An Energy-Efficient Comparator With Dynamic Floating Inverter Amplifier

Proposed paper from here.

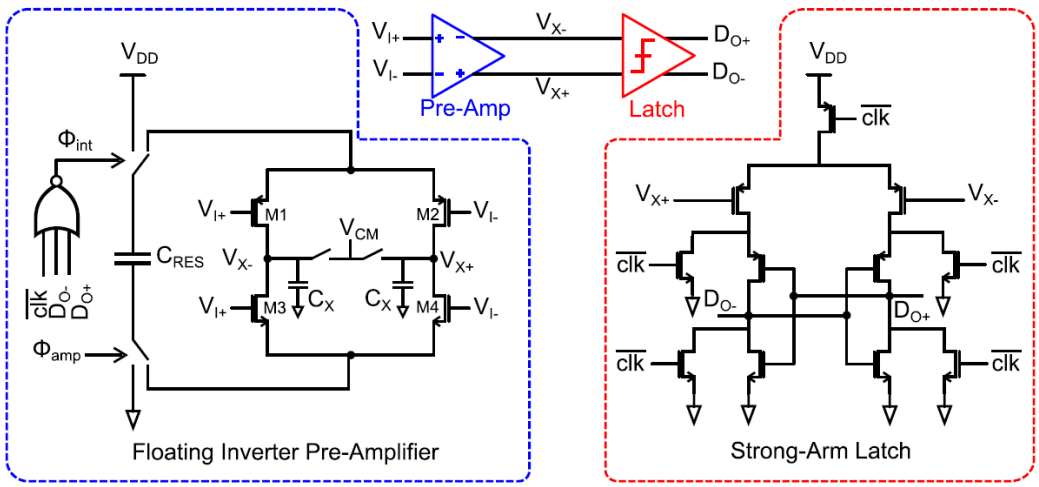
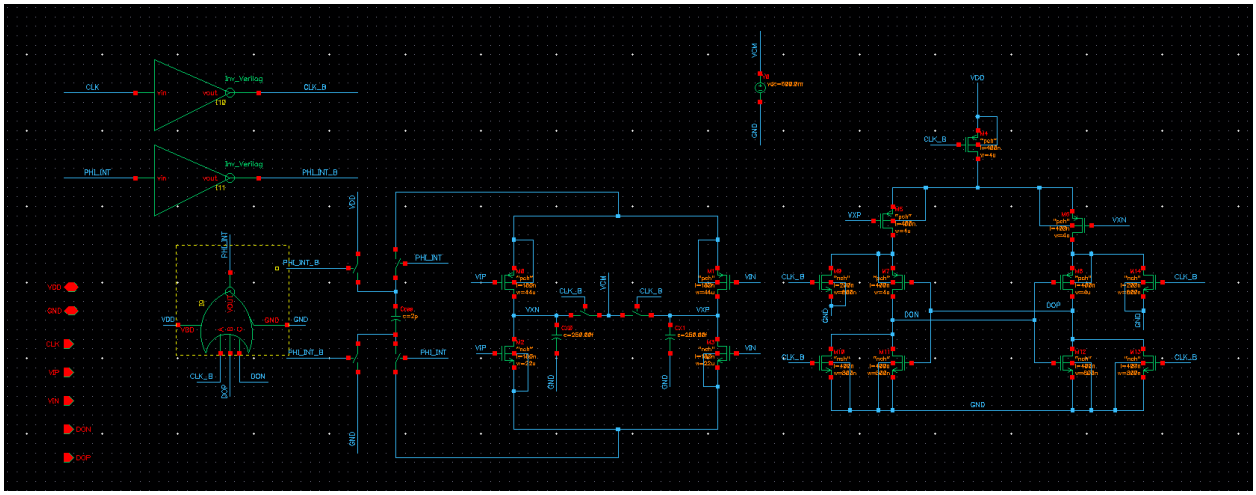


Fig. 2. Schematic of the proposed comparator with FIA.

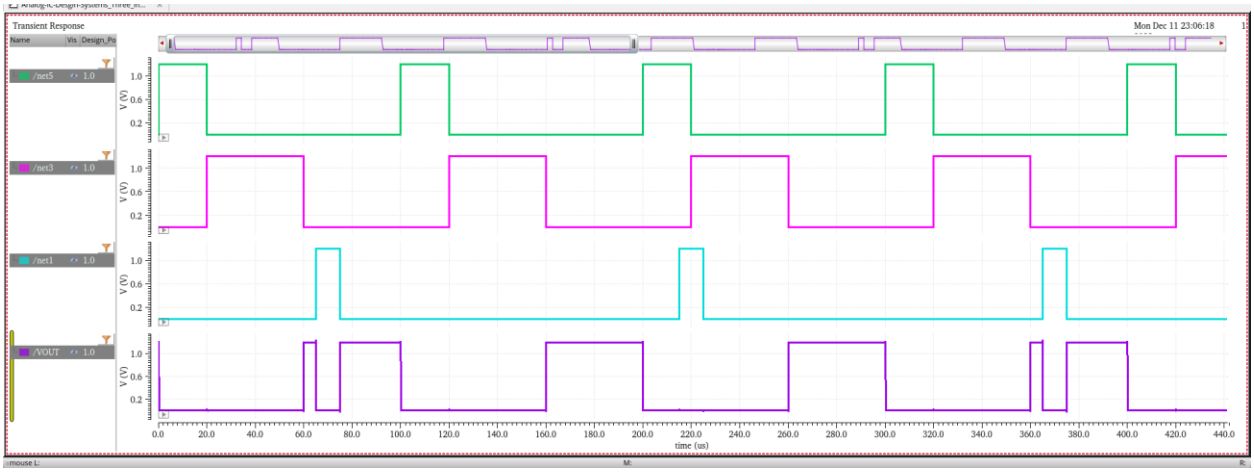
Schematics :



Sizing : NMOS in preamplifier -> 44u/180n PMOS -> 22u/180n

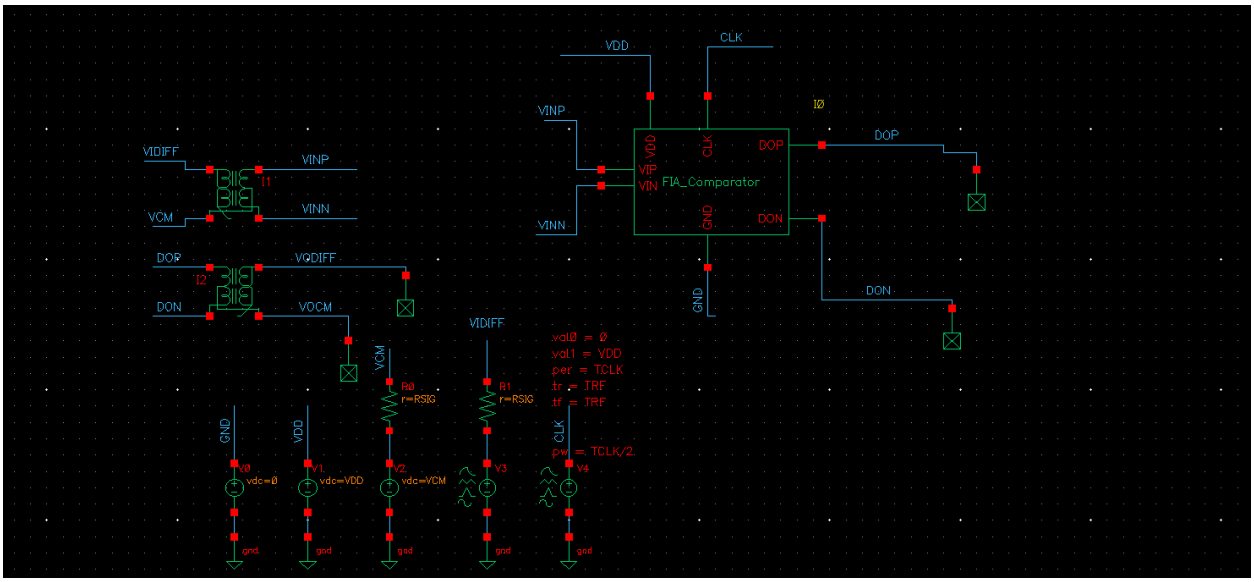
NMOS in latch ->0.8u/200n, lower transistor 0.8u/400n PMOS -> 4u/400n

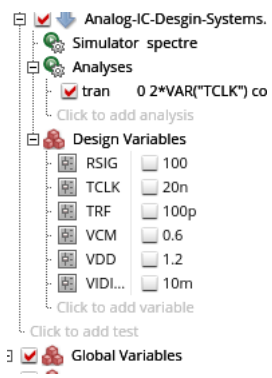
First testing the Three input NOR gate :



then the Comparator test bench :

First we will use a dc differential input:

