

4K Camera Agilex 5 FPGA E-Series 065B Arrow Eagle Board Demo Design

Version: **3.2.0**

Last updated: **2024-11-15**

Altera Confidential

© Altera Corporation. Altera, the Altera logo, the 'a' Logo, and other Altera marks are trademarks of Altera Corporation.

Altera and Intel warrant performance of its FPGA and semiconductor products to current specifications in accordance with Altera's or Intel's standard warranty as applicable, but reserves the right to make changes to any products and services at any time without notice.

Altera and Intel assume no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera or Intel.

Altera and Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

*Other names and brands may be claimed as the property of others.

Contents

| | | |
|------------|--|-----------|
| 1.0 | Introduction | 7 |
| 1.1 | Features | 7 |
| 1.2 | Terminology | 8 |
| 1.3 | Reference Documents | 10 |
| 2.0 | Release Version | 11 |
| 2.1 | Quartus Prime Version | 11 |
| 2.2 | Changes from the Previous Version | 11 |
| 3.0 | Functional Description | 12 |
| 3.1 | MIPI Input..... | 12 |
| 3.2 | Input TPG | 13 |
| 3.3 | Black Level Statistics | 13 |
| 3.4 | Clipper..... | 14 |
| 3.5 | Defective Pixel Correction..... | 14 |
| 3.6 | Adaptive Noise Reduction | 16 |
| 3.7 | Black Level Correction | 17 |
| 3.8 | Vignette Correction | 19 |
| 3.9 | White Balance Statistics | 20 |
| 3.10 | White Balance Correction | 21 |
| 3.11 | Demosaic | 22 |
| 3.12 | Histogram Statistics | 24 |
| 3.13 | Unsharp Mask Filter..... | 24 |
| 3.14 | Color Correction Matrix | 25 |
| 3.15 | Warp | 25 |
| 3.16 | TMO | 26 |
| 3.17 | 3D LUT | 28 |
| | 3.17.1 Generating LUT Files | 29 |
| 3.18 | Logo Overlay | 30 |
| 3.19 | 1D LUT | 30 |
| 3.20 | Frame Writer | 31 |
| 3.21 | Hard Processor System | 32 |
| 3.22 | Miscellaneous | 32 |
| 4.0 | Hardware and Software Setup | 33 |
| 4.1 | Pre-compiled Delivery Package | 33 |
| 4.2 | Source Delivery Package | 33 |
| 4.3 | Hardware and Software Requirements | 33 |
| | 4.3.1 Minimum Hardware and Software Requirements | 34 |
| | 4.3.2 Requirements for FPGA Development | 35 |
| | 4.3.3 Requirements for Software Development | 39 |
| 4.4 | Development Kit Setup | 39 |
| | 4.4.1 Flashing the MicroSD Card | 41 |

| | | |
|------------|--|-----------|
| 4.5 | Powering up the Development Kit | 41 |
| 4.5.1 | FPGA Workflow | 42 |
| 4.6 | Booting the Application | 43 |
| 5.0 | Graphical User Interface | 45 |
| 5.1 | Input Config Tab | 46 |
| 5.1.1 | Sensor Control | 46 |
| 5.1.2 | Sensor Color Profile | 47 |
| 5.1.3 | Test Pattern Generator | 48 |
| 5.1.4 | Input Select | 49 |
| 5.1.5 | Clipper | 49 |
| 5.1.6 | Autoexposure Controls | 51 |
| 5.2 | ISP Pipeline Tab | 52 |
| 5.2.1 | Defective Pixel Correction | 53 |
| 5.2.2 | Adaptive Noise Reduction | 53 |
| 5.2.3 | Black Level Correction | 54 |
| 5.2.4 | Vignette Correction | 54 |
| 5.2.5 | White Balance Correction | 55 |
| 5.2.6 | Demosaic | 55 |
| 5.2.7 | Unsharp Mask Filter | 56 |
| 5.2.8 | Color Correction Matrix | 56 |
| 5.3 | Output Config Tab | 57 |
| 5.3.1 | Warp | 57 |
| 5.3.2 | Tone Mapping Operator | 60 |
| 5.3.3 | 3D LUT | 60 |
| 5.3.4 | Logo | 61 |
| 5.3.5 | 1D LUT | 61 |
| 5.3.6 | Frame Writer | 63 |
| 5.4 | Pipeline Statistics Tab | 64 |
| 5.4.1 | Black Level Statistics | 65 |
| 5.4.2 | White Balance Statistics | 66 |
| 5.4.3 | Histogram Statistics | 67 |
| 6.0 | Document Revision History | 68 |

List of Figures

| | | |
|--------------|--|----|
| Figure 3-1. | 4K Camera Demo Design Block Diagram | 12 |
| Figure 3-2. | Black Level Statistics Block Diagram | 14 |
| Figure 3-3. | Defective Pixel Correction Block Diagram | 15 |
| Figure 3-4. | A Bayer CFA for an 8x8 section of an image and an example pixel neighborhood for green, red, and blue pixels | 15 |
| Figure 3-5. | Adaptive Noise Reduction Block Diagram | 16 |
| Figure 3-6. | BLC Function | 17 |
| Figure 3-7. | Black Level Correction Block Diagram | 18 |
| Figure 3-8. | Effects of Reflection Around Zero | 18 |
| Figure 3-9. | Vignette Correction Block Diagram | 19 |
| Figure 3-10. | A Mesh Zone with a Pixel of Interest | 20 |
| Figure 3-11. | White Balance Statistics Block Diagram | 20 |
| Figure 3-12. | Example of ratio calculation for 2x2 virtual pixels for a 6x6 Section of Image | 21 |
| Figure 3-13. | Packing Order of the Zones within a Region of Interest | 21 |
| Figure 3-14. | White Balance Correction Block Diagram | 22 |
| Figure 3-15. | An example of a 2x2 RGGB Bayer Color Filter Array (for an 8x8 pixel section of the image) | 23 |
| Figure 3-16. | Demosaic Block Diagram | 23 |
| Figure 3-17. | Histogram Statistics Block Diagram | 24 |
| Figure 3-18. | Warp Block Diagram | 26 |
| Figure 3-19. | Before (left) and after (right) TMO is applied to an example image | 27 |
| Figure 3-20. | TMO Block Diagram | 27 |
| Figure 3-21. | 3D LUT Color Transform Examples | 28 |
| Figure 3-22. | 3D LUT Block Diagram | 28 |
| Figure 3-23. | LUTCalc Format Settings | 29 |
| Figure 3-24. | 1D LUT Block Diagram | 31 |
| Figure 4-1. | Locating EMIF IP files | 36 |
| Figure 4-2. | Adding calibration file to the FPGA EMIF | 38 |
| Figure 4-3. | Adding calibration file to the HPS EMIF | 38 |
| Figure 4-4. | Cable Connections | 40 |
| Figure 4-5. | Switch Configurations | 40 |
| Figure 4-6. | Camera Cable Connection | 41 |
| Figure 4-7. | JTAG Settings | 42 |
| Figure 5-1. | ifconfig Example Output | 45 |
| Figure 5-2. | Input Config Tab | 46 |
| Figure 5-3. | Sensor Control | 46 |
| Figure 5-4. | Sensor Color Profile | 47 |
| Figure 5-5. | Test Pattern Generator | 48 |
| Figure 5-6. | Input Select | 49 |
| Figure 5-7. | Clipper | 49 |
| Figure 5-8. | Clipper Region of Interest | 50 |
| Figure 5-9. | Autoexposure Controls | 51 |
| Figure 5-10. | Autoexposure Region of Interest Editor | 52 |
| Figure 5-11. | ISP Pipeline Tab | 52 |
| Figure 5-12. | Defective Pixel Correction | 53 |

| | | |
|--------------|--------------------------------|----|
| Figure 5-13. | Adaptive Noise Reduction | 53 |
| Figure 5-14. | Black Level Correction | 54 |
| Figure 5-15. | Vignette Correction | 54 |
| Figure 5-16. | White Balance Correction | 55 |
| Figure 5-17. | Demosaic | 55 |
| Figure 5-18. | Unsharp Mask Filter | 56 |
| Figure 5-19. | Color Correction Matrix | 56 |
| Figure 5-20. | Output Config Tab | 57 |
| Figure 5-21. | Warp Full Controls | 58 |
| Figure 5-22. | Warp Corner Controls | 59 |
| Figure 5-23. | Warp Arbitrary Controls | 59 |
| Figure 5-24. | TMO Controls | 60 |
| Figure 5-25. | Logo | 61 |
| Figure 5-26. | 1D LUT | 62 |
| Figure 5-27. | Custom Curve Editor | 63 |
| Figure 5-28. | Frame Writer | 63 |
| Figure 5-29. | Pipeline Statistics Tab | 64 |
| Figure 5-30. | Black level Statistics | 65 |
| Figure 5-31. | White Balance Statistics | 66 |
| Figure 5-32. | Histogram Statistics | 67 |

List of Tables

| | | |
|------------|---------------------------|----|
| Table 1-1. | Terminology | 8 |
| Table 1-2. | Reference Documents | 10 |

1.0 Introduction

The 4K Camera Agilex 5 FPGA E-Series 065B Arrow Eagle Board Demo Design showcases a practical glass-to-glass camera solution. The exclusive support for industry-standard Mobile Industry Processor Interface (MIPI) D-PHY and MIPI CSI-2 interface on Agilex 5 FPGAs provides a powerful tool for camera product development. Supporting up to 2.5Gbps per lane and up to 8x lanes per MIPI interface enables seamless data reception from a 4K image sensor to the FPGA fabric for further processing. The MIPI CSI-2 IP converts pixel data to an AXI4-Streaming output, enabling connectivity to other IP cores within Intel's Video and Vision Processing (VVP) Suite.

This demo design includes an Image Signal Processing (ISP) subsystem incorporating many standard cameras processing IP cores such that raw sensor image data can be processed into RGB video. The ISP enables auto white balance and exposure correction by continuously collecting and analyzing runtime statistical data using the embedded Hard Processor System. This solution also includes adaptive local tone mapping for handling wide dynamic range scenes, 1DLUT and 3DLUT IPs for color transformations and HDR conversion, and a high-performance Warp IP core for geometric distortion correction. 4Kp30 streaming video is output via Altera DisplayPort IP.

The Demo Design runs with Quartus Prime Design Suite v24.2.

1.1 Features

- Selectable Input
 - a. MIPI D-PHY connectivity to Raspberry Pi High-Quality Camera (12.3 megapixels Sony IMX477 Sensor) with MIPI CSI-2 interface
 - b. Internal TPG supporting SMPTE Colorbars, Solid Colors, and moving Zone Plate patterns
 - c. 4Kp30 12bit Bayer
 - d. AXI4-Streaming interface
- ISP Pipeline (12bit Bayer)
 - a. Clipper
 - b. Image Signal Processing, including Black Level and White Balance Correction, Demosaic, Defective Pixel Correction, Adaptive Noise Reduction, Vignetter Correction, and Statistics.
- Video Pipeline (10bit RGB)
 - a. Sharpening
 - b. Warp IP
 - c. Tone Mapping Operator IP
 - d. 3D LUT IP
 - e. 1D LUT IP (12bit)

- Output
 - a. 4Kp30 10b DisplayPort
- Runtime Processing
 - a. Quad-core ARM Cortex CPU

1.2 Terminology

Table 1-1. Terminology

| Term | Description |
|------|--------------------------------------|
| AE | Auto Exposure |
| ANR | Adaptive Noise Reduction |
| API | Application programming interface |
| AWB | Auto White Balance |
| AXI | Advanced extensible interface |
| BLC | Black Level Correction |
| BLS | Black Level Statistics |
| BPS | Bits per-symbol |
| CCM | Color Correction Matrix |
| CFA | Color Filter Array |
| CPU | Central processing unit |
| CSC | Color Space Converter |
| CSR | Control/status registers |
| DDR | Double data-rate |
| DHCP | Dynamic host configuration protocol |
| DMA | Direct Memory Access |
| DMS | Demosaic |
| DP | DisplayPort |
| DPC | Defective Pixel Correction |
| EMIF | External memory interface |
| EOTF | Electrical-Optical Transfer Function |
| FMC | FPGA mezzanine card |
| FPGA | Field programmable gate array |
| FPS | Frames per second |
| FW | Firmware |
| GPIO | General purpose input-output |
| GUI | Graphical user interface |
| HDR | High-dynamic range |

| | |
|---------------|---|
| HDR HLG | HDR Hybrid Log Gamma |
| HDR PQ | HDR Perceptual Quantization |
| HPS | Hard Processor System |
| HS | Histogram Statistics |
| HW | Hardware |
| IP | Intellectual property |
| IO, I/O | Input/output |
| ISP | Image Signal Processing |
| JTAG | Joint Test Action Group |
| LAN | Local area network |
| LED | Light emitting diode |
| LSB | Least significant bit |
| LUMA | Luminance |
| LUT | Look Up Table |
| MIPI | Mobile Industry Processor Interface |
| MSB | Most significant bit |
| OBR | Optical Black Region |
| OETF | Opto-Electrical Transfer Function |
| OOTF | Opto-Optical Transfer Function |
| PIP | Pixels in parallel |
| PPI | PHY Protocol Interface |
| QPDS | Quartus Prime Design Software |
| RAM | Random-access memory |
| RGB | Red, green, and blue |
| RMS | Remosaic |
| ROI | Region Of Interest |
| RPi | Raspberry Pi |
| RX | Receive |
| SD | Secure Digital |
| SoC | System-on-chip |
| SSH | Secure shell |
| SW, S/W | Software |
| SW1, SW2, ... | Switch 1, Switch 2, ... |
| TMO | Tone Mapping Operator |
| TPG | Test Pattern Generator |
| TX | Transmit |
| UART | Universal asynchronous receiver-transmitter |

| | |
|-----|-----------------------------|
| UHD | Ultra-high definition |
| UI | User Interface |
| USM | Unsharp Mask |
| VC | Vignette Correction |
| VVP | Video and Vision Processing |

1.3 Reference Documents

Table 1-2. Reference Documents

| Document | Document No./Location |
|---|---|
| QPDS 24.2 | https://www.intel.com/content/www/us/en/software-kit/826844/intel-quartus-prime-pro-edition-design-software-version-24-2-for-windows.html |
| Video and Vision Processing Suite Altera FPGA IP User Guide | https://www.intel.com/content/www/us/en/docs/programmable/683329/24-2/about-the-video-and-vision-processing-suite.html |
| Agilex 5 E Series AXE5 Eagle Development Platform | https://github.com/ArrowElectronics/Agilex-5/wiki/Agilex-5-E-Series-AXE5-Eagle-Development-Platform |
| Bitec DP FMC | https://bitec-dsp.com/product/fmc-displayport-daughter-card-revision-8/ |
| Raspberry Pi High-Quality Camera | https://www.raspberrypi.com/products/raspberry-pi-high-quality-camera/ https://thepihut.com/products/flex-cable-for-raspberry-pi-camera-or-display-24-610mm https://thepihut.com/products/mini-tripod-for-raspberry-pi-high-quality-camera |
| Ultra Wide-Angle Lens | https://thepihut.com/products/ultra-wide-angle-c-mount-lens-for-raspberry-pi-hq-camera-3-2mm-focal-length |
| CRUVI ST4 to MIPI FFC Adapter | https://shop.trenz-electronic.de/en/CR00300-01-Camera-Adapter |
| Altera FPGA Streaming Video Protocol Specification | https://www.intel.com/content/www/us/en/docs/programmable/683397/current/about-the-intel-fpga-streaming-video.html |
| AMBA 4 AXI4-Stream Protocol Specification | https://developer.arm.com/documentation/ih0051/a/ |
| Avalon Interface Specifications – Avalon Streaming Interfaces | https://www.intel.com/content/www/us/en/docs/programmable/683091/20-1/streaming-interfaces.html |

2.0 Release Version

FPGA firmware version v3.2.0.
Software version v1.2.0.

2.1 Quartus Prime Version

This release is intended to be used with Intel Quartus Prime Pro Edition Software version 24.2.

2.2 Changes from the Previous Version

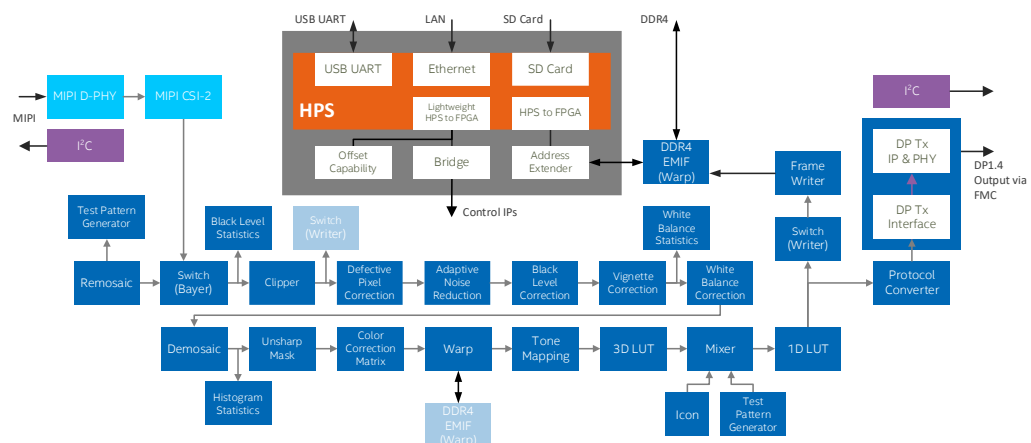
Initial version.

3.0 Functional Description

The 4K Camera Demo Design showcases a practical glass-to-glass 4Kp30 camera solution using standard Altera Connectivity and Video and Vision Processing (VVP) Suite IP Cores available through the Quartus Prime Design Software (QPDS).

