

Vision SDK Multi-Sensor Fusion Platform (a.k.a. MonsterCam)

(v02.07.00)

User Guide

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.
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	Introduction	4
1.1	References	4
1.2	Features on MonsterCam.....	4
1.3	Directory Structure.....	5
2	Installation Overview	5
2.1	PC Requirements.....	5
2.2	Software Requirements.....	5
2.3	Hardware Requirements.....	6
2.4	Software Installation	9
3	Build and Run	10
3.1	Building the MLO.....	10
3.2	Building the application	10
3.3	Boot modes	13
3.4	Load using SD card	13
3.5	Load using QSPI.....	13
3.6	Load using CCS.....	17
3.7	Running application	18
4	DM388 boot up and changing default binaries	19
4.1	Default DM388 binaries.....	19
4.2	Changing default DM388 binaries (file system)	19
5	Supported usecases on MSF	22
5.1	Single channel usecases.....	22
5.2	Stereo	22
6	Frequently Asked Questions.....	23
6.1	Build, install, load, PM related FAQs.....	23
6.2	MSF run related FAQs	23
7	Directory Structure Details.....	23
8	Revision History	23

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, TDA3xx family of SoCs

This document explains the HW/SW setup for Multi-Sensor Fusion Platform (MSF) also known as MonsterCam.

MSF is a TDA2XX based platform which can be used for developing fusion usecases and algorithms related to depth recognition in automotive. It provides way to interface with multiple sensors such as ToF, Thermopile, PIR along with cameras. It has interface for stereo camera sensors, high resolution main imager (mainly acts as front camera) and 3 FPD link cameras. The main imager is connected to SoC DM388. DM388 ISP is used for image processing (Conversion from RAW/BAYER to YUV). The processed stream is transferred to TDA2XX over video (i/p) ports for front camera algorithms to consume.

Vision SDK only supports s/w needed for TDA2XX, s/w running on DM388 is not part of Vision SDK, and neither is it supported. By default, MSF has a DM388 binary flashed which streams captured video stream to TDA2XX over video port. It is possible to change that binary using additional debug board.

1.1 References

Refer the below additional documents for more information about Vision SDK

Document	Description
VisionSDK_ReleaseNotes.pdf	Release specific information
VisionSDK_UserGuide_TDA2xx.pdf	This document. Contains install, build, execution information
VisionSDK_ApiGuide.CHM	User API interface details
VisionSDK_SW_Architecture.pdf	Overview of software architecture
VisionSDK_DevelopmentGuide.pdf	Developer Guide

1.2 Features on MonsterCam

Main Image sensor usecases and stereo usecases are supported on MonsterCam. **ToF, FPD Link board, thermopile, PIR are not supported through software in this release.**

1.3 Directory Structure

Once Vision SDK is installed; there will be two main directories installed – vision_sdk and ti_components.

Refer VisionSDK_UserGuide_TDA2xx.pdf for details of directory structure.

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 windows machine with about 6GB of free disk space. Building of the SDK is supported on windows environment.

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 Tera Term 4.72 or higher

This is used to get console prints from the MSF (TDA2XX and DM388 both)

This can be downloaded from

<http://ttssh2.sourceforge.jp/>

Based on your system's requirement you may need to download ftdi chip drivers actually have USB-> UART connection functional. It can be downloaded from here

<http://www.ftdichip.com/Drivers/VCP.htm>

2.2.2 Code Composer Studio

CCS is needed to debug the software. CCS can be downloaded from the below link. CCS version CCS version 6.0.1.00040 should be installed.

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

NOTE: Vision SDK can be used with CCS 5.4 and CCS 5.5 as well

2.3 Hardware Requirements

Hardware setup for MSF is described in this section

1. MSF unit including following boards
 - a. Main board (TDA2xx)
 - b. DM388 board / ISP Processor
 - c. Main Image Sensor Board
 - d. Stereo Sensor Board
 - e. Power Board
 - f. FPD link Board (SAT0074) (optional)
 - g. DM388 debug Board (optional)
2. Cables and connectors
 - a. Power cable connector
 - b. JTAG 20 pin connector (needed only for debug)
 - c. Internal cables
 - i. FPC cables for board to board data connections
 1. 1 for Main Image sensor board to DM388 board (at J7)
 2. 2 for Stereo sensor board to Main board (at J6 & J9)
 - d. External cables
 - i. Mini USB -> USB for PC based UART connection
 - ii. Power adapter (12V, 5 Amp)
 - iii. Power adapter connector
 - iv. HDMI Type A cable
 - v. HDMI Type D -> Type A cable (optional - only if DM388 directly connected to HDMI TV)
 - vi. Ethernet cable
 - vii. external FPC cable to connect DM388 debug board (optional)
3. 2 GB / 4GB FAT32 formatted bootable SD card

