Ultra96-V2 Documentation

Release latest

Jul 26, 2022

BOOTLOADER

1 Bootloader (U-boot) 3
- 1.1 Configure . . . 3
- 1.2 Compile . . . 3
- 1.3 Development . . . 3
- 1.4 Changelog . . . 3

2 Kernel (Linux) 5
- 2.1 Configure . . . 5
- 2.2 Compile . . . 5
- 2.3 Development . . . 5
- 2.4 Changelog . . . 5

3 RootFS (Linux) 7
- 3.1 Build . . . 7
- 3.2 Development . . . 7
- 3.3 Changelog . . . 7

4 Petalinux 9
- 4.1 Build . . . 9
- 4.2 Debug . . . 23
- 4.3 Changelog . . . 23

5 Vitis IDE 25
- 5.1 Build . . . 25
- 5.2 Debug . . . 25

For this course on Xilinx Zynq UltraScale+ MPSoC development, we have selected the Avnet-Ultra96-V2 platform as the reference development kit. There are several advantages in using this platform for an introductory course, enabling the potential reach to a wider audience:

- The Avnet Ultra96-V2 is powered by a Xilinx ZU3EG device that contains a fully featured processing system (PS) and a smaller programmable logic (PL) supported by free Xilinx development licenses.
- The Avnet Ultra96-V2 is one of the cheapest development kits based on the Xilinx Zynq UltraScale+ MPSoC devices, which makes it the ideal choice for both hobbyist and enterprise level courses.
- The Avnet Ultra96-V2 follows the 96 Boards standard promoted by Linaro for advanced ARM based platforms, so it supports a huge range of commercially available mezzanine and expansion cards.

CHAPTER

ONE

BOOTLOADER (U-BOOT)

- *configure*
  How to configure bootloader.
- *compile*
  How to compile bootloader.
- *development*
  Guidelines and release planning and check dependencies.
- *changelog*
  The bootloader development changelog.

1.1 Configure

1.1.1 1. —

1.1 —

1.1.1 —

1.2 Compile

1.3 Development

1.4 Changelog

CHAPTER

TWO

KERNEL (LINUX)

- *configure*
  How to configure kernel.
- *compile*
  How to configure kernel.
- *development*
  Guidelines and release planning and check dependencies.
- *changelog*
  The kernel development changelog.

2.1 Configure

2.2 Compile

2.3 Development

2.4 Changelog

CHAPTER

THREE

ROOTFS (LINUX)

- *build*
  How to create a rootfs image
- *development*
  Guidelines and release planning and check dependencies.
- *changelog*
  The rootfs development changelog.

3.1 Build

3.2 Development

3.3 Changelog

CHAPTER

FOUR

PETALINUX

- *Build*
  How to build in Petalinux
- *debug*
  Guidelines and release planning and check dependencies.
- *changelog*
  The petalinux development environment changelog.

