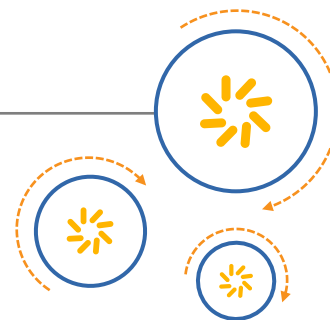




Qualcomm Technologies, Inc.



QCA4010.TX.2.2 Hostless SDK

Release Notes

80-YA116-21 Rev. A

November 3, 2017

Qualcomm is a trademark of Qualcomm Incorporated, registered in the United States and other countries. Other product and brand names may be trademarks or registered trademarks of their respective owners.

This technical data may be subject to U.S. and international export, re-export, or transfer ("export") laws. Diversion contrary to U.S. and international law is strictly prohibited.

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

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	November 2017	<p>Initial release. This document is based on 80-YA116-16_A with the following updates:</p> <ul style="list-style-type: none">▪ Added new features and a new column for 2.2 in the feature table in Chapter 2 Features.▪ Added Section 4.4 QCA4010.TX.2.2 CS limitations.▪ Added Section 5.7 QCA4010.TX.2.2 CS fixed issues/new features and 5.8 QCA4010.TX.2.2 CS known issues.▪ Updated Chapter 2 Features, 3 Supported Hardware, 4 Limitations, 5 Known Issues, and 6 Building and Running Demos for RB01/RB04 related information.

Contents

1 Introduction	6
2 Features	7
3 Supported Hardware	11
3.1 Development kit	11
4 Limitations	12
4.1 QCA4010.TX.1.1 CS limitations	12
4.2 QCA4010.TX.2.0 CS limitations	12
4.3 QCA4010.TX.2.1 CS limitations	12
4.4 QCA4010.TX.2.2 CS limitations	12
5 Known Issues	13
5.1 QCA4010.TX.1.1 known issues	13
5.2 QCA4010.TX.1.1 fixed issues/new features	13
5.3 QCA4010.TX.2.0 fixed issues/new features	14
5.4 QCA4010.TX.2.0 known issues	15
5.5 QCA4010.TX.2.1 fixed issues/new features	16
5.6 QCA4010.TX.2.1 CS known issues	17
5.7 QCA4010.TX.2.2 CS fixed issues/new features	17
5.8 QCA4010.TX.2.2 CS known issues	17
6 Building and Running Demos	18
6.1 Deliverables	18
6.1.1 Qualcomm Technologies packages on QDN	18
6.1.2 Third-party products	18
6.1.3 Reference documentation	19
6.2 Preparing the build environment	19
6.2.1 Installing Cadence Xtensa Compiler Toolchains and Tools	19
6.2.2 Installing AllJoyn software	23
6.2.3 Setting up Xtensa Xplorer for QTI Platform Development	23
6.3 Building the software	24
6.3.1 Building the firmware images	25
6.3.2 Building the firmware image for AllJoyn ServiceSample	27
6.4 Flashing instructions for RB01/RB02 or RB01/RB04 platform	28
6.4.1 Flashing image via JTAG	28
6.5 Running demo applications	28
6.5.1 Board setup and console connection with RB01/RB02 or RB01/RB04	29
6.5.2 WMI commands	29
6.6 Example Applications	31
6.6.1 Simple DNS Server	31
6.6.2 DNS Client	32
A GPIO Function Configuration	33

Tables

Table 6-1 Third-party products	18
Table 6-2 Reference documentation	19

1 Introduction

This document provides details on the QCA4010.TX.2.2 release for QCA4010/QCA4012. The release version is QCA4010.TX.2.2 (r00012.1). 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 QCA4010.TX.2.2 features included in this release.