2.3.1 MSF h/w terminologies

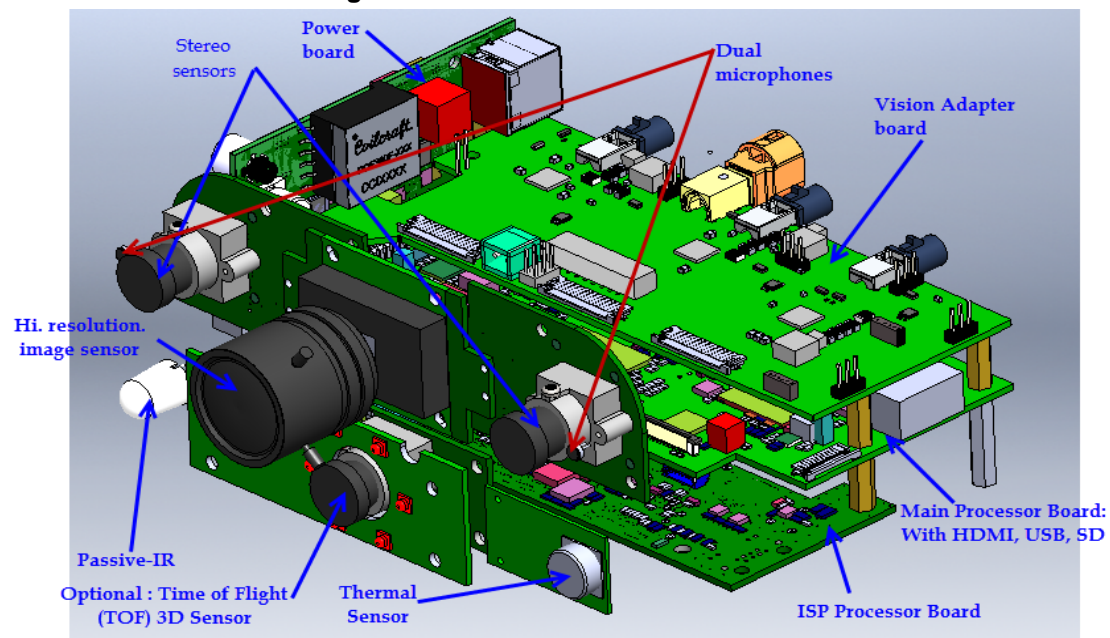


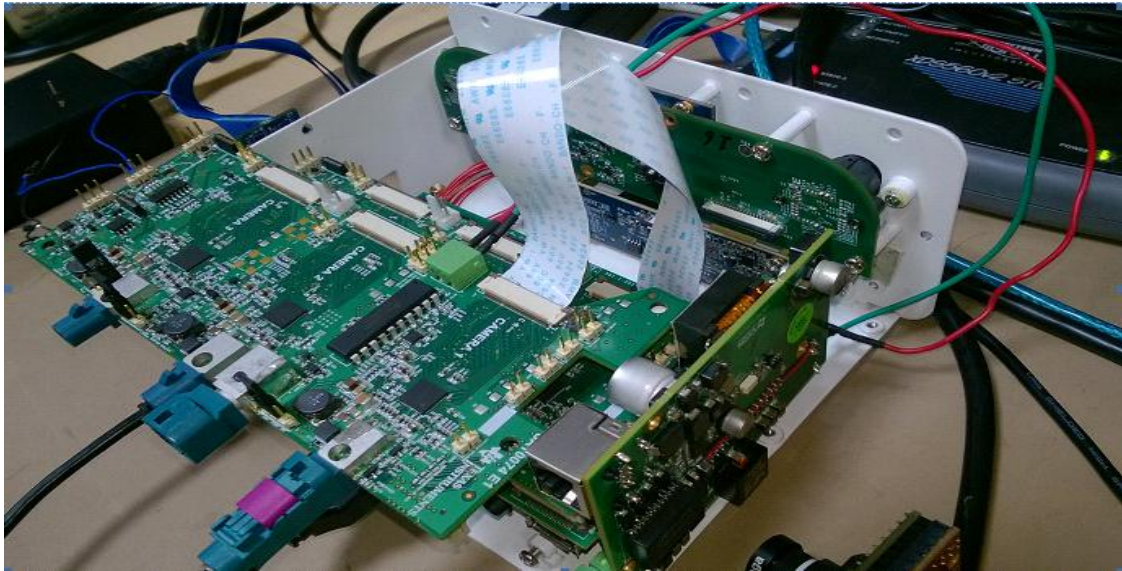
Figure 1 : MSF internal boards and sensors

IMPORTANT NOTE : For more detail on each board inside MSF please refer "[VisionSDK_HardwareSpecification_MultiSensorFusionPlatform.pdf](#)"

2.3.2 Default data connections & care abouts

By default inside the casing of MSF unit, Main Imager board is connected to DM388 board (bottom most) using 1 FPC cable (at J7) while Stereo sensor board is connected to Main Board using two FPC cables (at J6 and J9).

