Project Presentation

Hardware implementation of connected component labelling

Outline

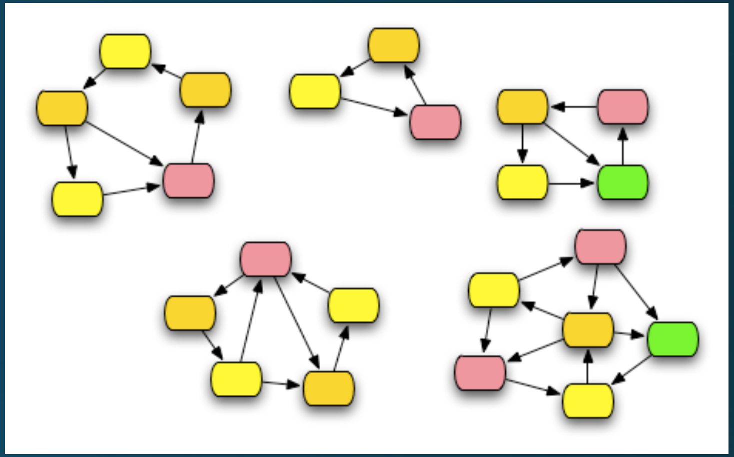
- Overview
- Challenges/Considerations
- Objective description
- Component description

Overview

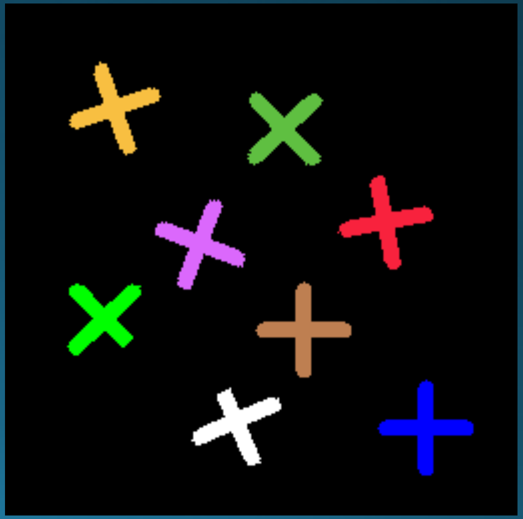
- A connected component labelling algorithm implemented in the FPGA, communicating through a device driver with the kernel to read and write the image pixel information.
- The output is a connected-component labeled image

Connected Component Labeling

- **Connected components.-** a region in a image within which there is an adjacency path between every pair of vertices



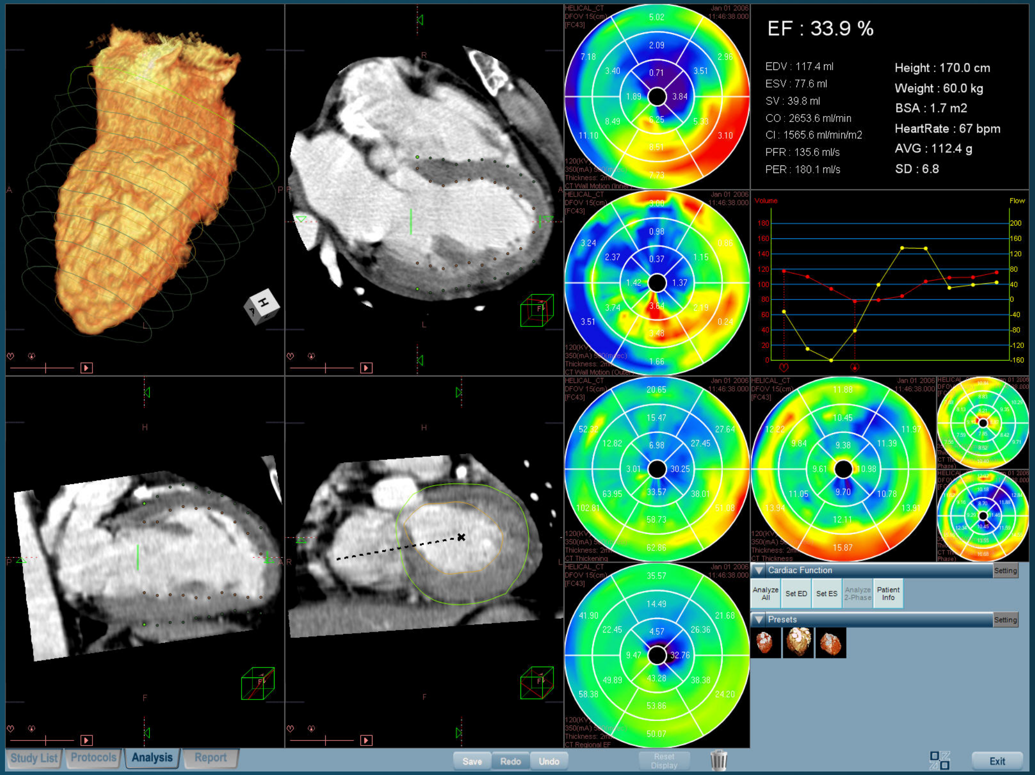
- **Connected-component labeling (CCL).-** algorithmic application to assign labels to connected components based on some heuristic



