CAD FOR VLSI

PROJECT 2:

TOPIC: SYSTOLIC ARRAY DESIGN

Problem Statement:

Build a '4x4' systolic array module using the MAC developed in assignment 1. After building the systolic array, perform Matrix Multiplication of two 4x4 matrices, containing random values, for both the data types (int8 and bfloat16).

Architecture:

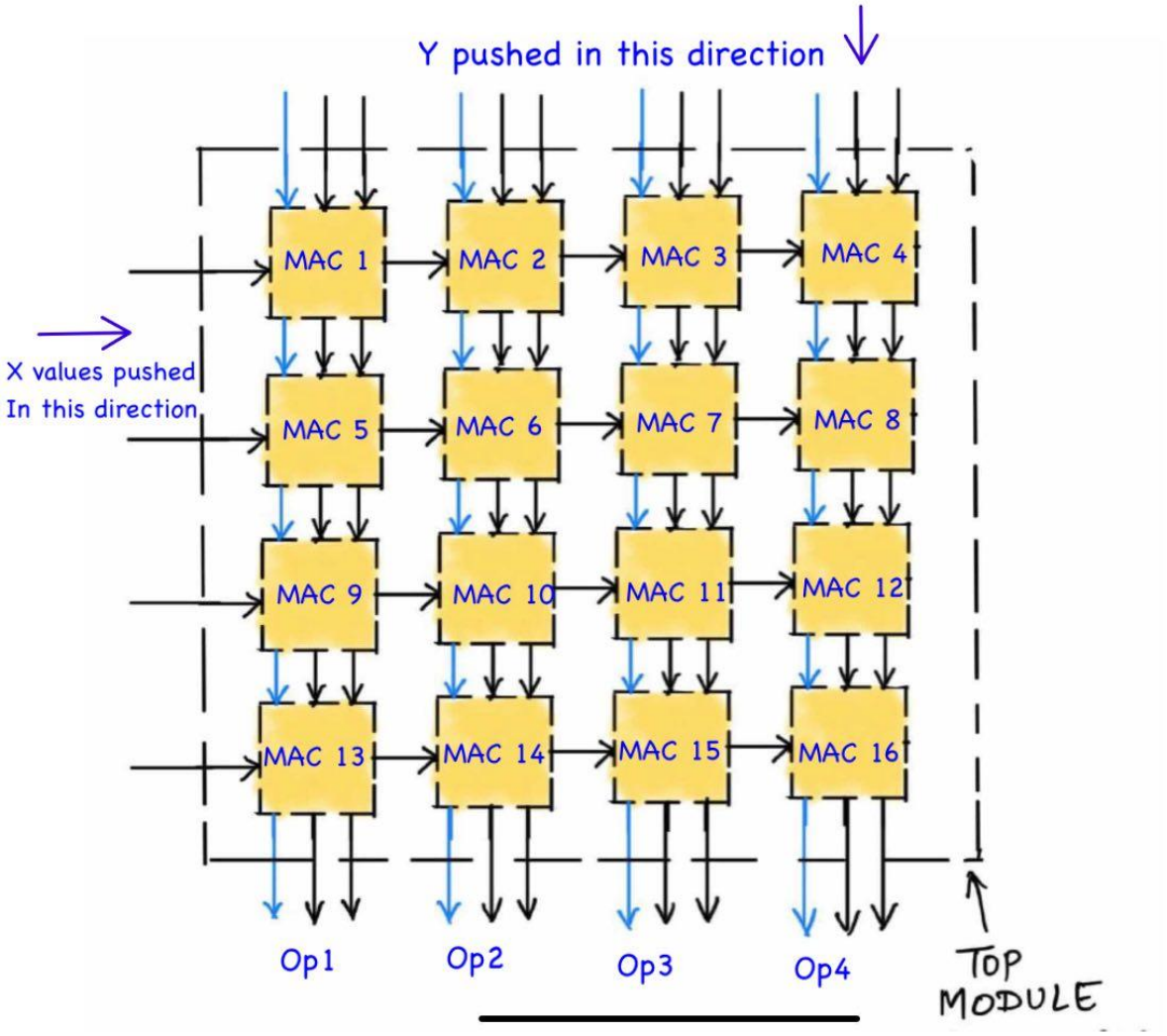
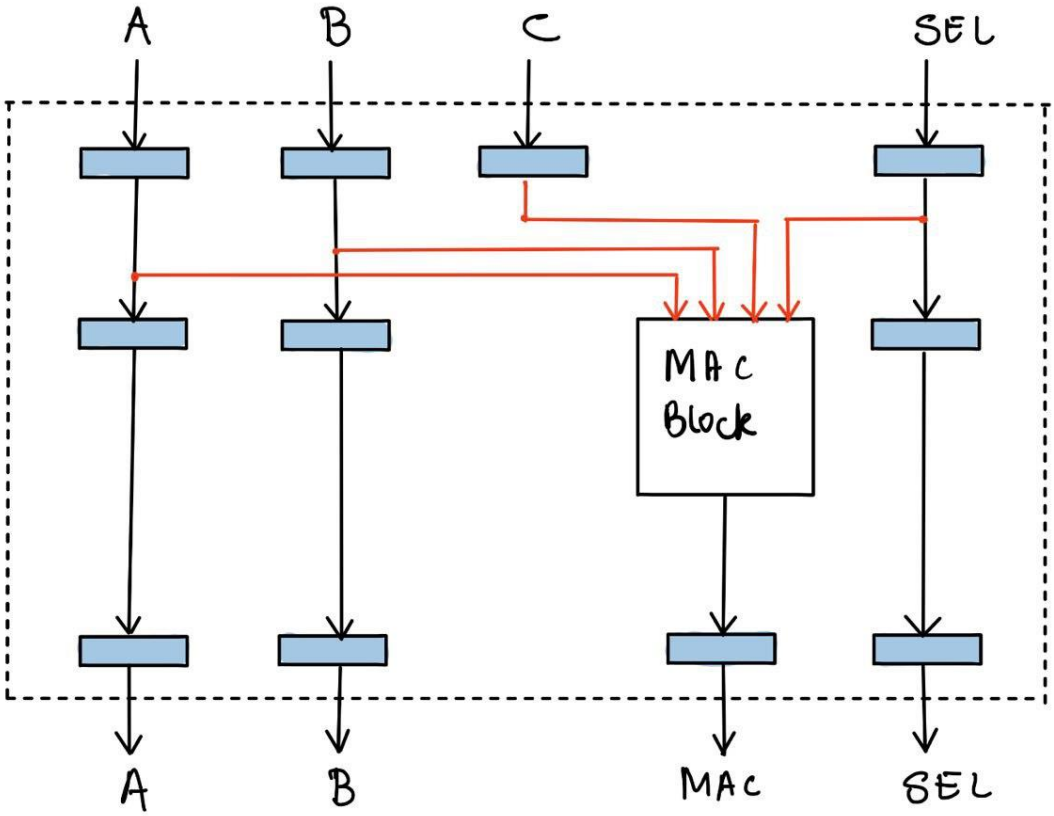


