

MULTI-MODE WIFI, BLUETOOTH, ZIGBEE

QCA4020: Multi-Mode Dual Band WiFi, Bluetooth 5, and ZigBee (802.15.4)



Purpose of the Document

The purpose of this document is to explain the QCA4020 which is multi-mode dual band WiFi, Bluetooth 5, and ZigBee (802.15.4). This document contains the features of the QCA4020 and how to configure it.

Document History

Version	Author	Date	Description
А	5G HUB	8.16.2021	Initial Document

Table of Contents

Purpos	se of the Document
Docum	nent History2
1	Introduction4
2	QCA4020 miniPCIe Layout5
3	QCA4020 miniPCle Pin Out6
4 4.1	How to flash Image to QCA4020 miniPCIe9 Method 1: Through Emergency Download Mode (EDL)9
4.2	Method 2: Through JTAG9
5	QCA4020 Pin Out
6	Software Tools to be Installed12
7 7.1	Setting Up the Software Development Environment12Python12
7.2	Java12
7.3	OpenOCD12
7.4	GNU ARM Toolchain13
7.5	Setup OpenOCD Plugin Usage with Eclipse13
8	Importing "Hello World" Application15
9	Building "Hello World" Application18
10	Flash the Application to the QCA4020 miniPCIe using USB21
11	Flash the Application to the QCA4020 miniPCle using JTAG23
12	Connecting QCA4020 to Serial Terminal Through UART26
13	Using Serial Terminal27
14	QCLI Demo Application

1 Introduction

This is a miniPCle card that supports multi-mode intelligent connectivity. It supports dual-band WiFi, Bluetooth 5, and ZigBee. It is based on Qualcomm QCA4020 System-On-Chip (SoC). It has low power SoC that integrates a Cortex M4F for application processing, Cortex M0 for network stack processing, and a separate processor for Wi-Fi stack designed to enable a highly concurrent multiple radio solution.

The QCA4020 SDK pre-integrates support for advanced security features and multiple software and cloud ecosystems.

Designed to address IoT fragmentation and support for interoperability, this solution is ideal for multiple IoT industries from home control and automation, networking, home entertainment and smart cities.

Feature Highlights

- Multi-mode SoC supporting dual band Wi-Fi, Bluetooth 5, and IEEE 802.15.4 concurrently
- Dedicated processor for Bluetooth LE LC and 15.4 MAC
- Dedicated processor for 802.11 a/b/g/n
- Zigbee 3.0 and OpenThread support
- Isolated power islands for low power operation
- Advanced hardware-based security featuring secure boot, trusted execution environment, encrypted storage, key provisioning and application-level security
- Comprehensive set of peripherals and interfaces: SPI, I2C, UART, HS-UART, ADC and GPIOs
- Integrated sensor hub for post-processing designed to enable low power sensor use cases
- Small package size allows for optimized form factors
- 300+KB RAM reserved for applications
- Bluetooth radio details: v5.0 with PA =+4dBm/+10dBm (for Long Range)
- 802.15.4 radio details: 2006 compliant, 15.4e, 2.4GHz DSSS +4dBm/+21dBm (for Long Range)

Specifications

Wi-Fi

Standards: 802.11a/b/g, 802.11n Wi-Fi Spectral Bands: 2.4 GHz, 5 GHz Peak Speed: 150 Mbps MIMO Configuration: 1x1 (1-stream)

Bluetooth

Bluetooth Specification Version: Bluetooth 5.0 Bluetooth Technology: Qualcomm Bluetooth mesh, Bluetooth Low Energy

802.15.4

LR-WPAN Protocol: Thread, Zigbee

USB

USB Version: USB 2.0

CPU CPU Clock Speed: Up to 128 MHz CPU Cores: Arm Cortex-M4F CPU, Arm Cortex-M0 CPU, Tensilica Xtensa

Security Support

Security Features: Application-level Security, Hardware-based Crypto Engine, Key Provisioning Security, Qualcomm[®] Trusted Execution Environment (TEE), Secure Boot, Secure Storage, Software Image Encryption, True Random Number Generator Wi-Fi Security: WPS Interface

2 QCA4020 miniPCle Layout

The following figure explains the QCA4020 miniPCIe. It has two headers P1 and P2 which exposes many of the GPIOs and interface of the QCA4020. In addition, it has a JTAG interface for debugging and flashing image. In addition, it has Emergency Download Mode (EDL) jumper header.



QCA402 M20 module
 miniPCle Interface
 J3 (EDL)
 J2 (configure JTAG interface)
 J1 (Force JTAG) 5- miniPCle socket
 P2 Header
 P1 Header
 JTAG Interface

Figure 1: QCA4020 miniPCle Layout.

The following figure shows the size and dimension of the QCA4020 miniPCIe.



Figure 2: QCA4020 miniPCIe Physical Dimension.

3 QCA4020 miniPCle Pin Out

The QCA4020 is miniPCIe card and interface. The following figure shows the pin out of the miniPCIe:



Figure 3: QCA4020 miniPCle Pin out.

To use the miniPCIe module, you can use USB adaptor (also know as Raspberry PI HAT). The miniPCIe pin mapping as in the table below. If the miniPCIe is used with he Raspberry PI HAT, miniPCIe pins are mapped to Raspberry PI for additional functionality and interface to Raspberry PI.

PIN #	QCA4020	QCA4020 GPIO PIN#	Raspberry PI PIN#
1	WAKE-UP	GPIO29_BE	
2	VCC_3V3		
3			
4			
5			
6			
7			
8			
9			
10			
11	M0&M4_UART0_RX	GPIO8_BE	GPIO 14 (8)
12			
13	M0&M4_UART0_TX	GPIO9_BE	GPIO 15 (10)
14			

Table	1.	0CA4020) miniP(^le Int	erface
Table	т.				chace.

