled before suspend, but IPv6 becomes available after suspend.

- **PWM**: PWM voltage is not within specification on Port7. Please check 80-ya116-4 document for PWM specification.

5.3 QCA4010.TX.2.0 fixed issues/new features

- **New Features**

  - HTTP Client enhancements: support for PUT/PATCH methods, chunked encoding, hanging GET, multiple HTTP client sessions, per-request timeout

  - HTTP Server enhancements: support for PUT/DELETE/HEAD methods, redirection of unknown URLs, restricting requests from LAN

  - New API to configure SSL contexts

  - New API to configure time used for TLS certificate verification

  - Added support for two concurrent SoftAP sessions.

  - New API to configure Custom IE in Beacon, Probe Response for HomeKit support

  - New API to configure buffer pool size

  - New API to control timeouts for TCP connection setup, SSL connection setup and Wi-Fi scan

- **Functionality**: Fixed the issue that ping fails with IP & MAC Level Fragmentation.

- **Early Beacon Termination:** Fixed the issue that EBT function fails when BLE Coex mode is enabled.

- **SoftAP**: Fixed the issue that the SoftAP SSID maximum length is 31 not 32.

- **AP+STA SCC/MCC**: Fixed the issue that AP+STA SCC/MCC routing function occasionally does not work.

- **MTBF**:

  - Fixed the issue that DUT is already connected to reference AP but occasionally fails to ping the reference AP until Wi-Fi scan is performed.

- Fixed the OTA HTTPS receive timeout issue.

- Fixed the issue that DUT fails to boot when NUM_IRAM_BANKS is changed in tunable_table.txt.

- Fixed the issue that qcom_dnsc_get_host_by_name fails to resolve hostname longer than 64 characters.

- Fixed the issue that after multiple times of power-on and power-off, a reference AP cannot be connected after disconnection.

5.4 QCA4010.TX.2.0 known issues

- **OTA:**

  - HTTP server function of OTA image occasionally does not work.

  - Upgrade/degrade DUT from HTTPS server occasionally fails.

- **HTTP**: When the Wi-Fi profile is null, the HTTP PUT method does not work.

- **MTBF:** DUT crashes with info "Addr err: NULL ptr deref"epc4=0x927973,AHB_err=0x80080000.

- **HTTP server:** HTTP PUT method to create Wi-Fi profile object fails, but updates are OK.

- **HTTP client:**

  □ HTTP client sometimes does not handle invalid certificate error from SSL client. This happens irrespective of TLS alert being enabled or not.

  □ After HTTPS client fails to get html once and times out, DUT fails to send SSL packet in the second time, and passes with v40.

- **Weave**: QCA4010 SDK hits exception after running 25+ hours with Weave.

6 Building and Running Demos

The QCA4010/12 software release is a mix of software to be downloaded from Qualcomm Technologies-hosted servers and software available from third-party vendor websites. This chapter discusses the procedure to combine Qualcomm Technologies and third-party software, and build/run the demos included in the Qualcomm Technologies release.

6.1 Deliverables

The QCA4010.TX.2.0 release consists of proprietary software by Qualcomm Technologies, Inc. and by third-party vendors.

- QCA4010/12 software package is hosted on Qualcomm Developer Network https://developer.qualcomm.com/hardware/qca4010-12. This package contains demonstrations and customer-use specific libraries in the source form and Qualcomm Technologies' proprietary code as firmware libraries. Qualcomm Technologies does not provide source of the firmware libraries. Package also contains tools and scripts needed to compile and build a standalone application that runs on QCA4010/12.

- QCA4010/12 use Tensilica core and as a result use Cadence Xtensa toolchain. See section 6.2.1 for installation of Cadence Xtensa toolchain.

- A script is used to download and compile AllJoyn applications for this release. See section 6.2.2 for details.

6.1.1 Qualcomm Technologies packages on QDN

User needs to register at the QDN site https://developer.qualcomm.com to access QCA4010/12 driver SDK, firmware and documentation.

6.1.2 Third-party products

Table 6-1 lists the packages to be downloaded from external websites.

Table 6-1 Third-party products

Vendor	Package	Type
Cadence	Xplorer-5.0.3-linux-installer.bin	OS
Cadence	pgblrm3_linux.tgz	OS
AllSeen Alliance	core/ajtcl	AllJoyn core
AllSeen Alliance	services/base_tcl	AllJoyn services

6.1.3 Reference documentation

Table 6-2 lists the supporting Qualcomm Technologies documents available for this release.

