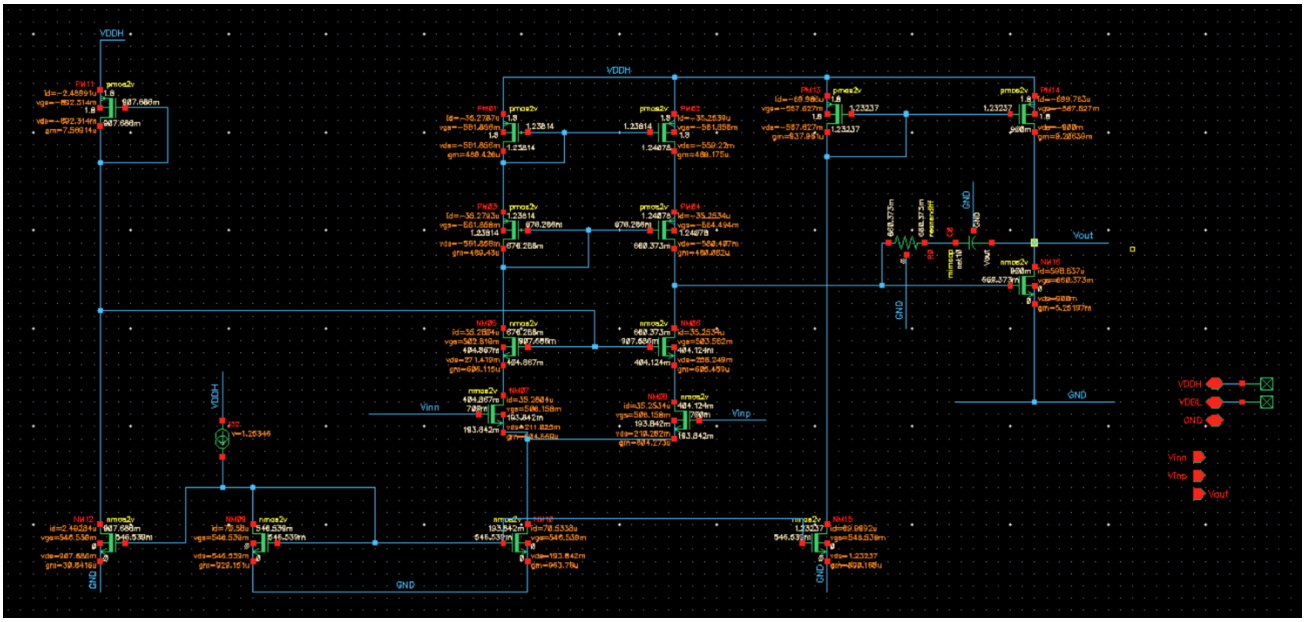
EE140 Final Project Report

Name: Dung Trong Dao

A. Overview

Schematic

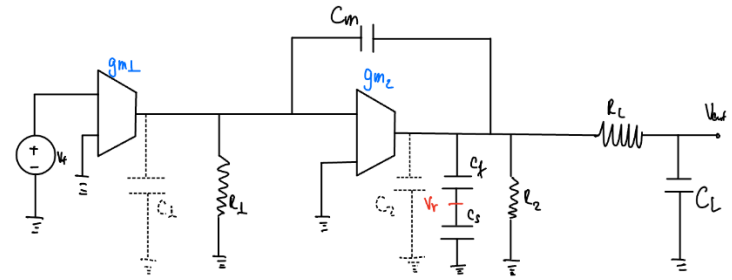


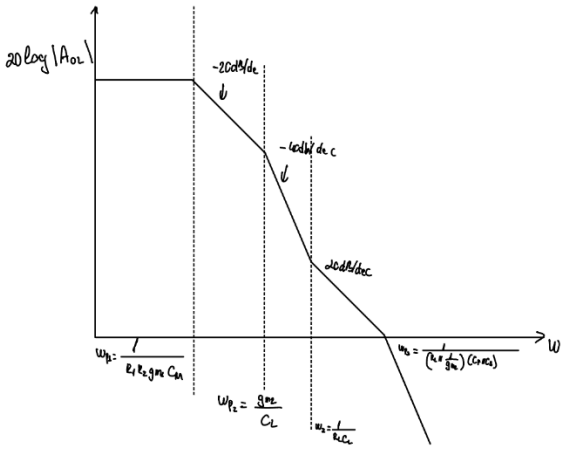
The first stage is a differential amplifier with a current mirror on top to enhance the gain of the first stage.
The bias voltage is approximately 900 mV, calculated as 2ΔV+Vth .
The first stage is designed to achieve a specific gain. In the second stage, I use a current mirror to generate the required current and a common-source amplifier to achieve the desired gain and meet the required slew rate.

Comparison table between the design specifications and my design

Parameter	Design Specifications	My Design
Settling Time (Tsettle) = max{Tsettle,rise ,Tsettle,fall }1	180ns	180ns
Total Error	≤ 0.2%	0.1%
Power Consumption	≤ **2. 5** mW	2mW
Output Voltage Swing	≥ 1.4V	1.4V
Maximum Current Mirror Ratio	10	10
Maximum Total Capacitance	10 pF	1 0pF
Maximum Total Resistance	200k Ohm	10K Ohm
CMRR at DC	≥ 70dB	70dB
PSRR at DC	≥ 50dB	50 dB
Phase Margin	≥ 45°	60°

B. Design





To design the circuit to meet the specified requirements, I focus on the maximum allowable error in estimating the gain. Additionally, since the phase margin is greater than or equal to 45°, I use this information to design gm_2. Once I determine the values of gm_1 (1st stage) and gm_2 (2nd stage), I use MATLAB to sweep and optimize the sizes of all the transistors in the circuit.

Designing 1st stage:

Let $C_c = 4.7\ pF$

First, estimate the static error, which is 0.05%. The equation is as follows:

$$\varepsilon_s = \frac{1}{A_{OL} * \beta} = 0.0005$$

$$A_{OL} = \frac{1}{0.0005 * \frac{1}{3}} = 6000$$

Secondly, estimate the dynamic error, which is 0.05%. The equation is as follows:

$$T_{setttling} = \frac{1}{60\ Hz * 272 * 340\ pixels} = 180\ ns$$

$$\varepsilon_d = \frac{Vout_{(\infty)} - Vout_{(t)}}{Vout_{(\infty)}} = e^{-w_u t} = 0.05\%$$

$w_u = \frac{\ln(0.0005)}{180ns} = 42.22 * 10^6\ rad/s \Rightarrow f_u = \frac{w_u}{2*\pi} = 6.72\ MHz$

We also have:

$w_{p1} = \frac{1}{R_1 * R_2 * gm_2 * C_c}$; $A_{OL} = R_1 * R_2 * gm_1 * gm_2$; $\beta * A_{OL} * w_{p1} = w_u * 1$

⇨ $gm_1 = \frac{w_u * C_c}{\beta} = \frac{42.22 * 10^6 \frac{rad}{s} * 4.7\ pF}{1/3} = 600\ \mu S$

When I have gm_1 and fu, I use MATLAB to determine all the transistor sizes and the V_{cm} required for the first stage. In the first loop, I sweep over the gm/Id range, and in the second loop, I sweep over the transistor lengths to find the maximum gm/Id with fu greater than 10 MHz Then, I determine the sizes of the PMOS and NMOS transistors.

Designing 2nd stage:

We have the second pole is: $w_{p2} = \frac{gm_2}{C_L}$

To determine the value of gm_{16} for the second stage and achieve a phase margin greater than 45 degrees, we want w_{p2} to be close to or equal to w_u, set :

$$w_{p2} = 2.5\ w_u$$

⇨ $gm_2 = \frac{C_L * gm_1 * \beta * 2.5}{C_c} = \frac{50pF * 600\ \mu S * 2.5 * 1/3}{4.7\ pF} = 5.32\ mS$

We also have:

$w_z = -\left(\frac{gm_2}{C_L}\right) * \frac{1}{1 - gm_{2*Rc}}$.