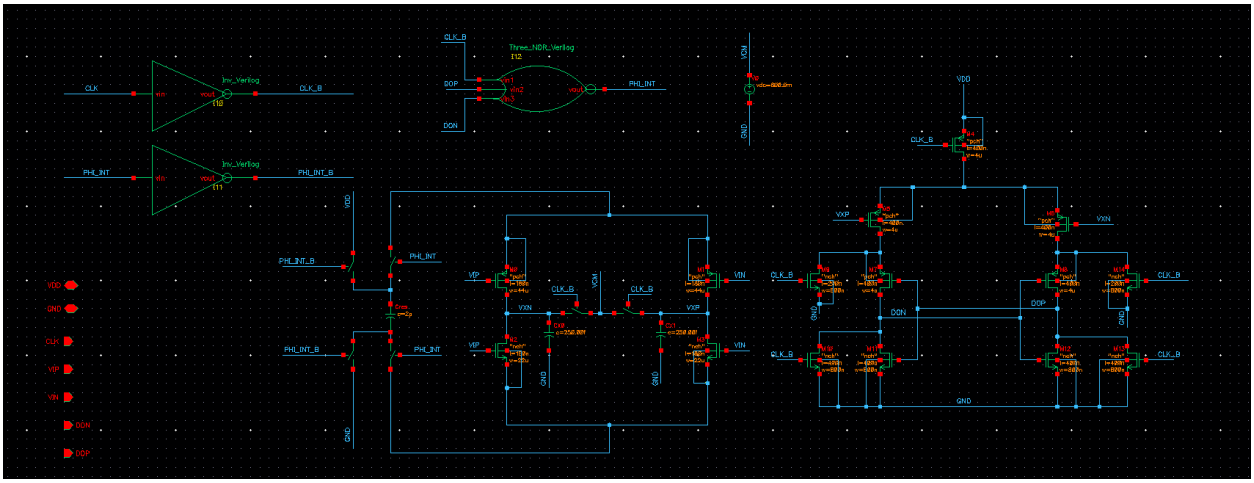


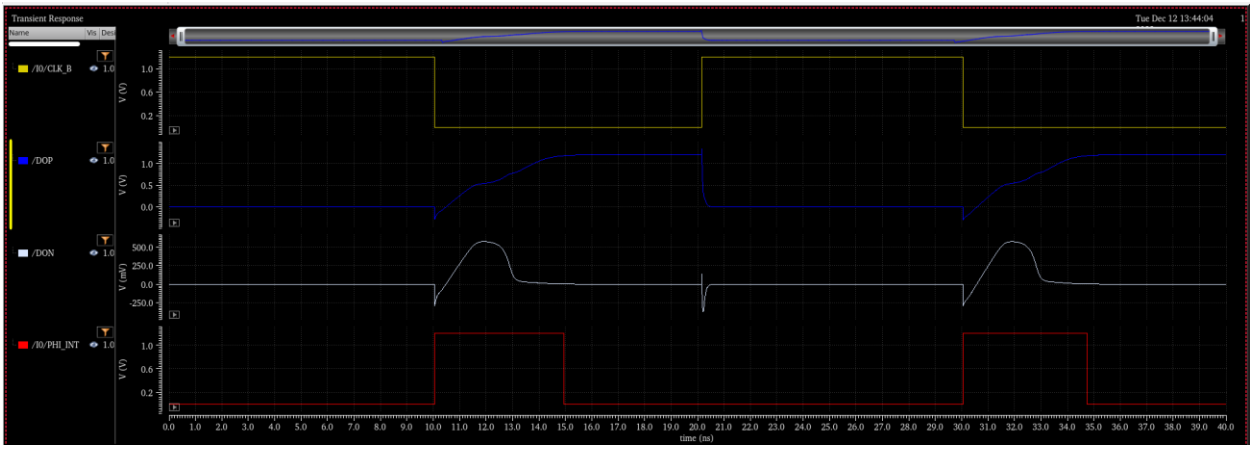


NOR gate is doing an unwanted behavior so we will try the behavioral model to check the effect of the comparator.

I0 (FIA_Comparator) | maestro | I12 (Three_NOR_Verilog)

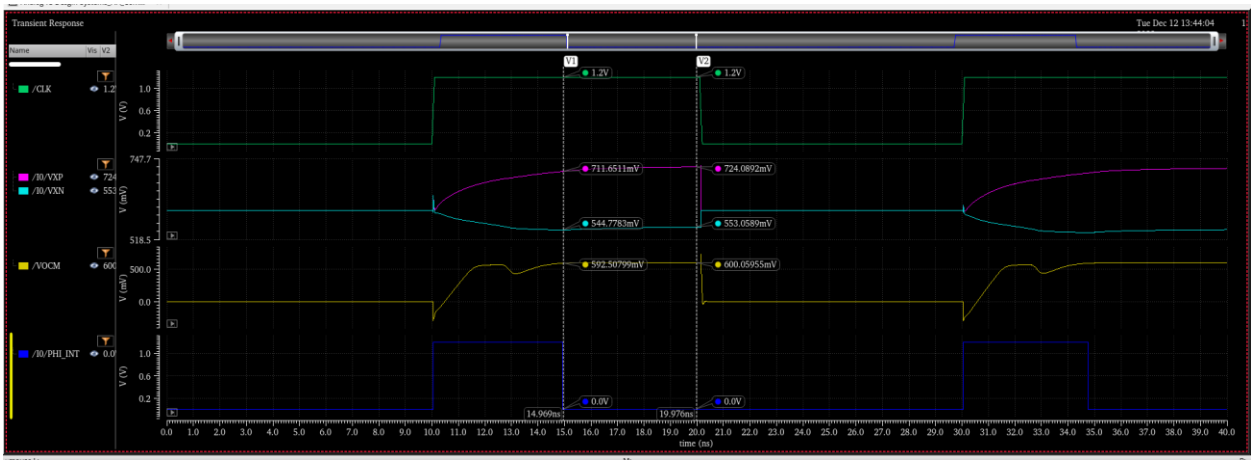
```
`include "discipline.h"
`include "constants.h"
// INSTANCE parameters
// vlogic_high = output voltage for high [V]
// vlogic_low = output voltage for high [V]
// vth = voltages above this at input are considered high [V]
// tdel, trise, tfall = {usual} [s]
//
module three_input_nor_gate(vin1, vin2, vin3, vout);
input vin1, vin2, vin3;
output vout;
electrical vin1, vin2, vin3, vout;
parameter real vlogic_high = 1.2;
parameter real vlogic_low = 0;
parameter real vth = 1.18;
parameter real tdel = 2p from [0:inf);
parameter real trise = 1p from (0:inf);
parameter real tfall = 1p from (0:inf);
real vout_val;
integer logic1, logic2, logic3;
analog begin
logic1 = V(vin1) > vth;
logic2 = V(vin2) > vth;
logic3 = V(vin3) > vth;
@ (cross(V(vin1) - vth, 1)) logic1 = 1;
@ (cross(V(vin1) - vth, -1)) logic1 = 0;
@ (cross(V(vin2) - vth, 1)) logic2 = 1;
@ (cross(V(vin2) - vth, -1)) logic2 = 0;
@ (cross(V(vin3) - vth, 1)) logic3 = 1;
@ (cross(V(vin3) - vth, -1)) logic3 = 0;
// define the logic function.
vout_val = (!(logic1 || logic2 || logic3)) ? vlogic_high : vlogic_low;
V(vout) <+ transition(vout_val, tdel, trise, tfall);
end
endmodule
```



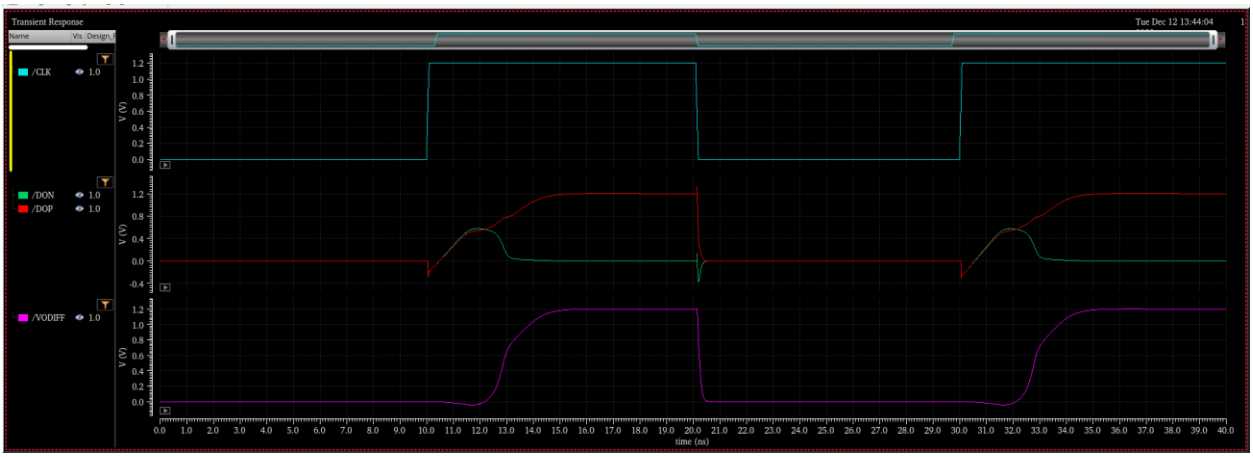


Seem to have better effect.