- Supported on the RB01/RB02 platform, hostless operation
- Supported on the RB01/RB04 platform, hostless operation

	TX.1.1	TX.2.0	TX.2.1	TX.2.2
Development tools				
Xtensa 10.0.3 (RE-2013.3)	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ThreadX V5.6	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Interfaces				
SPI master	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
UART	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
UART Power Save	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I ² C (master)	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I2C (slave)	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I ² S (master)	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
I ² S (slave)	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
PWM	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ADC	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Manufacturing				
MAC address programming	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
OTP programming	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Flash programming	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
USB-based thin mode (without MCU)	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Self-initialization	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Network stack				
IPv4	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
IPv6	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
IPv6 Off mode	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Network Services				
TCP	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
UDP	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
BSD socket	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
SSL	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

	TX.1.1	TX.2.0	TX.2.1	TX.2.2
HTTP server	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
HTTP client	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
SNTP	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
DHCP client	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
DHCP client release/renew	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
DHCP server	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
DNS server	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
DNS client	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ICMP	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
CoAP server			<input type="checkbox"/>	<input type="checkbox"/>
CoAP client			<input type="checkbox"/>	<input type="checkbox"/>
DTLS			<input type="checkbox"/>	<input type="checkbox"/>
MQTT client			<input type="checkbox"/>	<input type="checkbox"/>
Mutual SSL authentication			<input type="checkbox"/>	<input type="checkbox"/>
IPv6 router advertisement	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
DNS client v6	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
SSDP	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Applications/utilities				
Command line shell	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
OTA firmware upgrade	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Ping	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Fail-safe firmware upgrade	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Qonstruct tunables	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Secure storage			<input type="checkbox"/>	<input type="checkbox"/>
Demos				
Concurrent demo SCC (SoftAP +STA)	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Concurrent demo MCC (SoftAP + STA)	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Hostless UART	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
AllJoyn demo	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
AWS IoT demo			<input type="radio"/>	<input type="radio"/>
Bluetooth and Mesh demo			<input type="checkbox"/>	<input type="checkbox"/>
Software overlay demo				<input type="checkbox"/>
Wi-Fi features				
2.4 GHz HT20 STA	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5 GHz HT20 STA				
STBC, LDPC, MLD, MRC, Short GI	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
AMPDU Rx and Tx aggregation	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
AMSDU Rx	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Regulatory	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Antenna diversity				
Dynamic Rx diversity (per packet)	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

	TX.1.1	TX.2.0	TX.2.1	TX.2.2
Static Rx diversity/Tx diversity	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Low power				
Store-recall (suspend-resume)	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Raw mode	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Low power listen	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Green Tx	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Rate reporting	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Early Beacon Termination	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Security				
Open authentication	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
WEP	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
WPA/WPA2 PSK	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
WPS/WPS 2.0	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
SoftAP mode				
Soft AP	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Hidden SSID	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Soft AP: Multiple STAs	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Standards				
802.11b	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
802.11d	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
802.11g	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
802.11h (radar)	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
802.11n	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Certifications				
Wi-Fi 802.11n STA	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
WPS2.0	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Secure Boot	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Security Enhancement	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
IoT ecosystems				
Weave/libiota v1.1		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
AWS IoT v1.1			<input type="radio"/>	<input type="radio"/>
Encoders/Parsers				
ezXML parser	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
JSON parser	<input type="radio"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Bluetooth 4.1 (BLE Only)				
ANP		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ANS		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
BAS		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
CTS		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
DIS		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
FMP		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
HTP		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

	TX.1.1	TX.2.0	TX.2.1	TX.2.2
HIDS		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
HOGP		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
TIP		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
TPS		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
HRP		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
HRS		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
PXP		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
WLAN/BT Coex				
Coex		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Chip support				
QCA4010	○	○ <input type="checkbox"/>	○ <input type="checkbox"/>	○ <input type="checkbox"/>
CSR8811		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

3 Supported Hardware

3.1 Development kit

The QCA4010.TX.2.2 hostless SDK is supported on RB01/RB02 and RB01/RB04 platforms.

Table 3-2 Boards and modules for RB01/RB02 and RB01/RB04 platforms

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.
RB04	20-YA358-H1	The RB04 module uses the QCA4010 1 x 1 single-band 802.11b/g/n SoC and CSR8811 for Bluetooth. The RB04 module supports 2.4 GHz operation only with integrated balun solution. The 20-YA358-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.

