Application Note

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

# Xilinx Vitis AI *facedetect* and *resnet50* Demo on Trenz Electronic TE0802-02 with ZU2CG and 1 GB LPDD4

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

## Revision history

| Rev. | Date | Author | Description |
|---|---|---|---|
| 0 | 24.01.2023 | L.Kohout | Document creation |
| 1 | 09.02.2023 | L.Kohout | Fixed typos |
| 2 | 16.02.2023 | L.Kohout | HP ports mapping |
| | | | |

# Contents



# Acknowledgement

Acknowledgement to the StorAIge project and the corresponding Czech institutional support project No. 8A21009.

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 program and France, Belgium, Czech Republic, Germany, Italy, Sweden, Switzerland, Turkey.

# 1 Introduction

This document provides a tutorial describing setup and run Vitis AI 2.0 *facedetect* and *resnet50* demos on Trenz TE0802-02 board with ZU2CG and 1GB LPDD4 [1]. The system uses a Xilinx DPU unit to accelerate calculations. The DPU unit used in zcu104 or zcu102 does not fit into the used ZU2CG device. For this reason, a reduced DPU configuration is provided as well as precompiled models for demos. An example of the *facedetect* demo output can be seen in Figure 1. This text also describes how to move the system filesystem to the M.2 PCIe SSD.

# 2 Requirements

## 2.1 Hardware

- Trenz TE0802-02 board with ZU2CG and 1GB LPDD4 [1].
- USB webcam with USB cable, tested with ETERNICO ET201 Full HD webcam.
- M.2. PCIe SSD, tested device is Samsung SSD 970 EVO Plus 500GB.
- Micro USB cable.
- Ethernet cable.

## 2.2 Software

- Before following this guide, it is required to go through "TE0802 Vitis AI Tutorial https://wiki.trenz-electronic.de/display/PD/TE0802+test+board+Vitis+AI+Tutorial [2] from start with module *2* up to *Test 2: Run Vector Addition* section.

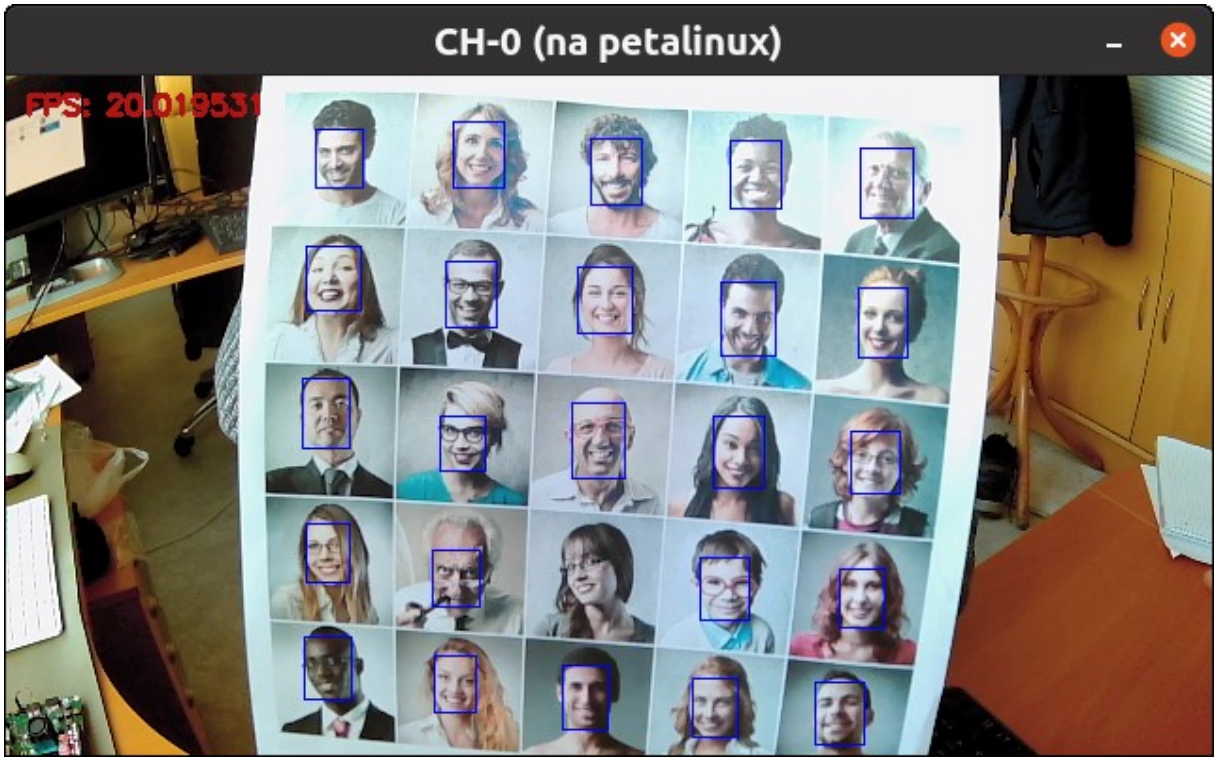



**Figure 1: Facedetect demo example output.**

# 3 Test 3: Vitis-AI demo

Starting point is *Test 2: Run Vector Addition* section of the TE0802 Vitis AI Tutorial: https://wiki.trenz-electronic.de/display/PD/TE0802+test+board+Vitis+AI+Tutorial [2].

