Vision SDK

(v02.07.00)

Linux User Guide

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.
www.ti.com/sc/docs/stdterms.htm

Mailing Address:

Texas Instruments
Post Office Box 655303
Dallas, Texas 75265

TABLE OF CONTENTS

1 Introduction 4
1.1 References 4
2 Installation Overview 5
2.1 PC Requirements 5
2.2 Software Requirements 5
2.3 Hardware Requirements 6
2.4 Software Installation & setup 6
2.5 Directory Structure 9
3 Build 11
3.1 Build for Linux + vision_sdk Capture + Display usecases 11
3.2 Build for AVB Capture(IPU1_0) -> Decode-> SGX + HDMI / LCD display (A15) 12
4 Run 14
4.1 Board Setup 14
4.2 Preparing SD card & Boot 14
4.3 Run demos 16
5 TDA2EX specific details 19
6 Revision History 19

1 Introduction

Vision Software Development Kit (Vision SDK) is a multi processor software development package for TI's family of ADAS SoCs. The software framework allows users to create different ADAS application data flows involving video capture, video pre-processing, video analytics algorithms, and video display. The framework has sample ADAS data flows which exercises different CPUs and HW accelerators in the ADAS SoC and shows customer how to effectively use different sub-systems in the SoC. Frame work is generic enough to plug in application specific algorithms in the system.

Vision SDK is currently targeted for the TDA2xx family of SoCs

This document particularly explains Linux part of vision SDK where A15 is supposed to run Linux as target OS while other cores in the SOC run SYSBIOS. Build environment is also assumed to be Linux and user should be familiar with basics of Linux to follow this document.

1.1 References

Refer the below additional documents for more information about Vision SDK

Document	Description
Vision_sdk/linux/docs/VisionSDK_ReleaseNotes.pdf	Release specific information
Vision_sdk/linux/docs/VisionSDK_UserGuide.pdf	This document. Contains install, build, execution information
Vision_sdk/docs/VisionSDK_UserGuide.pdf	Not relevant **unless explicitly mentioned in this document**
Vision_sdk/docs/VisionSDK_ReleaseNotes.pdf	Not relevant to this code drop
Vision_sdk/docs/VisionSDK_DataSheet.pdf	Not relevant to this code drop
Vision_sdk/docs/VisionSDK_ApiGuide.CHM	User API interface details
Vision_sdk/docs/VisionSDK_SW_Architecture.pdf	Overview of software architecture
Vision_sdk/docs/VisionSDK_DevelopmentGuide.pdf	Developer Guide on bios side only, partly relevant for this code drop
Vision_sdk/docs/VisionSDK_SurroundView_DemoSetUpGuide.pdf	Refer for all surround view use-cases.

2 Installation Overview

This chapter provides a brief description on the system requirements (hardware and software) and instructions for installing Vision SDK.

2.1 PC Requirements

Installation of this release needs a linux ubuntu 10.04/12.04 machine.

IMPORTANT NOTE: If you are installing ubuntu on a virtual machine, ensure its a 64 bit ubuntu.

2.2 Software Requirements

All software packages required to build and run the Vision SDK are included as part of the SDK release package except for the ones mentioned below

2.2.1 A15 Compiler, Linker

The linux installer for the linaro tools should be downloaded from below link

https://launchpad.net/linaro-toolchain-binaries/trunk/2013.03/+download/gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313_linux.tar.bz2

The tools need to be installed in $INSTALL_DIR/ti_components/os_tools/linux/linaro location.

IMPORTANT NOTE: A15 Compiler and linker MUST be installed before proceeding the build else compilation will fail. Also make sure the compiler is installed at the exact path mentioned above after installation of vision sdk.

