Vitis AI Lab 3 to 6

XILINX.

Agenda

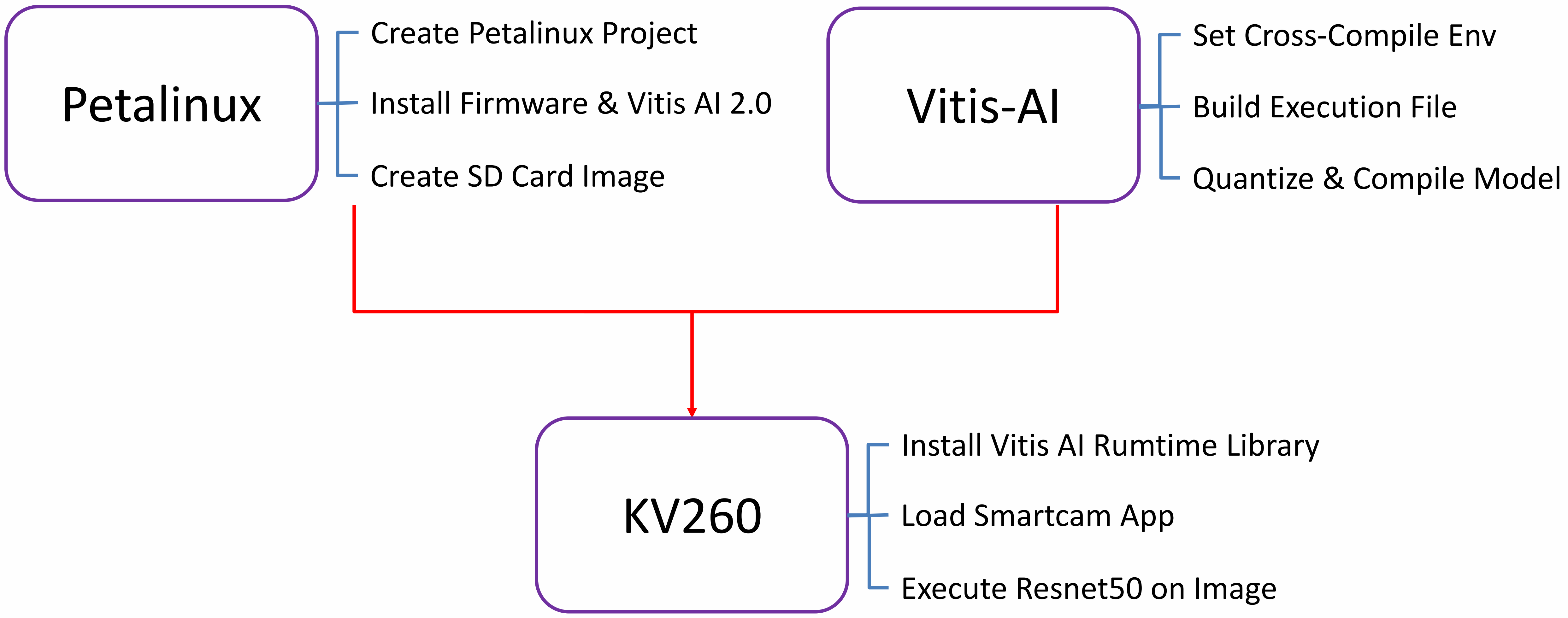
- Lab 3: Vitis AI Library (VART) – Using KV260
- Lab 4: Creating a Hardware Platform with the DPU Using the Vivado Design Suite Flow – Using KV260
- Lab 5: Creating a DPU Kernel Using the Vitis Environment Flow – Using KV260
- Lab 6: Creating a Custom Application – Using KV260

Lab 3: Vitis AI Library (VART) – Using KV260

Vitis AI Library (VART) – Using KV260

- Environment Setting
  1. Ubuntu 18.04
  2. PetaLinux Tools - Installer - 2021.2
  3. Kria K26 SOM Board Support Package - 2021.2
  4. Vitis-AI Lab 2.0

Vitis AI Library (VART) – Using KV260



Petalinux

- Build Petalinux

1. Dowload Petalinux Tools – Installer (2021.2)
2. Install Dependencies
3. ./petalinux-v2021.2-final-installer.run -d <custom path>
4. source <custom path>/settings.sh

Install Petalinux Tools

1. petalinux-create -t project -s /<kv260 BSP path>/xilinx-k26-starterkit-v2021.2-final.bsp -n kv260_os
2. cd ./kv260_os
3. petalinux-build

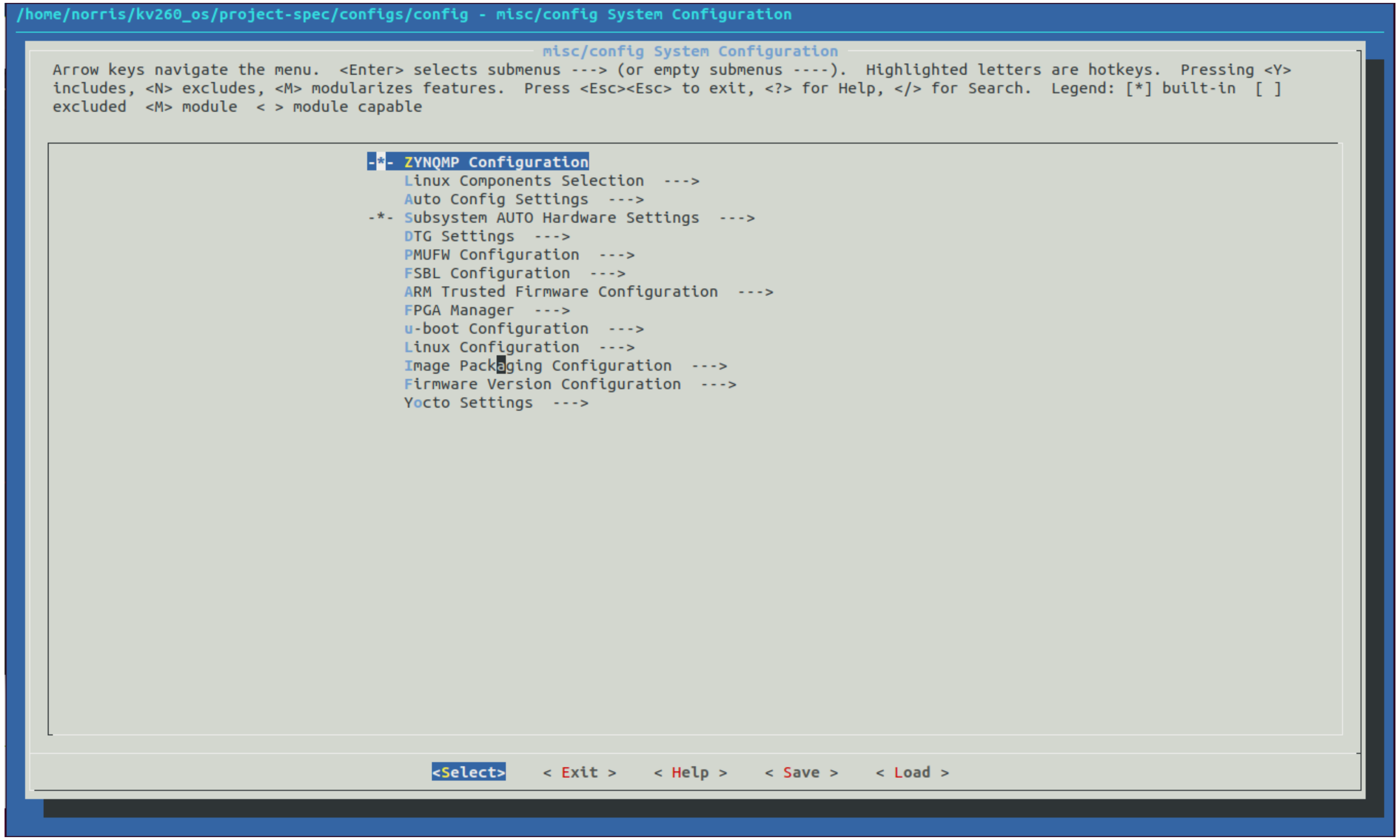
Build Petalinux Project

Petalinux

- Add Vitis AI 2.0 & Firmware to Petalinux
  1. cd components/yocto/layers/
  2. sudo rm -r meta-vitis-ai
  3. git clone -b rel-v2021.2 https://github.com/jlamperez/meta-vitis-ai.git meta-vitis-ai
  4. vi ~/kv260_os/build/conf/bblayers.conf
  5. delete ${SDKBASEMETAPATH}/layers/meta-vitis-ai
  6. add new meta-vitis-ai layer by petalinux-config

Petalinux - Vitis AI 2.0

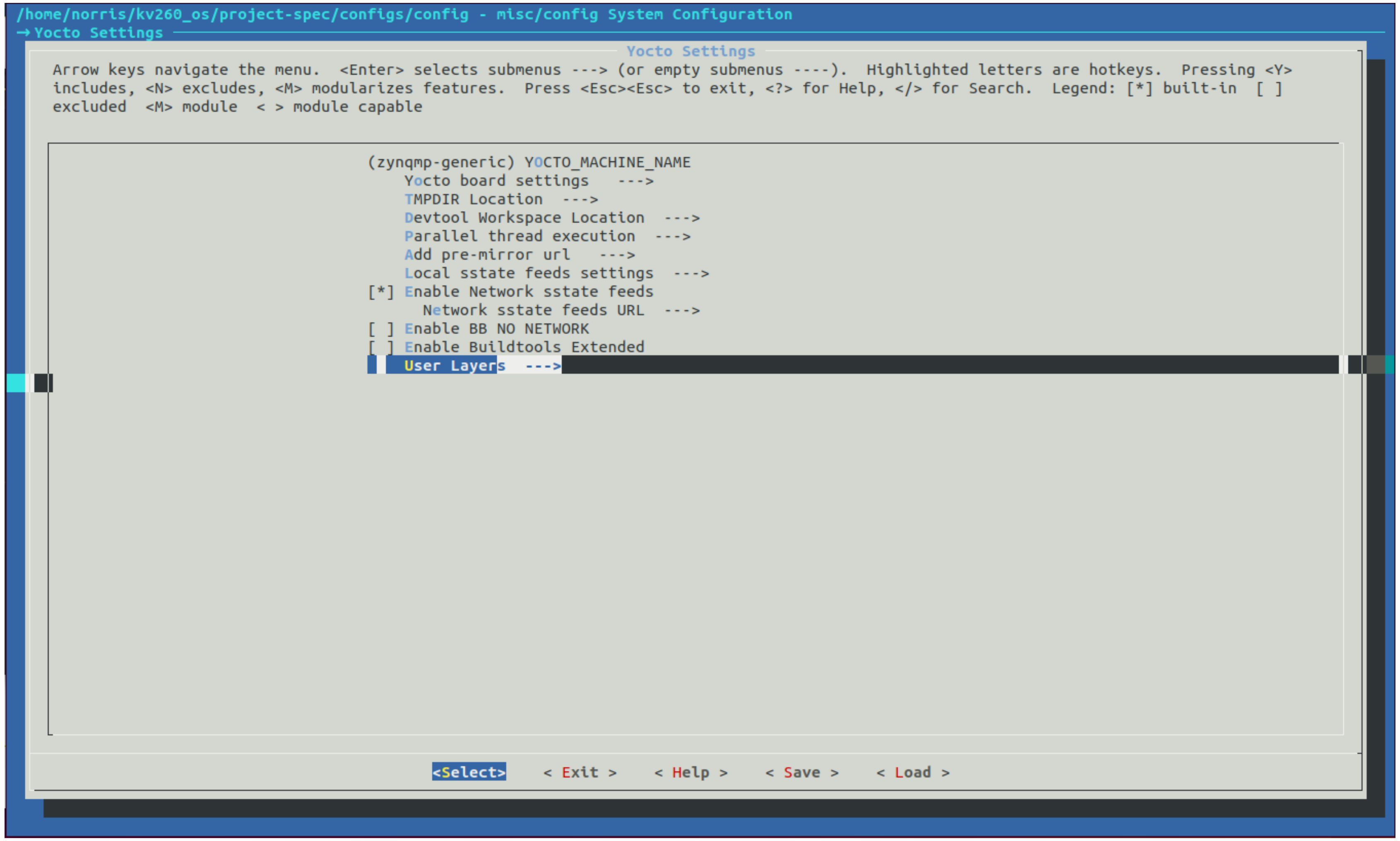
- petalinux-config



```
/home/norris/kv260_os/project-spec/configs/config - misc/config System Configuration
                              misc/config System Configuration
 Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing <Y>
 includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ]
 excluded  <M> module  < > module capable

                                   -*- ZYNQMP Configuration
                                       Linux Components Selection  --->
                                       Auto Config Settings  --->
                                   -*- Subsystem AUTO Hardware Settings  --->
                                       DTG Settings  --->
                                       PMUFW Configuration  --->
                                       FSBL Configuration  --->
                                       ARM Trusted Firmware Configuration  --->
                                       FPGA Manager  --->
                                       u-boot Configuration  --->
                                       Linux Configuration  --->
                                       Image Packaging Configuration  --->
                                       Firmware Version Configuration  --->
                                       Yocto Settings  --->

                         <Select>    < Exit >    < Help >    < Save >    < Load >
```

Petalinux - Vitis AI 2.0

- petalinux-config - Yocto Settings ---> User Layers

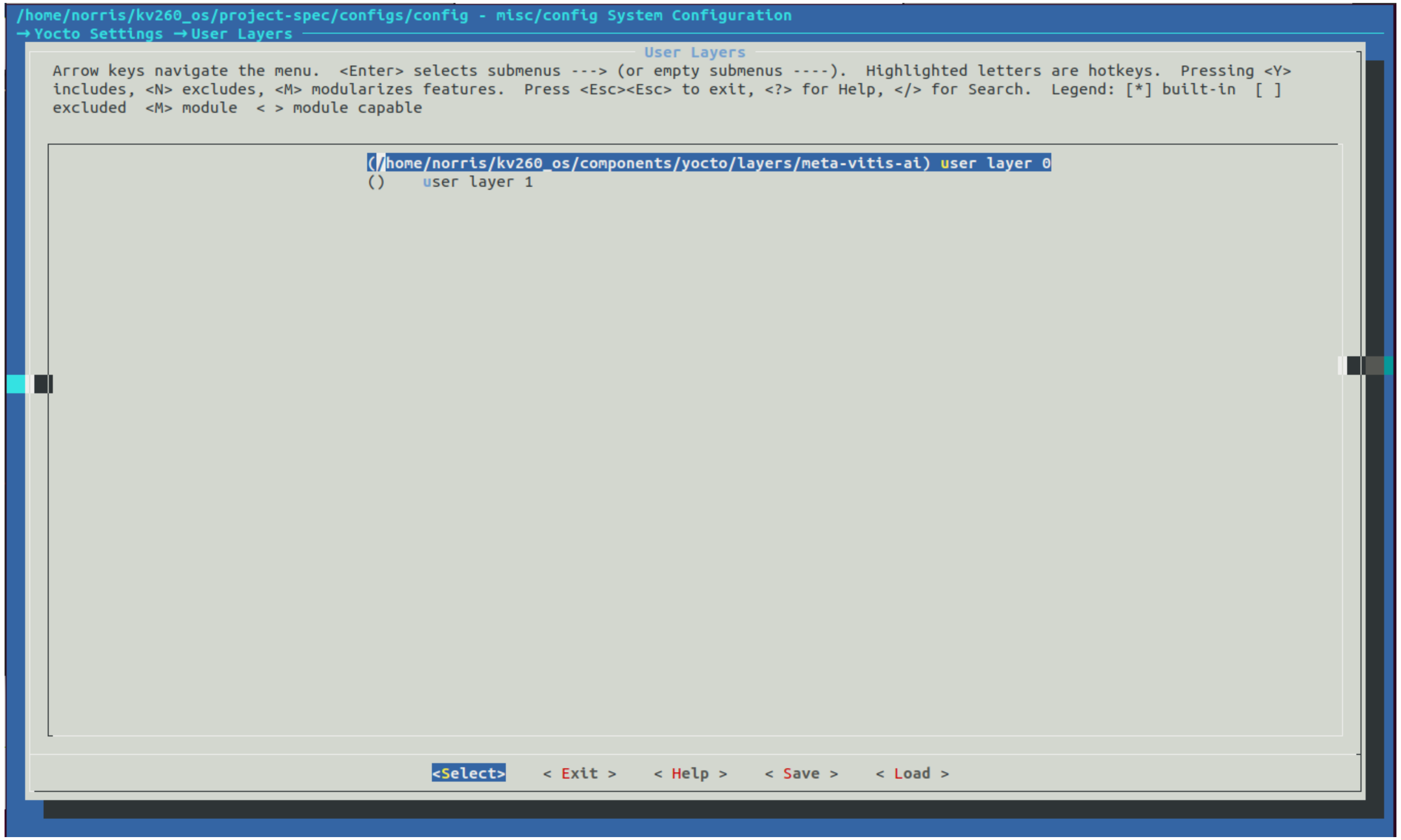


```
/home/norris/kv260_os/project-spec/configs/config - misc/config System Configuration
→ Yocto Settings ─────────────────────────────────────────────────────────────
                                  Yocto Settings
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing <Y>
  includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ]
  excluded  <M> module  < > module capable

                                  (zynqmp-generic) YOCTO_MACHINE_NAME
                                      Yocto board settings   --->
                                      TMPDIR Location  --->
                                      Devtool Workspace Location  --->
                                      Parallel thread execution  --->
                                      Add pre-mirror url   --->
                                      Local sstate feeds settings  --->
                                  [*] Enable Network sstate feeds
                                        Network sstate feeds URL  --->
                                  [ ] Enable BB NO NETWORK
                                  [ ] Enable Buildtools Extended
                                      User Layers  --->

                   <Select>    < Exit >    < Help >    < Save >    < Load >
```

