

Introduction [\(Ask a Question\)](#)

This application note describes the method to run H.264 video streaming over the Ethernet demo. This solution is developed using PolarFire® SoC FPGA video kit, which features an MPFS250T PolarFire SoC device. Microchip's PolarFire SoC devices combine RISC-V® based 5x core Microprocessor Subsystem (MSS), capable of running Linux®, and the PolarFire Fabric in a single device. This combination enables the partitioning of user designs between the MSS (C Source code) and fabric (RTL). Microchip's Libero® SoC enables the rapid development of RTL based designs for PolarFire SoC and many other device families. It also provides a wide range of IPs for a variety of applications such as video and imaging, signal processing, wired and wireless communications, and networking. Microchip's SoftConsole enables the rapid development of the C/C++ source code based applications targeted for all Microchip FPGA and SoC device families.

This demo design captures live stream from a camera on the PolarFire SoC video kit and performs H.264 compression in the fabric logic. Webserver application running on MSS allows user to connect to the PolarFire SoC video kit through Ethernet using the IP address of the video kit from the web browser. The web page on the browser allows user to control streaming of the live video from PolarFire SoC video kit to the connected computer system. On initiating the streaming from the web page, application triggers MSS that reads compressed stream of data from fabric and sends encoded H.264 RTP ethernet packets with the IP address of the system from which the stream request was initiated. User can play the video on computer using VLC Player, Gstreamer, or FFPlay applications. Web page allows user to download and pass the SDP file to the video player application after streaming is initiated.

PolarFire SoC FPGA video kit enables prototyping of Video and Imaging solutions. The kit supports the following key features among others:

- MPFS250T PolarFire SoC FPGA device
- MIPI CSI-2 interface
- FPGA Mezzanine Card (FMC) connector
- SD and eMMC card, LPDDR4, and DDR4 memories
- HDMI, Ethernet, PCIe®, and other interfaces
- CAN and mikroBUS

For more information about the video kit, see [MPFS250-VIDEO-KIT](#).

Table of Contents

| | |
|-----------------------------------------------------|----|
| Introduction..... | 1 |
| 1. Demo Requirements..... | 3 |
| 2. Demo Prerequisites..... | 4 |
| 3. Design Overview..... | 5 |
| 3.1. IP Blocks..... | 6 |
| 3.2. MSS Configuration..... | 6 |
| 3.3. Input and Output Ports..... | 8 |
| 3.4. Clocking Structure..... | 9 |
| 3.5. Reset Structure..... | 10 |
| 3.6. Resource Utilization..... | 11 |
| 4. Setting Up the Demo..... | 12 |
| 4.1. Setting Up the Hardware..... | 12 |
| 4.2. Setting Up the Serial Terminal..... | 12 |
| 4.3. Programming the Device..... | 14 |
| 5. Running Linux User Applications..... | 16 |
| 5.1. Flashing Linux .wic Image in eMMC Mode..... | 16 |
| 5.2. Flashing Linux .wic Image in SD Card Mode..... | 17 |
| 5.3. Running the H.264 Demo using GUI..... | 17 |
| 6. Appendix A: VLC Configurations..... | 23 |
| 7. Appendix B: Running the Tcl Script..... | 26 |
| 8. Revision History..... | 27 |
| Microchip FPGA Support..... | 28 |
| Microchip Information..... | 28 |
| Trademarks..... | 28 |
| Legal Notice..... | 28 |
| Microchip Devices Code Protection Feature..... | 29 |

1. Demo Requirements [\(Ask a Question\)](#)

The following table lists the setup requirement for running the demo.

Table 1-1. Demo Requirements

| Requirement | Description |
|---------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Hardware and Accessories | |
| PolarFire® SoC video kit | MPFS250-VIDEO-KIT Kit Contents: <ul style="list-style-type: none"> • PolarFire SoC Video Kit Board with (MPFS250TS-1FCG1152I) • Sony IMX334 4K30 Dual Camera Sensor • HDMI cable • Two Micro B USB cables • RJ45 Ethernet cable • 12V AC Adapter, 12V Power Cord |
| Image Sensor Module | LI-IMX334-MIPI-MICRO v1.0 |
| USB A to micro-B cable | Required for: <ul style="list-style-type: none"> • FPGA programming • UART interface for the terminal prompt from Linux® |
| Host PC | A host PC with USB, Ethernet port, and Windows® 10 and later, or Ubuntu 20.04 and later. |
| Ethernet cable | To communicate with the video kit while using the web GUI |
| Utility Software | |
| FPEXpress | FPGA Programming application |
| MobaXTerm , Putty , or TeraTerm | UART receiver-transmitter application at the host PC |
| USBImager | Free utility used for flashing binary images to USB sticks or SD or CF cards |
| 7-zip | Free archiving utility for compressing or decompressing files and is needed for extracting <code>.wic</code> file from <code>.wic.gz</code> |
| Admin privileges | Flashing Linux <code>.wic</code> images using USBImager requires admin privileges |
| VLC Media Player or GStreamer | To stream the H.264 video, use the following specified versions: <ul style="list-style-type: none"> • 3.0.16 (Windows) and later or 3.0.9.2 (Ubuntu) and later for VLC Media Player • 1.22.9 (Windows) and later for GStreamer |
| Web Browser | Chrome 96.0.4664.110 (Official Build) (64-bit) and later, or Microsoft Edge Version 96.0.1054.53 (Official build) (64-bit) and later |
| Flashable Binaries | |
| Linux <code>wic</code> image | Linux <code>.wic</code> Image |
| Job file | MPFS_VIDEO_KIT_H264_MM_DESIGN |



Important: The name of the binaries mentioned in the preceding table may be different from the rest of the guide as these keep changing in each releases, but the flashing and demo procedures remain the same.

2. Demo Prerequisites [\(Ask a Question\)](#)

Before you start, ensure that the following components are in place:

1. Download the Programming Job file
2. Download the Linux `.wic` image file

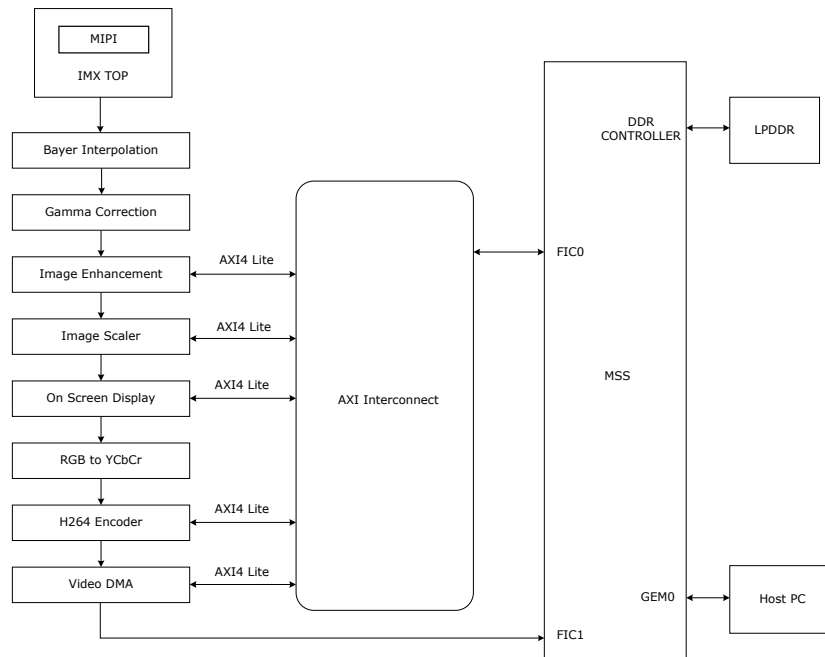


Important: To download the files, see [Demo Requirements](#) section.

3. Design Overview (Ask a Question)

The following figure shows the high-level block diagram of the reference design.

