ENEE640: Designing 8x4-bit Sequential MAX Circuit (Exam 2)

Nirvan Mishra
nmishra1@umd.edu

***Abstract*—This document describes the design of a sequential MAX circuit for eight unsigned 4-bit numbers (X0..X7). The circuit takes these eight inputs and outputs the maximum among the eight numbers. The report outlines the design approach and provides a transistor-level schematic for the core sorting element.**

I. Introduction

Digital circuits can be categorized into two main types: combinational and sequential. Combinational circuits generate outputs based solely on their current inputs. Sequential circuits, on the other hand, incorporate memory elements like registers to store past inputs and influence future outputs. This memory allows them to perform more complex tasks.

One type of sequential circuit is a max circuit, which identifies the largest value among its inputs. When dealing with a sequence of data, a simple combinational max circuit wouldn't suffice. It wouldn't remember the previous maximum value encountered.

This is where a register comes in. By incorporating a register into the max circuit, we create a **sequential max circuit**. This circuit can continuously track the maximum value seen so far in the data sequence. The register stores the current maximum, and the combinational logic compares new inputs to this stored value, updating the register if a larger value is found.

This approach allows the circuit to operate on a stream of data, identifying the overall maximum value across the entire sequence.

The design in utilizes a counter that counts the number of cycles, which in turn selects the input

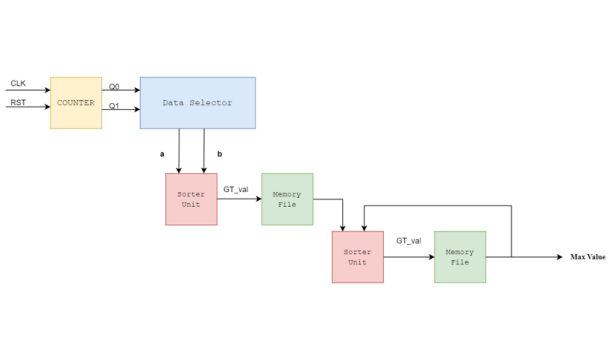

Fig. 1. Sequential Max Circuit

II. Design

A. Counter Unit

The counter unit counts the number of clock cycles (at each rising edge), the unit is built using a 1 bit register, that is constructed using D flip flop.

