Qualcomm Technologies, Inc.

CDT on Qualcomm Linux

80-70018-9 AC

April 10, 2025

Contents

1 Overview of configuration data table (CDT)

This guide is intended for developers who have full access to the proprietary software shipped with Qualcomm® Linux®.

The configuration data table (CDT) is an XML file that stores platform and device-specific information, including:

- Platform ID
- Major and minor versions
- Subtype
- OEM variant ID (OEM-ID)
- Number of key value pairs

This data is essential for various software modules, such as drivers and firmware, to dynamically detect and initialize the platform. From the above parameters, you can **customize only the OEM-ID** and not the other parameters.

CDT XML file is compiled into CDT binary and it's one of the important and mandatory binaries that need to be flashed onto UFS/eMMC partitions for the successful bootup of the device.

You can find the pre-compiled CDT binaries for different platforms at https://docs.qualcomm.com/bundle/publicresource/topics/RNO-250403001134/. All these binaries are compiled with Version 3 CDT. To differentiate customer board level changes, you can use OEM-ID. To use the OEM-ID, you need to update the CDT version to Version 5.

For detailed information about OEM-ID, see: Set up OEM variant ID

1.1 CDT general structure

A CDT consists of three primary sections:

1. The CDT header, consisting of:

- A magic number

- A version number
- Two reserved fields

2. The block metadata section
3. CDB0 - Platform ID

CDB (configuration data block) is a chunk of user-defined bytes. The following table describes the general CDT structure:

CDT component	Attribute name	Size and type	Value	Significance
CDT header	Magic number	32 bits, constant	0x43445400 (string CDT)	Magic number represents the existence of a successfully programmed CDT.
	Version number	unit16, little-endian	0x0001	CDT version: If the table format or the order of data blocks changes, the version number can be incremented to keep track of these changes. Currently, the CDT Version is 5.

CDT component	Attribute name	Size and type	Value	Significance
	Reserved	32 bits	0x0	Reserved for future use
	Reserved	32 bits	0x0	Reserved for future use
Metadata for CDB0	CDB0 offset	unit16, little-endian	Variable	Offset to the first byte of CDB0 in CDT.
	CDB0 size	unit16, little-endian	Variable	Size of CDB0 in bytes.

2 Set up OEM variant ID (OEM-ID)

You can modify the OEM-ID to differentiate user-specific platforms for Qualcomm reference device design changes. The OEM-ID supports values from 0-255.

- The value 0 is reserved for unmodified Qualcomm platforms.
- IDs 1-255 are available for use. In CDT Version 5, the OEM-ID is 8 bits.

The following tables show the format of CDT for Versions 3 and Version 5.

Table : Platform CDT format for Version 3

CDT version	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 6	Byte 7	Byte 8
03	0x03	Platform ID	Major version	Minor version	Subtype	Number of KVPs	Key 1	Value 1

Table : Platform CDT format with OEM-ID for Version 5

CDT version	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
05	0x05	Platform ID	Major version	Minor version	Subtype	OEM-ID	Number of KVPs	Key 1	Value 1

The following table shows the CDT values for different platforms along with OEM-ID field. For reference OEM-ID value is taken as 0 but you can modify it between values from 1 to 255.

Table : Platform CDT format details for Version 5

CDT platform ID format with OEM-ID	Byte 0: CDT version	Byte 1: Platform ID	Byte 2: Major version	Byte 3: Minor version	Byte 4: subtype	Byte 5: OEM-ID	Byte 6: KVPs	Byte 7: Key	Byte 8: value
QCS6490 Dragonwing™ RB3 Gen2 Core Development Kit	0x05	0x20	0x01	0x00	0x02	0x00	0X01	0X08	0X01
QCS6490 Dragonwing™ RB3 Gen2 Vision Development Kit	0x05	0x20	0x01	0x00	0x05	0x00	0X00	0X00	0X00
IQ-9075 Dragonwing™ Evaluation Kit	0x05	0x20	0x01	0x00	0x00	0x00	0X00	0X00	0X00
IQ-8300 Beta Evaluation Kit	0x05	0x19	0x01	0x00	0x01	0x00	0X00	0X00	0X00