- In our case the heuristic we used is 6 adjacency which is water tight

Applications

- Includes many automated image analysis applications:
  - OCR
  - Medical image analysis
  - Pathogen detection in image processing
  - Object detection and localization
  - Pattern/Gesture/Facial recognition



EF : 33.9 %

EDV : 117.4 ml
ESV : 77.6 ml
SV : 39.8 ml
CO : 2653.6 ml/min
CI : 1565.6 ml/min/m2
PFR : 135.6 ml/s
PER : 180.1 ml/s

Height : 170.0 cm
Weight : 60.0 kg
BSA : 1.7 m2
HeartRate : 67 bpm
AVG : 112.4 g
SD : 6.8

Technique

- On the first pass:
  - Iterate through each element of the data by column, then by row (Raster Scanning)
  - If the element is not the background
  - Get the neighboring elements of the current element
  - If there are no neighbors, uniquely label the current element and continue
  - Otherwise, find the neighbor with the smallest label and assign it to the current element
  - Store the equivalence between neighboring labels
- On the second pass:
  - Iterate through each element of the data by column, then by row
  - If the element is not the background
  - Relabel the element with the lowest equivalent label

Algorithm

- Neighborhood & label resolution

Top Left	Top
Current Left	Current

o	o
o	L(New)

L(TL)	X
X	L(TL)

o	o
L(CL)	L(CL)

o	L(T)
o	L(T)

o	L(T)
L(CL)	L(C)

