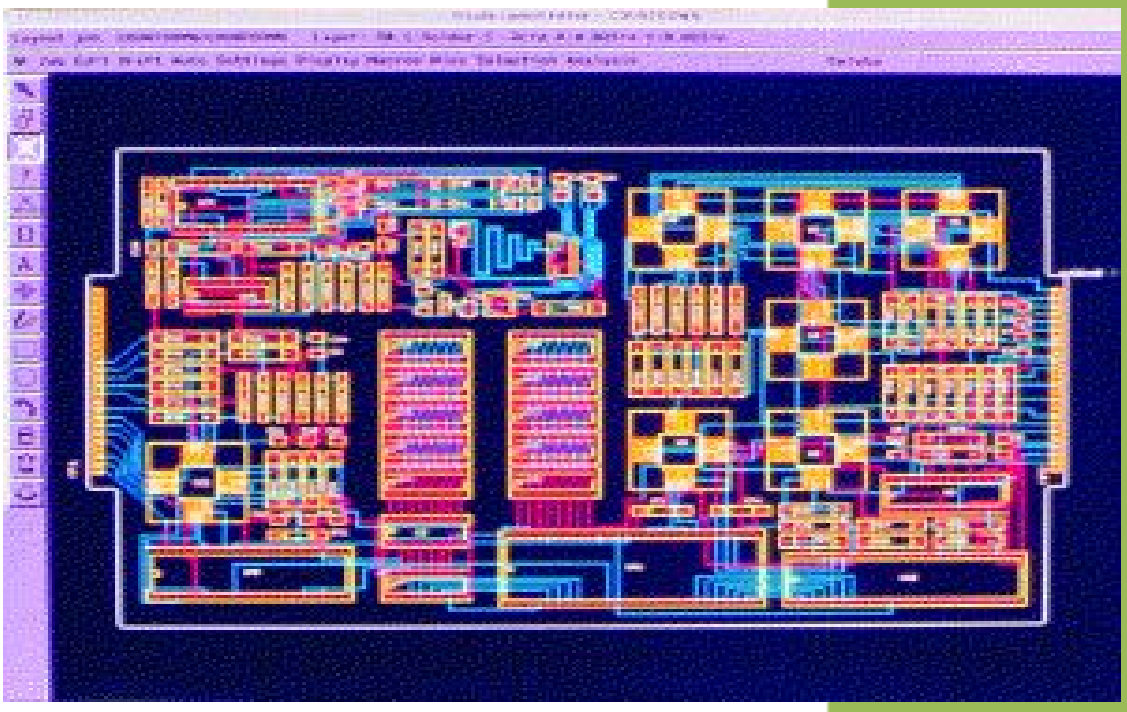
Computer Engineering Department
Faculty of Engineering
Cairo University

Spring 2018
CMP 305
VLSI

DCNN Accelerator

2018



VLSI Semester Term Project
Phase One

VLSI Project

Computer Engineering Department

Objectives

- To better understand Digital Design Flow
- To be proficient at VHDL
- To create real world hardware applications
- To understand the mapping between Algorithm Specifications & Hardware Implementation
- To understand Design Trade-offs
- To learn how to optimize Hardware Designs

Introduction

In real world applications, Convolution neural networks (CNN) achieved a great success in analyzing images. CNNs achieved 99.2% accuracy in classifying the human written digits. It achieved the state of the art in object detection and localizing the objects in an image. CNNs can generate images of human faces. It can colorize a grayscale image. CNNs used in the modern self-driving cars in lane detection, cars classification, and auto steering. Here are some of CNNs results:

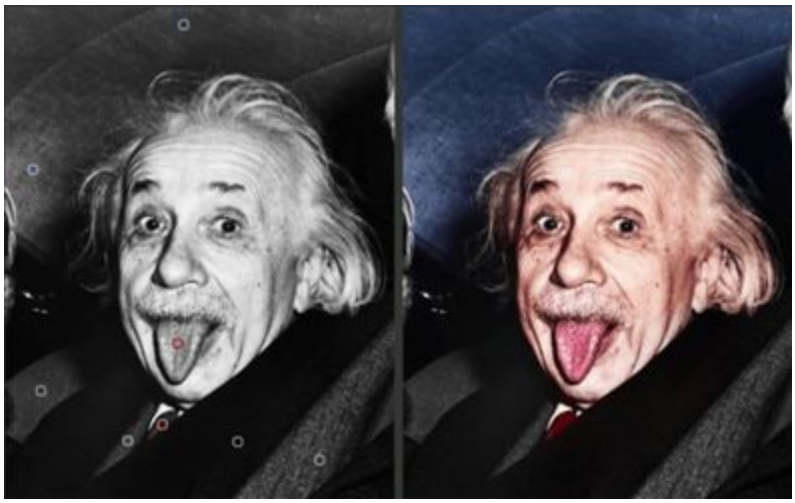


Figure 1-1: Image Colorization

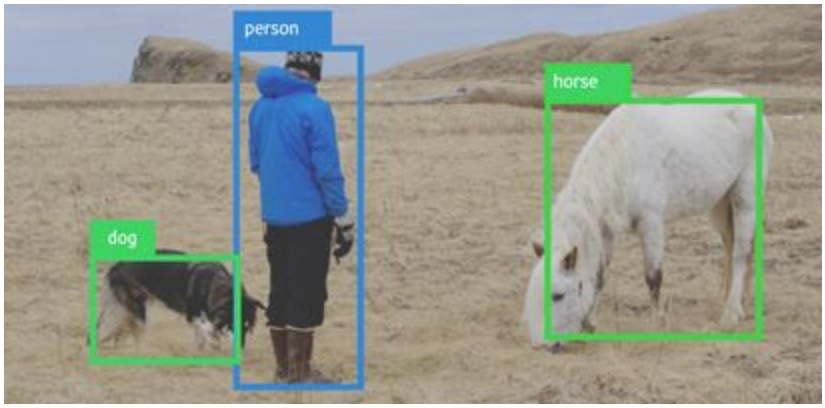


Figure 1-2: Object detection and localization

Figure 1-3: Lane Detection in self-driving cars.

In this project, you will design a detailed level design of a chip that helps the CPU in computing the convolution process, As you will see next section, convolution operation is very very costy for the CPU. So it is required to design the mentioned chip to help the CPU completing the process fast. You will also experience the whole cycle of Design & Fabricating a system on chip, best described as in Figure 1-4.

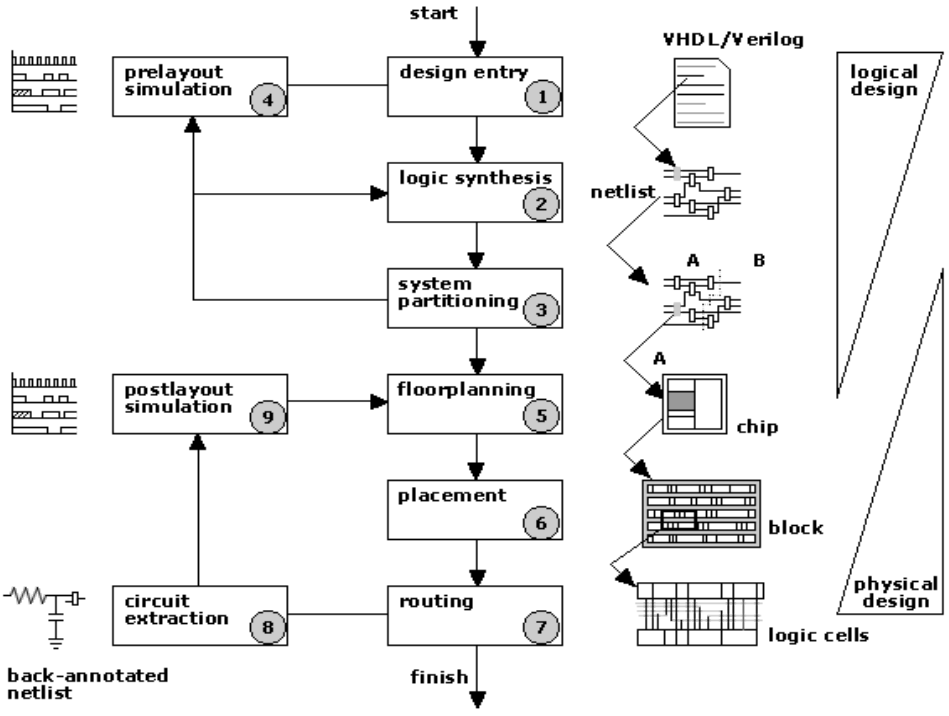


Figure 1-4: ASIC Chip Design Flow

Background

Discrete convolution for images basically takes two inputs, a matrix, and a filter where the size of the filter is less than the size of the input matrix. I.e. image (matrix) may be 256x256 and the filter may be 3x3.

Put the center of the filter on the first pixel of the image. Multiply filter values by the covered area of the matrix element wise. Sum over the result to get one pixel value in the output image. Repeatedly, move the filter and make the same steps over and over until you file the output matrix.