Video is ingested through an industry-standard MIPI directly connected sensor and processed through the Image Signal Processing (ISP) pipeline before output through DisplayPort (DP). The pipeline also utilizes additional non-ISP IP Cores from the VVP Suite, such as Test Pattern Generators (TPG) and VVP AXI4-S Switches. Furthermore, the design runs an embedded Linux Software Application (SW App) on the Hard Processor System (HPS) to provide real-time auto white balance (AWB) and auto exposure (AE) algorithms.

Figure 3-1. 4K Camera Demo Design Block Diagram



3.1 MIPI Input

The Raspberry Pi (RPi) High-Quality Camera provides a 2-lane MIPI interface running at a maximum of 2400Mbps per lane. The Camera sensor is a Sony IMX477 that outputs Bayer up to 12 bits per pixel.

The Altera MIPI D-PHY IP interfaces the FPGA directly to the RPi Camera via a MIPI cable. The Demo Design showcases 4Kp30 @12bits, and a MIPI lane rate of 1500Mbps provides sufficient bandwidth. The MIPI D-PHY is, therefore, configured for x2 lanes@1500Mbps with no skew calibration (as it's only needed for rates greater than 1500Mbps) and non-continuous clock mode. The camera has additional pins, including a Power Enable control and a slave I2C interface for powering and setting up the camera. The Power Enable pin is connected to an Altera PIO (Parallel Input Output) IP, which has an agent Avalon memory-mapped interface that allows runtime control via the HPS. Likewise, the I2C

interface is connected via an HPS I2C Controller. When the SW App starts, it powers the camera and sets it up for 1500Mbps lane speed, 3840x2160 resolution, raw12 pixel samples, no skew calibration, and blanking for 30FPS.

The Altera MIPI CSI-2 IP is connected to the MIPI D-PHY IP using a 2-lane 16-bit PHY Protocol Interface (PPI) bus. The bus effectively runs at 93.312MHz (3840x2160x30FPSx12b/2x16b). The CSI-2 IP is configured to output VVP AXI4-S Video Packets @ 4PiP (Pixels In Parallel) using a 297MHz clock.

3.2 Input TPG

The Input TPG feature allows you to test the Demo Design without the Camera input. It uses the VVP Test Pattern Generator IP and a non-QPDS IP called Remosaic (RMS) (supplied with the source project). The TPG has an RGB output, which the RMS turns into Bayer by discarding color information based on the camera's Color Filter Array (CFA) configuration. The TPG features several modes, including SMPTE Colorbars, solid colors, and zone plate patterns (not in all example designs). A VVP Switch is used to select the Input source.

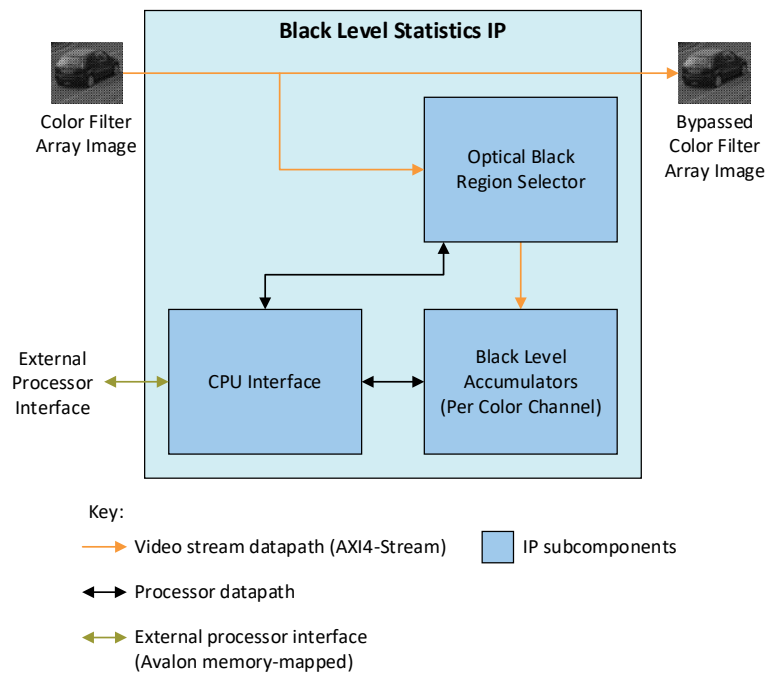
Related Information

- [VVP TPG Intel FPGA IP](#)
- [VVP Switch Intel FPGA IP](#)

3.3 Black Level Statistics

The Black Level Statistics IP (BLS) accumulates pixel values in a Region of Interest (ROI), which is usually associated with a shielded area of an imaging sensor called an Optical Black Region (OBR). The statistics are used to keep the sensor's black level value constant and compensate for any deviations due to temperature changes, for example. However, the RPi Camera does not pass the OBR, so a separate calibration step has been done to obtain a set of BLS values.

Figure 3-2. Black Level Statistics Block Diagram



The BLS calculates statistics across an OBR of a 2x2 CFA input from the sensor. The IP has dedicated accumulators for each CFA color channel that sum black pixel values. The SW App uses the BLS to set the Black Level Correction IP. The BLS passes the input image untouched to its output.

Related Information

[VVP Black Level Statistics Intel FPGA IP](#)

3.4 Clipper

The Clipper IP crops an active area from an input image and discards the remainder. It is used to remove the OBR from the sensor input. However, since the RPi Camera does not pass the OBR, the Clipper is configured to pass the entire 4K image.

Related Information

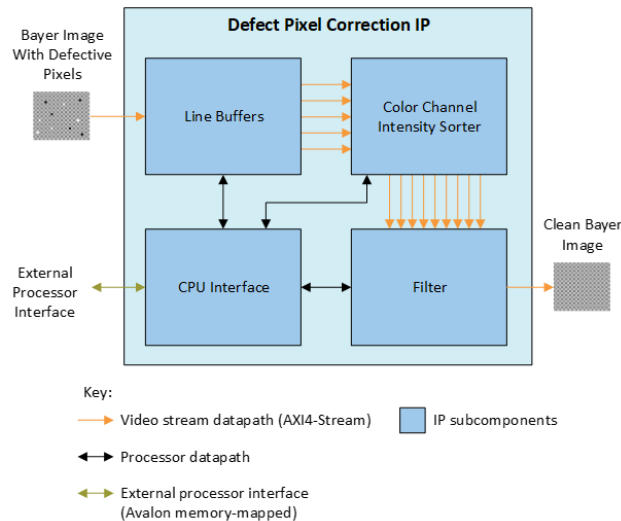
[VVP Clipper Intel FPGA IP](#)

3.5 Defective Pixel Correction

The Defective Pixel Correction IP (DPC) removes impulse noise associated with defective pixels in the sensor image. Such impulse noise is usually the result

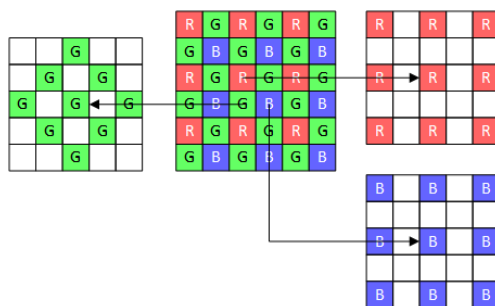
of hardware (HW) corruption in image sensor circuitry, and it manifests itself in specific pixels that show drastically wrong pixel intensities. The DPC operates on the 2x2 Bayer CFA image, identifies defective pixels using a configurable non-linear filter, and corrects them.

Figure 3-3. Defective Pixel Correction Block Diagram



The DPC works on a 5x5 pixel neighborhood. The Color Channel Intensity Sorter selects pixels closest to the center pixel and sorts them according to their intensities. A filter calculates a corrected pixel value from the sorted group of pixels depending on the sensitivity setting set via the agent Avalon memory-mapped interface connected to the HPS.

Figure 3-4. A Bayer CFA for an 8x8 section of an image and an example pixel neighborhood for green, red, and blue pixels



The DPC dynamically identifies and filters defect pixels and does not support static DPC. When you set the sensitivity level to the weakest value, the pixel value is altered only if its original value falls outside the value range of its 5x5 neighborhood. As the sensitivity increases, the range reduces closer to the median value.

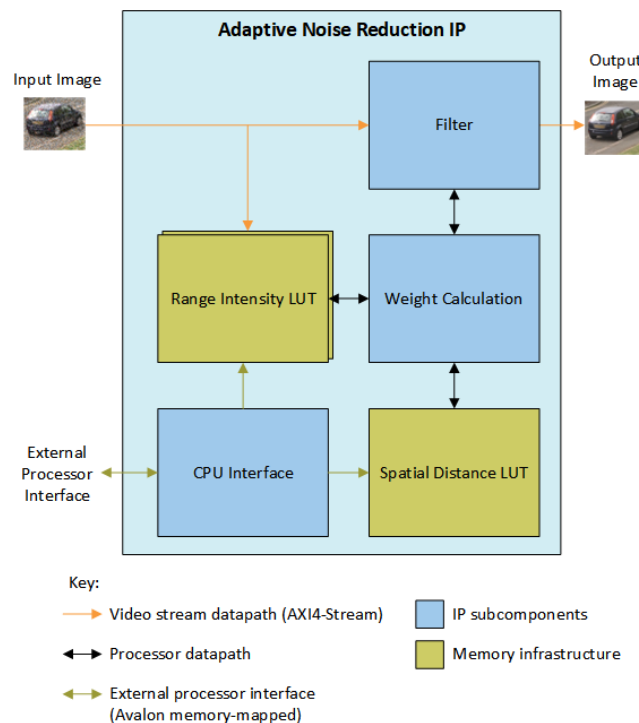
Related Information

[VVP Defective Pixel Correction Intel FPGA IP](#)

3.6 Adaptive Noise Reduction

The Adaptive Noise Reduction IP (ANR) is an edge-preserving smoothing filter that mainly reduces the independent pixel noise of an image. The IP operates on the 2x2 Bayer CFA image.

Figure 3-5. Adaptive Noise Reduction Block Diagram



The ANR IP uses a spatial weighted averaging filter that analyzes the scene and correlates similar pixels dynamically while generating the weights on the fly. The IP utilizes two LUTs when correlating the pixels, one for correlating pixel intensities and the other for correlating the spatial distance between the pixels.

The intensity range LUT is calibrated offline using the difference in noise level between two video images with identical content but different temporal noise. Using a LUT allows you to program different denoising strengths across the pixel intensities. For example, you may opt for stronger denoising of dark content to reduce shadow noise more aggressively while preserving the details on the mid tones and highlights.

The spatial distance LUT is used to make the ANR more versatile. It is programmed to create a weight distribution from the center pixel to the neighboring pixels. Traditional distributions like a Hamming or Hanning window to reduce ringing artifacts can be used, or the same value entries can be used for a rectangular distribution to maximize denoising capability. By default, the software configures a Gaussian distribution into the spatial distance LUT.

3.7 Black Level Correction

The Black Level Correction IP (BLC) operates on a 2x2 CFA input image and adjusts the minimum brightness level of the image, ensuring an actual black value is represented by the minimum pixel intensity. The sensor typically adds a pedestal value to the image, which must be removed once the noise has been reduced. The BLC IP subtracts a pedestal value from the input video stream and scales the result back to the full dynamic range.

Figure 3-6. BLC Function

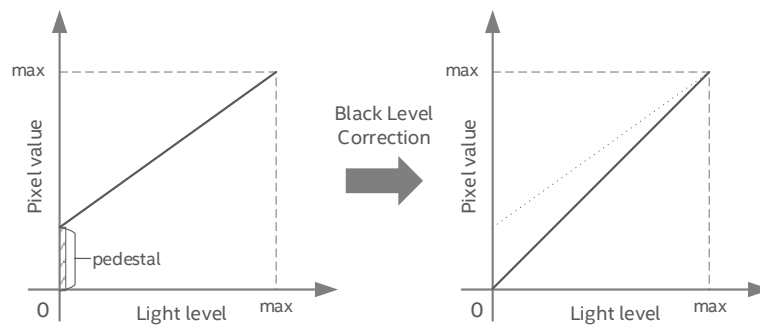
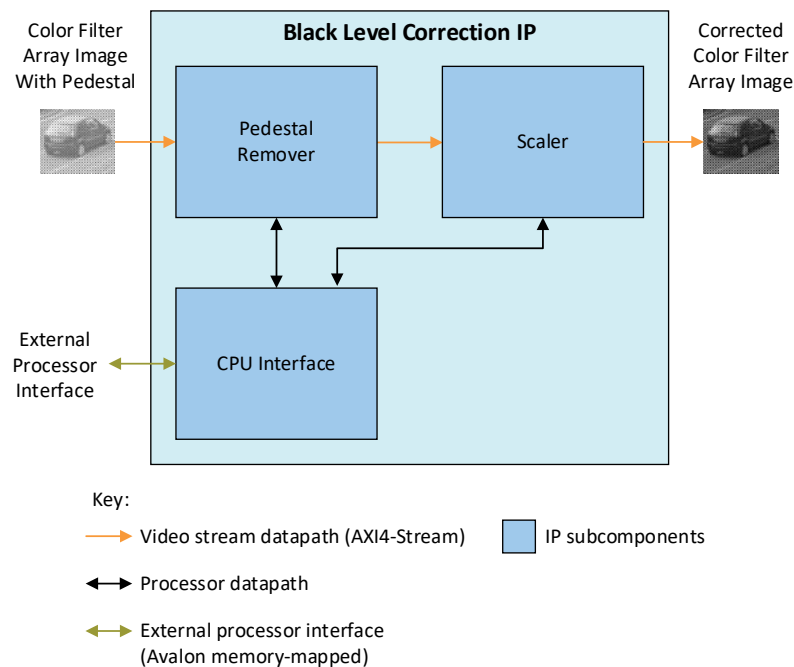
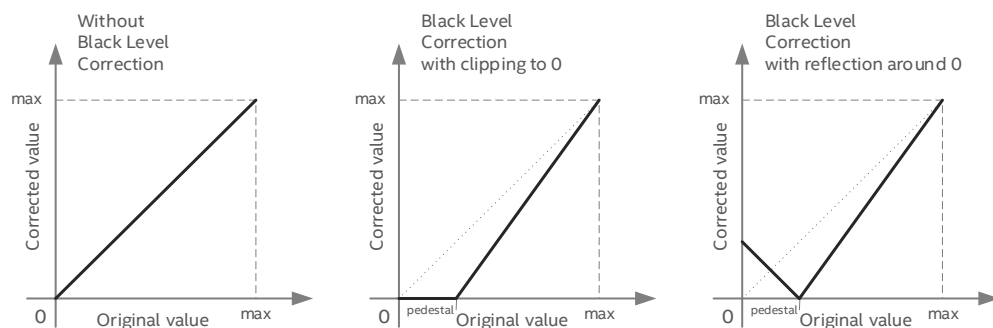


Figure 3-7. Black Level Correction Block Diagram



Using the BLS values, the SW App sets pedestal and scaler coefficient values for each of the 2x2 CFA color channels using the agent Avalon memory-mapped interface connected to the HPS. The pedestal remover subtracts the pedestal offset values from the input pixel values. The BLC reflects the negative values around zero.

Figure 3-8. Effects of Reflection Around Zero



The scaler part of BLC multiplies the pedestal remover value by the scaler coefficient, clipping to the maximum output pixel value should the calculation overflow.

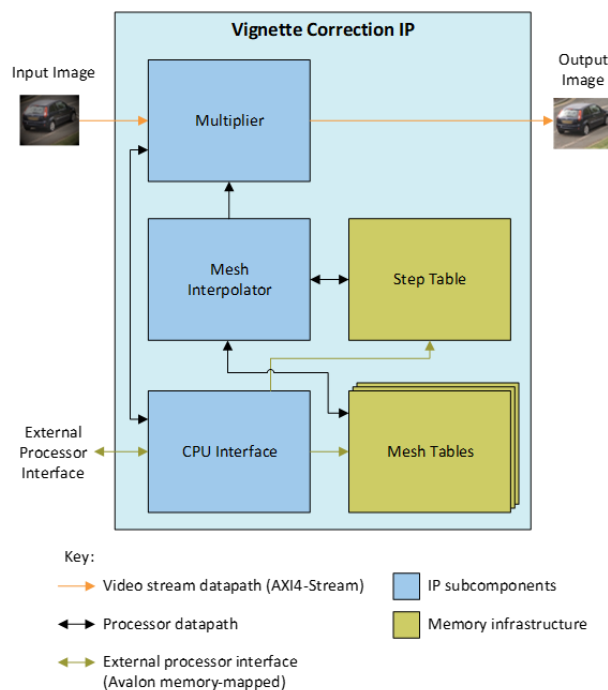
Related Information

[VVP Black Level Correction Intel FPGA IP](#)

3.8 Vignette Correction

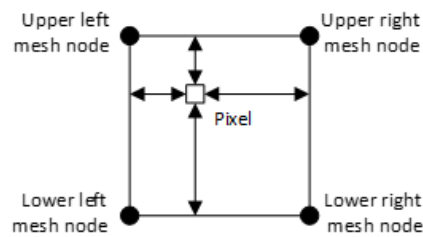
The Vignette Correction IP (VC) compensates for non-uniform intensity across the image caused by uneven light-gathering limitation of the sensor and optics. In the usual case, the non-uniformity can be caused by lens geometry transforms such as zoom, aperture, etc., where the center of the lens gathers more light compared to the outer regions. The VC corrects uniformity using a runtime mesh of coefficients and interpolating coefficients for any given pixel.

