OpenThread Porting Guide

QPG7015M/GP712

Application Note

1 Introduction

OpenThread is an open source implementation of Thread networking protocols developed by Thread group. It allows 802.15.4-capable devices to build robust dynamic mesh networks.

This Application Note gives the reader an overview on how to build and use Qorvo platform support library (libQorvo) to run the OpenThread stack on compatible chips.

The document is organized as follows:

- Chapter 1 gives an overview of the document.
- Chapter 2 provides architectural overview of libQorvo and OpenThread
- Chapter 3 provides an explanation on how to build libQorvo for target platform
- Chapter 4 provides an overview and instructions needed to build OpenThread
- Chapter 5 provides a smoke-test scenario to verify integrity of OpenThread build
- Chapter 6 provides an info regarding OpenThread usage in custom apps
- Chapter 7 provides an info regarding OpenThread build verification

Since the steps required to port the list of Qorvo chips mentioned below are the same, the chip name will be addressed generically by “the Qorvo device” in the remainder of this document.

The list of compatible Qorvo devices is as follows:

- QPG7015M
- GP712

Table of Contents

2 The Qorvo OpenThread Library

This chapter gives an overview of the Qorvo specific library for OpenThread (libQorvo).

2.1 Architectural Overview

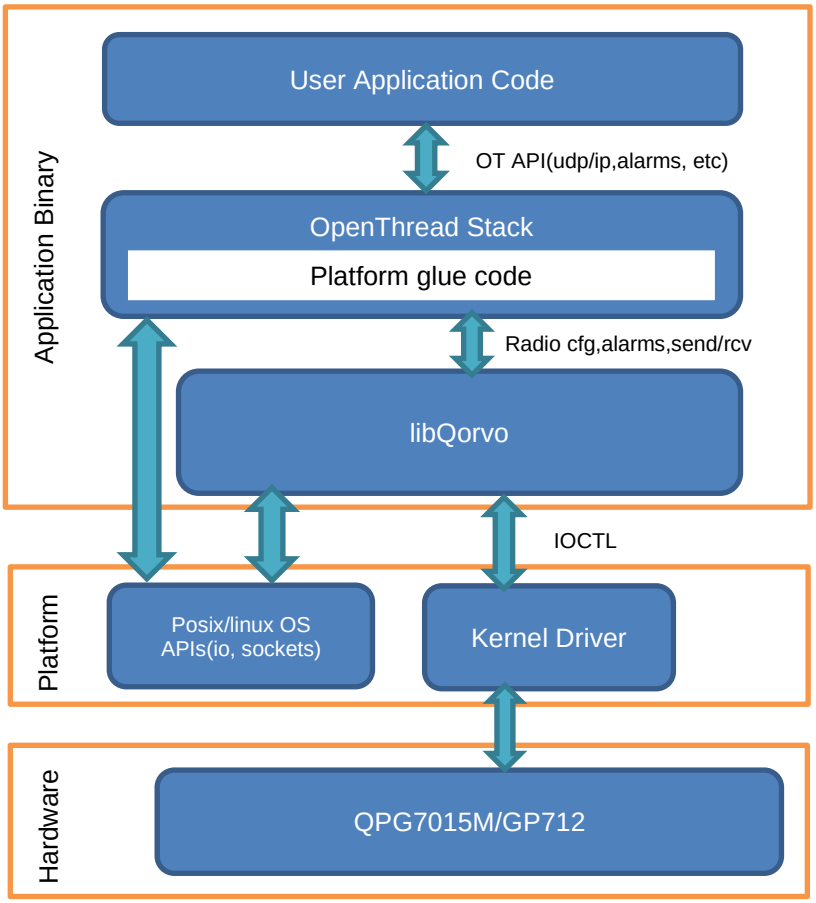

Figure 2: OpenThread Functional Scheme

The libQorvo library implements platform specific functionality and abstracts OpenThread from the Operating System (OS) and hardware. It provides timers, random and radio APIs in a way the OpenThread platform glue code can use it.

On the bottom side, it uses a Posix API to interact with OS and Kernel Driver via IOCTL to interact with radio hardware.