Equivalence

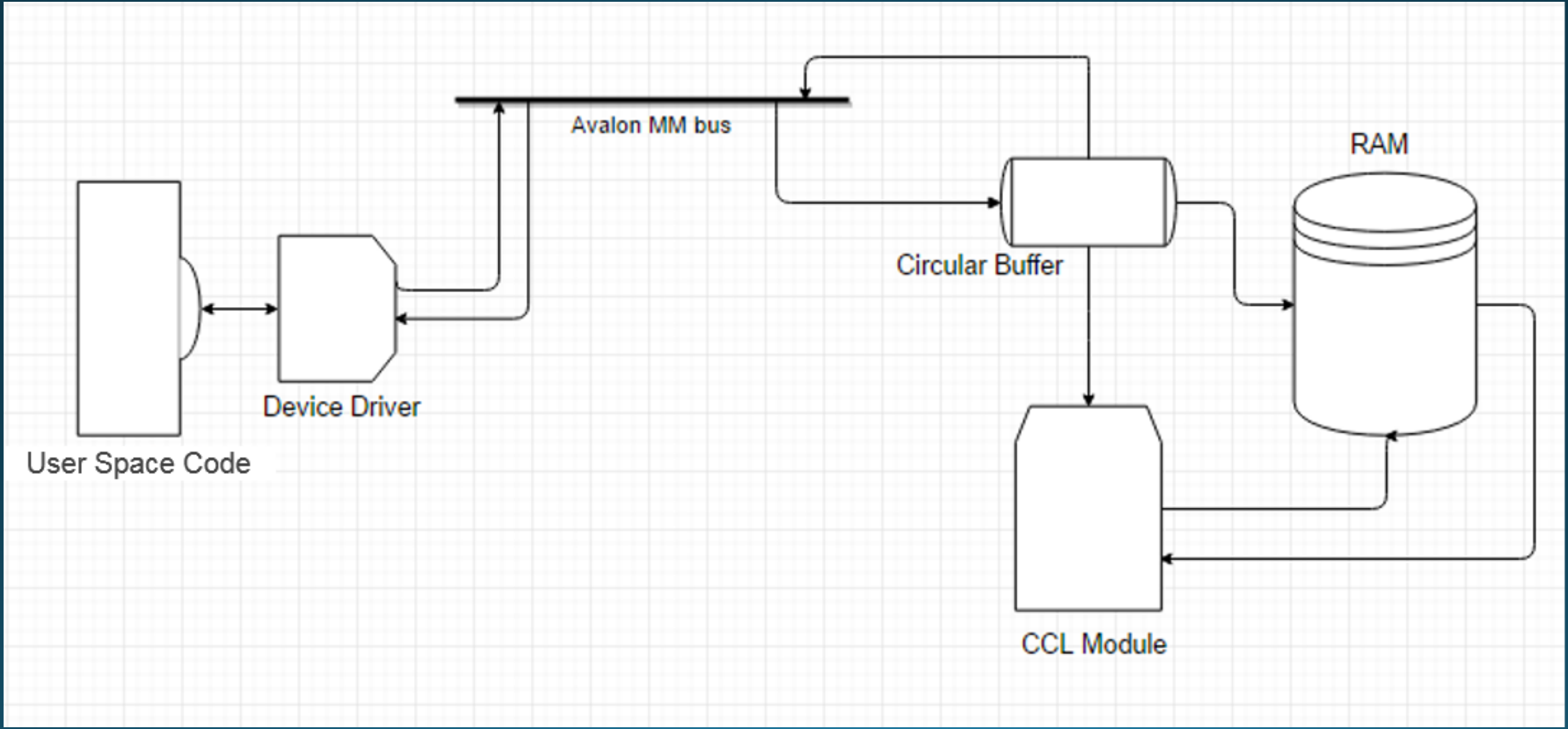
L(T) != L(CL) => LUT[Larger label] = Smaller label

L(C) = Smaller label

Proposed solution

- For the implementation of a CCL module and considering the previously mentioned limitations:
  - Implement a software code that decomposes the input image into pixel row pieces of information
  - Implement a device driver that writes and reads the pixel information to the FPGA through the Avalon MM bus
  - Implement a circular buffer that holds the pixel row information and its top neighbor for a sufficient amount of time to label it
  - Implement a set of modules that threshold the pixels, check neighbors, assign labels and update a label lookup table