15			
16			
17			
18			
19			
20			
21			
22	CHIP_PWD_L_BE	T4	
23			
24			
25			
26			
27			
28			
29			
30	I2C0_MASTER_SCL	GPIO10_BE	GPIO 19 (35)
31	DTR		33
32	I2C0_MASTER_SDA	GPIO11_BE	GPIO 18 (12)
33			
34			
35			
36	USB20_DM_BE	USB20_DM_BE	
37			
38	USB20_DP_BE	USB20_DP_BE	
39			
40			
41			
42	LED_ZIGBEE	GPIO13_BE	
43			
44			
45	SPI_SLAVE_CLK	GPIO18_BE	GPIO 11 (23)
46	LED_BLE	GPIO60_BE	
47	SPI_SLAVE_MOSI	GPIO23_BE	GPIO 9 (21)
48	LED_WLAN	GPIO12_BE	
49	SPI_SLAVE_MISO	GPIO20_BE	GPIO 10 (19)
50			
51	SPI_SLAVE_CS_N	GPIO19_BE	GPIO 8 (24)
52			

4 How to flash Image to QCA4020 miniPCIe

Flashing the image o the QCA4020 miniPCIe can be done using either of the following two methods:

4.1 Method 1: Through Emergency Download Mode (EDL)

In this step up, user can do the following:

- 1- Connect J3 (PIN 1 and PIN 2). This is the GPIO22_BE used for EDL.
- 2- Connect the USB cable between the PC and the USB port of M20.

4.2 Method 2: Through JTAG

In this step up, user can do the following:

- 1- Connect J1 (PIN 1 and PIN 2). This is the GPIO20_BE which used to force JTAG mode
- 2- Connect JTAG cable to the JTAG 10-PIN header which includes the signals TCK, TDI, TDO, TMS

Follow normal procedure to flash the *.elf file over the JTAG interface.

QCA4020 JTAG pins used as in the following Table:

QCA4020 PIN#	JTAG Signal
GPIO_50	JTAG3_BE_TCK
GPIO_51	JTAG3_BE_TDO
GPIO_52	JTAG3_BE_TMS
GPIO_53	JTAG3_BE_TDI

Table 2: JTAG Interface.

J2 on the QCA4020 is used to configure JTAG. Connecting J2 (PIN 1 and PIN2), force the following JTAG configuration:

Table 3	3: J2	Setting	for JTAG	Interface
---------	-------	---------	----------	-----------

Boot_Configure_B	Boot_Configure_B	Boot_Configure_BE	JTAG Interface
E_0GPIO_9_BE	E_1GPIO_25_BE	_2GPIO_18_BE	for M4
0	0	1	JTAG in GPIO[53:50]

5 QCA4020 Pin Out

The QCA4020 has abundant number of pins and GPIOs and functionalities which are exposed through P1 and P2 jumper headers. The following table summarizes all pins exposed through P1 and P2 jumper headers and their hardware functionalities.

Header and PIN	GPIO #	SPI or I2C orQSPI	UART	PWMADC/ SenseADC	РТА
P2 - PIN 1	3.3V				
P2 - PIN 2	GPIO4_BE				
P2 - PIN 3	GPIO5_BE				BT_ACTIVE
P2 - PIN 4	GPIO6_BE				WLAN_ACTIVE
P2 - PIN 5	GPIO7_BE				BT_PRIORITY
P2 - PIN 6	GPIO14_BE		HS_UART0_DM_CTS		
P2 - PIN 7	GPIO15_BE		HS_UART0_DM_TXD		
P2 - PIN 8	GPIO16_BE	I2C1_Master_SCL	HS_UART0_DM_RFR		BT_ACTIVE
P2 - PIN 9	GPIO17_BE	I2C1_Master_SDA	HS_UART0_DM_RXD		WLAN_ACTIVE
P2 - PIN 10	GPIO24_BE				
P2 - PIN 11	GPIO26_BE				
P2 - PIN 12	GPIO27_BE				

Table 4: P1 and P2 PIN Mapping.

P2 - PIN 13	GPIO28_BE			
P2 - PIN 14	GPIO30_BE			
P2 - PIN 15	GPIO31_BE			
P2 - PIN 16	GPIO32_BE			
P2 - PIN 17	GPIO33_BE			
P2 - PIN 18	GND			
P1 - PIN 10	A0 (SENSEADC_1_BE)			
P1 - PIN 9	A1 (SENSEADC_0_BE)			
P1 - PIN 8	GPIO_41_BE			
P1 – PIN 7	GPIO48_BE			
P1 - PIN 6	GPIO49_BE			
P1 - PIN 5	GPIO54_BE		SENSEADC2	
P1 - PIN 4	GPIO55_BE		SENSEADC3	
P1 - PIN 3	GPIO56_BE		SENSEADC4	
P1 - PIN 2	GPIO57_BE		SENSEADC5	
P1 - PIN 1	GPIO58_BE		SENSEADC6	

6 Software Tools to be Installed

Please install the following tools

- Python 2.7.X (Download Python | Python.org)
- Eclipse IDE for C/C++ (Eclipse IDE for C/C++ Developers | Eclipse Packages)
 This is a GUI-based integrated development environment
 Supported Version: Oxygen version Release 4.7.2
- Java:

Eclipse IDE has dependency on Java, JDK 8 or higher

- OpenOCD (<u>Download OpenOCD for Windows (gnutoolchains.com</u>)) version 0.10.0 [2017-06-09]
- **GNU Arm Embedded Toolchain** (<u>GNU ARM Toolchain</u>) version 6.x
- Qualcomm SDK for QCA4020: The SDK contains sample demo applications with source code to demonstrate different features and technologies that QCA4020 supports.