4.1 Build

4.1.1 1. Setup Build Environment

1.1 Docker

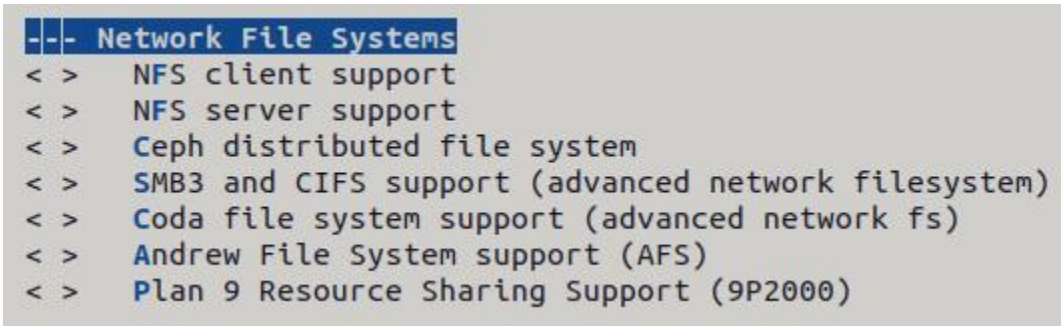
1.2 NFS

Host Server

1.2.1

- NFS

```
$ petalinux-config -c kernel
```



```
-|-|- Network File Systems
<*>   NFS client support
<*>     NFS client support for NFS version 2
<*>     NFS client support for NFS version 3
[ ]       NFS client support for the NFSv3 ACL protocol extension
<*>     NFS client support for NFS version 4
[ ]     Provide swap over NFS support
[ ]   NFS client support for NFSv4.1
[ ]   Root file system on NFS
[ ]   Use the legacy NFS DNS resolver
[*]   NFS: Disable NFS UDP protocol support
< >   NFS server support
[ ]   RPC: Enable dprintk debugging
< >   Ceph distributed file system
< >   SMB3 and CIFS support (advanced network filesystem)
< >   Coda file system support (advanced network fs)
< >   Andrew File System support (AFS)
< >   Plan 9 Resource Sharing Support (9P2000)
```

```
$ petalinux-config -c kernel -x build
```

1.2.2 NFS

```
$ sudo apt insall nfs-kernel-server
$ sudo apt install nfs-common
```

1.2.3 NFS

-

```
$ sudo vim /etc/exports

/nfs *(rw,sync,no_subtree_check,no_root_squash)
```

Table 1: Options

Option	Description
/nfs	( )
IP	IP (*:)
rw	Stands for read / write (ro: Read-only file system)
sync	Requires changes to be written to the disk beefore the are applied
no_subtree_check	Eliminates subtree checking
no_root_squash	root

- NFS

```
$ sudo exportfs -a
$ sudo exportfs -v    # or showmount -e

/nfs            <world>(sync,wdelay,hide,no_subtree_check,sec=sys,rw,secure,no_root_
↪squash,no_all_squash)

$ service nfs-kernel-server restart    # stop -> start
```

Target Board Client

1.2.4 NFS

- /proc/filesystems NFS

```
$ cat /proc/filesystems

nodev    nfs
nodev    nfs4
```

```
root@petalinux_u96v2_gem:~# cat /proc/filesystems
nodev   sysfs
nodev   tmpfs
nodev   bdev
nodev   proc
nodev   cgroup
nodev   cgroup2
nodev   cpuset
nodev   devtmpfs
nodev   configfs
nodev   debugfs
nodev   sockfs
nodev   pipefs
nodev   ramfs
nodev   hugetlbfs
nodev   rpc_pipefs
nodev   devpts
        ext3
        ext2
        ext4
        cramfs
        vfat
        msdos
        iso9660
nodev   ecryptfs
nodev   nfs
nodev   nfs4
nodev   cifs
nodev   smb3
nodev   jffs2
nodev   autofs
nodev   overlay
nodev   mqueue
nodev   gadgetfs
        btrfs
root@petalinux_u96v2_gem:~#
```

1.2.5 NFS

- NFS

```
$ petalinux-config -c rootfs
```

```
                                    Search Results
Symbol: nfs-utils [=y]
Type  : boolean
Prompt: nfs-utils
  Location:
    -> Filesystem Packages
      -> console
        -> network
(1)       -> nfs-utils
  Defined at /home/louis/work/project/mobis/tools/xilinx/petalinux/2021.2/bsp/petalinux_u96v
```

```
$ petalinux-build -c rootfs -x build
```