Figure 3-9. Vignette Correction Block Diagram



VC uses a rectangular mesh of coefficients to adjust the pixel intensities across an image, therefore tuning out vignetting and other sensor non-uniformities. VC operates independently on the 4 color channels of the 2x2 CFA input image. For all color channels, the mesh coefficients are derived during a calibration process in a controlled imaging environment, and the SW App writes them to the VC via the agent Avalon memory-mapped interface connected to the HPS. The precision of a mesh coefficient is fixed-point unsigned 8.11, with 8 integer bits and 11 fractional bits. The mesh divides the image into rectangular zones, with mesh points residing at the corners of the zones. For each pixel, the IP selects the 4 mesh points corresponding to the corners of the pixel's zone and interpolates them using the distance from the pixel location to the mesh points. A multiplier scales the input pixels using this interpolated coefficient.

Figure 3-10. A Mesh Zone with a Pixel of Interest



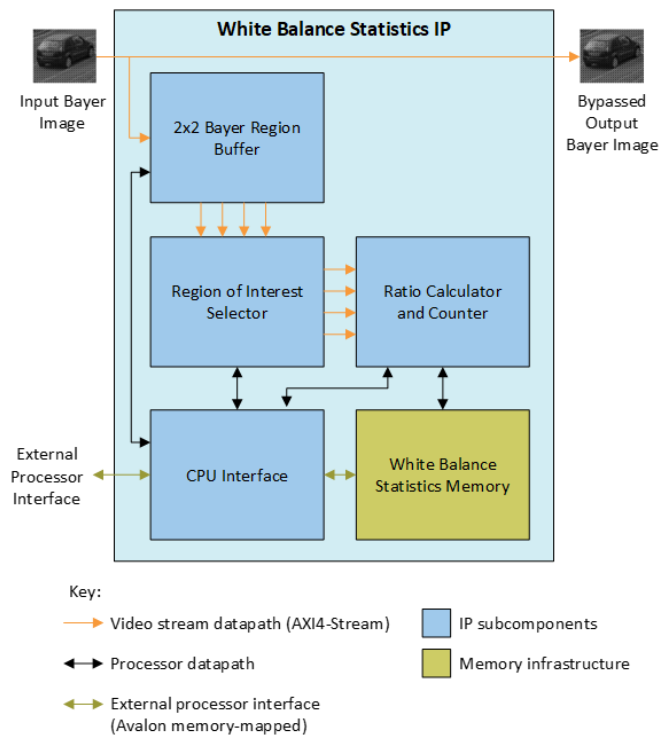
Related Information

[VVP Vignette Correction Intel FPGA IP](#)

3.9 White Balance Statistics

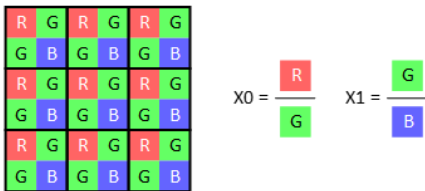
The White Balance Statistics IP (WBS) operates on a 2x2 Bayer CFA image and calculates red-green and blue-green ratios within a Region of Interest (ROI). The ROI is divided into 7x7 zones, and the WBS calculates independent statistics for all 49 zones. The SW App uses the WBS to set the White Balance Correction IP.

Figure 3-11. White Balance Statistics Block Diagram



A 2x2 Bayer region buffer creates a virtual pixel by grouping 4 pixels. The Ratio Calculator and Counter then calculate red-green and blue-green ratios for each virtual pixel within the ROI.

Figure 3-12. Example of ratio calculation for 2x2 virtual pixels for a 6x6 Section of Image



It then checks both ratios against lower and upper range thresholds and increments a zone counter if both ratios fall between their respective threshold values. If at least one of the ratios is out of range, both ratios for the 2x2 virtual pixel are discarded and do not contribute to the final statistics of that zone. Ratios of all virtual pixels that are not marked out of range are accumulated and transferred to their respective zone memory at the end of each zone. The SW App reads the zone memory via the agent Avalon memory-mapped interface connected to the HPS for every image. It calculates an average ratio by dividing the accumulated ratio by the number of counted 2x2 virtual pixels. If the number of counted pixels is too low, it indicates a zone with mixed content and is unsuitable for calculating white imbalance in the image.

The WBS passes the input image untouched to its output.

Figure 3-13. Packing Order of the Zones within a Region of Interest

ROI

| | | | | |
|---------|---------|---------|-------|---------|
| Zone 1 | Zone 2 | Zone 3 | | Zone 7 |
| Zone 8 | Zone 9 | Zone 10 | | Zone 14 |
| ... | | | | ... |
| Zone 43 | Zone 44 | Zone 45 | | Zone 49 |

Related Information

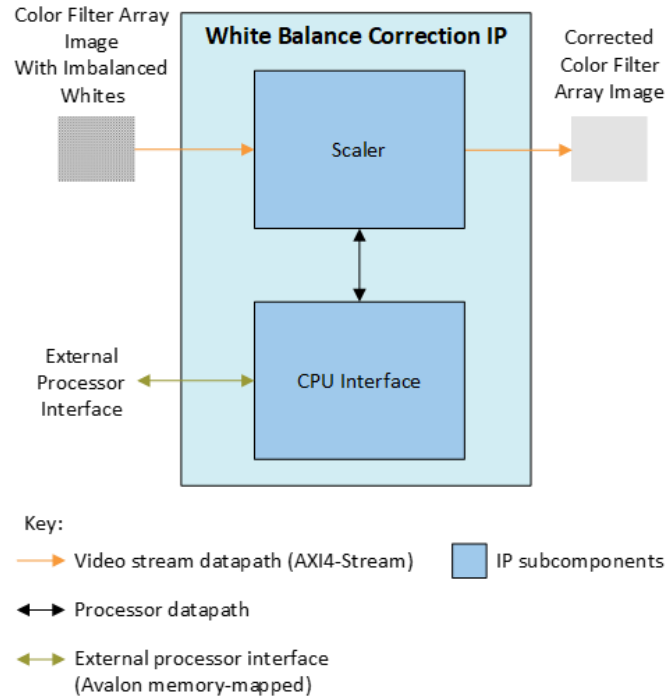
[VVP White Balance Statistics Intel FPGA IP](#)

3.10 White Balance Correction

The White Balance Correction IP (WBC) adjusts colors in a CFA image to eliminate color casts, which occur due to variations in color temperatures of

lighting conditions or differences in the light sensitivity of the CFA. Such processing ensures that white objects appear truly white without unwanted color tinting.

Figure 3-14. White Balance Correction Block Diagram



The WBC scaler multiplies the color channels of a 2x2 CFA input image by the scaler coefficient, clipping to the maximum output pixel value should the calculation overflow. The SW App uses the White Balance Statistics to generate scaler coefficient values for the WBC. It writes them via the agent Avalon memory-mapped interface connected to the HPS.

Related Information

[VVP White Balance Correction Intel FPGA IP](#)

3.11 Demosaic

The Demosaic IP (DMS) is a color reconstruction IP for converting a 2x2 Bayer CFA input image to an RGB output image. The DMS interpolates missing colors for each pixel based on its neighboring pixels.

Figure 3-15. An example of a 2x2 RGGB Bayer Color Filter Array (for an 8x8 pixel section of the image)

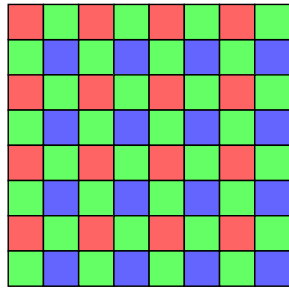
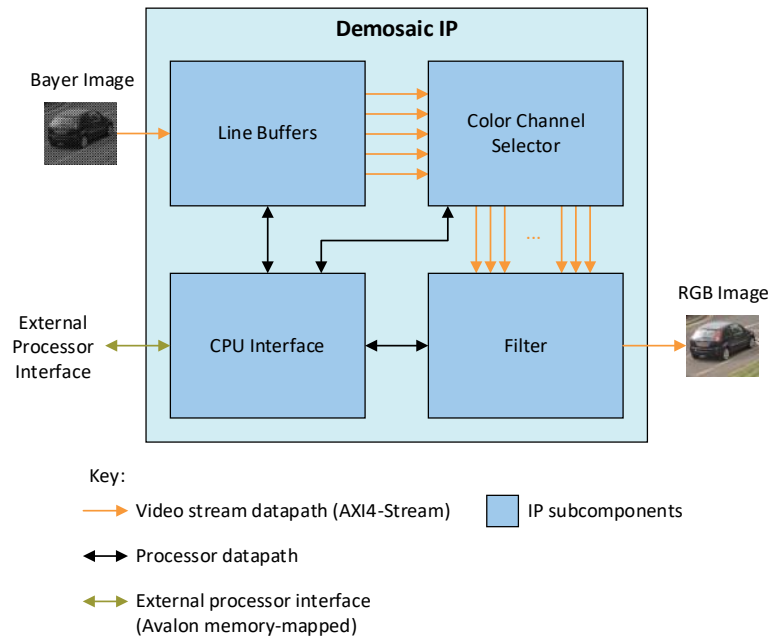


Figure 3-16. Demosaic Block Diagram



The DMS analyzes the neighboring pixels for every CFA input pixel and interpolates the missing colors to produce an RGB output pixel. The IP uses line buffers to construct pixel neighborhood information. The Color Channel Selector maps the pixels in the neighborhood depending on the position on the 2x2 CFA for the interpolating filter to calculate the RGB output.

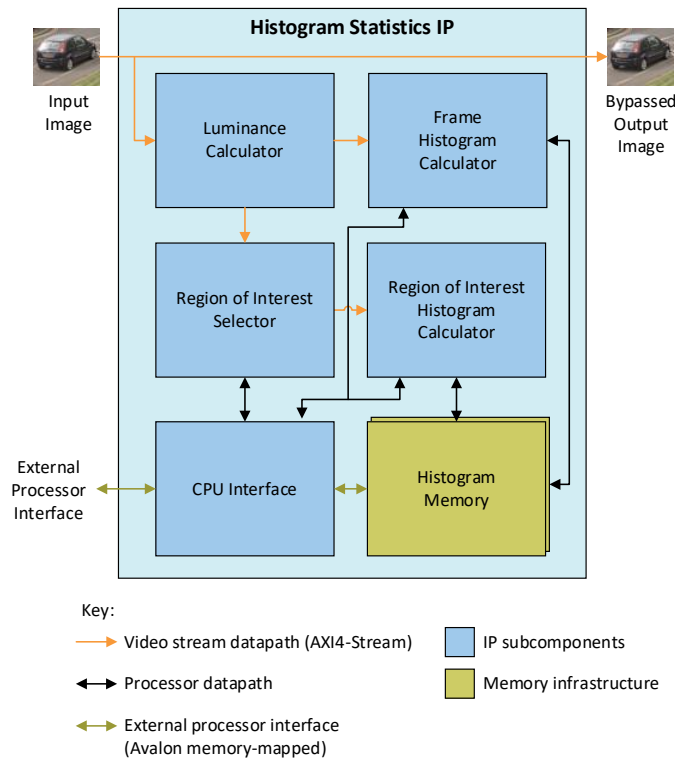
Related Information

- [VVP Demosaic Intel FPGA IP](#)

3.12 Histogram Statistics

The Histogram Statistics (HS) operates on RGB images, analyses the pixel values for each frame, and collects data to form a luma histogram. The IP is used to adjust sensor exposure settings.

Figure 3-17. Histogram Statistics Block Diagram



The HS IP calculates two luminance histograms on the RGB input image - a whole image histogram and an ROI image histogram. The RGB to Luma conversion is performed according to ITU-R BT.709. The IP passes the input image untouched to its output.

Related Information

- [VVP Histogram Statistics Intel FPGA IP](#)

3.13 Unsharp Mask Filter

The Unsharp Mask IP (USM) receives images and outputs them after performing an unsharp mask operation.

An unsharp mask is first created by converting the RGB input image into luma, then passed through a low-pass Gaussian blur filter. The Luma Mask is subtracted from the original luma before being scaled by the strength setting

to obtain the sharpened Luma output image. The ratio of Luma output over Luma input is finally used to scale the RGB input image to produce the RGB output image.

The unsharp mask has an agent Avalon memory-mapped interface to allow runtime control, such as changing the strength. A setting of zero is used as the bypass mode.

Related Information

- [VVP Unsharp Mask Intel FPGA IP](#)

3.14 Color Correction Matrix

The Color Correction Matrix (CCM) functionality is provided by the VVP Color Space Converter IP (CSC). Using the UI, you can add additional post-ISP processing color effects. The SW App folds the required color matrix multiplications into a single LUT and updates the CSC IP using the agent Avalon memory-mapped interface connected to the HPS.

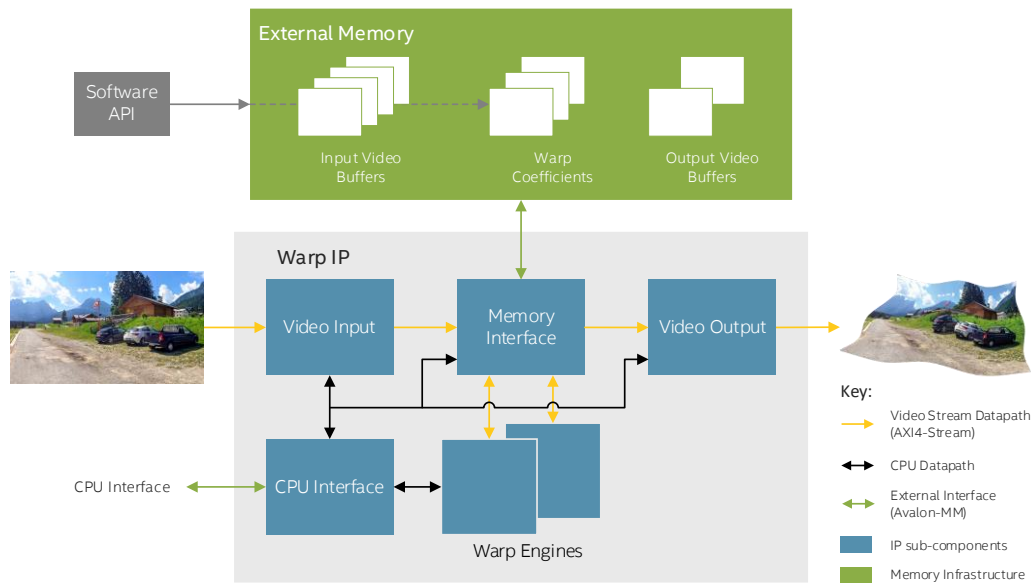
Related Information

- [VVP Color Space Converter Intel FPGA IP](#)

3.15 Warp

The Warp IP applies an arbitrary warp (or image transform) to an input image. It allows for lens distortion corrections (fisheye, for instance) and the ability to scale, rotate, and mirror the image.

Figure 3-18. Warp Block Diagram



The block diagram shows an external memory used to buffer the incoming and outgoing image data. The external memory also stores the coefficient tables used to control the Warp IP's operation and define the required image transform.

Warp engines operate on the buffered input images and reconstruct warped output images in the output buffers.

The Demo Design configures the Warp IP with two engines to support any arbitrary transformation at 4K resolution.

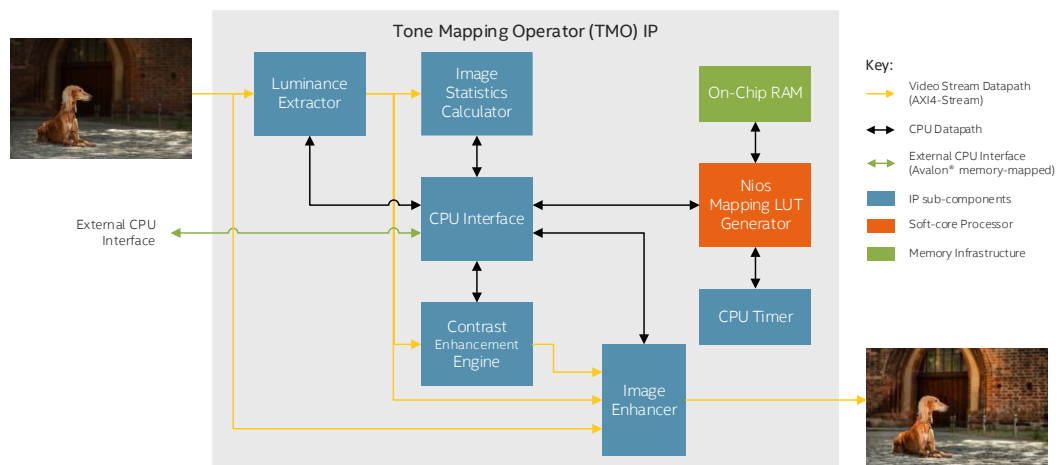
Related Information

- [Warp Intel FPGA IP](#)
- [VVP Warp Intel FPGA IP](#)

3.16 TMO

The Tone Mapping Operation IP (TMO) dynamically adapts the processing of an input image based on a regional (tile-based) approach. It improves the visibility of latent image detail and enhances the overall viewing experience.

Figure 3-19. Before (left) and after (right) TMO is applied to an example image

Figure 3-20. TMO Block Diagram


The luminance extractor converts RGB input to luma. The image statistics calculator uses luma to calculate a set of global and local statistics (4x4 grid) regarding the contrast of the input image. The LUT generator SW running on an embedded Nios V CPU analyzes the statistics, generates a set of mapping transfer functions, and converts them to LUTs. The contrast enhancement engine applies mapping transfer functions locally for better granularity. The image enhancer combines the information and calculates a set of weights that are to the input to generate contrast-enhanced output.

The TMO does not use image buffers; therefore, statistics collected for the previous image are used to enhance the current image.

An agent Avalon memory-mapped interface is connected to the HPS for IP control.

Related Information

- [Tone Mapping Operator \(TMO\) Intel FPGA IP](#)

- [VVP Tone Mapping Operator Intel FPGA IP](#)

3.17 3D LUT

The 3D LUT IP maps an image's color space to another using interpolated values from a LUT.