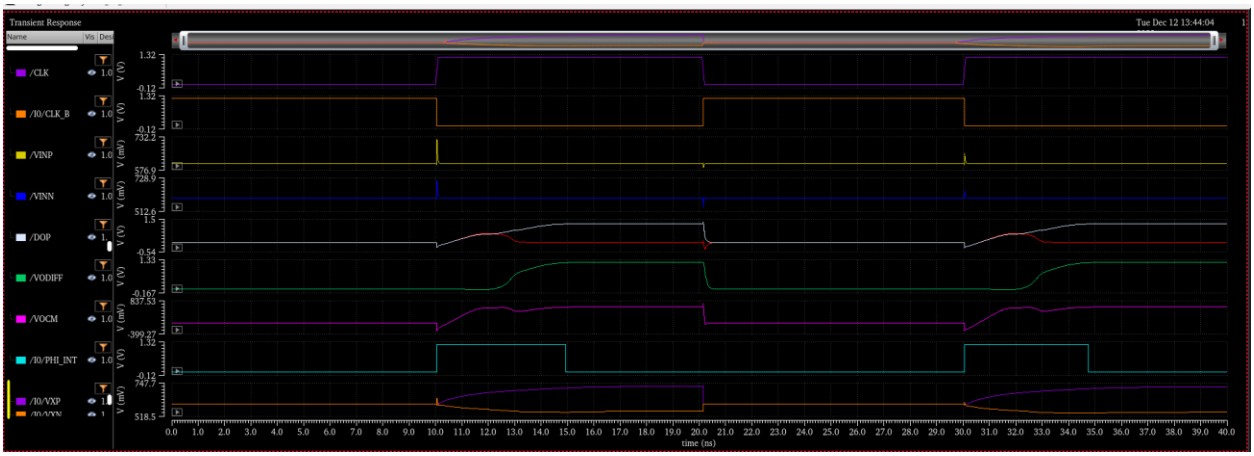
let's study separate transient signals (we will operate on the nominal corner to make sure of the operation of the basic comparator itself):



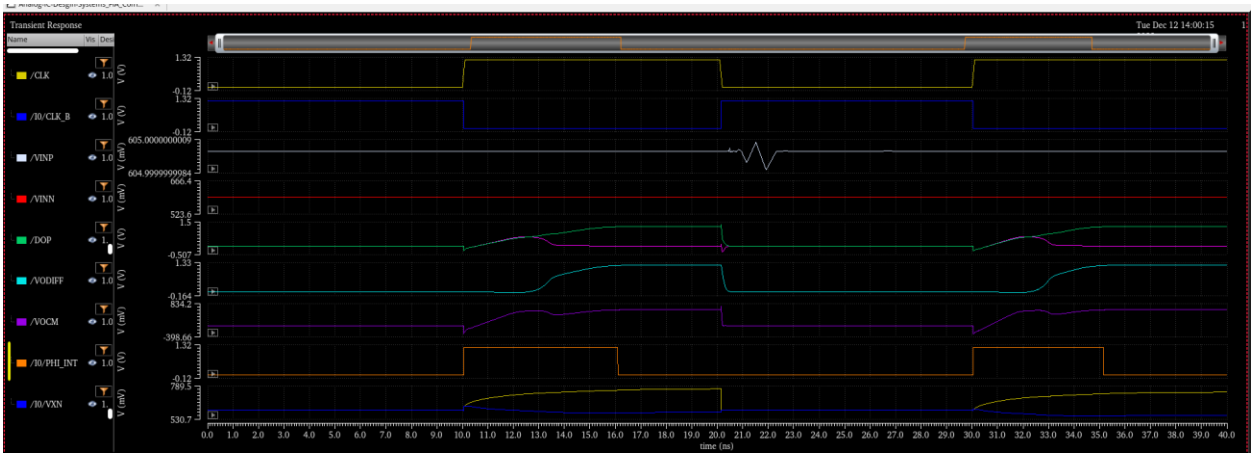
VXP and VXN shifting oppositely keeping VCM constant as we can see.



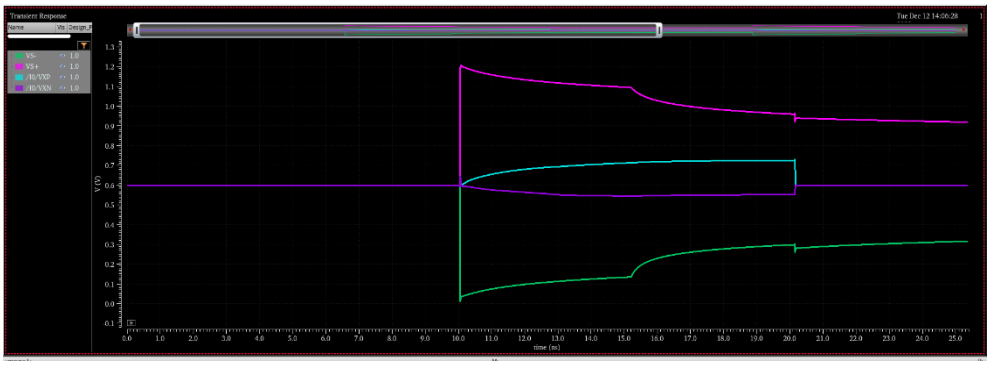
Latch regeneration operation.

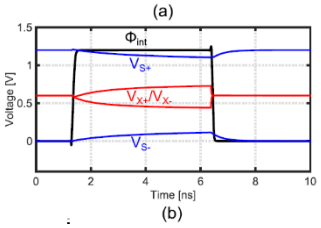


Effects at RSIG=100 ohm showing glitches at VINN and VINP due the effect of RC coupling at the source letting to charge and discharge times.

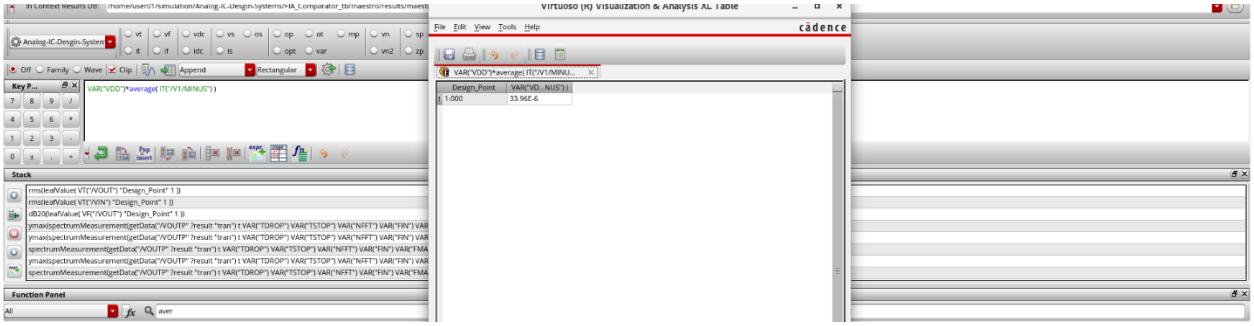


At RSIG = 0, but we will use RSIG = 100 ohm, more practical.

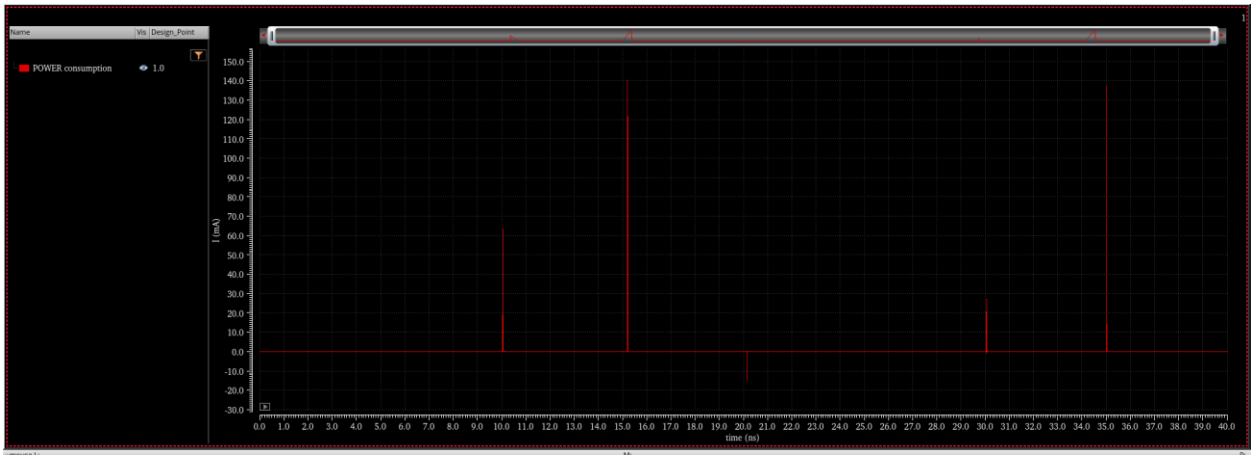




Calculating the power consumption:

