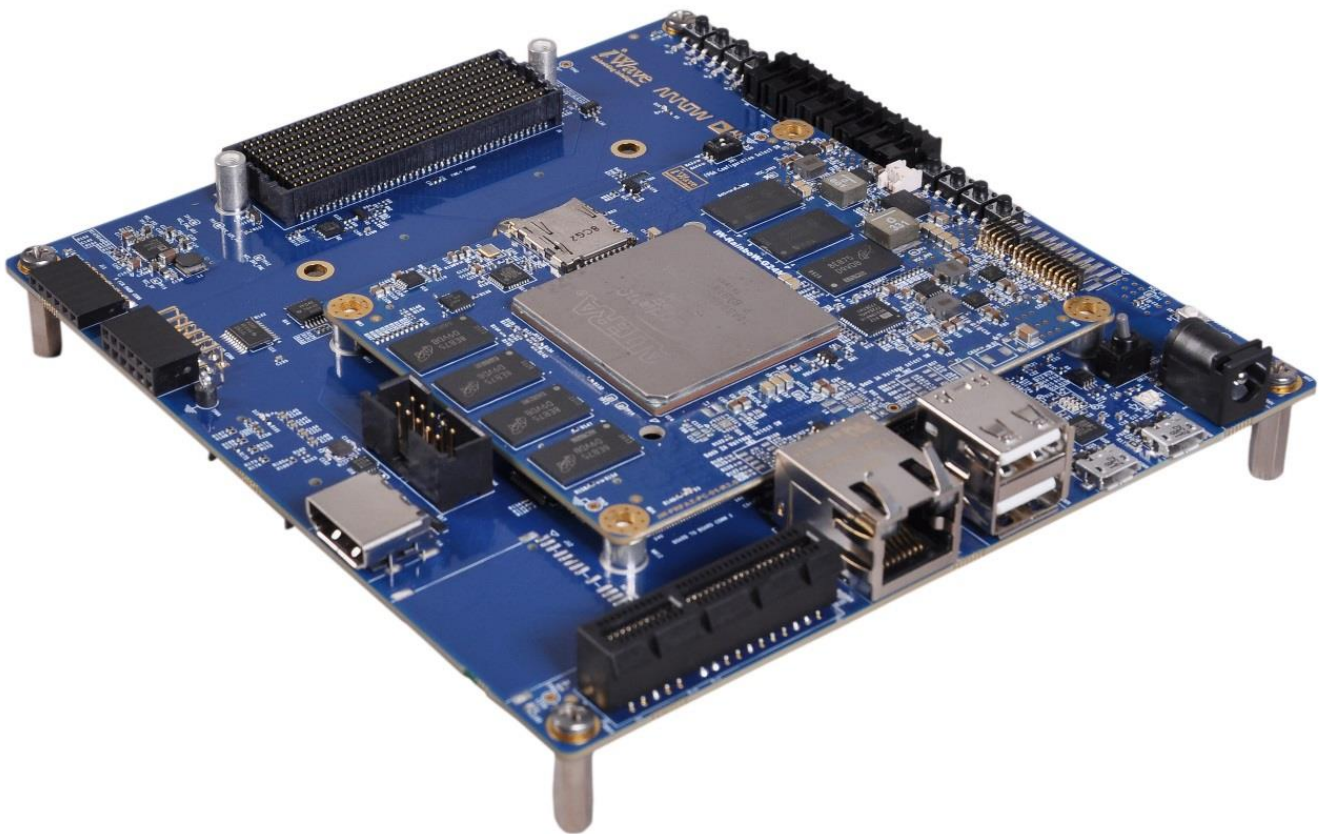


Arria10 SoC/FPGA FMC+ Development platform

Linux User Guide



Document Revision History

Document Number		iW-PRFXR-Arria10_SoC_FPGA-Linux4.9.78-SoftwareUserGuide-REL0.1
Revision	Date	Description
0.1	07 th May 2019	Draft Version

PROPRIETARY NOTICE: This document contains proprietary material for the sole use of the intended recipient(s). Do not read this document if you are not the intended recipient. Any review, use, distribution or disclosure by others is strictly prohibited. If you are not the intended recipient (or authorized to receive for the recipient), you are hereby notified that any disclosure, copying distribution or use of any of the information contained within this document is STRICTLY PROHIBITED. Thank you. "iWave Systems Tech. Pvt. Ltd."

Disclaimer

iWave Systems reserves the right to change details in this publication including but not limited to any Product specification without notice.

No warranty of accuracy is given concerning the contents of the information contained in this publication. To the extent permitted by law no liability (including liability to any person by reason of negligence) will be accepted by iWave Systems, its subsidiaries or employees for any direct or indirect loss or damage caused by omissions from or inaccuracies in this document.

CPU and other major components used in this product may have several silicon errata associated with it. Under no circumstances, iWave Systems shall be liable for the silicon errata and associated issues.

Trademarks

All registered trademarks and product names mentioned in this publication are used for identification purposes only.

Certification

iWave Systems Technologies Pvt. Ltd. is an ISO 9001:2015 Certified Company.



Warranty & RMA

Warranty support for Hardware: 1 Year from iWave or iWave's EMS partner.

For warranty terms, go through the below web link,

<http://www.iwavesystems.com/support/warranty.html>

For Return Merchandise Authorization (RMA), go through the below web link,

<http://www.iwavesystems.com/support/rma.html>

Technical Support

iWave Systems technical support team is committed to provide the best possible support for our customers so that our Hardware and Software can be easily migrated and used.

For assistance, contact our Technical Support team at,

Email : support.ip@iwavesystems.com
Website : www.iwavesystems.com
Address : iWave Systems Technologies Pvt. Ltd.
7/B, 29th Main, BTM Layout 2nd Stage,
Bangalore, Karnataka,
India – 560076

Table of Contents

1. INTRODUCTION	8
1.1 Purpose	8
1.2 Scope	8
1.3 List of Acronyms	8
2. BSP COMPILATION.....	9
2.1 BSP Driver details	9
2.1.1 Driver Source description	9
2.1.2 Device tree source description	9
2.2 Yocto compilation	10
2.2.1 Host Requirements.....	10
2.2.2 Host setup	10
2.2.3 Host package installation.....	10
2.2.4 Yocto project setup and build.....	11
2.3 Standalone compilation	13
2.3.1 Cross-compiler build.....	13
2.3.2 EDS Tool Installation	13
2.3.3 U-boot	17
2.3.4 Linux kernel	21
2.3.5 Kernel Device tree from FPGA Source	22
3. BINARY PROGRAMMING	23
3.1 Requirements.....	23
3.2 Programming the binaries	24
3.3 Core Detection using JTAG.....	25
4. Setting Up the Test Environment.....	29
4.1 Debug Port Setting	29
4.2 Powering ON Arria10 SoC/FPGA FMC+ Development Platform	31
4.3 Test Environment	32
5. U-BOOT TESTING AND BOOT CONFIGURATION.....	34
5.1 Basic commands.....	34
5.2 Basic device Tests.....	35
5.2.1 RAM Test.....	35
5.2.2 SD Test.....	35
5.2.3 Ethernet Test.....	36
5.3 Environment variables settings.....	38
5.3.1 Micro SD boot.....	38
5.3.2 TFTP boot	38
5.3.3 Default Environment Variable.....	39

6. LINUX PERIPHERAL TESTING.....	40
6.1 Block devices Test	40
6.1.1 Block Device Test.....	40
6.2 Network devices Test.....	41
6.2.1 Ethernet Test.....	41
6.3 HID Device Test	43
6.3.1 USB Keyboard.....	43
6.4 PCIe Test.....	44
6.5 NVMe Test.....	44
6.6 FPGA/HPS I/O Test	45
6.6.1 Dipswitch Test	45
6.6.2 LED/PUSH Button Test.....	45
6.7 FMC Loopback test.....	45
6.7.1 FMC transceiver Loopback	45
6.7.2 FMC GPIO Loopback.....	45
6.8 PMOD Loopback test	46
6.9 HDMI Test	46
6.10 Temperature Sensor test	46

List of Figures

Figure 1: EDS tool installation-1.....	14
Figure 2: EDS Tool installation-2	14
Figure 3: EDS Tool installation-3	15
Figure 4: EDS Tool Installation-4	15
Figure 5: EDS Tool Installation-5	16
Figure 6: EDS Tool Installation-6	16
Figure 7: BSP EDITOR-1	17
Figure 8: BSP EDITOR-2	18
Figure 9: BSP EDITOR-3	18
Figure 10: BSP EDITOR-4	19
Figure 11: BSP EDITOR-5	19
Figure 12: BSP EDITOR-6	20
Figure 13: Boot device memory layout.....	23
Figure 14: JTAG-1	25
Figure 15: JTAG-2	26
Figure 16: JTAG-3	26
Figure 17: JTAG-4	27
Figure 18: JTAG-5	27
Figure 19: JTAG-6	28
Figure 20: JTAG-7	28
Figure 21 : Debug Port Connection	29
Figure 22 : Power Supply Connection	31
Figure 23 : U-Boot On Terminal	32
Figure 24 : U-Boot Command Prompt	32
Figure 25 : Linux Command Prompt	33
Figure 26 : HDMI Screen	46

List of Tables

Table 1: Acronyms & Abbreviations..... 8

Table 2: Driver Source Path..... 9

Table 3: Device tree source..... 9

1. INTRODUCTION

1.1 Purpose

The purpose of this document is to help the software engineer to program and test the iW-RainboW-G24M-Arria10 FMC+ development platform and this will also guide to configure the Linux development environment in the Host PC and build the board support package.

1.2 Scope

The document describes the Programming Tool, U-Boot, Linux Operating systems and related software installed on the iW-RainboW-G24M-Arria10 FMC+ development platform. The Linux BSP is the collection of binaries, source code and support files that can be used to create a Linux kernel image and a root file system for iW-RainboW-G24M-Arria10 FMC+ development platform.

1.3 List of Acronyms

The following acronyms will be used throughout this document.

Table 1: Acronyms & Abbreviations

Acronyms	Abbreviations
BSP	Board Support Package
CPU	Central Processing Unit
DDR4	Double Data Rate 4
FMC	FPGA Mezzanine Card
I2C	Inter-Integrated Circuit
Kbps	Kilobits per second
MAC	Media Access Controller
MB	Mega Byte
Mbps	Megabits per sec
MHz	Mega Hertz
MMC	Multi Media Card
NVMe	Non-Volatile memory Express
PCIe	Peripheral Component Interconnect Express
PMOD	Peripheral Module interface
RGMII	Reduced Gigabit Media Independent Interface
SD	Secure Digital
SOM	System On Module
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
USB OTG	USB On The Go

2. BSP COMPILATION

2.1 BSP Driver details

This section explains about the features supported, the device driver details and path of the device drivers and device tree details in the Arria10 SoC BSP.

2.1.1 Driver Source description

This section explains about the device driver details and the corresponding path in the Linux BSP of Arria10 SOM.

Table 2: Driver Source Path

File Path	Description
Micro SD	
drivers/mmc/host/dw_mmc-pltfm.c	Micro SD driver controller file
Ethernet	
drivers/net/ethernet/stmicro/stmmac/dwmac-socfpga.c	Ethernet driver controller
USB	
drivers/usb/dwc2/platform.c	USB driver controller
Transceivers	
drivers/fmc/fmc_transceiver.c	Transceiver checker and generator test module
Thermal Sensor	
drivers/thermal/iw-thermal.c	Thermal Sensor driver
HPS/FPGA LED, PUSH Button, Dipswitch	
drivers/gpio/gpio-iwg24m_fpga.c	LED, Push button and Dipswitch driver
drivers/gpio/gpio-iwg24m_hps.c	
PCIe	
drivers/pci/host/pcie-altera.c	PCIe controller driver
HDMI	
drivers/video/fbdev/adv7511_altvip.c	HDMI controller driver
drivers/video/fbdev/altvipfb2-plat.c	Frame buffer controller driver

2.1.2 Device tree source description

This section explains about the device tree source code configuration and organization for iW-RainboW-G24M-Arria10 FMC+ development platform. The device tree source codes will be available in below path of the Linux kernel.

[~/arch/arm/boot/dts/](#)

Table 3: Device tree source

File Name	Description
socfpga_arria10_iwg24m_prfxr_sx480.dts	This device tree source includes SoC controller's configuration of Arria10 processors and defines the specific peripheral used in iW-RainboW-G24M-Arria10 FMC+ development platform.

2.2 Yocto compilation

To get the Yocto Project expected behaviour in a Linux Host Machine, the packages and utilities described below must be installed. An important consideration is the hard disk space required in the host machine. For example, when building on a machine running Ubuntu, the minimum hard disk space required is about 50 GB for the X11 backend. It is recommended that at least 120 GB be provided, which is enough to compile any backend.