By default FPD link board (Vision Adapter board) is disconnected with Main Board although it is connected to power board. At given instance of time either stereo board (board bearing stereo sensors) or FPD link board can be connected to main board. To connect FPD link board to main board, one needs to open the box and connect data cables carefully using twisted FPC cables. Example shown in the figure below below



Important Note: FPD link board is not supported in current vision sdk release.

2.3.3 Hardware setup

Following h/w setup needs to be done. Just to showcase connections, MSF unit has been opened from back. User need not open it. (JTAG is optional)

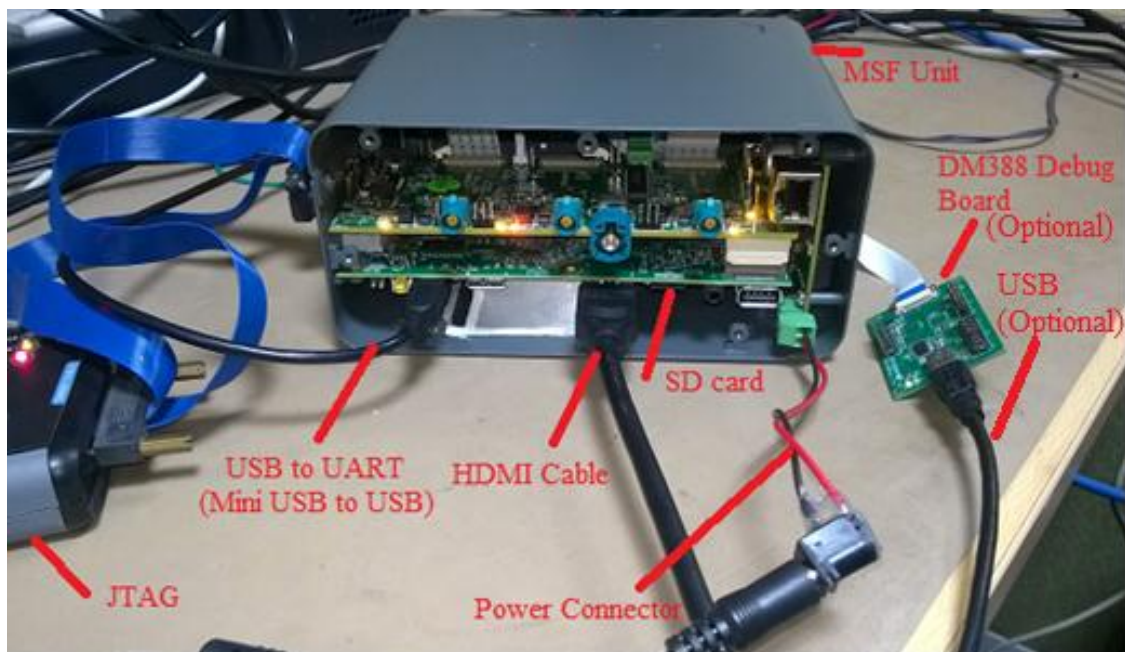


Figure 2 : MSF hardware setup

2.4 Software Installation

2.4.1 Windows

vision_sdk_xx_xx_xx_xx_setupwin32.exe is the SDK package installer.

Copy the installer to the path of your choice.

Double click the installer to begin the installation.

Follow the self-guided installer for installation.

IMPORTANT NOTE: On some computers running as administrator is needed. Right click on the installer and select option of "Run as administrator". If this is not done then you may see a message like "This program might not have installed correctly"

2.4.2 Linux

vision_sdk_xx_xx_xx_xx_setuptoolslinux.bin is the SDK package installer.

Download to INSTALL_DIR & execute

```
$> bash
$> export INSTALL_DIR=<Installation_directory_of_linux_installer>
$> cd $INSTALL_DIR
$> ./vision_sdk_xx_xx_xx_xx_setuptoolslinux.bin
```

Follow the installation instructions.

On completion of installation a folder by name VISION_SDK_xx_xx_xx_xx would have got created in the installation path.

2.4.3 Uninstall Procedure

To uninstall, double click on uninstall.exe created during installation in the folder VISION_SDK_xx_xx_xx_xx.

At the end of uninstall, VISION_SDK_xx_xx_xx_xx folder still remains. It is just an empty folder. It can be deleted manually.