3 Create CDT image

1. Create a CDT directory in your workspace. For example `/local/mnt/workspace/CDT`.
2. In the CDT directory, create a file named CDT.xml with the following code block, depending on your device type, for example, for QCS6490 Development Kit use, QCS6490_developmentkit_CDT.xml.

```
<?xml version="1.0" ?>
<dal>
<module name="config_data_table">
<driver name="NULL">
<device id="cdt_header">
<props name="magic-number"
type="DALPROP_ATTR_TYPE_BYTE_SEQ">
0x43, 0x44, 0x54, 0x00, end
</props>
<props name="version" type="DALPROP_ATTR_TYPE_BYTE_SEQ">
0x01, 0x00, end
</props>
<props name="reserved1" type="DALPROP_ATTR_TYPE_BYTE_SEQ">
0x00, 0x00, 0x00, 0x00, end
</props>
<props name="reserved2" type="DALPROP_ATTR_TYPE_BYTE_SEQ">
0x00, 0x00, 0x00, 0x00, end
</props>
</device>
<device id="cdb0">
<props name="platform_id"
type="DALPROP_ATTR_TYPE_BYTE_SEQ">
0x05, 0x20, 0x01, 0x00, 0x02, 0x00, 0x01, 0x08, 0x01  end
</props>
</device>
</driver>
</module>
</dal>
```

Note: The default CDT details for QCS6490 Development Kit are provided as an example. You can modify these values in the `CDT.xml` according to the platform specifications mentioned in the platform CDT format details: Platform CDT format with OEM-ID Version 5.

The Python script to generate `CDT.bin` is part of the `boot_images/boot/QcomPkg/Tools/cdt_generator.py` directory.

3. To create `CDT.bin`, use the following command format:

```
python cdt_generator.py <CDT XML file name>
     <binary CDT file name>
```

For example, the following commands show how to generate the CDT.bin for the QCS6490 Development Kit.

```
python boot_images/boot/QcomPkg/Tools/cdt_generator.py
QCS6490_developmentkit_CDT.xml QCS6490_developmentkit_
CDT.bin
```

Note: Follow steps 4 to 6 to flash new `CDT.bin` using PCAT. For fastboot, and QDL, methods the steps 4 to 6 aren't required. For more information, see Flash CDT.

4. Run the following command to generate `gpt_*.bin, zeros_*.bin, rawprogram3.xml`, and `patch3.xml`.

Note: In the following code snippet, -p 3 specifies the PHY partition 3. The -x indicates the input XML file.

```
 python <meta_build>/common/config/storage/ptool.py -x <meta_
build>/common/config/ufs/partition_ext.xml -p 3

Output:

.. container:: screenoutput

   <?xml version="1.0" ?>
   <data>
   <!--NOTE: This is an ** Autogenerated file ** -->
   < !--NOTE: Sector size is 4096bytes-->
   <program SECTOR_SIZE_IN_BYTES="4096" file_sector_offset="0"
filename="" label="ALIGN_TO_128K_1" num_partition_sectors="26"
```

```
partofsingleimage="false"      physical_partition_number="3"
readbackverify="false" size_in_KB="104.0" sparse="false" start_
byte_hex="0x6000" start_sector="6"/>
   <program SECTOR_SIZE_IN_BYTES="4096" file_sector_offset="0"
filename="" label="cdt" num_partition_sectors="32"
partofsingleimage="false" physical_partition_number="3"
readbackverify="false" size_in_KB="128.0" sparse="false" start_
byte_hex="0x20000" start_sector="32"/>
   <program SECTOR_SIZE_IN_BYTES="4096" file_sector_offset="0"
filename="" label="ddr" num_partition_sectors="256"
partofsingleimage="false" physical_partition_number="3"
readbackverify="false" size_in_KB="1024.0" sparse="false" start_
byte_hex="0x40000" start_sector="64"/>
   <program SECTOR_SIZE_IN_BYTES="4096" file_sector_offset="0"
filename="" label="last_parti" num_partition_sectors="0"
partofsingleimage="false" physical_partition_number="3"
readbackverify="false" size_in_KB="0" sparse="false" start_byte_
hex="0x140000" start_sector="320"/>
   <program SECTOR_SIZE_IN_BYTES="4096" file_sector_offset="0"
filename="gpt_main3.bin" label="PrimaryGPT" num_partition_
sectors="6" partofsingleimage="true" physical_partition_number=
"3" readbackverify="false" size_in_KB="24.0" sparse="false"
start_byte_hex="0x0" start_sector="0"/>
   <program SECTOR_SIZE_IN_BYTES="4096" file_sector_offset="0"
filename="gpt_backup3.bin" label="BackupGPT" num_partition_
sectors="5" partofsingleimage="true" physical_partition_number=
"3" readbackverify="false" size_in_KB="20.0" sparse="false"
start_byte_hex="(4096*NUM_DISK_SECTORS)-20480." start_sector=
"NUM_DISK_SECTORS-5."/>
   </data>
```