XILINX

Petalinux - Vitis AI 2.0

- petalinux-config - /home/xxx/<project_name>components/yocto/layers/meta-vitis-ai

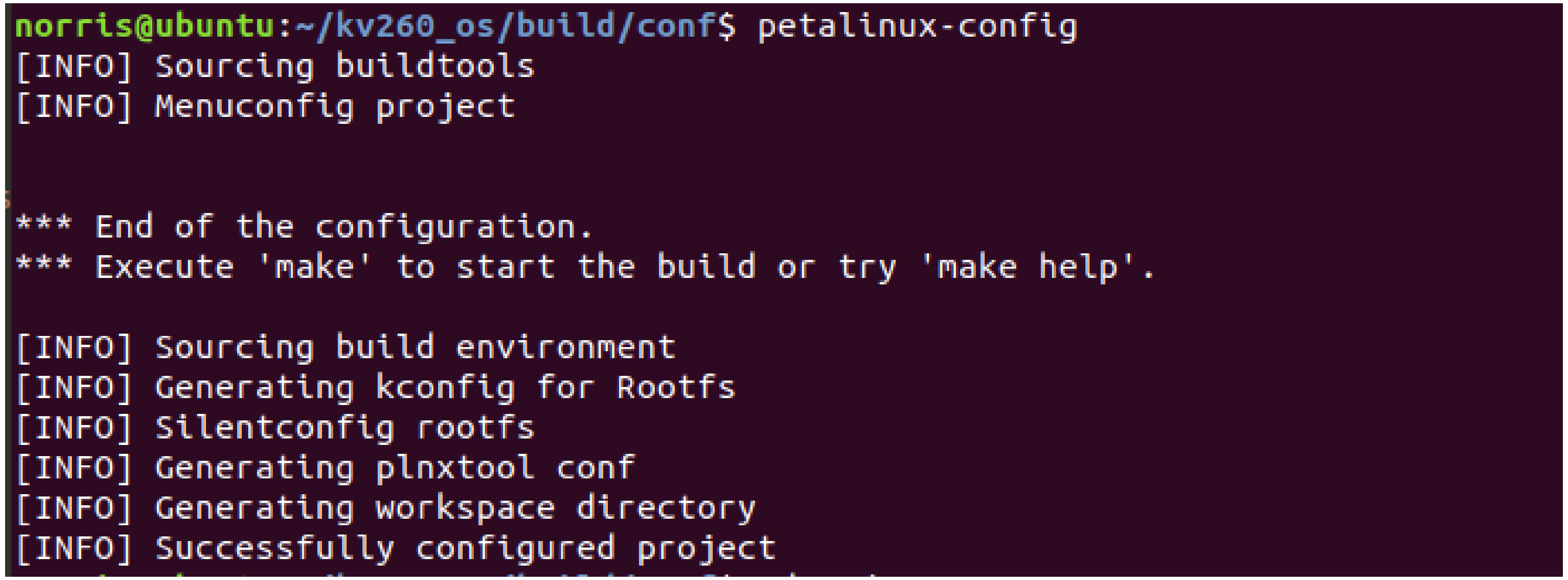


```
/home/norris/kv260_os/project-spec/configs/config - misc/config System Configuration
→ Yocto Settings → User Layers
                                    User Layers
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing <Y>
  includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ]
  excluded  <M> module  < > module capable

                 (/home/norris/kv260_os/components/yocto/layers/meta-vitis-ai) user layer 0
                 ()    user layer 1

                 <Select>    < Exit >    < Help >    < Save >    < Load >
```

XILINX.

Petalinux - Vitis AI 2.0

- petalinux-config



```
norris@ubuntu:~/kv260_os/build/conf$ petalinux-config
[INFO] Sourcing buildtools
[INFO] Menuconfig project


*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.

[INFO] Sourcing build environment
[INFO] Generating kconfig for Rootfs
[INFO] Silentconfig rootfs
[INFO] Generating plnxtool conf
[INFO] Generating workspace directory
[INFO] Successfully configured project
```

Petalinux - Vitis AI 2.0

- petalinux-config
  1. vi components/yocto/layers/meta-petalinux/recipes-core/images/petalinux-image-minimal.bb
  2. Add the following words

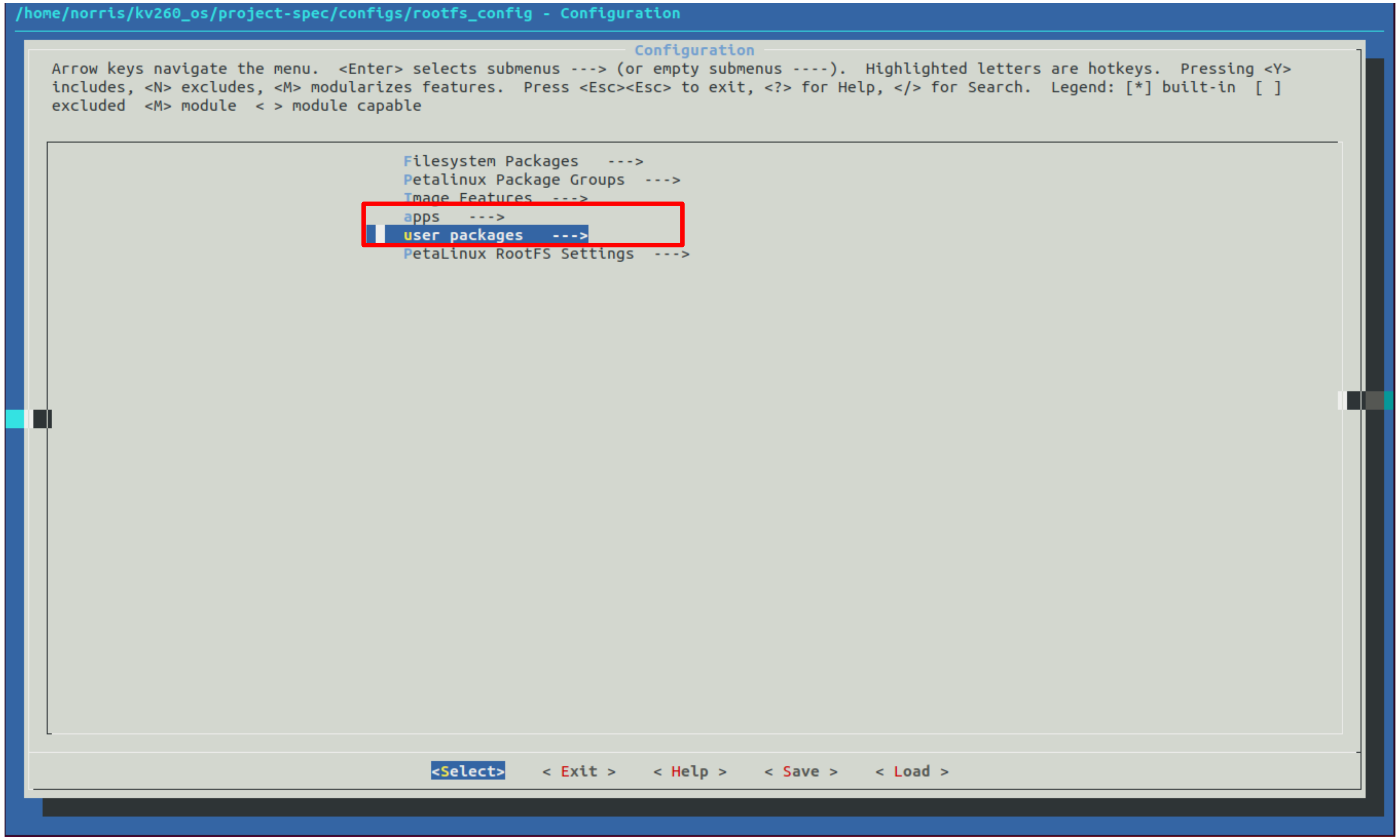
```
IMAGE_INSTALL_append = "packagegroup-petalinux-vitisai-dev \
                        packagegroup-petalinux-vitisai"
```

Petalinux

- Add Vitis AI 2.0 & Firmware to Petalinux
  1. cd ~/kv260_os
  2. git clone -b xlnx_rel_v2021.2 https://github.com/Xilinx/kv260-firmware
  3. petalinux-create -t apps --template fpgamanager -n karp-smartcam --enable --srcuri "./kv260-firmware/smartcam/kv260-smartcam.bit \
     ./kv260-firmware/smartcam/kv260-smartcam.xclbin \
     ./kv260-firmware/smartcam/shell.json \
     ./kv260-firmware/smartcam/kv260-smartcam.dtsi“
  4. petalinux-config -c rootfs

Petalinux - Firmware

- petalinux-config -c rootfs – apps & user packages

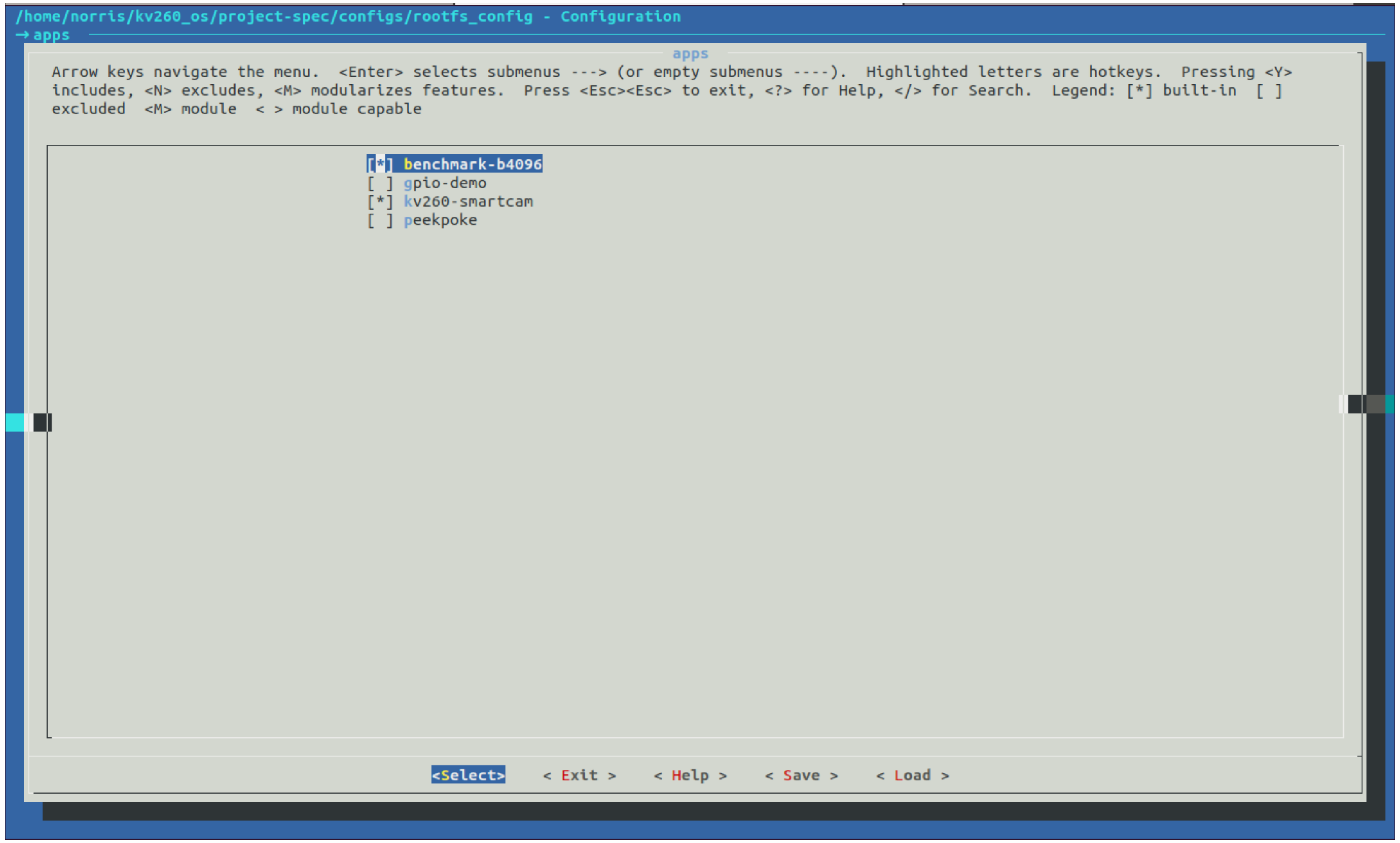


```
/home/norris/kv260_os/project-spec/configs/rootfs_config - Configuration
                                  Configuration
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing <Y>
  includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ]
  excluded  <M> module  < > module capable

                                  Filesystem Packages  --->
                                  Petalinux Package Groups  --->
                                  Image Features  --->
                                  apps  --->
                                  user packages  --->
                                  PetaLinux RootFS Settings  --->

                  <Select>    < Exit >    < Help >    < Save >    < Load >
```