3 Build and Run

This chapter provides an overview on, how to build and run bootloader (MLO) & application image (AppImage). The chapter mainly describes windows based build environment commands,

For Windows ensure BUILD_OS is selected as Windows_NT in Rules.make

```
BUILD_OS := Windows_NT
```

For Linux ensure BUILD_OS is selected as Linux in Rules.make

```
BUILD_OS := Linux
```

In case of Linux use "make" instead of "gmake", rest all commands / variables applies same as mentioned for windows in Rules.make

3.1 Building the MLO

On windows command prompt, go inside the directory
VISION_SDK_xx_xx_xx_xx\vision_sdk.

1. Open Rules.make in any edit and ensure

```
.....
ifeq ($(VSDK_BOARD_TYPE), )
    VSDK_BOARD_TYPE := TDA2XX_MC
endif
.....
```

Under vision_sdk directory run the command

```
$> gmake -s sbl_sd
```

3.1.1 Generating MLO

Go to VISION_SDK_xx_xx_xx_xx\vision_sdk\build\scripts

Edit sbl_mlo_create_tda2xx-mc.bat to correct paths

Execute sbl_mlo_create_tda2xx-mc.bat

MLO should be generated to VISION_SDK_xx_xx_xx_xx\vision_sdk\build\scripts\
mlo_tda2xx-mc folder

3.2 Building the application

On windows command prompt, go inside the directory
VISION_SDK_xx_xx_xx_xx\vision_sdk.

1. Open Rules.make in any edit and ensure

```
.....
ifeq ($(VSDK_BOARD_TYPE), )
    VSDK_BOARD_TYPE := TDA2XX_MC
endif
.....
```

2. Select TDA2XX's processors needed for your usecase / application

```
#
```

```
# Change below to include or exclude certain core's
#
PROC_DSP1_INCLUDE=yes
PROC_DSP2_INCLUDE=yes
PROC_EVE1_INCLUDE=yes
PROC_EVE2_INCLUDE=yes
PROC_EVE3_INCLUDE=yes
PROC_EVE4_INCLUDE=yes
PROC_A15_0_INCLUDE=yes
PROC_IPU1_0_INCLUDE=yes
PROC_IPU1_1_INCLUDE=yes
```

Select "no" for processor that are not needed.

e.g. for simple DM388 Capture + TDA2xx display usecase You need only IPU1_0 & A15_0 as "yes". This will save build time.

3. Save Rules.make after making changes

Build is done by executing gmake.

"gmake" is present inside XDC package. For "gmake" to be available in windows command prompt, the XDC path must be set in the windows system path.

IMPORTANT NOTE: xdc path is needed to be set in environment variables. If not, then set it using the set PATH =
<Install_dir>/ti_components/os_tools/windows/xdctools_x_xx_xx_xx;%PATH% in command prompt

Under vision_sdk directory run the command

```
$> gmake -s all
```

Executing above will automatically clean up previous builds. It will then build all the necessary components (Starterware, BSP drivers, EDMA drivers) and then the Vision SDK framework and examples.

On a successful build completion, following executables will be generated in the below path

```
\vision_sdk\binaries\vision_sdk\bin\tda2xx-mc
vision_sdk_arp32_1_release.xearp32F
...
...
vision_sdk_arp32_4_release.xearp32F
vision_sdk_c66xdsp_1_release.xe66
```

```
...  
...  
vision_sdk_c66xdsp_2_release.xe66  
vision_sdk_ipul_0_release.xem4  
vision_sdk_ipul_1_release.xem4  
...
```

If need be, incremental build can be done by following command

```
$> gmake -s
```

To speed up the incremental builds the following can be done as required

The build time can be made faster by following below steps.
in Rules.make set

```
BUILD_DEPENDANCY_ALWAYS = no
```

IMPORTANT NOTE: However make sure to do "gmake -s depend" before "gmake -s" in the following cases.

- when number of processors enabled is changed in Rules.make
- when DDR_MEM value in Rules.make is changed
- when NDK_PROC_TO_USE to use is changed
- when any .h or .c file in TI component is installed in ti_components is changed
- when any new TI component is installed in ti_components

If this "gmake -s depend" not done then build or execution may fail.