4.3 QCA4010.TX.2.1 CS limitations

On RB01/RB04, if BLE is initialized and the device is “reset” via gdb, BT initialization may occasionally fail on reboot. When this occurs, reboot the device with a full power cycle.

4.4 QCA4010.TX.2.2 CS limitations

On RB01/RB04, if BLE is initialized and the device is “reset” via gdb, BT initialization may occasionally fail on reboot. When this occurs, reboot the device with a full power cycle.

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 disconnection 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 buffer pool size
 - New API to control timeouts for TCP connection setup, SSL connection setup and Wi-Fi scan
 - Added dual-band support
 - Added support for RB01/RB04 HW
 - Added support for BT 4.1 (BLE)
 - Added support for WLAN/BT Coex
 - Added support for accuracy of 15.625 ms timer and socket function processing.
 - Provide mechanism to reuse a certificate/ca list from an existing SSL context. This avoids using multiple copies of the same certificate/ca list in memory.
 - Make HTTP client message body size configurable with max length as 10240 bytes.
- **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.
- **I2S:** Fixed the issue that there is no voice from headphone when testing speak function on port0.

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.
 - The DUT hangs up when connecting AP, and then disconnecting AP and launching ftp_ota upgrade. Enabling AP aggregation can fail in FTP upgrading with ERR1003.
- **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:**
 - 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.
 - DUT cannot establish HTTP connection as a HTTP client after repeatedly performing "HTTP/HTTPS connect-get-post-disconnect" about 600 times.
- **Weave:** QCA4010 SDK hits exception after running 25+ hours with Weave.
- **SnS MTBF:**
 - The DUT occasionally hangs with benchquit after receiving TCP or UDP traffic without epc info.
 - When the DUT works in SAP mode on the 5 GHz channel, it disconnects/reconnects frequently during overnight stress test.
- **Multiple UART:** UART data loss is > 8%. Verification failed on UART0 (115200) + UART2 (115200). UART data loss rate > 1.3% in power save, and no data loss in max performance.
- **BT BLE:** The DUT quickly continuously scans device when shaking remote device.
- **BT IOT:** BLE disconnections were occasionally noticed with some off-the-shelf bands. BLE connection setup occasionally takes long time (28s+) with some phone models.
- **Coex:** BLE connection drops with CI=100ms when running BLE loopback data and UDP Rx traffic simultaneously. The DUT cannot work on dual mode (central and peripheral). BLE connection dropped after Wi-Fi suspend 10s; the DUT hangs after running TCP Rx stress with BLE connection. Fail to update connect interval in ref device when the DUT works as central mode.
- **Coex throughput:** Under WLAN 11b traffic and BLE traffic mode, Wi-Fi Rx throughput value & BLE throughput value is low (WIFI Rx < 1Mbps & BLE < 16kbps); UDP Rx traffic is low (about 100kbps) when setting BLE connect interval=7.5ms.
- **Performance:** RvR TCP uplink test result is not good on 11NG mode. Down link ping is failed and reconnected to AP when testing MAC Fragmentation.
- **Performance:** When MAC fragmentation occurs, downlink ping fails and DUT gets reconnected to AP.
- **Coex:** Packet loss may be observed in IrDA in presence of BLE loopback traffic.

- **I2S:**
 - There is too much noise when recording audio file in port0. This issue is not seen in port1.
 - Player with serious noise when Audio speak in I2S0, and I2S1 works well. I2S audio play causes DUT crash when inputting a non-existent audio file to **ruby_play.exe**.
- **Store-recall:**
 - The DUT gets Assertion CRASH of “pc=0x94ba63” after store-recall and disconnecting DB120 AP under 5 GHz, passed with other APs.
 - When running store recall, DUT delays about 40s to wake up.

