

Building and Simulating an 8-bit SAR ADC in LTSpice

One of our biggest priorities in this digital environment is to interact quickly and easily with our inherited natural world. This challenge was met with a multitude of Analog to Digital Converters (ADCs) and Digital to Analog Converters (DACs). The Successive Approximation Registers Analog to Digital Converter (SAR ADC) is one of the more popular ADCs because of its high speed and resolution as well as its simple and small design [1]. The SAR ADC uses the binary search algorithm to convert an analog input into an N-bit digital output signal. A SAR ADC has four major components: the sample/hold circuit, the comparator, the SAR logic circuit, and the digital to analog converter. The SAR ADC's smaller components include a timing circuit that creates all the clock signals needed for the entire circuit as well as an output circuit that samples and outputs the final converted digital signal. The SAR ADC in this project uses 50nm technology and a 1V rail-to-rail voltage.

Other information that will be added to the introduction:

- Power input
- Specs from the testing

Tutorial Introduction:

The method in this paper will make use of sub-cell circuits; each circuit component will be built then tested separately. Once the circuit simulations are successful, a symbol is made for the circuit block then is tested once again. After each symbol is developed and combined with the next level circuit, the final SAR ADC will be made of only six circuit blocks. This hierarchy makes building and debugging much simpler for large scale circuits.

To start, these first circuits will need to be created:

1. An inverter
2. 3-input NAND logic gate
3. 2-input AND logic gate

Then, our next components to make are:

1. Inverter Buffer
2. Texas Instruments D Flip-Flop (TI DFF)
3. Comparator Biasing circuit

Finally, the last components we will build are:

1. SAR logic block
2. R2R DAC
3. Sample/Hold circuit
4. Comparator circuit
5. Output Door
6. Clocks Circuit

Once all components are finished and successfully simulated the SAR ADC will be finished with these last six components. Before starting the tutorial, the following model document needs to be downloaded and placed in the same folder that the components will be housed. [models.txt](#)

Tutorial Instructions:

Part 1.

The first step will be to create the inverters and logic gates. The LTSpice component models used for the PMOS is "pmos4" and the NMOS is "nmos4". Each PMOS and NMOS used in this circuit will have the same specifications to coordinate with the models.txt file. All PMOS transistors need to be named "P_50n" and edited to have a length of "50n" and a width of "1u". All NMOS transistors need to be named "N_50n" and have a length of "50n" and a width of "0.5u". These settings can be accessed by right clicking each component symbol. The finished inverter circuit can be seen in the following figures.

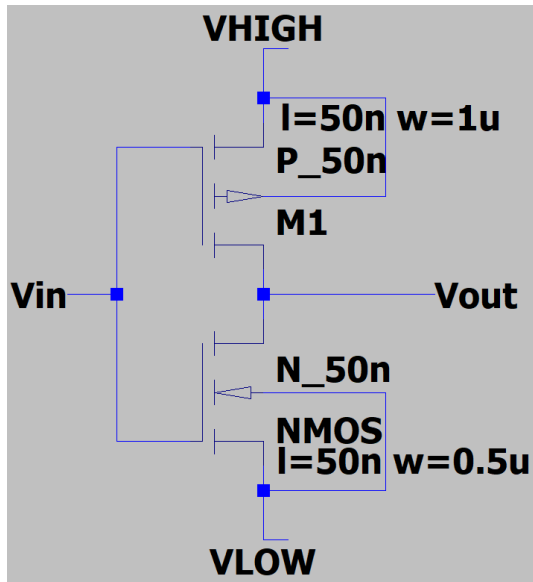


Figure 1.1 (left). Inverter logic gate comprised of one PMOS transistor on top and one NMOS transistor on the bottom. Note, PMOS and NMOS are connected drain to drain.

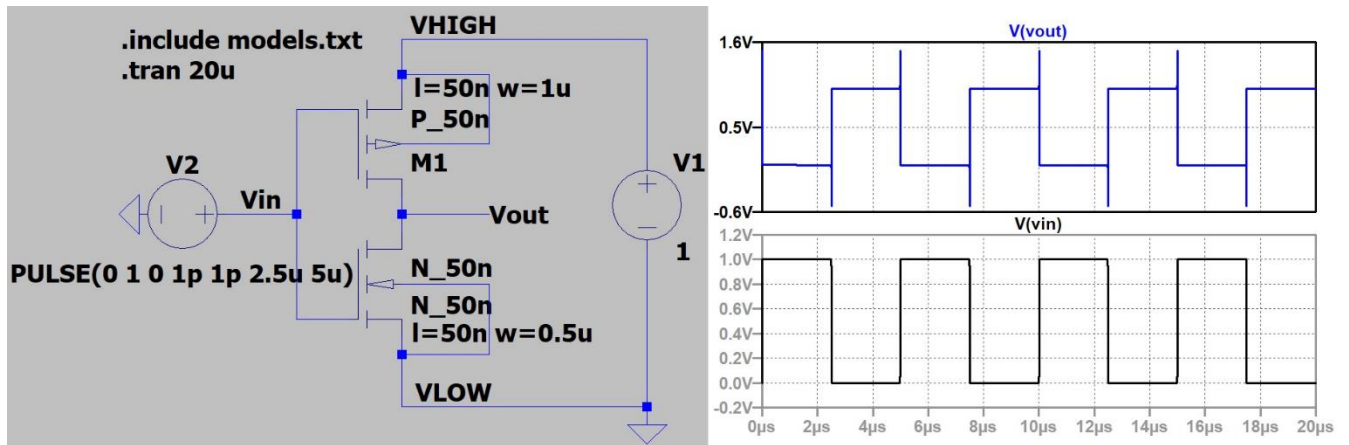


Figure 1.2. Successful simulation of the inverter circuit. Vout is the opposite of Vin.

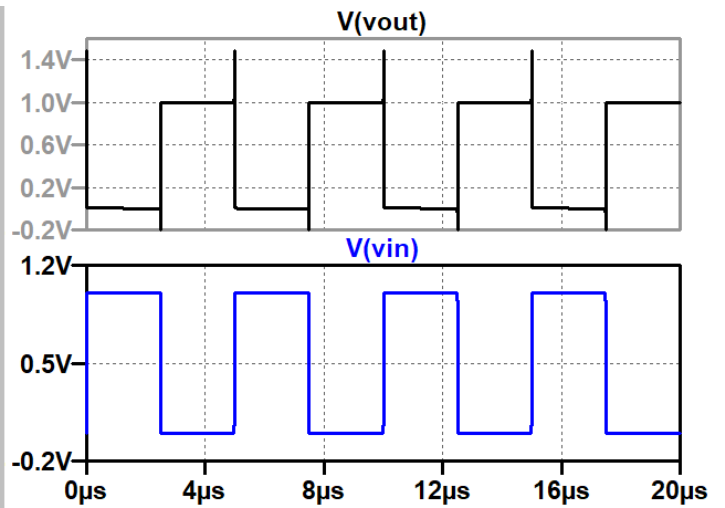
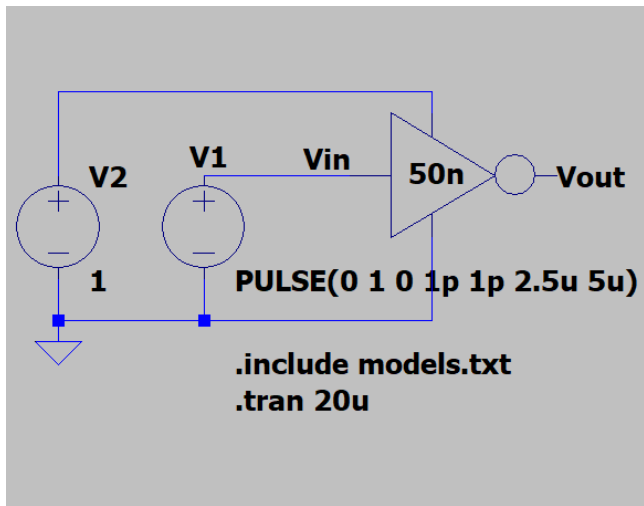


Figure 1.3. Successful inverter symbol simulation.

Now that the inverter is fully functional, the NAND and AND gates can be created using the same steps.

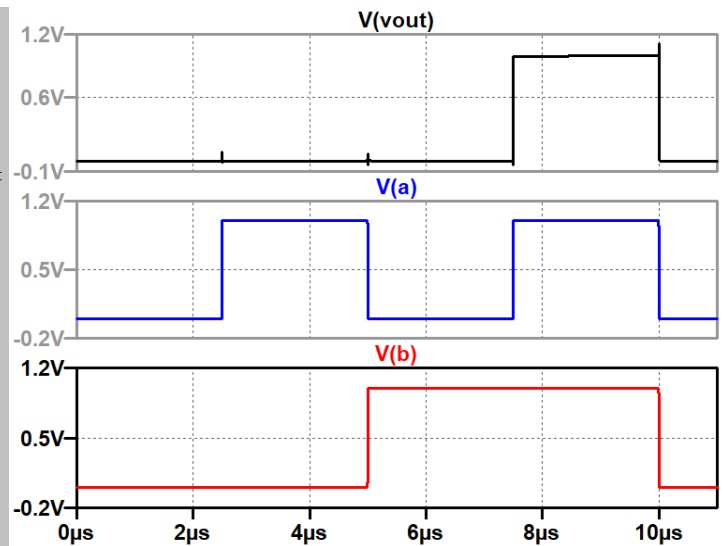
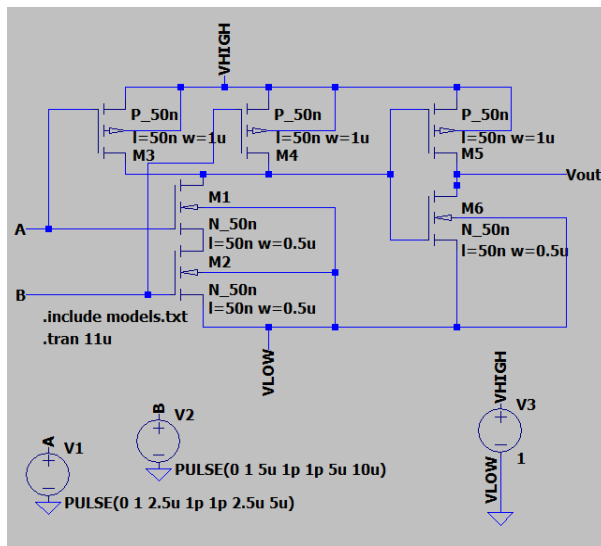


Figure 2.1. AND logic gate circuit simulation. The simulation lets us see all possible inputs. Vout will only go to 1 when both the inputs are 1.

