2x2 Matrix Multiplier

Alec Hill, Brendan Zarycky, Eric Niezabytowski, Dmytro Melnikov
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: AlecHill@oakland.edu, Melnikov@oakland.edu, BrendanZarycky@oakland.edu, Eniezabytowski@oakland.edu.

Matrix multiplication is a binary operation that takes two matrices, performs some multiplication and addition, to form a single matrix from the previous two. Matrix multiplication is applicable to many different branches of science as well as mathematics.

I. Introduction

The scope of this project is to design and implement a 2x2 matrix multiplier on our FPGA boards that will output the products on the seven segment display for both signed and unsigned numbers. Using what was learned in the labs, we made a block diagram (Figure 2 Appendix) to help create the system much like how our labs were presented. Components from previous labs, such as: multiplexers, adders, multipliers, decoders, and registers were used. These components incorporated together helped to achieve the working matrix multiplier.

II. Methodology

A. Matrix multiplication

Knowing how to multiply matrices is a key to understanding how to implement the various components of the circuit. For matrix multiplication, the number of columns in the first matrix must be equal to the number of rows in the second matrix. This project only focuses on 2x2 matrix multiplication, thus simplifying the architecture a little. Multiplication of the 2x2 matrix is shown below in Figure 1.

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} \times \begin{pmatrix} E & F \\ G & H \end{pmatrix} = \begin{pmatrix} AE+BG & AF+BH \\ CE+DG & CF+DH \end{pmatrix}$$

Figure 1: 2x2 Matrix Multiplication explained

This figure breaks down the steps to the multiplications which will help create the overall system.

B. Implementation on FPGA

After much discussion on how to design the circuit, the group came to a conclusion on what was thought to be the best system that will perform the function. It implemented a good chunk of the lab 5 circuit as the start for the system. Using a 3-bit address input for the decoder in lab 5, that same idea was used for the matrix multiplier. Eight numbers are needed for the starting matrices, which are being written to eight registers. The multiplication and addition will need components that were used in lab 2 and lab 3 such as the processing unit but instead of a 4-bit adder, the system will need an 8-bit adder since it is multiplying two 4-bit solutions which gives an 8-bit output. Before being sent to the display, a 4-1 multiplexer was implemented. This will send each value of the resulting matrix numbers to the serializer, and then on to the seven segment display manually. After the last multiplexer it will go through the serializer to display on the 7 segment display in two hex numbers and will use a 2-bit address as a select for the multiplexer to change the values manually.

III. Experimental Setup

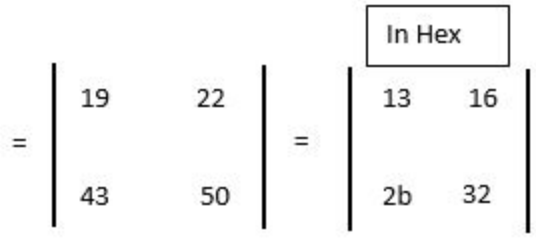
To verify the function of the project, the nexys-4 a7-50t board was used. Utilizing Vivado to design, test, synthesize, and implement the system onto the nexys board. To help verify the proper functioning of the system, a test bench was created and simulated to see potential results. The matrix below in Figure 3 was implemented into the test bench.

$$\begin{vmatrix} 1 & 2 \\ 3 & 4 \end{vmatrix} \times \begin{vmatrix} 5 & 6 \\ 7 & 8 \end{vmatrix}$$

Figure 3: Test bench Matrix

Following Figure 1, the matrices are multiplied as shown on the next page along with the answer to the matrix multiplication.

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



From start to finish, using an initial total of 8 numbers, with 4 numbers for each matrix were sent and stored to the correct corresponding registers. From the registers, values were sent to the correct multiplier where it multiplies to the corresponding number in the multiplying matrix (As shown on Figure 2). After the multiplication, an 8-bit number corresponding to each multiplication is then sent to the 8-bit adder along with the correct corresponding value for that value position in the resulting matrix. Once Added, they are sent to 4-1 multiplexor where they are manually selected to show on the 7-segment display. The 8 bit number must go through the serializer to show the 2 hex numbers on the display. The simulations in Figure 4 of the appendix help illustrate the different values that will show on the 7 segment display when the second display is active as well as when the first display is active.

For the hardware portion, a total of 10 switches were used. 4 for the data in, 3 for the address they are being written to, 2 for the select to show which corresponding number we want to show, as well as 1 switch to read the values and to write the values exactly like lab 5. Implementing a serializer to show two 7-segment displays, as well as a button to reset the values in the registers and restart the multiplication as a whole. The layout the switches, display and reset button is shown in Figure 5 in the appendix.

