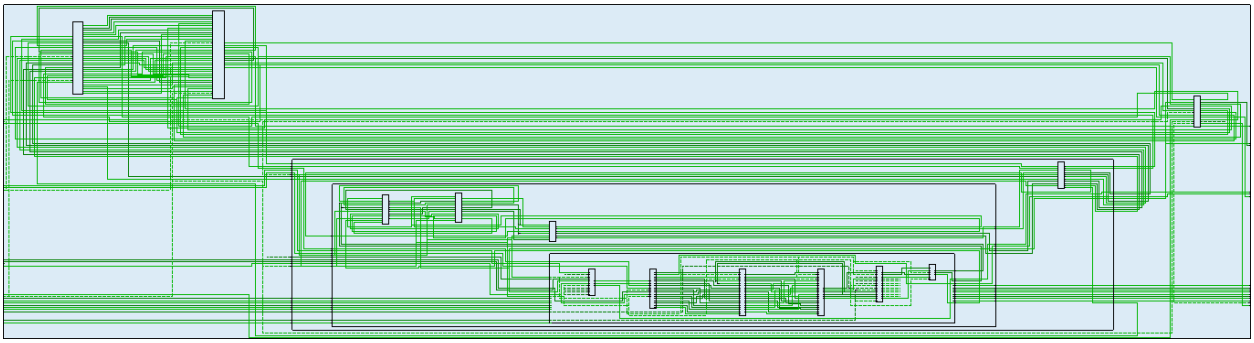
Introduction

Under Apertus Association in Google Summer of Code 2019, I worked in implementing lossless JPEG 1992 core and it's supporting software.

While raw images provide high quality, raw huge size limit FPS and cause difficulties in transition and storing.

My task was to implement lossless JPEG 1992 core to be placed in Axiom Beta and Axiom Micro FPGAs, the core will compress the sensor data (raw stream) with no loss in quality what so ever. This will enable higher FPS and/or lower bandwidth.

I have completed the main core that will compress the stream of data, I have also wrote software to control the core and the DMA and wrote several tools to manipulate, generate and analyze raw and lossless files. I have also worked in an open source DMA for the core, and tested open source library that can decode resulting files.



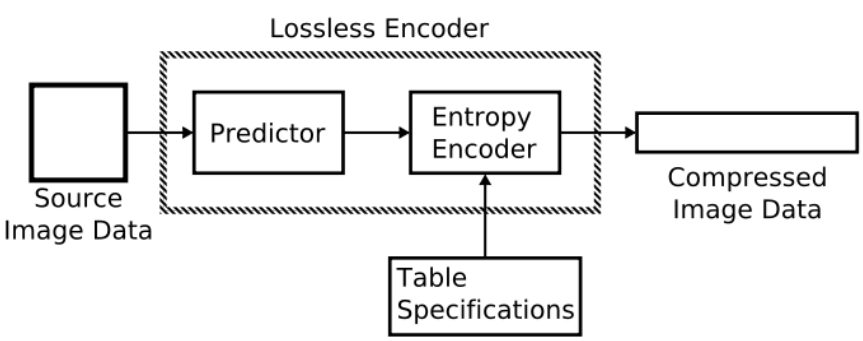
Core github link

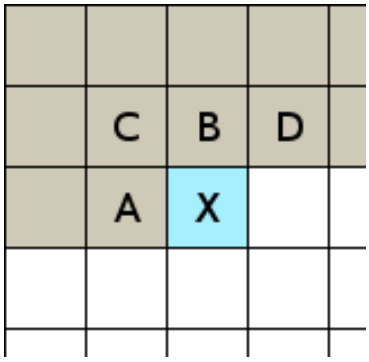
Library github link

Raw image tools github link

Lossless JPEG

Lossless JPEG was developed as a late addition to JPEG in 1993, using a completely different technique from the lossy JPEG standard. It uses a predictive scheme based on the three nearest (causal) neighbors (upper, left, and upper-left), and entropy coding is used on the prediction error.