Figure 3-1. High-level Block Diagram



The camera generates a live Full HD (FHD) video (1920x1080 picture resolution) to the Bayer Interpolation IP. The FPGA fabric receives these frames and performs processing as follows:

- IMX334 top block contains MIPI CSI2 Rx IP to get the camera data, which is passed to the Bayer Interpolation IP. Bayer Interpolation IP block converts each Bayer frame to RGB data frame. Specifically, each single 8-bit Bayer pixel data is converted to a 24-bit RGB pixel data.
- The 24-bit RGB pixel data is then passed through Image Processing IP's to add features like auto-brightness, R/G/B gains, contrast and so on.
- The On Screen Display (OSD) logic injects the text "COMPRESSION RATIO x" into the image frame, where x is the value of compression ratio input from software. Each character size is 16x16 pixels. The text characters and digits in 16x16 pixel format are stored in the ROM. The OSD logic injects the text at the location specified by the coordinates input from software. The text color can also be controlled by using the color input of OSD logic. The OSD logic takes RGB as the input frame format and gives RGB frame with the injected text in the specified coordinates as the output.
- RGB data is converted to YUV422 format using RGB to YUV convert IP.
- 1920x1080 image is scaled down to the selected resolution using Image Scaler IP and sent to H.264 Encoder IP. The resolution can be changed from the user application through GUI interface.
- The H.264 compressed data is written to the LPDDR memory through FIC1 AXI slave interface.
- Application running on MSS reads the compressed data from the LPDDR4 memory.
- On Linux, Fast Forward MPEG (FFMPEG) reads FIFO data and generates RTP packets to send over GEM.

RGB Gains, Compression quality factor, Compression Resolution, Contrast, Brightness, and so on are controlled by MSS through FIC3 Master interface.

3.1. IP Blocks [\(Ask a Question\)](#)

The following table lists the IP blocks used in the design and their functionality.

Table 3-1. IP Blocks

| IP Block | Description |
|--------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| PolarFire® Oscillator | The PolarFire RC Oscillator block generates a 2 MHz clock. |
| PolarFire Clock Conditioning Circuit (CCC) | The PolarFire CCC block generates multiple clocks required in the design. |
| PolaFire Clock Divider | The clock divider block performs a divide by operation. |
| H.264 Encoder | The H.264 block performs H.264 compression on YUV4:2:2 data input. |
| mipiccsi2rxdecoder | The MIPI CSI-2 receiver decoder block for PolarFire (MIPI CSI-2 Rx Decoder) decodes the data from the sensor interface. Data Type: RAW-10 Lane Width: 4 Number of Pixels: 1 Inputs Data Invert: 0 FIFO depth: 12 |
| PolarFire IOD Generic RX | The IP block operates with 500 Mbps data rate |
| CoreReset | The IP block synchronizes the reset to the respective clock domain. |
| Microprocessor SubSystem (MSS) | On the MSS hard IP, the Linux user space application receives the H.264 compressed frame from fabric FIFO and streams over Ethernet. For more information about the MSS configuration and LPRDDR4 memory partitioning, see MSS Configuration . |
| CoreAXI4Interconnect | Configuring multiple IP's in the video pipeline is done through FIC0 Initiator interface in MSS. CoreAXI4Interconnect IP is used to re-direct these IP register configuration requests to the corresponding IP's using AXI4-Lite interface at the output interface. |
| Bayer Interpolation | This IP block converts the 8-bit raw data to 24-bit RGB data |
| RGBtoYCbCr | This color space conversion IP block converts RGB 24-bit data format to YUV422 16-bit data format. |
| Gamma Correction | This IP block converts the pixel intensity to match with the perspective of human eye by using a logarithmic curve. |
| Image Enhancement | This IP block adjusts the brightness, contrast, and color balance through user controls. |
| IMX334 | This SmartDesign module receives the live camera feed and converts it into raw 8-bit parallel data. Each byte represents one pixel. |
| INIT MONITOR | The IP block triggers reset to the design. |
| Image Scaler | This IP scales down the input video to a resolution selected by user through web based graphical user interface. |
| MIPI Training Lite | This IP is used for clock data training for MIPI interface. |
| VDMA | Video DMA IP is used to write the frame data to the memory. |

3.2. MSS Configuration [\(Ask a Question\)](#)

The following table lists the configuration of MSS clock, peripherals, and memory.

Table 3-2. MSS Blocks

| MSS Block | Description |
|------------|-----------------------------------------------------------------------------|
| MSS Clocks | MSS PLL reference clock source: 125 MHz MSS CPU clock frequency: 600 MHz |

| MSS Block | Description |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Peripherals | <ul style="list-style-type: none"> eMMC: Used for storing Linux kernel images. Gigabit Ethernet MAC: Used for copying data to/from the Linux OS to Host PC. MMUART: Used to capture HSS boot log and Linux terminal. I²C: Used for initializing the camera. Fabric Interface Controller (FIC1) AXI4 Slave interface for LPDDR access from fabric. |
| DDR Memory | LPDDR4 with the default configuration. |
| Memory Partitioning | See DDR Memory Partition . |

3.2.1. DDR Memory Mappings [\(Ask a Question\)](#)

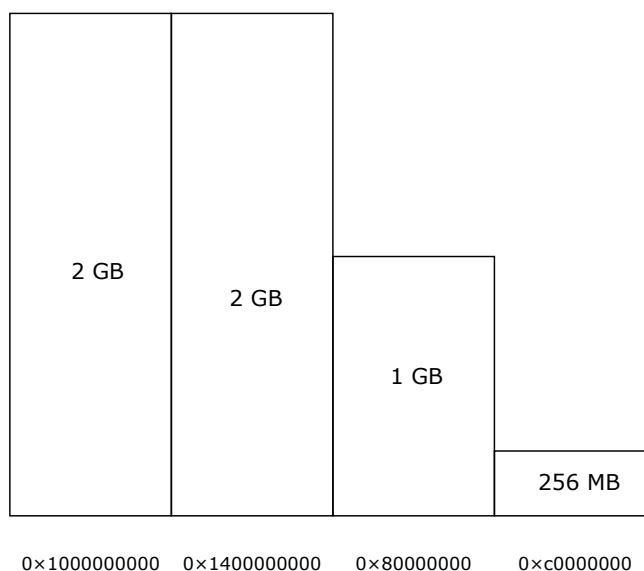
This section describes the DDR regions and their allocations. It also explains how these memory regions are defined in the Linux device tree so that the Linux user space can address these regions.

For more information about DDR memory partition regions, see the “Segmentation Blocks” section in [PolarFire SoC MSS Technical Reference Manual](#).

3.2.1.1. DDR Memory Partition [\(Ask a Question\)](#)

The following figure shows the memory mapping of the 2 GB on-board DDR memory in the Linux device tree.

Figure 3-2. LPDDR4 Memory Mapping



PolarFire SoC video kits come with 2 GB of DDR memory by default, partitioned as follows:

- 2 GB of DDR memory is mapped as listed:
 - 1 GB 32-bit cached memory at 0x80000000
 - 256 MB 32-bit non cached memory at 0xc0000000
 - 2 GB 64-bit cached memory at 0x100000000
 - 2 GB 64-bit non-cached memory at 0x140000000

The following figure shows the **MSS Configurator > DDR Memory Partition** settings used in the design.

Figure 3-3. DDR Memory Partition

| | MSS Offset Address | Range | MSS High Address | Physical DDR Offset Address | Physical DDR High Address |
|------------------|--------------------|--------|------------------|-----------------------------|---------------------------|
| Cached 1GB | 0x8000_0000 | 1 GB | 0xBFFF_FFFF | 0x0000_0000 | 0x3FFF_FFFF |
| Cached 16GB | 0x10_0000_0000 | 2 GB | 0x10_7FFF_FFFF | 0x0000_0000 | 0x7FFF_FFFF |
| Non-Cached 256MB | 0xC000_0000 | 256 MB | 0xCFFF_FFFF | 0x0000_0000 | 0xFFFF_FFFF |
| Non-Cached 16GB | 0x14_0000_0000 | 2 GB | 0x14_7FFF_FFFF | 0x0000_0000 | 0x7FFF_FFFF |

3.3. Input and Output Ports [\(Ask a Question\)](#)

The following table lists the key input and output ports in the design.