1. Create new directory *test_board_dpu_trd*:

```
mkdir -p ~/work/TE0802_02_240/test_board_dpu_trd
cd ~/work/TE0802_02_240/test_board_dpu_trd
```

2. Start Vitis, in Ubuntu terminal execute

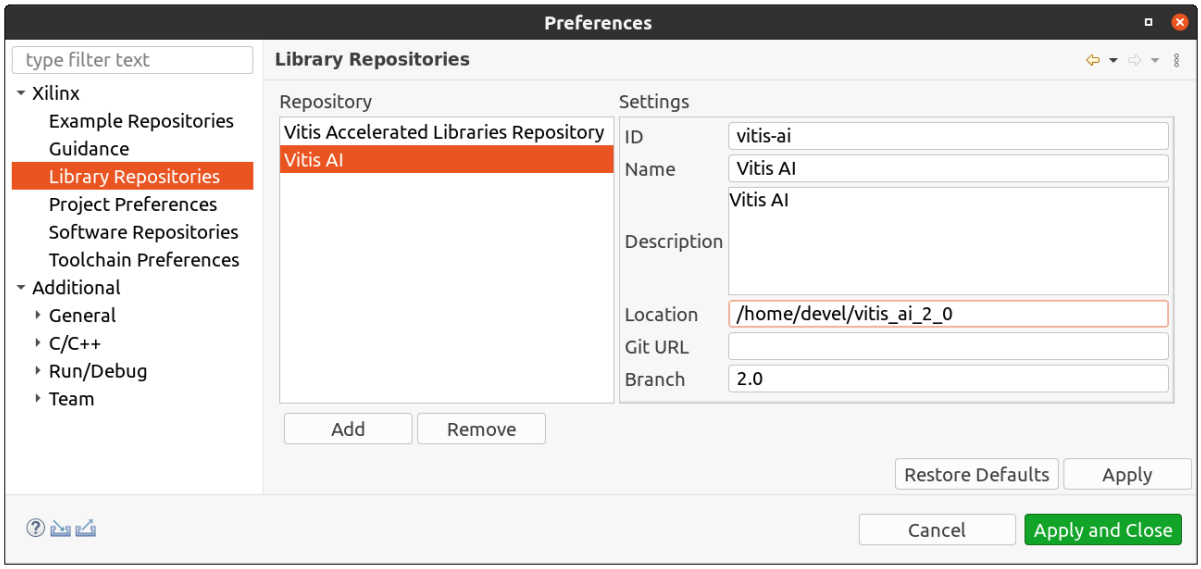
```
vitis &
```

In Vitis IDE Launcher, select your working directory

**~/work/TE0802_02_240/test_board_dpu_trd**

Click on Launch to start Vitis.

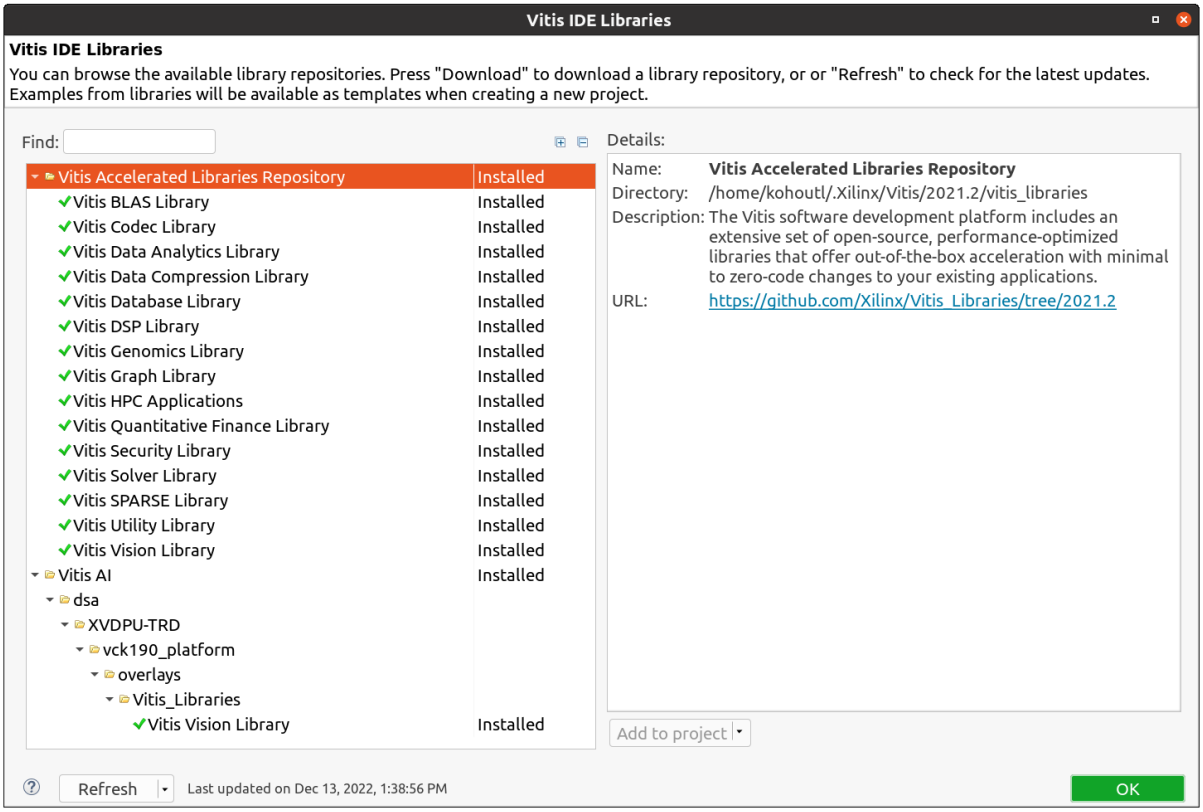
3. Add Vitis-AI Repository to Vitis:
   a. Open menu **Window→Preferences**.
   b. Go to **Library Repositories** tab.
   c. Add Vitis-AI by clicking **Add** button and fill the form as shown below, use absolute path to your home folder in field **Location**.




   d. Click **Apply and Close**

   *NOTE: Field "Location" says that the Vitis-AI repository from GITHUB has been already cloned into ~/vitis_ai_2_0 folder during the stage of the Petalinux configuration in previous tutorial. It is the same Vitis-AI 2.0 package downloaded from the branch 2.0. Use the absolute path to your home directory. It depends on the user name. The user name in the figure is "devel". Replace it by your user name.*

   e. Check that the library is correctly added, it appears in Libraries. Open menu **Xilinx→Libraries...**. You can see there just added Vitis-AI library marked as *Installed*.



