	Controlled version is located in the project repository. Printed versions are uncontrolled unless stamped “Controlled Copy” in RED.

iSSD v2.0

(NVMe based intelligent SSD Card)

Product Requirement ID	N/A
Version	A00-08
Date	10-05-2017
iCAP Classification	NXP Internal

REVIEW / OFFICIAL SIGN-OFF

Role	Name	Date

Acronyms and Definitions

ACRONYMS	DEFINITIONS
NVMe	Non-Volatile Memory Express
LS2	Layer Scape 2088a
FPGA	Field Programmable Gate Array
PCIe	Peripheral Component Interconnect Express
DIMM	Dual in line memory module

Table of Contents

List of Figures

List of Tables

1 Introduction

This User Manual is prepared for the reference to the Freescale, to work with the LS2088a NVMe based iSSD card for the initial delivery of the board support package.

1.1 Scope of the Setup

This document describes how to set up the **LS2088a NVMe based iSSD in** order to reproduce and validate the functionality of the product.

This document specifically helps in understanding the following sections,

- Software setup
- Hardware setup
- Booting the setup
- validating the setup
- Test Procedure of the interfaces present.

1.2 High level design

This is a two board solution where LS2088a is in the main board which is described as **iNIC** card whereas FPGA and the NVDIMMs are sitting on **Storage Card**, and the setup completely referred as **iSSD** card.

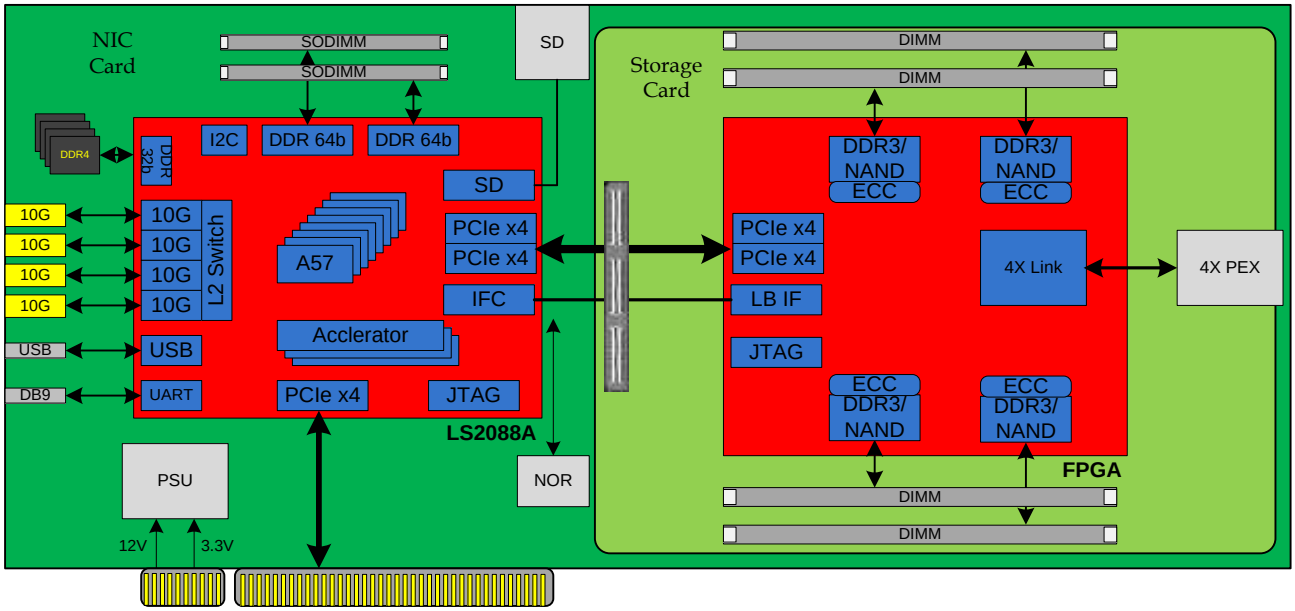


Figure 1: Block Diagram of iSSD

LS2 device interfaces utilized in the design:

- 4 x 10G SFP+
- 1 x USB IF
- 1 x 4-lane PCIe 3.0 end point for host interface (to communicate with host PC)
- 2 no. of x4 lane PCIe 3.0 Root complex interface (to interface NVMe Controller in FPGA)
- 1 x UART Interface for system console / debug
- 1 x SD card Interface
- 2 x DIMM for DDR4 interface

FPGA interfaces required in FP utilized in the design:

- 2 no. of x4 PCIe3.0 endpoint (for LS2 interface)
- 4 x DIMM support for DDR3/NAND controller interface

Figure 2: Front side assembly of the LS2088a iNIC card.

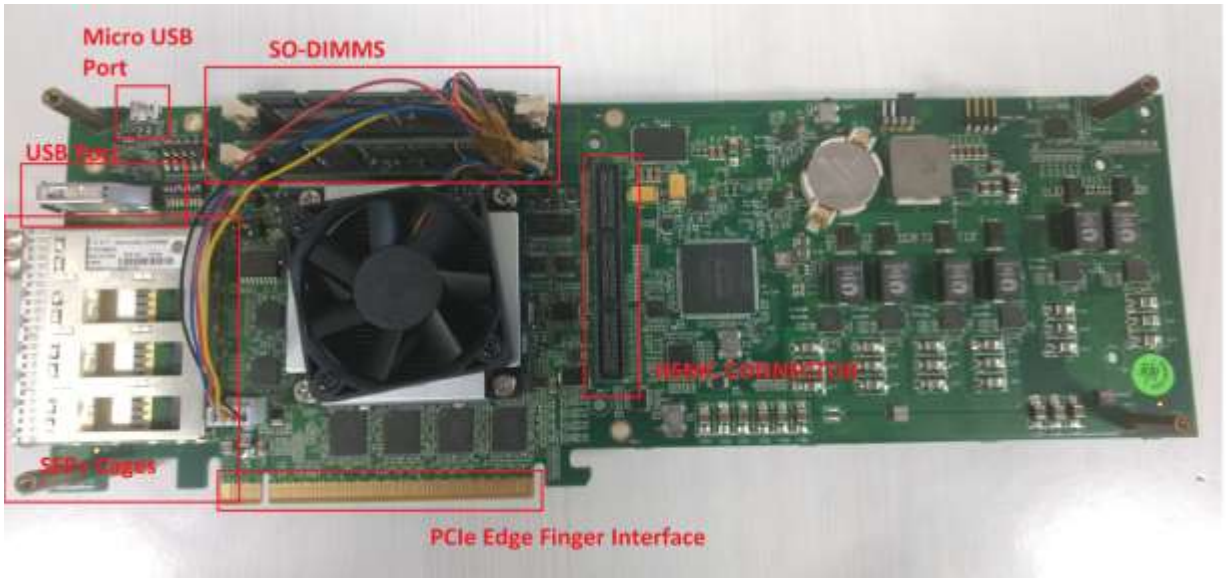


Figure 3: Interfaces in the LS2088a iNIC card

Figure 4: Back side assembly of the LS2088a iNIC card.

Figure 5: Front side assembly of the storage card

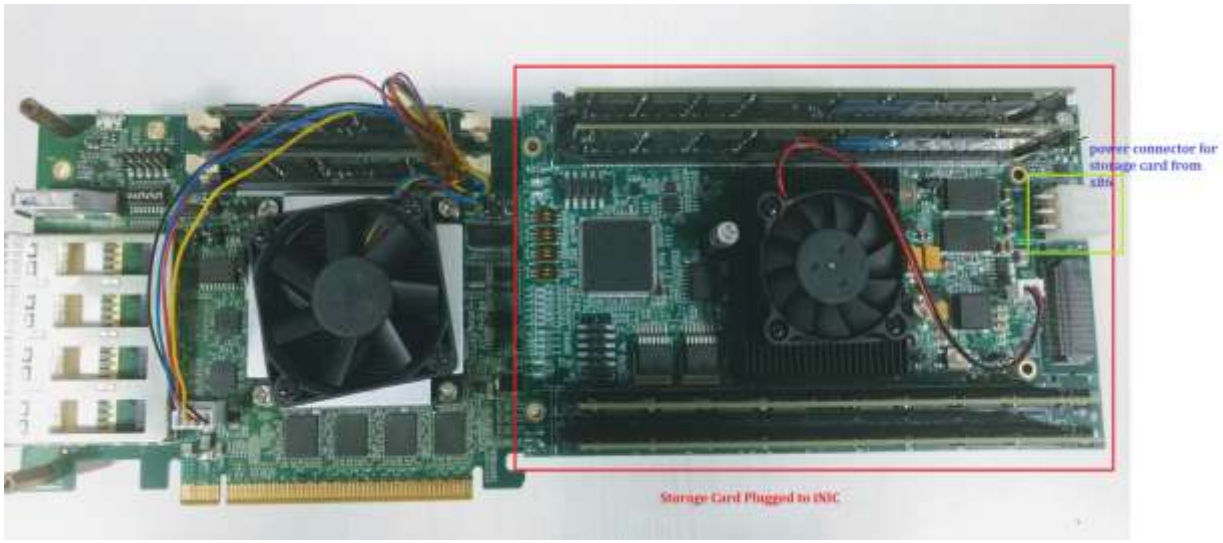


Figure 6: Complete iSSD card where storage card sits on the LS2088a iNIC card

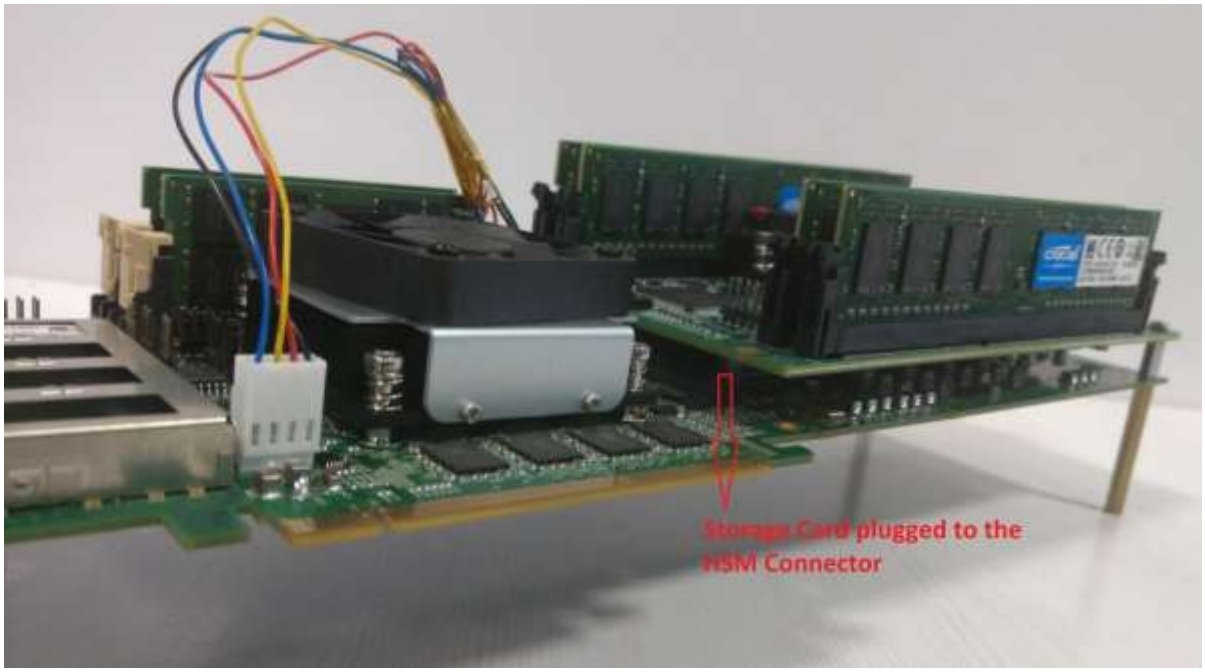


Figure 7: iSSD card side view

2 Hardware Setup

2.1 Hardware inventory

The following hardware inventories are required for the setup along with the iSSD card

- ATX12V (Seasonic) power supply/ x86system with PCI slot of the 1x16 lane.
- SD/SDHC card/ USB3.0.
- Micro UART cable for Serial console.
- Four no. of SFP+ transceiver and four pair of optical cables for each SFP+ module. The recommended SFP+ transceiver is **FTLX8571D3BCL 15-05 and the Optical Fiber part no is 12R9914.**
- Linux(x86) laptop/ PC for connecting to the serial console.
- DDR3 DIMM Module. Recommended DIMM module is **CT102472BA160B, Manufacturer – Crucial.**

2.2 Setting up LS2088a iSSD

2.2.1 Recommended Header and Switch settings in the iNIC card

S.No	Header	Description	Default state
1	**H1**	**1-2 short = RTC use Battery supply** **3-4 Short => Sourcing PCIe Clock from EDGE FINGER;** 3-4 Open => Sourcing PCIe Clock from ON BOARD	1-2: Short 3-4: Short
2	**H2**	IRF PWM controller programming header **Cable will connect to this Header for programming PWM controller**	Open
3	**H4**	VCC CPU CORE **Open - Do Not Sense**	open
4	**H5**	**1-2 Shorted => Boot from NOR BANK 4**; 1-2 Open => Boot from NOR BANK 0	Short
5	**H8**	TA PROGRAM MTR **Open – Default**	Open
6	**H9**	TA PROGRAM SFP+ **Open – Default**	Open
7	**H10**	**1-2 Open => USB HOST MODE**	Open
8	**H11**	**1-2 Shorted => RTC Clock selection** 2-3 = Open	1-2 Shorted

9	**SW2**	**NOR Flash Boot** **SW2.1 => OFF** **SW2.2 => ON** **SW2.3 => OFF** **SW2.4 => ON** **SW2.5 => OFF** **SW2.6 => ON** **SW2.7 => ON** **SW2.8 => ON** I2C RCW Boot SW2.1 => OFF SW2.2 => ON SW2.3 => ON SW2.4 => OFF SW2.5 => ON SW2.6 => OFF SW2.7 => ON SW2.8 => ON QSPI Boot SW2.1 => ON SW2.2 => OFF SW2.3 => ON SW2.4 => ON SW2.5 => OFF SW2.6 => OFF SW2.7 => ON SW2.8 => OFF	Recommended State NOR Flash BOOT: SW2.1 => OFF SW2.2 => ON SW2.3 => OFF SW2.4 => ON SW2.5 => OFF SW2.6 => ON SW2.7 => ON SW2.8 => ON

Table 1: Header and switch settings in iNIC card

2.2.2 Recommended Header settings in the Storage card

S.No	Header	Description	Default State
1	**H1**	Power Module M4 compensation pin	Open
2	**H2**	Power Module M7 compensation pin	Open
3	**H4**	Power Module M6 compensation pin	Open
4	**H5**	Power Module M5 compensation pin	Open
5	**H3**	Fan header. This connect to Heatsink connector	Connect to Heatsink

7	**J3**	CPLD JTAG Header – Used for programming CPLD	Open
8	**J6**	FPGA JTAG Header – Used for programming EPCQ flash	Open
9	**J4**	2x3, 12V Power input connector	Connect to ATX power connector
10	**J5**	2x31 connector. Used for connecting one storage card to another storage card through external cable.	Open
11	**J9**	HSMC Board to board connector. Used to connect with iNIC Card.	Connect to iNIC card

Table 2: Header settings in the Storage card

2.2.3 LED lights in iNIC card

Below are the Status LED's in the iNIC card,

LED no.	Color	Name	ON state	OFF state
D3	Green	HRESET_N	Asserted Hardware reset	Hardware reset is not asserted
D4	Green	Power status	Power sequence is fine	Power sequence is not fine
D5	Red	ASLEEP	LS2 is in sleep state	Ls2 is not in sleep state
D7	Red	Power status	Power supply to the board, failed	Power is Ok
D8	Green	12V supply	12V is available for the board	12V is not available for the board
D9,D10	Red	GPIO	LED's for debugging	

Table 3: Led lights present in the iNIC card and their state

2.2.4 DIMM Voltage Switch setting for SW1, SW2, SW3, SW4 in Storage Card

The following are Switches corresponds to DIMM connector.

Switch	DIMM connector

SW1	M4
SW2	M2
SW3	M1
SW4	M3

Table 4: Switches for the DIMM connectors

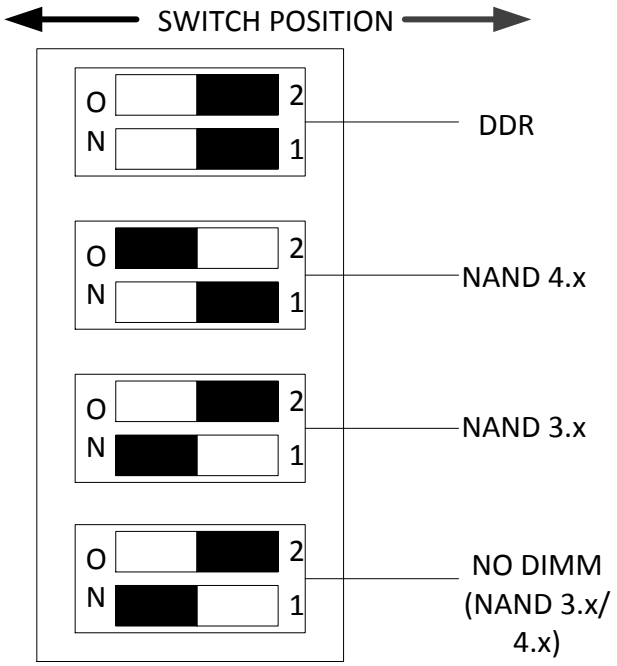


Figure 8:Switch position for selecting DIMM module

2.2.4.1 Recommended switch settings for DDR3

S.No	Header	Description	DIMM Module Type
1	SW1	SW1.1 = OFF SW1.2 = OFF	DDR3
2	SW2	SW2.1 = OFF SW2.2 = OFF	DDR3
3	SW3	SW3.1 = OFF SW3.2 = OFF	DDR3
4	SW4	SW4.1 = OFF SW4.2 = OFF	DDR3

Table 5: Switch Settings for DDR3

2.2.4.2 Recommended switch setting for NAND DIMM ONFI 4.X

S.No	Header	Description	DIMM Module Type
1	SW1	SW1.1 = OFF SW1.2 = ON	NAND DIMM ONFI 4.X
2	SW2	SW2.1 = OFF SW2.2 = ON	NAND DIMM ONFI 4.X
3	SW3	SW3.1 = OFF SW3.2 = ON	NAND DIMM ONFI 4.X
4	SW4	SW4.1 = OFF SW4.2 = ON	NAND DIMM ONFI 4.X

Table 6: Switch Settings for ONFI 4.X

2.2.4.3 Recommended switch setting for NAND DIMM ONFI 3.X

S.No	Header	Description	DIMM Module Type
1	SW1	SW1.1 = ON SW1.2 = OFF	NAND DIMM ONFI 3.X
2	SW2	SW2.1 = ON SW2.2 = OFF	NAND DIMM ONFI 3.X
3	SW3	SW3.1 = ON SW3.2 = OFF	NAND DIMM ONFI 3.X
4	SW4	SW4.1 = ON SW4.2 = OFF	NAND DIMM ONFI 3.X

Table 7: Switch Settings for ONFI 3.X

2.2.4.4 Recommended switch setting for MRAM

S.No	Header	Description	DIMM Module Type
1	SW1	SW1.1 = ON	MRAM

		SW1.2 = ON	
2	SW2	SW2.1 = ON SW2.2 = ON	MRAM
3	SW3	SW3.1 = ON SW3.2 = ON	MRAM
4	SW4	SW4.1 = ON SW4.2 = ON	MRAM

Table 8: Switch Settings for MRAM

2.2.5 LED lights in the Storage card

2.2.5.1 LED'S connected to CPLD

S.NO	LED	SILKSCREEN	DESCRIPTION
1	D10	LED 7	POWER FAIL
2	D11	LED 8	Configuration done
3	D12	LED 9	Thermal even of DDR
4	D13	P_GD	Power Good
5	D14	FLT_M1	Power fail M1
6	D3	FLT_M2	Power fail M2
7	D15	FLT_M3	Power fail M3
8	D1	FLT_M4	Power fail M4

Table 9: LED connected to CPLD

2.2.5.2 LED'S connected to FPGA

S.NO	LED	SILKSCREEN	DESCRIPTION

1	D4	LED 1	TBD
2	D5	LED 2	TBD
3	D6	LED 3	TBD
4	D7	LED 4	TBD
5	D8	LED 5	TBD
6	D9	LED 6	TBD

Table 10: LED connected to FPGA

2.2.5.3 POWER LED

S.NO	LED	SILKSCREEN	DESCRIPTION
1	D2	PWR_IN	VCC_AUX_12V

Table 11: POWER LED

2.2.6 Basic Setup for iSSD

2.2.6.1 Setup for NVMe iSSD

- In iNIC card, a jumper need to connected as shown below at PCI edge interface header **H1** to power up the iSSD through x86 Machine.