Table 3-3. Input and Output Ports

| Port Name | Direction | Description |
|------------------------------------------------------------------------------------------------------------------------------|------------------|-------------------------------------------------------------------------------------------------------------------------------|
| Camera RX, Reset, and Reference Clock Ports | | |
| CAM1_RST | Output | Input signal used to reset the camera sensor module. This signal comes from the MSS. |
| CAM1_RX_CLK_P CAM1_RX_CLK_N | Input | Input pads to receive the reference clock for the camera sensor. |
| CAM1_RXD[0] CAM1_RXD_N[0] CAM1_RXD[1] CAM1_RXD_N[1] CAM1_RXD[2] CAM1_RXD_N[2] CAM1_RXD[3] CAM1_RXD_N[3] | Input | Input pads to receive the live video from the camera sensor module. These pads are assigned to the Bank 7 I/Os. |
| MSS Peripheral Ports | | |
| REFCLK REFCLK_N | Input | Input ports for receiving MSS reference clock from the on-board 125 MHz oscillator. |
| RESET_N | Output | This port is assigned to AB12 pin of Bank 6 MSS DDR. This port is used to reset the on-board LPDDR4. |
| MMUART_0_RXD_F2M MMUART_0_TXD_M2F MMUART_1_RXD_F2M MMUART_1_TXD_M2F | Input and Output | Input and Output ports assigned to C7, B7, D4, and C4 pins. These pins are connected to the on-board CP2108 USB to UART chip. |
| USB_CLK | Input | USB clock from the on-board USB3340-EZK-TR IC. |
| USB_DATA0 USB_DATA1 USB_DATA2 USB_DATA3 USB_DATA4 USB_DATA5 USB_DATA6 USB_DATA7 | Inout | USB data ports assigned to Bank 2 MSS I/Os. |
| EMMC_CLK | Output | This port is assigned to AA8 pin of the Bank 4 to provide clock to the eMMC device. |
| EMMC_CMD | Inout | This port is assigned to AA9 pin of Bank 4 to receive and forward eMMC commands. |

Table 3-3. Input and Output Ports (continued)

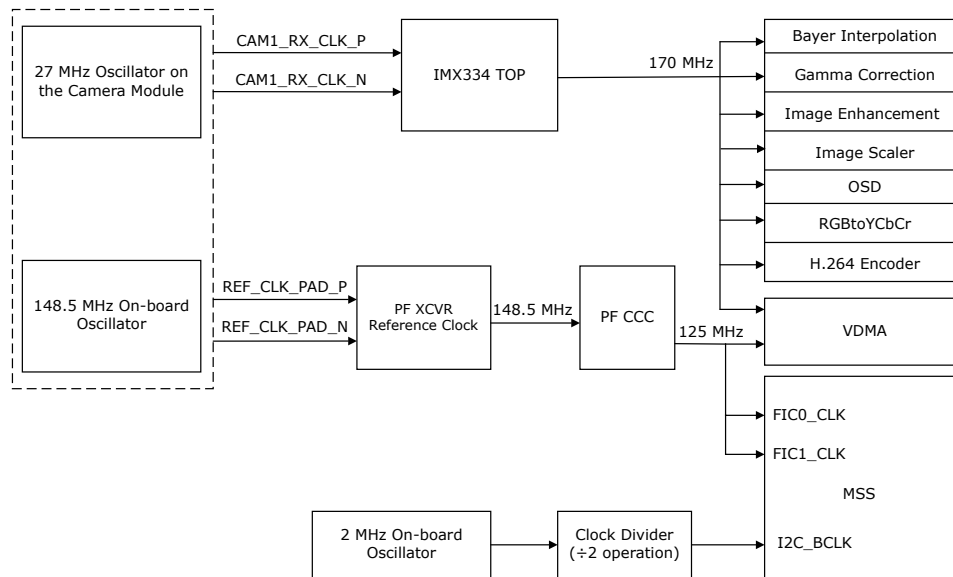
| Port Name | Direction | Description |
|--------------------------------------------------------------------------------------------------------------|-----------|--------------------------------------------------------------------------------------------------|
| EMMC_DATA0 EMMC_DATA1 EMMC_DATA2 EMMC_DATA3 EMMC_DATA4 EMMC_DATA5 EMMC_DATA6 EMMC_DATA7 | Inout | eMMC data ports assigned to Bank 4 MSS I/Os. |
| SDIO_SW_EN_N | Output | Enable Secure Digital Input Output (SDIO) port. |
| SDIO_SW_SEL0 SDIO_SW_SEL1 | Output | SDIO selection pin. |
| SGMII_RX1_N SGMII_RX1_P | Input | Input ports assigned to Bank 5. These pins are connected to the on-board VSC8662 PHY device. |
| SGMII_TX1_N SGMII_TX1_P | Output | Output ports assigned to Bank 5. These pins are connected to the on-board VSC8662 PHY device. |
| Transceiver Ports | | |
| REF_CLK_PAD_N REF_CLK_PAD_P | Input | Input ports for receiving the transceiver reference clock from an on-board 148.5 MHz oscillator. |
| LED2 LED3 | Output | GPIOs connected to on-board user LEDs. |

3.4. Clocking Structure [\(Ask a Question\)](#)

The following figure shows the clocking structure of the design.

- PF_CCC generates 125 MHz fabric clock from 148.5 MHz on-board reference clock. This generated clock is used to drive FIC0 and FIC1 interface of the MSS. This also acts as the AXI4-Lite interface clock.
- The on-board 27 MHz Oscillator on the camera module is used as a reference clock for IOD in the IMX334 block. The CCC inside the IMX334 uses reference clock from IOD and generates 170 MHz clock.

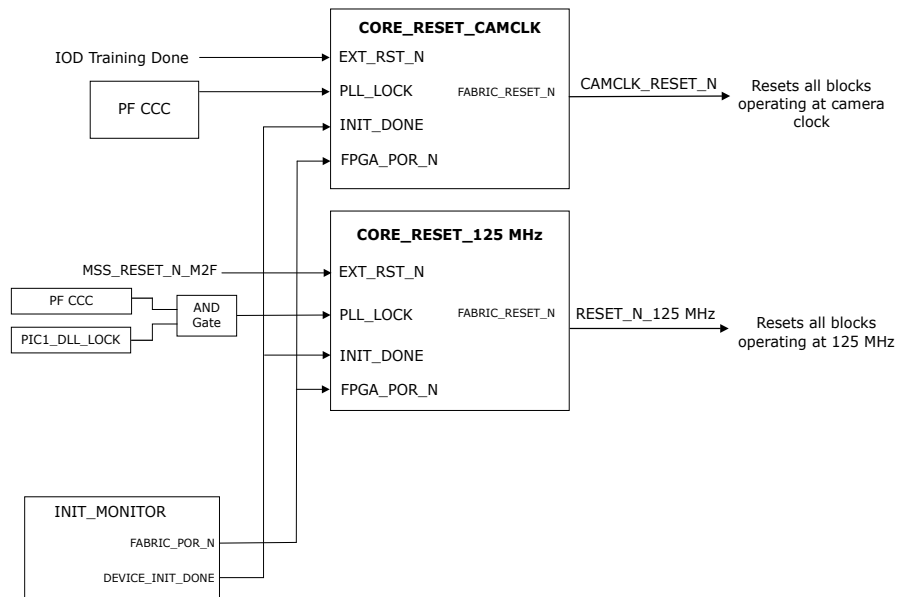
Figure 3-4. Clocking Structure



3.5. Reset Structure [\(Ask a Question\)](#)

The following figure shows the reset structure of the design.

Figure 3-5. Reset Structure



The INIT_MONITOR IP asserts the following signals:

- FABRIC_POR_N: This signal is asserted after the initialization of the fabric. FABRIC_POR_N is used to reset the MSS.
- DEVICE_INIT_DONE: This signal is asserted after the initialization of the entire PolarFire SoC device.

After asserting the FABRIC_POR_N, DEVICE_INIT_DONE, and MSS_RESET_N_M2F, the CORERESET_PF IP blocks generate the Reset signals.

3.6. Resource Utilization [\(Ask a Question\)](#)

The following figure shows the resource utilization of the design.

