Application Note

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

Support for TE0802-02-1BEV2-A board with Vitis AI 3.0 DPU and VGA display

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

Revision history

Rev.	Date	Author	Description
00	14.12.2023	L.K	Initial HW design with X11 on VGA display
01	27.12.2023	J.K	Extensible platform with test_vadd example, AI 3.0 support for AMD DPU in B512 configuration, X11 on VGA display.
02	11.1.2024	J.K	Setup of reserved CMA area to 256 MB. Defined new fixed size and position of simple frame buffer in the user device tree. VGA IP with support of different configurations of colour R,G and B 8 bit data in the 32 bit pixel representation. Support for display output with colour content represented as levels of gray. Description of setting up of Vitis embedded Linux project test_board_vga serving for compilation of the utility vga.elf
O3	15.1.2024	J.K	Includes examples of video streams with input generated as test pattern or from USB video camera and output to VGA X11 desktop on local VGA display and to the remote X11 desktop. Introduction explains how this application note fits into the EECONE project WP4 task 4.3 (second life of electronics due to modular design) and to T4.4 (extension of life due to methodology of support used custom platform for the in-time-evolving design tools and PetaLinux versions).

Contents

Acknowledgement

The EECONE project is supported by the Chips Joint Undertaking and its members, including the top-up funding by National Funding Authorities from involved countries under grant agreement no. 101112065.

https://zs.utia.cas.cz/index.php?ids=projects/eecone

https://eecone.com/eecone/home/

1 Introduction

EECONE project *https://eecone.com/eecone/home/* work package 4, task 4.3 is investigating measures to support second life of electronics due to modular design.

Work package 4 task 4.4 is investigating measures to support extension of life of electronics due to methodology of support used custom platform to adapt for the in-time-evolving design tools and embedded Linux PetaLinux operating system.

UTIA AV CR, v.v.i. (Institute of Information Theory and Automation of the Czech Academy of Sciences, in short UTIA) is not-for profit research institute located in Prague, Czech Republic. UTIA is involved as partner in both tasks, T4.3 and T4.4.

Both EECONE task require specification of comparable reference systems which are based on modular HW with potential for “second life” by reuse of modules or use cost optimized PCB HW without modularity.

Systems (with HW modularity or low cost single PCB) should be capable to perform similar challenging tasks. Systems have to be capable to accelerate in HW AI inference algorithms with video camera input for edge application like person detection, face detection, car-make or car-type detection and graphical output to local display or to the remote PC connected by wired Ethernet in a local network.

Systems should also support remote monitoring and control from remote PC connected by wired Ethernet in a local network.

The investigated measures and methodologies to support “second life” of electronic modules (T4.3) and measures to support extension of life of electronics (T4.4) due to methodology of support used custom platform to adapt for the in-time-evolving design tools and embedded Linux PetaLinux operating system. We target developers designing the final commercial, AI inference based edge applications, mainly in the area of home automation.

Based on these requirements UTIA have selected two types of systems:
- Low cost systems
- Modul based systems

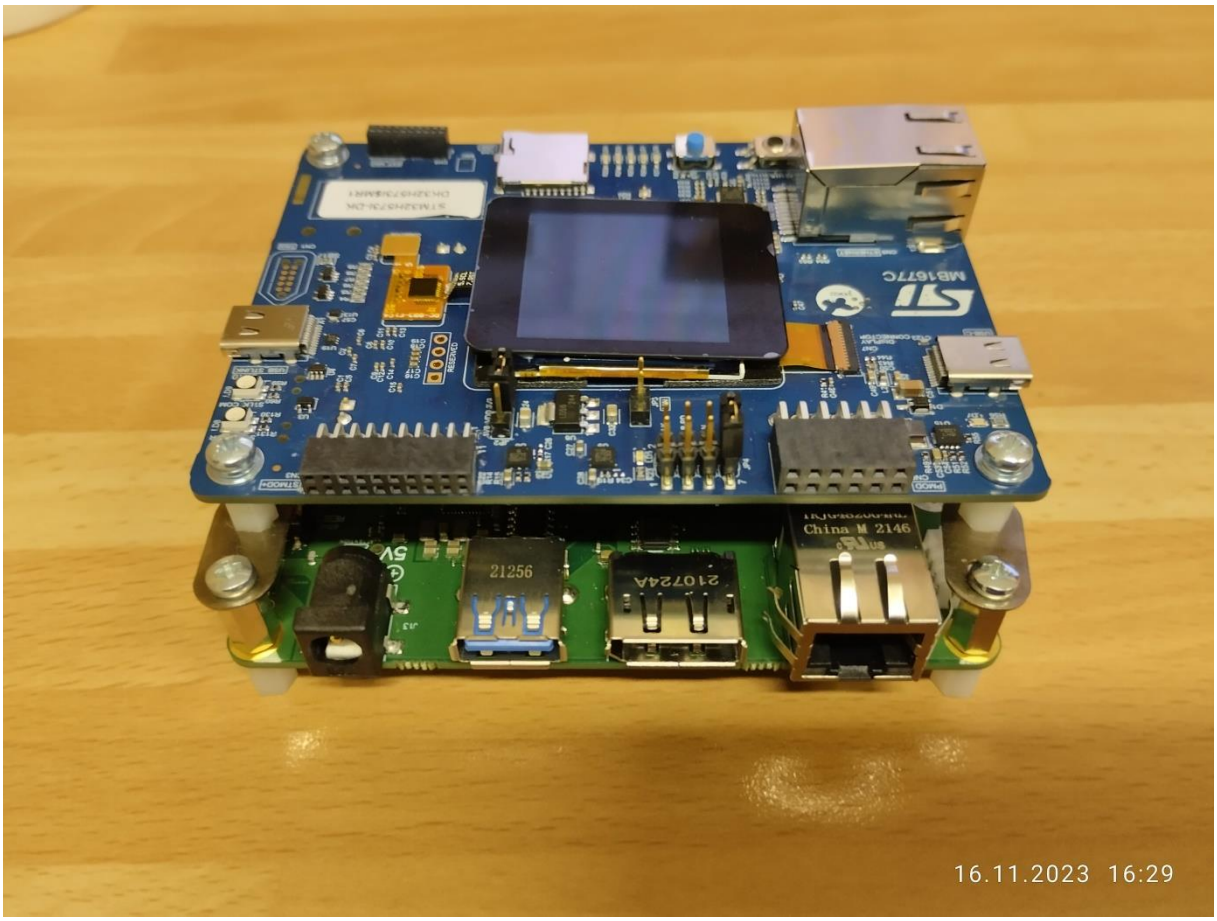
Both compared types of systems use STMicroelectronic STM32H573I-DK board for:
- local system control on small graphical touch screen display
- remote system control from www browser based on www-server or secure communication based on mqtt client. Board is supported by STMicroelectronic CubeMX SW framework and also by NetXDuo SW framework on top of ThreadX OS and FileX SW package.

The MCU used on STM32H573I-DK board is a 40nm chip with 32 bit ARM M33 MCU operating with 250 MHz clock, 2 MBytes of program flash memory and 640 KBytes of RAM.

Compared systems use 16nm AMD ZynqUltrascale+ device with 64 bit ARM A53 Microprocessor and programmable logic in the same device and Petalinux OS.
- Low cost systems have an AMD ZynqUltrascale+ device and DDR4 with all peripheral interfaces soldered on a single, low cost PCB
- Modul based systems have an AMD ZynqUltrascale+ device and DDR4 soldered on an 4x5 cm module connected by connectors to a carrier board with all peripheral interfaces

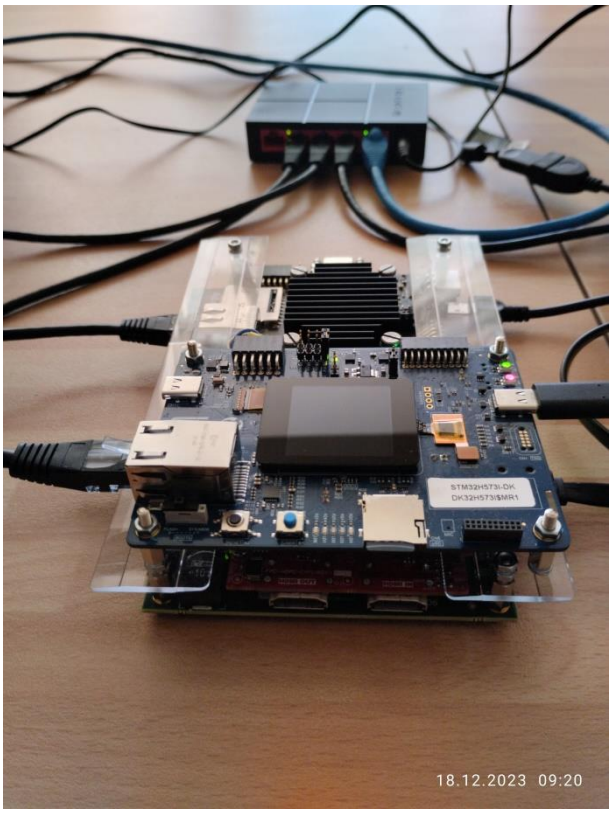
1.1 Low cost systems used by UTIA in EECONE T4.3 and T4.4

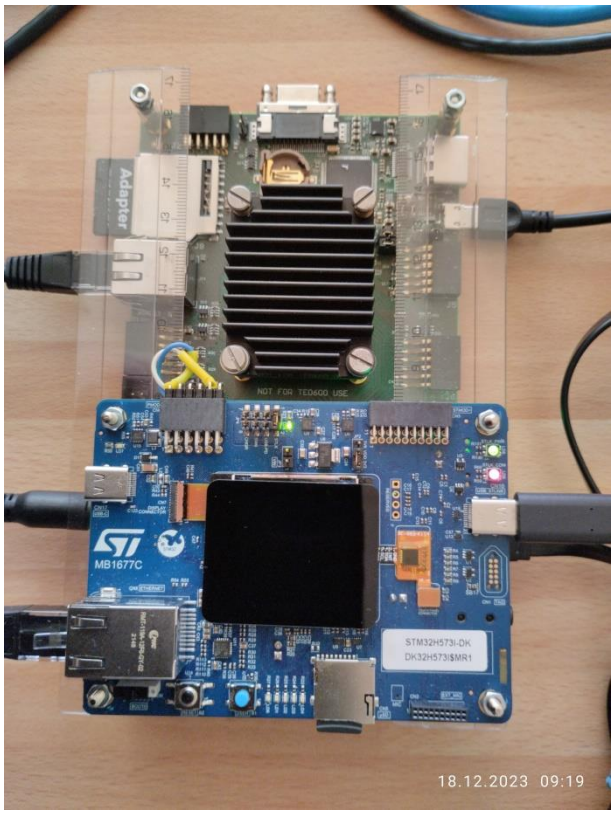
[1]	STM32H573I-DK	https://www.st.com/en/evaluation-tools/stm32h573i-dk.html	Local or remote system control (www-server or secure mqtt client) for [2], [3]
[2]	TE0802-02-1BEV2-A	https://shop.trenz-electronic.de/en/TE0802-02-1BEV2-A-MPSoC-Development-Board-with-AMD-Zynq-UltraScale-ZU1EG-and-1-GB-LPDDR4?c=474	AMD Vitis AI 3.0 AMD DPU in PL USB camera HD VGA display or remote X11 desktop
[3]	TE0802-02-2AEV2-A	MPSoC Development Board mit AMD Zynq™ UltraScale+™ ZU2 und 1 GB LPDDR4 | Trenz Electronic GmbH Online Shop (EN) (trenz-electronic.de)	AMD Vitis AI 3.0 AMD DPU in PL USB camera HD VGA display or remote X11 desktop



1.2 Module based systems used by UTIA in EECONE T4.3 and T4.4

[1]	STM32H573I-DK TE0701-06 Carrier Board for Trenz Electronic 4 x 5 Modules TE0821 or TE0820	https://www.st.com/en/evaluation-tools/stm32h573i-dk.html https://shop.trenz-electronic.de/en/TE0701-06-Carrier-Board-for-Trenz-Electronic-4-x-5-Modules?c=261	Local or remote system control (www-server or secure mqtt client) for [4], [5] Carrier Board for range of 4x5 cm modules [4], [5].
[4]	TE0821 Module: **17 module types** (to be supported)	https://shop.trenz-electronic.de/en/Products/Trenz-Electronic/TE08XX-Zynq-UltraScale/TE0821-Zynq-UltraScale/	AMD Vitis AI 3.0 AMD DPU in PL USB camera FULL HD HDMI display or remote X11 desktop
[5]	TE0820 Module: **100 module types** (to be supported)	https://shop.trenz-electronic.de/en/Products/Trenz-Electronic/TE08XX-Zynq-UltraScale/TE0821-Zynq-UltraScale/	AMD Vitis AI 3.0 AMD DPU in PL USB camera FULL HD HDMI display or remote X11 desktop





This application note and the accompanying evaluation package describe system **[2].** It is available for free public download from UTIA server dedicated to UTIA contributions to EECONE project:
https://zs.utia.cas.cz/index.php?ids=projects/eecone

It will be also available for free public download as **wiki** tutorial from Trenz-Electronic wiki server:
https://wiki.trenz-electronic.de/display/PD/Vitis+AI+and+Vitis+Acceleration+Tutorials+with+Trenz+Electronic+Modules

1.3 Objective of This Application Note and Evaluation Package

This application note and the accompanying evaluation package describe system **[2].**

This application note describes how to design custom HW platform with AMD DPU for Vitis 2022.2 AI 3.0 inference in configuration B512 and also with RGB video output to VGA display for Trenz Electronic evaluation board TE0802-02-1BEV2-A with AMD ZU1EG device.

