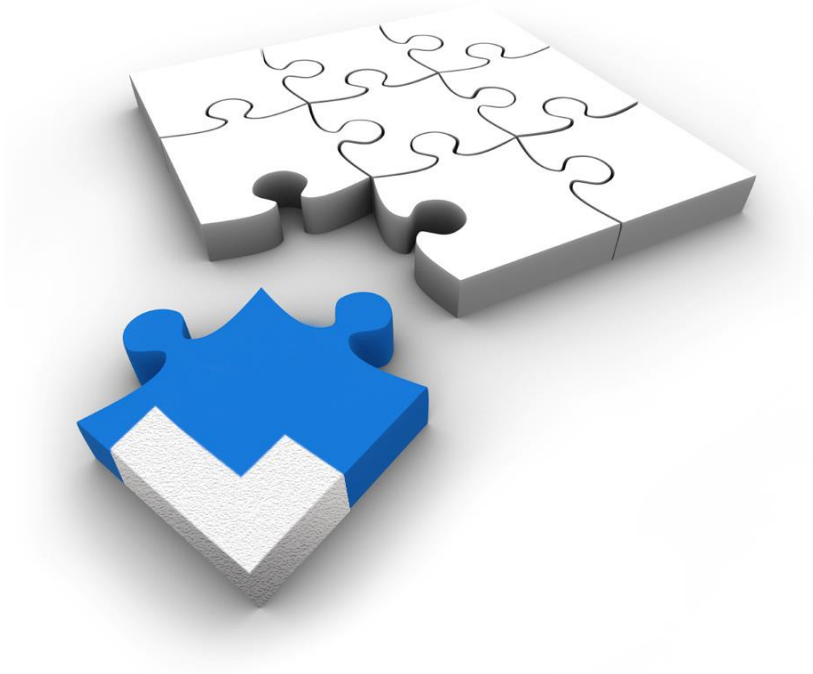
logiADAK-VDF-ZU

Video Design Framework - Reference Designs for Xylon logiVID-ZU Vision Development Kit

User‘s Manual

Version: 4.0.1

logiADAK_VDF_ZU_v4.0.1.docx



1 ABOUT THE FRAMEWORK

The logiADAK-VDF-ZU Video Design Framework enables Xylon logiVID-ZU Vision Development Kit users to quickly utilize the provided hardware platform for their own development of the Xilinx® All Programmable Zynq® UltraScale+™ MPSoC based embedded multi-camera vision systems.

The framework includes pre-verified logicBRICKS reference designs for video capture and the display output under the Linux operating system. Video capture can be either from:

- Xylon video cameras with the Maxim Integrated **GMSL2** high-speed digital video interface (GMSL2 version; see Figure 1) or

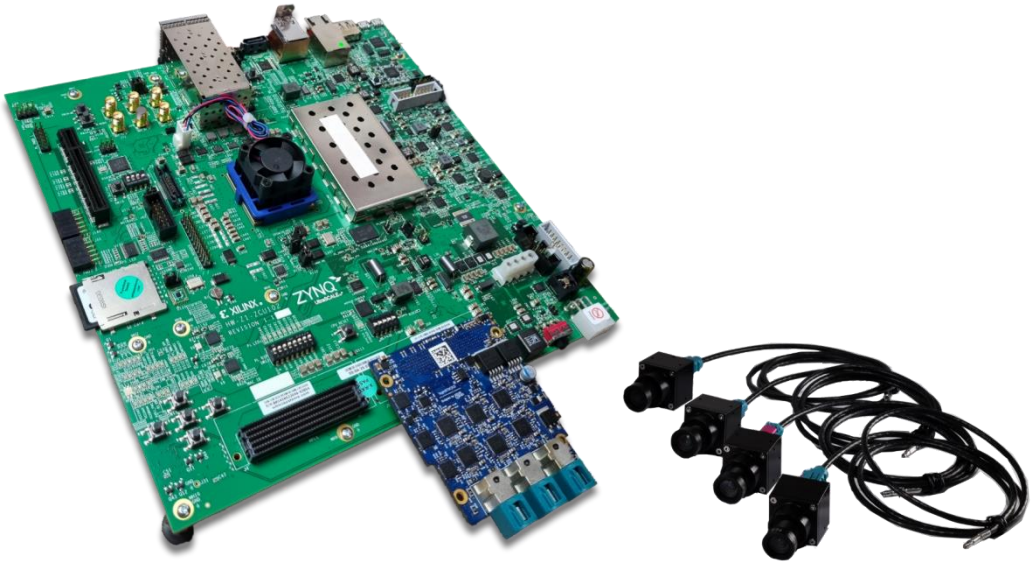


Figure 1: Xylon logiADAK-VDF-ZU Development Kit – GMSL2 Version

- Xylon video cameras with the TI® **FPD-Link III** high-speed digital video interface (FPD-Link III version; see Figure 2).

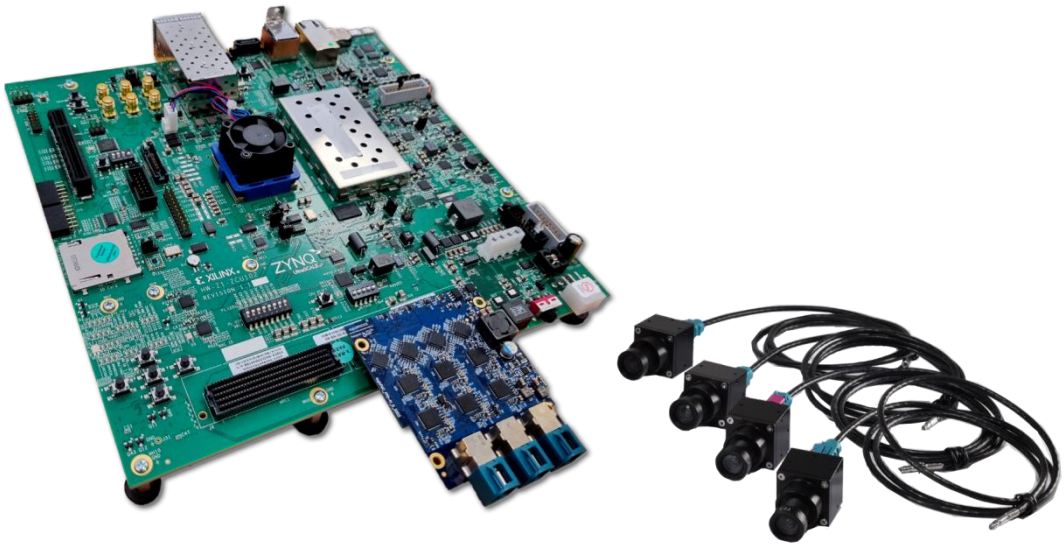


Figure 2: Xylon logiADAK-VDF-ZU Development Kit – FPD-Link III Version

Reference designs are prepared for hardware-centric Vivado® Design Suite and software is implemented in Vitis Unified Software Platform.

The complete camera-to-display MPSoC designs, which are compact and use just a fraction of available programmable logic, significantly save the design time. Instead of starting from scratch and having to spend months designing and building a new design framework, the logiADAK-VDF-ZU design framework users can immediately focus on specific vision-based parts of their next MPSoC design. The logiVID-ZU hardware platform can be installed on test vehicles (cars, robots...) and used in exhaustive tests, i.e. for testing of the new ADAS developments in the test vehicle and under different road conditions.

logiADAK-VDF-ZU reference designs include Xylon logicBRICKS IP cores and hardware design files prepared for Xilinx Vivado® Design Suite. Hardware designers can customize designs and add their own IP cores through the Vivado IP Integrator (IPI).

To provide the Vitis users the complete embedded C/C++ development experience, the supplied reference design includes the Sobel video filter example. This example shows kit users how to integrate their own vision processing logic between video input and video output IP cores, and how to implement it in programmable logic.

The Linux OS and software drivers for logicBRICKS IP cores enable software developers to efficiently work with the framework, without knowing the hardware implementation details.

The hardware designs are the same for both versions; GMSL2 and FPD-Link III. However there are changes to software between the versions. These changes will be described in the following chapters.

1.1 Programmable Logic Utilization

The logiADAK-VDF-ZU reference designs utilize just small fractions (Table 3) of available programmable logic resources in the Xilinx Zynq UltraScale+ MPSoC XCZU9EG device. Free resources can be utilized by users who can also alter the pre-defined logicBRICKS configurations and change the programmable logic utilization.

Table 1: FOUR-CAM Reference Design Programmable Logic Utilization

Family (Device)	F (MHz)			LUT[1]	LUTRAM[1]	FF[1]	IOB[2]	BRAM[1]	MULT/ DSP48/E	PLL / MMCM	BUFG	GTx	Design Tools
	mclk[4]	rclk	clk[5]										
Zynq UltraScale+[3] (XCZU9EG-2)	200	100	150/300	37833	4228	48710	38	58.5	137	1/3	23	3	Vivado 2021.1

Notes:

1) Assuming the following configuration: AXI Stream, YUV output, 32-bit AXI4-Lite register interface, 64-bit AXI4 memory interface with max. burst size of 64 words, scaling in both directions with multipliers (DSP48s), output stride set to 2048 pixels, logiCVC FIFO size multiplication factor 1x, MIPI RX Line Buffer Depth 256.
2) Assuming only video inputs are routed off-chip, register and memory interfaces are connected internally
3) Only burst size of 16 words is supported on HP ports in the Xilinx Zynq UltraScale+ MPSoC
4) logiWIN clock frequencies
5) Default/Additional clock for Vitis acceleration flow

Table 2: Free Programmable Logic Resources

	Available in XCZU9EG	FOUR-CAM
Look-Up Tables (LUTs)	274,080	~ 14%
Look-Up Tables as Memory (LUTRAM)	144, 000	~ 3%
Flip Flops (FFs)	548,160	~ 9%
Block RAM (18/36 kB BRAM)	912	~ 6%
DSP slices (MULT/DSP)	2,520	~ 5%

1.2 About Vitis Unified Software Platform

1.3 Hardware Requirements

The logiVID-ZU Vision Development Kit (Figure 1) includes the following hardware, which is utilized by reference designs provided in the logiADAK-VDF-ZU design framework:

- **logiVID-ZU-GMSL2**
  - 1x Xilinx Zynq UltraScale+ MPSoC ZCU102 Evaluation Kit (Manufacturer part No.: EK-U1-ZCU102-G)
    - Kit package contains:
      - ZCU102 Evaluation Board (Manufacturer part No.: HW-Z1-ZCU102 Rev. 1.1)
      - Power supply
      - Xilinx license slip
  - 1x Xylon GMSL2 FMC daughter card (part no.: logiFMC-GMSL2-9296A-12C)
  - 4x Xylon GMSL2 video camera (part no.: logiCAM-GMSL2-AR0231-05525FM)
    - Camera package contains:
      - Xylon video camera
      - FAKRA cable assembly (5m)
  - 2x Xylon 4 HFM Rosenberger connector adapter (part no: logiFMC-CBL-4HFM)
  - 1x SD card
    - High/Extreme Capacity, High/Ultra Speed

- **logiVID-ZU-FPD3**
  - 1x Xilinx Zynq UltraScale+ MPSoC ZCU102 Evaluation Kit (Manufacturer part No.: EK-U1-ZCU102-G)
    - Kit package contains:
      - ZCU102 Evaluation Board (Manufacturer part No.: HW-Z1-ZCU102 Rev. 1.1)
      - Power supply
      - Xilinx license slip
  - 1x Xylon FPD3 FMC daughter card (part no.: logiFMC-FPD3-954-12C)
  - 4x Xylon FPD3 video camera (part no.: logiCAM-FPD3-AR0231-05525FM)
    - Camera package contains:
      - Xylon video camera
      - FAKRA cable assembly (5m)
  - 1x Xylon 4 HFM Rosenberger connector adapter (part no: logiFMC-CBL-4HFM)
  - 1x SD card
    - High/Extreme Capacity, High/Ultra Speed

- **Table 3: Included Hardware**

GMSL2 version - logiVID-ZU-GMSL2[2]	FPD-Link III version - logiVID-ZU-FPD3[2]
1x Xilinx Zynq UltraScale+ MPSoC ZCU102 Evaluation Kit[1] (ver 1.1) with the XCZU9EG-FFVB1156-2 device	
1x Xylon GMSL2 Deserializer FMC Module (Product code: logiFMC-GMSL2-9296A-12C)	1x Xylon FPD-LINK III Deserializer FMC Module (Product code: logiFMC-FPD3-954-12C)
4x 2.3-Mpix Xylon GMSL2 Cameras	4x 2.3-Mpix Xylon FPD-LINK III Cameras
1x SD card	
4x FAKRA cable assemblies	
Power supply	

[1] – OEM kit version without the Xilinx Vivado Design Suite seat
[2] – logiVID-ZU is delivered in GMSL2 or FPD-Link III version

1.3.1 GMSL2 Deserializer FMC Module

The FOUR-CAM reference designs GMSL2 version delivered with the logiADAK-VDF-ZU require Xylon GMSL2 Deserializer FMC module (Figure 3), which comes as a part of the hardware kit deliverables (GMSL2 version).

