CS150 Final Project Report, Team 07

Sahar Mesri, Sagar Karandikar, cs150-bw, cs150-bn

December 10, 2013

Abstract

Abstract goes here.

1 Overview

1.1 Design - Block Diagrams

1.2 Brief Description of Major Submodules

1.2.1 Downsampler.v

The Downsampler module converts the original 800x600 image fed into our processing pipeline into a 400x300 image to align with our hardware constraints. The downsampler takes in the image data as a stream of pixels along with a valid signal that is only raised when the input pixel is valid data from the image. In addition to reducing image size, the downsampler is responsible for "generating" the blanking regions (sequences of black pixels) that are used to flush data through our gaussian pipeline. The downsampler ultimately outputs a scaled image as a stream of pixels along with a valid signal and a blankingregion signal that indicate whether the current output pixel is valid and/or a pixel in the blanking region respectively.

1.2.2 Upsampler.v

The Upsampler module converts the processed 400x300 image coming out of the processing pipeline back into a 800x600 image for display over the DVI interface. The upsampler takes in the processed image data as a stream of pixels along with a valid signal that is asserted only for data that is both valid pixel data from the image and not part of the blanking regions. It performs basic upscaling by effectively taking each pixel from the processed input and expanding it to cover a 2x2 box of pixels in the final output. The upsampler outputs a pixel at a time in the output image along with a validout signal that is asserted whenever the data on the output is valid pixel data that we ultimately wish to display (blanking regions are non-existant at this point).

1.2.3 Downsampler4x.v

This module is identical to Downsampler.v, excluding some constants that are modified to allow this version to downsample from 400x300 images to 200x150 images.

1.2.4 Upsampler4x.v

This module is identical to Upsampler.v, excluding some constants that are modified to allow this version to upsample from 200x150 images to 800x600 images.

1.2.5 DownsamplerWrap.v, Downsampler4xWrap.v

This module attaches the Downsampler to its output fifo, which mediates between the Downsampler and the gaussian pipeline. This module spans both the first and second

clock domains.

1.2.6 UpsamplerWrap.v, Upsampler4xWrap.v

This module attaches the Upsampler to its input fifo, which mediates between the gaussian pipeline and the Upsampler. This module spans both the second and third clock domains.

1.2.7 Check4.v, Check4_4x.v

This module connects the entire Downsampler, Octave, and Upsampler pipeline together. Additionally, it provides the switching functionality that allows our design to display any of the gaussians or differences on-demand.

1.2.8 octave.v

The Octave module contains 5 Gaussian filter blocks and computes the four Difference of Gaussian outputs. The module takes in a stream of pixels (one at a time), along with a valid signal and a blanking signal. It will output a stream of pixels (along with a valid signal) for each Gaussian filter and each Difference of Gaussian. There is no blanking output since blanking pixels are considered invalid at the output. This module also has a special output just for the next octave, which contains the data, valid, and blanking signal from one of the Gaussian filters (the fourth one in this case).

1.2.9 five_by_five_window.v

This module represents one Gaussian filter block, with five coefficients in the horizontal direction and five coefficients in the vertical direction. The input is a stream of pixels (one at a time), along with a valid signal and a blanking signal. The output is also a stream of pixels (after going through the Gaussian filter), along with a valid signal and a blanking signal. Pixels that are marked as blanking are also replaced with zero prior to the output.

1.2.10 x_window.v

This module is used to compute the horizontal portion of the Gaussian filter. It will multiply 5 pixels in the same row by their corresponding coefficients. It will then take the average of those results to find the output value, whose location within the frame corresponds to the center of those five pixels. The input is a stream of pixels (one at a time), along with a valid signal and a blanking signal. The output is also a stream of pixels (after computing the horizontal Gaussian), along with a valid signal and a blanking signal.

1.2.11 five_row_array.v

This module is used to store five rows of pixels at a time, which becomes the input into the y_window module. This module takes in a stream of input pixels (one at a time), along with a valid signal and a blanking signal. It outputs five pixels at a time, one from each line, along with a valid signal and a blanking signal. These five pixels have the same horizontal position in the frame, and come from consecutive lines within the frame. The center of these five consecutive lines is the pixel at the center of the Gaussian filter block, so the blanking output corresponds to the blanking input for this particular pixel.

1.2.12 y_window.v

This module computes the vertical portion of the Gaussian filter. It multiplies 5 input pixels (from the same column) by their corresponding coefficients, and averages those results to find the output value. The location of the output value within the frame corresponds to the center of the input pixels. The input is a stream of pixels (five at a time), along with a valid signal and blanking signal. The output is also a stream of pixels (after computing the vertical Gaussian), along with a valid signal and a blanking signal.

2 System Description

2.1 Datapath

2.1.1 Connections to SRAM

2.1.2 Gaussian

2.1.3 Other stuff

2.1.4 Downsampler

2.1.5 Upsampler

2.2 Control

3 Design Metrics

4 Conclusion