2 x 2 Matrix Multiplier

Jimmy Galat, Chris Folden, Dane Zetouna, Alec Spurlock
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: jagalat@oakland.edu , cbfolden@oakland.edu , dzetouna@oakland.edu , awspurlo@oakland.edu

A matrix organizes a group of numbers using specified arithmetic rules. Matrix multiplication is a binary operation that generates a single matrix from two matrices. This is applicable to many different mathematical disciplines as well as several branches of science.

I. INTRODUCTION

The purpose of this project is to design a 2 x 2 matrix that will output products. This applies to both signed and unsigned numbers. The group accomplished this by using resources obtained in the labs completed throughout this course. Some of these tools are: decoders, registers, a processing unit, an 8-bit adder, and multiplexers. These components were incorporated together to achieve the matrix multiplication.

II. METHODOLOGY

A. Understanding Matrix Multiplication

Before planning the layout of the project, it was essential to understand how matrix multiplications works. For matrix multiplication, the size of each matrix is relevant. The number of columns in the second matrix must equal the number of rows in the first matrix. Luckily, for this project, only a 2 x 2 matrix is being represented so this rule is met. After understanding the process for 2 x 2 matrix multiplication, an example was created. This is shown in **Figure 1** below:

$$\begin{bmatrix} e & f \\ g & h \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} ae + cf & be + df \\ ag + ch & bg + dh \end{bmatrix}$$

Figure 1: 2x2 Matrix Multiplication Process.

Figure 1 displays the systematic procedure for matrix multiplication. After understanding this, the next step was to begin the layout utilizing the tools learned from class and lab sessions.

B. Layout of the Project

After many ideas were discussed, the plan was to have a 4-bit address input into the decoder. This will select which of the 8 registers to write into and which registers to read the data from. From there, the data will be taken from the first 8 registers and then multiplied and added together in the format shown in **Figure 1**. This will be done using the material for processing units learned from previous labs. The result will then be stored into the last 4 of the total 12 registers. From there a second multiplexer with 4 to 1 characteristics will select which value of the corresponding matrix, and send it through the serializer and onto the 7-segment display in Hex. This will be done by changing the address of the decoder and switching from write to read. The layout for the first half of the project is shown in **Figure 2** below:

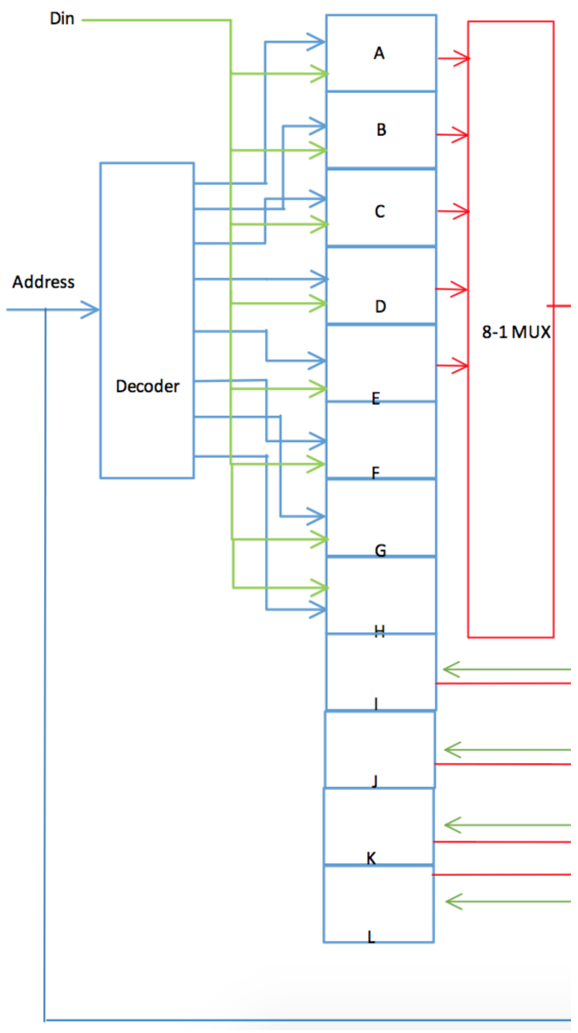

Figure 2: Decoder/Registers

In the first half of the Matrix-multiplication circuit, a 4-bit address line corresponding to each of the first 8 registers allows the 4-bit input from Din to be saved into the desired register. Each of the registers corresponds to each of the 8 spots in the multiplying 2 x 2 matrix as shown before in **figure 1**. The next step shown in figure 2 will be the 8 to 1 multiplexer. This multiplexer will pick and choose the proper register values to send through the processing unit that will then multiply the binary inputs.