Figure 3: Xylon GMSL2 Deserializer FMC Module

1.3.2 FPD-LINK III Deserializer FMC Module

The FOUR-CAM reference design FPD-Link III version delivered with the logiADAK-VDF-ZU require Xylon's FPD-Link III Deserializer FMC module (Figure 4), which comes as a part of the hardware kit deliverables (FPD-Link III version).

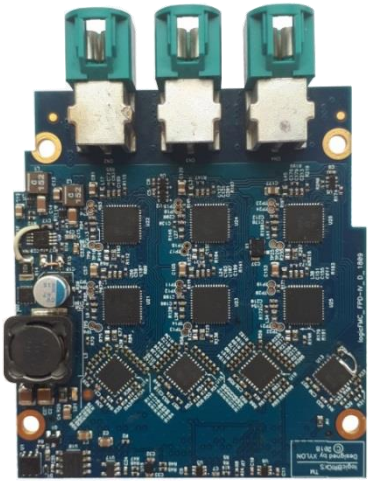


Figure 4: Xylon FPD Link III Deserializer FMC Module

1.3.3 Xylon GMSL2 Camera

For transmissions of high-definition uncompressed video and camera control data the logiVID-ZU development kit includes Xylon cameras compatible with the Maxim Integrated GMSL2 high-speed digital video interface. Each camera includes the ON Semiconductor AR0231 2.3-megapixel camera sensor that combines high-definition (HD) 1928x1208p30 Full HD video with the color high dynamic range (HDR) functionality, GMSL2 serializer (transmitter) board, short cable lead with a connector and FIFO Optics 05525FM narrow-angle lens.

All camera parts are enclosed in the aluminium housing designed by Xylon. Its rugged metal construction provides excellent lens and imager module protection and enables safe and easy test vehicle installations. The aluminium housing used is the same for the GMSL2 and FPD-Link III camera. The same camera cable is used for power, control data and Full HD video transmissions.

1.3.4 Xylon FPD-Link III Camera

For transmissions of high-definition uncompressed video and camera control data the logiVID-ZU development kit includes Xylon cameras compatible with the TI FPD-Link III high-speed digital video interface. Each camera includes the On Semiconductor AR0231 2.3-megapixel camera sensor that combines high-definition (HD) video with the color high dynamic range (HDR) functionality, image Co-

Processor, FPD-Link III serializer (transmitter) board, the FIFO Optics 05525FM narrow-angle lens and a short cable lead with a connector. The same camera cable is used for power, control data and HD video transmissions. Resolution of Xylon FPD-Link III camera is 1928x1208p30.

Figure 5. logiCAM with the FIFO Optics 05525FM lens

1.4 Software Requirements

The logiADAK-VDF-ZU reference designs and Xylon logicBRICKS IP cores are fully compatible with Xilinx development tools – Vitis Unified Software Platform 2021.1 and PetaLinux 2021.1. Future design releases shall be synchronized with the newest Xilinx development tools.

1.5 Design Deliverables

1.5.1 Vitis platform

- Reference design prepared for Vitis Unified Software Platform
- Supports Linux applications
- Vivado reference design that allows for instant design check-up and Vitis workspace for quick software changes
- Xylon evaluation logicBRICKS IP cores:
  - logiCVC-ML Compact Multilayer Video Controller
  - logiWIN Versatile Video Input
- Xylon IP Cores that are not sold separately but only serve to augment this specific reference design:
  - tUser-Trimmer

1.5.2 Software

- logiVIOF VideoIn-VideoOut Library
- Demo application sources
- Includes Linux kernel drivers for included logicBRICKS IP cores

1.5.3 Binaries

- Precompiled SD card image for the fastest demo startup:
  - boot.scr
  - boot.bin
    - First Stage Boot Loader
    - Universal Boot Loader
    - FPGA
    - Platform Management Unit Firmware
  - image.ub
    - kernel image
    - device tree blob
    - minimal Root File System
  - Four Camera demo
  - Sobel filter kernel binary (xclbin)

1.6 Reference Design

The logiADAK-VDF-ZU video design framework includes FOUR-CAM reference design which implements four parallel video inputs from Xylon cameras, implementation of Sobel filter using Vitis accelerated libraries and the display output. All video inputs are stored in the video memory, and by mean of the on-board push buttons, the user can select each of them for the single camera, all cameras or Sobel visualization full screen display output.

Reference design contains the Sobel filter as the example of the C-code based hardware accelerator (from Vitis accelerated libraries) implemented by Vitis Unified Software Platform. The Sobel filter example shows kit users how to integrate their own vision processing logic between video input and video output IP cores, and how to implement in programmable logic.

2 LOGICBRICKS IP CORES

2.1 About logicBRICKS IP Library

Xylon's logicBRICKS IP core library provides IP cores optimized for Xilinx All Programmable FPGA and SoC devices. The logicBRICKS IP cores shorten development time and enable fast design of complex embedded systems based on Xilinx All Programmable devices.

The key features of the logicBRICKS IP cores are:

- logicBRICKS can be used in the same ways as Xilinx IP cores within the Xilinx Vivado Design Suite, and require no skills beyond general tools knowledge. IP users setup feature sets and programmable logic utilization through implementation tools' Graphical User Interface (GUI).
- Each logicBRICKS IP core comes with the extensive documentation, reference design examples and can be evaluated on reference hardware platforms. Xylon provides evaluation logicBRICKS IP cores to enable risk-free evaluation prior to purchase.
- Broad software support – from bare-metal software drivers to standard software drivers for different operating systems (OS). Standard software support allows graphics designers and software developers to use logicBRICKS in a familiar and comfortable way.
- Xylon assures skilled technical support.

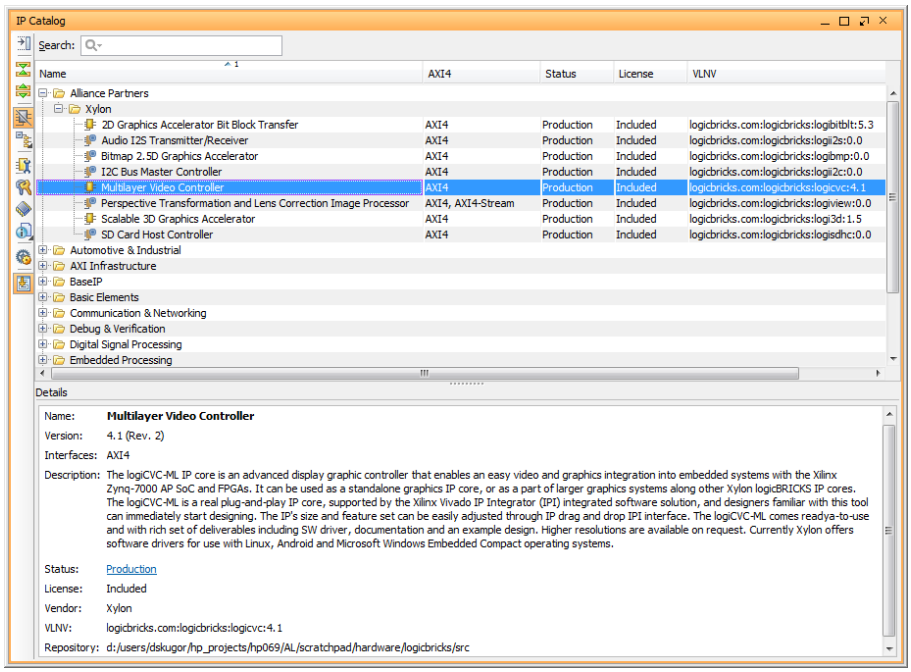


Figure 6: logicBRICKS IP Cores Imported into the Vivado IP Catalog

The Figure 6 shows logicBRICKS IP cores imported into Vivado Design Suite, while the Figure 7 shows a typical logicBRICKS IP core's configuration GUI.

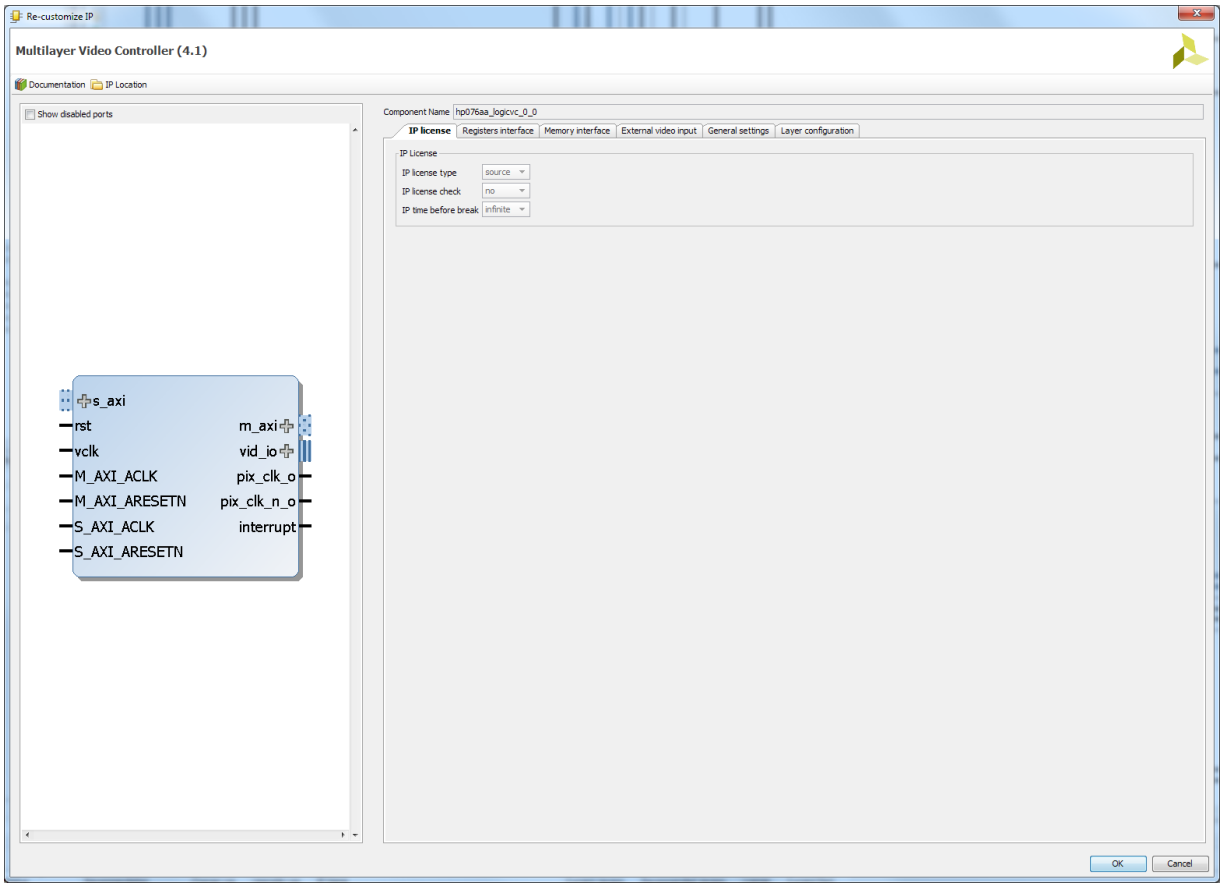


Figure 7: Example of logicBRICKS IP Configuration GUI

To access logicBRICKS IP cores' User's Manuals, double-click on the specific IP core's icon, and then the Documentation icon in the opened IP configuration GUI. Choose either the Product guide to open the manual, or the Change Log to open IP core's change log.

logicBRICKS User's Manuals contain all necessary information about the IP cores' features, architecture, registers, modes of operation, etc.