Figure 9: PCIe CLK selection Jumper

- Plug the storage card into the iNIC through HSMC connector

Note: Refer the Appendix 6.4 for *Screw mounting of the storage card* into the iNIC card.

- Plug the power supply cable to the storage card from the x86 PC, into the power slot(Refer figure 6 for power location).
- Ensure that there is proper cooling fan for the board.
- Insert the SD card and USB into the SD/SDHC card and USB slot as shown in the Figure 10 and figure 3 respectively.

.

Figure 10: SD card slot

- Connect the Micro USB cable from the board to the Linux laptop/PC.

- According to the bank which the user intends to use, the jumper is connected to the board at position as shown below.
- User can plug in the iSSD card(NVMe storage card mounted on iNIC) in the **PCI slot of the x86** PC.
- Switch ON the x86 Machine

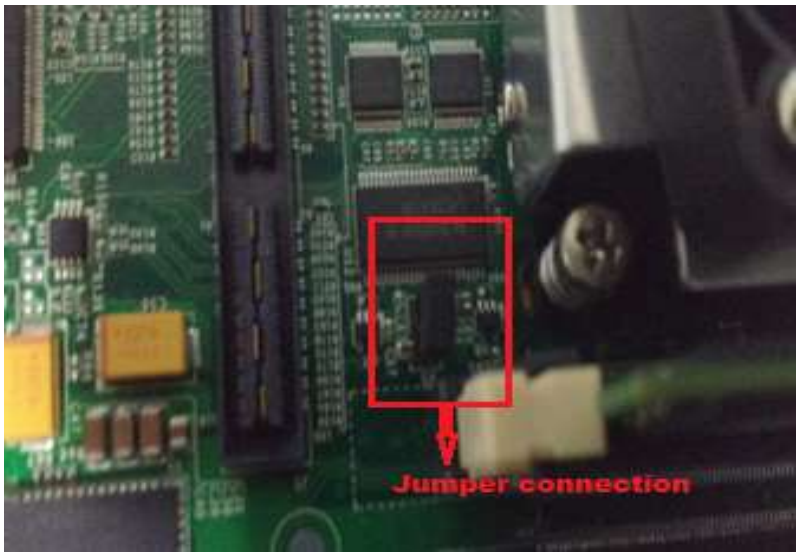


Figure 11: Header position to boot form 4

Note:

- Please use the bank 4 for flashing the upgraded images, bank 0 is used for the factory default and its Read only.
- If the user boot from Bank 0 with Un-bootable U-Boot image then, user cannot flash the bootable image into bank 0 through bank 4. So, it's recommended to use bank 4.
- If the Jumper is not connected, both the banks are accessible.

Precaution:

- Please Use Image **from Bank 4** and keep jumper inserted in **H5.**
- Remove Jumper from H5 only for Image upgrade in Bank4. This precaution will not allow user to erase Boot code in both Bank Accidentally.

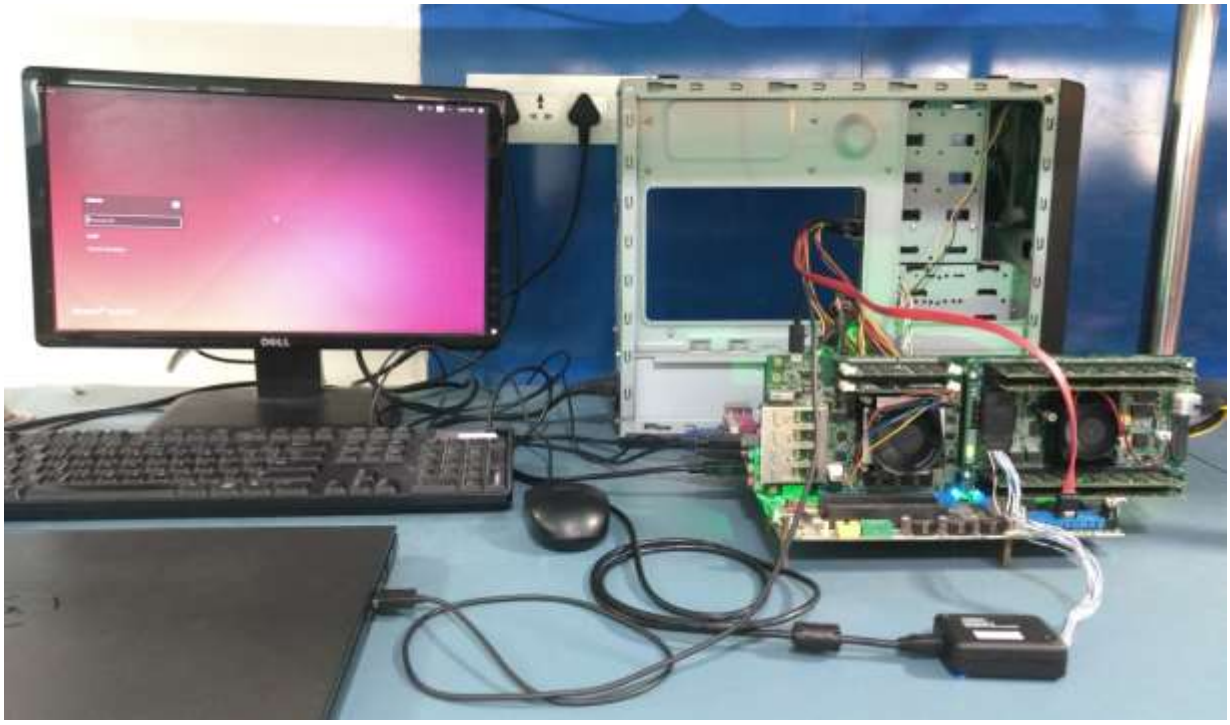


Figure 12: Complete set up of the NVMe iSSD

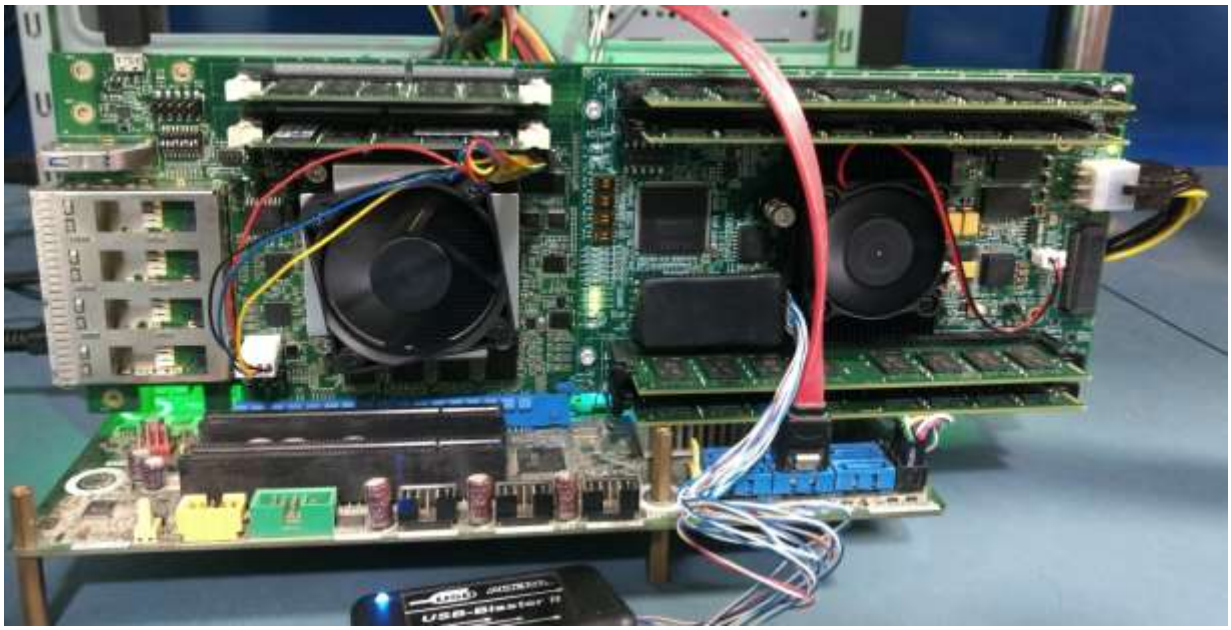


Figure 13: Front view NVMe setup of the LS2088a iSSD

Note: Check the board retention for secure setup(Refer Appendix section 6.5)

2.2.6.2 Set up for LS2088a iNIC functionality

This section describes the hardware setup with X86 machine as host and LS2088a card as iNIC card.

Figure 14: iNIC card plugged in x86 PCIe slot for iNIC functionality test

Follow the below procedure to make the hardware setup of the iNIC functionality,

- Do not power up the x86.
- Insert the LS2088a iNIC card properly in PCIe x16 slot of x86 Machine.
- Connect USB mini console cable to your laptop for LS2088 iNIC card console access.
- Insert SFP in the proper 10G port as shown in the figure.
- Connect 10G interface to some other 10G interface(x86) to do the ping test.

Note: Make sure Card is inserted properly

For test procedure and Test results, please refer LS2088_iNIC_Feature_Test_Guide.docx

2.2.6.3 Set up for the LS2088a iNIC RoCE ISER implementation

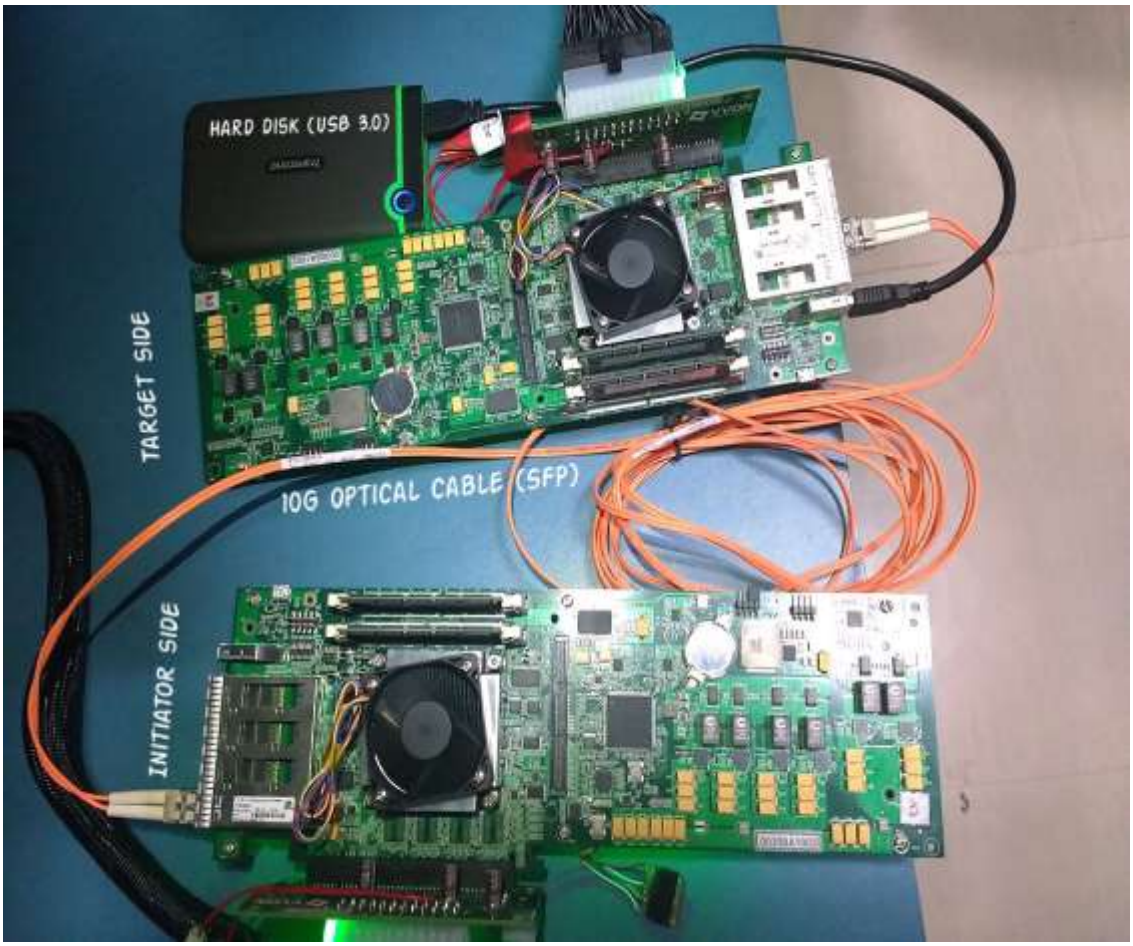


Figure 15: ISER implementation setup

2.2.7 Powering the iSSD card

2.2.7.1 Powering through the x86 systems

- User can plug the card into the x86 system so that once the x86 system is switched on, the iNIC PCI will be enumerated and the card is powered up.
- User can see the Power LED(D3) glowing up in **Green color** and LED(D5) in **Red** color glowing beside that, which ensures that board is powered up and initial RCW sequencer is happening. This is referred as auxiliary mode. The LED indication is shown below.

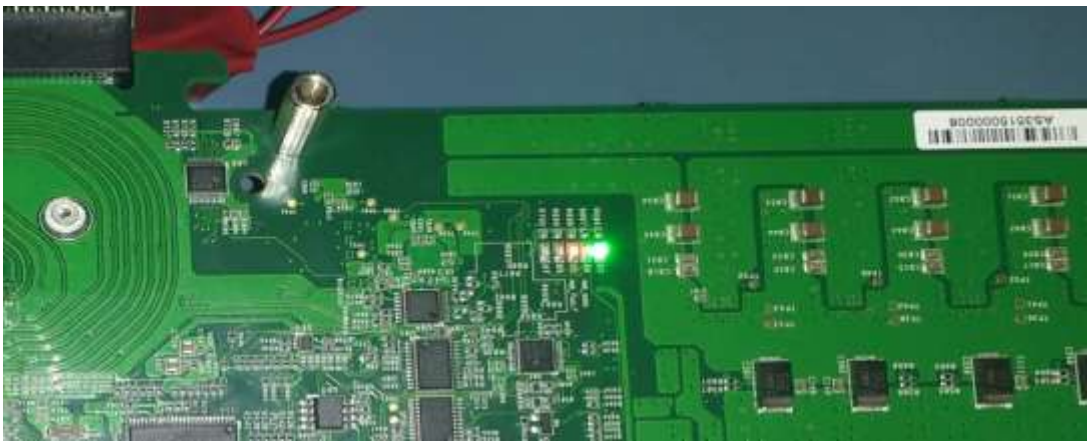


Figure 16:LED Status of the iSSD at initial power up

- If the board is not properly powered up, LED D7 will glow in **Red** color.
- Then LED(D8) will be glowing near the PCI edge connector and LED D5 in Red color glows off.This ensures that board is completely powered and RCW is loaded properly into the board and it will start booting and the LED status of the board is shown below,

Figure 17: LED status of the iSSD after complete powering up

2.2.7.2 Failures cases of the power up

POWER FAIL will glow in case of failure of power up as discussed in Table no.3

- CPLD is not properly programmed or powered up.
- Wrong RCW flashed into the NOR flash or RCW sequence failed.
- Incorrect RCW switch settings on the board (if so verify with Table no. 1).
- Jumper setting may not properly done, according to the type of the power supply (ATX/ x86 system).
- Jumper setting for the bank selection.
- Some on board voltage may be in poor condition.

2.2.8 After powering up the board

- Once the board is powered up it starts booting
- User can Hit any key to stop the auto boot to Linux and to access U-boot prompt or it will boot to the kernel.

Note:

If the user wants to power up the iSSD using the ATX supply and the PCIe power card, please refer the Appendix section 6.1 for the procedure of the setup.