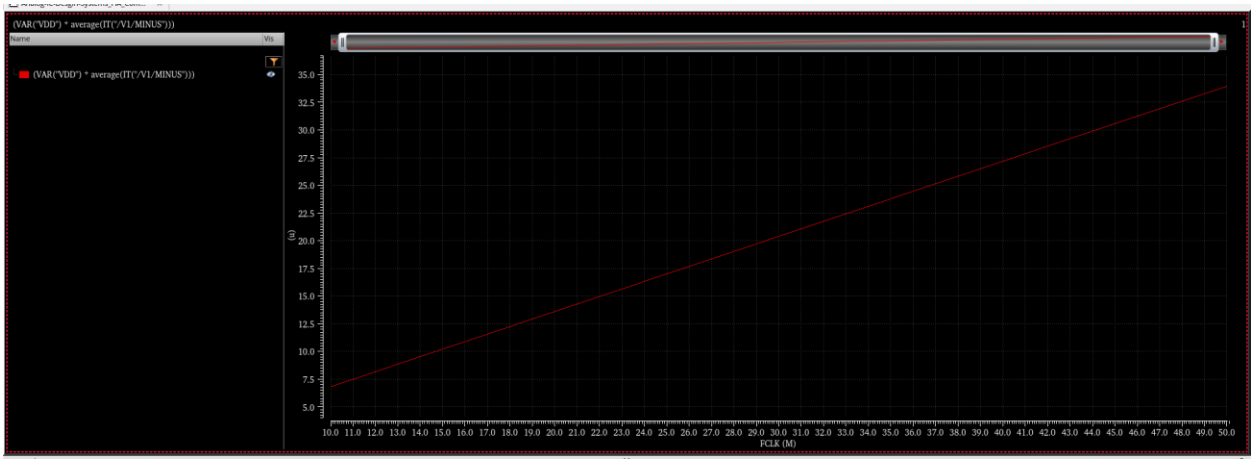


The complex draws an average power of 34uW.



Power consumption with time.

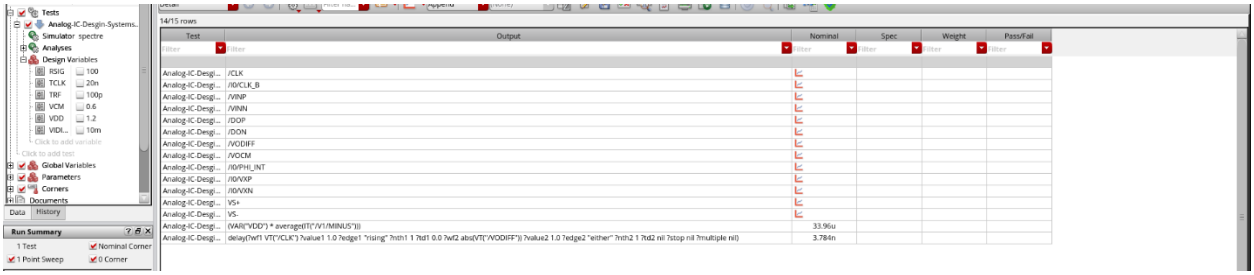
Set VDIFF to its original value. Set TCLK = 1/FCLK and set a linear sweep for FCLK 10M:10M:50M. Plot the average power consumption vs FCLK. Comment on the results.



what we can see here is that there is a linear relation between the CLK frequency (the switching frequency) and the power consumption, as the frequency increases the power consumption increases, and it is only rational because as the CLK frequency increases the switching times increase and thus the dynamic power consumption increases, what we can notice here is that the static power loss is very small, this is because this topology uses CMOS inverters that does not allow static current to pass, there is only a very small value.

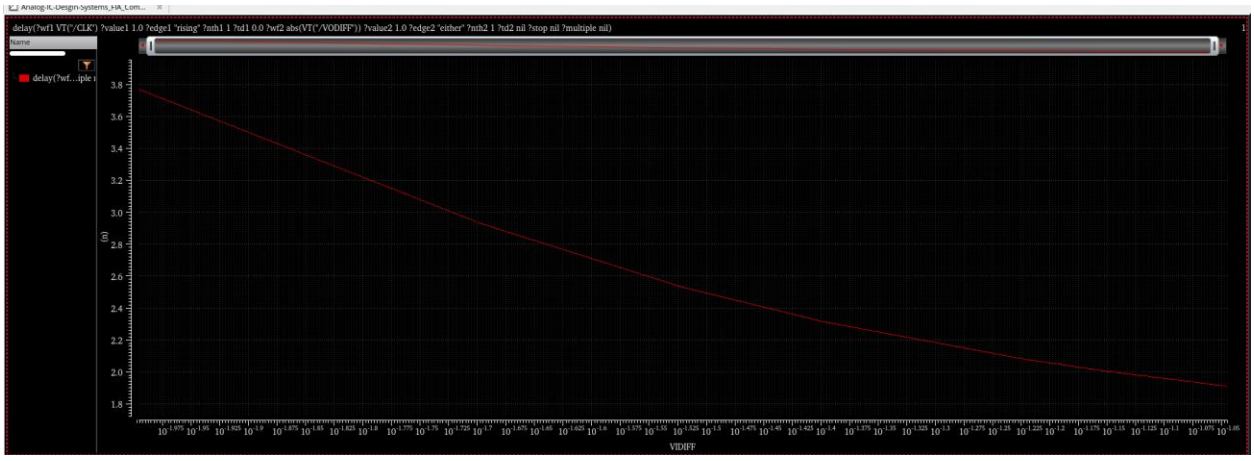
Calculate the delay using the calculator.

```
delay(?wf1 VT("/CLK") ?value1 1.0 ?edge1 "rising" ?nth1 1 ?td1 0.0 ?wf2 abs(VT("/VODIFF"))
?value2 1.0 ?edge2 "either" ?nth2 1 ?td2 nil ?stop nil ?multiple nil)
```



Set a logarithmic parametric sweep for the differential input voltage. Set VDIFF = 100uV to 100mV, 10 steps, logarithmic.

Run transient analysis and plot the delay vs VDIFF. Use log axis for VDIFF. Comment on the results.



Comment:

We know that the regenerative latch delay can be given by the following equation assuming full scale transition:

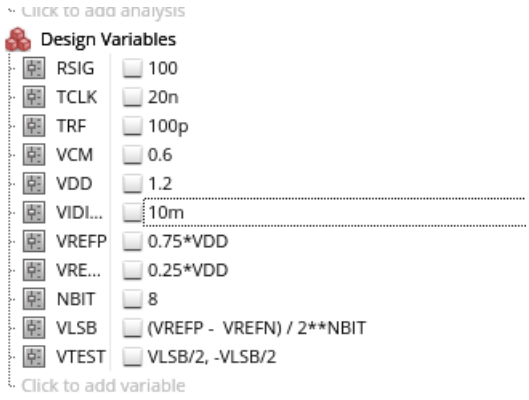
$$V(t) = \Delta V_{in} e^{+t/\tau} \rightarrow \; t_d = \tau \ln \frac{V_{FS}}{\Delta V_{in}} = \tau(\ln V_{FS} - \ln \Delta V_{in})$$

From this equation we can see the semi-linear relation between the delay time and the input voltage, as the input voltage increase the time delay required for the generation will decrease, and this is logically true if the input voltage was large it would take less time to reach the strong one or zero than when given small input voltage, also as the input voltage increases too much slewing effect comes into play and causes some sort of saturation and that is what we see at the end of the curve.

OVERLOAD (OVERDRIVE) RECOVERY TEST

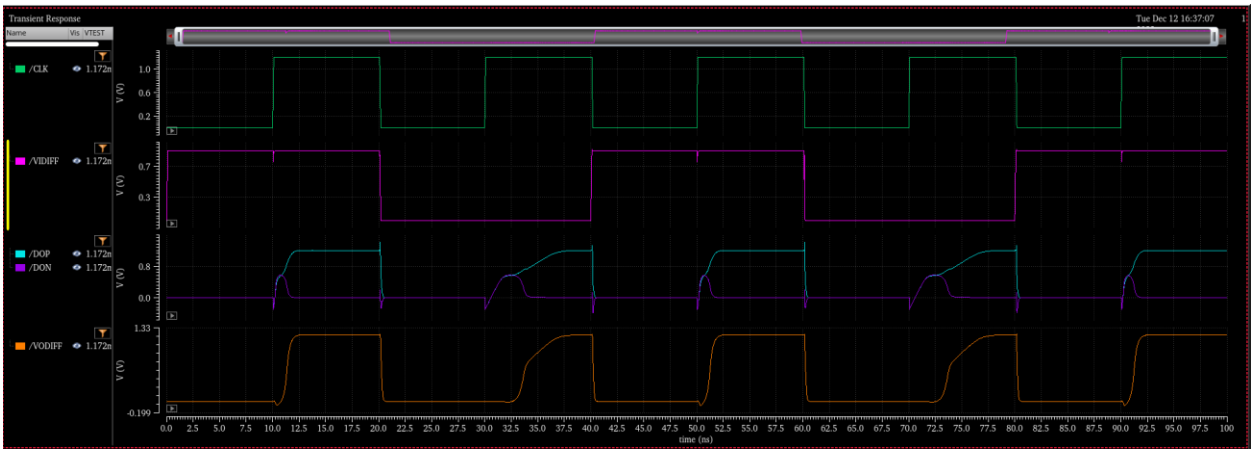
We will apply small input (±0.5 LSB) is applied to the comparator input in a cycle right after a full-scale input is applied; the comparator should be able to resolve to the right output in either case → memoryless.