2.2 Evaluation logicBRICKS IP Cores

Xylon offers free evaluation logicBRICKS IP cores which enable full hardware evaluation:

- Import into the Xilinx Vivado tools (IP Integration)
- IP parameterization through the GUI interface
- Simulation (if Xilinx tools support it)
- Bitstream generation

The logicBRICKS evaluation IP cores are run-time limited and cease to function after some time. Proper operation can be restored by reloading the bitstream. Besides this run-time limitation, there are no other functional differences between the evaluation and fully licensed logicBRICKS IP cores.

Evaluation logicBRICKS IP cores are distributed as parts of the Xylon reference designs: http://www.logicbricks.com/logicBRICKS/Reference-logicBRICKS-Design.aspx.

Specific IP cores can be downloaded from Xylon's web shop: http://www.logicbricks.com/Products/IP-Cores.aspx.

2.3 logicBRICKS IP Cores Used in This Design

2.3.1 logiCVC-ML Compact Multilayer Video Controller

The logiCVC-ML IP core is an advanced display graphics controller for LCD and CRT displays, which enables an easy video and graphics integration into embedded systems with Xilinx Zynq-7000 All Programmable SoC and FPGAs.

This IP core is the cornerstone of all 2D and 3D GPUs. Though its main function is to provide flexible display control, it also includes hardware acceleration functions: three types of alpha blending, panning, buffering of multiple frames, etc.

- Supports all Xilinx FPGA families
- Supports LCD and CRT displays (easily tailored for special display types)
- Display resolutions up to 8192x8129 (including 4K2K resolution)
- Available SW drivers for: Linux and Microsoft Windows Embedded Compact OS
- Support for higher display resolutions available on request
- Supports up to 5 layers; the last one configurable as a background layer
- Configurable layers' size, position and offset
- Alpha blending and Color keyed transparency
- Pixel, layer, or Color Lookup Table (CLUT) alpha blending mode can be independently set for each layer
- Packed pixel layer memory organization:
  - RGB – 8bpp, 8bpp using CLUT, 16bpp 5-6-5, 24bpp 8-8-8 and 30bpp 10-10-10
  - YCbCr – 16bpp (4:2:2), 20bpp (4:2:2), 24bpp (4:4:4), 30bpp (4:4:4)
- Configurable ARM® AMBA® AXI4 memory interface data width (32, 64, 128 or 256 bits)
- Programmable layer memory base address and stride
- Simple programming due to small number of control registers
- Support for multiple output formats:
  - Parallel display data bus (RGB or YCrCb) with 1, 2 or 4 pix per clock: 12x2-bit, 15, 16, 18, 20, 24, 30 bit
  - Digital Video ITU-656: PAL and NTSC
  - LVDS output format: 3 or 4 data pairs plus clock
  - Camera link output format: 4 data pairs plus clock
  - DVI output format (currently not supported in US and US+ devices)
- Supports synchronization to external parallel input

- Versatile and programmable sync signals timing
- Double/triple buffering enables flicker-free reproduction
- Display power-on sequencing control signals
- Parametrical VHDL design that allows tuning of slice consumption and features set
- Prepared for Xilinx Vivado tools

More info: http://www.logicbricks.com/Products/logiCVC-ML.aspx
Datasheet: http://www.logicbricks.com/Documentation/Datasheets/IP/logiCVC-ML_hds.pdf

2.3.2 logiWIN Versatile Video Input

The logiWIN IP core enables easy implementation of video frame grabbers. Input video can be decoded, real-time scaled, de-interlaced, cropped, anti-aliased, positioned on the screen... Multiple logiWIN instances enable processing of multiple video inputs by a single Xilinx device.

- Supports versatile digital video input formats:
  - ITU656 and ITU1120 (PAL and NTSC)
  - RGB
  - YUV 4:2:2
- Maximum input and output resolutions are 2048 x 2048 pixels
- Built-in YcrCb to RGB converter, YUV to RGB converter and RGB to YcrCb converter
- Embedded image color enhancements: contrast, saturation, brightness and hue for ITU and YUV separately
- Real-time video scale-up (zoom in) up to 64x
- Real-time video scale-down (zoom out) down to 16 times
  - Lossless scaling down to 2x, or 4x in cascade scaling mode
- Supports video input cropping and smooth image positioning
- Configurable register interface; ARM® AMBA® AXI4-Lite
- ARM® AMBA® AXI4 and AXI4-Lite bus compliant
- Compressed stencil buffer in BRAM (mask over output buffer)
- Supports pixel alpha blending
- Provides "Bob" and "Weave" de-interlacing algorithms
- Supported big and little Endianness memory layout
- Double or triple buffering for flicker-free video
- Prepared for Xilinx Vivado tools

More info: https://www.logicbricks.com/Products/logiWIN.aspx
Datasheet: https://www.logicbricks.com/Documentation/Datasheets/IP/logiWIN_hds.pdf

2.4 logicBRICKS IP Cores for Video Processing

Xylon offers several logicBRICKS IP cores for video processing on Xilinx All Programmable FPGA, SoC and MPSoC devices:

logiVIEW Perspective Transformation and Lens Correction Image Processor

Removes fish-eye lens distortions and executes programmable transformations on multiple video inputs in real time. Programmable homographic transformation enables: cropping, resizing, rotating, transiting and arbitrary combinations. Arbitrary non-homographic transformations are supported by programmable Memory Look-Up Tables (MLUT).

More info: http://www.logicbricks.com/Products/logiVIEW.aspx
Datasheet: http://www.logicbricks.com/Documentation/Datasheets/IP/logiVIEW_hds.pdf

logiISP Image Signal Processing (ISP) Pipeline

The logiISP Image Signal Processing Pipeline IP core is a full high-definition ISP pipeline designed for digital processing and image quality enhancements of an input video stream in Smarter Vision embedded designs based on Xilinx All Programmable devices.

More info: http://www.logicbricks.com/Products/logiISP.aspx
Datasheet: http://www.logicbricks.com/Documentation/Datasheets/IP/logiISP_hds.pdf

logiHDR High Dynamic Range (HDR) Pipeline

Ultra-High Definition (UHD, including 4K2Kp60) HDR pipeline for camera image quality enhancements. Enables extraction of the maximum detail from high-contrast scenes, i.e. scenes with objects highlighted by a direct sunlight and objects placed in extreme shades.

More info: http://www.logicbricks.com/Products/logiHDR.aspx
Datasheet: http://www.logicbricks.com/Documentation/Datasheets/IP/logiHDR_hds.pdf

3 GET AND INSTALL THE DESIGN FRAMEWORK

Customers entitled for the logiADAK-VDF-ZU installation package delivery get the unique FTP or web download account from Xylon. To purchase the design framework, please visit our online catalog: http://www.logicbricks.com/Products/logiADAK-VDF-ZU.aspx

3.1 Installation Process

Installation process is quick and easy. The logiADAK-VDF-ZU framework can be downloaded as a cross-platform Java JAR self-extracting installer. Please make sure that you have a copy of the JRE (Java Runtime Environment) version 6 or higher on your system to run Java applications and applets. Double-click on the installer's icon to run the installation.

At the beginning, you will be requested to accept the design framework license – Figure 8. For installation in Linux OS, please follow instructions:

http://www.logicbricks.com/logicBRICKS/Reference-logicBRICKS-Design/Xylon-Reference-Designs-Linux-Installation.aspx.

If you agree with the conditions from the Xylon license, click NEXT and select the installation path for your logicBRICKS reference design (Figure 9). The installation process takes several minutes. It generates the folder structure described in the paragraph 3.1.1 Folder Structure.

Figure 8: Installation Process – Step 1

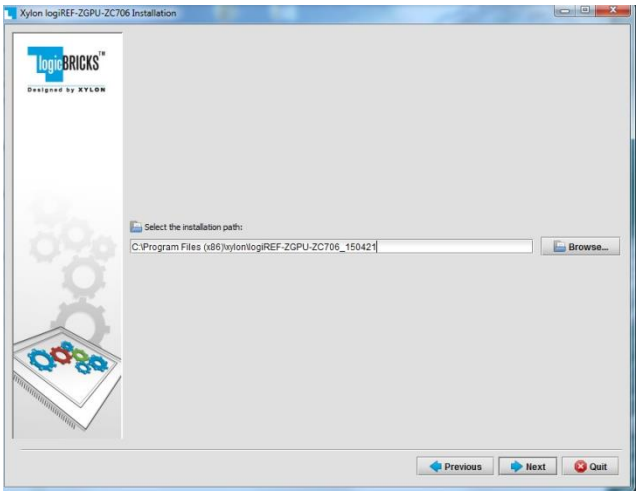


Figure 9: Installation Process – Step 2

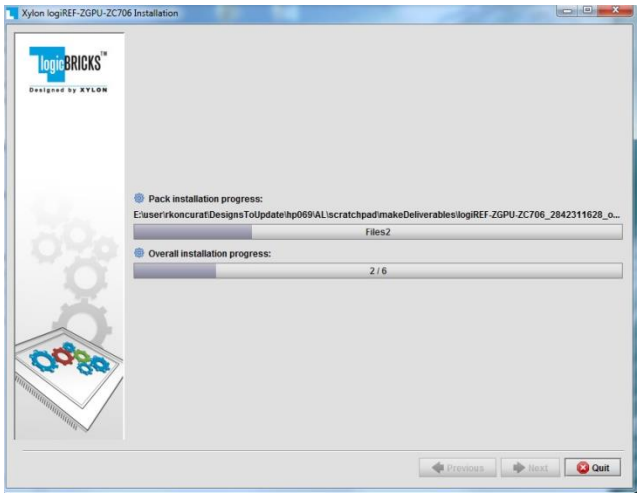


Figure 10: Installation Process – Step 3

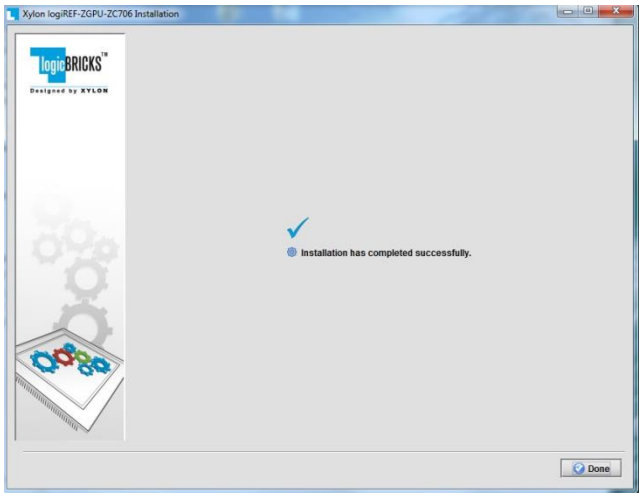


Figure 11: Installation Process – Step 4

3.1.1 Filesystem Permissions of the Installed Folder (Microsoft® Windows® OS)

The reference design installed in the default path `C:\Program Files\xylon` may inherit read-only filesystem permissions from the parent folder. This will block you in opening the hardware project file in Xilinx Vivado tools. Therefore it is necessary to change the filesystem permissions for the current user to "Full control" preferably.

To change the user permissions for `C:\Program Files\xylon` folder and all of its subdirectories, right click on the `C:\Program Files\xylon` folder and select "Properties". Under "Security" tab select "Edit". Select "Users" group in the list and check "Full control" checkbox in the "Allow" column.

Folder Structure

Figure 12 gives a top level view of the directories and files included with the logiADAK-VDF-ZU video design framework. Table 4 explains the purpose of directories.