Typical applications include:

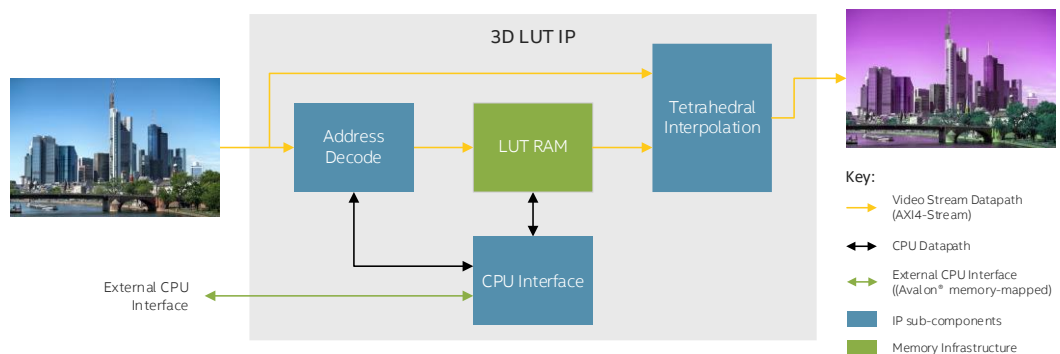
- Color space conversion
- Chroma keying
- Dynamic range conversion
- Artistic effects (sepia, hue rotation, color volume adjustment, etc.)

Figure 3-21. 3D LUT Color Transform Examples



(From top left: original, saturation, brightness increase, colorize (purple), colorize (green), desaturation)

Figure 3-22. 3D LUT Block Diagram



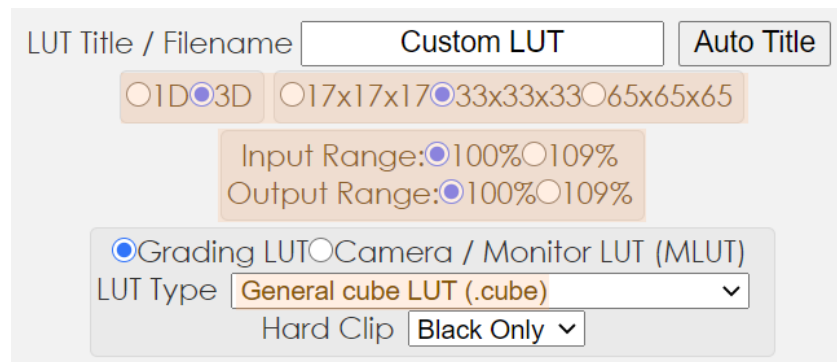
The 3D LUT uses the most significant bits (MSBs) of the 3 RGB color component inputs to retrieve data values from the LUT and the least significant bits (LSBs) to interpolate the final output value. An agent Avalon memory-mapped interface connected to the HPS handles the runtime control and LUT programming. The LUT memory is double-buffered, which not only allows for 2 separate LUTs to be loaded but also allows for a smooth transition to occur between them.

3.17.1 Generating LUT Files

Users are responsible for sourcing or generating LUTs for their products. LUTs are generally developed based on input (optical system, sensor, ISP, etc.) and display parameters. Various tools are available under open-source licenses to produce LUTs. One such tool is LUTCalc, which can be used online. When using it to generate LUTs for the 3D LUT IP, ensure that:

- The LUT is set to 3D
- Size matches the 3D LUT IP parameters (i.e. 17, 33, or 65 cube)
- Input and Output Range are 100%
- LUT Type is "General cube LUT (.cube)"

Figure 3-23. LUTCalc Format Settings



In general, any LUT file used with this design must follow these formatting conventions:

- RGB component order
- Components change first from left to right, i.e., R first, G second, B third
- The data type must match the IP GUI parameter and may either be:
 - normalized fixed- or floating-point numbers between 0.0 to 1.0
 - integers between 0 and 2LUT_DEPTH-1 (for example, 10-bit: 0 to 1023)
- The data type must be the same for the whole file

Related Information

- [3D LUT Intel FPGA IP](#)
- [VVP 3D LUT Intel FPGA IP](#)
- [LUTCalc GitHub page](#)

3.18 Logo Overlay

The Altera Logo Overlay feature uses a VVP Test Pattern Generator IP, a VVP Mixer IP, and a non-QPDS Icon IP (supplied with the source project). The TPG is used for the screensaver function and is configured as a 4K solid black image. It is also the base layer for the Mixer IP and is mixed with the ISP pipeline video layer and the Icon IP layer. The opacity of the Icon layer can be changed on the fly using the Avalon Memory-Mapped CPU interface.

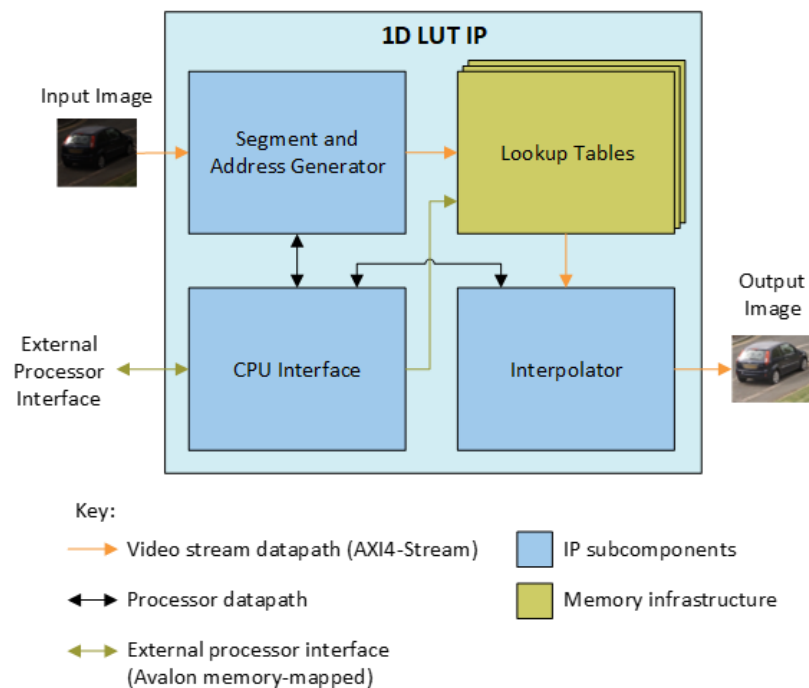
Related Information

- [VVP TPG Intel FPGA IP](#)
- [VVP Mixer Intel FPGA IP](#)

3.19 1D LUT

The 1D LUT IP uses a SW configurable LUT to apply a transfer function to the image. You may use it to implement OOTF, OETF, and EOTF transfer functions defined for video standards and legacy gamma compression or decompression. You may also change the LUT content arbitrarily to apply an artistic effect to the image.

Figure 3-24. 1D LUT Block Diagram



The 1D LUT IP calculates LUT addresses from the input pixels and interpolates fractional differences between LUT values to generate output pixels. The IP uses an independent LUT for each color plane. An agent Avalon memory-mapped interface connected to the HPS is used to configure the LUTs.

Related Information

- [VVP 1D LUT Intel FPGA IP](#)

3.20 Frame Writer

The Frame Writer IP receives and stores frames in FPGA external DDR memory. The frame writer has a host Avalon memory-mapped interface to allow connection to the external memory and an agent Avalon memory-mapped interface connected to the HPS to allow runtime control.

It operates in a single shot mode on RGB 12-bit images.

The HPS has a host Avalon memory-mapped interface to the FPGA external DDR, allowing it to read out captured images.

Related Information

- [VVP Frame Writer Intel FPGA IP](#)

3.21 Hard Processor System

The Hard Processor System (HPS) runs the embedded software used to set the external camera and internal IP and provide the auto white balance and auto exposure features. In addition, it runs a web server that allows you to interact with the demo via an Ethernet connection. The HPS has its own external DDR that is used exclusively for the SW. Additionally, the HPS is connected to the external FPGA fabric DDR to process Warp and Frame Writer IP data.

3.22 Miscellaneous

Other Altera Connectivity and VVP IP are used for the Demo Design.

Related Information

- [VVP Pixels in Parallel Converter Intel FPGA IP](#)
- [VVP Color Plane Manager Intel FPGA IP](#)
- [VVP Bits per Color Sample Adapter Intel FPGA IP](#)

4.0 Hardware and Software Setup

4.1 Pre-compiled Delivery Package

The deliverable design is HPS-first, which uses a combined SD card that contains both the software and firmware components and a JIC file:

```
4KCamera_FWa.b.c.jic
```

```
4KCamera_FWa.b.c_SWa.b.c.img
```

Where the numbers a, b, and c identify the software versions and subversions. The `.sof` file version is identified not by filename but by a unique sha1sum.

The `.img` file is the SD card image, which is to be written to a microSD card. It contains pre-compiled software.

The `.sof` file is the FPGA image and must be downloaded over JTAG to the hardware using the Quartus Programming Tool.

4.2 Source Delivery Package

The deliverable comes in two files:

```
Cam_Sol_v3_2_4K.tar
```

```
VvpIsp4kCameraDemo_v1.2.1.zip
```

Where the numbers a, b, and c (where applicable) identify the firmware and software versions and subversions.

The first deliverable file is the FPGA source project for building the FPGA using Quartus. Extract the delivery package using an uncompressing utility. For example, on Linux, use the command:

```
tar xzvf Cam_Sol_v3_2_4K.tar
```

The second deliverable is the Software source project. Extract the delivery package using an uncompressing utility.

4.3 Hardware and Software Requirements

The demo design requires three classes of hardware/software to run the design or further develop the SoC:

- A minimum set of hardware and software is required to run the demo design.

- A set of Intel Quartus Prime Design Suite components are required for FPGA development.
- A set of software tools and environments are required for software development.

4.3.1 Minimum Hardware and Software Requirements

The design requires the following hardware to run:

- Intel Agilex 5 FPGA E-Series 065B Premium Development Kit and PSU with the following standard components installed:
 - Hard Processor System expansion board
 - Minimum 8GB U3 Micro SD card
- Raspberry Pi High-Quality Camera Module (12.3 megapixel Sony IMX477 sensor):
 - With CS/C mount
 - Note that if the sensor body has been shipped with a removable outer ring (where the lens screws onto), you may have to remove it and screw the lens directly onto the thread to get a clear image. See below:



- Wide-angle Lens
- Tripod
- 15 pin flat MIPI cable (20cm length)
- CRUVI ST4 to MIPI FFC Adapter
- USB-Blaster II Download Cable
- Bitec DisplayPort FMC Daughter Card – Revision 8
- DisplayPort 2.0 (or HDMI with converter dongle) video sink

- 3840x2160 resolution up to 30 fps with 10b RGB support
- Network device with ethernet connection (with DHCP server)
- Application interface device (PC, tablet, phone, etc.) on the same local network that can run a web browser
- Terminal device (PC, tablet, phone, etc.) with a USB port
- Cables:
 - DisplayPort cable when the monitor supports DisplayPort input, or HDMI cable with 4kp60 converter dongle. Note that a high-quality dongle should be used as cheaper alternatives may not work.
 - Ethernet cable for network connectivity
 - Mini-USB B serial cable for HPS terminal connection
 - Micro-USB B JTAG Cable for FPGA programming

The design requires the following software to run:

- Quartus 24.2 Programmer Tool Installed
 - Ensure the ByteBlaster II USB Driver is installed
- Web browser on the application interface device
- Terminal emulator (PuTTY, TeraTerm, etc.) and FTDI* FT232R USB UART drivers installed on the terminal device
- SD Card Formatter Utility (from SD Association)
- SD Card Imager Utility (Win32 Disk Imager, for example)

The above is the minimum list of hardware and software required to run the pre-compiled demo design. Depending on your setup, you may require additional hardware, cables, and software.

4.3.2 Requirements for FPGA Development

The FGPA developers require the following components:

- Microsoft® Windows® or Linux PC or workstation
- The Intel Quartus Prime Design Suite Version 24.2, which includes the following:

Microsoft and Windows are trademarks of the Microsoft group of companies.

- Intel Quartus Prime Pro Edition
- Intel Agilex 5E device support
- Platform Designer
- Intel IP Library (including the VVP suites)

4.3.2.1 FPGA Development – Getting Started

- Ensure you have extracted the FPGA source deliverable package. On Windows it is preferable to extract the package in a folder with a short full path like `C:\demo\` to prevent maximum path character limit during compilation.
- Load Quartus and open the "agilex5_vvpisp.qpf" file or navigate to the "quartus" directory and for Linux type:

```
quartus agilex5_vvpisp.qpf&
```

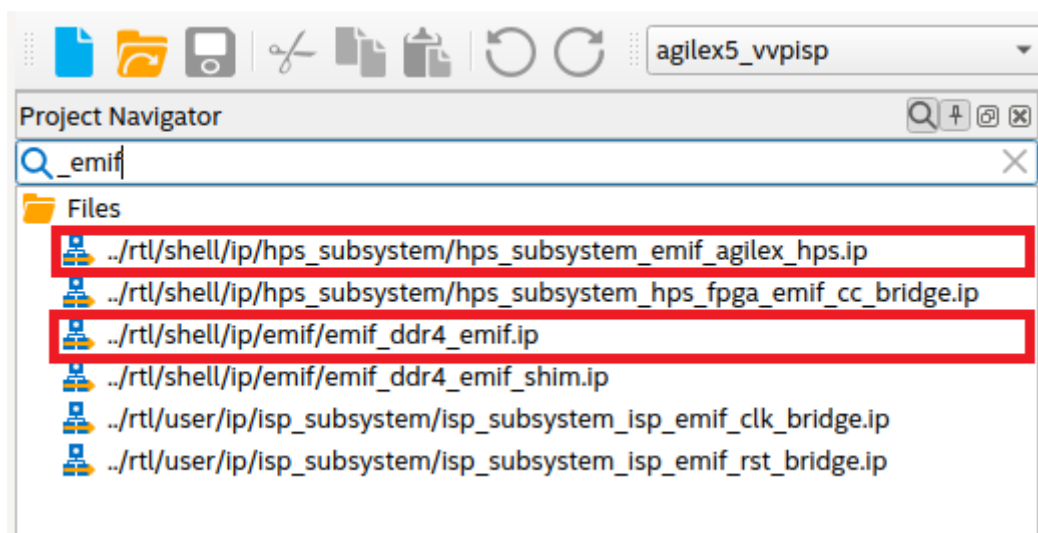
4.3.2.2 FPGA Development – Adding EMIF Calibration Files

Due to a limitation in Intel Quartus Prime Design Suite Version 24.2 you configure calibration file paths in the EMIF IPs manually after opening the project. Locate two EMIF IP files in the Quartus project navigator (Figure 4-1):

```
emif_ddr4_emif.ip
```

```
hps_subsystem_emif_agilex_hps.ip
```

Figure 4-1. Locating EMIF IP files



Add calibration files to the EMIF IPs as shown in Figure 4-2 and Figure 4-3 by right clicking on the selected file and selecting "Open".

Both EMIF IPs in the design use the file "LPDDR4_1GB_1333.qprs" located in the "non_qpds_ip/shell" directory.

Figure 4-2. Adding calibration file to the FPGA EMIF

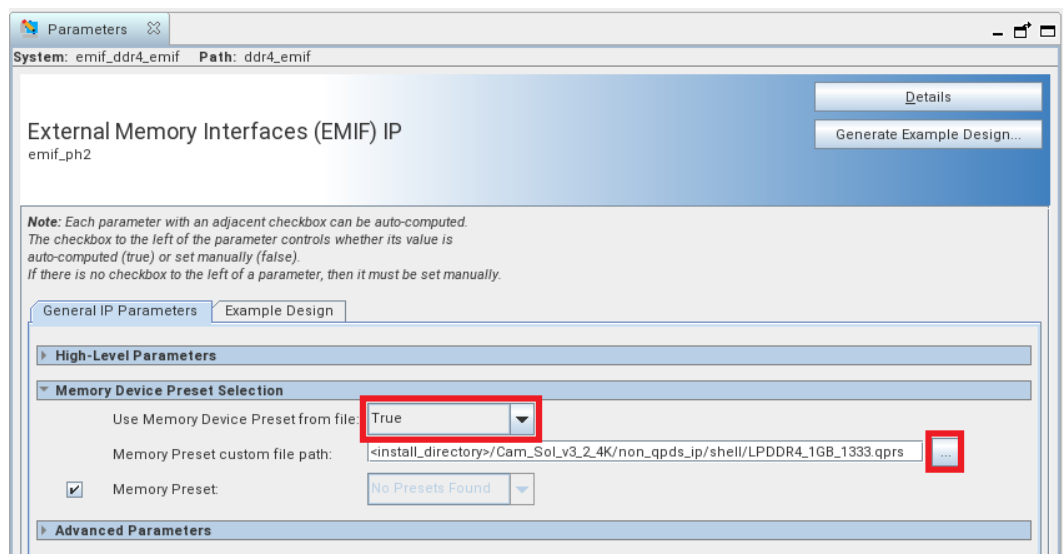
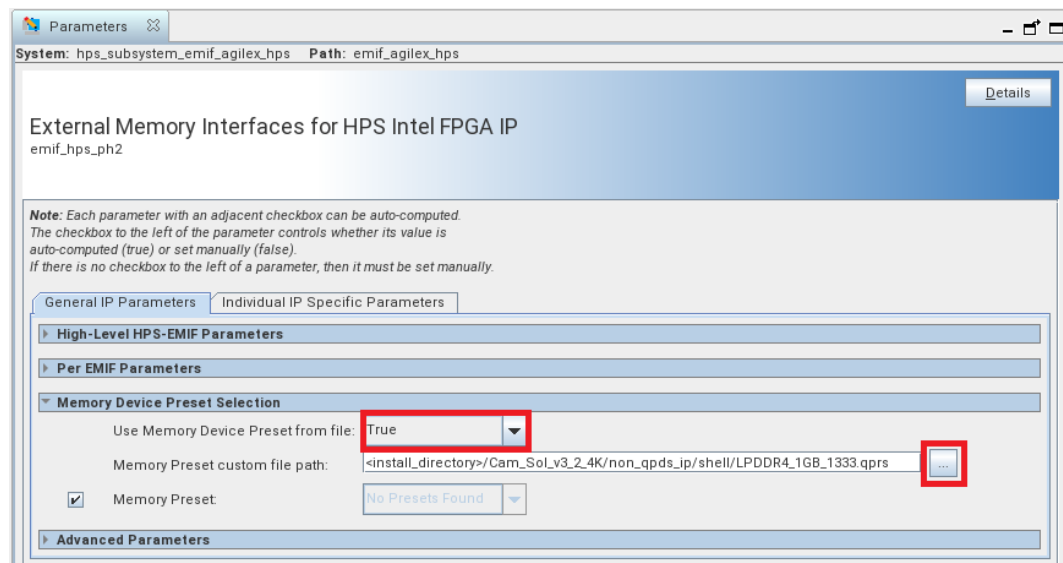


Figure 4-3. Adding calibration file to the HPS EMIF



4.3.2.3 FPGA Development – Compiling the Project

4.3.2.3.1 HPS First Programming File

You continue with the compilation of the example design in the Quartus GUI and generate a new .sof programming file. You may also explore, inspect, and modify the full system inside Platform Designer by opening top level system file agilex5_vvpisp_qsys.qsys from the project navigation pane.