3 Software Setup

3.1 Building the SDK from the ISO

To get the root privilege, issue the following command from the terminal

```
$ sudo visudo
```

Add following lines under “Members of admin group may gain root privileges”

```
%admin ALL=(ALL) ALL
%<user name> ALL=(ALL) ALL
<user name>  ALL = NOPASSWD: /usr/bin/apt-get
```

For example:

```
%madhu ALL=(ALL) ALL
madhu  ALL = NOPASSWD: /usr/bin/apt-get
```

Install the following package for squashfs support by issuing the following command,

```
$ sudo apt-get update
$ sudo apt-get install squashfs-tools
```

To build the SDK from the ISSD-SDK-V2.0-YYMMDD-yocto.iso, shall follow the bellow steps:

If .iso files are in two parts, then combine it using cat command:

cat ISSD-SDK-V2.0-YYMMDD-yocto.iso.part1 ISSD-SDK-V2.0-YYMMDD-yocto.iso.part2 >ISSD-SDK-V2.0-YYMMDD-yocto.iso

```
$ mkdir  /home</USER>/SDK-<version no>
$ mkdir  /home</USER>/SDK-<version no>/mnt
$ cp –r ISSD-SDK-V2.0-YYMMDD-yocto.iso /home</USER>/SDK-<version no>
$ cd /home</USER>/SDK-<version no>/
$ sudo mount –o loop ISSD-SDK-V2.0-YYMMDD-yocto.iso /home</USER>/SDK-<version no>/mnt/
$ cd /home</USER>/SDK-<version no>/mnt
$ ./install.sh
```

Note: It will ask for the installation path, user can specify the path as /home</USER>/SDK-<version no>

```
$cd /home</USER>/SDK-<version no>/<Layerscape path>
$ source fsl-setup-env –m ls2088aissd –j 24 –t 24
$ bitbake u-boot-mkimage-native
$ bitbake fsl-image-minimal
$ sudo ../issd-images 01:00:00
```

This will create issd_images folder in the image directory “tmp/deploy/images/ls2088aissd”

By default uimage in issd_images directory is ISSD uimage.

To build INIC uimage enable a macro (#define FSLU_NVME_INIC 1) in "tmp/work/ls2088aissd-fsl-linux/linux-ls2-sdk/4.0-r0/git/include/inic_config.h" file.

Note: By default this macro is disabled.

To build uimage_inic alone run these commands,

```
$bitbake -c compile virtual/kernel

$./tmp/work/x86_64-linux/u-boot-mkimage-native/v2015.01+gitAUTOINC+92fa7f53f1-r0/git/tools/mkimage -A arm64 -O linux -T kernel -C none -a 0x80080000 -e 0x80080000 -n "ISSD" -d tmp/work/LS2088aissd-fsl-linux/linux-qoriq/4.0-r0/build/arch/arm64/boot/Image
./tmp/deploy/images/LS2088aissd/issd_images/uimage_inic.bin > /dev/null
```

Now all the nine images will be presented in "tmp/deploy/images/ls2088aissd/issd_images" directory.

3.2 Firmware Upgrade Procedure

3.2.1 Generating issd-images tar

Images tar is used by firmware upgrade script (upgrade.sh) to flash images from linux. Images tar is named as issd-images_<version> (version given to create issd_images). Tar should contain all 10 images (refer Table 12 image naming format), flash_images.sh script and checksums.md5.

3.2.2 Steps to create issd-images tar

Follow the steps given below to create issd-images tar

```
$mkdir issd-images_01.00.00
$cd issd-images_01.00.00
$mkdir images
$cp tmp/deploy/images/ls2088aissd/issd_images/* ./images/
$cd ../
$find ./images/ -type f -print0 | xargs -0 md5sum > checksums.md5
$cp ..sources/meta-freescale/recipes-extended/merge-files/files/merge/usr/sbin/flash_images.sh .
$cd ../
$tar –cvzf issd-images_01.00.00.tar.gz issd-images_01.00.00
```

3.2.3 Flashing images from U-Boot

NOR flash is divided into two halves namely bank0 and bank4. It is possible to have different images in the two banks and boot one bank at a time, this will not affect other bank. Boot with jumper to work on bank4 (shown in the figure 18). Boot without jumper to work on bank0.

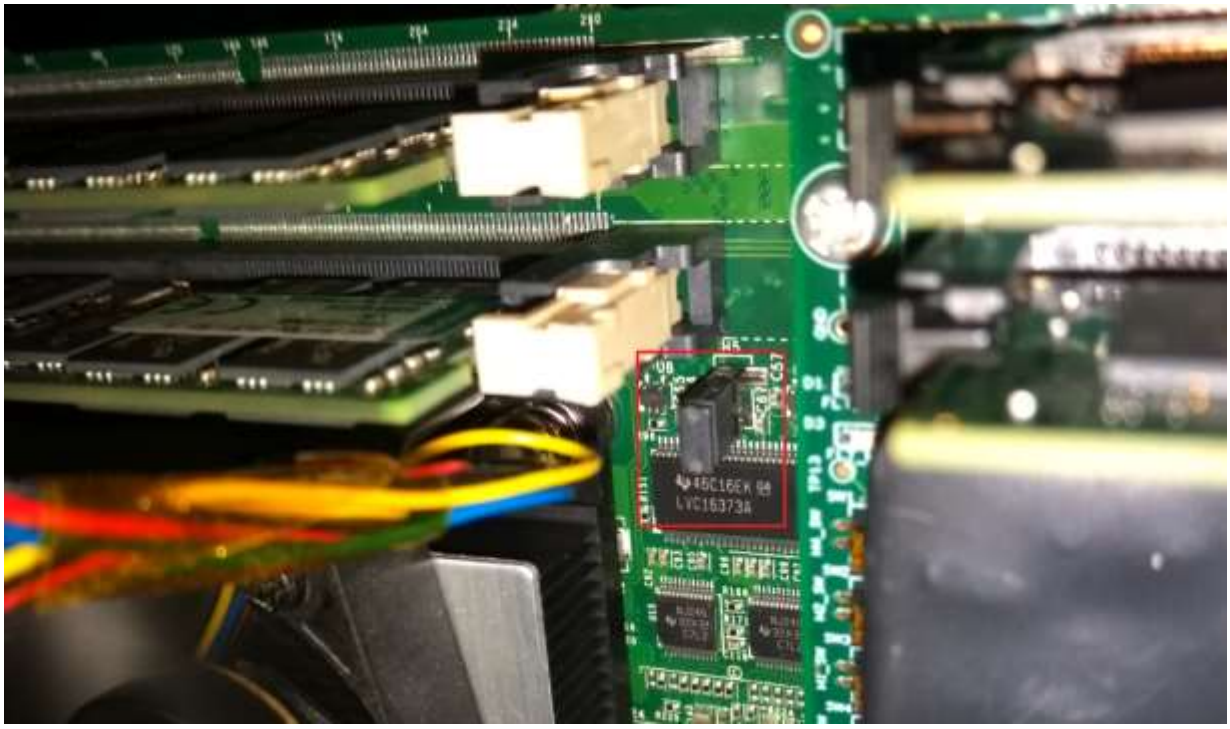
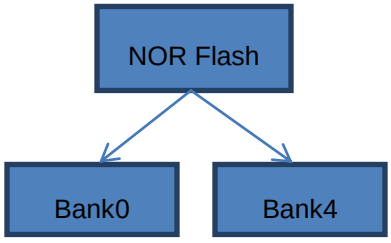


Figure 18: Bank4 jumper setting

Both banks allow flashing images from U-boot and Linux. From bank0, images on bank4 can be flashed based on addressing but from bank4, flashing images on bank0 is not supported. Once images are flashed on both banks, always work on bank4, don't work on bank0, because if improper images are flashed on bank4 and is not booted, one can switch to bank0 and flash proper images on the bank4 and start working on bank4. If bank0 is not booted because of the improper images flashed, cannot flash images on bank0 from bank4. So bank0 is considered to be factory default (images).



Addressing depends on current bank and the other bank, not on bank0 and bank4. For example,
If address of some image in the current bank is 0x580000000, then the address of that image in the other bank is 0x584000000 which means same steps to flash images on both banks and same address is used in the current bank regardless of bank0 and bank4.

Images and its partitions are shown below:

Partitions	File name	File size	Address	Allocated size
MTD 0	PBL	688 B	580000000 - 58001FFFF	128 KB
MTD 1	U-Boot	472 KB	580020000 - 5800BFFFF	640 KB
MTD 2	U-Boot Env	8 KB	5800C0000 - 5800DFFFF	128 KB
MTD 3	DPL	5 KB	5800E0000 - 5800FFFFF	128 KB
MTD 4	DPC	470 B	580100000 - 58011FFFF	128 KB
MTD 5	MC	2.3 MB	580120000 - 58039FFFF	2.5 MB
MTD 6	DTB	16 KB	5803A0000 - 5803BFFFF	128 KB
MTD 7	uimage_inic	7.2 MB	5803C0000 - 580C3FFFF	8.5 MB
MTD 8	uimage_issd	7.2 MB	580C40000 - 5814BFFFF	8.5 MB
MTD 9	Rootfs	13 MB	5814C0000 - 5835FFFFF	33.25 MB
MTD 10	Debug_log	-	583600000 - 583FFFFFF	10 MB

The commands to flash image on banks is given below (give files name accordingly):
Copy issd_images folder to USB/SD card and connect to iSSD. While booting stop counting by hit enter will stay on U-Boot and give the below commands to write images in the flash from USB.

Mount usb before start using it

```
=>usb start
```

To list the files in the folder (issd_images)

```
=>fatls usb 0 issd_images
```

To load the image in the RAM

```
=>fatload usb 0 <RAM address><file name>
```

Erase the particular partition

```
=>erase <starting address><ending address>
```

To copy the image from RAM to the partition

```
=>cp.b <RAM address><Partition starting address> $filesize
```

To unmount usb

```
=>usb stop
```

Commands with address to flash images on the current bank:

U-BOOT ENV

```
=>protect off 0x5800C0000 0x5800DFFFF
=>erase 0x5800C0000 0x5800DFFFF
```

PBL

```
=>fatload usb 0 0xa0000000 issd_images/pbl_01.00.00.bin;erase 0x580000000 0x58001FFFF;
cp.b 0xa0000000 0x580000000 $filesize
```

U-BOOT

```
=>fatload usb 0 0xa0000000 issd_images/u-boot_01.00.00.bin;erase 0x580020000 0x5800BFFFF;
cp.b 0xa0000000 0x580020000 $filesize
```

DPL

```
=>fatload usb 0 0xa0000000 issd_images/dpl_01.00.00.dtb;erase 0x5800E0000 0x5800FFFFF;
cp.b 0xa0000000 0x5800E0000 $filesize
```

DPC

```
=>fatload usb 0 0xa0000000 issd_images/dpc_01.00.00.dtb;erase 0x580100000 0x58011FFFF;
cp.b 0xa0000000 0x580100000 $filesize
```

MC

```
=>fatload usb 0 0xa0000000 issd_images/mc_01.00.00.itb;erase 0x580120000 0x58039FFFF;
cp.b 0xa0000000 0x580120000 $filesize
```

DTB

```
=>fatload usb 0 0xa0000000 issd_images/device-tree_01.00.00.dtb;erase 0x5803A0000 0x5803BFFFF;
cp.b 0xa0000000 0x5803A0000 $filesize
```

uImage-INIC

```
=>fatload usb 0 0xa0000000 issd_images/uimage_inic_02.00.00.bin;erase 0x5803C0000 x580C3FFFF;
cp.b 0xa0000000 0x5803C0000 $filesize
```

uImage-ISSD

```
=>fatload usb 0 0xa0000000 issd_images/uimage;erase 0x580C40000 0x5814BFFFF;
cp.b 0xa0000000 0x580C40000 $filesize
```

ROOTFS

```
=>fatload usb 0 0xa0000000 issd_images/rootfs_01.00.00.sqsh;erase 0x5814C0000 0x5835FFFFF;
cp.b 0xa0000000 0x5814C0000 $filesize
```

Commands with address to flash images on other bank is given below (from bank4 is not supported).

U-BOOT ENV

```
=>protect off 0x5840C0000 0x5800DFFFF
=>erase 0x5840C0000 0x5800DFFFF
```

PBL

```
=>fatload usb 0 0xa0000000 issd_images/pbl_01.00.00.bin;erase 0x584000000 0x58401FFFF;
cp.b 0xa0000000 0x584000000 $filesize
```

U-BOOT

```
=>fatload usb 0 0xa0000000 issd_images/u-boot_01.00.00.bin;erase 0x584020000 0x5840BFFFF;
cp.b 0xa0000000 0x584020000 $filesize
```

DPL

```
=>fatload usb 0 0xa0000000 issd_images/dpl_01.00.00.dtb;erase 0x5840E0000 0x5840FFFFF;
cp.b 0xa0000000 0x5840E0000 $filesize
```

DPC

```
=>fatload usb 0 0xa0000000 issd_images/dpc_01.00.00.dtb;erase 0x584100000 0x58411FFFF;
cp.b 0xa0000000 0x584100000 $filesize
```

MC

```
=>fatload usb 0 0xa0000000 issd_images/mc_01.00.00.itb;erase 0x584120000 0x58439FFFF;
cp.b 0xa0000000 0x584120000 $filesize
```

DTB

```
=>fatload usb 0 0xa0000000 issd_images/device-tree_01.00.00.dtb;erase 0x5843A0000
0x5843BFFFF;cp.b 0xa0000000 0x5843A0000 $filesize
```

uImage-INIC

```
=>fatload usb 0 0xa0000000 issd_images/uimage_inic_02.00.00.bin;erase 0x5843C0000
0x584C3FFFF;cp.b 0xa0000000 0x5843C0000 $filesize
```

uImage-ISSD

```
=>fatload usb 0 0xa0000000 issd_images/uimage;erase 0x584C40000 0x5854BFFFF;
cp.b 0xa0000000 0x584C40000 $filesize
```

ROOTFS

```
=>fatload usb 0 0xa0000000 issd_images/rootfs_01.00.00.sqsh;erase 0x5854C0000 0x5885FFFFF;
cp.b 0xa0000000 0x5854C0000 $filesize
```

Note: To flash from SD card, give **mmc** instead of usb in the above commands

3.2.4 Firmware upgrade running at kernel

The tarball issd-images_01.00.00.tar.gz contains the following files:

```
|--issd-images_01.00.00.tar.gz
|  |--checksums.md5
|  |--flash_images.sh
|  `--images
|     |--device-tree_01.00.00.dtb
|     |--dpc_01.00.00.dtb
|     |--dpl_01.00.00.dtb
|     |--fpga_01.00.0b.rpd
|     |--mc_01.00.00.itb
|     |--pbl_01.00.00.bin
|     |--rootfs_01.00.00.sqsh
|     |--u-boot_01.00.00.bin
|     |-- uimage_inic_02.00.00.bin
|     `-- uimage_issd_01.00.00.bin
```

Name of the binaries under images directory should be in following format.

1. Binary should have the binary key word as a part of its binary name.
2. Binary version should begin with _ and end with ".<binary extension>"

<binary name with its key word>_<binary version>.<binary extension>

Ex: pbl-updated_01.00.00.bin

Binary name	Key word of the binary	Binary extension
RCW image	pbl	bin
U-Boot image	u-boot	bin
DPL image	dpl	dtb
DPC image	dpc	dtc
MC image	mc	itb
device tree image	device	dtb
kernel image	uimage-inic uimage-issd	bin
Filesystem	rootfs	sqsh

FPGA image	fpga	rdb

Table 12: Image naming format

3.2.5 Preparing the SD card/USB

3.2.5.1 From laptop

Insert USB/ SD card in the laptop and copy the tar to the USB / SD card:

Note:

- Format the USB with fat32 partition.
- USB need to enabled at U-Boot before accessing it. User need to issue the U-Boot command as discussed in the section 5.3.2

```
$ cp<iSSD binary tar ball> /media/<USER>/<storage>
```

3.2.5.2 From target

- Insert USB/SD card in the SD/SDHC slot or USB slot of iSSD board respectively
- To power up the LS2088a iSSD board, please refer section 2.6
- Boot the LS2088a iSSD board.
- To get the SD card/USB device file, issue the following command:

```
$ ls – la  /dev/sd*  or fdisk –l
```

- Make directory to mount SD/USB under /run

```
$ mkdir /run/binary
```

- Mount the SD card/USB manually

```
$ mount /dev/<device node> /run/binary
```

3.2.6 Flashing the images from the SD/USB card

- Change the directory to mount directory, where SD card/USB has mounted.

```
$ cd /run/binary
```

- Run the upgrade.sh script, which will do the firmware upgradation after validating firmware image version.

```
$ upgrade.sh  <options> <mount_path>/issd-images_01.00.00.tar.gz
```

NOTE: Script uses */etc/version* file for version validation.