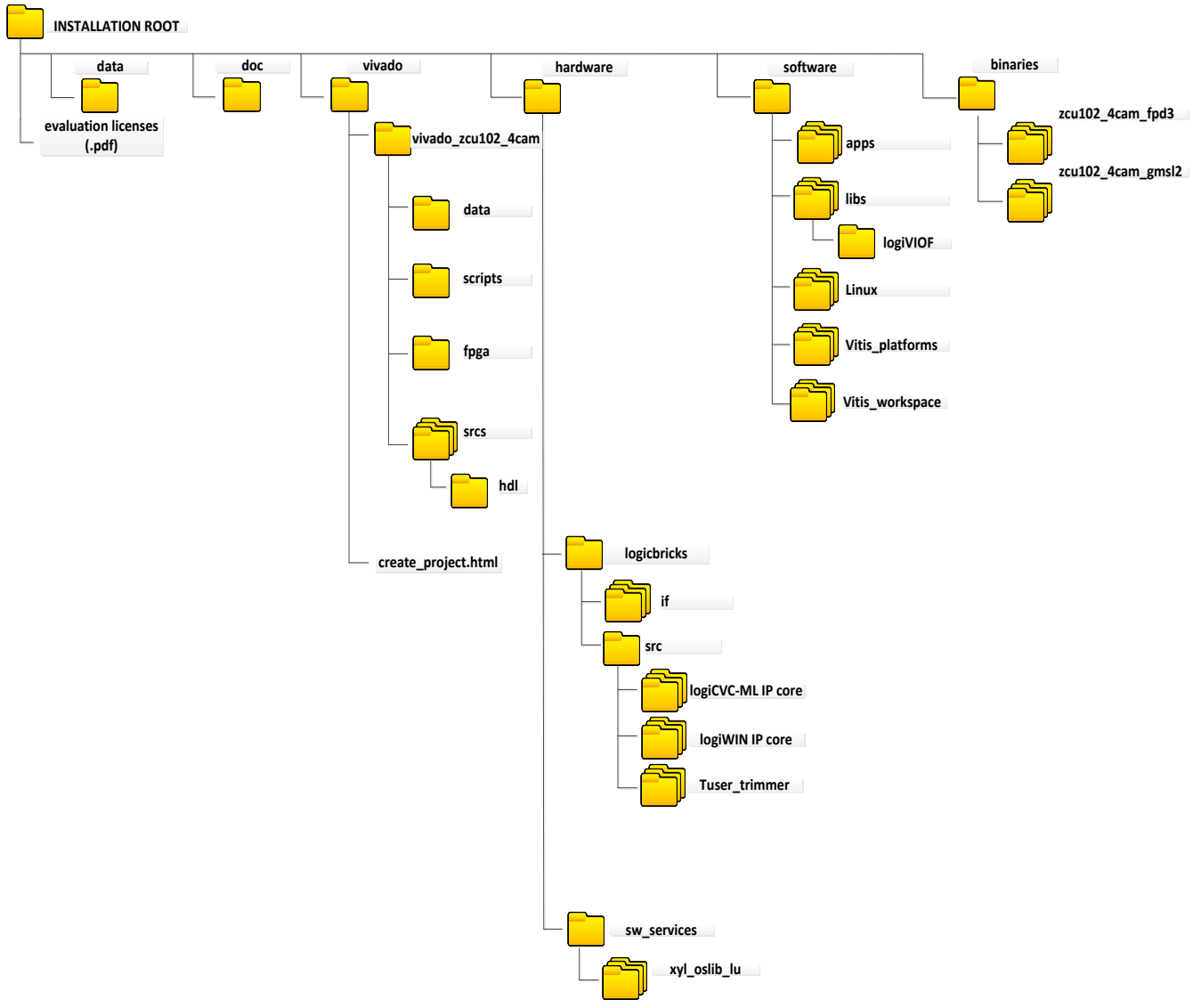


Figure 12: The Folder Structure

Folder	Purpose
INSTALLATION ROOT	-
data	Additional hardware documentation/datasheets/manuals.

doc		Project documentation.
vivado		
vivado_zcu102_4cam		This folder contains the complete Vivado project and files necessary for regenerating project from TCL scripts.
	data	Design constraints files (XDC).
	srcs	Block design GUI script and HDL wrappers.
	scripts	TCL scripts to create block design from scratch.
	fpga	ZCU102 reference design xsa file. (not in delivery)
hardware		
	logicbricks/if	Xylon custom IP core interfaces (bus definitions).
	logicbricks/src	Evaluation logicBRICKS IP cores. IP cores' User's Manuals are stored in doc subdirectories.
	sw_services	xyl_oslib_lu – Xylon Linux User Space OS abstraction library for Xilinx– use in Linux User Space applications.
software		
	apps	Demo applications source code files.
	libs/logiVIOF	logiVIOF source code files
	Linux	Linux PetaLinux files.
	Vitis_platforms	Initial files for building zcu102_4cam platform
	Vitis_workspace	Xilinx Vitis workspace folder for building applications.
binaries		
	zcu102_4cam_fpd3	Prepared binaries ready for SD card.
	zcu102_4cam_gmsl2	Prepared binaries ready for SD card.

Table 4: Explanation of the logiADAK-VDF-ZU folder structure

4 GETTING THE IP LICENSES

4.1 Getting logicBRICKS IP Licenses

The logiADAK-VDF-ZU installation comes with the evaluation versions of the logicBRICKS IP cores, and in order to be able to change the provided reference designs, you need to request the proper licenses from Xylon.

Please contact Xylon Technical Support Service support@logicbricks.com and immediately provide your Ethernet MAC ID number or Sun Host ID.

For instructions how to find your Ethernet MAC or host ID, please visit: http://www.logicbricks.com/Documentation/Article.aspx?articleID=KBA-01186-M0JXKD..

For each logicBRICKS IP core used in the logiADAK-VDF-ZU reference designs Xylon will generate and send to you separated e-mails with the license keys (file) and full instructions for setting up the license key and downloading the logicBRICKS IP core. Please follow the provided instructions.

If you experience any troubles during the registration process, please contact Xylon Technical Support Service – support@logicbricks.com.

Figure 13: E-mail with logicBRICKS License and Download Instructions

4.2 Getting the Xilinx HDMI 1.4/2.0 Transmitter Subsystem License

The logiADAK-VDF-ZU reference design comes with the the Xilinx HDMI 1.4/2.0 Transmitter Subsystem hierarchical IP, and in order to be able to change the provided reference designs, a valid license for the IP Core Bundle is needed. The HDMI 1.4/2.0 Transmitter Subsystem is a hierarchical IP that bundles a collection of HDMI IP sub-cores and outputs them as a single IP.

Digital code vouchers for the Xilinx HDMI 1.4/2.0 Transmitter Subsystem IP are delivered by Xylon to all buyers of the logiVID-ZU Evaluation Kit for use with the logiADAK-VDF-ZU reference design. Xylon does not offer an alternative direct method of acquiring the license. Voucher codes can only be redeemed once.

To redeem the voucher code for the license given to you by Xylon, please perform the following steps:

1. Go to the following link www.xilinx.com/getproduct and login with your Xilinx account.
2. After logging in you should see the page shown in Figure 14.
3. Input the voucher code in the section where it says "Have a Voucher to Redeem?" and then hit Redeem Now.
4. You will then get asked if you want to redeem your voucher for "LogiCORE, HDMI, Site License" and after saying yes it will populate the Certificated Bases Licenses section with the newly added license option. You can now select it there and generate the license which is tied to your desired Ethernet MAC ID number.

Figure 14: Xilinx License Page

If you experience any trouble during the redeeming process, please contact Xylon Technical Support Service – support@logicbricks.com.

5 LOGIADAK-VDF-ZU REFERENCE DESIGNS

The logiADAK-VDF-ZU framework contains Vivado design prepared for Vitis platform initialization.

Platform design is based on the logiADAK-VDF-ZU reference Vivado design with addition of the Sobel filter implementation in Vitis Unified Software Platform. Sobel filter is implemented between active input and display output. Filter implementation provides a simple example how to implement C-code based accelerator in the Programmable Logic (from Vitis accelerated libraries) by Vitis tool usage.

For implementing Sobel filter written in C-code in FPGA, Vitis uses Vivado HLS tool, which is called in the background. After that, Vivado design prepared for Vitis (Vitis platform) is linked with generated vision algorithm (in this case – Sobel filter) by v++ linker.

The Vitis platform supports Linux applications and contains Linux kernel drivers and libraries for the included logicBRICKS IP cores.

5.1 FOUR-CAM SoC Design and Memory Layout

This MPSoC design (Figure 15) supports four camera inputs, and the single display output. The following MPSoC resources are used for the Sobel filter implementation and are present in exported Vitis development platform:

- Clocks:
  - Clock ID 0: 150 MHz (default)
  - Clock ID 1: 300 MHz
- Platform interfaces:
  - zynq_ultra_ps_e_0: S_AXI_HP1_FPD, S_AXI_HP2_FPD
  - zynq_ultra_ps_e_0_axi_periph: M11_AXI
- Interrupts:
  - axi_intc_0: interrupts 0 to 31

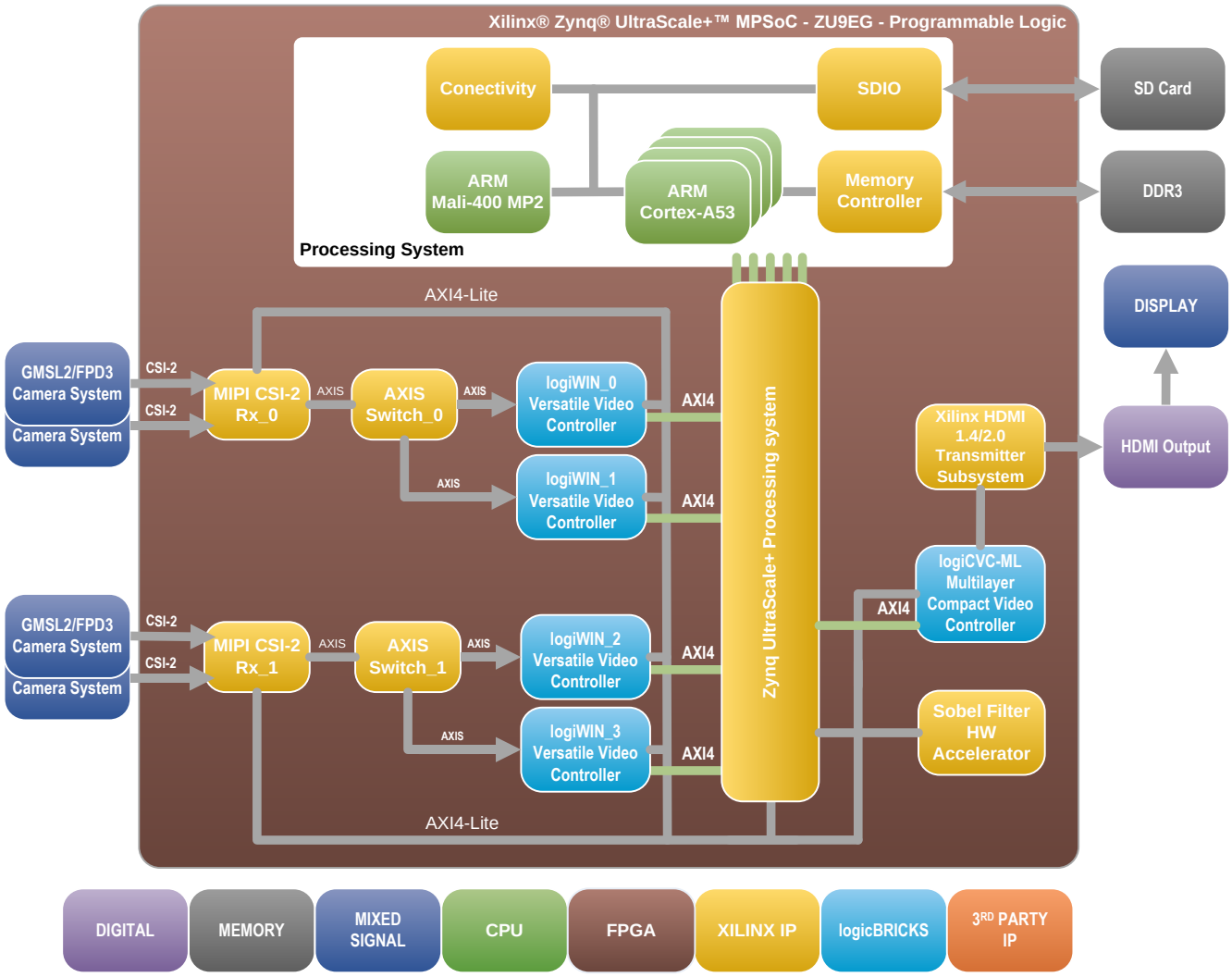


Figure 15: FOUR-CAM MPSoC Design– Block Diagram

(Clock Generator Module and other utility IP cores are not shown)

The Figure 16 shows the memory layout of video buffers. Each logiCVC layer has its own reserved memory space and multiple video buffers (not shown) which are dynamically allocated.