Table 6-2 Reference documentation

Doc. number	Doc. title
80-YA116-15 (this document)	IoE Wireless System Release (I-WSR) QCA4010.TX.2.0 Hostless SDK for QCA4010/12 Release Notes
80-YA116-14	QCA4010/QCA4012 Hostless SDK (QCA4010.TX.2.0) User Guide

6.2 Preparing the build environment

NOTE: A Linux system is required. These steps have been verified on an Ubuntu 12.04 32-bit system and an Ubuntu 14.04.1 32-bit system running a 3.13.x kernel.

NOTE: An Internet connection is required for most of the steps in this section, as installers and dependent packages must be downloaded.

1. Download the QCA4010.TX.2.0 package from Qualcomm Developer Network (QDN) here:

   https://developer.qualcomm.com/hardware/qca4010/tools

2. Unzip and extract the package into any directory. This directory is referred below as `<install dir>`.

6.2.1 Installing Cadence Xtensa Compiler Toolchains and Tools

1. Download the Xtensa Toolchain by executing <install dir>/**Xtensa.fetch.sh** script in the software release package. The toolchain installers are downloaded to a new folder **RE-2013.3-pgblrm** under the script's directory. For the configuration parameters, execute command.`/install*` in folder **pgblrm3**. The total size of the installers is about 2 GB.

2. Install Xplorer and Xtensa CLI tools.

   a. Execute the Xtensa Xplorer Setup Wizard **Xplorer-5.0.3-linux-installer.bin** from the Linux system and follow the onscreen instruction to install. Make sure Xplorer and Xtensa Development Tools are selected and record the location where the tools are installed (for step 3-a).

   b. After the installation is complete, allow the installer to start the Xplorer.

   c. Install the Xtensa license file supplied by Qualcomm Technologies at **target/license.dat**. In the Xplorer GUI, click **Install Software Keys** and navigate to the license file. Click **Finish** and close the window.

   d. In the Xplorer workbench, right click **Configurations** in the **System Overview** panel. Click **Find and Install a Configuration Build**. In the popup window, next to **Browse for build download to install**, click **Browse…** and navigate to target/RE-2013.3-pgblrm/pgblrm3_linux.tgz. Click **OK**, and wait for the installation to complete.

    e.   Exit Xtensa Xplorer.

3.   Setup environment variables for command line access to Xtensa tools.

    a.   Place the following variables either in a script to be sourced later, or in a .bashrc (or equivalent) startup file.

NOTE: The value to fill in for <TOOL_INSTALL_DIR> is the value recorded from step 2-a. As an example: /home/user/xtensa

```
export XTENSA_INST=<TOOL_INSTALL_DIR>
export XTENSA_CORE=pgblrm3
export XTENSA_ROOT=$XTENSA_INST/XtDevTools/install/builds/RE-2013.3-
    linux/pgblrm3
export XTENSA_SYSTEM=$XTENSA_ROOT/config
export XTENSA_TOOLS_ROOT=$XTENSA_INST/XtDevTools/install/tools/RE-
    2013.3-linux/XtensaTools
export LM_LICENSE_FILE=~/license.dat
export PATH=$PATH:$XTENSA_TOOLS_ROOT/bin
```

    b.   Either source the script created in 3-a, or open a new terminal if using .bashrc in 3-a.

    c.   Test the environment variable settings by compiling a simple program.
```
echo 'main(){}' > test.c
xt-xcc test.c
```

4.   (Optional) Install Xtensa OCD Daemon.

Installation of the Xtensa OCD Daemon (XOCD) is necessary to flash firmware images to the SP241/SP242 modules **via JTAG**.

Installation of the Xtensa OCD Daemon is required to fully utilize the capabilities of Xtensa Xplorer for QTI Platform Development (see section 6.2.3)

XOCD can be installed on either a Windows machine or a Linux machine. It can therefore be installed and run on:

□   the same Linux machine that Xtensa Xplorer and the Xtensa CLI tools were installed on in step 6.2.1.2.

□   a Windows machine that is on the same local network as the Linux machine.

    a.   Installing XOCD on Windows.

        Refer to the instructions in section 6.2 *Installing the Xtensa OCD Daemon (XOCD) on Windows* in the *Tensilica Debug User's Guide*. This document can be found at <TOOL_INSTALL_DIR>/XtDevTools/downloads/RE-2013.3/docs/ten_debug_guide.pdf.

    b.   Installing XOCD on Linux.

