Design and Implementation of 2x2 Matrix Multiplication

James Sherwood, Tim Kozen
Electrical Engineering Undergraduate Students
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: jsherwood@oakland.edu, tlkozen@oakland.edu

***Abstract*- This project outlines the development and implementation of a 2x2 matrix multiplication device. It is shown that this can be successfully realized using hardware input switches, and output 7 segment displays, in conjunction with Nexys A7 FPGA board programming. For simplicity, the device studied is limited to unsigned input and output elements. The successful operation is proven through simulation, and a full final demonstration.**

I. Introduction

The intent of this project is to implement a 2x2 matrix multiplier using VHDL and a programmable Nexys A7 FPGA board. The user must be able to input single digit matrix elements using controls on the FPGA board, and subsequently be able to view the elements of the resulting matrix through on-board display components. Matrix multiplication is a tedious process, and highly subject to human error. Calculations using digital logic have obvious advantages, especially when scaled.

The design of this matrix multiplication system requires the use of tools from every section of this semester's digital logic design course. From the fundamentals of binary unsigned numbers, to the cascading of logic gates for mathematical operations, to the storage of data using flip flops, and the encoding of that data to be displayed using 7 segment displays. This project provides an excellent case study in connecting these ideas together to create a useful tool.

II. Methodology

The matrix multiplication device is divided into three sections: User data input, mathematic operations, and resultant output display. These components are coded in a modular fashion and are detailed in the following sections.

A. User Data Input

To begin, each 2x2 matrix to be multiplied has four elements. These eight separate matrix elements are inputted and stored into the circuit via a decoder and register.

$$\begin{bmatrix} a0 & a1 \\ b0 & b1 \end{bmatrix} X \begin{bmatrix} c0 & c1 \\ d0 & d1 \end{bmatrix}$$

Numerical vales from 0 to 9 are inputted in binary form through five switches on a Nexys A7 FPGA board. Each of the eight input matrix elements have a corresponding switch also located on the FPGA board. In addition, there is also an enable EN switch that when selected will store the user inputted numerical value into one of eight register slots.

The user first selects the binary number which is to be stored by activating the appropriate switches. Then, the specific switch that corresponds to the desired matrix position is selected. Finally, the EN switch is activated and then deactivated. This procedure is the repeated for each successive number in the matrices to be multiplied.
When the matrix element position switch and the EN switch are selected at the same time a decoder recognizes that specific 9bit combination sends a select signal to a register that activates one of eight memory slots corresponding to the selections.

The input data is sent simultaneously to all register slots. However, only the register slot activated by the decoders signal will allow the data entry to be stored via a flip flop.

B. Mathematical Operations

The stored numbers are then passed through eight separate multipliers. The architecture for the eight separate multiplication operations are achieved through simple VHDL language. Two 2x2 matrices are multiplied using the following algorithm.

$$\begin{bmatrix} (a0c0 + a1d0) & (a0c1 + a1d1) \\ (b0c0 + b1d0) & (b0c1 + b1d1) \end{bmatrix}$$

The resulting values from the multipliers are represented by signals m0-m6, please reference fig 2.1 for schematic representation. These signals have a range of [0 to 81], requiring 7 bits to represent. These signals are passed to four 7-bit adders. The adders use entirely combinational circuits to perform the addition operation. The adder outputs are the final elements for the resultant matrix. These elements are represented by signals r0 – r3.

$$\begin{bmatrix} m_0 + m_1 & m_2 + m_3 \\ m_4 + m_5 & m_6 + m_7 \end{bmatrix} \rightarrow \begin{bmatrix} r_0 & r_1 \\ r_2 & r_3 \end{bmatrix}$$

Signals r0 – r3 have a range of [0 to 162] requiring 8 bits to represent. These four signals are terminated at a 4:1 MUX to be select-ably passed to the output circuit.

C. Resultant Data Output and Display

Once the resultant matrix elements have been calculated, the elements are displayed, one at a time, on the lower three 7 segment displays. The matrix element (0-3) is selected

through the use of two input switches. The combination of these switches creates the 2 bit signal "sel" in fig 2.1. This signal is input to the 4:1 MUX, which passes the corresponding element to its output.

To support display functionality, the 8-bit output of the 4:1 MUX is converted to three 4-bit BCD signals, one for each possible digit in the element value. This is accomplished using the "shift and add 3" method *See reference 2.*

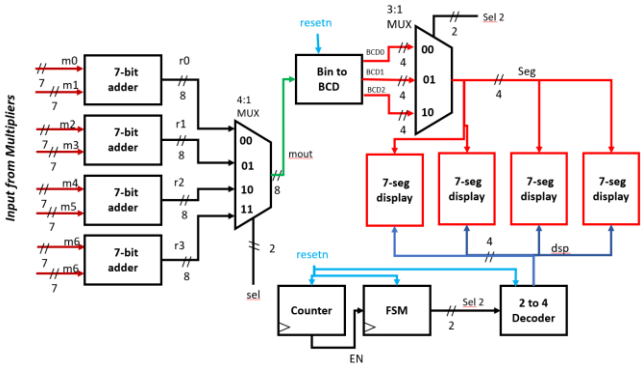

Fig 2.1: Adder and output circuits

Since there are four 7 segment displays, and only one 7 segment display driver, the displays must be activated serially. This means that for a short period of time (100us) each display is exclusively activated, displaying the corresponding element digit. All three displays used are activated, in order, in this way, repeatedly. Since the human eye cannot distinguish events faster than 60Hz, the complete element value is seen, and the serial activation is not observed.