Figure 3-6. Resource Utilization

| Module Name | Fabric 4LUT | Fabric DFF | Interface 4LUT | Interface DFF | Single-Ended I/O | Differential I/O Pairs | uSRAM (64x12) | LSRAM (20K) | Math (18x18) | Chip Globals | PLL |
|--------------------------|-------------|------------|----------------|---------------|------------------|------------------------|---------------|-------------|--------------|--------------|-----|
| Top | 33356 | 27240 | 8148 | 8148 | 98 | 15 | 64 | 167 | 38 | 8 | 2 |
| Primitives | 1 | 0 | 0 | 0 | 29 | 0 | 0 | 0 | 0 | 0 | 0 |
| CLOCKS_AND_RESETS_inst_0 | 2 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 1 |
| Primitives | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| CORERESET_CLK_125MHz | 1 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PF_CCC_CO_0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| PF_CLK_DIV_CO_0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| FIC_CONVERTER_0 | 4894 | 2016 | 504 | 504 | 0 | 0 | 42 | 0 | 0 | 1 | 0 |
| MSS | 2 | 0 | 0 | 0 | 78 | 10 | 0 | 0 | 0 | 0 | 0 |
| Video_Pipeline_0 | 28457 | 25208 | 7644 | 7644 | 0 | 5 | 22 | 167 | 38 | 5 | 1 |
| IMX334_IF_TOP_0 | 2936 | 3329 | 396 | 396 | 0 | 5 | 0 | 11 | 0 | 4 | 1 |
| Primitives | 1 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| CORERESET_PF_C2_0 | 2 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| CORERESET_PF_C3_0 | 0 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PF_CCC_C1_0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| PF_IOD_GENERIC_RX_CO_0 | 1427 | 1742 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 2 | 0 |
| mipi_lp_0 | 1506 | 1552 | 396 | 396 | 0 | 0 | 0 | 11 | 0 | 1 | 0 |
| VDMA_CO_0 | 1013 | 1112 | 624 | 624 | 0 | 0 | 1 | 17 | 0 | 0 | 0 |
| video_processing_0 | 24508 | 20767 | 6624 | 6624 | 0 | 0 | 21 | 139 | 38 | 1 | 0 |
| Bayer_Interpolation_CO_0 | 817 | 339 | 108 | 108 | 0 | 0 | 0 | 3 | 0 | 0 | 0 |
| Gamma_Correction_CO_0 | 465 | 22 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| H264_Iframe_Encoder_CO_0 | 21344 | 19133 | 4248 | 4248 | 0 | 0 | 21 | 88 | 23 | 1 | 0 |
| Image_Enhancement_CO_0 | 141 | 202 | 108 | 108 | 0 | 0 | 0 | 0 | 3 | 0 | 0 |
| Image_Scaler_top_0 | 810 | 675 | 1836 | 1836 | 0 | 0 | 0 | 48 | 3 | 0 | 0 |
| RGBtoYCbCr_CO_0 | 94 | 69 | 324 | 324 | 0 | 0 | 0 | 0 | 9 | 0 | 0 |
| delay_register_0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| osd_top_0 | 837 | 326 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

4. Setting Up the Demo [\(Ask a Question\)](#)

The demonstration involves the following steps:

- [Setting Up the Hardware](#)
- [Setting Up the Serial Terminal](#)
- [Programming the Device](#)

4.1. Setting Up the Hardware [\(Ask a Question\)](#)

Setting up the hardware involves interfacing the camera sensor module and the HDMI monitor with the PolarFire SoC video kit.

Follow these steps:

1. Connect the camera sensor module at **J10** on the video kit.
2. Ensure the jumper settings on the video kit are set as per the “Jumper Settings” section in [PolarFire SoC FPGA Video Kit User Guide](#).

4.2. Setting Up the Serial Terminal [\(Ask a Question\)](#)

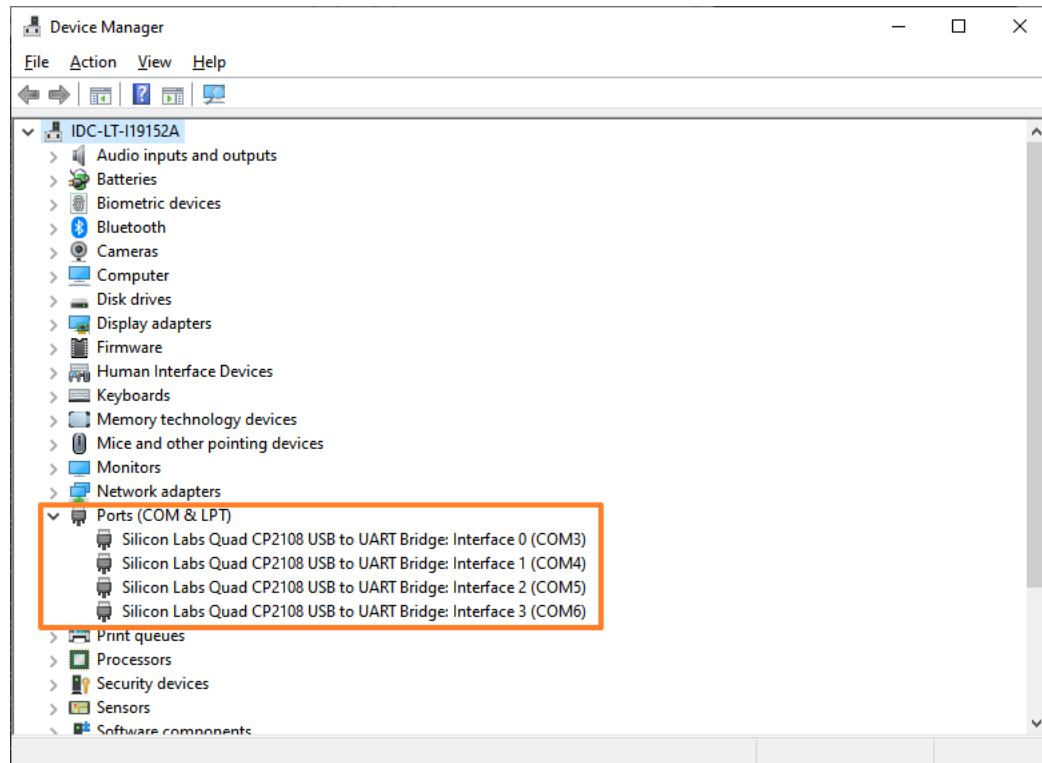
PolarFire SoC Video kit includes CP2108 USB to UART chip to interact with on-board Linux OS. Before you start:

- Download and install [CP210x Universal Windows Driver](#).
- Unzip and see the [CP210x_Universal_Windows_Driver_ReleaseNotes](#).

After installing the driver, follow these steps:


1. Connect USB cable at **J12** port on the PolarFire SoC video kit board to the host PC.
2. After connecting the power adapter to the board at **J39**, switch ON the boards power supply using the SW5 switch. This must detect the USB UART chip on the board at the host PC. You can confirm this at the host PC device manager, see the following figure.

Figure 4-1. Device Manager



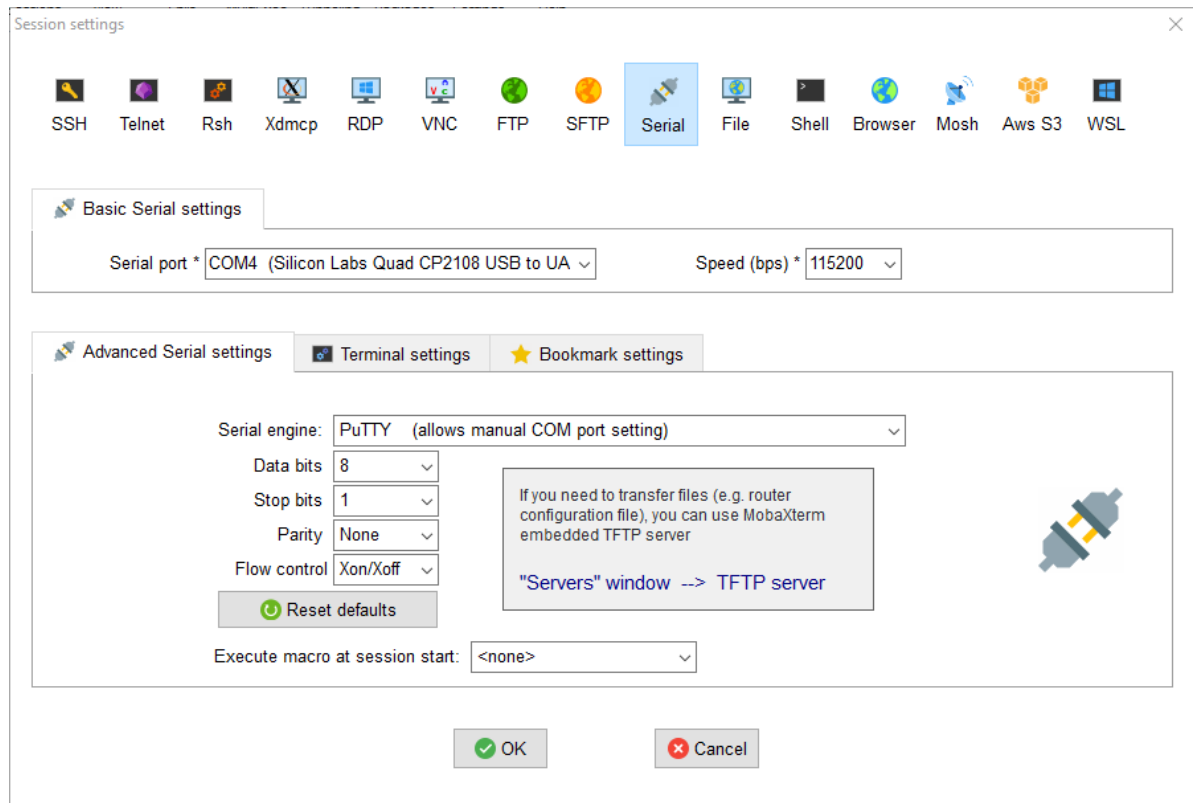
The interfaces on the host PC's device manager are active in the current demo project and display the messages when appropriately configured.