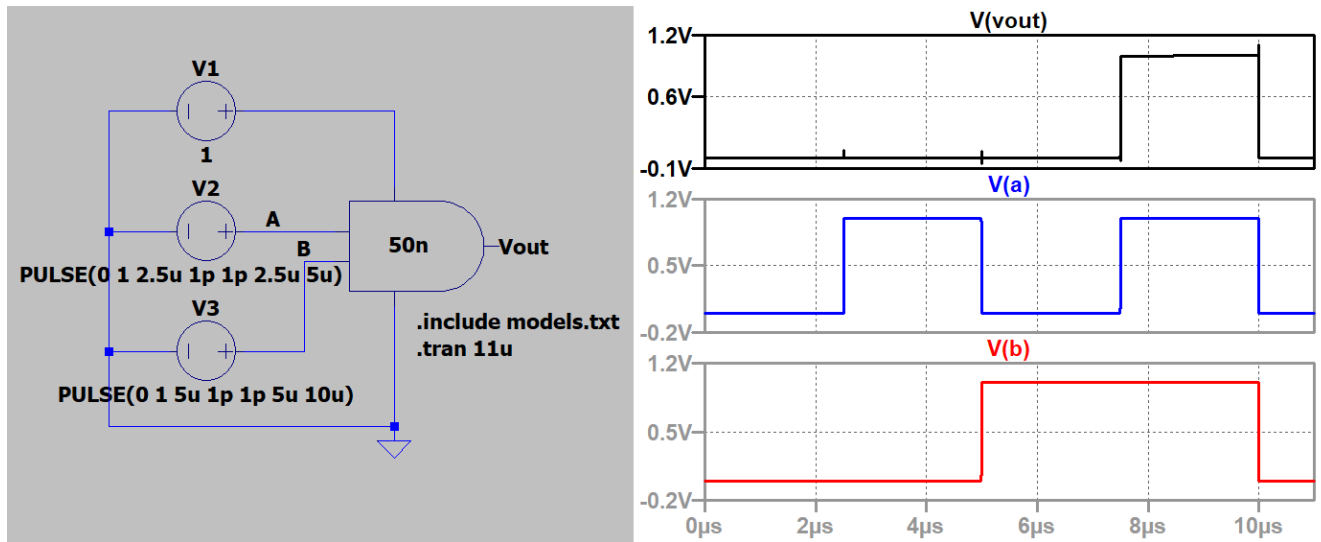


Figure 2.2. AND symbol simulation results.

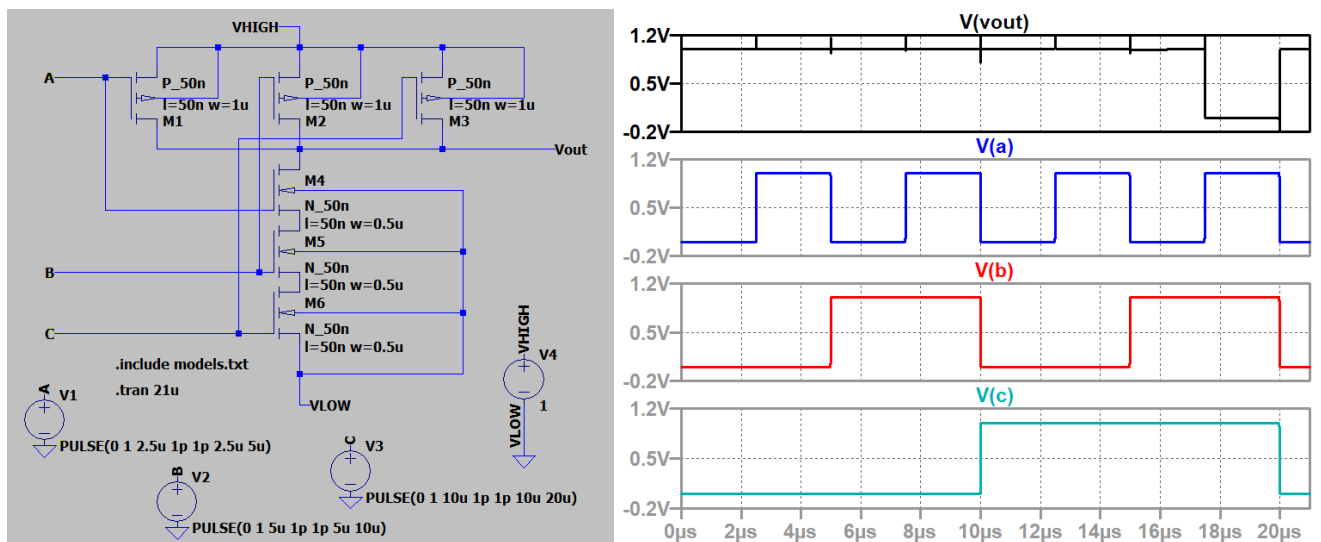


Figure 3.1. Three input NAND logic gate circuit simulation. Using the same counting method, all possible inputs can be seen from 0-0-0 to 1-1-1.

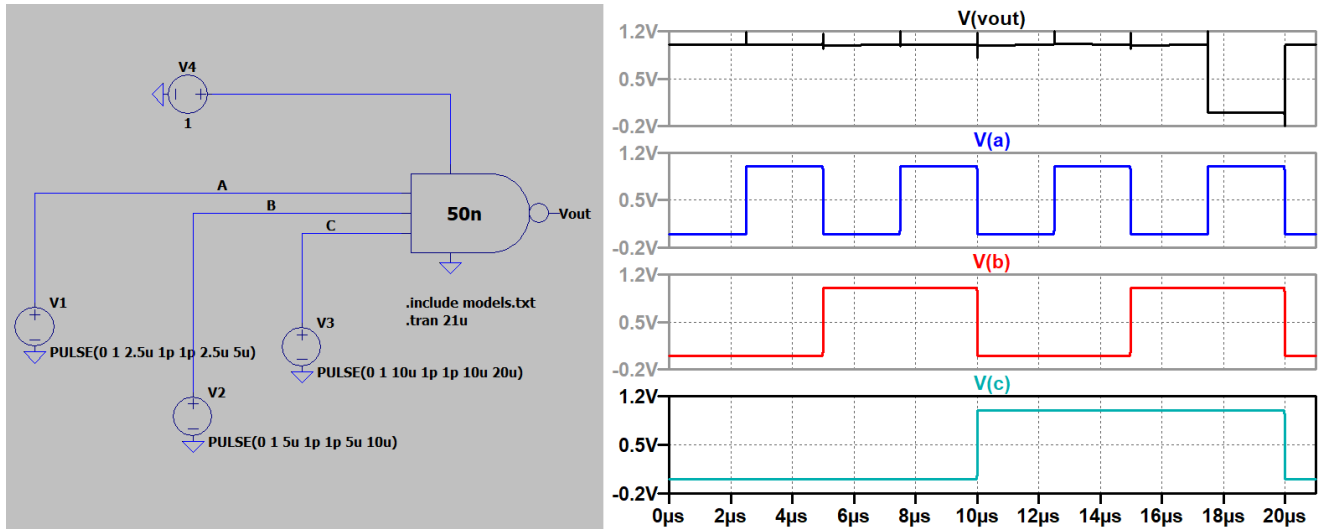


Figure 3.2. Three input NAND symbol simulation.

Part 2.

Now that the lower-level circuits have been built and successfully simulated, the next level circuits can be created. At this point we will build the inverter buffers, TI DFF, and the Comparator biasing circuit. These mid-level circuits will be the final building blocks for our SAR ADC's six circuit components.

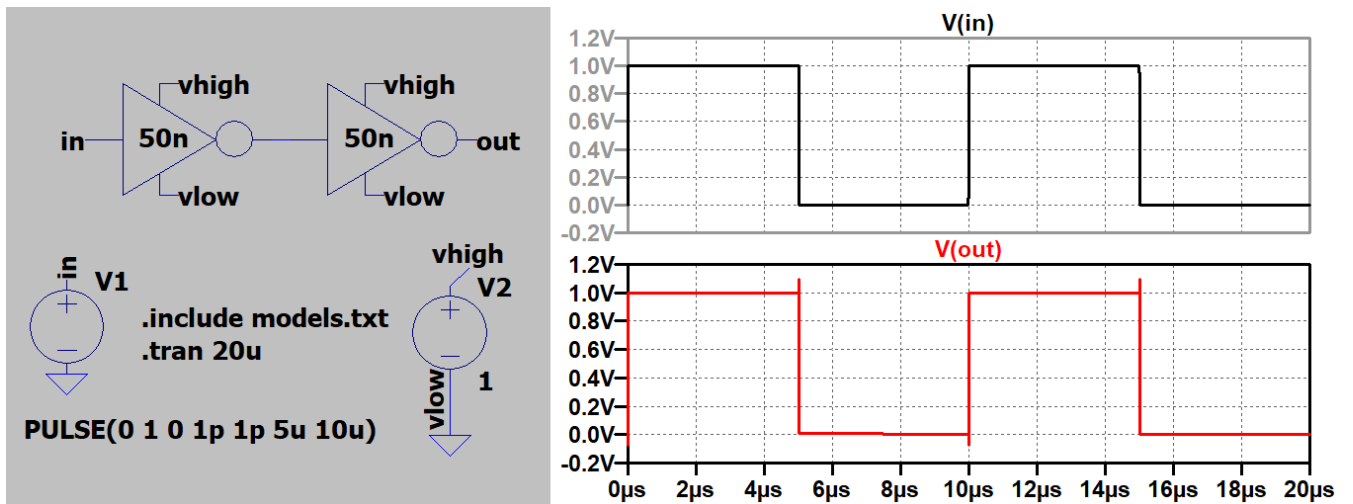


Figure 4.1. The buffer is used to prevent hold time violations in the circuit. The input and output are the same values.

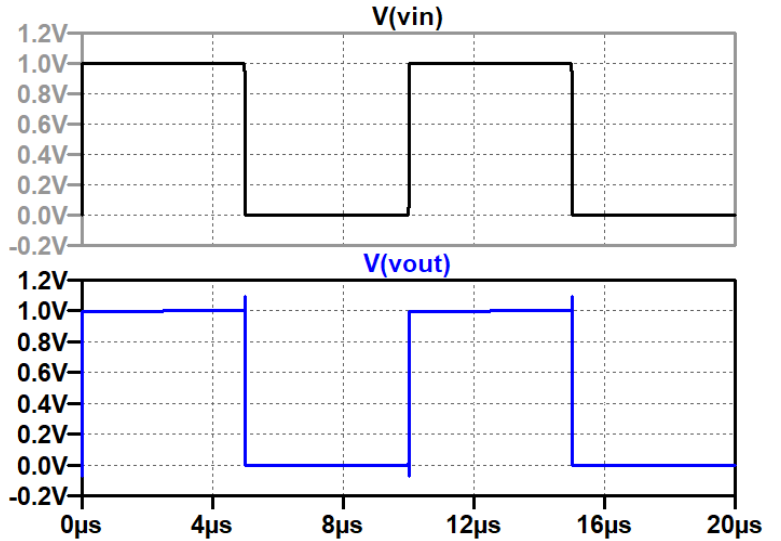
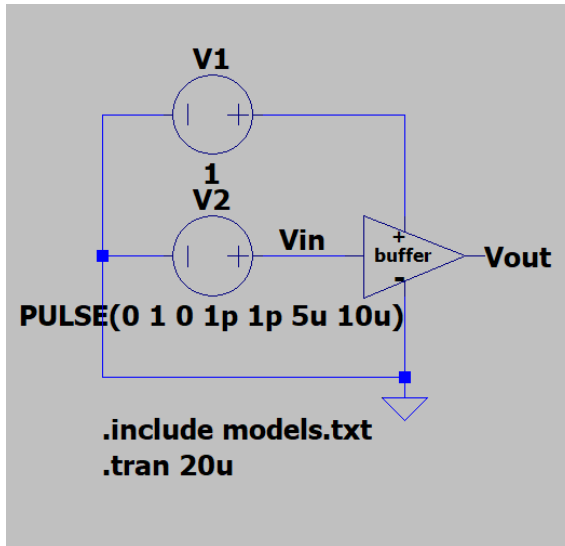


Figure 4.2. The buffer symbol simulation. The symbol is marked with the word “buffer” so as not to get it confused with the inverter or comparator. The “vhigh” and “vlow” outputs are also marked in case the buffer is rotated or mirrored with future use.