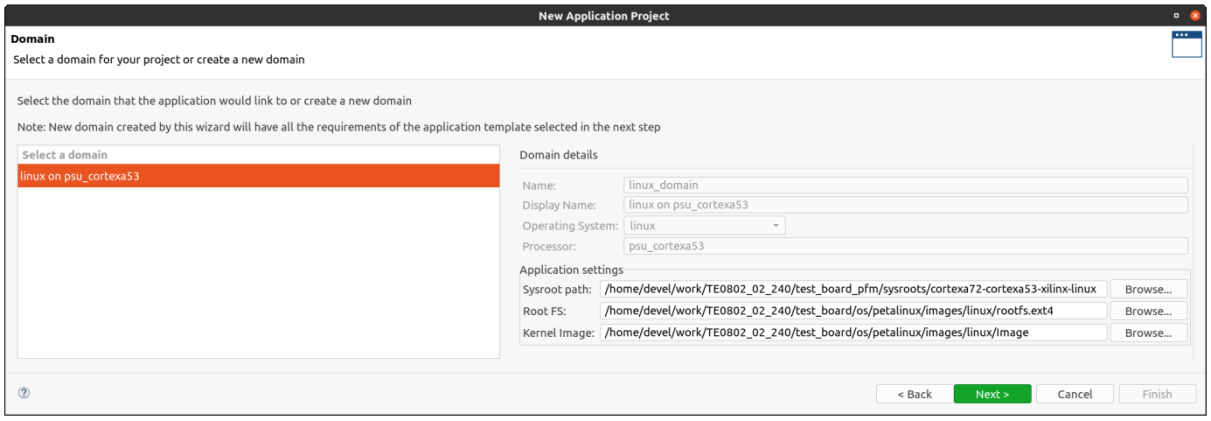

4. Create a Vitis-AI design for our TE0802_02_240 custom platform.
   a. Select **File→New→Application project**. Click **Next**. Skip welcome page if it is shown.
   b. Click on **+ Add** icon and select the custom extensible platform **TE0802_02_240_pfm[custom]** located in directory: *~/work/TE0802_02_240/test_board_pfm/TE0802_02_240_pfm/export/TE0802_02_240_pfm*.
   c. Click **Next**.
   d. In **Application Project Details** window type into **Application project name**:

      **dpu_trd**

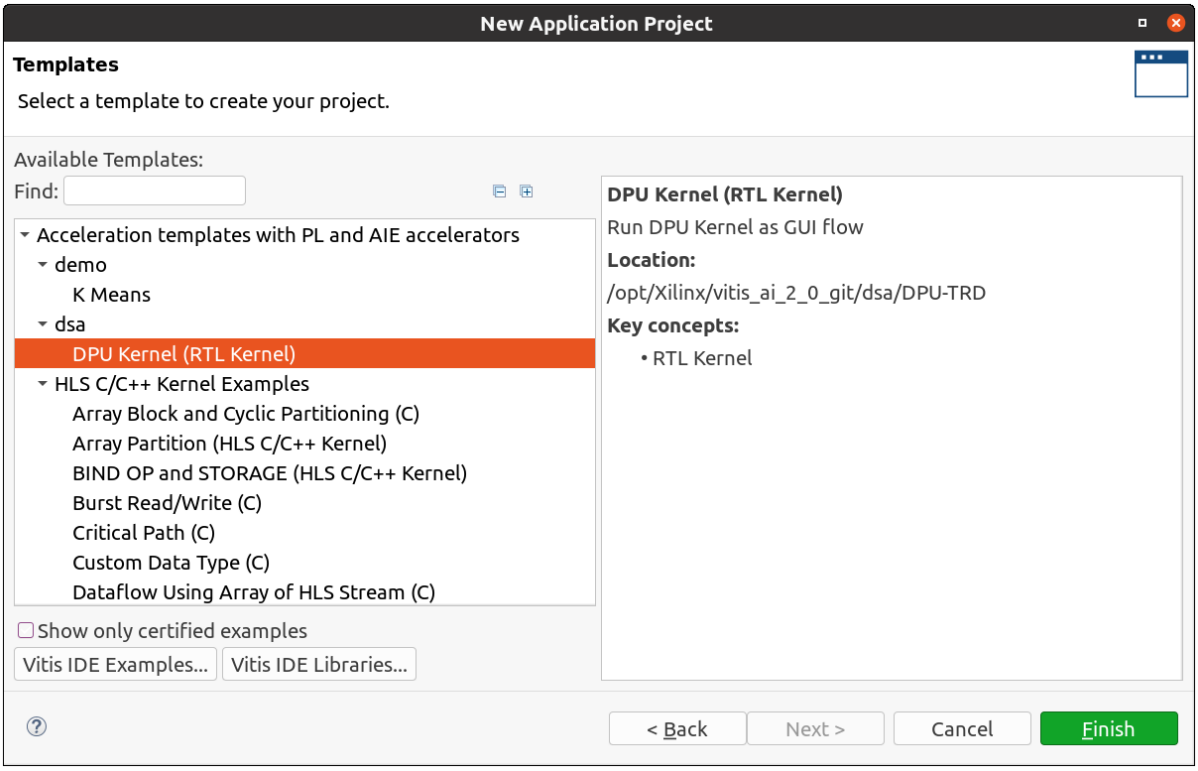
   e. Click **Next**.
   f. In **Domain window** type (or select by browse):

| Field | Value | Description |
|---|---|---|
| Sysroot path | ~/work/TE0802_02_240/test_board_pfm/sysroots/cortexa72-cortexa53-xilinx-linux | Petalinux compilation outputs from Trenz Vitis AI Tutorial |
| Root FS | ~/work/TE0802_02_240/test_board/os/petalinux/images/linux/rootfs.ext4 | |
| Kernel Image | ~/work/TE0802_02_240/test_board/os/petalinux/images/linux/Image | |

   g. Click **Next**.




h. In the **dsa** folder, select: **DPU Kernel (RTL Kernel)**.




i. Click **Finish**. New project template is created now.

5. Configure project and DPU:

   a. In dpu_trd window switch **Active build configuration** from Emulation-SW to **Hardware**.

