**EECE 416 Design Project**

**FALL 2019**

**SAR ADC**

**David Caldera and Moses Peace McCabe**

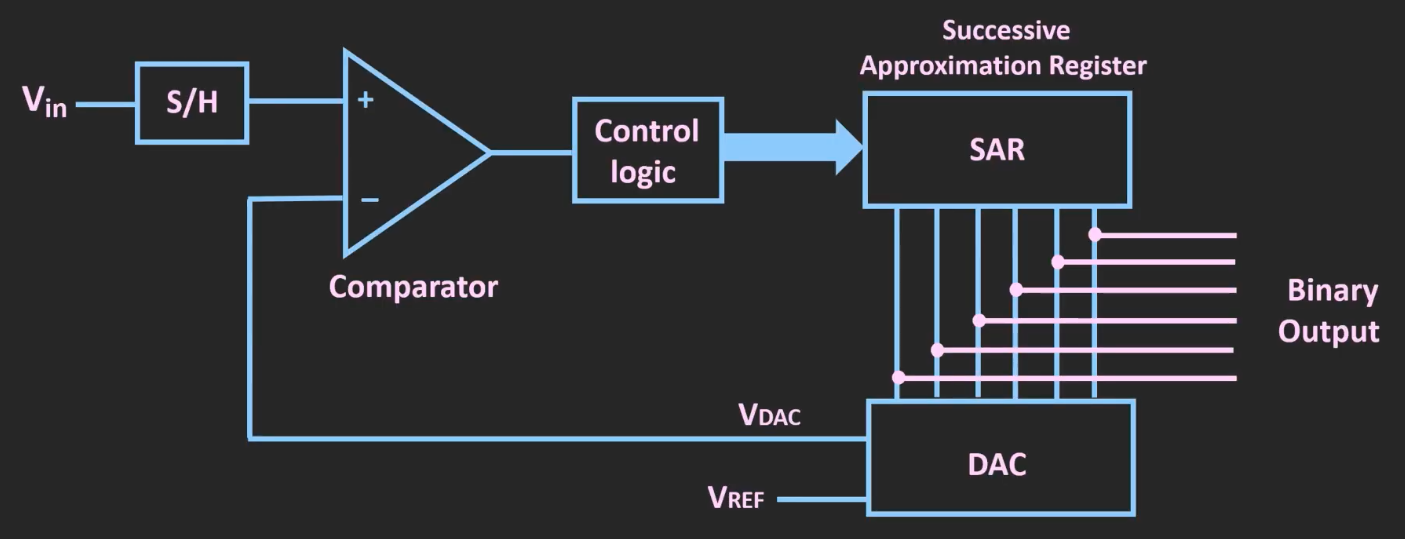
**ABSTRACT**

Successive approximation register (SAR) analog to digital converters (ADCs) is one of the most used architecture for application requiring sampling rate under 10 MSPS. The SAR-ADC converts a continuous analog waveform into a discrete digital representation via a binary search through all possible levels before finally converging upon a digital output for each conversion.

This project designed an 8-bits successive approximation register ADC uses two shifted register to perform its binary algorithm and an R-2R ladder DAC that convert the SAR register binary bits into an analog voltage to be compared by the comparator.

**1. PROJECT DESCRIPTION - Moses**

## The Successive approximation register ADC converts a continuous analog waveform into a discrete digital representation via a binary search through all possible levels before finally converging upon a digital output for each conversion. The ADC takes a continuous analog waveform as input. A Sample and Hold circuit perform a sample to the input base on its clock frequency and past it output to the comparator, which checks if the value is less or greater than the DAC output. The comparator outputs a one if the DAC voltage is less than Sample and Hold value (S/H), and a 0 if the DAC voltage is greater than the S/H output. The output of the comparator is fed into the SAR register, which produces a discrete digital representation via a binary search through all possible quantization levels. The SAR must run N (number of bits) faster than the S/H in other to produce all bits level. The DAC register converts the binary output of the SAR via an R-2R Ladder register, see figure below.

****

**START UP PROCEDURE**

**In the milestone 3 submission for this project, many unnecessary libraries and other files were included that were not needed or used in the final circuit design and simulation. The startup procedure will be provided assuming the user wants to use the down hierarchy feature of Electric and navigate through the internal circuitry used to build the circuit.**

**The final test we ran for the project was the testing of the combining the SAR, COMPARATOR, and DAC. The down hierarchy list starting from this circuit will be provided.**

**The following libraries need to open on Electric:**

1. **David\_Caldera\_ADC**
2. **SAR\_8Register**
3. **ADC**
4. **ADC---Copy**

**To navigate the SAR from the top level:**

| Down Hierarchy Flow | SAR Logic | Moses |
| --- | --- | --- |
| Library | Circuit | Top Level |
| David\_Caldera\_ADC | SAR\_COMP\_SAR\_ADC | Comparator Attach to SAR and DAC |
| SAR\_8Register | SAR\_DAC | DAC attached to SAR block |
| ADC | DAC\_8-bit | DAC circuit |
| ADC | SAR | SAR circuit |
| ADC | FF\_AS | Master-Slave Flip-Flop |
| ADC | NAND\_3 | 3-Input NAND\_3 |
| ADC | INV\_20 | INVERTER |

**To navigate the Comparator by hierarchy:**

| Down Hierarchy Flow | Comparator | David |  |
| --- | --- | --- | --- |
| Library | Circuit | Top Level |  |
| David\_Caldera\_ADC | SAR\_COMP\_SAR\_DAC | DAC attached to SAR block | |
| David\_Caldera\_ADC | COMP\_2P | Auto-Centering Comparator | |
| David\_Caldera\_ADC | NO\_CLK | Non-overlapping clk | |
| David\_Caldera\_ADC | FF\_AS | Singe Cycle Edge Trigger | |
| David\_Caldera\_ADC | OR\_2 | 2-Input OR | |
| David\_Caldera\_ADC | NAND\_3 | 3-Input NAND | |
| David\_Caldera\_ADC | NAND\_2 | 2-Input NAND | |
| David\_Caldera\_ADC | AND\_2 | 2-Input AND | |
| David\_Caldera\_ADC | INV\_2 | 2-Input AND | |

**To find and navigate the Edge Detector by hierarchy**

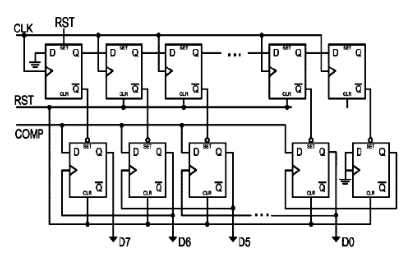
| Down Hierarchy Flow | Edge Detector | David |
| --- | --- | --- |
| Library | Circuit | Top Level |
| David\_Caldera\_ADC | Edge\_Detector | Edge Detector Circuit |
| David\_Caldera\_ADC | FF\_AS | Single Cycle Edge Trigger |
| David\_Caldera\_ADC | Other Logic Gates |  |

**To find and navigate the Edge Detector by hierarchy**

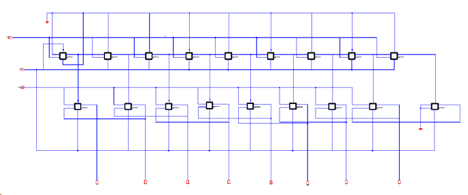
| Down Hierarchy Flow | CLOCK DIVIDER BY 8 | Designed by David |
| --- | --- | --- |
| Library | Circuit | Top Level |
| ADC---Copy | CLK\_DIV\_MS | Clock Divider Using Master Slave Flip Flops |
| ADC---Copy | FF\_MS | Master Slave Flip Flops |
| ADC---Copy | AND,NAND,NOR,OR | Logic Gates |

**2.1 SAR Logic/Register - Moses Peace McCabe**

The Successive-approximation-register (SAR) logic Register is the heart of the SAR-ADC (Analog to Digital converter) system. It generates digital signals that are fed into the R-2R Ladder DAC network. The SAR register takes in a digital true or false (one or zero) input and uses a binary tree algorithm to produce an 8-bit digital output. The SAR architecture used in this report is the exact logic structure used in milestone two, as shown below. This logic block implementation used two shift registers in other to perform the successive approximation routine, which is the binary tree algorithm. Each shift register is composed of a chain of nine D Flip-Flops. The shift register on the top is used as a sequencer and is synchronous with the internal clock. The internal clock is faster than the system clock, which allows the SAR register to perform binary algorithm and shift each bit. The bottom registers store the conversion value and outputs it into the R-2R DAC register, which, as the name implied, convert the output for the SAR into an analog value. This value is compared with each sampled and hold output via a comparator circuit, and the result is feed into the SAR register to perform another shift.

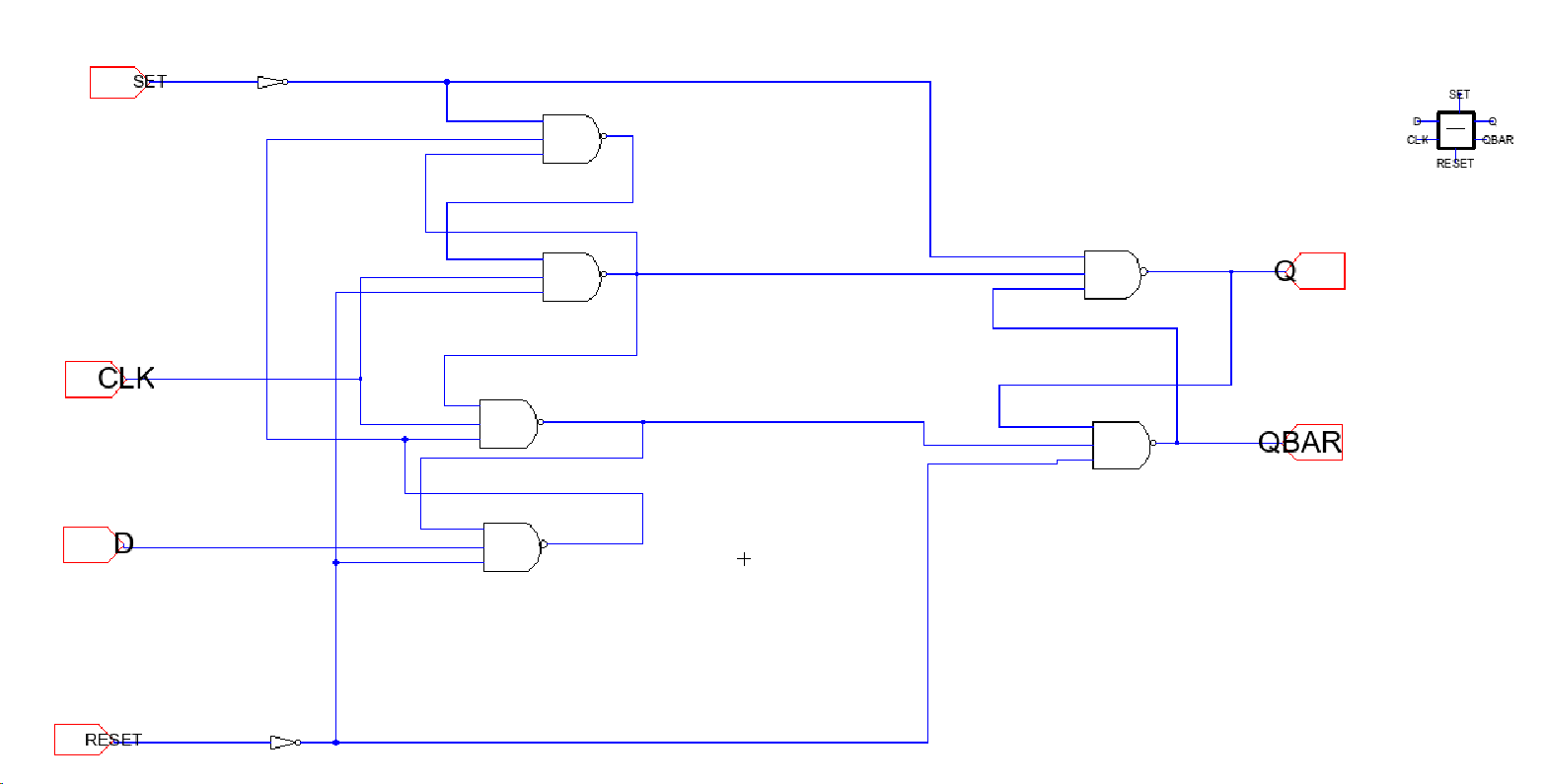


8-bit SAR Logic Register [1]



8-bit SAR Logic Register – Electric implementation

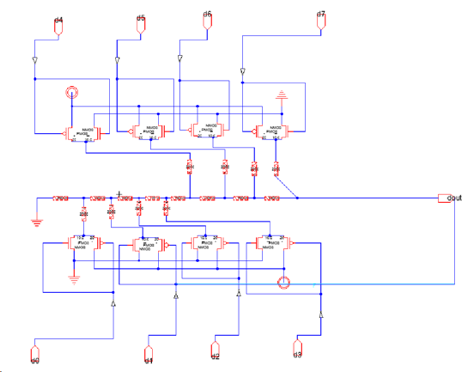
The D-Flip Flop used to construct the above figure was built by Mr. McCabe. The Flip Flop using a master-slave approach, see figure below. The input clock was designed by Mr. Caldera, see section 2.2 for figure

.

D-Flip Flop – Electric Schematic

**R-2R Ladder DAC**

The R-2R Ladder Digital to Analog Converter (DAC) used in this project converts digital data (binary) into an analog voltage. The DAC uses only two resistance values, R = 1000ohm and 2R = 2000ohm, to converts the parallel SAR output into an analog voltage, see figure below. Regardless of the input bits and the amount of resistor, the output impedance of the DAC is equal to R. Because of this simplicity in the calculation and weight of the output impedance, this structure is a simple, effective, accurate, and inexpensive way to create a DAC.