NOTE: These steps have been verified on an Ubuntu 12.04 32-bit system and an Ubuntu 14.04 32-bit system running a 3.13.x kernel. If your kernel is newer than 3.13.x, these steps may not work. In this case, contact your FAE for assistance.

        i   Install the Macraigor drivers.

Download the Macraigor support DEB from the Macraigor website: http://macraigor.com/gnu/mcgr-hwsupport-12.0-0_i386.deb. The Macraigor documentation is available for reference: http://macraigor.com/gnu_faq.htm#install%20mcgr_hwsupport%20DEB

(a) Install libjpeg62 dependency: `sudo apt-get install libjpeg62`

(b) `wget http://macraigor.com/gnu/mcgr-hwsupport-12.0-0_i386.deb`

(c) Install Macraigor support package: `sudo dpkg --install mcgr-hwsupport-12.0-0_i386.deb`

ii Make the following symlinks so the xt-ocd installer can find the Linux kernel sources it needs.

NOTE: Replace <linux-kernel-version> with the value for your machine, you can use the `uname -a` command to get the kernel version.

```
sudo ln -s /usr/src/linux-headers-<linux-kernel-version>-
generic/include/generated/utsrelease.h /usr/src/linux-headers-
<linux-kernel-version>-generic/include/linux/utsrelease.h

sudo ln -s /usr/src/linux-headers-<linux-kernel-version>-
generic/include/generated/uapi/linux/version.h /usr/src/linux-
headers-<linux-kernel-version>-generic/include/linux/version.h
```

iii Run the Xtensa OCD Daemon Setup Wizard.

`cd $XTENSA_INST/XtDevTools/downloads/RE-2013.3/tools/`

`sudo`

`./xt-ocd-10.0.3-linux-installer --mode xwindow`

Do not edit the default installation directory of: `/opt/tensilica/xocd-10.0.3`

On 'Select Components' screen, make sure the Macraigor Drivers and Libraries entry is checked

On 'Kernel Sources' screen, point to: `/usr/src/linux-headers-<linux-kernel-version>-generic`

NOTE: Replace <linux-kernel-version> with the value for your machine, you can use the `uname -a` command to get the kernel version.

Follow instructions for the remaining steps and let the installer complete.

iv After installation, on some systems modify **/opt/tensilica/xocd-10.0.3/topology.xml** to indicate the JTAG controller in use.

5. (Optional) Test xt-gdb and JTAG.

Once XOCD has been installed on either the same Linux machine as Xtensa Xplorer and the Xtensa CLI tools or on a Windows machine on the same local network, the following steps can be used to verify the setup.

a. Follow the instructions in section 6.5.1 to setup the Qualcomm Technologies IoE development kit and connect it to the computer that has XOCD installed on it via the JTAG.

NOTE: When using a virtual machine, make sure that the JTAG device is captured by the Virtual Machine before proceeding. Prior to launching xt-ocd, the JTAG may appear as an unknown device in your virtual machine device list, similar to: "Unknown device 08AC:2024"

b. Launch XOCD. Follow the steps below according to which machine has XOCD installed.

i   Launch XOCD on Windows.

XOCD can be launched from Start > All Programs > Xtensa OCD Daemon > Xtensa OCD Daemon or via a command prompt.

ii   Launch XOCD on Linux.

```
cd /opt/tensilica/xocd-10.0.3
./xt-ocd -dTD
```

NOTE: When using a virtual machine, you may need to recapture the JTAG device after launching xt-ocd. It should appear in your virtual machine device list similar to: "Macraigor usb2demon(tm)"

Example output:
```
XOCD 10.0.3
(c) 1999-2015 Tensilica Inc. All rights reserved.
Loading module "gdbstub" v2.0.0.11
Loading module "macraigor" v2.0.0.5
Loading module "jtag" v2.0.0.20
Loading module "xtensa" v2.0.0.40
Loading module "traxapp" v2.0.0.8
Loading module "trax" v2.0.1.22
 Total IR bits : 5
  TAP[0] irwidth = 5
 Total DR bypass bits : 1
Starting thread 'GDBStub'
Starting thread 'TraxApp'
0: TAP IR access succesful
```

c. On the Linux machine that has Xtensa Xplorer and the Xtensa CLI tools installed:

In a terminal, launch xt-gdb. Execute the commands below to connect to the xt-ocd instance from step b and run two gdb commands to verify the setup.