The second portion of the matrix multiplication circuit is shown in Figure 2: This portion starts by showing the proper 4-bit values in each register being multiplied accordingly. In matrix multiplication the matrix values after being multiplied together are then actually added to another set of multiplied values. For this task four 8-bit adders were used to add each of the 4 products from the multiplier. A total of 12 registers were used throughout the entire project. 8 registers as said before were used to saved the corresponding matrix values. Now the last 4 registers of the total 12 are going to be used to store the final products of the overall matrix multiplication. In total a 2 X 2 matrix multiplied will give a total 4 products as explained at the beginning of the project.

At this point in the circuit everything has been multiplied and stored into the correct registers. Now it is time for the outputted matrix values to be sent through a serializer in which outputs the binary values in Hex on the 7 segment display. Each value is chosen to be displayed by the 4 to 1 multiplexer.

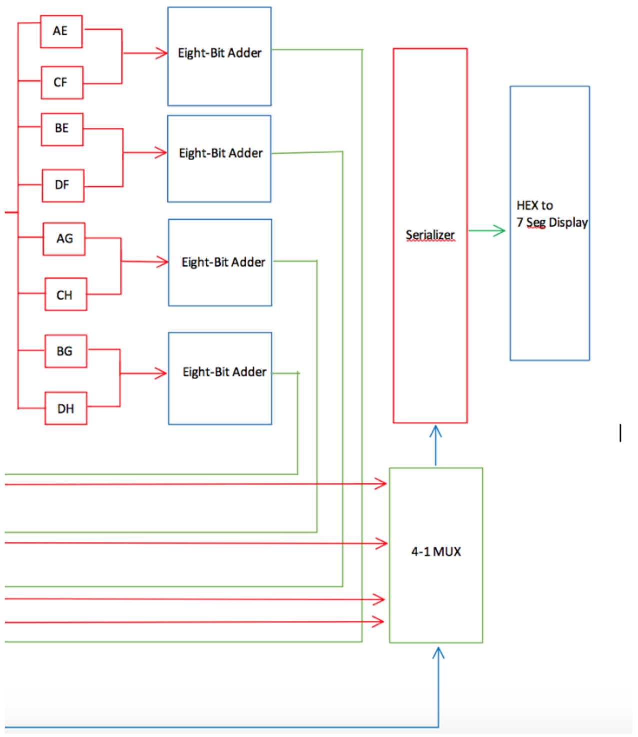

Figure 3: Multipliers/Adder/Display

As far as displaying the outputs, it was decided that the most effective method would be to cycle through each output value manually. Once the operator switch is switched high the user can read through the output matrix values by entering the binary representation for registers 8-12 in the 4-bit address input.

III. EXPERIMENTAL SETUP

For the hardware, a Nexys-4 DDR board was programmed using the design discussed in this report. For the software, Xilinix ISE 14.7 was used to design, synthesize, and implement the Nexyus-4 board. To verify proper functionality of the matrix, a test bench was created and functional simulations were performed.