R-2R Ladder DAC – Electric Schematic

The PMOS and NMOS transistor used on the above circuit was created by the help of the tutoring videos[2]. the PMOS was sized at 20 and the NMOS at 10. Using the inverters sizes as reference

**2.1.1 Design Requirements**

The successive approximation register needed to produce an 8-bit output based on the input. This required a circuit that performs bit shifting via a binary search algorithm to store each data. Since each out of the Sample and hold must result in an 8-bit SAR output produced by the SAR shifting algorithm. The SAR required 8 clocks to successfully approximate its output period before the next Sample and Hold value is updated to the comparator.

The Flip-Flop used to create the SAR must be fast and accept four inputs and two outputs. A pin for set, reset, bit, clock, and output 1 and 2. The set bit will allow the flop to store input to output 1 the reset pin will clear the flop, while the clock pin will allow the entire flop network to function/update at the same speed.

The R-2R Ladder DAC must convert binary to an analog voltage. The system must be fast and easily scalable to any desired number of bits. The DAC must use a resistor instead of a capacitor to an output voltage, which will make the calculation of the DAC easy. Since the DAC is a sub-circuit of the system output resistances of the DAC must be easily scalable and small. Uses only two values of resistors which make for easy and accurate fabrication and integration

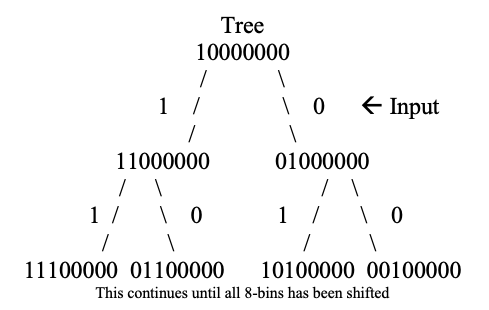
**2.1.2 Description of Final Circuit**

The successive approximation register for this project needed to accomplish two goals. To take in a binary input from the comparator and create an 8-bits output to be applied to the R-2R DAC register. This goal is accomplished by creating a SAR circuit with two shift registers in order to perform a binary tree algorithm. As stated each shift register is composed of a chain of nine D Flip-Flops. These Flip-Flops perform bit shifting depending on the SAR input. The shifting process accomplished by the Flops is known as a binary algorithm.

The successive approximation register algorithm function as such:

* The MSB of the SAR is initially set to 1 once the reset bit is high. The remaining bits, bit 7 through bit 0, are set to low.
* Since the SAR output 8-bits (n=8), the result will be 10000000.
* If the comparator output a 0, and the SAR shifted the MSB value to bit-1, and store the input to the MSB.
* If the comparator output is a 1, the SAR shifted the MSB value to bit-1 and copied the input into the MSB. In other words, the SAR keep it MSB as is and change bit-1 to 1, while the remaining bits are all 0.
* This process is repeated eight times for each of the samples and holds value.

This algorithm is best to view as a binary tree. Where the top of the tree is step one, and the body of the tree is the remaining steps described above.

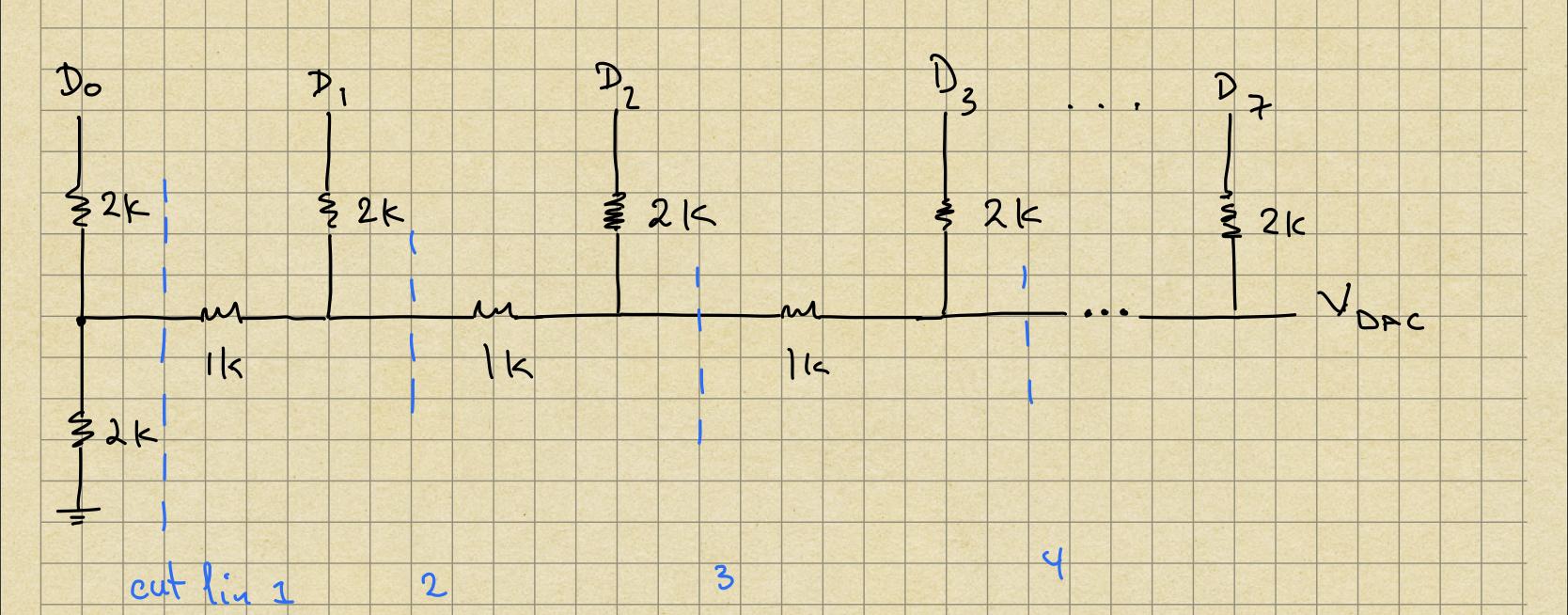


As stated previously, the binary search algorithm is being performed by the SAR shift registers. As shown in the figure, the logic block implementation used two shift registers in other to achieve the successive approximation routine. This top sequential device (shift registers) load a 1 once the reset is high and then moves or “shifts” it to its output once the reset is low and once every clock cycle. The production of each parallel-in parallel-out shift registers passes to the next second half parallel network set pins to which update the value of the DAC output based on the input (comparator value). This performed the binary search algorithm explained above.

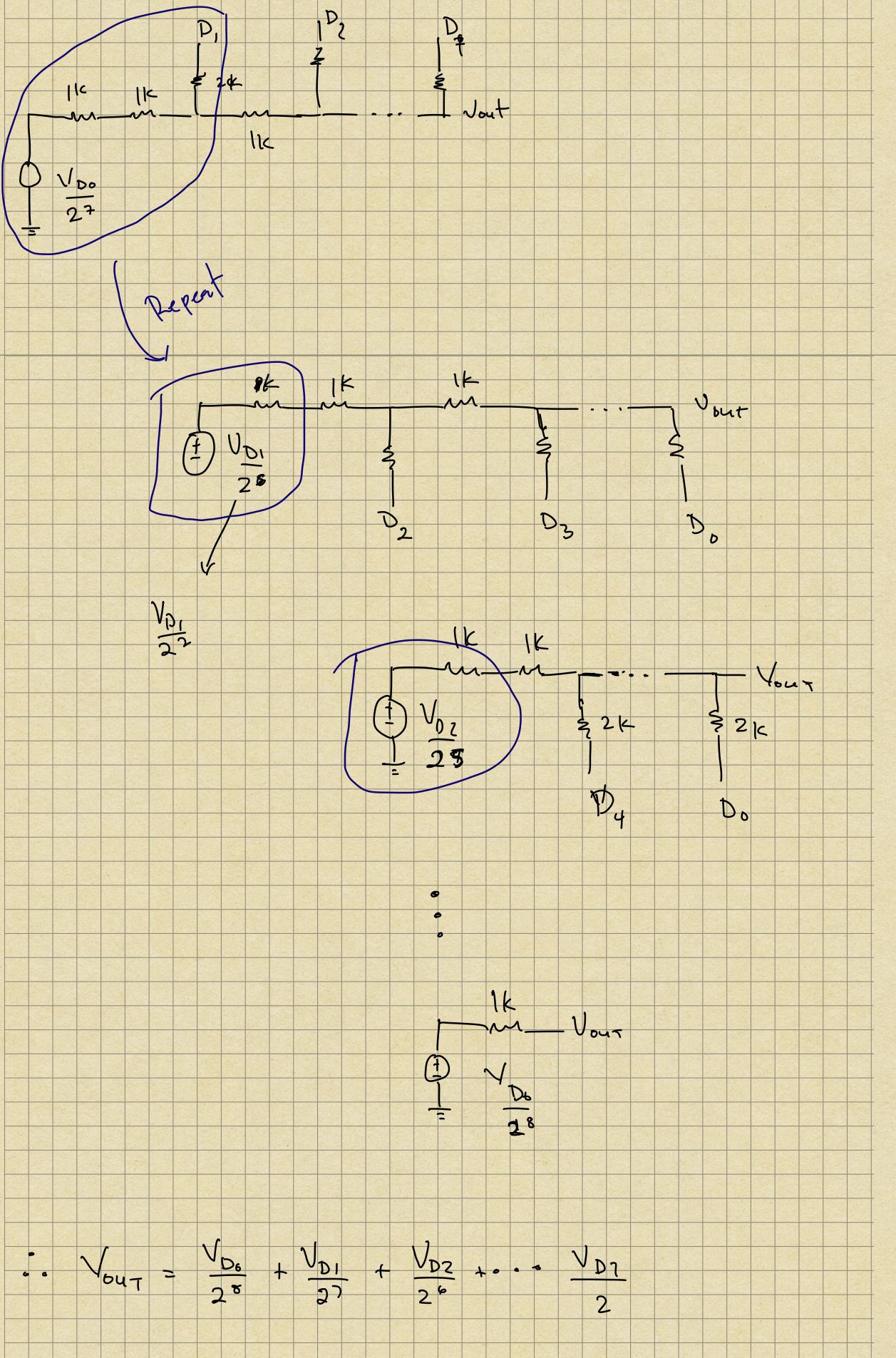
This algorithm will only function correctly if the Flip-Flop used in the SAR accept a Reset and Clock as an input. The SAR need a reset value to set the MSB output to 1 while allow the system to accept new value for the comparator after each sample. The clock on the SAR must run 8 times faster that the Sample and Hold clock. Giving the system enough time to sample each value. The four input and two output master-slave D Flip-Flop accomplish this goal.

**2.1.3 Design Calculations**

The R-2R Ladder DAC can easily be study using the Thevenin rule will stated that if a circuit containing linear elements, we can cut the circuit at any point and add a test voltage and a single series resistor to analysis the cut circuit. The test voltage is the open circuit voltage at the cut point, and the series resistor is the equivalent open circuit resistance with all voltage sources shorted. Applying this rule to our DAC speed up the calculation, see figure below.



We start by replacing the circuit to the left of the left-most cut-line with its Thevenin equivalent. This process continues methodically, step by step for each cut-line, substituting the equivalent circuit for each stage, we obtain the figure below.



From these figures we see that the voltage at each stage is showing as , Where N = 8.

**2.1.4 Design Evolution**

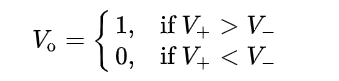
The foundation of the SAR register is the flip flop. The designed D Flip-Flop by Mr. Caldera, which was created with the help of the lecture notes, did not work when used to construct the SAR register. As a stand-alone circuit, the flip flop work, but once applied to the SAR register structure, the flip flop did not work. Mr. Caldera flip-flop suffers from timing problems; the output Q changes state before the timing pulse of the clock input has time to go off. Further research and collaboration with classmates led to the D flip flop use in this project, see figure 3. The reason for using this structure is that its functions as storage that both samples its input and stores data based on the transition of a clock pulse. If the combinational parts of a circuit could settle during the time, the clock signal was true, and the storage part of the circuit sampled the input and saved the result when the clock changed from true to false. This design works because the it flip-flop eliminates the timing problems

**2.2 “Clocked” Comparator + Timing Circuitry - David Caldera**

**2.2.1 Design Requirements**

For the “Clocked Comparator” and Timing circuitry portion of the SAR ADC, the objective was to design 3 functional blocks. The function blocks of the SAR ADC that I was in charge of was the comparator, clock divider, and edge detector. The requirement for the “Clocked” Comparator is to design a comparator structure that compares two inputs and outputs either high or low that is also time dependent. The objective of the comparator is to compare the output of the sample and hold circuit to the output of the digital-to-analog-converter (DAC) and send the correct output to the SAR.