Cleaning the build can be done by following command

```
$> gmake -s clean
```

3.2.1 Generating AppImage

Go to VISION_SDK_xx_xx_xx_xx\vision_sdk\

Edit MulticoreImageGen_tda2xx-mc.bat to ensure you have only paths for binaries of processors you mentioned "yes" in Rules.make also ensure profile mentioned matches with Rules.make

Execute MulticoreImageGen_tda2xx-mc.bat

AppImage should be generated in folder VISION_SDK_xx_xx_xx_xx\vision_sdk\binaries\vision_sdk\bin\tda2xx-mc\sbl_boot

3.3 Boot modes

Following boot modes are supported on MSF

SD

QSPI

Similar to TDA2XX EVM users can also choose to load using CCS to avoid building MLO, use JTAG for further run and debug

3.4 Load using SD card

1. Copy generated MLO and AppImage on the 2 GB / 4 GB FAT32 formatted bootable SD card. (Refer VisionSDK_UserGudie_TDA2xx.pdf for flashing procedure).

2. Insert the SD in SD card slot on main board (near HDMI connector)

3.5 Load using QSPI

3.5.1 Steps to generate qspi writer tools

NOTE: SBL qspi image is built from starterware package.

To build qspi Run the command **gmake -s sbl_qspi** from vision_sdk root dir

And run the **sbl_qspi_create.bat** placed at vision_sdk\build\scripts

This generates all required tools under vision_sdk\build\scripts\qspi

1. sbl_a15host_release.xa15fg
2. qspiFlashWriter_m4_release.xem4
3. sbl_qspi

(NOTE: QSPI should be built for TDA2XX_MC as board type)

3.5.2 Generating AppImage

Go to VISION_SDK_xx_xx_xx_xx\vision_sdk\

Edit MulticoreImageGen_tda2xx-mc.bat to ensure you have only paths for binaries of processors you mentioned "yes" in Rules.make also ensure profile mentioned matches with Rules.make

Execute MulticoreImageGen_tda2xx-mc.bat

AppImage should be generated in folder VISION_SDK_xx_xx_xx_xx\vision_sdk\binaries\vision_sdk\bin\tda2xx-mc\sbl_boot

3.5.3 Qspi Pin Settings and steps

Qspi pin settings:

SW2[1] =0

SW2[2] =1 booting from QSPI

1. Connect A15.

Select CortexA15_0, navigate to Scripts->DRA7xx MULTICORE Initialization
DRA7xx_MULTICORE_EnableALLCores

2. Connect M4 (IPU)

Halt A15 core, and Load image on M4

**C:\VISION_SDK_XX_XX_XX_XX\vision_sdk\build\scripts\qspi\
qspiFlashWriter_m4_release.xem4**

Run the core.

Console outputs

[Cortex_M4_IPU1_C0]

QSPI Flash writer application

MID - 1

DID - 18

Enter the File Name **C:**

VISION_SDK_XX_XX_XX_XX\vision_sdk\build\scripts\qspi\sbl_qspi

Enter the Offset in bytes (HEX) **0x00**

Erase Options:

0 -> Erase Only Required Region

1 -> Erase Whole Flash

2 -> Skip Erase

Enter Erase Option: **1**

Load Options:

0 -> fread using code (RTS Library)

1 -> load raw using CCS (Scripting console)

Enter Load Option: **0**

Read xxxxxx bytes from [100%] file...Done.

QSPI whole chip erase in progress

QSPI file write started

******QSPI flash completed sucessfully******

3. Reset the board and Repeat step 1 and 2.

Console outputs

[Cortex_M4_IPU1_C0]

QSPI Flash writer application

MID - 1

DID - 18

Enter the File Name

C:\VISION_SDK_XX_XX_XX_XX\vision_sdk\build\scripts\qspi\sbl_qspi

Enter the Offset in bytes (HEX) **0x00**

Erase Options:

0 -> Erase Only Required Region

1 -> Erase Whole Flash

2 -> Skip Erase

Enter Erase Option: **2**

Load Options:

0 -> fread using code (RTS Library)

1 -> load raw using CCS (Scripting console)

Enter Load Option: **0**

Read xxxxxx bytes from [100%] file...Done.

QSPI file write started

*****QSPI flash completed sucessfully*****

4. Reset the board and Repeat step 1 and 2.

[Cortex_M4_IPU1_C0]

QSPI Flash writer application

MID - 1

DID - 18

Enter the File Name

**c:\VISION_SDK_XX_XX_XX_XX\vision_sdk\binaries\vision_sdk\bin\tda2x
x-evm\sbl_boot\AppImage_BE**

Enter the Offset in bytes (HEX): **0x80000**

Erase Options:

0 -> Erase Only Required Region

1 -> Erase Whole Flash

2 -> Skip Erase

Enter Erase Option: **2**

Load Options:

0 -> fread using code (RTS Library)

1 -> load raw using CCS (Scripting console)