NOTE: Errors will occur if the environment variables in Step 3-a have not been set.

i   If XOCD is running on the same machine as xt-gdb:
```
xt-gdb
(xt-gdb) target remote localhost:20000
(xt-gdb) x/i $pc
(xt-gdb) reset
```

ii   If XOCD is running on a different machine than xt-gdb:
```
xt-gdb
(xt-gdb) target remote <IP Address of XOCD Machine>:20000
(xt-gdb) x/i $pc
```

```
(xt-gdb) reset
```

    d. Exit from xt-gdb.

    e. Exit from XOCD (either close xt-ocd on Linux, or XOCD on Windows).

6.2.2 Installing AllJoyn software

1. Extract the source code and directories from the release package.

2. Go to the target directory and run the installer.

```
cd target
./demo/install-alljoyn.sh
```

6.2.3 Setting up Xtensa Xplorer for QTI Platform Development

1. The following steps cover setting up the Xtensa Xplorer IDE to allow developers targeting QTI hardware such as the QCA4010/QCA4012 to create AllJoyn applications.

NOTE: These steps have been verified on an Ubuntu 12.04 32-bit system and an Ubuntu 14.04.1 32-bit system.

2. Make sure the following dependencies are installed. On Ubuntu, test by running:

```
sudo apt-get install python-setuptools

sudo easy_install -U pyserial

sudo apt-get install libxml2 libxml2-utils libxml2-dev
```

3. Run the installer, providing the Xtensa base installation directory path (same directory as step 6.2.1.3.a) as an argument:

```
cd target/demo/aj/alljoyn/sde
./install.sh --xtensadir $XTENSA_INST
```
Example: `./install.sh --xtensadir /home/user/xtensa`

4. Install XOCDM.

    a. Launch Xtensa Xplorer.

    b. Choose a workspace if prompted.

    c. In Xplorer, go to Tools > XOCDM Manager > Create zip for installation.

    d. XOCDM must be installed on the same machine that XOCD was installed on in section 6.2.1.4.

        i If XOCD was installed on a Linux machine:

          (a) Make sure Linux is selected, and choose a location to save the package.

          Example: `~/Downloads/xocdm.tar`

          (b) Click **Create**, wait for the process to compete, and then click **OK**.

          (c) Exit Xplorer.

          (d) Extract and setup XOCDM.

```
cd ~/Downloads
```

```
tar –xvf xocdm.tar

cp –r xocdm5.0.2 $XTENSA_INST/Xplorer-5.0.3/utils/xocdm5.0.2
```

    (e) Open the file $XTENSA_INST/Xplorer-5.0.3/utils/xocdm5.0.2/xocdm.ini.

    (f) Add this line at the end: 10.0.3=/opt/tensilica/xocd-10.0.3

    (g) Save and close the file.

  ii If XOCD was installed on a Windows machine:

    (a) Make sure Windows is selected, and choose a location to save the package.

    (b) Copy/move the package to the Windows machine.

    (c) Extract the package on the Windows machine.

    (d) In a command prompt, run start_xocdm.bat to verify the setup was successful.

    For more information and details, see the Xtensa Xplorer documentation on "Installing XOCD Manager". This can be accessed by launching Xtensa Xplorer, then going to Help > Help Contents, and performing a search for "Install XOCDM".

5. Documentation for developing an AllJoyn application using the plugin is available inside Xtensa Xplorer. Launch Xtensa Xplorer, go to Help > Help Contents, and expand the entry for "AllJoyn Development Guide".

6.3 Building the software

This section describes the process to build the hostless firmware binary image with SDK shell and demo applications. The build requires a Linux machine with Xtensa toolchain. The firmware source is available in the **target** folder in the release package.

■ SDK shell

The SDK shell allows users to run various commands to demonstrate the features available in this release. The shell also provides reference source for customer application development. Refer to *QCA4010/QCA4012 Hostless SDK (QCA4010.TX.2.0) User Guide* for details.

■ AllJoyn Service Sample

The AllJoyn Service Sample demonstrates how to use AllJoyn services and the core AllJoyn software. The sample implements a simulated air conditioner device.