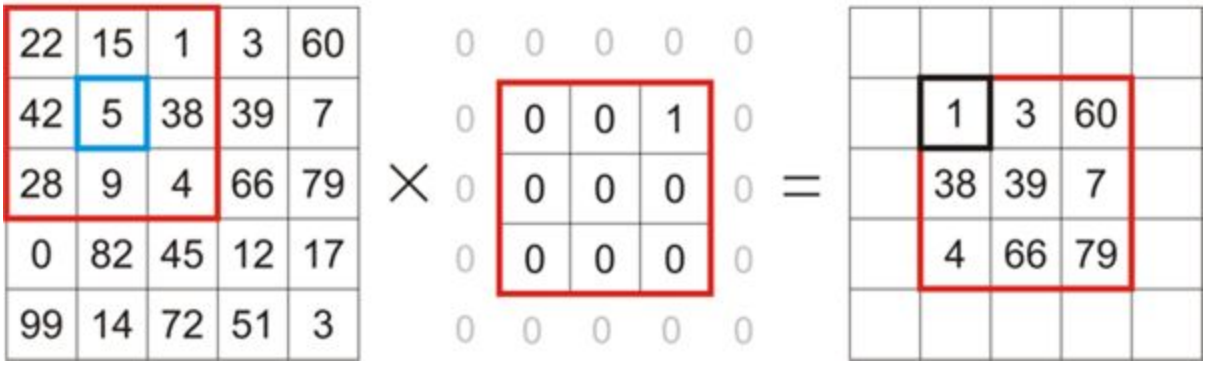


Figure 2-1: one Convolution step

CNNs contains many layers, and every layer contains N number of filters. And instead of human crafted filters like Sobell filter, and Canny. Neural network tries mathematically to make by itself the filter values.

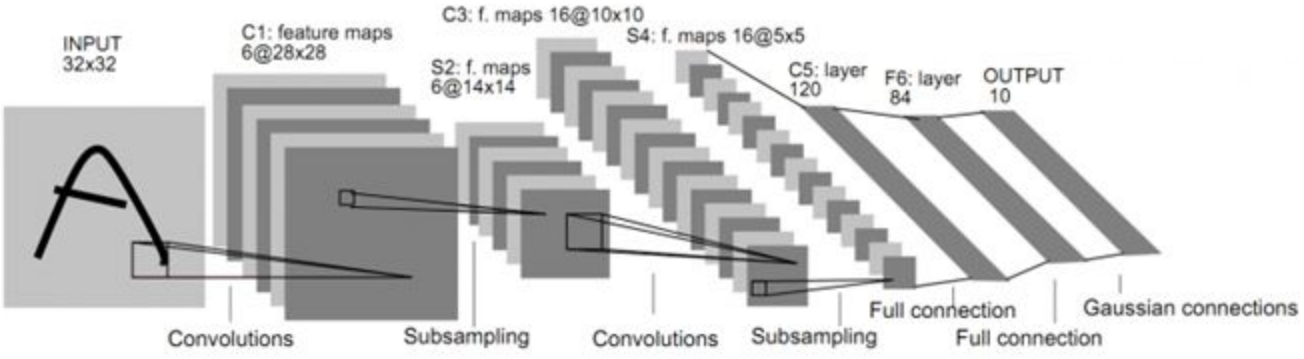


Figure 2-2: Full Convolution Neural Network

Design Requirements

You are required to build a detailed hardware design for a Convolution & Pooling layer algorithm for DCNN Accelerator. The system is built for grey-leveled images (each pixel has range between 0 to 255) as shown in figure 3. The main objective of the project is building the accelerator module.