Interface 0 displays the Hart Software Service (HSS) boot messages while Interface 1 displays U-Boot messages, Linux boot messages, and provides a Linux prompt.

 **Important:** For each user, the interfaces might be on a different COM# than what is shown in the preceding figure.

An application like MobaXterm/TeraTerm at the host PC is required to establish serial communication with the video kit board. The baud rate for such connections must be 115200 bps, see the following figure.

Figure 4-2. Serial Port Settings



3. Open Interface 0 and Interface 1.

➔ Important: The data rate must be configured for all interfaces to establish successful communication with the PolarFire SoC video kit.

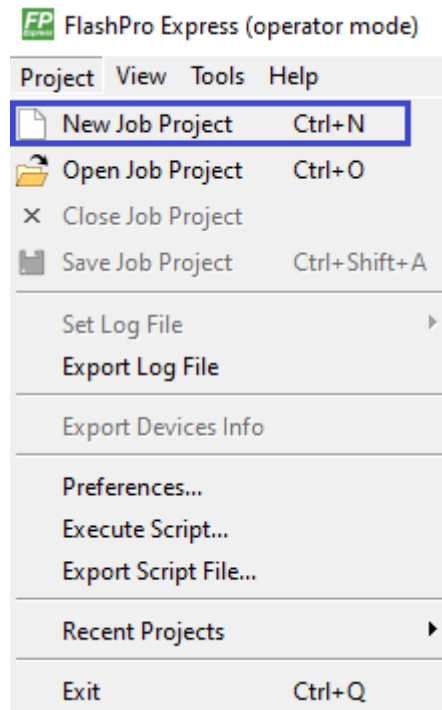
4.3. Programming the Device [\(Ask a Question\)](#)

This section describes the steps to program the PolarFire SoC device with the `VKPFSoC_H264.job` file using FlashPro Express.

Follow these steps:

1. Ensure to close pins 1 and 2 of J28 jumper.
2. Connect a micro-USB to **J5** from the host PC and start the FPEXpress software from **Windows > Start**.
3. Ensure to power on the PolarFire SoC video kit.
4. Select **New** or **New Job Project** from the **Project** menu to create a new job project, see the following figure.

Figure 4-3. New FlashPro Express Project



5. In the **New Job Project** dialog box, enter the following:
 - Programming job file: Click **Browse** and navigate to the location where the job file is located and select the file.
 - FlashPro Express job project location: Select **Browse** and navigate to the location where you want to save the project.
6. Click **Open**. The required programming file is selected and ready to be programmed in the device. The FlashPro Express window appears. Confirm that a programmer number appears in the Programmer field. If it does not, check the board connections and click **Refresh/Rescan Programmers**.
7. Click **RUN** to program the device. When the device is programmed successfully, a RUN PASSED status is displayed.
8. Close FlashPro Express (**Project > Exit**).
9. Power-cycle the board.

5. Running Linux User Applications [\(Ask a Question\)](#)

The Linux `.wic` image is packaged with all the demo applications. Extract the `core-image-minimal-dev-mpfs-video-kit-xxxx.rootfs.wic.gz` with 7zip utility to any folder. The extracted `core-image-minimal-dev-mpfs-video-kit-xxxx.rootfs.wic` image can be flashed either in eMMC or SD card. Irrespective of the flashing device, running the demo user applications remain the same.



Important: If the Linux image has multi-compression extensions like `wic.gz.zip`, extract till `.wic` is obtained.

5.1. Flashing Linux `.wic` Image in eMMC Mode [\(Ask a Question\)](#)

This section describes the steps to put the HSS in CLI mode, program the Linux images into eMMC using the USBImager application and initiate the Linux boot.

1. Do not insert SD card to the board. If present, SD card booting takes priority.
2. Remove the micro-USB cable connected to **J5** and connect it to **J19** (USB-OTG connector) of the board. When the USB cable is detected, the LED near **J19** glows in green.
3. Connect one more micro-USB cable to **J12** from the host PC.
4. **J18** and **J57** must be open.
5. Close pins 2 and 3 of **J36**.
6. Power ON the board and press Enter on Interface 0 while booting.



Important: The COM port number might be different for different users. To identify the COM port, see [Setting Up the Serial Terminal](#).

7. Follow the steps in [eMMC Content Update Procedure](#).
8. After programming the eMMC and rebooting, check the serial terminal application connected to Interface 1. Linux will be up and running, waiting for user to login.

Figure 5-1. Linux User Login

```

[ OK ] Reached target Preparation for Network.
Starting Network Configuration...
[ OK ] Started D-Bus System Message Bus.
[ OK ] Started User Login Management.
[ OK ] Started Network Configuration.
[ 15.921240] macb 20112000.ethernet eth1: PHY [20110000.ethernet-ffffffff:09] driver [Generic PHY] (irq=POLL)
[ 15.931273] macb 20112000.ethernet eth1: configuring for phy/sgmii link mode
[ 15.939521] pps pps0: new PPS source ptp0
[ 15.944065] macb 20112000.ethernet: gem-ptp-timer ptp clock registered.
Starting Netwo[ 15.955389] macb 20110000.ethernet eth0: PHY [20110000.ethernet-ffffffff:08] driver [Generic PHY] (irq=POLL)
rk Name Resoluti[ 15.966142] macb 20110000.ethernet eth0: configuring for phy/sgmii link mode
on...
[ 15.975297] pps pps1: new PPS source ptp1
[ 15.979902] macb 20110000.ethernet: gem-ptp-timer ptp clock registered.
[ OK ] Started Network Name Resolution.
[ OK ] Reached target Network.
[ OK ] Reached target Host and Network Name Lookups.
[ OK ] Started The Apache HTTP Server.
Starting Permit User Sessions...
[ OK ] Finished Permit User Sessions.
[ OK ] Started Getty on tty1.
[ OK ] Started Serial Getty on ttyS0.
[ OK ] Started Serial Getty on ttyS1.
[ OK ] Reached target Login Prompts.
[ OK ] Reached target Multi-User System.
Starting Record Runlevel Change in UTMP...
[ OK ] Finished Record Runlevel Change in UTMP.

OpenEmbedded nodistro.0 mpfs-video-kit ttyS1

mpfs-video-kit login: [ 18.032788] macb 20110000.ethernet eth0: Link is Up - 100Mbps/Full - flow control off
[ 18.040767] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready

```

This step demonstrated the process of running the Linux user application by programming the Linux user application image onto the on-board 8 GB eMMC NAND Flash using the `.wic` file. Upon completion, the Linux OS booted up successfully from the eMMC as indicated by the Linux OS login prompt displayed to the user.

5.2. Flashing Linux .wic Image in SD Card Mode [\(Ask a Question\)](#)

This section describes the steps to program the Linux images into SD card using the SD card reader and USBImager application, and initiate the Linux boot.