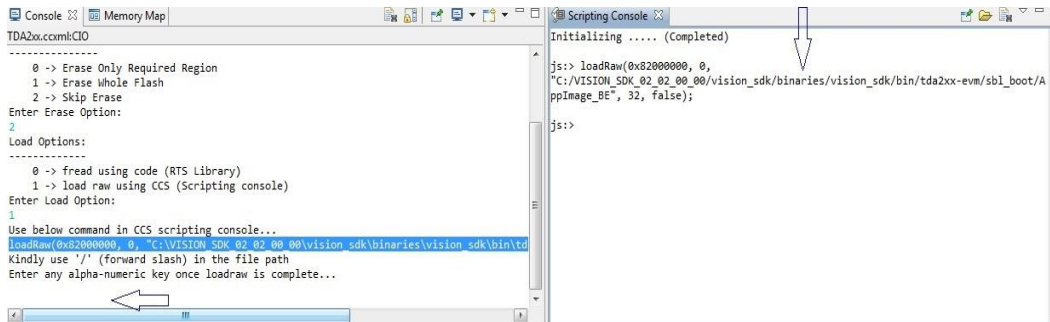
Enter Load Option: **1**

Open Scripting console window by clicking "Menu -> View -> Scripting console" and enter below command on scripting console as shown 3.5.3.1

`loadRaw(0x82000000, 0, "C:/VISION_SDK_XX_XX_XX_XX/vision_sdk/binaries/vision_sdk/bin/tda2xx-evm/sbl_boot/AppImage_BE", 32, false);`

In CCS console Enter any alpha-numeric key once loadraw is complete... as shown in 3.5.3.1

CCS console and scripting console



QSPI file write started

*******QSPI flash completed successfully*******

Power reset the board to boot from qspi.

3.6 Load using CCS

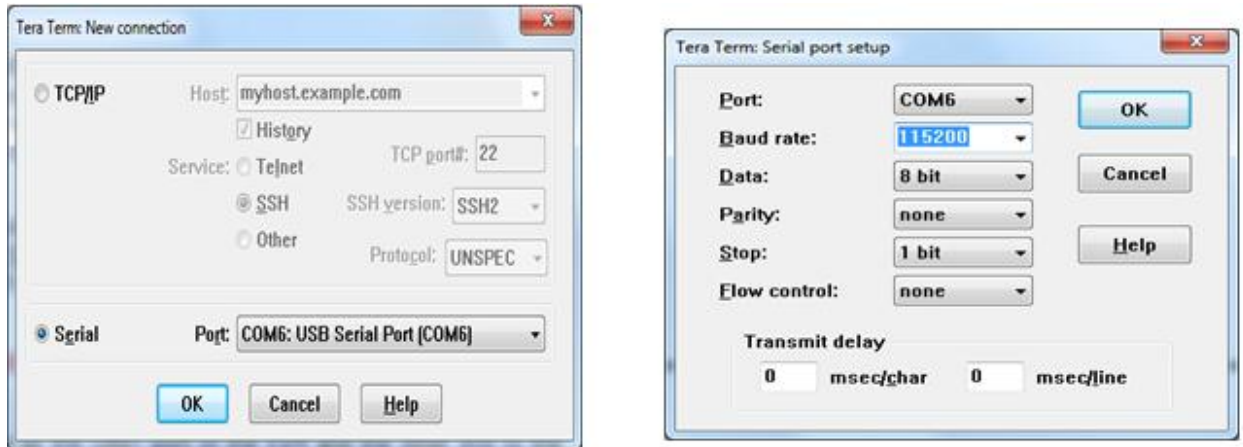
CSP Package for tda2xx-mc is kept under VISION_SDK_XX_XX_XX_XX/ti_components/ccs_csp/tda2xx install this package and ensure following in TDA2xx_ddr_config.gel

```
Set #define IS_EMIF2_AVAILABLE          (1)
Set #define MEMMAP_1GB_SINGLE_EMIFX1    (1)
Set #define MONSTER_CAM                  (1)
Also set ENABLE_ECC                      (0)
```

3.7 Running application

Please refer [section 4.1](#) before trying any single channel usecases

1. Power on the board with specified power adapter (12v, 5A), with a boot ready SD card inserted ([section 3.3](#)) or with images flashed in QSPI ([section 3.4](#))
2. Press yellow reset button (near Mini USB connector (USB->UART))
3. On PC, select the correct COM port in TeraTerm's connection with following settings



IMPORTANT NOTE: If you observe UART terminal does not response, Updating the USB to UART driver on PC may make it work on the failings PCs. You can download the drivers from the below link.

<http://www.ftdichip.com/Drivers/VCP.htm>

<http://www.ftdichip.com/Drivers/CDM/CDM%20v2.10.00%20WHQL%20Certified.exe>

6. Upon successful boot up you should see Usecase Menu on the console

7. Only single channel usecase and stereo usecase are valid for MSF, you need to input usecase number in order to run that usecase.