⇨ To avoid a pole to the RHS we set $R_c =$ 4k 0hm.

Then I use MATLAB to determine all the transistor sizes of the second stage.

C. Transistor and Bias Summary

The table below summarizes key parameters for each transistor, including dimensions, drain bias current (Id), gate-to-source voltage (|Vgs||V), transconductance (gm), and output conductance (gds).

CMOS #	Id(μA)	|Vgs| (mV)	Gm(μS)	Gds(μS)
PM01	35.27	561.85	489.42	9.40
PM02	35.27	561.85	489.17	9.41
PM03	35.27	561.85	489.43	9.40
PM04	35.27	564.49	480	9.18
NM05	35.27	502.81	606.115	30.84
NM06	35.27	503.86	605.46	32.06
NM07	35.27	506.15	604.67	36.41
NM08	35.27	506.15	604.27	36.47
NM09	70.54	546.54	929.15	21.64
NM10	70.54	546.54	963.76	59.21
PM11	2.49	892.31	7.56	156.58
NM12	2.49	546.54	30.84	450.44
PM13	69.98	576.62	937.96	18.01
PM14	699.76	576.62	9.20	149.08
NM15	69.99	546.54	890.16	10.77
NM16	598.83	660.37	5.25	109.79

D. Discussion:

Designing and implementing the first stage:

For the first stage, the primary focus was achieving sufficient gain and a high frequency fu. Based on lecture insights, a telescopic differential amplifier was chosen to meet the required gain of approximately 300. This topology provides a high intrinsic gain due to its structure while maintaining a relatively low power consumption.

To optimize the design, I utilized MATLAB, adapting the provided script from Exercise 6 to calculate the transistor sizes with a high gm/ID ratio. Initially, the design encountered significant challenges with mismatch at the current mirror located at the bottom of the circuit. This mismatch, primarily due to VDSV variations, significantly affected performance.

To address this issue, I fixed the VGSV of the PMOS transistors at 550 mV and evenly divided VDSV across each device, assigning approximately 230 mV. These adjustments helped reduce the VDSV mismatch and improved overall performance.

After determining gm_1=600 μS, I used MATLAB to sweep and find the optimal lengths and widths of all transistors in the first stage. Additionally, the MATLAB analysis provided the

current flow into the first stage, ensuring that the design met the required specifications while maintaining stability and efficiency.

$$A_{DC\_gain\_1st} = gm_{7,8}(gm_6 * ro_6 * ro_8 \,//\, gm_4 * ro_4 * ro_2)$$

After running several simulations, my design initially failed to meet the CMRR at DC (70 dB) and PSRR at DC (50 dB) requirements. To address this, I adjusted the lengths of the (PMOS) transistors and NMOS transistors at the current mirror slightly to meet the specifications, as ro is directly proportional to L. Increasing the transistor lengths resulted in higher ro, which improved the circuit's performance.

Designing and implementing the second stage:

In this stage, since gm_2 is very large, it requires a high current flow into the transistors in the second stage, which also results in the highest power consumption in the design. The high current, however, contributes to meeting the slew rate and settling time requirements.

Given the swing requirement of ≥1.4V, I fixed the VGS of the PMOS transistor at the top to 600 mV. Using the MATLAB script from Exercise 5a, I calculated the required current for the second stage. Based on this value, I implemented a current mirror with a ratio of 5 to supply the necessary current for the second stage.

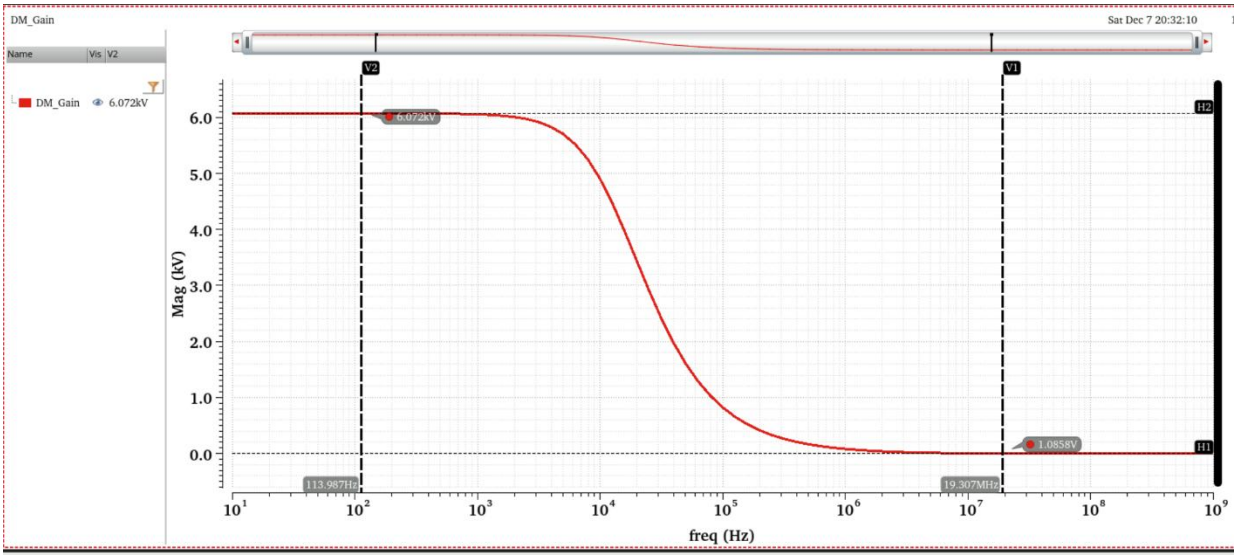
Initially, I designed the circuit with Cc = 4.7pF. However, due to unknown factors, I adjusted Cc to 1 pF, and increased the current in this stage to meet the settling time requirement of 180 ns.

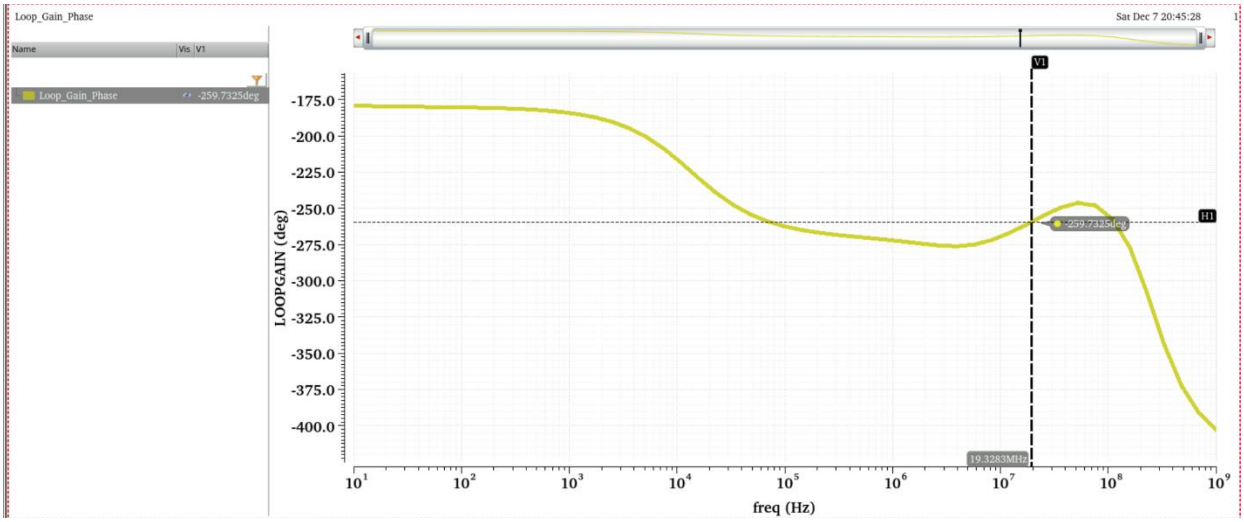
Difficulty in design:

1. Due to the limitation of the current mirror ratio being a maximum of 10, if the design does not meet the settling time requirement after using the maximum ratio, I must redesign the first stage to provide a larger current.
2. Because of power consumption constraints, I cannot set the current too high to achieve a better slew rate, as it would exceed the acceptable power budget.
3. Additionally, due to the mismatch in the current mirror, setting the bias voltage accurately becomes challenging.

