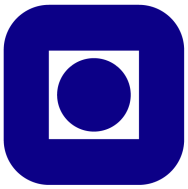

NTNU

TDT4295 - Computer Design Project

Project Report

Convolution team

Peter Aaser
Mattis Spieler Asp
Truls Fossum
Fredrik Haave
Øyvind Kjerland
Arnstein Kleven
Mathias Ose

October 24, 2015

Abstract

Contents

1 | Introduction

2 | Description & Methodology

2.1 Convolution

Some words on convolution. Background-ish

2.2 FPGA

The very heart of our computer is our custom made architecture implemented on our FPGA. In this section we will first describe the overall architecture of our convolution engine, and then we will drill down and examine the core modules. Modularity and extendibility is a core principle in our design, however in this section we will describe the processor as if only being able to do 3x3 convolutions.

2.2.1 Overall design

Convolution is a very regular task where data flows only forward. This means that our processor can be simplified, removing the need for a central control module. Instead each module simply operates under the assumption that all data inputs are correctly formatted and ordered and does its operations accordingly. In the figure we see this design principle

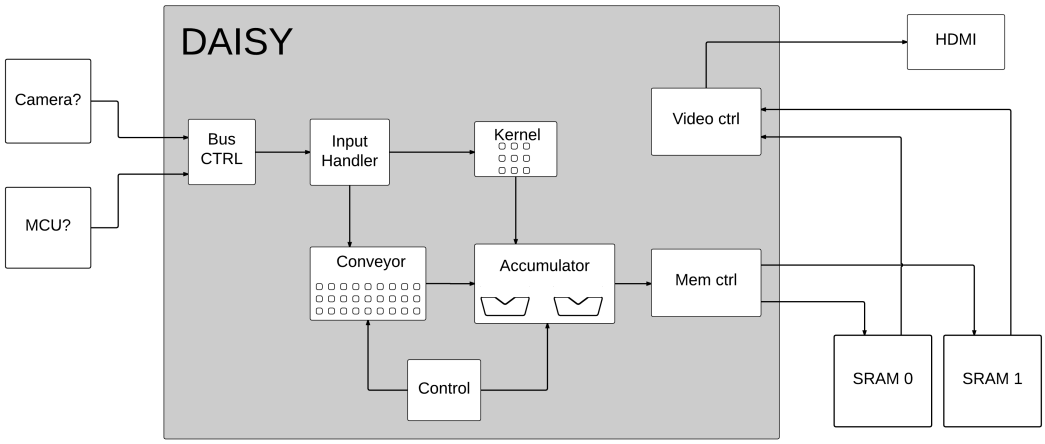

Figure 2.1: The data path for our processor, named daisy after the way it daisy chains control

reflected, no data ever flows backwards, only forwards. Breaking up the individual steps we get the typical flow of data.

1. Data from the bus is read by the bus controller. This controller serves as a clock domain crossover and is responsible for delivering data to the input handler in a clock domain crossing FIFO queue.

2. The input handler recieves data from the queue and does magic

3. After magic is performed the conveyor recieves the data. This unit is responsible for buffering pixel data, and deliver data to the accumulator in a set pattern to ensure a pixel is used in all its 9 contexts.

4. The Kernel buffer collects kernel data at program start from the input handler. After recieving 9 kernel values it will only deliver kernel data to the Accumulator

5. The accumulator recieves kernel data from the kernel buffer and performs its mapping function on the kernel value and the pixel from the conveyor belt. When an accumulator has accumulated all its pixels it flushes its value, resetting the accumulator register and writing the old data to the memory control unit.

6. Accumulated pixels are reassembled in the mem ctrl unit and written to one of the two SRAM banks, allowing us to double buffer.

7. After being buffered in SRAM pixel data is read by the video ctrl module which outputs video to an HDMI cable, ensuring crisp image quality served in a modern fashion.

2.3 Data in

We used an EBI bus! We handled data in an input buffer!

2.4 Convolver

This section encompasses the four modules forming the heart of the convolution engine. In the order they are covered, they are:

1. the conveyor belt

2. the kernel buffer

3. the accumulator.

4. the control unit

In order to not bog down the report with unescessary implementation detail we will cover an idealized implementation leaving the final implementation to the appendix. The two major parts of the convolution engine is the conveyor belt and the accumultion unit. By decoupling these we ensure a modular design allowing us to try a range of different approaches in our accumultion unit.