Use following steps to install the toolchain

```
$> cd $INSTALL_DIR/ti_components/os_tools/linux/linaro
$> tar –xjvf gcc-linaro-arm-linux-gnueabihf-4.7-2013.03-20130313_linux.tar.bz2
```

Ensure the toolchain is for 32 / 64 bit machine as per configuration of installation machine

If your machine is 64 bit and you have downloaded toolchain from link above

Execute following step on installation machine

$>sudo apt-get install ia32-libs

2.2.2 Linux kernel, uboot, sgx driver and target file system

Prior to this vision sdk release kernel, uboot, sgx & target filesystem were part of vision_sdk package. In this release they need to be separately downloaded / cloned. Links to download / clone are mentioned in software installation section 2.4

2.2.3 Other mandatory software packages for build

Ensure these packages/tools are installed on the installation machine

uname, sed, mkimage, dos2unix, dtrx, mono-complete, git, corkscrew

To install

```
$>sudo apt-get install <package_name>
```

2.2.4 Code Composer Studio

CCS is needed to load, run and debug the software. CCS can be downloaded from the below link. CCS version 5.4.0.00091 should be installed.

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

2.3 Hardware Requirements

Please refer $INSTALL_DIR/vision_sdk/docs/VisionSDK_UserGuide.pdf

IMPORTANT NOTE: This release supports vision sdk only on Rev E or higher version of EVMs

2.4 Software Installation & setup

Download VISION_SDK_XX_XX_XX_XX_setuplinux.bin

```
$> bash
$> ./ VISION_SDK_XX_XX_XX_XX_setuplinux.bin
```

Accept license agreement and give <installation_directory_absolute_path> e.g. /home/usename/foldername/VISION_SDK_XX_XX_XX_XX

```
$> cd <installation_directory_absolute_path>
$> export INSTALL_DIR=<installation_directory_absolute_path>
```

2.4.1 Install Linux kernel, uboot, sgx and file system

If you are setting up git first time you need to setup your .gitconfig and gitproxy with as shown below

IMPORTANT NOTE: These steps are just indicatory. You may have to figure out arguments (e.g. paths for .gitconfig / git-proxy.sh / arguments to corkscrew) based on your particular system.

1. Edit .gitconfig

```
$>vi /home/<username>/.gitconfig
[core]
        gitproxy = none for ti.com
        gitproxy = /home/<username>/git-proxy.sh
```

save and exit (ESC + :wq)

2. Create git-proxy.sh

```
$>vi /home/<username>/git-proxy.sh
exec /usr/bin/corkscrew proxyle01.ext.ti.com 80 $*
```

save and exit (ESC + :wq)

You should see following output if the setup is successful

```
$> git config --list
core.gitproxy=none for ti.com
core.gitproxy=/home/<username>/git–proxy.sh
```

IMPORTANT NOTE: Steps below are absolutely mandatory and pre-cursors before you move to building vision sdk

You can choose to perform steps 1 – 4 in parallel through different terminals, These need not be sequential, to open terminal use Ctrl + Alt +t

Step 1 - Clone kernel & apply patch

```
$> cd $INSTALL_DIR/ti_components/os_tools/linux/kernel
$> git clone git://git.omapzoom.org/kernel/omap.git
$> cd omap/
$> git checkout -b kernel_dev 2865b60202ac4dc9598e6dcf0e6a76381c5d7696
```

Apply patch for memreserve in dts file

TDA2XX

Ensure you are in $INSTALL_DIR/ti_components/os_tools/linux/kernel/omap directory

```
$> git apply ../linux-kernel-addon/patches/dra7xx_evm_linux_infoadas.patch
```

TDA2EX

Ensure you are in $INSTALL_DIR/ti_components/os_tools/linux/kernel/omap directory

```
$> git apply ../linux-kernel-addon/patches/dra72x_evm_linux_infoadas.patch
```

memcahe.ko source

There is additional kernel module needed for vision_sdk, ensure the source lies under $INSTALL_DIR/ti_components/os_tools/linux/kernel/linux-kernel-addon/memcache, this comes by default with installation.