This application note [1] is using AMD Vitis 2022.2 and PetaLinux 2022 tools installed on Ubuntu 20.04. The described configuration integrated AMD DPU IP, version v4.1.0, with architecture DPUCZDX8G_ISA1_B512.

Described board configuration can operate as small standalone computer with 1 Gb Ethernet connectivity, keyboard, mouse and simplified AMD X11 graphical desktop on VGA RGB444 display with fixed HD resolution 1280x720p60.

Support package for this application note will be available for public download from [1].

The TE0802-02-1BEV2-A is MPSoC Development Board with AMD Zynq™ UltraScale+™ ZU1EG and 1 GB LPDDR4. See [2].

The installed DPU in B512 configuration requires recompilation of Vitis AI 3.0 examples and inference models in the Vitis AI framework. This compilation process will be described in separate application note [3].

Tutorial for Vitis 2021.2.1 and Vitis AI 2.0 with AMD DPU in B800 configuration and without VGA display for Trenz-Electronic evaluation board TE0802-02-1BEV2-A can be downloaded from [4]. The described platform can use DisplayPort video output of Arm A53 processors in PS part of the device. However, not all DisplayPort monitors synchronize correctly with the board.

The VGA display output described here in [1], [3] is in comparison to [4] more robust and works with all VGA RGB 1280x720p60 compatible displays.

Important Note:

The Vitis 2021.2.1 framework with AI 2.0 DPU in B800 configuration and the corresponding precompiled AI 2.0 inference models used in [4] are different from the framework described in this application note [1]. Platform described in this application note [1] requires AI 3.0 DPU and the Inference model compilation described in application note [3].

2 Prepare Reference Design for Extensible Custom Platform with VGA display output

In Ubuntu terminal, source paths to Vitis and Vivado tools by

```
$ source /tools/Xilinx/Vitis/2022.2/settings64.sh
```

Download archive

https://shop.trenz-electronic.de/trenzdownloads/Trenz_Electronic/Development_Boards/TE0802/Reference_Design/2022.2/test_board/TE0802-test_board-vivado_2022.2-build_2_20230628100458.zip

with pre-build files from

https://shop.trenz-electronic.de/en/TE0802-02-1BEV2-A-MPSoC-Development-Board-with-AMD-Zynq-UltraScale-ZU1EG-and-1-GB-LPDDR4?path=Trenz_Electronic/Development_Boards/TE0802/Reference_Design/2022.2/test_board

to .zip archive

```
~/Downloads/TE0802-test_board-vivado_2022.2-build_2_20230628100458.zip
```

This Trenz Electronic archive contains bring-up scripts and pre-build files for creation of PetaLinux for several variants of TE0802 evaluation board.

This application note targets the TE0802-02-1BEV2-A evaluation board (with ZU01-EG device) and this is variant no 04.

Therefore unzip the archive file to the directory
`~/work/te0802_04_240_vga/test_board`

2.1 Reference HW for TE0802-02-1BEV2-A evaluation board

In Ubuntu terminal, change directory to the test_board directory:

```
$ cd ~/work/te0802_04_240_vga/test_board
```

Copy VGA IP core from evaluation package archive accompanying this application note and published in [1]:

```
ip_lib/vga_out_1_0
```

into directory

```
~/work/te0802_04_240_vga/test_board/ip_lib
```

Setup the StarterKit directory files for a Linux host machine.

In Ubuntu terminal, execute:

```
$ chmod ugo+rwx ./console/base_sh/*.sh
```

```
$ chmod ugo+rwx ./_create_linux_setup.sh
```

```
$ ./_create_linux_setup.sh
```

Select option (0) to open Selection Guide and press Enter.

Select variant 4 from the selection guide, press Enter and agree selection.

Create Vivado 2022.2 Project with option 1 and press Enter.

Vivado 2022.2 project will be generated for the selected variant.

The Vivado tool will be opened and Trenz Electronic HW project for the TE0802 test_board HW Design, will be generated.

In Vivado 2022.2 window Sources, click on `zusys_wrapper` and next on `zusys.bd` to open the HW diagram in the IP integrator.

2.2 VGA display.

Copy script
`vivado/te0802-VGA-2022-2.tcl`

from evaluation package archive accompanying this application note and published in [1] into directory
`~/work/te0802_04_240_vga/test_board/vivado`

In Vivado 2022.2 console, source:

```
source te0802-VGA-2022-2.tcl
```

Modified HW diagram includes HW IPs serving for VGA output, now.

2.3 HW support for Vitis Extensible Design Flow

Copy script

```
vivado/fast_track_v2.tcl
```

from evaluation package archive accompanying this application note and published in [1] into directory

`~/work/te0802_04_240_vga/test_board/vivado`

In Vivado 2022.2 console, source

```
source fast_track_v2.tcl
```

Created HW diagram includes VGA related extension of HW and also HW needed for the Vitis 2022.2 extensible design flow.

Validate block design.

Compile HW by

```
TE::hw_build_design -export_prebuilt
```

Compile the custom, Trenz Electronic first stage boot loader by command

```
TE::sw_run_vitis -all
```

Vitis is started.

Close Welcome page and compile

Close Vitis.

Close Vivado

Close Ubuntu terminal.

2.4 Copy Created Custom First Stage Boot Loader

Up to now, test_board directory has been used for the development.

```
~/work/te0802_04_240_vga/test_board
```

Create two new folders:

```
~/work/te0802_04_240_vga/test_board_pfm/pfm/boot

~/work/te0802_04_240_vga/test_board_pfm/pfm/sd_dir
```

Copy the custom first stage boot loader executable file created in section 2.3 from

```
~/work/te0802_04_240_vga/test_board/prebuilt/software/1eg_s1gb/fsbl.elf
```

to

```
~/work/te0802_04_240_vga/test_board_pfm/pfm/boot/fsbl.elf
```

3 Building Petalinux for Extensible Design Flow with Vitis AI 3.0 Support and VGA display

3.1 Vitis AI 3.0 support

Download the Vitis-AI 3.0 repository.
In browser, open page:

https://github.com/Xilinx/Vitis-AI/tree/3.0

Click on green Code button and download Vitis-AI-3.0.zip file.
Unzip
`Vitis-AI-3.0.zip`
file to directory
`~/Downloads/Vitis-AI`

Copy
`~/Downloads/Vitis-AI`
to
`~/work/Vitis-AI-3.0`

The directory

`~/work/Vitis-AI-3.0`

contains the Vitis-AI 3.0 framework, now.

To install the Vitis-AI 3.0 version of shared libraries into rootfs (when generating system image by PetaLinux) we have to copy recepies `recipes-vitis-ai` to the Petalinux project.

Copy
`~/work/Vitis-AI-3.0/src/vai_petalinux_recepies/recipes-vitis-ai`
to
`~/work/te0802_04_240_vga/test_board/os/petalinux/project-spec/meta-user/`

Delete file:
`~/work/te0802_04_240_vga/test_board/os/petalinux/project-spec/meta-user/recipes-vitis-ai/vart/vart_3.0_vivado.bb`
and keep only the unmodified file:
`~/work/te0802_04_240_vga/test_board/os/petalinux/project-spec/meta-user/recipes-vitis-ai/vart/vart_3.0.bb`

File `vart_3.0.bb` will create `vart` libraries for Vitis design flow with dependency on the AMD `xrt` software framework.

3.2 Building Petalinux for Extensible Design Flow Generation of Extensible Platform for Vitis with VGA display support

Open Ubuntu terminal in the directory

`~/work/te0802_04_240_vga/test_board/os/petalinux`

In Ubuntu terminal, source paths to Vitis and Vivado tools and to Petalinux by:

```
$ source /tools/Xilinx/Vitis/2022.2/settings64.sh
```

```
$ source /tools/Xilinx/PetaLinux/2022.2/tool/settings.sh
```

In Ubuntu terminal type

```
petalinux-config --get-hw-description=../../vivado
```

Set Petalinux to use EXT4 disk partition on SD card for the file system.
Change INITRD to EXT4:

```
Image Packaging Configuration -->
      Root filesystem type (INITRD) -->
            (X) EXT4 (SD/eMMC/SATA/USB)
```

Step up to

```
Image Packaging Configuration -->
```

modify `Root filesystem formats` from

```
cpio cpio.gz cpio.gz.u-boot ext4 tar.gz jffs2
```

to

```
ext4
```

Set `Petalinx boot arguments` to use EXT4, set CMA memory size to 256 MB and set `generic UIO` driver.

```
DTG Settings -->
      Kernel Bootargs -->
```

deactivate

```
generate boot args automatically
```

and set

```
user set kernel bootargs
```

string to

```
earlycon console=ttyPS0,115200 clk_ignore_unused root=/dev/mmcblk0p2 rw
rootwait cma=256M uio_pdrv_genirq.of_id=generic-uio
```

Close this Petalinux menuconfig.

Modify file

```
~/work/te0802_04_240_vga/test_board/os/petalinux/project-spec/meta-
user/conf/user-rootfsconfig
```

by appending lines:

```
CONFIG_xrt
CONFIG_xrt-dev
CONFIG_zocl
CONFIG_opencl-clhpp-dev
CONFIG_opencl-headers-dev
CONFIG_packagegroup-petalinux-opencv
CONFIG_packagegroup-petalinux-opencv-dev
CONFIG_dnf
CONFIG_e2fsprogs-resize2fs
CONFIG_parted
CONFIG_resize-part
CONFIG_packagegroup-petalinux-vitisai
CONFIG_packagegroup-petalinux-self-hosted
CONFIG_cmake
CONFIG_packagegroup-petalinux-vitisai-dev
CONFIG_mesa-megadriver
CONFIG_packagegroup-petalinux-x11
CONFIG_packagegroup-petalinux-v4lutils
CONFIG_packagegroup-petalinux-matchbox
CONFIG_vitis-ai-library
CONFIG_vitis-ai-library-dev
CONFIG_vitis-ai-library-dbg
CONFIG_packagegroup-petalinux-gstreamer
CONFIG_libomxil
CONFIG_packagegroup-core-ssh-dropbear
CONFIG_imagefeature-ssh-server-dropbear
```

```
CONFIG_imagefeature-ssh-server-openssh
CONFIG_openssh
CONFIG_openssh-sftp-server
CONFIG_openssh-sshd
CONFIG_openssh-scp
CONFIG_imagefeature-package-management
```

Configure Petalinux rootfs

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

In this menuconfig, go to

```
User Packages -->
```

and select/deselect user packages as follows:

```
[*] libomxil
[*] cmake
[*] dnf
[*] e2fsprogs-resize2fs
[*] gpio-demo
[*] imagefeature-package-management
[ ] imagefeature-ssh-server-dropbear
[*] imagefeature-ssh-server-openssh
[*] mesa-megadriver
[*] opencl-clhpp-dev
[*] opencl-headers-dev
[*] openssh
[*] openssh-scp
[*] openssh-sftp-server
[*] openssh-sshd
[ ] packagegroup-core-ssh-dropbear
[*] packagegroup-petalinux-gstreamer
[*] packagegroup-petalinux-matchbox
[*] packagegroup-petalinux-opencv
[*] packagegroup-petalinux-opencv-dev
[*] packagegroup-petalinux-self-hosted
[*] packagegroup-petalinux-v4lutils
[*] packagegroup-petalinux-vitisai
[*] packagegroup-petalinux-vitisai-dev
[*] packagegroup-petalinux-x11
[*] parted
[*] peekpoke
[*] resize-part
[*] vitis-ai-library
[ ] vitis-ai-library-dbg
[ ] vitis-ai-library-dev
```

```
[*] xrt
[*] xrt-dev
[*] zocl
```

Close this Petalinux menuconfig.

Configure Petalinux kernel

```
petalinux-config -c kernel
```

Switch on the simple framebuffer support by:

```
Device Drivers -->
      Graphics support -->
            Frame buffer Devices -->
                  -*- Support for frame buffer devices
                        <*> Simple framebuffer support
```

Ensure the following items are TURNED OFF by entering 'n' in the [ ] menu selection:

```
CPU Power Management -->
      CPU Idle -->
            CPU idle PM support

CPU Power Management -->
      CPU Frequency scaling -->
            CPU Frequency scaling
```

Close this Petalinux menuconfig.