XILINX

Petalinux - Firmware

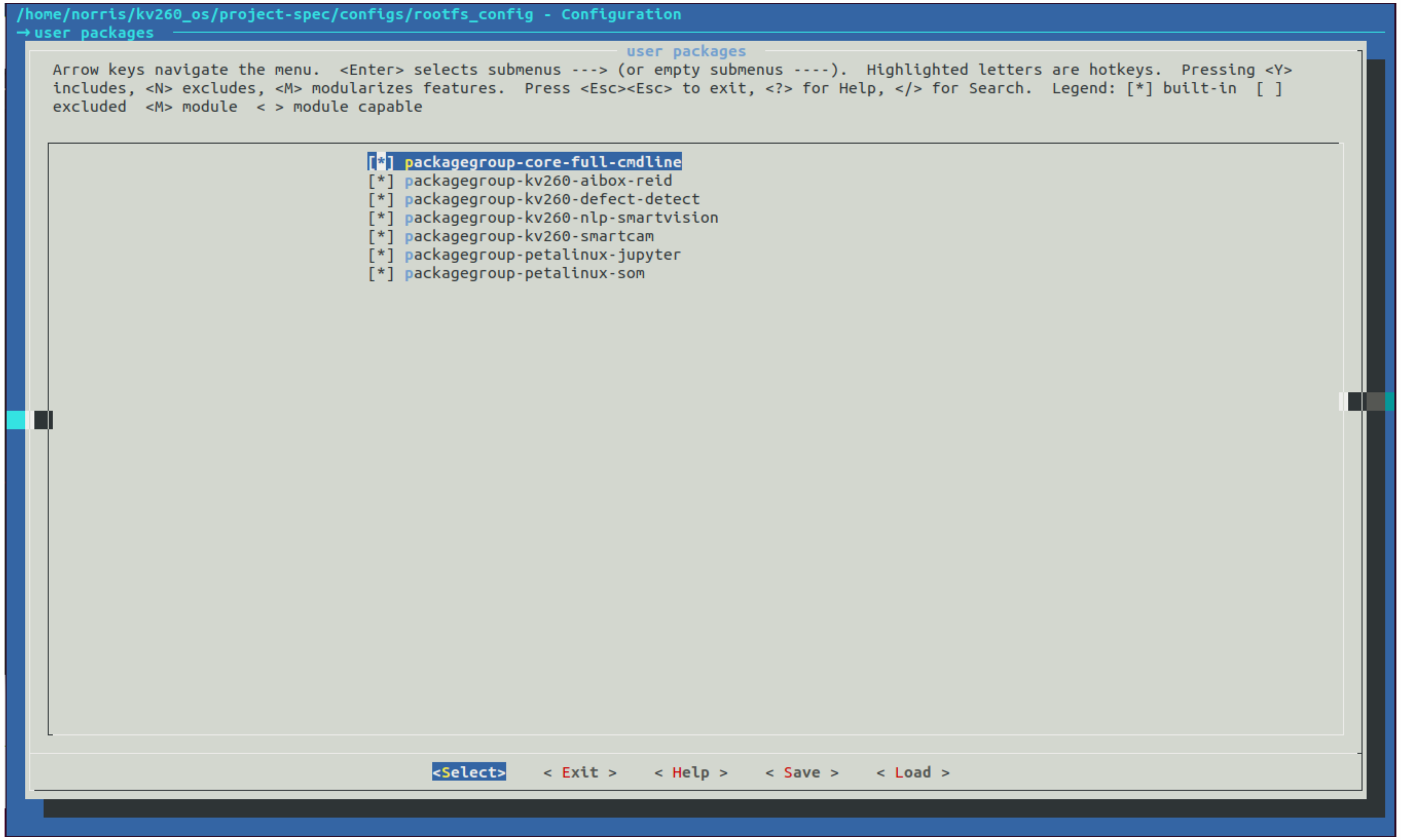
- petalinux-config -c rootfs – apps: check the karp-smartcam if has been selected



```
/home/norris/kv260_os/project-spec/configs/rootfs_config - Configuration
→ apps
                                     apps
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing <Y>
  includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ]
  excluded  <M> module  < > module capable

                                  [*] benchmark-b4096
                                  [ ] gpio-demo
                                  [*] kv260-smartcam
                                  [ ] peekpoke

                    <Select>    < Exit >    < Help >    < Save >    < Load >
```

Petalinux - Firmware

- petalinux-config -c rootfs – user packages

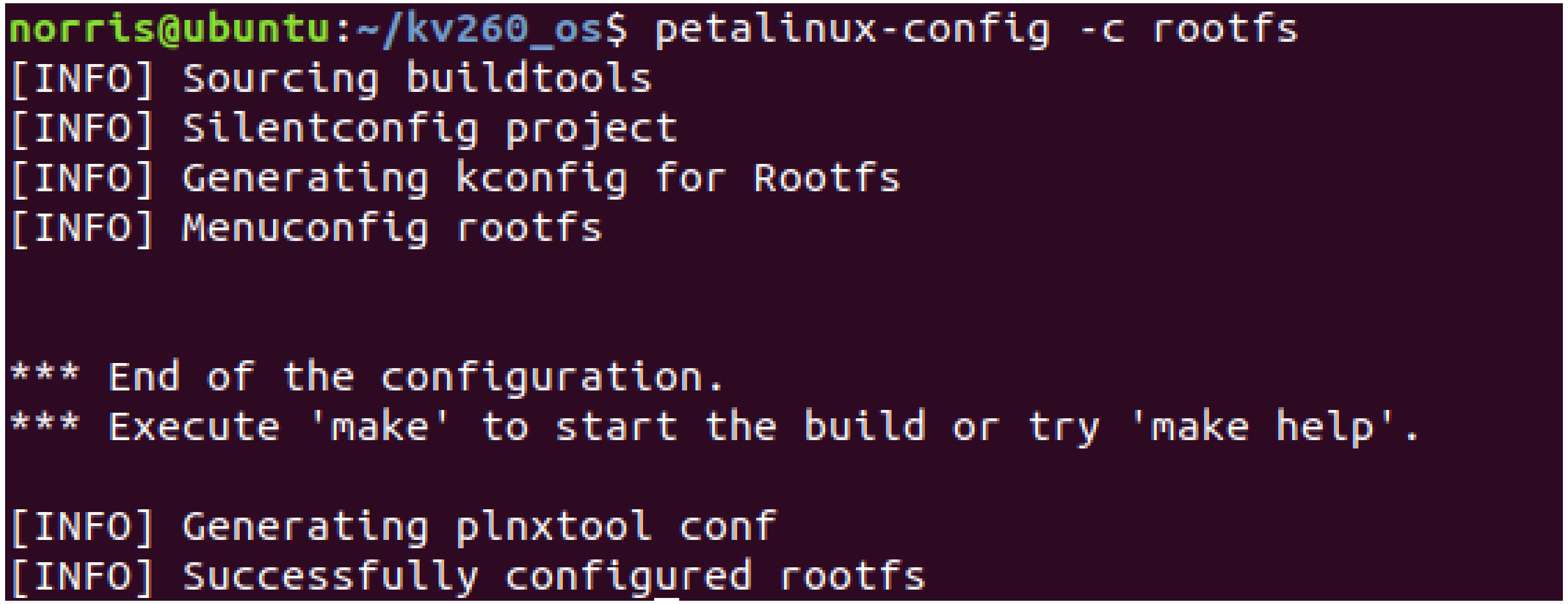


```
/home/norris/kv260_os/project-spec/configs/rootfs_config - Configuration
→ user packages
                                 user packages
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing <Y>
  includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ]
  excluded  <M> module  < > module capable

                      [*] packagegroup-core-full-cmdline
                      [*] packagegroup-kv260-aibox-reid
                      [*] packagegroup-kv260-defect-detect
                      [*] packagegroup-kv260-nlp-smartvision
                      [*] packagegroup-kv260-smartcam
                      [*] packagegroup-petalinux-jupyter
                      [*] packagegroup-petalinux-som

                <Select>    < Exit >    < Help >    < Save >    < Load >
```

XILINX.

Petalinux - Firmware

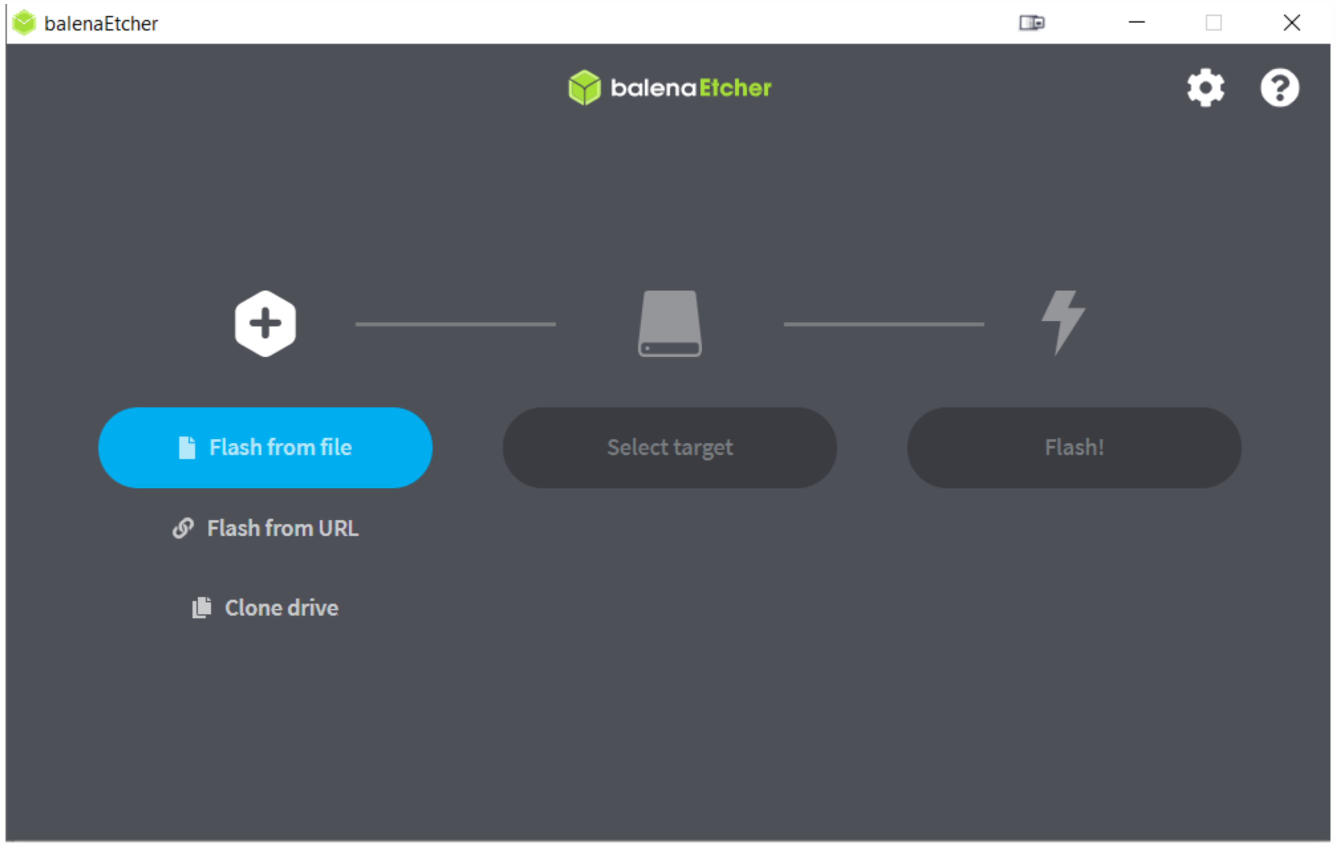
- petalinux-config -c rootfs



```
norris@ubuntu:~/kv260_os$ petalinux-config -c rootfs
[INFO] Sourcing buildtools
[INFO] Silentconfig project
[INFO] Generating kconfig for Rootfs
[INFO] Menuconfig rootfs


*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.

[INFO] Generating plnxtool conf
[INFO] Successfully configured rootfs
```

Petalinux

1. petalinux-build -c petalinux-image-minimal
2. petalinux-package --boot --u-boot --dtb images/linux/u-boot.dtb –force
3. petalinux-package –wic
4. The built image will locate at images/linux/petalinux-sdimage.wic



Vitis-AI

- Install Vitis AI 2.0 on Ubuntu 18.04

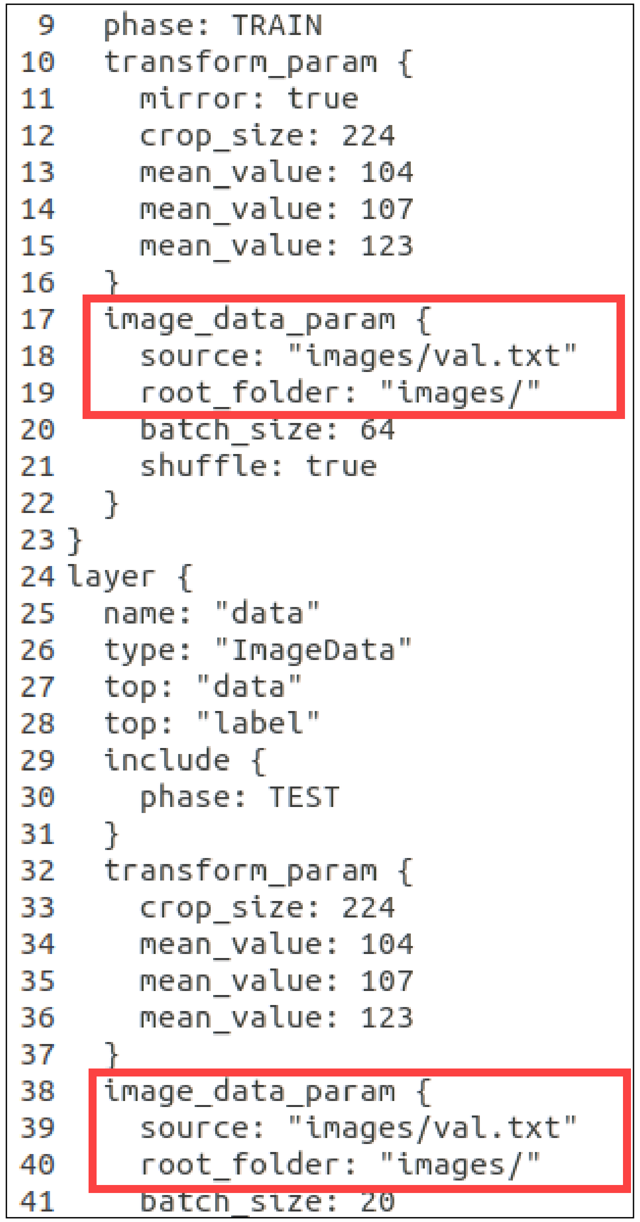
1. sudo apt-get install docker.io
2. sudo chmod 777 /var/run/docker.sock
3. docker pull xilinx/vitis-ai:2.0.0
4. git clone -b v2.0 https://github.com/Xilinx/Vitis-AI.git
5. cd Vitis-AI/setup/mpsoc
6. sudo ./ sdk-2021.2.0.0.sh
7. . /<sdk location>/ environment-setup-cortexa72-cortexa53-xilinx-linux
8. cd Vitis-AI/demo/VART/resnet50
9. bash -x build.sh

Set Cross-Compile Environment

Build resnet50 execution file

Vitis-AI

- Quantize Resnet50 Model (the flow is like Lab 1)
  1. ./docker_run.sh xilinx/vitis-ai:2.0.0
  2. conda activate vitis-ai-caffe
  3. cp -r training/vai_q_c Vitis-AI
  4. vim Vitis-AI/vai_q_c/lab/1_caffe_quantize_for_edge.sh
  5. modify the value of calib_iterc and test_iter from 10 to 2
  6. vim Vitis-AI/vai_q_c/lab/
     cf_resnet50_imagenet_224_224_7.7G_2.0/float/trainval.prototxt
  7. sh 1_caffe_quantize_for_edge.sh



```
    phase: TRAIN
    transform_param {
      mirror: true
      crop_size: 224
      mean_value: 104
      mean_value: 107
      mean_value: 123
    }
    image_data_param {
      source: "images/val.txt"
      root_folder: "images/"
      batch_size: 64
      shuffle: true
    }
}
layer {
    name: "data"
    type: "ImageData"
    top: "data"
    top: "label"
    include {
      phase: TEST
    }
    transform_param {
      crop_size: 224
      mean_value: 104
      mean_value: 107
      mean_value: 123
    }
    image_data_param {
      source: "images/val.txt"
      root_folder: "images/"
      batch_size: 20
```

Vitis-AI

- Compile Resnet50 Model to xmodel
  1. vim Vitis-AI/vai_q_c/lab/2_caffe_compile_for_edge.sh
  2. modify EDGE_TARGET=ZCU102 to EDGE_TARGET=KV260
  3. vim /opt/vitis_ai/compiler/arch/DPUCZDX8G/KV260/arch.json
  4. modify 4096 to 3136
  5. sh 2_caffe_compile_for_edge.sh