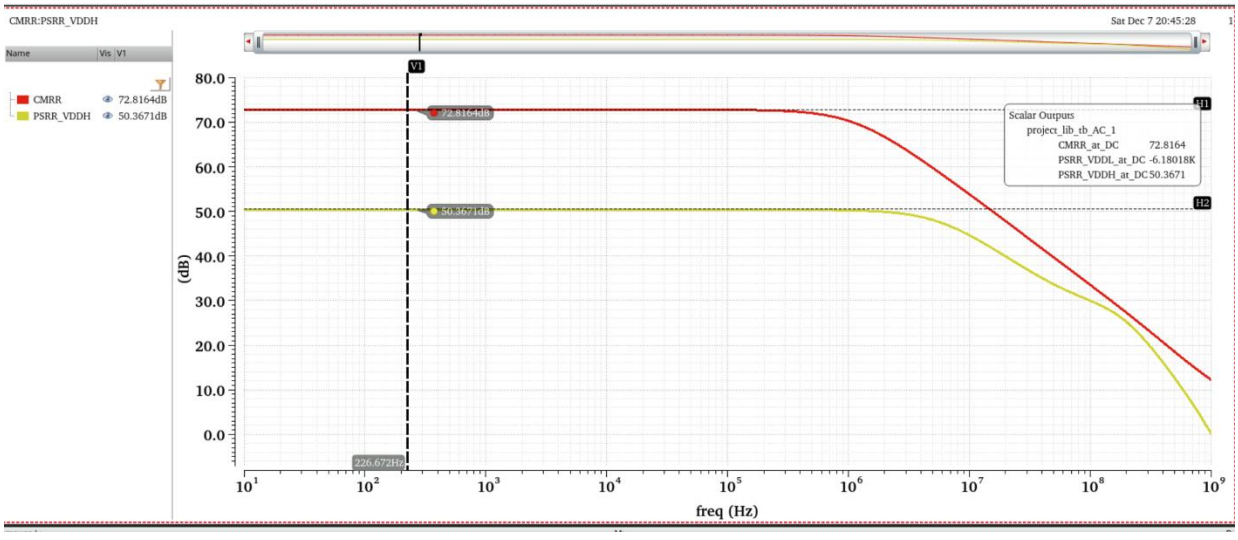
TB_AC :

1. Bode plots of the loop gain and phase margin.



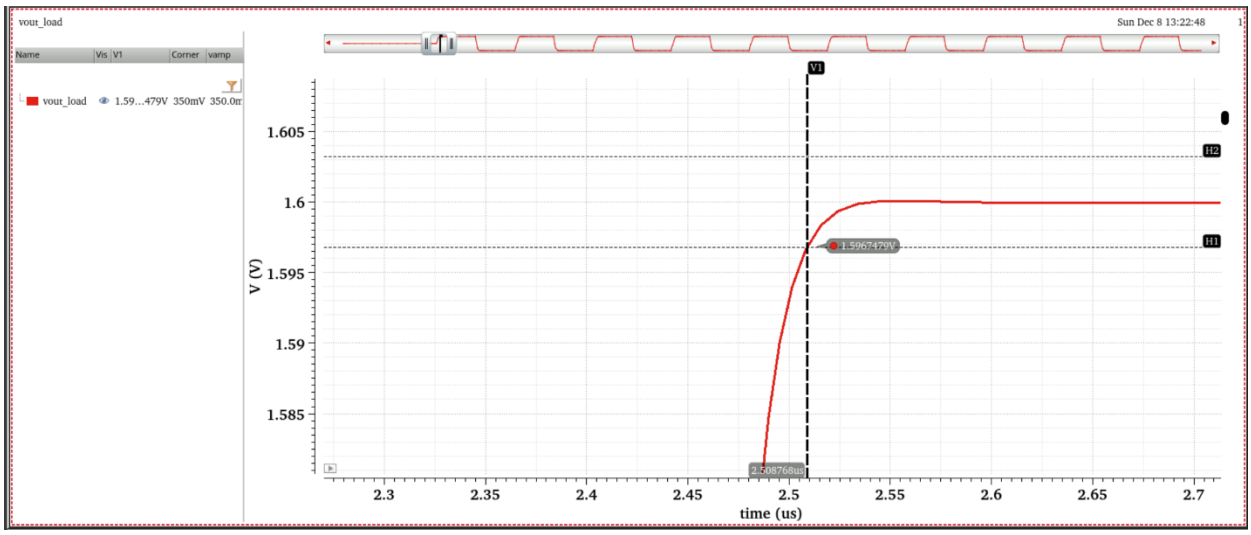


2. CMRR and PSRR vs frequency (only use VDDH in design).

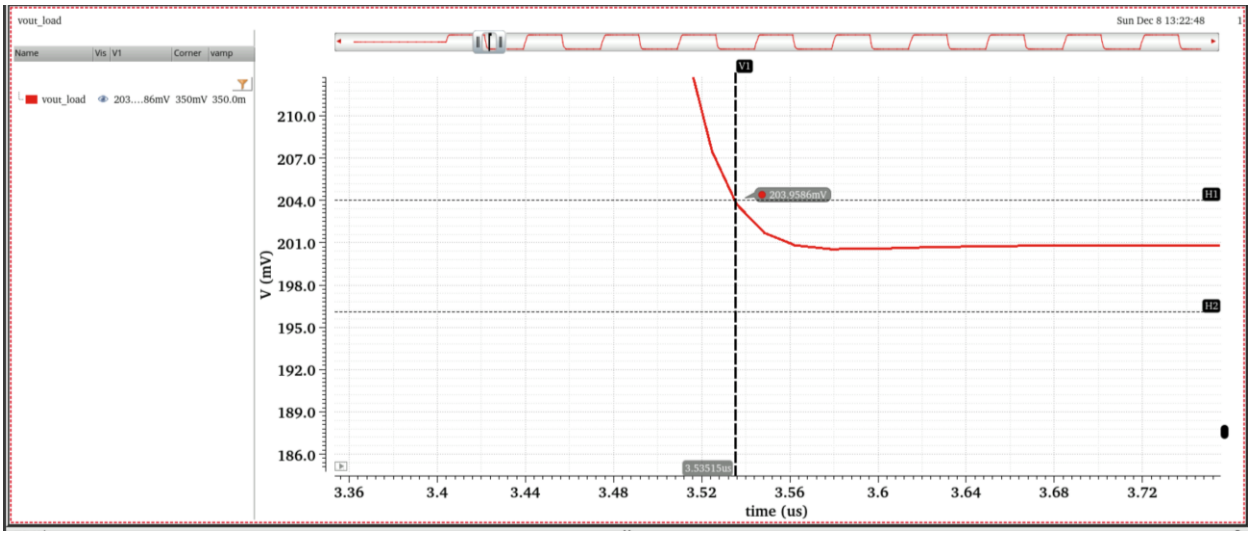


TB_Transient:

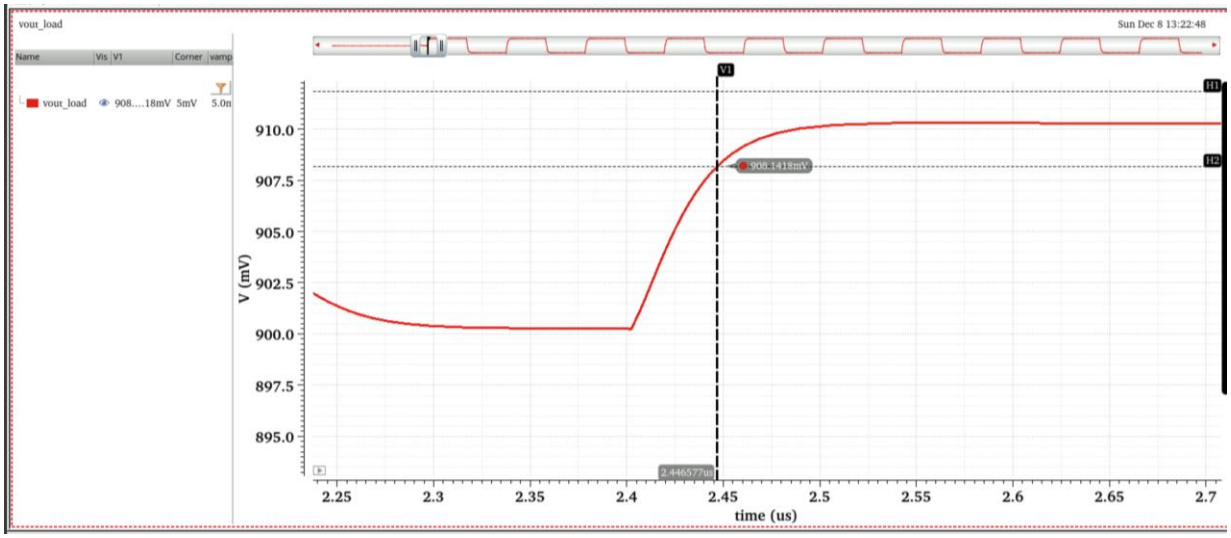
1. Waveform of the voltage at the load capacitor with legible markers for the settling time of the rising 1.4V output step.



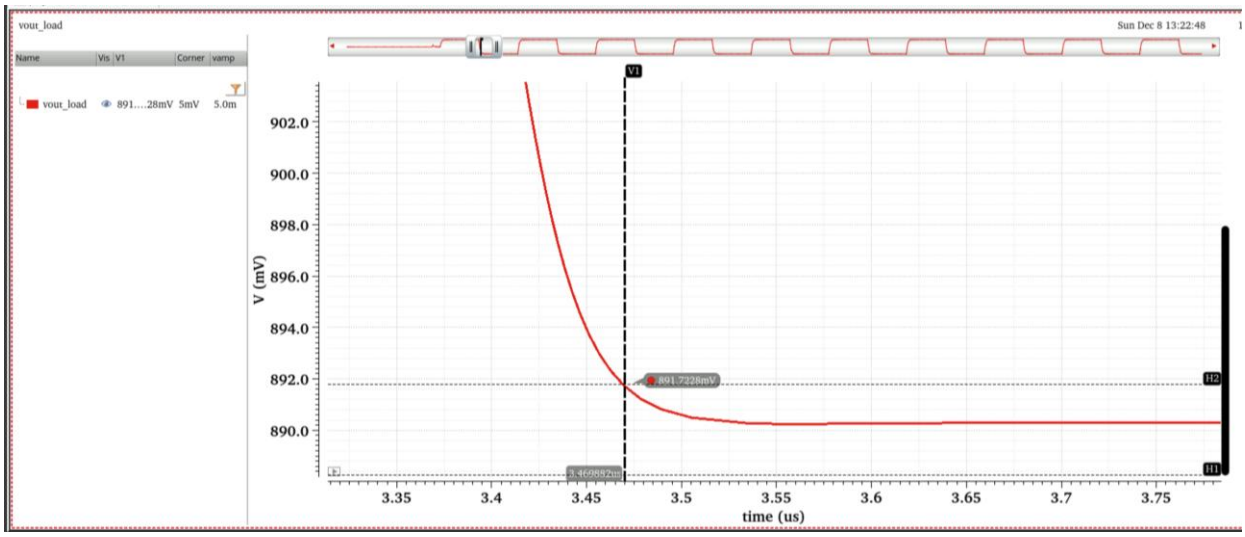
2. Waveform of the voltage at the load capacitor with legible markers for the settling time of the falling 1.4V output step.



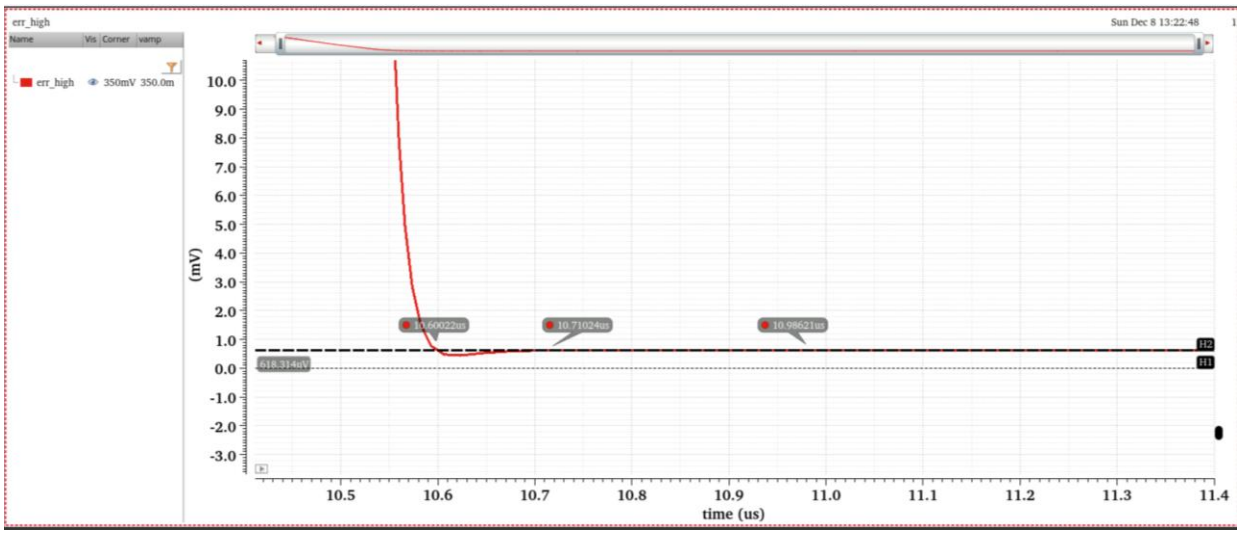
3. Waveform of the voltage at the load capacitor with legible markers for the settling time of the rising 20mV output steps.