3.3 Modify Device Tree to support output to the VGA display

In text editor, open file

```
~/work/te0802_04_240_vga/test_board/os/petalinux/project-spec/meta-
user/recipes-bsp/device-tree/files/system-user.dtsi
```

And append these lines:

```
/* FB */

/ {

 /*

 * reserved memory for Debian frame buffer

 *      1 frame = 1920 * 1080 * 4B = 0x007E9000 (ca. 8MB)

 *      1 frame = 1280 *  720 * 4B = 0x00384000 (ca. 3.6MB)

 *      it is assumed that the CMA allocates 256 MB from the top

 *      of the memory

 * for 1GB memory
```

```
 *      0x30000000 to 0x3FFFFFFF              CMA     (256MB)
 *      0x2FC00000 to 0x2FFFFFFF              FB 720p (004MB)
 */
 reserved-memory {
 #address-cells = <2>;
    #size-cells = <2>;
    ranges;
    reserved1: buffer@0x2FC00000 {
      no-map;
      reg = <0x0 0x2FC00000 0x0 (1280 * 720 * 4)>;
    };
  };
  /*
   * simple frame buffer driver for Debian xserver
   * keep the name "fbxserver0"
   */
  framebuffer1: fbxserver {      // VGA out
    #address-cells = <2>;
    #size-cells = <2>;
    compatible = "simple-framebuffer";
    reg = <0x0 0x2FC00000 0x0 (1280 * 720 * 4)>;
    width = <1280>;
    height = <720>;
    stride = <(1280 * 4)>;
    format = "a8b8g8r8";
  };
};

/delete-node/ &axi_vdma_0;

&amba_pl {
```

```
  axi_vdma_0: vdma_0@80010000 {
    #dma-cells = <1>;
    clock-names = "s_axi_lite_aclk", "m_axi_mm2s_aclk", "m_axis_mm2s_aclk";
    clocks = <&zynqmp_clk 71>, <&zynqmp_clk 71>, <&zynqmp_clk 71>;
    compatible = "xlnx,axi-vdma-6.3", "xlnx,axi-vdma-1.00.a";
    interrupt-names = "mm2s_introut";
    interrupt-parent = <&gic>;
    interrupts = <0 89 4>;
    reg = <0x0 0x80010000 0x0 0x10000>;
    xlnx,addrwidth = <0x20>;
    xlnx,flush-fsync = <0x1>;
    xlnx,num-fstores = <0x1>;
    dma-channel@80010000 {
      compatible = "xlnx,axi-vdma-mm2s-channel";
      interrupts = <0 89 4>;
      xlnx,datawidth = <0x20>;
      xlnx,device-id = <0x0>;
      xlnx,genlock-mode ;
    };
  };
};

&axi_vdma_0 {
    compatible = "generic-uio";
    status = "okay";
};
```

3.4 Build Petalinux

Build Petalinux image by Petalinux command in Ubuntu terminal

```
petalinux-build
```

This build can take several hours.

3.5 Build Sysroot Package

Build sysroot package by Petalinux command in Ubuntu terminal

```
petalinux-build --sdk
```

This build can take more than one hour.

The generated `sysroot` package `sdk.sh` is located in directory

```
~/work/te0802_04_240_vga/test_board/os/petalinux/images/linux
```

3.6 Generation of Extensible Platform for Vitis with VGA display

Copy these four files:

Files	From	To

`bl31.elf`
`pmufw.elf`
`system.dtb`
`u-boot-dtb.elf`
	`~/work/te0802_04_240_vga/test_board/os/petalinux/images/linux/`	`~/work/te0802_04_240_vga/test_board_pfm/pfm/sd_dir/`

Rename the copied file **u-boot-dtb.elf** to **u-boot.elf**

The directory
`~/work/te0802_04_240_vga/test_board_pfm/pfm/boot`
contains these five files:

```
bl31.elf
fsbl.elf
pmufw.elf
system.dtb
u-boot.elf
```

Copy these two files:

Files	From	To
`boot.scr` `system.dtb`	`~/work/te0802_04_240_vga/test_board/os/petalinux/images/linux/`	`~/work/te0802_04_240_vga/test_board_pfm/pfm/sd_dir/`

Copy single file:

File	From	To
`init.sh`	`~/work/te0802_04_240_vga/test_board/misc/sd/`	`~/work/te0802_04_240_vga/test_board_pfm/pfm/sd_dir/`

3.7 Generate Sysroot from Sysroot Package

In Ubuntu terminal, change the working directory to:
~/work/te0802_04_240_vga/test_board/os/petalinux/images/linux

In Ubuntu terminal, execute script enabling access to Vitis 2022.2 tools.
Execution of script serving for setting up PetaLinux environment is not necessary:

```
$ source /tools/Xilinx/Vitis/2022.2/settings64.sh
```

In Ubuntu terminal, execute script

```
$ ./sdk.sh
```

Script will ask for the directory where to create sysroot. Enter path.

```
~/work/te0802_04_240_vga/test_board_pfm
```

Confirm Y to create sysroot.

The sysroot directories and files for PC and for Zynq Ultrascale+ will be created in:

```
~/work/te0802_04_240_vga/test_board_pfm/sysroots/x86_64-petalinux-linux
~/work/te0802_04_240_vga/test_board_pfm/sysroots/cortexa72-cortexa53-
xilinx-linux
```

Once created, do not move these sysroot directories (due to some internally created paths).

3.8 Generate Extensible Platform with VGA Display

In Ubuntu terminal, change the working directory to:exit

```
~/work/te0802_04_240_vga/test_board_pfm
```

In Ubuntu terminal, execute script enabling access to Vitis 2022.2 tools.

```
$ source /tools/Xilinx/Vitis/2022.2/settings64.sh
```

Start the Vitis 2022.2 tool by executing

```
$ vitis &
```

In Vitis “Launcher”, set the workspace for the extensible platform compilation to:

```
~/work/te0802_04_240_vga/test_board_pfm
```

Click on “Launch” to launch Vitis

Close Welcome page.

In Vitis, select in the main menu:
`File -> New -> Platform Project`

Type name of the extensible platform:
`te0802_04_240_vga_pfm`

Click Next.

Choose for hardware specification for the platform file:
`~/work/te0802_04_240_vga/test_board/vivado/test_board_1eg_s1gb.xsa`

In “Software specification” select:
`linux`

In “Boot Components” unselect:
`Generate boot components`

These components have been already generated by PetaLinux.

New window `te0802_04_240_vga_pfm` is opened.

Click on `linux on psu_cortex53` to open window Domain: `linux_domain`

In “Description” write:
`xrt`

In “Bif File” find and select the pre-defined option:
`Generate Bif`

In “Boot Components Directory” select:
`~/work/te0802_04_240_vga/test_board_pfm/pfm/boot`

In “FAT32 Partition Directory” select:
`~/work/te0802_04_240_vga/test_board_pfm/pfm/sd_dir`

In Vitis IDE “Explorer” section, click on `te0802_04_240_vga_pfm` to highlight it.

Right-click on the highlighted `te0802_04_240_vga_pfm` and select `build project` in the open submenu. Platform is compiled in few seconds.
Close the Vitis 2022.2 tool by selection:
`File -> Exit`

Vitis 2022.2 extensible platform

`te0802_04_240_vga_pfm`

has been created in the directory:

`~/work/te0802_04_240_vga/test_board_pfm/te0802_04_240_vga_pfm/export/te0802_04_240_vga_pfm`

4 Platform Usage

4.1 Read Platform Info

With Vitis environment setup, platforminfo tool can report XPFM platform information.

```
platforminfo
~/work/te0802_04_240_vga/test_board_pfm/te0802_04_240_vga_pfm/export/te
0802_04_240_vga_pfm/te0802_04_240_vga_pfm.xpfm
```

4.2 Create and Compile Vector Addition Example

Create new directory **test_board_test_vadd** to test Vitis extendable flow example "vector addition"

```
~/work/te0802_04_240_vga/test_board_test_vadd
```

Current directory structure:

```
~/work/te0802_04_240_vga/test_board
~/work/te0802_04_240_vga/test_board_pfm
~/work/te0802_04_240_vga/test_board_test_vadd
```

Change working directory:

```
$ cd ~/work/te0802_04_240_vga/test_board_test_vadd
```

In Ubuntu terminal, execute script enabling access to Vitis 2022.2 tools.

```
$ source /tools/Xilinx/Vitis/2022.2/settings64.sh
```

In Ubuntu terminal, start Vitis by:

```
$ vitis &
```

In Vitis IDE Launcher, select your working directory

```
~/work/te0802_04_240_vga/test_board_test_vadd
```

Click on Launch to launch Vitis.

Select File -> New -> Application project. Click Next.

Skip welcome page if shown.

Click on "+ Add" icon and select the custom extensible platform te0802_04_240_vga_pfm[custom] in the directory:

```
~/work/te0802_04_240_vga/test_board_pfm/te0802_04_240_vga_pfm/export/te
0802_04_240_vga_pfm
```

We can see available PL clocks and frequencies.

Click `Next`.

In “Application Project Details” window type into Application project name: `test_vadd`
Click `Next`.

In “Domain window” type (or select by browse):

“Sysroot path”:
`~/work/te0802_04_240_vga/test_board_pfm/sysroots/cortexa72-cortexa53-xilinx-linux`

“Root FS”:
`~/work/te0802_04_240_vga/test_board/os/petalinux/images/linux/rootfs.ext4`

“Kernel Image”:
`~/work/te0802_04_240_vga/test_board/os/petalinux/images/linux/Image`
Click `Next`.

In “Templates window”, if not done before, update “Vitis IDE Examples” and “Vitis IDE Libraries”.

Select `Host Examples`
In “Find”, type: “vector add” to search for the “Vector Addition” example.

Select: “Vector Addition”
Click `Finish`
New project template is created.

In test_vadd window menu “Active build configuration” switch from “SW Emulation” to “`Hardware`”.

In “Explorer” section of Vitis IDE, click
on: `test_vadd_system[te0802_04_240_vga_pfm]` to select it.

Right Click on: `test_vadd_system[te0802_04_240_vga_pfm]` and select in the opened sub-menu:
`Build project`

Vitis will compile HW kernel and application SW for Arm host.

In test_vadd_kernels subproject, compile the krnl_vadd from C++ SW to HDL HW IP source code

In test_vadd_system_hw_link subproject, compile the krnl_vadd HDL together with te0802_04_240_vga_pfm into new, extended HW design with new accelerated (krnl_vadd) will run on the default 240 MHz clock. This step can take some time.

In test_vadd subproject, compile the vadd.cpp application example.

4.3 Run Compiled test_vadd Example Application from USB Terminal

The sd_card.img file is output of the compilation and packing by Vitis. It is located in the directory:
~/work/te0802_04_240_vga/test_board_test_vadd/test_vadd_system/Hardware/package/sd_card.img

Write the sd card image from the sd_card.img file to SD card.

Insert the SD card to the TE0802 evaluation board.

Connect PC USB terminal (115200 bps) cable to the TE0802 evaluation board.

Connect Ethernet cable to the TE0802 evaluation board.

Power on the TE0802 board.

In PC, find the assigned serial line COM port number for the USB terminal. In case of Win 10 or Win 11, use device manager.

In PC, open serial line terminal with the assigned COM port number. Speed 115200 bps.

In PC terminal, type:

```
sh-5.0# cd /run/media/mmcblk0p1/
sh-5.0# ./test_vadd krnl_vadd.xclbin
```

The application test_vadd should run with this output:

```
INFO: Reading krnl_vadd.xclbin
Loading: 'krnl_vadd.xclbin'
Trying to program device[0]: edge
Device[0]: program successful!
TEST PASSED
sh-5.0#
```

The Vitis application has been compiled to HW and evaluated on custom system with extensible custom te0802_04_240_vga_pfm platform.

In PC terminal type:

```
# halt
```

System is halted. Messages relate to halt of the system can be seen on the USB terminal.

The SD card can be safely removed from the the TE0802 board, now.

The TE0802 board can be disconnected from power.

4.4 Enable KBD, Mouse, VGA Video Display with QOS Support

Connect USB KBD and USB mouse to TE0802 board.

Connect VGA video display to TE0802 board and power on the display.

On the TE0802 board, reset button to start the system. USB terminal starts to display booting information. VGA display remains black with no signal at this stage.

In PC USB terminal, find TE0802 board IP address (assigned by the network DHCP server) by command:

```
ifconfig eth0
```

On PC, start utility for `sftp` Ethernet file transfer to the TE0802 board.

Copy the executable precompiled binary file

`sw\prebuilt\vga.elf`
from evaluation package accompanying this application note [1] to TE0802 board file

```
/run/media/mmcblk0p1/vga.elf
```

Copy file

```
sw\scripts\01vga.sh
```

from evaluation package accompanying this application note [1] to T E0802 board file

```
/etc/X11/Xsession.d/01vga.sh
```

Copy file

```
sw\scripts\xorg.conf
```

from evaluation package accompanying this application note [1] to TE0802 board file (old file will be overwritten)

```
/etc/X11/xorg.conf
```

Copy scripts optimizing the data transfers between the DPU and the memory controller from the PC to the home folder on the board.

To copy them use SFTP from

```
dpu_sw_optimize
```

to

```
/home/root
```

Switch back to the board terminal.