See the help message to know all the options,

```
$ upgrade.sh help
```

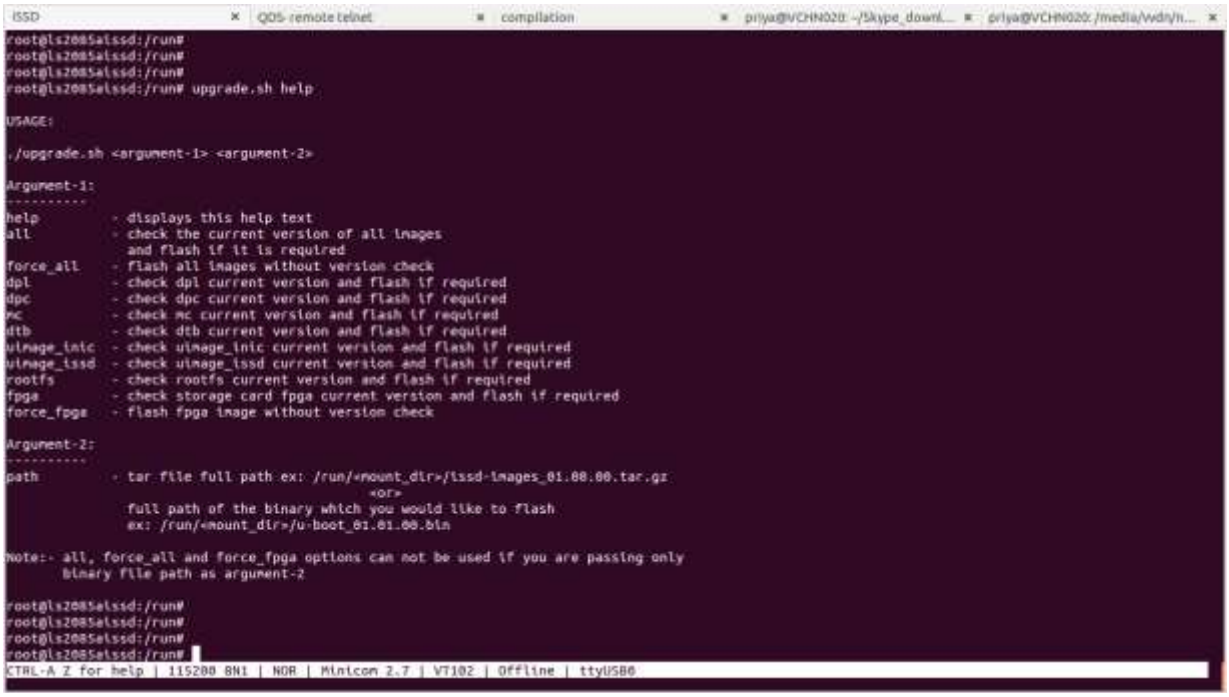


Figure 19: ***Help message of upgrade script***

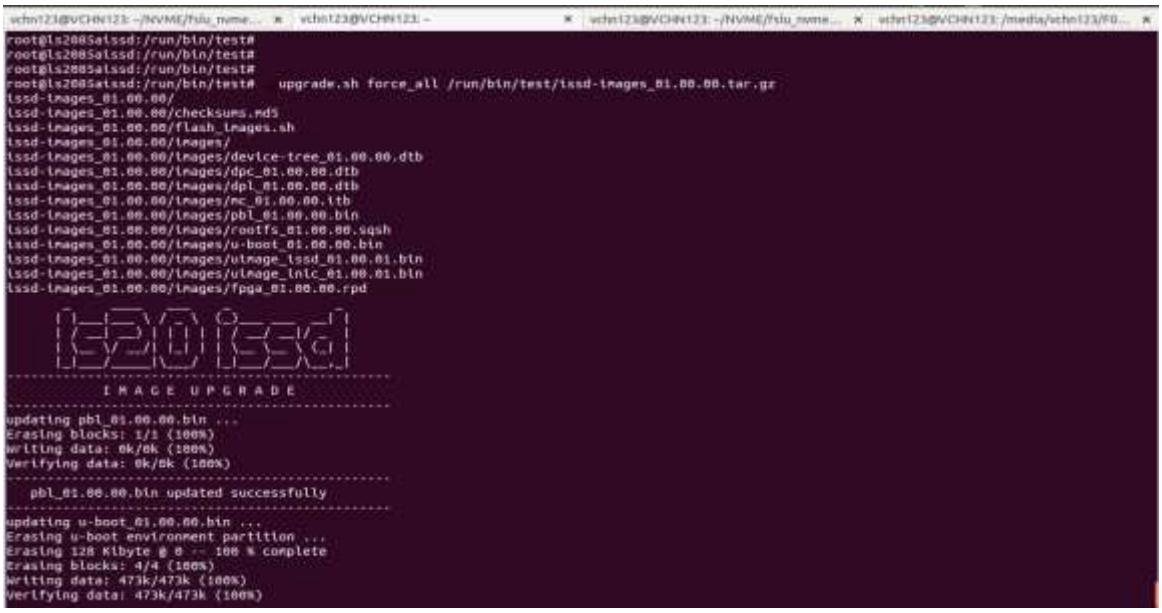


Figure 20: ***force upgrade of the all images***

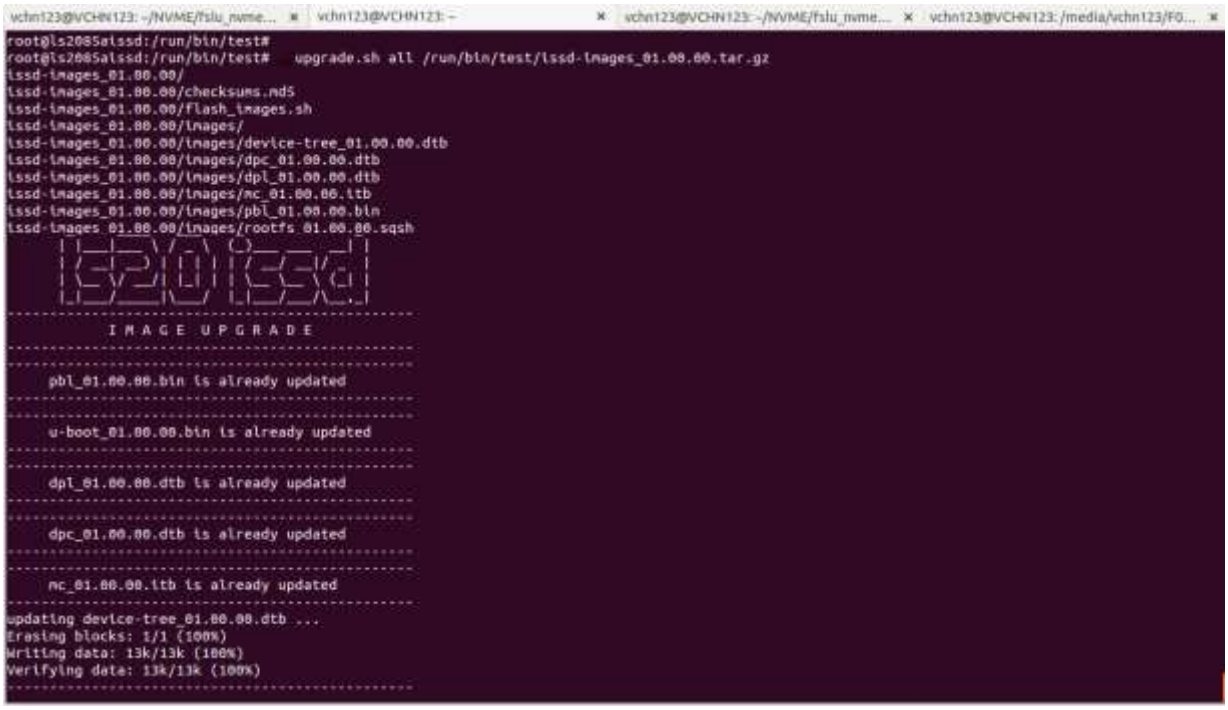


Figure 21: Upgrading all the images at once

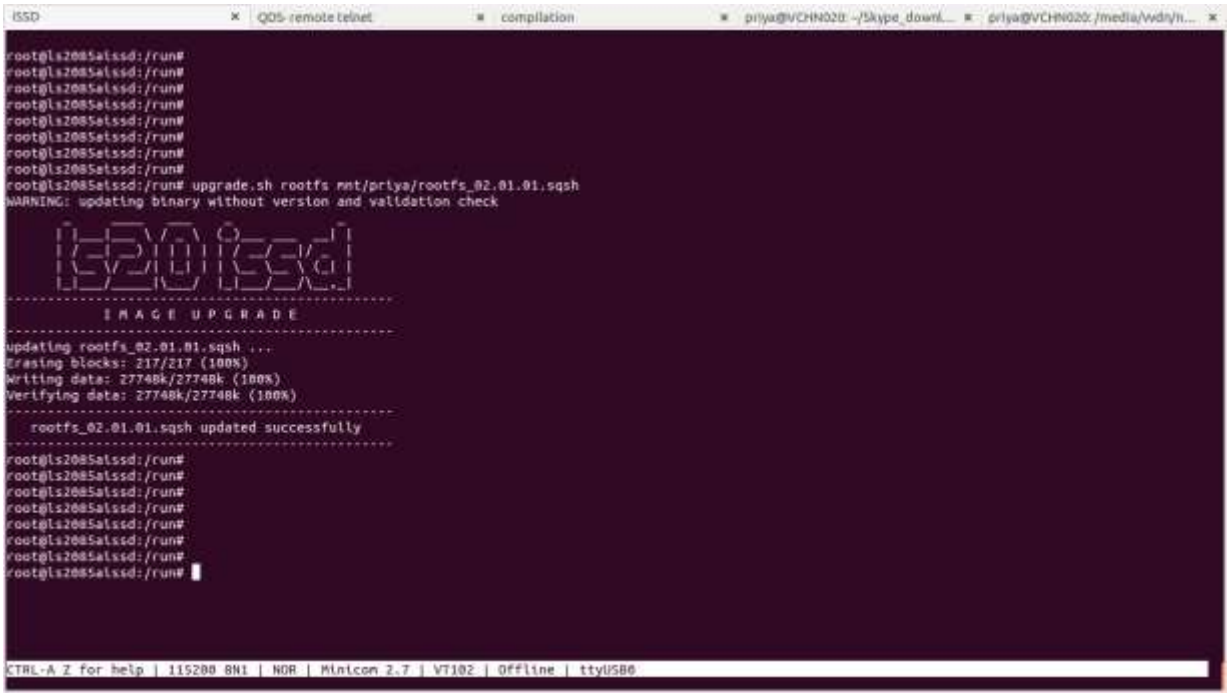


Figure 22: Upgrading root-fs alone

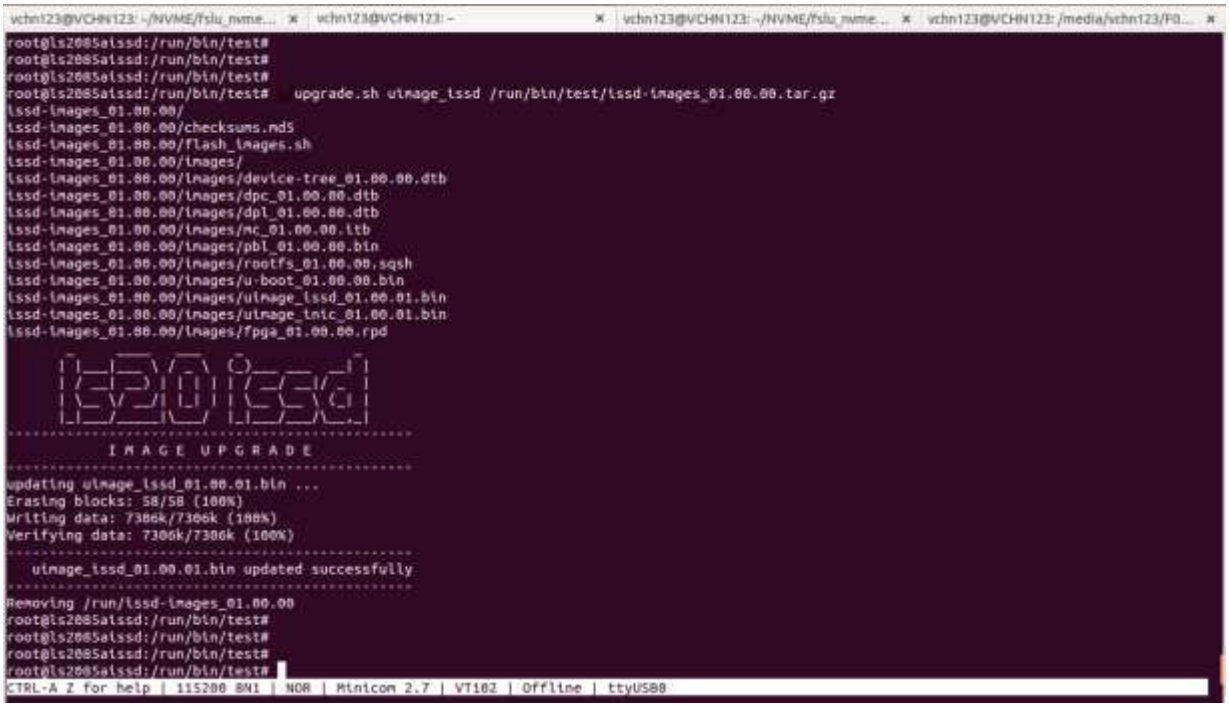


Figure 23: Upgrading Kernel image_iSSD alone

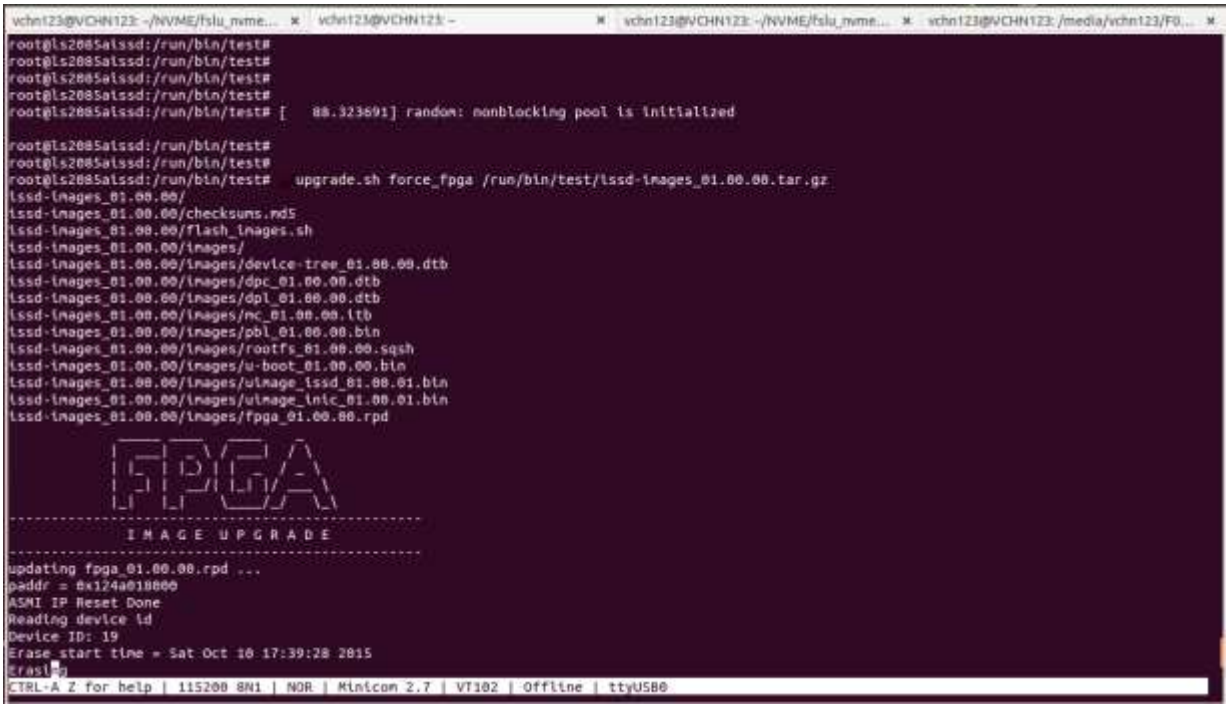


Figure 24: FPGA image upgrade

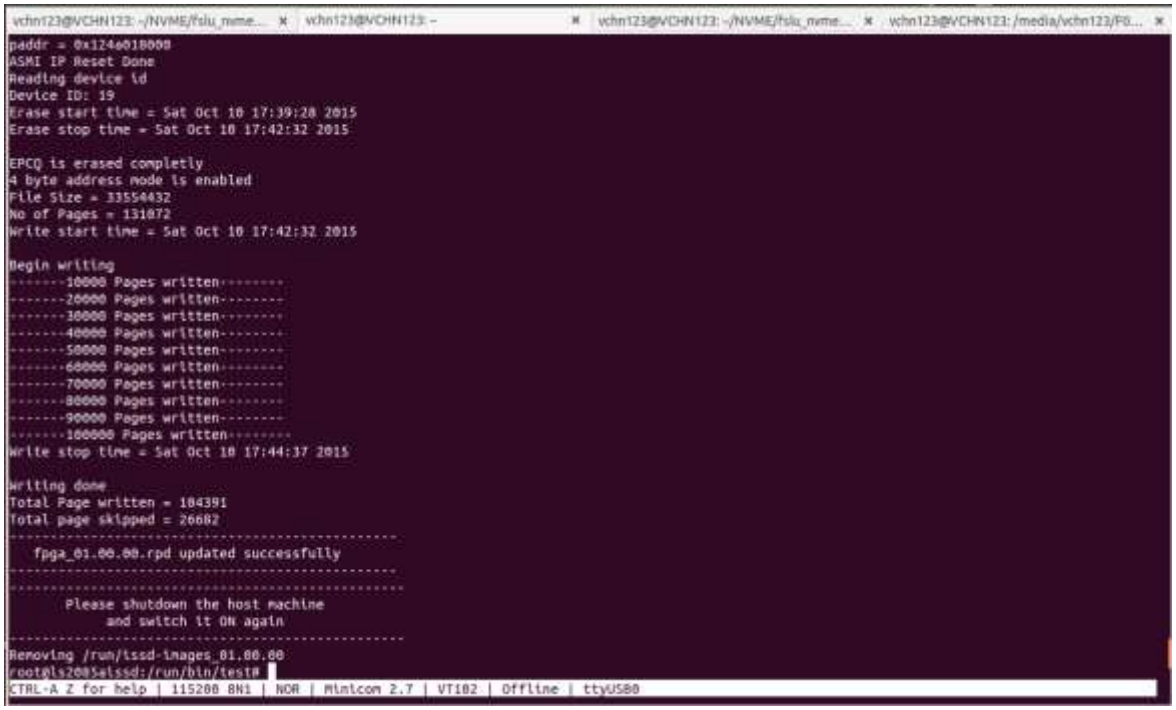


Figure 25:End of the FPGA image upgrade

- Run version.sh script to know the current version of all the images running on the target board.

```
$ version.sh
```

```
root@ls2085aissd:~# version.sh
 ========================================
        Image           Version
 ========================================
    PBL                 01.00.00
    DPL                 01.00.00
    DPC                 01.00.00
    MC                  01.00.00
    Kernel              01.00.00
    U-boot              01.00.00
    rootfs              01.00.00
    CPLD_NIC            A10
    CPLD_SSD            A01
    FPGA                01.00.01
root@ls2085aissd:~# version.sh checksum
 ================================================================
          Image                 Version              Checksum
 ================================================================
          PBL                   01.00.00              c42301d6
          DPL                   01.00.00              36e0ce85
          DPC                   01.00.00              e985ecab
          MC                    01.00.00              a4f23f6e
          Kernel                01.00.00               --
          U-boot                01.00.00               --
          rootfs                01.00.00               --
          CPLD_NIC              A10                    --
          CPLD_SSD              A01                    --
          FPGA                  01.00.01               --
root@ls2085aissd:~#
```

Figure 26: Check version and checksum of all images running in iSSD

3.3 Flashing the CPLD image using Quartus

Quartus software is used to flash the image. Quartus is a software which supports various device families such as cyclone IV, cyclone V, stratix V and etc.

TheQuartus 14.1 software shall be installed in the windows OS (64 bit), to flash the CPLD image on both iNIC and Storage Card.

Note: During installation, dialog box opens with device list. Select the stratix V to install device support for stratix V.

3.3.1 USB-Blaster Connection

USB-Blaster is connected to the storage card to open the programmer for flashing the storage card CPLD image as shown in the figure 27.