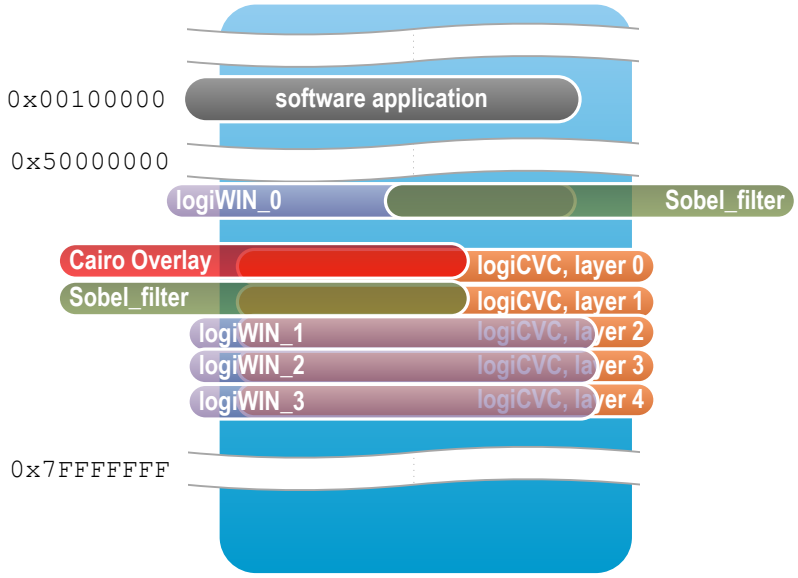


Figure 16: FOUR-CAM Design Memory Layout

5.2 Video Input/Output Synchronization

FOUR-CAM design use hardware synchronization implemented between logiWIN and logiCVC-ML IP cores. SoC systems implementing video input units have video input frame rates and video output frame rates rarely equal, and need to implement frame rate conversions from lower to higher frame rate, or vice versa. For camera #1 and Sobel filter visualization logiVIOF is used, where software triple buffering is used, while hardware synchronization is implemented for the remaining 3 cameras.

5.2.1 logiWIN Hardware Buffering Implementation

Double/triple buffering state machine is placed outside the logiWIN. The logiWIN output pin *curr_vbuff[1:0]* sends to the double/triple buffer external controller information to which buffer the logiWIN currently writes. On the input pin *next_vbuff[1:0]* the double/triple buffer external controller sends to the logiWIN information to which buffer the next frame should be written. For double buffering *next_vbuff* value changes between 0 and 1, and for triple buffering between 0, 1 and 2. If double/triple buffering is not in use, the *next_vbuff* must be set to 0. Output signal *sw_vbuff_req* signals to the external controller that the current buffer *curr_vbuff[1:0]* is written and requests buffer switching. External controller grants buffer switching over *sw_vbuff_grant* together with the pointer to the next buffer *next_vbuff[1:0]*.

5.2.2 logiCVC-ML Hardware Buffering Implementation

External video synchronization requires three separate frame buffers (buffer_0, buffer_1 and buffer_2 implemented in the video memory). In SoC designs with the logiCVC-ML display controller IP core, three frame buffers must be setup for every logiCVC-ML graphic layer. The triple buffering method provides an advantage over the double buffering synchronization method, since the video input units do not have to wait on buffers swapping as there is always a spare frame buffer for new frame data writing.

To support this feature, the logiCVC-ML uses video input synchronization control port which consists of *e_curr_vbuff*[C_NUM_OF_LAYERS*2-1:0] and *e_switch_vbuff*[C_NUM_OF_LAYERS-1:0] input signals, and *e_next_vbuff* [C_NUM_OF_LAYERS*2-1:0] and *e_switch_grant*[C_NUM_OF_LAYERS-1:0] output signals.

With the input signals *e_current_vbuff*[n*2+1:n*2] and *e_switch_vbuff*[n] external video source signals to logiCVC-ML layer n the currently written buffer and when to switch buffers (typically on the end of its active frame of external video source). With the output signal *e_switch_grant*[C_NUM_OF_LAYERS-1:0] the logiCVC-ML grants the video source to start writing its next frame to *e_next_vbuff*[n*2+1:n*2] buffer.

The logiCVC-ML IP core is constantly sampling *e_current_vbuff* and *e_switch_vbuff* inputs with the memory clock. When *e_switch_vbuff* high state is detected, the logiCVC samples *e_current_vbuff* and asserts *e_switch_grant* along with the associated *e_next_vbuff*. External logic should constantly sample *e_switch_grant* signal, and when it detects that *e_switch_grant* is high, it should sample *e_next_vbuff* and de-assert *e_switch_vbuff*. When logiCVC detects *e_switch_vbuff* low, it de-asserts *e_switch_grant* signal on the next memory clock cycle. *e_switch_vbuff* and *e_switch_grant* signals are used as handshake signals between logiCVC and external logic. This kind of implementation supports buffers switching between logiCVC and external logic running on synchronous and on asynchronous clocks.

To enable external frame buffer synchronization for a specific graphic layer, user has to enable it by setting the EN_EXT_VBUFF_SW bit to 1 in the corresponding layer control register.

If external video input signals are connected to the logiCVC-ML's video input synchronization control port and synchronization are turned off (EN_EXT_VBUFF_SW=0), logiCVC-ML will always signal the external video input to write data to buffer 0, i.e. *e_next_vbuff*[n*2+1:n*2]=0. At the same time, logiCVC-ML will work in the CPU synchronization mode so it will read memory buffer, which is defined with layer address register.

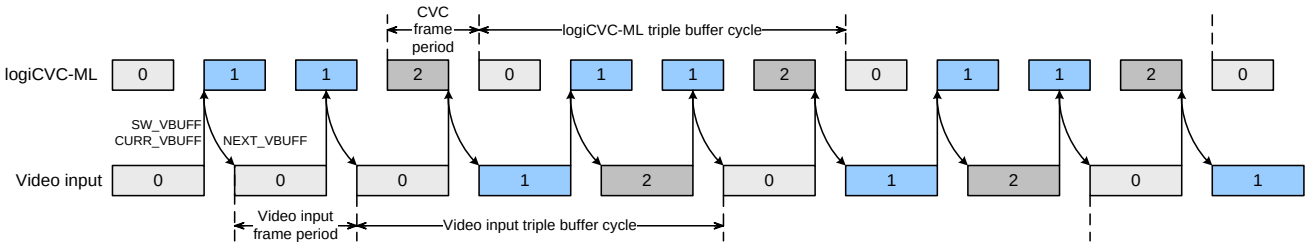


Figure 17: Triple Buffering Example when logiCVC-ML Refresh Rate is Higher than Video Input

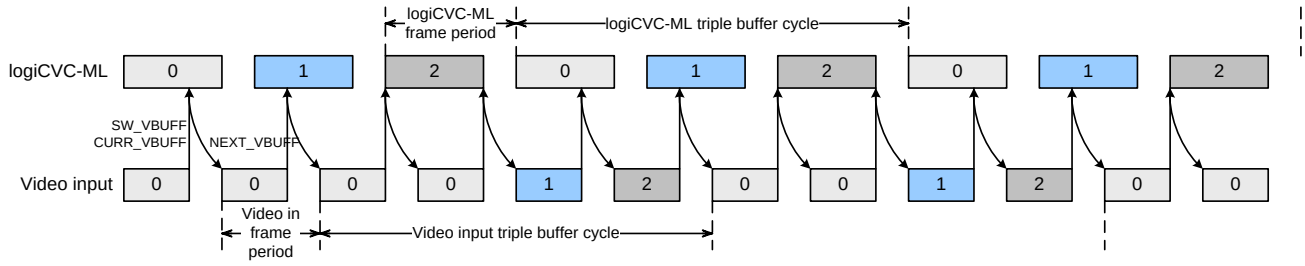


Figure 18: Triple Buffering Example when logiCVC-ML Refresh Rate is Lower than Video Input

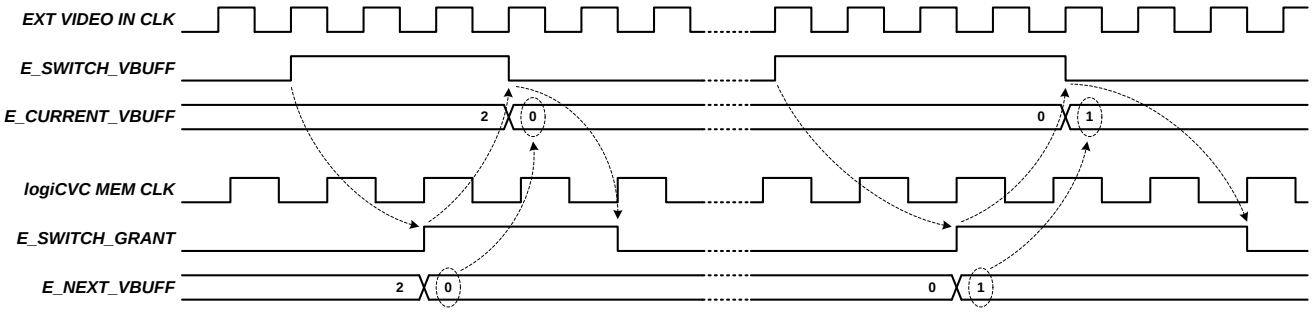


Figure 19: External buffer control signals timing diagram

5.3 Restoring Full MPSoC Design from Xylon Deliverables

Xylon provides all necessary design files and TCL scripts to enable full project restore in the Xilinx Vivado Design Suite 2021.1. To regenerate the Vivado designs, please open Vivado Design Suite 2021.1.

Navigate to ***logiADAK-VDF-ZU_vX_Y_Z/vivado/vivado_zcu102_4cam/scripts*** folder for the FOUR-CAM design.

To regenerate the selected design, run the `project.tcl` script (type `source project.tcl` to Vivado Tcl Shell as shown by Figure 20). Upon the design regeneration, open the Vivado project and the Vivado Block Diagram should be the same as shown in Figure 21.

Please note that design synthesis and implementation is only validated using Vivado Design Suite 2021.1 on Linux operating system.

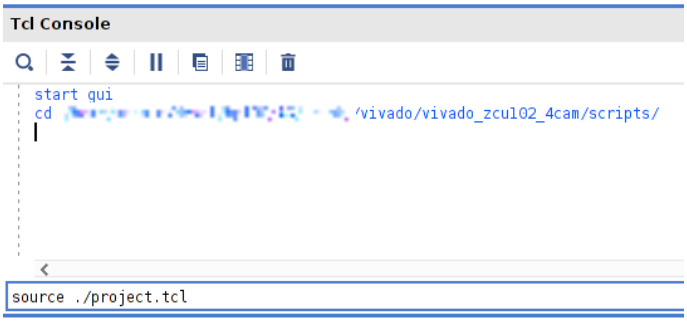


Figure 20: Vivado 2021.1 Tcl Shell – Regenerating the Project Example

Please note that design regeneration takes some time. Also, during the regeneration process, the Vivado GUI may close and re-open, therefore please do not be alarmed by these events.