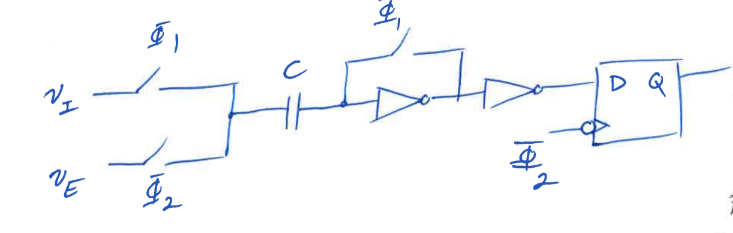
The two inputs of a comparator are typically called V+ and V-. Standard behavior of a comparator is as shown below:



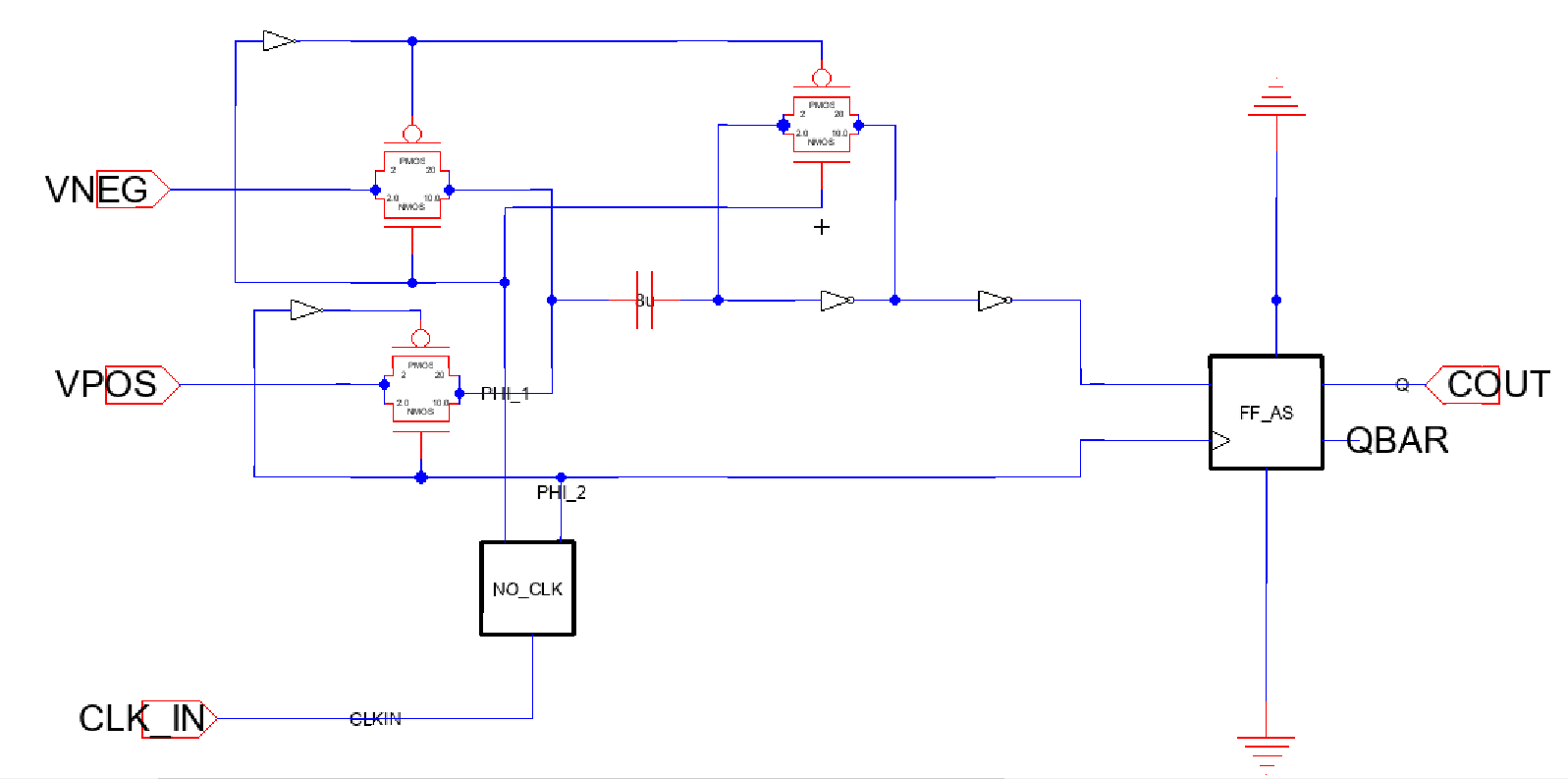
The design of the SAR ADC requires the synchronization of the functional blocks with respect to certain clocks. The implementation of the SAR ADC requires two clocks, a “slow clock” to drive the sample and hold circuitry, and a “fast clock to drive the SAR Control Logic and comparator. The clock divider designed for the ADC takes a pulse signal (clock) and outputs a pulse with a slower frequency. The state machine designed for the SAR Logic has 8 states which requires 1 full clock cycle to process each output of the ADC. This results in the SAR logic needing 8 clock cycles to complete approximating the sampled value. To accomplish this effect, the “fast” clock will feed the SAR circuit while also passing it through a clock divider circuit that divides the clock frequency by 8.

**2.2.2 Description of Final Circuit**

The structure used for the comparator block of the ADC is the “auto-centering” comparator. The figure below is a generalized circuit schematic of a commonly used comparator structure. In the context of the ADC, the comparator accepts two inputs being the sample and hold output into the negative terminal while the positive terminal receives the DAC output. The negative and positive terminals of the comparators are represented in the schematic using Vi and Ve respectively. 𝛷1 and 𝛷2 represent the non-overlapping clocks the circuits needs to drive the switching circuitry. The switching circuitry was implemented using transmission gates with nmos and pmos transistors.

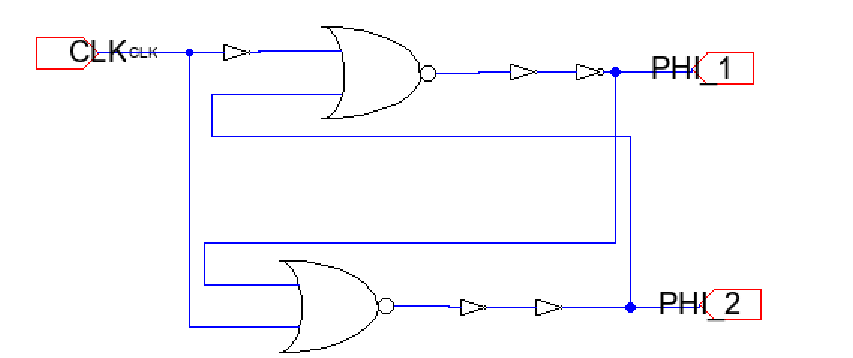
****

**Electric VLSI schematic of comparator circuit.**



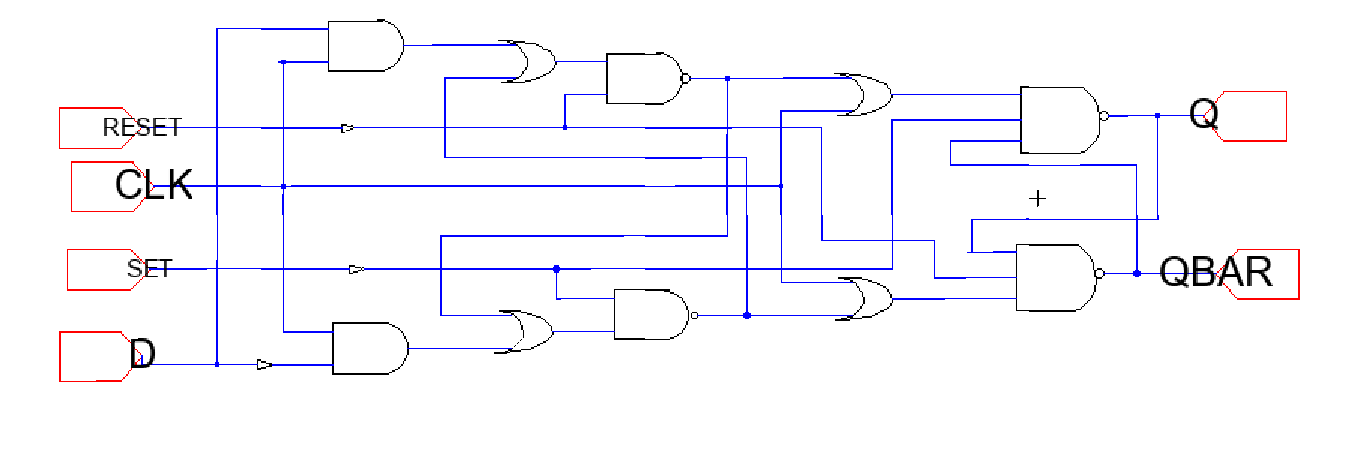
**Two-phase non-overlapping clock generator schematic for comparator.**

This is the NO\_CLK circuit block in the above image. The non-overlapping clock accepts the master clock of the ADC as input, and outputs two clocks that drive the comparators transmission gates.



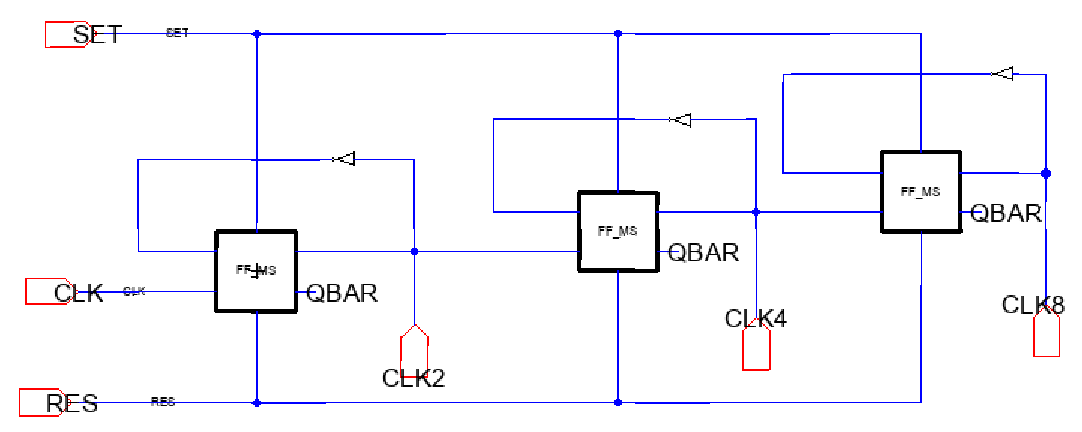
**Single Clock Phase Edge Triggered Flip Flop with asynchronous set and reset pins schematic**

This flip flop design is the FF\_AS circuit block in the comparator structure above. This structure accepts the four standard inputs of D, CLK, SET, RESET. The outputs available in this structure are Q and QBAR. This flip flop structure is a positive edge triggered latch whose output Q is supposed to follow D in the positive edge of the clock. This structure also offers the asynchronous set and reset inputs to the flip flops. When the set pin is set high, Q goes high independent of the clock.



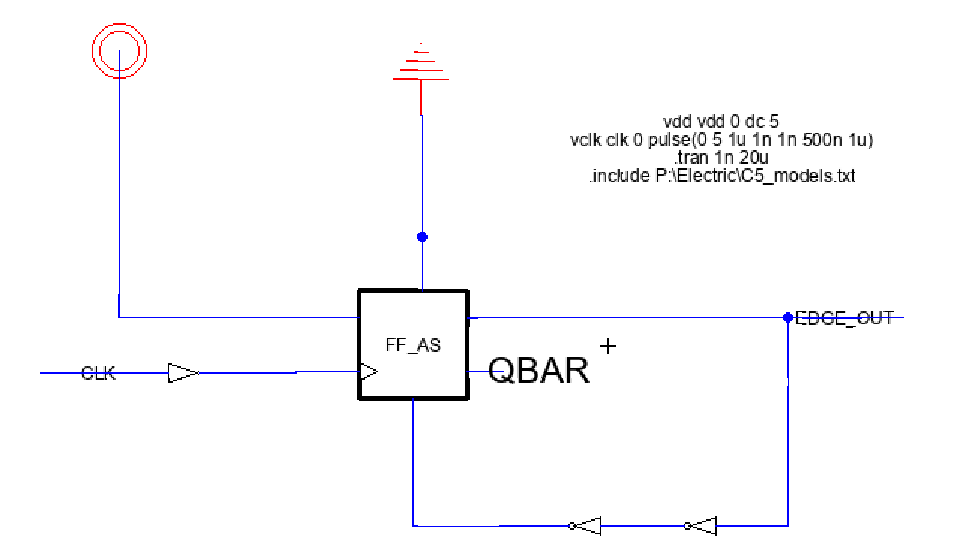
**Electric VLSI Clock Divider-by-8 circuit schematic**

The clock divider circuit uses the master slave flip flop design built by Mr. McCabe to create the clock divider circuit. The clock divider is built using three clock divider by 2 in series. Each flip flop in the clock divider design serves as a clock divider by feeding the inverted output back into the D input of the flip flop. The output of the flip flop is being fed into the clock input of the next divider stage. For this structure I am allowing access to the output of each stage which are inputs clock signal divided into 2,4, and 8.



**Electric VLSI Edge Detector Circuit**

The image below shows the top level circuit for the edge detector. The edge detector is built with the single cycle edge triggered flip flop. The edge detector circuit used accepts two inputs D, CLK, SET, and RESET, and outputs an EDGE\_OUT output which is a short impulse. The purpose of edge detector is to reset the SAR Control logic after one cycle of the slow clock. The resetting of the SAR logic sets all of the output bits to zero allowing the ADC to begin approximating the next sample.



**2.2.3 Design Calculations**

**2.2.4 Design Evolution**

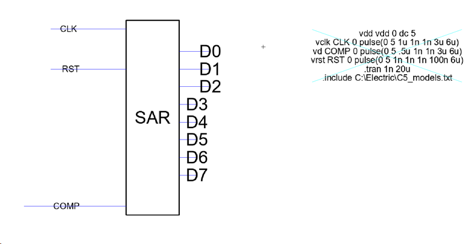
The two portions of the project that I spent the most time on is the building of different flip flop designs and comparator structures on Electric VLSI. Most of the functional blocks of the ADC required some form of edge triggered flip-flop with asynchronous set and reset lines. I built 4 different flip flop structures including PTL and static CMOS structures. The first edge triggered flip flop that worked was the single phase flip flop structure on the slides provided to us by the instructor. Later on in the project I noticed that the flip flop was being triggered by the negative edge of the clock which is not what the gate level ADC was designed with on milestone 2. To compensate for this issue, an inverter gate was added to the clock input line allowing it be rise on the positive edge of the clock.

