International Institute of Information Technology
Bangalore

5x5 Systolic Array Implementation for 3x3 Matrix Multiplication

Ayyappa Koppuravuri (IMT2020555)
Ayyappa.Koppuravuri@iiitb.ac.in

August 5, 2022

Aim

Write a program for 3x3 Matrices multiplication using 5x5 systolic array model in Verilog to integers, IEEE-754 Single precision floating point numbers, and IEEE-754 Half precision floating point numbers. Also Synthesis the code, implementation, generating bit streams using the software Vivado, and programming it on an FPGA (Basys-3 board/Zed Board).

Objectives Accomplished

- Written a code in verilog for matrix multiplication using 5x5 systolic array model and was able to test it successfully in all the three data types mentioned in above objective.**The code is expected to be generalised in a way that it is applicable to all the data types which are similar to IEEE-754 numbers but varying with N,e,m, and $N <= 32$ where N is total no of bits, e is total no of exponent bits, and m is no of mantissa bits.**
- Synthesis the code in Vivado and confirmed output in Post-Synthesis simulation for all the three data types.
- Runned Implementation in Vivado and tested the output using Post-Implementation simulation for all the three data types.
- Able to generate Bit streams in Vivado for all the data types and Programmed it on FPGA boards either Basys-3/ Zed board according to resource constrains. But, The output observed in ILA is not as expected.

Objectives remaining to be accomplised/Future Objectives

- Programming on FPGA Boards again for all the three data types and debug it.
- Finding a way to decrease the no of clock cycles for accessing Block Rams. Because, The total no of clock cycles used for accessing inputs are around 18 and the total no of clock cycles taken for systolic array for multiplication is around 9.
- Testing it for Bfloat-16 data type also.
- Linking all the Codes of different data types to a single project.
- Using this for its applications like Convolution.

Code

link for the code - Systolic Array

Model of the Systolic array implemented

PE(Processing element)

The Processing Element in this particular systolic array has three inputs **c**, **b**, and **c** which has outputs **a'** = **a**, **b'** = **b**, and **c'** = **c** + **a** * **b**. The operation **c** + **a** * **b** can be considered as the common operation programmed in these processing elements. The outputs **a'**, and **b'** are the outcome of the property of the **PE** acting as a hardware Register holding data for a clock cycle. These operations are briefly described using a block diagram in **Fig. 1**.

There is no much change in the implementation of PE for other data types as two functions namely fadd, and fmul are created in the PE's of floating point numbers for their addition, and mutiplication.

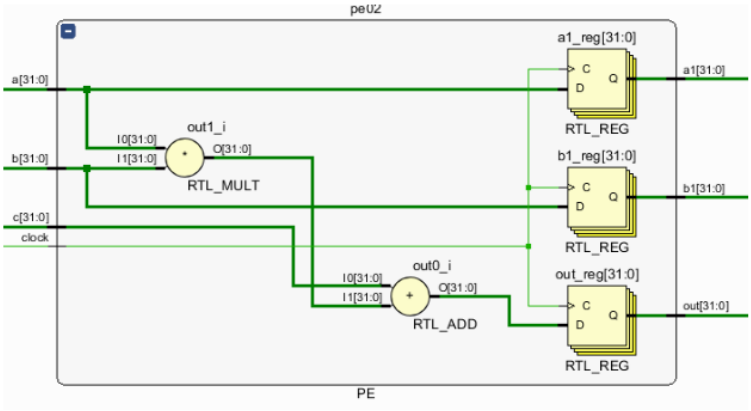

Figure 1: Internal Structure of the PE used in the Systolic Array for integer data type

0.1 Delay block

A Delay block is just a group of flipflops for holding data for a clock cycle which is triggered to all the Delay blocks and Processing Elements at a time. These are used to make the Systolic Array symmetrical. In this case the Systolic Array is **5x5** structured. The inputs to the delay block are **a**, and **b**. The outputs of the delay block are $a' = a$, and $b' = b$. The schematic of delay block is shown in **Fig. 2**

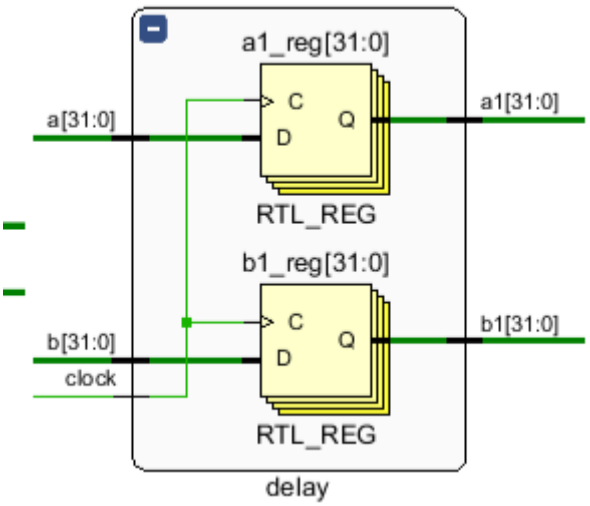

Figure 2: Schematic of Delay Block used in the Systolic Array

Systolic Array

The systolic array used for matrix multiplication of 3x3 matrices have inputs $A_{3x3}, and B_{3x3}$.