3 Building libQorvo for Custom Platform

The OpenThread implementation provides network stack source and platform-independent APIs to build applications on top of it. Device-specific code is separated from platform libraries such as libQorvo. The Qorvo OpenThread package provides following components:

- libQorvo sources for building on the target platform including needed components
- Makefile with buildflags
- Toolchain defines for make

3.1 Customizing Compiler Settings

Toolchain path and settings can be configured by editing `compiler_defines.mk` in `make\compilers\<toolchain>\` directory. Target compiler can be selected by editing COMPILER variable in Makefile, i.e. `COMPILER:=rpi_bcm2708` will use `make\compilers\rpi_bcm2708\compiler_defines.mk`

Basic `compiler_defines.mk` toolchain configuration includes:

Path to toolchain root:

```
TOOLCHAIN       ?=$(ROOTDIR)/gpHub/TOOL_RPi/bin/arm-gcc-4.7.1
```

Architecture name:

```
ARCH            ?=arm
```

Toolchain prefix:

```
CROSS_COMPILE   ?=$(ARCH)-bcm2708hardfp-linux-gnueabi-
```

System root definition:

```
SYSTEMROOT      ?=$(TOOLCHAIN)/arm-bcm2708hardfp-linux-gnueabi/sysroot
```

Default compiler flags:

```
FLAGS_COMPILER += -Os
FLAGS_COMPILER += -march=armv6j
FLAGS_COMPILER += -Wall
```

C compiler flags:

```
CFLAGS_COMPILER += $(FLAGS_COMPILER_FILTERED)
```

C++ compiler flags:

```
CXXFLAGS_COMPILER += $(FLAGS_COMPILER_FILTERED)
CXXFLAGS_COMPILER +=-Wno-reorder
```

Flags for assembler:

```
ASFLAGS_COMPILER+=
```

Linker options:

```
LDFLAGS_COMPILER+=--sysroot=$(SYSTEMROOT)
```

Library options:

```
LIBFLAGS_COMPILER+=
```

3.2 Building libQorvo

After the compiler is configured to build libQorvo, run make on the Makefile that is included in the source archive:

```
make -f Makefile.QorvoGP712_rpi_cli_ftd_socket
```

If everything is configured correctly, it will generate a library file at:
`Work/QorvoQPG7015M_rpi_cli_ftd_socket/libQorvoQPG7015M_rpi_cli_ftd_socket.a`
Or
`Work/QorvoGP712_rpi_cli_ftd_socket/libQorvoGP712_rpi_cli_ftd_socket.a`

4 OpenThread

For more information on OpenThread software, several guides and news can be found on
`https://openthread.io/`

4.1 Project Structure Overview

The upstream OpenThread repository can be found at:
`https://github.com/openthread/openthread`

The current tested version for Qorvo chips is based on:

```
9ea34d1e2053b6b2a80e1d46b65a6aee99fc504a
```

The upstream OpenThread Border Router repository can be found at:
`https://github.com/openthread/ot-br-posix`

The current tested version for Qorvo chips is based on:

```
a4880eb0a782adb9e6104cb00aa285d0be3a669a
```

Example applications for the Qorvo devices can be found at:
https://github.com/Qorvo/QGateway.git

OpenThread and OpenThread Border Router repositories are available as git submodules of QGateway repository together with Raspberry Pi toolchains repository.

Please check “QPG7015M instructions” and “GP712 instructions” for:

- Setting up Cross Compile Environment
- Building OpenThread RCP and CLI applications from sources
- Building OpenThread Border Router

5 Using OpenThread CLI application

5.1 Starting thread using command line

Once the OpenThread CLI app is built, you can run it on the target and test OpenThread functionality. Before starting the OpenThread app, ensure all platform drivers are loaded.

Run the CLI app:

```
./<platform>-ot-cli-ftd
```

After the app started is, there will be a command line interface available.

The list of available CLI commands can be retrieved using `help`.

The application starts in a default disconnected state. To test networking:

Set the network PanID:

```
> panid 0xcafe
Done
```

Bring the network interface up:

```
> ifconfig up
Done
```

Start the thread stack:

```
> thread start
Done
```

After that, the node will start a thread partition.