Before committing to the auto-centering comparator, I had attempting on building the StrongARM Latch Comparator and Rail-to-Rail Comparator with Adaptive Power control, and the conventional latch comparator given in the articles provided by the instructor.

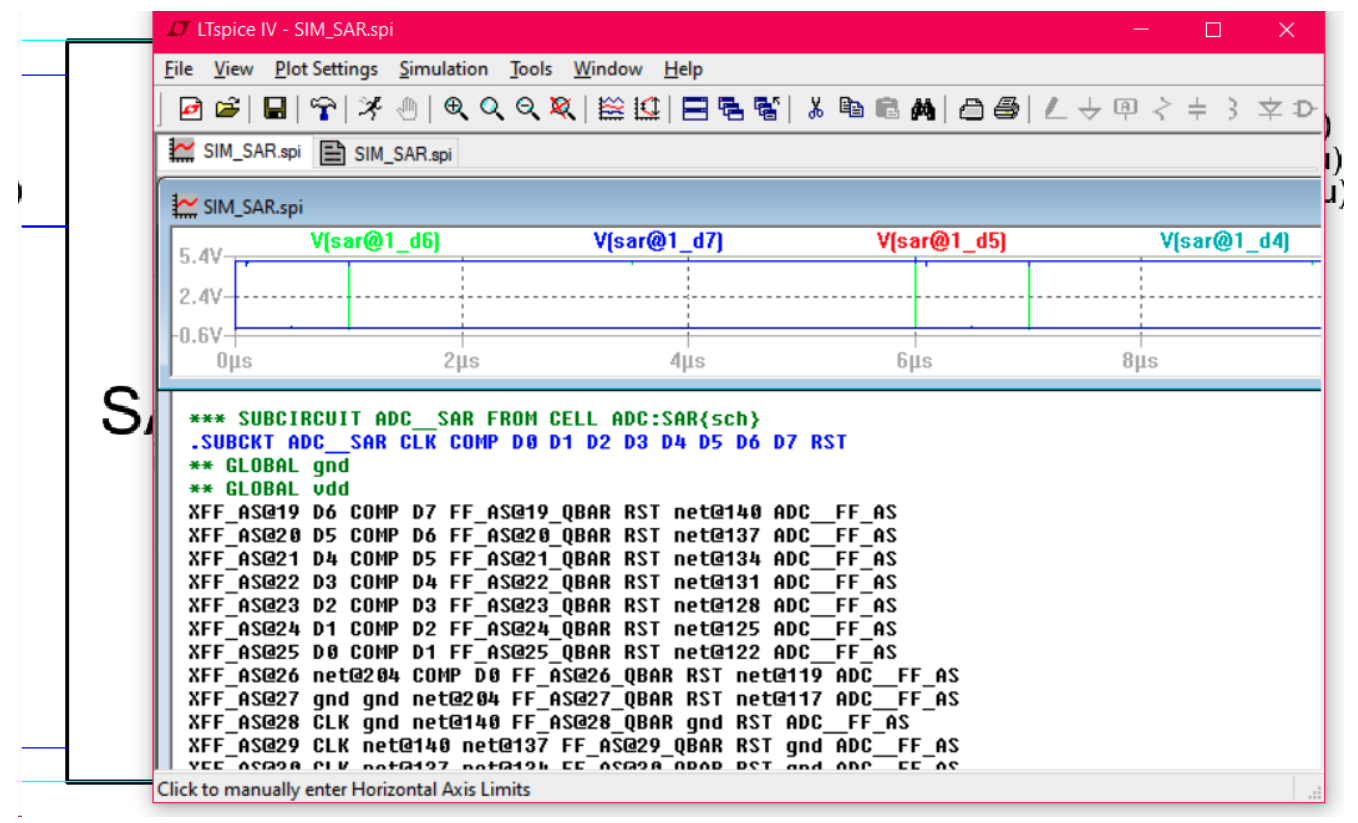
* The built circuits and simulations of these comparator structures could be found in:
* 1. Library: David\_Caldera\_ADC Cell: C\_R\_R
* 2.. Library: David\_Caldera\_ADC Cell: C\_StrongARM
* 3. Library: David\_Caldera\_ADC Cell: SIM\_CMP\_RR

**3. CIRCUIT TESTING AND DATA ANALYSIS**

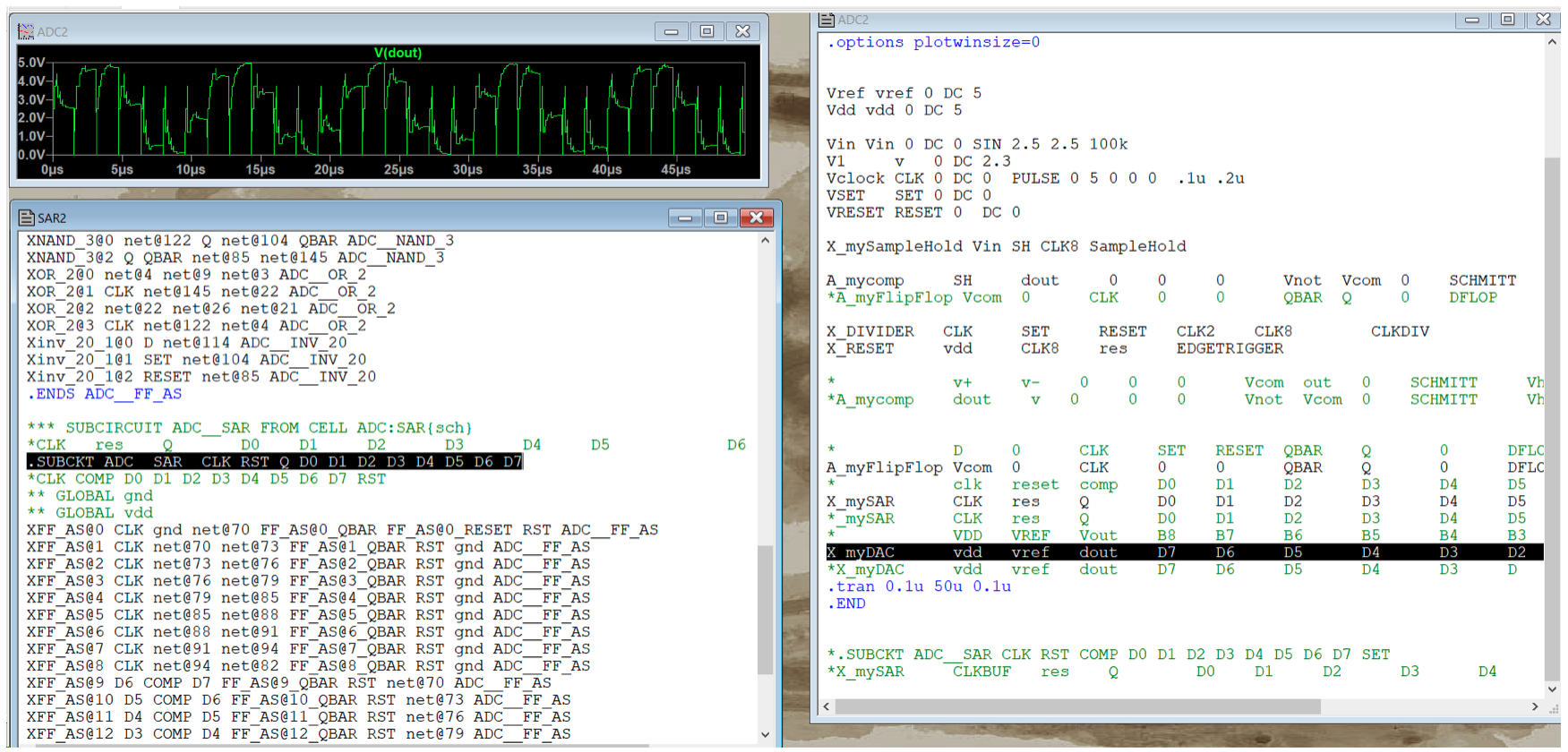
**3.1 {SAR Logic/Register} -- Moses**

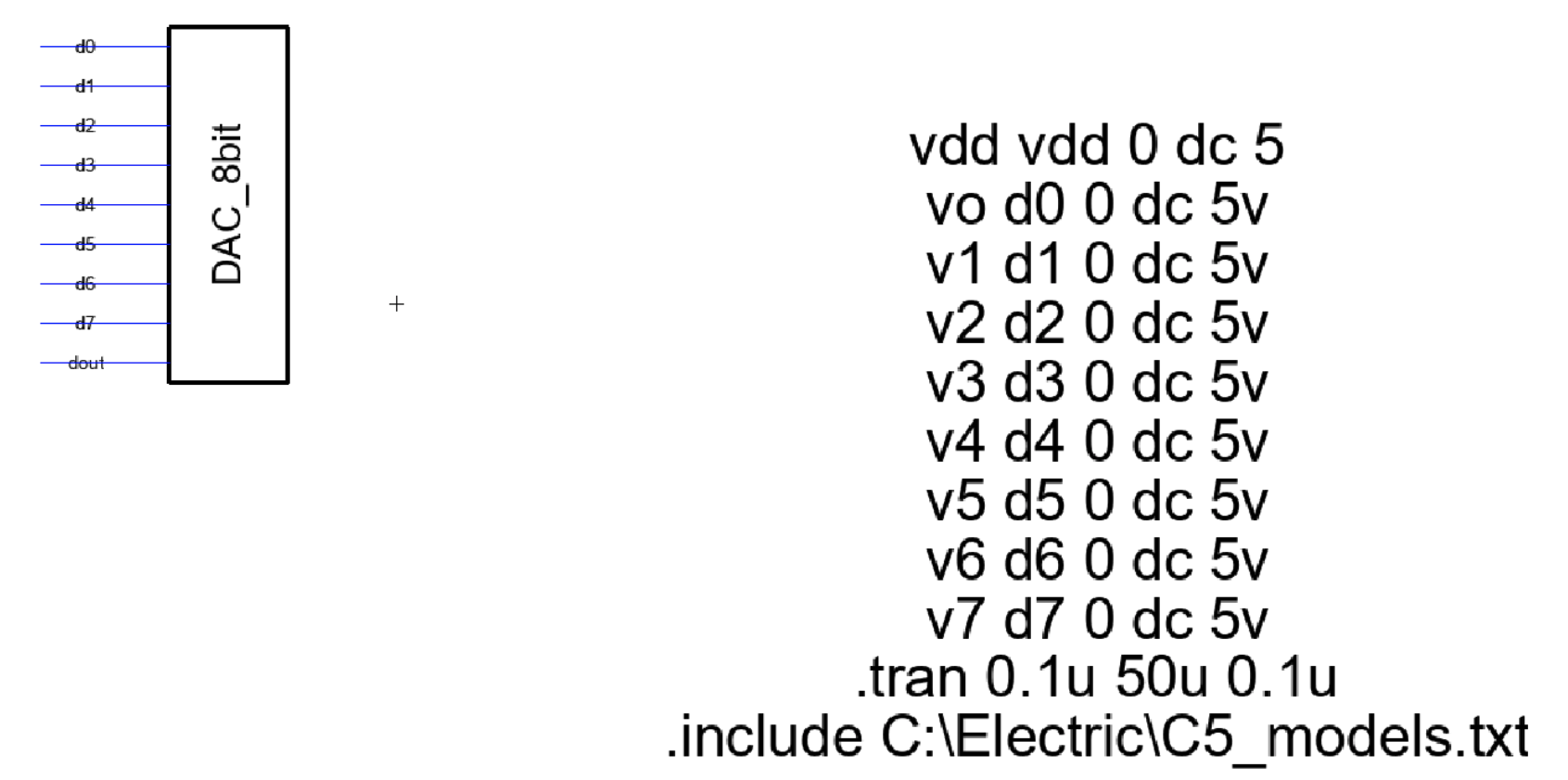


To test the SAR, I defined pulse input for the input pin and ran LTspice. Electric created a netlist file, see figure above



I copied and pasted the SIM\_SAR.spi file into notepad and save it as a circle (.cir) file called SAR2.cir . I open SIM2.cir in netlist and copied and pasted the sub-circuit into my working ADC from milestone 2. This allowed us to view the output of the DAC in LTspice simulation window. Allow us to test the SAR register, see figure below.





For testing of the 8-bit R-2R Ladder DAC register. I performed the same step applied to the SAR register.