2.2.1 Host Requirements

- A Linux host PC with Ubuntu version 14.04.
- Root permission on the Development Host.
- Cross compiler package for Arria10 SoC.

2.2.2 Host setup

- This document assumes that Ubuntu PC is used. Not a requirement, but the packages may be named differently and the method of installing them may be different.
- The recommended Ubuntu version is 14.04.
- Minimum hard disk space required is about 50 GB and 4GB of RAM.

Note: Earlier versions may cause the Yocto Project build setup to fail, because it requires python versions which are available only starting with Ubuntu 12.04.

2.2.3 Host package installation

- Open a terminal window and install the below packages in host PC.

```
$ sudo apt-get update
```

```
$ sudo apt-get upgrade
```

```
$ sudo apt-get install phablet-tools sed wget cvs subversion git-core coreutils unzip texi2html texinfo  
libstd1.2-dev docbook-utils gawk python-pysqlite2 diffstat help2man make gcc build-essential g++  
desktop-file-utils chrpath libgl1-mesa-dev libglu1-mesa-dev mercurial autoconf automake groff libtool  
xterm lib32z1 lib32stdc++6
```

2.2.4 Yocto project setup and build

To setup the yocto, follow the procedure given below.

- Set the Git username and create a build directory using below commands.

```
$ git config --global user.name "Your Name"
```

```
$ git config --global user.email "Your Email"
```

```
$ mkdir iwq24m-release-bsp
```

```
$ cd iwq24m-release-bsp
```

- Download the altera.xml file and repo utility using below commands.

```
$ wget http://releases.rocketboards.org/release/2018.05/qsrq/src/altera.xml
```

```
$ wget http://commondatastorage.googleapis.com/git-repo-downloads/repo
```

- Change the permission on repo file using below command.

```
$ chmod 777 repo
```

- Export the PATH to current directory execute below command.

```
$ export PATH=$PATH:~/iwq24m-release-bsp
```

- Download angstrom-manifest using below commands.

```
$ repo init -u git://github.com/Angstrom-distribution/angstrom-manifest -b angstrom-v2017.06-  
yocto2.3
```

- Create a local manifest directory and move altera.xml file to local manifest using below commands.

```
$ mkdir -p .repo/local_manifests
```

```
$ mv altera.xml .repo/local_manifests/
```

- Complete the download using below command.

```
$ repo sync
```

- Set the Machine ID and setup environment using below command.

```
$ MACHINE=iwq24m . ./setup-environment
```

- To add the meta-altera-refdes layer to conf/bblayers.conf execute below command.

```
$ sed -i '/meta-altera/a \ \ ${TOPDIR}\layers\meta-altera-refdes \\' conf/bblayers.conf
```

- To remove the meta-atmel layer and meta-gumstix-community from conf/bblayers.conf execute below command.

```
$ sed -i '/meta-atmel/d' conf/bblayers.conf
```

```
$ sed -i '/meta-gumstix-community/d' conf/bblayers.conf
```

- Un-tar the Source-Code.tar.gz file from deliverables.

```
$ cd <path to iW-PRFXR-RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables>/iW-PRFXR-  
RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables/Source-Code/
```

```
$ tar -xvf Source-Code.tar.gz
```

- The Yocto patch file is located in below path in deliverables.

```
$ <path_to_Source-Code>/Source-Code/Linux/Yocto/PATCH000-iW-PRFXR-SC-01-RX.X-RELX.X-  
YoctoPyro_basic_customization.patch
```

- To apply the Yocto patch file, execute the below commands.

```
$ patch -Np1 < <path_to_Yocto_patch_file>/PATCH000-iW-PRFXR-SC-01-RX.X-RELX.X-  
YoctoPyro_basic_customization.patch
```

- Execute below command to compile kernel images and yocto filesystem for iW-G24M platform.

```
$ bitbake gsr-d-console-image virtual/bootloader
```

- After successful compilation the binaries are placed in below path.

```
<path_to_iwg24m-release-bsp>/deploy/glibc/images/iwg24m/Angstrom-gsr-d-console-image-glibc-ipk-  
v2017.06-iwg24m.rootfs.tar.gz
```

```
<path_to_iwg24m-release-bsp>/deploy/glibc/images/iwg24m/zImage--<linux_revision>-iwg24m-<date &  
time>.bin
```

```
<path_to_iwg24m-release-bsp>/deploy/glibc/images/iwg24m/zImage--<linux_revision>-  
socfpga_arria10_iwg24m_prfxr_sx480--<date & time>.dtb
```

```
<path_to_iwg24m-release-bsp>/deploy/glibc/images/iwg24m/u-boot-arria10-iwg24m-v2014.10-r1.img
```

- Rename the binary files as mentioned below.

```
Angstrom-gsr-d-console-image-glibc-ipk-v2017.06-iwg24m.rootfs.tar.gz -> rootfs.tar.gz
```

```
zImage--<linux revision>-iwg24m-<date & time>.bin -> zImage
```

```
zImage--<linux revision>-socfpga_arria10_iwg24m_prfxr_sx480--<date & time>.dtb ->  
socfpga_arria10_iwg24m_prfxr_sx480.dtb
```

- Bootable U-Boot binary can be created by adding header required by BootROM using mkpimage provided by Altera.
- Change the directory to embedded folder of the EDS tool as mentioned below. Refer Section **EDS Tool Installation** to install EDS tool.

```
$ cd <path to SOCEDS>/embedded
```

- Start an Embedded Command Shell by running the below command.

```
$ ./embedded_command_shell.sh
```

- Copy u-boot-arria10-iwg24m-v2014.10-r1.img file to SOCEDS directory as mentioned below

```
$ cp <path_to_iwg24m-release-bsp>/deploy/glibc/images/iwg24m/u-boot-arria10-iwg24m-v2014.10-  
r1.img <path to SOCEDS directory>/embedded/
```

- To create uboot bootable binary, execute below command.

```
mkpimage -hv 1 -o uboot_w_dtb-mkpimage.bin u-boot-arria10-iwg24m-v2014.10-r1.img u-boot-arria10-  
iwg24m-v2014.10-r1.img u-boot-arria10-iwg24m-v2014.10-r1.img u-boot-arria10-iwg24m-v2014.10-  
r1.img
```

- This above command creates the following binary files in the current folder.

```
uboot_w_dtb-mkpimage.bin - bootable image
```

2.3 Standalone compilation

This section explains procedure and detailed information about compiling the U-boot and Kernel for Arria10 SOC without Yocto.

- Un-tar the Source-Code.tar.gz file from deliverables.

```
$ cd <path to iW-PRFXR-RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables>/iW-PRFXR-  
RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables/Source-Code/  
$ tar -xvf Source-Code.tar.gz
```

2.3.1 Cross-compiler build

This section provides the detailed information and process for building the cross-compiler and this cross-compiler can be used for standalone compilation of Linux kernel.

- Open a terminal window and give the below mentioned commands.

```
$ wget https://releases.linaro.org/archive/14.04/components/toolchain/binaries/gcc-linaro-arm-linux-  
gnueabi-4.8-2014.04_linux.tar.bz2  
$ tar -xjf gcc-linaro-arm-linux-gnueabi-4.8-2014.04_linux.tar.bz2  
$ export PATH=$PATH:~/gcc-linaro-arm-linux-gnueabi-4.8-2014.04_linux/bin  
$ export CROSS_COMPILE=arm-linux-gnueabi-
```

2.3.2 EDS Tool Installation

The Uboot can be generated using EDS tool. This section describes step by step procedure to install the EDS tool.

- Download the SoC Embedded Design Suite “SoCEDSProSetup-18.1.0.222-linux.run” from the below link.
<http://fpgasoftware.intel.com/soceds/>
- For more information on EDS tool refer below link.
http://fpgasoftware.intel.com/soceds/18.1/?edition=pro&platform=linux&download_manager=d1m3

Note: Here SoCEDSProSetup-18.1.0.222-linux.run used.

- Enter into EDS tool downloaded directory and change to executable.

```
$ cd <EDS tool downloaded directory>; chmod +x S SoCEDSProSetup-18.1.0.222-linux.run
```

- Create a directory to install EDS tool.

```
$ mkdir G24M-EDS
```

- Execute the below command to install the EDS tool.

```
$ ./SoCEDSProSetup-18.1.0.222-linux.run
```

- Select next step as mentioned below.

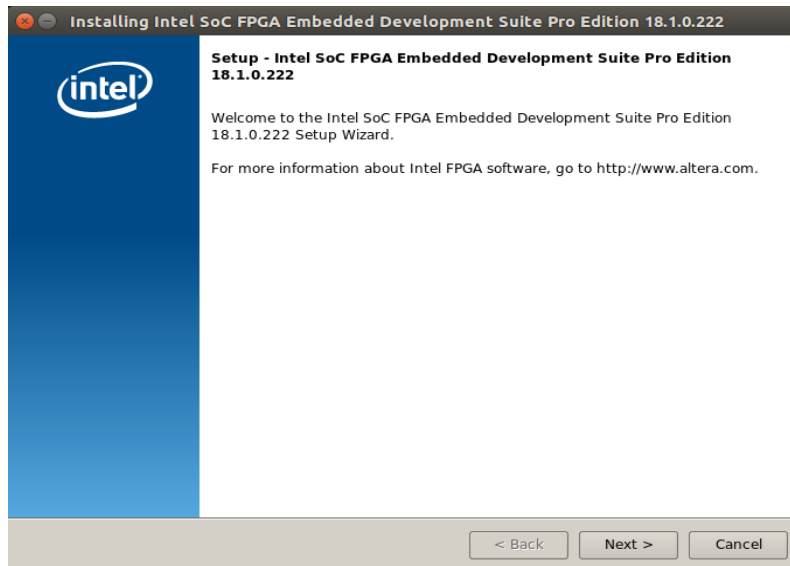


Figure 1: EDS tool installation-1

- Accept the agreement and select next step as mentioned below.

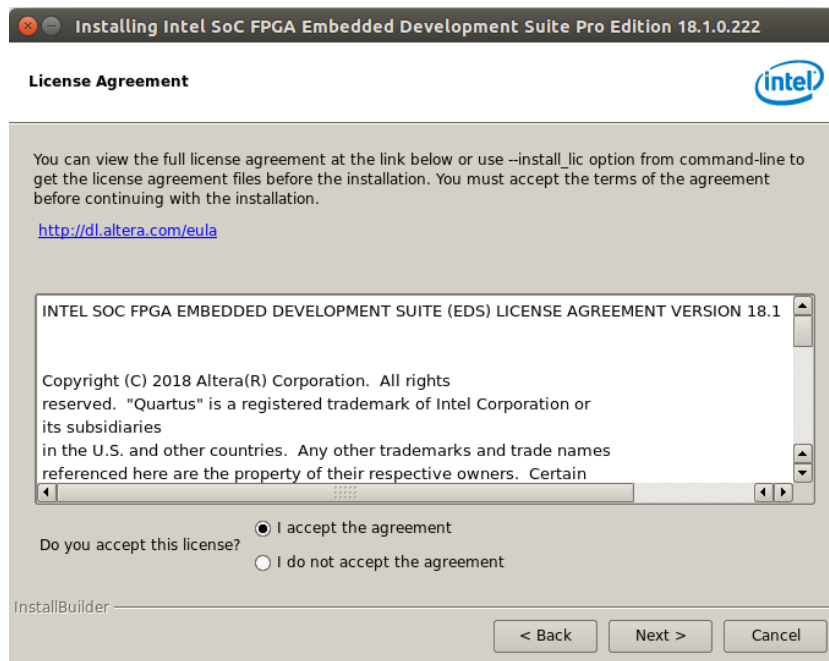


Figure 2: EDS Tool installation-2

- Select a directory to install the EDS tool as mentioned below.