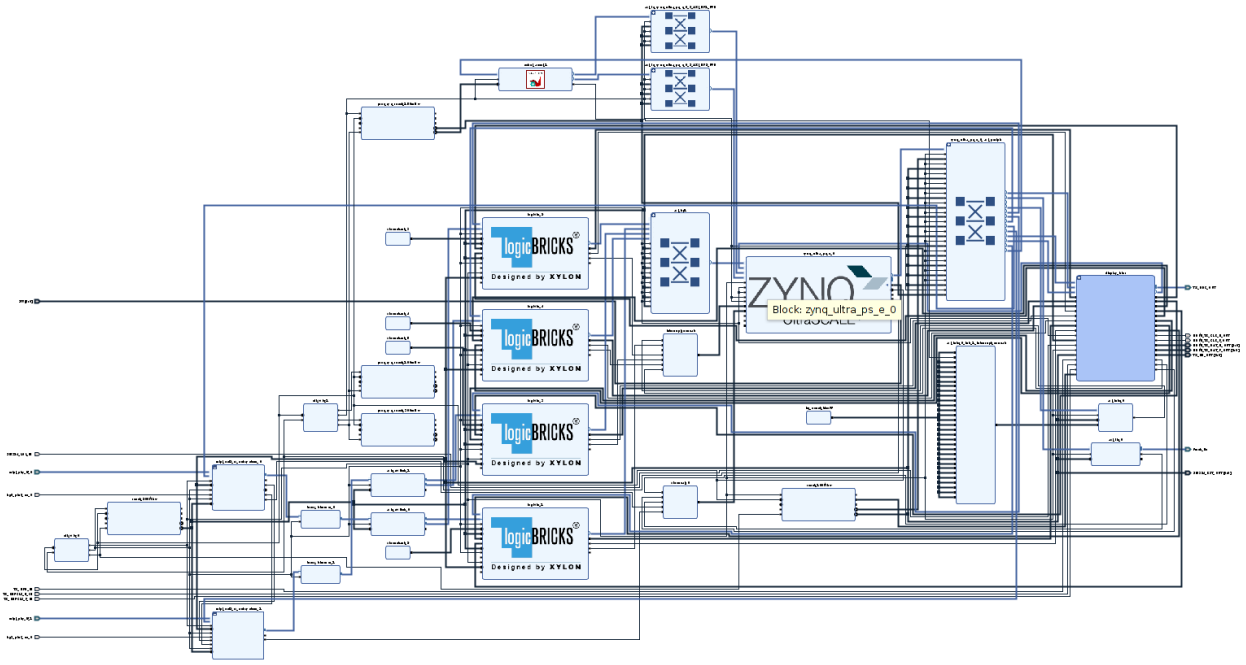


Figure 21: FOUR-CAM Vivado IP Integrator Block Diagram

5.4 Vitis Platform and the Hardware Accelerator implementing Sobel filter

In the Vitis development environment in logiADAK-VDF-ZU application, Sobel filter is used as an example of using Vitis accelerated libraries for accelerating various, already implemented, hardware functions.

In Vivado Design Suite, hardware platform is exported to be used in Vitis software development environment, where Sobel filter is implemented using Vitis accelerated libraries.

In the provided software demo application, the Sobel filter algorithm is contained in one C++ function `sobel_camera()(vitis_sobel.cpp` source code file).This function is used as a simple example of the C-code based hardware accelerator implementation generated by the Vitis tool.

You can use broad range of pre-optimized libraries that you can call as a software API to bring plug-n-play acceleration in your application or customize to design your own accelerator for Xilinx platforms across edge to cloud.

Vitis accelerated-libraries are accessible to all developers through GitHub and scalable across all Xilinx platforms. Develop your applications using these optimized libraries and seamlessly deploy across Xilinx platforms at the edge, on premise or in the cloud without having to re-implement your accelerated application.

For more information see: https://xilinx.github.io/Vitis_Libraries/

5.5 Software Description

Software components:

- v4l2 driver, https://github.com/logicbricks/driver_v4l2_logiwin, patch for driver v1.02 is located:
  `../software/Linux/zcu102_4cam/project-spec/meta-xylon/recipes-kernel/logiwin/files/0001-Petalinux-2021.1-kernel-uplift.patch`
- DRM driver, https://github.com/logicbricks/driver_drm_logicvc, patches for driver is located:
  `../software/Linux/zcu102_4cam/project-spec/meta-xylon/recipes-kernel/logicvc/files/0001-Petalinux-2021.1-kernel-uplift.patch`
- logiVIOF library, for detailed documentation see:
  `../software/libs/logiVIOF/doc/html/index.html`
- plf+ library (for GMSL2 and FPD3 cameras initialization)
  `../software/Linux/zcu102_4cam/project-spec/meta-xylon/recipes-apps/plf`

logiVIOF DRM implementation is not yet fully documented.

Software architecture is shown in the picture below.

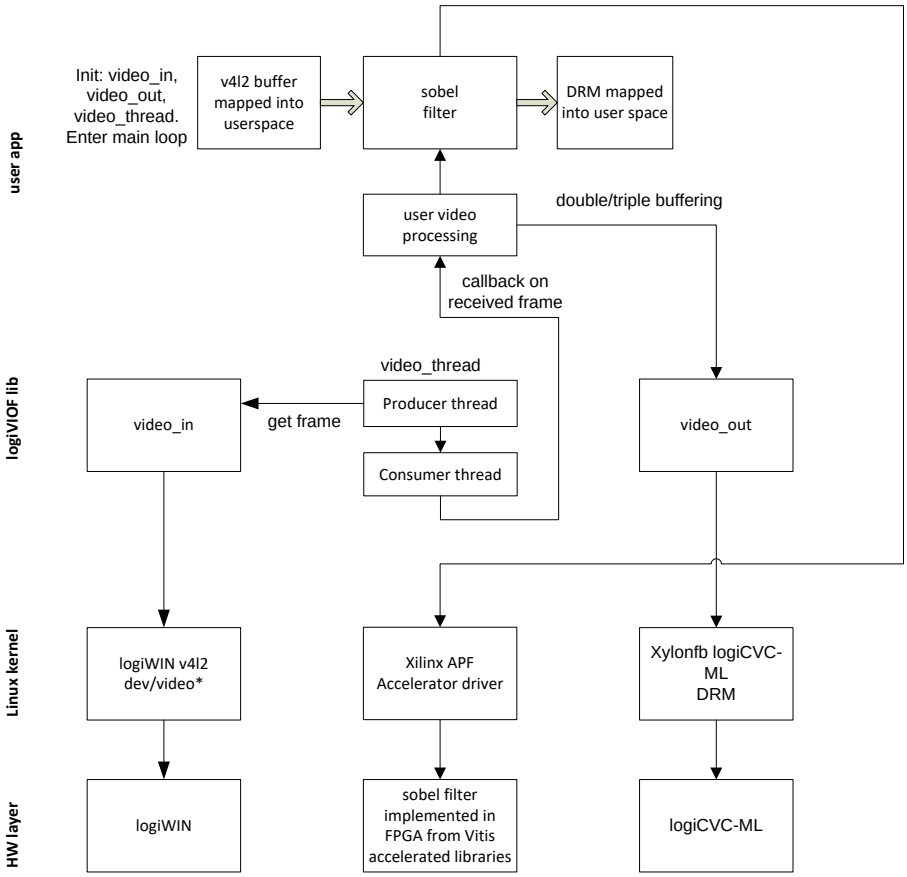


Figure 22: Software architecture

5.5.1 Demo application

There is FOUR-CAM demo application, in two versions: GMSL2 and FPD-Link III.

Application for both versions is the same, difference being only initialization scripts on SD card (`plf_fpd3` for FPD-Link III and `plf_gmsl2` for GMSL2 version).

Application demonstrates how to display the flicker-free video from an external video input by using the software/hardware buffering synchronization, and shows how to use Vitis accelerated libraries by implementing Sobel filter demonstration on one camera input.

Video input and output for Camera #1 (on which Sobel filter is demonstrated) is initialized by the logiVIOF library (Figure 22). For other cameras and their respective logiWINs, linuxuserspace driver is used for logiWIN initialization and hardware buffering for synchronization with logiCVC.
Main parts used in the demo are:

- **`plfConfiguration()`**
- **`menu_loop()`**
  - **`demoInit()`**
  - **`demoStart()`**
  - **`demoStop()`**
  - **`demoDeinit()`**

`plfConfiguration()` initializes Xylon Deserializer FMC module and Xylon cameras using `plf+` library and .xml and .ini scripts located on SD card. **`demoInit()`** initializes video in and video out:

```
demoInit():
   ▪ vout_init()            //create logiVIOF VideoOut interface to control
                            //DRM device (xylon DRM dev/dri/logicvc*)
   ▪ vout_layerEnable()     //enable logiCVC layer
   ▪ vout_layerDisable()    //disable logiCVC layer
   ▪ vin_init()             //
   ▪ logiWIN_initialize()   // initialize logiWINs for cameras other than
                            // camera #1
   ▪ init_overlay()         // initialize Cairo overlay graphics
```

After the **`demoInit()`** part, the SW app enters the **`menu_loop()`** function and checks the on-board push-buttons and keyboard buttons. User can display each of the cameras and toggle Sobel filter visualization.

```
change_overlay()          // change overlay graphics between modes - one
                          // camera / four cameras
demoStart():
   ▪ logiWIN_set_format ()     //set input format if mode is changed
                               // between one camera / four cameras
   ▪ video_thread_initialize() // create logiVIOF VideoOut interface to
                               //   control   framegrabber   (xylon   v4l2
                               // dev/video0)

video_layers_mode():
   ▪ vout_layerSetPositionAndSize()//depending on mode set layer geometry
```

```
                                        //for one / four cameras
```

- `vout_layerEnable()  //depending on mode enables one`
  `//or all four layers`

Upon the **`demoStart()`** execution, the program enters the main loop and checks the on-board push-buttons and keyboard buttons. User can stop the application by calling the **`demoStop()`** function:

`demoStop():`

- `vout_layerDisable()  //disables all layers`
- `logiWIN_disable()  // disable all logiWINs`

The **`demoStart()`** function is called with the user options after the end of the **`demoStop()`** function.

If user exits the application, the **`demoDeinit()`** function deinitializes video inputs and the video output:

`demoDeinit():`

- `vin_deinit()  // deinitializes VideoIn`
- `vout_deinit()  // deinitializes VideoOut`

5.5.2 Input resolution and the frame rate

Both reference designs use 1928x1208@30 input video resolution for Xylon cameras.

5.5.3 Output resolution and the frame rate

Both reference designs use 1920x1080@60 output resolution.

6 QUICK START

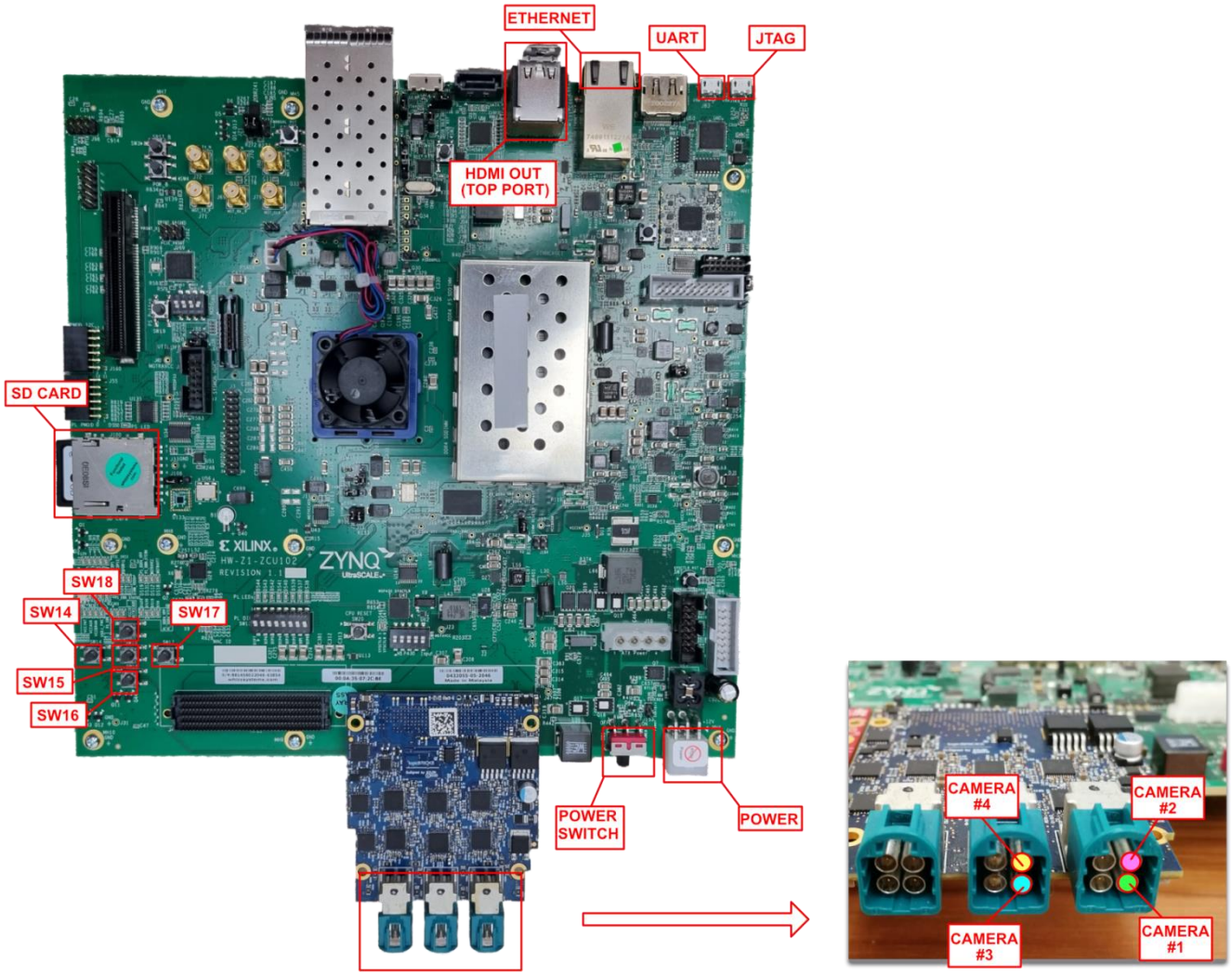


Figure 23: Four-CAM HW Setup (GMSL2 Version)