Step 2 - Clone u-boot

```
$> cd $INSTALL_DIR/ti_components/os_tools/linux/u-boot
$> git clone git://git.omapzoom.org/repo/u-boot.git
$> cd u-boot/
```

```
$> git checkout -b uboot_dev b89e568d9c356d756501b16caad5734970ec25d1
```

Step 3 - Clone sgx drivers

```
$> cd $INSTALL_DIR/ti_components/os_tools/linux/sgx
$> git clone git://git.ti.com/graphics/omap5-sgx-ddk-linux.git
$> cd omap5-sgx-ddk-linux/
$> git checkout -b sgx_dev 0f3561a47cf7a1b968fb64446a0be10abc9a15d5
```

Step 4 - Download and untar file system

Download file system from this location

http://downloads.ti.com/dsps/dsps_public_sw/glsdk/7_00_00_04/exports/arago-glsdk-multimedia-image-dra7xx-evm.tar.gz

and keep under following directory

$INSTALL_DIR/vision_sdk/linux/targetfs

Untar file system

You need to be root user to do this

```
$> sudo –s / sudo
root> bash
root> export INSTALL_DIR=<installation_directory_absolute_path>
root> cd $INSTALL_DIR/vision_sdk/linux/targetfs
root> tar xzvf arago-glsdk-multimedia-image-dra7xx-evm.tar.gz
root> chmod 777  -R ../targetfs
root> exit
```

2.4.2 Un-installation

```
$> rm -rf $INSTALL_DIR
```

2.5 Directory Structure

Once Vision SDK is installed (i.e. directory named VISION_SDK_XX_XX_XX referred as **$INSTALL_DIR** in entire document); there will be two main directories installed under**–** vision_sdk, ti_components.

Folder Description

Folders are briefly described in table below

Directory	Description
vision_sdk	Root directory of vision SDK
vision_sdk/linux	Linux part of vision sdk
vision_sdk/linux/boot	Pre-built images before the build, built images after the build
vision_sdk/linux/build	Support files for building linux part of release package
vision_sdk/linux/docs	Documentation for Linux part of vision SDK
vision_sdk/linux/examples	Usecases that run with Linux + vision sdk
vision_sdk/linux/scripts	Scripts related to linux part of vision SDK
vision_sdk/linux/src	Links sources
vision_sdk/linux/targetfs	Root file system
vision_sdk/build	Support files for building enti e release package
vision_sdk/build/makerules	Make rules for components and cores
vision_sdk/build/tda2xx	Rules specifically for TDA2xx device
vision_sdk/docs	Documentation for vision SDK
vision_sdk/examples	Sample Use case / application folder
vision_sdk/examples/tda2xx	Use case for TDA2xxx device
vision_sdk/examples/tda2xx/include	All include files needed for use case
vision_sdk/examples/tda2xx/src	All source files needed for use case
vision_sdk/include	All the include files for the framework
vision_sdk/include/link_api	Interface files for all the links
vision_sdk/src	All the source files for the framework
vision_sdk/src/links_a15	Source files for individual links present on A15
vision_sdk/src/links_common	Files which are for common across all links
vision_sdk/src/links_dsp	Source files for individual links present on DSP
vision_sdk/src/links_eve	Source files for individual links present on EVE
vision_sdk/src/links_ipu	Source files for individual links present on IPU
vision_sdk/src/main_app	Folder for main() functionality
vision_sdk/src/utils_common	Common utilities used in framework
ti_components	Root directory of tools accompanying vision SDK
ti_components/algorithms_codecs/eve_sw_xx_xx_xx_xx	EVE Kernels Library
ti_components/algorithms_codecs/framework_components_x_xx_xx_xx	Framework Components
ti_components/algorithms_codecs/ivahd_hdvicp20api_xx_xx_xx_xx_xxxx	The HDVICP library is an interface between HDVICP2 hardware and the codec

