# ZCU102 Vitis DPU TRD (Vitis AI 3.0)-Vitis Flow

March 26, 2023

**Overview:**

This tutorial is expanded version of tutorial from <https://github.com/Xilinx/Vitis-AI/tree/master/dpu> , where you can get [DPUCZDX8G\_VAI\_v3.0.tar.gz/DPUCZDX8G\_VAI\_v3.0/prj/Vitis/README.md](https://www.xilinx.com/bin/public/openDownload?filename=DPUCZDX8G_VAI_v3.0.tar.gz). The necessary files for this tutorial are also provided at this link:

We are going to use Vitis AI 3.0 (DPU IP v4.1) , Vitis Tool 2022.2 , XRT 2022.2 in this tutorial.

## Base Platform and [ZYNQMP common image](https://www.xilinx.com/member/forms/download/xef.html?filename=xilinx-zynqmp-common-v2022.2_10141622.tar.gz) for ZCU102

In this step

1. Copy the ZCU102 base platform from the Vitis Installation directory, or you can clone the base bsp link and build it locally.

You can find the base platform for ZCU102 here in your Vitis installation directory:

Or we alos have provided the base platform on download link mentioned at top.

**Creating Base Platform yourself:** the base platform can also be created for custom board or ZCU102 by following here: Vitis Platform Creation ZCU104 or KV260 [[Link](https://github.com/Xilinx/Vitis-Tutorials/tree/2022.2/Vitis_Platform_Creation)]. For ZCU102 the steps mentioned for Platform creation for ZCU104 matches. For KV260 some of the final steps are different else KV260 also have most steps common with ZCU104.

1. Dowload the [ZYNQMP common image](https://www.xilinx.com/member/forms/download/xef.html?filename=xilinx-zynqmp-common-v2022.2_10141622.tar.gz) from Petalinux Web Page [[Link](https://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/embedded-platforms/2022-2.html)]. This common image can be used for any of MPSoC Boards, Xilinx Boards or other vendors board or custom carrier board. If you have follow above platform creation tutorial then you will create the “[ZYNQMP common image](https://www.xilinx.com/member/forms/download/xef.html?filename=xilinx-zynqmp-common-v2022.2_10141622.tar.gz)” file yourself which you can use on following steps.

## Building the Overlay

Here we are going to change the DPU Architecture to B512, the default Vitis-DPU-TRD archtecture if B4096 for ZCU102.

While changing the DPU architecture we will require to “compile the resnet50” in Vitis AI tool ourself using the arch.json generated by this step.

The steps going to follow here:

1. Sourcing Vitis, XRT
2. Exporting ZCU102 Base platform
3. Exporting [ZYNQMP common image](https://www.xilinx.com/member/forms/download/xef.html?filename=xilinx-zynqmp-common-v2022.2_10141622.tar.gz)
4. Running the make command, here we will make the DPU only (no SoftMax, for faster building overlay).

At the end of this Step, we will get the SD\_IMG for ZCU102, which we will burn on SD card using the Balena Etcher tool.

**Here is the Build LOG from our process:**

## Booting the ZCU102

First, burn the SD card with SD\_IMG using Balena Etcher.

The steps we are going to perform here are:

1. Zip the samples directory from **DPUCZDX8G\_VAI\_v3.0/app/samples**, this folder has multiple symbolic link. So we are going to copy samples.zip into “ZCU102’s BOOT partition”. So compress:

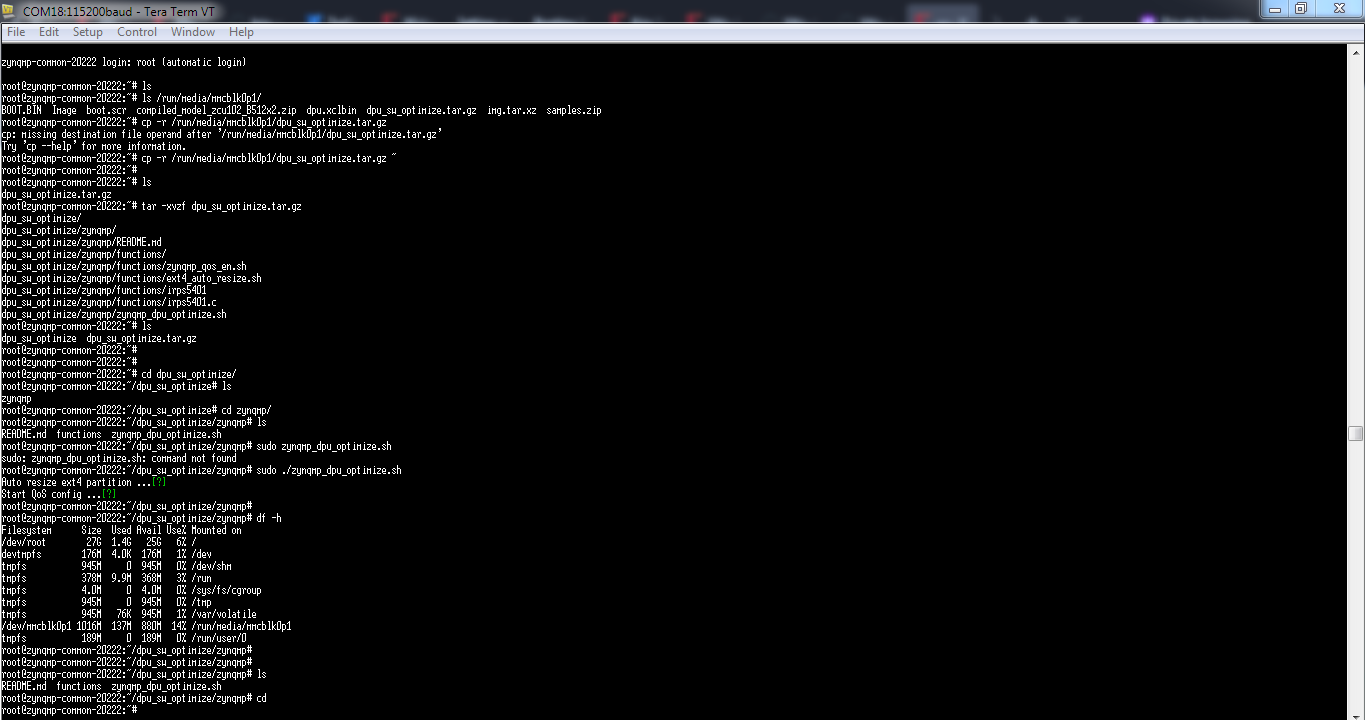
DPUCZDX8G\_VAI\_v3.0/app/samples as samples.zip or similar.

1. Also zip the models from **DPUCZDX8G\_VAI\_v3.0/app/models** and copy the zip into “BOOT partition” on ZCU102-SD-Image card.
2. Also copy “**dpu\_sw\_optimize.tar.gz**” to BOOT partition, this will be useful for us for “resizing the boot partition memory”.
3. Now setup the ZCU102 on SD Boot mode.
4. Plug the SD Card on ZCU102, setup power connection, UART connection. On PC, you can connect UART connection to terminal program like GTKterm, Teraterm, Putty or any. Setup the UART in 115200 baud rate, ZCU102 will give you 4 UART ports, you can connect Serial Port number 18.
5. Now power on the Board, you will start getting UART messages on your terminal.
6. Here you can copy the Zip or tar files we have at BOOT parition in to Home partition of the ZCU102. After copying you can extract those in home partition.

Example command (our BOOT parition is accessible at /run/media/mmcblk0p1):

cp -r /run/media/mmcblk0p1/dpu\_sw\_optimize.tar.gz

1. You can run the bash script inside “dpu\_sw\_optimize”, that script will resize our parition.



1. a

This boot is general boot which we have ran to see the boot flow and test the default “resnet50.xmodel”. Even though our DPU architecture is changed to two core B512, the default “resnet50.xmodel” based run must give us “fingerprint error” or “fingerprint mismatch” error.