1. Defined values DC value for each input (d0 – d7).

2. Run Spice in Electric and check the DAC output (dout).

3. Repeat step 1,with different input values

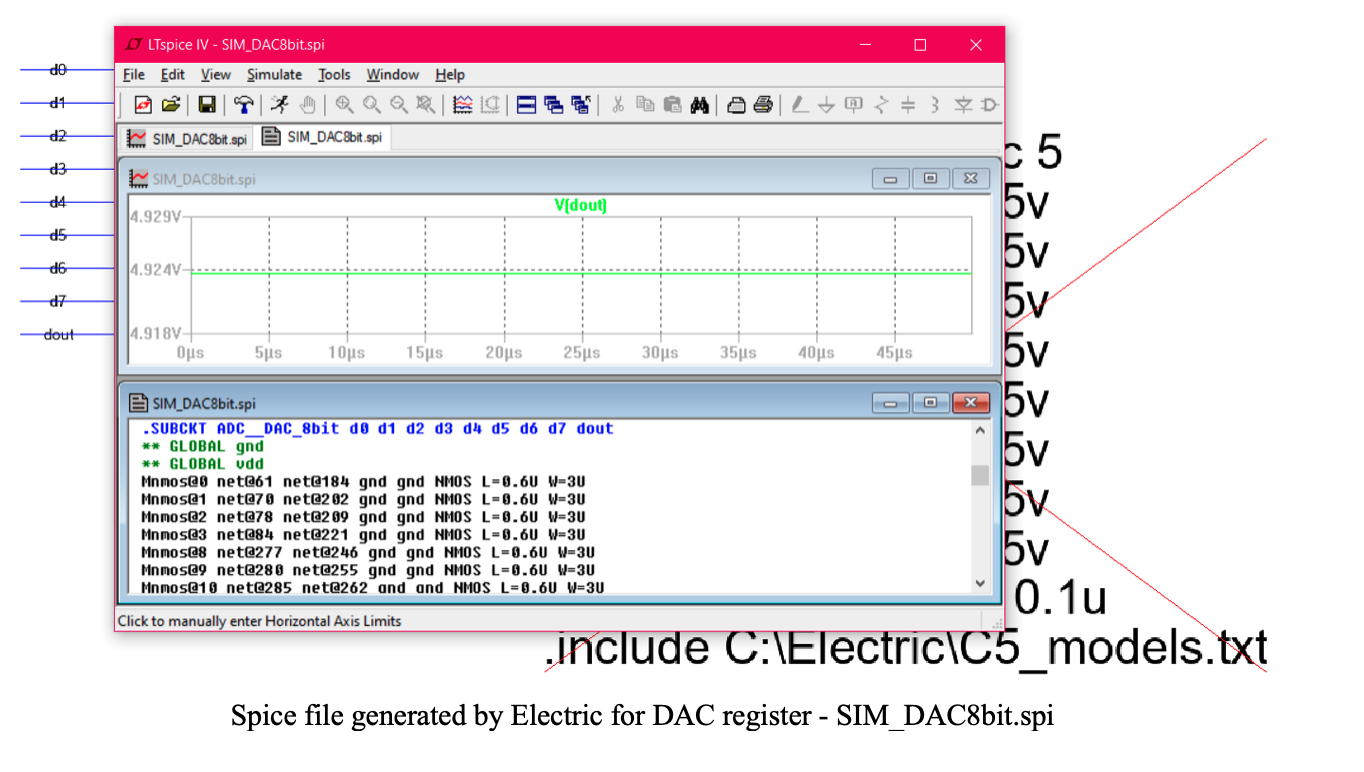
4. Repeat step 2. Compare value with quick hand calculation.

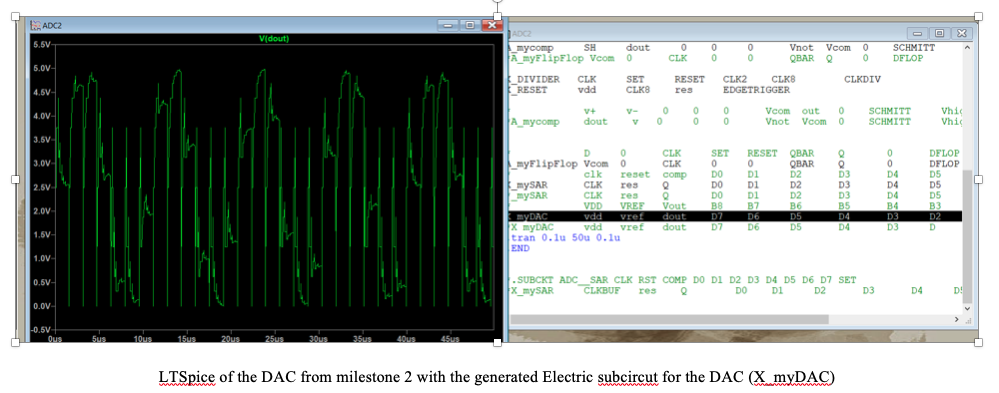
5. Copy the SIM\_DAC8bit.spi file into notepad and create a circuit file (

6. Open the circuit file in LTspice and copy the subcircuit name into the working SAR-ADC from milestone 2.

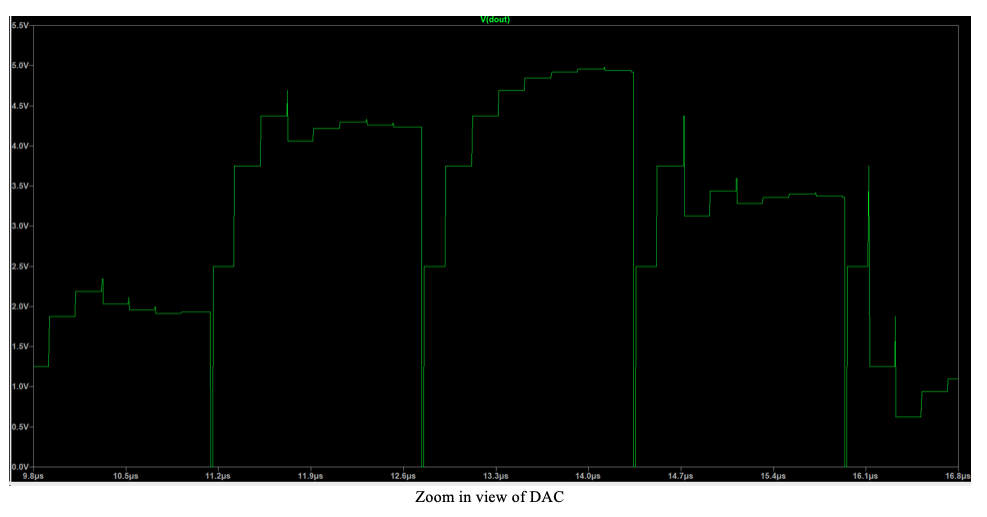
7. Run the SAR-ADC file

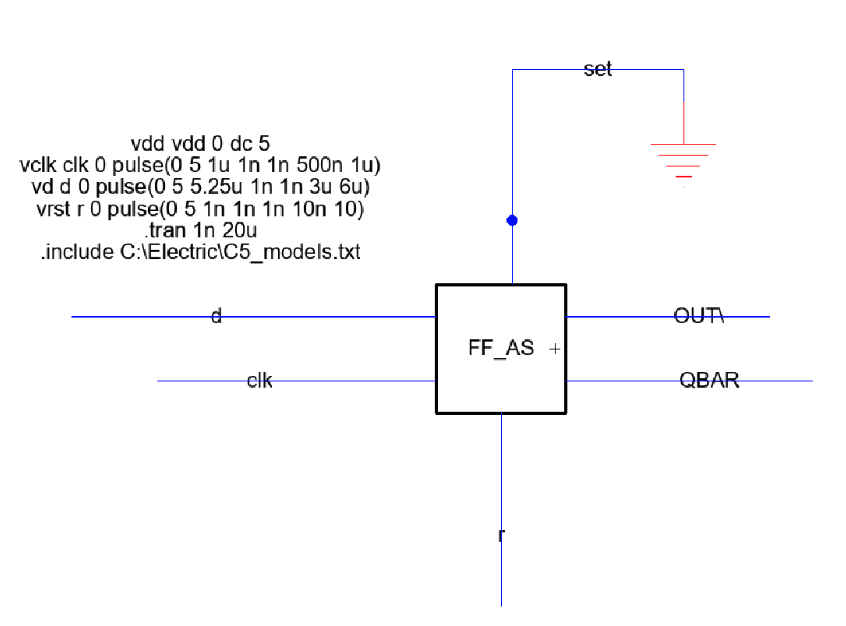
8. Run the LTSpice simulation window to check DAC output





LTSpice of the DAC from milestone 2 with the generated Electric subcircut for the DAC (X\_myDAC)





For testing of the D Flip Flop DAC register. I performed the same step applied to the DAC register.

1. Defined values pulse for each input (d, set, and reset)

2. Run Spice in Electric and check the Flip Flop output (out).



To access the file used to perform these tests. Please follow the steps below.

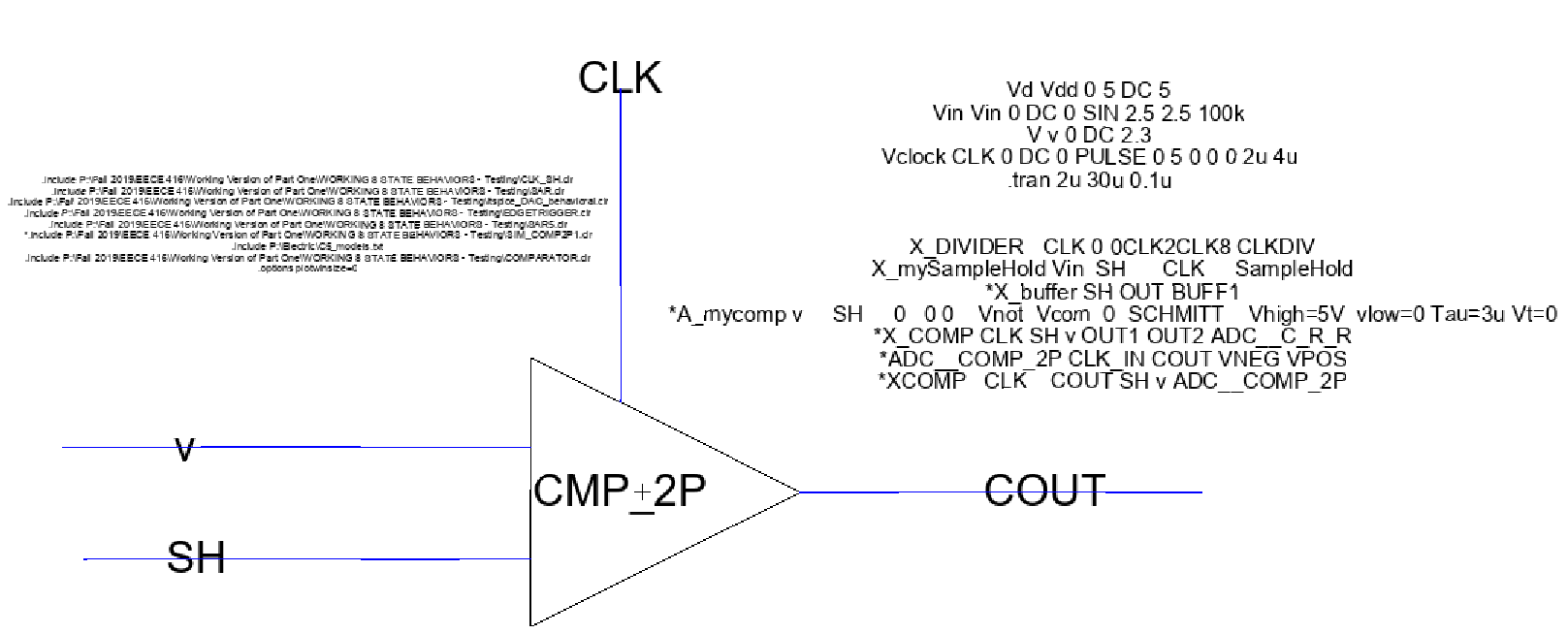
1. download the zip file sent to you in milestone 2 title “ElectricSubmission”
2. unzip the file and click the file title “Electric”
3. open Electric
4. From Electric, locate the file icon and select open library, or click ctrl O
5. Locate the unzip file and click it
6. In the unzip file locate ADC.jelib and click open
7. In electric Explorer under Libraries the ADC should be there. If not repeat step 4 – 7.
8. Click the + icon next the ADC
9. To view the DAC schematic and icon
   1. locate the file title DAC\_8bit and click the + icon next to it
   2. Click DAC\_8bit{sch} to open DAC schematic
   3. Click DAC\_8bit{ic} to open the DAC icon
   4. To run test on the DAC locate the file “SIM\_DAC8bit{sch} and run spice.
   5. Run the spice code locate in this schematic.
10. To view the SAR register schematic, icon and run test
    1. Locate the file title SAR and click the + icon next to it
    2. Click SAR{sch} - to view schematic
    3. Click SAR{sch} - to view icon
    4. And the end of the ADC file is a title SIM\_SAR{sch} - open and used the spice code within this file for testing.
11. To view D flip flop schematic and icon
    1. Locate the file title “FF\_AS” and click the + icon next to it
    2. Click the FF\_AS{sch} for schematic
    3. Click the FF\_AS{ic} for icon
    4. Locate the file “SIM\_FF\_AS{sc} and use the spice code to test the circuit.

**3.2 “Clocked” Comparator & Timing Circuitry - David**

**\*Due to miscommunication of the work done on milestone 3, a description of how to navigate the files will be provided so test described in the section could be replicated.**

The components of the ADC that I tested were the comparator, clock divider, and edge detector circuitry. Each component was tested with the functional ADC circuit designed in Milestone 2. For example, the clock divider structure in milestone 2 was designed using LTSpice ideal A source flip flops. In the context of the full ADC, once the transistor level flip flop was built in Electric, the ideal model was replaced with the Electric generated model. A description along with images of the simulations will be provided.