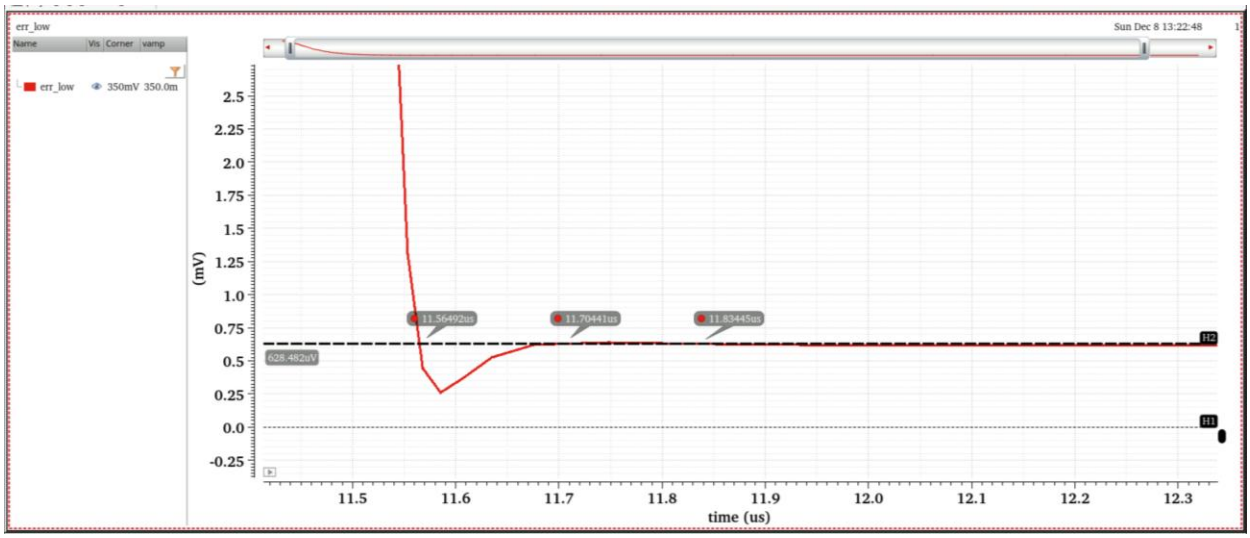
4. Waveform of the voltage at the load capacitor with legible markers for the settling time of the falling 20mV output step.



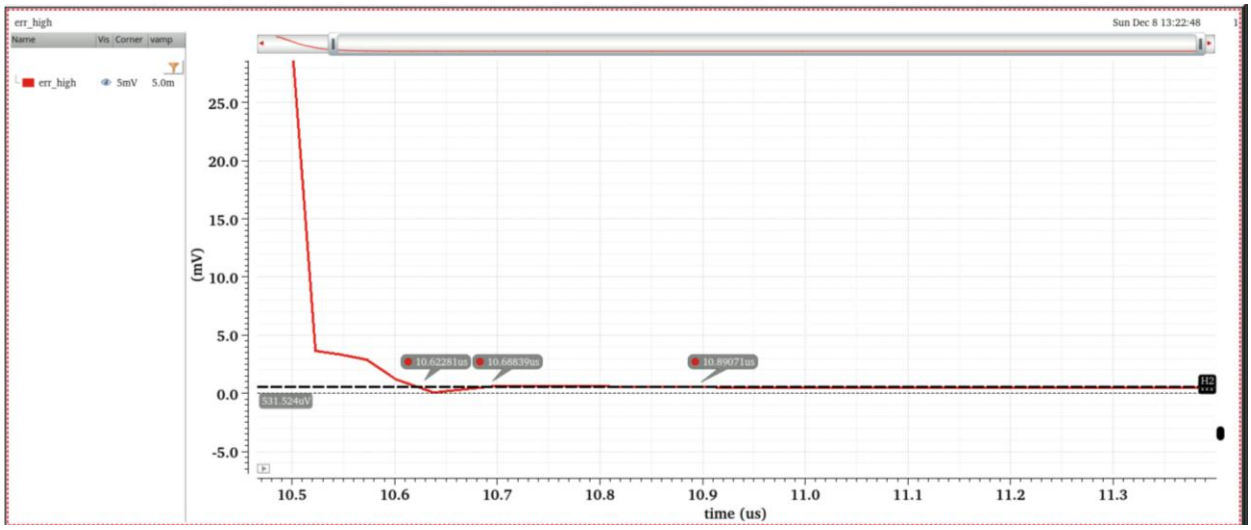
5. Settling error for rising 1.4V output steps at the load capacitor with legible markers



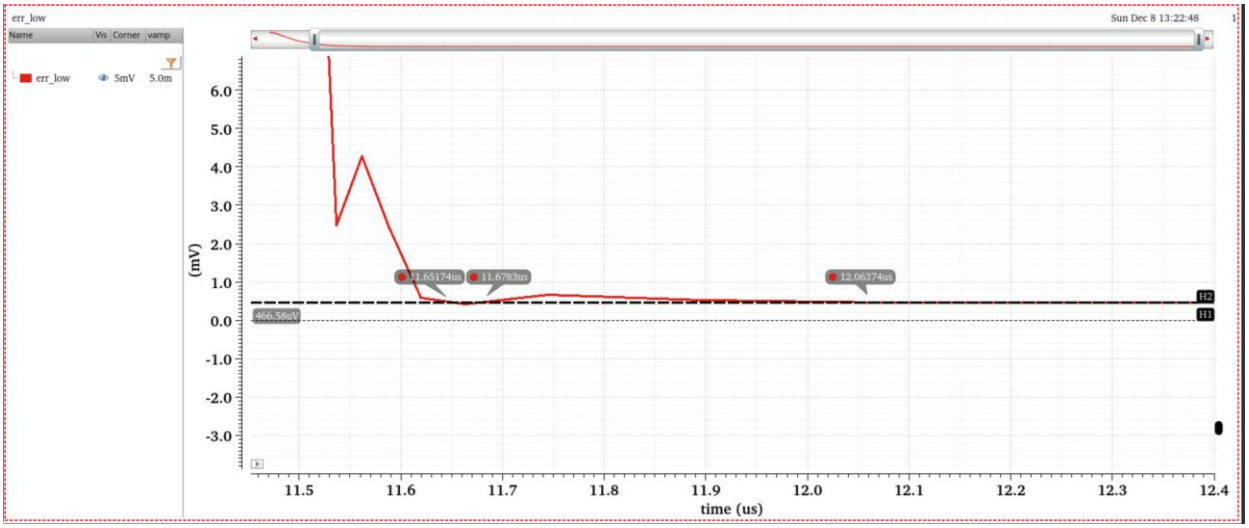
6. Settling error falling 1.4V output steps at the load capacitor with legible markers



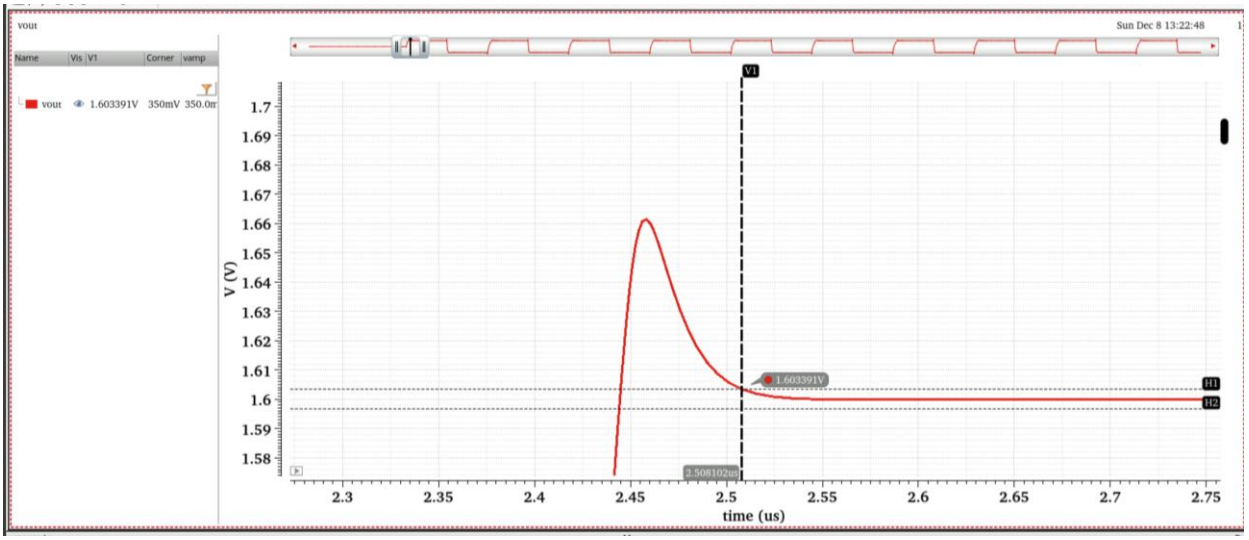
7. Settling error for rising 20mV output steps at the load capacitor with legible markers