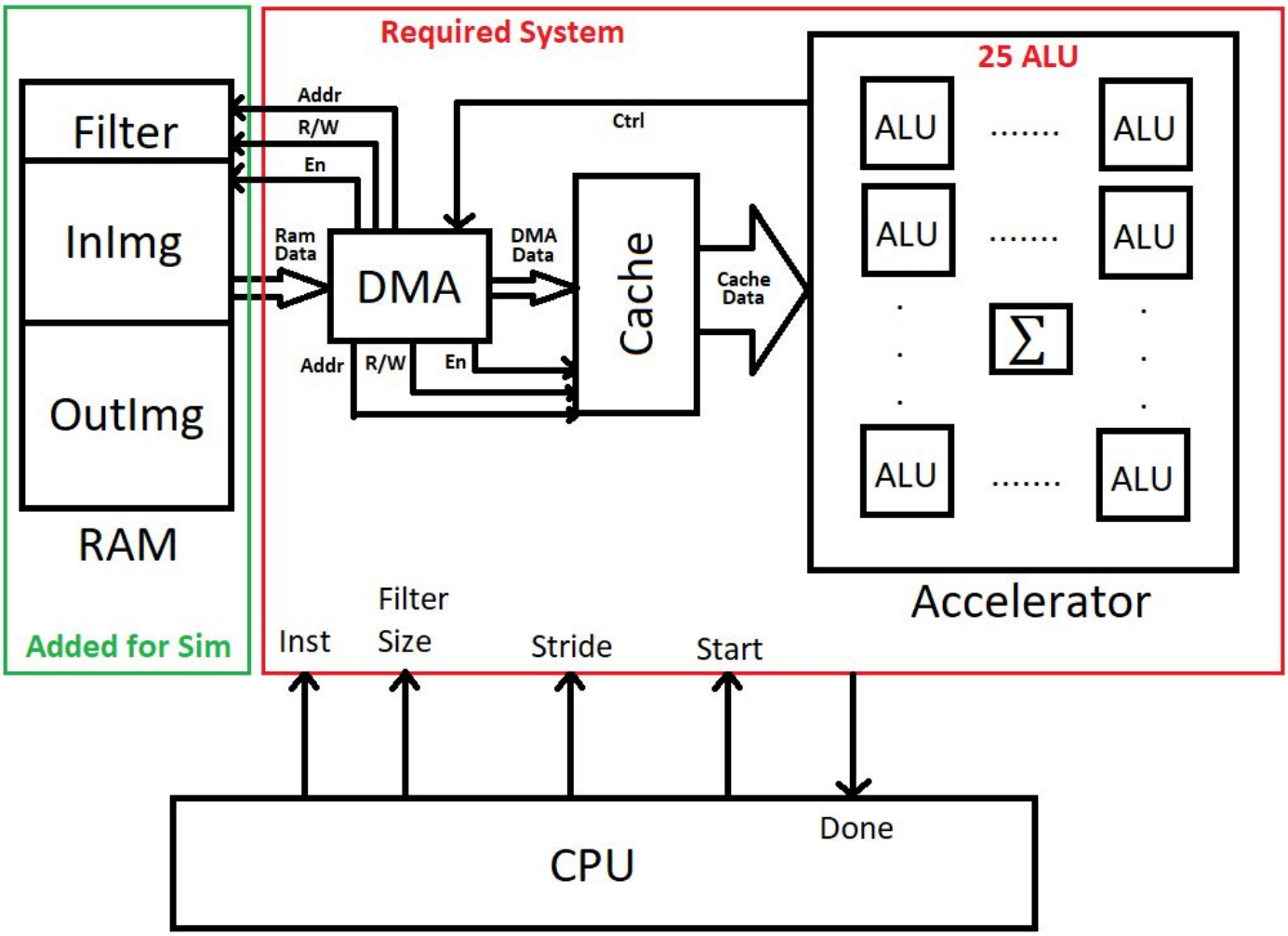


Figure 3: A simplified system design

Take in consideration that your design will be implemented in VHDL and synthesized in the next phases. So try to make the design as clear as possible to easily implement it. Prior optimization is a plus.

Application Scenario

Intel company has assigned your team for designing & implementing DCNN accelerator module to be added into Intel GPUs. The accelerator operates on images of size 256x256 pixels. The CPU loads the required image into a specific part in the GPU RAM, where the image is saved in row-based order (1st row in the image "pixels from (0,0) to (0,255)" are saved first in memory locations from 0 to 255, 2nd row in the image "pixels from (1,0) to (1,255)" are saved in memory locations from 256 to 511, and so on). The image index starts from the top left as pixel number (0,0).
Loading the image into the GPU RAM is out of our responsibility. After the CPU loads the RAM, a "Start" signal is sent to the accelerator to start processing the image.

The processing is done according to 3 options:

1. "Inst" chooses the operation type (convolution or average pooling)
2. "Size" chooses the filter size (3x3 filter or 5x5 filter)
3. "Stride" chooses the filter stride length (move the window by 1 pixel or by 2 pixels each iteration)

For any operation the accelerator sends a command to the DMA to fetch the filter and the image window. After the DMA finishes loading, the accelerator process the window and write the result back in the cache. The accelerator sends a command to the DMA to save the result & fetch the next window, and so on.
Finally, after the accelerator finishes iterating on the whole image, it sends "Done" to the CPU.