Hostless software package directories

```
target
|-- bddata
|   |-- boardData_2_1_MP1_1124.bin
|-- bin
|   |-- raw_flashimage_AR401X_REV6_IOT_hostless_single.bin  //Sample binary
|-- demo       //Demo apps source code
|   |-- aj
|   |   |--alljoyn
|   |   |   |--sde
|   |   |--allseen
|   |-- sdk_flash
```

```
|   |-- sdk_proxy
|   |-- sdk_shell
|-- image
|   |-- sdk_flash.out
|-- lib
|-- tool
|-- sdkenv.sh
```

6.3.1 Building the firmware images

Run the following commands on the standard "bash" shell.

1. Setup the environment variables required for building the binary image.

   source sdkenv.sh

2. Build the target demo for application.

   To use sdk_shell application, go to **target/demo/sdk_shell** and build the IOT demonstration.

   ```
   cd target/demo/sdk_shell
   make clean all
   make
   ```

   This step generates an image to the **target/image** directory.

   The **iot_demo.out** image is a sample application demonstrating OTA, HTTP, SSL, DNS, and so on.

3. Define and build the images.

   a. Make sure the standard library **libxml2** is installed. On Ubuntu, test by running:

      ```
      sudo apt-get install libxml2 libxml2-dev
      ```

   b. Select the **tunable_input.txt** file and modify this file according to build image type:

      ```
      cd target/tool/tunable
      ```

      To build an RB02 iot_demo image:

      ```
      cp tunable_input_mp1_hostless_4bitflash.txt tunable_input.txt
      echo OP_FLAVOR $OP_FLAVOR >> tunable_input.txt
      echo BAND 1 >> tunable_input.txt
      echo "export FLASH_FLAVOR=\"_IOT_MP1\"" >> tunable_input.txt
      echo "export BD_FILENAME=$SDK_ROOT/bddata/boardData_2_1_SP241.bin" >>
      tunable_input.txt
      ```

NOTE: For RB02 board, only single band is supported, so BAND must be set to 1 only. FLASH_FLAVOR can be set to differentiate the names for RB02 image.

   c. Run Qonstruct.

      ```
      cd target/tool
      ./qonstruct.sh --qons /tmp/qons
      ```

   d. Follow the instructions at Qonstruct to build the demonstration images with various combinations of features.

ap+sta	AP+STA concurrency
Mcc	Multi-channel concurrency
Scc	Single channel concurrency
1bitflash	Single mode SPI flash access
4bitflash	Quad mode SPI flash access

- (Optional) Enable HTTP server. Specify the HTML page list in the following line:
  ```
  export HTMLTOFLASH_LIST=""
  ```

Various images can be generated to the **target/bin** directory based on the Qonstruct configuration.
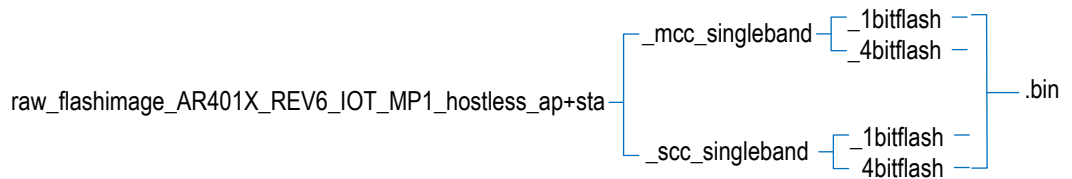
6.3.2 Building the firmware image for AllJoyn ServiceSample

This section provides the steps required for building the AllJoyn ServiceSample application. Before starting follow the steps section 6.3.1.

1. Go to AllJoyn Service Sample directory and build the demo:
   ```
   cd
   target/demo/aj/alljoyn/applications/sample_apps/tcl/ServicesSamples/targ
   et/aj_qca4004/aj_ServiceSample
   make clean
   make
   ```

2. (Optional) If /tmp/qons/tunable_input.txt does not exist (installation in section 6.3.1 not performed), run qonstruct.sh to generate it.
   ```
   cd target/tool
   ./qonstruct.sh --qons /tmp/qons
   ```

3. Open /tmp/qons/tunable_input.txt for editing. Uncomment and modify the following line to point at the AllJoyn ServiceSample application file:
   ```
   export APPS_OUTFILE="<YOUR_PATH>/target/image/aj_ServiceSample.out"
   ```

4. Run Qonstruct again to generate the firmware image.
   ```
   cd target/tool
   ./qonstruct.sh --qons /tmp/qons
   ```

This step generates one image to the directory target/bin/:

- raw_flashimage_AR401X_REV6_IOT_hostless_unidev_dualband_aj_ServiceSample.bin

6.4 Flashing instructions for RB01/RB02 platform

Users can use either JTAG (with OCD Daemon) or USB (with ART tool) to flash the firmware images to the RB02 module.