Building with HPS First configuration allows the HPS to boot and program the FPGA from the SD Card.

You reference the new `.sof` programming file for U-Boot HPS in command line to generate the final programming file.

On Windows:

```
> cd <extracted_tarball_directory>\quartus\output_files  
  
> <quartus_install_path>\quartus\bin64\quartus_pfg.exe -c agilex5_vvpisp.sof -o  
hps_path=./u-boot-spl-dtb.hex -o hps=ON -o flash_loader=A5ED065BB32AE4SR0 -o  
mode=ASX4 -o device=MT25QU02G vvp-isp-agilex5_axe5_eagle.jic
```

On Linux:

```
cd <extracted_tarball_directory>/quartus/output_files  
  
$ <quartus_install_path>\quartus/bin/quartus/quartus_pfg -c agilex5_vvpisp.sof -o  
hps_path=./u-boot-spl-dtb.hex -o hps=ON -o flash_loader=A5ED065BB32AE4SR0 -o  
mode=ASX4 -o device=MT25QU02G vvp-isp-agilex5_axe5_eagle.jic
```

The jic file `vvp-isp-agilex5_axe5_eagle.hps.jic` is programmed one time only. The subsequent rbf file `vvp-isp-agilex5_axe5_eagle.core.rbf` is used as the FPGA image.

4.3.3 Requirements for Software Development

The example design software is compiled with a Yocto project software stack tailored for the Altera Agilex 5, which requires a Linux development system that meets specific system requirements.

The software comes with a Docker recipe for instantiating the build environment. The only pre-requisite is that Docker is installed, and the daemon is operational on your local system. You can find instructions for installing Docker here: <https://docs.docker.com/engine/install/>

4.3.3.1 Software Development – Getting Started

- From the root source directory (yocto), run the following command build a complete SD card image:

```
./yocto_build_script.sh
```

4.4 Development Kit Setup

IMPORTANT! Handle ESD-sensitive equipment (boards, SD Cards, Camera, etc) only when adequately grounded and at an ESD-safe workstation.

Figure 4-4. Cable Connections

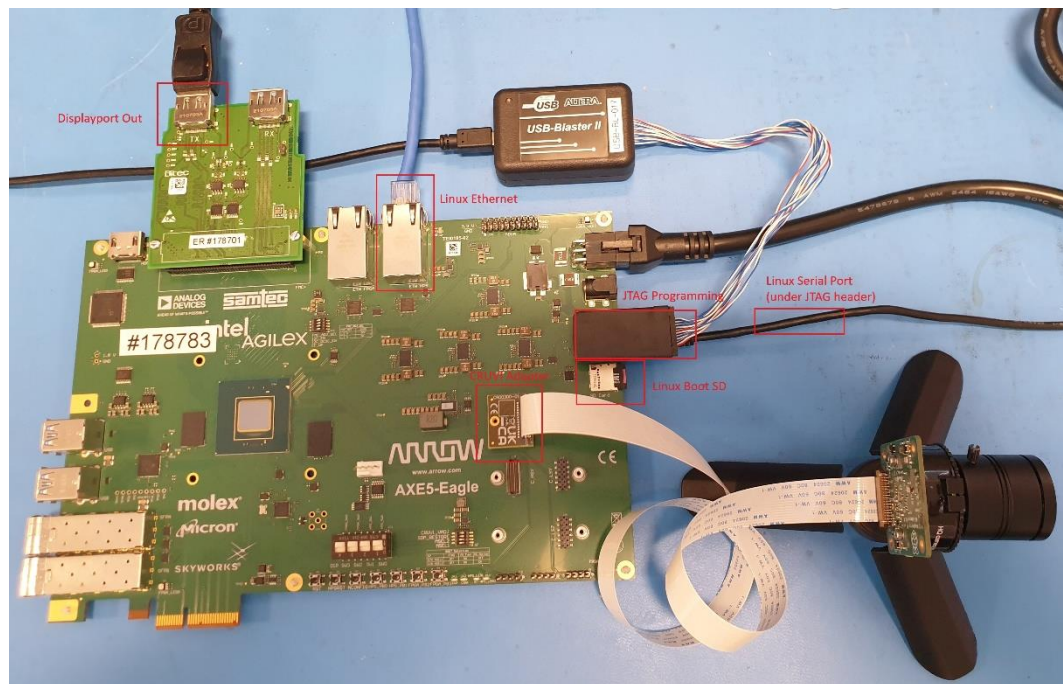
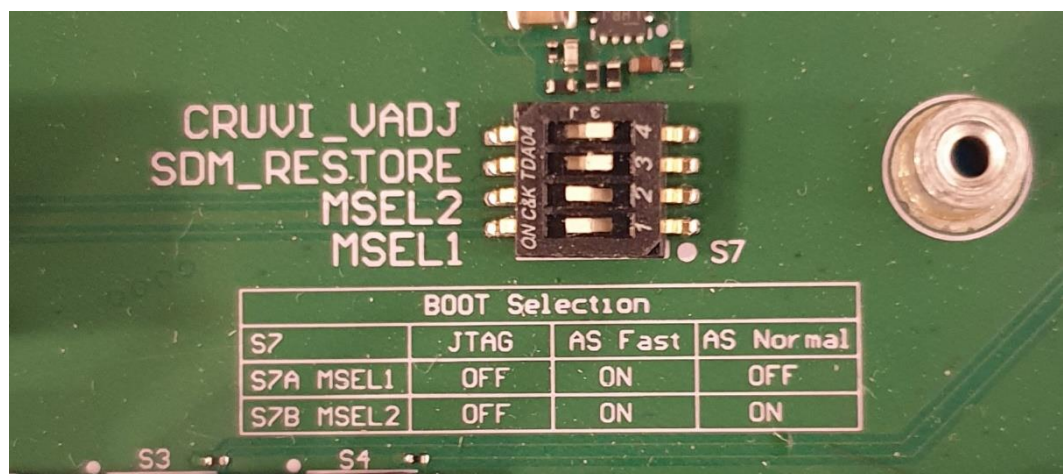


Figure 4-5. Switch Configurations

Ensure the switch configurations are as shown:



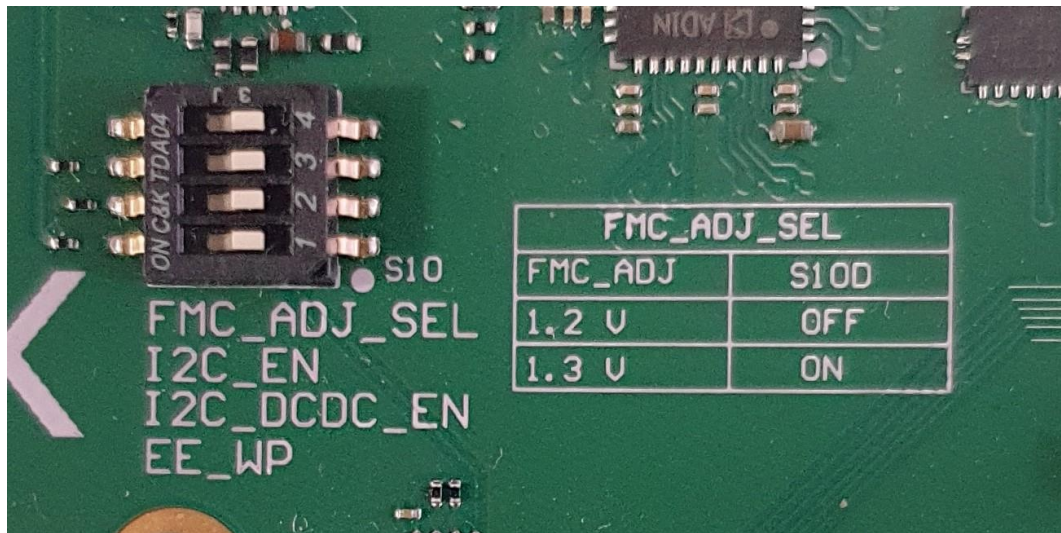


Figure 4-6. Flashing the MicroSD Card

Requirements:

- MicroSD Card with at least 8GB capacity and rated for U3 speed
- MicroSD Card Reader/Writer
- `.img` file stored on your local computer
- SD Card Formatter Utility (from SD Association)
- SD Card Imager Utility (Win32 Disk Imager, for example)

Format the MicroSD Card using an SD Card Formatter Utility. Then, using the MicroSD Card Writer and Imager Utility, flash the `.img` file to the MicroSD Card. Install the MicroSD Card into the Development Kit. Note that the `.img` file is specifically for the HPS-first workflow.

4.5 Powering up the Development Kit

The following steps describe how to set up the Development Kit to run the demo design:

- Follow the previous sections to setup the Development Kit and to flash and install the MicroSD Card
 - Make sure the video sink is ready.
- Start running a terminal emulator over the USB UART interface (115200 baud, 8 bits, no parity, 1-bit stop, and no flow control).

- Power up the board by plugging in the power adapter.

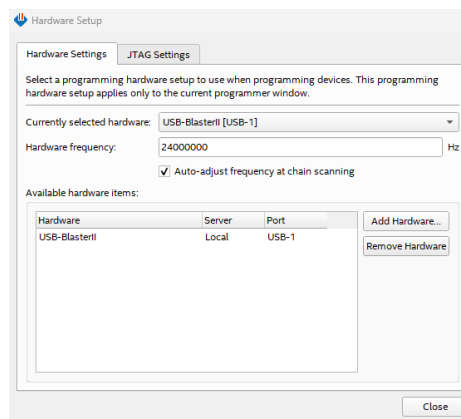
4.5.1 FPGA Workflow

A single workflow is supported for working with the Development Kit: HPS-first, where the board programs itself from a `.rbf` file located on the SD card.

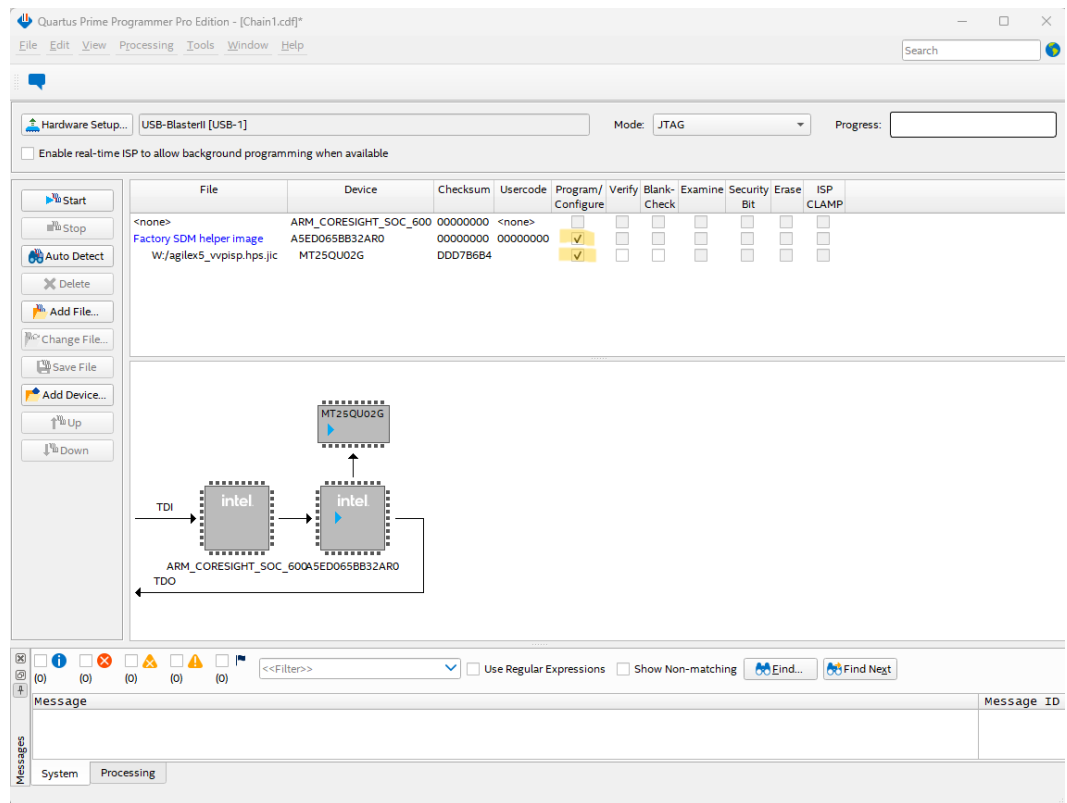
4.5.1.1 HPS-First Setup

- If these steps have been successfully completed previously, they do not need to be repeated.
- For the initial HPS-first setup, program the given `.jic` to the Development Kit.
 - Open Quartus Programmer Tool
 - If you do not see "USB-BlasterII [USB-1]" in the box next to "Hardware Setup," you must click the Hardware Setup button.
 - Ensure the USB-Blaster is selected under the "Currently selected hardware" dropdown box. Then press close.

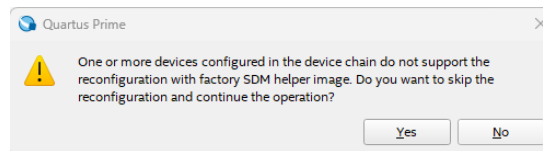
Figure 4-7. JTAG Settings



- On the main programmer screen, click "Auto Detect," the device chain will update with two devices: A5EC065BB32AR0, and ARM_CORESIGHT_SOC_600.
- Right-click A5EC065BB32AR0, and select "Change File" to select the `.jic`
- The UI will update as follows: (Hashes may not be the same)



- Ensure the highlighted boxes "Program/Configure" are checked with a tick. Then click "Start". The warning window will appear; click "Yes" and the programming will proceed.



- The FPGA will be programmed with the JIC.
 - o Power off the board.
 - o Insert the SD card flashed with HPS-first .img file, and power on.

4.6 Booting the Application

- With the board set up, the HPS Arm processor will start up and boot into an embedded Linux OS. Observe the terminal output and wait for the startup sequence to complete to the Linux prompt.

- If you see no output, ensure you have selected the correct serial port.
- At the prompt, type "root," then press enter.
- Next, type "ifconfig" and press enter. This will obtain the device IP address (inet address of eth0), which you will use to connect to the web server running in the FPGA.
- To start the application, type "./VvpIspDemo", then press 'enter'.
- Enter the IP address in a web browser on a device on the same local network to run the application GUI.

5.0 Graphical User Interface

The 4K Camera Demo Design software application has a web-server GUI that is used to demonstrate the features of the design. The web address of the GUI is printed to the terminal screen when the application starts running.

From a PC connected to the same LAN as the development board, open a web browser and enter the IP address of the board into the address bar.

An alternative way to find the IP address is to log in as root using a blank password and then use the `ifconfig` command from the terminal. Find the `inet addr` for the `eth0` interface. Using the `ipv6`-address to connect to the web server is also possible.

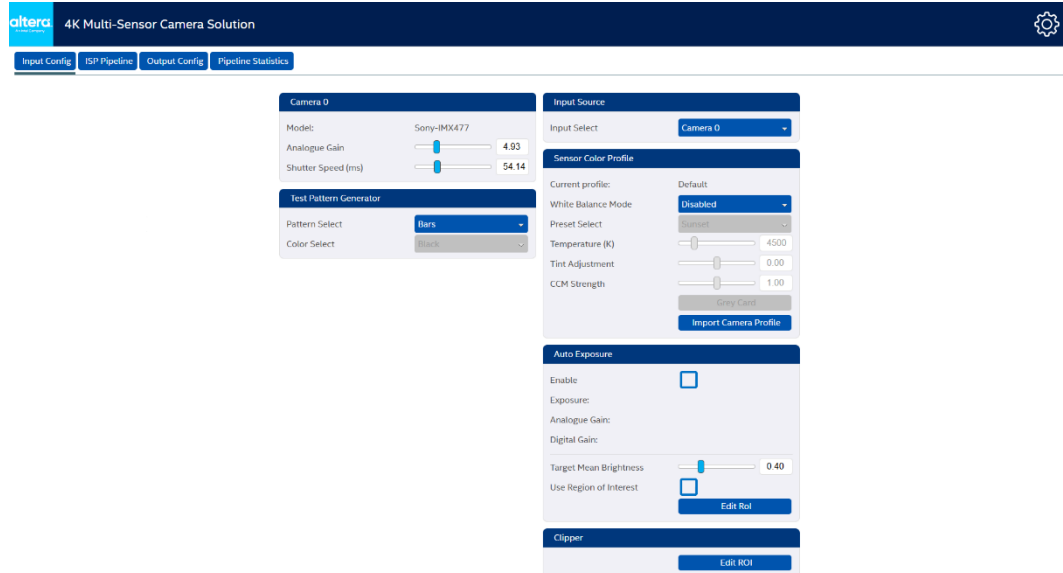
Figure 5-1. ifconfig Example Output

```
root@agilex5devkit:~#  
root@agilex5devkit:~#  
root@agilex5devkit:~#  
root@agilex5devkit:~# ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 10.102.200.129 netmask 255.255.254.0 broadcast 10.102.201.255  
    inet6 fe80::88e8:7fff:fe85:6acd prefixlen 64 scopeid 0x20<link>  
    ether 8a:ce:00:7f:85:6a:cd txqueuelen 1000 <Ethernet>  
    RX packets 13579 bytes 882681 (861.9 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 22567 bytes 11855083 (11.3 MiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
    device interrupt 23  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 <Local Loopback>  
    RX packets 168 bytes 15172 (14.8 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 168 bytes 15172 (14.8 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
root@agilex5devkit:~#
```

The UI is optimized for displaying the full screen on a 1920x1080 screen. Press F11 on your browser to go full screen.