KV260

- Create Folder
  1. mkdir -p Vitis-AI/demo/VART/resnet50
  2. mkdir -p /usr/share/vitis_ai_library/models/resnet50/
- Copy the file of resnet50 execution and label
  1. cd <Vitis-AI on Ubuntu>/demo/VART/resnet50/
  2. scp -r resnet50 words.txt <KV260的username>@<IP>:~/Vitis-AI/demo/VART/resnet50/
- Copy VART install files to KV260
  1. cd <Vitis-AI on Ubuntu>/setup/
  2. scp -r mpsoc <KV260的username>@<IP>:~/
- Copy the images for inference
  1. wget https://www.xilinx.com/bin/public/openDownload?filename=vitis_ai_runtime_r2.0.0_image_video.tar.gz
  2. tar -zxvf vitis_ai_runtime_r2.0.0_image_video.tar.gz -C ~/Vitis-AI/demo/VART/

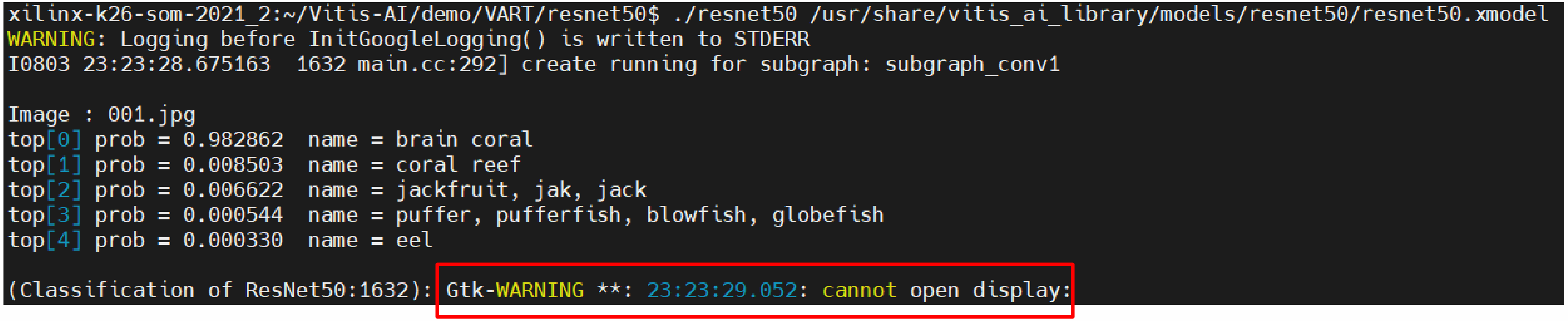
KV260

- The xmodel KV260 will run
  1. cd < Vitis-AI on Ubuntu > /vai_q_c/lab/cf_resnet50_imagenet_224_224_7.7G_2.0/vai_c_output_KV260/
  2. scp -r md5sum.txt meta.json resnet50.xmodel <username of KV260>@<IP>:/ usr/share/vitis_ai_library/models/resnet50/

KV260 – Install VART & Run Demo

- Install VART
  1. cd mpsoc/VART
  2. sh bash target_vart_setup.sh
- Load App
  1. sudo xmutil unloadapp
  2. sudo xmutil loadapp karp-smartcam
- Run Demo
  1. cd ~/Vitis-AI/demo/VART/resnet50/
  2. ./resnet50 /usr/share/vitis_ai_library/models/resnet50/resnet50.xmodel

KV260 – Demo Result

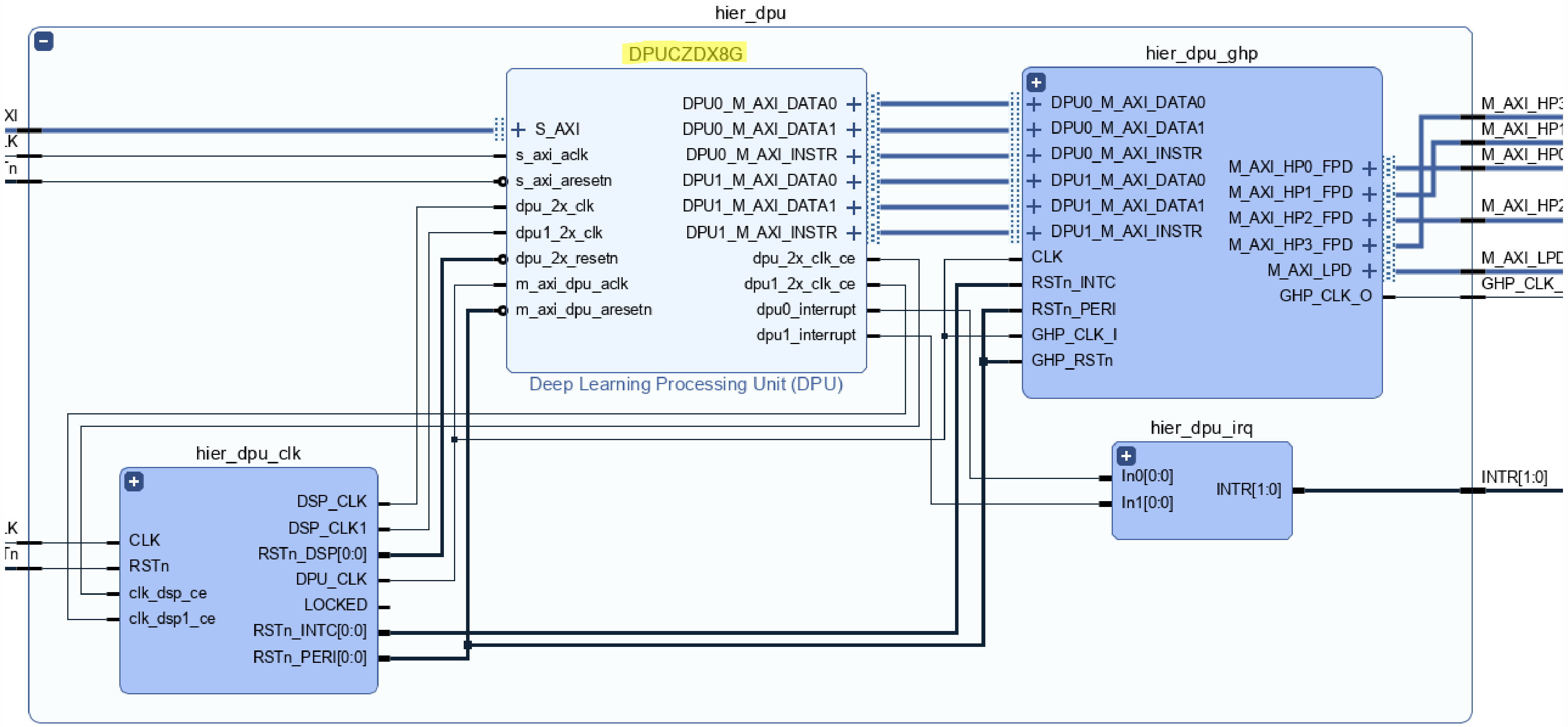


```
xilinx-k26-som-2021_2:~/Vitis-AI/demo/VART/resnet50$ ./resnet50 /usr/share/vitis_ai_library/models/resnet50/resnet50.xmodel
WARNING: Logging before InitGoogleLogging() is written to STDERR
I0803 23:23:28.675163  1632 main.cc:292] create running for subgraph: subgraph_conv1

Image : 001.jpg
top[0] prob = 0.982862  name = brain coral
top[1] prob = 0.008503  name = coral reef
top[2] prob = 0.006622  name = jackfruit, jak, jack
top[3] prob = 0.000544  name = puffer, pufferfish, blowfish, globefish
top[4] prob = 0.000330  name = eel

(Classification of ResNet50:1632): Gtk-WARNING **: 23:23:29.052: cannot open display:
```

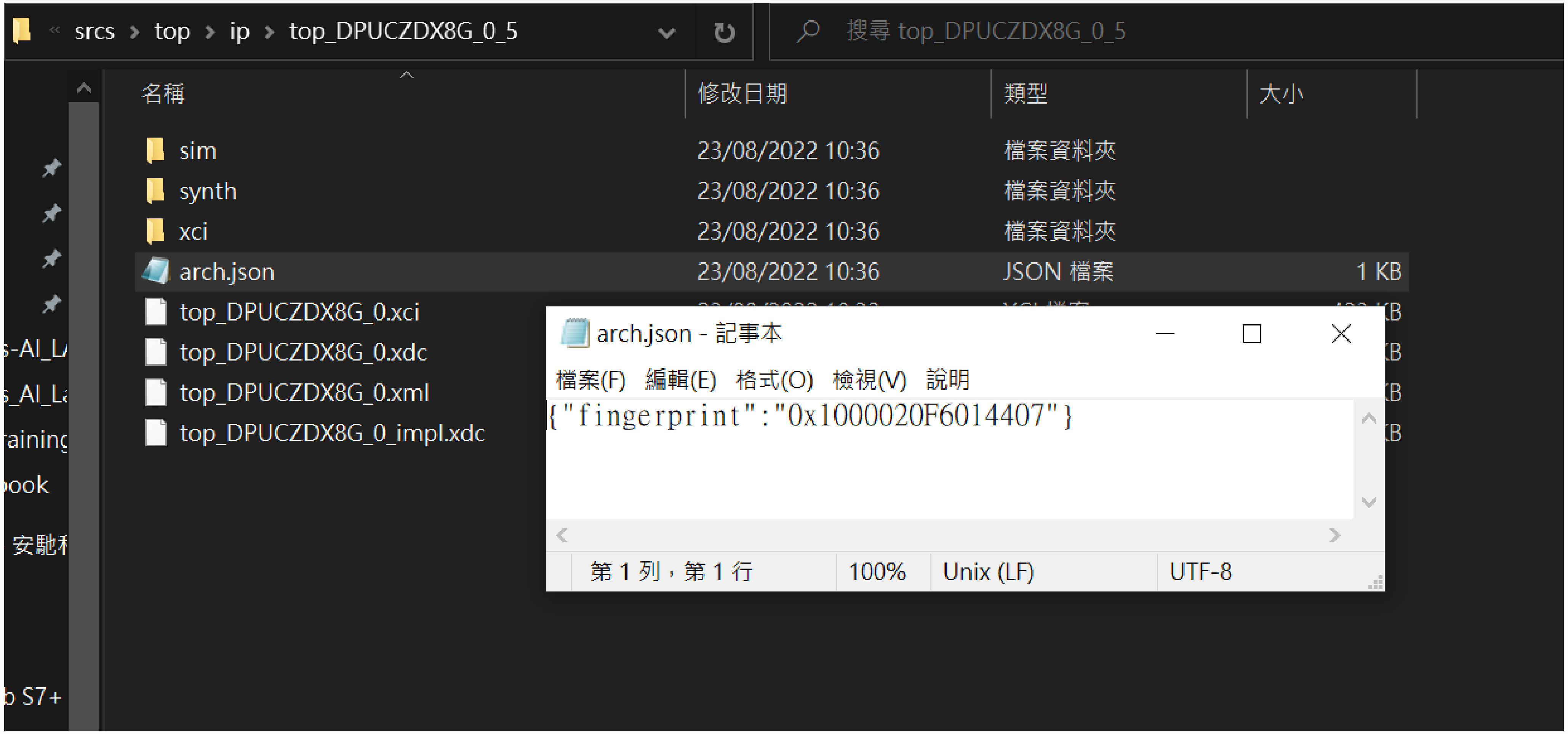
Can’t not display the image through X11-forwarding/Gtk

Lab 4: Creating a Hardware Platform with the DPU Using the Vivado Design Suite Flow – Using KV260

Lab 4: Creating a Hardware Platform with the DPU Using the Vivado Design Suite Flow



Lab 4: Creating a Hardware Platform with the DPU Using the Vivado Design Suite Flow



srcs › top › ip › top_DPUCZDX8G_0_5

搜尋 top_DPUCZDX8G_0_5

名稱	修改日期	類型	大小
sim	23/08/2022 10:36	檔案資料夾	
synth	23/08/2022 10:36	檔案資料夾	
xci	23/08/2022 10:36	檔案資料夾	
arch.json	23/08/2022 10:36	JSON 檔案	1 KB
top_DPUCZDX8G_0.xci			
top_DPUCZDX8G_0.xdc			
top_DPUCZDX8G_0.xml			
top_DPUCZDX8G_0_impl.xdc			

arch.json - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

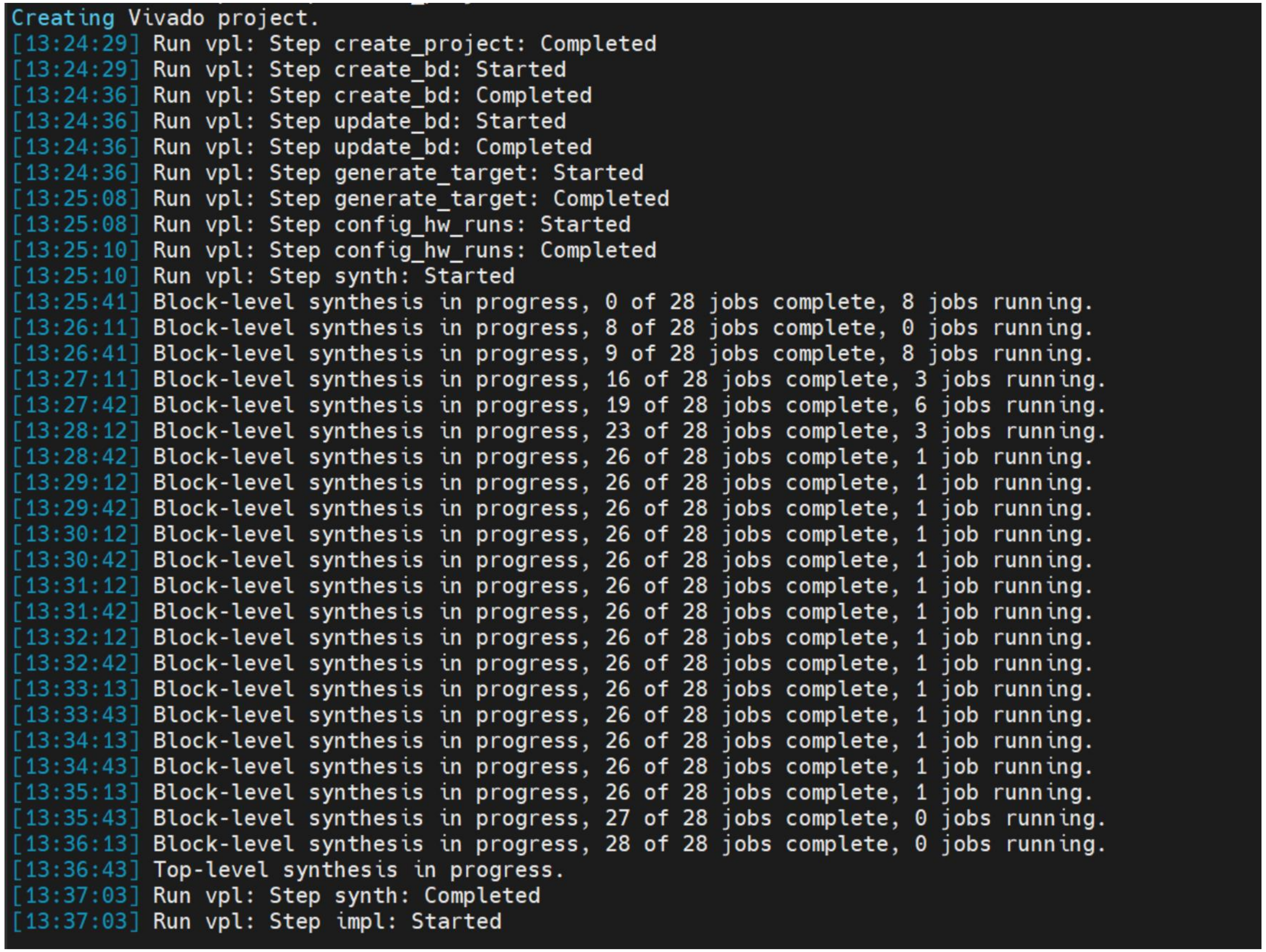
```
{"fingerprint":"0x1000020F6014407"}
```

第 1 列，第 1 行 100% Unix (LF) UTF-8

Lab 5: Creating a DPU Kernel Using the Vitis Environment Flow – Using KV260

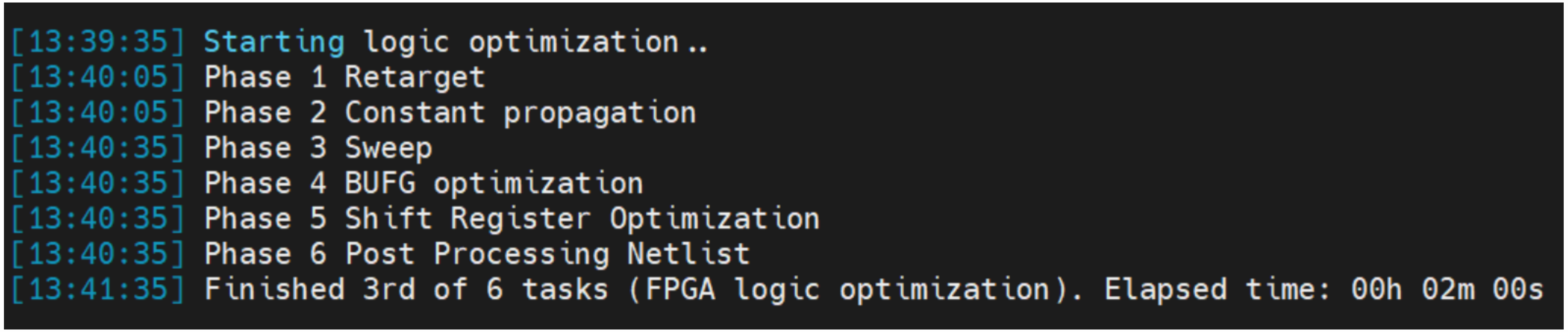
XILINX.