5.5 QCA4010.TX.2.1 fixed issues/new features

- **New features:**
 - Implemented two-way cert authentication in SSL on QCA4010/QCA4012.
 - Reuse of certificate or CA list across multiple SSL contexts is allowed.
 - New ART2 command is added to support flash commands such as querying flash ID.
 - Added support for CoAP server and CoAP client. Four concurrent CoAP clients are supported. The CoAP server supports five connections with exception that only one connection is supported over DTLS. CoAP is supported over both UDP and TCP. CoAP works with both TLS and DTLS connections.
 - Added support for external antenna diversity.
 - Added support for new TLS ciphers:
 - TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8
 - TLS_DHE_RSA_WITH_AES_128_CBC_SHA
 - TLS_DHE_RSA_WITH_AES_256_CBC_SHA
 - Added support for stack protection.
 - Added APIs to support upgrading image via the HTTPS web server.
 - Added support for AWS IoT SDK 1.1.
 - Added support for the MQTT client.
- **Fixed issues:**
 - **HTTP server:** HTTP PUT method to create Wi-Fi profile object fails, but updates are OK.
 - **HTTP client:**
 - After HTTPS client fails to get html once and times out, DUT fails to send SSL packet the second time.
 - DUT cannot establish HTTP connection as a HTTP client after repeatedly performing "HTTP/HTTPS connect-get-post-disconnect" about 600 times.

5.6 QCA4010.TX.2.1 known issues

- **Store-recall:** When running store recall, DUT delays about 40s to wake up. This issue occurs only when the store recall duration is > 15 mins.
- **MQTT client:** The DUT as a MQTT client may time out and exit while processing Publish/PubRec message. A fix for this issue can be provided via the Support site.
- **SSL/DTLS:** The DUT as TLS or DTLS server cannot sometimes establish connection with OpenSSL client using ECC certification.

5.7 QCA4010.TX.2.2 fixed issues/new features

- **New features:**
 - Added support for memory overlays. Different modules can be loaded into same memory region reserved for largest module.
 - Added support for the Server Name Indication (SNI).
 - Added support to periodically sync SNTP client to SNTP.
 - Increased the host name size for HTTP connections from 40 to 256 bytes.
 - Added support for a higher accuracy timer that gets invoked from an ISR context.
- **Fixed issues:**
 - **MQTT client:** The DUT as a MQTT client may time out and exit while processing Publish/PubRec message. A fix for this issue can be provided via the Support site.
 - **SSL/DTLS:** The DUT as TLS or DTLS server cannot sometimes establish connection with OpenSSL client using ECC certification.

5.8 QCA4010.TX.2.2 known issues

- **Store-recall:** When running store recall, DUT delays about 40s to wake up. This issue occurs only when the store recall duration is > 15 mins.
- **UART FW Upgrade:** In UART FW Upgrade mode, upgrade and degrade fails when using ART2 to burn the image. A fix for this issue can be made available through the Support site.
- **Wi-Fi + BLE (Mesh) coexistence**
 - WLAN as SoftAP may get disconnected with error code 4 when running peak WLAN + Mesh + GATT + LOT traffic.
 - BLE may get disconnected during BLE + Wi-Fi (uplink) + MESH traffic when BT priority = 0. With recommended setting of BT priority = 2, this issue is not observed.
 - On one occasion, the DUT was not responsive when running Mesh + LOT + peak TCP traffic in WLAN SAP mode.

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.2.2 release consists of proprietary software by Qualcomm Technologies, Inc. (QTI) 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-YA116-21 (this document)	QCA4010.TX.2.2 Hostless SDK Release Notes
80-YA116-22	QCA4010/QCA4012 Hostless SDK (QCA4010.TX.2.2) 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.2.2 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 Explorer.
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 Explorer 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 Explorer 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 Explorer 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 complete, 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.2.2) 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.

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

- To use `sdk_shell` application with Bluetooth:

Go to **target/demo/sdk_shell** and build the IOT demo.

```
cd target/demo/sdk_shell
make clean all
make -f Makefile.bt
```

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

- **bt_demo.out**: This image is a demonstration with OTA, HTTP, SSL, DNS, Bluetooth, 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 RWDATASET_MAXSIZE 0xf000 >> 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.