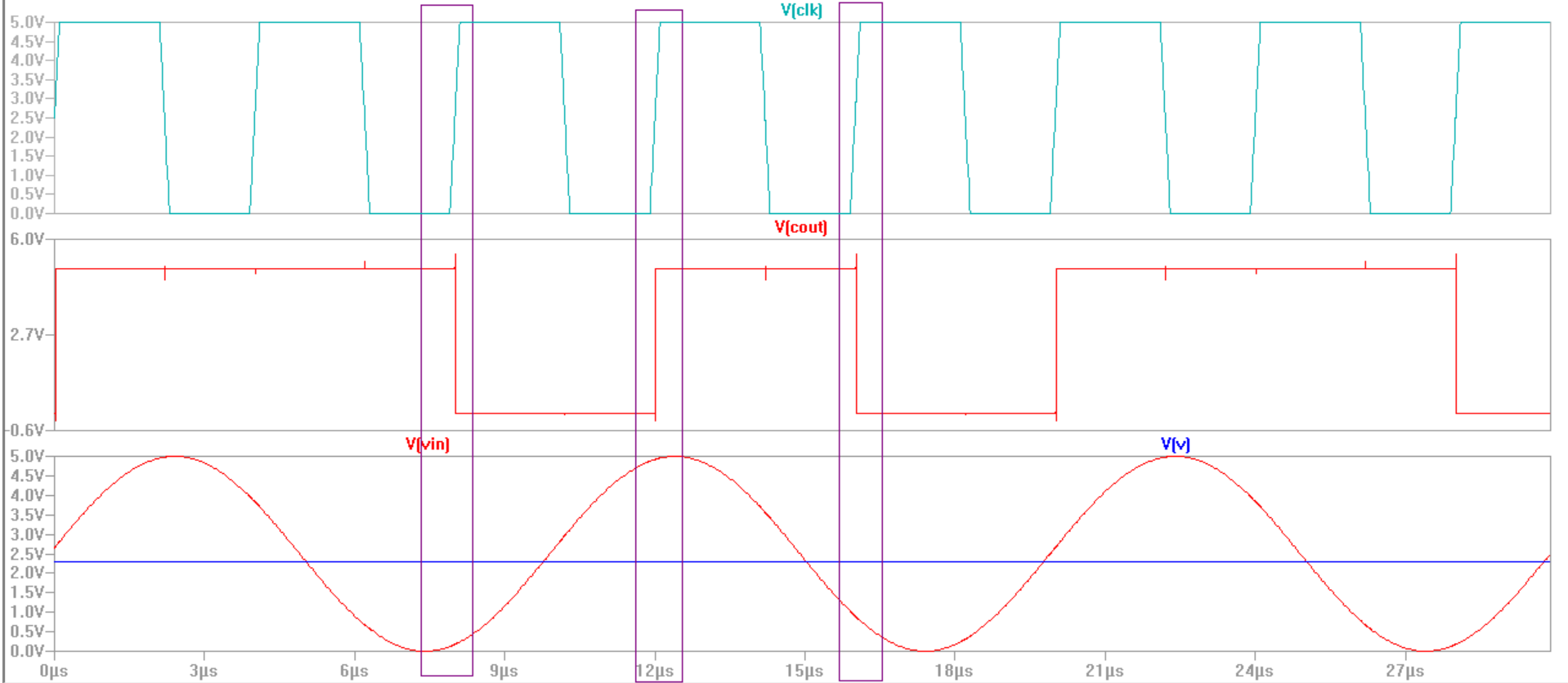
To test the functionality of the comparator independent of the DAC and SAR Logic, a sine wave was generated before to simulate the changing output of the DAC. A sine wave was fed into the sample and hold circuit to emulate how SAR holds a constant values between cycles of the fast clock. Since the SAR only changes its output every cycle and feeds the DAC, this results in a sample and hold effect in the DAC output. The Sample and Hold output goes to the negative goes into the negative terminal of the comparator while a constant 2.3V goes to the positive terminal. The circuit built is “clock comparator” meaning that the comparator only makes the comparison when the clock is high.



\* This test is located in: Library: David\_Caldera\_ADC Cell: SIM\_CMP\_2P of the submitted zip file.

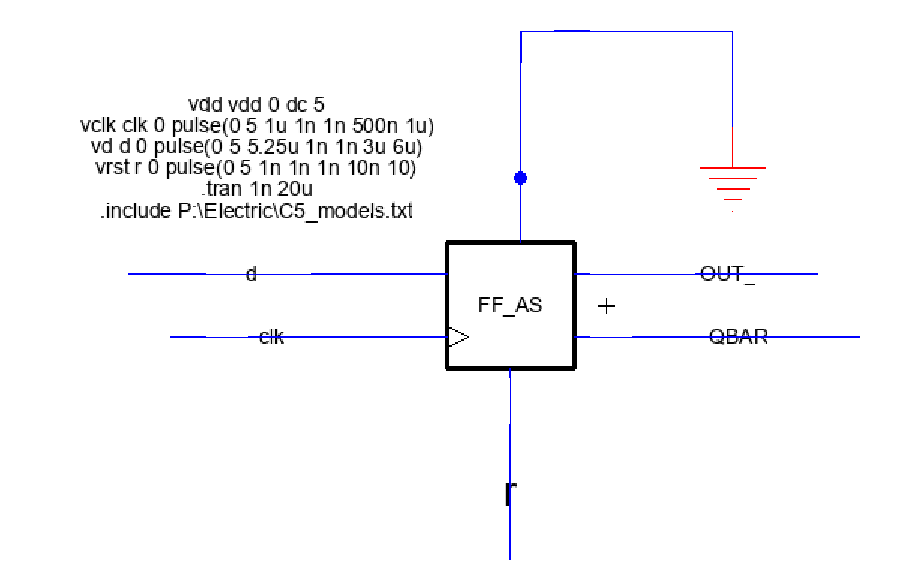
**Comparator Simulation of above model.**

The image shows the comparator simulations in Spice. The plots are split in three panes. The top panel showing a clock signal called “clk”. The middle panes showing a pulse like signal is the comparator output called cout. The bottom pane is showing the sinusoidal input and the constant 2.3V. The sinusoidal signal is called vin and the constant 2.3 is called v. The image outlines where the comparator output changes in black rectangles. The first rectangle on the captures the three panes when the clk signal goes high. When the clock goes high, vin is less than v

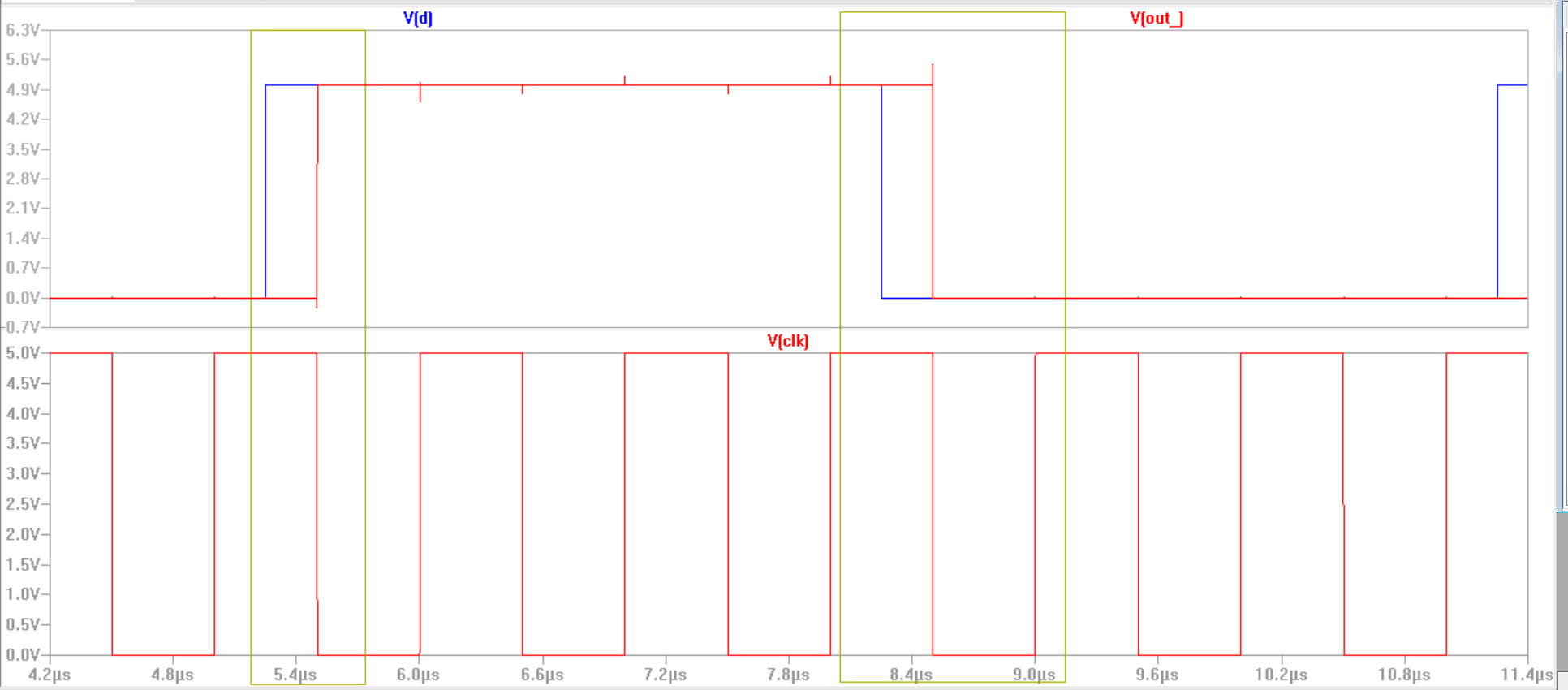


**Electric VLSI Single Phase Edge Triggered Flip flop simulations**

For testing the flip-flop,, a clock pulse with a period of 1us, a D input pulse of 6us, and a reset pulse of 10s were fed into the flip flop. The output Q of an edge triggered flip flop should follow D on the positive edge of the clock signal. In order to simulate the true behavior of the flip flop, the D needs to be on for a fixed amount of time before the clock goes on in order to get correct out.



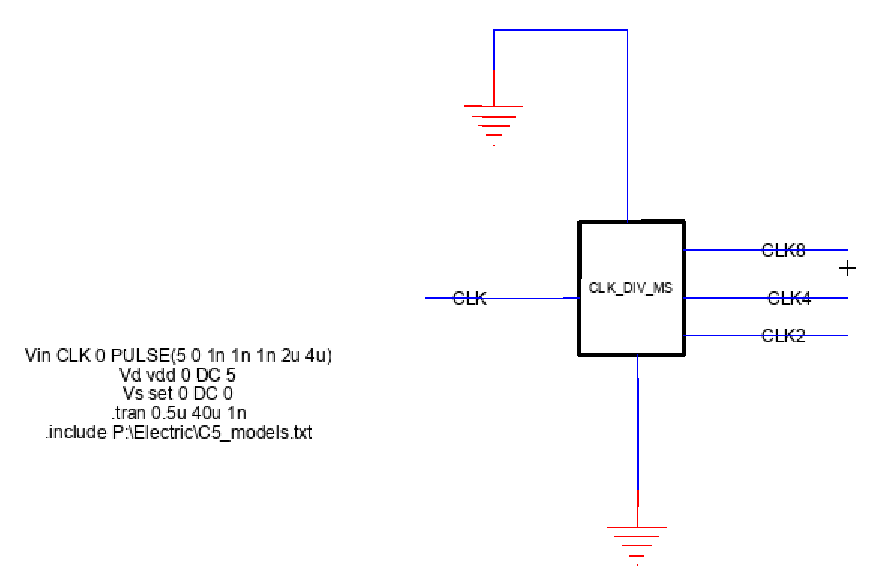
The image below shows the flip flop simulation in Spice. The plots are split in 2 panes. The bottom panel showing a clock signal called “clk”. The top panes showing the D input called d’ and the Q output called out\_’. The expected behavior of typical edge triggered flip flops is that they are triggered by the positive edge of the clock signal. The image outlines where the triggering of the flip flop happens. An observation that I did not notice until later on in the project is that Q’ follows D in the negative edge of the clock. This matter was taken care of by adding an inverted to the clock input to have it respond to the negative edge of the clock. The yellow rectangles in the outline these occurrences.



\* This test is located in: Library: David\_Caldera\_ADC Cell: SIM\_FF\_AS of the submitted zip file.