Lab 5: Creating a DPU Kernel Using the Vitis Environment Flow

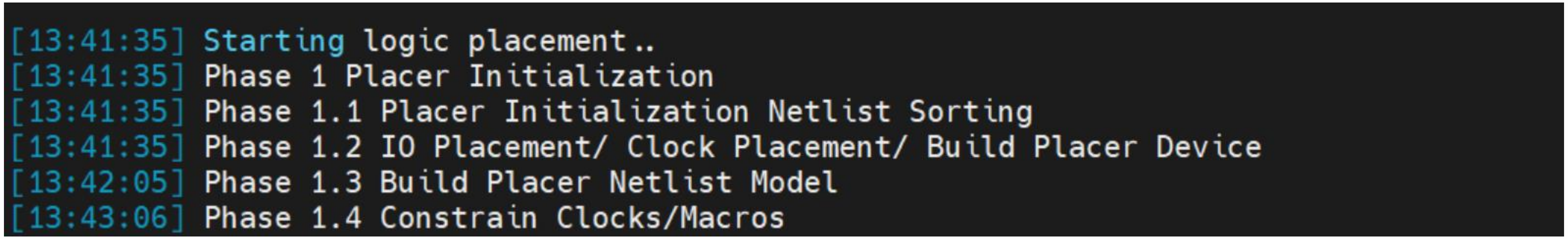


```
Creating Vivado project.
[13:24:29] Run vpl: Step create_project: Completed
[13:24:29] Run vpl: Step create_bd: Started
[13:24:36] Run vpl: Step create_bd: Completed
[13:24:36] Run vpl: Step update_bd: Started
[13:24:36] Run vpl: Step update_bd: Completed
[13:24:36] Run vpl: Step generate_target: Started
[13:25:08] Run vpl: Step generate_target: Completed
[13:25:08] Run vpl: Step config_hw_runs: Started
[13:25:10] Run vpl: Step config_hw_runs: Completed
[13:25:10] Run vpl: Step synth: Started
[13:25:41] Block-level synthesis in progress, 0 of 28 jobs complete, 8 jobs running.
[13:26:11] Block-level synthesis in progress, 8 of 28 jobs complete, 0 jobs running.
[13:26:41] Block-level synthesis in progress, 9 of 28 jobs complete, 8 jobs running.
[13:27:11] Block-level synthesis in progress, 16 of 28 jobs complete, 3 jobs running.
[13:27:42] Block-level synthesis in progress, 19 of 28 jobs complete, 6 jobs running.
[13:28:12] Block-level synthesis in progress, 23 of 28 jobs complete, 3 jobs running.
[13:28:42] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:29:12] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:29:42] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:30:12] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:30:42] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:31:12] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:31:42] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:32:12] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:32:42] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:33:13] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:33:43] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:34:13] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:34:43] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:35:13] Block-level synthesis in progress, 26 of 28 jobs complete, 1 job running.
[13:35:43] Block-level synthesis in progress, 27 of 28 jobs complete, 0 jobs running.
[13:36:13] Block-level synthesis in progress, 28 of 28 jobs complete, 0 jobs running.
[13:36:43] Top-level synthesis in progress.
[13:37:03] Run vpl: Step synth: Completed
[13:37:03] Run vpl: Step impl: Started
```

Lab 5: Creating a DPU Kernel Using the Vitis Environment Flow



```
[13:39:35] Starting logic optimization..
[13:40:05] Phase 1 Retarget
[13:40:05] Phase 2 Constant propagation
[13:40:35] Phase 3 Sweep
[13:40:35] Phase 4 BUFG optimization
[13:40:35] Phase 5 Shift Register Optimization
[13:40:35] Phase 6 Post Processing Netlist
[13:41:35] Finished 3rd of 6 tasks (FPGA logic optimization). Elapsed time: 00h 02m 00s
```



```
[13:41:35] Starting logic placement..
[13:41:35] Phase 1 Placer Initialization
[13:41:35] Phase 1.1 Placer Initialization Netlist Sorting
[13:41:35] Phase 1.2 IO Placement/ Clock Placement/ Build Placer Device
[13:42:05] Phase 1.3 Build Placer Netlist Model
[13:43:06] Phase 1.4 Constrain Clocks/Macros
```

Lab 6: Creating a Custom Application – Edge – Using KV260

XILINX.

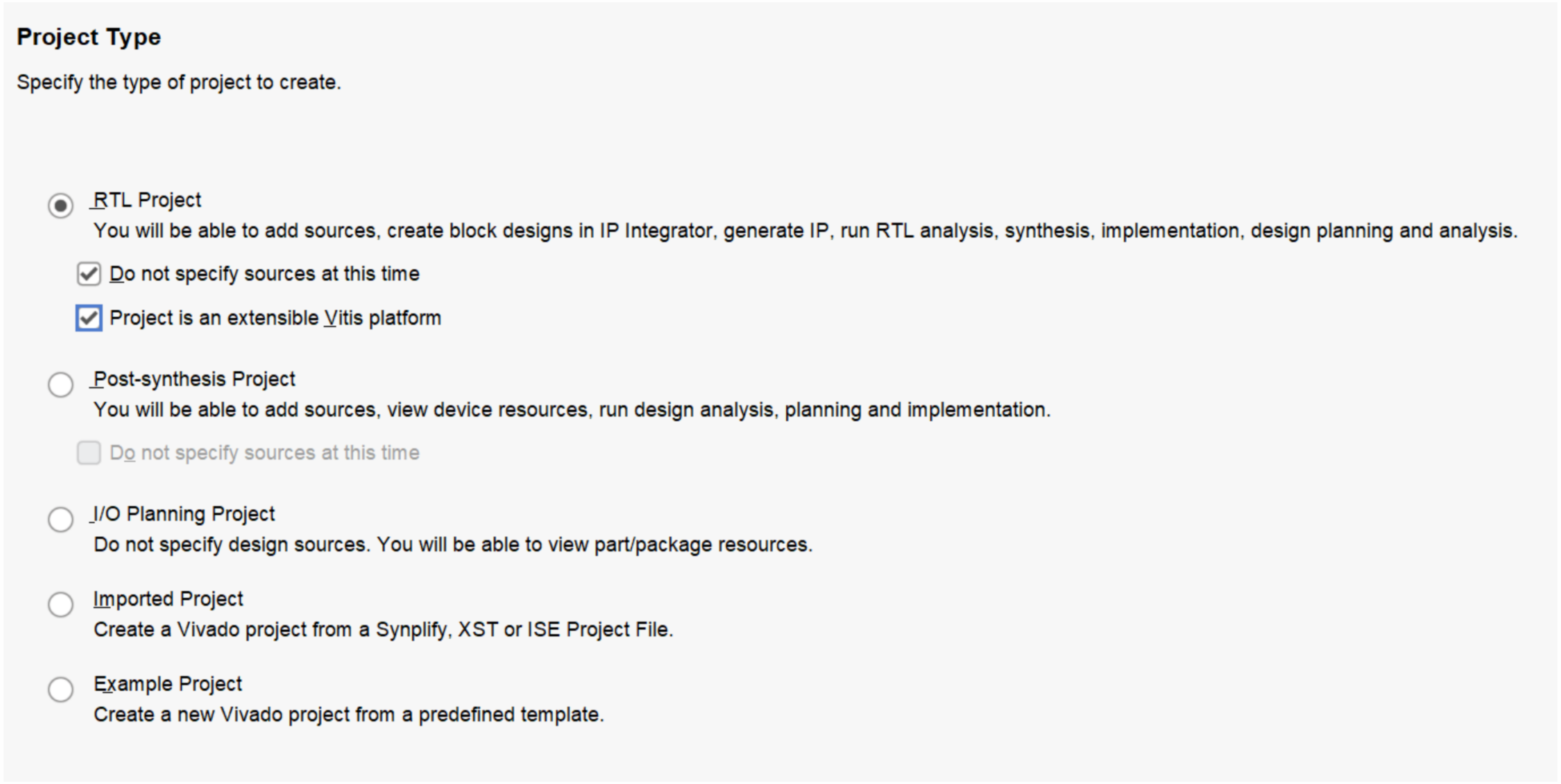
Lab 4 to 6: Using KV260

Overview

Vivado	Create Block Design for KV260
Petalinux	Create Petalinux Project; Generate Device Tree
Vitis	Create Platform & Application

XILINX.

KV260 – Vivado part



Project Type

Specify the type of project to create.

● RTL Project
You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.

[✓] Do not specify sources at this time

[✓] Project is an extensible Vitis platform

○ Post-synthesis Project
You will be able to add sources, view device resources, run design analysis, planning and implementation.

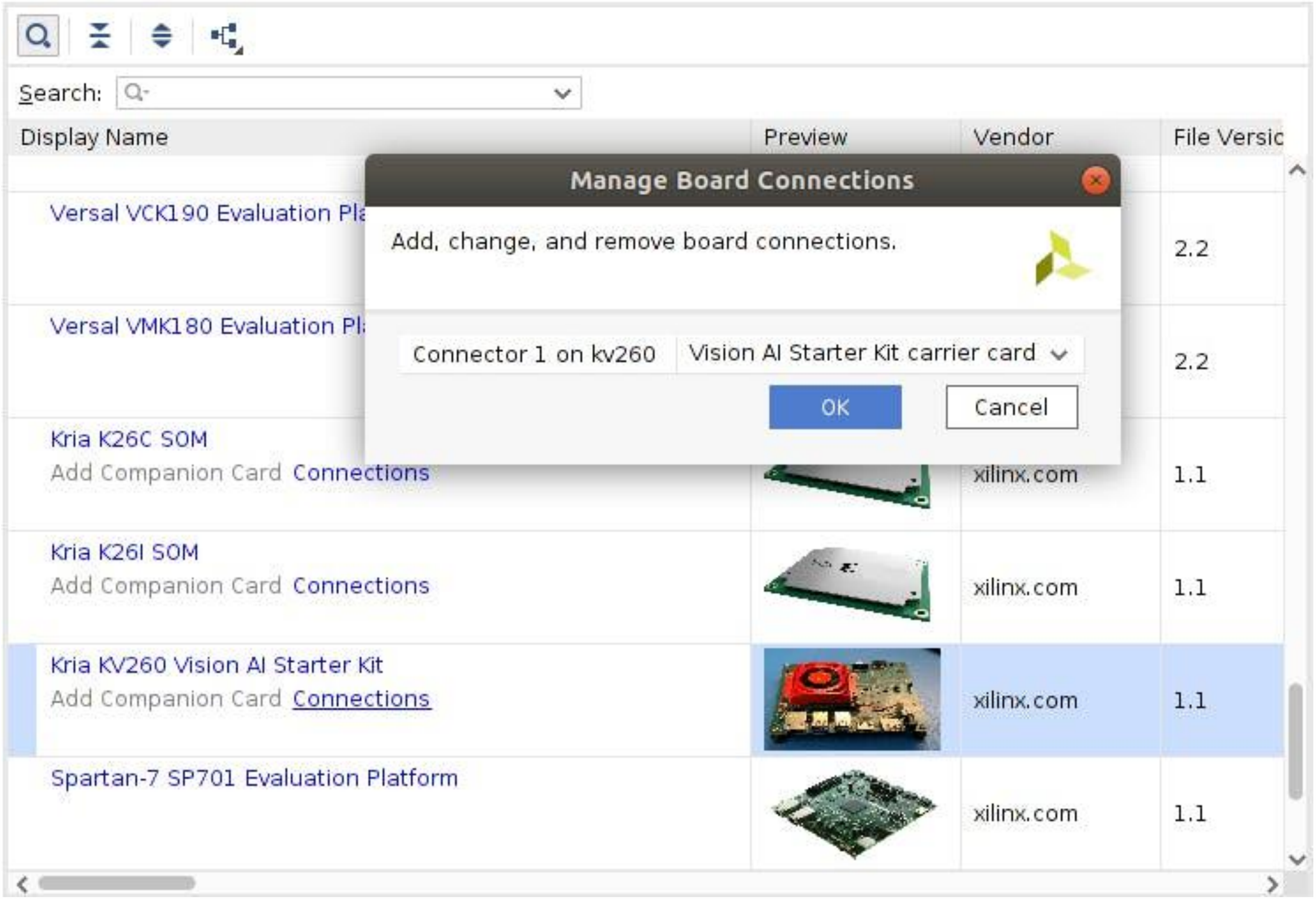
[ ] Do not specify sources at this time

○ I/O Planning Project
Do not specify design sources. You will be able to view part/package resources.

○ Imported Project
Create a Vivado project from a Synplify, XST or ISE Project File.

○ Example Project
Create a new Vivado project from a predefined template.