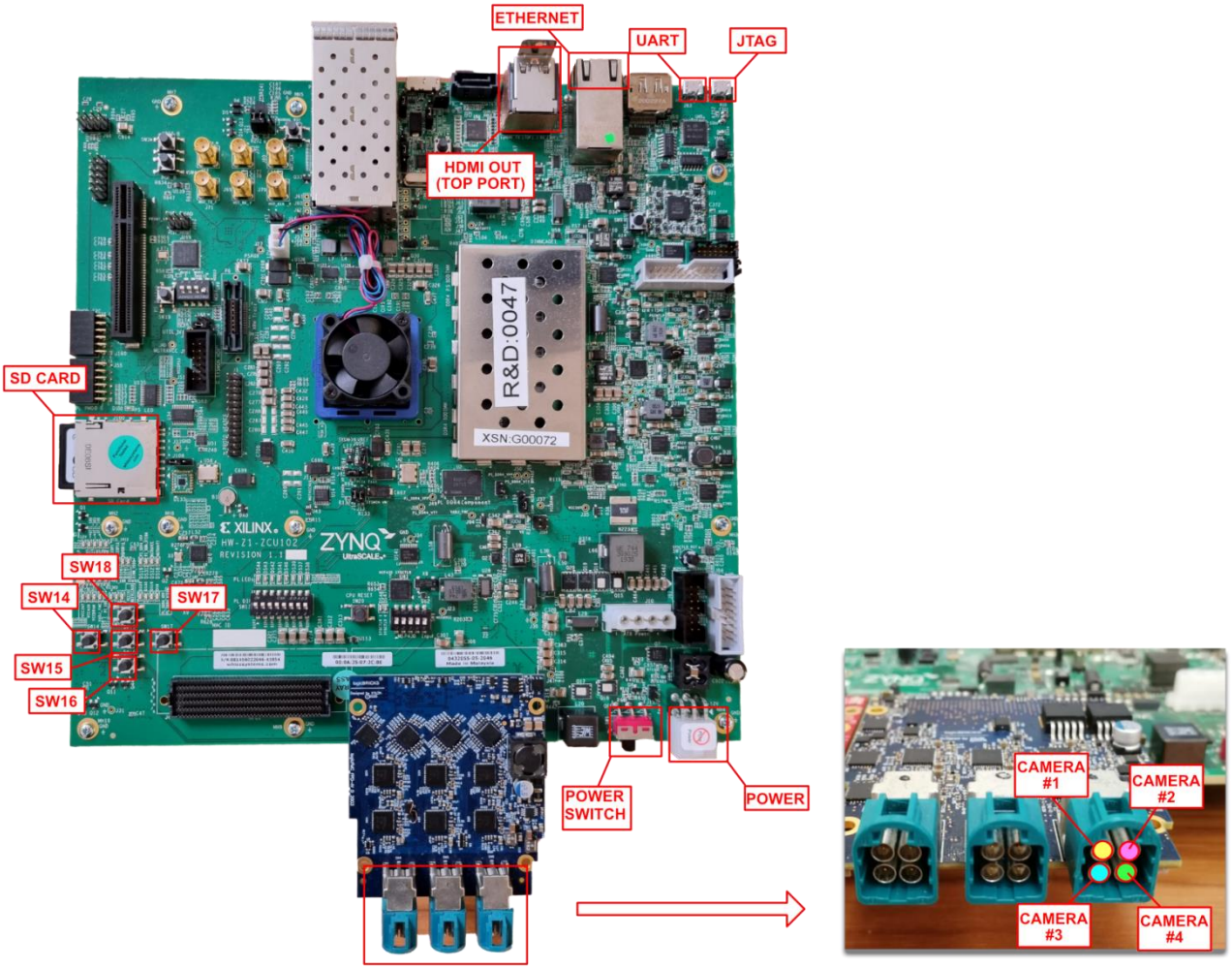


Figure 24: Four-CAM HW Setup (FPD-Link III Version)

For FOUR-CAM design cameras must be plugged into inputs labeled from CAMERA #1 to CAMERA #4.

6.1 Run the Precompiled Linux Demo Examples

To enable rapid testing of the hardware setup, Xylon provides application demo binaries in the *binaries* folder of the deliverables. Connect hardware as depicted on Figure 23 or Figure 24.

To run the four cameras demo for Xylon FPD-Link III cameras, copy the content of the *binaries/zcu102_4cam_fpd3* folder to the root folder of the SD card.

To run the four cameras demo for Xylon GMSL2 cameras, copy the content of the *binaries/zcu102_4cam_gmsl2* folder to the root folder of the SD card.

SD card should be formatted as FAT32.

Optionally, you can use a serial terminal program (baud rate 115200 8N1) and the USB UART connection to the ZCU102 board to monitor the system's operation.

For full explanation of the ZCU102's features and settings, please check the documentation Xilinx XTP426.

6.2 Demo controls

Start the FOUR-CAM demo design and the display will show the video stream from one of the four attached video cameras. Change the displayed camera input by pressing the number '1', '2', '3' or '4' on a keyboard, or by pressing SW14 or SW17 push-buttons on the ZCU102 board. Toggle demo mode between one camera, or all cameras by pressing the number '5' on a keyboard, or by pressing the SW15 push-button on the ZCU102 board. Toggle Sobel filter visualization on camera #1 by pressing 's' on keyboard or SW18 on the ZCU102 board. Stop the application by pressing 'q' on the keyboard.

6.3 Change the Delivered Software

6.3.1 Xilinx Development Software

The logiADAK-VDF-ZU video design framework reference designs and Xylon logicBRICKS IP cores are fully compatible with Xilinx development tools – Vivado Design Suite 2021.1. Future design releases shall be synchronized with the newest Xilinx development tools.

Licensed users of Xilinx tools can use their existing software installation for the logiADAK-VDF-ZU evaluation and modifications.

6.3.2 Set Up Linux System Software Development Tools

Set of ARM GNU tools are required to build the Linux software and applications. The complete tool chain for the Zynq UltraScale+ MPSoC can be obtained from the Xilinx ARM GNU Tools wiki page: http://wiki.xilinx.com/Install+Xilinx+Tools. Access to tools requires a valid, registered Xilinx user login name and password.

6.3.3 Set Up git Tools

Git is a free Source Code Management (SCM) tool for managing distributed version control and collaborative development of software. It provides the developer a local copy of the entire development project files and the very latest changes to the software.

Visit https://www.xilinx.com/html_docs/xilinx2019_1/SDK_Doc/SDK_tasks/sdk_working_with_git.html to get instructions how to use Xilinx git.

To get the latest version of Xylon logicBRICKS software drivers for Linux operating system, please visit Xylon's git: https://github.com/logicbricks.

6.3.4 Setting up the Vitis workspace

All logiADAK-VDF-ZU software applications are delivered in the source code to enable users to do software customizations. If the hardware platforms have been changed it is necessary to rebuild the platform inside Vitis workspace. This paragraph explains how to setup the Xilinx Vitis environment for these customizations.

Quick steps required to modify deliverables to specific needs:

1. Open the Vivado design in `INSTALLATION_ROOT/vivado` in Vivado 2021.1**,** make changes (optionally), generate device image and export .XSA file (go to: File -> Export platform -> Hardware -> Pre-synthesis)
2. Build petalinux located in `INSTALLATION_ROOT/software/Linux/ zcu102_4cam`:
   - a. petalinux-config --get-hw-description=./<path_to_xsa> (import HW description generated in step 1.)
   - b. petalinux-build (build petalinux project)
   - c. petalinux-build --sdk (build petalinux SDK)
   - d. petalinux-package --sysroot (export sysroot)

3. From `INSTALLATION_ROOT/software/Linux/zcu102_4cam/images/linux`:

   - Copy *bl31.elf*, *zynqmp_fsbl.elf (*rename to *fsbl.elf*), *u-boot.elf* and *pmufw.elf* to `INSTALLATION_ROOT/software/Vitis_platforms/boot` directory
   - Copy *boot.scr*, *image.ub* and *rootfs.cpio.gz* to `INSTALLATION_ROOT/software/Vitis_platforms/image` directory

NOTE: Windows OS support is limited to the Vitis embedded software development flow. The Vitis acceleration flow is only supported on Linux. For more information on how to setup development environment see: https://github.com/Xilinx/Vitis-Tutorials/tree/2021.1/Getting_Started/Vitis

4. Create and build Vitis platform:
   - Open Vitis development IDE
   - As workspace set `INSTALLATION_ROOT/software/Vitis_workspace`
   - Go to File -> New -> Platform Project...
     - Platform project name: *hp130ag*
     - Create a new platform from hardware (XSA) -> Browse... -> choose .xsa hw file (generated in step 1.)
     - Software Specification:
       - Operating system: linux

        - Processor: psu_cortexa53
        - Uncheck "Generate boot components"
    - Click Finish
- Open *hp1130ag/platform.spr* -> linux on psu_cortexa53
    - Bif File: Browse... -> select .bif file from `INSTALLATION_ROOT/software/Vitis_platforms/boot` directory
    - Boot components Directory: Browse... -> select `INSTALLATION_ROOT/software/Vitis_platforms/boot` directory
    - FAT32 partition directory: Browse... -> select `INSTALLATION_ROOT/software/Vitis_platforms/image` directory
    - Linux Rootfs (optional): Browse... -> select `INSTALLATION_ROOT/software/Vitis_platforms/image/rootfs.cpio.gz`
    - Sysroot Directory: Browse... -> select dir (from step 2. d.) `INSTALLATION_ROOT/software/Linux/zcu102_4cam/images/linux/sdk/sysroots/cortexa72-cortexa53-xilinx-linux`
- Right click `hp130ag` -> Build project

5. Import and build application:
    - Go to File -> Import -> Eclipse workspace or zip file
        - Select root directory: `INSTALLATION_ROOT/software/Vitis_workspace`
        - Uncheck "Copy projects into workspace"
        - Check all projects to import into workspace
        - Click Finish
    - Open `app_zcu102_4cam_system.sprj`:
        - Change Active build configuration to Hardware
        - Change sysroot to correct path (inside platform export folder, e.g. `Vitis_workspace/PLATFORM_ROOT/export/PLATFORM_NAME/sw/PLATFORM_NAME/linux_domain/sysroot/cortexa72-cortexa53-xilinx-linux`)
        - Change Kernel image to correct path (inside platform export folder, e.g. `Vitis_workspace/PLATFORM_ROOT/export/PLATFORM_NAME/sw/PLATFORM_NAME/linux_domain/image/image.ub`)
    - Right click `logiVIOF` -> Build Project
    - Right click `app_zcu102_4cam_system` -> Build project *

* This will build all system projects, including kernel, hardware linking and software application. This process takes a while.