IMPORTANT NOTE: If you are running any single channel Capture + Display usecase on MSF, ensure you wait for 1 minute to allow DM388 to complete booting and start streaming. Tda2xx is booting Sysbios but dm388 is booting Linux. Select usecase number after this.

Rest all the steps are exactly same as TDA2XX EVM and they are motioned in VisionSDK_UserGuide_TDA2xx.pdf

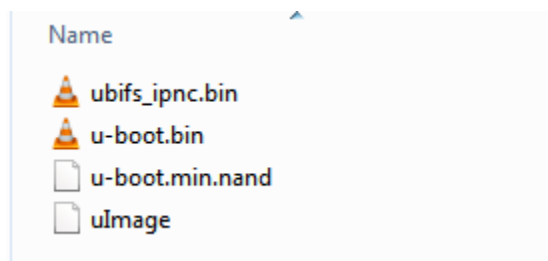
4 DM388 boot up and changing default binaries

DM388 is powered up by TDA2XX main board. In the boot up sequence TDA2XX issues a reset to DM388 through GPIO. It means DM388 boots up after TDA2XX in MSF. By default, binaries for DM388 are flashed into DM388's NAND and there is no need to change these binaries. These are Linux binaries (MLO, u-boot and uImage) along with file system (UBIFS) that contains DM388's firmware. DM388 streams the captured video over VOUT1 and TDA2xx receives it over VIN2A port.

DM388 board in MSF supports NAND only. For SD boot there is hardware modification needed as there is no switch to select the boot mode. It is not encouraged to use SD boot mode for DM388, instead NAND + Ethernet should suffice.

4.1 Default DM388 binaries

Binaries package corresponding to this VisionSDK release is available in pre-built binaries package VisionSDK_pre-built_binaries/dm388/VISION_SDK_MSF_DM388_binaries_01.00.00.00.zip. After unzipping this you should see following contents



You need to ensure these binaries (especially filesystem) are flashed into DM388's NAND, without which, on Rev B.1 MSF, you may see inverted image. You need to change these binaries before you run any algorithms on single channel usecases to get right output.

4.2 Changing default DM388 binaries (file system)

Vision SDK does not include software release for DM388 but in case if these binaries need to be changed (most like when you want to tune the image), steps below can be followed.

1. Connect the debug board to MSF through external FPC cable and mini-USB to USB cable to PC. It is shown in "Hardware setup section" 2.3.3. **Also connect Ethernet cable to MSF to the RJ45 port near TDA2xx's JTAG connector (That is DM388's Ethernet port).**

Note: If you don't have debug board contact FAE to get one.

2. Power up MSF and press reset button (refer section 3.3 "Running Application")
3. Open another tera term connection with right COM port, (one you already opened for TDA2xx UART)

4. At times you may have to plug out and plug in DM388 USB cable connected to debug board, with this you should see very small LED blinking on the debug board and characters on the DM388's UART console
5. *Remember to flash images in NAND you need to stop DM388 at u-boot.*
6. With this connection if you see DM388 has gone ahead and booting kernel press yellow reset button again, you will see DM388 booting again and hit enter at uboot. In either case, you should see following console on first 6 successful steps.

[illegible]

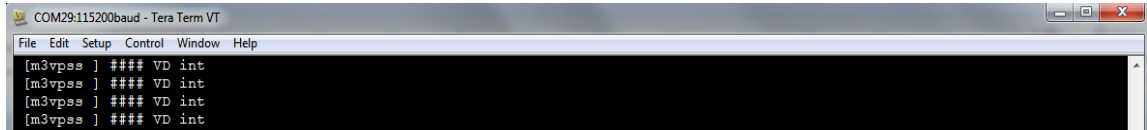
7. You need to setup a tftp server on your PC, where you can keep updated image
8. Use following commands to flash ubifs image to NAND

```
DM388_IPNC# setenv serverip <ip_of_pc_where_tftp_is_running>
DM388_IPNC# setenv dnsip <dns_server_ip_on_your_network>
DM388_IPNC# setenv dnsp2 <dns_server_ip_on_your_network>
DM388_IPNC# setenv gatewayip <gatewayip_on_your_network>
DM388_IPNC# setenv netmask 255.255.252.0
DM388_IPNC# setenv ipaddr <any_static_ip>

DM388_IPNC# nand scrub 0x006C0000 0x7940000;mw.b 0x81000000 0xFF
0x50000000
```

```
DM388_IPNC# tftp 0x81000000 ubifs_ipnc.bin;nand erase 0x006C0000
0xB940000;nand write 0x81000000 0x006C0000 0x5000000
DM388_IPNC# boot
```

To conclude DM388 is booting properly look for following continuous print on DM388's console



If you hit enter here you should see **root@DM385_IPNC** as console, VD_int should keep coming.