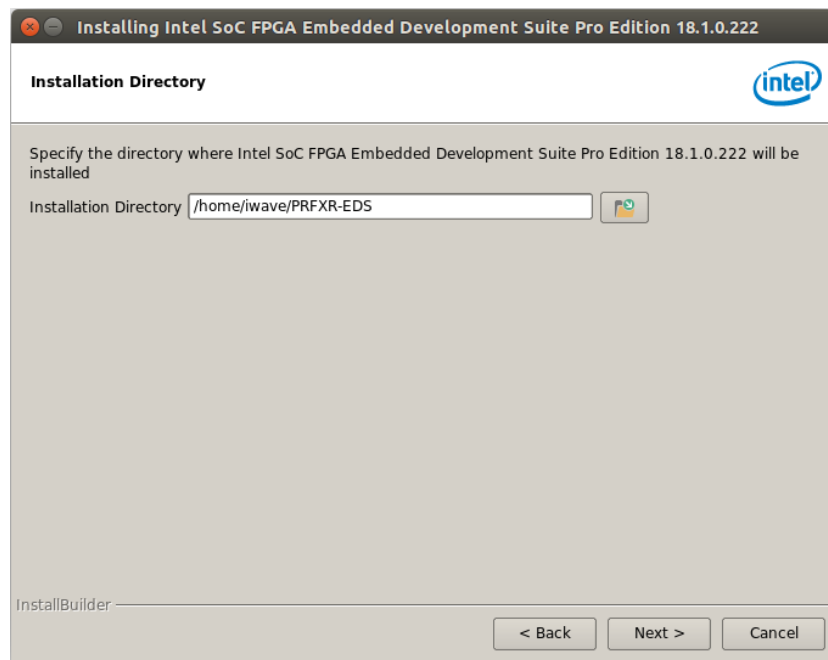


Figure 3: EDS Tool installation-3

- Select SoC Embedded Design Suite (EDS) option as mentioned below.

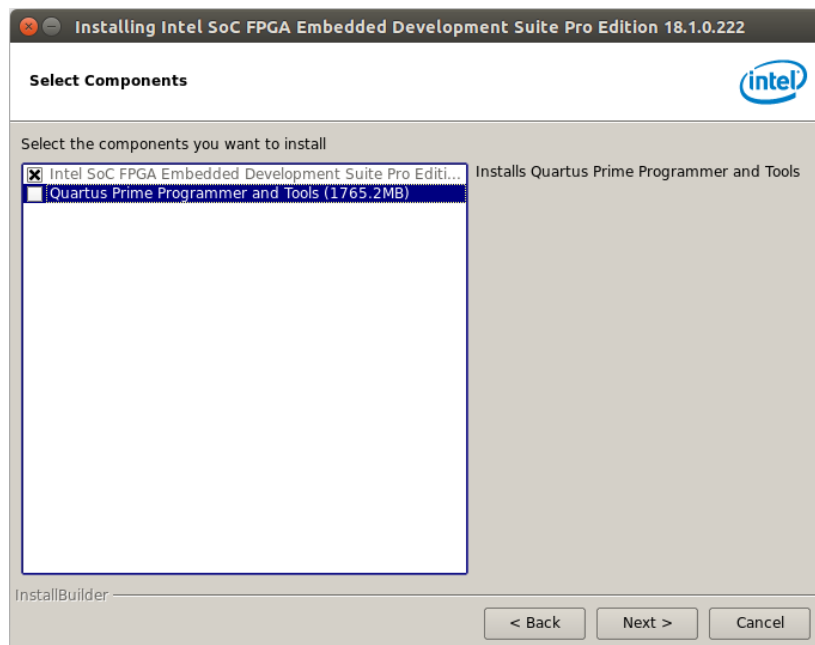


Figure 4: EDS Tool Installation-4

- EDS package will be unpackaged into selected directory as shown below.

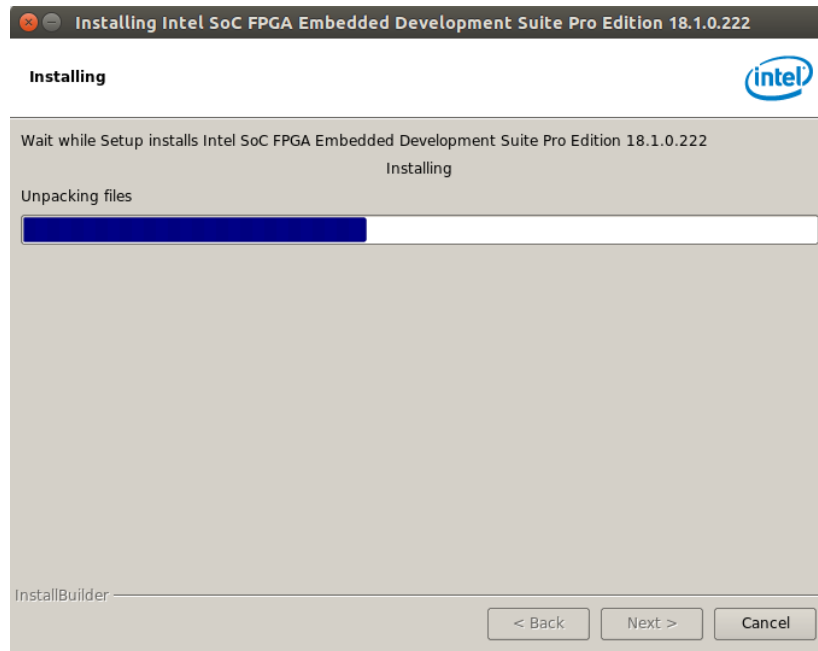


Figure 5: EDS Tool Installation-5

- Un select the Launch DS-5 installation and finish the installation as mentioned below

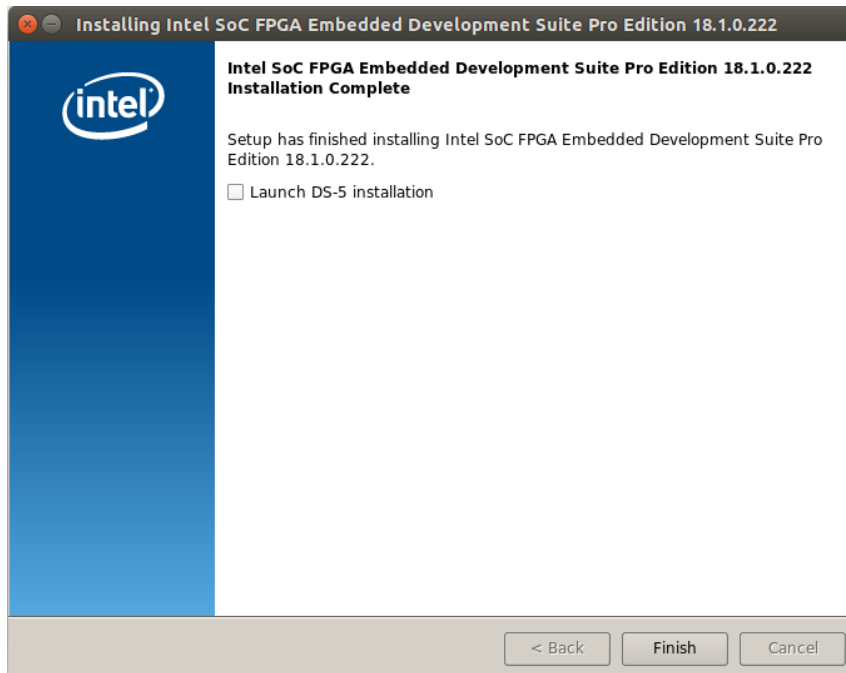


Figure 6: EDS Tool Installation-6

2.3.3 U-boot

This section provides the detailed information and process for building the u-boot binary image. EDS Tool should be installed in the PC.

- Change the directory to embedded folder of the EDS tool.

```
$ cd G24M-EDS/embedded
```

- Start an Embedded Command Shell by running the below command.

```
$ ./embedded_command_shell.sh
```

- Un-tar the FPGA design file from deliverables.

```
$ cd <path to iW-PRFXR-RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables>/iW-PRFXR-RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables/Source-Code/FPGA
```

```
$ tar -xvf iW-PRFXR-SY-01-RX.X-RELX.X-SX480.tar.gz
```

- The hps_isw_handoff folder is located in the below path.

```
<path_to_Source-Code>/Source-Code/FPGA/iW-PRFXR-SY-01-RX.X-RELX.X-SX480/hps_isw_handoff
```

- Copy the hps_isw_handoff folder to below mentioned path.

```
$ cp -rf <path_to_hps_isw_handoff>/hps_isw_handoff examples/hardware/a10_soc_devkit_ghrd/
```

- Start the BSP Editor.

```
$ bsp-editor
```

- In the BSP Editor window, go to File -> New HPS BSP. This opens the New BSP window.

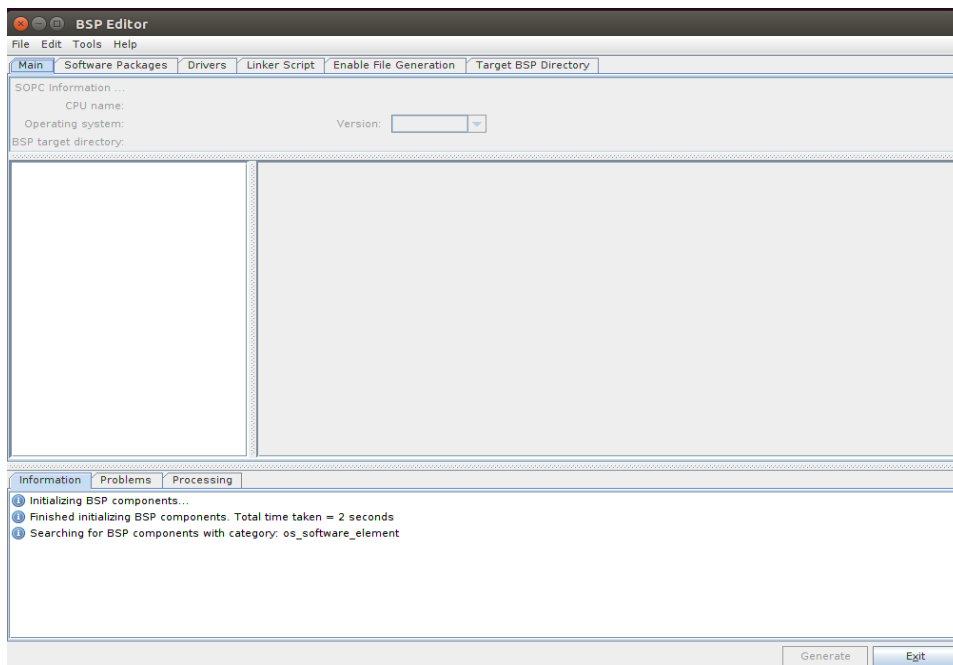


Figure 7: BSP EDITOR-1

- In the New BSP window, click the Browse button for the Preloader Settings Directory. This opens a file browser window.

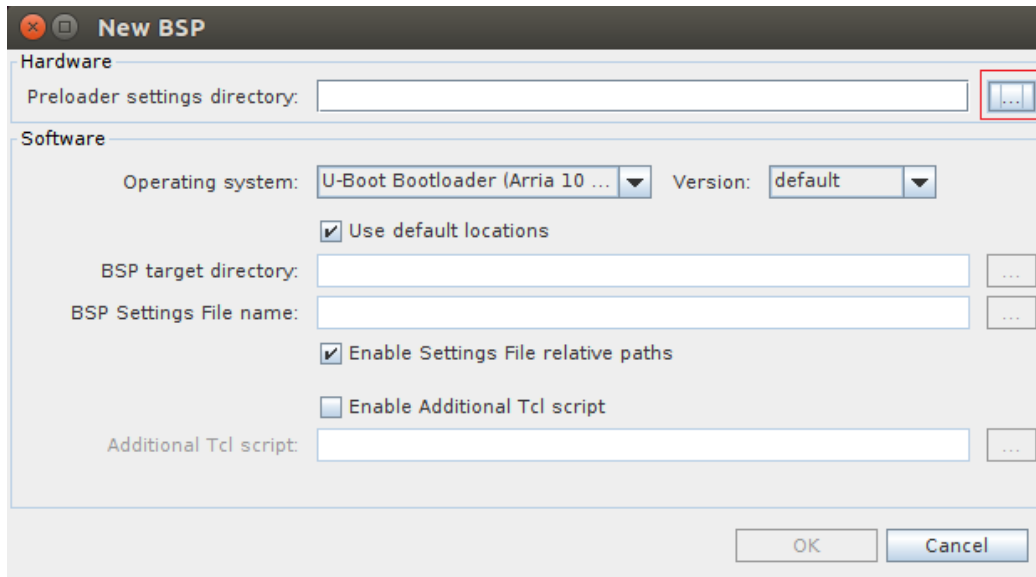


Figure 8: BSP EDITOR-2

- Browse to the folder examples/hardware/a10_soc_devkit_ghrd/hps_isw_handoff and click Open to get back to the New BSP window.