The whole process process takes **8** clock cycles of duration, In which each row of the Matrix A is passed to a00 , a10, and a20 and the first column of Matrix B is passed to b00 , b01, and b02. The second row of the Matrix A is passed to a10 , a20, and a30, and the second column of the Matrix B is passed to b01 , b02 , and b03 respectively. Similarly, The third of row of the Matrix A is passed to a20 , a30 , and a40, and the third column of the Matrix B is passed to b02 , b03 and b04 respectively in the first consecutive clock cycles.Other values remains as zero. This is given in **Fig. 3**

The output of the Matrices Multiplication namely D is collected at c35, c45, c55, c35 , and c45 of the systolic array. D00, D01, D02, D10, and D20 are collected at c55, c45, c35, c54, and c53 indices of the systolic array at 6^{th} clock cycle of the process.Similarly, The indices D12, D11, D21 of the output matrix are collected at the indices c45, c55, and c54 of the systolic array at 7^{th} clock cycle of the process. Lastly, the index D22 of the

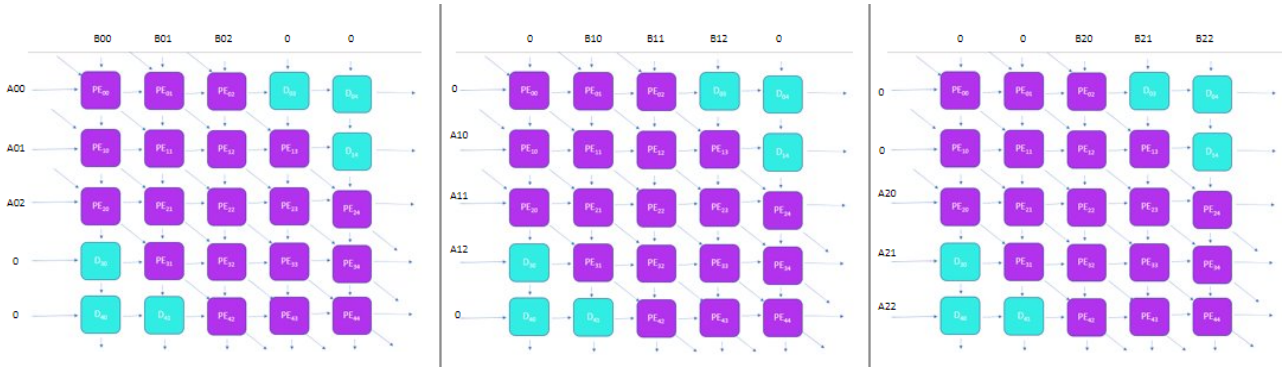

Figure 3: Description of Inputs in Systolic array

Matrix D is collected at the index c55 of the systolic array at 8^{th} clock cycle of the process.

The above explanation is for the case, If initially all the Zeros are not passed through the systolic array.

The Structure of the model we are using for Matrix multiplication is given in **Fig. 4** along with appropriate indexing used for above description.

$$\text{where A} = \begin{bmatrix} A00 & A01 & A02 \\ A10 & A11 & A12 \\ A20 & A21 & A22 \end{bmatrix} \text{,B} = \begin{bmatrix} B00 & B01 & B02 \\ B10 & B11 & B12 \\ B20 & B21 & B22 \end{bmatrix} \text{, and D} = \begin{bmatrix} D00 & D01 & D02 \\ D10 & D11 & D12 \\ D20 & D21 & D22 \end{bmatrix}$$