      *NOTE: DPU configuration is stored in file dpu_conf.vh, it is located in dpu_trd_kernels/src/prj/Vitis directory*.

   b. Remove the **dpu_conf.vh** file and replace it by the one enclosed with this application note (ZIP file: dpu/dpu_conf.vh).

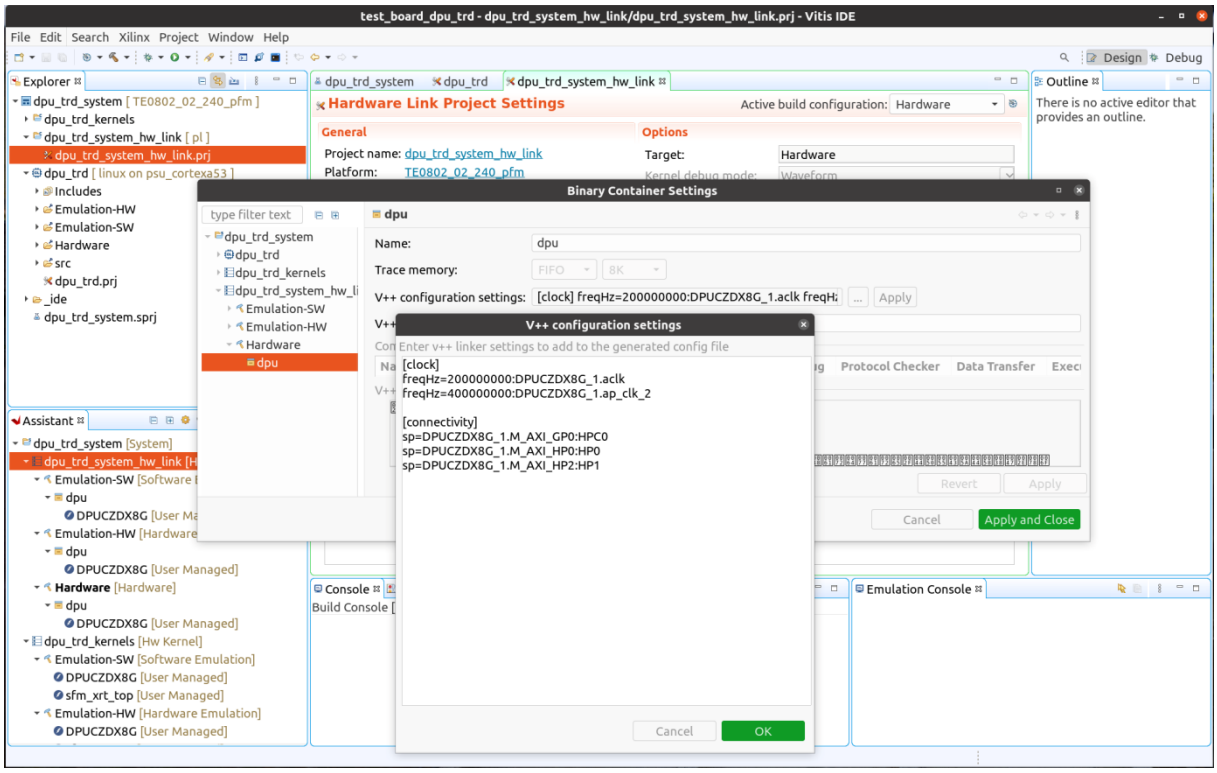
```
//Setting the arch of DPU, For more details, Please read the PG338

/*====== Architecture Options ======*/
// |-----------------------------------------------------|
// | Support 8 DPU size
// | It relates to model. if change, must update model
// +-----------------------------------------------------+
// | `define B512
// +-----------------------------------------------------+
// | `define B800
// +-----------------------------------------------------+
// | `define B1024
// +-----------------------------------------------------+
// | `define B1152
// +-----------------------------------------------------+
// | `define B1600
// +-----------------------------------------------------+
// | `define B2304
// +-----------------------------------------------------+
// | `define B3136
// +-----------------------------------------------------+
// | `define B4096
// |-----------------------------------------------------|

`define B1152

// |-----------------------------------------------------|
// | If the FPGA has Uram. You can define URAM_EN parameter
// | if change, Don't need update model
// +-----------------------------------------------------+
// | for zcu104 : `define URAM_ENABLE
// +-----------------------------------------------------+
// | for zcu102 : `define URAM_DISABLE
// |-----------------------------------------------------|

`define URAM_DISABLE

//config URAM
`ifdef URAM_ENABLE
    `define def_UBANK_IMG_N          5
    `define def_UBANK_WGT_N          17
    `define def_UBANK_BIAS           1
`elsif URAM_DISABLE
    `define def_UBANK_IMG_N          0
    `define def_UBANK_WGT_N          0
    `define def_UBANK_BIAS           0
`endif

// |-----------------------------------------------------|
// | You can use DRAM if FPGA has extra LUTs
// | if change, Don't need update model
// +-----------------------------------------------------+
// | Enable DRAM  : `define DRAM_ENABLE
// +-----------------------------------------------------+
// | Disable DRAM : `define DRAM_DISABLE
// |-----------------------------------------------------|

`define DRAM_DISABLE
```

```
//config DRAM
`ifdef DRAM_ENABLE
    `define def_DBANK_IMG_N          1
    `define def DBANK WGT N          1
    `define def_DBANK_BIAS           1
`elsif DRAM_DISABLE
    `define def_DBANK_IMG_N          0
    `define def_DBANK_WGT_N          0
    `define def_DBANK_BIAS           0
`endif

// |-----------------------------------------------------|
// | RAM Usage Configuration
// | It relates to model. if change, must update model
// +-----------------------------------------------------+
// | RAM Usage High : `define RAM_USAGE_HIGH
// +-----------------------------------------------------+
// | RAM Usage Low  : `define RAM_USAGE_LOW
// |-----------------------------------------------------|

`define RAM_USAGE_LOW

// |-----------------------------------------------------|
// | Channel Augmentation Configuration
// | It relates to model. if change, must update model
// +-----------------------------------------------------+
// | Enable  : `define CHANNEL_AUGMENTATION_ENABLE
// +-----------------------------------------------------+
// | Disable : `define CHANNEL_AUGMENTATION_DISABLE
// |-----------------------------------------------------|

`define CHANNEL_AUGMENTATION_ENABLE

// |-----------------------------------------------------|
// | DepthWiseConv Configuration
// | It relates to model. if change, must update model
// +-----------------------------------------------------+
// | Enable  : `define DWCV_ENABLE
// +-----------------------------------------------------+
// | Disable : `define DWCV_DISABLE
// |-----------------------------------------------------|

`define DWCV_ENABLE

// |-----------------------------------------------------|
// | Pool Average Configuration
// | It relates to model. if change, must update model
// +-----------------------------------------------------+
// | Enable  : `define POOL_AVG_ENABLE
// +-----------------------------------------------------+
// | Disable : `define POOL_AVG_DISABLE
// |-----------------------------------------------------|

`define POOL_AVG_ENABLE

// |-----------------------------------------------------|
// | support multiplication of two feature maps
// | It relates to model. if change, must update model
// +-----------------------------------------------------+
```

```
// | Enable  : `define ELEW_MULT_ENABLE
// +-----------------------------------------------------+
// | Disable : `define ELEW_MULT_DISABLE
// |-----------------------------------------------------|

`define ELEW_MULT_ENABLE

// +-----------------------------------------------------+
// | RELU Type Configuration
// | It relates to model. if change, must update model
// +-----------------------------------------------------+
// | `define RELU_RELU6
// +-----------------------------------------------------+
// | `define RELU_LEAKYRELU_RELU6
// |-----------------------------------------------------|

`define RELU_LEAKYRELU_RELU6

// |-----------------------------------------------------|
// | DSP48 Usage Configuration
// | Use dsp replace of lut in conv operate
// | if change, Don't need update model
// +-----------------------------------------------------+
// | `define DSP48_USAGE_HIGH
// +-----------------------------------------------------+
// | `define DSP48_USAGE_LOW
// |-----------------------------------------------------|

`define DSP48_USAGE_HIGH

// |-----------------------------------------------------|
// | Power Configuration
// | if change, Don't need update model
// +-----------------------------------------------------+
// | `define LOWPOWER_ENABLE
// +-----------------------------------------------------+
// | `define LOWPOWER_DISABLE
// |-----------------------------------------------------|

`define LOWPOWER_DISABLE

// |-----------------------------------------------------|
// | DEVICE Configuration
// | if change, Don't need update model
// +-----------------------------------------------------+
// | `define MPSOC
// +-----------------------------------------------------+
// | `define ZYNQ7000
// |-----------------------------------------------------|

`define MPSOC
```