5. Copy the device programmer *prog_firehose_ddr.elf, prog_firehose_lite.elf* files from the `<BOOT-SI-ROOT>/boot_images/boot/QcomPkg/SocPkg/xxx/Bin/LA/DEBUG/` directory to your CDT directory.
6. Modify the `filename=""` parameter in the `rawprogram3.xml` file by adding the `CDT.bin` filename.

```
<program SECTOR_SIZE_IN_BYTES=”4096” file_sector_
offset=”0” filename=”” label=”cdt” num_partition_
sectors=”32” partofsingleimage=”false” physical_
partition_number=”3” readbackverify=”false” size_in_
KB=”128.0” sparse=”false” start_byte_hex=”0x20000” start_
sector=”32”/>
```

4 Manage kernel device tree

A kernel device tree is a data structure that describes the hardware components of a system. It includes details about the CPU, memory, buses, and peripherals.

Device tree source files (.dts or .dtsi) are compiled into a .dtb binary for the Qualcomm Linux kernel.

4.1 Match CDT with DTS

The CDT and device tree are matched during device bootup. Take RB3Gen2 Development Kit as an example. The platform ID and subtype values are 0x20 (Decimal equivalent 32) and 0x2 respectively. These values will be searched and matched with the .dts files (arch/arm64/boot/dts/qcom/qcs6490-addons-rb3gen2-vision-mezz.dts) whose qcom,board-id parameter value has <32 0x2>. Similarly, qcom,oem-id also if defined will be searched in the .dts files and matched to differentiate customer board level changes.

The example shows the details about OEM-ID and board-id in DTS.

```
{
      model = "Qualcomm Technologies, Inc. Robotics RB3gen2 addons
vision mezz platform with OEM-ID 0";
      compatible = "qcom,qcs6490-addons-rb3gen2-vision-mezz", "qcom,
qcm6490";
      qcom,board-id = <32 0x2>, <32 0x602>; qcom,oem-id = <0x00>;
```

After the device has booted up, check the device tree information in the `cat proc/device-tree/model` file. For example, if you are using a Qualcomm Development Kit, you should see the following output.

```
cat /proc/device-tree/model
Qualcomm Technologies, Inc. Robotics RB 3 Gen 2 addons vision mezz
platform with OEM-ID 0.
```

Note: In the example, the OEM-ID value is 0x0. By default, in the DTS OEM-ID isn't defined and you must add the OEM-ID property as needed. You can choose any value from 1-255. Ensure that the OEM-ID defined in the DTS matches the CDT.

5 Flash CDT

Once the CDT.bin is created, you can flash the generated CDT image using either fastboot commands, Product Configuration Assistant Tool (PCAT) or the Qualcomm Downloader (QDL).

5.1 Flast CDT using Fastboot

Prepare the CDT binary and flash it using fastboot by running the following commands:

```
adb reboot bootloader
fastboot devices
fastboot flash cdt <cdt bin filename>
```

5.2 Flash CDT using PCAT

To flash the CDT image using PCAT, follow these instructions on either Linux or Windows operating systems.

1. Enable 9008 EDL mode by running the `adb reboot edl` command.
2. Open PCAT.
3. Connect the device to the host computer and ensure that the 9008 port is detected.
4. Click the *Software download* > *Dowload* button.
5. Select *Meta/FlatBuild*, click *Browse*, go to the CDT package location, and click *Open*.
6. To set the *DeviceProgrammer*, click *Browse*, go to the CDT location, select `prog_firehose_ddr.elf`, and click *Open*.
7. Click *XMLs*, select `rawprogram3.xml`, and click *Open*. Select `patch3.xml` and click *Open*.
8. Click *Download* ` and wait until you see the following message in the PCAT activity log

```
Build loaded successfully on the device.
```

The download is now complete.

5.3 Flash CDT using QDL

For instructions on flashing the CDT using QDL, see https://docs.qualcomm.com/bundle/publicresource/topics/80-70018-254/flash_images.html#flash-cdt

6 References

Related documents

Title	Document number
Document	
Qualcomm Linux Boot Guide	80-70018-4
Qualcomm Linux Boot Guide - Addendum	80-70018-4A

Acronyms and terms

Definition	Acronym
CDT	Configuration data table
CPU	Central processing unit
DTB	DeviceTree blobs
DTS	DeviceTree source
eMMC	Embedded multimedia card
PCAT	Product configuration assistant tool
Identification	ID

LEGAL INFORMATION

Your access to and use of this material, along with any documents, software, specifications, reference board files, drawings, diagnostics and other information contained herein (collectively this "Material"), is subject to your (including the corporation or other legal entity you represent, collectively "You" or "Your") acceptance of the terms and conditions ("Terms of Use") set forth below. If You do not agree to these Terms of Use, you may not use this Material and shall immediately destroy any copy thereof.