- To build an RB04 bt_demo image:

```
cp tunable_input_mp2_hostless_4bitflash.txt tunable_input.txt
echo OP_FLAVOR \$OP_FLAVOR >> tunable_input.txt
echo BAND 1 >> tunable_input.txt
echo RWDATASET_MAXSIZE 0xf000 >> tunable_input.txt
echo "export FLASH_FLAVOR=\"_IOT_MP2\"" >> tunable_input.txt
echo "export BD_FILENAME=\$SDK_ROOT/bddata/boardData_2_1_MP2_1124.bin"
>> tunable_input.txt
```

NOTE: For RB04 board, only single band is supported, so BAND must be set to 1 only. FLASH_FLAVOR can be set to differentiate the names for RB04 image. When the OP_FLAVOR is selected correctly for RB04, the bt_demo.out will be used when compiling the RB04 image. Check **qonstruct.sh** for reference. For detailed information about bt_demo with CSRmesh, refer to *Building CSRmesh with QCA4010/QCA4012 Application Note (80-YA675-3)*.

- c. Modify the following values in the tunable_input.txt file.

Line	Name	Old Value	New Value
88	GPIO6_ACTIVE_CONFIG	0x90007800	0x80007800
92	GPIO10_ACTIVE_CONFIG	0x90007808	0x80007808
94	GPIO12_ACTIVE_CONFIG	0x90007808	0x80007808
95	GPIO13_ACTIVE_CONFIG	0x90007808	0x80007808
103	GPIO21_ACTIVE_CONFIG	0xd0000000	0x80000000
133	GPIO6_INACTIVE_CONFIG	0x90007800	0x80007800
137	GPIO10_INACTIVE_CONFIG	0x90007808	0x80007808
139	GPIO12_INACTIVE_CONFIG	0x90007808	0x80007808
140	GPIO13_INACTIVE_CONFIG	0x90007808	0x80007808
148	GPIO21_INACTIVE_CONFIG	0xd0000000	0x80000000

- d. Run Qonstruct.

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

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

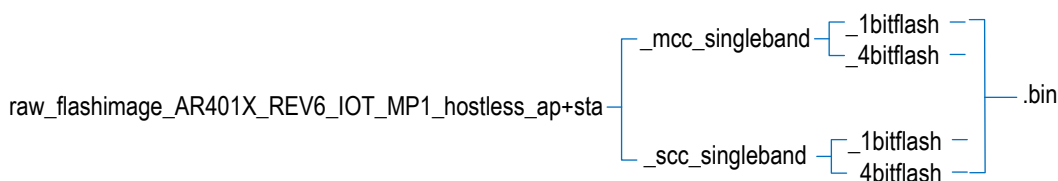
unidev	1 device
---------------	----------

ap+sta	AP+STA concurrency
mcc	Multi-channel concurrency
scc	Single channel concurrency
uart_at	Hostless UART
singleband	2.4 GHz only
1bitflash	Single mode SPI flash access
4bitflash	Quad mode SPI flash access

- f. (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 in 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/target/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 or RB01/RB04 platform

Users can use either JTAG (with OCD Daemon) or USB (with ART tool) to flash the firmware images to the RB02/RB04 module.