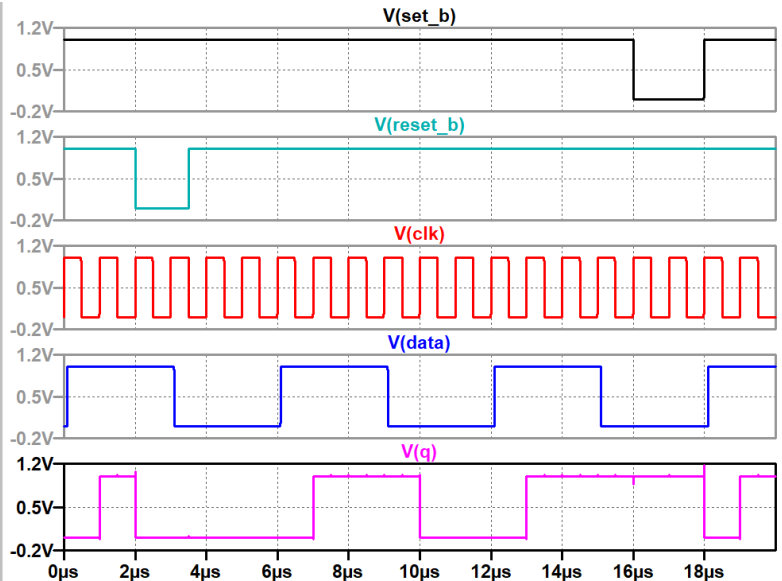
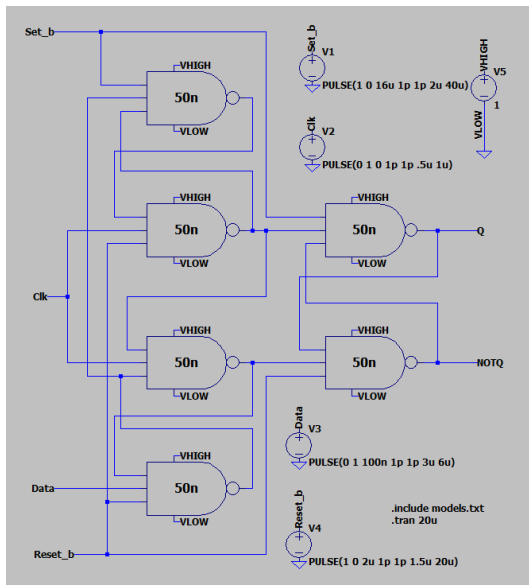


Figure 5.1. TI DFF circuit simulation. The data signal is delayed slightly so that the data signal is distinct from the rising edge clock signal. The reset function can be seen in the first 4us. Normal Flip Flop functions can be seen from 4us to 12us. The set function can be seen in the last 6us.

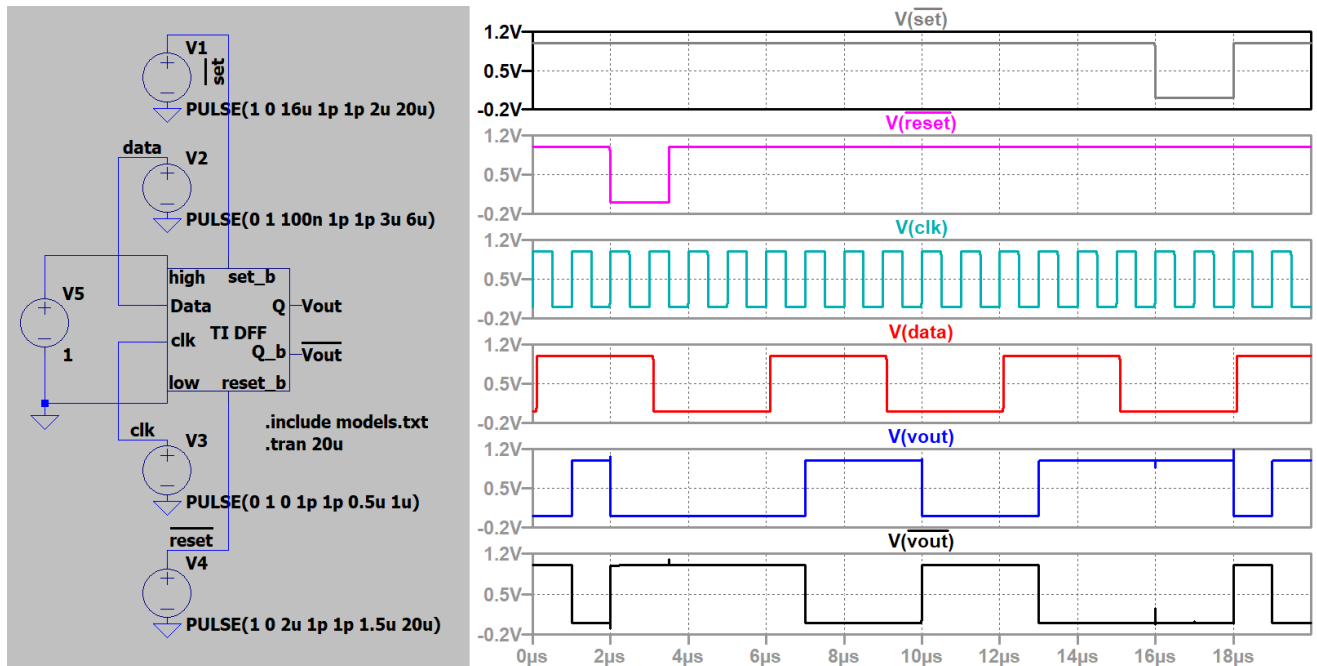
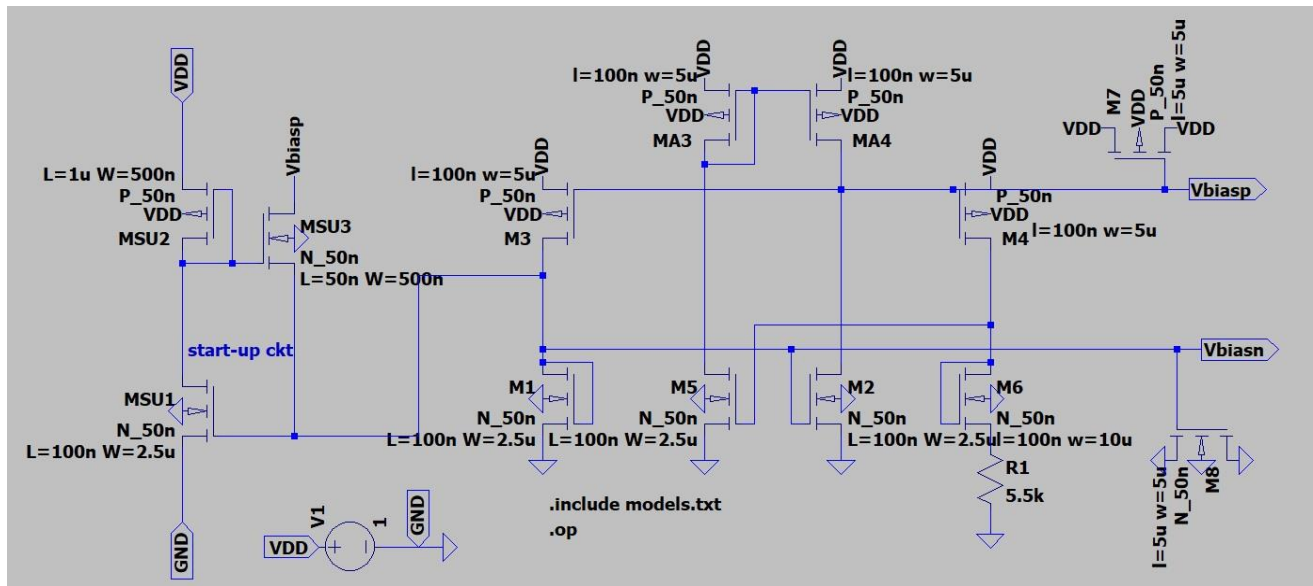


Figure 5.2. Successful TI DFF symbol simulation. The Q not output is also seen here.

The comparator biasing circuit was provided by Dr. Li who had retrieved it from Dr. Jacob Baker's book [1]. This circuit can be found in figure 20.22 on page 629. In the following figure, the operating point simulation is provided for a full view of the circuit.



```

--- Operating Point ---
V(n002) : 0.0339458 voltage
V(vdd) : 1 voltage
V(vbiasn) : 0.353762 voltage
V(vbiasp) : 0.651916 voltage
V(n003) : 0.354512 voltage
V(n004) : 0.0553655 voltage
V(n001) : 0.636762 voltage
V(m6#tbody) : 3.07481e-012 voltage
V(m6#tbody) : 8.3024e-013 voltage
V(m2#tbody) : 5.5295e-012 voltage
V(m2#tbody) : 6.38069e-013 voltage
V(m8#tbody) : 3.95579e-012 voltage
V(m8#tbody) : 3.95579e-012 voltage
V(m5#tbody) : 5.40122e-012 voltage
V(m5#tbody) : 6.23448e-013 voltage
V(m1#tbody) : 3.00114e-012 voltage
V(m1#tbody) : 3.45861e-013 voltage
V(msu3#tbody) : 5.83271e-012 voltage
V(msu3#tbody) : 3.59656e-012 voltage
V(msu1#tbody) : 2.93018e-013 voltage
V(msu1#tbody) : 3.69178e-014 voltage
V(m7#tbody) : 1 voltage
V(m7#tbody) : 1 voltage
V(ma3#tbody) : 1 voltage
V(ma3#tbody) : 1 voltage
V(ma4#tbody) : 1 voltage
V(ma4#tbody) : 1 voltage
V(m4#tbody) : 1 voltage

V(m4#tbody) : 1 voltage
V(m3#tbody) : 1 voltage
V(m3#tbody) : 1 voltage
V(msu2#tbody) : 1 voltage
V(msu2#tbody) : 1 voltage
Id(M6) : 1.00661e-005 device_current
Ig(M6) : 3.40335e-010 device_current
Ib(M6) : -4.23424e-013 device_current
Is(M6) : -1.00665e-005 device_current
Id(M2) : 1.18632e-005 device_current
Ig(M2) : 1.25612e-010 device_current
Ib(M2) : -6.66399e-013 device_current
Is(M2) : -1.18633e-005 device_current
Id(M8) : -1.14075e-008 device_current
Ig(M8) : 2.2817e-008 device_current
Ib(M8) : -2.10976e-012 device_current
Is(M8) : -1.14075e-008 device_current
Id(M5) : 1.19247e-005 device_current
Ig(M5) : 1.32544e-010 device_current
Ib(M5) : -6.51024e-013 device_current
Is(M5) : -1.19248e-005 device_current
Id(M1) : 1.00454e-005 device_current
Ig(M1) : 1.65903e-010 device_current
Ib(M1) : -3.61478e-013 device_current
Is(M1) : -1.00455e-005 device_current
Id(M3) : 8.1014e-011 device_current
Ig(M3) : -8.99622e-011 device_current
Ib(M3) : -1.00579e-012 device_current
Is(M3) : 9.95393e-012 device_current
Id(Ms1) : 5.34067e-006 device_current

Ig(Ms1) : 1.97873e-010 device_current
Ib(Ms1) : -3.67629e-014 device_current
Is(Ms1) : -5.34087e-006 device_current
Id(M7) : 1.37197e-009 device_current
Ig(M7) : -2.74777e-009 device_current
Ib(M7) : 3.83574e-012 device_current
Is(M7) : 1.37197e-009 device_current
Id(Ma3) : -1.19216e-005 device_current
Ig(Ma3) : -1.54745e-009 device_current
Ib(Ma3) : 3.82955e-013 device_current
Is(Ma3) : 1.19231e-005 device_current
Id(Ma4) : -1.18606e-005 device_current
Ig(Ma4) : -1.55065e-009 device_current
Ib(Ma4) : 3.67476e-013 device_current
Is(Ma4) : 1.18621e-005 device_current
Id(M4) : -1.00666e-005 device_current
Ig(M4) : 9.65524e-012 device_current
Ib(M4) : 6.66484e-013 device_current
Is(M4) : 1.00666e-005 device_current
Id(M3) : -1.00687e-005 device_current
Ig(M3) : 1.82283e-011 device_current
Ib(M3) : 6.67248e-013 device_current
Is(M3) : 1.00687e-005 device_current
Id(Msu2) : -5.32782e-006 device_current
Ig(Msu2) : -1.2764e-008 device_current
Ib(Msu2) : 9.00347e-009 device_current
Is(Msu2) : 5.33158e-006 device_current
I(R1) : 1.00665e-005 device_current
I(V1) : -4.92638e-005 device_current

