

Vision SDK

(v03.xx)

OpenCX

Copyright © 2014 Texas Instruments Incorporated. All rights reserved.

Information in this document is subject to change without notice. Texas Instruments may have pending patent applications, trademarks, copyrights, or other intellectual property rights covering matter in this document. The furnishing of this documents is given for usage with Texas Instruments products only and does not give you any license to the intellectual property that might be contained within this document. Texas Instruments makes no implied or expressed warranties in this document and is not responsible for the products based from this document.

IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgment, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Customers are responsible for their applications using TI components.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards ought to be provided by the customer so as to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, license, warranty or endorsement thereof.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations and notices. Representation or reproduction of this information with alteration voids all warranties provided for an associated TI product or service, is an unfair and deceptive business practice, and TI is neither responsible nor liable for any such use.

Resale of TI's products or services with *statements different from or beyond the parameters* stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service, is an unfair and deceptive business practice, and TI is not responsible nor liable for any such use.

Also see: [Standard Terms and Conditions of Sale for Semiconductor Products.](http://www.ti.com/sc/docs/stdterms.htm)
www.ti.com/sc/docs/stdterms.htm

Mailing Address:

Texas Instruments
Post Office Box 655303
Dallas, Texas 75265

Copyright © 2014, Texas Instruments Incorporated

TABLE OF CONTENTS

1	OpenCV: Introduction	4
1.1	Building cmem	4
1.2	Steps to load cmem kernel module in Linux	4
1.3	Running OpenCV test-bench on Vision SDK BIOS	4
1.4	Cloning and Building TI OpenCV source.....	5
2	OpenCL: Compiling OpenCL Kernels	6
2.1	Linux	6
2.2	SYSBIOS	6
2.3	Compilation of OpenCL kernels of Vision SDK use-cases	6
3	Revision History	8

1 OpenCV: Introduction

OpenCV provides open compute library and OpenCL offers open standard for heterogeneous programming. Both these open components are enabled in Vision SDK. This document describes steps needed for OpenCL to work in Linux Vision SDK.

1.1 Building cmem

In order to build cmem, follow the instructions in the below link

http://processors.wiki.ti.com/index.php/CMEM_Overview

The git link and checkout tag of the cmem repo are mentioned in the 'VisionSDK_Linux_UserGuide'.

In the above wiki, refer to the 'Build Environment Setup' and 'Building Test Binaries' sections to set up the build environment to build cmem user and kernel modules.

While configuring the makefile generation use the below command instead of the one mentioned in the wiki so that shared libraries are generated instead of static libraries.

```
./configure --host=arm-linux-gnueabi --prefix=$PWD
```

The build process remains same otherwise.

1.2 Steps to load cmem kernel module in Linux

For OpenCL to work in Linux, cmem kernel module is required. Cmem provides a contiguous range of memory to OpenCL device. So, we need to load this kernel module after the system has booted up.

Once Linux Vision SDK has booted up, navigate to the below folder

```
$> cd /opt/vision_sdk
```

Then execute the script 'load_cmem_ko.sh' to load the cmem kernel module.

```
$> ./load_cmem_ko.sh
```

Now, run usual steps described in VisionSDK_Linux_UserGuide.pdf, as below

```
$> ./camnodes.sh
```

```
$> ./vision_sdk_load.sh
```

```
$> ./apps.out
```

Select the demo to run

1.3 Running OpenCV test-bench on Vision SDK BIOS

In order to run OpenCV test-bench from the BIOS environment, a Vision SDK usecase is added.

1.3.1 Enabling OpenCV test-bench:

Follow the below steps to enable OpenCV test-bench:

- Enable building the OpenCV test-bench use-case by setting the make variable `ENABLE_OPENCV_TESTS` to `yes` in `cfg.mk` of `tda2xx_evm_bios_opencx/cfg.mk` file
- The OpenCV test-bench is built as library and linked to this Vision SDK usecase

1.3.2 Configuring the test-bench

In order to configure the test-bench do the following:

- The test-bench entry function names are similar to its counterpart executable names. E.g `opencv_test_imgproc.exe` could be run by calling `test_imgproc()` and setting the appropriate args in the `algplugin`.
- Similarly, for `opencv_perf_imgproc.exe`, the counterpart function entry point would be `perf_test_imgproc()`
- Next download the test data from the below link
 - https://github.com/opencv/opencv_extra
- Copy the `testdata` folder to the SD card before running any `opencv` test that reads any file from the `testdata` folder

1.4 Cloning and Building TI OpenCV source

TI OpenCV source repository contains works related to

- Enabling cross build on ARM A15 running BIOS or Linux
- Offload of certain functions from A15 to C66x DSP
- Test bench changes to enable running in BIOS environment

This repository can be cloned with the below command

`git clone https://bitbucket.itg.ti.com/scm/opencv/tiopencv.git`

In order to build `opencv` source as part of vision sdk build, follow the below procedures

- Navigate to `'ti_components'` folder
- Then, navigate to `'algorithms_codecs'`
- Now, clone the `'tiopencv'` repo using the above mentioned command
- Set the env variable, `'BUIILD_OPENCV_SRC'` to `'yes'` in the `'tda2xx_evm_<OS>_opencx'` config
- Now, executing `make` with the target `'opencx'` will build `vision_sdk` and also `opencv` for the appropriate configuration and the `opencv` build contents will be placed in one of the below folders inside `'tiopencv'` folder
 - `build_bios_debug`
 - `build_bios_release`
 - `build_linux_debug`
 - `build_linux_release`

In order to clean, use `'opencx_clean'` target with `make`.