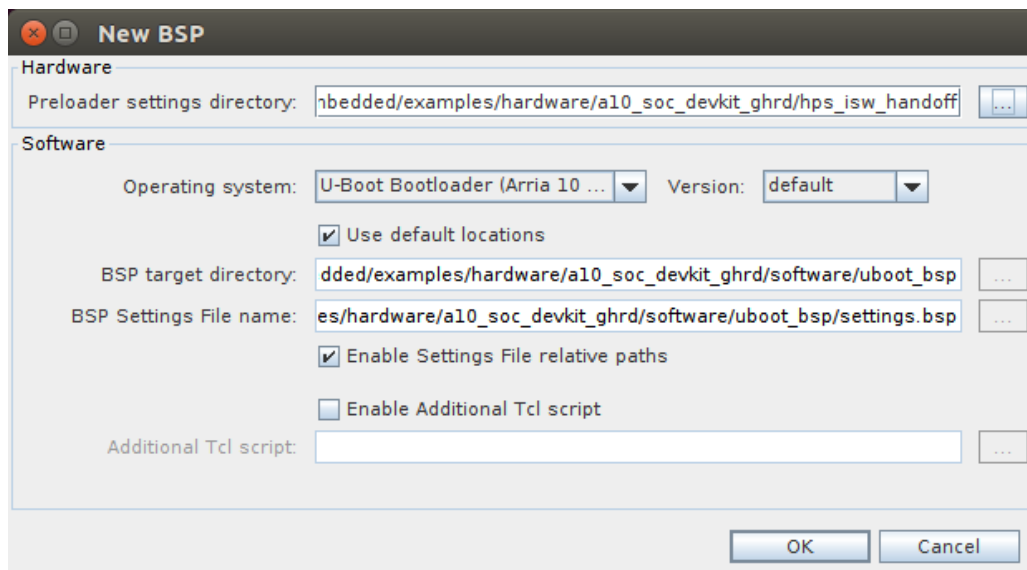


Figure 9: BSP EDITOR-3

- In the New BSP window, select the Operating System to be U-Boot Bootloader (Arria 10 HPS) and leave the Use default locations enabled. This creates the new BSP in the embedded/examples/hardware/a10_soc_devkit_ghrd/software/uboot_bsp folder.

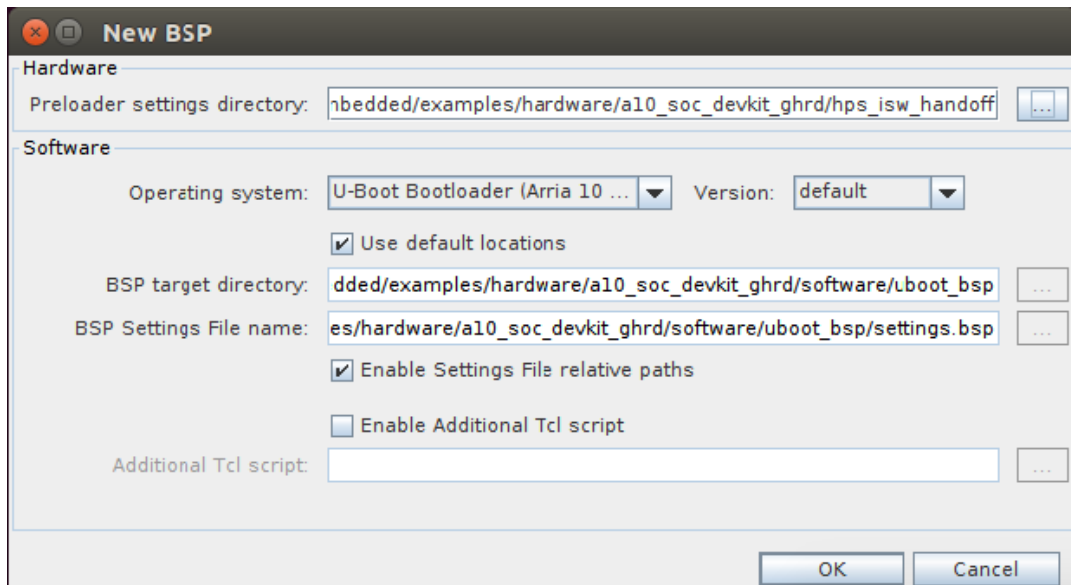


Figure 10: BSP EDITOR-4

- Click OK. That creates the BSP settings file and close the New BSP window
- Edit BSP settings as follows:

boot_device: Boot from SD/MMC (default)

model: iWG24M Arria 10 SOC FPGA FMC+ Development Platform

rbf_filename: ghrd_10as066n2.rbf

disable_uboot_build: unchecked (default)

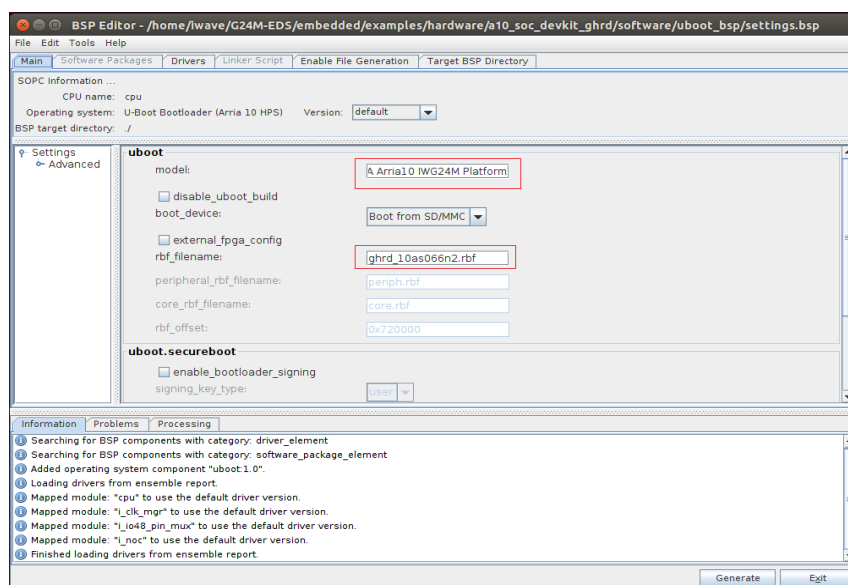


Figure 11: BSP EDITOR-5

- In BSP Editor window, click Generate to generate the source code and the U-Boot Device Tree.

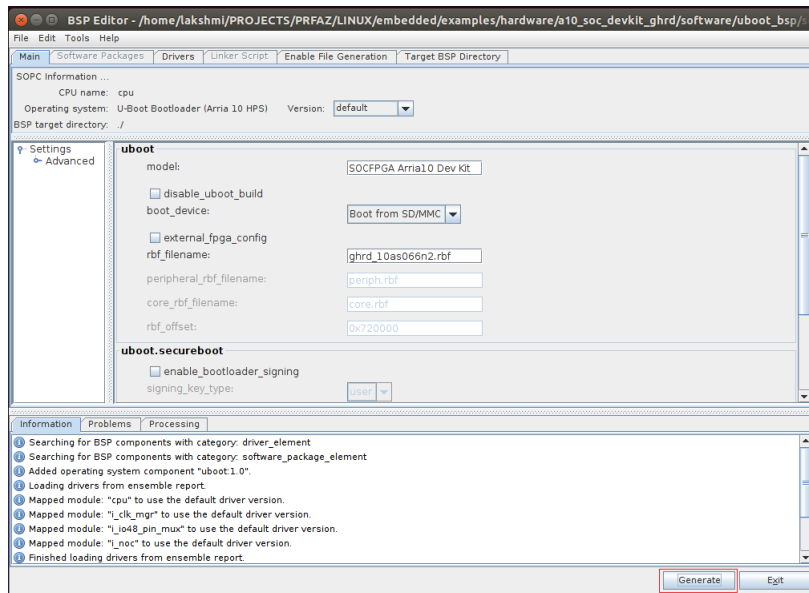


Figure 12: BSP EDITOR-6

- In BSP Editor Window, click Close to close the generator window.
- Go to the U-Boot folder

```
$ cd examples/hardware/a10_soc_devkit_ghrd/software/uboot_bsp
```

- Run make to create uboot-socfpga folder.

```
$ make
```

- Copy the Uboot patch file to the current folder.

```
$ cp <path_to_Source-Code>/Source-Code/U-Boot/PATCH001-iW-PRFXR-SC-01-RX.X-RELX.X-Linux4.9.78-UBoot_basic_customization.patch .
```

- Apply the uboot patch

```
$ make clean
```

```
$ cd uboot-socfpga
```

```
$ patch -Np1 < ../PATCH001-iW-PRFXR-SC-01-RX.X-RELX.X-Linux4.9.78-UBoot_basic_customization.patch
```

```
$ cd ..
```

```
$ make
```

- The above command creates the following binary files in the current folder.

```
uboot_w_dtb-mkpimage.bin - bootable image
```

- Refer the **BINARY PROGRAMMING** section to update the Uboot binary.

2.3.4 Linux kernel

This section provides the detailed information and process for building the kernel images.

- Download the kernel source code by running the below mentioned command.

```
$ git clone https://github.com/altera-opensource/linux-socfpga
```

- Checkout the required version of linux-socfpga using below commands

```
$ cd linux-socfpga
```

```
$ git checkout -b linux-socfpga ACDS18.0_REL_GSRD_PR
```

- Copy the kernel patch file to previous directory.

```
host@host/<directory>~$ cp <path_to_Source-Code>/Source-Code/Linux/Kernel/PATCH002-iW-PRFXR-SC-01-RX.X-RELX.X-Linux4.9.78-Kernel_basic_customization.patch ../
```

- To apply the patch file, execute the below command.

```
host@host/<directory>/linux-socfpga~$ patch -Np1 < ../PATCH002-iW-PRFXR-SC-01-RX.X-RELX.X-Linux4.9.78-Kernel_basic_customization.patch
```

- To export the Cross Compiler and tool chain path, execute the below command.

```
host@host/<Directory>/linux-socfpga ~$ export ARCH=arm
```

```
host@host/<Directory>/linux-socfpga~$ export PATH=$PATH:~/gcc-linaro-arm-linux-gnueabi-4.8-2014.04_linux/bin
```

```
host@host/<Directory> linux-socfpga ~$ export CROSS_COMPILE=arm-linux-gnueabi-
```

- To configure the kernel for iW-RainboW-G24M-Arria10 FMC+ development platform, execute the below command.

```
host@host/<Directory>/linux-socfpga~$ make iw-g24m_arria10_defconfig
```

- To compile the kernel module drivers and kernel image, execute the below commands.

```
host@host/<Directory>/linux-socfpga~$ make
```

- After successful compilation, kernel image (zImage) and device tree (.dtb) will be created in the below path respectively.

```
~/linux-socfpga/arch/arm/boot/zImage
```

```
~/linux-socfpga/arch/arm/boot/dts/socfpga_arria10_iwg24m_prfxr_sx480.dtb
```

- Refer the **BINARY PROGRAMMING** section to update the Linux kernel binary.

2.3.5 Kernel Device tree from FPGA Source

- Un-tar the FPGA design file from deliverables.

```
$ cd <path to iW-PRFXR-RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables>/iW-PRFXR-  
RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables/Source-Code/FPGA  
$ tar -xvf iW-PRFXR-SY-01-RX.X-RELX.X-SX480.tar.gz
```

- The hps_isw_handoff folder is located in the below path.

```
<path_to_Source-Code>/Source-Code/FPGA/iW-PRFXR-SY-01-RX.X-RELX.X-SX480/HPS_SYS/HPS_SYS.sopci  
nfo
```

- Copy the sopcinfo file from the FPGA source to the SOCEDS directory using below command

```
$ cp <path_to_SOPCINFO>/HPS_SYS.sopcinfo <path_to_SOCEDS_directory>/embedded/
```

- Copy the Board XML files SOCEDS directory using below command

```
$ cp <path_to_Source-Code>/Source-Code/FPGA/Board_XML_file/*  
<path_to_SOCEDS_directory>/embedded/
```

- Change the directory to embedded folder of the EDS tool as mentioned below

```
$ cd <path to SOCEDS>/embedded
```

- Start an Embedded Command Shell by running the below command.

```
$ ./embedded_command_shell.sh
```

- To create kernel device tree source, execute the below command

```
$ socp2dts -input HPS_SYS.sopcinfo --output socfpga_arria10_iwg24m_prfxr_sx480.dts --board  
hps_a10_common_board_info.xml --board hps_a10_devkit_board_info.xml --bridge-removal all --clocks
```

Note: The dts file created by the above command is used in the Linux kernel.

- To create device tree blob, execute the below command

```
$ dtc -I dts -O dtb -o socfpga_arria10_iwg24m_prfxr_sx480.dtb socfpga_arria10_iwg24m_prfxr_sx480.dts
```

- The above command creates the following binary in current folder.

```
$ socfpga_arria10_iwg24m_prfxr_sx480.dtb -> Kernel device tree
```

3. BINARY PROGRAMMING

The `make_sdimage.py` is a tool which can be used to create a bootable SD card image. The tool runs in a Linux system and is used to partition the micro SD card and program the binaries in to the card.