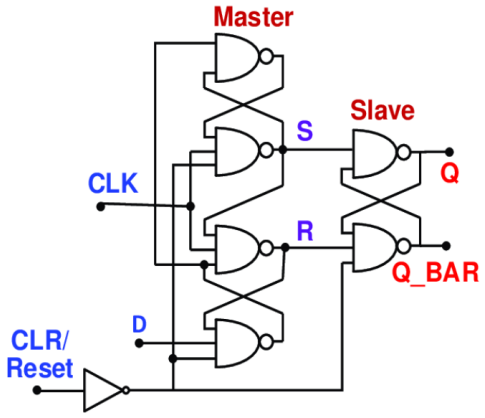

Fig. 2. D Flip Flop with asynchronous RESET

RESET	CLK	D	Q	$\bar{Q}$
1	x	x	0	1
0	↑	0	0	1
0	↑	1	1	0

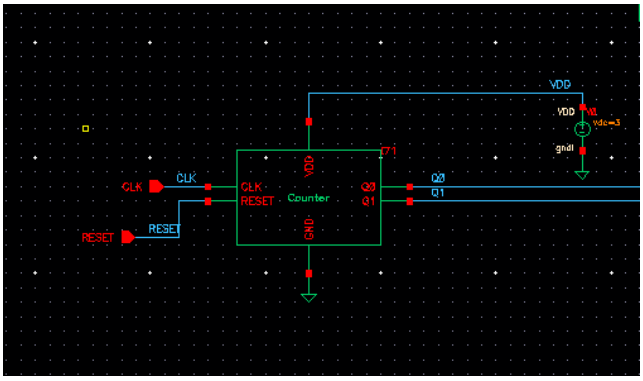

Fig. 3. 2 bit Counter

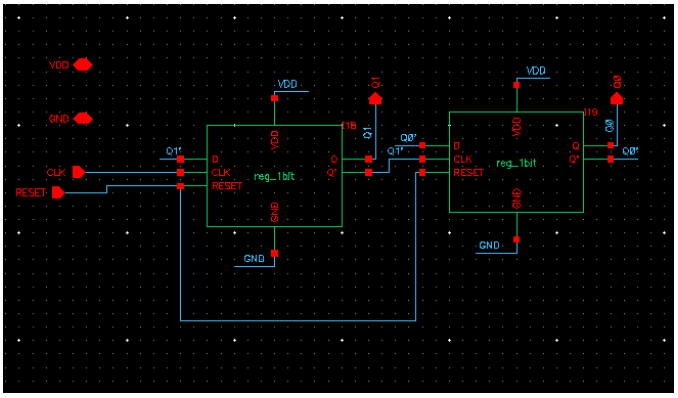

Fig. 4. Counter schematic

CLK	Q0(MSB)	Q1
1	0	0
1	0	1
1	1	0
1	1	1

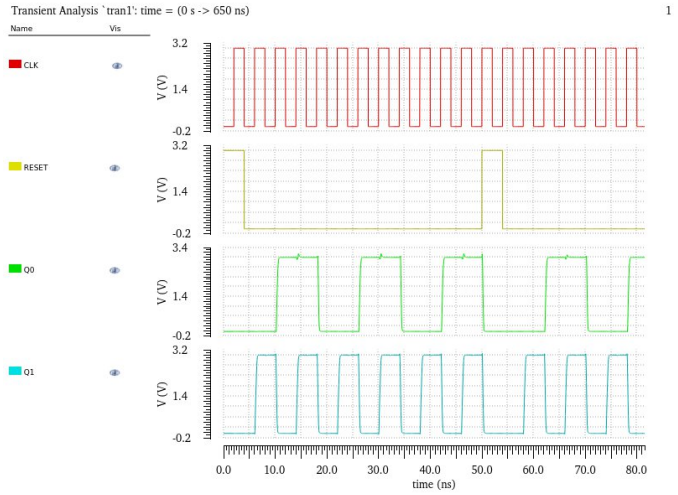

Fig. 5. Waveform

B. Data Selector

The data selector module is essential to the data selection process because it provides inputs to the sorter unit. It receives two inputs, Q0 and Q1, which are obtained from the counter output and operate on a clock cycle basis. The sorter unit then carefully examines these inputs to determine which one has a higher value. The output of the higher value is then forwarded by the sorter unit.

CLK	Q0(MSB)	Q1	a	b
1	0	0	X7	X6
1	0	1	X5	X4
1	1	0	X3	X2
1	1	1	X1	X0

Boolean Expressions':

$$a = \overline{Q0}\cdot\overline{Q1}\cdot X7 + \overline{Q0}\cdot Q1\cdot X5 + Q0\cdot\overline{Q1}\cdot X3 + Q0\cdot Q1\cdot X1 \quad (1)$$

$$b = \overline{Q0}\cdot\overline{Q1}\cdot X6 + \overline{Q0}\cdot Q1\cdot X4 + Q0\cdot\overline{Q1}\cdot X2 + Q0\cdot Q1\cdot X0 \quad (2)$$