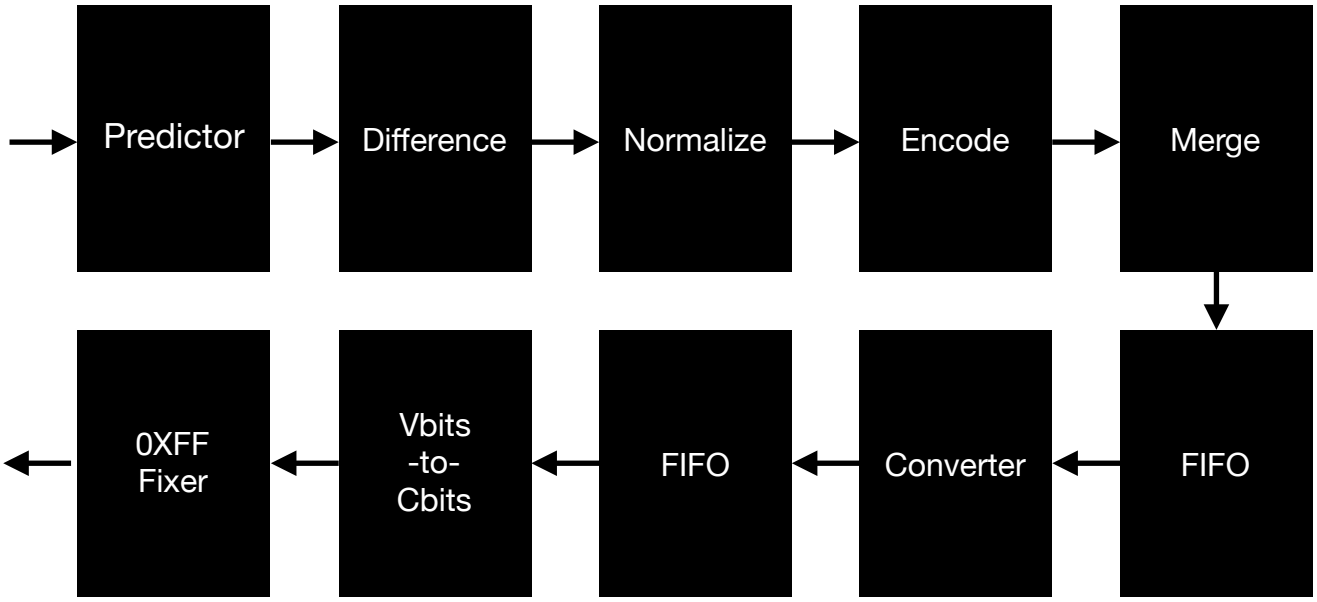


I will short it as **LJ92**.

LJ92 In Axiom Cameras

Lj92 will be placed in the USB3.0 pipeline, It will provide a typical compression of (**55%** to **75%**) which practically will provide more FPS and/or lower bandwidth with no effect on the quality what so ever.