1.2.6 NFS

-

```
$ sudo vi /etc/fstab

x.x.x.x:/nfs /mnt/nfs nfs defaults 0 0
```

1.3 CIFS

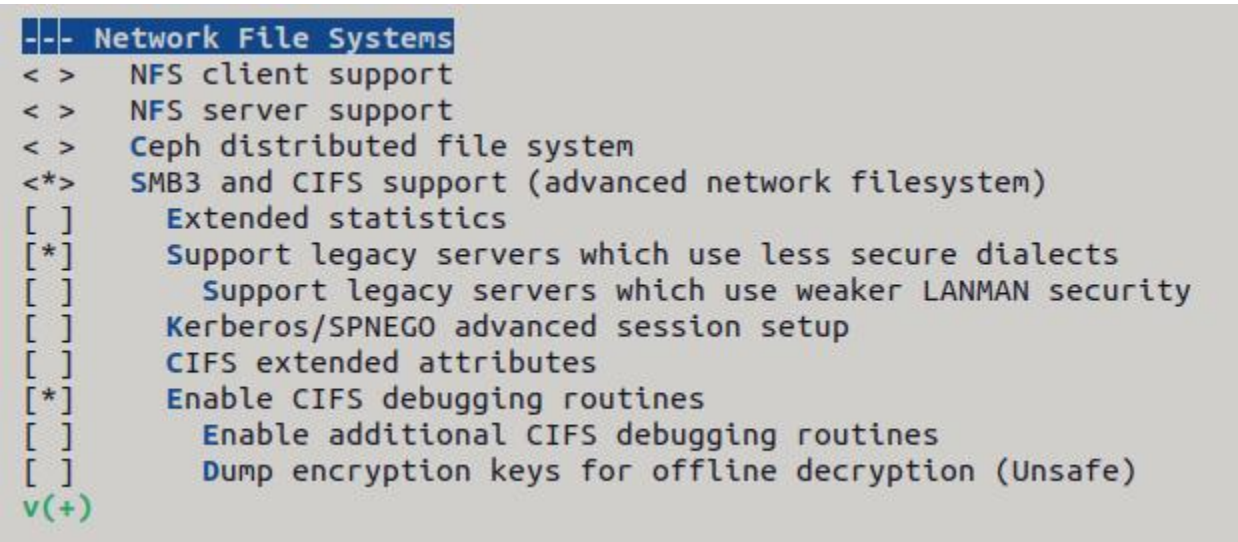
Host Server

1.3.1

- CIFS

```
$ petalinux-config -c kernel
```

```
-*- Network File Systems
< >   NFS client support
< >   NFS server support
< >   Ceph distributed file system
< >   SMB3 and CIFS support (advanced network filesystem)
< >   Coda file system support (advanced network fs)
< >   Andrew File System support (AFS)
< >   Plan 9 Resource Sharing Support (9P2000)
```




```
$ petalinux-config -c kernel -x build
```

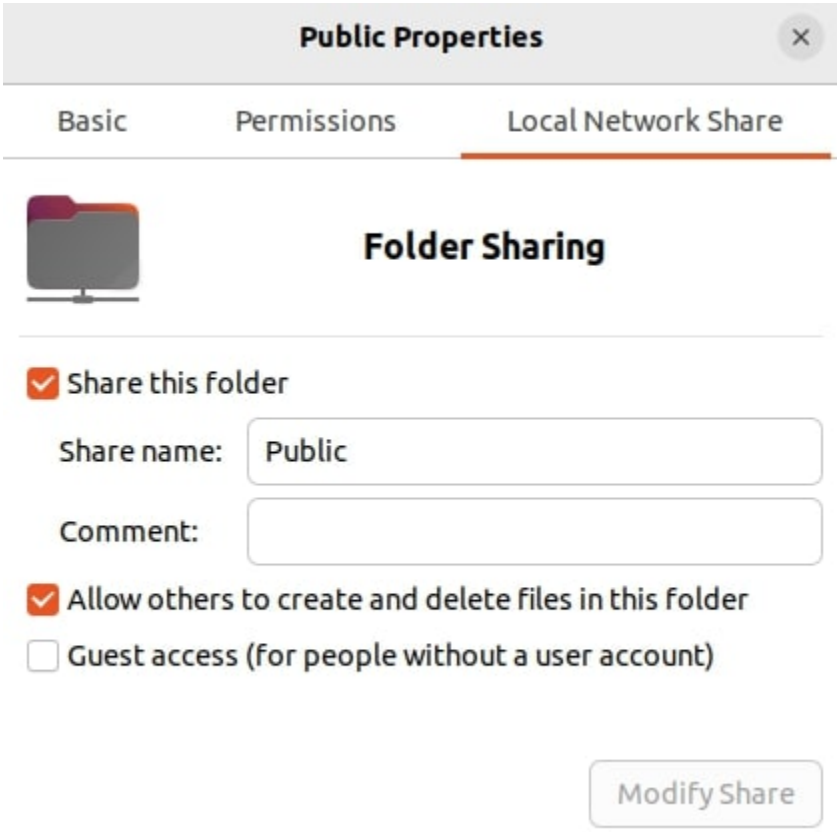
1.3.2 CIFS

```
$ sudo apt insall cifs-utils
```

