MIXED-SIGNAL EDGE ACCELERATOR FOR REAL-TIME MMWAVE PLATFORM VIBRATION COMPENSATION

DESIGNS AND TESTS DOCUMENTATION

To simulate the SPICE files, first source the *.bashrc* file in the top-level directory.

- *source caddy.bashrc*

This will use the ngspice version (33) already installed on the caddy. To run any of the spice files, simply call:

- *ngspice netlist_name.spice*

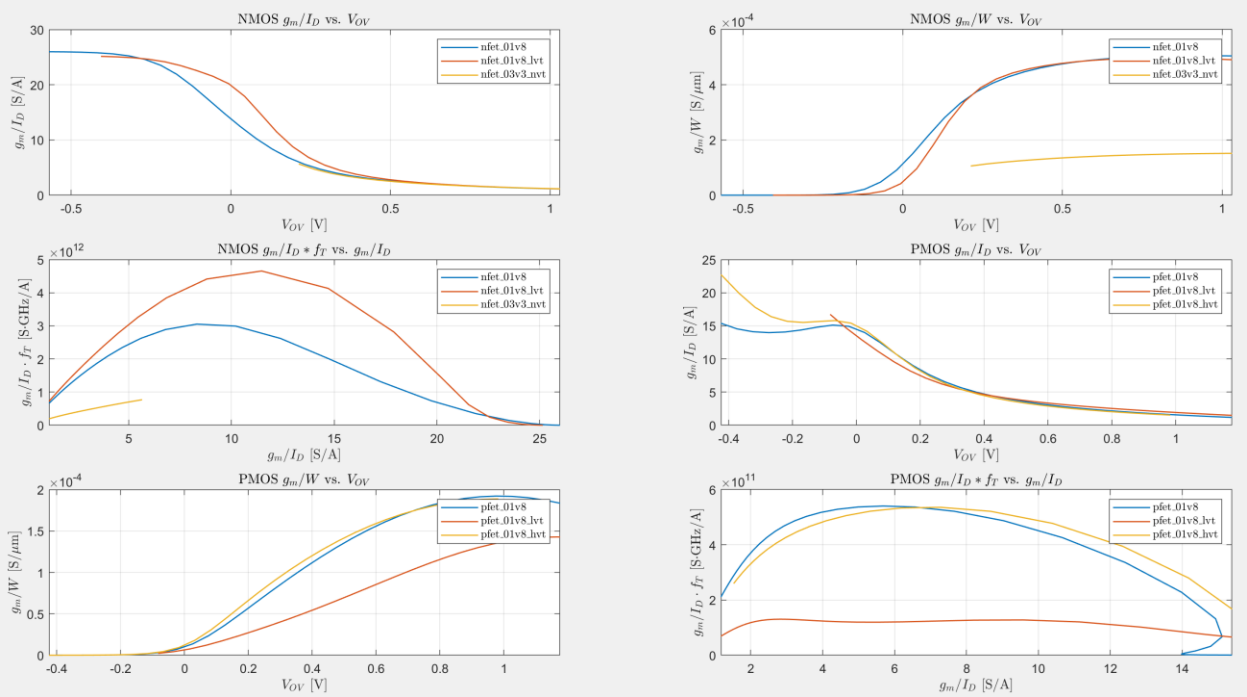
Note that the netlist file will include the *…commands.spice* file, so ngspice should be called on the netlist file itself. For instance, for the fully differential OTA, one would invoke:

- *ngspice diff_fold_casc_ota.spice*

Design space exploration and device characterization utilizes the following MATLAB scripts:

- *matlab_scripts/mosSKY130_characterization.m*
- *matlab_scripts/mosSKY130_gm_id_analysis.m*

These simply read in the generated characterization data files for the LVT/HVT NMOS and PMOS transistors in the Skywater 130 nm PDK (included in the MATLAB scripts repository directory) and generate plots illustrating the devices' key design characteristics (e.g. g_m/I_D, f_T, etc.). They also form the basis for the table look-up design process implemented in *look_up.m*. Below are example characterization plots generated by *mosSKY130_gm_id_analysis.m*.



For the actual chip design, results can be visualized using the MATLAB script, *spice_post_process.m*. This script is divided into sections corresponding to the various design blocks, such that the user can choose which block to analyze on its own. Simply run the corresponding standalone section of the script. Note that the script references the file *load_ngspice.m* and thus must be run in the same directory as this auxiliary helper script. For all analyses, path names may have to be updated to reflect the repository structure, since the scripts were simply run locally (not in the repository) during the design and debugging process.

For SPICE simulations, many of the *..commands* files include multiple analyses. Simply uncomment the selected analysis to be investigated. In many cases, different analysis require different sources (e.g. transient vs. AC analyses), and so the corrected input stimulus sources must also be commented/uncommented in the commands SPICE files. Likewise, the correct *.raw* output file name must also be commented/uncommented.

I. Fully Differential Operational Transconductance Amplifier (OTA)

Schematic: *designs/diff_fold_casc_ota.sch*
Symbol: *designs/diff_fold_casc_ota.sym*
SPICE netlist: *simulations/diff_fold_casc_ota.spice*
Simulations/tests file: *simulations/diff_fold_casc_ota_commands.spice*

- This file includes the parameter values for the final design.
- Used for all simulations except the transient step response.

Transient simulation schematic: *designs/diff_ota_tb.sch*
Transient simulation SPICE netlist: *simulations/diff_ota_tb.spice*
Transient simulation test file: *simulations/diff_ota_tb_commands.spice*

- This file includes the appropriate parameter values.

