VC8000D OpenMax

Compiling & Running Guide

4/2/2019

**Legal Notices**

**COPYRIGHT INFORMATION**

This document contains proprietary information of VeriSilicon Holdings Co., Ltd. VeriSilicon reserves the right to make changes to any products herein at any time without notice. VeriSilicon does not assume any responsibility or liability arising out of the application or use of any product described herein, except as expressly agreed to in writing by VeriSilicon; nor does the purchase or use of a product from VeriSilicon convey a license under any patent rights, copyrights, trademark rights, or any other of the intellectual property rights of VeriSilicon or third parties.

**DISCLOSURE/RE-DISTRIBUTION LIMITATIONS**

The information contained herein is not to be used by or disclosed to third parties without the express written permission of an officer of VeriSilicon Holdings Co., Ltd.

(VeriSilicon Distribution Level **D: CONFIDENTIAL – NOT FOR REDISTRIBUTION**).

**TRADEMARK ACKNOWLEDGMENT**

VeriSilicon, the VeriSilicon logo, Hantro and the Hantro logo are the trademarks of VeriSilicon Holdings Co., Ltd. in the United States and/or other jurisdictions. All other trademarks are the property of their respective holders.

For our current distributors, sales offices, design resource centers, and product information, visit our web site located at [*http://www.verisilicon.com*](http://www.verisilicon.com).

For technical support, please email [hantro-support@verisilicon.com](mailto:hantro-support@verisilicon.com?subject=Hantro%20Support%20Query).

VeriSilicon Confidential. Copyright © 2018 by VeriSilicon Holdings Co., Ltd. All rights reserved worldwide.

1. How to compile video\_decoder for VC8000D formats
2. cd openmax\_il/source/decoder
3. Run script build\_dec\_libs.sh to build VC8000D decoder libraries.

Usage: build\_dec\_libs.sh <target> <options>

Available targets:

clean clean build

pclinux build decoder libraries for HW C-model testing

arm\_pclinux build decoder libraries for HW C-model testing at arm platform

arm\_linux build decoder libraries for HW running at ARM platform

Available options:

64 64-bit environment

rel release mode

After building, below libraries may be generated.

libdwl.a DWL library

libtbcommon.a Common testbench library

libvc8kd.a VC8000D CModel library(this lib would not be generated if target “arm\_linux” used)

libdechevc.a HEVC ctrlSW library

libdecvp9.a VP9 ctrlSW library

libdech10p.a H264 ctrlSW library

libdecmpeg4.a MPEG4 ctrlSW library

libdecmpeg2.a MPEG2 ctrlSW library

libdecvc1.a VC1 ctrlSW library

libdecvp6.a VP6 ctrlSW library

libdecjpeg.a JPEG ctrlSW library

libdecvp8.a VP8/WEBP ctrlSW library

libdecavs.a AVS ctrlSW library

libdecrv.a RV ctrlSW library

libcommon.a Common ctrlSW library

1. Modify Makefile in current folder to fit customer's demand

e.g.

64BIT\_BUILD=y build 64-bit target

USE\_HWMODEL=y use HW C-model rather than Real HW platform

1. Run "make" to build VC8000D OMX IL Decoder Component

Available targets:

pclinux build image and video decoder omxil component for HW C-model testing

arm\_pclinux build image and video decoder omxil component for HW C-model testing at ARM platform

arm build image and video decoder omxil component for HW running at ARM platform

clean deletes generated output

Two dynamic link libraries would be generated in this folder.

libOMX.hantro.VC8000D.video.decoder.so

libOMX.hantro.VC8000D.image.decoder.so

1. cd folder "test" and modify Makefile in this folder to fit customer's demand

e.g.

## Select which decoder HW is used. Do not enable both at the same time.

64BIT\_BUILD = y build 64-bit target

1. Run "make" to build VC8000D OpenMAX IL Decoder TB

Available targets:

pclinux build OMX testbench for HW C-model testing

arm\_pclinux build OMX testbench for HW C-model testing at ARM platform

arm build OMX testbench for HW running at ARM platform

clean deletes generated output

The executable file “video\_decoder” would be generated in current folder.

1. How to run video\_decoder for VC8000D formats

Usage: ./video\_decoder [options]

-id <n> The test case id which is specified by user.

-h264 The test format which is specified by user.

-guard <n> Specify the guard margin between nBufferCountMin to nBufferCountActual of external buffers when no DynamicPortConfig happens for adaptive streams.

-iw <n> The width of input buffer which is specified by user.

-ih <n> The height of input buffer which is specified by user.

-w <n> The width of output buffer which is specified by user.

-h <n> The height of output buffer which is specified by user.

-i <stream> The location of input stream which is specified by user.

-tiled Enable tile output(raster output by default).

-rd <dir> The path of reference yuv file which is specified by user.

-rf <yuv> The name of reference yuv file which is specified by user.

-nc No comparison with reference yuv, the output file called "temp.yuv" would be generated.

e.g.

*./video\_decoder -id case\_0001 -h264 -guard 2 -iw 480 -ih 480 -w 540 -h 480 -i /home/testdata/stream.hevc*

*-tiled -rd /home/yuv\_reference -rf case\_0001.yuv*

*./video\_decoder –mpeg4 -guard 0 -iw 1920 -ih 1088 -w 1920 -h 1088 -i /home/testdata/stream.mpeg4 –nc*

*./video\_decoder -id case\_0001 -hevc -guard 2 -iw 480 -ih 480 -w 540 -h 480 -i /home/testdata/stream.hevc*

*-tiled -rfc -rd /home/yuv\_reference -rf case\_0001.yuv*

*./video\_decoder -vp9 -iw 1920 -ih 1088 -w 1920 -h 1088 -i /home/testdata/stream.vp9 –nc*

1. Notes
2. How to configure cross compiler to run HW or HW C-model on ARM platform.

A quickly way to set up cross compiler is using “export”.

export TOOLCHAIN=/lhome/vivscm/working/gcc-linaro-aarch64-linux-gnu-4.9-2014.07\_linux

export PATH=$TOOLCHAIN/bin:$PATH

export CROSS=aarch64-linux-gnu-

export ARCH=-march=armv8-a

1. Please make sure the consistency of compiling option.

For example, If USE\_64BIT\_ENV is enabled when customer compiling decoder library, customer should also make sure 64BIT\_BUILD = y when compiling OMX IL Component and OMX IL TB.

If customer use target “versatile” to build G1 decoder library or use “ENV=arm\_linux” to build G2 decoder library, customer should also use target “arm” to build OMX IL Component and OMX IL TB.