1.3.3 Samba

```
$ sudo smbpasswd -a [ID]
```

1.3.4



Target Board Client

1.3.5 CIFS

- CIFS

```
$ vim ./components/yocto/layers/meta-petalinux/recipes-core/images/petalinux-image-user.
↪bb
```

```
DESCRIPTION = "OSL image definition for Xilinx boards"
LICENSE = "MIT"

require petalinux-image-user.inc
```

```
$ vim ./components/yocto/layers/meta-petalinux/recipes-core/images/petalinux-image-user.
↪inc

#
cifs-utils \
nfs-utils \
```

```
COMMON_INSTALL = " \
    openssh-sftp-server \
    tcf-agent \
    mtd-utils \
    bridge-utils \
    can-utils \
    pciutils \
    kernel-modules \
    cifs-utils \
    nfs-utils \
    ethtool \
    "
```

1.2.6 CIFS

- CLI

```
$ mount -t cifs -o user=xxx,password=xxx \\x.x.x.x\cifs /mnt/cifs
# mount -t cifs -o user=[ID],password=[PASSWORD] \\[IP]\[ ] [ ]
```

-

```
$ sudo vi /etc/fstab

//x.x.x.x/Public /mnt/cifs cifs user=xxx,password=xxx,_netdev 0 0
```

4.1.2 2. Compile

```
$ source ./settings.sh
$ petalinux-build    # Full build
```

2.1 Bootloader Compile

```
$ petalinux-build -c u-boot -x clean
$ petalinux-build -c u-boot -x cleansstate
$ petalinux-build -c u-boot -x mrproper
$ petalinux-config -c u-boot
$ petalinux-build -c u-boot -x build
```

2.2 Kernel Compile

```
$ petalinux-build -c kernel -x clean
$ petalinux-build -c kernel -x cleansstate
$ petalinux-build -c kernel -x mrproper
$ petalinux-config -c kernel
$ petalinux-build -c kernel -x build
```

4.1.3 3. Create Rootfs

Small rootfs:

```
$ vim ./components/yocto/layers/meta-petalinux/recipes-core/images/petalinux-image-user.
↪bb
$ vim ./components/yocto/layers/meta-petalinux/recipes-core/images/petalinux-image-user.
↪inc

$ petalinux-build -c petalinux-image-user -x clean
$ petalinux-build -c petalinux-image-user -x cleansstate
$ petalinux-build -c petalinux-image-user -x mrproper
$ petalinux-build -c petalinux-image-user -x build
```

Normal rootfs:

```
$ petalinux-build -c rootfs -x clean
$ petalinux-build -c rootfs -x cleansstate
$ petalinux-build -c rootfs -x mrproper
$ petalinux-config -c rootfs
$ petalinux-build -c rootfs -x build
```

Mount rootfs:

```
$ mkdir rootfs/
$ sudo mount -t ext4 rootfs.ext4 rootfs/
$ ls rootfs/
$ sudo umount rootfs/
```