1) Legal Notice.

This Material is being made available to You solely for Your internal use with those products and service offerings of Qualcomm Technologies, Inc. ("**Qualcomm Technologies**"), its affiliates and/or licensors described in this Material, and shall not be used for any other purposes. If this Material is marked as "**Qualcomm Internal Use Only**", no license is granted to You herein, and You must immediately (a) destroy or return this Material to Qualcomm Technologies, and (b) report Your receipt of this Material to qualcomm.support@qti.qualcomm.com. This Material may not be altered, edited, or modified in any way without Qualcomm Technologies' prior written approval, nor may it be used for any machine learning or artificial intelligence development purpose which results, whether directly or indirectly, in the creation or development of an automated device, program, tool, algorithm, process, methodology, product and/or other output. Unauthorized use or disclosure of this Material or the information contained herein is strictly prohibited, and You agree to indemnify Qualcomm Technologies, its affiliates and licensors for any damages or losses suffered by Qualcomm Technologies, its affiliates and/or licensors for any such unauthorized uses or disclosures of this Material, in whole or part.

Qualcomm Technologies, its affiliates and/or licensors retain all rights and ownership in and to this Material. No license to any trademark, patent, copyright, mask work protection right or any other intellectual property right is either granted or implied by this Material or any information disclosed herein, including, but not limited to, any license to make, use, import or sell any product, service or technology offering embodying any of the information in this Material.

THIS MATERIAL IS BEING PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, WHETHER EXPRESSED, IMPLIED, STATUTORY OR OTHERWISE. TO THE MAXIMUM EXTENT PERMITTED BY LAW, QUALCOMM TECHNOLOGIES, ITS AFFILIATES AND/OR LICENSORS SPECIFICALLY DISCLAIM ALL WARRANTIES OF TITLE, MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR A PARTICULAR PURPOSE, SATISFACTORY QUALITY, COMPLETENESS OR ACCURACY, AND ALL WARRANTIES ARISING OUT OF TRADE USAGE OR OUT OF A COURSE OF DEALING OR COURSE OF PERFORMANCE. MOREOVER, NEITHER QUALCOMM TECHNOLOGIES, NOR ANY OF ITS AFFILIATES AND/OR LICENSORS, SHALL BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY EXPENSES, LOSSES, USE, OR ACTIONS HOWSOEVER INCURRED OR UNDERTAKEN BY YOU IN RELIANCE ON THIS MATERIAL.

Certain product kits, tools and other items referenced in this Material may require You to accept additional terms and conditions before accessing or using those items.

Technical data specified in this Material may be subject to U.S. and other applicable export control laws. Transmission contrary to U.S. and any other applicable law is strictly prohibited.

Nothing in this Material is an offer to sell any of the components or devices referenced herein.

This Material is subject to change without further notification.

In the event of a conflict between these Terms of Use and the *Website Terms of Use* on www.qualcomm.com, the *Qualcomm Privacy Policy* referenced on www.qualcomm.com, or other legal statements or notices found on prior pages of the Material, these Terms of Use will control. In the event of a conflict between these Terms of Use and any other agreement (written or click-through, including, without limitation any non-disclosure agreement) executed by You and Qualcomm Technologies or a Qualcomm Technologies affiliate and/or licensor with respect to Your access to and use of this Material, the other agreement will control.

These Terms of Use shall be governed by and construed and enforced in accordance with the laws of the State of California, excluding the U.N. Convention on International Sale of Goods, without regard to conflict of laws principles. Any dispute, claim or controversy arising out of or relating to these Terms of Use, or the breach or validity hereof, shall be adjudicated only by a court of competent jurisdiction in the county of San Diego, State of California, and You hereby consent to the personal jurisdiction of such courts for that purpose.

2) Trademark and Product Attribution Statements.

Qualcomm is a trademark or registered trademark of Qualcomm Incorporated. Arm is a registered trademark of Arm Limited (or its subsidiaries) in the U.S. and/or elsewhere. The Bluetooth® word mark is a registered trademark owned by Bluetooth SIG, Inc. Other product and brand names referenced in this Material may be trademarks or registered trademarks of their respective owners.

Snapdragon and Qualcomm branded products referenced in this Material are products of Qualcomm Technologies, Inc. and/or its subsidiaries. Qualcomm patented technologies are licensed by Qualcomm Incorporated.