Important: This step is required only to boot from SD card instead of on-board 8 GB eMMC NAND Flash. Skip this step to boot the Linux from eMMC Flash and run the demo.

1. Power OFF the board.
2. Follow the steps in [SD Card Content Updated Procedure](#).

5.3. Running the H.264 Demo using GUI [\(Ask a Question\)](#)

To run the H.264 demo using GUI, perform the following steps:

1. Insert the dual-camera sensor module in **J10** on the PolarFire SoC video kit. Ensure to remove the camera lens cap.
2. Do not connect any Ethernet cable from the host to the video kit.
3. On the windows host, open command prompt in Admin mode and type `ipconfig`.

Figure 5-2. Executing the ipconfig Command

```

C:\WINDOWS\system32>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet 2:

    Connection-specific DNS Suffix  . : microchip.com
    IPv4 Address. . . . .             : 10.43.224.0
    Subnet Mask . . . . .            : 255.255.255.0
    Default Gateway . . . . .        :

Ethernet adapter Ethernet:

    Media State . . . . .            : Media disconnected
    Connection-specific DNS Suffix  . :
  
```

4. Make a note of the Ethernet adapter name displayed against the Media disconnected interface. It is **Ethernet** in this case.
5. Set the host IP address to 192.168.2.X, where X is any integer from 2 to 255 and the Subnet mask to 255.255.255.0 using the following command. This is a one-time process and need not to be done every time the video kit is connected.

```
netsh interface ip set address "Ethernet" static 192.168.2.100 255.255.255.0
```

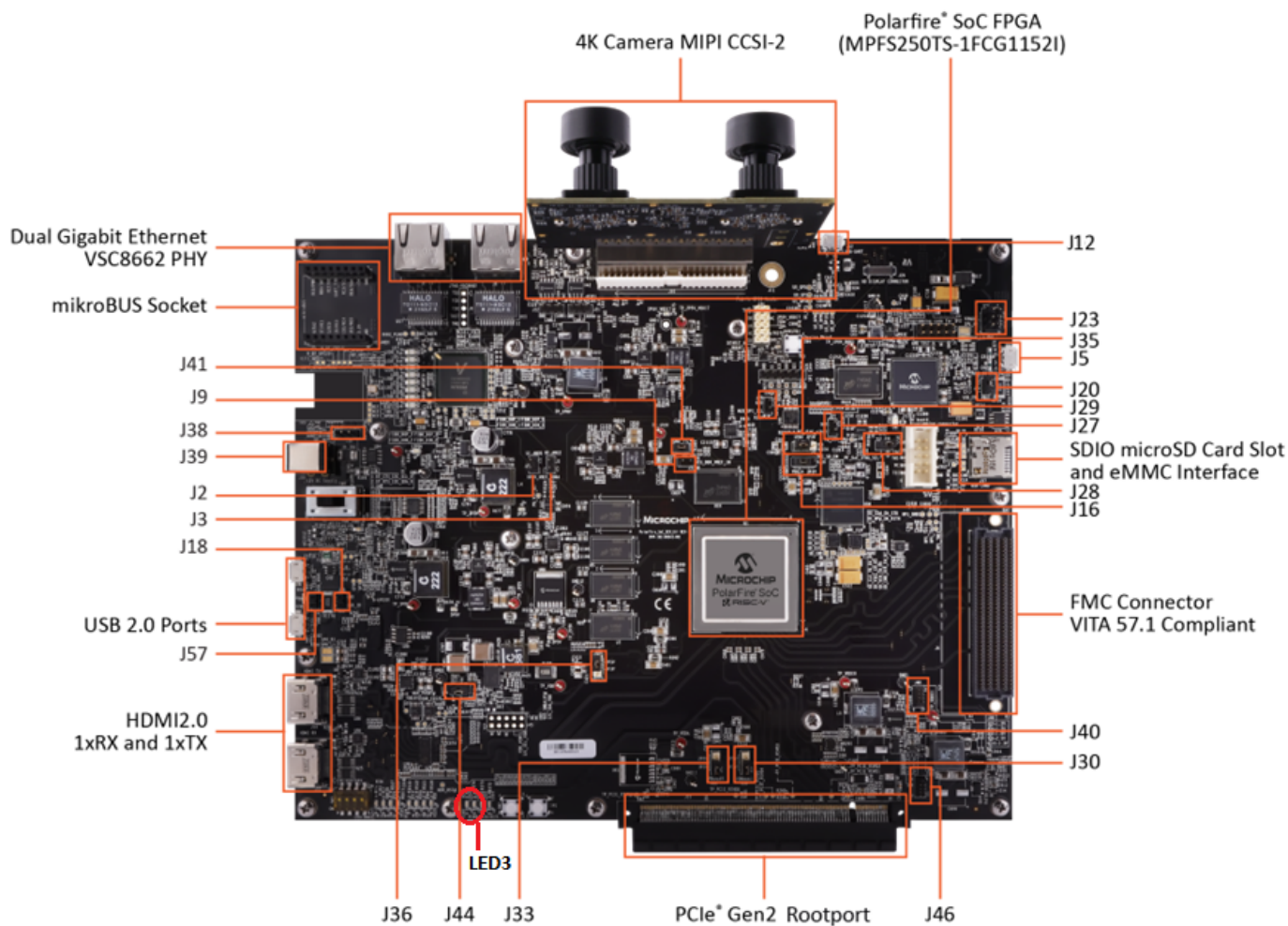
Figure 5-3. Setting the Host IP Address

```

C:\WINDOWS\system32>netsh interface ip set address "Ethernet" static 192.168.2.100 255.255.255.0
  
```

6. Connect an Ethernet cable from video kit **eth0** port to the PC Ethernet port. There are two Ethernet ports on the video kit. The Ethernet port **eth0** is next to the camera mount.
7. Power ON the board and wait for the system boot sequence to complete. The LED highlighted (LED 3) in the following figure glows green once the Linux boot up sequence is completed.
8. Restart the board if LED 3 does not glow.

Figure 5-4. PolarFire SoC Video Kit—LED Indication



9. Log in to Linux using the user name `root` (password is not required).
10. To configure the board for static IP, follow the steps mentioned in [Steps to configure static IP addresses](#).
11. Reboot the video kit and log in again into Linux using the user name `root`.
12. Type `ifconfig`, on the serial terminal connected to Interface 1 (Linux COM Port).
13. It must report some IP, which indicates that the Ethernet port is working, see the following figure.

Figure 5-5. Fetching the Ethernet Port Status

```

OpenEmbedded nodistro.0 mpfs-video-kit ttyS1
mpfs-video-kit login: [ 26.096881] macb 20110000.ethernet eth0: Link is Up - 100Mbps/Full - flow control off
[ 26.104846] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
root
This is version v2023.02 of the Polarfire SoC Yocto BSP.

Updated images and documentation are available at:
  https://github.com/polarfire-soc/
* FPGA reference design
  https://github.com/polarfire-soc/icicle-kit-reference-design/releases
* Yocto releases
  https://github.com/polarfire-soc/meta-polarfire-soc-yocto-bsp/releases
* Buildroot External
  https://github.com/linux4microchip/buildroot-external-microchip
root@mpfs-video-kit:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.2.1 netmask 255.255.255.0 broadcast 192.168.2.255
    inet6 fe80::204:a3ff:fe65:672e prefixlen 64 scopeid 0x20<link>
    ether 00:04:a3:65:67:2e txqueuelen 1000 (Ethernet)
    RX packets 675 bytes 197986 (193.3 KiB)
    RX errors 0 dropped 4 overruns 0 frame 0
    TX packets 87 bytes 5964 (5.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 13