To 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

To automate start of scripts from the

```
/home/root/dpu_sw_optimize
```

folder follow this steps:

Copy file

```
sripts/qos.sh
```

to the

```
/etc/init.d/
```

folder on the board, use SFTP. Make it executable:

```
chmod +x /etc/init.d/qos.sh
```

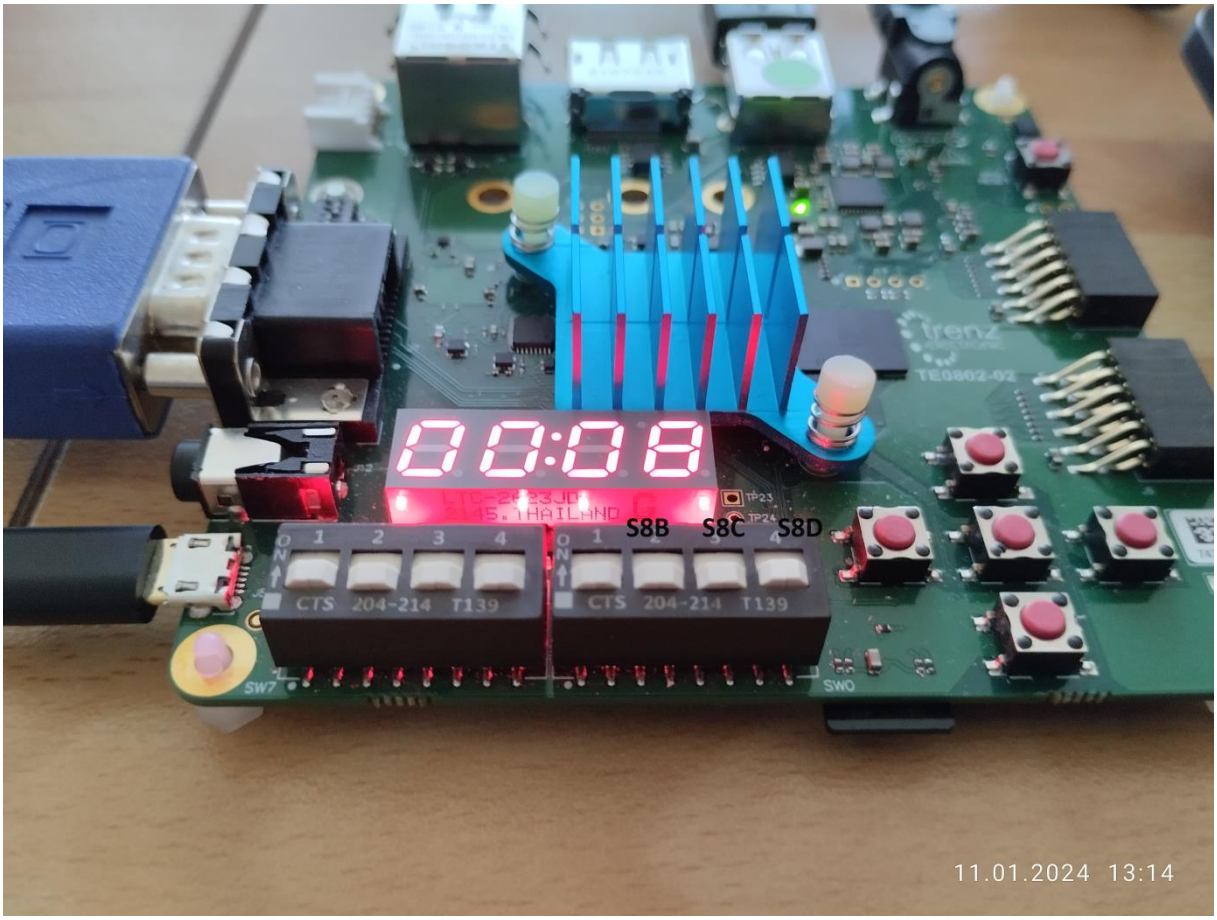
Create a link of the qos.sh file for the `runlevel 5`.

```
ln -s /etc/init.d/qos.sh /etc/rc5.d/S99qos
```

Reboot the TE0802 board by typing in the PC USB terminal

```
reboot
```

The TE0802 board is rebooted with VGA video output enabled on the display.



Switches [S8B, S8C, S8D] (marked in black) are set to the default [OFF,OFF,OFF] state.

The simplified AMD X11 desktop is created with several icons. The resolution is fixed to the HD with resolution 1280x720p60. Setting of switches [S8B, S8C, S8D] = [OFF,OFF,OFF] defines output to VGA display as HD with colours converted to levels of gray.

4.5 Display of Single Frame Buffer

Implemented VGA supports different configurations of R G and B 8 bit colour data positions in the 24 bit representation of each pixel. It is controlled by 3 switches S8B, S8C, S8D of the evaluation board.

S8B fpga pin M1 PCB name USER_SW[2]	**S8C** fpga pin P2 PCB name USER_SW[1]	**S8D** fpga pin P3 PCB name USER_SW[0]	**Colour byte combination**		
OFF	OFF	OFF	All colour combinations displayed as **gray**		
ON	OFF	OFF	G	B	R
OFF	**ON**	**OFF**	**B**	**G**	**R**
ON	**ON**	**OFF**	**R**	**G**	**B**
OFF	OFF	ON	B	R	G
ON	OFF	ON	R	B	G
OFF	ON	ON	G	R	B
ON	ON	ON	All colour combinations displayed as **gray**		

The basic X11 GUI is correctly displayed with this switch combination

`[S8B,S8C,S8D] = [ON,ON,OFF]`

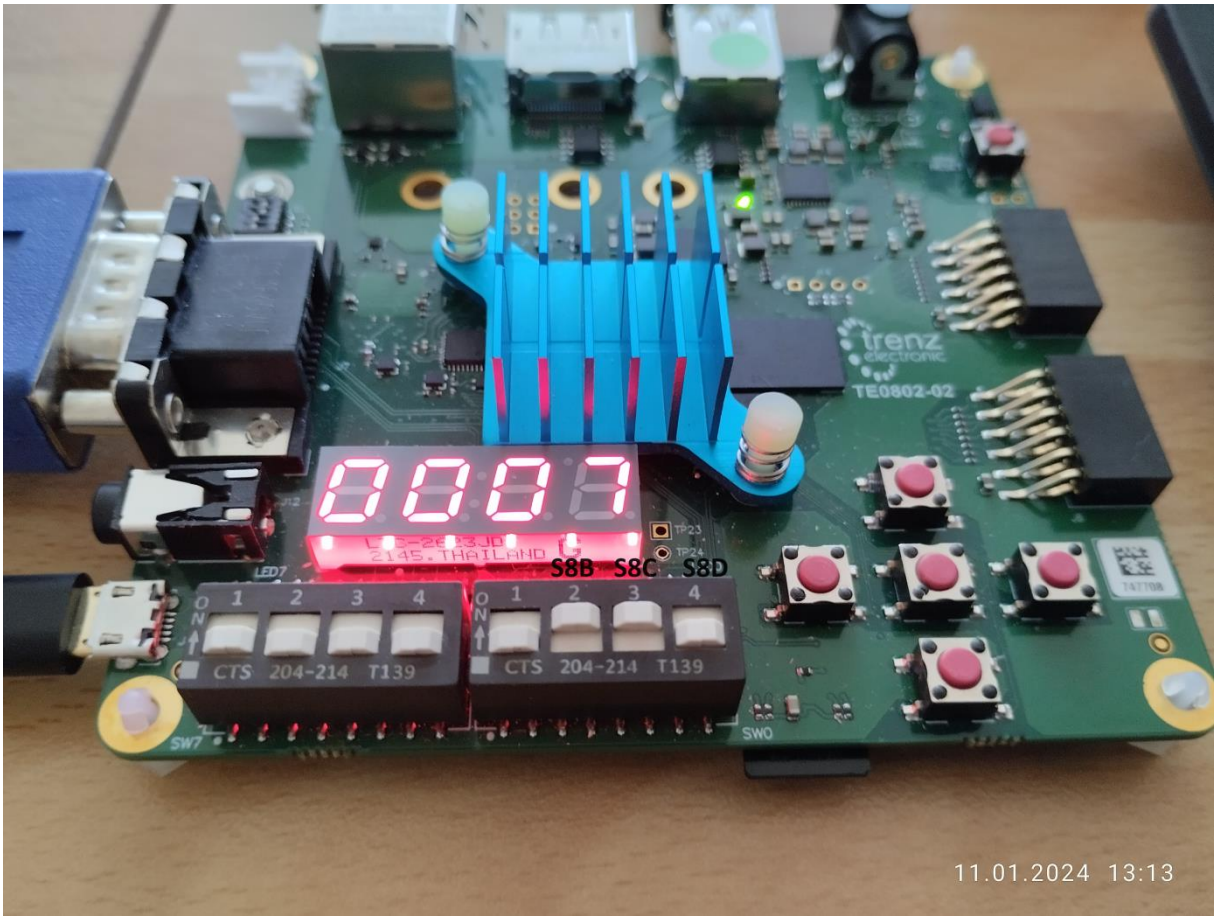
Vitis AI 3.0 applications generate output to the simple frame buffer in format BGR888. Therefore, the X11 output images generated by Vitis AI 3.0 applications can have swapped B and R. These X11 single frame outputs are correctly displayed with this switch combination

`[S8B,S8C,S8D] = [OFF,ON,OFF]`

All combinations of buffers can be displayed as gray image by these two switch combinations

`[S8B,S8C,S8D] = [OFF,OFF,OFF]`

`[S8B,S8C,S8D] = [ON,ON,ON]`



Switches [S8B, S8C, S8D] (marked in black) are set to [ON,ON,OFF] state.

You can change setting of switches [S8B, S8C, S8D] in runtime.

Setting of switches [S8B, S8C, S8D] = [ON,ON,OFF] defines output to VGA monitor as HD with colour RGB444 output.

4.6 User Control of X11 GUI

Mouse and keyboard connected to the TE0802 board are used as user input devices.

Click on “Terminal” icon (A Unicode capable `rxvt` terminal emulator).

Terminal emulator `rxvt` opens as full screen graphic window.

In terminal emulator, use keyboard connected to the TE0802 board and type:

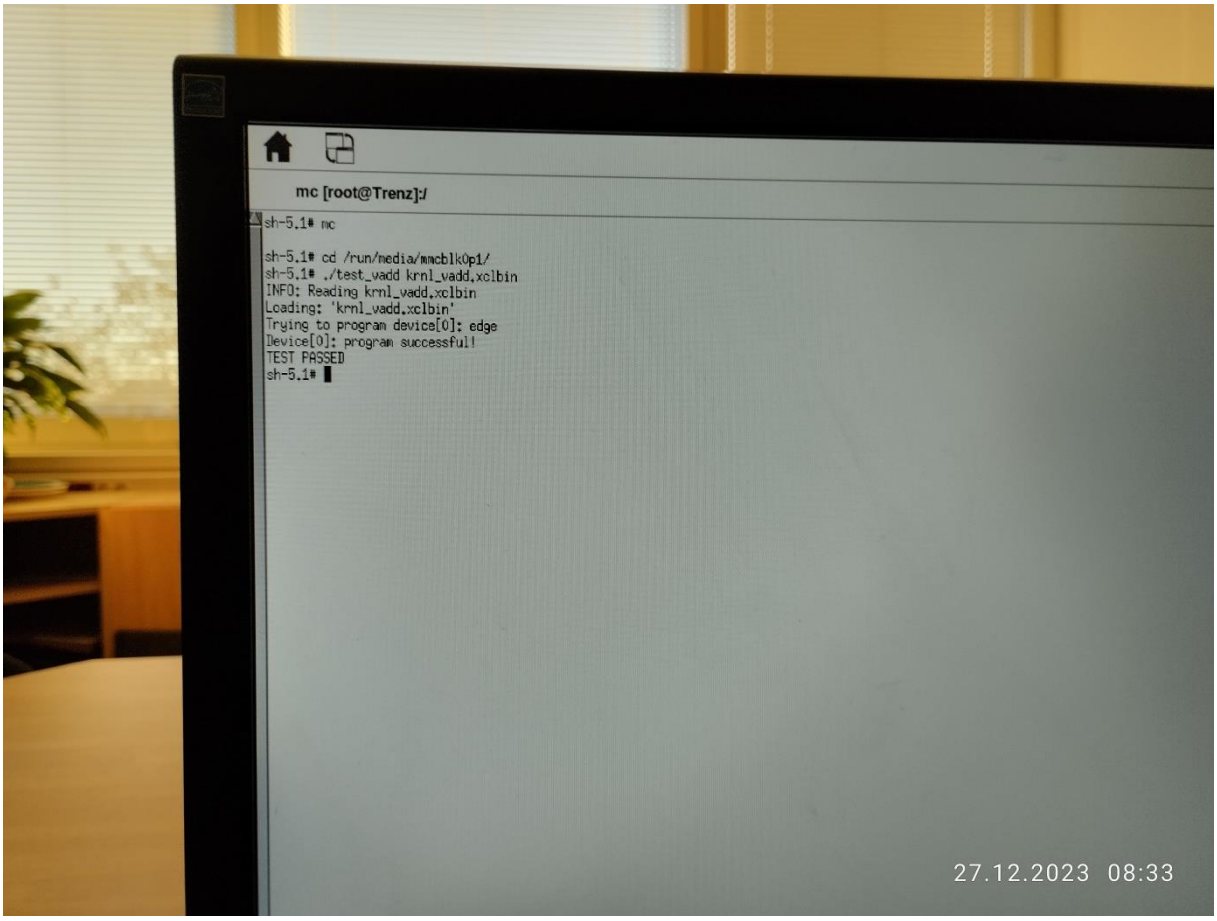
```
root@Trenz:~# cd /run/media/mmcblk0p1/
root@Trenz:~# ./test_vadd krnl_vadd.xclbin
```