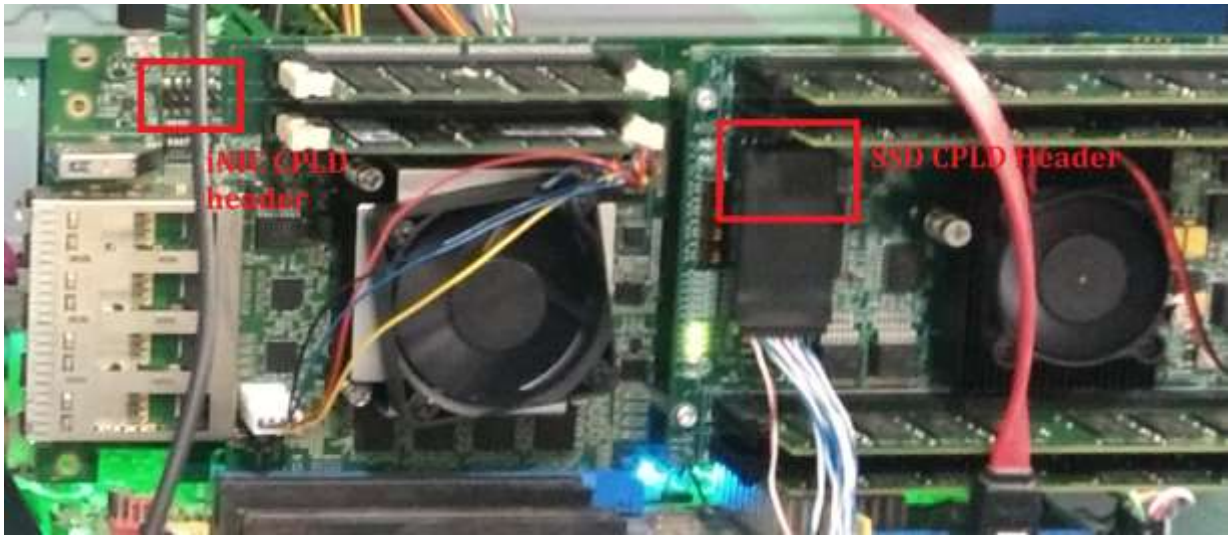


Figure 27: USB-Blaster Connection

Note: USB-Blaster should be connected to the iNIC CPLD JTAG Header to open the programmer for flashing the iNIC card CPLD image.

3.3.2 Steps to flash the CPLD image using Quartus

Following are the steps to flash the CPLD image:

- Open the Quartus (14.1) Software. When it is opened, a new window appears as like figure 28.
- Click on the *programmer* field which will open one more window that looks like figure 29. Verify whether the USB-Blaster is selected in the hardware setup. If USB-Blaster is not selected then select it. Also choose *Mode* as *JTAG*.
- Click on *Auto Detect* tab to detect device. On successful detection, the window looks exactly as figure 30.
- Choose the CPLD image file through *change file* option by *right clicking* as shown in the figure 31.
- Open the .pof file as shown in figure 32and select the checkbox as shown in the figure 33. On successful open, that file name is replaced with selected file name.
- Finally, click the *start* tab to start flashing the image. On successful flashing, 100% is shown in the progress box as shown in the figure 34.