This modification is necessary for successful implementation of the DPU on the used module.

c. Go to **dpu_trd_system_hw_link** and open **dpu_trd_system_hw_link.prj** file. Remove **sfm_xrt_top** kernel from binary container by right clicking on it and choosing remove. **Reduce number of DPU kernels to one**.




d. Configure connection of DPU kernels. On the same tab right click on **dpu** and choose **Edit V++ Options**.

Click "**...**" button on the line of V++ Configuration Settings and modify configuration as follows:

```
[clock]
freqHz=200000000:DPUCZDX8G_1.aclk
freqHz=400000000:DPUCZDX8G_1.ap_clk_2

[connectivity]
sp=DPUCZDX8G_1.M_AXI_GP0:HPC0
sp=DPUCZDX8G_1.M_AXI_HP0:HP1
sp=DPUCZDX8G_1.M_AXI_HP2:HP2
```

*NOTE: The platform provides HP0, HP1, HP2 and HP3 ports to be used. As the HP0 port is shared with the display port of the board, it is recommended to use ports HP1 and HP2. It reduces the risk of image flickering on the DP monitor.*

6. Build DPU_TRD application

   a. In *Explorer* section of Vitis IDE, click on **dpu_trd_system[TE0802_02_240_pfm]** to select it.

   b. Right Click on **dpu_trd_system[TE0802_02_240_pfm]** and select in the opened sub-menu **Build project**.

7. Run DPU_TRD on the board

   a. Write **sd_card.img** to SD card using SD card reader. The image file is an output product of the Vitis compilation and packaging phase. It is located in directory:

   **~/work/TE0802_02_240/test_board_dpu_trd/dpu_trd_system/Hardware/package/**

*NOTE: For writing the image file to the SD card can be used BalenaEtcher tool, downloadable from https://www.balena.io/etcher. The tool is available for Windows, Linux or MAC platforms.*

b. Boot the board and open a terminal connection, USB UART or SSH.

- UART connection via USB UART/JTAG (connector J8 on the board). On your host machine identify the device to be connected (COM port on Windows or ttyUSB device on Linux). On Windows machine it can be any port. On Linux machine are USB serial ports counted from zero. As the board provides two virtual devices and the second one is the UART, the device will be **/dev/ttyUSB1** (in case that only one board is connected). To connect the board use PUTTY, for instance. The settings are:
    - Baud rate – 115200
    - Data bits – 8
    - Stop bits – 1
    - Parity – none
    - Flow control – none

  *NOTE: If you want to forward graphical applications started from this terminal to the board monitor connected via display port, you should also set DISPLAY variable correctly (export DISPLAY=:0.0).*

- SSH connection via Ethernet (preffered). To find out the board IP address use UART connection, execute command:

```
ifconfig
```

  Or from Linux host machine, you can use *arp-scan* command. Example:

```
sudo arp-scan -l -I enp4s0
Interface: enp4s0, type: EN10MB, MAC: c4:6e:1f:01:b9:0d, IPv4:
10.42.0.1
Starting arp-scan 1.9.7 with 256 hosts
(https://github.com/royhills/arp-scan)
10.42.0.102       80:1f:12:d0:7d:0a   Microchip Technology Inc.

1 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9.7: 256 hosts scanned in 1.997 seconds (128.19
hosts/sec). 1 responded
```