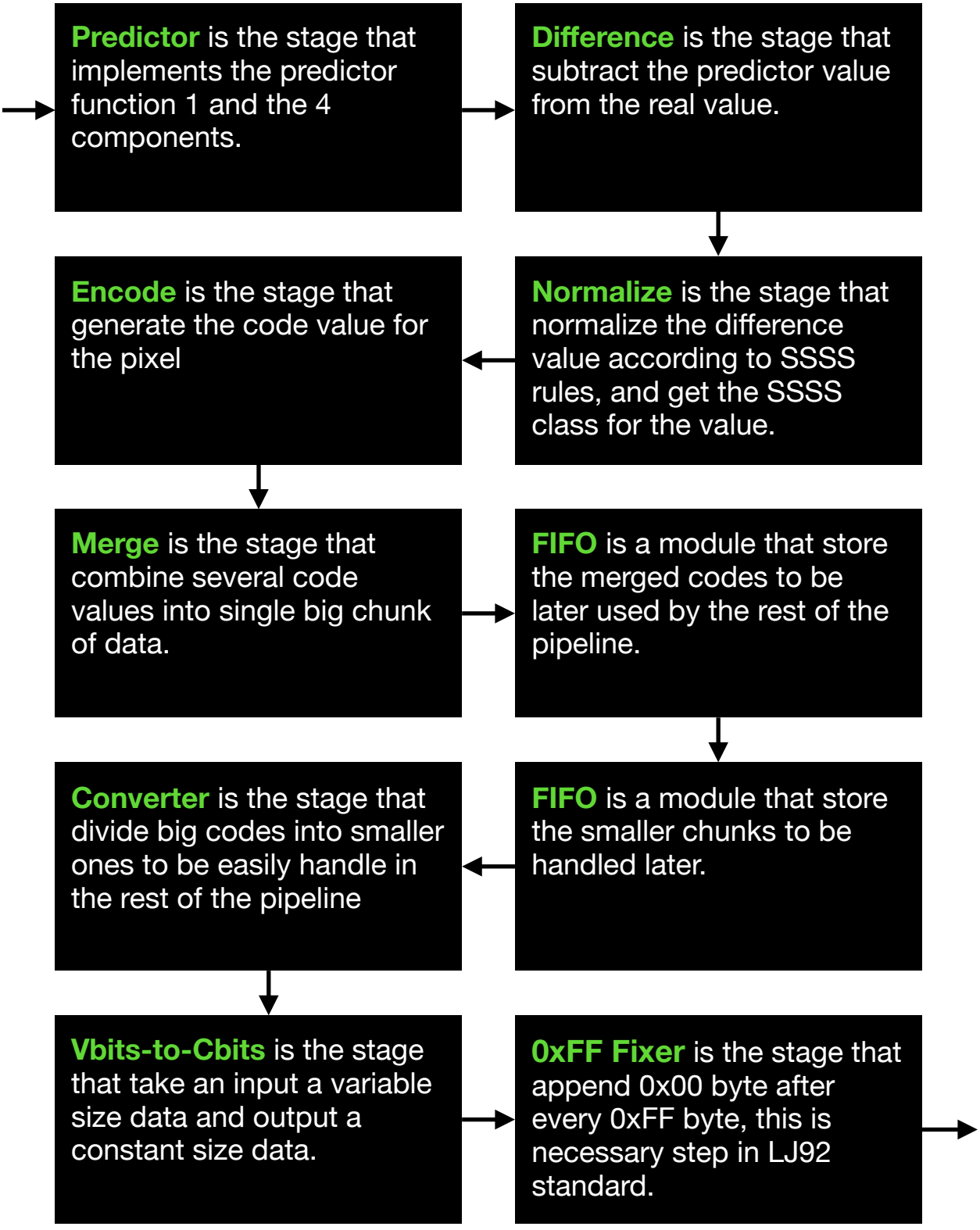
LJ92 Pipeline



LJ92 Beta/Micro Technical Details

- Input: 2 pixels per cycle for Axiom Beta and 1 pixel per cycle for Axiom Micro.
- Output: 16bits of data.
- **LJ92** uses predictor function 1.
- **LJ92** uses 4 components.
- **LJ92** handles 12bits images.
- **LJ92** adds marker at the beginning and ending of each frame.
- **LJ92** stream is [starting_marker][compressed_frame][ending_marker]…
- starting_marker is 16bytes of 0xFF.
- ending_marker is 16bytes of 0xFF in case of complete frame. or 0xFFFE in case of incomplete frame.

LJ92 Pipeline In Details



Receiver End

In receiver end, the receiver will receive a stream from the camera, the receiver must detect and extract each frame, append the header and the tailer to every frame.

The receiver will make use of **the following information** when generating the header.

- **LJ92** uses predictor function 1.
- **LJ92** uses 4 components.
- **LJ92** bit depth is 12bits.
- **LJ92** SSSS as following unless changed via AXI-Lite interface.
  (0b1110, 4)
  (0b000, 3)
  (0b001, 3)
  (0b010, 3)
  (0b011, 3)
  (0b100, 3)
  (0b101, 3)
  (0b110, 3)
  (0b11110, 5)
  (0b111110, 6)
  (0b1111110, 7)
  (0b11111110, 8)
  (0b111111110, 9)
- **LJ92** width and height are only set via AXI-Lite interface.

Please refer to **lj92_image.cpp** to check how header and tailer are generated using those information.

Sender End

Axiom cameras are generally the senders end, The **LJ92** will be placed between the memory reader and the USB3.0 module.

The **LJ92** core will be in IDLE state until the first valid input, and the core will terminate after all pixels are inputted and processed.

The core must be reseted in FPGA before every frame.

Here is the steps needed to run the core:

- Reset the core or load the bit file.
- Set the wanted height and width.
- Set the wanted maximum allowed cycle before force end the frame.
- Change the SSSS information if needed - default in previous slide.
- Setting or changing the data happens via AXI-Lite Interface with no particular order.
- Only after setting all the data correctly you may start the recording.
- Between every frame and the other you must reset the core and wait one cycle until the height, width and allowed_cycles are set again.
- After resetting, The SSSS data will stay the same, the height, width and allowed_cycles will get erased and set in the next cycle since all the core is reseted but not the core_axi_lite module.