After the firmware image is downloaded to the RB02/RB04 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 (for RB02), JP3.2&3 (for RB04), JP11.2&1(Test mode), JP5.2&3(IOT mode), JP10.2&3(HOST0), JP9.2&1(HOST1) to make RB02/RB04 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
```

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.2.2) User Guide* for demo usages.

6.5.1 Board setup and console connection with RB01/RB02 or RB01/RB04

1. Use a USB to RS232 adaptor cable to connect the RB01/RB02 or RB01/RB04 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:

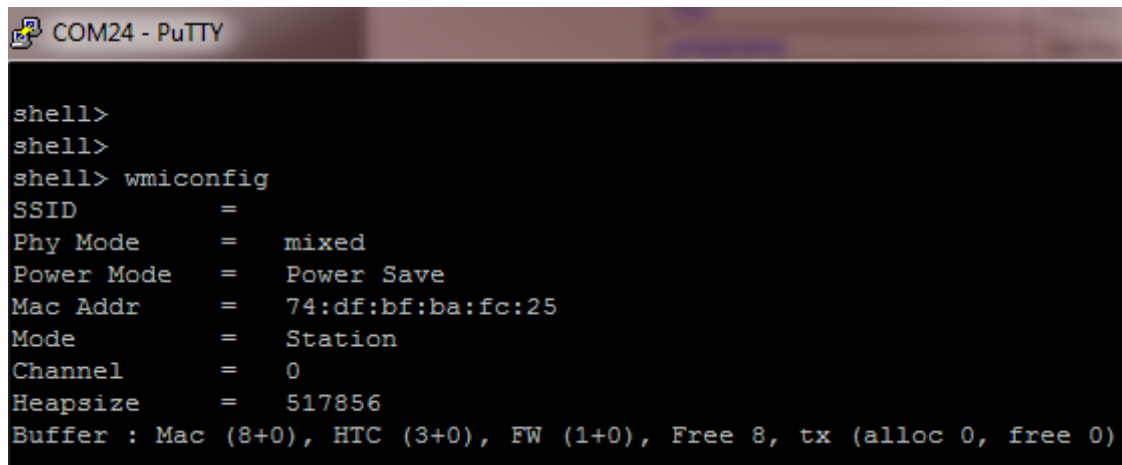


```
COM24 - PuTTY
shell>
shell>
shell>
shell>
shell>
```

6.5.2 WMI commands

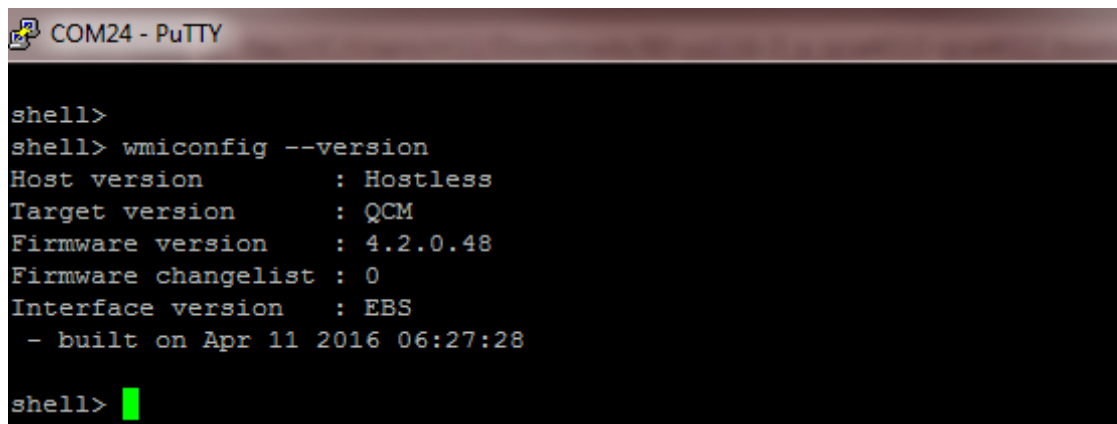
You can use the `wmiconfig --help` command to view the syntax of available commands. The following screenshots demonstrate the output of a few WMI commands.

wmiconfig



