1 Introduction

This User manual describes how to use the Radio Control Console application.
This application allows the user to issue commands to, and retrieve information from, the Qorvo chip running test firmware from a PC console.
The specification of the supported test modes is outside the scope of this manual. They are handles in the documentation of test component software and documentation package.

Table of Contents

2 System Overview

The Radio Console Application interfaces with the Qorvo Test Component Driver DLL. This driver connects to the Software. This Test Component software itself consists of an test component Extension DLL running on the test PC and test component firmware part running on the Device Under Test (DUT) (see 3.2.1.1).

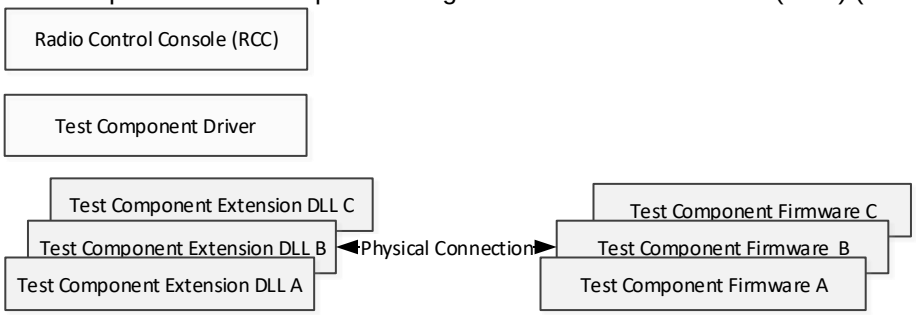

Figure 1: Software Components

When starting the Radio Console Application, the software locates the Test component dll and sets up a connection with the test component firmware running on the DUT.
The physical link the test component dll uses to interface with the DUT is configurable. A UART, Ethernet and ADB over USB are available [1].

3 Software Installation

This chapter describes the Radio Control Console installation and configuration procedure.

3.1 Pre-requisites

3.1.1 .NET Framework

The application requires .NET runtime framework 4.5.1 or higher. A check for the installed version is done when starting the application. If required, download .NET framework 4.5.1 or later from Microsoft (or Microsoft .NET Downloads).

3.1.2 Physical connection between DUT and PC in place

3.1.2.1 UART

A UART link in-between the DUT running the PTC firmware and PC running the Console Application needs to be in place. The UART IO mapping of the DUT is defined in the release notes of the Product Test Software.

3.1.2.2 Ethernet

An ethernet link is also possible and supported by the RadioControlConsole by using the ***-e <<hostname or ipaddress>>*** argument upon startup.

3.1.2.3 ADB

The radio console can also interface with test component firmware via an Android Debug Bridge. The radioControlConsole is to be called in combination with the **–as=<<adb shell reference>> --ap=<<test component elf>>** arguments.

3.2 Testcomponent DLL

RadioControlConsole uses TestComponent libraries to connect and configure the target DUT. At the time of writing there are 2 defined TestComponent types:

1. **PTC** = Product Test Component

   PTC allows to configure the target machine into a certain state. The commands from PTC configure

[1] Test component release notes list the supported physical links. Not all packages support all links.

the radio chip in test modes enabling measuring the RF performance during design validation, certification or production test.

2. **CTC** = Coexistence Test Component

   CTC allows to set the coexistence configuration on the target DUT. The commands from CTC will configure the Coexistence settings.

These DLLs define the allowed configuration and its values. It also contains the required version and product identification, so it must map 1-to-1 to the target firmware.
Using the <<H>> command in the RadioControlConsole you can find out what the defined attributes and modes are for that target, together with its supported values.

3.2.1 PTC

A PTC DLL with matching product identification field and version number as the one programmed on the DUT should be available. Note PTC DLL's are distributed together with PTC firmware files.

3.2.1.1 PTC Firmware

The DUT needs to run PTC firmware. Note each PTC firmware variant can be identified with a unique combination of an identification field and version number.

3.2.1.2 PTC DLLs

The RadioControlConsole.exe application requires the PTC DLLs to be able to communicate to the PTC firmware. The default location where the application will search for the DLLs

1. An **Extensions** folder next to the executable.
2. If that folder does not exist, the application will go look in the parent folder of the current folder for the **Extensions** folder.

It is possible to override this folder by starting up the executable with the **-f** or **--extensionsfolder=** parameter (see RadioControlConsole.exe -h for more information).

If the application is unable to find any compatible DLLs in the specified folder, a new window will be shown in which you can select the correct folder. Note that this will be shown every time unless you copy the DLLs inside that folder to the **Extensions** folder or startup the application with the correct folder set as a parameter.

3.2.2 CTC

It should be noted that the user cannot connect to the DUT using CTC and PTC using 1 RadioControlConsole!

It is possible to connect with two RadioControlConsoles to the same target DUT if the physical connection allows this. E.g. when using ethernet you can make a connection to PTC on port A and a connection to CTC on port B.
In order to make sure that the RCC application is using the correct library, it is required to add the ***–coex*** argument upon execution of the RCC application.

```
RadioControlConsole.exe --coex
```

CTC extension DLL can reside in the same folder as PTC DLLs. The Coex argument will make sure that the application only looks for the CTC DLLs.