After the firmware image is downloaded to the RB02 module, setup the jumpers on the RB01 board following the instructions in section 6.4.1 to enable hostless operation.

6.4.1 Flashing image via JTAG

Before start, make sure the Xtensa OCD and JTAG are correctly configured as described in section 6.2.1.

Board setup

1. Set up the jumper so that board EJTAG is enabled and bootstrap is in hostless mode.

   Pull on the Jumper connecting JP3.1&2, JP11.2&1(Test mode), JP5.2&3(IOT mode), JP10.2&3(HOST0), JP9.2&1(HOST1) to make RB02 module in hostless mode. Pull on the Jumper connecting JP11.2&3(Test mode) to make module in JTAG mode.

2. Use a type-B USB cable to connect PC USB port and RB01 POWER-USB USB port to power up the RB01 board.

3. Use a USB JTAG Wiggler to connect PC USB port and JP12 of the RB01 board.

Flashing procedure

Before start, make sure the Xtensa OCD daemon is up and running on a Windows or Linux machine. On Linux machine:

1. Navigate to the binary image folder.

   ```
   cd target/image
   ```

2. Run the following command to flash the image.

   ```
   xt-gdb -x gdb.sdk_flash
   target remote <Xtensa_host_IP>:20000
   (xt-gdb) sdk_flash ../bin/<raw_flashimage*>.bin
   ```

3. Wait until xt-gdb prints "sdk_flash is DONE".

4. Reset the board.

   ```
   (xt-gdb) reset
   (xt-gdb) cont
   ```

NOTE: For RB02, the following updates are required to the corresponding tunable txt files (target\tool\tunable\tunable_input_mp1_hostless_4bitflash.txt and target\tool\tunable\tunable_input_mp1_hostless_1bitflash.txt).

line 88: change the value of GPIO6_ACTIVE_CONFIG from 0x90007800 to **0x80007800**

line 92: change the value of GPIO10_ACTIVE_CONFIG from 0x90007808 to **0x80007808**

line 94: change the value of GPIO12_ACTIVE_CONFIG from 0x90007808 to **0x80007808**

line 95: change the value of GPIO13_ACTIVE_CONFIG from 0x90007808 to **0x80007808**

line 103: change the value of GPIO21_ACTIVE_CONFIG from 0xd0000000 to **0x80000000**

line 133: change the value of GPIO6_INACTIVE_CONFIG from 0x90007800 to **0x80007800**

line 137: change the value of GPIO10_INACTIVE_CONFIG from 0x90007808 to **0x80007808**

line 139: change the value of GPIO12_INACTIVE_CONFIG from 0x90007808 to **0x80007808**

line 140: change the value of GPIO13_INACTIVE_CONFIG from 0x90007808 to **0x80007808**

line 148: change the value of GPIO21_INACTIVE_CONFIG from 0xd0000000 to **0x80000000**

6.5 Running demo applications

After the firmware/demo image is downloaded to the IoE development kit, users can use the applications to test and verify the functionality. Refer to *QCA4010/QCA4012 Hostless SDK (QCA4010.TX.2.0) User Guide* for demo usages.

6.5.1 Board setup and console connection with RB01/RB02

1. Use a USB to RS232 adaptor cable to connect the RB01/RB02 kit to a Windows-based PC through J1.6 (module RXD), J1.7 (module GND), and J1.8 (module TXD).

NOTE: The labels for TXD (J1.6) and RXD (J1.8) on the board are opposite to the actual settings.

2. Start a serial terminal application from the PC and select the lower port to connect using the port setting: 115200, 8, n, 1, no flow control.

3. The serial terminal application displays:
   ```
   shell>
   ```

A GPIO Function Configuration

This chapter contains the GPIO configuration options supported by the QCA4010/QCA4012 hardware.

NOTE: Not all the configurations listed below are tested or verified by Qualcomm Technologies. The default configurations for this release are provided in the **tunable_input.txt** file in the SDK package.