4.1.4 4. Create Boot Images

```
$ cd ./petalinux_u96v2/bsp/images/linux
$ petalinux-package --boot --fsbl zynqmp_fsbl.elf --fpga system.bit --pmufw pmufw.elf --
↪u-boot --force
```

4.1.5 5. Flash Images

```
$ cd ./petalinux_u96v2/bsp/images/linux
```

5.1 INITRAMFS

- RAM-based File System (INITRAMFS, JTAG)

1. Petalinux Configuration for INITRAMFS

```
$ petalinux-config
```

```
                                    Search Results
Symbol: SUBSYSTEM_ROOTFS_INITRAMFS [=y]
Type  : boolean
Prompt: INITRAMFS
  Location:
    -> Image Packaging Configuration
(1)   -> Root filesystem type (<choice> [=y])
  Defined at /home/louis/work/project/mobis/tools/xilinx/petalinux/2021.2/bsp/petalinux_u96v
  Depends on: <choice>
```

```
    Root filesystem type (INITRAMFS)  --->
(petalinux-image-user) INITRAMFS/INITRD Image name
(image.ub) name for bootable kernel image
(cpio cpio.gz cpio.gz.u-boot ext4 tar.gz jffs2) Root filesystem formats
(0x1000) DTB padding size
[*] Copy final images to tftpboot
(/tftpboot) tftpboot directory
```

1. Create RootFS

```
$ petalinux-build -c petalinux-image-user -x build
$ mkdir initramfs
$ tar -xf rootfs.tar.gz -C initramfs
or
$ sudo mount -t ext4 rootfs.ext4 initramfs
```

1. Kernel Configuration for INITRAMFS

```
$ petalinux-config -c kernel
```

```
Symbol: BLK_DEV_RAM [=y]
Type  : tristate
Defined at drivers/block/Kconfig:295
  Prompt: RAM block device support
  Depends on: BLK_DEV [=y]
  Location:
    -> Device Drivers
(1)   -> Block devices (BLK_DEV [=y])
```

```
    Scheduler features  ----
[*] Control Group support  --->
[*] Namespaces support  --->
[*] Checkpoint/restore support
[ ] Automatic process group scheduling
[ ] Enable deprecated sysfs features to support old userspace too
[ ] Kernel->user space relay support (formerly relayfs)
-*- Initial RAM filesystem and RAM disk (initramfs/initrd) suppor
(/home/louis/work/project/mobis/tools/xilinx/petalinux/2021.2/bsp
(0)   User ID to map to 0 (user root)
```

```
< >   Network block device support
< >   STEC S1120 Block Driver
< >   Promise SATA SX8 support
<*>   RAM block device support
(16)    Default number of RAM disks
(65536) Default RAM disk size (kbytes)
< >   Packet writing on CD/DVD media (DEPRECATED)
< >   ATA over Ethernet support
<*>   Xen virtual block device support
<M>   Xen block-device backend driver
```

1. Modify boot argument

```
$ vim ./project-spec/meta-user/recipes-bsp/device-tree/files/
↪system-user.dtsi
```

Note: Modify 'chosen' node.

```
bootargs = "earlycon console=ttyPS0,115200 clk_ignore_unused
root=/dev/ram0 rw rootwait quiet
```

---

2. Apply the modification to DTB and Create linux kernel image included RooFS

```
$ petalinux-build -c kernel -x build
```

Note: chosen DTB DTS .

```
$ dtc -I dtb -O dts -f system.dtb -o system.dts
```

2. Create BOOT.BIN

```
$ petalinux-package --boot --fsbl zynqmp_fsbl.elf --fpga design_1_wrapper.
↪bit --pmufw pmufw.elf --u-boot --force
```