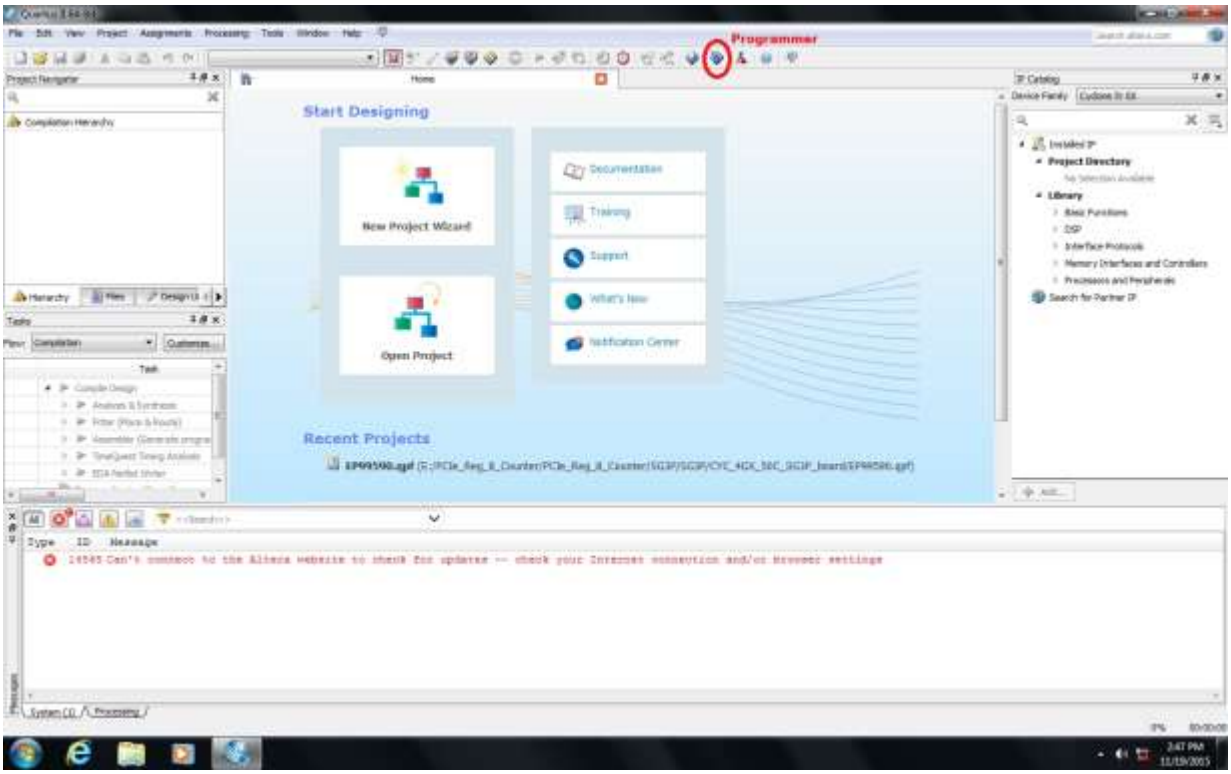


Figure 28: Quartus Starting window

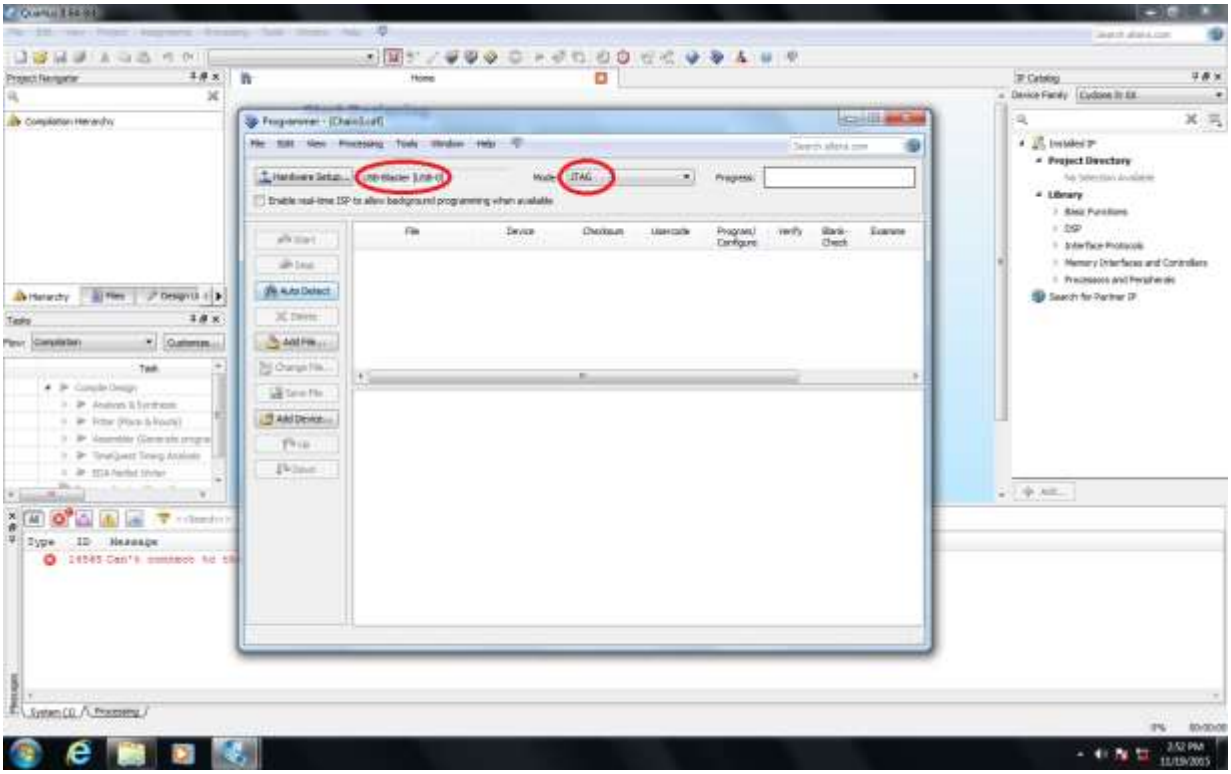


Figure 29: Quartus Programmer window

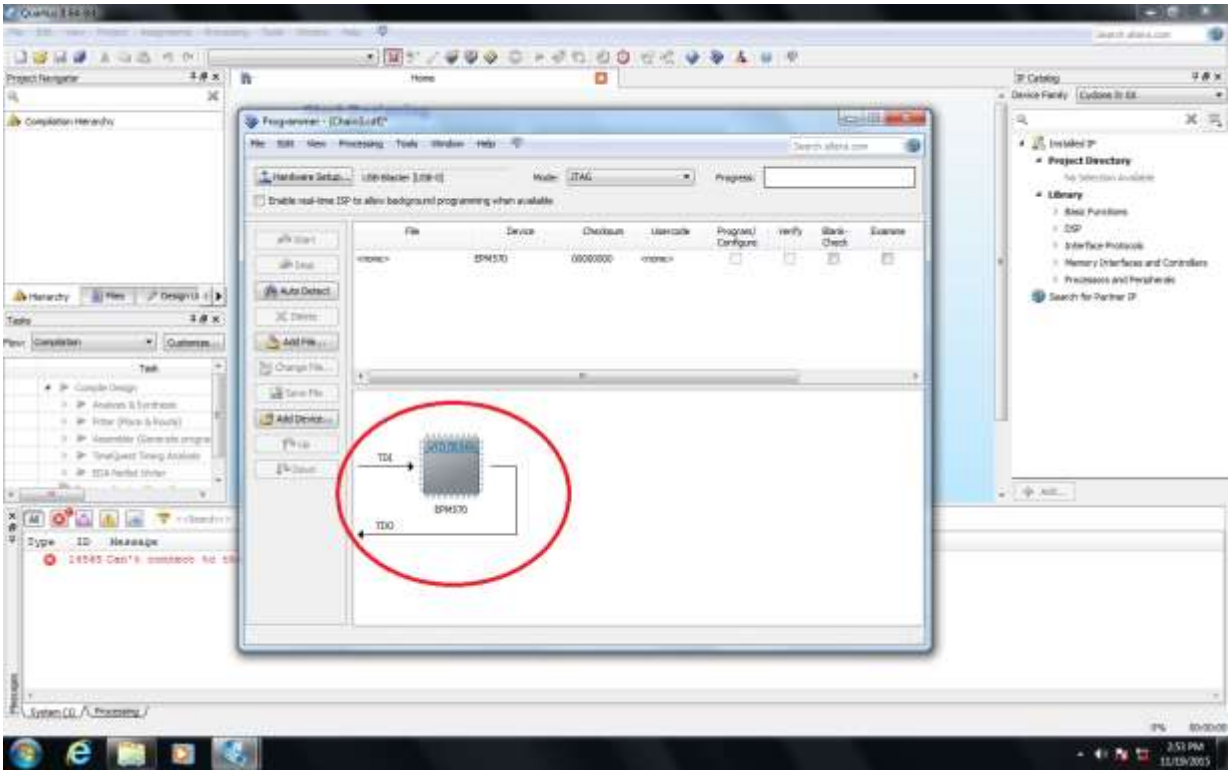


Figure 30: Detecting device

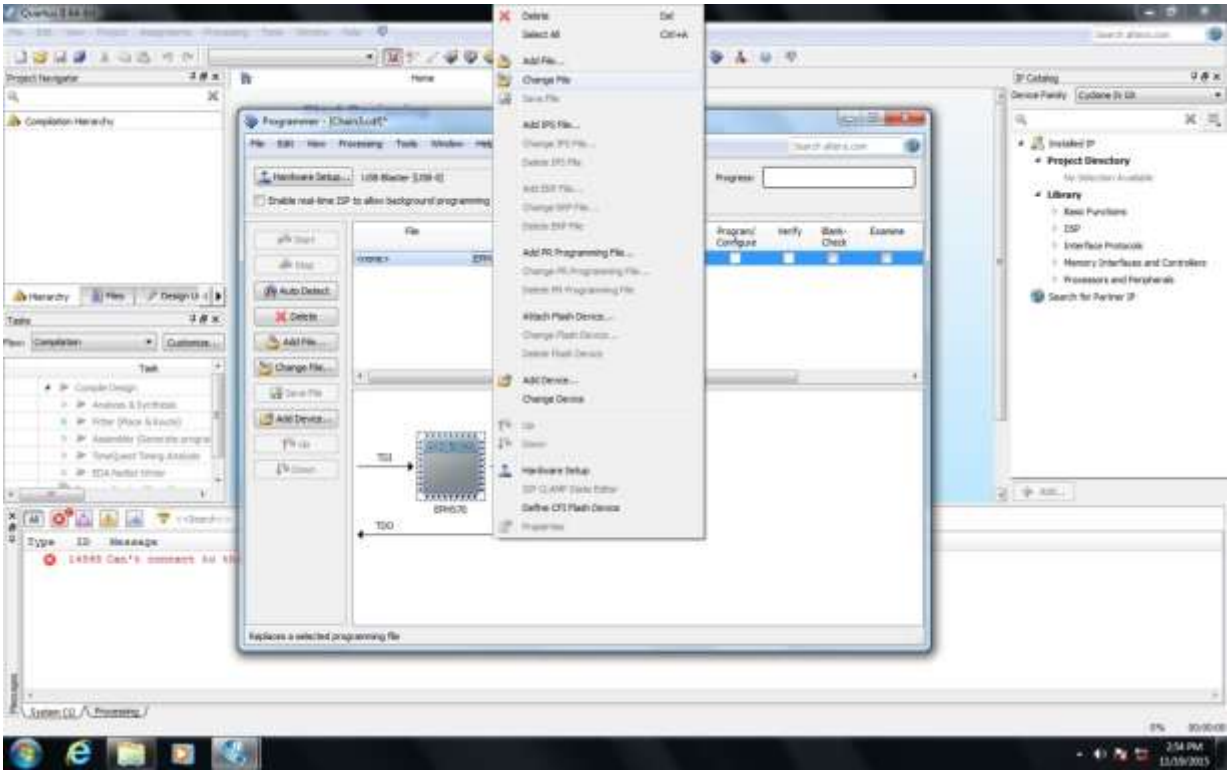


Figure 31: Changing the image file to be flashed

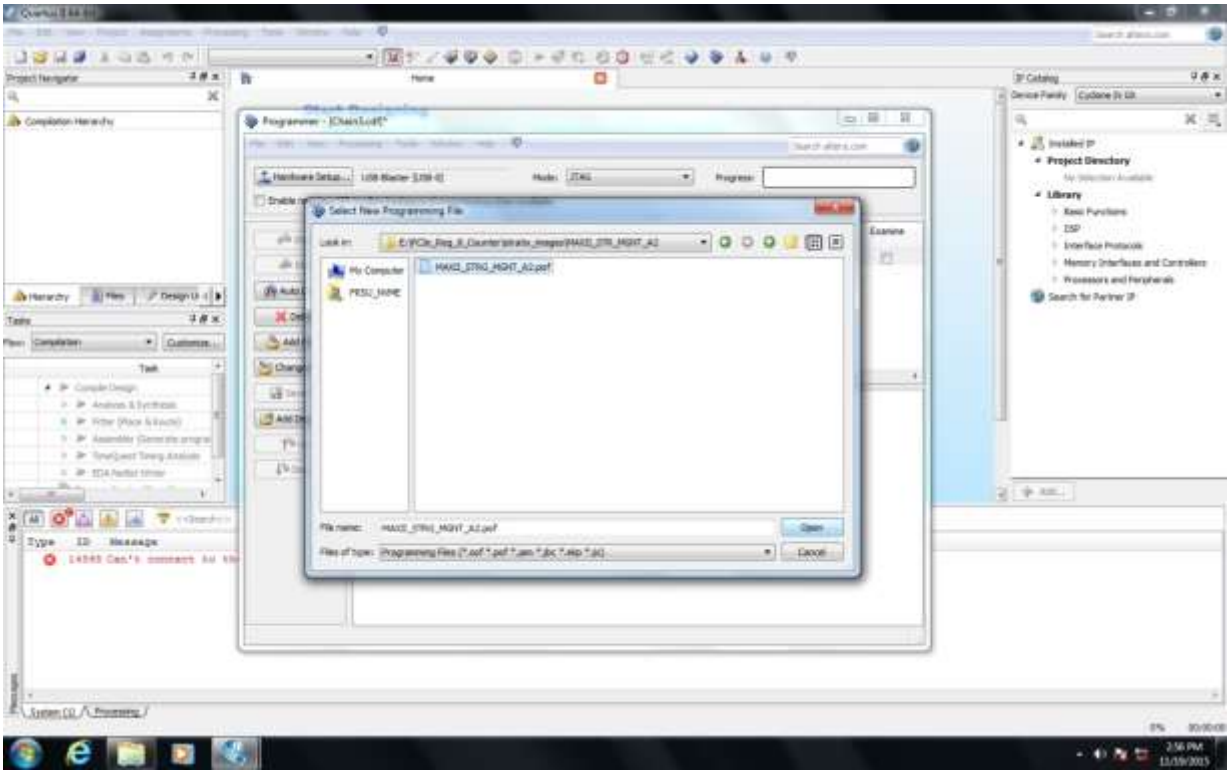


Figure 32: Opening the image file to be flashed

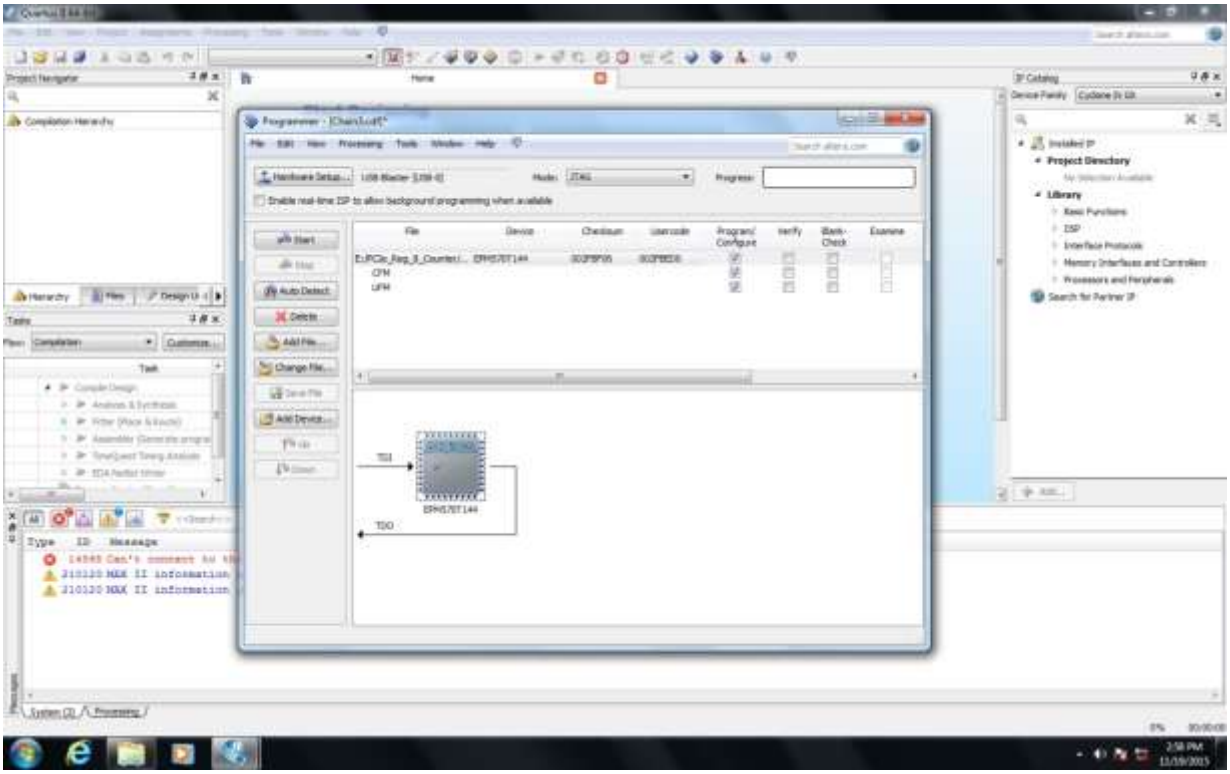


Figure 33: Selecting the image file to be flashed on the device

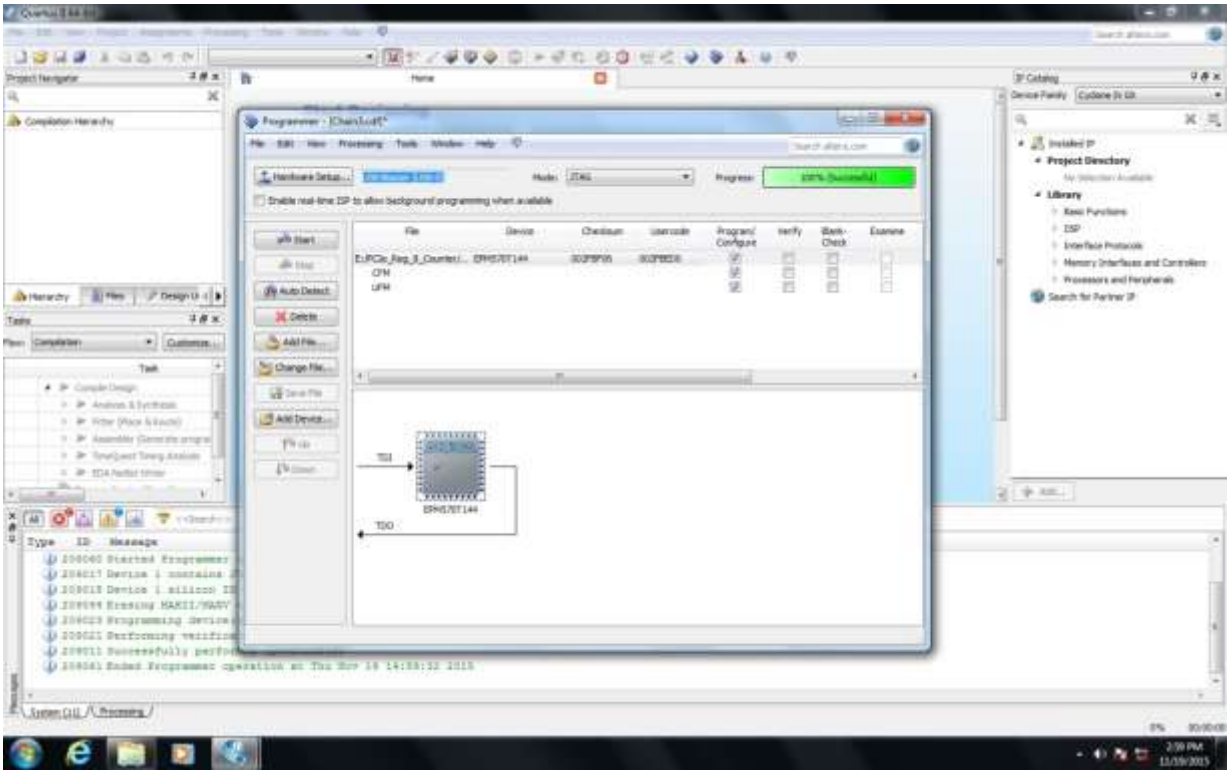


Figure 34: Flashing the image on the device

3.4 Flashing the FPGA image

3.4.1 Using Quartus Software

USB-Blaster should be connected to the JTAG Header (J6) in the storage card to open the programmer for flashing the FPGA image as shown in the figure 35.

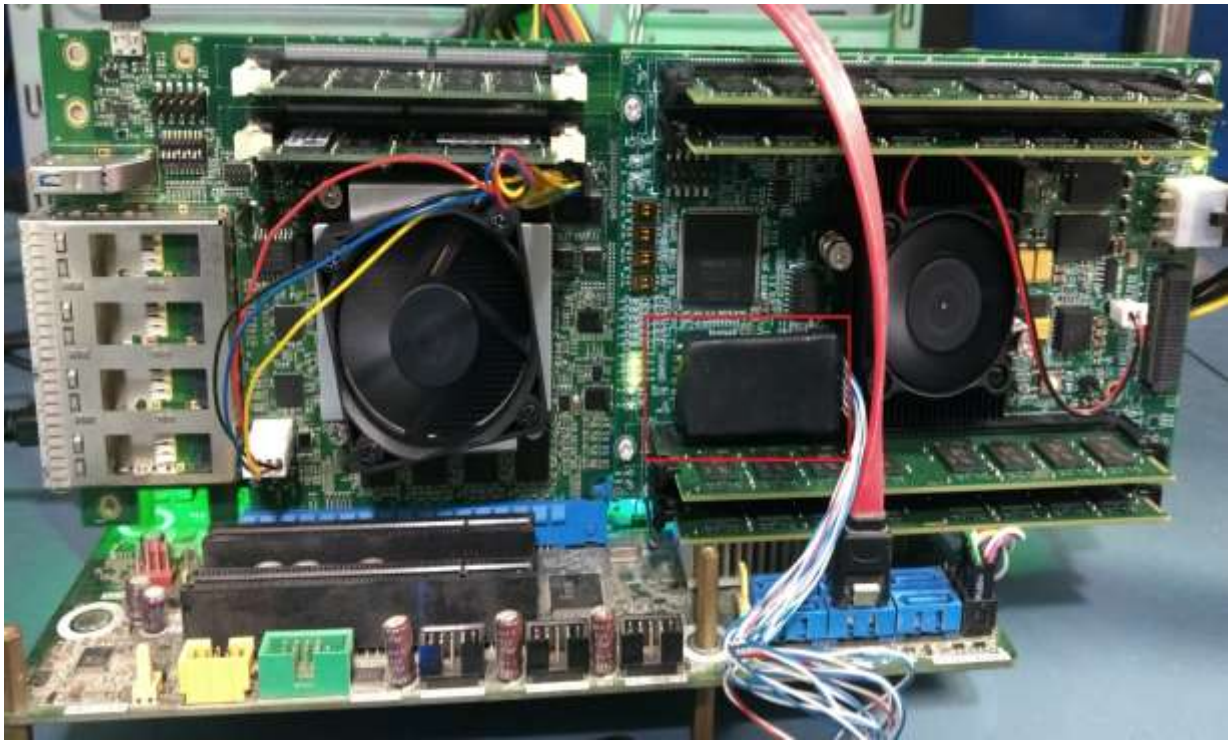


Figure 35: USB Blaster Connection

3.4.1.1 Steps to flash the FPGA image using Quartus

- Open the Quartus Software. When it is opened, a new window appears as like figure 28.
- Click on the *programmer* field which will open one more window that looks like figure 29. Verify whether the USB-Blaster is selected in the hardware setup. If USB-Blaster is not selected then select it. Also choose *Mode* as *JTAG*.
- Click on *Auto Detect* tab to detect device and Select the option **5SGXEA3K3** in case of A3 fpga is used otherwise select the option **5SGXEA7K3** if A7 fpga is used When the device list pop up window opens.
- Choose the FPGA image file through *change file* option by *right clicking on the file appears*.
- Open the .jic file of FPGA image and select the checkbox. On successful open, that file name is replaced with selected file name.
- Finally, click the *start* tab to start flashing the image. On successful flashing, 100% is shown in the progress box as like shown in the figure 34.

3.4.2 From Linux

FPGA image can be upgraded from Linux Once FPGA image is flashed using USB Blaster.

This feature is supported by upgrade.sh script, not only FPGA image, all images are upgraded from linux using upgrade.sh script. tar contains all images including FPGA image, should be passed as a path, as shown below.

```
./upgrade.sh  force_fpga  <path>
```

Eg: ./upgrade.sh force_fpga /run/mnt/issd-images_01.00.00.tar.gz

This will take few minutes (max 5 mins) to upgrade FPGA image, don't hit ctrl-C while flashing image.

On successful completion will get the print like below.

Image updated successfully. Please shutdown the host machine and switch it ON again.

Note: Make sure the FPGA image version used is compatible for the FPGA device version

3.5 Adding packages in SDK

SDK has Toolchain for cross compiling and sysroot for headers and libraries needed for the target machine. Cross compilers and sysroots are added in the running environmental variables by the SDK provided script (in sources directory). Bitbake is used for building packages and to do so it requires dot bb file (<package name>.bb file) for the package.

3.5.1 Steps to add package

- Make a directory with the package name in anyone of the recipes folder in the meta-fsl-arm.
- Again make a directory with the name of *files (or with package name again)* inside the package directory
- Keep the source files inside the *files* directory. If package source contains many files, can create tarball and keep the tarball inside the *files* directory. Also keep *Readme.txt* file inside files directory and checksum of this file will be given in the .bb file
- Place the .bb file of the package, outside of *files* directory i.e under package directory.
- To add the binaries in the image, the package name should be added in the appropriate bitbake image (fsl-image-minimal, fsl-image-core and etc...).

Once the above steps are done, whenever bitbake generate image that will contain binaries of the package added.

3.5.2 Steps to compile the package alone

Bitbake generate binaries from the source with the help of .bb file of that package. Follow the below steps for building package separately.

```
bitbake -c clean -f <package name given in the .bb file>

bitbake -c compile -f <package name given in the .bb file>

bitbake  <package name given in the .bb file>
```

3.5.3 Example

Assume hello.c, Readme.txt and hello.bb files (content is shown below) are present in the home directory.

Cross compiling package for target machine and adding binary in the image (fsl-image-minimal) is shown below.

Change directory to meta-fsl-arm using *cd* command, and do the following steps:

```
meta-fsl-arm$
meta-fsl-arm$       cd recipes-user
recipes-user$       mkdir hello
recipes-user$       cd hello
hello$     mkdir files
hello$     cp ~/hello.c  files
hello$     cp ~/Readme.txt  files
hello$     cp ~/hello.bb  .
```

The content of the hello.bb file is shown below:

```
DESCRIPTION =  "hello program"
LICENSE = "GPLv2"
LIC_FILES_CHKSUM = "file://README.txt;md5=85bc148ae2df6154c0cfd11bbd46c3bd"
PR = "r0"
SRC_URI = "file://hello.c  \
        file://Readme.txt"
S = "${WORKDIR}"

do_compile() {
            cd ${S}
            ${CC}  -o hello hello.c
}

do_install() {
            mkdir -p ${D}/usr/
            mkdir -p ${D}/usr/sbin/
cp  ${S}/hello ${D}/usr/sbin/
}
```

Once the above steps are done, add the package name in the image file (assume fsl-image-minimal) .

The content of the “sources/meta-freescale/images/fsl-image-minimal.bb” file is shown below:

```
# Copyright(C) 2014-2015 Freescale
#
# The minimal rootfs with basic packages for boot up
#

include recipes-core/images/core-image-minimal.bb
require images/fsl-image-deploy.inc

inherit fsl-utils

IMAGE_INSTALL += " \
        echo-server \
        kernel-image \
        restool \
        setserial \
        kernel-modules \
        kmod \
        pciutils \
        i2c-tools \
        memtool \
        mtd-utils \
        mtd-utils-jffs2 \
        bash \
        iperf \
        i2capp \
        merge-files \
        flashfpga \
        issd-nvme \
        fio \
        sysklogd \
        hello  \
"
```

```
IMAGE_FSTYPES = "ext2.gz tar.gz"

ROOTFS_POSTPROCESS_COMMAND += "rootfs_add_sdk_version;"
```

Issue the following command from build directory to cross compile for target machine and to add binary (hello) in the image .

```
build_ls2088aissd_release$          bitbake fsl-image-minimal
```

3.5.3.1 Cross compiling for target machine

Issue the following two commands from the build directory to cross compile the package alone for target machine

```
build_ls2088aissd_release$          bitbake -c compile -f hello
build_ls2088aissd_release$          bitbake hello
```

Binary will be generated in the work directory.

4 Setting the serial console

Once the Board is power up, It will check the RCW configuration in the NOR flash and it is validated by the system CPLD. Once the verification the RCW is loaded and boots up the board. Initially we will the U-Boot prints and the prompt of it, and we can boot the board to the kernel prompt.

User can use the Serial communication program *minicom* from the Linux laptop to see the output of the U-Boot, Kernel and their prompt.

4.1.1 Serial communication program

If user doesn't have minicom installed, they can install by following command,

```
$ sudo apt-get install minicom
```

4.1.2 Configuring the minicom

- Open the minicom using the command.

```
$ sudo minicom –s
```

- Connect the Micro USB cable to the Laptop/ PC to launch the minicom.
- Select the Serial port set up in item list as shown below,

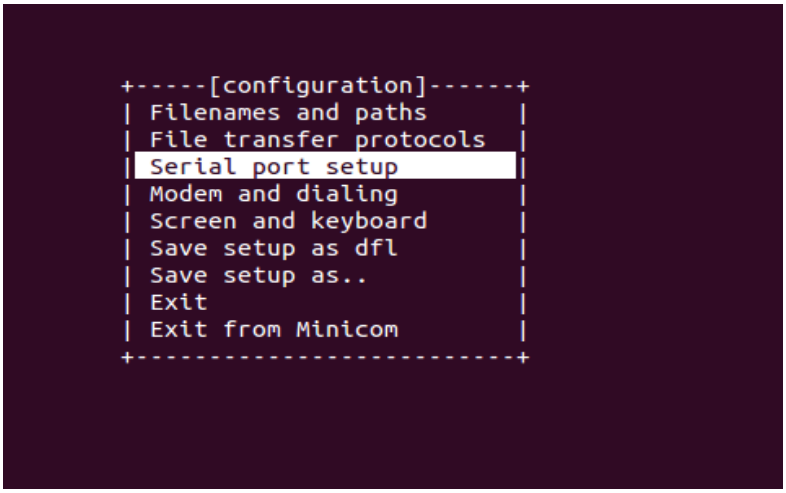


Figure 36: Configuring Minicom

- Select the serial device as "/dev/ttyUSB0" and make sure that the other settings are configured as shown below.

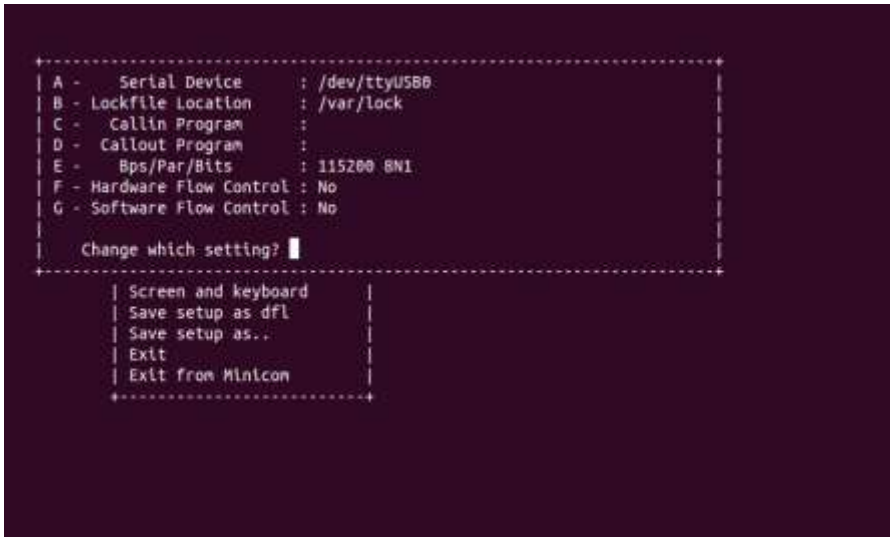


Figure 37: Serial port selection in minicom

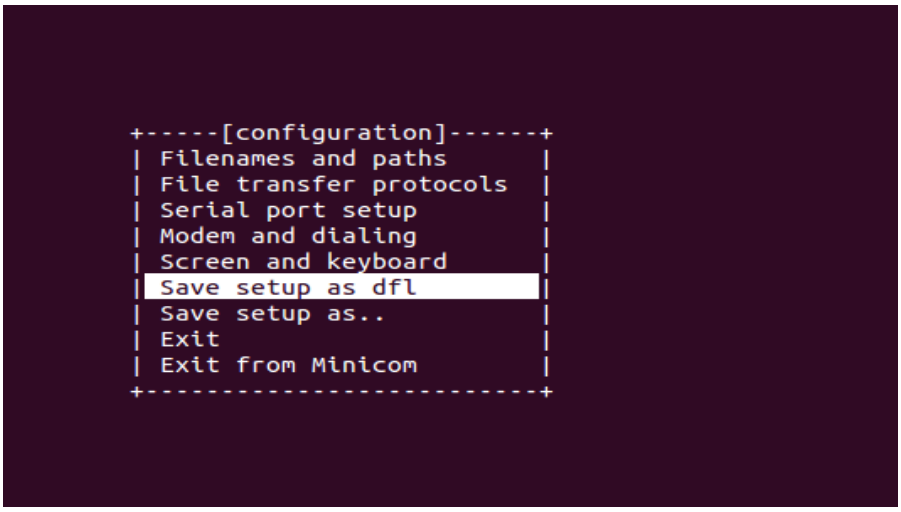


Figure 38: saving minicom configuration

- Press the "*save setup as dfl*" .
- Hold on the **Exit** and power up board and then press the Exit field of the minicom. User can observe the booting.