IV. Results

The matrix multiplier was successful in multiplying 4-bit inputs in the matrix multiplication equation. Using many components that were used before, like the bit adder which was used in lab 2, the multiplier which was the top file in lab 3, the same decoder and registers that were implemented in lab 5, along with the multiplexer which was used in a few labs. Also the serializer which helped display two hex numbers on the 7-segment display.

The results obtained from running the simulations that are shown in the appendix in Figure 4, showed that the values coming out were correct as the math operations would get the same answers in multiple different scenarios with different numbers in each place. This helped to conclude that everything was connected correctly and that it could display any number up to 255 since only two 7-segment displays were employed.

Conclusions

This project was very successful in multiplying unsigned 4 bits numbers in a 2 by 2 matrix. We were able to implement a part of almost every lab to help make our function systems.

We originally wanted to implement a BCD to 7-segment display rather than a hex to 7-segment display, but the dead line quickly came up and our group was not sure how to implement this in our working system. Improvements that could be made would be the BCD to 7-segment display as it would be easier for everyone to read the output, as well as multiplication of signed numbers. An improvement that we also considered is with the seven segment display with the data in, for example as we put a switch combination for the Data in to the address we would then see that value is displayed on the seven segment display. Another improvement we can make is with the multiplexer and display. If we used the seven segment display to show currently selected matrix solution position with the value of that position. This helpful to follow the overall flow of the matrix. A change that could be made if instead of manually going through the results, if we could implement a display or graphic interface so that we have a matrix format that we can input our values into. This will make it easier to track our 8 inputs and solutions. This can also are current issue the max limit of numbers that can imputed as well.

The project turned out exactly as we anticipated by simply multiplying two, 2 by 2 matrices to form one resulting matrix. Knowing what improvement we could make made us proud of what we got done in the short time we had to create this multiplier. Our system was able to store 8 values in the registers, multiply and add the correct values together, as well as display the correct value we wanted to see when we wanted to see it.

References

[1] OWL, “Recursive Matrix Multiplication Strassen Algorithm.” *Mathematics Stack Exchange*, 1 Sept. 1966, math.stackexchange.com/questions/1854288/recursive-matrix-multiplication-strassen-algorithm/1854422.

[2] Daniel Llamocca’s Unit 7 serializer example project.