```

Figure 6.1. Comparator Biasing Circuit with operating point (".op") simulation results.

Part 3.

For this part of the project, we will be creating the last circuit blocks needed for the final SAR ADC. In this section we will build the SAR Logic block, the R2R DAC, the Sample/Hold circuit, the output door circuit, and the clocks circuit.

The SAR Logic schematic is from an outside source [3]. The Logic was used in a SAR ADC microchip, so it has buffers to prevent a real-world issue called hold time violation. These hold violations will interfere with a microchip's timing and will cause this ADC process to become defective.

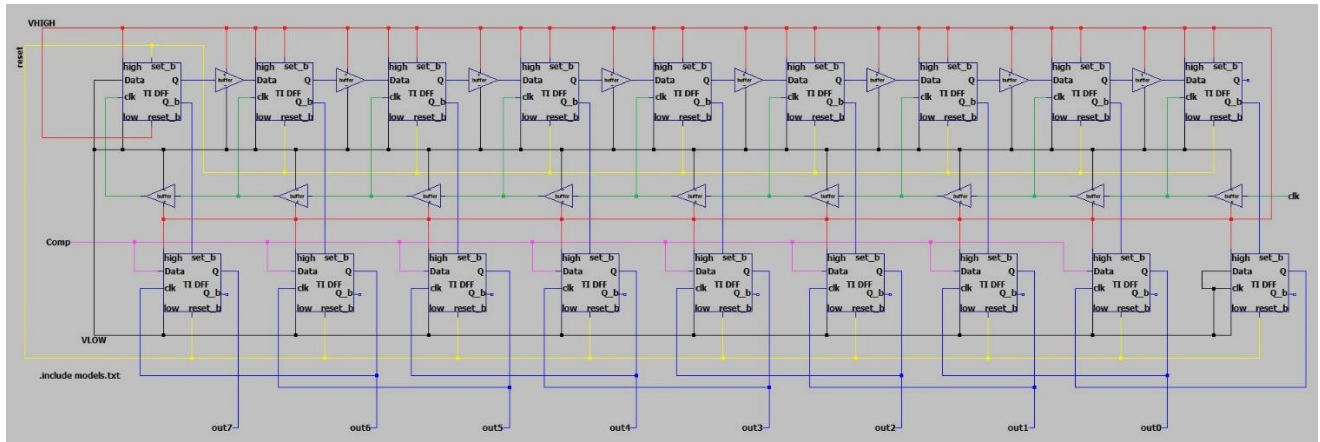


Figure 7.1. SAR Logic circuit block color coded for easier referencing. All “VHIGH” is red, “VLOW” is black, “reset” is yellow, “clk” is green and “Comp” is purple/pink. Keep in mind, the TI DFFs have the “set_not” and “reset_not” outputs so this may look different than a traditional SAR Logic circuit.

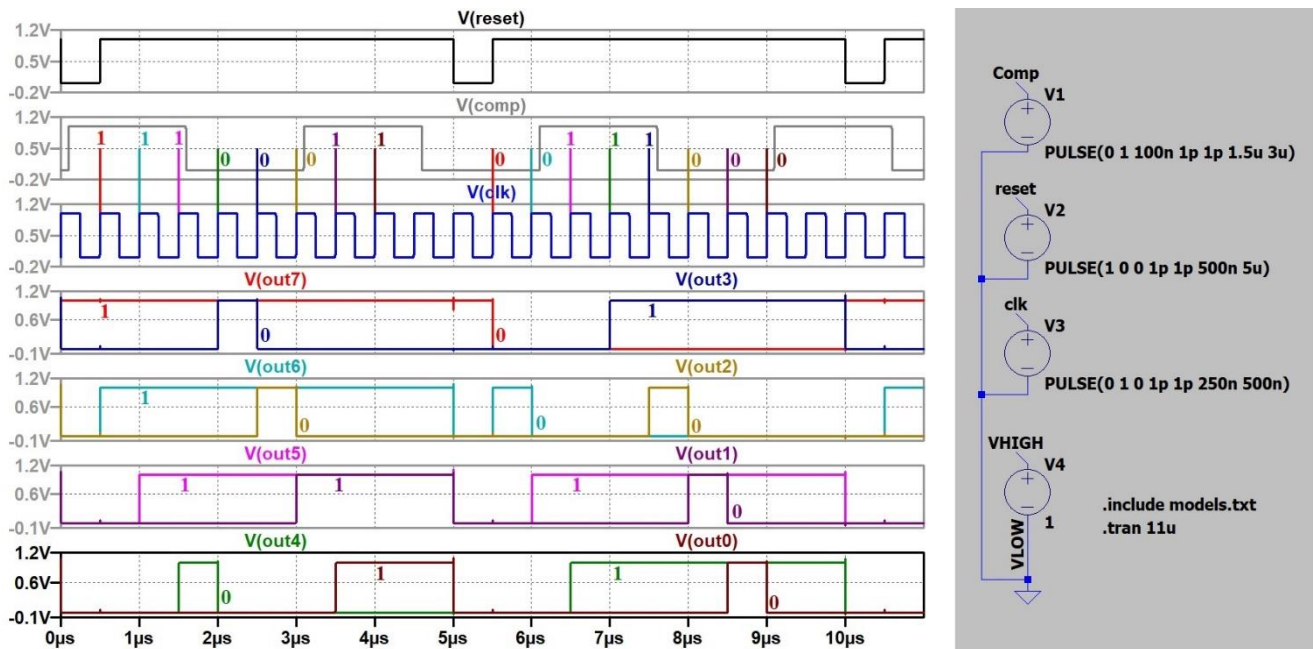


Figure 7.2. SAR Logic Circuit simulation. Because the circuit is so large this figure only contains the input voltages of the circuit. The inputs are meant to correlate to the labels in figure 7.1. The results to look for with this simulation can be seen in the colored vertical lines. The output sampling is triggered by the following TI DFF’s Q rising edge (as seen in the blue lines of figure 7.1). So, each signal is set at the moment the following TI DFF gets its initial high value.

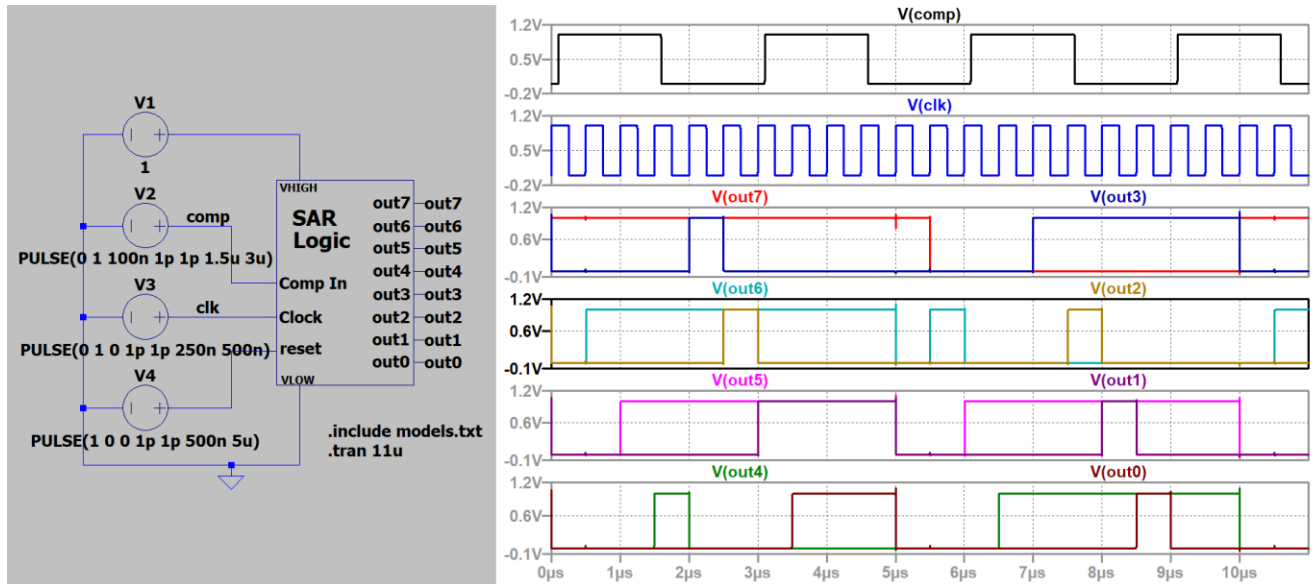


Figure 7.3. SAR Logic symbol simulation. The input signals are the same as those in figure 7.2.