5 Test procedure

5.1 Pre-test checklist

- Proper switch settings and jumper settings (refer table no:1) before powering ON.
- Inserting the XFI modules before powering up is recommended than hot plugging.
- Ensure proper cooling fan is connected to the board and it's working properly when board is powered up.
- Insert the SD card/ USB3.0 properly, before powering up .
- Check the LED's glowing or not when powered up as discussed in the section 2.2.9.1 else refer the section 2.2.9.3 for failure cases.

Note: Insert the external devices to the iSSD before powering it. Hot plugging of devices is not recommended

5.2 Checking the interfaces at booting

- Once the board is booted, User can see the following U-Boot prints.

```
U-Boot 2015.07-rc1Layerscape2-SDK+gfaa9145 (Dec 21 2015 - 10:44:34)


SoC:  LS2088E (0x87010010)
Clock Configuration:
      CPU0(A57):1600 MHz  CPU1(A57):1600 MHz  CPU2(A57):1600 MHz
      CPU3(A57):1600 MHz  CPU4(A57):1600 MHz  CPU5(A57):1600 MHz
      CPU6(A57):1600 MHz  CPU7(A57):1600 MHz
      Bus:     600  MHz  DDR:     1333.333 MT/s     DP-DDR:   1333.333 MT/s
Reset Configuration Word (RCW):
      00: 40282830 40400040 00000000 00000000
      10: 00000000 00200000 00200000 00000000
      20: 0e412980 00002580 00000000 00000000
      30: 00000a08 00000000 00000080 00000080
      40: 00000000 00000000 00000000 00000000
      50: 00000000 00000000 00000000 00000000
      60: 00000000 00000000 00027000 00000000
      70: 3f350000 00000000 00000000 00000000
Model: Freescale Layerscape 2085a QDS Board
I2C:   ready
Board Arch: V1, Board Version: A
CPLD Version: A10
DRAM:  Initializing DDR....using SPD
Detected UDIMM 18ASF1G72HZ-2G1A1
Detected UDIMM 18ASF1G72HZ-2G1A1
15.5 GiB
DDR    15.5 GiB (DDR4, 64-bit, CL=9, ECC on)
      DDR Controller Interleaving Mode: 256B
DDR Chip-Select Interleaving Mode: CS0+CS1
Waking secondary cores to start from fff21000
```

```
All (8) cores are up.
Using SERDES1 Protocol: 53 (0x35)
Using SERDES2 Protocol: 63 (0x3f)
Flash: 128 MiB
MMC:   FSL_SDHC: 0
EEPROM: NXID v1
PCIe1: disabled
PCIe2: Root Complex no link, regs @ 0x3500000
PCIe3: Endpoint no link, regs @ 0x3600000
PCIe4: Root Complex no link, regs @ 0x3700000
In:    serial
Out:   serial
Err:   serial
crc32+
fsl-mc: Booting Management Complex ... SUCCESS
fsl-mc: Management Complex booted (version: 8.0.2, boot status: 0x1)
fsl-mc: Deploying data path layout ... SUCCESS
DPNI4, DPNI3, DPNI2, DPNI1 [PRIME]
Hit any key to stop autoboot:  0
=>
--
```

From the above U-boot log , one can observe the major interfaces are up and each interface initialization is discussed below

5.2.1 CPLD version

```
Board Arch: V1 Board Version: A
CPLD Version: A10
```

5.2.2 DDR Verification

```
DRAM:  Initializing DDR....using SPD
Detected UDIMM 18ASF1G72HZ-2G1A1
Detected UDIMM 18ASF1G72HZ-2G1A1
15.5 GiB
      DDR    15.5 GiB (DDR4, 64-bit, CL=9, ECC on)
      DDR Controller Interleaving Mode: 256B
      DDR Chip-Select Interleaving Mode: CS0+CS1
```

5.2.3 Serdes Configuration:

```
Using SERDES1 Protocol: 53 (0x35)
Using SERDES2 Protocol: 67 (0x3f)
```

5.2.4 SDHC interface

```
MMC:   FSL_SDHC: 0
```

5.2.5 PCIe interface

```
PCIe2: Root Complex x4 gen3, regs @ 0x3500000
01:00.0    - 1957:0953 - Mass storage controller
PCIe2: Bus 00 - 01
PCIe3: Endpoint x4 gen3, regs @ 0x3600000
PCIe4: Root Complex no link, regs @ 0x3700000
```

5.2.6 Ethernet interface:

```
DPNI1 [PRIME], DPNI2, DPNI3, DPNI4
```

5.2.7 Environment of bootloader

- **Default environment variables**

```
=> prt
baudrate=115200
bootargs=ubootver=01.00.00 cpld-ssd-ver=A01 console=ttyS0,115200 root=/dev/mtdblock9 ro rootfstype=squashfs earlycon=uart8250,mmio,0x21c0500,115200 default_hugepa
gesz=2m hugepagesz=2m hugepages=16 cma=256m
bootcmd=cp.b $kernel_start_lnic $kernel_load $kernel_size  && cp.b $dtb_start $dtb_load $dtb_size && bootm $kernel_load - $dtb_load
bootdelay=2
bootimg=lnic
dtb_load=0x80000000
dtb_size=0x20000
dtb_start=0x5003A0000
eth1addr=00:04:9F:01:09:04
eth2addr=00:04:9F:01:09:05
eth3addr=00:04:9F:01:09:06
ethact=DPNI1
ethaddr=00:04:9F:01:09:03
ethprime=DPNI1
fdt_high=0xa0000000
hwconfig=fsl_ddr:bank_intlv=auto
initrd_high=0xffffffffffffffff
kernel_addr=0x100000
kernel_load=0xa0000000
kernel_size=0x880000
kernel_start_lnic=0x5003C0000
kernel_start_lssd=0x500C40000
loadaddr=0x80100000
ramdisk_addr=0x800000
ramdisk_size=0x2000000
stderr=serial
stdin=serial
stdout=serial

Environment size: 945/8188 bytes
=>
```

Figure 39: U-Boot Environment of the iNIC

If the user want to boot the iSSD for the accessing the storage card, user may need to use the following U-boot command

```
=>setenv bootimg issd
=> saveenv
```

Important Note:

Once the environment of the bootloader is changed for the iSSD then user **need to reboot** the x86 system if the iNIC is powered through the x86 system mandatorily.

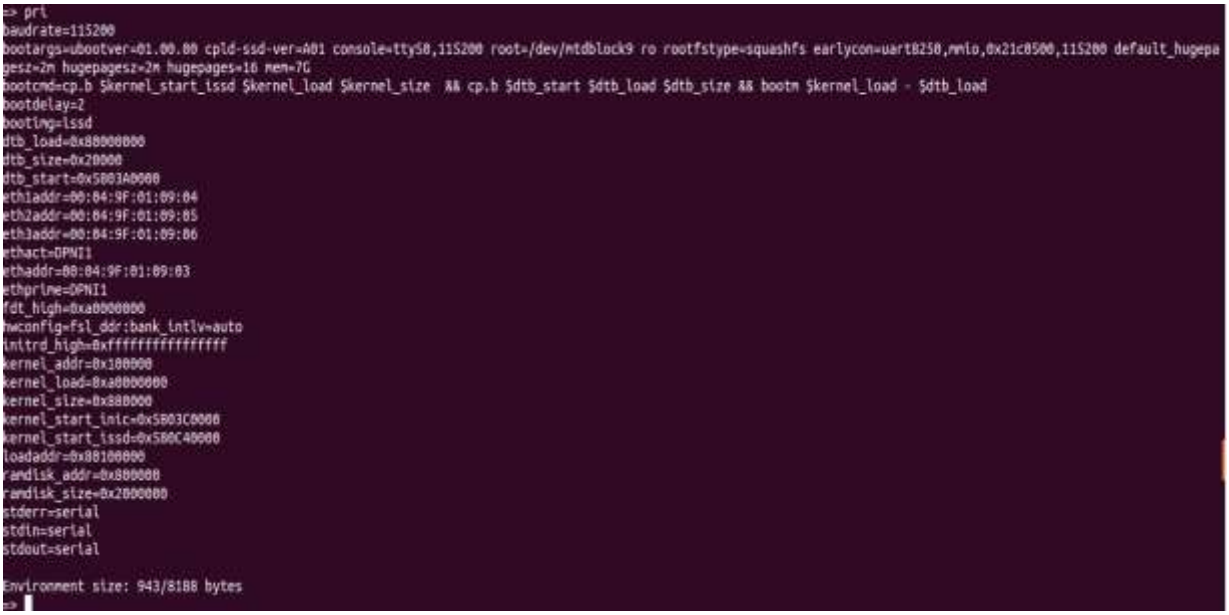


Figure 40: U-Boot Environment for the storage card

5.3 Checking interfaces at U-Boot using CLI/ U-Boot commands

5.3.1 SD card interface Testing

```
=>mmcinfo
Device: FSL_SDHC
Manufacturer ID: 3
OEM: 5344
Name: SL08G
Tran Speed: 50000000
Rd Block Len: 512
SD version 3.0
High Capacity: Yes
Capacity: 7.4 GiB
Bus Width: 4-bit
```

5.3.2 USB interface testing

Plug the USB at the USB slot(shown below) before powering up board.

```
=>usb start
(Re)start USB...
USB0:   Register 200017f NbrPorts 2
```

```
Starting the controller
USB XHCI 1.00
scanning bus 0 for devices... 2 USB Device(s) found
      scanning usb for storage devices... 1 Storage Device(s) found
=>usb tree
USB device tree:
 1  Hub (5 Gb/s, 0mA)
 |  u-boot XHCI Host Controller
 |
 +-2  Mass Storage (480 Mb/s, 200mA)
      JetFlash Mass Storage Device UYNGOZR7
```

5.3.3 SPF interface:

```
=>ping 192.168.0.3
Using DPNI1 device
host 192.168.0.3 is alive
```

5.4 Loading the Kernel

- If user stops the boot at U-boot then the following command need to issue at the U-Boot prompts to load the Kernel.

```
=>boot
```

- If the Kernel image is place in the RAM address, then the command is as follows,

```
=>bootm <load/ram address>
```

5.5 Testing the 10G interfaces at the Kernel

Procedure:

- Assign IP for the corresponding connected ports on both the boards of same domain
- For example,

```
=>ifconfig ni3 192.168.0.3 for first LS2088a
=>ifconfig ni2 192.168.0.4 for second LS2088a
```

Where ni3 and ni2 are the ethernet interfaces in LS2088A 3rd and 2nd SFP+ ports.

- Now try to ping each other using **ping 192.168.0.3** from first LS2088A board and **ping 192.168.0.4** from second and observe the packets received and transmitted in their corresponding console
- Also verify the same using **ifconfig -a** command after the ping operation, Tx and Rx count of corresponding interfaces should be increased.
- Also try pinging between the boards at u-boot level itself. For example, both the boards at u-boot level, one at u-boot and other at kernel.
- While trying to ping the board at u-boot,
- Assign IP using the command

```
=>set ipaddr <ip_address>
```

- For example, set ipaddr 192.168.0.3
- Then try to ping the board from other board

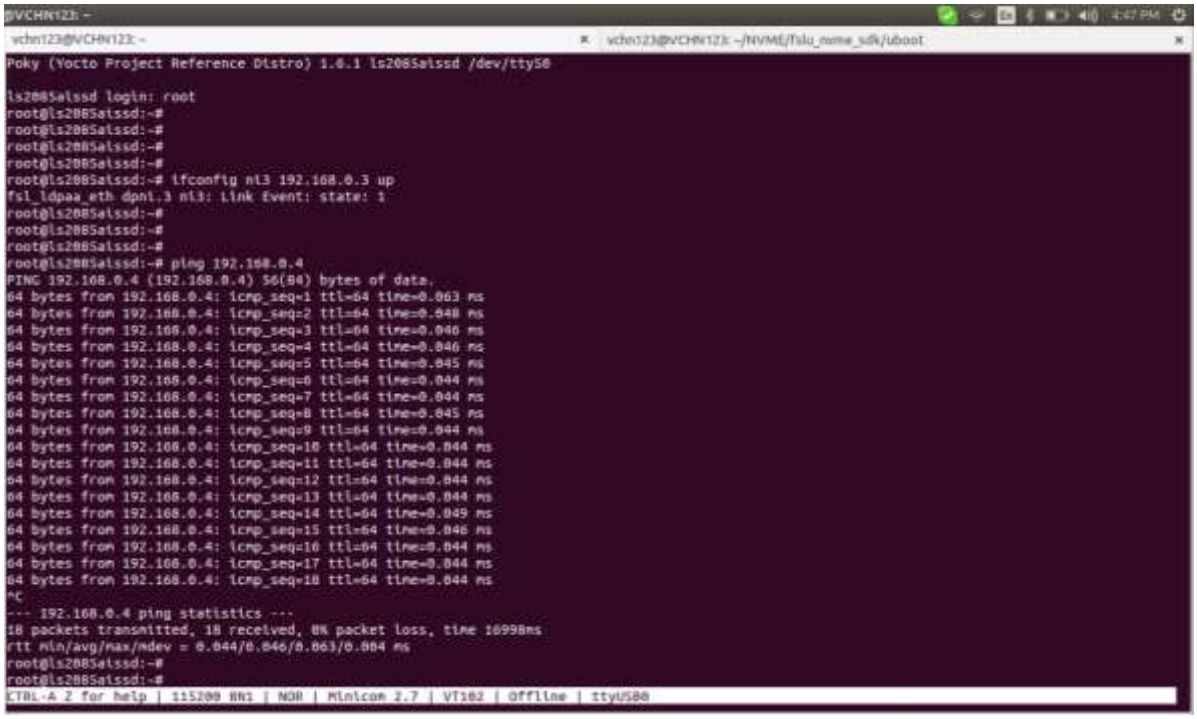


Figure 41: 10G interface test result

5.6 Testing PCIe interface at Kernel

PCIe interface can be tested at the Kernel prompt through the following command, and it will list the clear details of the PCIe interface as shown below,

```
=>lspci -vv
```

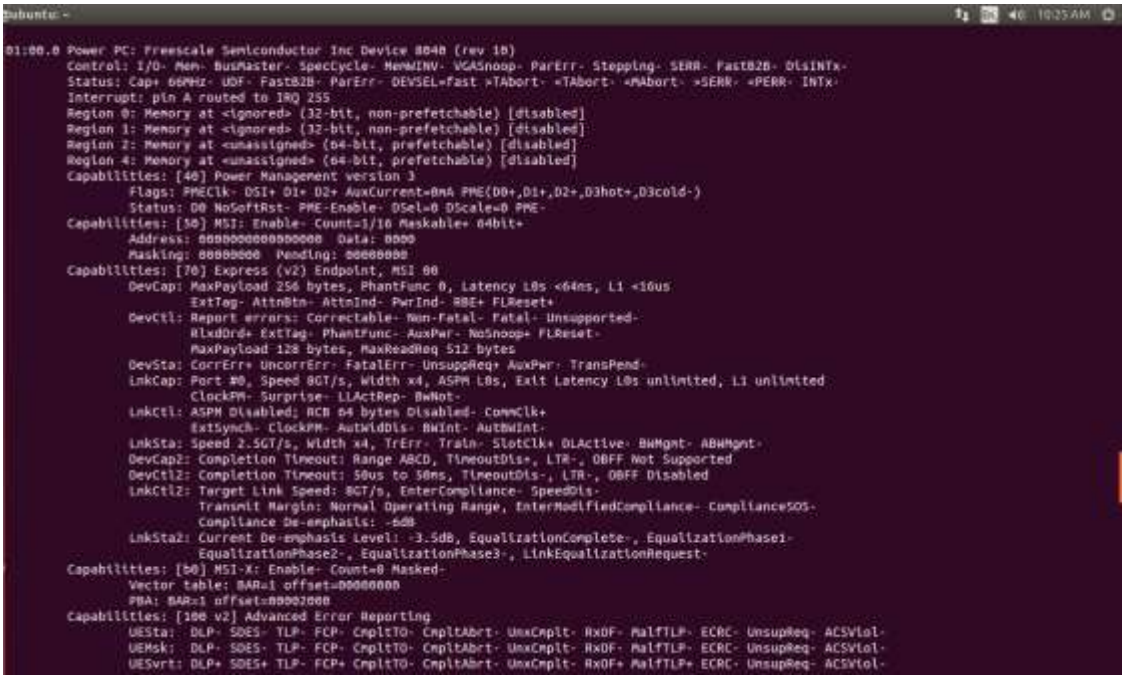


Figure 42: PCIe test result

5.7 Power off

- Initially the power button of the board should be switched off and then ATX power supply, in case it is powered through ATX power supply.
- If it is plugged into X86, powering down the system is enough.

6 Appendix

6.1 Using PCIe power card

6.1.1 Before powering up

This section describes the required setup of the board, before powering it up.

- Connect the ATX 24V pin connector to the PCIe power card and then connect iNIC card to the PCIe slot in PCIe Power card.
- The PCIe power card is shown below,

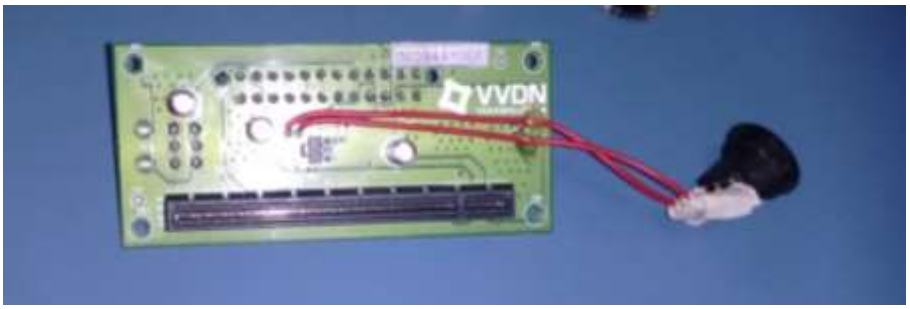


Figure 43: Front side assembly of PCIe power card

Figure 44: Back side assembly of the PCIe power card

Figure 45: Powering though ATX power supply and PCIe card

6.1.2 Powering through ATX power supply and PCIe power card

- Switch on the main power supply and then ATX power supply connected to the board
- User can see the Power LED(D3) glowing up in **Green color** and LED(D5) in **Red** color glowing beside that, which ensures that board is powered up and initial RCW sequencer is happening. This is referred as auxiliary mode. The LED indication is shown in the figure 15.

