Application Note

ÚTIA Akademie věd České republiky
Ústav teorie informace a automatizace AV ČR, v.v.i.

All VART Examples from Xilinx Vitis AI 2.0 for Trenz Electronic board TE0808 SoM + TEBF0808 Carrier

Zdeněk Pohl, Lukáš Kohout, Jiří Kadlec
zdenek.pohl@utia.cas.cz, kohoutl@utia.cas.cz, kadlec@utia.cas.cz

Revision history

Rev.	Date	Author	Description
01	8.12.2022	Z.P.	document creation
02	30.12.2022	Z.P.	fixed error in petalinux configuration

Contents

Acknowledgement

This project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 101007321. The JU receives support from the European Union's Horizon 2020 research and innovation programme and France, Belgium, Czech Republic, Germany, Italy, Sweden, Switzerland, Turkey.

1 Description

This document provides tutorial how to setup and run all VART demos present in Vitis AI library 2.0 on Trenz TE0808 SoM attached to TEBF0808 carrier board.

2 Requirements

1. Hardware:
    a. Trenz TE0808 SoM installed on TEBF0808 and power source.
    b. Display Port Cable.
    c. Display port monitor with FHD support.
    d. USB webcam with USB cable, tested with See3CAM_CU30 - 3.4 Mpix Low Light USB Camera (Color).
    e. Ethernet UTP cable.
    f. 16GB SD card
2. Software:
    a. Finished “Test 3: Vitis-AI Demo” [1] example from TE0808 StarterKit Vitis AI Tutorial, i.e. it is possible to run dpu_trd (resnet50) demo.
    b. SD Card image created in “Test 3: Vitis-AI Demo” have to be re-generated as described in this application note.

3 How to Build VART Examples and Install Models

1. First the SD Card image used in “Test 3: Vitis-AI Demo” must be extended by additional gstreamer plugins as the VART demos are mostly running on video sources. Steps to add gstreamer plugins:
    a. Go back to place where petalinux was compiled in Trenz Vitis-AI Tutorial and add to ~/work/TE0808_24_240/StarterKit/os/petalinux/project-spec/meta-user/conf/user-rootfsconfig
    following line:

```
CONFIG_packagegroup-petalinux-gstreamer
```

    b. At the same folder open file “petalinuxbsp.conf” and add lines:

```
LICENSE_FLAGS_WHITELIST_append = “ commercial”
IMAGE_INSTALL_append = “ gstreamer1.0-plugins-ugly”
IMAGE_INSTALL_append = “ gstreamer1.0-libav”
```

    c. Setup petalinux environment.
    d. Run:

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

    e. Make sure that in “user packages” submenu is **packagegroup-petalinux-gstreamer** is checked.
    f. Rebuild petalinux:

```
petalinux-build
```

    g. Open Vitis workspace used to generate **dpu_trd** project.
    h. Build the **dpu_trd** project. New rootfs.ext4 file built in previous steps will be added to **sd_card.img**

i. Write **sd_card.img** file to 16GB SD Card.
j. Boot the board with new image and resize partition as learned in Trenz Vitis AI Tutorial.

2. Get scripts **init_VART.sh** and **VART_build_all.sh**
3. Edit **init_VART.sh** script and set correct paths to:
    a. Vitis AI github repository path (see Vitis AI Starterkit Tutorial, the path should be ~/vitis_ai_2_0):

```
VITIS_AI_DIR=~/vitis_ai_2_0/
```

    b. Path to installed platform SYSROOT (see Vitis AI Starterkit Tutorial, the path should be ~/work/te0808_24_240/StarterKit_pfm):

```
PLATFORM_SYSROOTS_DIR=~/work/te0808_24_240/StarterKit_pfm
```

4. Start downloading support files and building all examples:

```
./VART_build_all.sh all
```

5. Connect UTP and power cable to TE0808+TEBF0808. Power on the board.
6. Connect your PC to TE0808 using SFTP.
7. Copy 'VART' folder content to board using SFTP:

Copy all content of:

```
~/vitis_ai_2_0/demo/VART
```

to target board TE0808 folder:

```
/home/root
```

8. In PC open folder ~/vitis_ai_2_0/models/AI-Model-Zoo/ and use script to get all available precompiled models from Xilinx, call:

```
python3 downloader.py
```

when asked for input fill: "all", then enter "0" for all and enter "2" for zcu102 & zcu104 & kv260. Wait until all models in form of tar.gz archives are downloaded.