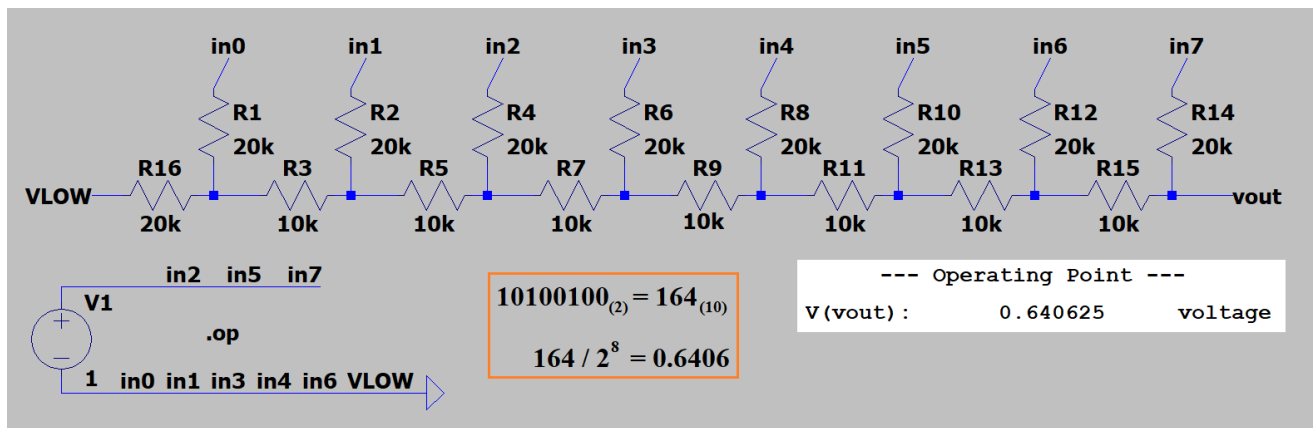


Figure 8.1. R2R DAC circuit simulation with correct voltage output. The orange box shows the binary to decimal conversion and the calculation for the analog value. Vref here is V1=1V.

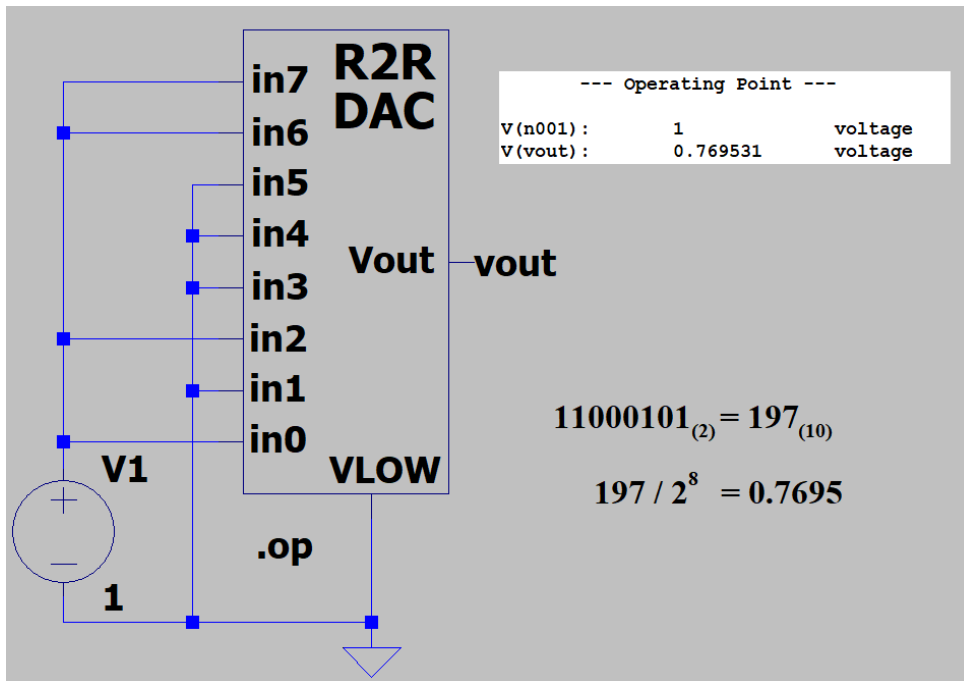


Figure 8.2. R2R ladder symbol simulation and verification calculations.

The following Sample/Hold circuit was taken from [2].

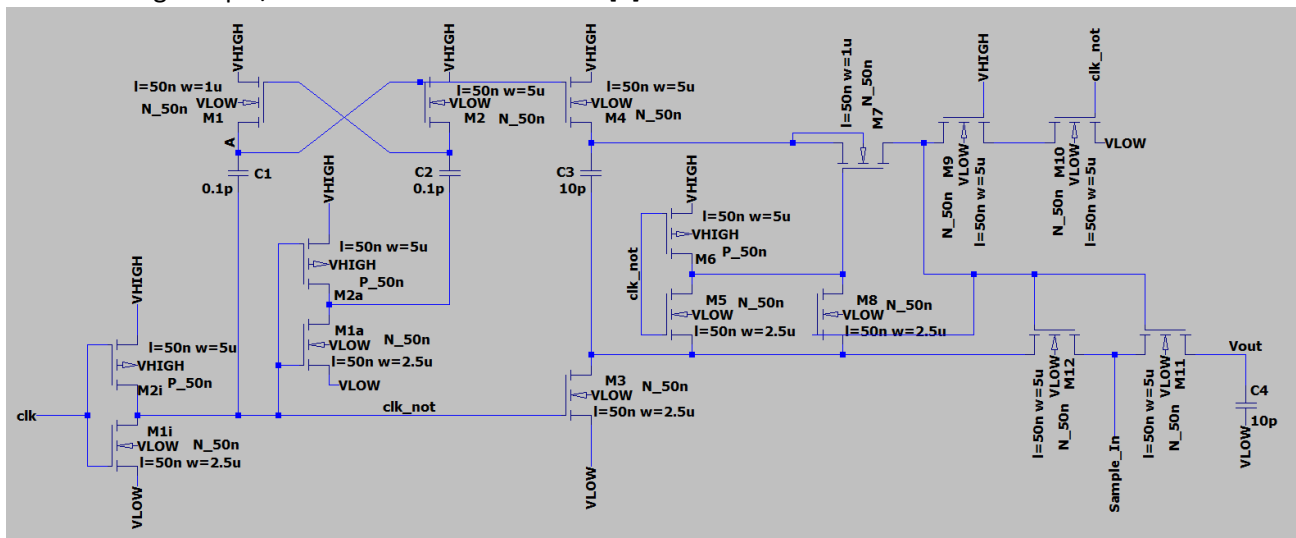


Figure 9.1. Abo's Sample and Hold Circuit.

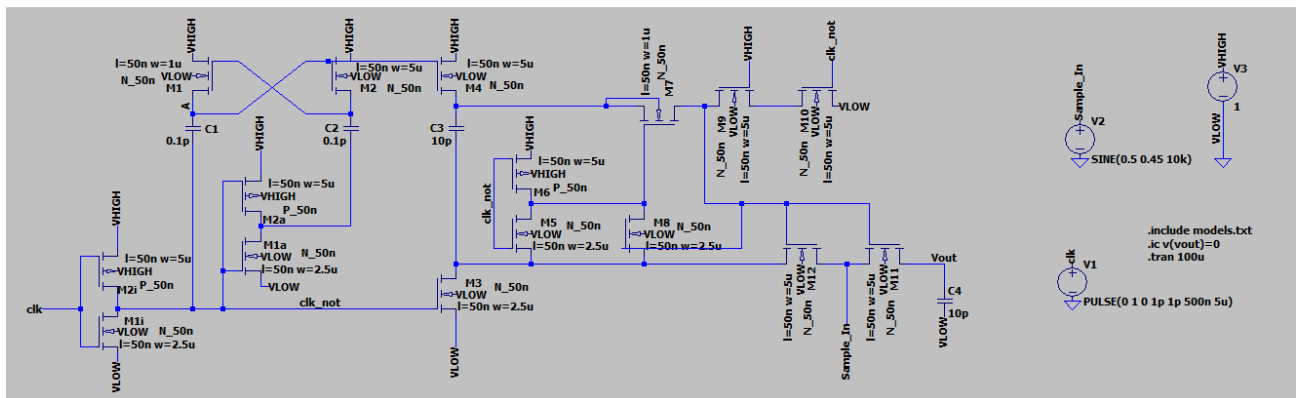
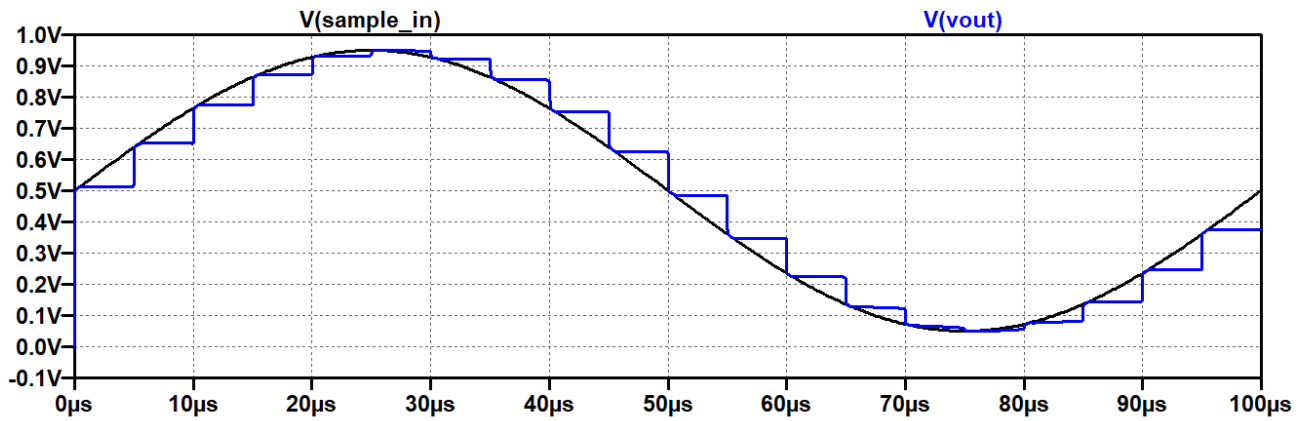


Figure 9.2. Sample/Hold circuit simulation. The input values are the signals that will be used for the SAR ADC. The “Sample_in” signal will be the SAR ADC’s V_{in} signal.

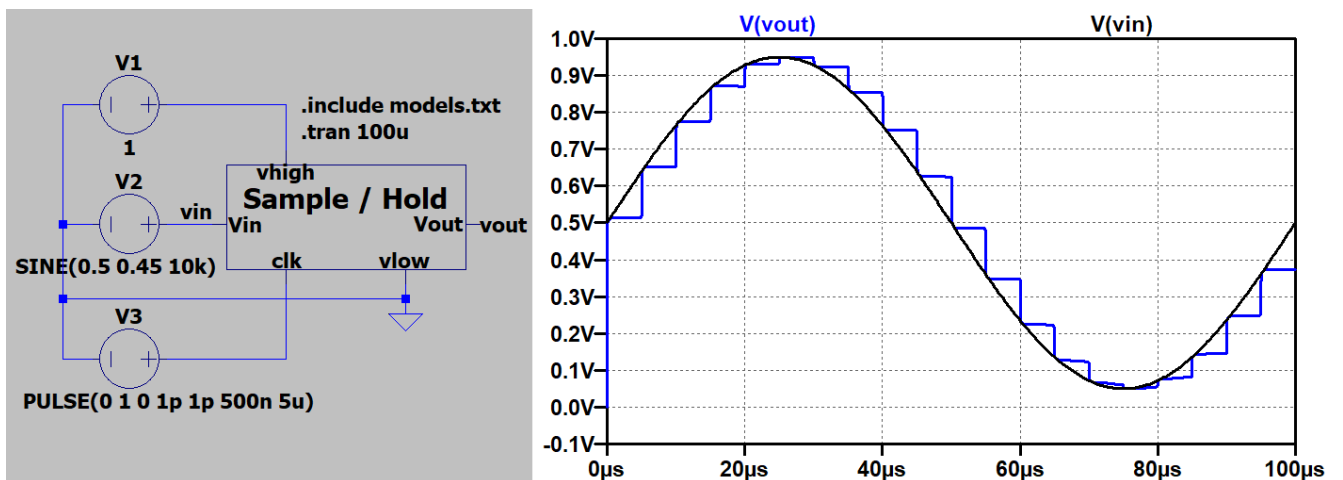


Figure 9.3. Sample/Hold simulation with the symbol.