7 Setting Up the Software Development Environment

7.1 Python

After installing Python, add the path to python.

Example:

If python.exe is in the folder

C:\CRMApps\Apps\Python276-64

set path as follows:

%PATH%=%PATH%:C:\CRMApps\Apps\Python276-64

7.2 Java

After installing Java, add path to Java.

Example:

If Java.exe is in the folder

C:\ProgramData\Oracle\Java\javapath

set path as follows:

%PATH%=%PATH%: C:\Program Files\Java\jdk1.8.0_161\bin

7.3 OpenOCD

OpenOCD plugin is required to establish the connection between Eclipse IDE and onboard FTDI JTAG debugger. After installation, add the path to OpenOCD.

Example:

If openocd.exe is in the folder

C:\Program Files\OpenOCD-20170609\bin

set path as follows:

```
set %PATH%=%PATH%:C:\Program Files\OpenOCD-20170609\bin
```

7.4 GNU ARM Toolchain

Install the ARM toolchain by running the ".exe" file and make sure you select the option to "Add path to environment variables" during the final step.

7.5 Setup OpenOCD Plugin Usage with Eclipse

Do the following steps:

1- Go to **Help > Install new software** in EclipseIDE and make sure the following plugin are installed and enabled by default.



Preferences			— 🗆 X
type filter text	Available Software Sites		<
 > General > C/C++ ChangeLog > Docker > Help Install/Update Available Software Sites > Library Hover > MCU > Mylyn > Oomph > Remote Development > RPM > Run/Debug > SWTChart Extensions > Terminal > TextMate > Tracing Validation > Version Control (Team) > XML 	type filter text Name Image: State of the state o	Location http://download.eclipse.org/releases/2021-03 http://download.eclipse.org/rols/cdt/releases/10.2 https://download.eclipse.org/cols/cdt/releases/10.2 https://download.eclipse.org/releases/10.2 https://download.eclipse.org/releases/latest/ https://download.eclipse.org/linuxtools/update http://download.eclipse.org/clipse/updates/4.19 http://download.eclipse.org/tracecompass/master/repository http://download.eclipse.org/tracecompass.incubator/stable-6.3.	× Add Edit Remove Reload Enable Import Export
? 🖬 🖆 😑		Apply and	Close Cancel

2- Set path to openOCD. Restart the Eclipse IDE and under **Window -> preferences**. Set path to openOCD as seen below:

Preferences			– 🗆 X
type filter text	Global Op	enOCD Path	← → ⇒ 8
> C/C++ ChangeLog > Docker > Help > Install/Update	Configure the specifically, After installic configure the specifical specifical specifical specifical specifical specification of the s	ne location where xPack OpenOCD is installed. The values are stored within Eclipse. Unless re they are used for all projects in all workspaces. ng OpenOCD updates, restart Eclipse for the defaults to be re-evaluated and use the Restore ie new location.	defined more Defaults button to
> Library Hover	Executable:	openocd.exe	
MCU Global Arm Toolchains Paths Global Build Tools Path Global OpenOCD Path Global QEMU Path Global RISC-V Toolchains Paths Global SEGGER J-Link Path Workspace Arm Toolchains Paths Workspace QenOCD Path Workspace QEMU Path Workspace QEMU Path Workspace RISC-V Toolchains Paths Workspace SEGGER J-Link Path > Mylyn	Folder:	C:\NB_IoT_Projects\PCBandGitHub\Tools\OpenOCD-20170609-0.10.0\bin Browse	xPack
> Oomph > Remote Development	,	Restore Defa	ults Apply
? ù 4 0		Apply and Clos	e Cancel

3- After installing the Qualcomm SDK, **QCA4020 OEM SDK+CDB**, Demos samples are in the following folder:

arget\quart\demo\	

QCA4020 miniPCle

- 4- Install the QCA plugin jar file available at
- <SDK_source>/target/quartz/demo/EclipseSupportFiles
 - Copy the jar file (QCA402x_plugin.jar) to the "dropin" folder under the Eclipse IDE installed folder.
 - Restart the Eclipse IDE if running. To restart Eclipse, click on the **File->Restart** after the plug-in is installed.
- 5- Make sure all environment variable are set up correctly

Edit environment variable	×
C:\Ilsers\hossa\AnnData\Ioca\Programs\Puthon\Puthon27	New
C:\Program Files (x86)\GNU Arm Embedded Toolchain\10.2020-g4-maior\bin	INEW
C:\NB_loT_Projects\PCBandGitHub\Tools\OpenOCD-20170609-0.10.0\bin	Edit
C:\Program Files\Java\idk1.8.0 291\bin	
%USERPROFILE%\AppData\Local\Microsoft\WindowsApps	Browse
C:\Users\hossa\AppData\Local\GitHubDesktop\bin	
C:\MinGW\bin	Delete
C:\Users\hossa\AppData\Local\Programs\Microsoft VS Code\bin	
C:\Users\hossa\AppData\Roaming\npm	
	Move Up
	Move Down
	Edit text
OK	Cancel

8 Importing "Hello World" Application

Install Eclipse project files for sample demo applications.
 To install there is eclipseSupport.bat for Windows and eclipseSupport.sh for Linux in the following folder:

<SDK_source>/target

2- Open a terminal window and navigate to

<SDK_source>/target

3- Run the following command

sh eclipseSupport.sh

Or

eclipseSupport.bat

After executing the script, the Eclipse project files **.***cproject*, **.***project* and **.***settings* folder are updated in the respective folders of the demo application.