IMPORTANT: Vitis AI library 2.0 has an error in one of 'yaml' metafiles. Before download process is started it is needed to fix it:
In folder model-list/pt_pointpainting_nuscenes_2.0
Open 'model.yaml' file for editing and replace complete line:

```
download link: download link
```

With line

```
download link:
https://www.xilinx.com/bin/public/openDownload?filename=pointpainting_nuscenes_40000_64_0_pt-zcu102_zcu104_kv260-r2.0.0.tar.gz
```

9. Connect to target board TE0808 and create folder for models:

```
/usr/share/vitis_ai_library/models
```

10. Copy all downloaded *.tar.gz files to TE0808 board using SFTP to folder:

```
/usr/share/vitis_ai_library/models
```

11. Open ssh terminal to TE0808 board and continue on target board.
12. (Optional step) Set correct date and time:

```
date -s "2 OCT 2006 18:00:00"
hwclock --systohc
```

13. Go to `/usr/share/vitis_ai_library/models` and extract all:

```
cat *.tar.gz | tar xvzf - -i
```

14. (Optional step) Remove archives to save space on SD card:

```
rm *.tar.gz
```

15. Set environment variables:

```
export XLNX_VART_FIRMWARE=/mnt/sd-mmcblk1p1/dpu.xclbin
```

DISPLAY must be set only when X11 forwarding is NOT used:

```
export DISPLAY=:0.0
```

16. (recommended step) Test on one example - resnet50:
    a. Open readme file located in VART folder, and find command to execute resnet50 demo:

```
./resnet50 /usr/share/vitis_ai_library/models/resnet50/resnet50.xmodel
```

    b. Run example demo (using command found in previous step):

```
cd /home/root/VART/resnet50
./resnet50 /usr/share/vitis_ai_library/models/resnet50/resnet50.xmodel
```

See result in terminal.

4 Tested Demos

Commands needed to execute individual demos can be found in VART/README.md

Following sections will show example results of execution each demo.

4.1 Demo: adas_detection

Command:

./adas_detection video/adas.webm /usr/share/vitis_ai_library/models/yolov3_adas_pruned_0_9/yolov3_adas_pruned_0_9.xmodel

or

./adas_detection video/adas.avi /usr/share/vitis_ai_library/models/yolov3_adas_pruned_0_9/yolov3_adas_pruned_0_9.xmodel

Input:
AVI or webm video
Output:

Summary:
Adas detection running on provided small resolution dashboard cam video.

4.2 Demo: inception_v1_mt_py

Command:

python3 inception_v1.py 1 /usr/share/vitis_ai_library/models/inception_v1_tf/inception_v1_tf.xmodel

Input:

-

Output:

In terminal

Summary:

Inception implemented in python.

4.3 Demo: pose_detection

Command:

./pose_detection video/pose.webm /usr/share/vitis_ai_library/models/sp_net/sp_net.xmodel /usr/share/vitis_ai_library/models/ssd_pedestrian_pruned_0_97/ssd_pedestrian_pruned_0_97.xmodel

Or

./pose_detection video/pose.mp4 /usr/share/vitis_ai_library/models/sp_net/sp_net.xmodel /usr/share/vitis_ai_library/models/ssd_pedestrian_pruned_0_97/ssd_pedestrian_pruned_0_97.xmodel

Input:

mp4 or webm video file

Output:

Summary:

Demo shows pose detection of pre recoded fitness video. Persons in mirror are also detected.

4.4 Demo: resnet50

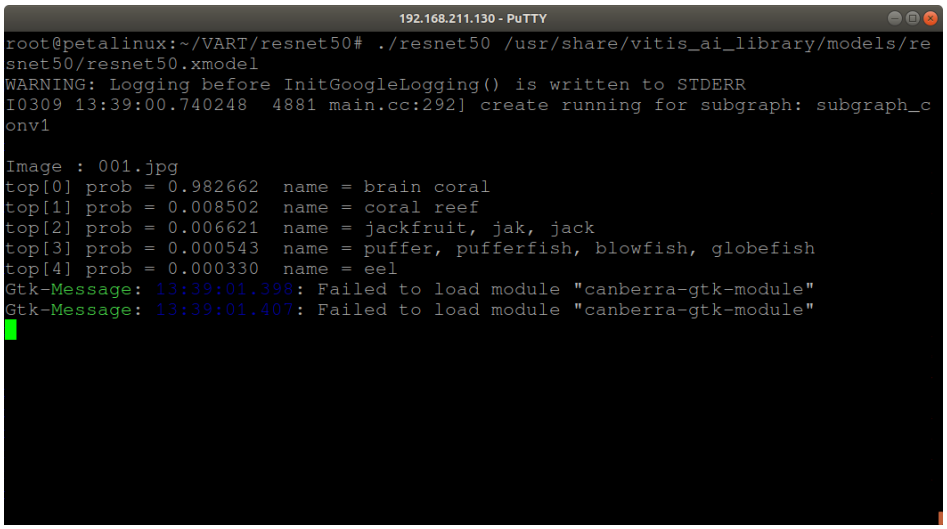
Command:

./resnet50 /usr/share/vitis_ai_library/models/resnet50/resnet50.xmodel

Input:

Automatically uses image: images/001.jpg

Output:



Summary:

Demo takes file images/001.jpg and does classification. Output can be found in terminal. Path to input file seems to be fixed (no path to image in command)

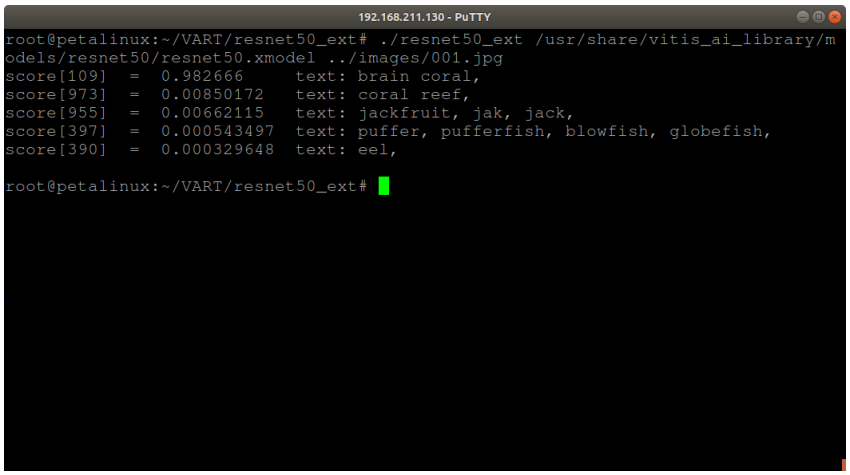
4.5 Demo: resnet50_ext

Command:

./resnet50_ext /usr/share/vitis_ai_library/models/resnet50/resnet50.xmodel ../images/001.jpg

Input:

Output:



Summary:

Like resnet50 but now there is possible to add input file(s) in command line.

4.6 Demo: resnet50_mt_py

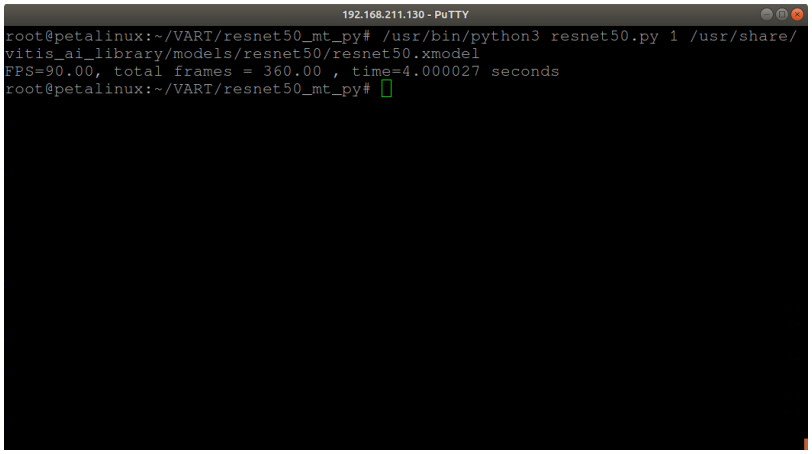
Command:

python3 resnet50.py 1 /usr/share/vitis_ai_library/models/resnet50/resnet50.xmodel

Input:

-

Output:



Summary:

resnet50 in python. It is not clear where is input file defined and where is output.