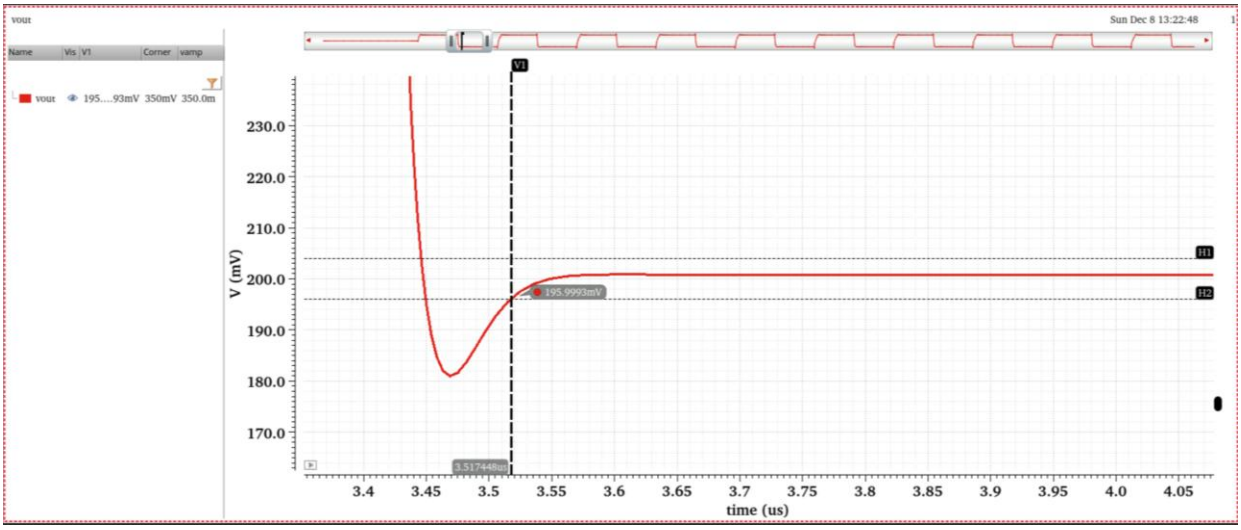
8. Settling error for falling 20mV output steps at the load capacitor with legible markers



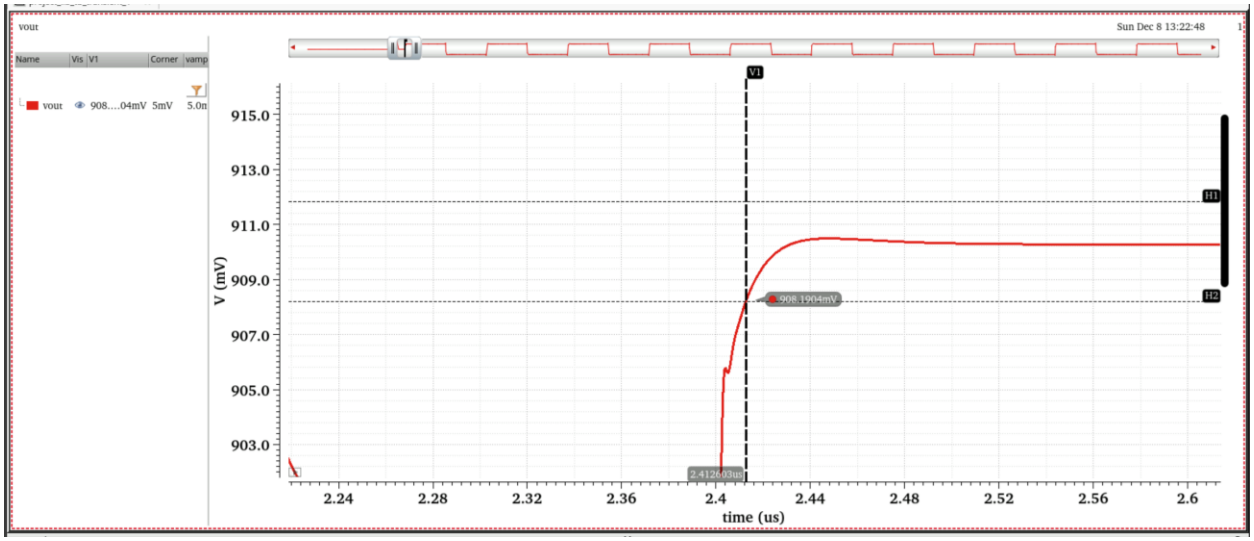
9. Waveform of the voltage at amplifier output for the rising 1.4V output steps.



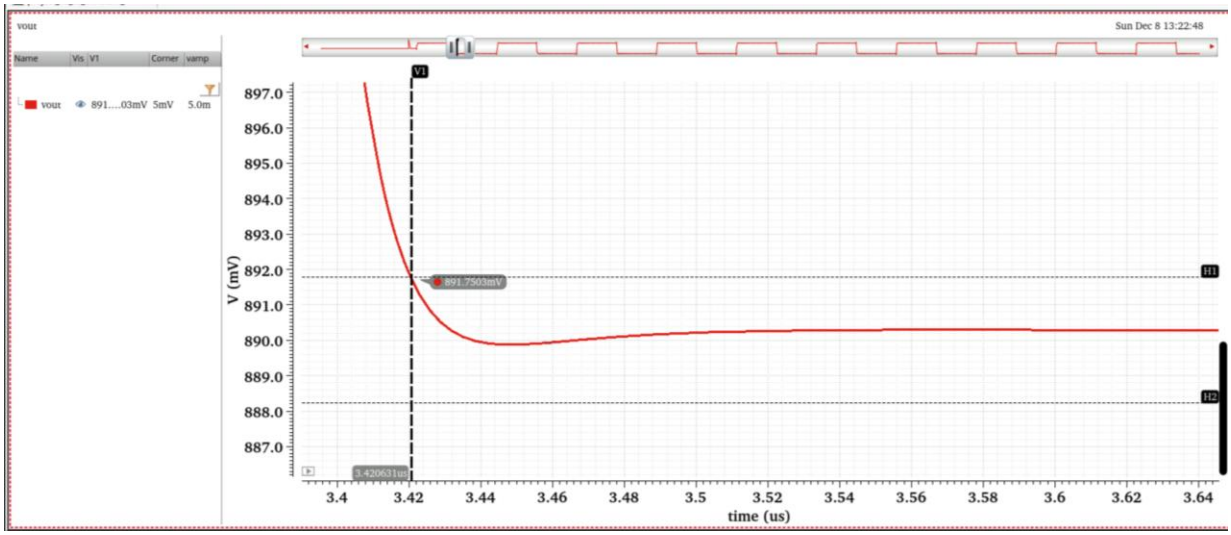
10. Waveform of the voltage at amplifier output for the falling 1.4V output steps.