Scenario Details

1. CPU sends Start='1' & (Inst, Size, Stride)
2. Accelerator sends to DMA to read "Filter"
3. DMA reads "Filter" from RAM and save it in Cache
4. DMA sends Filter_Ack to Accelerator
5. For each Window in the Image (256x256)
    i. Accelerator sends to DMA to read Win [iter]
    ii. DMA reads window [iter] from "InImg" and save it in Cache
    iii. DMA sends Data_Ack to Accelerator
    iv. Accelerator runs the required operation & save result in Cache
    v. Accelerator sends to DMA to save result
    vi. DMA writes the result in "OutImg"
    vii. DMA sends Result_Ack to Accelerator
6. Accelerator sends Done='1' to CPU

Convolution Algorithm

Convolution operation in DCNN is essentially a dot product between 2 matrices Filter & Window.

1) Sum = 0
2) For each row i
    a) For each col j
        i) Sum = Sum + (Filter[i,j] * Window[i,j])
3) Result = Sum

Pooling Algorithm

1) Sum = 0
2) For each row i
    a) For each col j
        i) Sum = Sum + Window[i,j]
3) If Filter_Size == 3x3
    a) Result = Sum << 3 (Where "<<" is Shift Right)
4) Else If Filter_Size == 5x5
    a) Result = Sum << 5

Radix-2 Booth's Multiplication Algorithm

Binary multiplication uses shifting operations to compute the multiplication of 2 N-bit numbers. It could be implemented in a single-cycle (Shift-Add Approach) or multi-cycle (Booth's algorithm) approaches. **You will use Booth Algorithm in your mini ALUs**.

1) Assuming that the 2 operands are called M1 and M2
2) Set values for A, P, S
   Where A is a (2N+1) bits & the highest N bits are set to be M1 (the rest are "0")
   And S is a (2N+1) bits & the highest N bits are set to be "M1 2's complement" (the rest are "0")
   And P is a (2N+1) bits & the highest N bits are set to be "M2" (the rest are "0")
3) Loop for N iterations:
    a) If the Least Significant (Rightmost) 2 bits == 00 or 11
        i) Do Nothing
    b) If the Least Significant (Rightmost) 2 bits == 01
        i) P = P + A
    c) If the Least Significant (Rightmost) 2 bits == 10
        i) P = P + S
    d) Shift P 1 bit to the right
4) End Loop
5) Result = P[1 to N] (Ignore the least significant bit)

System Specifications

The system should have the following I/O ports:

Port	Direction	Size
Clk	IN	1 bit
Rst	IN	1 bit
Start	IN	1 bit
Inst	IN	1 bit
Size	IN	1 bit
Stride	IN	1 bit
Done	OUT	1 bit
RAM_En	OUT	1 bit
RAM_RW	OUT	1 bit
RAM_Address	OUT	Your choice
RAM_DataIn	IN	40 bits
RAM_DataOut	Out	8 bits

The RAM is an external memory (outside your system) that you will implement for simulation purposes.

The RAM should have the following structure:

Port	Direction	Size
Clk	IN	1 bit
Rst	IN	1 bit
En	IN	1 bit
R/W	IN	1 bit
Address	IN	Your choice
DataIn	IN	8 bits
DataOut	OUT	40 bit

References for more Informations

1. https://en.wikipedia.org/wiki/Booth%27s_multiplication_algorithm
2. https://en.wikipedia.org/wiki/Binary_multiplier
3. http://www.geoffknagge.com/fyp/booth.shtml
4. https://en.wikipedia.org/wiki/Convolutional_neural_network
5. http://cs231n.github.io/convolutional-networks/
6. http://machinelearninguru.com/computer_vision/basics/convolution/image_convolution_1.html
7. http://aishack.in/tutorials/image-convolution-examples/

Rules & Regulations

- Team are 3-4 members.
- You are free to design your system as you want as long as you perform the required functionality.
- You could add any additional I/O ports and modules according to your need, but you have to justify your design choices.
- Your design should be modifiable to meet the design constraints in the next phases.
- Take care that your design is logically mappable to hardware or you will have to repeat it all again.
- Open your mind and don't limit yourself .
- You are not allowed to copy from any external resources in your implementation .
- You are allowed to consult external resources for Design but Do your OWN & you have to fully understand it.
- **Grades are based Mainly on Individual work + your team work . if you didn't work and the project was complete you will still get a zero grade. And we really mean it.**
- The Document is variable to change with a previous notification.

Deliverables

- Hardware Design Document:
  - Detailed Units Architecture with connections.
  - Detailed Sub-units Design in the logic gates Level.
  - Design assumptions and limitation
  - Your Names

Deadline: Sunday 1/4/2018 at Section Time

Discussion: will be scheduled later