CTC works similar as PTC, except for the naming difference and commands.

4 Application execution

There are several possible command line arguments you can give to the application. Running

```
RadioConsoleControl.exe -h
```

will give more information on that.

When the application is started without a connection argument (see Connection chapter), the window in Figure 2 and Figure 3 is shown, allowing you to switch between a Serial connection or Ethernet connection. The user can fill in the required settings.

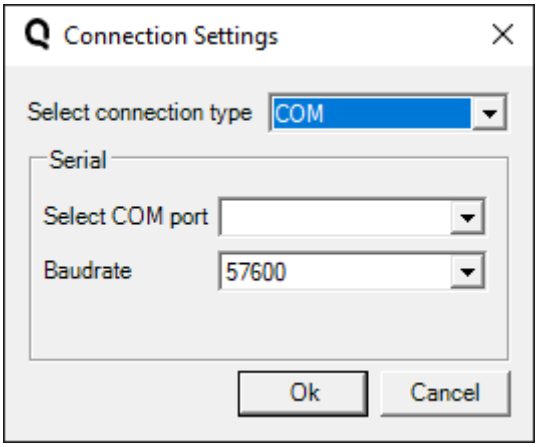

Figure 2: COM port settings

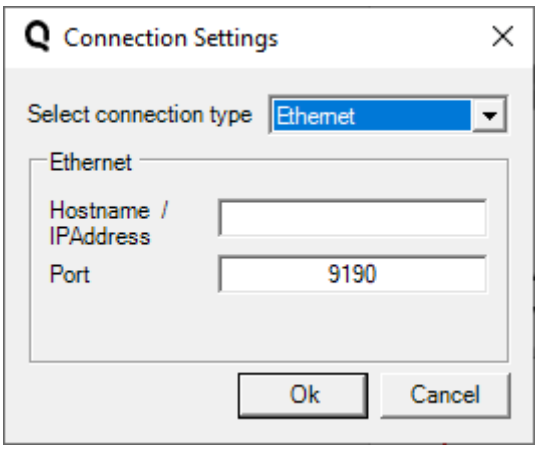

Figure 3: Ethernet settings

For the ethernet connection, the user can specify a hostname or IP address.
When clicking ok, the connection will be made based on the selected connection type and settings for that type.

If the connection is successfully established, the identification procedure will start.

5 Connection

5.1 Serial

Default the COM port selection window will be shown. To avoid the showing of the separate window, the user can specify the settings as command line arguments :

```
RadioControlConsole.exe -c COM1 -b 57600
```

5.2 Ethernet

To connect to a DUT that has an Ethernet connection on which the test component firmware is running :

```
RadioControlConsole.exe -e gateway1234 -p 9190
```

This specifies the hostname or ipaddress and the port which the test component has defined to use.

5.3 ADB Support

There is also provision to connect to ADB enabled platforms. This will require the ADB shell to be installed on the PC running the RadioControlConsole.
Install ADB from :

```
https://dl.google.com/android/repository/platform-tools_r26.0.0-windows.zip
```

in a separate folder on the PC : e.g. C:\{adb_folder}.
Make sure that this ADB connectivity is functional :
Go to : C:\{adb_folder}, open a CMD window and run 'adb devices'. This should show the connected ADB device.

To Connect to the ADB device using RadioControlConsole :

```
RadioControlConsole.exe --as=C:\{adb_folder}\adb.exe
--ap=/path/to/TestComps_firmware/in/DUT/TestCompsFirmware.elf
```

6 Identification procedure

The firmware that is programmed on the DUT requires a unique PTC DLL that contains the supported modes and attributes. The RadioControlConsole has a mechanism that searches for the matching dll.
First the application will take a random DLL to get the target identification. This will return a version of the target firmware together with a defined product name (*not* the filename: renaming the file does not change this). This version and product name are also defined *inside* the DLL and this combination must match with the target in order for it to be valid for use. So it is possible to have the same versions but different product name DLL and vice versa.
This procedure involves no user interaction.

Once the identification procedure is finished and a valid DLL has been found, the application is ready for use.

7 Command execution

After the connection and identification procedure have been completed successfully, it is now possible to start entering commands. All attribute and mode commands are dependent upon the loaded DLL. To get an overview of what commands are supported on the loaded DLL (and thus the programmed firmware) you can request "HELP" on the console.
Here we will give on overview of the general actions.

7.1 HELP

Running the HELP command on the console will show the supported modes and attributes together with its allowed values.
E.g. :

```
ATTRIBUTES
MODE_ATTRIBUTE_A : set attribute A
---→ Supported values :
A1,A2
MODE_ATTRIBUTE_B : set attribute B
---→ Supported values :
B1,B2
```

```
MODES
==========
 MODE : Set a mode on/off
```

```
     --→ MODE_ATTRIBUTE_A, MODE_ATTRIBUTE_B
```

MODE can have values on/off or just a single execution. For more details on this, please refer to the PTC/CTC Specific user manuals [5][6].
The mode command can be executed in the following ways :

```
MODE A1 B1 ON
```

Or

```
MODE_ATTRIBUTE_A A1
MODE_ATTRIBUTE_B B1
MODE ON
```

The first command will actually execute the latter sequence.