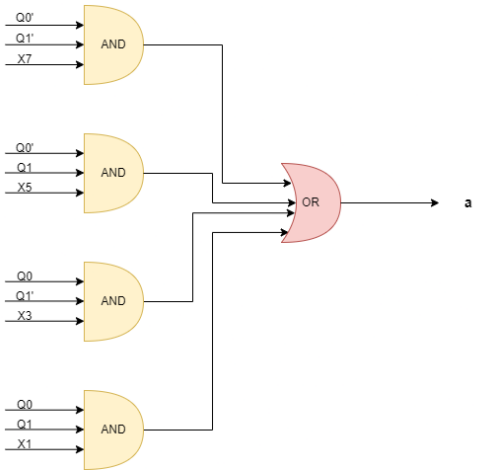

Fig. 6. Circuit for 'a' Output pin

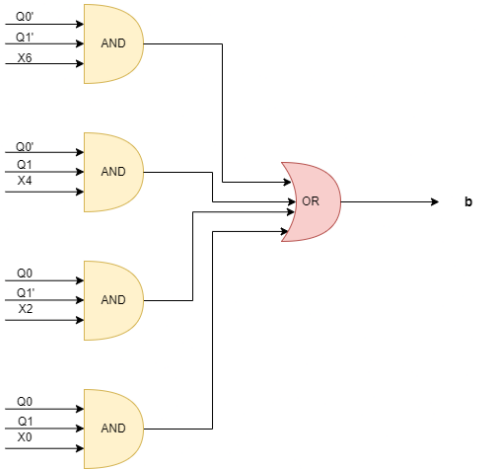

Fig. 7. Circuit for 'b' Output pin

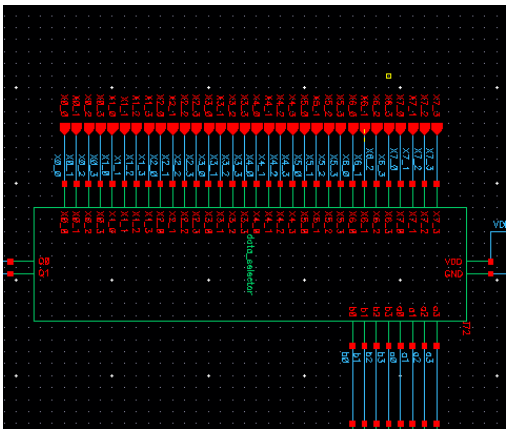

Fig. 8. Data Selector

C. Sorter Unit

The sorter unit, consists of comparator and swapping circuit. In the sorter unit, first the comparator circuit takes the two inputs, compares them, produces A_GT_B; A_LT_B; and A_EQ_B. Along with the inputs and these outputs of comparator circuit as selection lines, to the swapping circuit, swaps the two inputs depending on which selection line is high.

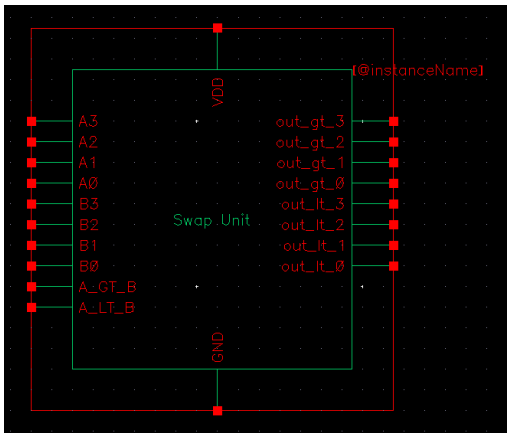

Fig. 9. Sorter Unit

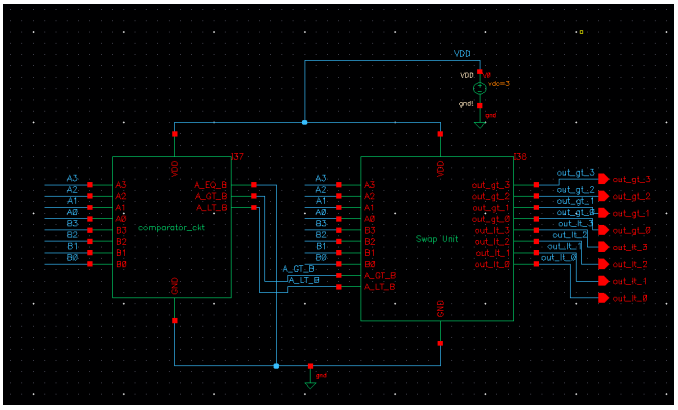

Fig. 10. Sorter Schematic

Sorting Process:

Figure 11 depicts a single step in a sorting algorithm, a bitonic sorting variation, that compares two binary numbers, A and B, to determine their order.

The comparator starts by comparing the Most Significant Bits (MSBs) of A and B. In binary representation, MSB is the leftmost bit, which holds the greatest weight in determining the overall value.

A (binary 4): In this case, A's binary representation is 0100 (assuming 4 bits). Therefore, the MSB of A is 0. B (binary 8): B's binary representation is likely 1000 (assuming 4 bits). Therefore, the MSB of B is 1. Since A's MSB (0) is less than B's MSB (1), the comparator determines that A is less than B. This comparison result is typically indicated by a high signal on a designated output line, often named A_LT_B in this case.