Pin name	Description		
GPIO[0]			
	GPIO configuration	**Interface**	**Signal**
	0	GPIO	SDIO CMD
	1	SPI	CS
	2	SPI	CS
	3	GPIO	SDIO
	4	SPI	CS
	7	Debug	OBS[0]
	8	Keypad	OUT[0]
GPIO[1]			
	GPIO configuration	**Interface**	**Signal**
	0	GPIO	SDIO data
	1	SPI master	MOSI
	2	SPI master	MOSI
	3	GPIO	SDIO data
	4	SPI master	MOSI
	6	-	ST RF Active
	7	Debug	OBS[1]
	8	Keypad	OUT[1]

Pin name	Description		
GPIO[2]			
	GPIO configuration	**Interface**	**Signal**
	0	GPIO	SDIO data
	1	UART	RXD
	2	UART	RXD
	3	GPIO	SDIO data
	5	UART	TXD
	6	-	ST RF status
	7	Debug	OBS_BUS[2]
	8	Keypad	IN[0]
GPIO[3]			
	GPIO configuration	**Interface**	**Signal**
	0	GPIO	SDIO data
	1	UART	TXD
	2	UART	TXD
	3	GPIO	SDIO data
	4	SPI	INT
	5	UART	RXD
	6	-	ST RF freq
	7	Debug	OBS_BUS[3]
	8	Keypad	IN[1]
GPIO[4]			
	GPIO configuration	**Interface**	**Signal**
	0	GPIO	SDIO data
	1	SPI master	MISO
	2	SPI master	MISO
	3	GPIO	SDIO data
	4	SPI master	MISO
	5	UART	RTS
	6	-	ST RC Req/Ack
	7	Debug	OBS_BUS[7]
	8	keypad	OUT[2]

Pin name	Description		
GPIO[5]	**GPIO configuration**	**Interface**	**Signal**
	0	SDIO	CLK
	1	SPI /I2C master	CLK
	2	SPI /I2C master	CLK
	3	SDIO	CLK
	4	SPI /I2C master	CLK
	5	UART	CTS
	7	Debug	OBS_BUS[5]
	8	Keypad	OUT[3]
GPIO[6]	**GPIO configuration**	**Interface**	**Signal**
	0	JTAG	TDI
	1 - 4	PWM	CH 0
	5	SPI master	MISO
	7	Debug	OBS_BUS[6]
GPIO[7]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	CLK REQ IN
	1 - 4	PWM	CH 1
	5	SPI master	MOSI
	7	Debug	OBS_BUS[7]
GPIO[8]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	CLK REQ OUT
	1 - 4	PWM	CH 2
	5	SPI master	CS
	7	Debug	OBS_BUS[8]
	8	Keypad	OUT[4]
GPIO[9]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[9]
	1 - 4	PWM	CH - 3
	5	SPI / I2C master	SCK
	6	I2S	BCLK
	7	Debug	OBS_BUS[9]
	8	Keypad	OUT[5]

Pin name	Description		
GPIO[10]	**GPIO configuration**	**Interface**	**Signal**
	0	JTAG	TMS
	1	PWM	CH- 4
	2 - 3	I2C	SCK
	4 - 5	PWM	CH - 4
	6	I2S	SDI
	7	Debug	OBS_BUS[10]
	8	Keypad	IN[2]
GPIO[11]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[11]
	1	PWM	CH - 5
	2	I2C	SDA
	3	I2C	SDA
	4	PWM	CH - 5
	5	PWM	CH - 5
	6	I2S	SDO
	7	Debug	OBS_BUS[11]
	8	Keypad	IN[3]
GPIO[12]	**GPIO configuration**	**Interface**	**Signal**
	0	JTAG	TCK
	1	PWM	CH - 6
	2	BT	Frequency
	3	BT	Frequency
	4	PWM	CH - 6
	5	PWM	CH - 6
	6	I2S	WS
	7	Debug	OBS_BUS[12]
	8	Keypad	OUT[6]
	9	-	SW_0_0