Directory	Description
ti_components/algorithms_codecs/ivahd_jpegvdec_xx_xx_xx_xx_xxxx	MJpeg decoder files
ti_components/algorithms_codecs/vlib_c66x_x_x_x_x	Video Processing Functions Library
ti_components/cg_tools	All the code gen tools
ti_components/cg_tools/windows/arm_x_x_x	Tools needed for ARM CPU cores
ti_components/cg_tools/windows/arp32_x_x_x	Tools needed for ARP32 core
ti_components/cg_tools/windows/c6000_x.x.x	Tools needed for C66x DSP
ti_components/drivers	All the drivers used in Vision SDK
ti_components/drivers/bsp_xx_xx_xx_xx	Driver components for all the peripherals
ti_components/drivers/edma3_lld_xx_xx_xx_xx	Driver for system DMA usage
ti_components/gel/DRA7xx	Gel files modified for Vision-Sdk
ti_components/networking	Networking related tools
ti_components/networking/avbtp_xx_xx_xx_xx	AVB Stack files
ti_components/networking/ndk_x_xx_xx_xx	Network Development Kit
ti_components/networking/nsp_vayu_x_xx_xx_xx	Network Development Kit Support Package
ti_components/drivers/starterware_xx_xx_xx_xx	Lowest level SW interface for programming HW registers
ti_components/os_tools	Operating System related tools
ti_components/os_tools/bios_x_xx_xx_xx	BIOS operating sytem used in Vision SDK
ti_components/os_tools/ipc_x_xx_xx_xx	Interprocessor communication files
ti_components/os_tools/windows/xdctools_x_xx_xx_xx	XDC tools related files

3 Build

Important Note: In this release reference to variations in platform names would be found, basically they resemble same platform for informational adas

tda2xx as dra7x or vice versa

tda2ex as dra72 or vice versa

For building binaries follow these steps

3.1 Build for Linux + vision_sdk Capture + Display usecases

Note: If you are trying this after 3.2 ensure you do a 'make clean' and manually delete $INSTALL_DIR/vision_sdk/binaries folder and then proceed

1. You must have followed all the steps in software installation and setup section 2.4 before you proceed
2. Select platform in Rules.make

   TDA2XX

```
VSDK_BOARD_TYPE := TDA2XX_EVM
```

   TDA2EX

```
VSDK_BOARD_TYPE := TDA2EX_EVM
```

3. Ensure in $INSTALL_DIR/vision_sdk/Rules.make

```
A15_TARGET_OS := Linux
```

4. Build the linux dependencies, this will build kernel, u-boot and sgx drivers & memcache.ko

   NOTE: This step is needed only first time you build or if you make any changes to u-boot/kernel/sgx drivers, otherwise this can be skipped.

```
$> cd $INSTALL_DIR/vision_sdk
$> make linux
$> make linux_install
```

5. Build the sdk

```
$>make –s
```

6. Step 5 will ensure your firmware binaries and linux application's .out are copied into your targetfs i.e. $INSTALL_DIR/vision_sdk/linux/targetfs/lib/firmware and $INSTALL_DIR/vision_sdk/linux/targetfs/opt/vision_sdk respectively. It will also execute make linux_install to copy all binaries needed to boot into $INSTALL_DIR /vision_sdk/linux/boot and $INSTALL_DIR /vision_sdk/linux/targetfs/boot
7. The file system under $INSTALL_DIR/vision_sdk/linux/targetfs now can be used as either NFS or rootfs on sd card.
   a. Tar the file system and keep it in $INSTALL_DIR/vision_sdk/linux/boot/ folder, you need to have root/sudo access on the machine to do this.

```
$>sudo –s or $>sudo
root>export INSTALL_DIR=<installation_directory_absolute_path>
root>cd  $INSTALL_DIR/vision_sdk/linux/targetfs
root>tar czvf arago-glsdk-multimedia-image-dra7xx-evm.tar.gz  ./*
root>mv ./ arago-glsdk-multimedia-image-dra7xx-evm.tar.gz
          $INSTALL_DIR/vision_sdk/linux/boot/
root>exit
```

NFS + SD boot

Important Note: You need to keep file system in two places (idiosyncrasy of 3.14 kernel). The zImage and dra7-evm-infoadas.dtb is picked up from /boot folder on the sd card always while firmware binaries are picked up from $INSTALL_DIR/vision_sdk/linux/targetfs/lib/firmware

$INSTALL_DIR/vision_sdk/linux/targetfs is used as nfs

For NFS to work, $INSTALL_DIR/vision_sdk/linux/targetfs needs to be exported from /etc/exports of installation machine.

Refer section 4.2 to prepare SD card for NFS.