To drive the serial activation of the 7 segment displays, a finite state machine (FSM) is used. The FSM state diagram is shown in fig 2.2.

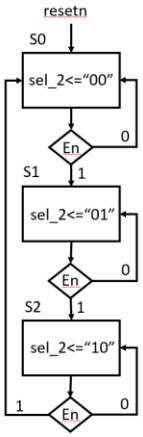

Fig2.2: FSM state diagram

The FSM is triggered to change state using a counter generated "En" signal. This bit is flipped to '1' every 10,000 clock ticks (100us). The output of the FSM is the "sel_2" signal used by both the 3:1 MUX and the 2 to 4 decoder to activate the correct 7 segment display, and display the correct value simultaneously.

III. Experimental Setup

Simulation of the addition operation, and output displays in shown in fig 4.1. For this simulation, the eight 7-bit inputs to the adders are hard-coded into the test bench. The allows the correct adder output to be verified, as well as the input and output of the FSM.

IV. Results

The following timing simulation and accompanying diagram represents the input section of the matrix multiplier. It shows that the inputs values are stored into the registers. In this case, the stored numbers are integers ranging from 2 through 9. . The radix has been switched to unsigned integers for ease of demonstration.

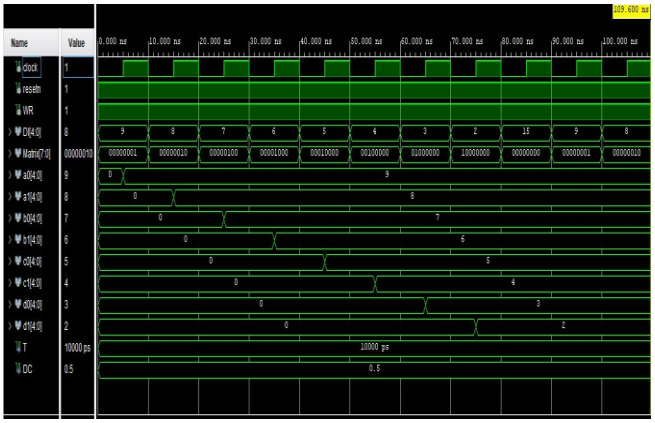

Fig 4.1: Multiplier input simulation

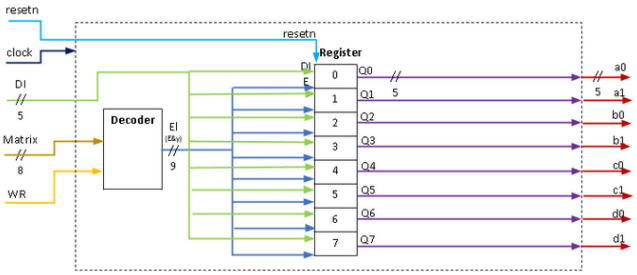

Fig 4.2: Multiplier and input circuits

The below figure shows the testbench of one of the multipliers. As can be seen, when the inputs to be multiplied are 5 and 9 or 6 and 7 the result is 45 and 42 respectively. Again, the radix has been switched to unsigned integers for ease of demonstration.

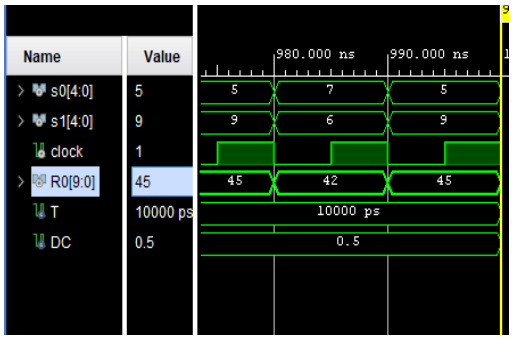

Fig 4.3: Multiplier simulation

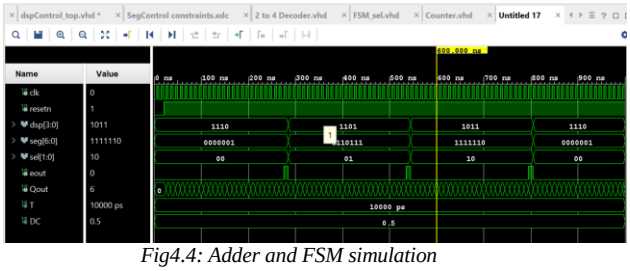

Fig4.4: Adder and FSM simulation

With all simulations successful, the hardware is programmed with the completed top file. The successful demonstration can be view using the following link:
[Kozen Sherwood ECE 2700 Final Project Demo - YouTube]

V. Conclusions

The 2x2 matrix multiplier was successfully implemented on the FPGA hardware. This project provided an excellent opportunity to cascade the various components learned through out the semester.

One of the largest challenges in the development of this device turned out to be the binary to BCD converter (algorithm that we did not learn in ECE 2700). Also, learning that there is only one 7 segment display driver on the Nexys A7 FPGA board, necessitating the serial display system (Counter, FSM, and MUX).

Even though the project was successful, there are areas that could be improved: The input elements could be increased in range or improved to accept signed numbers. The input could also be improved to use a more user-friendly mechanism (perhaps a keyboard and computer monitor). The output could similarly be improved to display all resultant matrix elements simultaneously (again, perhaps a computer screen).

References

1. Ronald J. Tocci, Neal S. Widmer, and Gregory L Moss "Digital Systems Principles and Applications" Pearson Education, Inc., 330 Hudson Street , New York New York 10013 (2011).

2. "VHDL Code for Binary to BCD Converter". Invent Logic, All about FPGAs. VHDL Code for Binary to BCD converter (allaboutfpga.com)