5.1 Input Config Tab

Figure 5-2. Input Config Tab

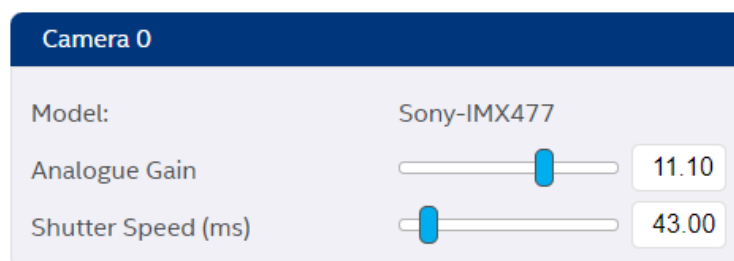


The Input Config Tab controls two separate parts of the application:

- The Input Stage corresponds to selecting which input to use and the parameters for those input sources. It also includes the Clipper, located immediately before the primary ISP cores.
- The Control Loops, corresponding to the Automatic White Balance and Auto Exposure systems.

5.1.1 Sensor Control

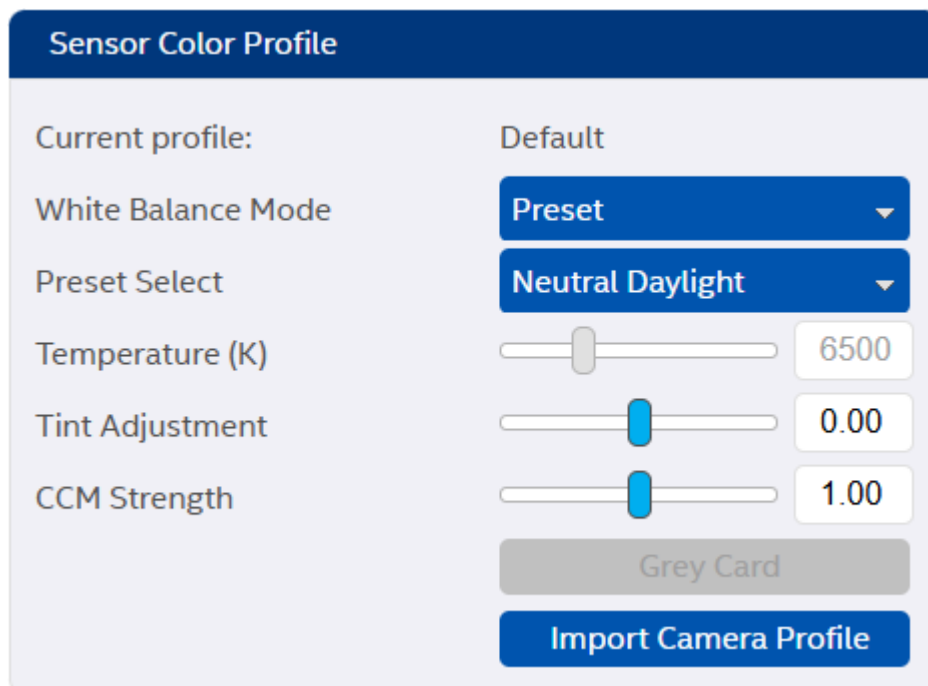
Figure 5-3. Sensor Control



The Sony IMX77, sensor control UI provides a few simple controls for the Raspberry Pi camera used in the demo. The Analogue Gain slider programs logarithmic table entries to the camera to provide linear gain scaling, allowing the image to be brightened or dimmed. The Shutter Speed slider controls the shutter speed, which is directly related to the output framerate. It can be adjusted to minimize any flickering from artificial light sources.

5.1.2 Sensor Color Profile

Figure 5-4. Sensor Color Profile



<NEW CONTROLS REQUIRING DESCRIPTION>

The Sensor Color Profile UI allows control of the Automatic White Balance system built into the demo. Its profiles are premade and created using the in-app Calibrator Tool running on live video processed by the ISP. However, the Calibrator Tool requires expert use and is hidden from general use.

There are four modes of operation:

- Disabled. In Disabled mode, the White Balance system is turned off. White Balance Correction and Black Level Correction parameters are unlocked so the user can access them. Post-process color corrections utilizing the Color Correction Matrix are turned off automatically.
- Manual. In this mode, the user specifies the current temperature of the scene the camera is pointing at by using the Temperature Slider. **The temperature must be set correctly;** otherwise, incorrect coefficients

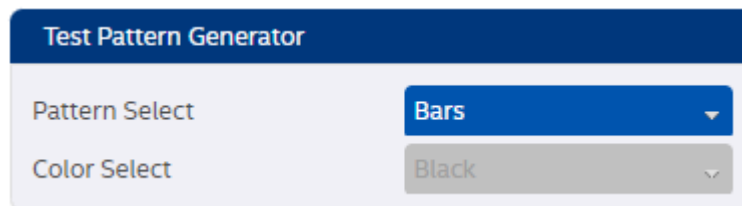
will be used, causing the image to look incorrect. As a guide, temperatures between 5000-6500K usually provide a safe option.

- **Preset.** Preset mode operates similarly to Manual mode; however, the Temperature Slider is blocked from user control. Instead, a Preset Select dropdown box is used to specify the temperature from a set of well-known, industry-standard temperature values.
- **Automatic.** Automatic mode turns off all user input and instead uses the White Balance Statistics IP core to attempt to guess the scene's color temperature. It then decides which coefficients to apply and automatically color-corrects the scene.

Camera Profiles can be used for any setting, including artistic effects. The Import Camera Profile button can load a profile into the application.

5.1.3 Test Pattern Generator

Figure 5-5. Test Pattern Generator

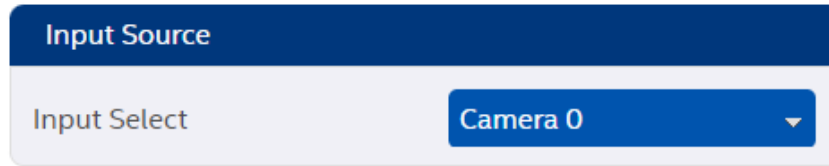


The Test Pattern Generator UI allows you to control the Input Test Pattern Generator. It operates in the RGB domain and has four modes of operation:

- **Color Bars.** SMPTE standard color bars.
- **Solid Color.** Use the Color Select dropdown box to set the color from the list (including SMPTE standard color bar colors and mid-level grey).
- **Zone Plate.** Arbitrary-sized and positioned concentric rings that get wider and narrower as they get further from their origin. Used to introduce frequency into the image to generate interference detection.
- **Rainbow.** Uses the Solid Color mode but cycles through the Color Select list.

5.1.4 Input Select

Figure 5-6. Input Select



The Input Select UI controls which input is selected:

- Input TPG corresponds to the TPG configured in the Test Pattern Generator UI. The input TPG output is passed through a Remosaic IP Core to convert it to Bayer format that is consistent with the camera input. It can be used to verify the system without the Camera input.
- Output TPG corresponds to a TPG configured in Color Bars mode that feeds the output DisplayPort IP Core. It can be used to verify the video sink device.
- Camera corresponds to the Raspberry Pi Sony IMX477 sensor output connected to the MIPI cable port.

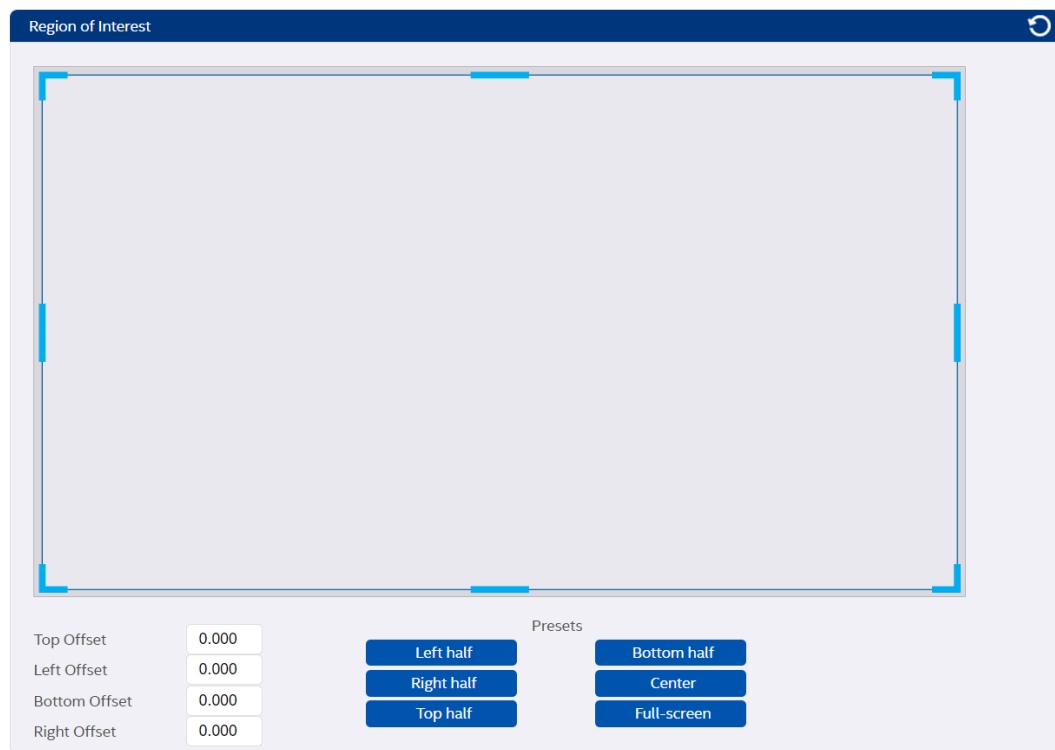
5.1.5 Clipper

Figure 5-7. Clipper



The Clipper UI serves as a simple "Edit ROI" button to open a much larger UI.

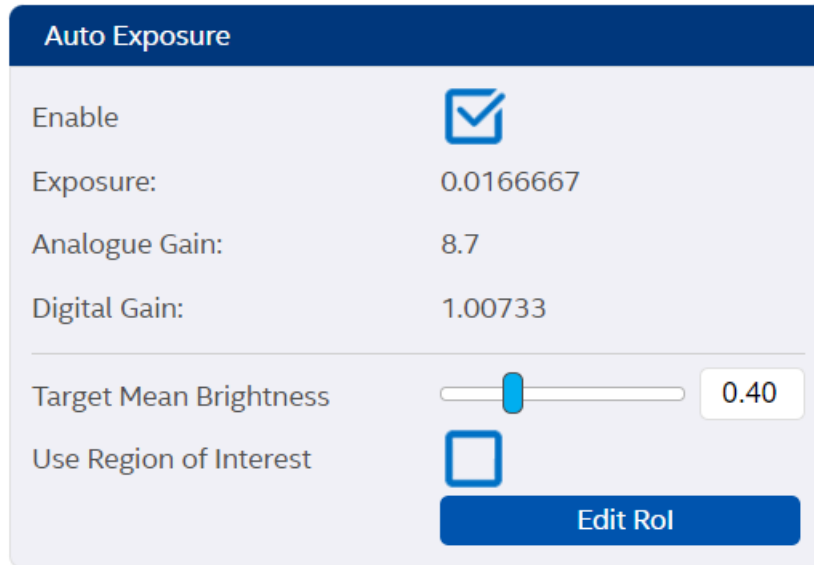
Figure 5-8. Clipper Region of Interest



The ROI selector allows the user to select a region of the screen that will be cropped and stretched to fill the display. This can be done by using a mouse to stretch and drag the window or manually entering values. Some basic presets are also available to set common ROI values quickly.

5.1.6 Autoexposure Controls

Figure 5-9. Autoexposure Controls

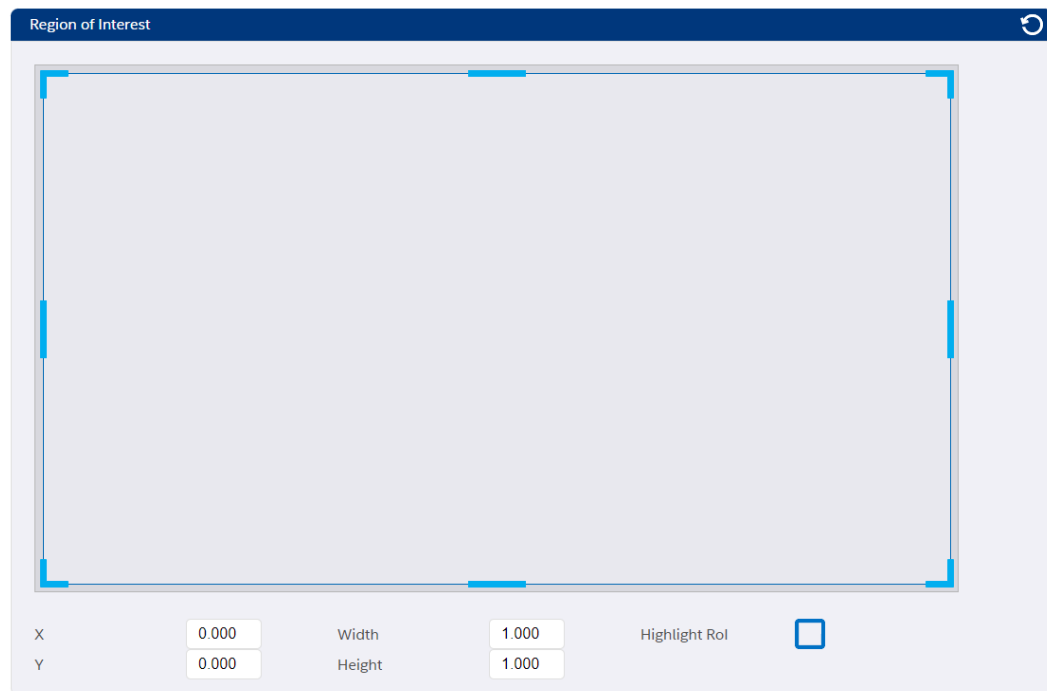


The image shows a software control panel titled "Auto Exposure". It contains several settings: "Enable" is checked with a blue checkmark icon; "Exposure:" is set to 0.016667; "Analogue Gain:" is set to 8.7; "Digital Gain:" is set to 1.00733; "Target Mean Brightness" is controlled by a slider set to 0.40; and "Use Region of Interest" is unchecked with an empty square icon. Below the "Use Region of Interest" checkbox is a blue button labeled "Edit Rol".

| Auto Exposure | |
|--------------------------|-------------------------------------|
| Enable | <input checked="" type="checkbox"/> |
| Exposure: | 0.016667 |
| Analogue Gain: | 8.7 |
| Digital Gain: | 1.00733 |
| Target Mean Brightness | <input type="range" value="0.40"/> |
| Use Region of Interest | <input type="checkbox"/> |
| Edit Rol | |

Only available when the Camera Input is selected, the Autoexposure UI controls an autoexposure algorithm that uses data from the Histogram Statistics IP to control sensor shutter speed and gain. When using the ROI, the algorithm will prioritize the region specified using the region editor (shown below) when calculating an exposure level.

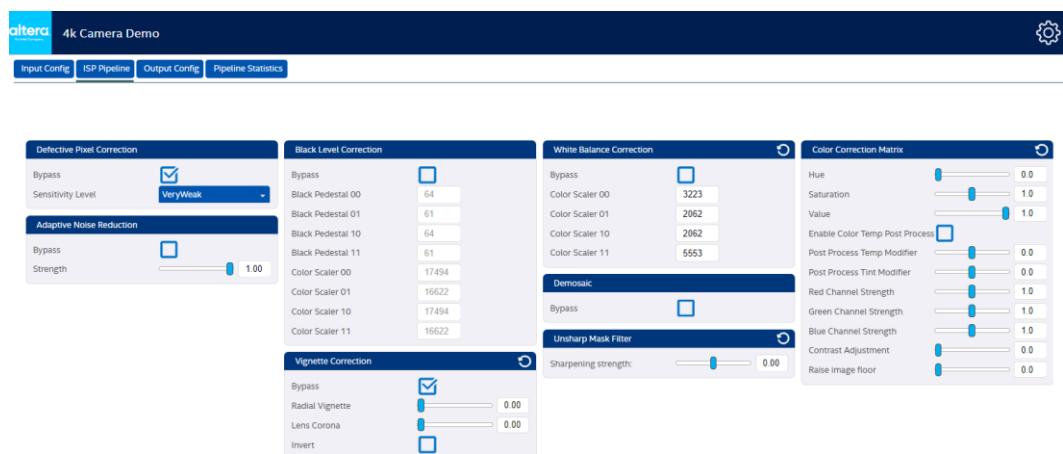
Figure 5-10. Autoexposure Region of Interest Editor



The ROI editor allows the user to select a region of the screen. This can be done by using a mouse to stretch and drag the window or manually entering values.