3. JTAG Downloads

```
$ petalinux-boot --jtag --u-boot --fpga --bitstream design_1_wrapper.bit --
↪pmufw pmufw.elf
$ petalinux-boot --jtag --kernel --fpga --bitstream design_1_wrapper.bit --
↪pmufw pmufw.elf
```

```
INFO: Configuring the FPGA...
INFO: Downloading bitstream: design_1_wrapper.bit to the target.
INFO: Downloading ELF file: /home/louis/work/project/mobis/tools/xilinx/petalinux/2021.2/bsp/docker-p
etalinux_u96v2_gem/images/linux/pmufw_custom.elf to the target.
INFO: Downloading ELF file: /home/louis/work/project/mobis/tools/xilinx/petalinux/2021.2/bsp/docker-p
etalinux_u96v2_gem/images/linux/zynqmp_fsbl.elf to the target.
INFO: Loading image: /home/louis/work/project/mobis/tools/xilinx/petalinux/2021.2/bsp/docker-petalinu
x_u96v2_gem/images/linux/Image at 0x00200000
INFO: Loading image: /home/louis/work/project/mobis/tools/xilinx/petalinux/2021.2/bsp/docker-petalinu
x_u96v2_gem/images/linux/system.dtb at 0x00001000
INFO: Downloading ELF file: /home/louis/work/project/mobis/tools/xilinx/petalinux/2021.2/bsp/docker-p
etalinux_u96v2_gem/build/misc/linux-boot/linux-boot.elf to the target.
INFO: Loading image: /home/louis/work/project/mobis/tools/xilinx/petalinux/2021.2/bsp/docker-petalinu
x_u96v2_gem/images/linux/boot.scr at 0x20000000
INFO: Downloading ELF file: /home/louis/work/project/mobis/tools/xilinx/petalinux/2021.2/bsp/docker-p
etalinux_u96v2_gem/images/linux/bl31.elf to the target.
```

```
$ df -h
```

```
Filesystem                Size      Used Available Use% Mounted on
devtmpfs                558.0M         0    558.0M   0% /dev
tmpfs                   832.2M     96.0K    832.1M   0% /run
tmpfs                   832.2M    124.0K    832.1M   0% /var/volatile
/dev/mmcblk0p1          125.0M     49.2M     75.8M  39% /media/sd-mmcblk0p1
/dev/mmcblk0p2           14.1G    147.8M     13.3G   1% /media/sd-mmcblk0p2
root@petalinux_u96v2_gem:~#
```

Note: PMUFW / FSBL bitstream XSA (Export Hardware) Vitis IDE ‘Platform project’ .

Warning: `--u-boot` u-boot CLI linux kernel RooFS .

5.2 eMMC

- Flash-based File System (eMMC, JTAG)

1. Petalinux Configuration for Ext4 File System

```
$ petalinux-config
```

```
                                    Search Results
Symbol: SUBSYSTEM_ROOTFS_EXT4 [=y]
Type  : boolean
Prompt: EXT4 (SD/eMMC/SATA/USB)
  Location:
    -> Image Packaging Configuration
(1)   -> Root filesystem type (<choice> [=y])
  Defined at /home/louis/work/project/mobis/tools/xilinx/petalinux/2021.2/bsp/petalinux_u96v
  Depends on: <choice>
```

```
||  Root filesystem type (EXT4 (SD/eMMC/SATA/USB))  --->
(/dev/mmcblk0p2) Device node of SD device (NEW)
(image.ub) name for bootable kernel image
(cpio cpio.gz cpio.gz.u-boot ext4 tar.gz jffs2) Root filesystem formats
(0x1000) DTB padding size
[*] Copy final images to tftpboot
(/tftpboot) tftpboot directory
```

```
$ petalinux-build -c petalinux-image-user -x build
```