* Performing the down hierarchy command on Electric will go to
  + 1. Library: ADC ---Copy Cell : FF\_MS ( Master-Slave FF configuration)
  + 2. Library: ADC ---Copy Cell: NAND3 and INV\_20

Clock Divider-8 simulation subckt schematic



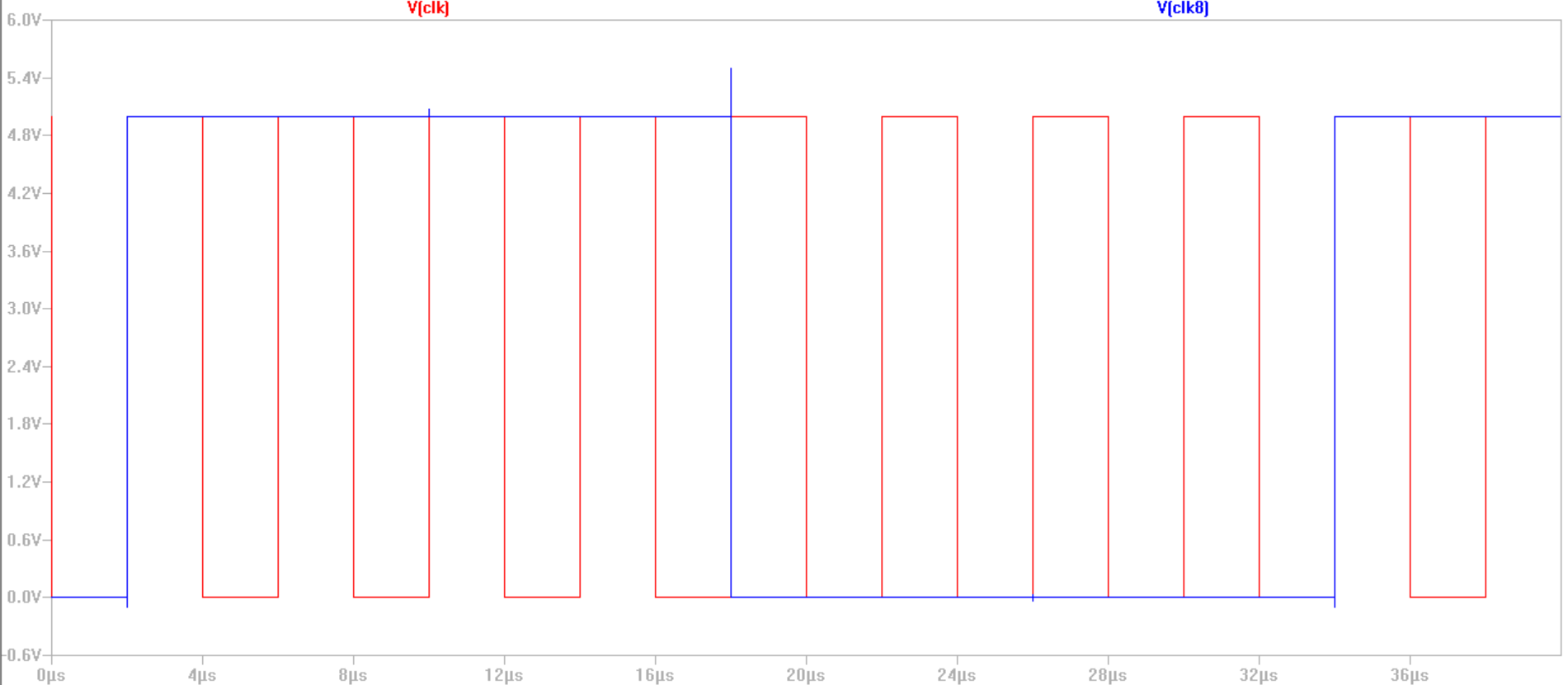
\* This test is located in: Library: ADC---Copy Cell: CLK\_DIV\_MS of the submitted zip file.

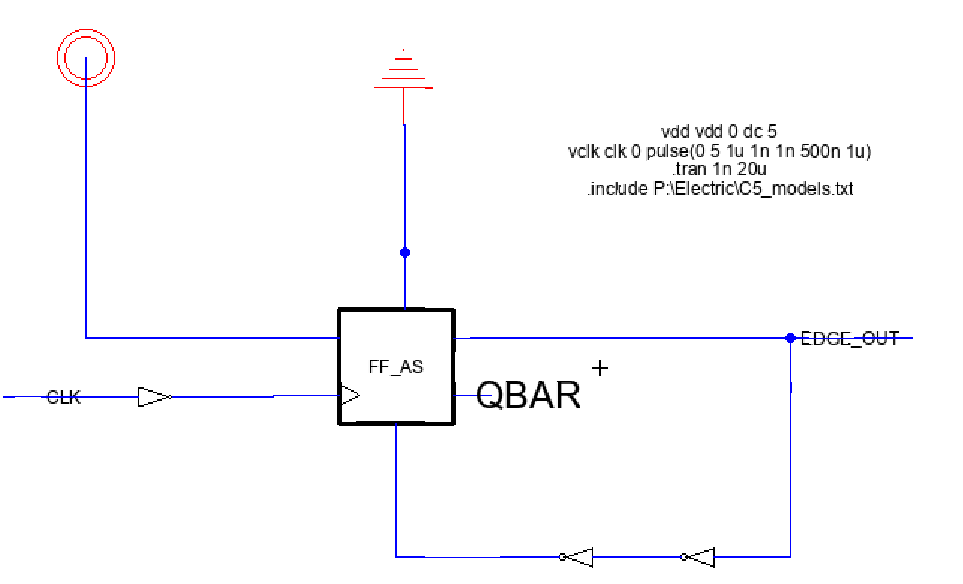
* Performing the down hierarchy command on Electric will go to
  + 1. Library: ADC ---Copy Cell : FF\_MS ( Master-Slave FF configuration)
  + 2. Library: ADC ---Copy Cell: NAND3 and INV\_20

\* This flip flop is used in the clock divider and SAR Logic

The objective of the clock divider circuit is to divide the input signal frequency by a factor of 8. The image below is the spice simulation of the clock divider with a pulse with a fixed frequency being fed as the input and the output being a pulse signal with a frequency 8 times less than that of the input.

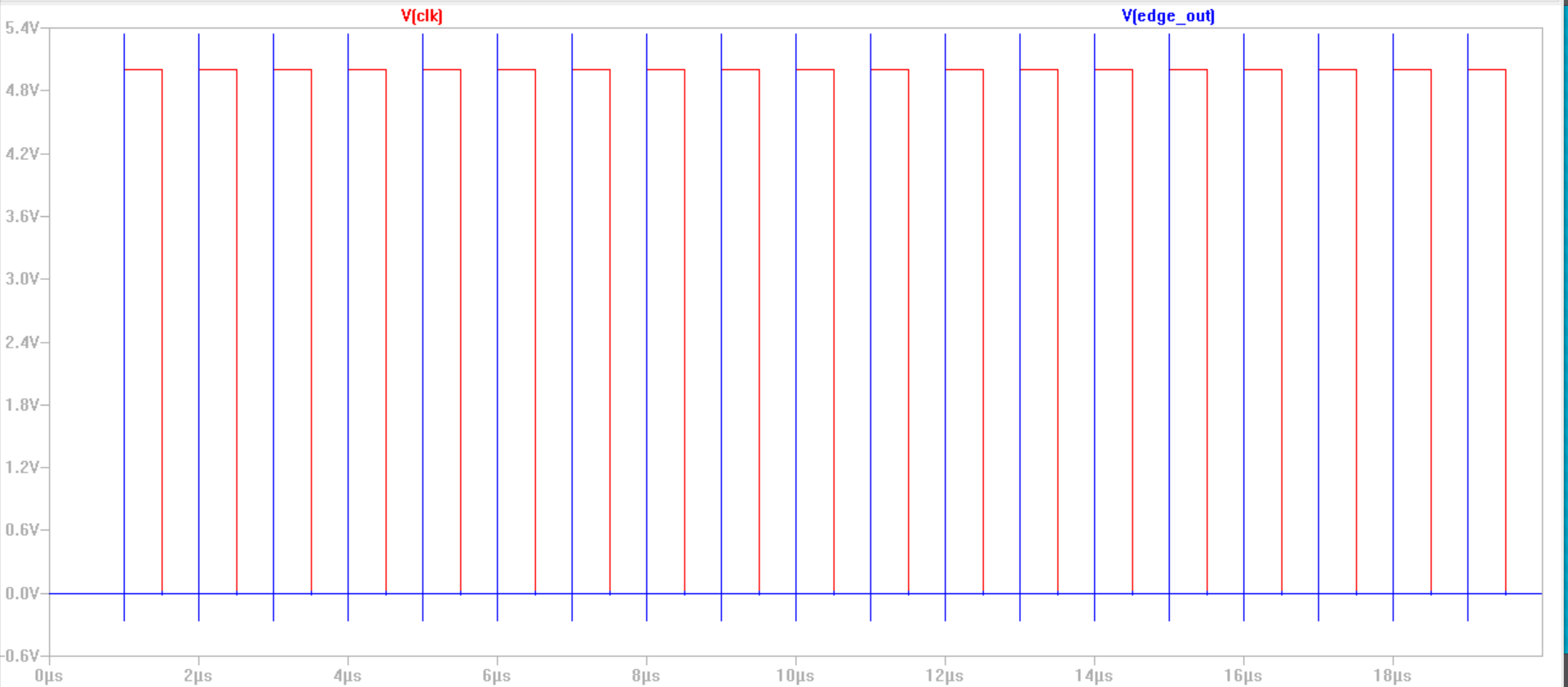
**Electric VLSI Edge Detector Schematic and testing parameters**





**Spice Edge Detector Simulation**

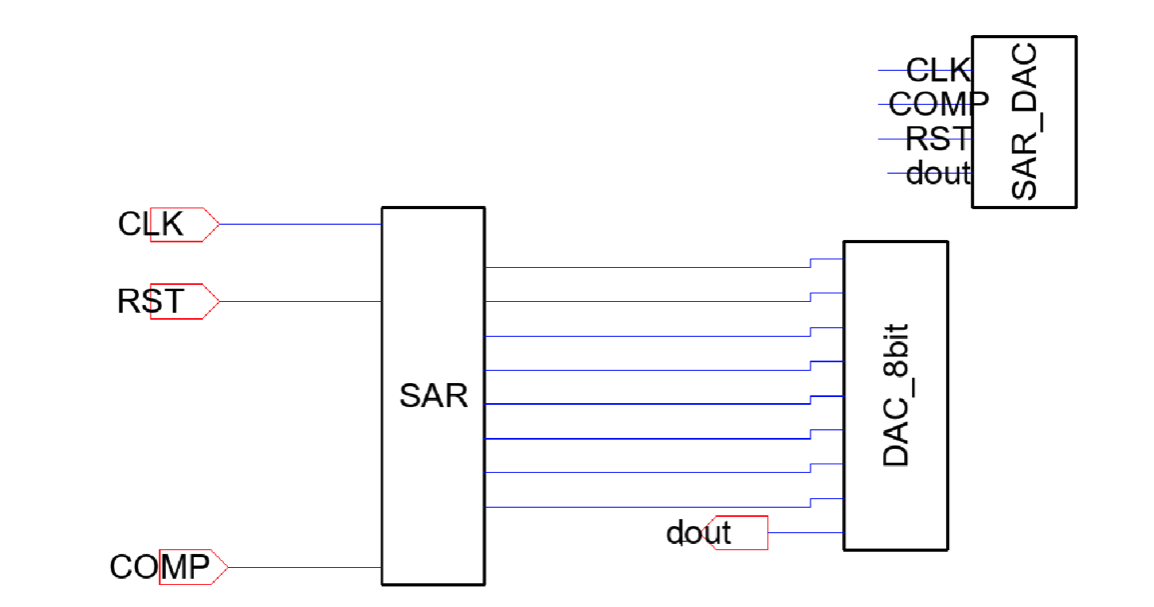
**To test the edge detector circuit a clock pulse with a 1us was given into the CLK of the flip flop. A constant 5V is being fed into the D input of the flip flop. This circuit was designed to give a short impulse at the positive edge of the slow clock. The simulation below shows the output, edge\_out, to have a short 5V impulse at every positive edge of the CLK input.**



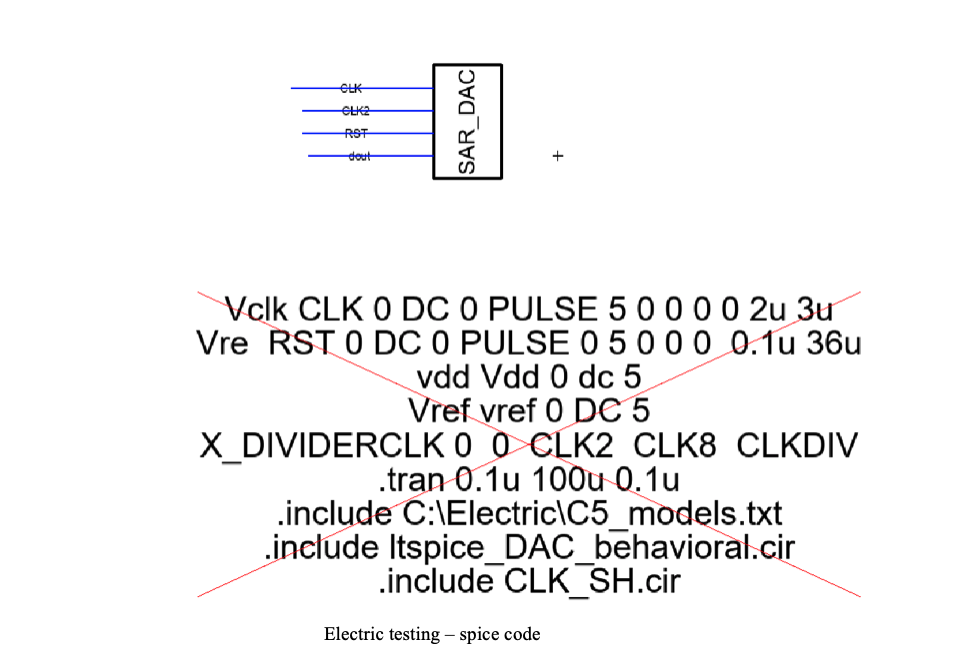
**3.3 System Demonstration**

The next figures show the processing of adding sub-circuit together to create a complete DAC-SAR register circuit.