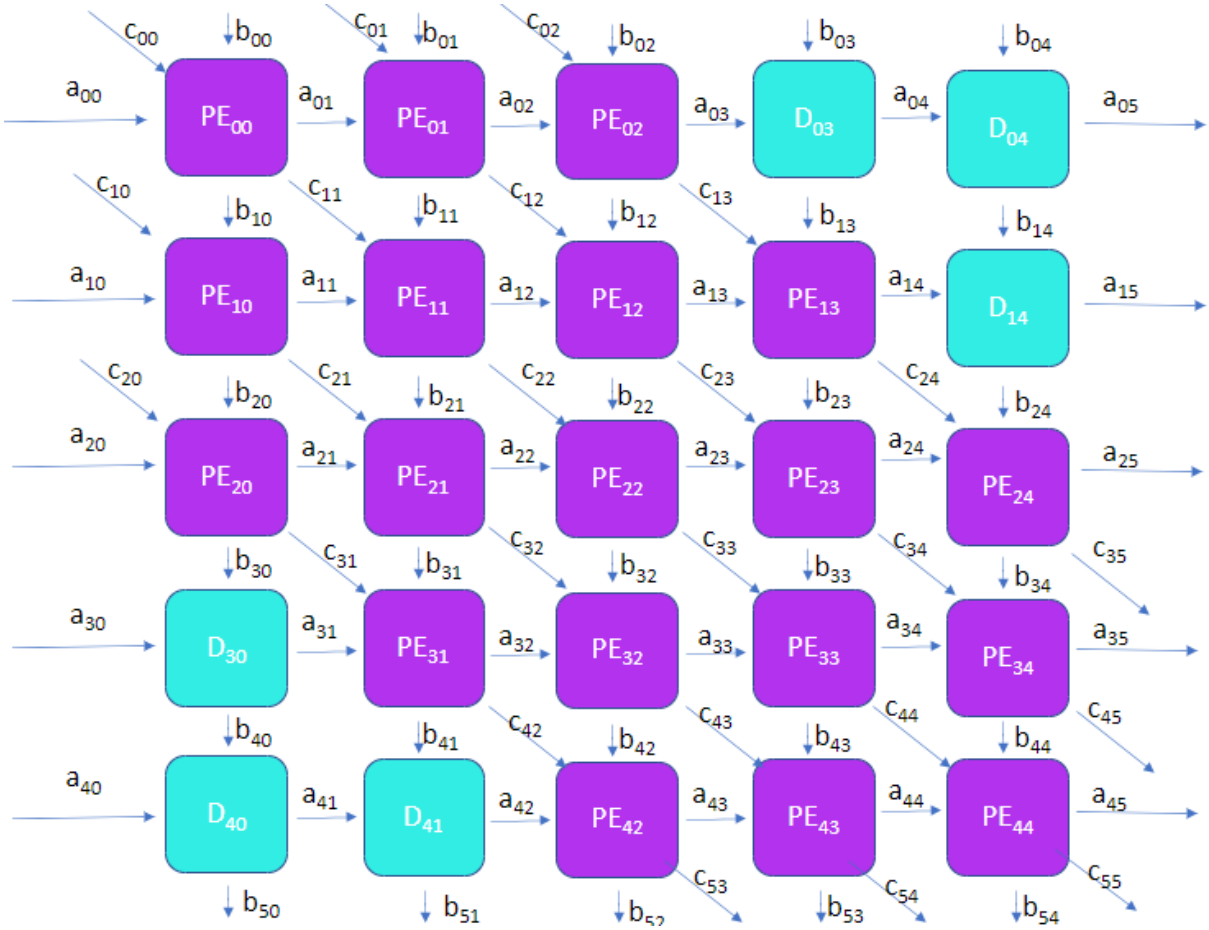

Figure 4: Schematic of Systolic Array

Results

Integers

For the Matrices A = $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$,B = $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$

The output Matrix is : D = $\begin{bmatrix} 30 & 36 & 42 \\ 66 & 81 & 96 \\ 102 & 126 & 150 \end{bmatrix}$

