Cairo University - Faculty of Engineering
Computer Engineering Department
VLSI

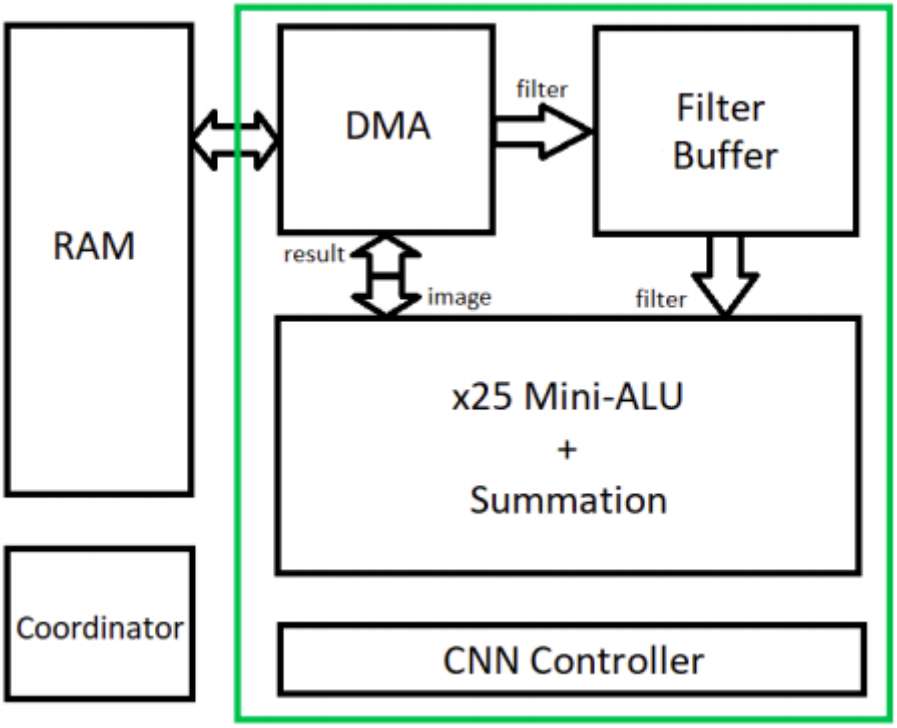
DCNN Accelerator
CNN Layer
Phase 1

Name	Section	BN
Ahmed Mostafa	1	7
Seif Eldin Elsaeed	1	27
François Adham	2	9
Naiera Magdy	2	30

Submitted to:

Eng/ **Abd El Rahman Abo Taleb**

Academic Year:
2020 – 2021



1- Modules:

1.1- DMA:

DMA has 4 modes of operation:

Code	Operation
00	Read window 5x5
01	Write value “16 bits”
10	Read number of filters with size 5x5
11	Read bias numbers

- Reading filters:
  - Takes the number of filters from CNN module
  - Read filters one by one
  - Save filters in Filter-Buffer module

- Reading Bias:
  - Read bias weights from RAM
  - Save the weights in Filter-Buffer module

1.2- Filter-Buffer:

Has 2 buffers the first one is the filter buffer (1920x5x5) each element is 16 bit of width, the other one is for bias weights has 120 elements each of them is 16 bits width.

1.3- CNN:

Contains the algorithm of the layer and controlling events, Convolution module as well as the Pooling module.

1.4- Convolution:

Contains 25 mini-ALU that performs the convolution process using Radix-2 Booth's Multiplication Algorithm and returns the result to CNN.

1.5- Pooling:

Takes a window of the image, performs pooling over the first 4 elements and returns the result to CNN to be saved in the RAM.

1.6- RAM:

- Reading from RAM:
  - Takes address and offset.
  - Reads the first 5 elements from address, the second 5 elements from address + offset.
  - Offset is equal to image size to get a window and equal to 5 in case of filter.
  - Returns the result (5x5) to Filter Buffer in case of filter and to CNN module in case of Window.
- Writing to RAM:
  - Takes the address and the value to be written.
  - Write one word per clock.

2- Assumption and Limitations:

- **Assumptions:**
  - Filters, biases then the image are stored in the RAM in sequence with fixed locations (until integration with I/O module).
  - We have 6 layers, 3 convolution, 2 pooling

We have synthesized our design with reduced RAM size and filter buffer size to be able to finish the synthesis.

- **Limitations:**
  - Image is **32x32**
  - Write **one** word only in the ram at a time
  - Fixed point number is **5 bits** signed integer and **11 bits** mantissa.

3- Testbench:

- We tried a simple test case with all filter weights = 2^{-6}(represented as 0020 hexadecimal), bias weights = 0 and image weights = 1 (represented as 0800 as hexadecimal). All saved in the mem file attached with the file.
- Final output of the CNN are 120 elements all = 5.7109375 (represented as 2DB0 as hexadecimal).
- We provided a python script that simulates this sequence of inputs with one filter.
- To test the CNN you have to write the data of RAM in the LoadMem.mem file all represented as fixed point numbers with radix hexadecimal with the following addresses:
  - All Filters: 0 → 50549 (size: 5x5x6 + 5x5x96 + 5x5x1920)
  - All Biases: 50550 → 50691 (size:6 + 16 + 120)
  - Input Image: 50691 → 51715 (size: 32x32x1)
- You will find the output of each layer at the following addresses:
  - Layer 0 conv: 51716 → 56419 (size: 28x28x6)
  - Layer 1 pooling: 56420 → 57595 (size: 14x14x6)
  - Layer 2 conv: 57596 → 59195 (size: 10x10x16)
  - Layer 3 pooling: 59196 → 59595 (size: 5x5x16)
  - Layer 4 conv: 59596 → 59715 (size: 1x1x120)