Information about reset/clk signals:

- **clk** and **rst** signals do connect to all the core but not the core_axi_lite module.
- **axi_lite_clk** and **axi_lite_rst** do connect to the core_axi_lite only.
- **rst** and **axi_lite_rst** are both active high signals.
- Before every frame, you must only reset the “**rst**” signal but not the “**axi_lite_rst**” signal.
- **clk** and **axi_lite_clk** MUST connect to the same **clk** source.

Receiver Recording Options

Axiom cameras will be connected to recorder devices through USB3.0 modules.

The recorder devices responsibility to extract the frames from the stream and encapsulate it with suitable container like DNG or MLV.

For uncompressed raw, the receiver end can easily extract every frame as every frame is the same size, that is not the case with lossless stream since every frame is different size.

To be able to extract every lossless frame, there is a marker preceding and following every frame.

For now the receiver end should look for the starting marker and ending marker to extract every frame. then add this frame to the container.

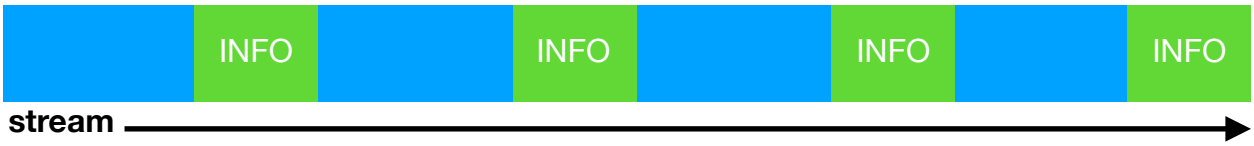
While this method is very simple, it is processing consuming for the receiver and it may limit the FPS in low end recorders.

Future work:

As mentioned, the marker method is simple but it may limit FPS, to overcome this problem I propose a way that may be implemented in the future.

The proposed method is to put information about the stream in known offsets in the stream.

using this way, the receiver will know where to look to find information about the stream like where every frame start and end.



LJ92 FPGA Core Software

In normal mode of operation where the core is placed between the memory reader and the USB3.0 module, the only software interface is AXI-Lite interface.

AXI-Lite interface propertise:

- Read and write to SSSS values.
- Read and write the height and width of the frame.
- Read and write the allowed cycle before force ending the frame.
- Read 8 debugging register - only if the core synthesized with the debugging module enable.

The core also implements the AXI4-Stream interface which allow it to be used with any DMA that handle AXI4-Stream components.

In the Github repository there is a software directory that contains code and instructions to describe how to use the core with DMA in ZYNQ. This intended only for testing purposes.

LJ92 Software:

- Uses Xilinx DMA, or open source DMA (soon).
- Write the necessary data via AXI-Lite to test RAW12 images.
- Read and display all the data in registers (height, width, allowed cycles, SSSS values, debug registers)
- Set needed info for DMA and start the DMA.
- Validate the LJ92 result against pre computer LJ92 file.
- Validate the markers.
- Accurately Calculate time consumed in every step.

Please refer to **LJ92-software** to get the software and read the instructions to test the core.

LJ92 Tools and Utilities

LibLJ92 Library:

- Written by Andrew Baldwin and maintained by Magic Lantern community to encode and decode LJ92 files.
- Support decoding of a single scan with single/multiple components.
- Support decoding of all predictor functions.
- Support decoding of 2 to 16 bits color depth and height and width up to 65535.
- Support encoding in predictor function 6 only.
- Support encoding in single scan with single component only.
- Support encoding of 2 to 16 bits color depth and height and width up to 65535.
- Auto generate optimal Huffman tree in encoding.

Raw Image Tools:

- Tools written to manipulate Raw images, Including LJ92 encoding with different options to generate LJ92 files to test the core against.
- Don't support decoding.
- Support encoding in all predictor functions.
- Support encoding in single scan with single/multiple components.
- Support encoding of 2 to 16 bits color depth and height and width up to 65535.
- Huffman tree is inputted manually, not generated.
- The tool support several other things like generating DNG or PNM and displaying analysis of LJ92 images.

Liblj92 github link

Raw image tools github link

References & Useful Links

- Lossless JPEG [wikipedia.org]
- How DNG compresses raw data with lossless JPEG92 [thndl.com]
- CCITT Rec. T.81 (1992 E) - PDF [w3.org]
- aaalgo/ljpeg [github.com]
- ilia3101/MLV-App/liblj92 [github.com]
- FaresMehanna/MLV-File-Format/LJPEG-1992 [github.com]
- FaresMehanna/RAW-Image-Tools [github.com]
- FaresMehanna/JPEG-1992-lossless-encoder-core [github.com]