In Figure 4, the inputted numbers into the matrix were set up in the test bench as:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} x \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

Figure 4:

The math operations are the following:

$$= \begin{bmatrix} 1x5 + 2x7 & 1x6 + 2x8 \\ 3x5 + 4x7 & 3x6 + 4x8 \end{bmatrix}$$

$$= \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$$

The numbers were sent and stored into the 8 corresponding registers, selected and multiplied, added together accordingly, and stored into the final-product registers. This simulation's purpose is to illustrate that the circuit correctly multiplied the 2 X 2 matrices properly and stored them into the proper registers. **Figure 5** below illustrates this:

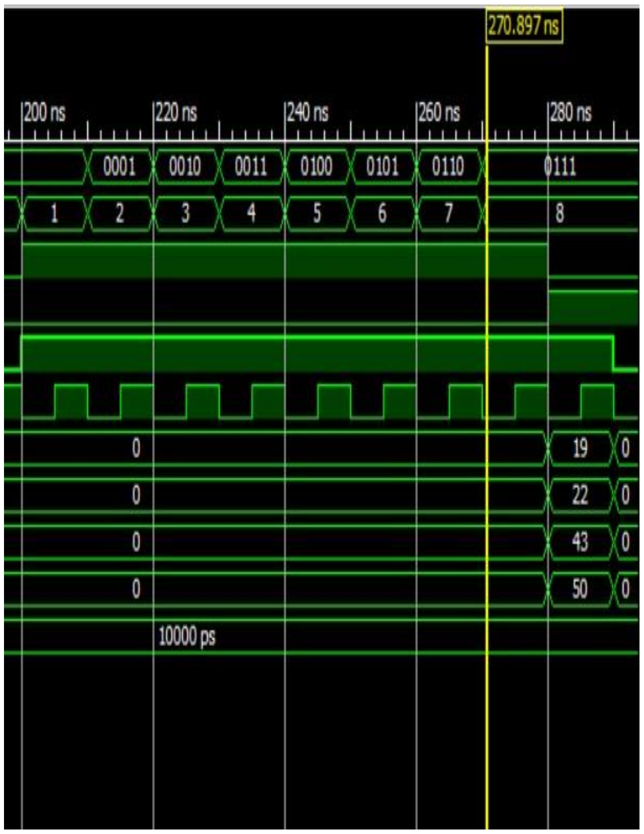

Figure 5: Timing Diagram for register inputs

The next simulation test bench the team set up would have the goal to demonstrate the entire final product onto the seven-segment display in Hex. With the same numbers inputted into the matrix in this simulation, just as before the numbers were stored, multiplied, added together and stored into the final product registers. Now illustrated in **Figure 6** will be the final output onto the seven-segment display. Now within this simulation it is set up to automatically display each output, unlike in the experimental program on the FPGA board where these values have to be cycled through manually.