The high signal on A_LT_B triggers the swapping unit. This unit takes the original inputs A and B and swaps their order to ensure the larger value (B) comes after the smaller value (A) in the sorted sequence.

out_{gt} (greater than): This output line becomes 8, which was the original value of B. out_{lt} (less than): This output line becomes 4, which was the original value of A.

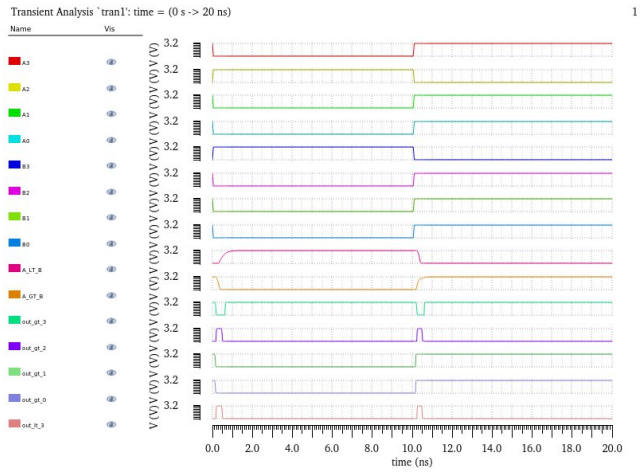

Fig. 11. Sorting Unit output

D. Memory File

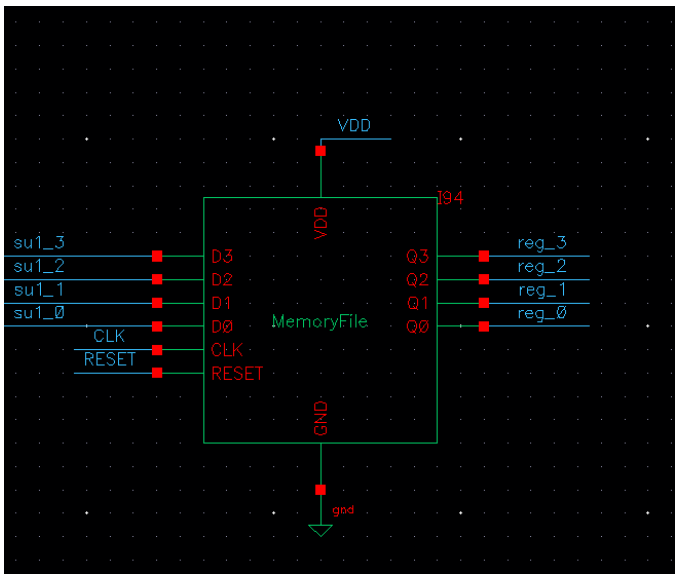

Fig. 12. Memory File

A 4-bit memory file constructed with D flip-flops (DFFs) operates as a fundamental component in digital systems, facilitating the storage and retrieval of binary data. Each DFF within

the memory file represents one bit of the overall memory capacity. When reset is high (1), all D flip-flops within the

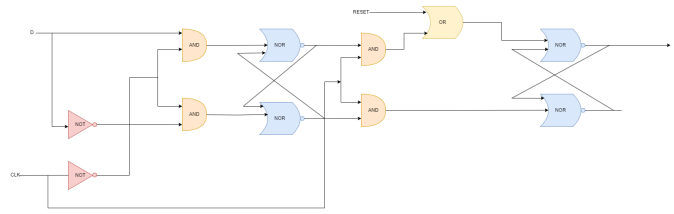

Fig. 13. 1 bit DFF

memory file are set to 0. When the reset signal is asserted, it triggers the DFFs to reset to their default state, setting all bits to 0. Subsequently, as data is inputted into the memory file, each D flip-flop stores one bit of the binary data. For instance,

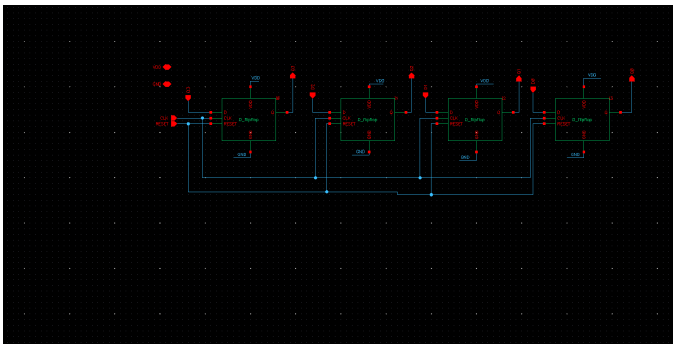

Fig. 14. Memory File

let's denote the D flip-flops as DFF0, DFF1, DFF2, and DFF3, representing the four bits of the memory file. The data input lines are labeled D0, D1, D2, and D3 respectively, and the output lines are Q0, Q1, Q2, and Q3.