You can check the current node role using the `state` command:

```
> state
Leader
Done
```

Depending on network configuration and device role, the state may change dynamically (i.e., a device can become child, router). For more information, check the Thread protocol specification.

5.2 Commissioning another device to thread network

If OpenThread was built with commissioning functionality enabled (`COMMISSIONER=1`), you can try joining another device to the thread network.

Start the commissioner from the CLI console:

```
commissioner start
```

Add another device using the `joiner` command by specifying credentials of another device:

```
> commissioner joiner add * PASSWD
Done
```

Instead of using a wildcard in this command, you can also filter joining devices by eui64.

On the **second device**, start joining:

```
> ifconfig up
Done
> joiner start PASSWD
Done
```

Wait for successful joining message:

```
Join success!
```

And start the thread network:

```
> thread start
Done
```

Use the ipaddr command to check the IP addresses of node:

```
> ipaddr
fdde:ad00:beef:0:0:ff:fe00:fc00
fdde:ad00:beef:0:0:ff:fe00:800
fdde:ad00:beef:0:5b:3bcd:deff:7786
fe80:0:0:0:6447:6e10:cf7:ee29
Done
```

Try pinging from the **first node** to verify networking works:

```
> ping fdde:ad00:beef:0:5b:3bcd:deff:7786
8 bytes from fdde:ad00:beef:0:5b:3bcd:deff:7786: icmp_seq=1 hlim=64
time=24ms
```

6 Customizing Applications

Instead of using CLI/NCP interface, you can create custom application or/and integrate the OpenThread stack to an existing one. In OpenThread projects, the apps are located at `openthread/apps/<appname>`

Every OpenThread application needs to be linked with:

- /third_party/Qorvo/$(PLAT)/libQorvoApi.a
- /src/core/libopenthread-mtd.a (for Minimal Thread Devices) or /src/core/libopenthread-ftd.a (for Full Thread Devices) depending on target device.

7 Verification of the Port

Basic validation scenarios can be found at
`https://openthread.io/guides/porting/validate_the_port`

Abbreviations

CLI	Command Line Interface
IOCTL	Input/Output Control
NCP	Network Co-Processor
NVM	Non Volatile Memory
OT	OpenThread
SDK	Software Development Kit

Important Notice

The information contained herein is believed to be reliable; however, Qorvo makes no warranties regarding the information contained herein and assumes no responsibility or liability whatsoever for the use of the information contained herein. All information contained herein is subject to change without notice. Customers should obtain and verify the latest relevant information before placing orders for Qorvo products. The information contained herein or any use of such information does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other intellectual property rights, whether with regard to such information itself or anything described by such information. **THIS INFORMATION DOES NOT CONSTITUTE A WARRANTY WITH RESPECT TO THE PRODUCTS DESCRIBED HEREIN, AND QORVO HEREBY DISCLAIMS ANY AND ALL WARRANTIES WITH RESPECT TO SUCH PRODUCTS WHETHER EXPRESS OR IMPLIED BY LAW, COURSE OF DEALING, COURSE OF PERFORMANCE, USAGE OF TRADE OR OTHERWISE, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.** Without limiting the generality of the foregoing, Qorvo products are not warranted or authorized for use as critical components in medical, life-saving, or life-sustaining applications, or other applications where a failure would reasonably be expected to cause severe personal injury or death.

Document History

Version	Date	Section	Changes
1.00	05 Jun 2018		First version.
1.01	12 Sep 2018		Added patching info
1.02	13 Nov 2018		Updated patching info, split platform makefiles, minor fixes
1.03	14 Nov 2018		Review.
1.04	03 Apr 2019		Added QPG7015M
1.05	20 Jan 2021		Remove obsolete steps, update build configuration to include the rcp binary
1.06	13 Jul 2021		Update openthread build instructions, reference documentation in qpg-openthread GIT repo
1.07	10 Aug 2021		Added build flag for FTD builds
1.08	15 Feb 2022		OpentThread build referencing QGateway repository