Change the differential input to be a pulse source with the following parameters: one value (initial) = VREFP, zero value (pulse) = VTEST, period = 2*TCLK, rise/fall time = TRF, delay = 0, pulse width = TCLK. Assume the comparator will be used in an 8-bit ADC. Set global parameters as shown below.



After running transient analysis :

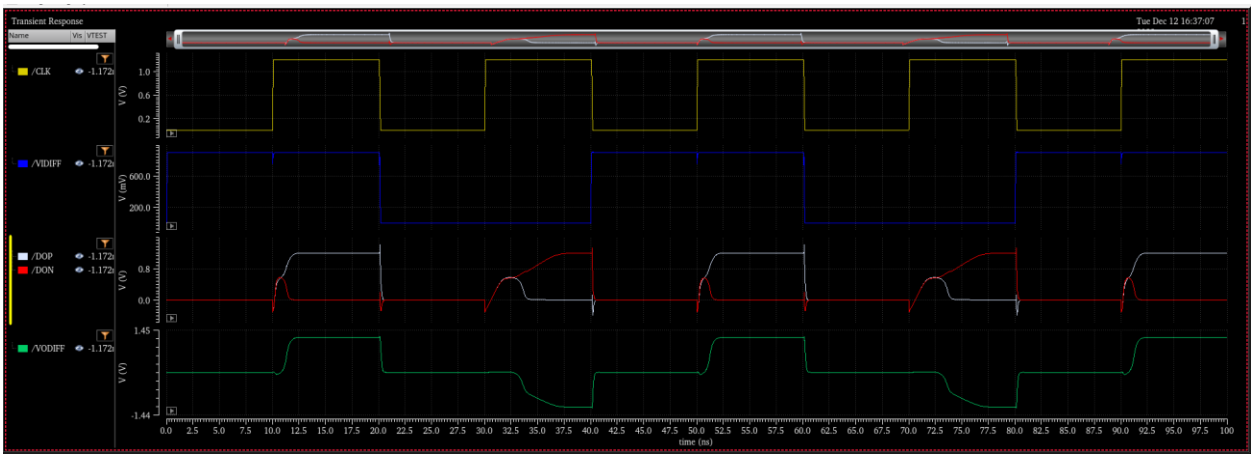
Case 1 : applying +VLSB/2 after a full-scale input:



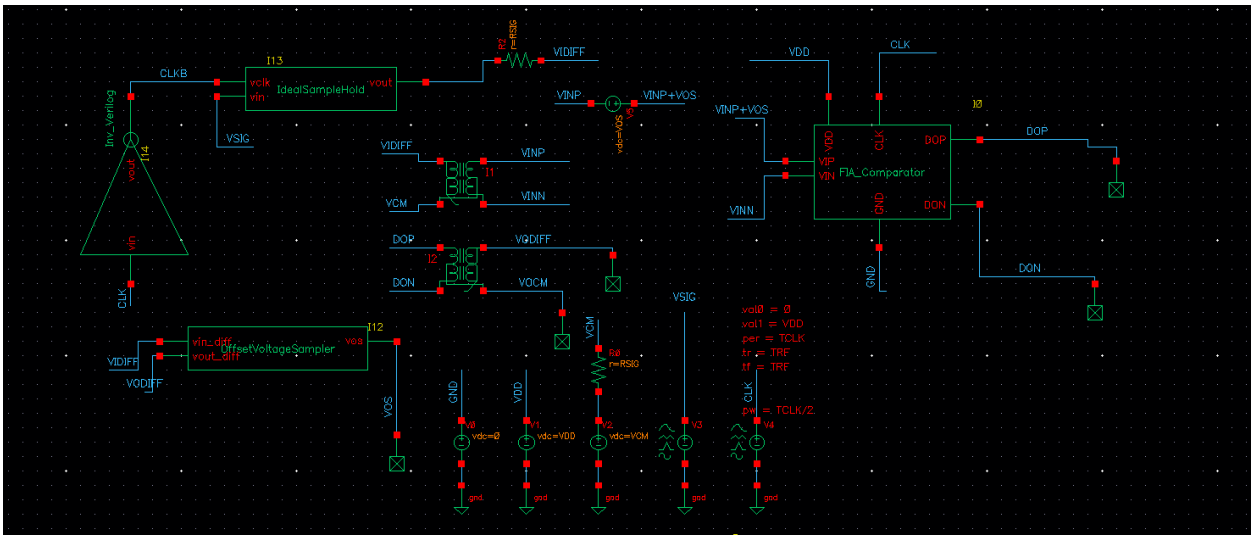
the comparator responded with the right output; this indicates no reverse hysteresis.
or the hysteresis effect is less than VLSB/2.

Case 2 : applying -VLSB/2 after a full-scale input:

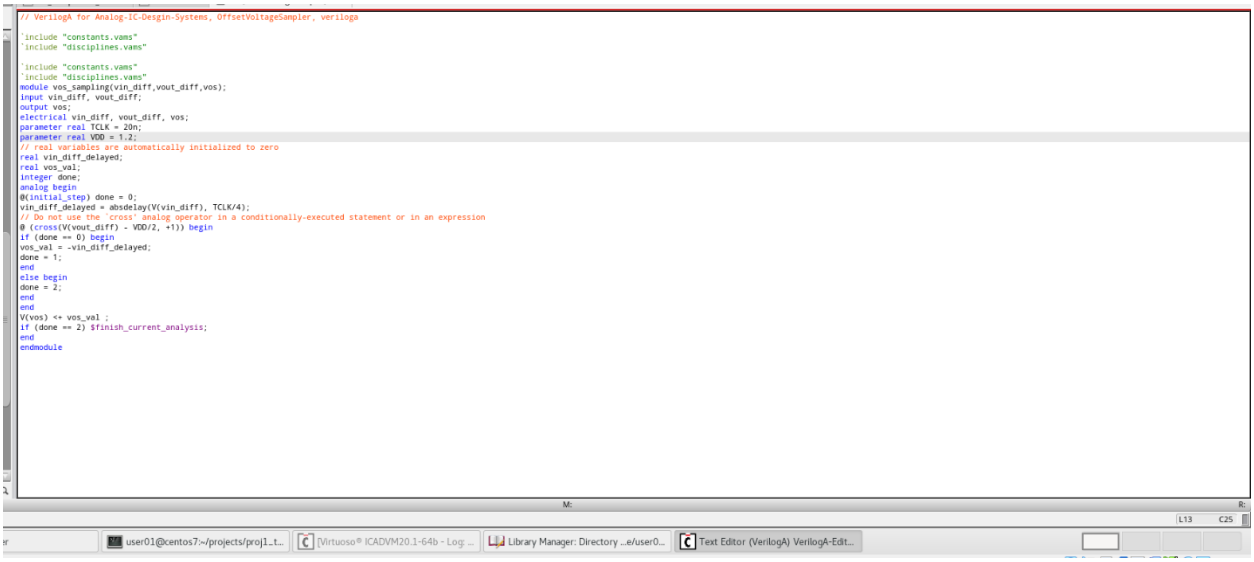
Applying a very small input after the full-scale input, the circuit responded in the right manner meaning that the design is free from hysteresis or hysteresis is less than 0.5 VLSB if you want to check just lower the value of the TEST voltage lower than 0.5 VLSB.
HINT: It's important to understand that the normal operation of the circuit is to start with the precharge phase first when the CLK = 0, so that any initial state in the circuit capacitors would not affect me, and after that continue to the amplification or the evaluation phase where the main gold is.



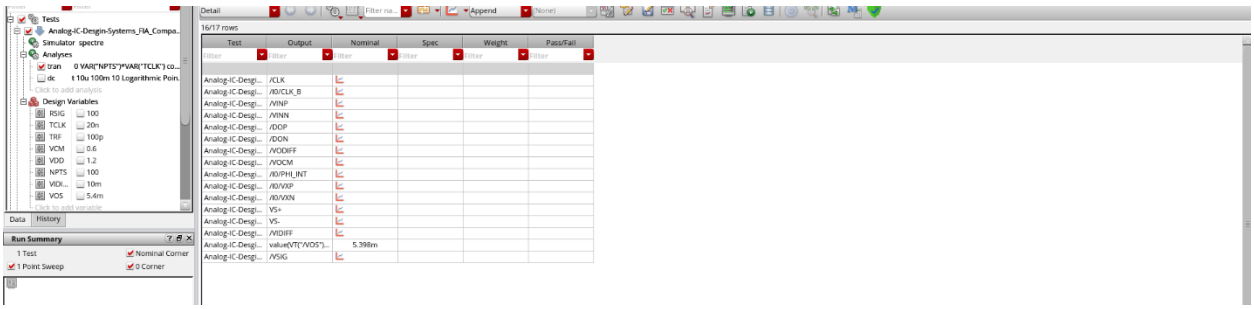
Offset voltage simulation using an artificial method.