4- Open Eclipse application and Go to **File->Open Project** and choose the folder of the "Hello World" application and check the box beside the application. Click **Finish**

Import Projects from File System or Archive				– 🗆 X
Import Projects from File System or Archive				
😣 Invalid directory or archive file				
Import source: C:\QCA4020\SDK\qca4020-or-3-4_qca_	oem_sdk-cdb-master-2e23b7b0933311a33dc090ed0 f6 47de9b31d7	954\target\quartz\demo\Helloworld_demo	~	Directory Archive
type filter text				Select All
8 Folder	Import as			Deselect All
				0 of 1 selected
Close newly imported projects upon completion				Hide aiready open projects
Use installed project configurators to:				
Search for nested projects				
Detect and configure project natures				
Working sets				
Add project to working sets				New
Working sets:				✓ Select
			Sho	ow other specialized import wizards
?			< Back Next >	Finish Cancel

5- To debug the project, set the Debug Config. Go to **Run->Debug Configuration->GDB OpenOCD**. Set the Application and OpenOCD options as show in the images below. Select Quartz.elf as the C/C++ application. If you have already built the image, select Disable auto build. If image is not built, select Use workspace settings.

Debug Configurations	- 0	×
Create, manage, and run configurations	Ŕ	Š.
Image: Second	Name: Helloworld_demo Main	
Filter matched 13 of 13 items	Revert Apply	
(?)	Debug Clo	se

6- In the Debugger window, provide the openocd executable path with config option

-f \${project_loc}\build\gcc\qca402x_openocd.cfg

7- Provide the arm-none-eabi-dgb executable path for the GDB client as seen below.

Debug Configurations		— 🗆	\times
Create, manage, and run configurations		Ť,	ñ
Image: Construction of the second	Name Helloworld_demo Main % Debugger Startup Source Common Startup OpenOCD Setup Start OpenOCD locally Start OpenOCD locally Browner Executable path: C:\Tools\OpenOCD-20170609-0.10.0/bin\Openocd.exe Browner Actual executable C:\Tools\OpenOCD-20170609-0.10.0/bin\Openocd.exe Browner Cohonge it use the global or workspace preferences pages or the project properties page) GOB port: 3333 Telnet port: 6666 Config options: -f C:s\QCA4020:SDK\qca4020-or-3-4.qca_oem_sdk-cdb-master-2e23b7b0933311a33dc090ed0f647de9b31d7954\target\quartz\dem Config options: -f C:s\QCA4020:SDK\qca4022.openocd.cfg Allocate console for the telnet connection GDB Client Setup	vvse Variables	×
Filter matched 13 of 13 items	Reve	ert Apply	
0	De	bug Close	

9 Building "Hello World" Application

To build the application, do the following:

- 1. Go to Project properties->Configure build. Right-click on the project name in project explorer and go to Properties
- 2. Verify the build command and the build directory in the Builder settings tab.
- 3. Go to C/C++ build and set the build command to "build.bat", and the build directory to "path to source" as in the following screenshots.
- 4. Click Apply and Close

Properties for Helloworld_demo		— 🗆 X
type filter text	C/C++ Build	
 > Resource Builders > C/C++ Build > C/C++ General Linux Tools Path Project Natures Project References Run/Debug Settings Task Repository Task Tags > Validation WikiText 	Configuration: Default [Active] Builder Settings Behavior Refresh Policy Builder Builder Use default build command Build command Build command: build.bat Makefile generation Generate Makefiles automatically Expand Env. Variable Refs in Makefiles Build location Build location Build directory: S{workspace_loc:/Helloworld_demo/build/gcc}	V Manage Configurations Variables Variables Workspace File system Variables Restore Defaults Apply
0		Apply and Close Cancel

Properties for Helloworld_demo			
type filter text	C/C++ Build		← → ⇒ 8
 > Resource Builders > C/C++ Build > C/C++ General 	Configuration: Default [Active]	~ │ [Manag	e Configurations
Linux Tools Path Project Natures Project References	Builder Settings Behavior	Refresh Policy	
Run/Debug Settings Task Repository Task Tags > Validation WikiText	Build settings	□ Enable parallel build ◎ Use optimal jobs (12) ○ Use parallel jobs: 12 ♀ ○ Use unlimited jobs	
	Workbench Build Behavior Workbench build type:	Make build target:	
	Build on resource save (Auto buil	d) all	Variables
	Note: See Workbench automatic bui	ld preference	
	Build (Incremental build)	t 4020 cdb 2p0	Variables
	Clean	clobber	Variables
		Restore Default	ts Apply
?		Apply and Close	Cancel

5. Build the application and you will see the following message in the Console Window

eclipse-workspace - Eclipse	e IDE —		×
File Edit Source Refactor	Navigate Search Project Run Window Help		
📑 🕶 🔚 👘 🛛 😒 🖛 🔦 🕶	ਛੇ ! 💷 ! 🐂 ! 🔌 ! 📸 ▾ 🚳 ▾ 💕 ▾ 🧭 ▾ 🖉 ▾ 🚱 ▾ 🥵 ▾ ! 🎥 🗁 🖋 ▾ ! 💷 🔳 🗉		
월 ▼ 禄 ▼ ♡ ⊂? ↔ ▼		۹ 🖻	Ec
Project Explorer	🔐 Problems 🖉 Tasks 🖳 Console 🔲 Properties 🛛 🗱 🕂 🕀 😚 😫 📰 🔚 🚍 🛃 📰	📑 🝷 🖻	
> 😂 Helloworld_demo	CDT Build Console [Helloworld_demo]		
	18:51:55 **** Incremental Build of configuration Default for project Helloworld_demo ****	~	. <u>×</u>
	build.bat t 4020 cdb 2p0		1
	<pre>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>></pre>		
	Building Quartz Helloworld Demo for v2 Chipset		
	RTOS threadx		
	CHIPSET VARIANT qca4020		Ð
	Exporting Device config files		
	GENERATING DEVCFG		
	Building hello_world.c		8
	Building palypal.c		87
	Building export/platform cem som.c		
	Building export/platform_oem_mom.c		
	Building export/DALConfig_devcfg.c		
	Building exportVALConig_tom.c		
	Building export/devcfg fom data.c		
	Building export\UsrEDL.c		
	Building export/flash_init_config.c		
	Attempting to convert windows paths: output\LinkerLibs.txt		
	Linking		
	Hashing		
	18:53:16 Build Finished. 0 errors, 0 warnings. (took 1m:21s.44ms)		
		~	4