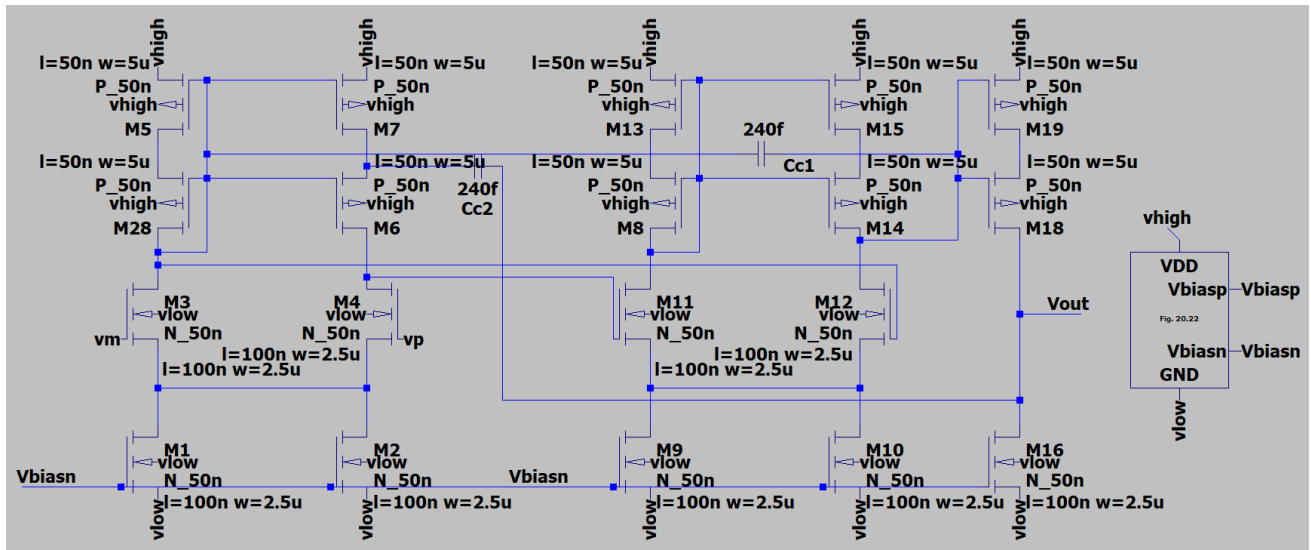


Figure 10.1. Comparator schematic. This comparator is also from Dr. Baker's CMOS book [1]. This circuit can be found on page 820 in figure 24.61.

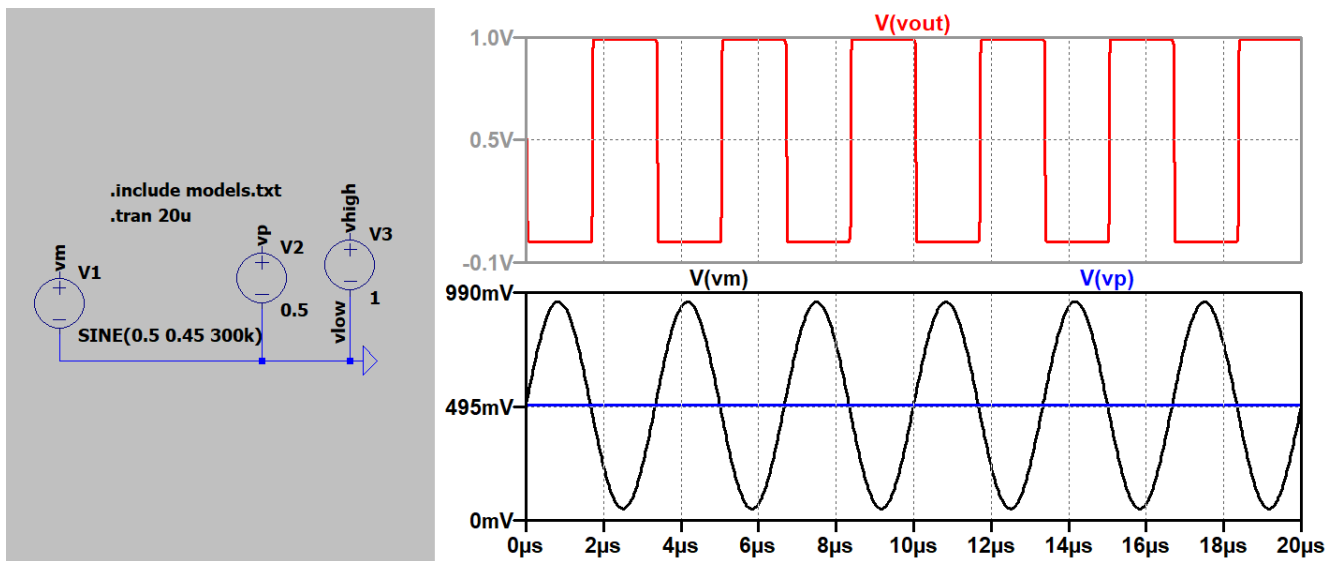


Figure 10.2. Comparator circuit simulation with just the voltage input signals shown. When $v_p < v_m$ then $v_{out} = 0$, and when $v_p > v_m$, $v_{out} = 1$.

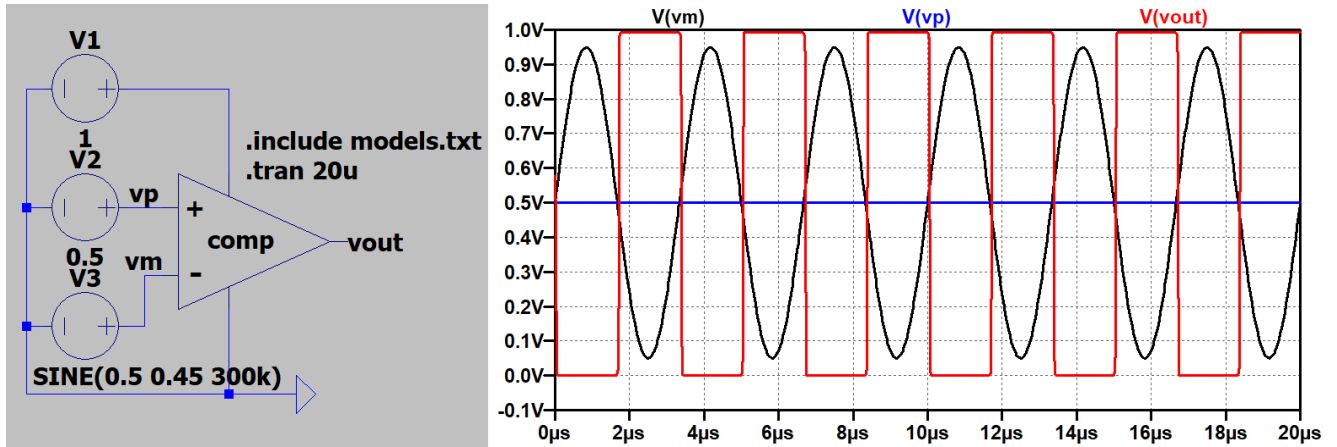


Figure 10.3. Comparator symbol simulation check.

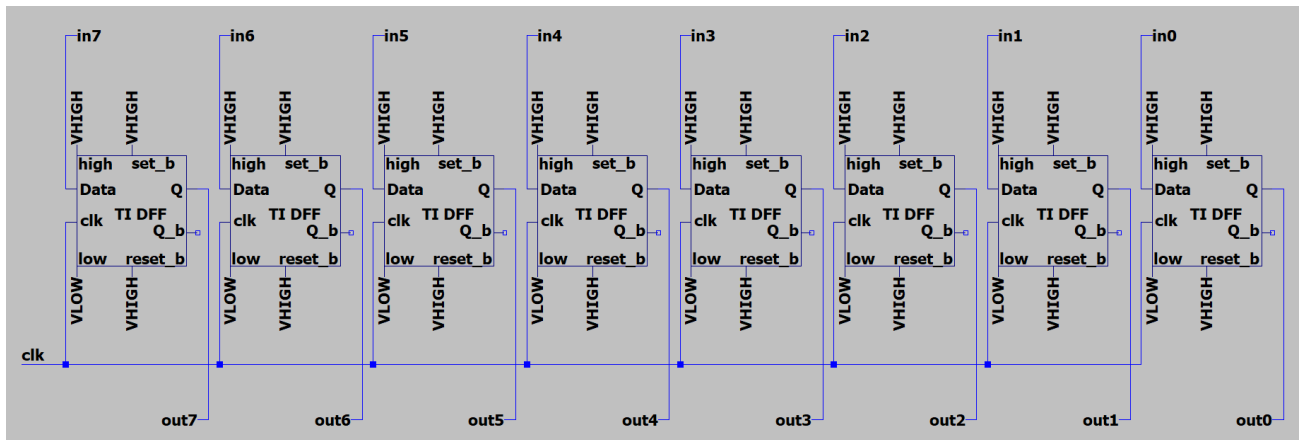


Figure 11.1. Parallel in parallel out shift register Output Door circuit schematic. This circuit is used to sample the output right after the conversion reset cycle is complete and right before the SAR Logic is reset for the next cycle.

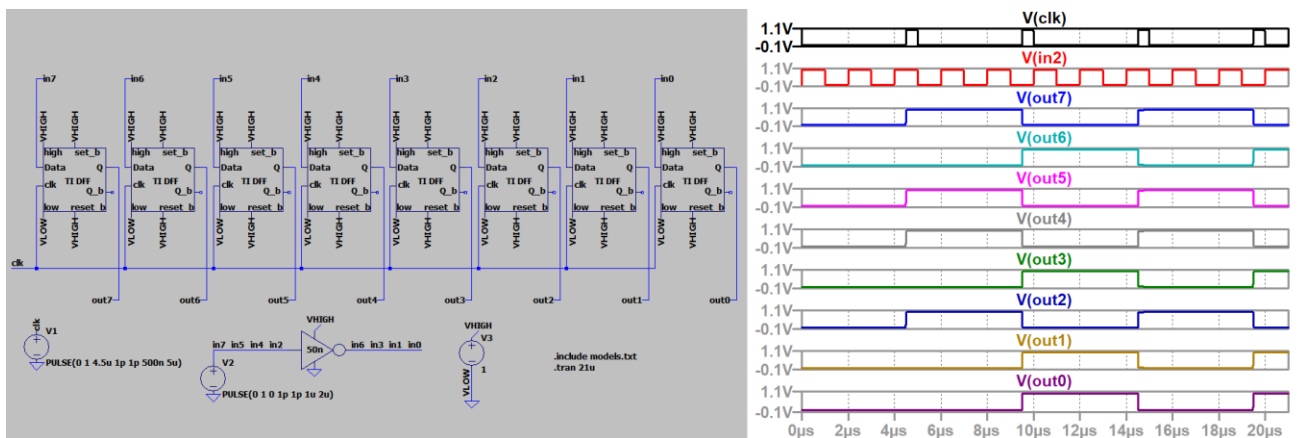


Figure 11.2. Circuit simulation for the output door. I added the inverter just to get some variety in the outputs. As you can see, the input can vary and is only sent to the output when the clock signal has a rising edge.