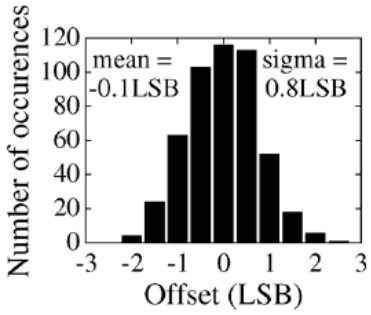
Add an ideal sample and hold circuit , and an offset voltage sampler with the following Verilog code:



```
// VerilogA for Analog-IC-Desgin-Systems, OffsetVoltageSampler, veriloga

`include "constants.vams"
`include "disciplines.vams"

`include "constants.vams"
`include "disciplines.vams"
module vos_sampling(vin_diff,vout_diff,vos);
input vin_diff, vout_diff;
output vos;
electrical vin_diff, vout_diff, vos;
parameter real TCLK = 20n;
parameter real VDD = 1.2;
// real variables are automatically initialized to zero
real vin_diff_delayed;
real vos_val;
integer done;
analog begin
@(initial_step) done = 0;
vin_diff_delayed = absdelay(V(vin_diff), TCLK/4);
// Do not use the `cross` analog operator in a conditionally-executed statement or in an expression
@ (cross(V(vout_diff) - VDD/2, +1)) begin
if (done == 0) begin
vos_val = -vin_diff_delayed;
done = 1;
end
else begin
done = 2;
end
end
V(vos) <+ vos_val ;
if (done == 2) $finish_current_analysis;
end
endmodule
```

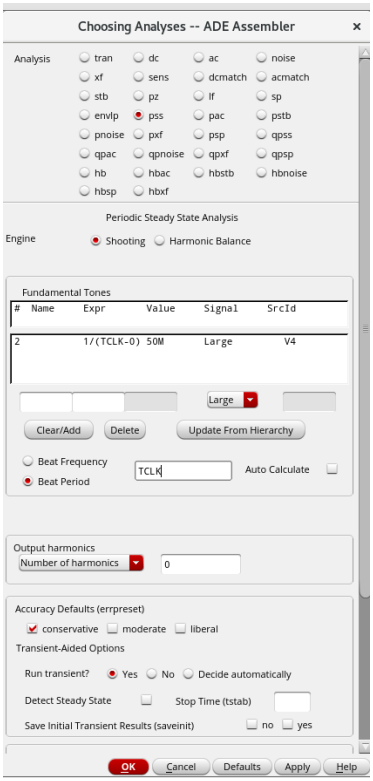


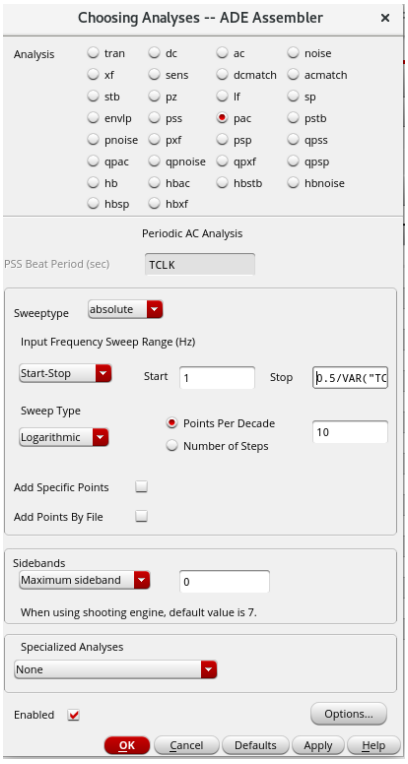
The shown offset is 5.398 mv while the offset we put is 5.4mv showing that the mean of the circuit offset is less than -0.1LSB (in mean). Of course, this is just systematic offset we still need Montecarlo simulation to check random offsets.



Checking the gain of the comparator

In this part we would like to plot the frequency response of the comparator. It is time variant (not an LTI system). Since the behavior of the circuit changes with time according to the clock signal, it is not possible to use normal ac analysis. The circuit doesn't have an operating point (dc steady state), but the clock signal which changes the circuit state is periodic (periodic steady state). A specialized type of analysis name periodic steady state (PSS) is used to analyze such type of circuits. PSS can be followed by other types of analysis such as PAC (periodic equivalent of ac analysis) and pnoise (periodic equivalent of noise analysis).

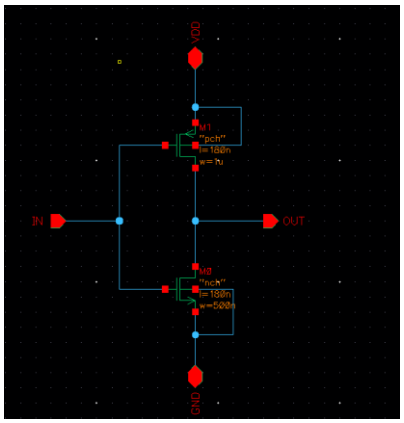


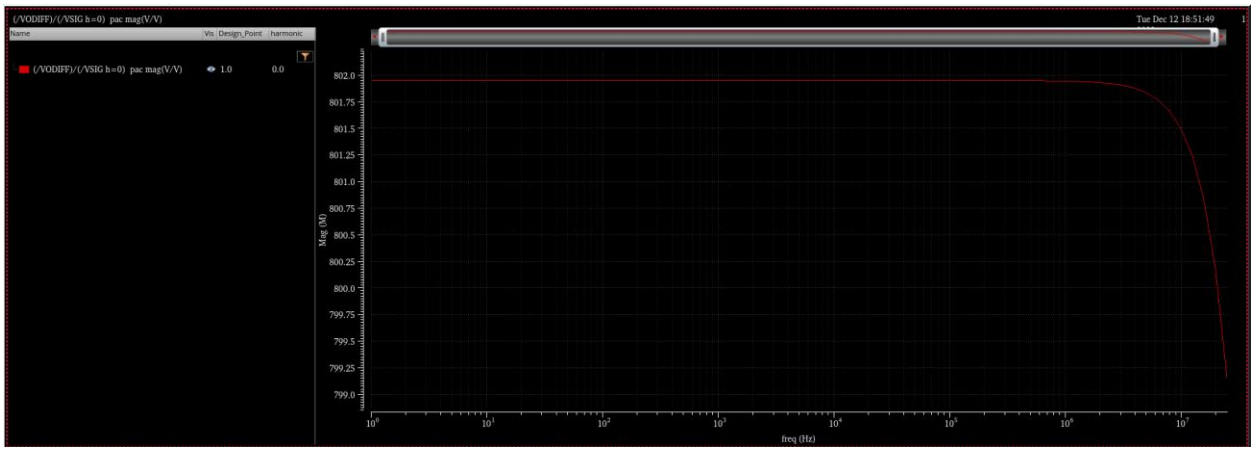


It seems that Spectre has identified an issue during the periodic steady state (PSS) analysis of the Verilog-A module named **inv_gate**. The error message indicates that the PSS analysis doesn't support behavioral module components with hidden states, and it specifically mentions the hidden state variable **vout_val** declared in line 12.

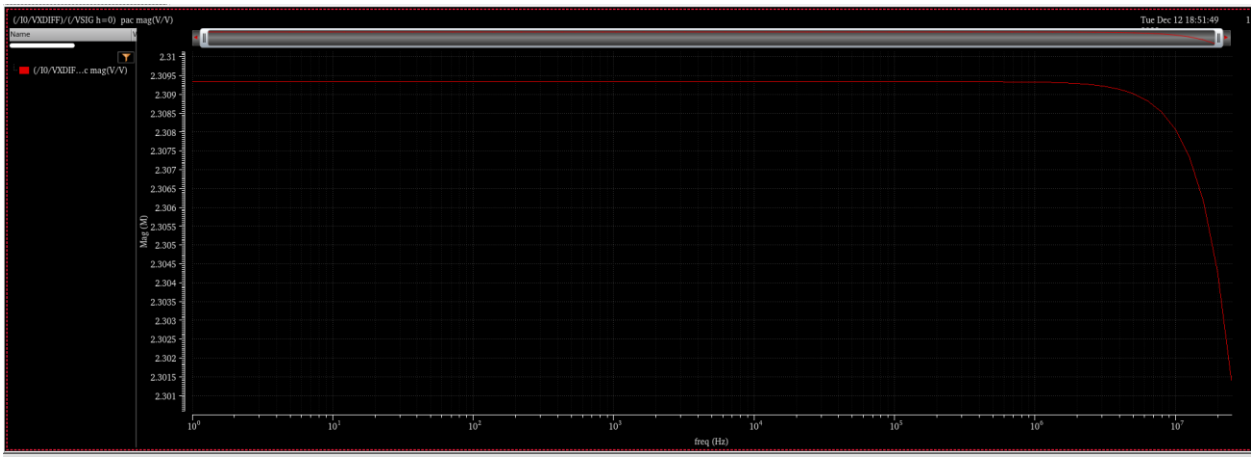
Thus, we will turn to a CMOS inverter to carry on the analysis.

```
Multithreading is disabled due to the size of the design being too small.
Error found by spectre during periodic steady state analysis `pss'.
    ERROR (SPCRTRF-15177): PSS analysis doesn't support behavioral module components with hidden states found in component 'inv_gate'.  Skipped.

        /home/user01/projects/proj1_tsmc65/Analog%2dIC%2dDesgin%2dSystems/Inv_Verilog/veriloga/veriloga.va, declared in line 12: Hidden state variable: vout_val

        Rewrite the module to remove the hidden state or if you confirm this hidden state will not affect the simulation result, you can ignore those hidden states in RF analysis by adding attribute 'ignore_hidden_state' before the module declaration. For example, (* ignore_hidden_state
```





Total gain.