Architecture/Block diagram



Main functionality

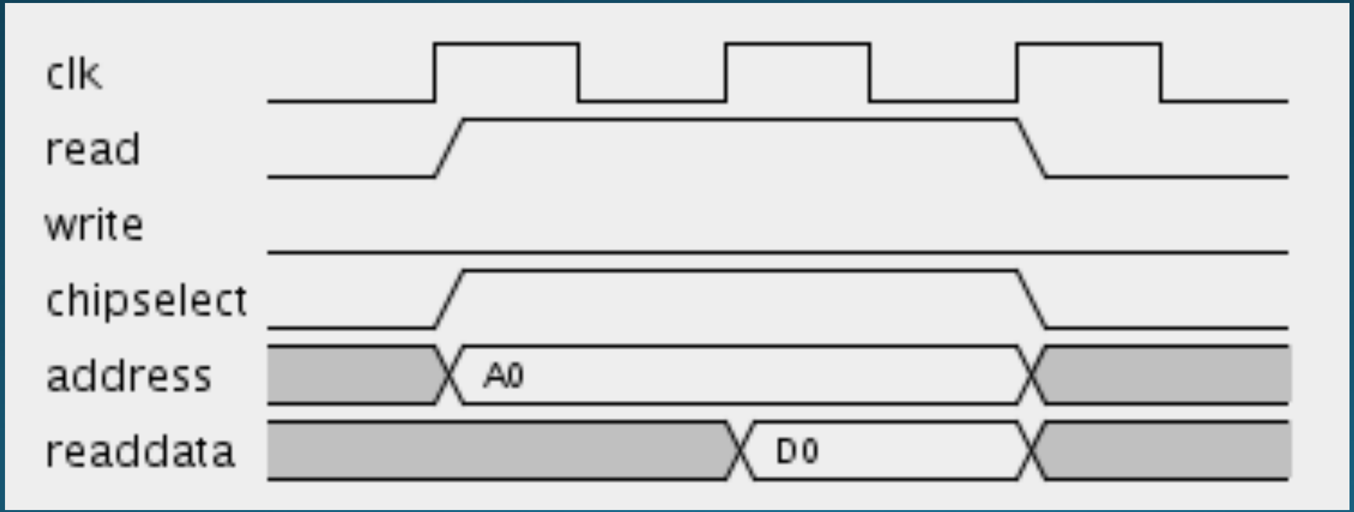
- Pixel information is obtained in the kernel
- The image row pixel information is sent to the FPGA and stored
- The first stage of labeling of the processing is conducted on each row and read back. The procedure is repeated until the full image is processed
- The kernel will initiate a second transfer of the image for label resolving with the pixels being written and read to and from the FPGA

Block/Module

- Kernel C Code
- Device Driver
- Avalon MM bus
- Circular buffer
- RAM
- Connected Component labeler

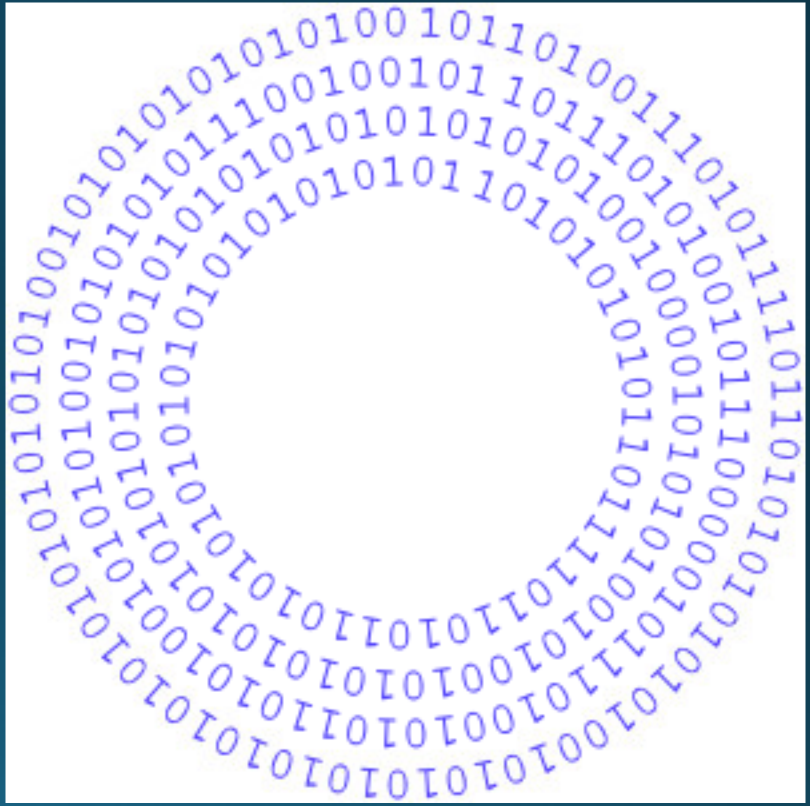
Avalon MM Bus

- Memory mapped interfaces
- Master (Device Driver) – Slave communication
- Implemented in Qsys