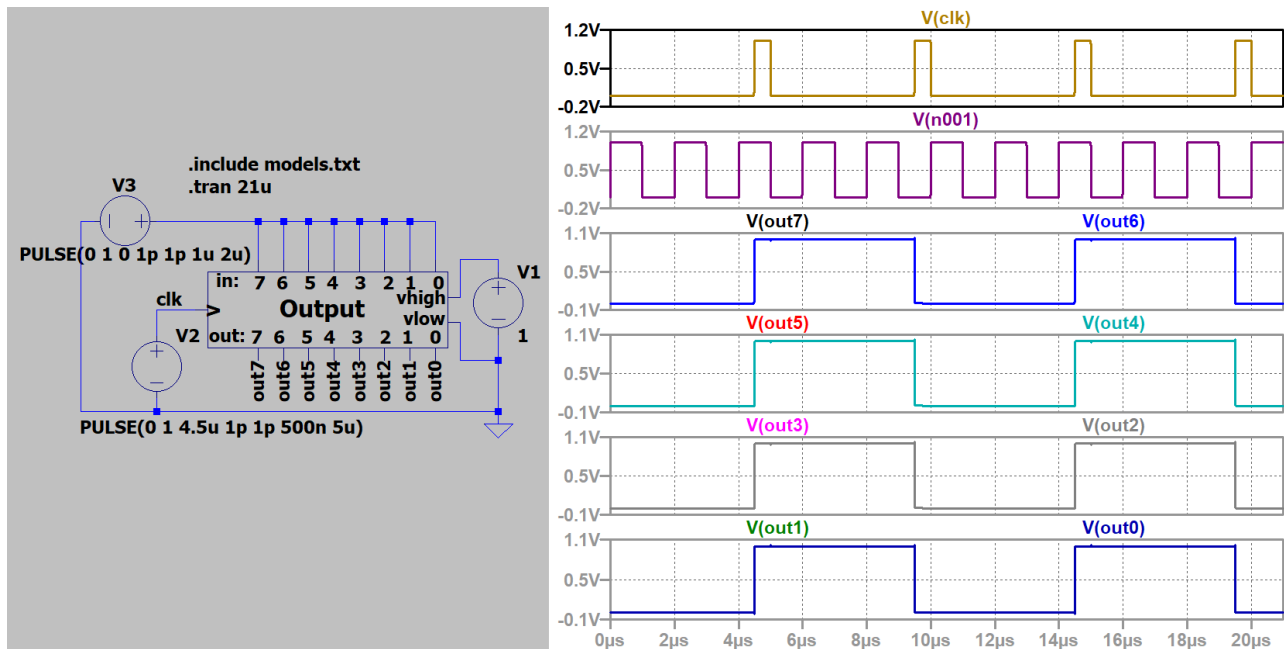


Figure 11.3. Output door symbol simulation results. The outputs were doubled up for a better view.

The last component that we need to make is the clocks circuit. This circuit will take an input of 4MHz and output all the clock signals we need for the SAR ADC circuit. These signals include the 2MHz 50% duty cycle clock signal for the SAR Logic block, the 200kHz 10% duty cycle Sample/Hold clock signal, the 200kHz 10% duty cycle inverted SAR Logic Reset signal, and the 200kHz 10% duty cycle Output Door clock signal. The Sample/Hold signal is an inversion of the Reset signal.

The circuit uses a 4-bit counter and is reset when the counter reaches the number ten which, in binary, is the value "1010". This permits 10 SAR Logic periods (0-9) for the conversion process. The first period is when the Sample/Hold and Reset signals are active and is used to set the initial "1000 0000" values for the SAR Logic block. The next nine cycles are used by the SAR Logic block to determine the binary value. The Output Door will then sample and output the digital values. The clocks circuit contains two main parts, the counter circuit and the logic gates used to detect specific binary values.

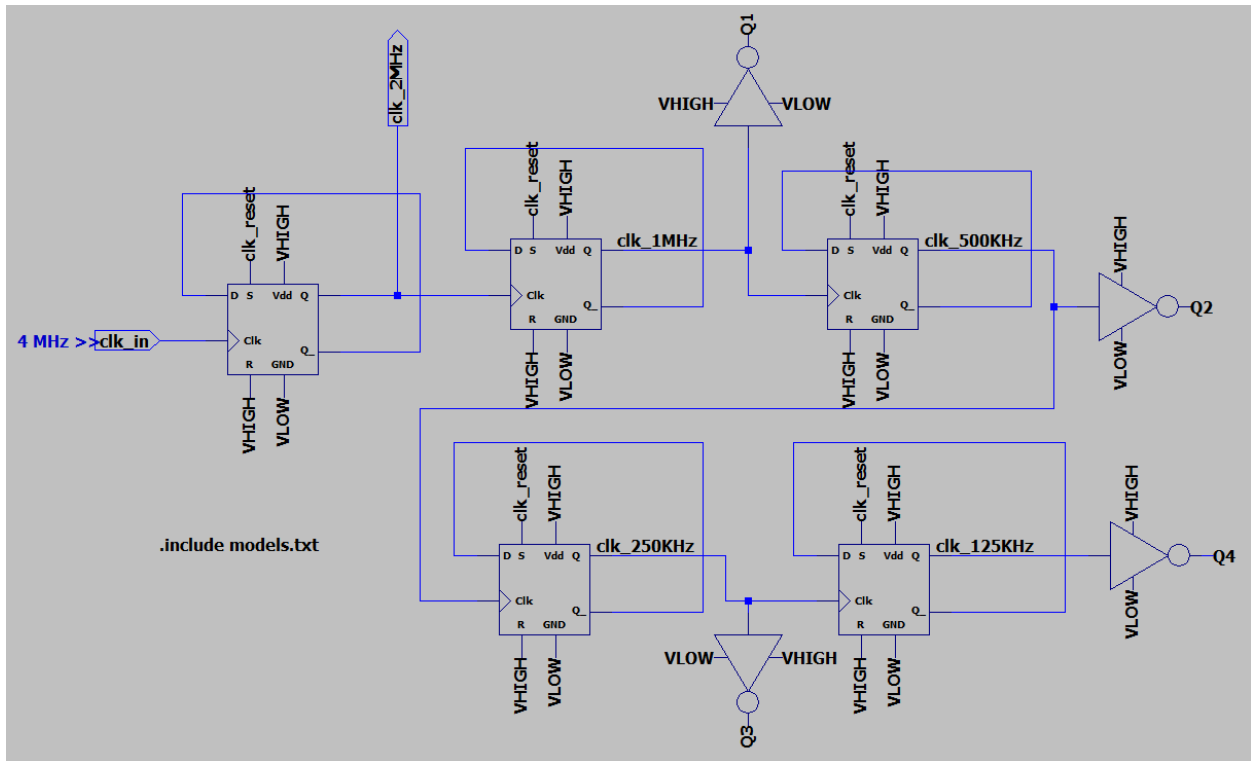


Figure 12.1. Counter portion of the clocks circuit. The output of each counter divides the input frequency signal by 2. In this figure, each output is labeled with its new frequency.

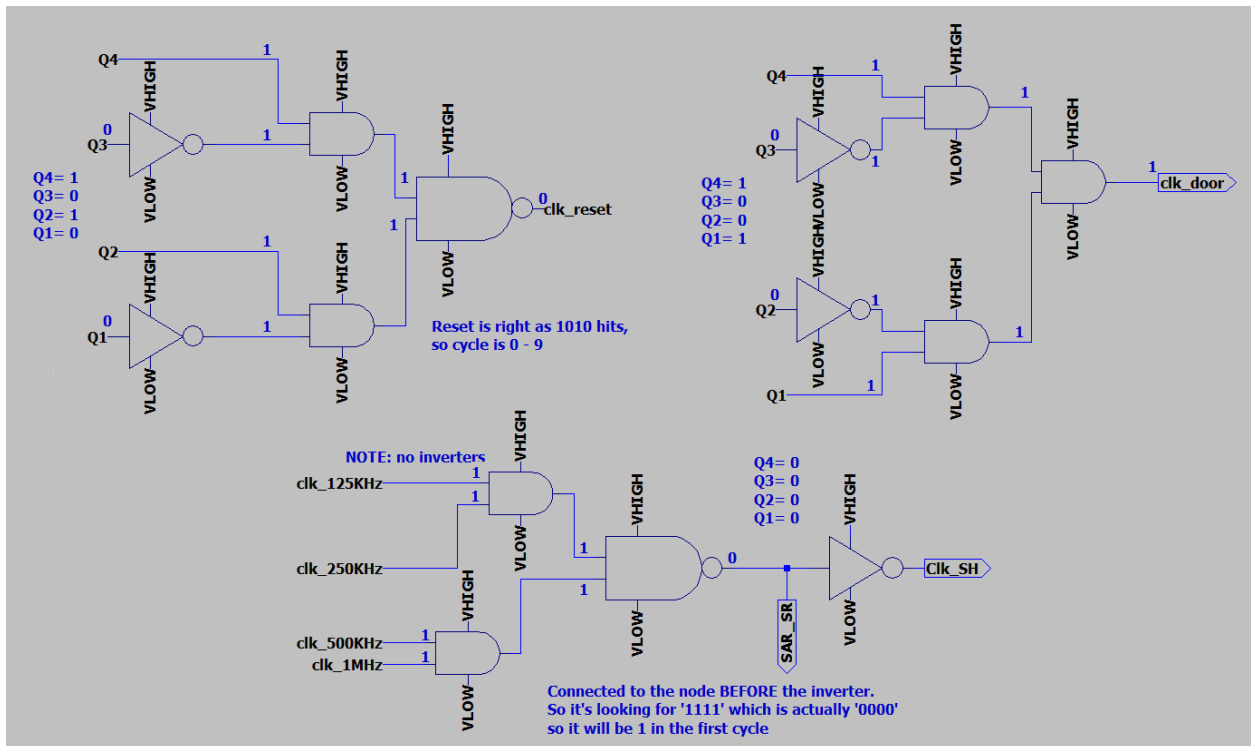


Figure 12.2. Logic gate portion of the clocks circuit. The Q labels show the output that each logic setup is seeking out from the counter. The bottom circuit's input references the TI DFF output nodes before the inverter, to keep the circuit simple. Otherwise there would be much more inverters than needed.