2. Kernel Configuration for Ext4 File System

```
$ petalinux-config -c kernel
```

```
Symbol: EXT4_FS [=y]
Type  : tristate
Defined at fs/ext4/Kconfig:29
  Prompt: The Extended 4 (ext4) filesystem
  Depends on: BLOCK [=y]
  Location:
(1) -> File systems
Selects: JBD2 [=y] && CRC16 [=y] && CRYPTO [=y] && CRYPTO_CRC32C [=y] &
Selected by [y]:
  - EXT3_FS [=y] && BLOCK [=y]
Selected by [n]:
  - EXT4_KUNIT_TESTS [=n] && BLOCK [=y] && KUNIT [=n]
```

```
<*> The Extended 3 (ext3) filesystem
[ ]   Ext3 POSIX Access Control Lists
[ ]   Ext3 Security Labels
-*- The Extended 4 (ext4) filesystem
[*]   Use ext4 for ext2 file systems
[*]   Ext4 POSIX Access Control Lists
[*]   Ext4 Security Labels
[ ]   Ext4 debugging support
[ ] JBD2 (ext4) debugging support
< > Reiserfs support
```

1. Modify boot argument

```
$ vim ./project-spec/meta-user/recipes-bsp/device-tree/files/
↪system-user.dtsi
```

Note: Modify 'chosen' node.

```
bootargs = "earlycon console=ttyPS0,115200 clk_ignore_unused
root=/dev/mmcblk0p2 rw rootwait quiet
```

2. Apply the modification to DTB and Create linux kernel image included RooFS

```
$ petalinux-build -c kernel -x build
```

Note: chosen DTB DTS .

```
$ dtc -I dtb -O dts -f system.dtb -o system.dts
```

3. Create BOOT.BIN

```
$ petalinux-package --boot --fsbl zynqmp_fsbl.elf --fpga design_1_wrapper.
↪bit --pmufw pmufw.elf --u-boot --force
```

5.2 SD Card

Partition:

```
$ sudo fdisk /dev/sdx
$ sudo fdisk -l

Device     Boot   Start      End  Sectors  Size Id Type
/dev/sdx1          2048  2099199  2097152    1G  c W95 FAT32 (LBA)
/dev/sdx2       2099200 31205375 29106176 13.9G 83 Linux
```

Format:

```
$ sudo mkfs -t ext4 /dev/sdx2
```

Specify mount directory:

```
$ sudo vim /etc/fstab

UUID=5AA3-7D75 /media/louis/SD_BOOT vfat defaults 0 0
UUID=2749244d-79ab-4493-87b1-2dace4105cbb /media/louis/SD_ROOTFS ext4 defaults 0 0
```

Insert SD Card and Check mount info:

```
$ dmesg | tail
$ mount
```

Write boot images `BOOT.BIN` `image.ub` `boot.scr` to BOOT partition:

```
$ sudo cp BOOT.BIN image.ub boot.scr /media/louis/SD_BOOT
```

Write rootfs images `rootfs.ext4` to ROOTFS partition:

```
$ sudo dd if=rootfs.ext4 of=/dev/sdx2
or
$ make rootfs/
$ mount -t ext4 rootfs.ext4 rootfs/
$ sudo cp -rf rootfs/* /media/louis/SD_ROOTFS
$ sync
```

5.4 NFS

Host:

```
$ sudo cp BOOT.BIN boot.scr image.ub /mnt/shared/images/u96v2-v2021.2-images/linux/
$ sudo cp rootfs.ext4 /mnt/shared/images/u96v2-v2021.2-images/linux/
```

Target Board:

```
$ ifconfig eth0 up x.x.x.x or ifup eth0 ( /etc/network/interface )
$ cp /mnt/cifs/images/u96v2-v2021.2-images/linux/BOOT.BIN
$ cp /mnt/cifs/images/u96v2-v2021.2-images/linux/image.ub
$ reboot
```

4.2 Debug

4.3 Changelog

CHAPTER

FIVE

VITIS IDE

- ***Build***
  How to build using Vitis IDE tool
- ***Debug***
  How to debug using Vitis IDE tool

5.1 Build

5.2 Debug