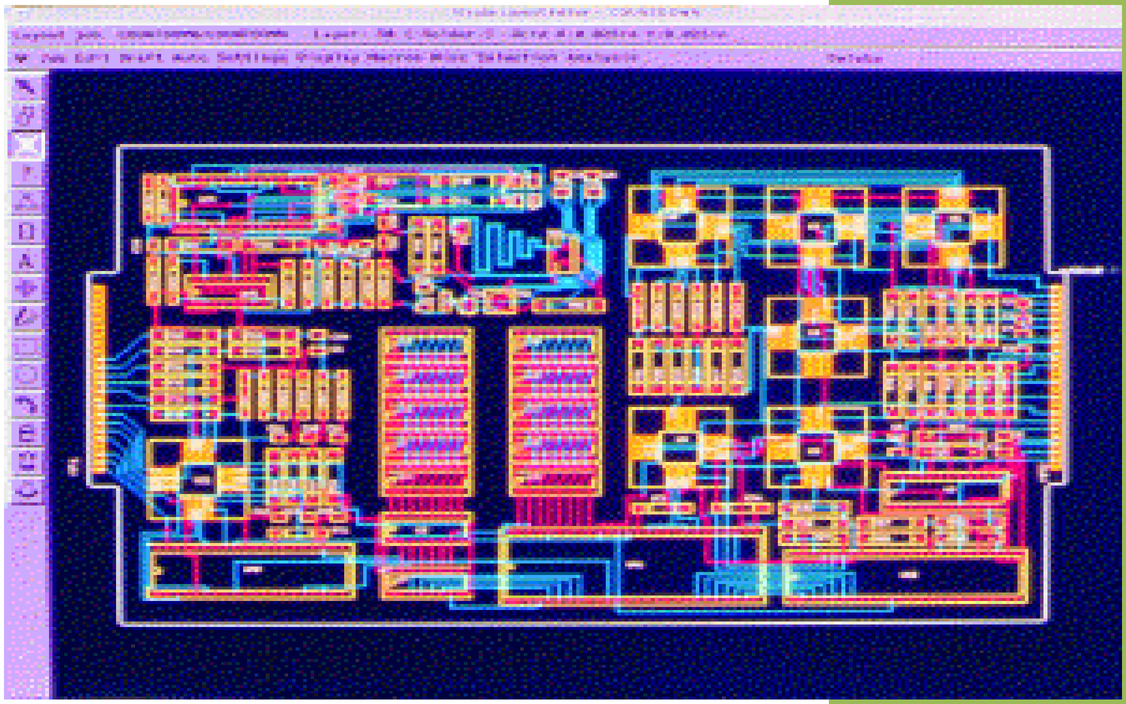
Computer Engineering Department
Faculty of Engineering
Cairo University

Spring 2021
CMP 305
VLSI

DCNN Accelerator

2021



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 Figure 1-2. CNNs can generate images of human faces. It can colorize a grayscale image Figure 1-1. CNNs used in the modern self-driving cars in lane detection, cars classification, and auto steering Figure 1-3. Here are some of CNNs results:

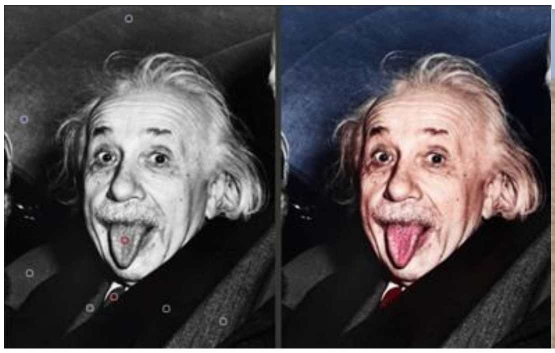


Figure 1-1: Image Colorization

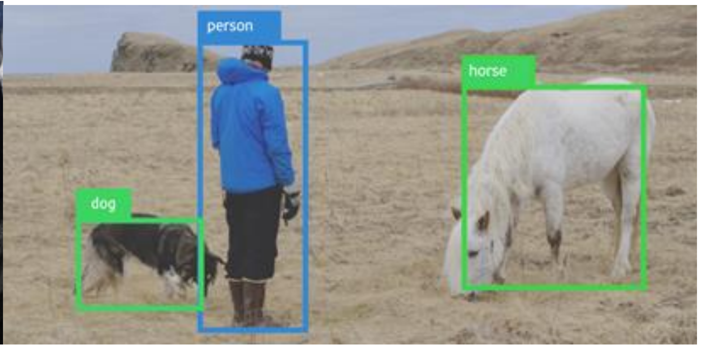


Figure 1-2: Object detection and localization

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

In this project, you will design a detailed low-level design of a chip that applies a CNN classifier over a grayscaled image (MNIST handwritten digits dataset). The chip should be a stand-alone chip that reads the image & CNN layers from user, applies the layers (convolution / pooling) consequently, and generates the output label (0 - 9).

As you will see next section, convolution operation is very very costy. So, it is required to design the mentioned chip to help the CPU completing the process fast, Such chips are called **Accelerators**. You will also experience the whole cycle of Design & Fabricating a system on chip, best described as in Figure 1-4.

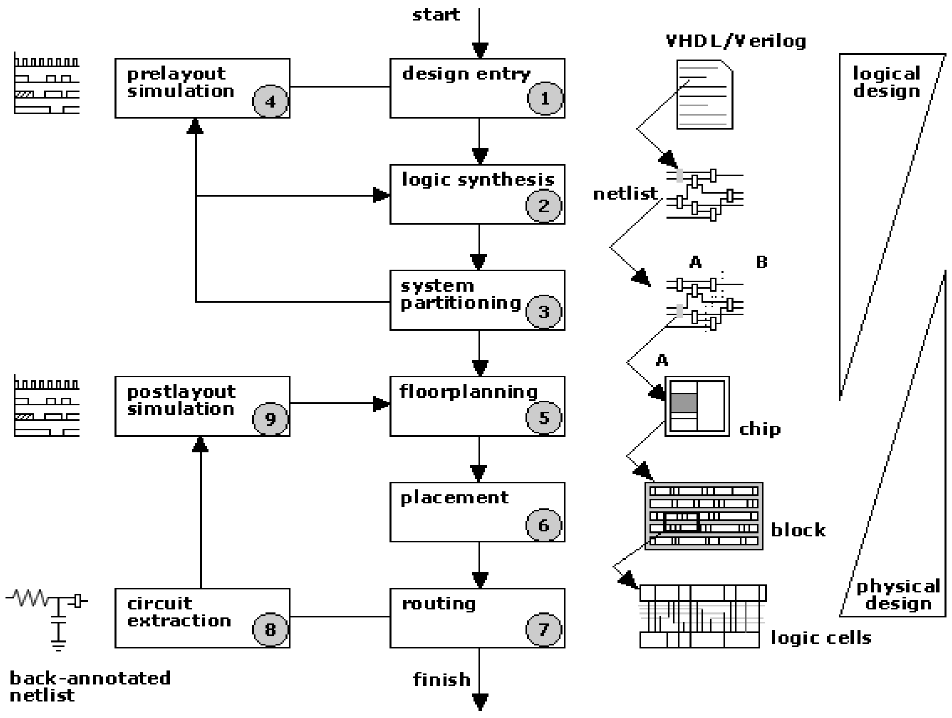


Figure 1-4: ASIC Chip Design Flow

Application Scenario

Apple company has assigned your team for designing & implementing DCNN accelerator to be added to its new IPhoneX12. The accelerator operates on grayscale images of size 28x28 pixels (MNIST dataset images). The CPU loads the required image into a specific part in the RAM.

The process is divided into 3 modules:

1. Loading the image & CNN layers (IO Module)

   Loading the image & CNN info into the accelerator RAM will be done using compressing (SW) and passing the compressed files using a parallel port (16-bit), then your hardware will uncompress the data and save it in its designated locations.

2. Applying Layers one by one (CNN Module)
   Applying the layers will start after the loading step is done (a done signal is passed from module 1 to module 2). The CNN layers are applied consecutively (Layer1 is applied & its result is saved in the RAM, then layer2 is applied & its result is also saved in the RAM, and so on).
3. Applying fully connected layer and out the label (FC Module)
   Finally, the last layer is a fully connected layer that is applied on the last CNN layer to generate the classification label.