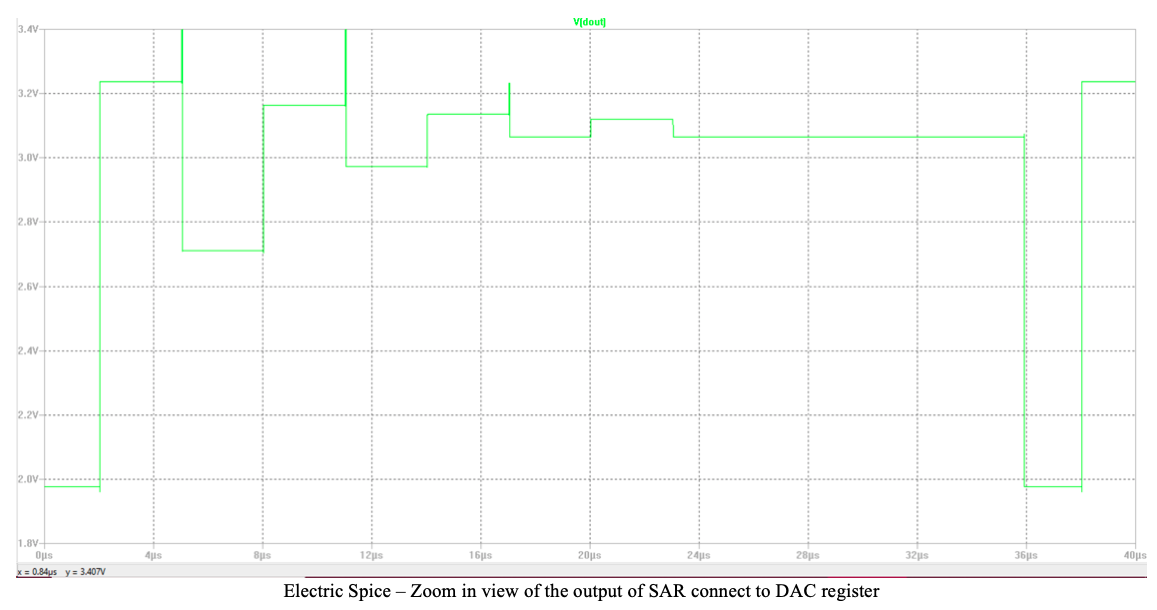
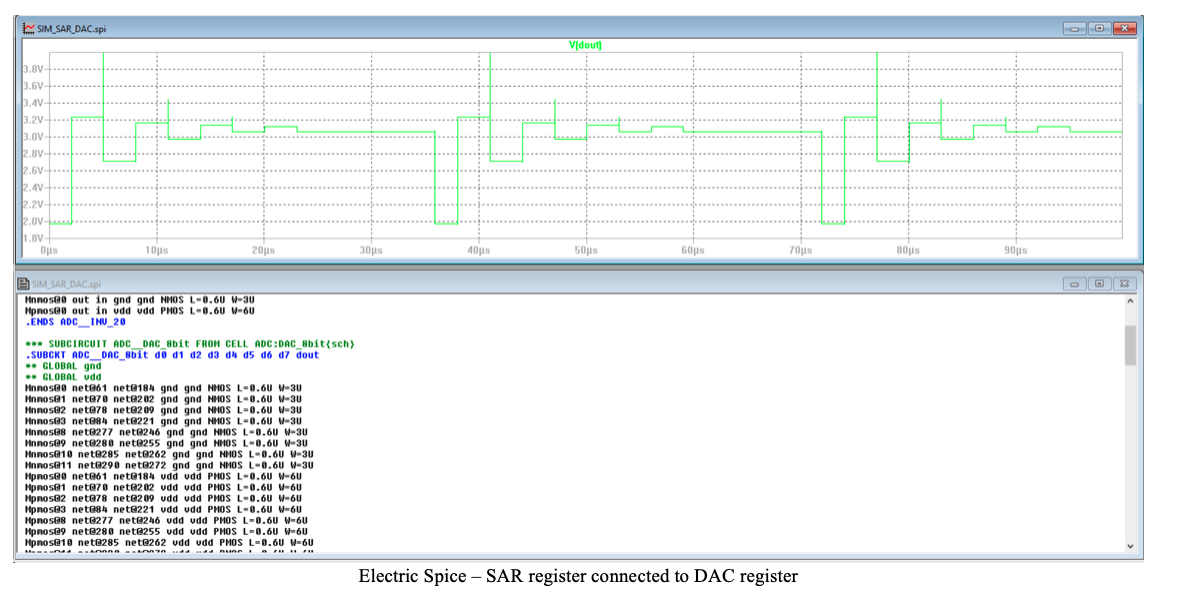
The SAR register and the DAC register are connected by passing the output of the SAR as the input of the DAC, see figure below. The circuit icon for this consisted of three input (clk, rst, and comp) and one output (dout).



To test this connected used the behavioral circuit created in milestone 2 as a subcircuit input into Electric, see figure below.ss



I ran the connected SAR and DAC register in Electric using Spice, see figure below



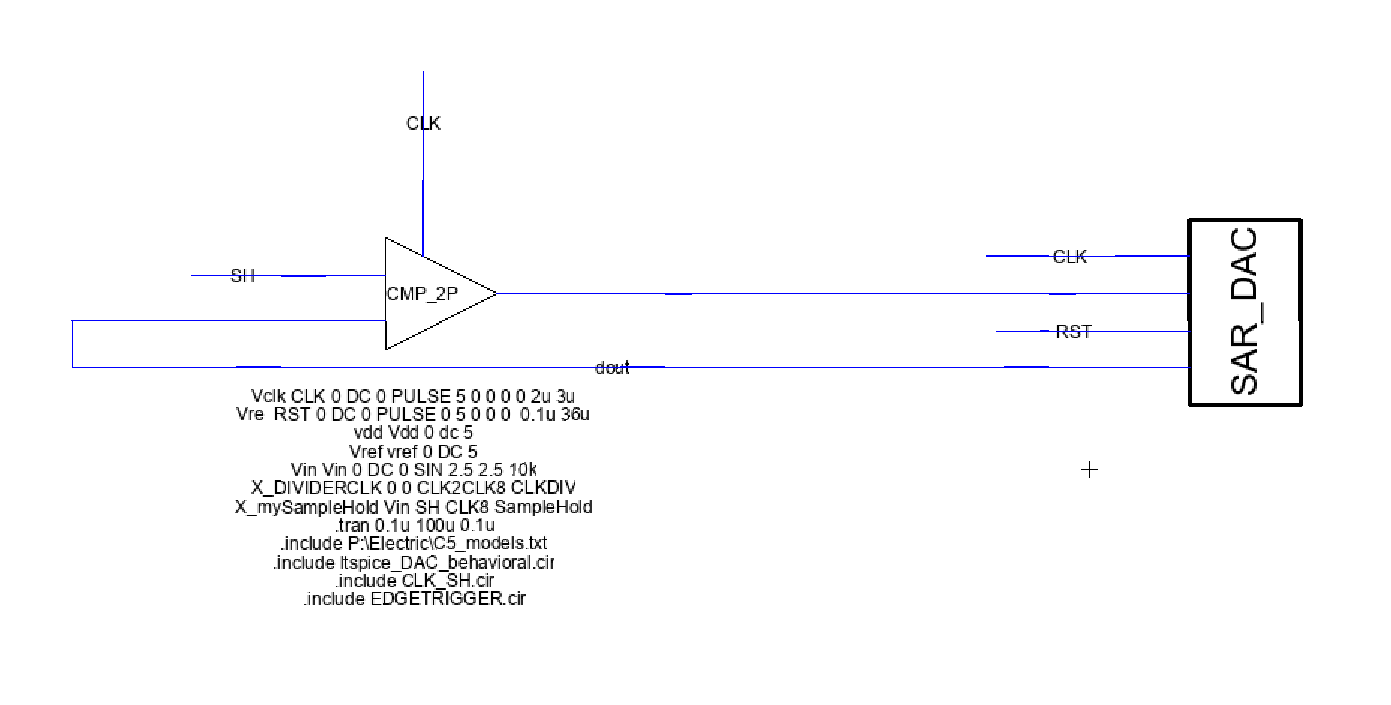
As shown in the above figures the design requirement was met and the circuit pass all test performed by Electric Spice and as a subcircuit to the circuit created in milestone 2.

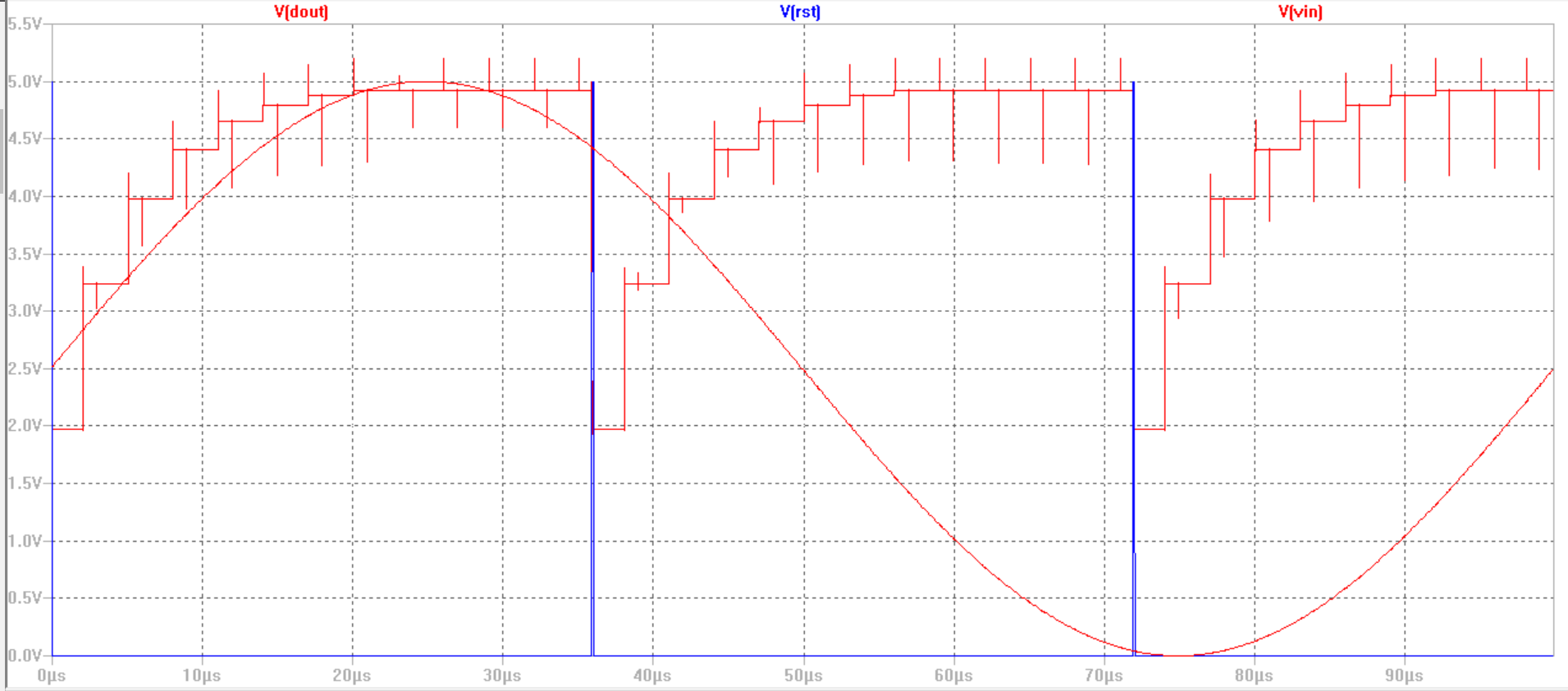
To access the file used to perform these tests. Please follow the steps below.

1. open Electric
2. From Electric, locate the file icon and select open library, or click ctrl O
3. Locate the unzip file and click it
4. In the unzip file locate SAR\_8Register.jelib and click open
5. In electric Explorer under Libraries the SAR\_8Register should be there. If not repeat step 1 – 5.
6. Click the + icon next the SAR\_8Register
7. To view the schematic and icon DAC with the SAR register connected together
   1. locate the file title SAR\_DACand click the + icon next to it
   2. SAR\_DAC{sch} to open DAC schematic
   3. SAR\_DAC{ic} to open the DAC icon
   4. To run test on the DAC locate the file “SIM\_SAR\_DAC{sch} and run spice code.

**Comparator, SAR, and DAC circuits simulation.**

Here is the furthest implementation of the ADC that we managed to reach in the time frame given to us. The functioning SAR and DAC circuit are tied to one block and connecting that output to the comparator.





**4. DISCUSSION**

The most challenging part of the design process was using the Electric VLSI software. Throughout the semester, most of the roadblocks we went through while working on the project had to do with Electrics poor user interface. The wiring scheme that the software offers makes it difficult to make simple circuits due to lag and sensitivity of user input.

The last test we managed to run before the milestone was due was connecting a functional SAR block and DAC to the comparator. The output seemed to be iterating through the SAR logic but the DAC output incremented to 5V every 8 cycles of the master clock. This is the result of the comparator giving a high output for every cycle of the fast clock. The two areas that may have caused this issue are design flaws in the comparator, or a non-synchronization issue between major functional blocks. Due to time constraints we were unable to perform extensive testing on each of different blocks most likely being the cause of not being able to get the functioning blocks to work together.

**5. CONCLUSION**

The overall project of designing an ADC was a meaningful project as it helped reinforce concepts in our other senior level classes. In particular, spending the semester working on the ADC has helped get a more intuitive understanding of the data acquisition systems used in Digital Signal Processing and Senior Design. This project has helped us have a more intimate understanding of specifications such as sampling rate, resolution, and accuracy. Unfortunately we were unable to complete the overall objective of designing a fully functional 8-bit SAR ADC on the transistor level. Even though we were unable to finish the project in the time frame, I do intend to follow-up with this project and complete it to the later.

A topic that may be interesting for this class is the development of a Phase Lock Loop system. While developing the timing circuitry for this project and limiting the sample rate due to having to slow the master clock. It would be interesting to develop circuitry that speeds up the internal clock such as phase lock loop circuitry does in microcontrollers. Any theme that correlates directly with other class will help drastically to get students interested and involved. The circuitry involved in the microcontrollers that are used in Embedded Systems such as UART, PLL, I2C, or other similar projects on a miniature scale would be interesting.