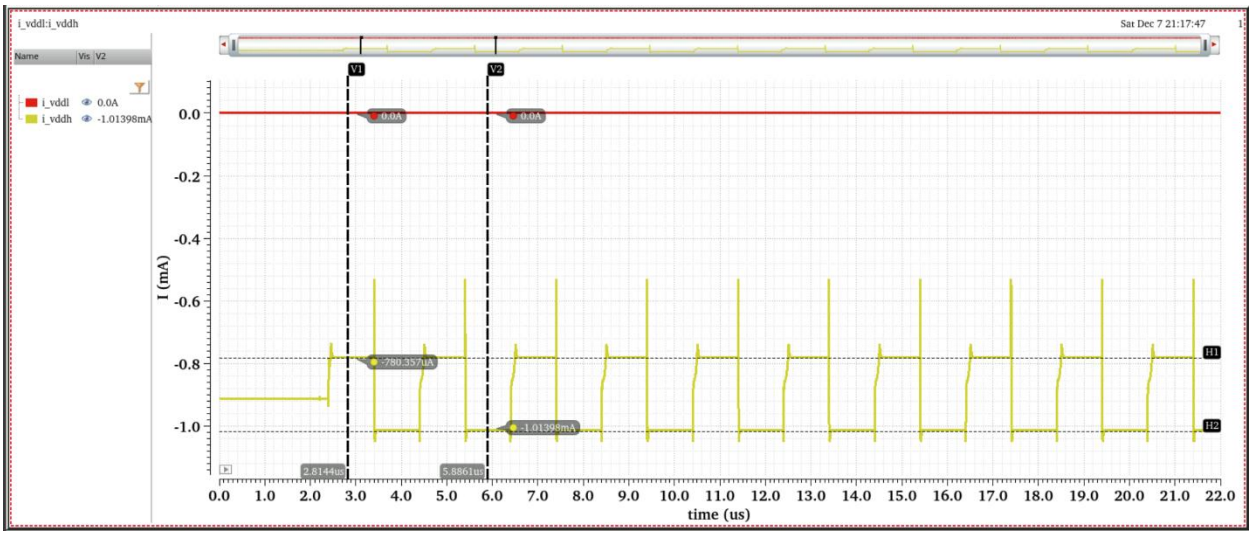
11. Waveform of the voltage at amplifier output for the rising 20mV output steps.



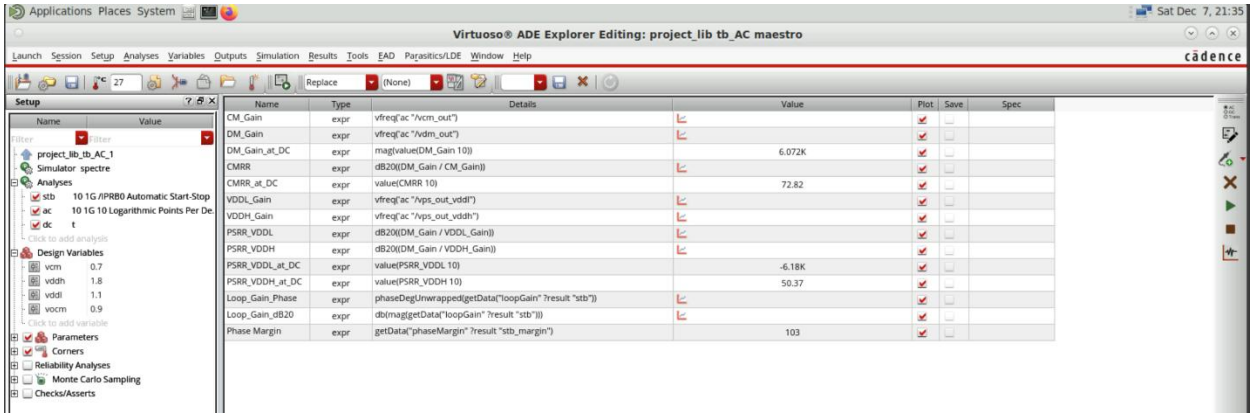
12. Waveform of the voltage at amplifier output for the falling 20mV output steps.



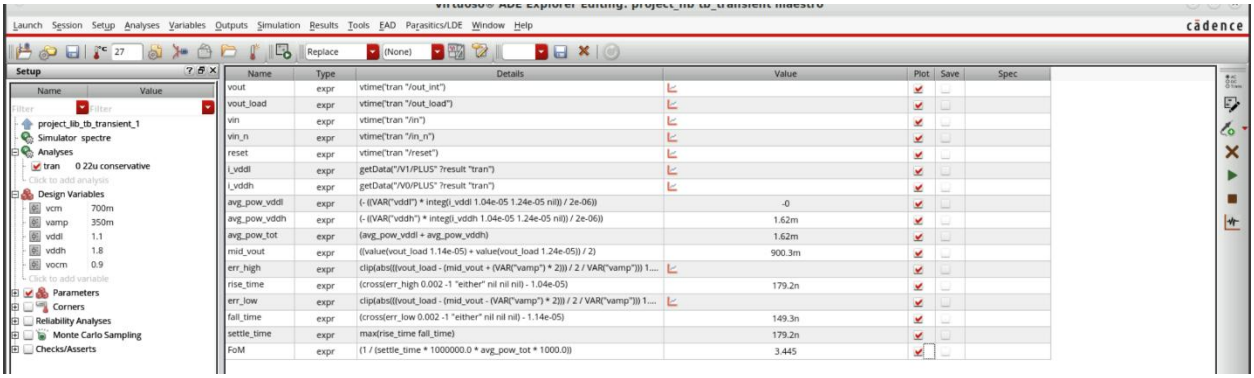
13. Waveform of the currents drawn from VDDH and VDDL (only use VDDH).



14. AC_Testbench



15. Transient_Testbench



16. Final result table

Parameter	Design Specifications	My Design
Settling Time (Tsettle) = max{Tsettle,rise ,Tsettle,fall }1	180ns	167ns
Total Error	≤ 0.2%	0.065%
Power Consumption	≤ **2. 5** mW	1.62mW
Output Voltage Swing	≥ 1.4V	1.4V
Maximum Current Mirror Ratio	10	10
Maximum Total Capacitance	10 pF	1 pF
Maximum Total Resistance	200k Ohm	4K Ohm
CMRR at DC	≥ 70dB	72.82dB
PSRR at DC	≥ 50dB	50.37dB
Phase Margin	≥ 45°	103°
Figure of Merit (FoM) to Maximize	$\frac{1}{T_{settle}(Us) * P_{total}(mW)}$	3.445

E. Conclusion

Throughout this project, I applied a telescopic differential amplifier for the first stage to achieve the required gain and utilized a current mirror with an appropriate ratio for the second stage to supply the large current required. The second stage's high current demand was critical to meeting performance specifications.

In the first stage, I encountered difficulties with the VDS mismatch when stacking the current mirror on top of the differential amplifier. This mismatch prevented me from properly generating the bias voltage for the first stage. To resolve this issue, I determined that the bias voltage needed to be approximately 900 mV. I designed a current mirror and optimized the transistor sizes to produce an output voltage around 900 mV, which I then used as the bias voltage. I also faced challenges in meeting the settling time requirements. I recognized that improving the settling time required increasing w_u, but doing so affected other factors, such as the gain, gm_1, and current in the first stage. Achieving the correct balance between these competing parameters required iterative adjustments and fine-tuning.

Ultimately, after performing hundreds of simulations and carefully adjusting the transistor sizes, I was able to meet all the requirements. Through this project, I gained valuable skills in designing IC circuits and learned the importance of connecting and analyzing all provided information. Understanding the relationships between design parameters was key to developing effective solutions for my circuit.