Important Note: Most likely you need not update the MLO, uboot and uImage for DM388. Even in case of tuning of main imager you will have to update just ubifs image (file system).

4.2.1 Flashing uboot and kernel images to NAND (optional)

Default flashed binaries should suffice. In case you need to update these binaries please refer commands below

Uboot MIN

```
DM388_IPNC# tftp 0x81000000 u-boot.min.nand;nand erase 0x0 0x20000;nand
write.i 0x81000000 0x0 0x20000
```

Uboot 2nd stage

```
DM388_IPNC# tftp 0x81000000 u-boot.bin;nand erase 0x20000 0x60000;nand
write.i 0x81000000 0x20000 0x60000
```

Kernel

```
DM388_IPNC# tftp 0x81000000 uImage;nand erase 0x00280000 0x00300000;nand
write.i 0x81000000 0x00280000 0x300000
```

If you change any of the "bootargs" use saveenv command to save these to nand flash. "printenv" is used print the current bootarg values.

5 Supported usecases on MSF

5.1 Single channel usecases

In all single channel usecases capture is happening on DM388 and output of an algorithm is displayed through HDMI on TDA2XX.

To run the usecase select respective option from runtime menu.

Since MSF doesn't have HDMI input to stream in video content, usecases that need HDMI input are not supported from Runtime Menu.

5.2 Stereo

Usecases in runtime menu under "Stereo Use-cases, (TDA2x MonsterCam ONLY)" are supported

Important Note: Before running any stereo usecase it is mandatory to calibrate stereo cameras to get best results from the algorithm. There is a calibration usecase supported by vision_sdk that helps user calibrate MSF for stereo. Refer VisionSDK_MultiSensorFusionStereoCalibrationGuide.pdf in docs folder in vision_sdk package.

For option / usecase "Network capture + Stereo (DSPx, EEx) + PD+TSR+LD+SOF (DSPx, EEx) + Display (HDMI)", ensure you select OPPMODE=opp_high and EMIFMODE=DUAL_EMIF_2X512MB in vision_sdk/build/makerules/build_sbl.mk as shown below

```
SBL_TDA2XX_MC_OPTIONS= $(SBL_COMMON_OPTIONS) PLATFORM=tda2xx-mc  
EMIFMODE= DUAL_EMIF_2X512MB FORCE_OPPMODE=TRUE OPPMODE=opp_high
```

Refer VisionSDK_UserGuide_TDA2xx for building MLO after this.

6 Frequently Asked Questions

6.1 Build, install, load, PM related FAQs

Refer VisionSDK_Userguide_TDA2XX.pdf for detailed FAQs

6.2 MSF run related FAQs

Q: Output of the stereo usecase is noisy, what can be done?
Ans: Ensure you have calibrated stereo sensors for the unit that you have. Stereo algorithm works on right and left images captured from stereo sensors and calculates disparities. If cameras are not calibrated the entire output won't be a correct depth map.
Q: When I run main imager (single channel) usecases, nothing is working, what could be wrong?
Ans: MSF unit that you have received will have default DM388 binary flashed in to NAND. This might be old. To rectify this first run single channel capture + display usecase (without any algorithm), observe the output on HDMI display if you see an inverted image or misalignment of the video on the screen, you need to update DM388's binary (only file system i.e. ubifs image) into DM388's NAND. The binary can be found in VisionSDK_pre-built_binaries package and procedure to flash is documented in section 4.2
Q: When I connect mini USB to USB to my PC, I don't see anything on TeraTerm/HyperTerminal
Ans: Most likely you have missed installing FDTI chip drivers refer section 2.2.1
Q: Quality of Main Imager output is not very good; colors are not very accurate, what can be done?
Ans: DM388 has not been tuned for production quality output; this can still be done through DCC tuning tool on PC based over Ethernet. Sensor tuning is a very involved activity and TI recommends approaching third parties for the same.
Q: When I run capture + display usecase I see a grey screen for a minute or so, why?
Ans: DM388 has not started capturing data. Basically DM388 boots linux while Tda2xx boot bios, TDAXX displays as grey frame until valid data is streamed by DM388 over video ports

7 Directory Structure Details

Refer VisionSDK_Userguide_TDA2XX.pdf

8 Revision History

Version	Date	Revision History
1.0	17th February 2015	Initial Version
2.0	25 th February 2015	Added DM388 binary details & corrections in build
3.0	3 rd March 2015	Review comments implemented & added FAQs
4.0	3 rd July 2015	Calibration usecase updates

« « « § » » »