The DCNN info includes:
1. Total number of layers
2. CNN Layer 1 info
3. CNN Layer 2 info
4. ….
5. Fully connected layer info

The CNN layer info includes:
1. The layer type (Convolution or Pooling)
2. The convolution window size (3x3 filter or 5x5 filter)
3. The layer depth (the number of filter windows in this layer)
4. The filter window values (if convolution layer)

The fully connected layer info includes:
1. The NN weight values

Advanced Modules

Below are some advanced ideas that you can use in design to enhance time/power and maybe area. Don't try to do all of them, they are totally optional:

- Customize your Hardware to work on compressed format instead of decompressing the data.
- You can start the CNN module while loading data, but make sure the processing worth it and the output will be correct.
- You will have several images as inputs, it will be better if you load the weights only once beware that your ram size might be limited keep place for the intermediate results/output as well.
- You might design the CNN & FC using the same hardware.
- You might use pipelining concept to process faster.
- You might use different adder/Multiplier algorithms with justification.

Design Requirements

Design Phase:

You are required to build a detailed hardware design for a DCNN Accelerator forward process (classifier/inference only). The system is built for grey-leveled images (each pixel has range between 0 to 255) as shown in figure 2.

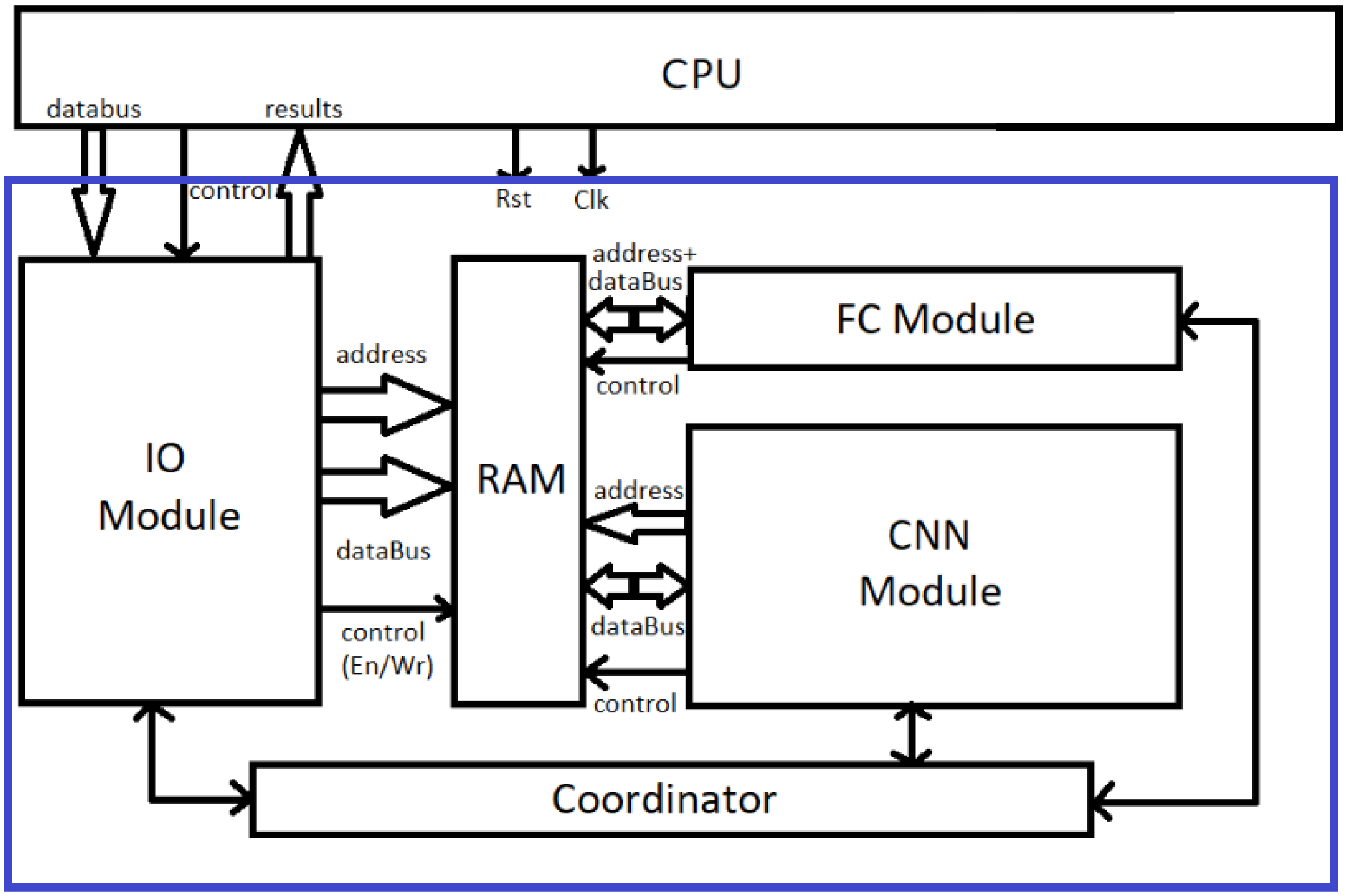


Figure 2: A simplified system design

Take in consideration that your design will be implemented in Verilog 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.

IO Module Description

The image & CNN info will be compressed before sending them to the DCNN accelerator (this part is a software script that applies the compression algorithm on the input). Then the compressed files are sent over a parallel port (16bit) to the DCNN accelerator. The received data will be decompressed by your hardware and saved on a local RAM.

More Details Here on IO Module Design Requirements.

CNN Module Description

The CNN layers are read and applied one by one in the CNN module. For each layer the module loops over the image using sliding 2D windows and computing the filter result for the middle pixels. The result of each layer is saved in the local RAM to be processed by the next layer.

More Details Here on CNN Module Design Requirements.