5.2 ISP Pipeline Tab

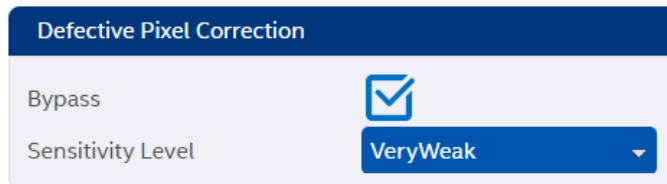
Figure 5-11. ISP Pipeline Tab



The ISP Pipeline Tab contains the controls corresponding to the primary ISP's core functionality, except for the 1D LUT IP core, which lives in the Output Config Tab.

5.2.1 Defective Pixel Correction

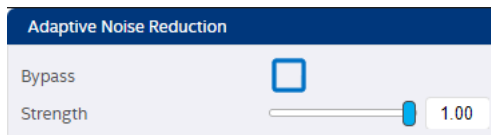
Figure 5-12. Defective Pixel Correction



The Defective Pixel Correction UI is used to select Bypass mode or the strength of the Sensitivity Level. The stronger the level, the softer the image will look. "VeryWeak" is the default sensitivity level and works well at correcting defective pixels.

5.2.2 Adaptive Noise Reduction

Figure 5-13. Adaptive Noise Reduction



The Adaptive Noise Reduction UI is used for selecting Bypass mode or the strength of the denoising algorithm when the Automatic White Balance mode is not Disabled.

5.2.3 Black Level Correction

Figure 5-14. Black Level Correction

| Black Level Correction | |
|------------------------|--------------------------|
| Bypass | <input type="checkbox"/> |
| Black Pedestal 00 | 63 |
| Black Pedestal 01 | 61 |
| Black Pedestal 10 | 63 |
| Black Pedestal 11 | 61 |
| Color Scaler 00 | 17203 |
| Color Scaler 01 | 16622 |
| Color Scaler 10 | 17203 |
| Color Scaler 11 | 16622 |

Black Level Correction is mostly locked out to the user as it uses pre-calculated coefficients during Calibration. The user can only access the Bypass option in a typical operation with the pre-calculated coefficient values displayed. The Bypass button is disabled when Automatic White Balance mode is enabled.

5.2.4 Vignette Correction

Figure 5-15. Vignette Correction

| Vignette Correction | |
|---------------------|-------------------------------------|
| Bypass | <input checked="" type="checkbox"/> |
| Radial Vignette | <input type="range"/> 0.00 |
| Lens Corona | <input type="range"/> 0.00 |
| Invert | <input type="checkbox"/> |

Vignette Correction is part of the Calibration process; therefore, the coefficients are pre-calculated and loaded for the sensor. The Vignette Correction UI itself serves as an example of what the Vignette Correction IP core is doing. When out of Bypass, the Radial Vignette slider simulates radial vignetting as commonly seen on fisheye lenses. Likewise, the Lens Corona simulates the effects of a strong light source on an imperfect lens, showing as a bright spot in the center. Note that these options function by brightening the image, as darkening will cause an imbalance in color ratios and cause corrupted colors in

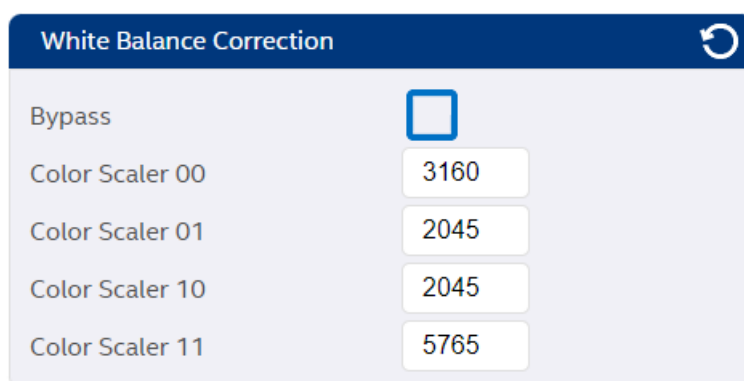
saturated regions (for example, around ceiling lights). It is recommended that Autoexposure is enabled to account for this brightened image automatically.

The Invert button is an artistic effect that shows the Vignette Correction coefficients that would be applied to the image to correct the effect.

The settings can be reset to a fully disabled state by clicking the Reset button on the top right of the UI header bar.

5.2.5 White Balance Correction

Figure 5-16. White Balance Correction



The White Balance Correction UI is a panel with a dark blue header bar containing the title "White Balance Correction" and a circular reset icon. Below the header, there is a "Bypass" checkbox which is currently unchecked. Underneath, four color scalers are listed, each with a corresponding numerical value in a light blue box: "Color Scaler 00" with value 3160, "Color Scaler 01" with value 2045, "Color Scaler 10" with value 2045, and "Color Scaler 11" with value 5765.

| Control | Value |
|-----------------|--------------------------|
| Bypass | <input type="checkbox"/> |
| Color Scaler 00 | 3160 |
| Color Scaler 01 | 2045 |
| Color Scaler 10 | 2045 |
| Color Scaler 11 | 5765 |

The White Balance Correction UI corresponds to the IP Core's four multipliers that control the scaling applied to the input Bayer CFA. The Bayer pattern of the camera in the Demo design is BGGR. Hence, Scaler 00 corresponds to blue strength, 01 and 10 correspond to green, and 11 correspond to red.

Sensible defaults are stored and can be quickly set using the Reset button on the top right of the UI header bar.

The White Balance Correction UI is disabled when Automatic White Balance mode is enabled.

5.2.6 Demosaic

Figure 5-17. Demosaic



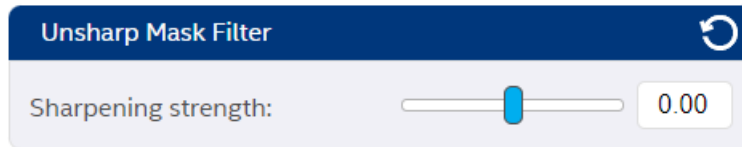
The Demosaic UI is a simple panel with a dark blue header bar containing the title "Demosaic". Below the header, there is a "Bypass" checkbox which is currently unchecked.

| Control | Value |
|---------|--------------------------|
| Bypass | <input type="checkbox"/> |

The Demosaic IP Core has virtually no user input. However, Bypass can be used to see the raw Bayer pattern as a monochrome image.

5.2.7 Unsharp Mask Filter

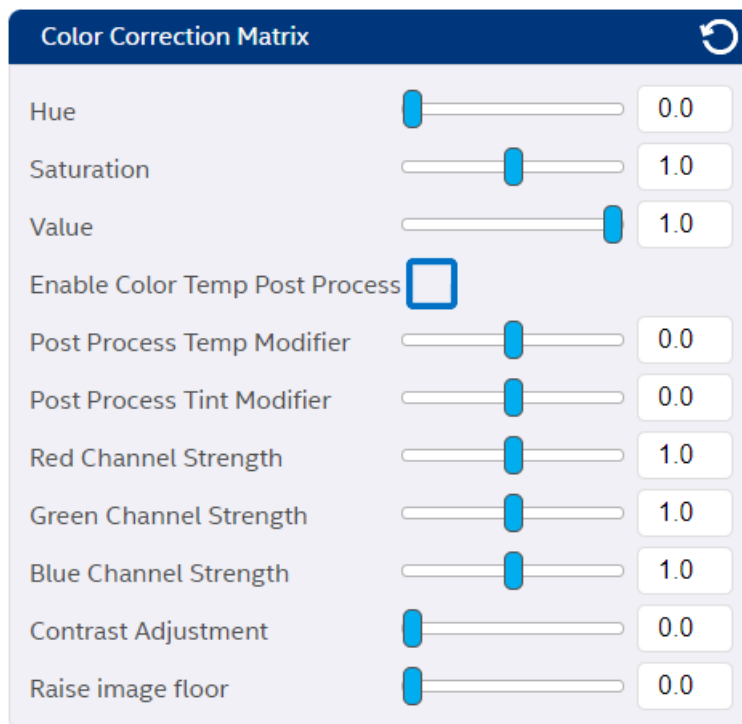
Figure 5-18. Unsharp Mask Filter



The Unsharp Mask Filter UI controls the strength of sharpening applied to the image. Positive strengths will sharpen the image, while negative strengths will soften the image. Clicking the Reset button in the top right of the UI header will set strength back to 0.

5.2.8 Color Correction Matrix

Figure 5-19. Color Correction Matrix



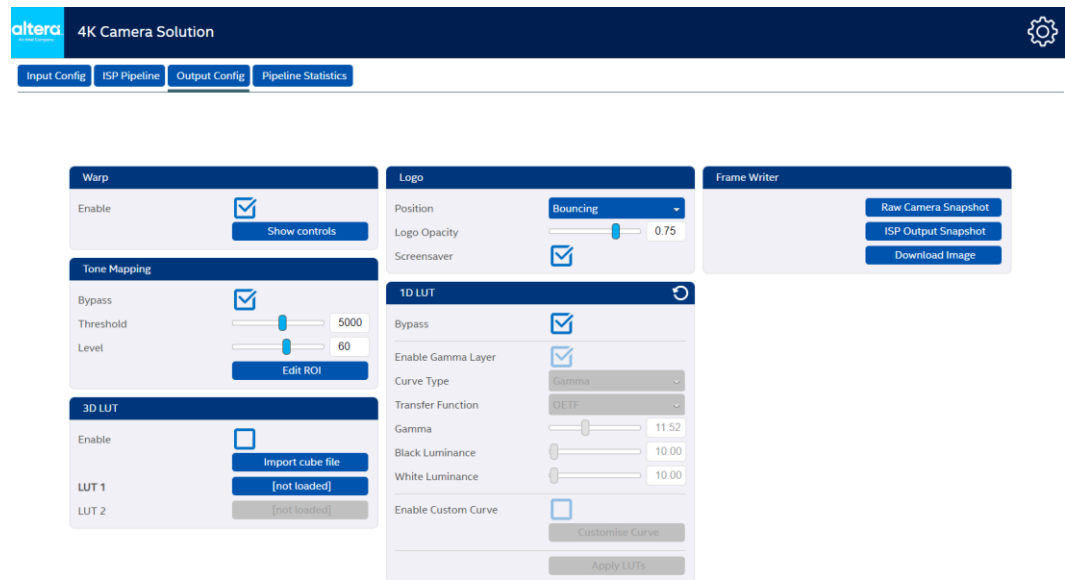
The Color Correction Matrix UI is a novel use case of the Color Space Converter IP. It allows for modification of Hue, Saturation, and Value (Brightness).

It also has additional post-process effects, such as post-process color temperature filtering, which can apply artistic effects to images. Individual channels can be scaled, and simple contrast algorithms can be applied to increase the image's dynamic range.

All sliders can be reset to their default values by clicking the Reset button in the top right of the UI header.

5.3 Output Config Tab

Figure 5-20. Output Config Tab

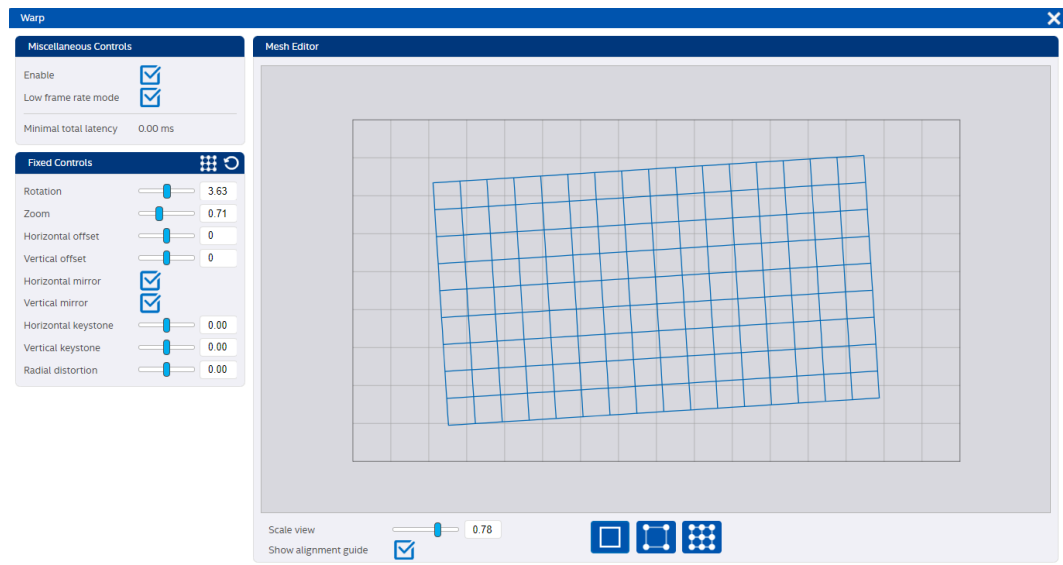


The Output Config Tab contains UI controls for IP Cores typically located after the core ISP pipeline up to the DisplayPort output.

5.3.1 Warp

The Warp UI is a simple interface where the Warp IP can be enabled or disabled using the checkbox. Click the "Show controls" button to open the full warp controls.

Figure 5-21. Warp Full Controls



The dialog box that pops up has 3 blue buttons at the center/bottom, which switch between the three warp editing modes. If any transformations go outside the warp IP's capability, the mesh will change color from blue to red, indicating an invalid configuration. When the parameterization ranges become valid again, the mesh color will change to blue.

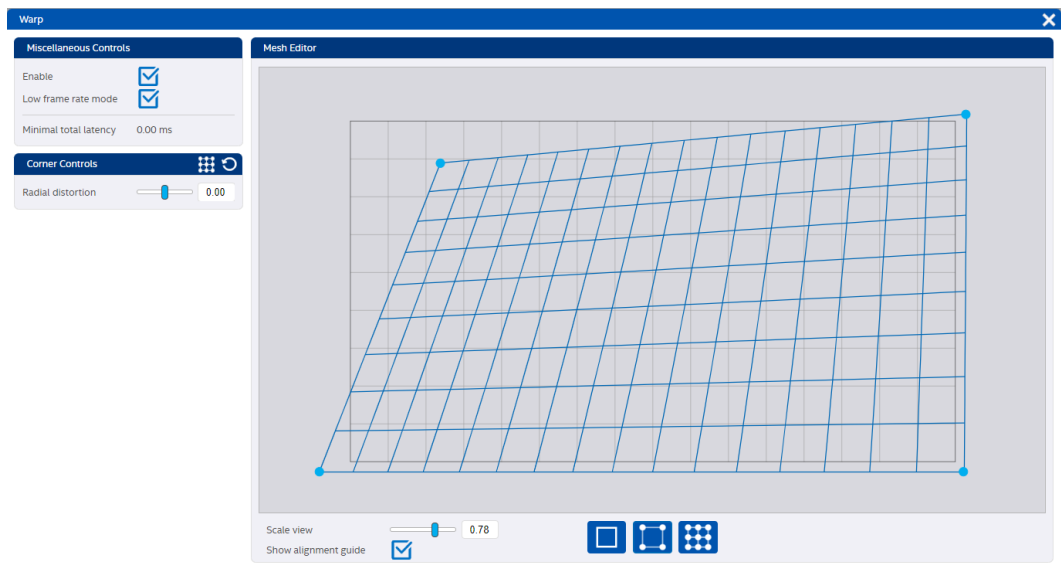
The "Fixed Controls" panel has high-level controls for specifying a warp based on various mathematical transforms.

The "Scale view" slider will let you reduce the target window size in the Mesh Editor, which makes it possible to view transforms that could extend outside the target windows.

The "Show alignment guide" lets you turn off the 16x9 grid of squares.

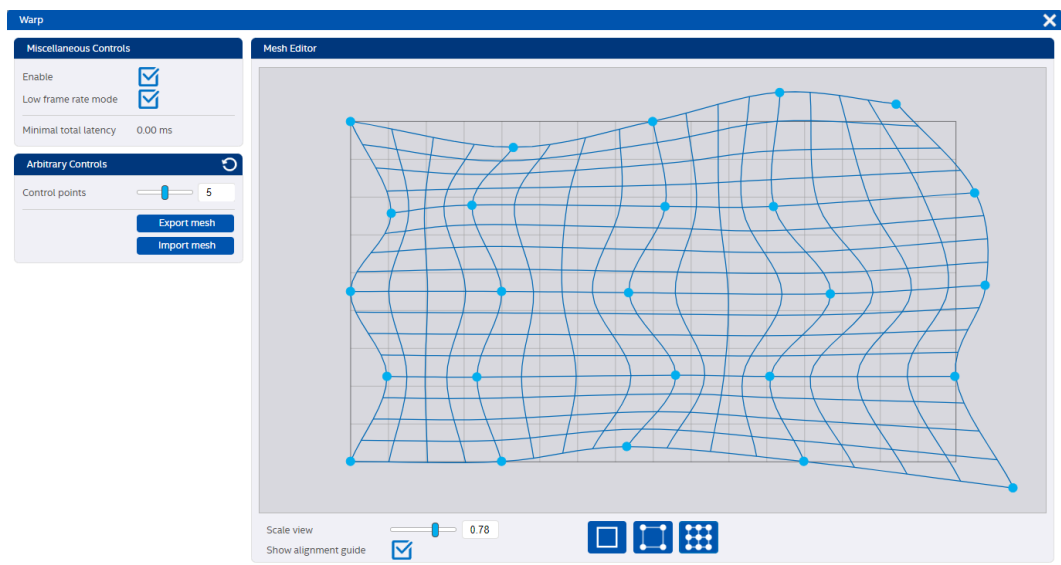
In the header of the "Fixed Controls" panel, there are two buttons:

- "Convert to arbitrary mesh" converts the current transformation to an arbitrary mesh for manual adjustment.
- "Reset" – restored the controls to their default settings.