3.1 Requirements

To program the binaries for iW-RainboW-G24M-Arria10 FMC+ development platform, following Items are required:

- Micro SD card reader.
- Host PC (Linux).
- Micro SD
- Binary files (`uboot_w_dtb-mkpimage.bin`, `zImage`, `socfpga_arria10_iwg24m_prf_xr_sx480.dtb`, `rootfs.tar.gz`, `ghrd_10as066n2.rbf`)
- The below figure shows the memory layout for the boot device.

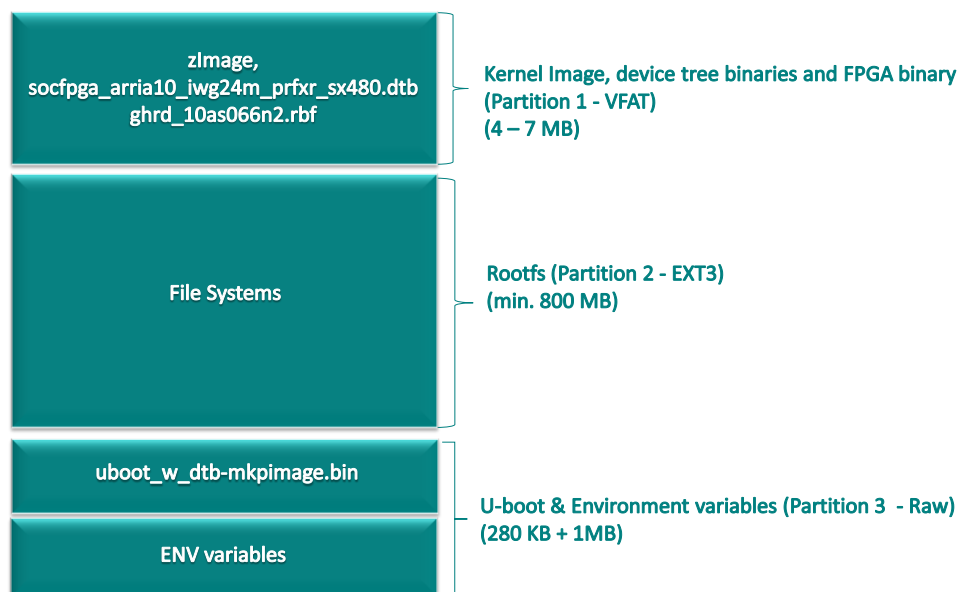


Figure 13: Boot device memory layout

3.2 Programming the binaries

- Connect a Micro SD card to a host PC using Card Reader.
- The programming tool is available in the below path of the deliverables.

```
<path_to_iW-PRFXR-RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables>/iW-PRFXR-RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables/Programming Tool/make_sdimage.py
```

- Create a folder Programming_Tool and Change the directory to Programming Tool.

```
host@host~$ mkdir Programming_Tool
```

```
host@host~$ cd Programming_Tool
```

- Copy the programming script file into Programming_Tool directory and change the permission.

```
host@host/Programming_Tool~$ cp /<path to Programming tool>/make_sdimage.py .
```

```
host@host/Programming_Tool~$ chmod +x make_sdimage.py
```

- The prebuilt binaries are available in the deliverables in the below path.

```
iW-PRFXR-RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables/Binaries/Binaries.tar.gz
```

- Un-tar the Binaries.tar.gz file from deliverables to program the default binaries.

```
$ tar -xvf <iW-PRFXR-RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables>/iW-PRFXR-RX.X-RELX.X-Arria10-SX480-Linux4.9.78-YoctoPyro_Deliverables/Binaries/Binaries.tar.gz
```

- Copy all the Binaries to the current folder.

```
host@host/Programming_Tool~$ cp <path_to_Binaries>/Binaries/FPGA/ghrd_10as066n2.rbf .
```

```
host@host/Programming_Tool~$ cp <path_to_Binaries>/Binaries/Uboot/u-boot_w_dtb-mkpimage.bin .
```

```
host@host/Programming_Tool~$ cp <path_to_Binaries>/Binaries/Linux/* .
```

- Create a folder for file system.

```
host@host/Programming_Tool~$ mkdir rootfs
```

```
host@host/Programming_Tool~$ cd rootfs
```

- Execute the below command to Untar the file system.

```
host@host/Programming_Tool/rootfs~$ sudo tar -xvzf ../rootfs.tar.gz
```

```
host@host/Programming_Tool/rootfs~$ sync
```

```
host@host/Programming_Tool/rootfs~$ sudo cp -rf <path_to_Binaries>/Binaries/Linux/iwtest .
```

```
host@host/Programming_Tool/rootfs~$ sync
```

```
host@host/Programming_Tool/rootfs~$ cd ../
```

- Execute the below command to create the bootable sd card image.

```
host@host/Programming_Tool/~$ sudo ./make_sdimage.py -f -P u-boot_w_dtb-mkpimage.bin,num=3,format=raw,size=10M,type=A2 -P rootfs/*,num=2,format=ext3,size=1500M -P zImage,ghrd_10as066n2.rbf,socfpga_arria10_iwg24m_prfxr_sx480.dtb,num=1,format=vfat,size=500M -s 2G -n sd_card_image_a10.img
```

- Execute the below command to program the bootable sd card image to Micro SD.

```
host@host/Programming_Tool/~$ sudo dd if=sd_card_image_a10.img of=/dev/sd<x> bs=1M
```


3.3 Core Detection using JTAG

This section explains how to detect HPS and FPGA core using On Board USB Blaster in iW-RainboW-G24M-Arria10 FMC+ development platform.

Test procedure

- Connect the one end of USB OTG cable to Windows PC (Quartus tool must be installed in Windows 7 PC) and other end to the On-Board USB Blaster connector on the board(J4).
- Open Quartus Tool.
- Run Tools -> Programmer in Quartus tool as shown below.

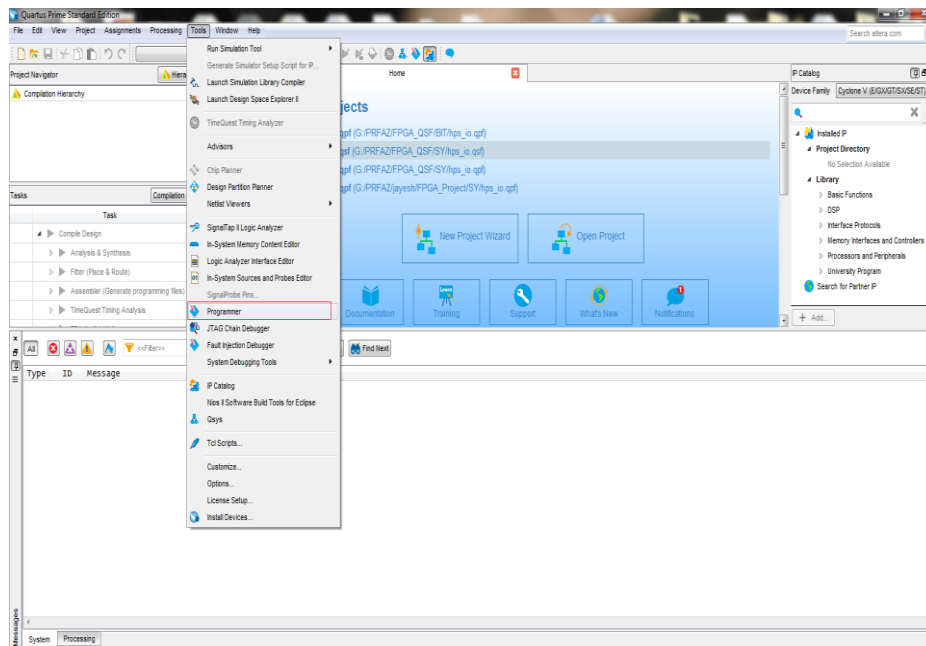


Figure 14: JTAG-1

- Programmer window will be displayed as shown below and Click on Hardware Setup.

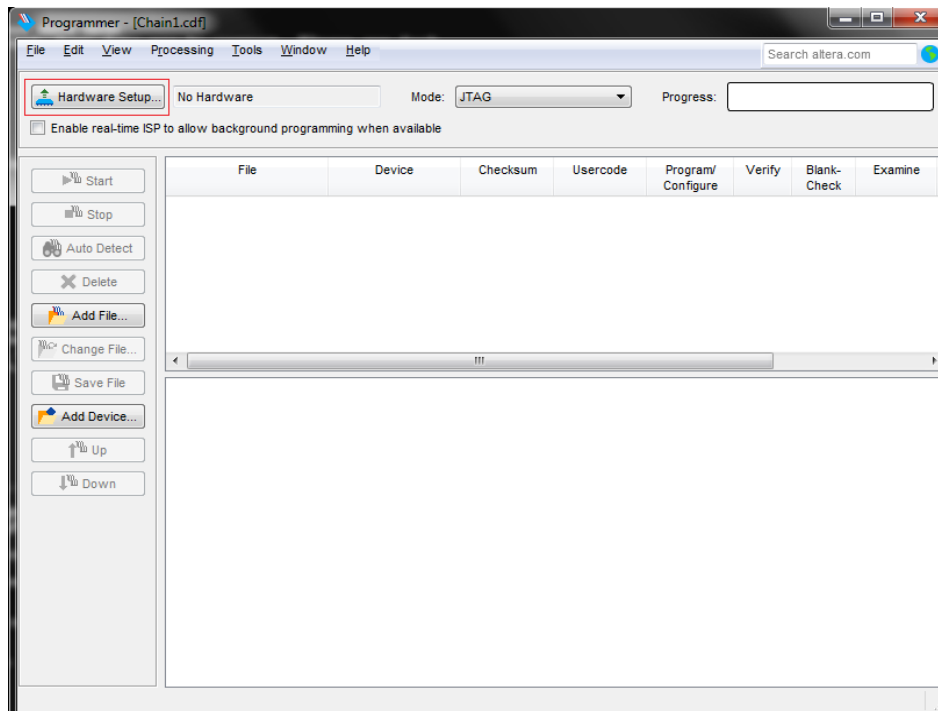


Figure 15: JTAG-2

- Hardware setup window will be displayed as shown below.

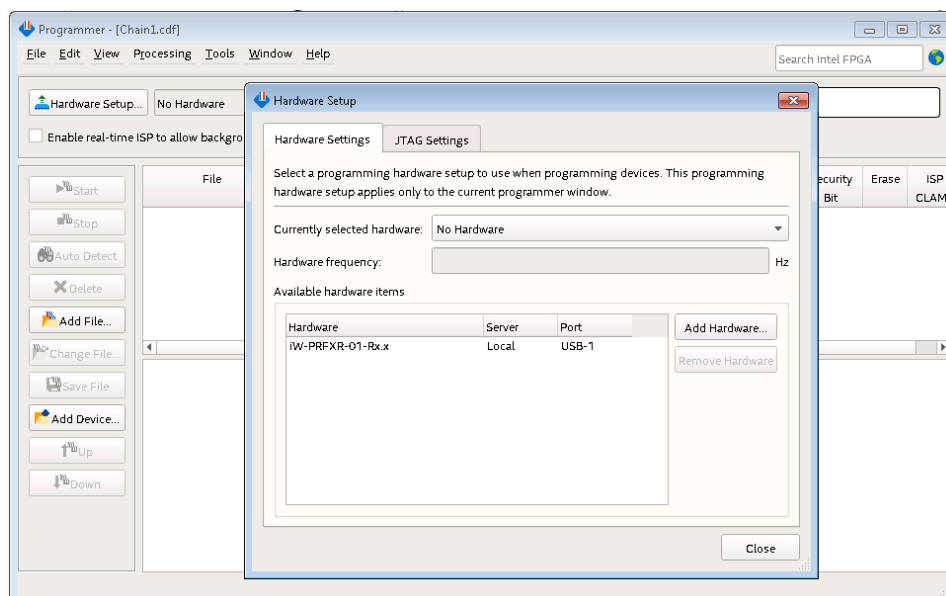


Figure 16: JTAG-3

- Select iW-PRFXR-01-Rx.x[USB-1] in Currently selected hardware and Click on Close:

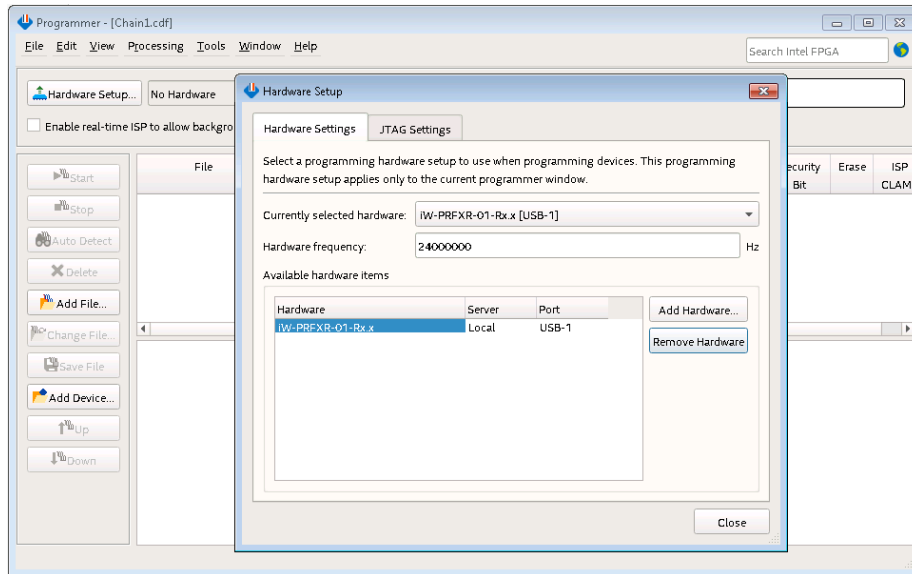


Figure 17: JTAG-4

- Click on Auto Detect in Programmer window.