7.2 Execute a script

The user can define a sequence of commands to be executed in a separate text file. This file can be loaded and executed by the RadioControlConsole application. The script consists of the commands as the user would enter them, each command on a separate line. # can be used to comment lines.
To execute the script, the user can:

1. Use the command line argument upon execution:

```
RadioControlConsole.exe -script TEST1.txt
```

2. Use a command in the console:

```
RUN TEST1.txt
```

If the last command in the script is EXIT, then the RadioControlConsole.exe application will be closed.

7.3 VERSION

This command will give you the version of the DLL and the acutal filename of the selected DLL.

7.4 Use User specific Configuration

The user can define in an xml file what colors to use and define a shortcut for a (set of) command(s). This file can be loaded upon startup of the application:

```
 RadioControlConsole.exe -u UserConfig.xml
```

An example xml:

```
<UserConfiguration xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
     <Colors>
            <Foreground>Yellow</Foreground>
            <Background>Black</Background>
     </Colors>
     <KeyBindings>
            <KeyBinding>
                   <Key>F1</Key>
                   <Command>CH 11</Command>
            </KeyBinding>
            <KeyBinding>
                   <Key>F2</Key>
                   <Command>CH 15</Command>
            </KeyBinding>
```

```
            <KeyBinding>
                  <Key>F3</Key>
                  <Command>CH 20</Command>
            </KeyBinding>
            <KeyBinding>
                  <Key>F4</Key>
                  <Command>CH 25</Command>
            </KeyBinding>
            <KeyBinding>
                  <Key>F5</Key>
                  <Command>CW U</Command>
                  <Command>SetCW On</Command>
            </KeyBinding>
            <KeyBinding>
                  <Key>F6</Key>
                  <Command>AN 0</Command>
            </KeyBinding>
            <KeyBinding>
                  <Key>F7</Key>
                  <Command>AN 1</Command>
            </KeyBinding>
            <KeyBinding>
                  <Key>F8</Key>
                  <Command>RX ON</Command>
            </KeyBinding>
            <KeyBinding>
                  <Key>F9</Key>
                  <Command>RX OFF</Command>
            </KeyBinding>
            <KeyBinding>
                  <Key>F10</Key>
                  <Command>SETCW OFF</Command>
            </KeyBinding>
            <KeyBinding>
                  <Key>F11</Key>
                  <Command>r</Command>
            </KeyBinding>
            <KeyBinding>
                  <Key>F12</Key>
                  <Command>p</Command>
            </KeyBinding>
      </KeyBindings>
</UserConfiguration>
```

References

[1] IEEE Standard for Low-Rate Wireless Personal Area Networks (WPANs); IEEE Std 802.15.4 – 2015

[2] Bluetooth 4.2 Specification

[3] GP_P330_AN_13550_Integrate_PTC_into_SDK.pdf

[4] PTC User Manual

[5] CTC User Manual

Abbreviations

ADB	Android Debug Bridge
CCA	Clear Channel Assessment
CLI	Command Line Interface
CSMA-CA	Carrier Sense Multiple Access with Collision Avoidance
DUT	Device Under Test
ED	Energy Detect
FCS	Frame Check Sequence
IO	Input Output
LQI	Link Quality Indication
PAN	Personal Area Network
PER	Packet Error Rate
PIP	Packet-In-Packet Resynchronization
PTC	Product Test Component
RCC	Radio Control Console
RF	Radio Frequency
RSSI	Received Signal Strength Indication
RX	Receive(r)
TX	Transmit(ter)
UART	Universal Asynchronous Receiver and Transmitter

Important Notice

Document History

Version	Date	Section	Changes
0.01	16 Nov 2016		First draft.
0.02	16 Nov 2016	4	Review
0.10	19 Jan 2017	2 4 5.2.2 6 5.2.3 7 References	Added reference to PTC firmware section Added COM port settings window Added attribute commands: - BLETESTPACKET - PACKETCOUNT / PACKETINTERVAL / PACKETLENGTH - PHY - PI - SCANCOUNT / SCANINTERVAL Added applicable modes (BLE/RF4CE) Added mode commands: - ED - INFO Grouped all mode commands in same section Added applicable modes (BLE/RF4CE) References added
0.11	20 Jan 2017		Reviewed

Version	Date	Section	Changes
0.20	14 March 2017	5	Added identification mechanism information
0.30	7-Sept 2017	9 5.2	Added customcommand Added attributes and modes for register read/write Added mapping clock on GPIO functionality
0.31	2 Nov 2017	**Error! Reference source not found.** All	Expand note on BLE statistics to include both RX and TX Update document formatting
0.4	18 Dec 2017	5	Update extension folder location and handling
1.0	9 Jan 2018	2, 3.2	Added PTC driver to system overview section Added information on PTC DLL identification and selection mechanism Released
1.1	10 Oktober 2018	6	Added attribute command : - BLEDATARATE
1.2	16 January 2020	4	Added new connection setting window