The application test_vadd with HW accelerated vector addition will run with this output on the X11 terminal emulator:

```
INFO: Reading krnl_vadd.xclbin
Loading: 'krnl_vadd.xclbin'
Trying to program device[0]: edge
Device[0]: program successful!
```

```
TEST PASSED

sh-5.0#
```



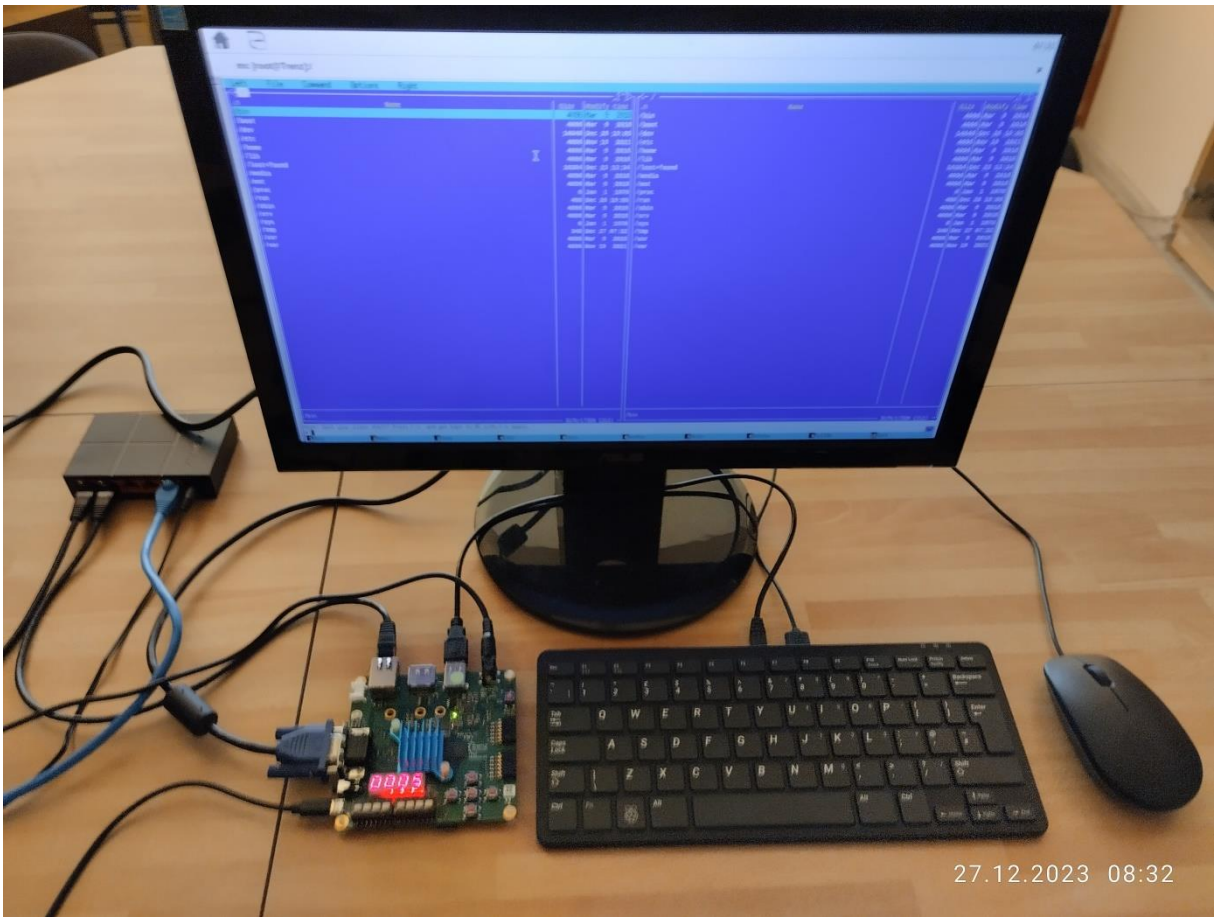
The TE0802 board is running the PetaLinux OS and drives simple version of an X11 GUI on VGA monitor. Application `test_vadd` has been started from xrvt terminal emulator.

The Vitis application has been compiled to HW and evaluated on custom system with extensible custom te0802_04_240_vga_pfm platform.

4.7 Multiple X11 windows

You can go to the X11 desktop and open second `rxvt` terminal emulator as a new full screen X11 window.

You can run (for example) `mc` application in this second terminal. It can be used for most common operations with files (copy, edit, unzip, …).



Mouse controlled X11 desktop GUI can be used to select, which terminal is visible in full screen.

You can stop `mc` application by pressing the TE0802 keyboard key F10.

4.8 Halt the TE0802 Board

Close all open rxvt terminal emulators by click on ”x” icon (in the upper right corner) or by typing:

```
root@Trenz:~# exit
```

In X11, click on ”Shutdown” icon to close filesystem of the PetaLinux running on the TE0802 board and halt the system.

System is halted, now.

Messages related to halt of the system can be seen on the PC USB terminal.

The VGA output to the display is also switched off and the vga display is dark.

The SD card can be safely removed from the TE0802 board, now.

The TE0802 board can be disconnected from power, now.

If you power ON the board with inserted SD card, the PetaLinux with X11 output to the enabled VGA display will start automatically.

If you create new SD card with new example, the steps described in section 4.4 have to be repeated to enable the automatic start of the VGA display output.

4.9 Vitis project test_board_vga (optional)

The executable `vga.elf` has been used for start of video-dma. It starts output of X11 desktop on HD VGA display (1280x720p60). This executable can be re-created from source code in Vitis project targeting compilation of C application for embedded flow.

Create new directory `test_board_vga` for Vitis embedded flow project `vga`

```
~/work/te0802_04_240_vga/test_board_vga
```

Current directory structure:

```
~/work/te0802_04_240_vga/test_board
~/work/te0802_04_240_vga/test_board_pfm
~/work/te0802_04_240_vga/test_board_test_vadd
~/work/te0802_04_240_vga/test_board_vga
```

Change working directory:

```
$ cd ~/work/te0802_04_240_vga/test_board_vga
```

In Ubuntu terminal, execute script enabling access to Vitis 2022.2 tools.

```
$ source /tools/Xilinx/Vitis/2022.2/settings64.sh
```

In Ubuntu terminal, start Vitis by:

```
$ vitis &
```

In Vitis IDE Launcher, select your working directory

```
~/work/te0802_04_240_vga/test_board_vga
```

Click on Launch to launch Vitis.

Select File -> New -> Application project. Click Next.

Skip welcome page, if shown. Click on `+Add` icon and select the custom extensible platform `te0802_04_240_vga_pfm[custom]` in the directory:

```
~/work/te0802_04_240_vga/test_board_pfm/te0802_04_240_vga_pfm/export/te0802_04_240_vga_pfm
```

We can see available PL clocks and frequencies.

Click Next.

In “Application Project Details” window type into Application project name: vga
Click Next.

In “Domain window” type (or select by browse):

“Sysroot path”:
~/work/te0802_04_240_vga/test_board_pfm/sysroots/cortexa72-cortexa53-xilinx-linux

“Root FS”:
~/work/te0802_04_240_vga/test_board/os/petalinux/images/linux/rootfs.ext4

“Kernel Image”:
~/work/te0802_04_240_vga/test_board/os/petalinux/images/linux/Image
Click Next.

Select Embedded software development templates
Select: Linux Hello World
Click Finish
New project template vga_system is created.

In vga_system window menu “Active build configuration” switch from “SW Emulation” to “Hardware”.

Created C template contains simple helloworld.c source code.

```
~/work/te0802_04_240_vga/test_board_vga/vga/src/helloworld.c
```

Delete helloworld.c

Copy from package accompanying this application note:
sw\src\vga\src

Source code file main.c and content of directories dt and vdma

```
~/work/te0802_04_240_vga/test_board_vga/vga/src/main.c

~/work/te0802_04_240_vga/test_board_vga/vga/src/dt/*

~/work/te0802_04_240_vga/test_board_vga/vga/src/vdma/*
```

In “Explorer” section of Vitis IDE, click on: vga_system[te0802_04_240_vga_pfm] to select it.

Right Click on: vga_system[te0802_04_240_vga_pfm] and select in the opened sub-menu:
Build project

Vitis will compile C application SW for Arm host, Linux target.

Compiled executable vga.elf is located in

```
~/work/te0802_04_240_vga/test_board_vga/vga/Release/vga.elf
```

Use your sftp utility to copy this compiled executable vga.elf to TE0802 test board file

```
/run/media/mmcblk0p1/vga.elf
```

On the TE0802 test board, close all running demos and use `reboot` command to reboot the TE0802 test board. This will test the created `vga.elf` utility. Linux will reboot and the X11 desktop will open automatically on the VGA display.

5 Vitis AI 3.0 DPUCZDX8V_VAI_v3.0 Installation

This section explains how to download and compile the Vitis AI 3.0 compatible DPU for the TE0802 evaluation board.

Download and install Vitis 2022.2 project with configurable DPU instance from repository:

https://github.com/Xilinx/Vitis-AI/tree/3.0/dpu

This page contains table with supported targets. Use the line of this table dedicated to the DPUCZDX8G DPU for MPSoC and Kria K26 devices. It is link for download of the programmable logic based DPU, targeting general purpose CNN inference with full support for the Vitis AI ModelZoo. It supports either the Vitis or Vivado flows on 16nm Zynq® UltraScale+™ platforms.

Therefore, click on the Download link in the column:
`Reference Design`

https://www.xilinx.com/bin/public/openDownload?filename=DPUCZDX8G_VAI_v3.0.tar.gz

This will result in download of file:

`~/Downloads/DPUCZDX8V_VAI_v3.0.tar.gz`

It contains directory
`~/Downloads/DPUCZDX8V_VAI_v3.0`

Copy this directory to the directory:
`~/work/DPUCZDX8V_VAI_v3.0`

It contains HDL code for the DPU and also source files and project files to test the DPU with AI resnet50 inference example.

5.1 Create and Build Vitis AI 3.0 DPU Design

Create new directory `test_board_dpu_trd` to test Vitis extendable flow example
`test_dpu trd`

`~/work/te0802_04_240_vga/test_board_dpu_trd`

Current directory structure:

```
~/work/te0802_04_240_vga/test_board
~/work/te0802_04_240_vga/test_board_pfm
~/work/te0802_04_240_vga/test_board_test_vadd
~/work/te0802_04_240_vga/test_board_dpu_trd
~/work/te0802_04_240_vga/test_board_vga
```

Change working directory:
$cd ~/work/te0802_04_240_vga/test_board_dpu_trd

In Ubuntu terminal, start Vitis 2022.2 by:
```
$ vitis &
```

In Vitis IDE Launcher, select your working directory
`~/work/te0802_04_240_vga/test_board_dpu_trd`
Click on Launch to start Vitis.

5.2 Add DPU Project template to the Vitis Extensible Flow

In Vitis, open menu `Window → Preferences`

Go to `Library Repository` tab

Add Vitis-AI by clicking `Add` button and `Fill Settings` like:

```
ID                vitis-ai
Name              Vitis AI
Description       Vitis AI
Location          /home/devel/work/DPUCZDX8G_VAI_v3.0
Git URL
Branch
```

Use absolute path to your home folder in field "Location":

Field "Location" says that the Vitis-AI repository from github has been cloned into
`~/work/DPUCZDX8V_VAI_v3.0`

Hovewer., the absolute path to your home directory is needed. In our case:
`/home/devel/work/DPUCZDX8V_VAI_v3.0`

The user name in the figure is `devel` in our case. Replace it by your user name.

Keep `Git URL` and `Branch` empty. Click `Apply` and `Close`.

The added library appears in Libraries. In Vitis, open menu `Xilinx → Libraries...`

You can find the installed `Vitis-AI` library marked as `Installed`

5.3 Configure Project for the Vitis Extensible Flow with DPU

Select
`File -> New -> Application project`.
Click
`Next`

Skip welcome page if it is shown.

Click on
+Add
icon and select the custom extensible platform
```
te0802_04_240_vga_pfm[custom]
```
in the directory:
```
~/work/te0802_04_240_vga/test_board_pfm/te0802_04_240_vga_pfm/export/te0802_04_240_vga_pfm
```
We can see available PL clocks and frequencies. PL4 with 240 MHz clock is has been set as default in the platform creation process. Click:
`Next`

In “Application Project Details” window type into Application project name:
`test_dpu_trd`
Click
`Next`

In “Domain window” type (or select by browse):

“Sysroot path”:
`~/work/te0802_04_240_vga/test_board_pfm/sysroots/cortexa72-cortexa53-xilinx-linux`

“Root FS”:
`~/work/te0802_04_240_vga/test_board/os/petalinux/images/linux/rootfs.ext4`

“Kernel Image”:
`~/work/te0802_04_240_vga/test_board/os/petalinux/images/linux/Image`

Click
`Next`

In “Templates window”, if not done before, update `Vitis IDE Examples` and `Vitis IDE Libraries`.

In “Find”, type:
`dpu`
to search for the `DPU Kernel (RTL Kernel)` example.

Select: `DPU Kernel (RTL Kernel)`

Click
`Finish`

New project template with DPU instance is created.