KV260 – Vivado part



Search:

Display Name	Preview	Vendor	File Versic
Versal VCK190 Evaluation Pla			2.2
Versal VMK180 Evaluation Pl			2.2
Kria K26C SOM Add Companion Card Connections		xilinx.com	1.1
Kria K26I SOM Add Companion Card Connections		xilinx.com	1.1
Kria KV260 Vision AI Starter Kit Add Companion Card Connections		xilinx.com	1.1
Spartan-7 SP701 Evaluation Platform		xilinx.com	1.1

Manage Board Connections

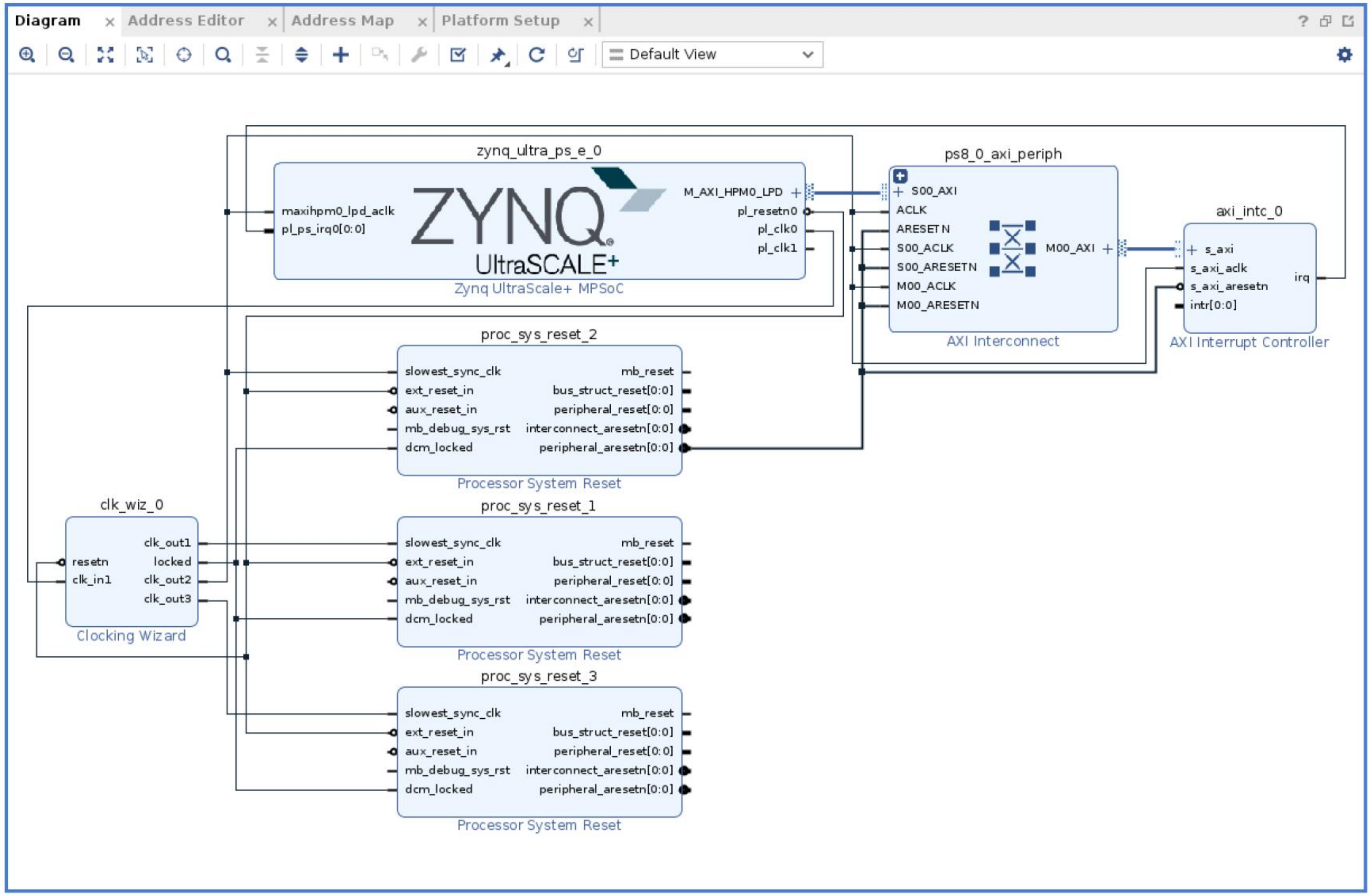
Add, change, and remove board connections.

Connector 1 on kv260 — Vision AI Starter Kit carrier card

XILINX.

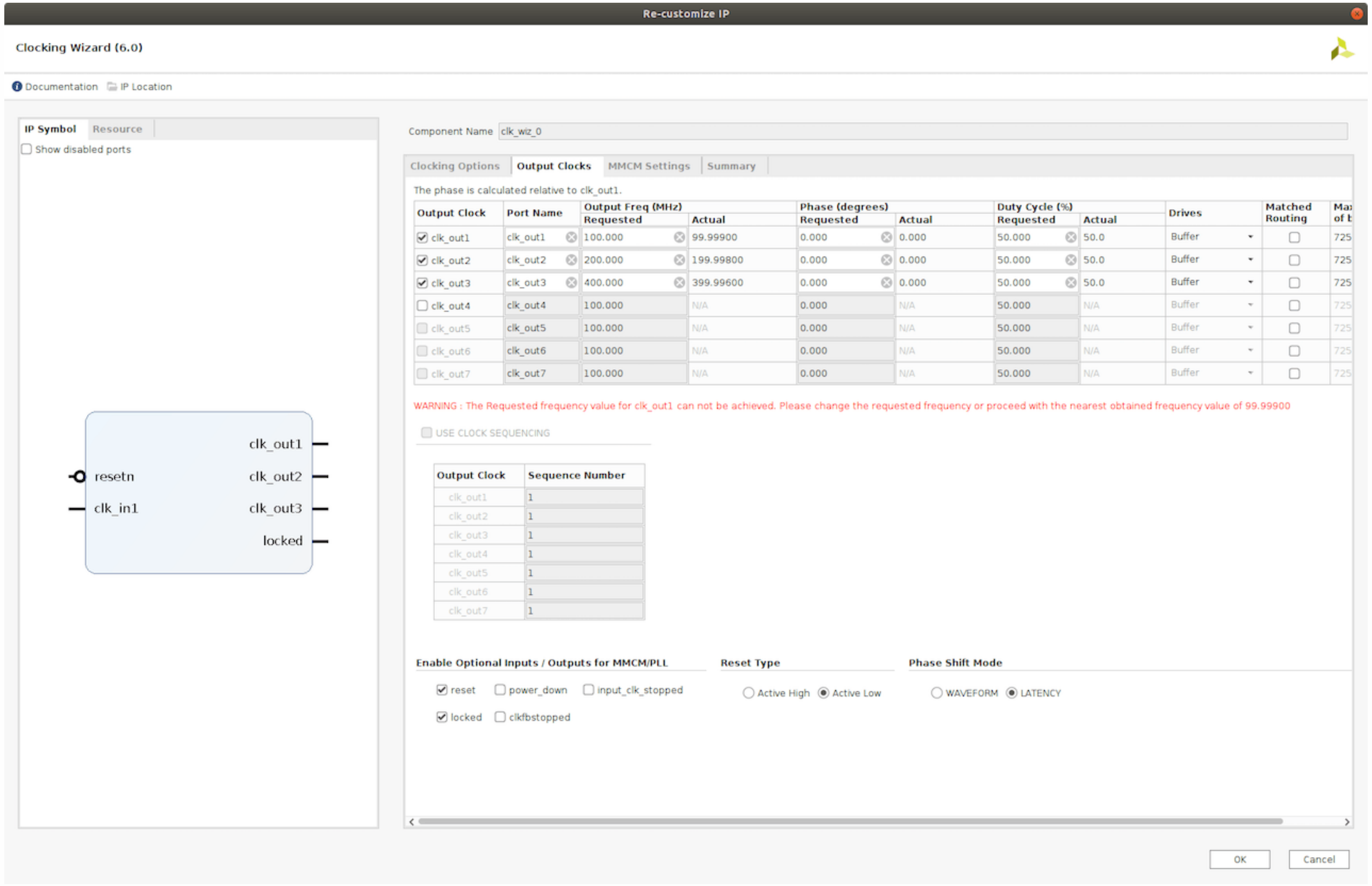
KV260 – Vivado part

- Xilinx KV260 github tutorial



KV260 – Vivado part

- Xilinx KV260 github tutorial



Component Name: clk_wiz_0

The phase is calculated relative to clk_out1.

Output Clock	Port Name	Output Freq (MHz)		Phase (degrees)		Duty Cycle (%)		Drives	Matched Routing	Ma of b
		Requested	Actual	Requested	Actual	Requested	Actual			
✓ clk_out1	clk_out1	100.000	99.99900	0.000	0.000	50.000	50.0	Buffer	☐	725
✓ clk_out2	clk_out2	200.000	199.99800	0.000	0.000	50.000	50.0	Buffer	☐	725
✓ clk_out3	clk_out3	400.000	399.99600	0.000	0.000	50.000	50.0	Buffer	☐	725
☐ clk_out4	clk_out4	100.000	N/A	0.000	N/A	50.000	N/A	Buffer	☐	725
☐ clk_out5	clk_out5	100.000	N/A	0.000	N/A	50.000	N/A	Buffer	☐	725
☐ clk_out6	clk_out6	100.000	N/A	0.000	N/A	50.000	N/A	Buffer	☐	725
☐ clk_out7	clk_out7	100.000	N/A	0.000	N/A	50.000	N/A	Buffer	☐	725

WARNING : The Requested frequency value for clk_out1 can not be achieved. Please change the requested frequency or proceed with the nearest obtained frequency value of 99.99900

☐ USE CLOCK SEQUENCING

Output Clock	Sequence Number
clk_out1	1
clk_out2	1
clk_out3	1
clk_out4	1
clk_out5	1
clk_out6	1
clk_out7	1

Enable Optional Inputs / Outputs for MMCM/PLL

✓ reset ☐ power_down ☐ input_clk_stopped

✓ locked ☐ clkfbstopped

Reset Type

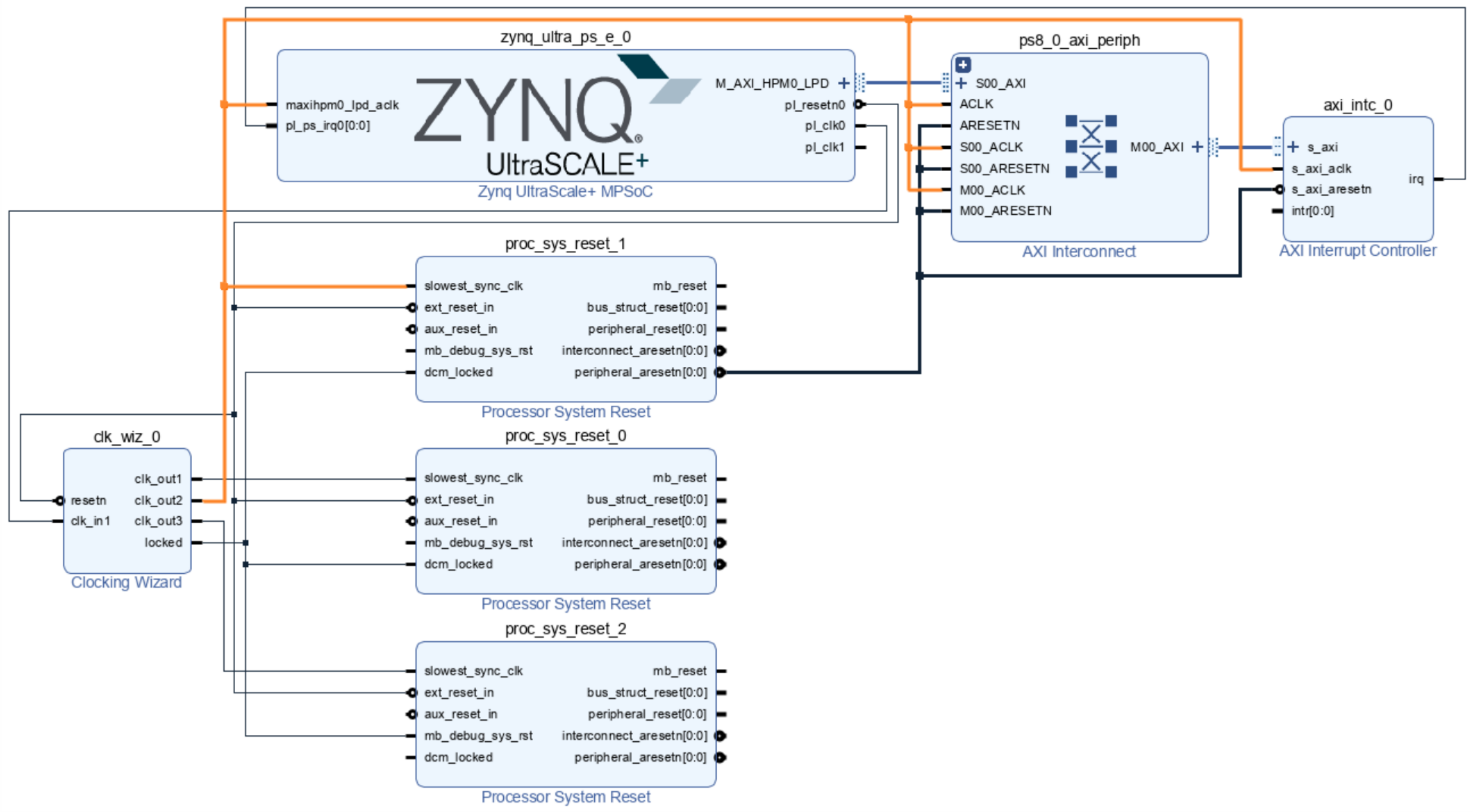
Active High [✓ Active Low]

Phase Shift Mode

WAVEFORM [✓ LATENCY]

KV260 – Vivado part

- Xilinx KV260 github tutorial



KV260 – Vivado part

- Xilinx KV260 github tutorial

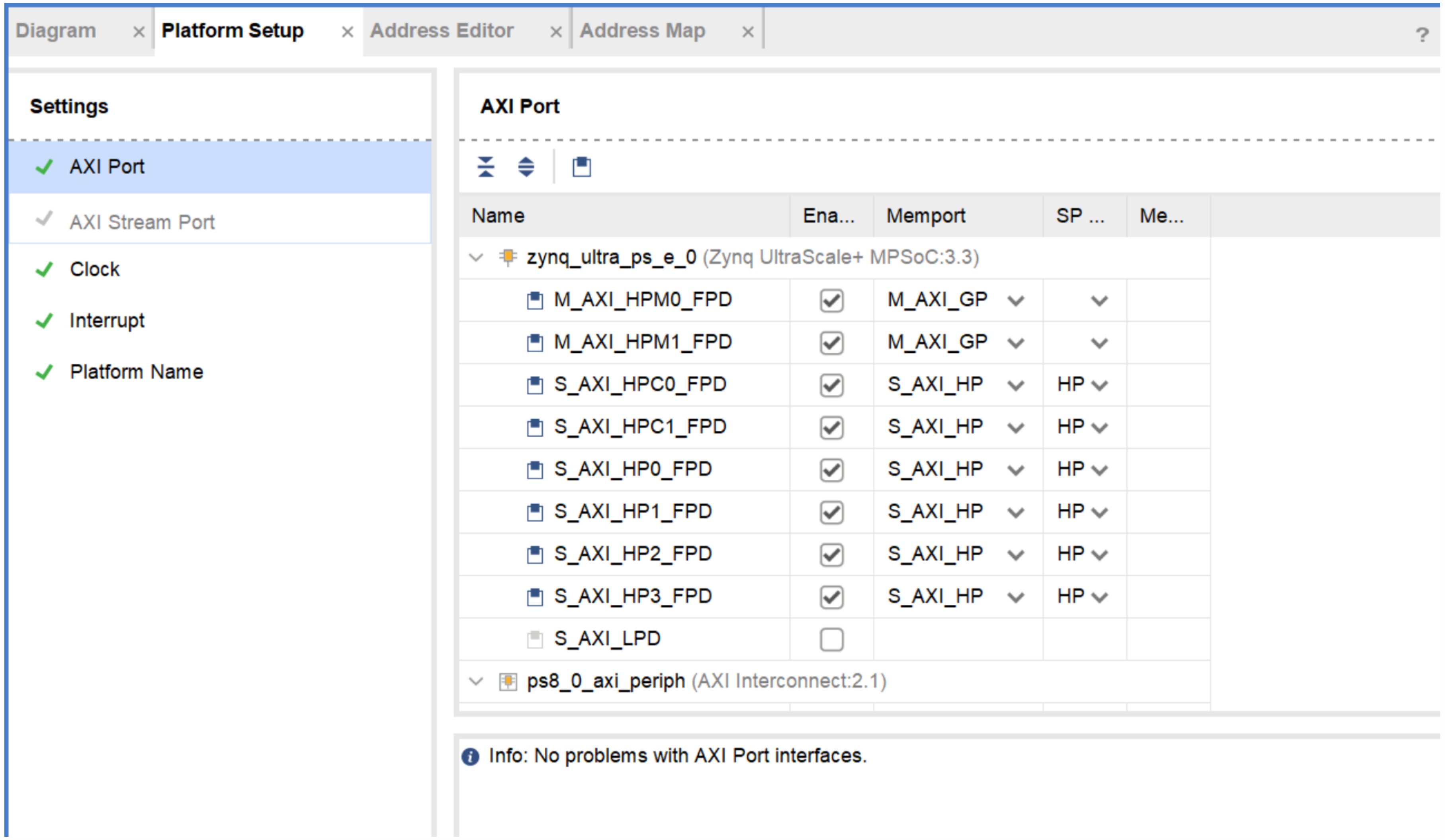


Diagram | Platform Setup | Address Editor | Address Map

Settings

- ✓ AXI Port
- ✓ AXI Stream Port
- ✓ Clock
- ✓ Interrupt
- ✓ Platform Name

AXI Port

Name	Ena...	Memport	SP ...	Me...
zynq_ultra_ps_e_0 (Zynq UltraScale+ MPSoC:3.3)				
M_AXI_HPM0_FPD	✓	M_AXI_GP		
M_AXI_HPM1_FPD	✓	M_AXI_GP		
S_AXI_HPC0_FPD	✓	S_AXI_HP	HP	
S_AXI_HPC1_FPD	✓	S_AXI_HP	HP	
S_AXI_HP0_FPD	✓	S_AXI_HP	HP	
S_AXI_HP1_FPD	✓	S_AXI_HP	HP	
S_AXI_HP2_FPD	✓	S_AXI_HP	HP	
S_AXI_HP3_FPD	✓	S_AXI_HP	HP	
S_AXI_LPD	☐			
ps8_0_axi_periph (AXI Interconnect:2.1)				

Info: No problems with AXI Port interfaces.