SD only boot

The tarred file system needs to be flashed on sd card along with uboot and uenv.txt refer section 4.2

3.2 Build for AVB Capture(IPU1_0) -> Decode-> SGX + HDMI / LCD display (A15)

Note: if you are trying this after 3.1 ensure you do a 'make clean' and manually delete $INSTALL_DIR/vision_sdk/binaries folder and then proceed

1. You must have followed all the steps in software installation and setup section 2.4
2. Ensure A15_TARGET_OS := Linux in $INSTALL_DIR/vision_sdk/Rules.make
3. Ensure NDK_PROC_TO_USE=ipu1_0 in $INSTALL_DIR/vision_sdk/Rules.make
4. Patch the dra7-evm.dtb as shown below

```
diff --git a/arch/arm/boot/dts/dra7-evm.dts b/arch/arm/boot/dts/dra7-evm.dts
index 3719c5a..350b26e 100644
--- a/arch/arm/boot/dts/dra7-evm.dts
+/*
&mac {
      status = "okay";
      pinctrl-names = "default", "sleep";
@@ -947,7 +951,7 @@
      phy-mode = "rgmii";
```

```
        dual_emac_res_vlan = <2>;
};
+*/
&davinci_mdio {
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&davinci_mdio_default>;
```

Basically you need to comment out mac, cpsw_emac0 and cpsw_emac1 from arch/arm/boot/dts/dra7-evm.dts and rebuild the dra7-evm-infoadas.dtb file

5. Build the linux dependencies, this will build kernel, u-boot and sgx drivers

   NOTE: This step is needed only first time you build or if you make any changes to u-boot/kernel/sgx drivers , otherwise this can be skipped.

```
$> cd  $INSTALL_DIR/vision_sdk
$> make kernel_menuconfig
```

Need to manually disable Ethernet and EDMA driver from kernel through menuconfig

Go to device drivers->Network device support & disable "Ethernet driver support"

Go to **device drivers-> DMA Engine support & disable "TI EDMA support"**

Save the config and exit

```
$> make linux
$> make linux_install
```

6. Disable all cores in Rules.make except IPU1_0 and A15 using PROC_<CORE_NAME>_INCLUDE in Rules.make
7. Refer steps 4 to 7 in section 3.1

Note: AVB is not tested on TDA2ex

4 Run

4.1 Board Setup

Refer $INSTALL_DIR/vision_sdk/docs/VisionSDK_UserGuide_TDA2xx.pdf

Note: The setup is different for 1ch / 4ch LVDS capture + SGX display and AVB->Decode->SgxDisplay usecase.

4.1.1 Capture switch setting

Video Config pins needs to set for different capture inputs

VIDEO CONFIG switch settings (SW3 on TDA2xx Vision Application Board (set for Ov10635 in Original version of CPLD))

Capture Type	**Hardware controlled pin settings Vision Application Board (Rev C CPLD) (default cpld image)**							
	1	2	3	4	5	6	7	8
OV10635	**OFF**	**ON**	**OFF**	**ON**	**OFF**	**ON**	**OFF**	**ON**
LVDS	**OFF**	**OFF**	**ON**	**OFF**	**OFF**	**ON**	**OFF**	**ON**
HDMI	**OFF**	**OFF**	**ON**	**ON**	**OFF**	**ON**	**OFF**	**ON**