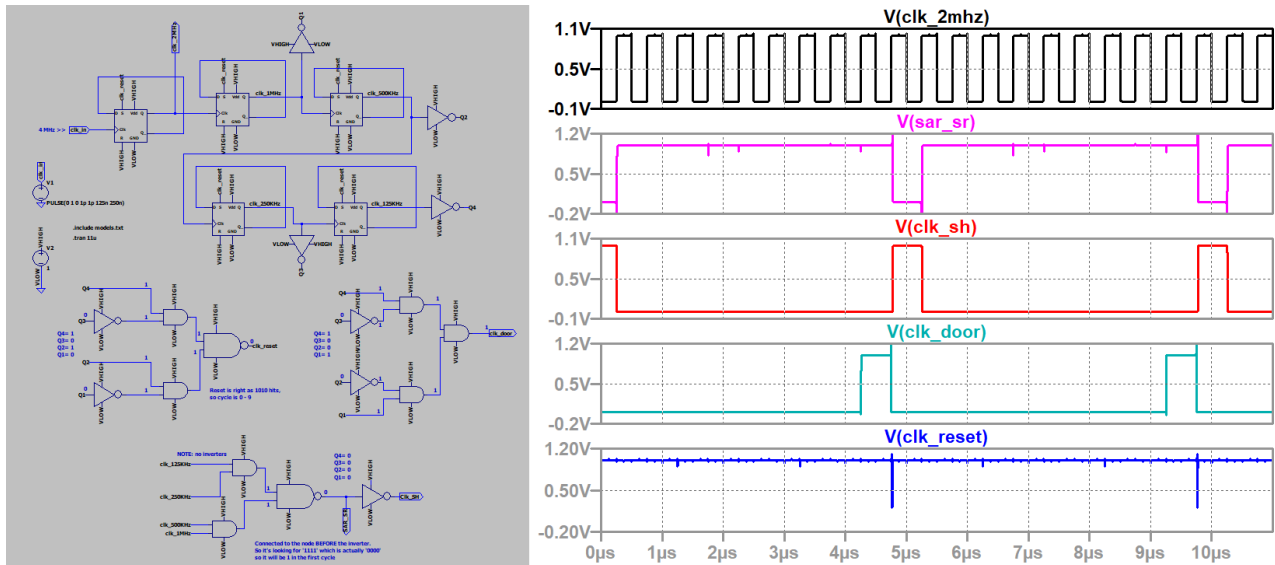


Figure 12.3. Clocks circuit simulation. The “clk_in” signal is 4MHz, which is “PULSE(0 1 0 1p 1p 125n 250n)” in LTSpice. The very first cycle is a half period, so the first rising edge is period 1. The “clk_2mhz” will go through 10 periods before resetting. You can see that “clk_door” is 1 for the last period and “clk_sh” is 1 in the first period.

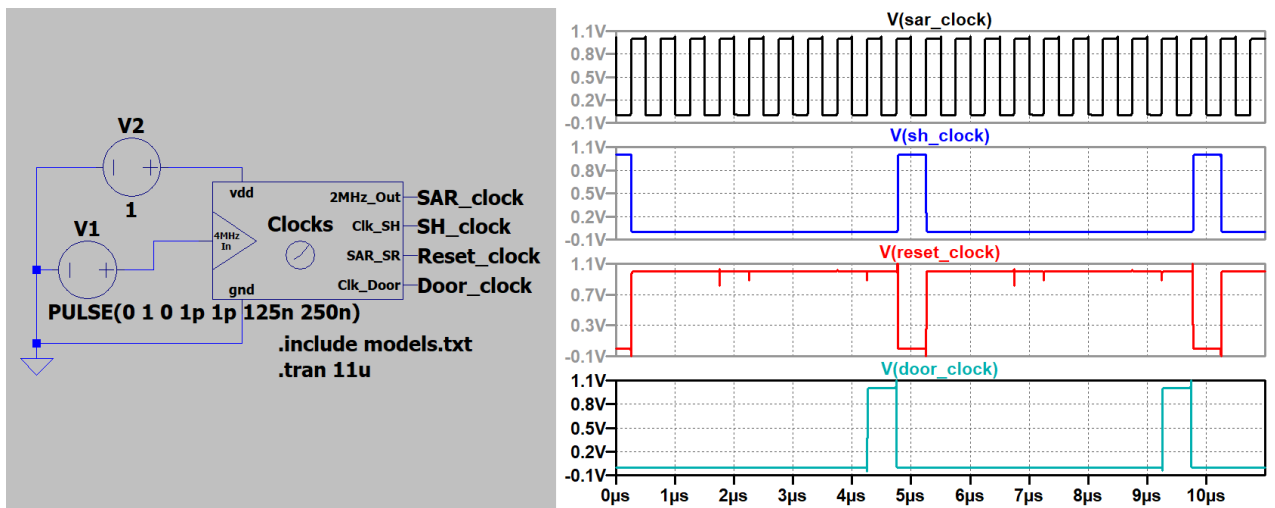


Figure 12.4. Clocks symbol simulation.

Part 4.

Now that all the components have been built and tested, we can assemble the final SAR ADC circuit. For a more comprehensive simulation result, a DAC will be connected to the output bits as seen in Figure 13.1.

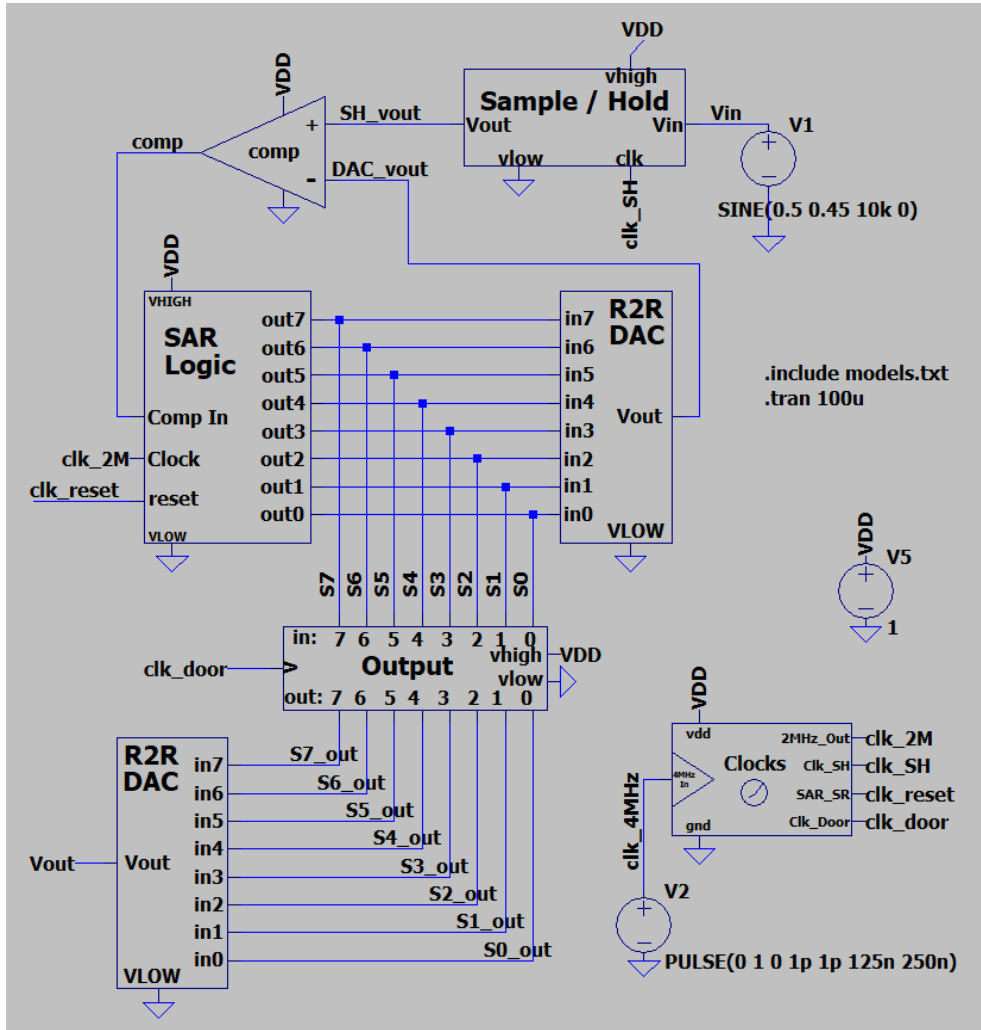


Figure 13.1. (left) The SAR ADC circuit schematic with a second R2R DAC connected to S7_out through S0_out.

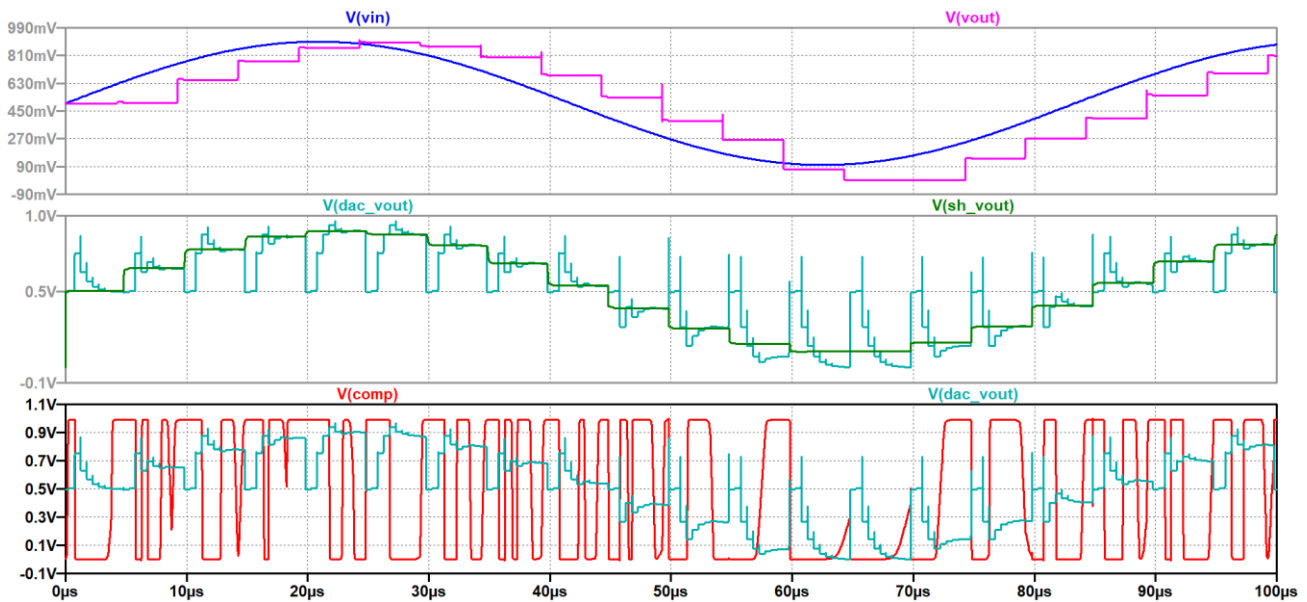


Figure 13.2. SAR ADC simulation results. In the top graph is V_{in} and V_{out} from the secondary DAC at the very bottom of the Figure 13.1. The middle graph shows the SAR Logic converting to the Sample/Hold output value. The lower graph is the comparator directing the SAR Logic as seen through DAC_vout .

This project is an LTSpice adaptation from [3]'s project. Major differences include a different comparator (from [1]), a simpler set/reset function, a different timing logic block, an R2R DAC, the Output Door circuit, and a slightly different SAR Logic circuit.

References:

[1] R. Jacob Baker, *CMOS: Circuit Design, Layout, and Simulation*, Third Edition. Piscataway, NJ: IEEE Press, 2010, pp. 931-1007.

[2] Abo, Andrew M., and Paul R. Gray. "A 1.5-V, 10-bit, 14.3-MS/s CMOS pipeline analog-to-digital converter", *Solid-State Circuits, IEEE Journal of* 34.5 (1999): 599-606.

[3] Columbia project, reference still in progress.