FC Module Description

The last CNN layer result is passed to a fully connected neural network that generates the probability for each label. Finally a softmax layer is used to choose the classifier prediction.

More Details Here on FC Module Design Requirements.

Synthesis and PnR Phase:

Each sub-team will implement & synthesis its module. You will walk through the ASIC Design flow to generate final working Netlist file for your design delivered in phase 1. RTL implementation and validation are the 1st and 2nd steps in ASIC design flow as shown in Figure 1. You will use Mentor tools as you practiced in the labs to complete the ASIC Design flow.

- Check that your top level design has the pins mentioned in “Detailed Interfacing” section.
- Implement your Design using VHDL.
- Simulate your Design using Do File.
- Synthesize your Design using Oaysis-RTL on **NCSU_FreePDK_45nm** Technology.
- Verify the synthesized netlist generated using your Do File.
- Place & Route your netlist using Nitro-SoC.
- Generate GDSII file & verify your design for DRC & LVS.

Tips

- Use Source Control Software such as git to avoid messing up at integration time.
- Always unit test your code to make sure it is working before integration. And add comments as much as you can.
- Define interfaces between modules clearly to make your world much easier.
- Make A **Good detailed design,** it is the key to the working project, do it with extra care.

Rules & Regulations

- Team will be 9-12 members which will be divided into 3 subteams.
- 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

Design Phase:

- Each subteam has its design deliverables defined in each subteam document.
- Overall Hardware design documents:
  - Your names
  - Overall system design and interfacing signals between the modules.
  - Design assumptions and limitation
  - Overall system scenario (Steps, FSM, …)
- Testbed Scripts
- Modules' (Entity) VHDL codes.
- Verification DO Files + IO files.

Synthesis Phase:

- Synthesis Tcl Script
- Synthesis Design Constraint file
- Synthesis Netlist file
- Synthesis Reports (Timing / Area)
- Project Documentation:
    - Project File Tree
    - Each Module (Entity) Description
    - I/O Description
    - Timing Diagrams for Initialization / Processing / Results
    - Verification Results (Behavioral / Netlist)
    - Design Flow Reports

Deadline: Design Phase by 23rd May 2021 11:59 PM at the blackboard submission link.