  *NOTE: The SSH connection should have X11 forwarding activated, if you want to forward graphical applications to your host machine. You should also set DISPLAY variable correctly (export DISPLAY=:10.0).*

c. Resize the second partition of the SD card to its maximal size, from the board terminal use command *resize-part*:

```
resize-part /dev/mmcblk0p2
/dev/mmcblk0p2
Warning: Partition /dev/mmcblk0p2 is being used. Are you sure you want to
continue?
Yes/No? yes
End?  [4295MB]? 100%
Information: You may need to update /etc/fstab.

resize2fs 1.45.6 (20-Mar-2020)
Filesystem at /dev/mmcblk0p2 is mounted on /media/sd-mmcblk0p2; on-line
resizing required
old_desc_blocks = 1, new_desc_blocks = 2
The filesystem on /dev/mmcblk0p2 is now 3539968 (4k) blocks long.
```

d. Switch from board terminal back to your host PC.

e. Copy demo source files from the PC to the home folder on the board, connect the board using SFTP and copy to **/home/root** on the board two folders:

**~/vitis_ai_2_0/demo/VART/resnet50** to **/home/root**
**~/vitis_ai_2_0/demo/VART/common** to **/home/root**

f. Keep SFTP connection active and copy to the board /home/root folder:

**~/vitis_ai_2_0/demo/Vitis-AI-Library/samples/facedetect** to **/home/root**

g. Unpack the ZIP archive provided with this application note and copy image and model files from folder **dpu/B1152_2CG** to the home folder of the board using SFTP. Copy following folders to **/home/root** (overwrite existing folders from previous step):

**dpu/B1152_2CG/facedetect**
**dpu/B1152_2CG/images** all to **/home/root**
**dpu/B1152_2CG/resnet50**

*NOTE: Pre-compiled models for custom DPU are provided in the ZIP archive enclosed with this application note. Use **Appendix A** if you want to compile models for used DPU configuration again.*

h. Copy scripts optimizing the data transfers between the DPU and the memory controller from the PC to the home folder on the board. These scripts are available in attached ZIP file. To copy them use SFTP:

**dpu_sw_optimize** to **/home/root**

i. Switch back to the board terminal.

j. Optimize the data transfers between the DPU and the memory controller, execute:

```
cd /home/root/dpu_sw_optimize/zynqmp
chmod -R +x *
./zynqmp_dpu_optimize.sh
```

*NOTE: The scripts are modified compare to scripts provided by Xilinx.*

k. Build demo executables. For both, **/home/root/resnet50** and **/home/root/facedetect**, folders execute:

```
chmod +x build.sh
./build.sh
```

l. Set needed environment variable pointing the DPU firmware.

```
export XLNX_VART_FIRMWARE=/mnt/sd-mmcblk0p1/dpu.xclbin
```

m. Run **resnet50** demo

```
cd /home/root/resnet50
./resnet50 resnet50.xmodel
```

The demo will use image located in the *../images* folder and the result will be shown in the terminal. The input image will be shown on your host machine screen or on the board monitor depending on the *$DISPLAY* variable.

n. Run **facedetect** demo

```
cd /home/root/facedetect
./test_performance_facedetect densebox_640_360.xmodel \
                              test_performance_facedetect.list
```

The files listed in .list file will be used to evaluate DPU performance for *densebox_640_360* model. The performance of the DPU unit in this configuration is **79 FPS**.

To run *facedetect* demo using USB webcam first connect USB webcam and then start the demo:

```
./test_video_facedetect densebox_640_360.xmodel 0 -t 1
```

The result can be seen on your host machine screen or on the board monitor depending on the *$DISPLAY* variable. The performance of the system is **20 FPS** for both, result displayed on the monitor connected via display port and result forwarded output via SSH.

*NOTE: The performance of the system depends on used USB camera. To be more precise, it depends on the size of the input image from the camera that has to be resized to the DPU facedetect model requirement (640x360). This resizing is done by ARM processor and this workload of the ARM processor affects performance.*

# 4 Filesystem on M.2 PCIe SSD

TE0802 is equipped with a M.2 PCIe connector (U5) providing a SSD connection. To migrate the file system from the SD card to SSD, follow steps bellow.

1. Create a primary partition on the SSD, use the board terminal.

   a. Get info about all storages available to the board.

```
parted -l
  Model: Samsung SSD 970 EVO Plus 500GB (nvme)
  Disk /dev/nvme0n1: 500GB
  Sector size (logical/physical): 512B/512B
  Partition Table: loop
  Disk Flags:
  Number  Start  End    Size   File system  Flags
  1      0.00B  500GB  500GB  ext4

  Model: SD SD16G (sd/mmc)
  Disk /dev/mmcblk0: 15.5GB
  Sector size (logical/physical): 512B/512B
  Partition Table: msdos
  Disk Flags:
  Number  Start   End     Size    Type     File system  Flags
  1      1049kB  1024MB  1023MB  primary  fat32        boot
  2      1024MB  4295MB  3271MB  primary  ext4
```

   b. Find a mount point of the SSD. The drive is automatically mounted to the system.

```
mount
  /dev/mmcblk0p2 on / type ext4 (rw,relatime)
  devtmpfs on /dev type devtmpfs
  (rw,relatime,size=240512k,nr_inodes=60128,mode=755)
  proc on /proc type proc (rw,relatime)
  sysfs on /sys type sysfs (rw,relatime)
  debugfs on /sys/kernel/debug type debugfs (rw,relatime)
  configfs on /sys/kernel/config type configfs (rw,relatime)
  tmpfs on /run type tmpfs (rw,nosuid,nodev,mode=755)
  tmpfs on /var/volatile type tmpfs (rw,relatime)
  /dev/nvme0n1 on /run/media/nvme0n1 type ext4 (rw,relatime)
```

```
/dev/mmcblk0p2 on /media/sd-mmcblk0p2 type ext4 (rw,relatime)
/dev/mmcblk0p1 on /media/sd-mmcblk0p1 type vfat (rw,relatime,gid=100,
fmask=0002,dmask=0002,allow_utime=0020,codepage=437,iocharset=iso8859-1,
shortname=mixed,utf8,errors=remount-ro)
devpts on /dev/pts type devpts (rw,relatime,gid=5,mode=620,ptmxmode=000)
```