The output and ***.elf** files are generated and available under the **output** folder:

📕 🛃 📕 🖛 output					– 🗆 ×			
File Home Share	v	/iew			~ ?			
Pin to Quick Copy Paste access Clipboard	X Cu Co Pas	t py path ste shortcut Move Copy to - Organize	New folder New	Edit Dpen • History	Select all Select none Invert selection Select			
\leftarrow \rightarrow \checkmark \uparrow \blacksquare « quartz \rightarrow demo \rightarrow Helloworld_demo \rightarrow build \rightarrow gcc \rightarrow output \checkmark \eth \checkmark Search output								
★ Quick access	^	Name	Date modified 8/18/2021 7:14 PM	Type File folde	Size			
Desktop Downloads	*	LinkerLibs.txt	8/18/2021 7:14 PM	Text Doc FLE File	ument 4			
Documents	*	Quartz.ld	8/18/2021 7:15 PM	LD File	472			
DocumentsNBackup DocumentsNBackup	*	Quartz.map	8/18/2021 7:15 PM 8/18/2021 7:15 PM	Linker A ELE Eile	ddress Map 9,811 490			
PCBandGitHub	*	Quartz_HASHED.hash	8/18/2021 7:15 PM	HASH Fi	le 1			
GitHubRepository	*	Quartz_HASHED_combined_hash.mbr	8/18/2021 7:15 PM 8/18/2021 7:15 PM	MBN File HD File	e 1 1			
Quectel UofW	*	Quartz_HASHED_merged.elf	8/18/2021 7:15 PM	ELF File	486			
Docs&Fig&Video	*	Quartz_HASHED_nohash.elf Quartz_HASHED_phdr.pbn	8/18/2021 7:15 PM 8/18/2021 7:15 PM	ELF File PBN File	486			
DataSheets	*	Quartz_nocompact.elf	8/18/2021 7:15 PM	ELF File	19,428			
Invoices	* 🗸	· <			>			
13 items								

10 Flash the Application to the QCA4020 miniPCIe using USB

You can flash an application to the QCA4020 miniPCIe over the USB port. To do so, follow the following steps:

- 1- Connect a jumper on J3 to short-circuit it. This puts the QCA4020 in EDL (Emergency Download Mode) which allows to download an image to the QCA4020 flash.
- 2- Connect a Micro USB cable between USB port on the USB adaptor (HAT) and a USB port on a Windows PC.

Open the device manager and you shall see Qualcomm DLoader is enumerated on a serial port (i.e., COM40 in this case).

着 Device Manager	_	×
File Action View Help		
V 📇 LAPTOP-HOSSAMNEW		^
> 📫 Audio inputs and outputs		
 > Biometric devices > Bluetooth > Cameras > Computer > Disk drives > Display adapters > Display adapters > Firmware > Human Interface Devices > Imaging devices > Jungo Connectivity > Keyboards > Memory technology devices > Mice and other pointing devices > Monitors > Network adapters > Ports (COM & LPT) = Intel(R) Active Management Technology - SOL (COM3) = Qualcomm HS-USB QDLoader 9008 (COM40) = Print queues > Print queues 		
> E Printers		~

3- Use the python-based tool called **qflash.py** (which is included in the QCA4020 SDK). The tool allows flashing images over USB. The tool is available at the following folder:

target\build\tools\flash

If the tool is invoked without any optional parameters, it does he following:

- a) Generate a default firmware descriptor table.
- b) Generate a default partition table.
- c) Flash the default sample application ***.elf** files to the qca4020 flash.

When you run this tool with **-h** parameter, you get the following as in this screenshot:



4- Run the following command to flash the application to the QCA4020

```
python qflash.py --comm_port 40 --app C:\QCA4020\SDK\qca4020-or-3-4_qca_oem_sdk-cdb-
master-
2e23b7b0933311a33dc090ed0f647de9b31d7954\target\quartz\demo\Helloworld_demo\build\g
cc\output\Quartz_HASHED.elf
```

Noe the image to be flashed is the Quartz_HASHED.elf.

You shall see the output as in the below screen shoot which indicates the image is successfully flashed to QCA4020.



11 Flash the Application to the QCA4020 miniPCIe using JTAG

You can use JTAG interface on the QCA4020 miniPCIe to flash the image. A SEGGER J-link hardware is used wit connect and interface to the JTAG.

On the QCA4020 miniPCIe, do the following

- Connect J1 (PIN1 and PIN2). This force JTAG mode
- Connect J3 (PIN1 and PIN2). This enable JTAG interface

On Windows environment variables, add the following environment variable:

- JLINK_PATH set it to J-link GDB server executable file (e.g., C:\Program Files (x86)\SEGGER\JLink)
- CLIENT_PATH set it to ARM GNU tools (e.g., C:\Program Files (x86)\GNU Arm Embedded Toolchain\10 2020-q4-major\bin)

Using Eclipse IDE, go to **Run->External Tools->External Tools Configurations** and add a new configuration. Point to the **flash.bat** file in the SDK which is used to configure and launch GDB server. Click **Apply** and then **Run**.