In dpu_trd window menu “Active build configuration” switch from “SW Emulation” to “`Hardware`”.

File `dpu_conf.vh` located at `dpu_trd_kernels/src/prj/Vitis` directory contains the initial default DPU configuration (defined for larger ZU devices).

In text editor, replace content of file `dpu_conf.vh` with this new content:

```
/*
```

```
* Copyright 2019 Xilinx Inc.
*
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*
*    http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
*  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
* See the License for the specific language governing permissions and
* limitations under the License.
*/

//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 B512

// |-----------------------------------------------------|
// | 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

// |-----------------------------------------------------|
// | ALU parallel Configuration
// | It relates to model. if change, must update model
// +-----------------------------------------------------+
// | setting 0  : `define ALU_PARALLEL_DEFAULT
// +-----------------------------------------------------+
// | setting 1  : `define ALU_PARALLEL_1
// |-----------------------------------------------------|
// | setting 2  : `define ALU_PARALLEL_2
// |-----------------------------------------------------|
// | setting 3  : `define ALU_PARALLEL_4
// |-----------------------------------------------------|
// | setting 4  : `define ALU_PARALLEL_8
// |-----------------------------------------------------|

`define ALU_PARALLEL_DEFAULT

// +-----------------------------------------------------+
// | CONV RELU Type Configuration
// | It relates to model. if change, must update model
// +-----------------------------------------------------+
// | `define CONV_RELU_RELU6
// +-----------------------------------------------------+
// | `define CONV_RELU_LEAKYRELU_RELU6
// |-----------------------------------------------------|

`define CONV_RELU_LEAKYRELU_RELU6

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

```
// | `define ALU_RELU_RELU6
// +-----------------------------------------------------+
// | `define ALU_RELU_LEAKYRELU_RELU6
// |-----------------------------------------------------|

`define ALU_RELU_RELU6

// |-----------------------------------------------------|
// | argmax or max Configuration
// | It relates to model. if change, must update model
// +-----------------------------------------------------+
// | enable  : `define SAVE_ARGMAX_ENABLE
// +-----------------------------------------------------+
// | disable : `define SAVE_ARGMAX_DISABLE
// |-----------------------------------------------------|

`define SAVE_ARGMAX_ENABLE

// |-----------------------------------------------------|
// | 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
```

It selects DPU configuration B512 and maximal use of DSP48 blocks.

This selection can fit into the PL logic of the ZU01-EG device together with the VGA display support.

5.4 Configure Connection of DPU Kernel

Go to `dpu_trd_system_hw_link` and double click on `dpu_trd_system_hw_link.prj`.

Remove `sfm_xrt_top` kernel from binary container by right clicking on it and choosing remove.

Reduce number of DPU kernels from 2 to 1.
On the same tab, right click on `dpu` and choose:

`Edit V++ Options`

Click "`...`" button on the line of
`V++ Configuration Settings`
and modify the 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:HP2
```

```
sp=DPUCZDX8G_1.M_AXI_HP2:HP3
```

5.5 Build the test_dpu_trd Project

In "Explorer" section of Vitis IDE, click on:
`dpu_trd_system[te0802_04_240_vga_pfm]`
to select it.

Right Click on:

`dpu_trd_system[te0802_04_240_vga_pfm]`

and select in the opened sub-menu:
`Build project`

Compilation takes some time (approximately 30 minutes).

Created extended HW with integrated DPU with configuration B1024 can be open and analysed in Vivado 2022.2

6 Prepare SD card for TE0802 with test_dpu_trd DPU and VGA

The **sd_card.img** file is output of the compilation and packing by Vitis. It is located in directory:
`~/work/te0802_04_240_vga/test_board_test_dpu_trd/dpu_trd_system/Hardware/package/`

In Windows 10 (or Windows 11) PC, install program `Win32DiskImager` for this task. Win32 disk imager can write raw disk image to removable devices.
https://win32diskimager.org/

Boot the board and open terminal on the board either by connecting serial console connection, or by opening Ethernet connection to ssh server on the board, or by opening terminal directly using window manager on board. Continue using the embedded board terminal.

6.1 Enable KBD, Mouse, VGA and QOS Support

Enable KBD, Mouse and VGA Video Display as described in section 4.4.

Reboot TE0802 board to get VGA dispaly output, keyboard and mouse.

6.2 Resize EXT4 Partition

Check ext4 partition size by:

```
root@Trenz:~# cd /
root@Trenz:~# df .
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/root               564048    398340    122364  77% /
```

Resize partition

```
root@Trenz:~# resize-part /dev/mmcblk0p2
```

Check ext4 partition size again, you should see:

```
root@Trenz:~# df . -h
Filesystem                Size      Used Available Use% Mounted on
/dev/root                 6.1G    390.8M      5.4G   7% /
```

The available size would be different according to your SD card size.

Set DISPLAY variable:

```
root@petalinux:~# export DISPLAY=:0.0
```

Set path to Xilinx Firmware:

```
root@petalinux:~# export
XLNX_VART_FIRMWARE=/run/media/mnt/mmcblk0p1/dpu.xclbin
```

6.3 Test the Integrated DPUCZDX8G_ISA1_B512

The integrated DPU can be tested by command:

```
xdputil query
```

```
root@Trenz:~# export DISPLAY=:0.0
root@Trenz:~# export XLNX_VART_FIRMWARE=/run/media/mmcblk0p1/dpu.xclbin
root@Trenz:~# xdputil query
{
    "DPU IP Spec":{
        "DPU Core Count":1,
        "IP version":"v4.1.0",
        "generation timestamp":"2023-02-21 21-30-00",
        "git commit id":"7d32c41",
        "git commit time":2023022121,
        "regmap":"1to1 version"
    },
    "VAI Version":{
        "libvart-runner.so":"Xilinx vart-runner Version: 3.0.0-
c5d2bd43d951c174185d728b8e5bcda3869e0b39  2023-12-23-13:14:25 ",
        "libvitis_ai_library-dpu_task.so":"Xilinx vitis_ai_library
dpu_task Version: 3.0.0-c5d2bd43d951c174185d728b8e5bcda3869e0b39  2023-
01-13 06:58:30 [UTC] ",
        "libxir.so":"Xilinx xir Version: xir-
c5d2bd43d951c174185d728b8e5bcda3869e0b39 2023-12-23-13:13:09",
```

```
        "target_factory":"target-factory.3.0.0
c5d2bd43d951c174185d728b8e5bcda3869e0b39"
    },
    "kernels":[
        {
            "DPU Arch":"DPUCZDX8G_ISA1_B512",
            "DPU Frequency (MHz)":300,
            "IP Type":"DPU",
            "Load Parallel":2,
            "Load augmentation":"enable",
            "Load minus mean":"disable",
            "Save Parallel":2,
            "XRT Frequency (MHz)":300,
            "cu_addr":"0xa0010000",
            "cu_handle":"0xaaaadd524910",
            "cu_idx":0,
            "cu_mask":1,
            "cu_name":"DPUCZDX8G:DPUCZDX8G_1",
            "device_id":0,
            "fingerprint":"0x101000056010200",
            "name":"DPU Core 0"
        }
    ]
}
root@Trenz:~#
```


We can see one running DPU IP, version v4.1.0, with architecture `DPUCZDX8G_ISA1_B512`.

Vitis AI 3.0 examples require re-compilation of SW for the B512 DPU architecture and also require re-compilation of inference models for the B512 DPU architecture.

This process will be described in a separate application note [3].

6.4 VGA Display with Touch Screen

System with VGA display with touch screen GUI.

The touch interface can potentially replace mouse and keyboard in some application scenarios, if applications are designed to be selected by touch screen user interface.

6.5 Used Programmable Logic Resources

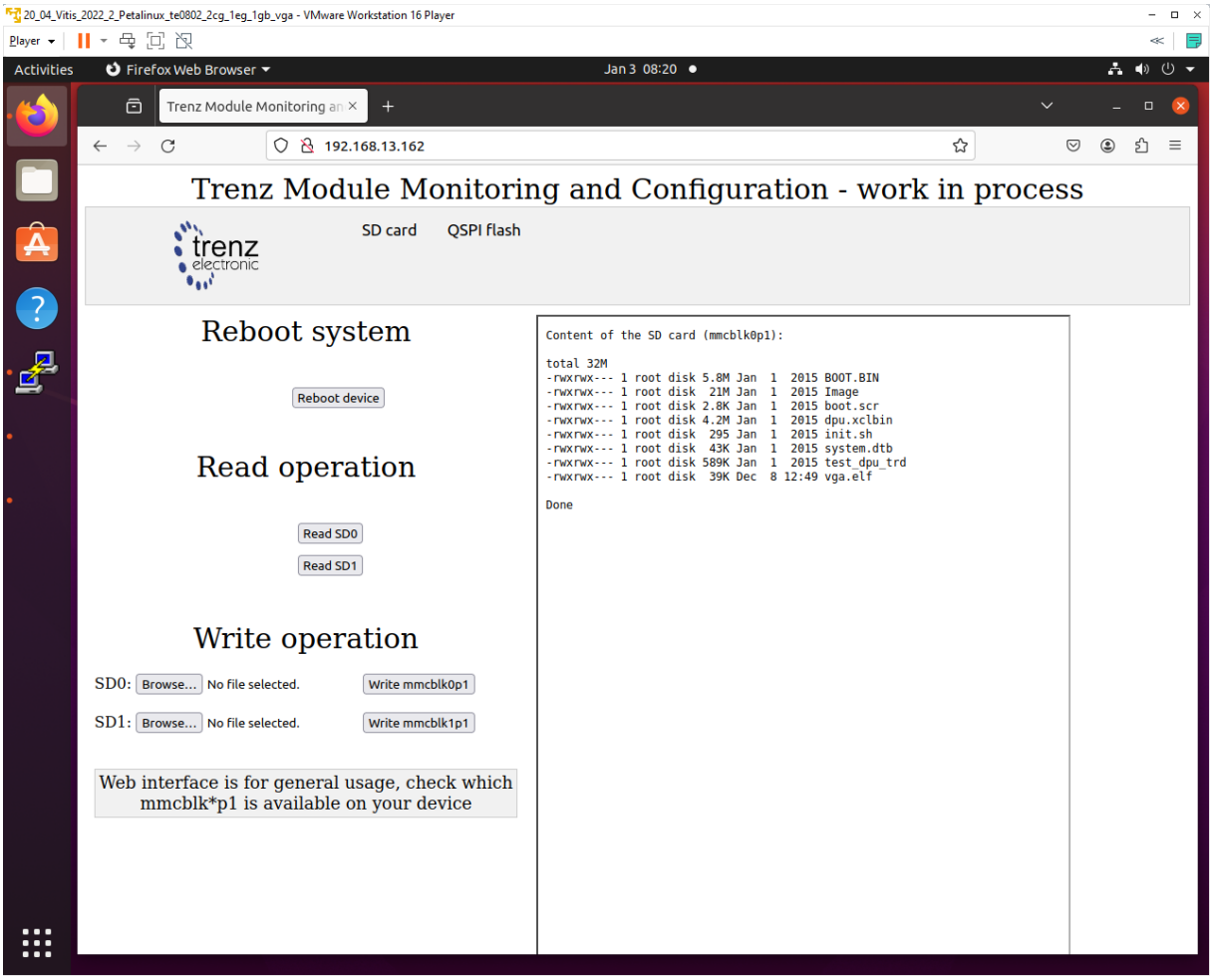
The design with integrated AMD DPUCZDX8G DPU in the configuration B512 and VGA display support is using these programmable logic resources of the ZU1EG device:

Resource	Used Instances [%]	Used Instances	Total Instances in ZU1EG
LUT	88.77	33234	37440
FF	62.56	46916	74880
BRAM	72.68	78.5	108
DSP	62.2	120	216

6.6 Remote Monitoring and Configuration Support

The configured OS includes work in progress version of a remote monitoring and configuration support server. It can be used for remote reading of content of the SD card partition `mmcblk0p1`.

Button `Reboot device` can be used for system reboot. Ethernet connection is lost, but remote PC www browser remains open and waits for possible reconnection.



After reboot of the evaluation board, the network DHCP server assigns Ethernet address to the evaluation board.

If the network DHCP address assignment algorithm assigns the identical Ethernet address, the page can be refreshed and the connection is re-established again.

If the network DHCP address assignment algorithm assigns different Ethernet address, the connection has to be established on the new Ethernet address.

6.7 Remote Control from Ubuntu X11 Desktop.

The configured OS also supports X11 desktop on remote PC via Ethernet.
In remote PC in Ubuntu OS, in `PuTTY` terminal utility with `ssh` Ethernet connection to the board with enabled `X11 forwarding`.

Opening.