c. Unmount the drive from the file system to allow it to be configured.

```
umount /run/media/nvme0n1/
```

d. Format the whole disk space with the EXT4 file system. This step delete all files on the drive.

```
mkfs.ext4 /dev/nvme0n1
  mke2fs 1.45.6 (20-Mar-2020)
  Found a gpt partition table in /dev/nvme0n1
  Proceed anyway? (y,N) y
  Discarding device blocks: done
  Creating filesystem with 122096646 4k blocks and 30531584 inodes
  Filesystem UUID: 746fb23e-c8ee-4b1f-b9da-d9b8d7665eb7
  Superblock backups stored on blocks:
  32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
  4096000, 7962624, 11239424, 20480000, 23887872, 71663616, 78675968,
  102400000

  Allocating group tables: done
  Writing inode tables: done
  Creating journal (262144 blocks): done
  Writing superblocks and filesystem accounting information: done
```

e. Mount the SSD back to the file system:

```
mount /dev/nvme0n1 /run/media/nvme0n1/
```

f. Link the SSD mount point to common place in the file system:

```
ln -s /run/media/nvme0n1/ /mnt/nvme0n1
```

*NOTE: In the /mnt folder there are also available mount points of the SD card partitions.*

2. Copy the file system to the SSD.

a. Copy file **rootfs.ext4** from the petalinux image folder on your host PC to the SSD connected to the board, use SFTP.

**~/work/TE0802_02_240/test_board/os/petalinux/images/linux/rootfs.ext4**
to
**/mnt/nvme0n1**

b. Switch to the board terminal.

c. Create a folder to where the rootfs.ext4 image file will be mounted:

```
mkdir -p /mnt/rootfs
```

d. Mount the rootfs.ext4 image file to provide its content to the file system.

```
mount -o loop /mnt/nvme0n1/rootfs.ext4 /mnt/rootfs
```

e. Copy the rootfs.ext4 image file content to the SSD:

```
cp -r /mnt/rootfs/* /mnt/nvme0n1
```

f. Unmount the rootfs.ext4 image file:

```
umount /mnt/rootfs
```

3. Set the system to use the SSD as a main file system.
   a. Reboot the system.
   ```
   reboot
   ```
   b. Interrupt the booting sequence at the u-boot stage. Hit any key to stop autoboot
   ```
   Hit any key to stop autoboot:  0
   ZynqMP>
   ```
   c. Modify boot arguments to use the SSD:
   ```
   setenv bootargs "earlycon console=ttyPS0,115200 clk_ignore_unused
   root=/dev/nvme0n1 rw rootwait cma=512M"
   ```
   d. Save new boot arguments
   ```
   saveenv
   ```
   *NOTE: In case you want to just test the boot arguments omit the saveenv command. Than for the next boot the previous saved arguments will be used.*
   e. Continue the booting sequence:
   ```
   boot
   ```

*NOTE 1: To run the demo from the file system located on the SSD, perform steps from Section 3, points 7.d to 7.n.*

*NOTE 2: The previous file system will be still available, but not used by the system. Its mount point will be /mnt/sd-mmcblk0p2.*

# 5 Automations and Optimizations

Some steps can be automatized to avoid doing them after each boot.

## 5.1 QoS

To automatize starting the scripts from the **/home/root/dpu_sw_optimize** folder follow next steps:

1. Copy file **sripts/qos.sh** from the provided ZIP file to the **/etc/init.d/** folder on the board, use SFTP. Make it executable:
   ```
   chmod +x /etc/init.d/qos.sh
   ```
2. Create a link of the **qos.sh** file for the runlevel 5.
   ```
   ln -s /etc/init.d/qos.sh /etc/rc5.d/S99qos
   ```

## 5.2 Monitor Optimizations

1. To set a display resolution to 800x600 automatically during the boot time, copy file **sripts/dp_mon.sh** from the provided ZIP file to the **/etc/profile.d/** folder on the board, use SFTP. Make it executable:
   ```
   chmod +x /etc/profile.d/dp_mon.sh
   ```
2. A monitor connected via the display port will go into sleep mode when keyboard and mouse are in idle state. To disable this behavior append an extra section to the file **/etc/X11/xorg.conf** (modified file is included within the attached ZIP file):
   ```
   Section "ServerFlags"
       Option "BlankTime" "0"
   EndSection
   ```

## 5.3 DPU Firmware

To set a variable pointing the DPU firmware automatically during the boot time, follow next step:

- Copy file **sripts/dpu_fw.sh** from the provided ZIP file to the **/etc/profile.d/** folder on the board, use SFTP. Make it executable:

```
chmod +x /etc/profile.d/dpu_fw.sh
```

# 6 Used Resources

| Resource | Utilization | | Available | Utilization % | |
|---|---|---|---|---|---|
| | All | DPU | | All | DPU |
| LUT | 34180 | 30815 | 47232 | 72.37 | 65.24 |
| LUTRAM | 3325 | 2497 | 28800 | 11.55 | 8.67 |
| FF | 53528 | 46267 | 94464 | 56.66 | 48.98 |
| BRAM | 123 | 123 | 150 | 82.00 | 82.00 |
| DSP | 226 | 224 | 240 | 94.17 | 93.33 |
| IO | 51 | 0 | 82 | 62.20 | 0.00 |
| BUFG | 7 | 0 | 196 | 3.57 | 0.00 |
| MMCM | 1 | 0 | 3 | 33.33 | 0.00 |

# 7 Power Consumption

Conditions of the power consumption measurement:

- Measured with wattmeter Electrobock EMF-1
  - Voltage range: 190-276 VAC (accuracy +/-1%)
  - Current range: 0.01-16A (+/-1%)
  - Power range: 0.2-4416 W (+/-1%)
- Measurement covers these components:
  - Power adapter.
  - Connected to the Ethernet.
  - USB connections: keyboard, mouse, USB camera.
  - Connected to the monitor using the display port, used resolution is 800x600.
  - Running demo: *facedetect* with output displayed on the monitor.
  - The SSD is connected to the board and the system runs from the file system which is located on the SSD.