**Test 1: run the example with default resnet50.xmodel**

**First error, Xmodel not found:**

This error is due to our model was not at the run directory. While you copy the resnet50.xmodel in home directory, you will not face this error.

root@zynqmp-common-20222:~# env LD\_LIBRARY\_PATH=samples/lib XLNX\_VART\_FIRMWARE=/run/media/mmcblk0p1/dpu.xclbin samples/bin/resnet50 img/bellpeppe-994958.JPEG

WARNING: Logging before InitGoogleLogging() is written to STDERR

F1119 17:23:20.634860 644 serialize\_v2.cpp:705] [UNILOG][FATAL][XIR\_READ\_PB\_FAILURE][failed to read pb file] file = resnet50.xmodel

\*\*\* Check failure stack trace: \*\*\*

Aborted

**Test 2 error, fingerprint mismatched:**

This is important error, as it tells us “dpu is working” but model’s fingerprint is not matched. And this error is expected error, cause we have changed the DPU architecture to B512, while default was B4096.

root@zynqmp-common-20222:~# cp /run/media/mmcblk0p1/app/model/resnet50.xmodel ~

root@zynqmp-common-20222:~# ls

dpu\_sw\_optimize dpu\_sw\_optimize.tar.gz img model resnet50.xmodel samples samples.zip

root@zynqmp-common-20222:~# env LD\_LIBRARY\_PATH=samples/lib XLNX\_VART\_FIRMWARE=/run/media/mmcblk0p1/dpu.xclbin samples/bin/resnet50 img/bellpeppe-994958.JPEG

WARNING: Logging before InitGoogleLogging() is written to STDERR

W1119 17:26:32.613843 659 dpu\_runner\_base\_imp.cpp:733] CHECK fingerprint fail! model\_fingerprint 0x101000056010407 is un-matched with actual dpu\_fingerprint 0x101000056010200. Please re-compile xmodel with dpu\_fingerprint 0x101000056010200 and try again.

F1119 17:26:32.614053 659 dpu\_runner\_base\_imp.cpp:695] fingerprint check failure.

\*\*\* Check failure stack trace: \*\*\*

Aborted

Now we have to go for “re-compiling” resnet50 model from Model Zoo in Vitis AI 3.0 GPU or CPU tool.

## Compiling the Resnet50 Model as of DPU Architecture

Now we are going to Vitis AI 3.0 for “re-compiling” the resnet50 model from Model zoo:

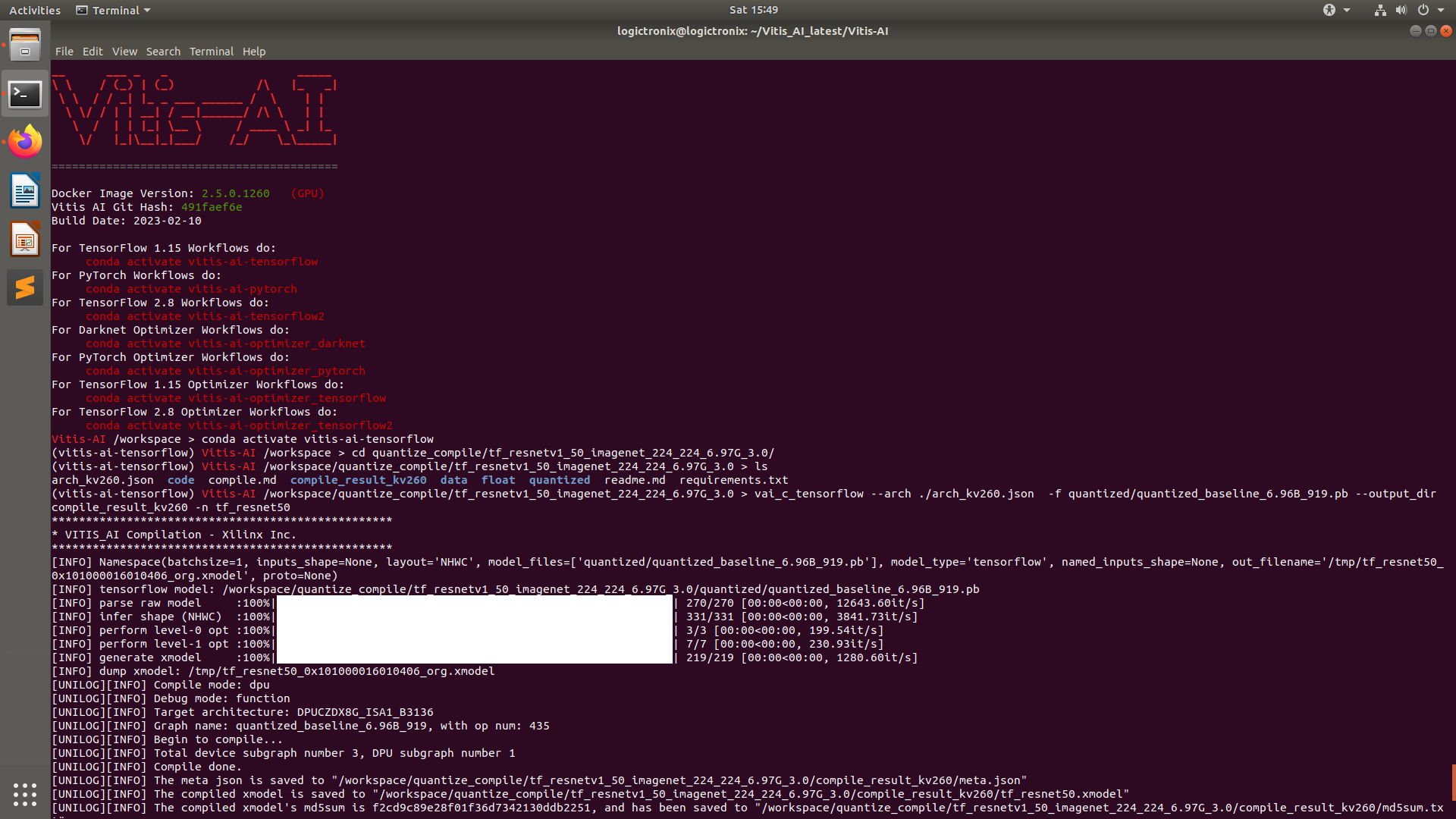
1. First download the resnet50 model from Model zoo, there are multiple resnet50 models in Model zoo. We have to find the matching version of resnet50 model based on the input hight x width of the image model takes.
   1. We have downloaded the tensorflow version of resnet50, there is also pytorch version of resnet50 models which can also be used. [Since Vitis AI 2.5](https://github.com/Xilinx/Vitis-AI/tree/2.5/model_zoo/model-list), caffe/darknet models are not supported so there is tensorflow(v1 and v2) and pytorch models only at Vitis AI 3.0 Model Zoo.
   2. Here is the model link, we have used: <https://github.com/Xilinx/Vitis-AI/tree/master/model_zoo/model-list/tf_resnetv1_50_imagenet_224_224_6.97G_3.0> , there are 3 different tf\_resnetv1\_50 models at model zoo, we have downlaoded un-pruned version while other two are pruned in different scale.
2. Now we have copied the our DPU fingerprint in arch.json, we can also copy the arch.json from the “binary\_container1” directory which is from “Step B”.