External Tools Configurations		- 0	×
Create, manage, and run configurations S External tool location does not exist		©, 1	-
🗅 🖻 🎾 🗎 🗶 🖻 🏹 🗸	Name: JTAG Flashing		
type filter text	🕐 Main 🗞 Refresh 🙀 Build 🚾 Environment 🛄 Common		
✓ ♀ Program ♀ JTAG Flashing	Location: C\OCA020.SDK\ors4020-or-3-4 ora eem sdk-cdb-master-2e23b7be933311a33dc090ed0f647de9b31d7954taroet\ouart>\demo\Helloworld demo\Helloworld demo\buildtoc <filsh.bat< td=""><td></td><td></td></filsh.bat<>		
Q OpenOCD_JTAG_Flashing	Browse Workspace Browse File System	Variables	f
	Working Directory:		
	C:\QCA4020\5DK\qca4020-or-3-4_qca_oem_sdk-cdb-master-2e23b7b0933311a33dc090ed0f647de9b31d7954\target\quartz\demo\Helloworld_demo\build\gcc\		
	Browse Workspace Browse File System	Variables	
	Arguments:		
			~
	Note: Enclose an argument containing spaces using double-quotes (").	Variables	
Filter matched 3 of 3 items	Show Command Line Revert	Apply	
?	Run	Close	

You will see the following screenshot and prompted to select device or core. Choose **Cortex-M4**. Click **OK**.

roject Explorer 🖂	E	871-0	le hello.	world.c 🕄							
S Helloworld d								•/			
C:\F	Program Files (x86)\SEGGER\JLink\JLi	inkGDBServerCL.exe						- 🗆 ×			
SEGGER	t J-Link GDB Server V6.92	Command Line V	ersion					^			
JLinkA	RM.dll V6.92 (DLL compil	ed Dec 18 2020	13:12:28								
C				14441- 16				1			
G	DB Server start settings	ice onspecified	-enuran	fittie -it	JTAG -Speed 1	000 -noir -no	iocarnosco	11y -port 2331			
GDBIni				SEGGER J-Linl	k V6.92 - Target devi	ce settings				? ×	
GDB Se	rver Listening port:	2331		Colored Devices a	0117				Inde Fredera and	C #0	
SWO nar	w output listening port:	2332		Selected Device: N	2017				utue Endian •	core +0 •	
Accent	remote connection:	Ves		Manufacturer	Device	Core	NumCores	Flash Size		^	
Genera	te logfile:	off			~		✓ Filter	Filter			
Verify	/ download:			Unspecified	ARM7	ARM7	1				
Init r	regs on start:	off		Unspecified	ARM9	ARM9	1	-			
Single	run mode:	off		Unspecified	ARM11	ARM11	1	-			
Target	connection timeout:	0 ms		Unspecified	Cortex-A5	Cortex-A5	1	-		_	
	J-Link related settings-			Unspecified	Cortex-A7	Cortex-A7	1	-		_	
J-Link	Host interface:	USB		Unspecified	Cortex-49	Cortex-A8	1			_	
J-Link	script:	none		Unspecified	Cortex-A12	Cortex-A12	î			_	
J-LLIK	Target related settings-			Unspecified	Cortex-A15	Cortex-A15	1	-			
Target	device:	Unspecified		Unspecified	Cortex-A17	Cortex-A17	1	-			
Target	interface:			Unspecified	Cortex-A53	Cortex-A53	1	-		_	
Target	: interface speed:	1000kHz		Unspecified	Cortex-A57	Cortex-A57	1	-		_	
larget	endian:	little		Unspectified	Contex-A72	Cortex-A72	1	-		_	
Connec	ting to J-Link			Unspecified	Cortex-M0+	Cortex-M0	1	-		_	
J-Link	is connected.			Unspecified	Cortex-M1	Cortex-M1	1	-			
				Unspecified	Cortex-M3	Cortex-M3	1	-			
				Unspecified	Cortex-M4	Cortex-M4	1	-			
			Con:	Unspecified	Cortex-M7	Cortex-M7	1	•		_	📕 💥 🔆 🖓 🐘 🚮 🐼 😓 🛃 🐨 🛨 🖬 🕇
			JTAG Flas	Unspecified	Cortex-M23	Cortex-M23	1			_	1d7954\target\quartz\demo\Helloworld_demo\build\gcc\flash.bat (Sep 5, 2021, 2:22:42 PM)
			Could I	Unspecified	Cortex-M33	Cortex-M33	1	-		_	<pre>Wed0f647de9b31d7954\target\quartz\demo\Helloworld_demo\build\gcc\output</pre>
			Could	Unspecified	Cortex-PISS	Cortex-PISS Cortex-PIA	1	-		_	Bed8f647de9b31d7954\target\quartz\demo\Helloworld_demo\build\gcc\outpu
			[2021-6	Unspecified	Cortex-R5	Cortex-R5	1	-			Function:gen_xml_program
			[2021-4	Unspecified	Cortex-R8	Cortex-R8	1				
			Startin	Unspecified	STAR	STAR	1				
			[2021-6	Unspecified	PIC32MX	MIPS	1	•			i-gdb.exe, looks like it is not a GDB executable File:flash through gd
			[2021-6	Innerified	IDTC30M7	Mirrnántiv	11				fo). File:flash_through_gdb.py:490 Function:gdb_check_availability
			GNU gdt						OK	Cancel	
			License				· · · ·	- 01			
			This is	free software	: you are free	to change and r	redistribute	it.			
			There 1	s NO WARRANTY,	to the extent	permitted by Ia	574.				
			<								

Eclipse will start flashing the image to the QCA4020 miniPCIe over JTAG. You will see the following screenshot.