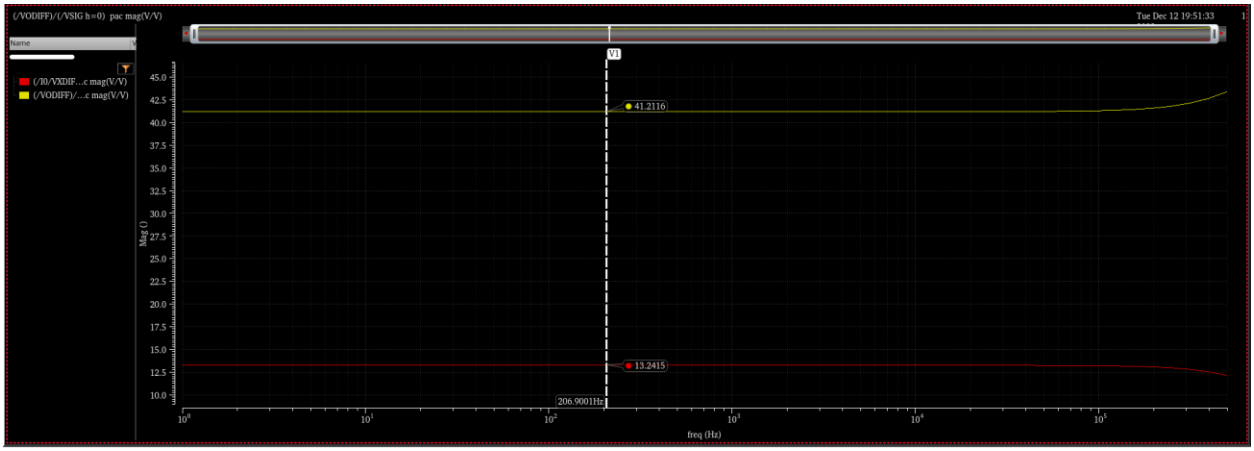


Preamplifier gain.

Comment :

The gain in the paper was 60 and due to output impedance it leads to a gain of 30, but something wrong happened, I believe I am supposed to calculate the gain of non-LTI systems like this, but I believe the error is with how I conduct the simulation.

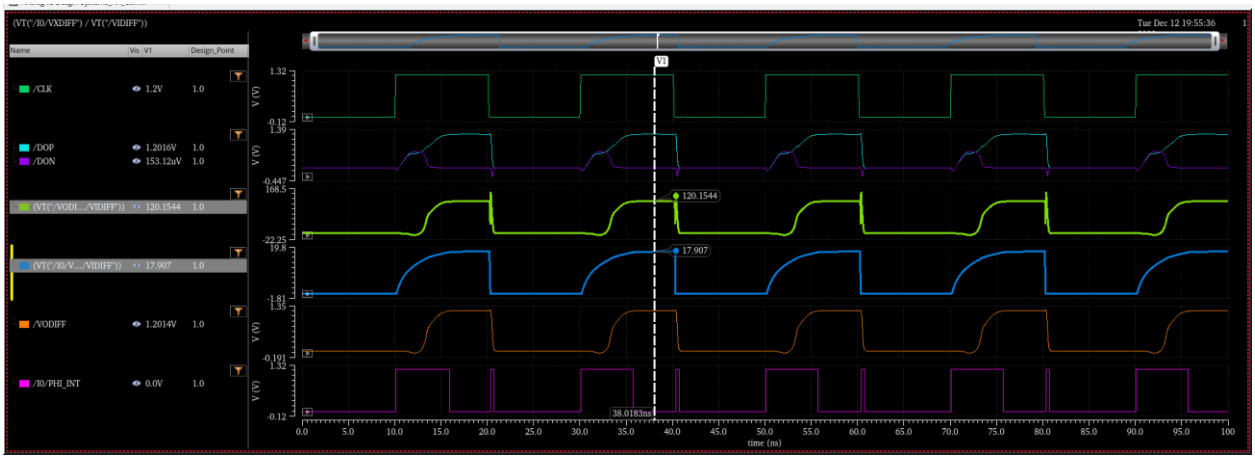
Turns out the gain is dependent on the CLOCK period, and since we chose a clock of TCLK = 20ns, that is very small which gave us these results, let's try another clock period and see.



This is at TCLK of 1us, see how the gain changes in this simulation.

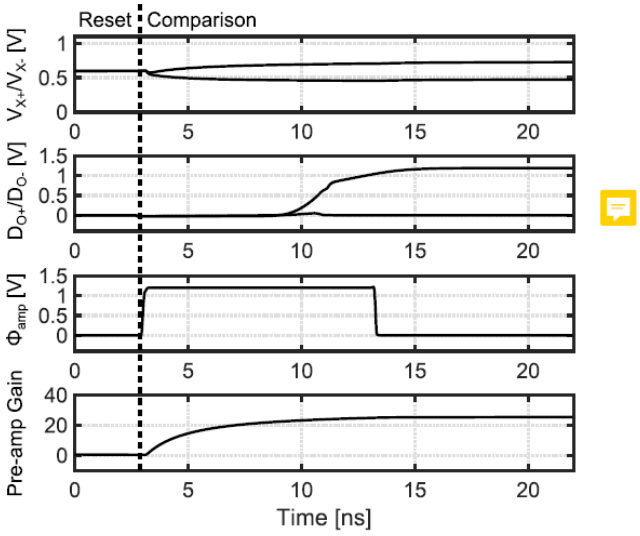
Let's Try another method of gain calculation :

We will try to plot the preamp gain and the total comparator gain vs time at TCLK = 20ns,

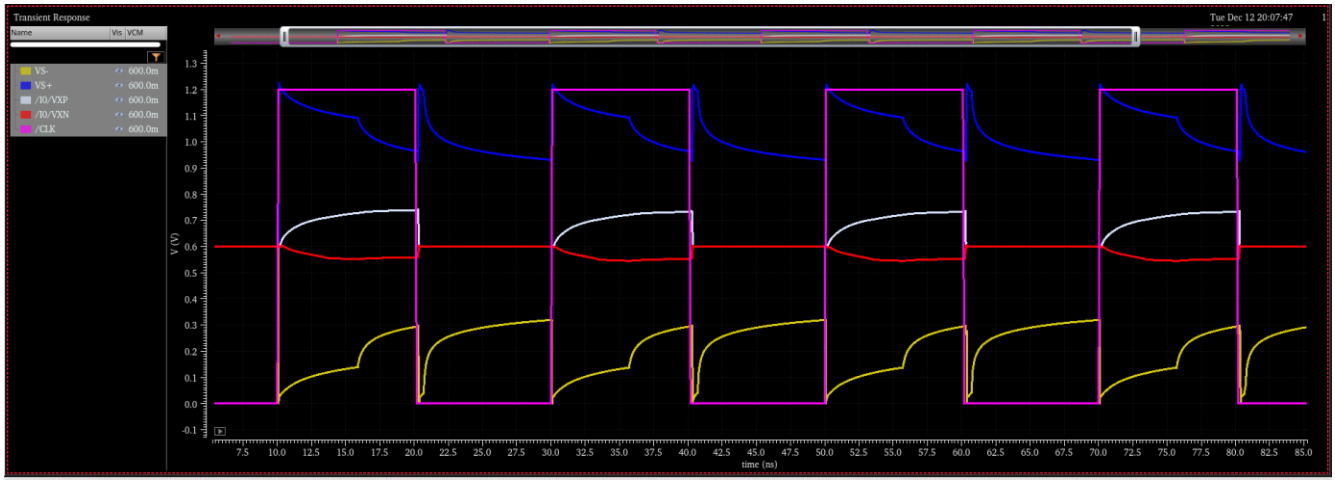


Total comparator gain is 120 v/v, and the preamp gain is 18 v/v.

The analysis in the paper is pretty much similar to the analysis here in figures, although the numbers could be a little off, but this might be due to design accuracy in several parts.