```

14. Check if the connection is successful by pinging the video kit, see the following figure.

Figure 5-6. Pinging the Video Kit

```

C:\WINDOWS\system32>ping 192.168.2.1

Pinging 192.168.2.1 with 32 bytes of data:
Reply from 192.168.2.1: bytes=32 time=1ms TTL=64
Reply from 192.168.2.1: bytes=32 time=1ms TTL=64
Reply from 192.168.2.1: bytes=32 time<1ms TTL=64
Reply from 192.168.2.1: bytes=32 time=3ms TTL=64

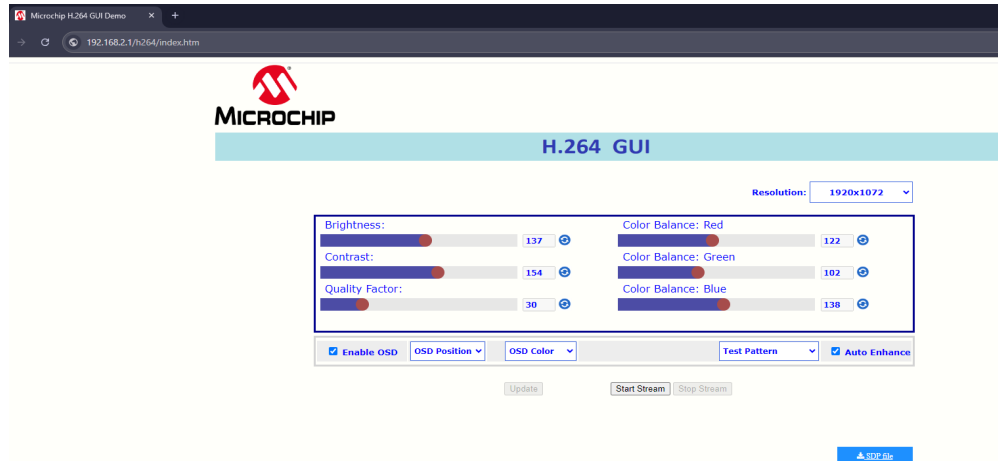
Ping statistics for 192.168.2.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 3ms, Average = 1ms

C:\WINDOWS\system32>

```

15. From the host PC, open the web browser. Enter the IP address of the PolarFire SoC video kit, depending on the Gigabit Ethernet MAC (GEM) you are connected to (J6: 192.168.2.1 or J7: 192.168.2.2), in the address bar and press **Enter**. The H.264 demonstration GUI loads in the browser, see the following figure.

Figure 5-7. H.264 GUI



16. Click the **Start Video** to initiate the video streaming. GStreamer or VLC player can be used to play the video stream. GStreamer has minimal latency as compared to VLC player. Following list contains the procedure to open the video using GStreamer and VLC player:

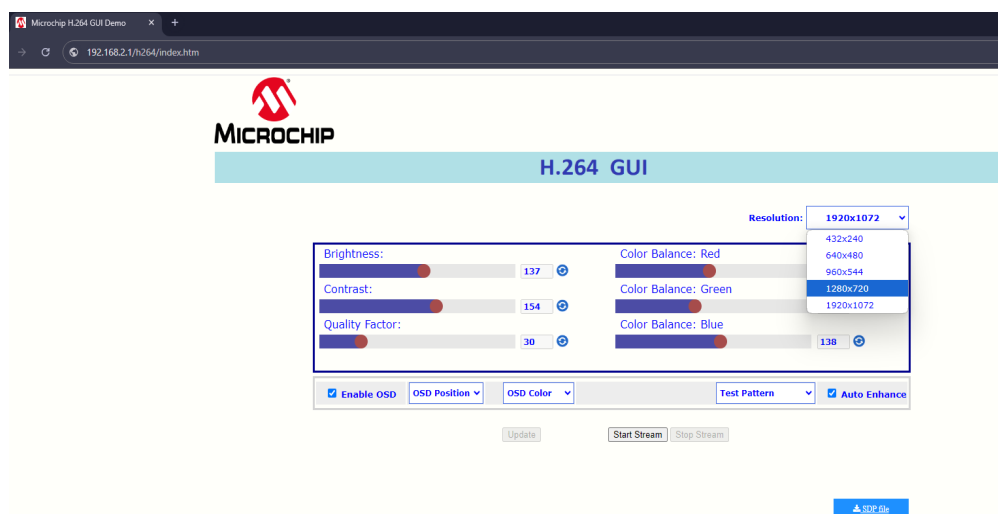
- To play the video using GStreamer in Windows command prompt or Linux terminal, use the following command:

```
gst-launch-1.0 udpsrc port=10000 ! application/x-rtp, payload=96 ! rtpH264depay !
h264parse ! avdec_h264 ! autovideosink
```

- The H264 GUI creates a Session Description Protocol (SDP) file. VLC player uses this file to accept streaming packets from the Ethernet cable. Do the following procedure to open the video using SDP file:
 - To download the SDP file, click **Download SDP file**.
 - To play the video stream, open the SDP file with the VLC player in Windows®/Linux® operating systems. In Linux, you can also execute `vlc` command in the terminal, with the SDP file as argument (for example, `vlc video.sdp`) to play the video stream.

17. Observe a live stream video captured from one of the cameras in the PolarFire SoC video kit. This is a scaled and H.264 compressed video of 1280x720 resolution. To change the resolution, select the resolution and click **Update** as shown in the following figure.

Figure 5-8. Select Resolution



18. Observe the live stream video in the VLC player, as shown in the following figure. Adjust the VLC configurations to reduce the lag in streaming. For more information about VLC configurations, see [Appendix A: VLC Configurations](#).

Figure 5-9. Live Stream Video



This concludes the demo.

To get back from static to DHCP (dynamic), use the following command in the Windows command-line prompt:

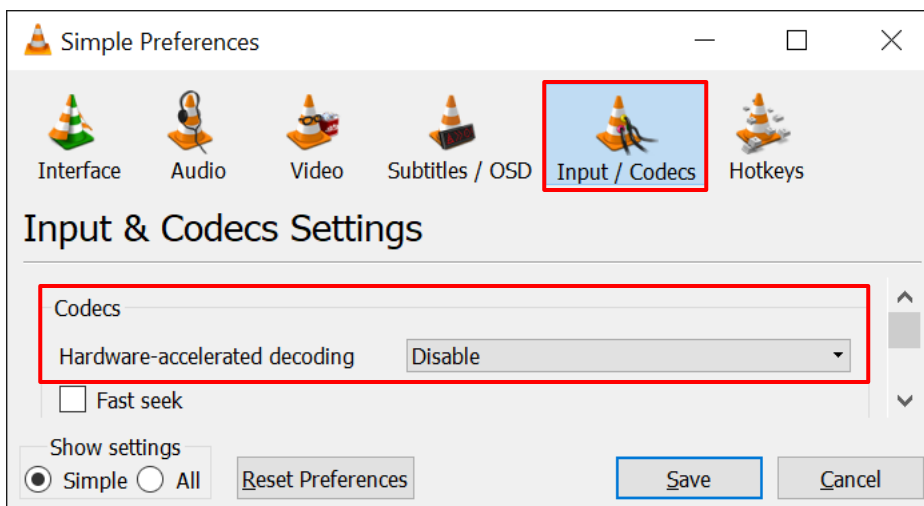
```
netsh interface ip set address name="Ethernet" dhcp
```

6. Appendix A: VLC Configurations [\(Ask a Question\)](#)

Launch the VLC player and configure the following for H.264 streaming.