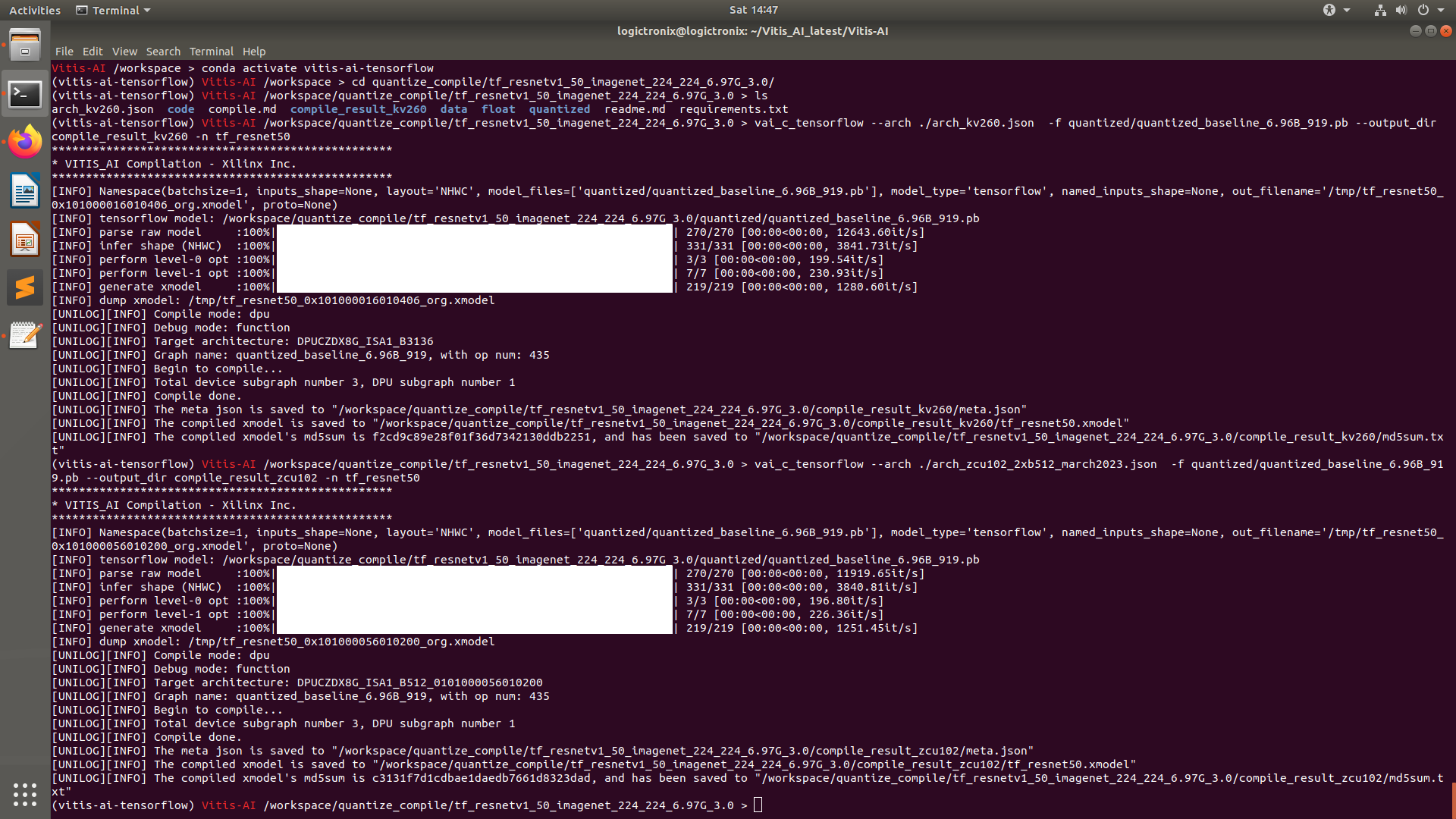
Our DPU Architecture’s fingerprint is (for B512 DPU architecture, Vitis AI 3.0):

{"fingerprint":"0x101000056010200"}

Note: we can also compile resnet50 and some generic model in older Vitis AI CPU or GPU version to, mainly we have to consider about “our DPU architecture on this tutorial is B512 and its new feature which are at Vitis AI 3.0 version or **DPU IP Version 4.1”**.

On our case Vitis AI 2.5 tool version can support our DPU architecture (fingerprint from Vitis AI 3.0 or DPU IP 4.1). For testing, Vitis AI 1.4 did not recognize the DPU fingerprint as few new layer and features are available on DPU IP after the Vitis AI 2.5. We already have installed Vitis AI 2.5 tool, so we are following it, else we will use Vitis AI 3.0 necessarily.





So ,we are compiling our resnet50 tensorflow model with Vitis AI 2.5 (which will be compatible to Vitis AI 3.0 compilation and supported by our SD\_Image).

The compilation command for our model compilation is:

vai\_c\_tensorflow --arch ./arch\_zcu102\_2xb512\_march2023.json -f quantized/quantized\_baseline\_6.96B\_919.pb --output\_dir compile\_result\_zcu102 -n tf\_resnet50

After compilation, we will get the compiled Xmodel, we can now rename the **tf\_resnet50.xmodel** as **resnet50.xmodel**.