2.4.1 The conveyor belt

In order to convolute an image we make several sweeps over the image as shown in fig:Sweeps. fig:SweepFrontier shows the frontier of each sweep, highlighting the three rows currently residing in the conveyor. The job of the conveyor belt is to maintain the

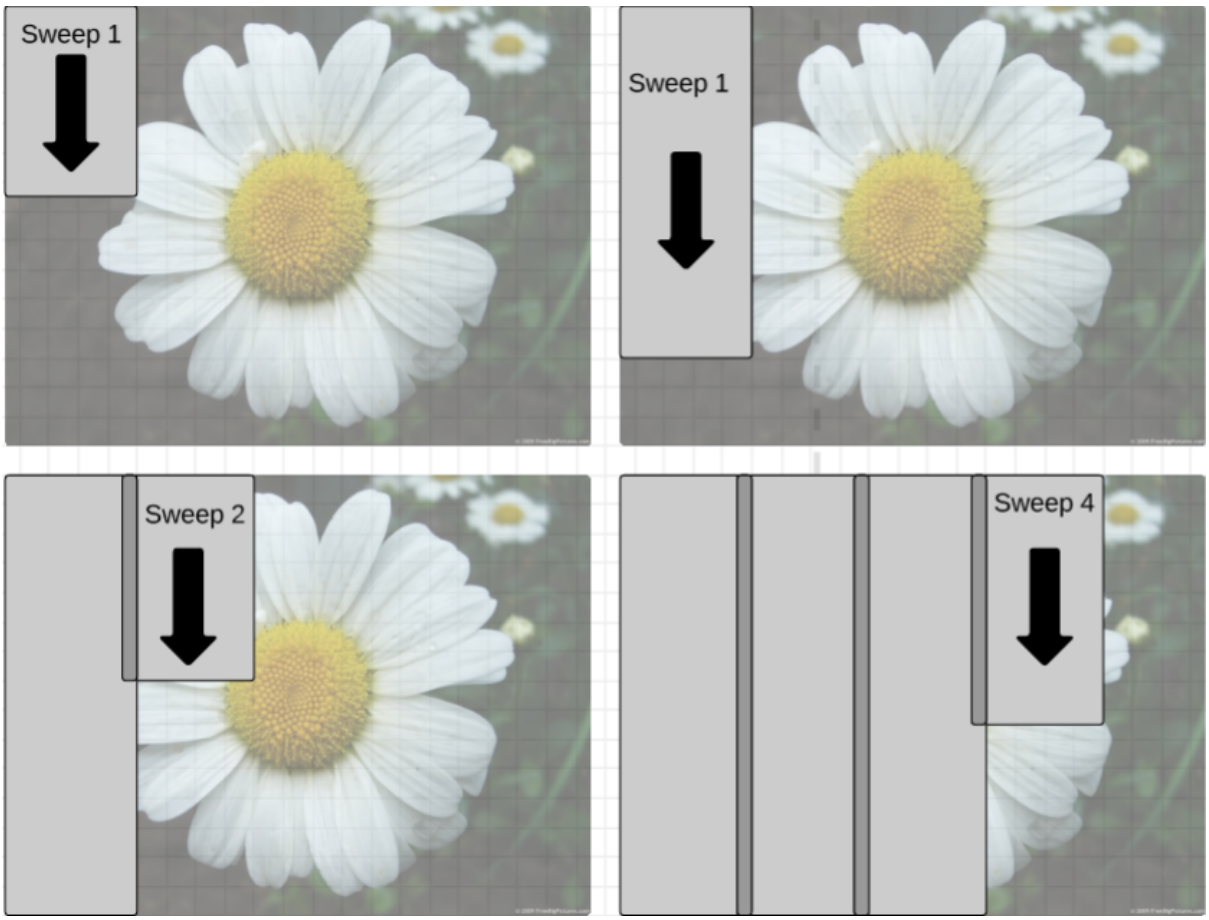

Figure 2.2: The sweep pattern used to collect data for convolution

convolution frontier and to feed the accumulators with data. An ideal representation of the conveyor belt is shown in fig:Ideal