KV260 – Vivado part

- Xilinx KV260 github tutorial

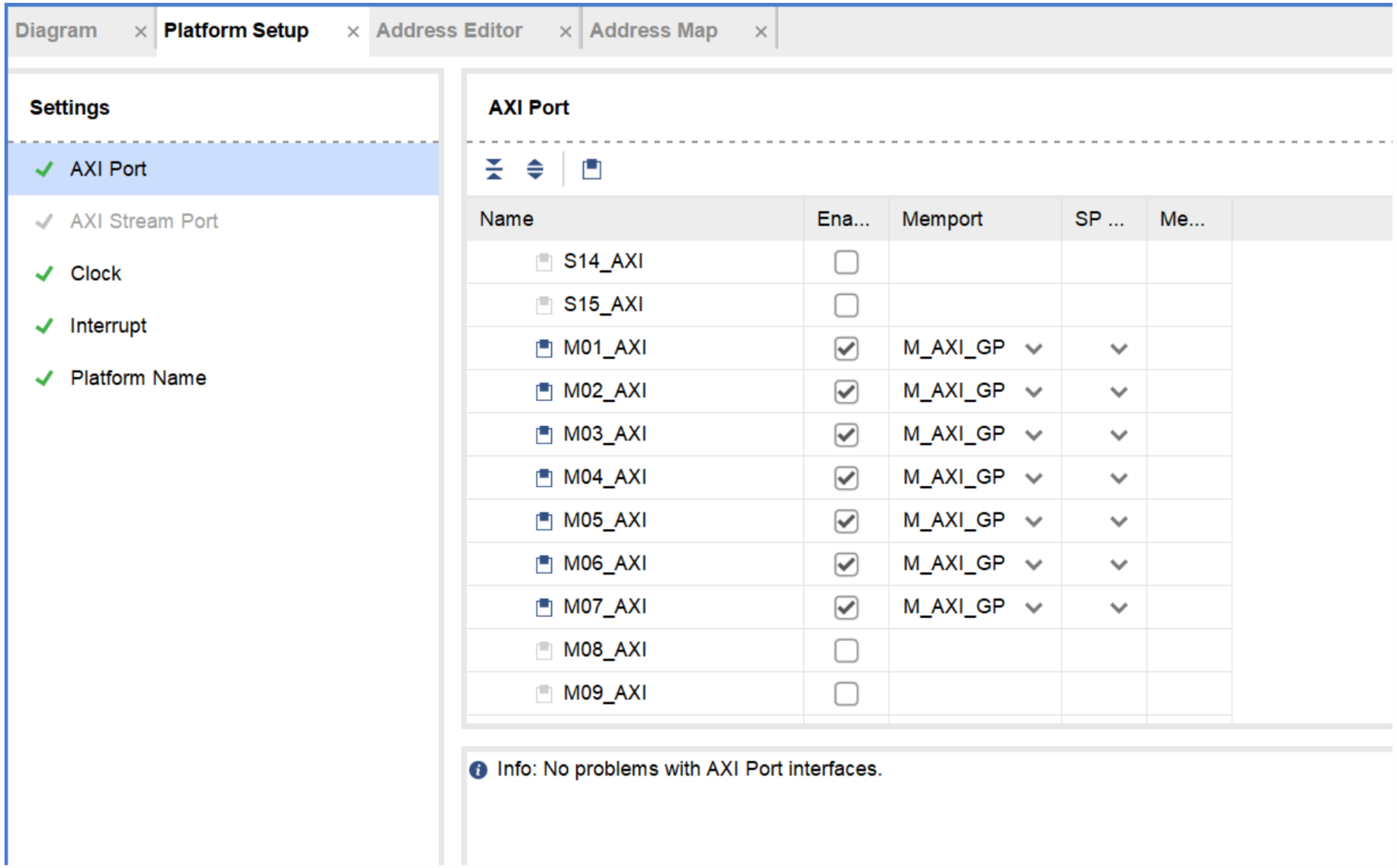


Diagram | Platform Setup | Address Editor | Address Map

Settings

- ✓ AXI Port
- ✓ AXI Stream Port
- ✓ Clock
- ✓ Interrupt
- ✓ Platform Name

AXI Port

Name	Ena...	Memport	SP ...	Me...
S14_AXI	☐			
S15_AXI	☐			
M01_AXI	✓	M_AXI_GP		
M02_AXI	✓	M_AXI_GP		
M03_AXI	✓	M_AXI_GP		
M04_AXI	✓	M_AXI_GP		
M05_AXI	✓	M_AXI_GP		
M06_AXI	✓	M_AXI_GP		
M07_AXI	✓	M_AXI_GP		
M08_AXI	☐			
M09_AXI	☐			

Info: No problems with AXI Port interfaces.

KV260 – Vivado part

- Xilinx KV260 github tutorial

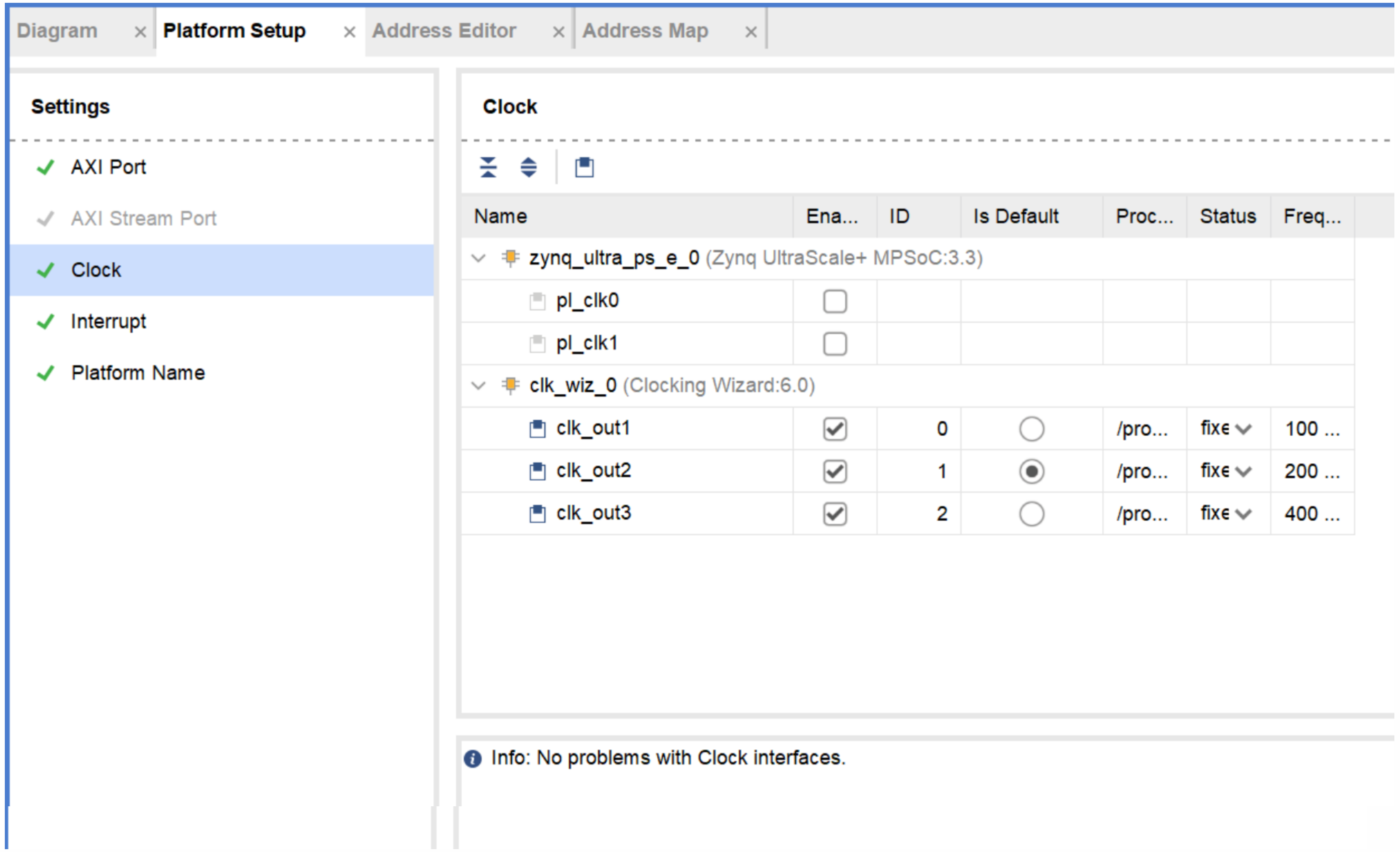


Diagram | Platform Setup | Address Editor | Address Map

Settings

- ✓ AXI Port
- ✓ AXI Stream Port
- ✓ Clock
- ✓ Interrupt
- ✓ Platform Name

Clock

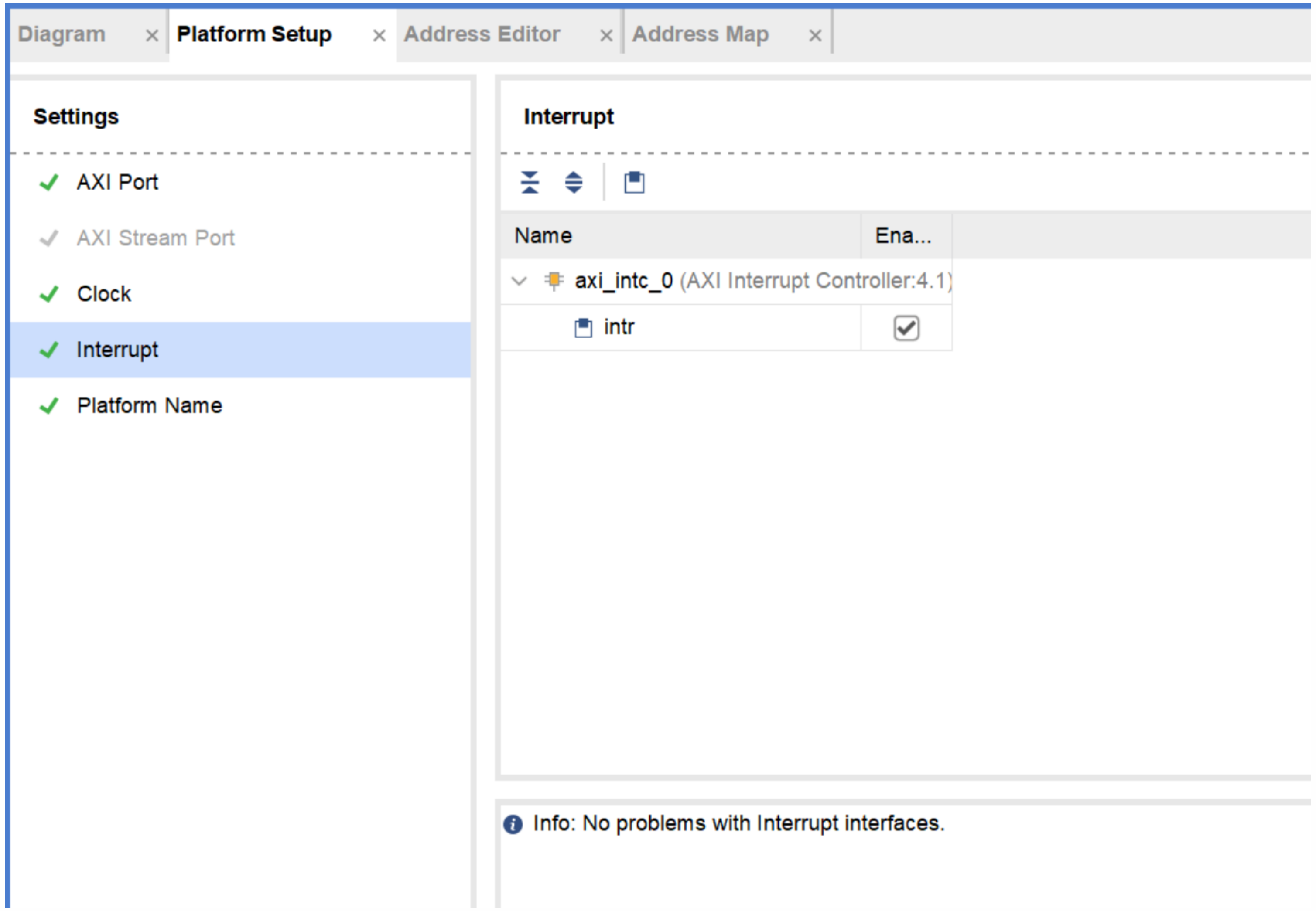
Name	Ena...	ID	Is Default	Proc...	Status	Freq...
zynq_ultra_ps_e_0 (Zynq UltraScale+ MPSoC:3.3)						
pl_clk0	☐					
pl_clk1	☐					
clk_wiz_0 (Clocking Wizard:6.0)						
clk_out1	✓	0	○	/pro...	fixe	100 ...
clk_out2	✓	1	●	/pro...	fixe	200 ...
clk_out3	✓	2	○	/pro...	fixe	400 ...

Info: No problems with Clock interfaces.

XILINX.

KV260 – Vivado part

- Xilinx KV260 github tutorial



Name	Ena...
axi_intc_0 (AXI Interrupt Controller:4.1)	
intr	✓

Info: No problems with Interrupt interfaces.

