# Applying Simple CMOS Gates

Aidan Sharpe

## I. Introduction

In our previous exercises, we constructed several simple CMOS gates. Specifically, we made an inverter, a transmission gate, and a two-input NAND gate. In this exercise, we combined these simple gates together to obtain more complex behaviors.

We used transmission gates and the inverter to construct a two to one multiplexer, and we used multiple NAND gates to create a half-adder.

## II. Half Adder

A half adder is a device that has two inputs $A$ and $B$, and two outputs: the sum ($Y$) and the carry ($C$). For simplification reasons, we opted to keep our carry out inverted ($\sim C$). The relationship between the inputs and outputs is seen in table I. The sum is high when the sum of $A$ and $B$ is one, and low when the sum modulo two is zero. The carry out is only high when the sum of $A$ and $B$ is two. Therefore, the inverted carry out $\sim C$ is always high unless the sum of $A$ and $B$ is two, in which case, $\sim C$ is low.

TABLE I
Half Adder Truth Table

| $A$ | $B$ | $Y$ | $\sim C$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |

We explored several half adder designs during this exercise, and we settled on using a simple design with four NAND gates combined as shown in figure 1. We ultimately decided on this design for its combination of speed and simplicity.

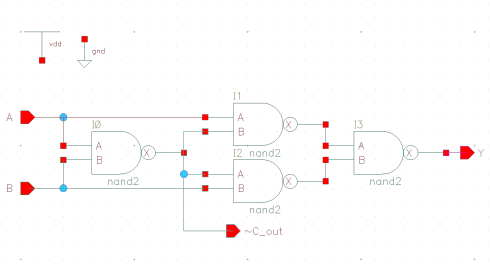


Fig. 1. Symbol-level schematic for half adder

To validate our design matches the desired behavior, we assembled a simulation that tests every combination of $A$ and $B$. This was done with two square waves, each at 50% duty cycle. The input to $A$ had a 1[ns] period of oscillation, while the input to $B$ had a 2[ns] period. Together, the signals create a two-bit binary counter. The signals from the simulation, seen in figure 2, matched the expected behavior. The teal trace, the inverted carry out $\sim C$, is only low when both $A$ and $B$ are high. Additionally, the magenta trace, the sum $Y$, is high when only one of $A$ and $B$ is high.

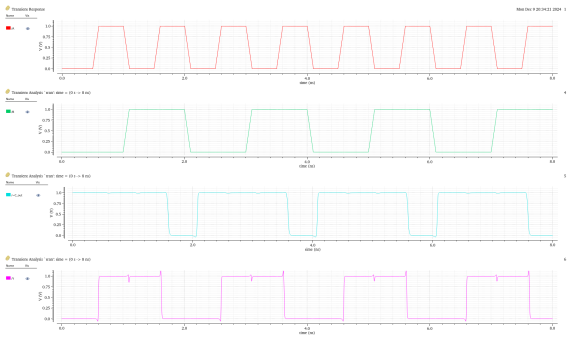

Fig. 2. Logic signals for half adder

Taking a closer look at the traces in figure 2, it is important to note that rise time for the output is actually smaller than the rise time of the inputs. This effect means that the switching time for our transistors is faster than our input.

## III. Two-to-One Multiplexer

Our two-to-one multiplexer (mux) is a signal control device. The control signal $S_0$ selects which of the two inputs $A_0$ or $A_1$ is passed to the output $Y$.

TABLE II
Two-to-one multiplexer truth table

| $S_0$ | $Y$ |
|---|---|
| 0 | $A_0$ |
| 1 | $A_1$ |

Our design used two transmission gates—one for each input—to determine if the signal should be passed through or not. We also used an inverter to send the opposite control signal to one of the transmission gates. This way, only one of the two signals would be passed through at any given time. This setup is realized with a schematic in figure 3.

Again, we tested that our design matched the expected behavior via simulation. We set up our control signal as a square wave with a 50% duty cycle and a 2[ns] period of oscillation. Since transmission gates directly pass the input to the output, we are not restricted to passing a "pure" one or zero. Therefore, to test which signal is being passed through, we set $A_0$ to 300[mV] and $A_1$ to 600[mv]. This way, as seen in figure 4, when $S_0$ is low, the output is 300[mV], and when the $S_0$ is high, the output is 600[mV].