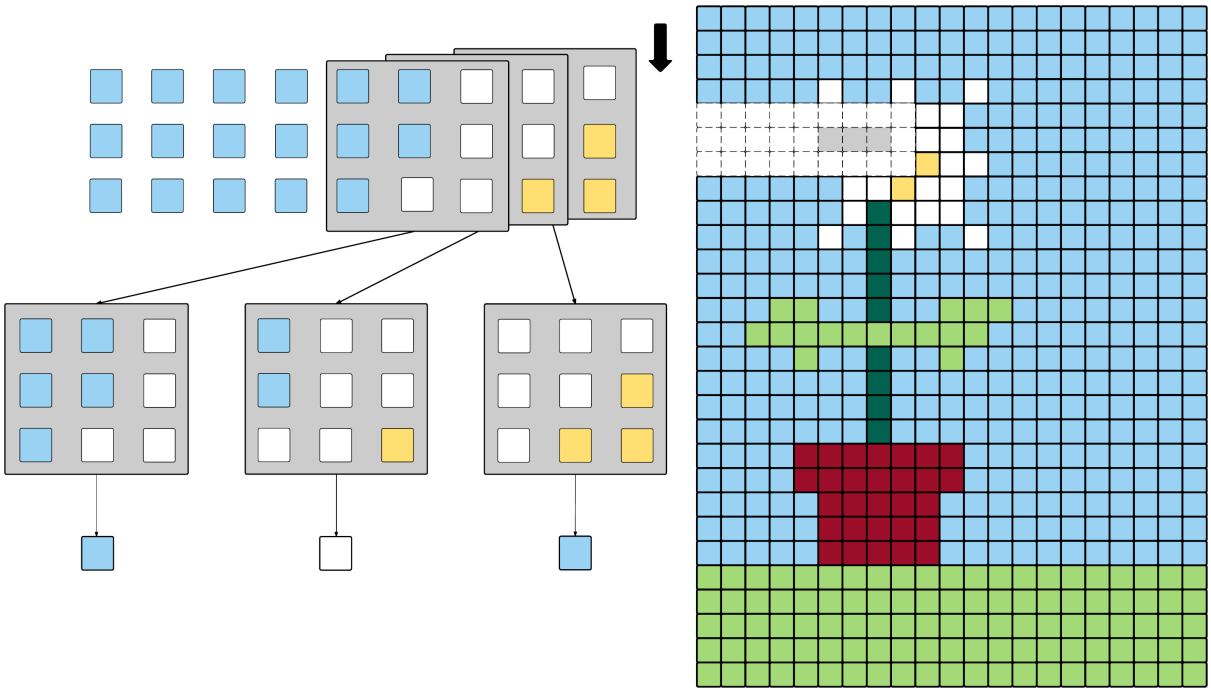

Figure 2.3: The image three greyed out pixels in the source image represent the three pixels which the neighbourhoods in grey boxes help calculate. The window is three pixels deep and nine pixels wide

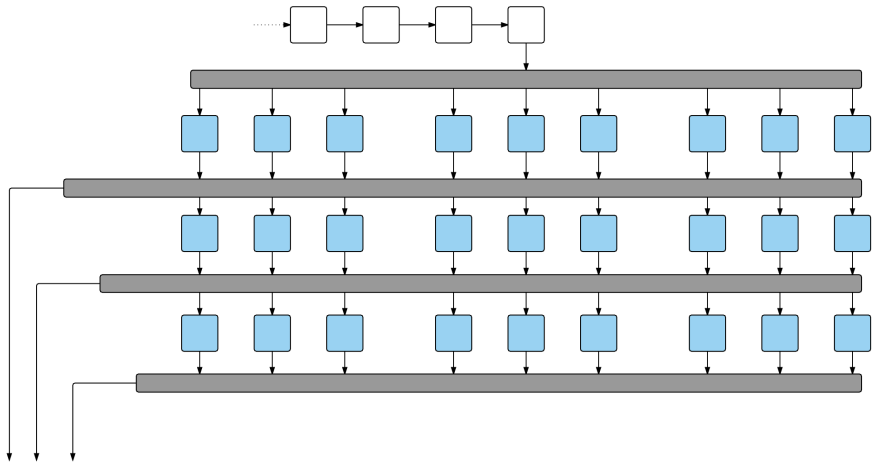

Figure 2.4: TODO

3 Results

3.1 Discussion

4 | Evaluation of assigment

5 | Conclusion

References

[1] Edgar Xavier Ample. Foo. Technical report, Foxford University, 3000.

[2] Mathias Ose. ma.thiaso.se. `http://ma.thiaso.se/`, 2014.