```
COM24 - PuTTY
shell>
shell>
shell> wmiconfig
SSID =
Phy Mode = mixed
Power Mode = Power Save
Mac Addr = 74:df:bf:ba:fc:25
Mode = Station
Channel = 0
Heapsize = 517856
Buffer : Mac (8+0), HTC (3+0), FW (1+0), Free 8, tx (alloc 0, free 0)
```

wmiconfig --version

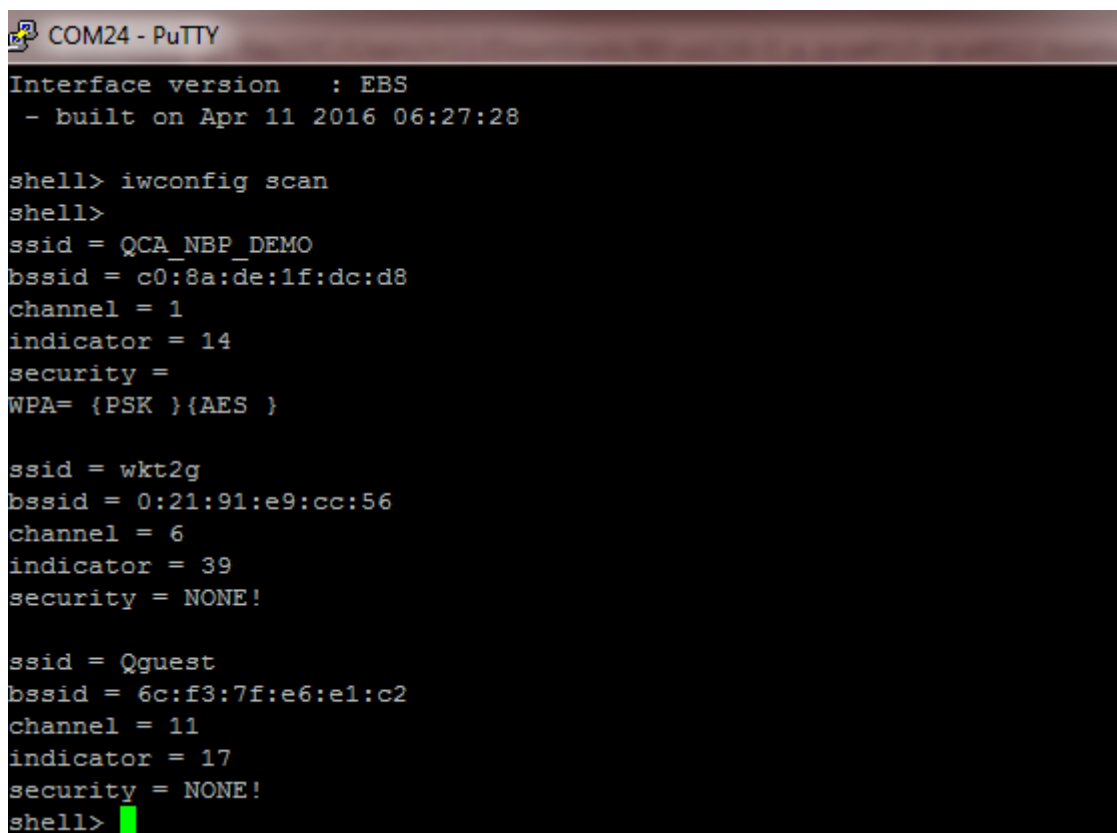


```
COM24 - PuTTY

shell>
shell> wmiconfig --version
Host version      : Hostless
Target version    : QCM
Firmware version  : 4.2.0.48
Firmware changelist : 0
Interface version : EBS
- built on Apr 11 2016 06:27:28

shell>
```

iwconfig scan



```
COM24 - PuTTY

Interface version : EBS
- built on Apr 11 2016 06:27:28

shell> iwconfig scan
shell>
ssid = QCA_NBP_DEMO
bssid = c0:8a:de:1f:dc:d8
channel = 1
indicator = 14
security =
WPA= {PSK }{AES }

ssid = wkt2g
bssid = 0:21:91:e9:cc:56
channel = 6
indicator = 39
security = NONE!

ssid = Qguest
bssid = 6c:f3:7f:e6:e1:c2
channel = 11
indicator = 17
security = NONE!
shell>
```

The following example commands demonstrate how to associate RB01/RB02 to an Access Point (AP) and to obtain IP address. In this example, the AP's SSID is "iot", the passphrase is "12345678", and the security protocol is WPA2.