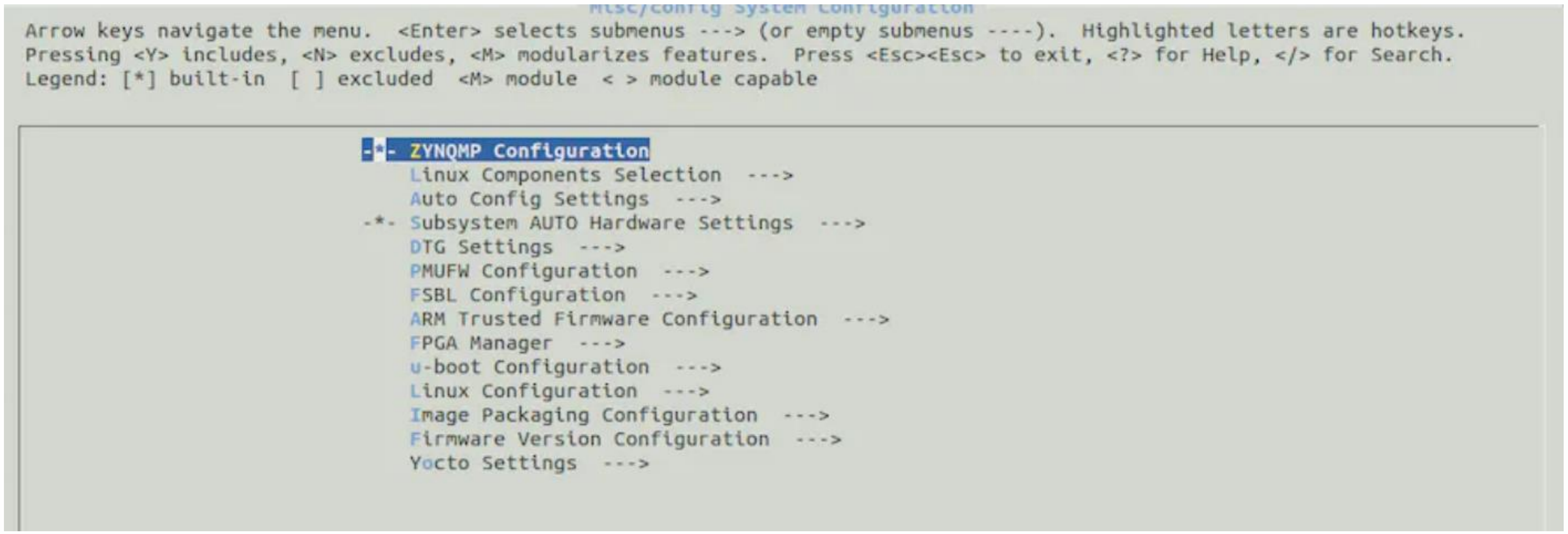
XILINX

KV260 – Vivado part

- Xilinx KV260 github tutorial
    1. Validation
    2. Create HDL Wrapper
    3. Generate Block Design
    4. Generate Bitstream
    5. Export XSA

KV260 – Petalinux part

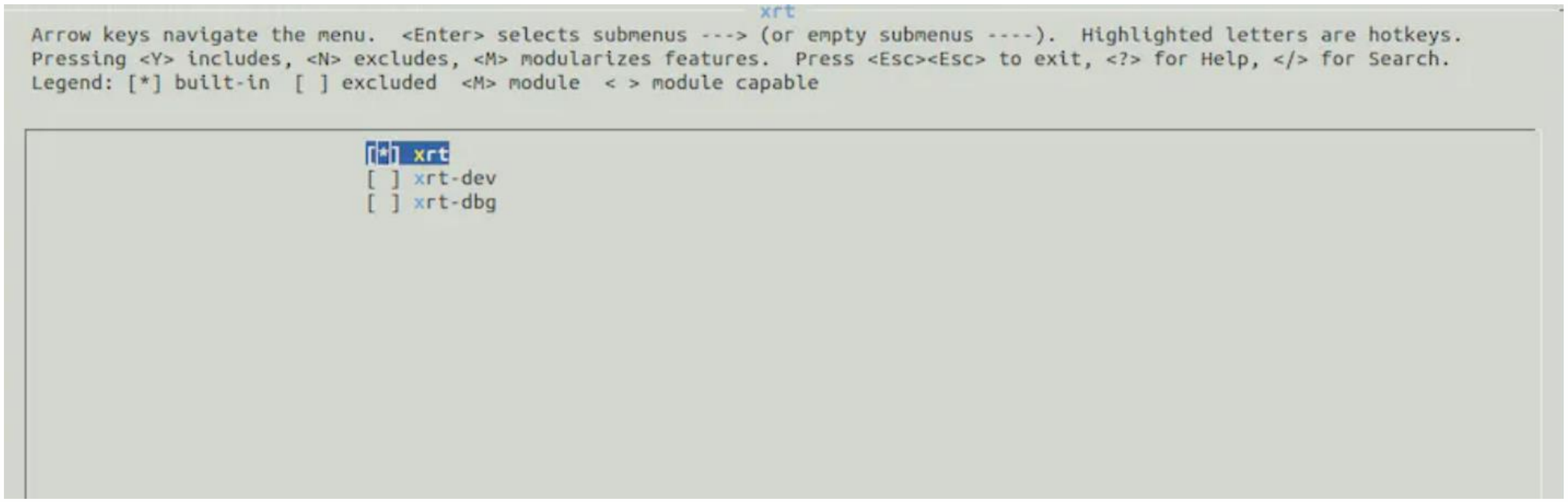
- Create Petalinux Project
  1. Petalinux 2021.2
  2. xilinx-k26-som-v2021.2-final.bsp
  3. petalinux-config --get-hw-description=/<path to xsa>



```
                         misc/config System Configuration
Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.
Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.
Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

                          -*- ZYNQMP Configuration
                              Linux Components Selection  --->
                              Auto Config Settings  --->
                          -*- Subsystem AUTO Hardware Settings  --->
                              DTG Settings  --->
                              PMUFW Configuration  --->
                              FSBL Configuration  --->
                              ARM Trusted Firmware Configuration  --->
                              FPGA Manager  --->
                              u-boot Configuration  --->
                              Linux Configuration  --->
                              Image Packaging Configuration  --->
                              Firmware Version Configuration  --->
                              Yocto Settings  --->
```

KV260 – Petalinux part

- Create Petalinux Project
  1. Petalinux 2021.2
  2. xilinx-k26-som-v2021.2-final.bsp
  3. petalinux-config --get-hw-description=/<path to xsa>
  4. petalinux-config -c rootfs



```
                                        xrt
 Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.
 Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.
 Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

                         [*] xrt
                         [ ] xrt-dev
                         [ ] xrt-dbg
```

XILINX.

KV260 – Petalinux part

- Create Petalinux Project
  1. Petalinux 2021.2
  2. xilinx-k26-som-v2021.2-final.bsp
  3. petalinux-config --get-hw-description=/<path to xsa>
  4. petalinux-config -c rootfs
  5. petalinux-build
  6. petalinux-build --sdk

KV260 – Petalinux part

- Device Tree Generator - convert .XSA into a device tree source file (.dtsi)

1. sudo apt install device-tree-compiler
2. git clone https://github.com/Xilinx/device-tree-xlnx
3. source /tools/Xilinx/Vitis/2021.2/settings64.sh
4. xsct (Xilinx Software Command-Line)

xsct

5. hsi open_hw_design <design_name.xsa>
6. hsi set_repo_path <path to device-tree-xlnx repository>
7. hsi create_sw_design device-tree -os device_tree -proc psu_cortexa53_0
8. hsi set_property CONFIG.dt_overlay true [hsi::get_os]
9. hsi set_property CONFIG.dt_zocl true [hsi::get_os]
10. hsi generate_target -dir <desired_dts_filename>
11. hsi close_hw_design [hsi current_hw_design]

XILINX.

KV260 – Petalinux part

- Compile Device Tree Source into Device Tree Blob
    1. source /tools/Xilinx/PetaLinux/2021.2/settings.sh
    2. dtc -@ -O dtb -o pl.dtbo pl.dtsi

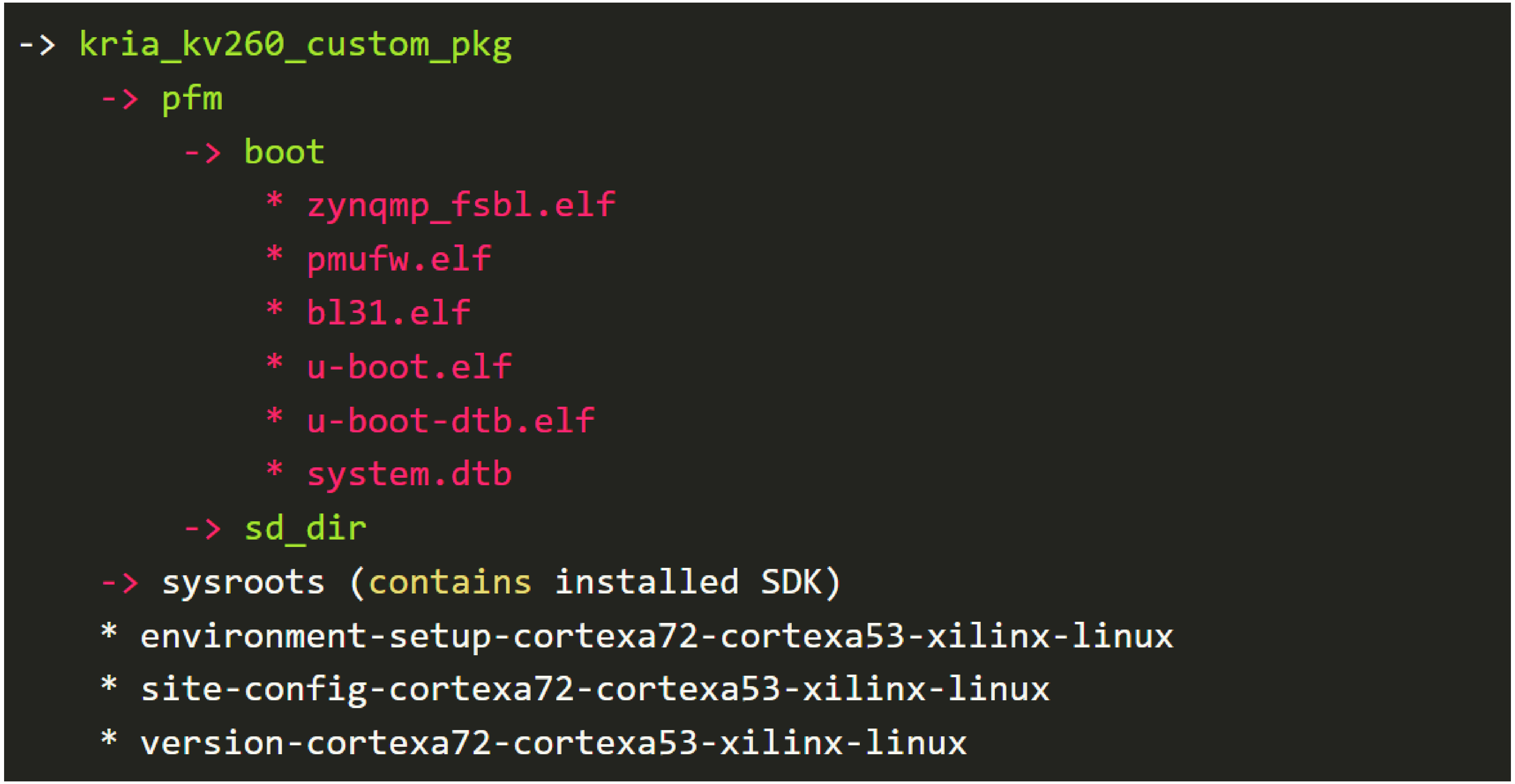
KV260 – Vitis part

- Prep Boot Components
  1. source /tools/Xilinx/PetaLinux/2021.2/settings.sh
  2. ./sdk.sh –d <your path>
  3. In <petalinux project path>/images/linux, you will find the following files:
     - ➢ zynqmp_fsbl.elf
     - ➢ pmufw.elf
     - ➢ bl31.elf
     - ➢ u-boot-dtb.elf
     - ➢ u-boot.elf
     - ➢ system.dtb

     copy these files to <your path>/pfm/boot

KV260 – Vitis part

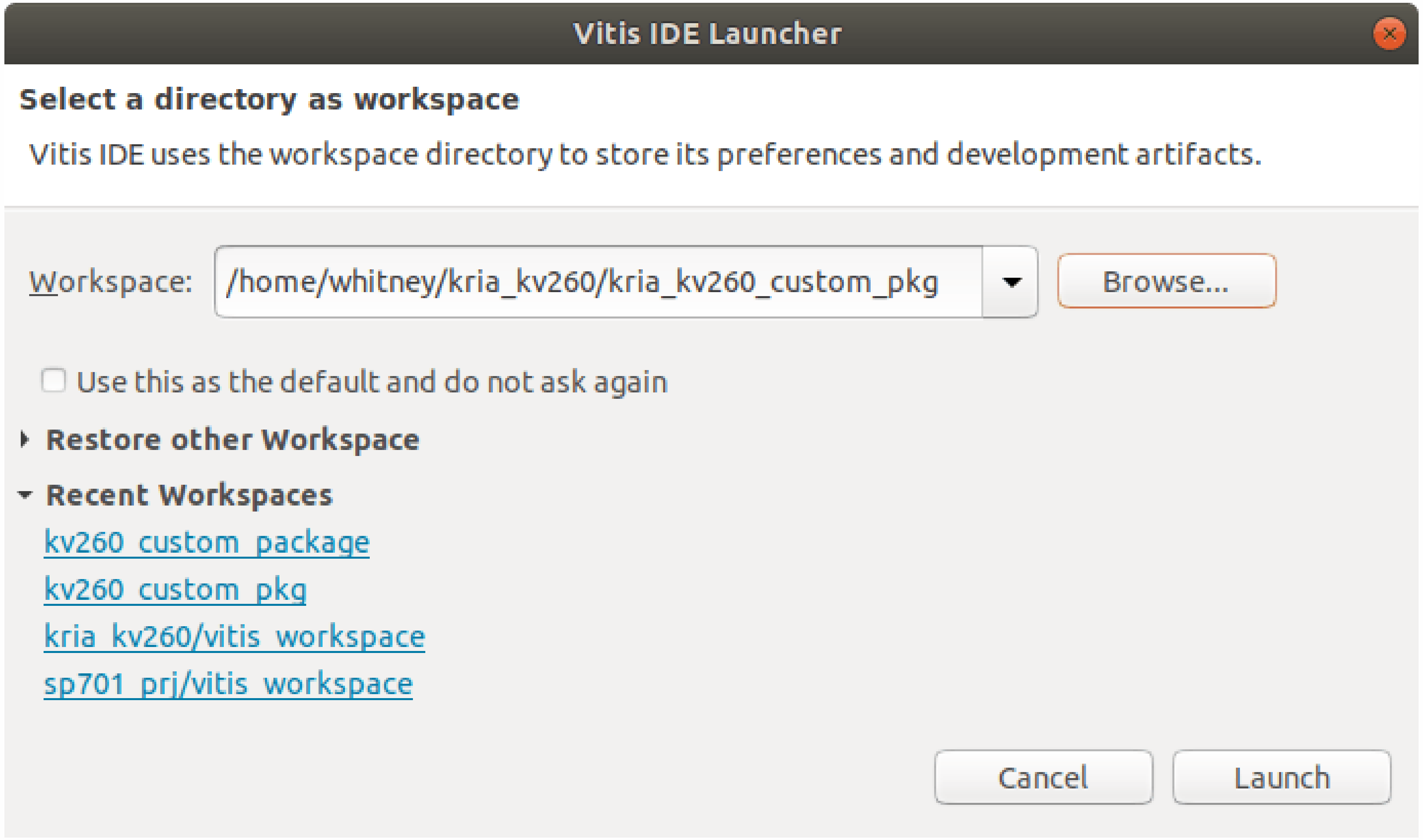
- Prep Boot Components



```
-> kria_kv260_custom_pkg
    -> pfm
        -> boot
            * zynqmp_fsbl.elf
            * pmufw.elf
            * bl31.elf
            * u-boot.elf
            * u-boot-dtb.elf
            * system.dtb
        -> sd_dir
    -> sysroots (contains installed SDK)
    * environment-setup-cortexa72-cortexa53-xilinx-linux
    * site-config-cortexa72-cortexa53-xilinx-linux
    * version-cortexa72-cortexa53-xilinx-linux
```

KV260 – Vitis part

- Create Vitis Platform



Vitis IDE Launcher

Select a directory as workspace

Vitis IDE uses the workspace directory to store its preferences and development artifacts.

Workspace: /home/whitney/kria_kv260/kria_kv260_custom_pkg

☐ Use this as the default and do not ask again

Restore other Workspace

Recent Workspaces

kv260_custom_package
kv260_custom_pkg
kria_kv260/vitis_workspace
sp701_prj/vitis_workspace