Figure 1 Systolic Array Architecture

We changed our MAC module a little bit for getting out the input values to output ports to feed the same to next MAC module according to the architecture.



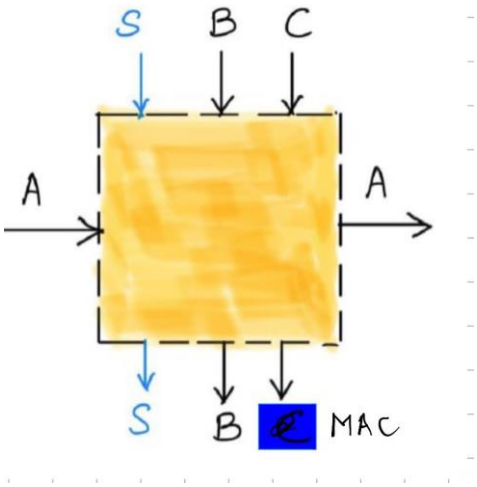


Figure 2 Building block of the Systolic Array

Our building block takes 3 cycles to gives correct MAC output and passing A,B and SEL value.

Testing of module:

We performed a matrix multiplication operation using this systolic array.

Two (4X4) matrices X and Y are generated randomly. Based on the value of mode in the python code, if mode=1, matrices with integer values are populated and if mode=0, matrices with bfloat16 values are populated.

Then matrix multiplication was performed in the following manner:

- We used cocotb framework to test the module.
- As a first step, Y values were pushed into the array, as mentioned in the project document, starting from last row to first row. Each subsequent row is passed after an interval of 3 clock cycles as it is the latency of our basic building block. Y values are pushed through B ports of the array.

Sequence in which values are pushed	**Column1**	**Column2**	**Column3**	**Column4**
1	Y[3][0]	Y[3][1]	Y[3][2]	Y[3][3]
2	Y[2][0]	Y[2][1]	Y[2][2]	Y[2][3]
3	Y[1][0]	Y[1][1]	Y[1][2]	Y[1][3]
4	Y[0][0]	Y[0][1]	Y[0][2]	Y[0][3]

- After Y values have been pushed, X values are pushed again after every 3 clock cycles in the following manner:

Time Step	**Row 1**	**Row 2**	**Row 3**	**Row 4**
1	X[0][0]	0	0	0
2	X[1][0]	X[0][1]	0	0
3	X[2][0]	X[1][1]	X[0][2]	0
4	X[3][0]	X[2][1]	X[1][2]	X[0][3]
5	0	X[3][1]	X[2][2]	X[1][3]
6	0	0	X[3][2]	X[2][3]
7	0	0	0	X[3][3]

- Output come out in the following fashion from the output ports in the following fashion:

Time Step	**Op1**	**Op2**	**Op3**	**Op4**
4	OUT[0][0]	0	0	0
5	OUT[1][0]	OUT[0][1]	0	0
6	OUT [2][0]	OUT [1][1]	OUT [0][2]	0
7	OUT [3][0]	OUT [2][1]	OUT [1][2]	OUT [0][3]
8	0	OUT [3][1]	OUT [2][2]	OUT [1][3]
9	0	0	OUT [3][2]	OUT [2][3]
10	0	0	0	OUT [3][3]

We also created a python model named as Systolic_Array_model in which we pass both X and Y matrices and mode i.e. whether it has to run in integer mode or bfloat16 mode.

Mode=1 means integer mode and Mode=0 means Bfloat16 mode

Matrix multiplication is done with the help of python script and both our module results and python script results are displayed on the terminal to compare them.