4.2 Preparing SD card & Boot

1. Connect micro SD card to installation machine
2. *NFS + SD boot*

   Ensure $INSTALL_DIR/linux/scripts/uenv_nfs.txt is copied as uenv.txt

   Update uenv.txt in linux/boot folder to have right nfs path & serverp

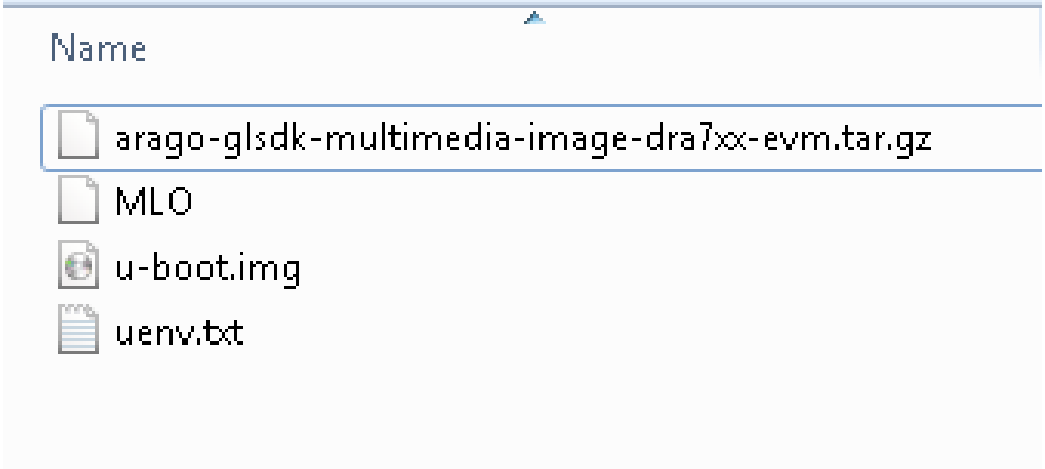
   SD only boot

   Ensure $INSTALL_DIR/linux/scripts/uenv_sd.txt is copied as uenv.txt

   Use dos2unix command on uenv.txt

   ```
   $> dos2unix $INSTALL_DIR/linux/boot/uenv.txt
   ```

3. Ensure $INSTALL_DIR/linux/boot folder appears as shown in picture after build



4. Format SD card and create two partitions (boot (FAT32) and rootfs(ext4)) using script below, <parent_device_name> can be found using "mount" command

```
$>sudo –s or sudo
root> bash
root> export INSTALL_DIR=<installation_directory_absolute_path>
root> cd $INSTALL_DIR/vision_sdk/linux/scripts
root> ./mksdboot.sh     --device /dev/<parent_device_name> --sdk $INSTALL_DIR/vision_sdk
root> exit
```

8. Disconnect SD card from installation machine and insert it into EVM micro SD slot.
9. Set hardware pin settings for SD Boot

   Make sure the Boot Mode Select Switch is set for the SD boot mode **on TDA2xx Base EVM**. This is done by setting the pins SYSBOOT (SW2+SW3) [0:15] to the below shown position

SYSBOOT	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	SW2								SW3							
SW Pin	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8
SW Position	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	ON	OFF	OFF	OFF	OFF	OFF	OFF	ON

10. Power on the EVM, press power reset

4.3 Run demos

1. Log in as root on the UART console

$> dra7xx-evm login: root
root@dra7xx-evm:~#

HDMI/LCD display should appear as shown in picture below
This release by default supports HDMI not LCD.

Following errors may be seen on display and the message can be ignored.
failed to load module: /usr/lib/gbm/gbm_dri.so: cannot open shared object file: No such file or directory
failed to load module: /usr/lib/gbm/gbm_gallium_drm.so: cannot open shared object file: No such file or directory

3. Proceed on UART console & execute

(TDA2XX) and (TDA2EX single channel)

```
$> cd /opt/vision_sdk
$> ./camnodes.sh
```

```
$> ./vision_sdk_load.sh
$> ./vision_sdk_linux_demo.out
Select the demo to run
```

(TDA2XX – TIDA00455/OV490 multi-channel use-case)

```
$> cd /opt/vision_sdk
$> ./camnodes.sh
$> ./vision_sdk_load.sh
$> ./ vision_sdk_ov490_pinmux.sh
$> ./vision_sdk_linux_demo.out
Select the demo to run
```

(TDA2EX LVDS)

```
$> cd /opt/vision_sdk
$> ./camnodes.sh
$> ./set_vip_mux.sh
$> ./vision_sdk_load.sh
$> ./vision_sdk_linux_demo.out
Select the demo to run
```