APPENDIX

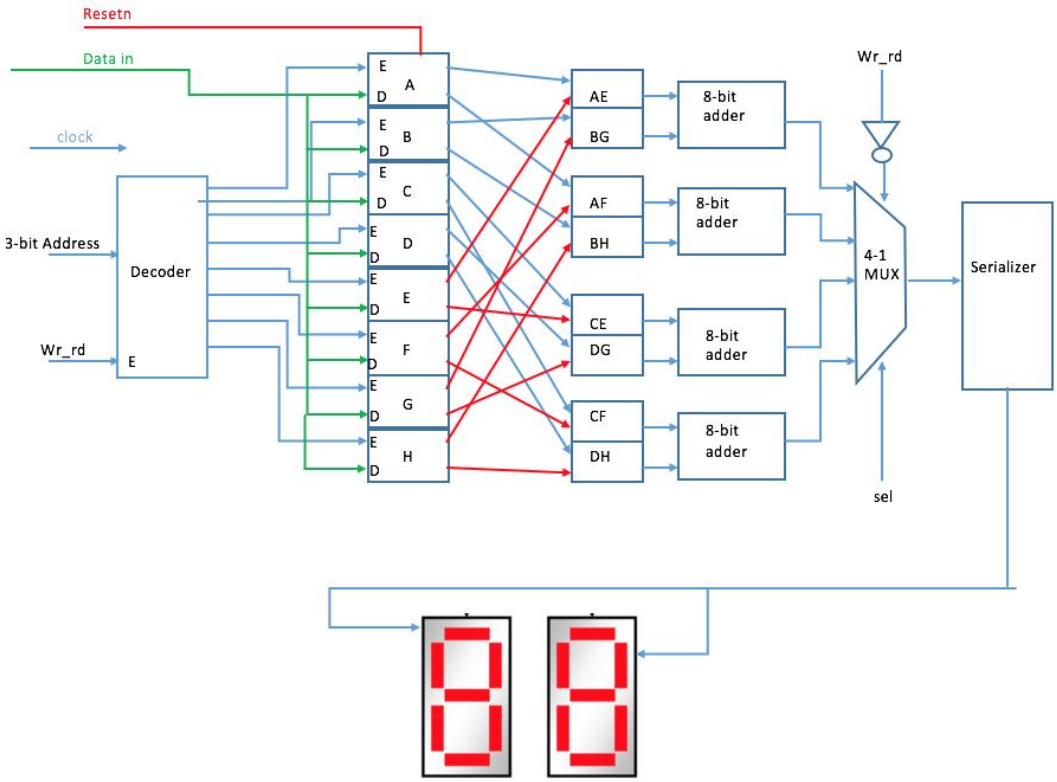


Figure 2: Block Diagram of Whole System

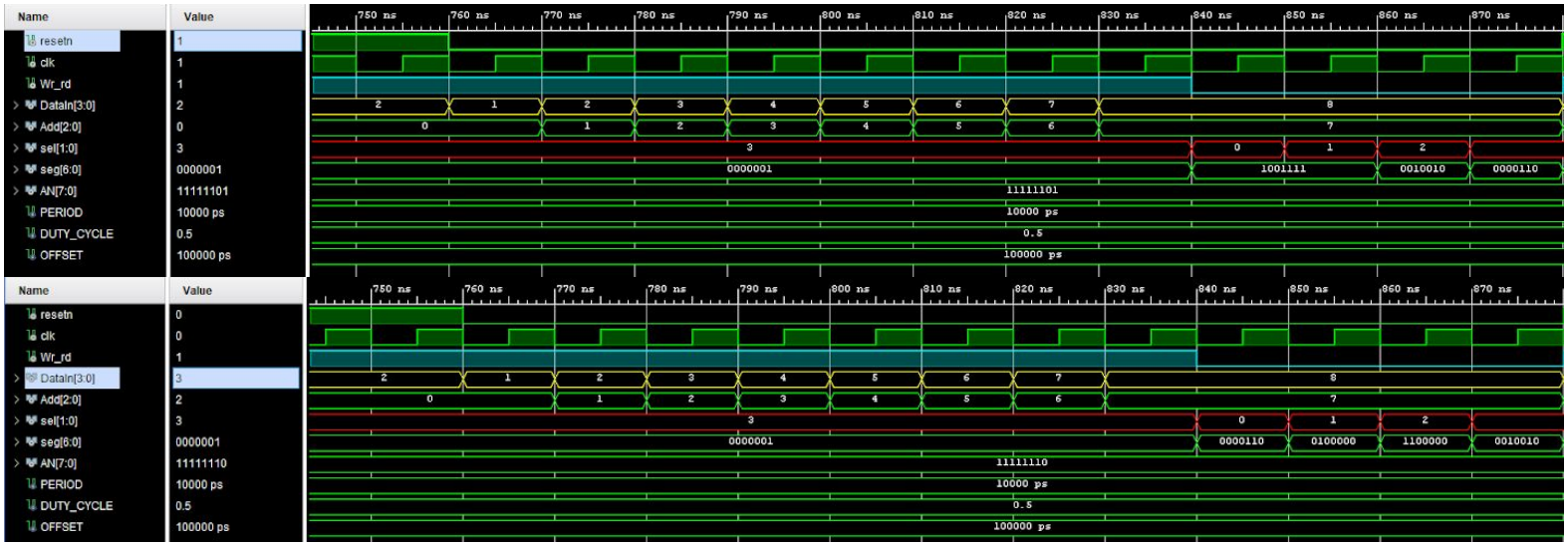


Figure 4: Simulations for each 7-Segment Display

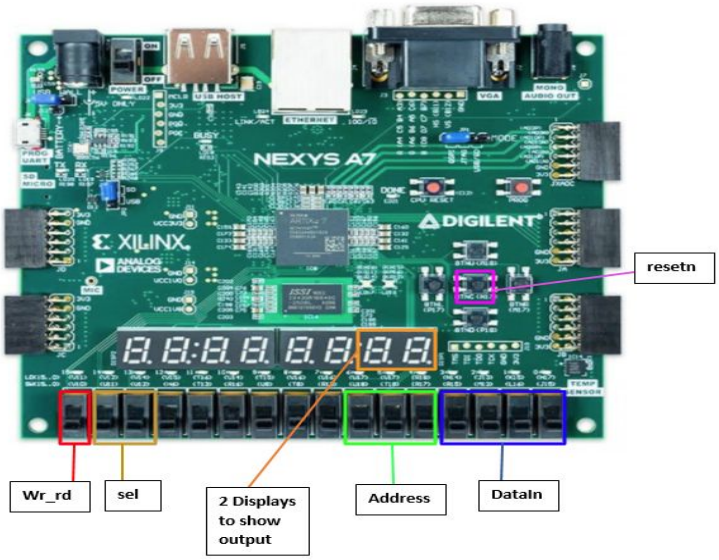


Figure 5: Nexys a7-50t Set up333