- If the board is not properly powered up, LED D7 will glow in **Red** color.
- Switch on the push button/Power button on the board.
- Now LED(D8) will be glowing near the PCI edge connector and LED D5 in Red color glows off.
- This ensures that board is completely powered and RCW is loaded properly into the board and it will start booting and LED indications will be as shown in the figure 16

6.2 iNIC card tested Interfaces

S.No	Interface/ Function	Tested (Y/N)	Notes/Comments
1	**Code Warrior JTAG**		
	Connect the Code Warrior JTAG on the H7 connector and check for LS2 detection	**YES**	
2	**DDR4 SODIMM**		
	Access the DDR4 memory space and verify Write/ Read using Code warrior IDE 1. Run U-boot by accessing two DDR4 SODIMM modules 2. Run kernel by accessing two DDR4 SODIMM modules	**YES**	
3	**Parallel NOR Flash**		
	Perform Write / Read on the NOR Flash using JTAG using command interface in Code warrior IDE 1. U-boot code ran from NOR flash 2. Kernel code ran from NOR flash	**YES**	
4	**Boot code programming**		
	Program the boot code through Codewarrior IDE using JTAG	**YES**	
5	**LS2 Booting & UART (Debug)**		
	Check for LS2 boot prompt through micro-USB (UART) port connected to a host PC	**YES**	
6	**PCIe x4 Endpoint interface (PCIe3)**		
	Connect the board to x16 PCIe slot of a mother board and check the LS2 is detected as a PCIe x4 peripheral, GEN3 Speed	**YES**	
7	**PCIe x4 Root complex interface (PCIe2)**		
	Connect Storage card on the HSMC connector, apply power to storage card & check for the Storage card detection as PCIe x4 device	**YES**	
8	**PCIe x4 Root complex interface (PCIe4)**		
	Connect Storage card on the HSMC connector, apply power to storage card & check for the Storage card detection as PCIe x4 device	**YES**	
9	**USB 3.0**		

	Configure USB 3.0 in Host mode and connect a mass storage and check for its detection 1. Test the USB interface in full speed mode (480 Mb/s) 2. Test the USB interface in super speed mode (5 Gb/s)	**YES**	
	Configure USB 3.0 in Device mode and connect to a Host PC and check for its detection	**NO**	Not tested.
10	**SD Card**		
	Connect a SD Card and check for its detection 1. Test SD Card detection in U-boot and Kernel 2. Run kernel image from SD Card	**YES**	
11	**XFI (Port 1 to Port 4)**		
	1. Connect one port SFP+ in NIC card to another SFP+ port in another board 2. ping test in Kernel	**YES**	
12	**SPI Flash**		
	Verify Write / Read	**NO**	No use case Identified for this SPI Flash. So this interface not tested.
13	**Failsafe Boot**		
	Erase the working boot code, insert Jumper on H5 and check for LS2 booting.	**YES**	
14	**EEPROM 1 (0X50)**		
	Verify Write / Read 1. Verify I2C address by i2c_scan command in U-boot 2. Read/write the EEPROM location in u-boot 3. I2C RCW BOOT done from this EEPROM	**YES**	
15	**EEPROM 2 (0X57)**		
	Verify Write / Read 1. Verify I2C address by i2c_scan command in U-boot 2. Read/write the EEPROM location in u-boot	**YES**	
16	**Temperature sensor 1**		
	Read temperature sensor value and cross verify with the Thermometer reading 1. Verify I2C address by i2c_scan command in U-boot 2. Read the temperature value	**YES**	

17	**Temperature sensor 2**		
	Read temperature sensor value and cross verify with the Thermometer reading 1. Verify I2C address by i2c_scan command in U-boot 2. Read the temperature value	**YES**	
18	**QSPI Flash**		
	Verify Write / Read	**NO**	Not tested. Currently no support available in Code warrior to Program this QSPI device.

Table 13: Tested interfaces in the iNIC card

6.3 Storage Card Tested Interfaces:

S.No	Interface/ Function	Tested (Y/N)	Comments
1	**JTAG CPLD**		
	Connect USB Blaster and check for the detection of CPLD	**YES**	
2	**JTAG FPGA**		
	Connect USB Blaster and check for the detection of FPGA	**YES**	
3	**I2C interface**		
	Check the I2C Probe	**YES**	
4	**PCIe x4 Endpoint interface (PCIe 1/2)**		
	Connect Storage card to Gen3 Supported Mother board using add on card and check for the detection as PCIeX4 device Check PCIe Write and read operation	**YES**	
	Connect Storage card on the HSMC connector, apply power to storage card & check for the Storage card detection as PCIe x4 device	**YES**	
5	**Temperature sensor**		
	Check for the detection of "4C" in the I2C Bus	**YES**	
6	**EEPROM**		
	Check for the detection of "50" in the I2C Bus	**YES**	
7	**DDR_NVDIM_M1 Interface**		

	Insert the UDIM DDR3 module and check the SPD through I2C (51)		
	Insert UDIM DDR3 module and do read and write operation	**YES**	
8	**DDR_NVDIM_M2 Interface**		
	Insert the UDIM DDR3 module and check the SPD through I2C (52)		
	Insert UDIM DDR3 module and do read and write operation	**YES**	
9	**DDR_NVDIM_M3 Interface**		
	Insert the UDIM DDR3 module and check the SPD through I2C (53)		
	Insert UDIM DDR3 module and do read and write operation	**YES**	
10	**DDR_NVDIM_M4 Interface**		
	Insert the UDIM DDR3 module and check the SPD through I2C (54)		
	Insert UDIM DDR3 module and do read and write operation	**YES**	
11	**8x SerDes Interface**		
	Connect 2 storage card Through EPLSP cable and Do write and read operation	Yet to verify	

Table 14: Tested interfaces in Storage card

6.4 Mounting the Storage card into iNIC card

This section describes how to mount the Storage card into iNIC card through HSMC connector. User need to follow properly the following steps.

- Hold the iNIC card in the right hand and Storage card in the left hand.
- Align the position of the both the cards i.e., iNIC above storage and gently place Storage card into HSMC connector of the iNIC as shown in the below figure.

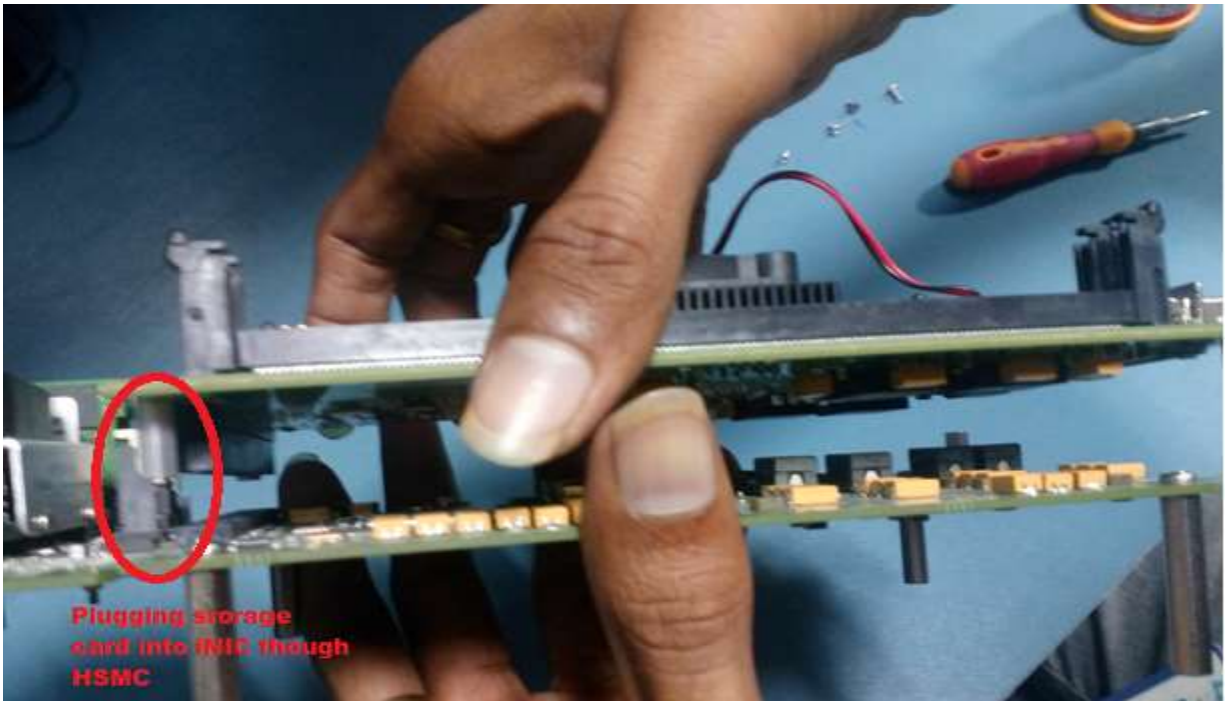


Figure 46: Alignment of both the cards before mounting

- Place the setup down on the flat surface, and then gently press at both the ends of the Storage card as shown below,

Note: please don't apply much force while mounting the card.

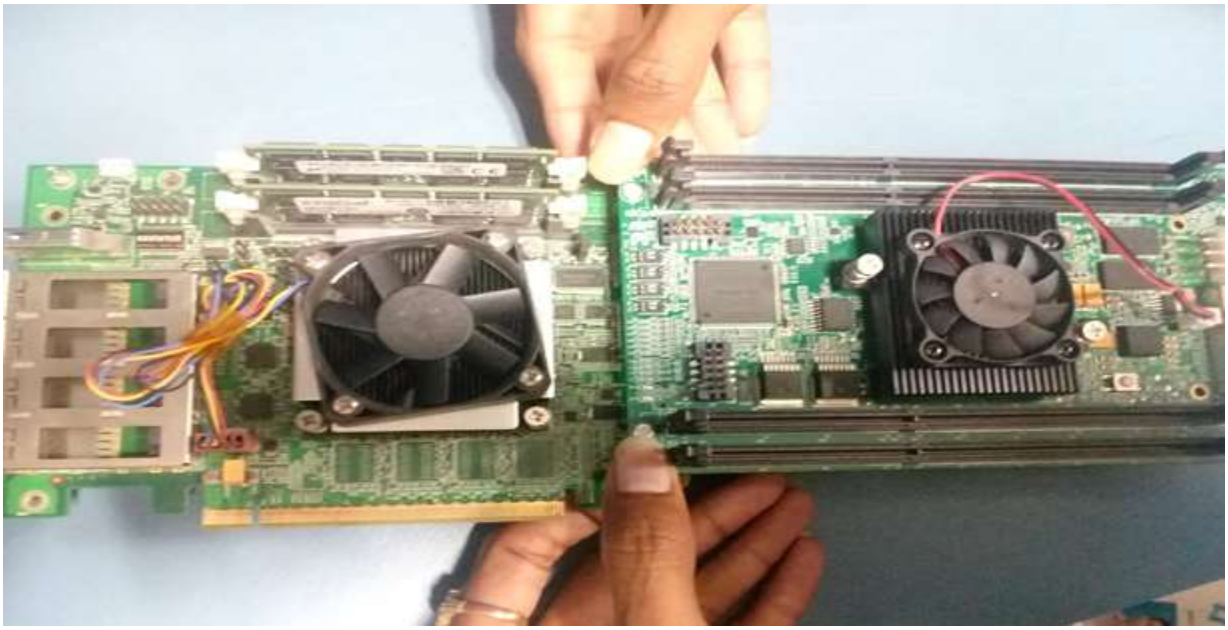


Figure 47: Mounting Storage card properly

- Flip the board then properly insert the screws into the holes of the iNIC as shown in the below figure,

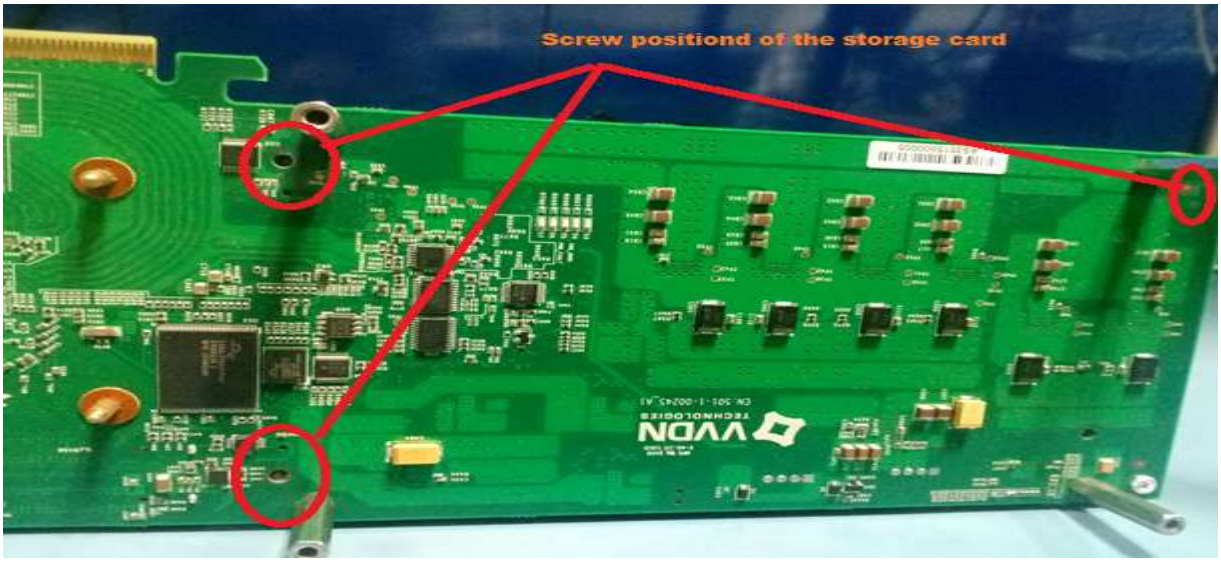


Figure 48: Screw positions of the iSSD

6.5 Board Retention

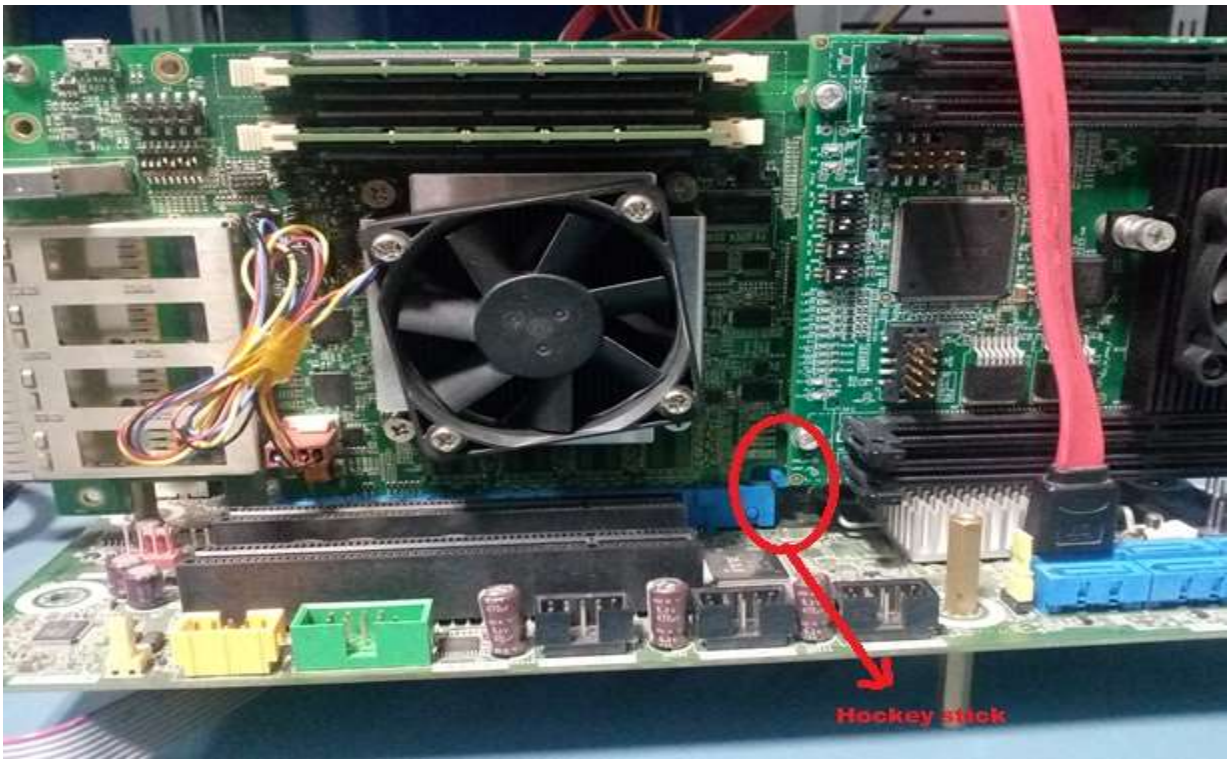


Figure 49: Hockey stick position for safety

Pull the Hockey stick up once the board is plugged into the PCIe slot of the x86 Machine. This is required so that the board will not be tilted or moved from the Mother board if we move it also.

7 FAQs

1). **My x86 machine doesn't detect the LS2 Endpoint.**

Check the H1 Jumper in the iNIC card. The header should hold two Jumper so as to get the PCIe clock from the x86 machine.

2) **How to Flash the Images to Bank4 from the Bank 0 (factory reset state)**

Stop the LS2 at the u-boot prompt. Having the images in the USB follow the commands mentioned in the Section 3.2.3 Flashing images from u-boot (Page No. 32).

3) **If we flash the LS2085 Images to LS2088 boards, will it work?**

No. The images of LS2085 cannot be flashed to the LS2088 board. There is no Backward compatibility.

4) **Versions of storage card Images.**

There are two versions of storage cards. One with A3 FPGA chip and another one with A7 FPGA chip. The Images for two boards are available in https://github.com/DFC-OpenSource/DFC-sdk/tree/LS2088_iSSD/images. If the A7 image is flashed to A3 or vice versa, the FPGA card will not be properly configured.

For A3 FPGA use version number 03.01.00.rpd

For A7 FPGA use version number 03.03.00.rpd

Note: Will keep on updating the Queries asked by customers.

8 Additional Info

1. With SDK_LS2088AISSD_v1.0 When compiling the SDK with fsl-image minimal it will throw error to fetch the dpl_examples git source. To avoid the error, Change the following in .bb file as follows.

- Open the .bb file in the following path *"<Yocto_dir>/sources/meta-freescale/recipes-dpaa2/dpl-examples/dpl-examples_git.bb*".
- Comment the already present SRC_URI.
- Add this instead *SRC_URI = "file://dpl_ls2088aissd_git.tar.bz2;name=tarball"*
- Compile the SDK using fsl-image-minimal