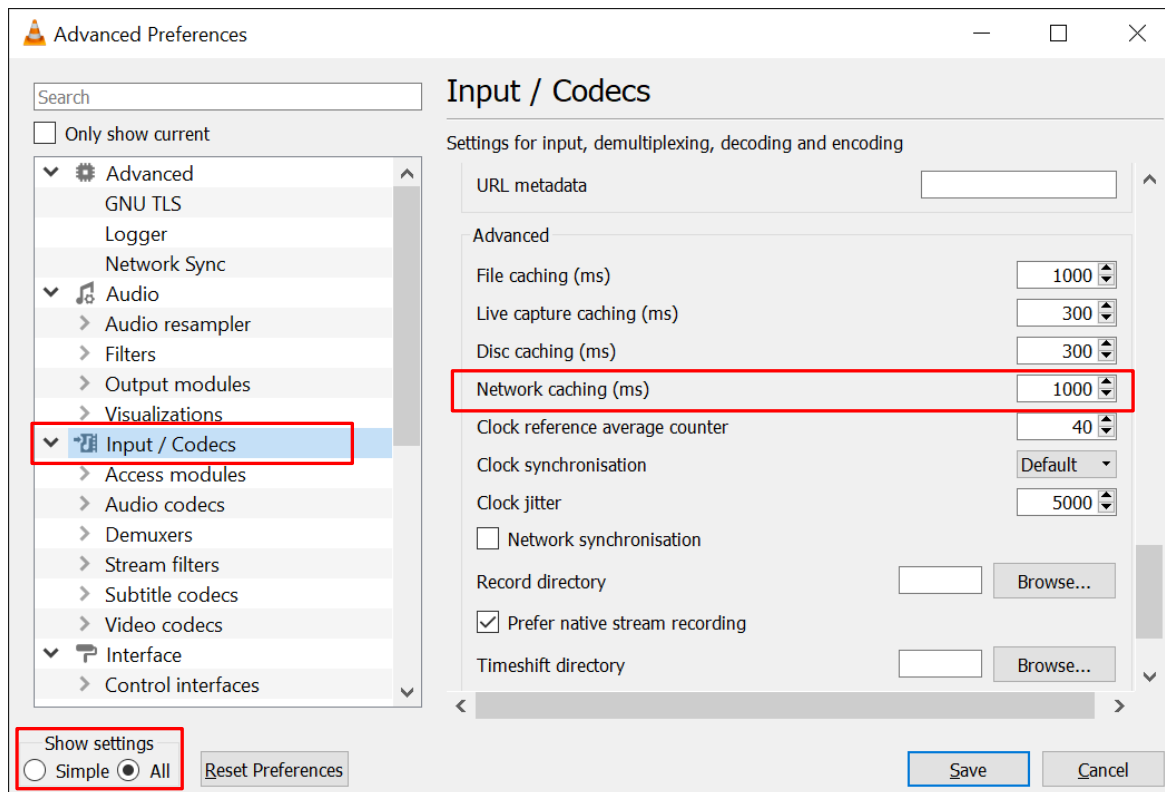
1. Disable the hardware acceleration.
 - a. On the VLC media player menu bar, select **Tools > Preference**.
 - b. In the **Simple Preferences** window, click the **Input / Codecs** tab and set the **Hardware-accelerated decoding** as **Disable**.

Figure 6-1. Simple Preference—Input / Codecs



2. Alter the Network Caching.
 - a. In the VLC media player menu bar, select **Tools > Preference**.
 - b. Select **All** under **Show settings** for **Advanced Preferences**.
 - c. In the **Advanced Preferences** window, select **Input / Codecs** and alter the **Network Caching (ms)** if the file you are trying to play is located on a network share. Increasing the Network Caching (ms) value increases the latency of video playing in VLC.

Figure 6-2. Advanced Preferences—Input / Codecs



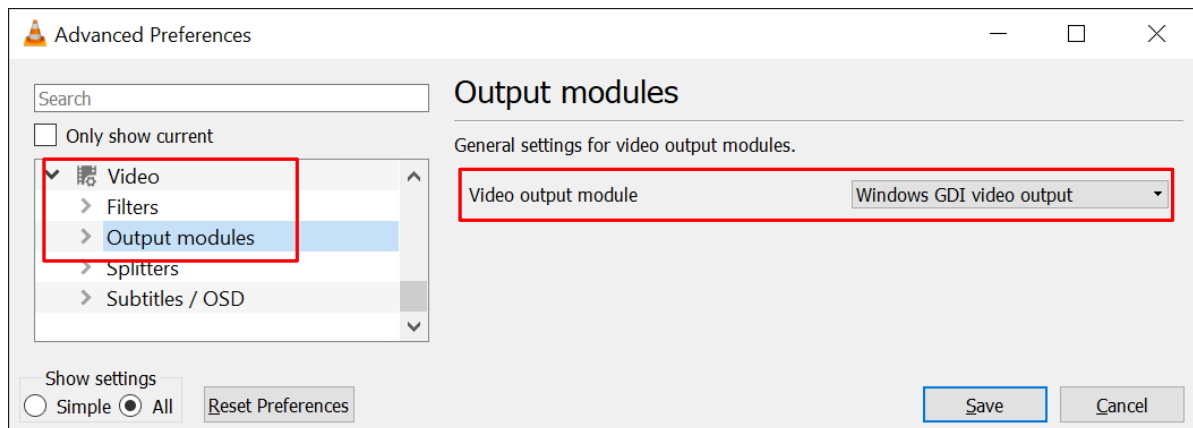
3. Configure the FFMPEG Video Decoder Parameters.
 - a. Select **Input / Codecs** > **Video codecs** > **FFmpeg**.
 - i. Clear **Direct rendering**.
 - ii. Select **Allow speed tricks** and set **Skip the loop filter for H.264 decoding** to **None**.
 - iii. Set **Threads** to 2.

Figure 6-3. Advanced Preferences—Video Codecs



4. Configure the video output module.
 - a. Select **Video > Output modules** and set **Video output module** as **Windows GDI video output**.

Figure 6-4. Advanced Preferences—Video Output Modules



5. Click **Save**.


7. Appendix B: Running the Tcl Script (Ask a Question)

Tcl scripts are provided in [PolarFire SoC Video Kit Reference Design](#).

To run Tcl, follow these steps:

1. Launch the Libero software
2. Select **Project > Execute Script....**
3. Click **Browse** and select `MPFS_VIDEO_KIT_REFERENCE_DESIGN.tcl` from the downloaded **TCL_Scripts** directory.
4. Add any required arguments (for example, `HSS_UPDATE PROGRAM`)
5. Click **Run**.

After successful execution of the Tcl script, the Libero project is created within the **top** directory.

 **Important:** The H.264 Encoder Encrypted IP is Licensed. If the license is not available, the Tcl generates the design with an Evaluation license which expires after an hour's use on the hardware. So, the streaming stops after an hour.

For more information about Tcl scripts, see `readme.md`.

See [Tcl Commands Reference Guide](#) for more information about Tcl commands. Contact Technical Support for any queries about running the Tcl script.

8. Revision History [\(Ask a Question\)](#)

The revision history table describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

| Revision | Date | Description |
|----------|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| E | 06/2025 | The following is the list of changes in the revision E of the document: <ul style="list-style-type: none"> Updated GitHub link related to the eMMC content update procedure in the Flashing Linux .wic Image in eMMC Mode section. Updated GitHub link related to the SD Card content update procedure in the Flashing Linux .wic Image in SD Card Mode section. |
| D | 10/2024 | The following is the list of changes in the revision D of the document: <ul style="list-style-type: none"> Updated Figure 3-1 in the Design Overview section Updated Table 3-1 in the IP Blocks section Updated DDR Memory Partition section Updated Figure 3-4 and Figure 3-5 in the Design Overview section Updated Figure 3-6 in the Resource Utilization section |
| C | 05/2024 | The following is the list of changes in the revision C of the document: <ul style="list-style-type: none"> Updated the software versions in the Demo Requirements section Updated the webpage links in the Demo Prerequisites section Added GStreamer information and updated the GitHub link for the static IP addresses configuration in the Running the H.264 Demo using GUI section |
| B | 03/2023 | <ul style="list-style-type: none"> Replaced all instances of “SEV” with “Video” throughout the document. Updated for Libero SoC v2022.3. Added reference to GitHub for programming a Linux® image on the on-board eMMC and SD Card. See Flashing Linux .wic Image in eMMC Mode and Flashing Linux .wic Image in SD Card Mode. |
| A | 04/2022 | The first publication of this document. |

Microchip FPGA Support

Microchip FPGA products group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, and worldwide sales offices. Customers are suggested to visit Microchip online resources prior to contacting support as it is very likely that their queries have been already answered.

Contact Technical Support Center through the website at www.microchip.com/support. Mention the FPGA Device Part number, select appropriate case category, and upload design files while creating a technical support case.

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

- From North America, call **800.262.1060**
- From the rest of the world, call **650.318.4460**
- Fax, from anywhere in the world, **650.318.8044**

Microchip Information

Trademarks

The “Microchip” name and logo, the “M” logo, and other names, logos, and brands are registered and unregistered trademarks of Microchip Technology Incorporated or its affiliates and/or subsidiaries in the United States and/or other countries (“Microchip Trademarks”). Information regarding Microchip Trademarks can be found at <https://www.microchip.com/en-us/about/legal-information/microchip-trademarks>.

ISBN: 979-8-3371-1406-4

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products are strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.