Pin name	Description		
GPIO[13]	**GPIO configuration**	**Interface**	**Signal**
	0	JTAG	TDO
	1	PWM	CH - 7
	2	BT	ACTIVE
	3	BT	ACTIVE
	4	PWM	CH - 7
	5	PWM	CH - 7
	6	I2S	MCLK
	7	Debug	OBS_BUS[13]
	8	Keypad	OUT[7]
	9	-	SW_0_1
GPIO[14]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[0]
	1	I2C master	SDA
	2	BT	Priority
	3	BT	Priority
	4	I2C master	SDA
	5	I2C master	SDA
	6	I2S	SCLK
	7	Debug	OBS_BUS[14]
GPIO[15]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[15]
	1	I2C master	SCK
	2	WLAN	Active
	3	WLAN	Active
	4	I2C master	SCK
	5	I2C master	SCK
	7	Debug	OBS_BUS[15]
GPIO[16]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[16]
	1-5	I2S	SCLK
	6	I2S	SDI
	7	Debug	OBS_BUS[16]
	8	Keypad	IN[6]

Pin name	Description		
GPIO[17]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[17]
	1-5	I2S	SDI
	6	I2S	SDO
	7	Debug	OBS_BUS[17]
	8	Keypad	IN[7]
GPIO[18]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[18]
	1-5	I2S	SDO
	6	I2S	WS
	7	Debug	OBS_BUS[0]
GPIO[19]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[19]
	1-5	I2S	WS
	6	I2S	MCLK
	7	Debug	OBS_BUS[1]
GPIO[20]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[20]
	1-5	I2S	MCLK
	6	BT	Freq
	7	Debug	OBS_BUS[2]
GPIO[21]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[21]
	1-4	HUART	RTS
	5	HUART	CTS
	6	BT	Active
	7	Debug	OBS_BUS[3]

Pin name	Description		
GPIO[22]			
	GPIO configuration	**Interface**	**Signal**
	0	GPIO	GPIO[22]
	1-4	HUART	CTS
	5	HUART	RTS
	6	BT	Priority
	7	Debug	OBS_BUS[4]
GPIO[23]			
	GPIO configuration	**Interface**	**Signal**
	0	GPIO	GPIO[23]
	1-4	HUART	RXD
	5	HUART	TXD
	6	Wlan	Active
	7	Debug	OBS_BUS[5]
GPIO[24]			
	GPIO configuration	**Interface**	**Signal**
	0	GPIO	GPIO[24]
	1-4	HUART	TXD
	5	HUART	RXD
	7	Debug	OBS_BUS[6]
GPIO[25]			
	GPIO configuration	**Interface**	**Signal**
	0	GPIO	GPIO[25]
	1-5	I2C	SDA
	7	Debug	OBS_BUS[7]
GPIO[26]			
	GPIO configuration	**Interface**	**Signal**
	0	GPIO	GPIO[26]
	1-5	I2C	CLK
	7	Debug	OBS_BUS[8]
GPIO[27]			
	GPIO configuration	**Interface**	**Signal**
	0	GPIO	GPIO[27]
	1-5	I2S	BCLK
	7	Debug	OBS_BUS[9]

Pin name	Description		
GPIO[28]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[28]
	1-4	Debug UART	TXD
	5	Debug UART	RXD
	7	Debug	OBS_BUS[8]
GPIO[29]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[26]
	1-4	Debug UART	RXD
	5	Debug UART	TXD
	7	Debug	OBS_BUS[11]
GPIO[30]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[30]
	1-5	I2S	SDI
	6	-	ST RF Active
	7	Debug	OBS_BUS[12]
	8	Keypad	IN [4]
GPIO[31]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[31]
	1-5	I2S	SDO
	6	-	ST RF status
	7	Debug	OBS_BUS[13]
	8	Keypad	IN [5]
GPIO[32]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[32]
	1-5	I2S	WS
	6	-	ST RF status
	7	Debug	OBS_BUS[14]
	8	Keypad	IN [6]

Pin name	Description		
GPIO[33]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[33]
	1-5	I2S	MCLK
	6	-	ST RF Req/Ack
	7	Debug	OBS_BUS[15]
	8	Keypad	IN [7]
GPIO[34]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[32]
	1-4	QUAD SPI master	CLK
GPIO[35]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[35]
	1-4	QUAD SPI master	CS0
GPIO[36]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[36]
	1-4	QUAD SPI master	DIO0
GPIO[37]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[37]
	1-4	QUAD SPI master	DIO1
GPIO[38]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[38]
	1-4	QUAD SPI master	DIO2
GPIO[39]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[39]
	1-4	QUAD SPI master	DIO3

Pin name	Description		
GPIO[40]	**GPIO configuration**	**Interface**	**Signal**
	0	GPIO	GPIO[40]
	1-4	QUAD SPI master	CS1
	5	-	CLK OBS
	6	-	ADC HW trigger