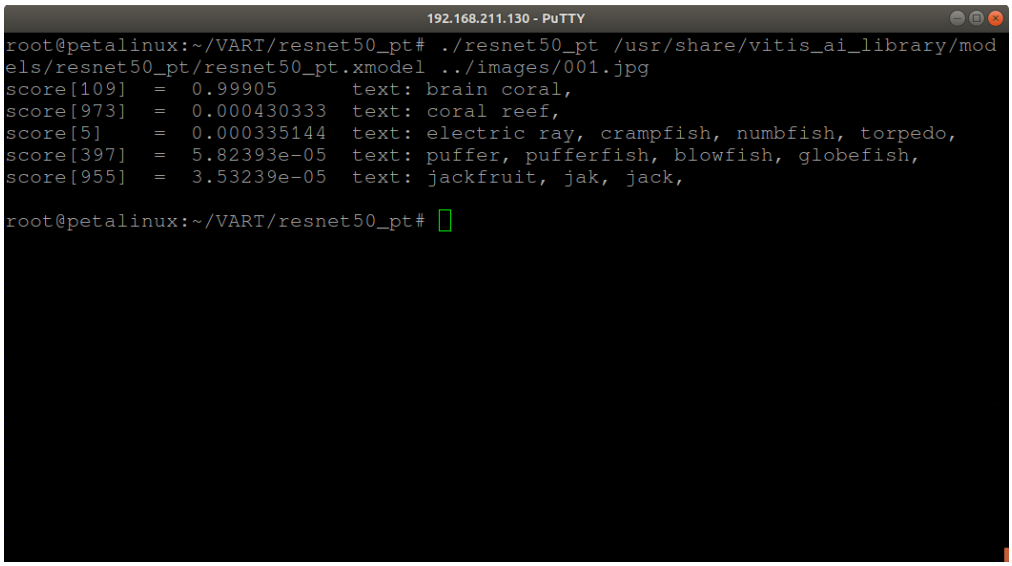
4.7 Demo: resnet50_pt

Command:

./resnet50_pt /usr/share/vitis_ai_library/models/resnet50_pt/resnet50_pt.xmodel ../images/001.jpg

Input:

Output:



Summary:

Another implementation of resnet50. It is not clear what exactly are differences between resnet50_* demos in VART.

4.8 Demo: segmentation

Command:

./segmentation video/traffic.webm /usr/share/vitis_ai_library/models/fpn/fpn.xmodel

Input:

mp4 or webm video

Output:

Summary:

ADAS segmentation on pre-recorded dashboard video.

4.9 Demo: squeezenet_pytorch

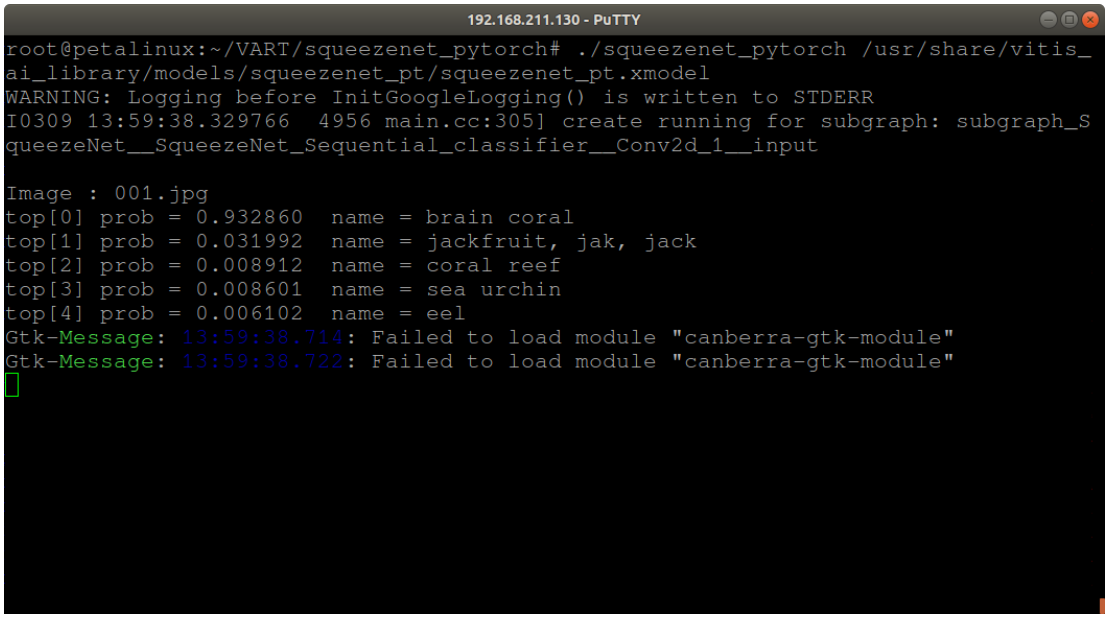
Command:

./squeezenet_pytorch /usr/share/vitis_ai_library/models/squeezenet_pt/squeezenet_pt.xmodel

Input:

Fixed input image

Output:



Summary:

Classification of image by squeezenet.

4.10 Demo: video_analysis

Command:

./video_analysis video/structure.webm /usr/share/vitis_ai_library/models/ssd_traffic_pruned_0_9/ssd_traffic_pruned_0_9.xmodel

Input:

webm or mp4 video

Output:

Summary:

Traffic surveillance analysis from pre-recorded video.

5 References

[1] TE0808 Starterkit Vitis AI Tutorial, Trenz Electronic Wiki: https://wiki.trenz-electronic.de/display/PD/TE0808+Starterkit+Vitis+AI+Tutorial