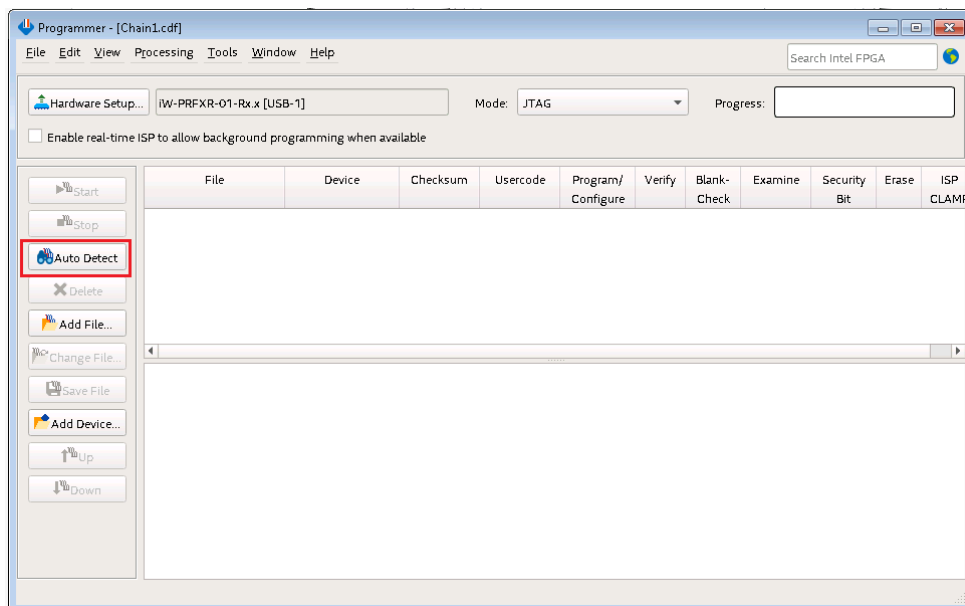


Figure 18: JTAG-5

- List of devices will be displayed. Select 10AS048H3 device and click OK.

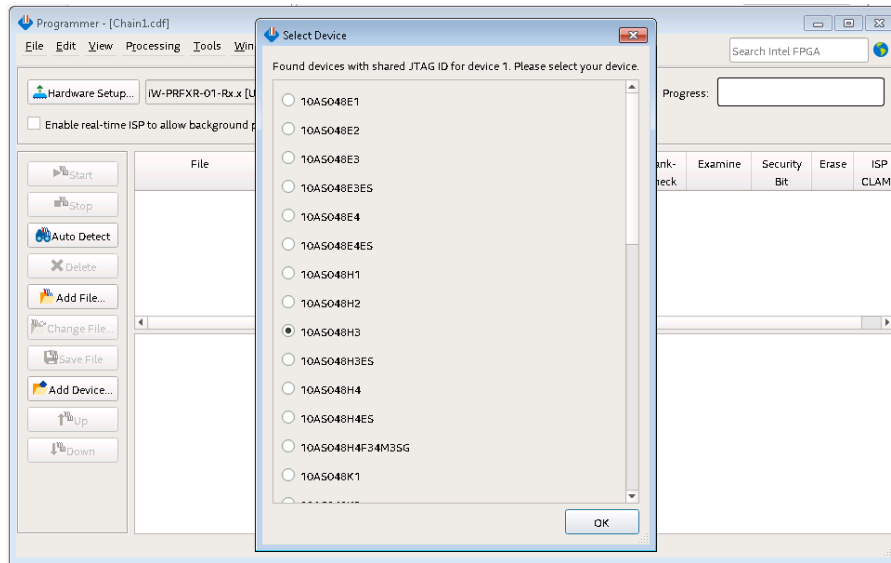


Figure 19: JTAG-6

- HPS and FPGA cores will be detected as shown below.

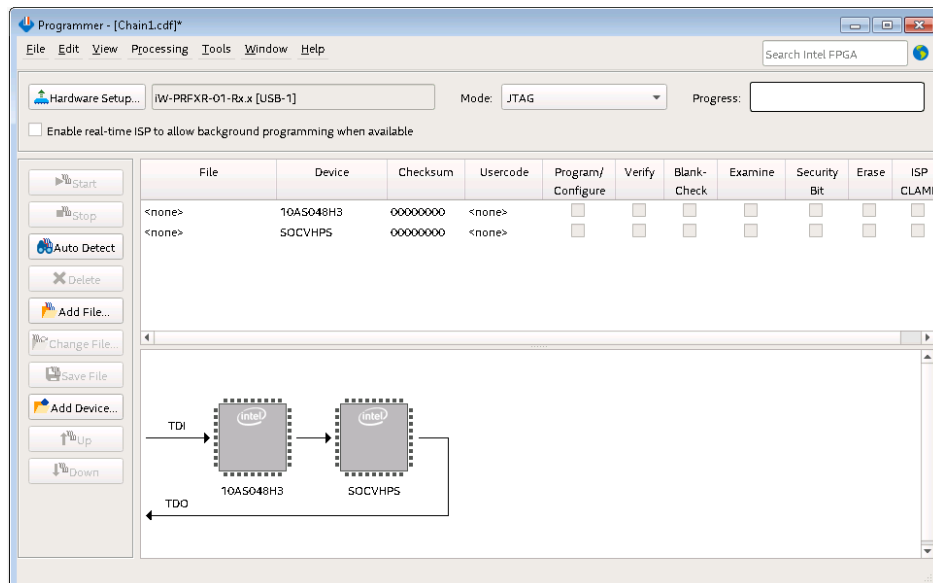


Figure 20: JTAG-7

4. Setting Up the Test Environment

This section describes the step by step procedure to setup the test environment for iW-RainboW-G24M-Arria10 FMC+ development platform.

- Setting up the Debug port
- Power ON the Development platform

4.1 Debug Port Setting

- Connect Type A end of USB cable to PC and Micro AB end of USB cable to Development platform's debug Micro USB connector(J3) as shown below.

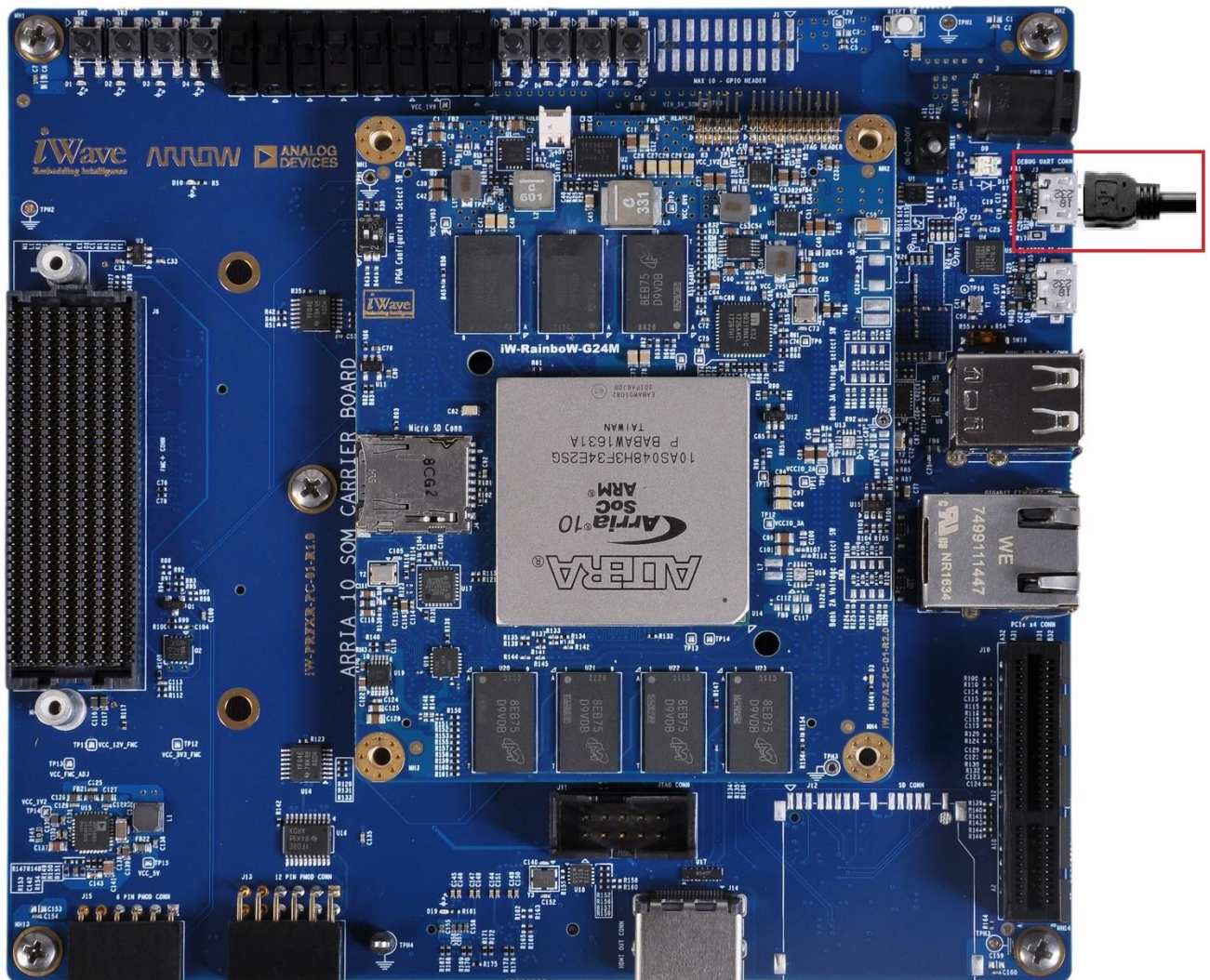


Figure 21 : Debug Port Connection

- Install the driver for Debug Port in Host PC/Laptop using the below link.

<http://www.ftdichip.com/Products/ICs/FT232R.html>

- Setup the Debug Terminal with the below parameter settings.

Baud Rate : 115200

Data bits : 8

Parity : None

Stop Bits : 1

Flow Control : None

4.2 Powering ON Arria10 SoC/FPGA FMC+ Development Platform

iW-Rainbow-G24M-Arria10 SoC/FPGA FMC+ Development Platform comes with 12V, 5A power supply with universal plugs. Please follow the below procedure to power on the Development platform.

- Connect the 12V power supply plug to the power connector (J2) of the iW-RainboW-G24M-Arria10 Soc/FPGA FMC+ development platform as shown below and switch ON the power supply.