```
wmiconfig --p 12345678
wmiconfig --wpa 2 CCMP CCMP
```

```
wmiconfig --connect iot
wmiconfig --ipdhcp
wmiconfig --ipconfig
ping <Access Point IP address>
```

6.6 Example Applications

This section introduces how to try the following two example applications:

- Simple DNS server
- DNS client

6.6.1 Simple DNS Server

DNS Server translates domain names to IP addresses and each domain is managed by a DNS server. A DHCP server assigns the TCP/IP address to a DHCP client. The DHCP client advertises its HOST name as part of DHCP exchange. DNS Server associates the TCP/IP address assigned by DHCP Server to a client with the HOST name advertised by the DHCP client. DHCP Server and DNS Server interact to create the database.

In this example application, the simple DNS server is enabled only in Soft AP mode.

Test the simple DNS Server

To test the simple DNS server, prepare the following components:

- The simple DNS server, which is a Development Kit configured to the Soft AP mode.
- Two clients: STA-A and STA-B. The clients can be either Development Kit or Windows PC.

Follow the steps below to set up the environment and test the simple DNS server:

1. Execute the following command to configure the Development Kit to the Soft AP mode:

```
wmiconfig --mode ap
wmiconfig --channel 6
wmiconfig --connect ioeap
wmiconfig --ip_dns_local_domain <domain_name>
```

2. Configure the HOST name on the two clients, which will associate to the Soft AP.

It is recommended to configure the HOST name before acquiring IP address. On a Development Kit client, use the following commands:

```
wmiconfig --iphostname <host_name>
wmiconfig --ipdhcp
```

On a Windows PC, use the `ipconfig` command to display the configured IP address and HOST name configured on STA-A.

NOTE: Make sure Development Kit clients use unique MAC addresses.

3. From STA-A, ping STA-B using the host name.
At this time, STA-A sends a DNS query to the Soft AP to learn the IP address of STA-B.

The application on STA-A can now interact with the application on STA-B through the Soft AP by hostname. The **custom_ip_resolve_hostname** API can also be used to learn the IP address of the peer by domain name.

6.6.2 DNS Client

DNS Client resolves and caches domain DNS names from the Domain Name System (DNS). DNS Client queries a configured DNS Server for the IP address if a requested DNS name does not exist in its cache. When DNS Client receives the requested address, it stores the name and address in its cache for future requests. The DNS Client service is enabled by default.

DHCP (if enabled) can configure one of the DNS servers. Up to three DNS servers can be configured manually. The throughput demo application has an enhanced ping command to work with the DNS Client service.

WMI commands can add and delete the DNS server, and can resolve hostname for IPv4 and IPv6.

Test DNS Client

Follow the steps below to configure and test DNS Client:

1. Add IP address of the DNS Server:

```
wmiconfig --ip_dns_server_addr <ipaddr>
```

2. Resolve domain names by using the following commands:

```
wmiconfig --ip_resolve_hostname [<host_name> <domain_type>]
```

```
wmiconfig --ip_gethostbyname [<host_name>]
```

```
wmiconfig --ip_gethostbyname2 [<host_name> <domain_type>]
```

3. Ping the host:

```
ping <host_name>
```

The `<domain_type>` parameter specifies the IP version of the requested address. Use one of the following values:

- **AF_INET**: The DNS Client tries to get an IPv4 address for a given name by sending DNS Query of type `DNS_TYPE_IPADDR`.
- **AF_INET6**: The DNS Client tries to get an IPv6 address for a given name by sending DNS Query of type `DNS_TYPE_AAAA`.

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 data
	1	SPI master/I2C master	MOSI/I2C_SDA
	2	SPI master/I2C master	MOSI/I2C_SDA
	3	GPIO	SDIO data
	4	SPI master	MOSI
	6	-	ST RF Active
	7	Debug	OBS[1]
	8	Keypad	OUT[1]
	GPIO[1]	GPIO configuration	Interface
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
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]

Pin name	Description		
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]
	GPIO[10]	GPIO configuration	Interface
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]	

Pin name	Description		
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
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
0xB-0xF	GPIO	GPIO	
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]	

Pin name	Description		
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]
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]
	8	I2C Master	I2C_SCK
GPIO[19]	GPIO configuration	Interface	Signal
	0	GPIO	GPIO[19]
	1-5	I2S	WS
	6	I2S	MCLK
	7	Debug	OBS_BUS[1]

Pin name	Description		
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]
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]

Pin name	Description		
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]
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]

Pin name	Description		
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]
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

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