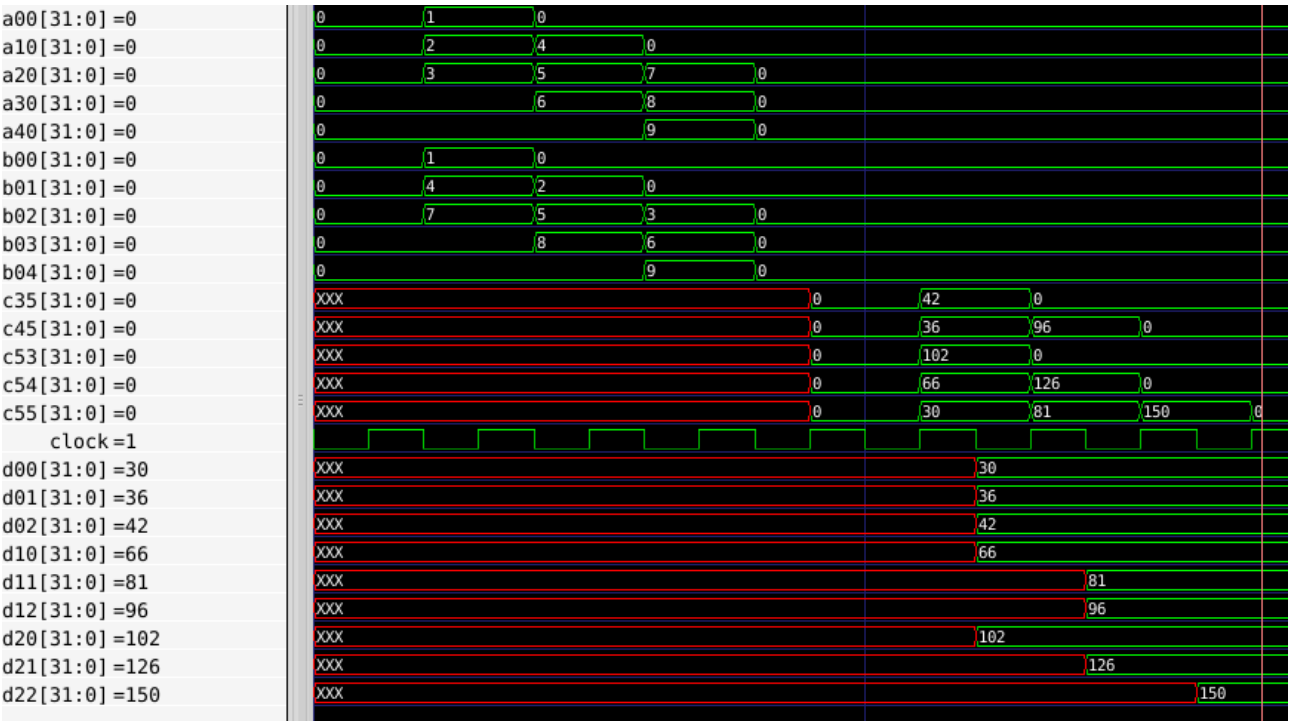

Figure 5: waveform for an example of integer datatype taken from Verilog

IEEE-754 Single Precision Floating Points

For the Matrices

$$A = \begin{bmatrix} 01000000110010000000000000000000 & 01000000000001011100001010001111 0 & 01000000010110011001100110011001 \\ 11000000100010011001100110011001 & 00111111100011001100110011001100 & 01000000101100000000000000000000 \\ 01000001000010101011100001010001 & 11000001000100110011001100110011 & 00000000000000000000000000000000 \end{bmatrix},$$

$$B = \begin{bmatrix} 00111111010000000000000000000000 & 01000001010000000000000000000000 & 01000001000000011110101110000101 \\ 01000001010001010111000010100011 & 00000000000000000000000000000000 & 01000000110111100001010001111010 \\ 01000001000000011110101110000101 & 11000000111010111000010100011110 & 01000000000000000000000000000000 \end{bmatrix}$$

The output Matrix is

$$D = \begin{bmatrix} 01000010011011001100100101101010 & 01000010110001010011000100100110 & 01000001000110000001010101001101 \\ 01000010010111000000100100110110 & 11000001010101101110000101001000 & 11000001000111111110111110011100 \\ 11000010110101100000110100001101 & 01000010110100000001010001111010 & 01000010101110110111000110101000 \end{bmatrix}$$

Figure 6: waveform for an example of Single precision datatype

IEEE-754 Half Precision Floating Points

For the Matrices

$$A = \begin{bmatrix} 0100011001000000 & 0100000001011100 & 0100001011001100 \\ 1100010001001100 & 0011110001100110 & 0100010110000000 \\ 0100100001010101 & 1100100010011001 & 0000000000000000 \end{bmatrix},$$

$$B = \begin{bmatrix} 0011101000000000 & 0100101000000000 & 0100001000000000 \\ 0100101000101011 & 0000000000000000 & 1100011101011100 \\ 0100100000001111 & 0100011011110000 & 0100000000000000 \end{bmatrix}$$

The output Matrix is

$$D = \begin{bmatrix} 0101001101100101 & 0101011000101010 & 0100100011000010 \\ 0101001011100000 & 1100101010110000 & 1100100011111110 \\ 1101011010101110 & 0101011010000000 & 0101010111011010 \end{bmatrix}$$