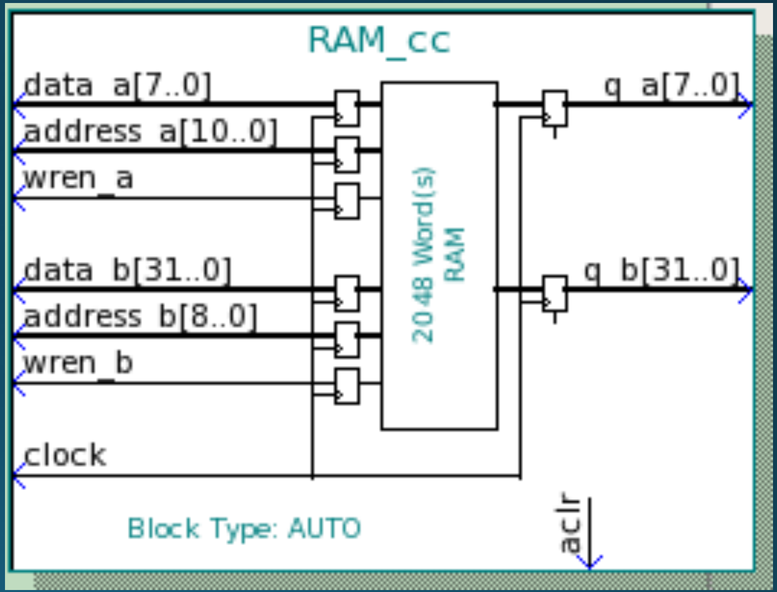
Circular Buffer

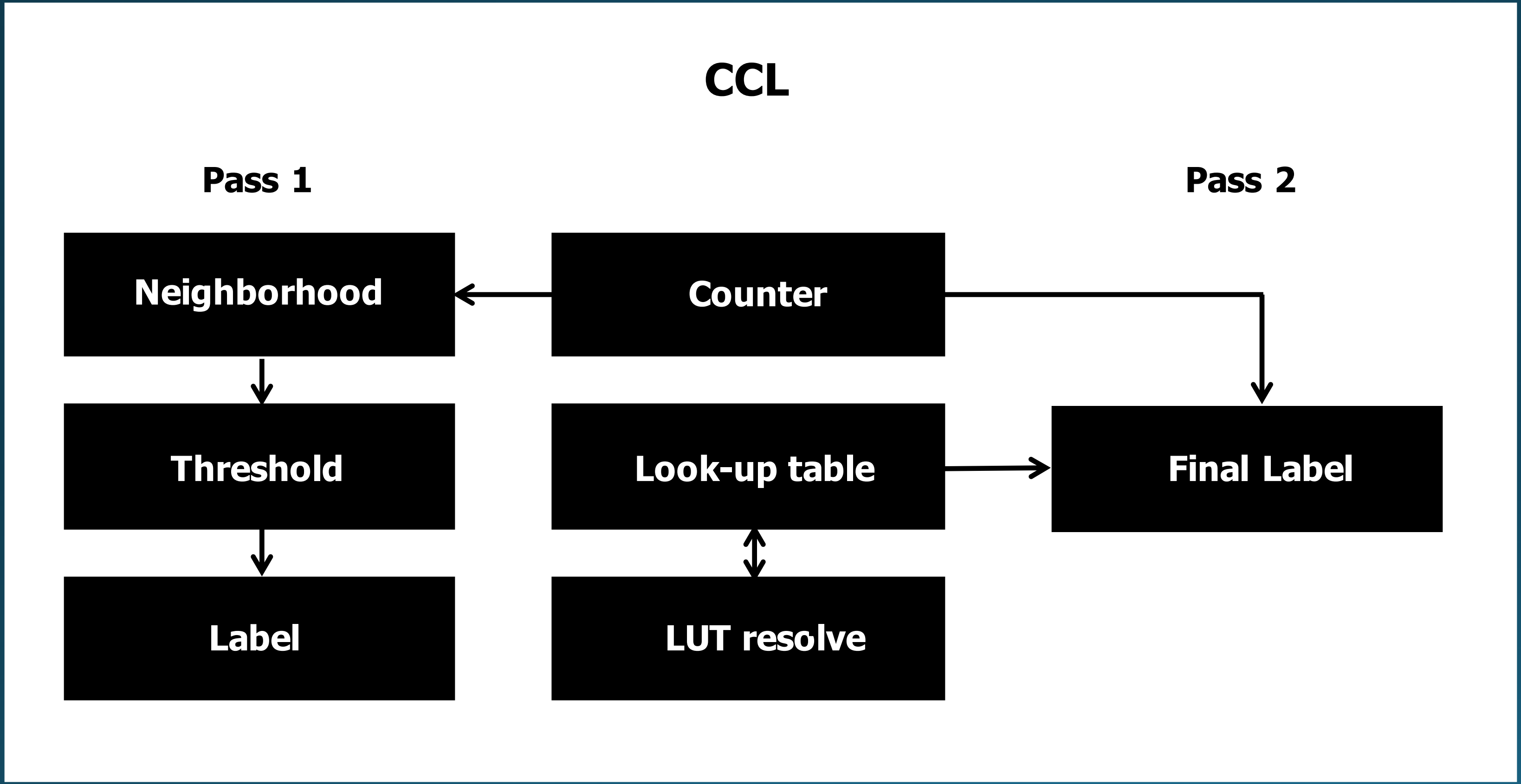
- Accounts for the limitations in FPGA memory size
- Handles pixel information writing and reading for 3 image rows to and from the device driver one row at a time
- Handles signaling for CCL modules: active and top row
- As rows are being processed, the software will be told to read out the labels and clear the circular buffer for new row pixels to be written into