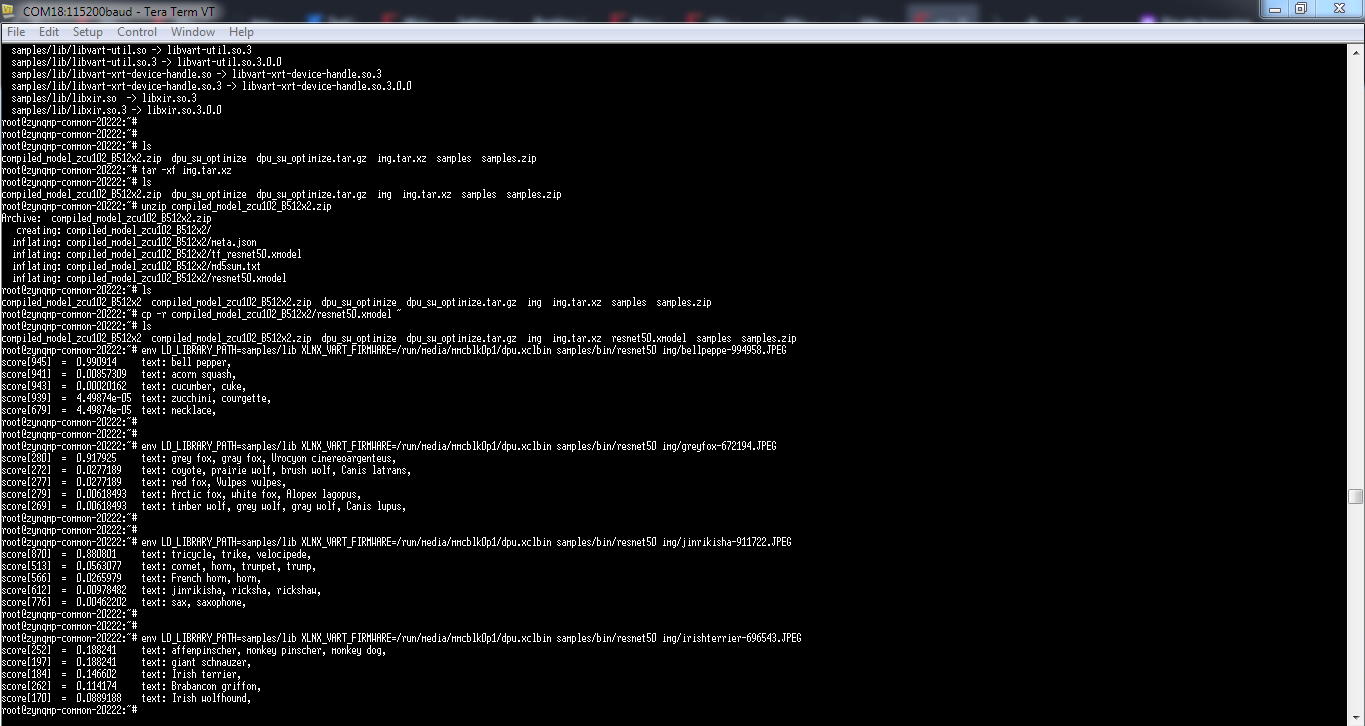
**Now lets again copy the resnet50.xmodel into ZCU102’s BOOT partition.**

## Running the Demo

As we already have bring up the ZCU102 in “Step C”. Now lets plug the SD card (with BOOT Partition including newly compiled resnet50.xmodel) in ZCU102.

Power on the board and now you can see the UART messages as in previous Step C. After boot completes, lets follow these steps:

1. First enter “ls” on current directory
2. Now copy the “resnet50.xmodel” from BOOT partition into the home directory
3. Now lunch the command for running the application, following UART log or screenshot shows the test of resnet50 with different images:



## Debug Points or Hints on Vitis DPU TRD with ZCU102 or Custom Board

For debugging,

1. Launching the Application after boot completes can hint the possible error like:
   1. DPU ERROR
   2. Fingerprint Error
   3. Xmodel error
2. Lsmod command

Lsmod can give the running or active nodes like Zocl, Mali, DMAproxy, UIO etc. This can help us for either Zocl available on the boot or not.

We have included the complete BOOT.LOG along with lsmod message-log in footer of this tutorial.

1. Dmesg command

Dmesg can give the detail Kernel level log, so if there is any error on hardware level or software level then we can get that in Dmesg LOG.

## Common issues from Xilinx Forum and Git-Issues