Note: If BUILD_OPENCV_SRC is set to 'yes', always use the target opencx to build and clean

2 OpenCL: Compiling OpenCL Kernels

In this section we describe the steps for compiling the OpenCL CL kernels with Vision SDK. The Open CL kernels are compiled with the compiler **clocl**. Clocl is in available at <ti_components\opencv_rtos_Rel_VER_XXX>\packages\ti\clocl. More details about clocl and OpenCL in general are available at

<http://downloads.ti.com/mctools/esd/docs/opencv/index.html>

2.1 Linux

The following steps are to be followed for offline compilation of OpenCL Kernels.

1. Set your system Path to //OpenCL_Installation_Folder/packages/ti/clocl.
2. Set the following variable to
 - a. TARGET_ROOTDIR = //YOUR_TARGET_FS/
3. Now you can compile your CL files.

2.2 SYMBIOS

In case of symbios the CL files must be compiled on a host Linux machine.

2.2.1 Basic Compilation of Kernels:

This step is valid when there is no change in the DSP image from the standard Vision SDK release.

1. Set your system Path to //OpenCL_Installation_Folder/packages/ti/clocl.
2. Set TARGET_ROOTDIR = //vision_sdk_Installation_Folder/examples/tda2xx/src/opencv
3. Now you can compile your CL files.
4. If you are including the CL files as pre-compiled header then now you have to build the vision SDK.

2.2.2 Advanced Steps:

These steps are required if there is any change in the DSP 1 Code or Tools or any new component is added or there is a memory map change.

1. Do a clean build of Vision SDK on a linux host.
2. Copy dsp.syms and dsp.syms.obj from
//vision_sdk_Installation_Folde/binaries/tda2xx_evm/bios/opencv/vision_sdk/bin/tda2xx-
evm to
//vision_sdk_Installation_Folder/examples/tda2xx/src/opencv/usr/share/ti/opencv
3. Now follow the Basic Compilation steps described in section 4.2.1.
- 4.

2.3 Compilation of OpenCL kernels of Vision SDK use-cases

In order the compile the OpenCL kernels of Vision SDK use-cases, the following are required to be done.

2.3.1 Setting up Environment variables

Set up the following environment variables before building vision sdk

1. The following two steps are needed only if opencl kernel is compiled without vision sdk
 - a. Add clocl path to PATH variable. (*this is already taken care in vision sdk*)
 - i. This can be found inside opencl folder inside ti_components for BIOS and
 - ii. Inside <targetfs path>/usr/share/ti/opencl/bin/x86 for Linux
 - b. Set TI_OCL_CGT_PATH variable pointing to dsp compiler root path (this is already taken care in vision sdk)
2. Add dsp compiler bin path to PATH variable
 - a. i.e. add the compiler binary path to PATH variable
 - b. e.g export PATH=<path_to_dsp_compiler_8.1.x>/bin:\$PATH

2.3.2 BIOS:

Follow the below steps:

3. The OpenCL kernels should reside inside the folder named 'kernel' of the respective algplugin
4. A Makefile is required to be present inside this folder and this can be just copied from one of the existing examples. E.g. from 'openclframecopy' algplugin
5. Once the above two steps are done, building vision sdk with target named 'opencx' is required which builds these OpenCL kernels along with usual Vision SDK build
 - a. i.e. make opencx -s -j
6. *Note: if there is no intention of building OpenCV do not set BUILD_OPENCV_SRC to yes in 'tda2xx_evm_<OS>_opencx' config*

2.3.3 Linux:

Follow the below steps:

1. The first two steps mentioned in BIOS section, 2.3.1 are to be followed
2. Then, add a build target named 'opencl_build' to the MAKEFILE.MK file of the algplugin
3. Then, in addition to performing make on the algplugin MAKEFILE.MK default target, perform make on the algplugin MAKEFILE.MK 'opencl_build' target
 - a. This is done from the 'MAKEFILE_adas.MK' found in <vision_sdk_path>/hlos/examples/ folder

3 Revision History

Version	Date	Revision History
1.0	18th October 2016	Initial Version
1.1	25 th October 2016	Addressed review comments
1.2	9 th June 2017	Added steps to build OpenCV from source
1.3	5 th July 2017	Minor corrections in path and footer
1.4	22 nd March 2018	Updated OpenCV repo path

« « « § » » »