👄 eclipse-workspace - Eclipse IDE		-	0	\times
File Edit Source Refactor Navigate	ch Project Run Window Help			_
📑 • 🗟 🕼 📓 🖶 💌 🕨	■ ※ 3. ③ 2. [計 元 文] ② 操 • ◎ + ● • ● ◎ ◎ ペ • 〕 ④ ◎ + ◎ • ◎ + ◎ + ◎ + ◎ + ◎ + ◎ + ◎ + ◎ +	Q	1	祖谷
Project Explorer 😒	🖻 😘 🍸 🖁 🖓 🔲 🔯 Atelia_world.c 🖾			
> 🚰 Hellowold_demo	<pre>65 PAL_COMPOL_MATT_STRUM_LITEAA("Nello World From QCAADDB minHCle using JTAGV'n"); 74 PAL_COMPOL_MATT_STRUM_LITEAA("Nello World From QCAADDB minHCle using JTAGV'n"); 75 Sitep(1000); //Sitep for 1 sec 65 } 76 } 78 /// (print This function is used to pre initialize resource 72 @print This function is used to pre initialize resource</pre>			
Moiting pagiston	GGBPUINdUnkdDBsverCLexe – L X			8
Reading all regi	- 07.2004/02/2/ 5			B
Read 4 bytes @ a Reading all regi	55 05100462C2 (Data = 0x0801489A) 5			
Read 4 bytes @ a Reading all regi Read 4 bytes @ a Downloading 4 by Reading all regi	ss 9x10404(27) (Duta - 0x601489A) 5s 9x1044(27) (Duta - 0x601489A) 6 didress 0x1000ETD4			
Read 4 bytes @ a Downloading 4 by Reading all regi Read 4 bytes @ a	55 061004/C27 (Data = 0+K801489A) g address 0x44000088 \$5 0x1004027 (Data = 0x6801489A)			
Reading 64 bytes	ddress (9x1046114)			-
Setting breakpoi	55 0X10040140 (URTa = 0X152U) address 0X104084346, SISE = 2, BPHandle = 0x0000			
Starting target	d @ address 0x10046146			
Reading all regi				
Read 4 bytes @ a Downloading 1 by Reading all regi	a do traine (0 kt) = 0/41F0E92D) @ address 0x1000056D			~
Read 4 bytes @ a	55 0x10046146 (Data - 0x41F0E92D)			
Reading all regi	g 80/r85 0X18081E00	(Sep 5, 2021, 2:2	5:23 PM)	U
Read 4 bytes @ a Downloading 8 by Reading all regi	ss 8x1004(164 ()data = 8x41F0E52D) @ address 0x10081E88 5			^
Read 4 bytes @ a	SSE UX4004Date	ine 123.		
	<pre>Breakpoint 1, main_c () at /local/mt/workspacs/CWBuilds/DDT.QC.34.74-00081-QDM-2_20218020_023452/b/loces_proc/core/storage/tools/jtagprogrammer/src//ltagProgrammer.cll2 13 /local/mt/workspacs/CWBuilds/DDT.QC.34.74-00081-QDM-2_20218020_023452/b/loces_proc/core/storage/tools/jtagprogrammer/src//ltagProgrammer.cll2 14 /local/mt/workspacs/CWBuilds/DDT.QC.34.74-00081-QDM-2_20218020_023452/b/loces_proc/core/storage/tools/jtagprogrammer/src//ltagProgrammer.cll2 15 /local/mt/workspacs/CWBuilds/DDT.QC.34.74-00081-QDM-2_20218020_023452/b/loces_proc/core/storage/tools/jtagprogrammer/src//ltagProgrammer.cll2 15 /local/mt/workspacs/CWBuilds/DDT.QC.34.74-00081-QDM-2_20218020_023452/b/local/mt/workspacs/cDM-2_20218020_023452/b/local/mt/workspacs/cDWB-2_20218020_023452/b/local/mt/workspacs/cDWB-2_20218020_023452/b/local/mt/workspacs/cDWB-2_20218020_023452/b/local/mt/workspacs/cDWB-2_20218020_023452/b/local/mt/workspacs/cDWB-2_20218020_02 14 /local/mt/workspacs/cDWB-2_20218020_023452/b/local/mt/workspacs/cDWB-2_20218020_023452/b/local/mt/workspacs/cDWB-2_20218020_023452/b/local/mt/workspacs/cDWB-2_20218020_023452/b/local/mt/workspacs/cDWB-2_20218020_023452/b/local/mt/wo</pre>	lelloworld_de	mo\build	d∖gc
	<pre>Preadpoint 1, main_c () at /local/mt/werkspac/f000ullis/JCT (27.3.4.7.4-00000204-2_00210001_023452/b/locaw_proc/core/storage/cool//tagprogrammer/sc//JtagProgrammer.cll23 133 in /local/mt/workspac/f000ullis/JCT (27.3.4.7.400001_023452/b/locaw_proc/core/storage/cool//tagprogrammer/sc/ mtomp.loc exectable has been specified and krapt does not support attemp.loc exectable has been specified and krapt does not support 0.0244546 in 72 () 0.0000000000000000000000000000000000</pre>			ļ
	Transfer rate: 82 XB/sec, 643B bytes/write. Breakpoint 1 at 0x18046146: file /local/unt/workspace/CNUBuilds/IOT.QZ.3.4.r1-00001-QI04-2_20218201_023452/b/ioesw_proc/core/storage/tools/ttagprogrammer/src//JtagProgrammer.c, 11 	.ne 123.		*
	6			>

When flashing is completed. You will see the following screen shoot with the output:



After finishing flashing the image, make sure to remove J1 & J3 on QCA4020 miniPCle to exit JTAG mode.

12 Connecting QCA4020 to Serial Terminal Through UART

You can use the QCA4020 miniPCIe when mounted on USB adaptor (HAT) with any device that has UART (Tx/Rx) interface such as a Raspberry PI board, Arduino board or any other hardware board, with UART interface.