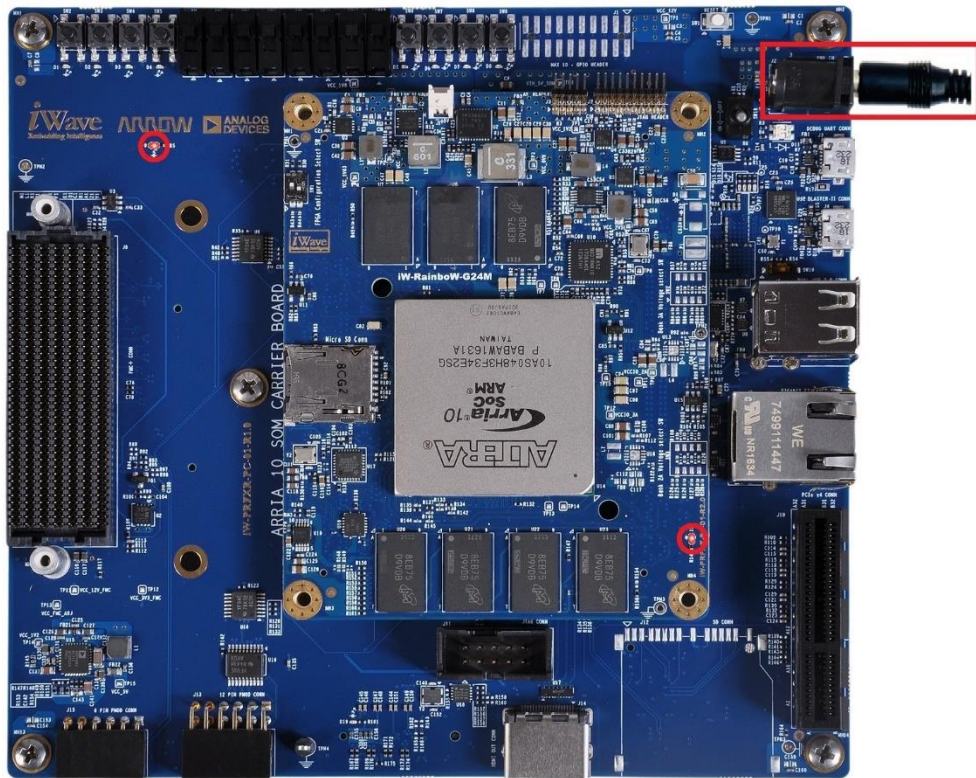


Figure 22 : Power Supply Connection

- Once Power is applied to iW-RainboW-G24M Arria10 SoC/FPGA FMC+ Development Platform, the Red Power LEDs in the Arria10 SoC/FPGA SOM and carrier board glow as shown in the above image.

4.3 Test Environment

Once power is applied to iW-RainboW-G24M-Arria10 FMC+ development platform as explained in the previous section, the boot message displays on the debug terminal of the PC/Laptop which is connected to the Development platform.

In Linux Release, U-boot boot messages appears in Hyper Terminal as shown below.

```
U-Boot 2014.10 (May 07 2019 - 14:43:56)

CPU : Altera SOCFPGA Arria 10 Platform
BOARD : iW-RainboW-G24M Arria 10 SOCFPGA FMC+ Development Platform
I2C: ready
DRAM: WARNING: Caches not enabled
SOCFPGA DWMMC: 0
FPGA: writing ghrd_10as066n2.rbf ...
Full Configuration Succeeded.
DDRCAL: Success
SDRAM: Initializing ECC 0x00000000 - 0x80000000
SDRAM-ECC: Initialized success with 2148 ms
INFO : Skip relocation as SDRAM is non secure memory
Reserving 2048 Bytes for IRQ stack at: ffe386e8
DRAM : 2 GiB
WARNING: Caches not enabled
MMC: In: serial
Out: serial
Err: serial
Model: iWG24M Arria 10 SOCFPGA FMC+ Development Platform

Board Info:
      BSP Version      : iW-PRFXR-SC-01-R1.0-REL0.1-Linux4.9.78
      SOM Version      : iW-PRFAZ-AP-01-R2.2

Net: dwmac.ff802000
Hit any key to stop autoboot: 2
```

Figure 23 : U-Boot On Terminal

Immediately after power on, press any key in debug terminal to go to the U-boot command prompt as shown below. Otherwise Linux will launch automatically.

```
U-Boot 2014.10 (May 07 2019 - 14:43:56)

CPU : Altera SOCFPGA Arria 10 Platform
BOARD : iW-RainboW-G24M Arria 10 SOCFPGA FMC+ Development Platform
I2C: ready
DRAM: WARNING: Caches not enabled
SOCFPGA DWMMC: 0
FPGA: writing ghrd_10as066n2.rbf ...
Full Configuration Succeeded.
DDRCAL: Success
SDRAM: Initializing ECC 0x00000000 - 0x80000000
SDRAM-ECC: Initialized success with 2148 ms
INFO : Skip relocation as SDRAM is non secure memory
Reserving 2048 Bytes for IRQ stack at: ffe386e8
DRAM : 2 GiB
WARNING: Caches not enabled
MMC: In: serial
Out: serial
Err: serial
Model: iWG24M Arria 10 SOCFPGA FMC+ Development Platform

Board Info:
      BSP Version      : iW-PRFXR-SC-01-R1.0-REL0.1-Linux4.9.78
      SOM Version      : iW-PRFAZ-AP-01-R2.2

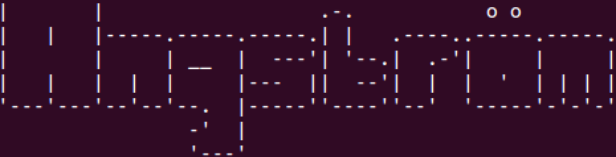
Net: dwmac.ff802000
Hit any key to stop autoboot: 0
iWave-G24M>
```

Figure 24 : U-Boot Command Prompt

To Login in Linux, enter "root" in terminal and you will get the Linux command prompt as shown below. Once you get the prompt you are done with Test Environment setup on Linux delivery.

```
[ OK ] Starting Lightning Fast Webserver With Light System Requirements...  
[ OK ] Started NFS status monitor for NFSv2/3 locking..  
Starting Network Name Resolution..  
Starting Permit User Sessions..  
[ OK ] Started Permit User Sessions.  
[ OK ] Started Lightning Fast Webserver With Light System Requirements.  
[ OK ] Started Network Name Resolution.  
[ OK ] Started Getty on tty1.  
[ OK ] Started Serial Getty on ttyS0.  
[ OK ] Reached target Login Prompts.
```

--O--



The Angstrom Distribution iwq24m ttyS0

Angstrom v2017.06 - Kernel

iwq24m login: root

Last login: Wed Apr 3 04:56:41 UTC 2019 on ttyS0

root@iwq24m:~#

Figure 25 : Linux Command Prompt

5. U-BOOT TESTING AND BOOT CONFIGURATION

This part of the document explains about testing the peripherals in u-boot level and loading the Linux OS for iW-RainboW-G24M-Arria10 FMC+ development platform.

- iW-RainboW-G24M-Arria10 FMC+ development platform boot from Micro SD as boot media device.
- Connect debug UART with host PC and Power ON the iW-RainboW-G24M-Arria10 FMC+ development platform.

5.1 Basic commands

- To find available commands and descriptions in U-Boot level type the below command.

```
iWave-G24M> help
```

- The Available commands are displayed in command prompt as shown below.

```
bdinfo - print Board Info structure  
boot - boot default, i.e., run 'bootcmd'  
bootm - boot application image from memory  
saveenv - save environment variables to persistent storage  
setenv - set environment variables
```

- To display the platform information, execute the below command.

```
iWave-G24M> bdinfo
```

- The platform information will be displayed in command prompt as shown below.

```
arch_number = 0x00000000  
boot_params = 0x00000100  
DRAM bank = 0x00000000  
-> start = 0x00000000  
-> size = 0x40000000  
eth0name = dwmac.ff802000  
ethaddr = 00:01:02:03:04:05  
current eth = dwmac.ff802000  
ip_addr = <NULL>  
baudrate = 115200 bps  
TLB addr = 0xFFE3C000  
relocaddr = 0x40000000  
reloc off = 0x00000000  
irq_sp = 0xFFE34EE8  
sp start = 0x3FEFF000
```

5.2 Basic device Tests

In U-Boot level, the supported devices are

- RAM
- Micro SD
- Ethernet

5.2.1 RAM Test

- In iW-RainboW-G24M-Arria10 FMC+ development platform RAM physical address is from 0x00000000 to 0x7FFFFFFF.
- To write the data into RAM location, execute the below command.

```
iWave-G24M> mw <RAM_addr> <DATA> <No_of_location_to_be_write>
```

- To display the data in the RAM location, execute the below command.

```
iWave-G24M> md <RAM_addr> <No_of_location_to_be_display>
```

- To test the RAM read/write, execute the below command.

```
iWave-G24M> mtest <RAM_addr_start> <RAM_addr_end> <DATA> <No_of_times>
```

Example

```
iWave-G24M> mtest 0x10000000 0x10500000 0xAABBCCDD 0x1
```

Testing 10000000 ... 10500000:

Tested 1 iteration(s) with 0 errors.

Note: Accessing the restricted RAM area or other physical address may cause unpredictable behaviour. Make sure, you are not entering the restricted area RAM address.

5.2.2 SD Test

- Initialize the particular SD/eMMC device by the below command.

```
iWave-G24M> mmc dev 0
```

switch to partitions #0, OK

mmc0 is current device

- To display the SD/eMMC device information, execute the below.

```
iWave-G24M> mmcinfo
```

Device: SOCFPGA DWMMC

Manufacturer ID: 3

OEM: 5344

Name: SS08G

Tran Speed: 50000000

Rd Block Len: 512

SD version 3.0

High Capacity: Yes

Capacity: 7.4 GiB

Bus Width: 4-bit

5.2.3 Ethernet Test

- To set the MAC address and IP address for the platform and to save the environment variables, execute the below command.

```
iWave-G24M> setenv ethaddr "<MAC addr>"
iWave-G24M> dhcp

Speed: 1000, full duplex
BOOTP broadcast 1
*** Unhandled DHCP Option in OFFER/ACK: 2
*** Unhandled DHCP Option in OFFER/ACK: 2
DHCP client bound to address 192.168.*.* (10 ms)
*** Warning: no boot file name; using 'COA80321.img'
Using dwmac.ff802000 device
TFTP from server 0.0.0.0; our IP address is 192.168.*.*; sending through gateway 192.168.2.254
Filename 'COA80321.img'.
Load address: 0x8000
Loading: *
TFTP error: 'File not found' (1)
Not retrying...'
iWave-G24M> saveenv
```

- To ping any IP address from the platform, execute the below command.

```
iWave-G24M> ping <any_ip_addr>
```

Example

```
iWave-G24M> ping *****

Speed: 1000, full duplex
Using dwmac.ff802000 device
host <any_ip_addr> is alive
```

5.2.3.1 TFTP & NFS Host PC setup

This section describes to setup a TFTP server and NFS server in Ubuntu Linux distributions Host PC.

- The following host pc setup is required only once per host.
- Install the nfs-kernel-server, tftpd-hpa server

```
$ sudo apt-get install nfs-kernel-server tftpd-hpa -y
```

TFTP server

- Edit tftp configuration file to upload the file to tftp server.

```
$ sudo vi /etc/default/tftpd-hpa  
TFTP_USERNAME="tftp"  
TFTP_DIRECTORY="/var/lib/tftpboot "  
TFTP_ADDRESS=":69"  
TFTP_OPTIONS="--secure "
```

- Change line TFTP_OPTIONS="--secure" to TFTP_OPTIONS="--secure --create" as mentioned below

```
TFTP_USERNAME="tftp"  
TFTP_DIRECTORY="/var/lib/tftpboot"  
TFTP_ADDRESS=":69"  
TFTP_OPTIONS="--secure --create "
```

Note : The default root directory where files are stored is /var/lib/tftpboot.

- Change the ownership of the directory.

```
$ sudo chown -R tftp /var/lib/tftpboot
```

- Start the tftp services,

```
$ sudo service tftpd-hpa stop  
$ sudo service tftpd-hpa start
```

- Verify the TFTP is running correctly or not

```
$ sudo service tftpd-hpa status
```

NFS

- Open file /etc/exports by below comment

```
$ sudo vim /etc/exports
```

- Insert the following line in /etc/exports file

```
<path to rootfs> *(rw,sync,no_root_squash)
```

Example

```
/home/iwave/test/rootfs *(rw,sync,no_root_squash)
```

- If you change this configuration file , you have to restart the NFS server:

```
$ sudo /etc/init.d/nfs-kernel-server stop  
$ sudo /etc/init.d/nfs-kernel-server restart
```

5.3 Environment variables settings

By default, the environment variables are saved in the boot device.