When new data is presented to the memory file, it's synchronized with the clock signal, causing the D flip-flops to latch the input data. Upon the rising edge of the clock, the data on the D input of each flip-flop is transferred to its output, thereby preserving the stored information until the next clock cycle.

III. Sequential Max Ciruit

This mechanism outlines a sequential comparison process implemented over multiple clock cycles to determine the maximum value among a series of inputs.

During the first clock cycle, the inputs x7 and x6 undergo comparison using the initial sorter unit. The maximum value is then captured and stored in a register. In the subsequent clock cycle, the output of this register, representing the previously determined maximum value, is juxtaposed with the input values x5 and x4 using the second sorter unit. Through this comparison, the new maximum value is identified.

This iterative process continues over subsequent clock cycles, with each cycle focusing on a new set of inputs. In the third clock cycle, the comparison occurs between x3 and x2 to update the current maximum value. Similarly, in the fourth clock cycle, the comparison takes place between x1 and x0, ultimately concluding the determination of the overall maximum value among the input set. By systematically com-

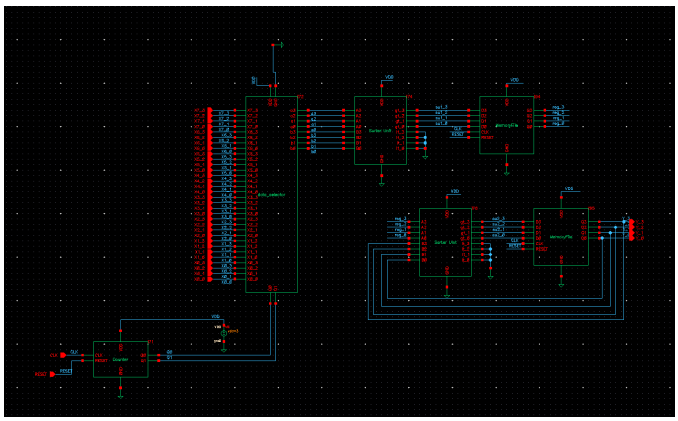

Fig. 15. Max Circuit

paring subsets of inputs across consecutive clock cycles, this mechanism ensures a thorough evaluation of the entire input range while efficiently identifying the maximum value within the dataset. This approach leverages the sorter units in tandem with registers to streamline the comparison process, enabling accurate and sequential determination of the maximum value over multiple iterations.

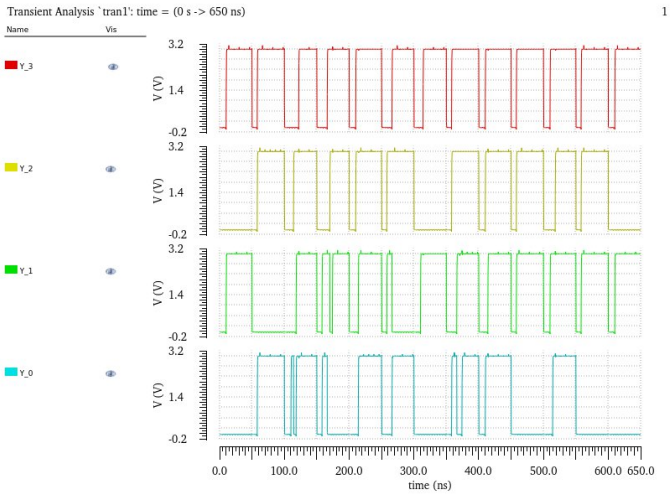

Fig. 16. Output

IV. Result

```
>> max_check('mishra_IO.csv', 3)
Congratulations, your output is error-free for all inputs

ans =

  logical

   1
```

Fig. 17. Functionaity Result