6. Format SD card to have single FAT32 partition.

7. Copy executables to SD card
    - From `app_zcu102_4cam_system/Hardware/package/sd_card` copy all content to SD card

- From
  - `INSTALLATION_ROOT/binaries/zcu102_4cam_fpd3` for FPD-Link III version cameras copy:
    - directory containing overlay logo (`logo`)
    - plf+ directory for initialization of Xylon cameras (`plf_fpd3`)
    - `platform_desc.txt` and `init.sh` for XRT and startup initialization.
  - `INSTALLATION_ROOT/binaries/zcu102_4cam_gmsl2` for GMSL2 version cameras copy:
    - directory containing overlay logo (`logo`)
    - plf+ directory for initialization of Xylon cameras (`plf_gmsl2`)
    - `platform_desc.txt` and `init.sh` for XRT and startup initialization

8. Setup board into SD boot mode, connect UART cable (optional) and power ON

6.4 Software Instructions – Linux Software

Xylon provides Linux logiCVC DRM and V4L2 logiWIN driver. Zynq UltraScale+ tool chain, Linux kernel and file system used for development and demonstrations of Xylon drivers are provisions of Xylon.

- Linux kernel building instructions and `dts` files can be found in `software\Linux\zcu102_4cam` folder.
- Running Linux applications with the ZCU102 board setup for the precompiled SD card image

6.5 Debugging Software Application with the TCF Agent

1. Launch the Vitis workspace
2. Open **`Debug Configurations`** window (Figure 25). To create a new Debug Configuration, double click on the **`System Project Debug`** section on the left hand side of the Debug Configuration GUI (Figure 26).

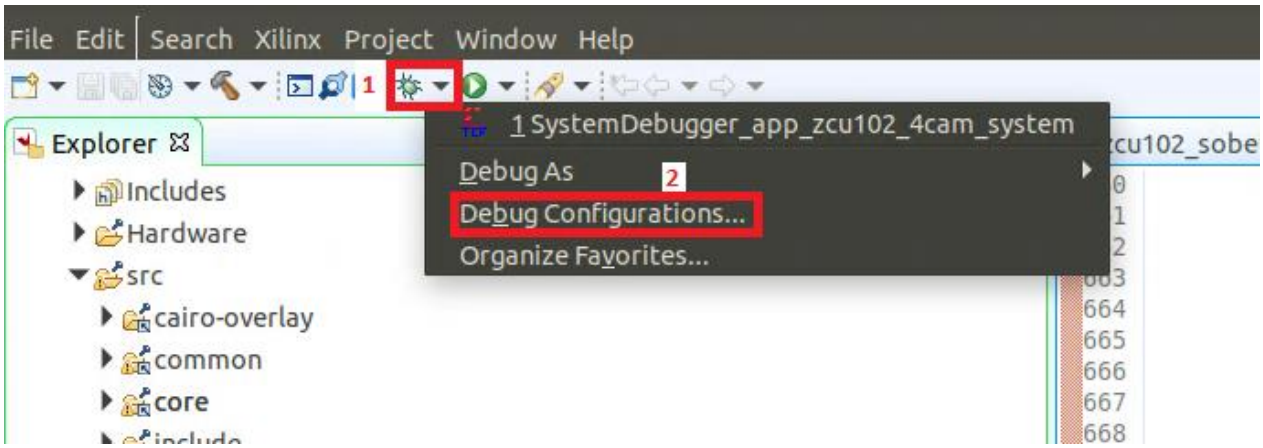


Figure 25: Vitis Workspace – Open Debug Configurations

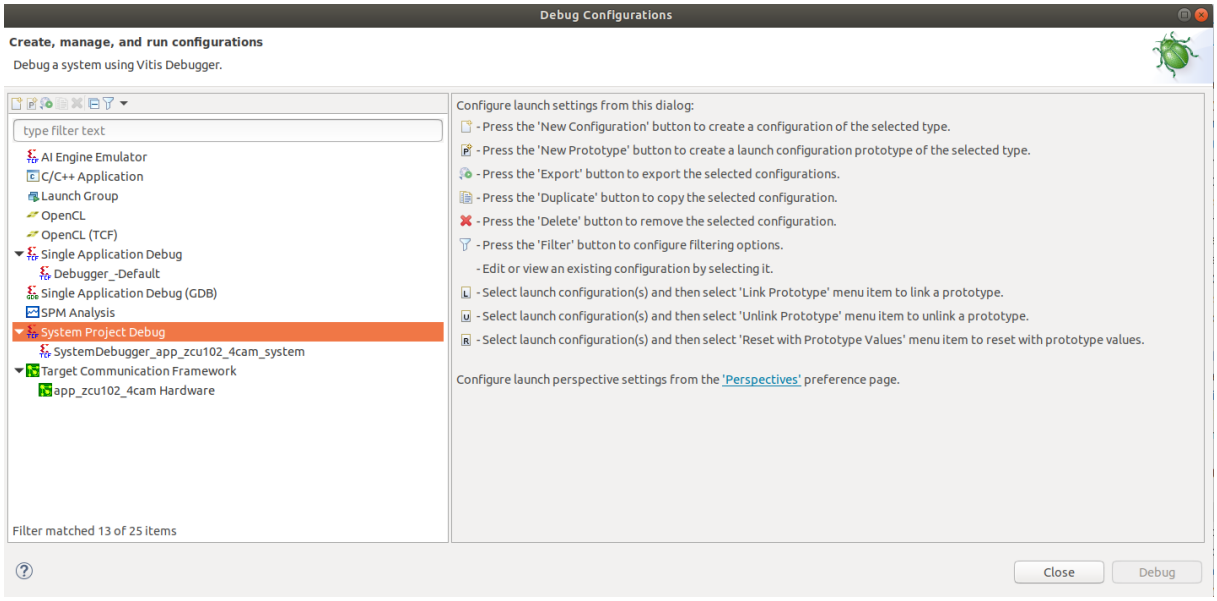


Figure 26: Vitis Workspace – Debug Configurations

3. Create new connection; set **`Host`** to correct target IP and set **`Port`** to 1534

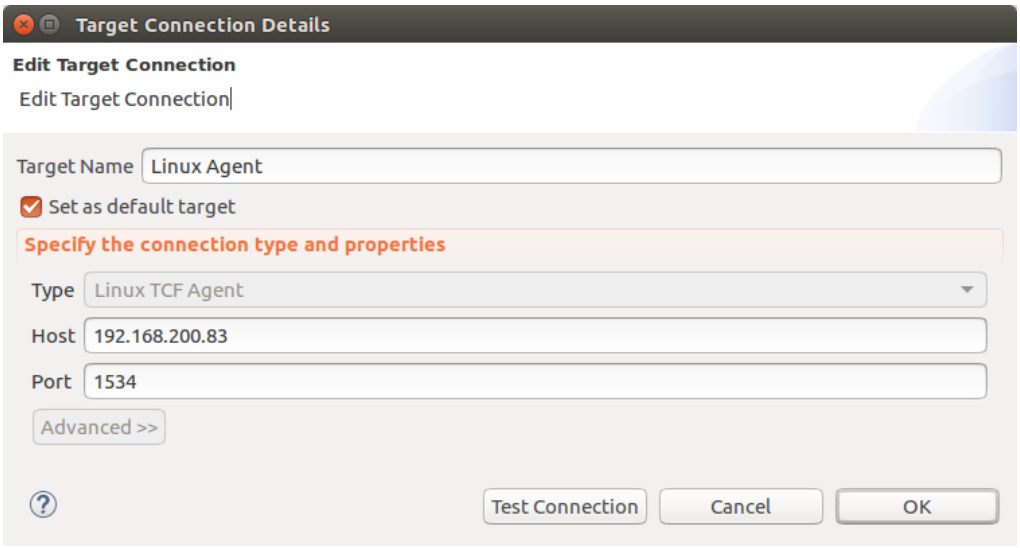


Figure 27: Vitis Workspace – Create the new Connection

4. Set project which is debugged (click on **`Browse`**…)
5. Select application which is intended to debug.
6. Set Linux TCF Agent to created connection in step 4.
7. Enter **`Remote Working Directory and Program Arguments.`**

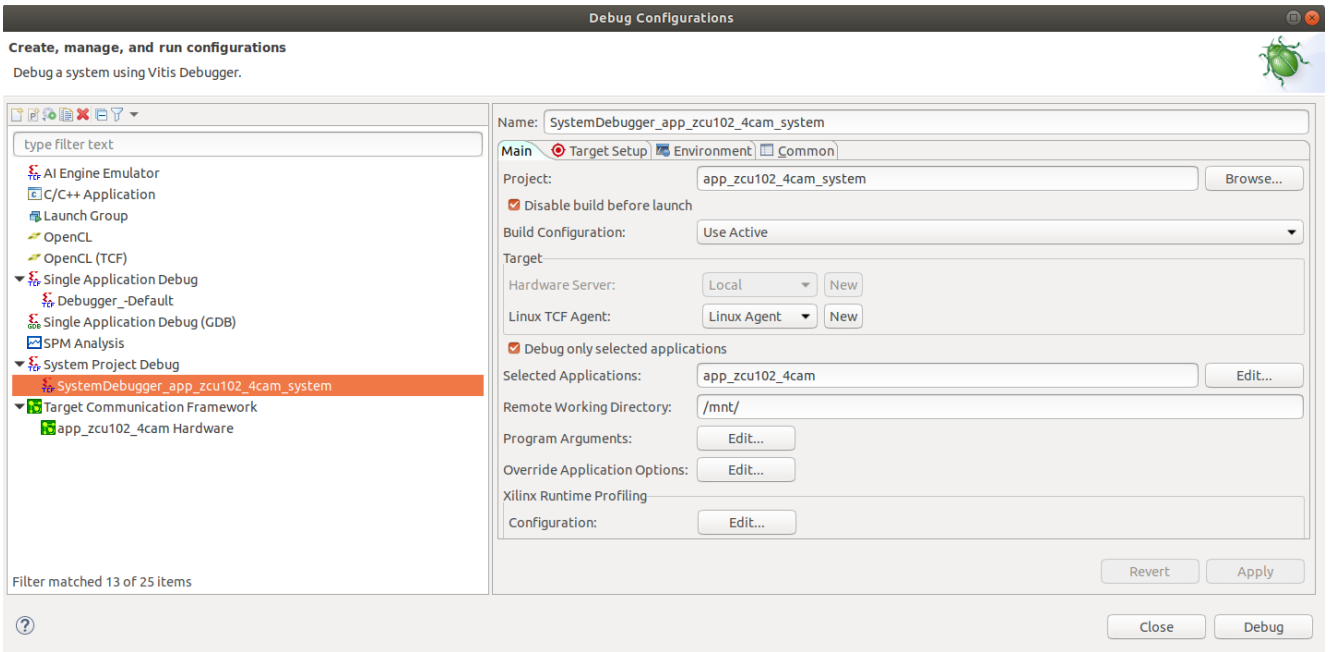


Figure 28: Vitis Workspace – Main Tab

8. If shared libraries are used set paths to in the **`Environment`** tab
9. Start **`Debug`**

7 REVISION HISTORY

Version	Date	Author	Approved by	Note
1.00.a	July 28th, 2017	A.Bogdanic, D.Stimac	G. Galić	Initial draft
2.00.a	September 4th, 2017	A.Bogdanic, D.Stimac	G. Galić	Initial release
2.01.a	September 7th, 2018	D.Perisic	D.Perisic	Camera interface changed to Xylon FPD-Link III Deserializer FMC module.
2.0.2	October 30th 2018	M.Posavec	R. Končurat	Update to Vivado Design Suite 2017.4
2.0.3	July 19th 2019	K. Mlakar	R. Končurat	Updated to support cameras with MIPI interface
3.0.1	December, 6th, 2019	A. Stanisic		Updated to Xilinx tools (Vivado Design Suite, SDK and PetaLinux) version 2019.1. Fixed Figure 25 (HDMI-CAM and FOUR-CAM designs were switched). Unified GMSL2 and FPD-Link III documentation.
3.0.2	April 28th, 2020	R.Soldat		Added 26 and modified Figure 25 to only show HDMI GMSL2 Setup instead of both.
3.1.0.	July 14th, 2021	R.Soldat		Updated Figure 1 and Figure 2 to show latest camera and FMC versions. Added Figure 5.
4.0.1	October 25th, 2021	S. Mišković		Updated to Xilinx tools (Vivado Design Suite, Vitis and Petalinux) version 2021.1. Added Sobel filter implementation. Removed HDMI-CAM HW and SW from document. Updated Figure 15 to show HDMI 1.4/2.0 Transmitter Subsystem IP Core.