5.3.1 Micro SD boot

- To load the kernel and file systems from the Micro SD, the environment variables should be set as shown below.

```
iWave-G24M> setenv bootcmd 'run core_rbf_prog; run callscript; run mmcload;run set_initstate; run mmcboot'
```

```
iWave-G24M> setenv mmcboot 'setenv bootargs console=ttyS0,115200 root=${mmccroot} rw rootwait vmalloc=100M consoleblank=0;fpgabr 1;bootz ${loadaddr} - ${fdtaddr}'
```

```
iWave-G24M> setenv mmcload 'mmc rescan;${mmccloadcmd} mmc 0:${mmccloadpart} ${loadaddr} ${bootimage};${mmccloadcmd} mmc 0:${mmccloadpart} ${fdtaddr} ${fdtimage}'
```

```
iWave-G24M> saveenv
```

- To boot the platform, execute the below command.

```
iWave-G24M> boot
```

5.3.2 TFTP boot

- To load the kernel and file systems through the TFTP, the environment variables should be set as shown below.

```
iWave-G24M> setenv fdt_addr '0x808000'
```

```
iWave-G24M> setenv serverip '<server ip address>'
```

```
iWave-G24M> setenv fdt_file 'socfpga_arria10_iwg24m_prfxr_sx480.dtb'
```

```
iWave-G24M> setenv nfsroot '<path to rootfs>'
```

```
iWave-G24M> setenv bootargs_base 'console=ttyS0,115200'
```

```
iWave-G24M> setenv bootargs_net 'setenv bootargs ${bootargs_base} root=/dev/nfs ip=dhcp nfsroot=${serverip}:${nfsroot},v3,tcp vmalloc=100M consoleblank=0'
```

```
iWave-G24M> setenv bootcmd_net 'dhcp;tftpboot ${loadaddr} ${serverip}:${bootimage};tftpboot ${fdt_addr} ${serverip}:${fdtimage}; run bootargs_net;fpgabr 1;bootz ${loadaddr} - ${fdt_addr}'
```

```
iWave-G24M> setenv bootcmd 'run set_initstate;run bootcmd_net'
```

```
iWave-G24M> saveenv
```

- To boot the platform, execute the below command.

```
iWave-G24M> boot
```

Note: For TFTP boot below mentioned changes needs to be done file system to disable the Angstrom Connection Manager to prevent loss of connection to the NFS server during Angstrom boot.

```
sudo mv lib/systemd/system/connman.service lib/systemd/system/connman.service.nouse
```

5.3.3 Default Environment Variable

- To restore the default environment variables, execute the below commands

```
iWave-G24M> env default -f -a
```

```
iWave-G24M> saveenv
```

```
iWave-G24M> reset
```

6. LINUX PERIPHERAL TESTING

This part of the document explains about testing the peripherals in Linux OS level for iW-RainboW-G24M-Arria10 FMC+ development platform.

- Connect debug UART with host PC and Power ON the iW-RainboW-G24M-Arria10 FMC+ development platform.

6.1 Block devices Test

The iW-RainboW-G24M-Arria10 FMC+ development platform supports the below block devices.

- Micro SD
- USB host

Testing device Requirements

To test the block devices supported by iW-RainboW-G24M-Arria10 FMC+ development platform, following items are required.

- USB memory stick
- Micro SD

6.1.1 Block Device Test

- The Micro SD / USB-OTG(as Host) will mount in below mentioned directories.

Micro SD - */run/media/mmcblk0p1, /run/media/mmcblk0p2, ... etc*

USB-OTG(as Host) - */run/media/sda1, /run/media/sda2, ... etc*

- To view the contents, execute the below command.

```
root@iwg24m:~# cd <mount_directory>
```

```
root@iwg24m:~/<mount_directory># ls
```

- To create a directory and remove a directory from the mounted partition, execute below commands respectively.

```
root@iwg24m:~/<mount_directory># mkdir <directory_name>
```

```
root@iwg24m:~/<mount_directory># rm -rf <target_directory>
```

- To copy a file to the mounted partition, execute below command.

```
root@iwg24m:~/<mount_directory># cp <source_file> <Destination>
```

- To exit from the mount folder, execute below command.

```
root@iwg24m:~/<mount_directory># cd /home/root
```

- To unmount the device, execute the below command.

```
root@iWave-G22M~# umount <Mount Directory>
```

Note: The rootfs mounted partition of any block device will not be mounted

6.2 Network devices Test

The iW-RainboW-G24M-Arria10 FMC+ development platform supports the below mentioned network device.

- Ethernet

Testing device Requirements

To test the Network device supported by iW-RainboW-G24M-Arria10 FMC+ development platform, following items are required.

- Ethernet connection

6.2.1 Ethernet Test

This section explains how to test the Ethernet in the iW-RainboW-G24M-Arria10 FMC+ development platform.

- Connect the Ethernet cable and to enable the Ethernet device, execute the below command.

```
root@iwg24m:~# ifconfig eth0 up
```

- To set the IP address using DHCP, execute the below command.

```
root@iwg24m:~# udhcpc -i eth0
```

```
udhcpc (v1.22.1) started
```

```
run-parts: /etc/udhcpc.d/00avahi-autoipd exited with code 1
```

```
Sending discover...
```

```
Sending select for *****...
```

```
Lease of ***** obtained, lease time 600
```

```
run-parts: /etc/udhcpc.d/00avahi-autoipd exited with code 1
```

```
/etc/udhcpc.d/50default: Adding DNS 192.168.2.254
```

```
/etc/udhcpc.d/50default: Adding DNS 192.168.2.2
```

```
/etc/udhcpc.d/50default: Adding DNS 192.168.2.4
```

- To check the IP address set, execute the below command.

```
root@iwg24m:~# ifconfig
```

```
eth0    Link encap:Ethernet HWaddr 00:01:02:03:04:05
```

```
inet addr:***** Bcast:***** Mask:255.255.255.0
```

```
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
```

```
RX packets:28783 errors:0 dropped:87 overruns:0 frame:0
```

```
TX packets:15286 errors:0 dropped:0 overruns:0 carrier:0
```

```
collisions:0 txqueuelen:1000
```

```
RX bytes:6218139 (5.9 MiB) TX bytes:4454482 (4.2 MiB)
```

- To test Ethernet, execute the below command.

```
root@iwg24m:~# ping <any_ip_addr>
```

```
PING ***** (*****): 56(84) bytes of data.
```

```
64 bytes from *****: icmp_seq=1 ttl=64 time=0.206 ms
```

```
64 bytes from *****: icmp_seq=2 ttl=64 time=0.160 ms
```

6.2.1.1 File transfer using TFTP server

- To receive any file from TFTP server to iW-RainboW-G24M-Arria10 FMC+ development platform, execute the below command

```
root@iwg24m:~# tftp -g <server_ip> -r <file_name>
```

- To transmit any file from iW-RainboW-G24M-Arria10 FMC+ development platform to TFTP server (host PC), execute the below command

```
root@iwg24m:~# tftp -p <server_ip> -l <file_name>
```

6.2.1.2 Folder mount from NFS

- To mount any folder from NFS server (Host PC) to iW-RainboW-G24M-Arria10 FMC+ development platform, execute the below command

```
root@iwg24m:~# mount -o nolock -t nfs <server_ip>:./<filepath> <mnt_directory>
```

- To view the NFS mounted files and folders, execute below command.

```
root@iwg24m:~# ls <mnt_directory>
```

*Note: To configure the host PC (under Linux OS) for TFTP and NFS server, refer the **TFTP & NFS Host PC setup** section.*

6.3 HID Device Test

The iW-RainboW-G24M-Arria10 FMC+ development platform supports the below Human Interface devices.

- USB HID devices

Testing device Requirements

To test the Human Interface Devices supported by iW-RainboW-G24M-Arria10 FMC+ development platform, following Items are required.

- USB keyboard.

6.3.1 USB Keyboard

- Insert the USB Keyboard in iW-RainboW-G24M-Arria10 FMC+ development platform USB slot. The following message will be displayed in command prompt.

```
root@iwg24m:~# [ 150.287308] usb 1-1.2: new low-speed USB device number 4 using dwc2
[ 151.378872] input: Dell KB216 Wired Keyboard as /devices/platform/soc/ffb40000.usb/usb1/1-1/1-1.2/1-1.2:1.0/0003:413C:2113.0002/input/input1
[ 151.458014] hid-generic 0003:413C:2113.0002: input: USB HID v1.11 Keyboard [Dell KB216 Wired Keyboard] on usb-ffb40000.usb-1.2/input0
[ 151.889455] input: Dell KB216 Wired Keyboard as /devices/platform/soc/ffb40000.usb/usb1/1-1/1-1.2/1-1.2:1.1/0003:413C:2113.0003/input/input2
[ 151.967502] hid-generic 0003:413C:2113.0003: input: USB HID v1.11 Device [Dell KB216 Wired Keyboard] on usb-ffb40000.usb-1.2/input1
root@iwg24m:~#
```

- Type on the keyboard to enter the command in HDMI console.

6.4 PCIe Test

The iW-RainboW-G24M-Arria10 FMC+ development platform supports the PCIe4 End point devices. This section explains how to list the connected PCIe End point devices in the iW-RainboW-G24M-Arria10 FMC+ development platform.

- Connect the PCIe device in iW-RainboW-G24M-Arria10 FMC+ development platform before powering ON.
- To list out the PCIe device, connected with board, execute the below command.

```
root@iwg24m:~# lspci
```

```
0000:00:00.0 PCI bridge: Altera Corporation Device e000 (rev 01)
```

```
0001:00:00.0 PCI bridge: Altera Corporation Device e000 (rev 01)
```

```
0001:01:00.0 USB controller: ASMedia Technology Inc. Device 2142
```

Note: To test the PCIe device, the corresponding device driver has to be included in the kernel.

6.5 NVMe Test

The iW-RainboW-G24M-Arria10 FMC+ development platform supports the NVMe4 End point devices. This section explains how to list the connected NVMe End point devices in the iW-RainboW-G24M-Arria10 FMC+ development platform.

- Connect the NVMe device in iW-RainboW-G24M-Arria10 FMC+ development platform before powering ON.
- To list out the NVMe device, connected with board, execute the below command.

```
root@iwg24m:~# lspci
```

```
0000:00:00.0 PCI bridge: Altera Corporation Device e000 (rev 01)
```

```
0000:01:00.0 Non-Volatile memory controller: Samsung Electronics Co Ltd Device a804
```

```
0001:00:00.0 PCI bridge: Altera Corporation Device e000 (rev 01)
```

Note: Use command 'df -h' to check if the device is mounted. If the device is not mounted, mount device manually using below command.

- To mount NVMe device, execute below command

```
root@iwg24m:~# mkdir /run/media/nvme0n1p1
```

```
root@iwg24m:~# mount -t vfat /dev/nvme0n1p1 /run/media/nvme0n1p1
```

6.6 FPGA/HPS I/O Test

The iW-RainboW-G24M-Arria10 FMC+ development platform supports 4 LED, 4 PUSH buttons and 4 Dipswitches from both FPGA and HPS.

6.6.1 Dipswitch Test

- To know the status of the HPS dipswitches, execute the below command.

```
root@iwg24m:~# /iwtest/hps_dip <dipswitch_number>  
dipswitch_number = 1,2,3,4
```

Example

```
root@iwg24m:~# /iwtest/hps_dip 1  
Dip Switch 1 Status - High/On
```

- To know the status of the FPGA dipswitches, execute the below command.

```
root@iwg24m:~# /iwtest/pio_dip <dipswitch_number>  
dipswitch_number = 1,2,3,4
```

Example

```
root@iwg24m:~# /iwtest/pio_dip 1  
Dip Switch 1 Status - High/On
```

6.6.2 LED/PUSH Button Test

The HPS and FPGA Push Buttons are mapped to LED's. Press each of the PUSH button and the corresponding LED glows.

6.7 FMC Loopback test

The iW-RainboW-G24M-Arria10 FMC+ development platform BSP supports 2 types of FMC loopback test.

- FMC transceiver Loopback
- FMC GPIO Loopback

6.7.1 FMC transceiver Loopback

- To test the FMC transceiver loopback, execute below command.

```
root@iwg24m:~# insmod /lib/modules/4.9.78-ltsi/kernel/drivers/fmc/fmc_transceiver.ko
```

- To remove the inserted module, execute below command.

```
root@iwg24m:~# rmmod /lib/modules/4.9.78-ltsi/kernel/drivers/fmc/fmc_transceiver.ko
```

6.7.2 FMC GPIO Loopback

- To test the FMC gpio loopback, execute below command.

```
root@iwg24m:~# /iwtest/fmc_gpio_loopback
```

6.8 PMOD Loopback test

The iW-RainboW-G24M-Arria10 FMC+ development platform supports 6 Pin PMOD and 12 Pin PMOD.

- To test the 6 Pin PMOD loopback test, execute below command.

```
root@iwg24m:~# /iwtest/pmod_loopback1
```

- To test the 12 Pin PMOD loopback test, execute below command.

```
root@iwg24m:~# /iwtest/pmod_loopback2
```

6.9 HDMI Test

The iW-RainboW-G24M-Arria10 FMC+ development platform supports full HD HDMI display of resolution 1080p@60Hz.

- Connect the HDMI to iW-RainboW-G24M-Arria10 FMC+ development platform before powering on the system and check below screen appears on the HDMI monitor after completion of the boot.



Figure 26 : HDMI Screen

6.10 Temperature Sensor test

- Execute below command to print the temperature.

```
root@iwg24m:~# cat /sys/class/temperature/show_temperature
```