Log in to the evaluation board as user `root` with pswd `root`
Start two `rxvt` terminal emulators by typing in `PuTTY` terminal:

```
rxvt &
rxvt &
```

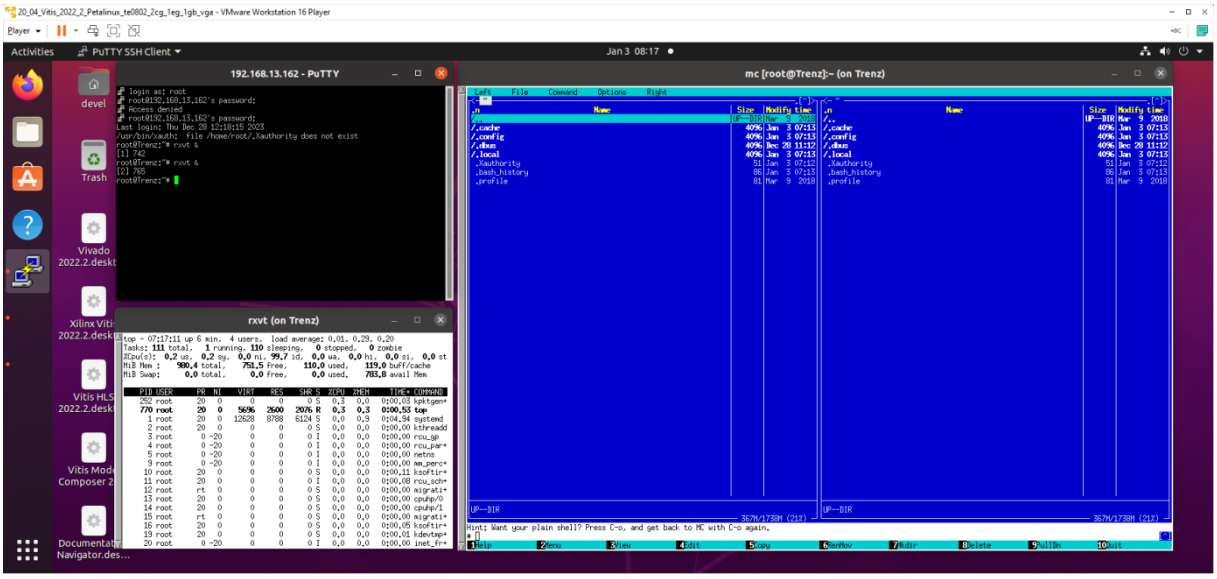
In first `rxvt` terminal emulator window start utility

```
top
```

In second `rxvt` terminal emulator start

```
mc
```

You can see two applications running on the evaluation board with output on the remote desktop. Remote PC kbd and mouse are used for control of these applications.



Closing.
On remote PC, close top utility by `Ctrl-C`. Stop mc utility by F10.
Close open terminal emulators by typing exit or by mouse click on x icon in the right top corner of terminal emulator window. Close `PuTTY` connection by typing exit or by mouse click on x icon in the right top corner of `PuTTY` window.

6.8 Remote Control in x-session-manager on Ubuntu X11 Desktop.

The configured OS also supports x-session-manager on X11 desktop on remote PC connected via Ethernet to the evaluation board.

Opening.
In remote PC in Ubuntu OS, start `PuTTY` terminal utility with ssh Ethernet connection to the board with enabled `X11 forwarding`.
Log in to the evaluation board as user `root` with pswd `root`
In PuTTY terminal, start `x-session-manager` by typing:

```
x-session-manager &
```

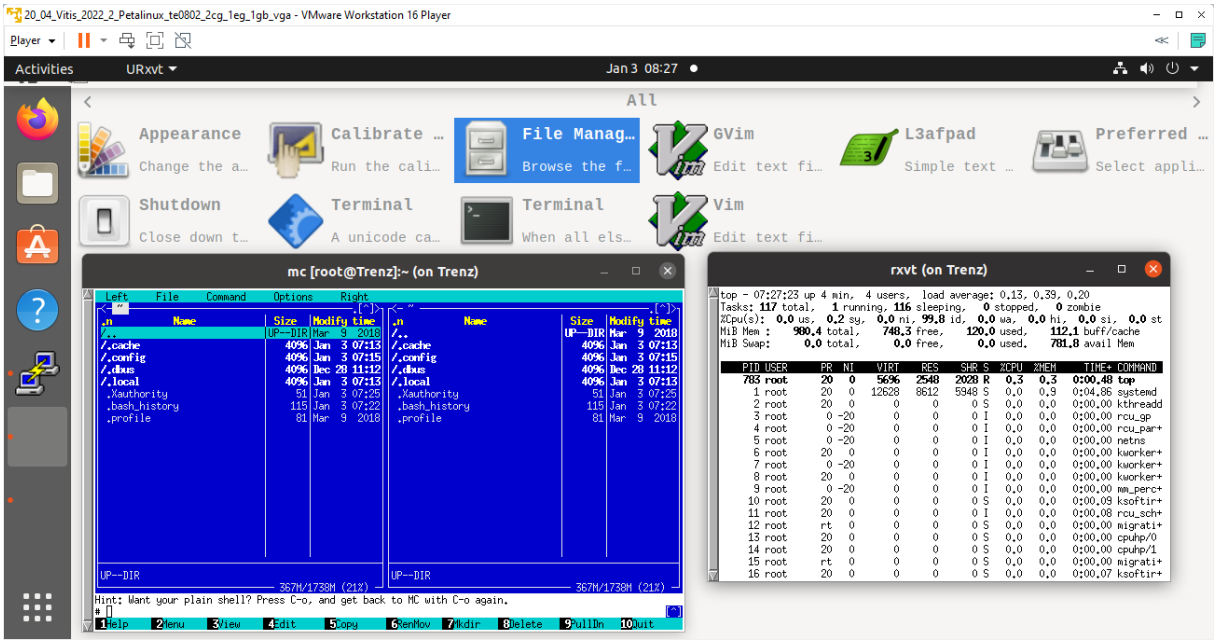
The desktop (displayed on the VGA display of the evaluation board) is also displayed in the remote PC X11 desktop. Start two `rxvt` terminal emulators by typing in `PuTTY` terminal:

```
rxvt &

rxvt &
```

In first `rxvt` terminal emulator window start utility
`top`
In second `rxvt` terminal emulator start
`mc`

You can see two applications running on the evaluation board with output on the remote desktop. Remote PC kbd and mouse are used for control of these applications.



Closing.
On remote PC, close top utility by `Ctrl-C`. Stop mc utility by key `F10`.
Close open terminal emulators by typing exit or by mouse click on x icon in the right top corner of terminal emulator window. Close `PuTTY` connection by typing exit or by mouse click on x icon in the right top corner of `PuTTY` window.

6.9 Display Test Pattern and Test USB Camera

Complete video chain can be tested with output to the X11 desktop on the local HD VGA display or with output to the remote X11 desktop.

To display the test pattern, use this `gstreamer` command:

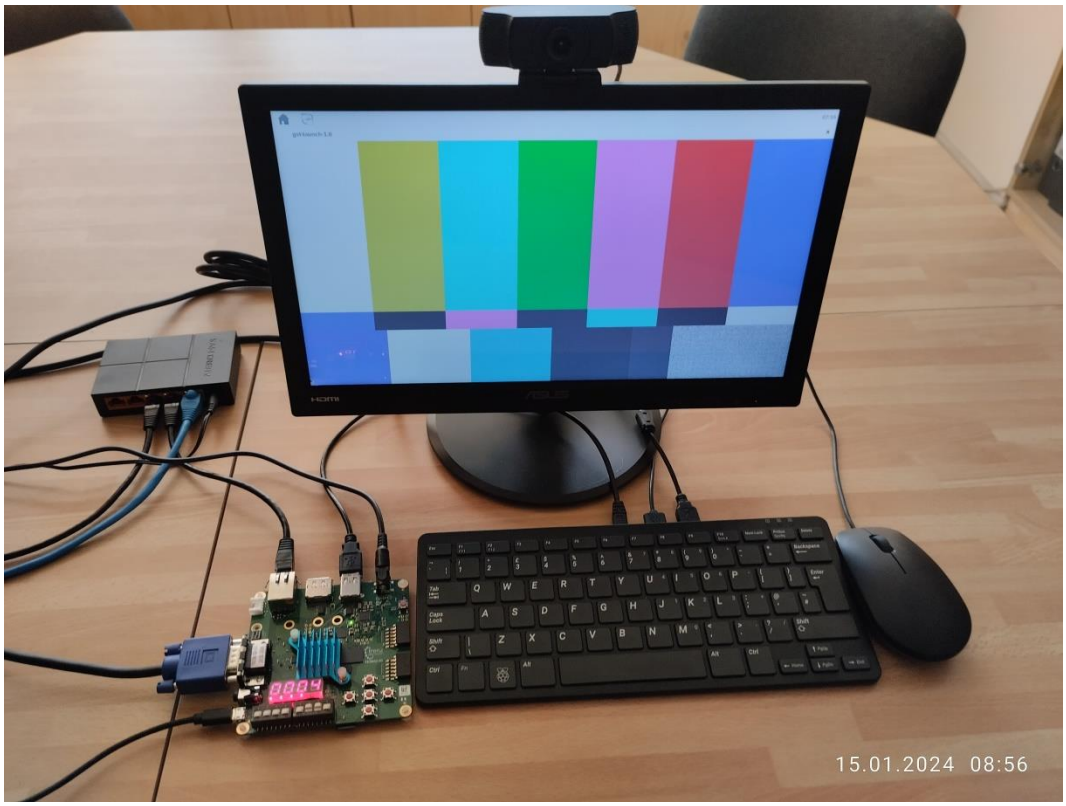
```
gst-launch-1.0 videotestsrc ! ximagesink
```

To display USB camera video, use this `gstreamer` command:

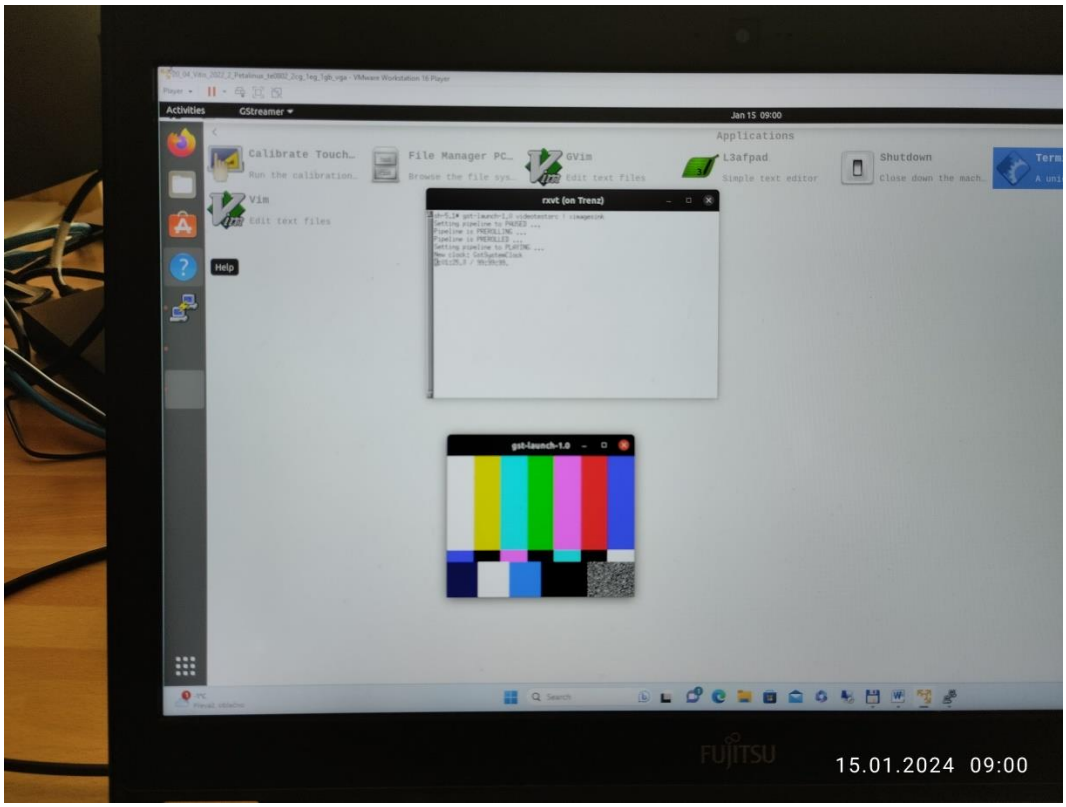
```
gst-launch-1.0 v4l2src device=/dev/video0 ! videoconvert ! ximagesink
```

Video output is directed to the local HD VGA display, if the command is started from local X11 console.