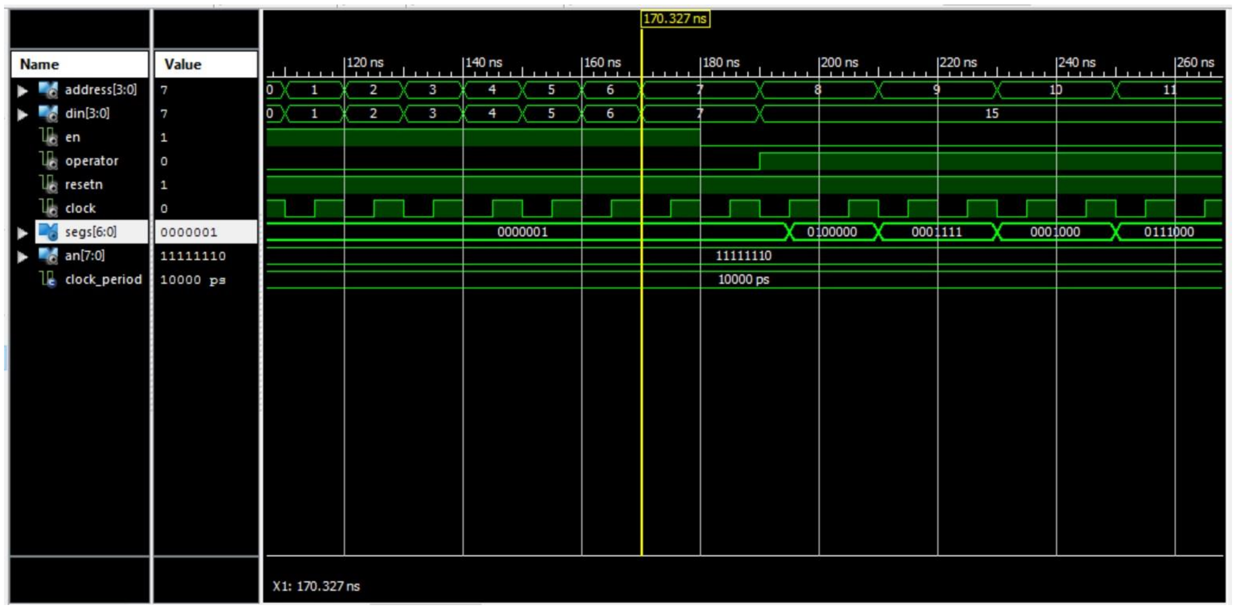

Figure 6: Final Timing Diagram/Seven-Segment Display Output

As seen in **Figure 6**, the inputted numbers of 1, 2, 3, 4, 5, 6, 7, and 8 all go through the circuit and properly multiply accordingly. Now what this simulation illustrates is the seven-segment display of 19, 22, 43, and 50 in binary.

IV. Results

The 2 X 2 matrix multiplier overall was successful in multiplying 4-bit inputs in matrix multiplication. This included using the following components learned in our labs throughout the semester: Read/Write into a register, choosing a register address with a multiplexer, a processing unit, five-bit adder, a serializer, and HEX to 7 segment display conversion.

The hardware portion of the project included the use of a total of 10 switches, the seven-segment display, and the reset button. **Figure 7** illustrates the whole layout for the switches. 4 switches as shown were used for the 4-bit Data-in, and another 4 switches were used for the 4-bit address selector.

The far left two switches were used as the Enabler/Writer while the other was implemented as the operator/Reader. The enabler allowed for the inputs to be written into the registers as needed. The Operator starts the math operations.

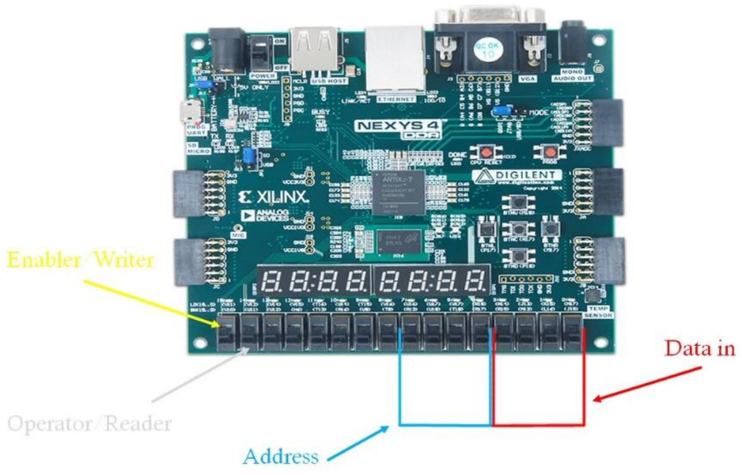

Figure 7: Nexys-4 DDR board layout

V. Conclusions

Overall this project was successful in multiplying Unsigned numbers in a 2 X 2 matrix. As a team we were able to implement what we learned throughout the course and implement the needed blocks of code to perform the given task.

A few problems occurred during the duration of the project. Originally the goal was to be able to multiply both signed and unsigned numbers and output the correct products. However time was not in our favor and this part of the task could not be completed. If we could make improvements to our 2 X 2 matrix multiplier, being able to multiply signed integers would be at the top of the list. Along with implementing signed integers, being able to automatically display all of the 4 final products without manually cycling through would make this a much more practical calculator.

Our project turned out to do exactly what original task was though with simply multiplying a 2 X 2 matrix and displaying the results. The two improvements listed were extra characteristics we as a group wanted to implement but were not able to get that far with the time we had. Our board successfully stores data, processes it, and displays to the user the proper multiplication of two matrices.

VI. References

[1] Weisstein, Eric W.. *Matrix*. (1999-2013). MathWorld-A Wolfram Web Resource. Accessed on December 17, 2013

[2] Daniel Llamocca Website