We confirmed our schematic matches the desired behavior, so we moved on to creating a layout. To make using multiple gates together easier, we made some modifications to our initial layouts. Most importantly, we wanted all horizontal interconnects to be on metal1 and all vertical interconnects to be on metal2. Adding this rule of thumb made routing much easier. For short runs, however, we prioritized using less vias, so metal1 runs vertically in some areas. Our final layout is seen in figure 5. Another helpful modification was lining up the power rails for all designs. This modification made the rails

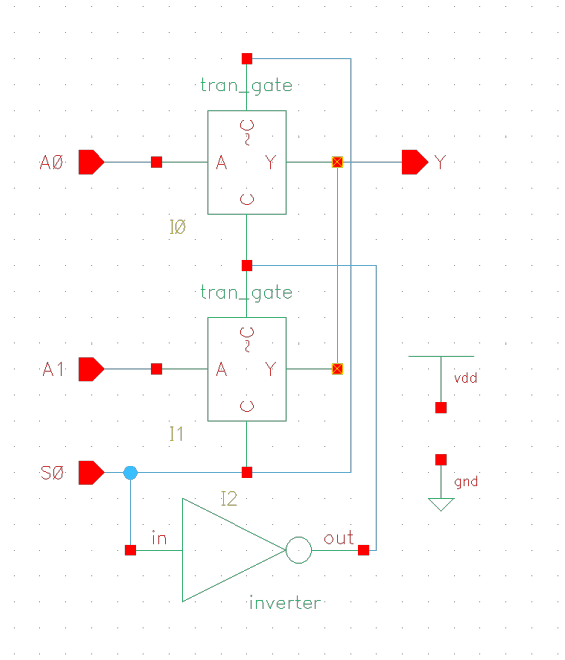


Fig. 3. Two-to-one multiplexer schematic

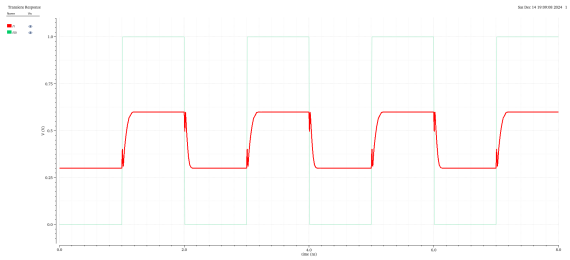

Fig. 4. Two-to-one multiplexer logic validation

clearly defined layout boundaries, and it also added additional space for routing interconnects.

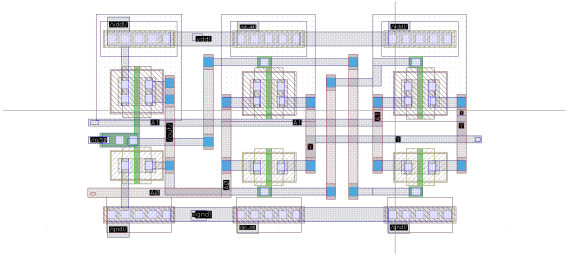

Fig. 5. Layout for two-to-one multiplexer

This process involved some unforeseen steps such as re-running DRC and LVS checks on the sub-designs, and re-spacing the elements within the sub-designs to create a more uniform top-level design.

## IV. Conclusion

This exercise proved the conclusion of the last exercise. We determined that by encapsulating fine details into lower-level designs, both higher-level schematic and layout design become much easier. For example, our half adder design used four two-input NAND gates. Each of our NAND gates contained four MOSFETs [1]. Therefore, if we were to design the half adder at a transistor level, we would have had to lay out sixteen transistors.

Going forward, we will continue to abstract our desings away from the transistor level. For example, higher-level designs may employ the half adder symbol to effectively place all sixteen transistors and their interconnects in one click.

By building up a design layer-by-layer, a highly complex design suddenly becomes attainable.

## References

[1] David Money Harris Neil H. E. Weste. *CMOS VLSI Design a Circuits and Systems Perspective, Fourth Edition.* Pearson, 2011.