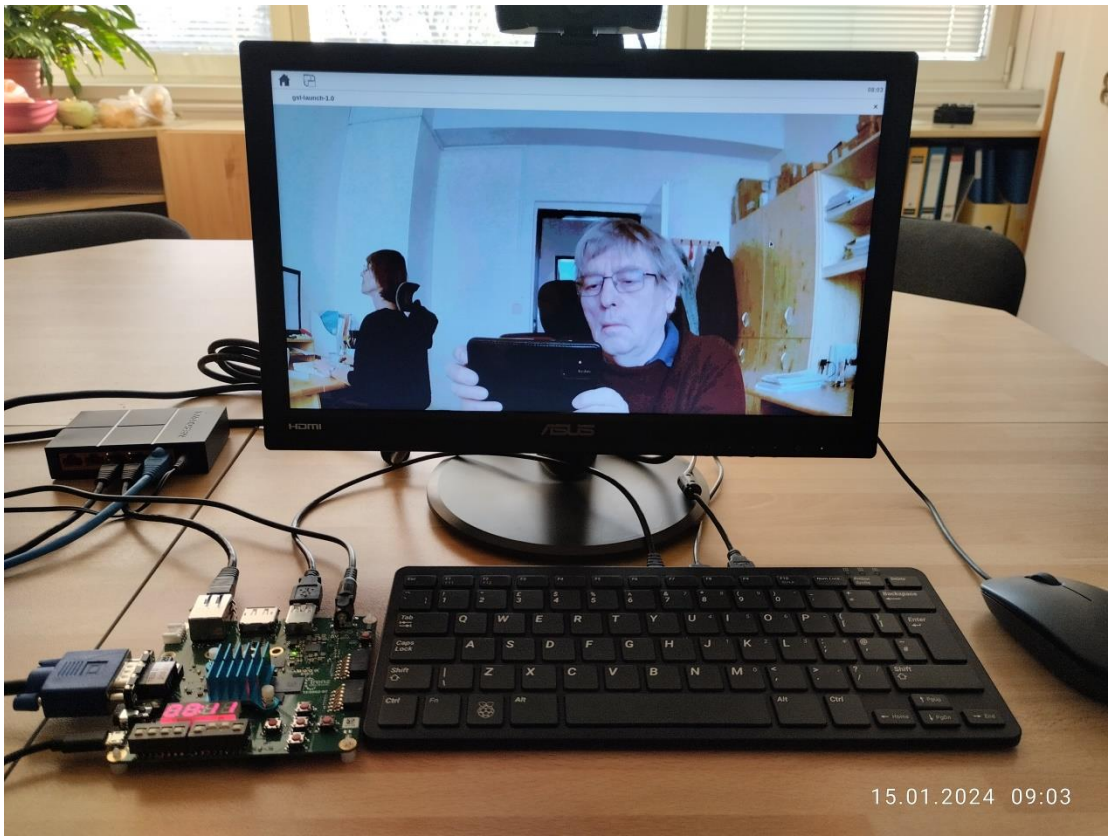
Video output is directed to the remote X11desktop, if the command is started from the remote X11 console.



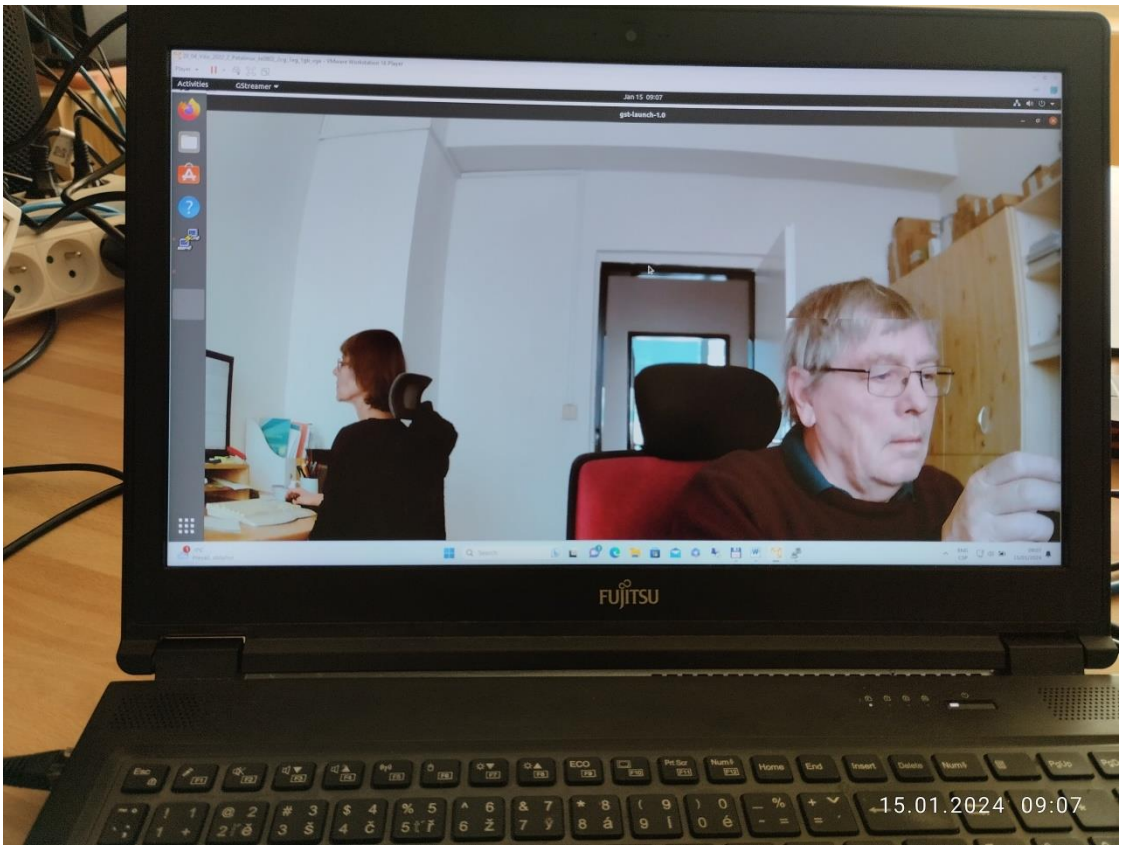
Test pattern is converted to local HD VGA RGB444 display



Test pattern is displayed on remote PC X11 desktop



Full HD video from USB camera is converted to local HD VGA RGB444 display.



Full HD video from USB camera is displayed as Full HD on remote PC X11 desktop.

6.10 Vitis AI 3.0 TE0802-02-1BEV2-A board, 1EG Device, DPU inB512 Configuration

Vitis AI 3.0 exampes	Performance with input from camera e2e [FPS]	Power with camera and VGA [W]	Performance with input from file e2e [FPS]	Power with input from file [W]	GigaOps with input from file e2e [Gops]
Face detection Model: pt_face-mask-detection_512_512_0.67G_3.0	17.3	6.9	30.9	6.2	20.7
Vehicle make Model: pt_vehicle-make-classification_VMMR_224_224_3.64G_3.0	18.9	7.4	22.3	6.7	81.1
Vehicle type Model: pt_vehicle-type-classification_CarBodyStyle_224_224_3.64G_3.0	19.1	7.4	22.3	6.7	81.1
Vehicle color Model: pt_vehicle-color-classification_VCoR_224_224_3.64G_3.0	18.9	7.4	22.3	6.7	81.1
General classification Model: pt_resnet50_imagenet_224_224_8.2G_3.0	8.1	7.2	9.3	6.5	76.2
General classification Model: pt_resnet50_imagenet_224_224_0.3_5.8G_3.0	9.2	7.1	11.1	6.5	64.3
General classification Model: pt_resnet50_imagenet_224_224_0.4_4.9G_3.0	9.8	7.1	12.2	6.5	59.7
General classification Model: pt_resnet50_imagenet_224_224_0.5_4.1G_3.0	10.2	7.0	12.9	6.4	52.8
General classification Model: pt_resnet50_imagenet_224_224_0.6_3.3G_3.0	11.4	7.0	14.9	6.4	49,1
General classification Model: pt_resnet50_imagenet_224_224_0.7_2.5G_3.0	11.8	6.9	16.2	6.3	40.5

Measurement conditions:

- TE0802-02-1BEV2-A board with ZU01EG device 1 GB DDR4
- DPU in B512 configuration
- USB WWW camera ETERNICO with sensor JX_F23, 1920x1080, max 20 FPS
- Keyboard RPi
- Mouse RPi
- VGA display (1280x720p60)
- Power supply 5V/4A
- Power measured at the 230V power plug

7 References

[1]
Support for STM32H573I-DK web server. (Application note, with evaluation package, UTIA). Not published, yet. To be published for public access from:
https://zs.utia.cas.cz/index.php?ids=projects/eecone
This application and evaluation package will be based on the STM32CubeH5 Firmware Examples for STM32H5xx Series Application based on NetXDuo: **Nx_WebServer.**
This STM application provides an example of Azure RTOS NetX Duo stack usage on STM32H573G-DK board, it shows how to develop Web HTTP server based application.
https://htmlpreview.github.io/?https://raw.githubusercontent.com/STMicroelectronics/STM32CubeH5/master/Projects/STM32CubeProjectsList.html

[2]
Lukáš Kohout, Jiří Kadlec, Zdeněk Pohl: Support for TE0802-02-1BEV2-A board with Vitis AI 3.0 DPU and VGA display (Application note with evaluation package, UTIA). Draft has been provided to partners for review 14.1.2024. **To be published 15.2.2024** for public free access from:
https://zs.utia.cas.cz/index.php?ids=projects/eecone

[3]
Lukáš Kohout, Jiří Kadlec, Zdeněk Pohl: Support for TE0802-02-2AEV2-A board with Vitis AI 3.0 DPU and VGA display (Application note, with evaluation package, UTIA). Draft has been provided to partners for review 14.1.2024. **To be published 15.2.2024** for public access from:
https://zs.utia.cas.cz/index.php?ids=projects/eecone

[4]
Support for module-based systems with TE0821 modules on TE0701 carrier board with Vitis AI 3.0 DPU (Application note, with evaluation package, UTIA). Not published, yet. To be published for free public access from:
https://zs.utia.cas.cz/index.php?ids=projects/eecone

[5]
Support for TE0820 modules with Vitis AI 3.0 DPU (Application note, with evaluation package, UTIA). Not published, yet. To be published for free public access from:
https://zs.utia.cas.cz/index.php?ids=projects/eecone

[6]
Description of compilation of Vitis AI 3.0 models for different configurations of AMD DPUs, (Application note, with evaluation package, UTIA). Not published, yet. To be published for free public access from:
https://zs.utia.cas.cz/index.php?ids=projects/eecone

[7]
TE0802-02-1AEV2-A test board Vitis AI Tutorial
(Vitis 2021.2.1, Vitis AI 2.0, no VGA display support)
https://wiki.trenz-electronic.de/display/PD/TE0802-02-1AEV2-A+test+board+Vitis+AI+Tutorial

[8]
TE0802-02-1AEV2-A test board Vitis AI Tutorial
(Vitis 2021.2.1, Vitis AI 2.0, no VGA display support)
https://wiki.trenz-electronic.de/display/PD/TE0802-02-1AEV2-A+test+board+Vitis+AI+Tutorial

[9]
TE0802-02-1AEV2-A test board Vitis AI Tutorial
(Vitis 2021.2.1, Vitis AI 2.0, no VGA display support)

https://wiki.trenz-electronic.de/display/PD/TE0802-02-1AEV2-A+test+board+Vitis+AI+Tutorial

[10] MPSoC Development Board with AMD Zynq™ UltraScale+™ ZU2CG and 1 GB LPDDR4, Trenz Electronic. https://shop.trenz-electronic.de/en/TE0802-02-1BEV2-A-MPSoC-Development-Board-with-AMD-Zynq-UltraScale-ZU2CG-and-1-GB-LPDDR4?c=474

[11] TE0821 test board Vitis AI Tutorial, v19, prepared by UTIA and published by Trenz Electronics 2.1.2023 (before EECONE start). https://wiki.trenz-electronic.de/display/PD/TE0821+test+board+Vitis+AI+Tutorial

[12] TE0821 Resources, TE0821 - 4 x 5 cm SoC module with Xilinx Zynq UltraScale+, prepared and published by Trenz Electronics https://wiki.trenz-electronic.de/display/PD/TE0821+Resources

[13] TE0820 test board Vitis AI 3.0 Tutorial, v25, has been prepared by UTIA (before EECONE start) for Trenz Electronics and it has been published by Trenz Electronics 31.8.2023. https://wiki.trenz-electronic.de/display/PD/TE0820+test+board+Vitis+AI+3.0+Tutorial

[14] TE0820 - 4 x 5 cm SoC module with AMD Zynq™ UltraScale+™ , prepared and published by Trenz Electronics https://wiki.trenz-electronic.de/display/PD/TE0820+Resources

[15] Lukáš Kohout, Zdeněk Pohl and Jiří Kadlec: Xilinx Vitis AI facedetect Demo on Trenz Electronic TE0820 4EV SoM with TE0701 06 Carrier Board and Avnet HDMI In/Out FMC Card; ; (Application note, with evaluation package, UTIA). Published 7.2.2023, (before EECONE start, with acknowledgement to project StorAIge No. 101007321). https://zs.utia.cas.cz/index.php?ids=results&id=te0820_4ev_vitis_ai_vcu

[16] Lukáš Kohout, Zdeněk Pohl and Jiří Kadlec: Xilinx Vitis AI facedetect and resnet50 Demo on Trenz Electronic TE0802 02 with ZU2CG and 1 GB LPDD4; Published 16.2.2023, (before EECONE start, with acknowledgement to project StorAIge No. 101007321). https://zs.utia.cas.cz/index.php?ids=results&id=te0802_2cg_vitis_ai_resnet50

[17] Zdeněk Pohl, Lukáš Kohout, Jiří Kadlec: Xilinx Vitis AI 'facedetect' and 'resnet50' Demo on Trenz Electronic TE0821-01-2cg-4GB SoM + TE0706-3 Carrier; (Application note, with evaluation package, UTIA). Published 18.1.2023, (before EECONE start, with acknowledgement to project StorAIge No. 101007321). https://zs.utia.cas.cz/index.php?ids=results&id=te0821_2cg_vitis_ai