MATLAB SCRIPTS

WeekW3_designed_stage1.m

```
nch = importdata("nch_2v.mat");
pch = importdata("pch_2v.mat");
L3 = 1;
VDS3 = 0.24;
VGS3=0.55;


fu = 1e8;
ft = fu * 10;
CL = 1e-12
VDD = 1.8;
fixed_VGSP = 0.55;
fixed_VDSN=0.23;
L_range = nch.L;
gm_ID_range = 2:1:30;
fixed_gm1 =600e-6;


% Sweep over gm_ID_range
for i = 1: length(gm_ID_range)
    % Find the best PMOS L
    p_gds_id_vs_l = look_up(pch, 'GDS_ID', 'L', L_range , 'VGS', fixed_VGSP , 'GM_ID', gm_ID_range(i));% Your code to look up pmos gds/ID acro
    p_ft_vs_l = look_up(pch , 'GM_CGG', 'L', L_range, 'VGS', fixed_VGSP, 'GM_ID',gm_ID_range(i))/2/pi;% Your code to look up pmos ft across l
    M = p_ft_vs_l > ft ;
    if(any(M))
        % We know gds_id will decrease as L increases , so find the highest L
        best_idx = max(find(M==1));
        p_gds_id = p_gds_id_vs_l(best_idx);
        best_p_l(i) = L_range(best_idx);
        % Knowing PMOS GDS and L, we can optimize the NMOS L and gm_ID
        for j = 1:length(L_range)
            % Find the gm_ID that maximizes overall gain while meeting ft constraint
            n_gds_id = look_up(nch, 'GDS_ID', 'GM_ID',gm_ID_range ,'L', L_range(j) , 'VDS', fixed_VDSN)% Your code to find the NMOS gds_id acr
            n_ft = look_up(nch , 'GM_CGG', 'GM_ID',gm_ID_range, 'L',L_range(j), 'VDS', fixed_VDSN)/2/pi;% Your code to find the NOMS fT across
            av = gm_ID_range./((n_gds_id + p_gds_id)');% Your code to find Av from gm_ID_range , n_gds_id and p_gds_id
            K = n_ft > ft ;
            if (any(K))
              max_av = max(av(find(K ==1)));
              best_idx = find(av== max_av);
              best_n_gm_ID(i,j) = gm_ID_range(best_idx);
              best_Av(i,j) = av(best_idx);
            else
              best_n_gm_ID(i,j) = NaN;
              best_Av(i,j) = NaN;
            end
        end
    else
        best_n_gm_ID(i ,:) = NaN;
        best_Av(i, :) = NaN;
        best_p_l(i) = NaN;
    end
end
% Find the best Av
max_gain = max(max(best_Av));
[gmp_ID_idx,n_L_idx] = find(best_Av == max_gain);
opt_p_L = best_p_l(max(gmp_ID_idx));
opt_n_gm_ID = best_n_gm_ID(max(gmp_ID_idx) , max(n_L_idx));
opt_n_L = L_range(max(n_L_idx));% Your code to find the optimum nmos
n_cdd_w = look_up(nch,'CDD_W','GM_ID',opt_n_gm_ID,'VDS',fixed_VDSN,'L',opt_n_L);% Your code to find the nmos CDD /W at the optimum gm_ID , L,
n_JD = look_up(nch,'ID_W','GM_ID',opt_n_gm_ID,'L',opt_n_L,'VDS',fixed_VDSN);% Your code to find the nmos JD at the optimum gm_ID , L, VDS
p_cdd_w = look_up(pch,'CDD_W','VGS',fixed_VGSP,'VDS',fixed_VGSP,'L',opt_p_L);% Your code to find the pmos CDD /W at the optimum VGS , VDS , L
p_JD = look_up(pch,'ID_W','VDS',fixed_VGSP,'VGS',fixed_VGSP,'L',opt_p_L)% Your code to find the pmos JD at the optimum VGS , VDS , L
% Iterate to find W
I = fixed_gm1  / opt_n_gm_ID;
Wn = I / n_JD;
Wp = I / p_JD;
I_ref= 2*I;

JD3 = look_up(nch,'ID_W','VGS',VGS3,'L',L3,'VDS',VDS3);
W3=I_ref/JD3
%find VGS_M1
VGS_M1 = look_upVGS(nch , 'GM_ID', opt_n_gm_ID , 'L', opt_n_L);
VICM = VGS_M1 + VDS3;
```

Editor - /home/cc/ee140/fa24/class/ee140-aai/gpdk045_LUTs/weekw_3_project3_stage2.m

WeekW3_designed_stage1.m | weekw_3_project3_stage2.m

```
nch = importdata("nch_2v.mat");
pch = importdata("pch_2v.mat");

fu = 1e8;
ft = fu * 10;
CL = 1e-12;
VDD = 1.8;

Vout = 0.9;
fixed_gm2=6.67e-3;
VGS=0.55;

L_range = nch.L;
LP_range = pch.L;
gm_ID_range = 2:1:30;
% Sweep over Vout
    % Find the best PMOS L
    p_gds_id_vs_l = look_up(pch, 'GDS_ID', 'L', L_range , 'VGS', VGS , 'VDS', Vout);% Your code to look up pmos gds/ID across l given VGS ,VDS
    p_ft_vs_l = look_up(pch , 'GM_CGG', 'L', L_range, 'VGS', VGS, 'VDS',Vout)/2/pi;% Your code to look up pmos ft across l given VGS ,VDS
    M = p_ft_vs_l > ft ;
    if(any(M))
        % We know gds_id will decrease as L increases , so find the highest L
        best_idx = max(find(M==1));
        p_gds_id = p_gds_id_vs_l(best_idx);
        opt_p_L= L_range(best_idx);
        % Knowing PMOS GDS and L, we can optimize the NMOS L and gm_ID
        for j = 1:length(L_range)
            % Find the gm_ID that maximizes overall gain while meeting ft constraint
            n_gds_id = look_up(nch, 'GDS_ID', 'GM_ID',gm_ID_range ,'L', L_range(j) , 'VDS', Vout)% Your code to find the NMOS gds_id across th
            n_ft = look_up(nch , 'GM_CGG', 'GM_ID',gm_ID_range, 'L',L_range(j), 'VDS', Vout)/2/pi;% Your code to find the NOMS fT across the g
            av = gm_ID_range./((n_gds_id + p_gds_id)');% Your code to find Av from gm_ID_range , n_gds_id and p_gds_id
            K = n_ft > ft ;
            if (any(K))
                max_av = max(av(find(K ==1)));
                best_idx = find(av== max_av);
                best_n_gm_ID(j) = gm_ID_range(best_idx);
                best_Av(j) = av(best_idx);
            else
                best_n_gm_ID(j) = NaN;
                best_Av(j) = NaN;
            end
        end
    else
        best_n_gm_ID(j) = NaN;
        best_Av(j) = NaN;
    end
% Find the best Av
max_gain = max(max(best_Av));
n_L_idx = find(best_Av == max_gain);
opt_n_gm_ID = best_n_gm_ID(n_L_idx);
opt_n_L = L_range(n_L_idx);% Your code to find the optimum nmos L
% Find the real widths of the devices
n_cdd_w = look_up(nch,'CDD_W','GM_ID',opt_n_gm_ID,'VDS',Vout,'L',opt_n_L);% Your code to find the nmos CDD /W at the optimum gm_ID , L, VDS
n_JD = look_up(nch,'ID_W','GM_ID',opt_n_gm_ID,'L',opt_n_L,'VDS',Vout);% Your code to find the nmos JD at the optimum gm_ID , L, VDS
p_cdd_w = look_up(pch,'CDD_W','VGS',VGS,'VDS',Vout,'L',opt_p_L);% Your code to find the pmos CDD /W at the optimum VGS , VDS , L
p_JD = look_up(pch,'ID_W','VDS',Vout,'VGS',VGS,'L',opt_p_L)% Your code to find the pmos JD at the optimum VGS , VDS , L
% Iterate to find W
I_ref = fixed_gm2 / opt_n_gm_ID;
Wn = I_ref / n_JD;
Wp = I_ref / p_JD;
VGS_M6 = look_upVGS(nch , 'GM_ID', opt_n_gm_ID , 'L', opt_n_L);
```