You can also use and connect the QCA4020 miniPCIe/USB adaptor (HAT) to a USB port in the computer through a USB-to-UART cable. You can use cable such as this one: <u>USB TO UART TTL (Wires) Serial Cable (PL2303HX) MCP00102W Programmer Arduino Compatible in</u> <u>Elecrow bazaar!</u>

Such a cable has four wires colored as follow:

Red: +5V

White<mark>: Tx</mark>

Green: Rx

Black: GND

Connect the **White** cable to **UART_TX** and **Green** cable to **UART_RX**. Also connect the **Black** cable to the **GND**).

As in the picture below, connect UART_TX to SJ2 (PIN2) AND UART_RX to SJ1 (PIN2).



Make sure when you connect the USB-to-UART cable, it shows correctly in Windows device manager and all its driver is installed as in this screenshot.



13 Using Serial Terminal

It is recommended to use Tera Term tool as the serial terminal. You can download it from here:

https://osdn.net/projects/ttssh2/downloads/54081/teraterm-4.72.exe/

Launch Tera Terminal and select the Serial option and select USB-to-Serial port.

Tera Term: New connecti	on	×
O T CP/IP	Host: myhost.example.com	1 ~
	✓ History	
	Service: O Telnet	TCP port#: 22
	• SSH SSH	version: SSH2 ~
	○ Other IP v	ersion: AUTO ~
● Serial	Port: COM31: USB Serial F	Port (COM31) v
	OK Cancel I	Help

a Term: Serial port setup and o	connection			×
Port:	COM31	~	New setting	1
Speed:	115200	~		1
Data:	8 bit	\sim	Cancel	1
Parity:	none	\sim		
Stop bits:	1 bit	\sim	Help	
Flow control:	none	\sim		
Transm 0	it delay] msec/char	0	msec/line	
Device Friendly Na Device Instance ID Device Manufactur Provider Name: FT Driver Date: 8-16-2 Driver Version: 2.1	ume: USB Seria I: FTDIBUS\VID rer: FTDI DI 017 2.28.0	ll Port († _0403+	COM31) PID_6001+7&1C8BEEF7&	

In Tera Term, choose **Setup->Serial port** and configure serial ports according to the following:

And now you are ready to see the output from "hello World" application on the serial monitor.

🔟 C(DM31 - Te	era Term	VT				—	×
File E	dit Setu	up Co	ntrol Wind	low Help				
Hello	World	from	QCA4020	miniPCIe				^
Hello	World	from	QCA4020	miniPCIe				
Hello	World	from	QCA4020	miniPCIe				
Hello	World	from	QCA4020	miniPCIe				
Hello	World	from	QCA4020	miniPCIe				
Hello	World	from	QCA4020	miniPCIe				
Hello	World	from	QCA4020	miniPCIe				
Hello	World	from	QCA4020	miniPCIe				
Hello	World	from	QCA4020	miniPCIe				
Hello	World	from	QCA4020	miniPCIe				
Hello	World	from	QCA4020	miniPCIe				
								~

If you are using JTAG mode to flash the image to the QCA4020 miniPCIe, you shall see the following output on Tera terminal

	🔟 C	OM31 - Te	era Term	VT					-	\times
F	File E	dit Setu	up Co	ntrol Wind	dow Help					
Η	ello	World	from	QCA4020	miniPCIe	using	JTAG			^
H	ello	World	from	QCA4020	miniPCIe	using	JTAG			
H	ello	World	from	QCA4020	miniPCIe	using	JTAG			
H	ello	World	from	QCA4020	miniPCIe	using	JTAG			
Η	ello	World	from	QCA4020	miniPCIe	using	JTAG			
H	ello	World	from	QCA4020	miniPCIe	using	JTAG			
H	ello	World	from	QCA4020	miniPCIe	using	JTAG			
H	ello	World	from	QCA4020	miniPCIe	using	JTAG			
H	ello	World	from	QCA4020	miniPCIe	using	JTAG			
H	ello	World	from	QCA4020	miniPCIe	using	JTAG			
										~

14 QCLI Demo Application

CLI demo application is a comprehensive demo that provides a mechanism to demonstrate different and all features and technologies that QCA4020 miniPCIe supports. It also provides reference implementation and usage of customer facing QAPIs.

The QCLI-demo application is available with the QCA4020 SDK and you can follow the same steps described before in Eclipse to build and flash it to the QCA4020 miniPCIe. Once flashed, you can see the output on the serial terminal as in the screenshot below. Also, you can exercise connecting to your home/office WLAN



The following screenshot shows how to connect to an WLAN access point.

💆 COM31 - Tera Term VT	_	×
File Edit Setup Control Window Help		
WLAN\$> Enable		^
WLAN\$> scan		
WLAN: Scan result count:0		
WLAN\$> scan		
WLAN: ssid = TPLink WLAN: bssid = b0:be:76:cb:d6:5b WLAN: indicator = 10 WLAN: security = MRAN: security = RSNAMPM2 = (FSK > (TKIP AES) WLAN: shell) Scan result count:1		
WLAN\$> SetDevice 1		
WLAN\$> SetWpaPassphrase Yusuf050209		
WLAN\$> SetWpaParameters WPA2 CCMP CCMP		
WLAN\$> Connect TPLink		
WLAN: WLAN: Setting SSID to TPLink WLAN:		
WLAN\$> WLAN: devId 1 Disconnected MAC addr 00:00:00:00:00:00		
WLAN\$> Connect TPLink		
WLAN: WLAN: Setting SSID to TPLink WLAN:		
WLAN\$> WLAN: devid 1 Disconnected MAC addr 00:00:00:00:00:00:00 WLAN: devid - 1 1 CONNECTED MAC addr b0:be:76:cb:d6:5b WLAN: 4 way handshake success for device=1 		
WLGN\$>		\mathbf{v}