Figure 5-22. Warp Corner Controls

In "Corner Controls" mode, each of the 4 corners of the transformation rectangle can be adjusted by dragging the blue circles.

There is also a control to apply additional radial distortion.

Figure 5-23. Warp Arbitrary Controls

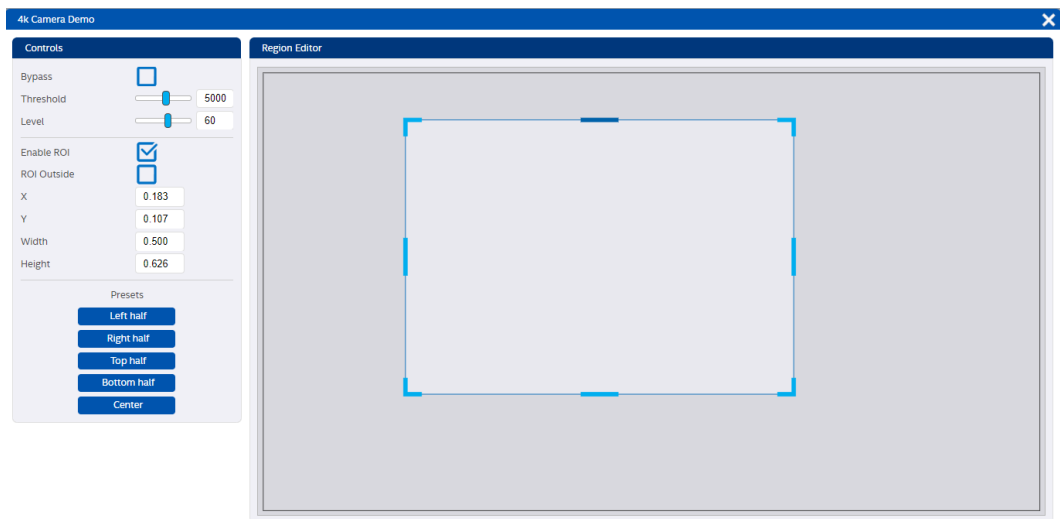
The mesh editor in "Arbitrary Controls" mode lets you manually adjust the mesh by dragging any control points. The number of control points can be changed with the slider.

The "Export mesh" and "Import mesh" buttons allow you to export mesh files to your PC and import them later or onto another device.

5.3.2 Tone Mapping Operator

The TMO UI has controls to bypass the tone mapping and sliders to adjust the threshold and level. The threshold slider controls the tone mapping strength. The level slider controls the alpha blending percentage between fully processed TMO output and its original input. The tone mapping feature can use a region of interest (ROI), which can be adjusted from a pop-up panel. The ROI panel is accessed by clicking the "Edit ROI" button on the main TMO panel.

Figure 5-24. TMO Controls

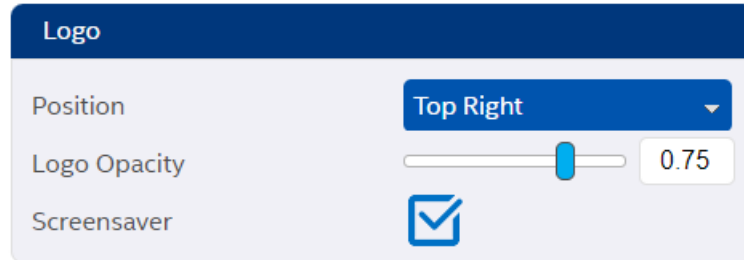


5.3.3 3D LUT

The 3D-LUT UI has controls to bypass the IP. Cube files on the local computer can be uploaded to the IP using the "import cube file" button. The 3D-LUT IP is double-buffered and can hold two LUTs at the same time. The last LUT file imported to the IP overwrites the non-active buffer and becomes active when loaded. The user can switch active buffers after loading both. The LUT contents are cleared when the application is restarted, or the board is power cycled.

5.3.4 Logo

Figure 5-25. Logo



This UI controls the position and opacity of the Altera logo on the screen. The Logo is also a Screensaver function, which can be turned on/off here. The Logo can be positioned on each corner or in a novelty mode where it bounces across the screen. The opacity slider determines how transparent the Logo is, where 1 is fully opaque, and 0 is fully transparent.

When Screensaver is checked, after a few minutes of UI inactivity, the screen output will drop to a black background, and the Logo will bounce across the screen. This is to prevent burn-in on screens that may be sensitive, such as OLED. The Screensaver will persist until a UI element is interacted with.

5.3.5 1D LUT

The 1D LUT IP allows the user to apply a transfer function to the output video image to match the video sink. The IP can be bypassed, or a LUT-based curve can be applied from a dropdown list of industry-standard transfer functions. The default transfer function is an OETF Gamma Curve, which performs pre-monitor gamma correction.

However, besides the preset curves, the IP can accept any arbitrary curve generated through the Custom Curve Editor. Furthermore, the IP supports multiple different layers of curves, allowing for nearly limitless sets of custom, artistic gamma effects. These arbitrary curves are generated through the Custom Curve Editor. The current configuration is set up for two layers: One for Gamma, and one for a Custom Curves.

Figure 5-26. 1D LUT

The screenshot shows the '1D LUT' configuration window. It has a title bar with a refresh icon. The settings are as follows:

| Control | Value / State |
|---------------------|-------------------------------------|
| Bypass | <input checked="" type="checkbox"/> |
| Enable Gamma Layer | <input checked="" type="checkbox"/> |
| Curve Type | Gamma |
| Transfer Function | OETF |
| Gamma | 11.52 |
| Black Luminance | 10.00 |
| White Luminance | 10.00 |
| Enable Custom Curve | <input type="checkbox"/> |
| Customise Curve | Button (disabled) |
| Apply LUTs | Button (disabled) |

The 1D LUT UI has four sections in its controls.

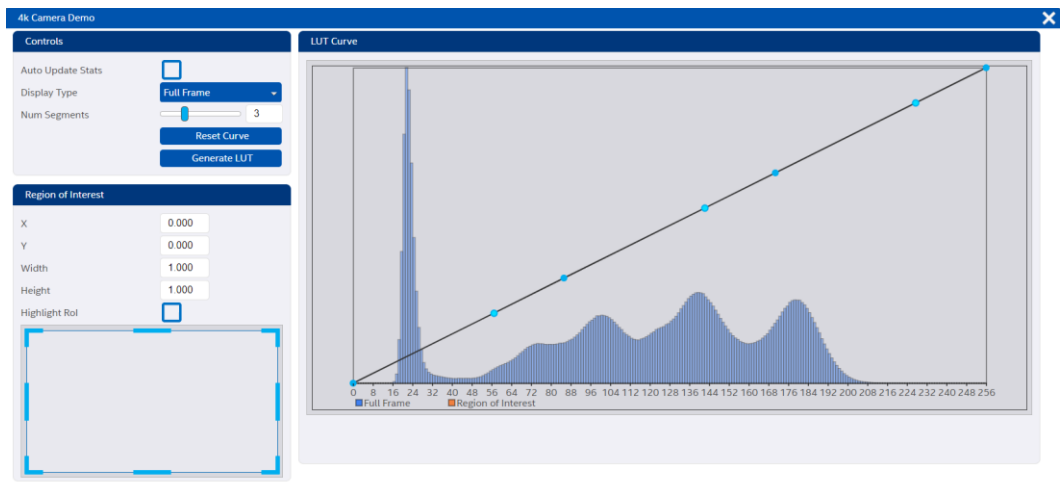
The first is the Bypass checkbox, which can entirely disable the 1D LUT IP. If Bypass is toggled on, then the rest of the UI is greyed out.

The second set of controls is for configuring the Gamma settings. To enable the controls, "Enable Gamma Layer" must be toggled on. The options allow for choosing different Transfer Functions and Curve Types, and the ability to pass parameters for the different LUT generations.

The third set controls the Custom Curve, if the "Enable Custom Curve" button is enabled, then the "Customise Curve" button becomes active, allowing for modifying of a custom curve using the Custom Curve Editor.

The final control is the "Apply LUTs" button. When all changes are finalized, clicking the button will collapse the 2 LUT layers into a single set of coefficients, which are then downloaded into the 1D LUT IP. This button is greyed out if neither Gamma nor Custom Curve layers are active.

Figure 5-27. Custom Curve Editor



The Custom Curve Editor contains a miniaturized version of the Histogram Statistics UI. Reference the Histogram Statistics UI description (within the Pipeline Statistics Tab section) to understand what the Display Type and ROI controls do.

On the LUT curve screen, the LUT is built by a spline specified by the dots on the line. The dark blue nodes correspond to direct spline points, whereas the light blue nodes apply weight to the spline curve between points. The number of segments can be controlled using the Num Segments Slider - which adds more nodes to the line for manipulation. Once editing is completed, click the "Generate LUT" button to save the values. The Custom Curve Editor box can be closed by clicking the X at the top right of the dialogue box.

5.3.6 Frame Writer

Figure 5-28. Frame Writer



The Frame Writer UI allows you to take screenshots of the raw sensor input image or the final processed output image. The input image post-Clipper IP is a 12-bit Bayer image, while the output image post 1D-LUT IP is a 10-bit RGB image. The IP captures the TIFF image in a 16-bit MSB-aligned format. By

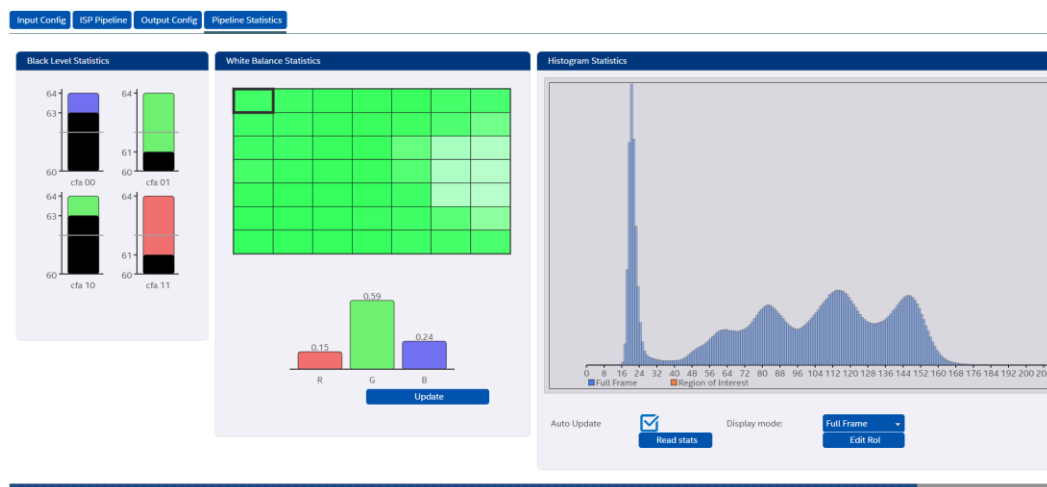
turning on and off various IPs in the video pipeline, the output image can be used to capture partially processed images.

The process for taking screenshots is:

- Click either the "Raw Camera Snapshot" button for the input image capture or the "ISP Output Snapshot" button for the output image. The output will flicker while the image is captured by the external DDR. Flickering stops once the capturing process finishes. The UI may also go dark and be unresponsive while the HPS converts the capture to TIFF. It will recover once the process is complete.
- Once the output and UI have recovered, click the "Download Image" button to download the TIFF image to your local computer.

5.4 Pipeline Statistics Tab

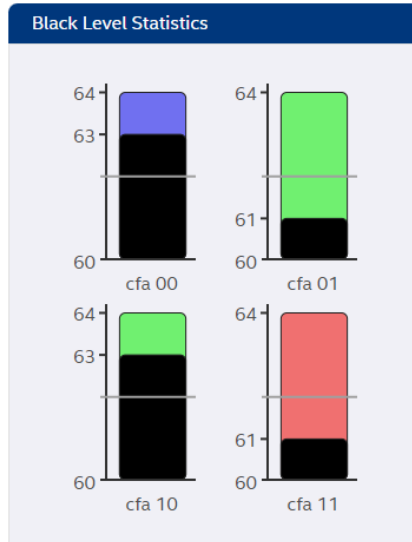
Figure 5-29. Pipeline Statistics Tab



The Pipeline Statistics Tab contains UI controls to help visualize the Statistics IP Cores within the ISP pipeline. The Controls available are the Black Level Statistics, White Balance Statistics, and Histogram Statistics.

5.4.1 Black Level Statistics

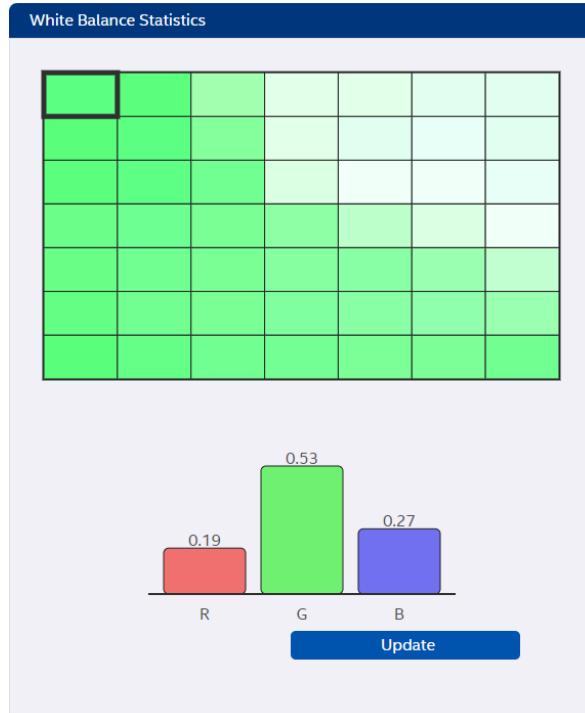
Figure 5-30. Black level Statistics



The Black Level Statistics UI provides only status information from the Automatic White Balance algorithm. It shows the current profiled black level on each of the Bayer channels. The channels are color-coded so that they match the CFA alignment of the input Bayer pattern (for example, BGGR, as shown here).

5.4.2 White Balance Statistics

Figure 5-31. White Balance Statistics



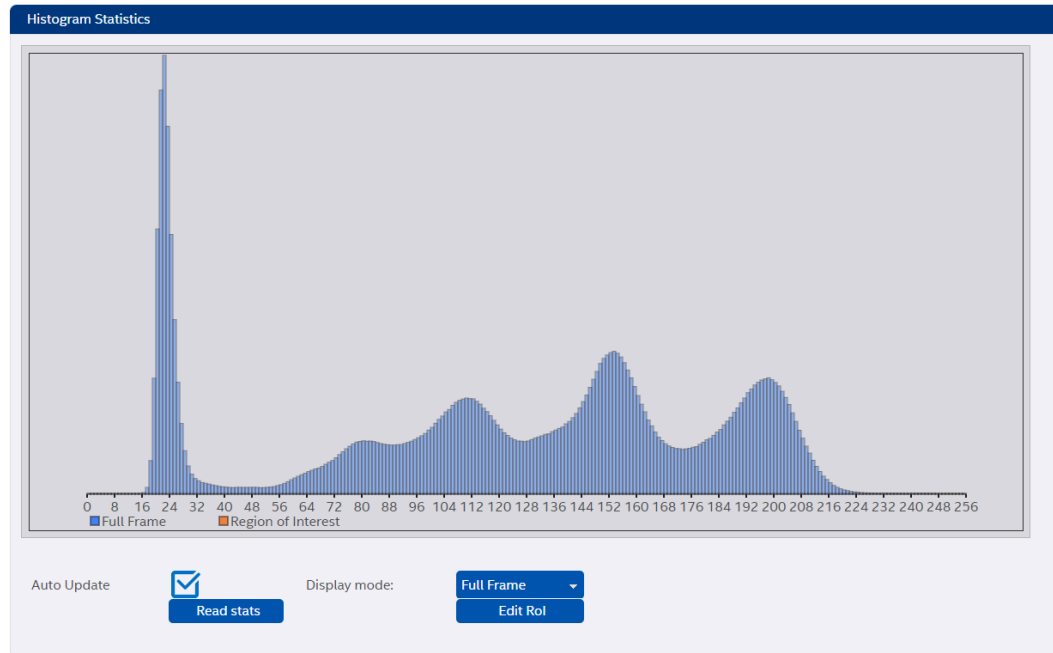
The White Balance Statistics UI shows a color ratio representation of a captured scene that the camera can see. The scene is split into 7x7 zones, and the UI represents the color ratios in each zone. Clicking on a grid square will select it, and the ratio of colors in that zone will be displayed as a bar graph.

It is important to note that the color grid doesn't capture luminosity, only color ratios. Therefore, results in some zones may appear unusual. For example, black and white sections will be indistinguishable as their color ratios are the same.

Since the UI works on a captured scene, use the "Update" button to force a scene capture update.

5.4.3 Histogram Statistics

Figure 5-32. Histogram Statistics



The Histogram Statistics UI shows a luminosity histogram of the given scene. The Autoexposure algorithm uses the Histogram Statistics as part of the brightness control loop present within that system.

The histogram automatically updates at a frequency of 1Hz, which can be disabled using the "Auto Update" checkbox. Manual updates are triggered by clicking the "Read Stats" button.

The Histogram Statistics produces two histograms, one for the entire frame and the other for a Region of Interest. The display mode dropdown box allows either the display of histograms separately or for both histograms to be overlaid simultaneously, distinguishable by different colors.

The region of interest is specified by clicking the "Edit RoI" button and using the ROI editor panel.

6.0 Document Revision History

| Date | Version | Changes |
|--------------|---------|------------------|
| [2024-11-15] | [3.2.0] | Initial release. |
| | | |
| | | |
| | | |
| | | |