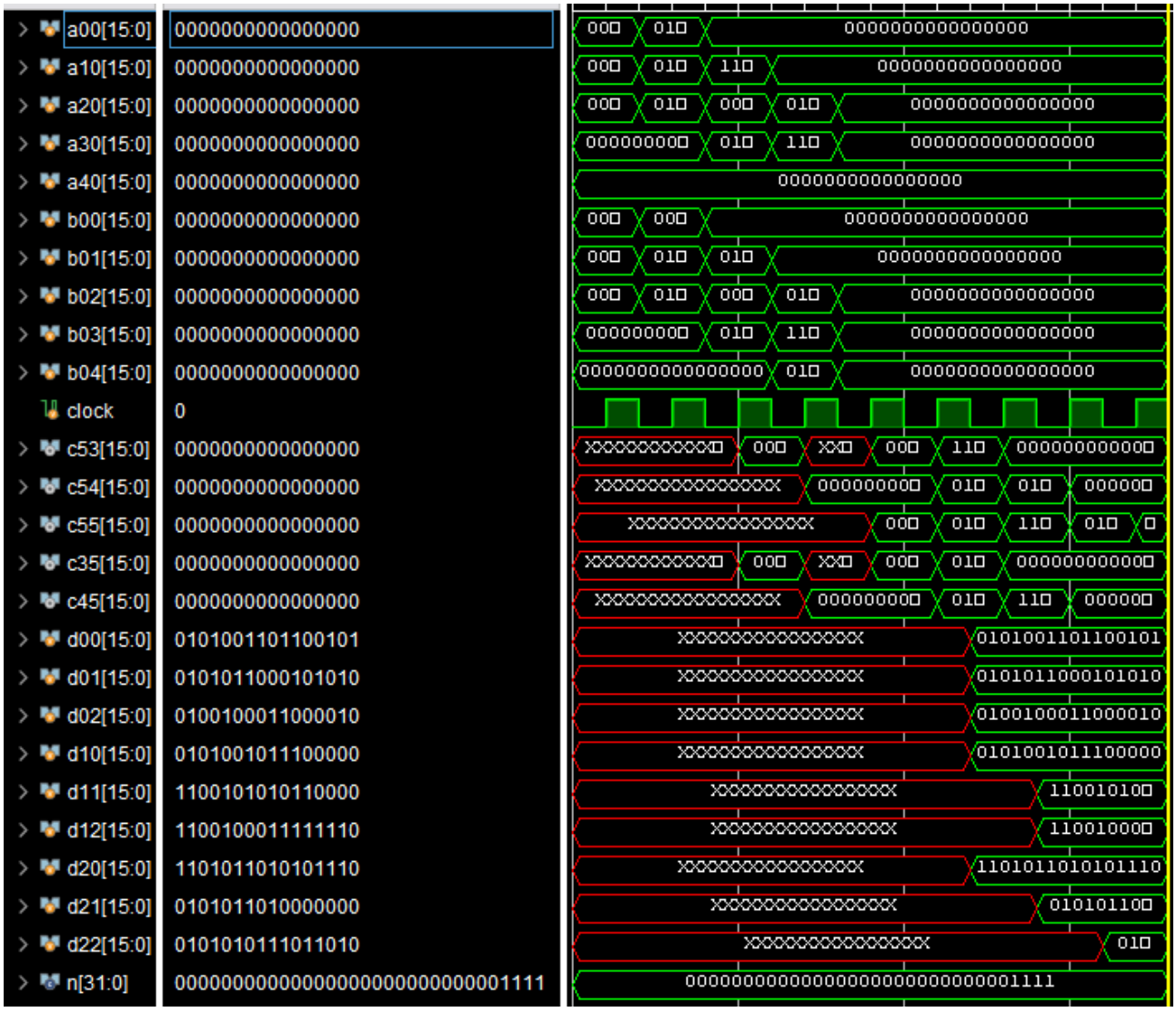

Figure 7: waveform for an example of Half precision datatype

Observations

- PE and Delay blocks to be triggered only either with Posedge clock or negedge clock else Vivado cannot allocate required flipflops to the blocks required.
- Integer data type is taken as 32 bit sized numbers in this model.
- Single precision numbers required Zed board to program on because of the resource constraint i.e, unsufficient no of LUT's
- Block Ram used in this model have 2 clock cycles delay. The code is written accordingly to this condition.
- To get rid of place-design error while implementation caused because of unsufficient no of ports VIO(Virtual input/Output) and ILA(Integrated Logic Analyser) are added as well as to give inputs easily.

Resources

- Youtube video for Systolic array
- IEEE-754 numbers
- IEEE-754 Floating point binary Arithmetic
- Flaoting point Addition
- Floating point Substraction(will be used to add two opposite signed numbers)
- Multiplication of Floating point numbers
- Half Precision numbers
- Bfloat 16