The power consumption was measured for three situations:

1. The whole system running without the *facedetect* demo: **7.3 W**.
2. Running the DPU performance *facedetect* demo: **8.8 W**
3. Running the *facedetect* demo where the output is shown on the monitor: **8.8 W**.

*NOTE: The power consumption of the system running the facedetect demo depends on used USB camera. To be more precise, it depends on the size of the input image from the camera that has to be resized to the DPU facedetect model requirement (640x360). This resizing is done by ARM processor and this workload of the ARM processor affects the consumption.*

# 8 Package Content

```
.
├── dpu
│   ├── arch.json
│   ├── B1152_2CG
│   │   ├── facedetect
│   │   │   ├── densebox_640_360.prototxt
│   │   │   ├── densebox_640_360.xmodel
│   │   │   └── images
│   │   │       ├── 001.jpg
│   │   │       ├── 002.jpg
│   │   │       ├── 003.jpg
│   │   │       ├── 004.jpg
│   │   │       ├── 005.jpg
│   │   │       ├── 006.jpg
│   │   │       ├── 007.jpg
│   │   │       ├── 008.jpg
│   │   │       ├── 009.jpg
│   │   │       └── 010.jpg
│   │   ├── images
│   │   │   └── bellpeppe-994958.JPEG
│   │   └── resnet50
│   │       └── resnet50.xmodel
│   ├── compile_cf_model.sh
│   └── dpu_conf.vh
├── dpu_sw_optimize
│   └── zynqmp
│       ├── functions
│       │   ├── ext4_auto_resize.sh
│       │   ├── irps5401
│       │   ├── irps5401.c
│       │   └── zynqmp_qos_en.sh
│       ├── README.md
│       └── zynqmp_dpu_optimize.sh
└── scripts
    ├── dp_mon.sh
    ├── dpu_fw.sh
    ├── qos.sh
    └── xorg.conf
```

# 9 References


[1]. Trenz Electronic, „TE0802 Development Board,“ 25 01 2023. [Online]. Available: https://wiki.trenz-electronic.de/display/PD/TE0802+Development+Board.

[2]. Trenz Electronic, „TE0802 test board Vitis AI Tutorial,“ 16 02 2023. [Online]. Available: https://wiki.trenz-electronic.de/display/PD/TE0802+test+board+Vitis+AI+Tutorial.

# A. Compile Models for Custom DPU

1. Prerequisites
   a. Go to https://github.com/Xilinx/Vitis-AI/tree/2.0 and follow instructions from the readme file to install Docker.
   b. We do still assume that *TE0802 test board Vitis AI Tutorial* [2] has been already implemented.
   c. We do also assume that the tutorial in this text has been followed up to the step where the *dpu_trd* project has been built (Section 3 up to step 6.b).
   d. This part of the guide is following instructions provided in Xilinx Vitis AI Github repository and Avnet tutorial at hackster.io

      https://www.hackster.io/AlbertaBeef/vitis-ai-2-0-flow-for-avnet-vitis-platforms-06cfd6

      where more details can be found.
2. Go to Vitis AI library and download models for *resnet50* and *facedetect*.
   a. Go to model folder and run:

```
cd ~/vitis_ai_2_0/models/AI-Model-Zoo
python3 downloader.py
```

   b. In downloader script choose:

```
> Input: cf densebox
> 2
> 1
```

      ZIP archive with **densebox** model will be downloaded from Xilinx

      Run **downloader.py** again to download also resnet50:

```
> Input: cf resnet50
> 1
```

      ZIP archive with resnet model will be downloaded

   c. Unzip both archives downloaded in previous step.
3. Locate file **arch.json** within the compiled **dpu_trd** project and copy it to the **~/vitis_ai_2_0/models/AI-Model-Zoo** folder. The file contains fingerprint of current DPU unit.
4. Find script file **compile_cf_model.sh** provided with this tutorial and copy it to **~/vitis_ai_2_0/models/AI-Model-Zoo**.
5. Pull the docker container (only first time):

```
docker pull xilinx/vitis-ai:2.0.0.1103
```

6. In the folder ~/vitis_ai_2_0 run docker and change directory to model Zoo:

```
sh -x docker_run.sh xilinx/vitis-ai:2.0.0.1103
cd ~/vitis_ai_2_0/models/AI-Model-Zoo
```

7. Create directory for output (see compile_cf_model.sh script):

```
mkdir compiled_output
```

8. Compile models:

```
conda activate vitis-ai-caffe
  (vitis-ai-caffe) $ source ./compile_cf_model.sh densebox_640_360 cf_densebox_wider_360_640_1.11G_2.0
  (vitis-ai-caffe) $ source ./compile_cf_model.sh resnet50 cf_resnet50_imagenet_224_224_7.7G_2.0
```

9. Compiled models can be found in output folder as **.xmodel* files. To successfully run *facedetect* demo, it is also needed to have **.prototxt* file which can be found in the package provided with this application note.

## Disclaimer

This disclaimer is not a license and does not grant any rights to the materials distributed herewith. Except as otherwise provided in a valid license issued to you by UTIA AV CR v.v.i., and to the maximum extent permitted by applicable law:

(1) THIS APPLICATION NOTE AND RELATED MATERIALS LISTED IN THIS PACKAGE CONTENT ARE MADE AVAILABLE "AS IS" AND WITH ALL FAULTS, AND UTIA AV CR V.V.I. HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and

(2) UTIA AV CR v.v.i. shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under or in connection with these materials, including for any direct, or any indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or UTIA AV CR v.v.i. had been advised of the possibility of the same.

Critical Applications:
UTIA AV CR v.v.i. products are not designed or intended to be fail-safe, or for use in any application requiring fail-safe performance, such as life-support or safety devices or systems, Class III medical devices, nuclear facilities, applications related to the deployment of airbags, or any other applications that could lead to death, personal injury, or severe property or environmental damage (individually and collectively, "Critical Applications"). Customer assumes the sole risk and liability of any use of UTIA AV CR v.v.i. products in Critical Applications, subject only to applicable laws and regulations governing limitations on product liability.