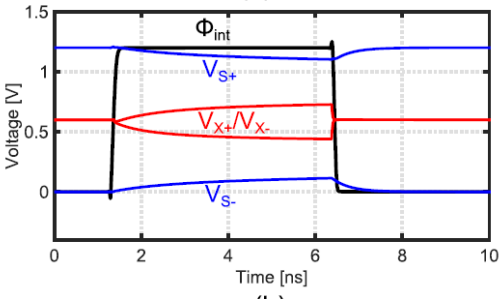


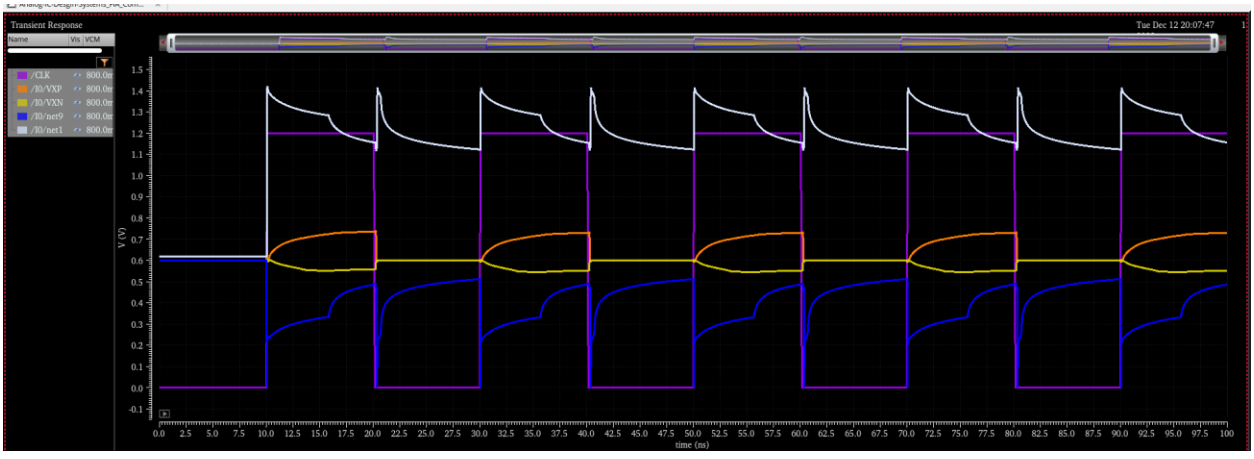
One last thing we will discuss is the input common mode insensitivity:



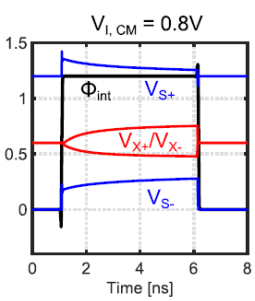
(a)

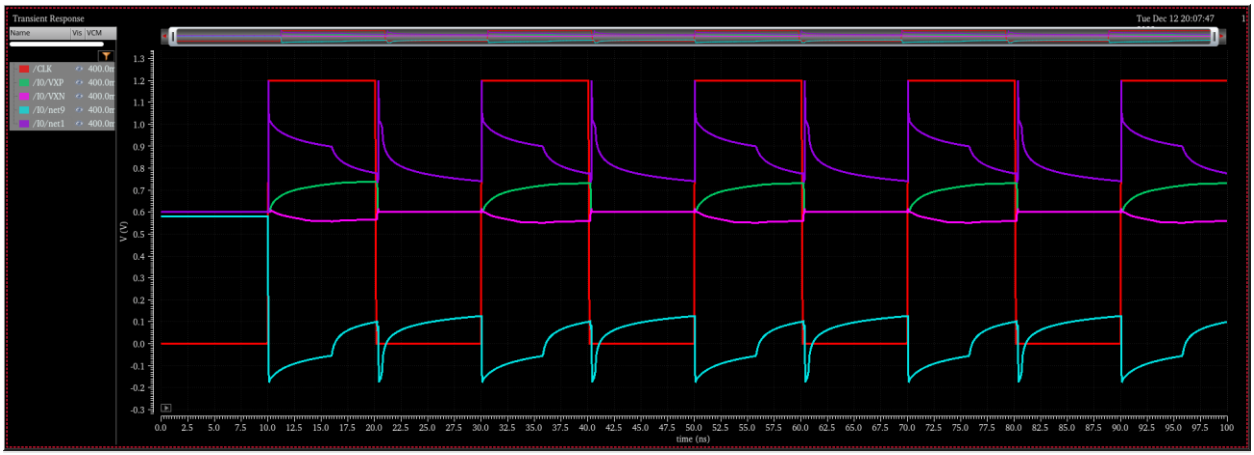
CASE 1 : VICM = 0.6V



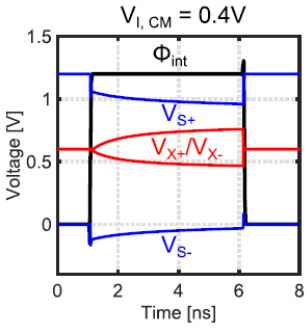


CASE 2 : VICM = 0.8v

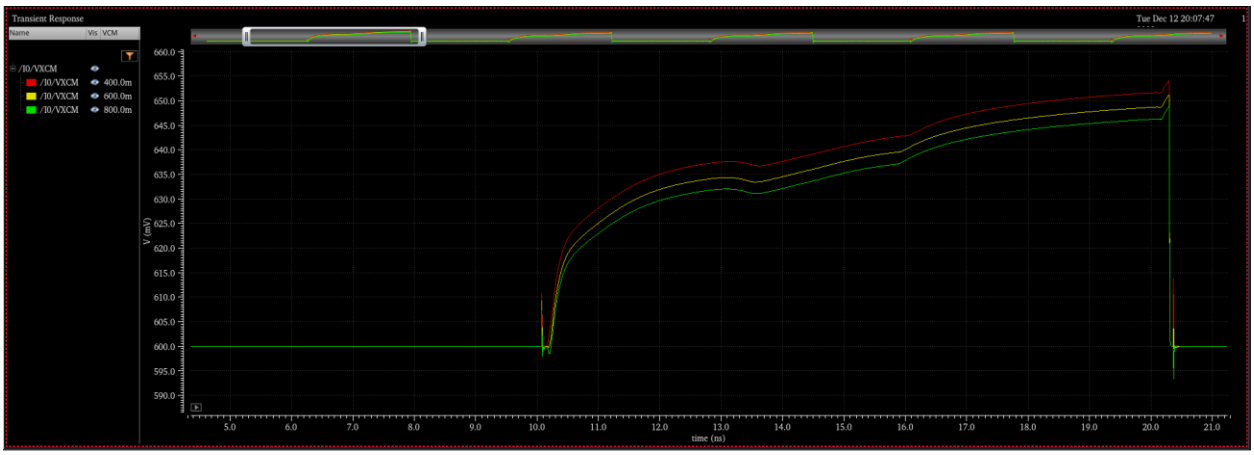




CASE 3 : VICM = 0.4v



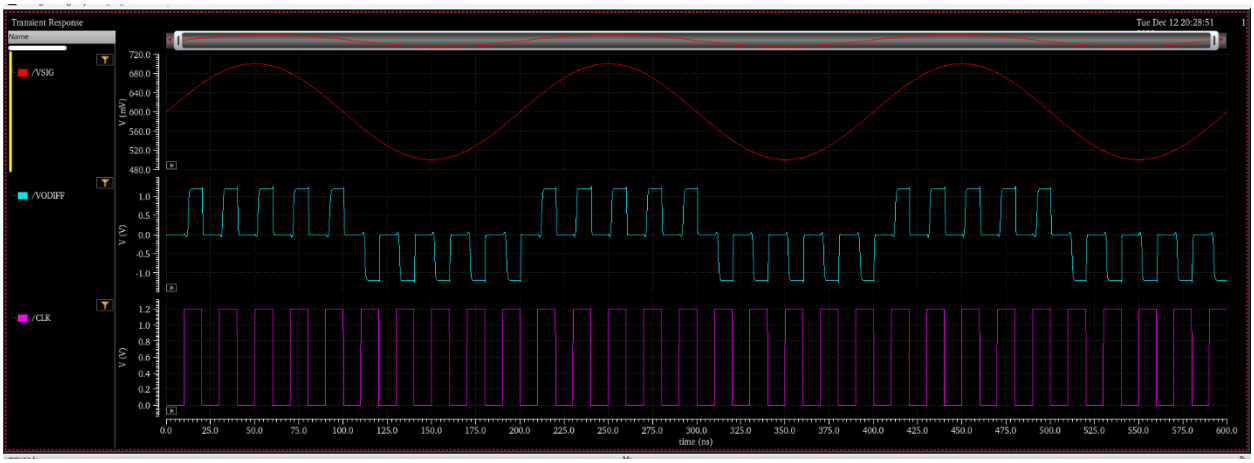
The analysis of the three cases is discussed thoroughly in the paper.



This is VXCM in the three cases the variation between them is very small since this is not the ideal case, where there are parasitics, there will be an effect on the output common mode voltage of the pre-amplifier which is given by the following equation and discussed thoroughly in the paper.

$$\Delta V_{X,\mathrm{CM}} \approx -2 \cdot \Delta V_{I,\mathrm{CM}} \cdot \frac{C_P}{C_X}.$$

Sine Wave TEST



Considering the 0.6v as the threshold of comparison, the comparator seems to work well for a sine wave test giving a polar non-return to zero line encoded signal which is the essence of digitization of analog signals.

thank you...