RAM

- Dual-port, single-clock, 2048 byte-size memory locations
- Port A data and output bus is 4 byte wide to handle read/write operations from the device driver
- Port B data and output bus is 1 byte which operates with CCL module




CCL
Pass 1
Pass 2
Neighborhood
Counter
Threshold
Look-up table
Final Label
Label
LUT resolve

Challenges/Considerations

- Limited memory capacity in the FPGA.- can not store a full image in it at once
- Limited Avalon bus width.- only 4 bytes can be sent through the bus at a time
- Architecting and implementing a multi-stage sequential algorithm on a FPGA
- Identifying source of errors: ex. Little Endian format used for RAM, Additional delay introduced in the read cycle because of the use of the dual port RAM

References

- https://www.google.com/imgres?imgurl=http%3A%2F%2Fwww.bigmessowires.com%2Fwp-content%2Fuploads%2F2015%2F08%2Fcircular-buffer2.png&imgrefurl=http%3A%2F%2Fwww.bigmessowires.com%2F2015%2F08%2F&docid=P8io7DtQm3wqMM&tbnid=IlfkUOnm2kT2dM%3A&w=300&h=300&bih=599&biw=1366&ved=0ahUKEwiZg6uVrdTMAhVENj4KHR8hCVY4rAIQMwgqKCcwJw&iact=mrc&uact=8
- http://ps4daily.com/2013/04/playstation-4-developers-didnt-know-about-8-gb-ram-until-reveal/
- http://www.terasic.com.tw/cgi-bin/page/archive.pl?No=83
- http://goodmath.scientopia.org/2007/10/24/making-graph-algorithms-fast-using-strongly-connected-components/
- https://www.google.com.ec/imgres?imgurl=http%3A%2F%2Fhomepages.inf.ed.ac.uk%2Frbf%2FHIPR2%2Fimages%2Fart8lab2.gif&imgrefurl=http%3A%2F%2Fhomepages.inf.ed.ac.uk%2Frbf%2FHIPR2%2Flabel.htm&docid=QBVuw1xM9LuHIM&tbnid=pZLwFlEmwT3EnM%3A&w=256&h=256&bih=599&biw=1366&ved=0ahUKEwiW3Nzt1NTMAhUGbB4KHYFVBkg4ZBAzCAcoBDAE&iact=mrc&uact=8
- http://stackoverflow.com/questions/29427242/difference-between-connected-component-labeling-and-image-segmentation