4.3.1 Application Constraints

- PD + TSR usecase in this release is tuned for pre-recorded content. This usecase is just a demonstration of DSP and EVEs along with linux in picture, does not show very good output on live content.
- 3DSRV usecase will only run with very low FPS (non-real time) on TDA2EX as there is only one GPU (SGX544) on TDA2Ex. However TDA2x has 2 SGX544 cores
- "Exit" the VSDK Linux side app while switching from one usecase to another. This is due to some known issue in the current DRM driver, Will fix this in next release, till then please follow the below steps to switch across different UCs,
    - '0' – to stop the current UC
    - 'x' – exit the SDK Linux side app
    - ./vision_sdk_linux_demo.out – to re-run the SDK Linux side app
    - Enter the menu option for new UC

4.3.2 Linux + vision_sdk 1ch / 4ch Capture(IPU1_0) -> SGX + display (A15)

Only 4 ch LVDS capture + display is show below on LCD, 1ch VIP output will also be similar except mosaic, You can also print detailed statistics of each Link using "Chains Run-time Menu" option 'p'

4.3.3 Linux + vision_sdk 1ch Capture(IPU1_0)-> Encode -> Decode->SgxDisplay

This usecase is just proof of concept and not real usecase for vision sdk. Users can ignore this usecase.

4.3.4 Linux + vision_sdk AVB Capture(IPU1_0) -> Decode->SgxDisplay

You should see similar output in 2x2 format in HDMI/LCD display for the encoded video streams being streamed from Linux PC.

IMPORTANT NOTE: Ensure Ethernet cable is connected to port Ethernet0 (Ethernet port close to HDMI), and run the avb talker application after you choose this option

Remove dra7-ipu2-fw.xem4 from /lib/firmware on the target filesystem

4.3.4.1 Running avb talker binary on host PC

1. Copy $INSTALL_DIR/ti_components/networking/avbtp_talker/avbtp_talker.sh and 1.AVI (input file) to host PC
2. On Host machine (connected to EVM by Ethernet cable), run following command

   $> cd <avb_talker_install_directory>

   $> ./avbtalker.sh 1.AVI 1.AVI 1.AVI 1.AVI

4.3.5 4ch LVDS capture, 3D surround view and SGX display

Refer VisionSDK_UserGuide_TDA2xx.pdf (section 2.3 Hardware Requirements) & VisionSDK_SurroundView_DemoSetUpGuide.pdf to set-up the surround view demos and run the use-case.

Erase the calibration table, when run the demo flirt time on any new set-up after boot-up by

rm –rf /home/root/.calibtable

4.3.6 TIDA00455/OV490 based capture, 3D surround view and SGX display

Refer to section 4.3 for steps to run the use-case. In case of green artifacts on display, try changing toggling polarity of capture using the command "omapconf write 0x4A002534 0x0" or "omapconf write 0x4A002534 0x5". 0x5 is the recommended setting – it might vary based on different OV490 firmwares/board versions/etc. This command can be run before running vision_sdk_linux_demo.out or while the use-case is running from another terminal by connecting via telnet.

5 TDA2EX specific details

1. The default dtb for TDA2EX is dra72-evm-infoadas.dtb which works well for Single channel capture. Update uenv.txt in linux/boot folder to have right dtb file name.
2. For LVDS capture, only SD boot works as Ethernet is been disabled after board Modification. (Refer BSP user guide section "**Changes needed for 4 Channel Multi-deserializer on TDA2EX EVM"** for more information).
3. For LVDS usecase - Use only uenv_sd.txt(Renamed as uenv.txt) under /linux/boot for LVDS and the dtb used is dra72-evm-infoadas-LVDS.dtb. Update uenv.txt to have appropriate dtb file name.
4. In order to build dra72-evm-infoadas-LVDS.dtb, change DEFAULT_DTB variable in Rules.make to dra72-evm-infoadas-LVDS.dtb and rebuild kernel.

   Note: By default dra72-evm-infoadas.dtb is built.

   Note: Only Single channel capure use cases (Option 1 & 2) and LVDS capture usecases (option 4) are validated.

6 Revision History

Version	Date	Revision History
1.0	30th June 2014	Initial Version
2.0	30th July 2014	Updated for vision sdk 2.03
3.0	17 th Dec 2014	Updated for 2.05
4.0	28th Feb 2015	Updated for 2.06

«««« § »»»»