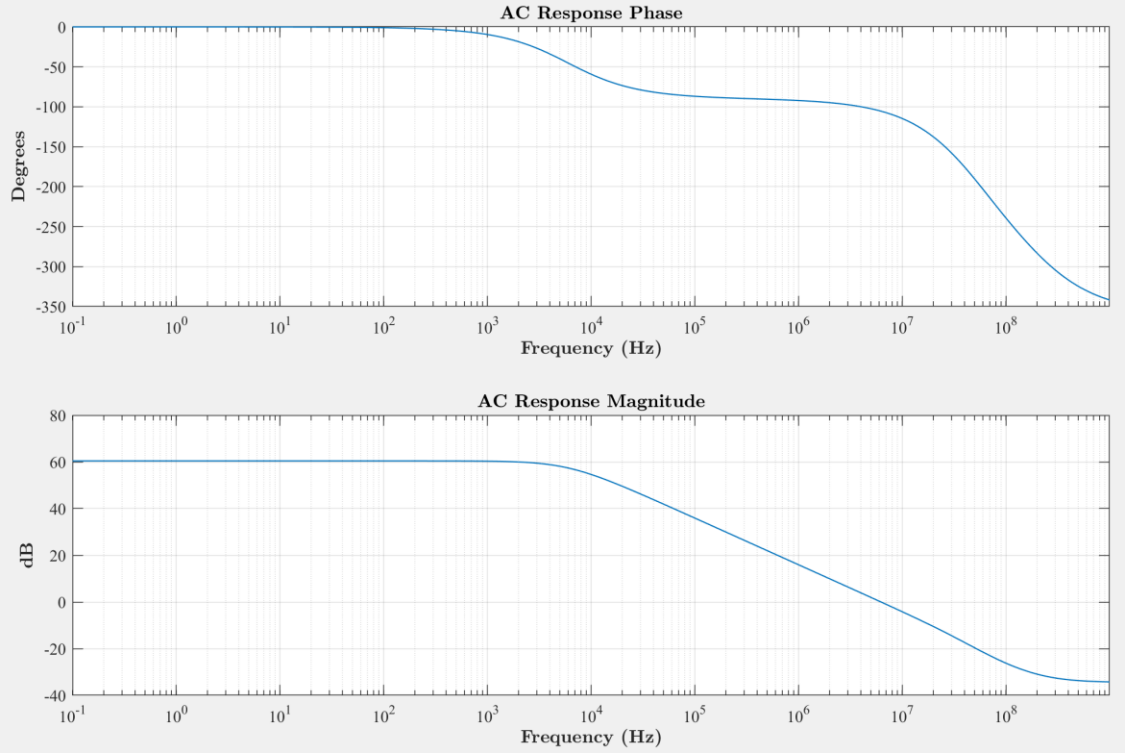
Analysis: “Differential OTA/Op-Amp Analysis” section of *spice_post_process.m*.

Tests:

a. Operating point simulation: performed in all of the control blocks.

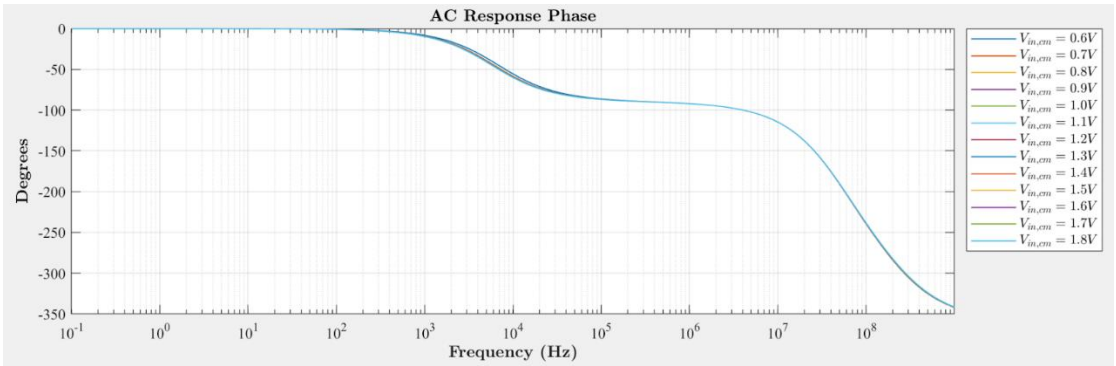
b. Single-point AC analysis: implemented in Control Block 1. Simply uncomment the AC analysis line. To analyze, ensure that the correct *.raw* file name is uncommented in the MATLAB script and that the AC analysis sweep Boolean is set to false.

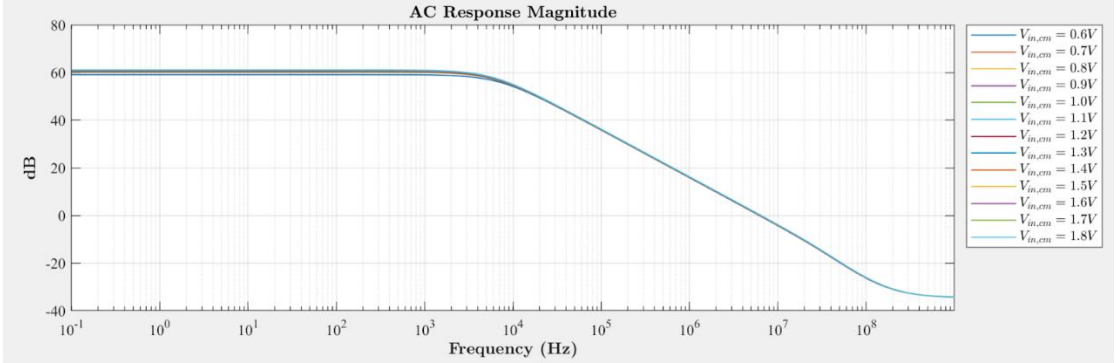
   Expected plot for *vincm* = *vcmdes* = 1.0 V:

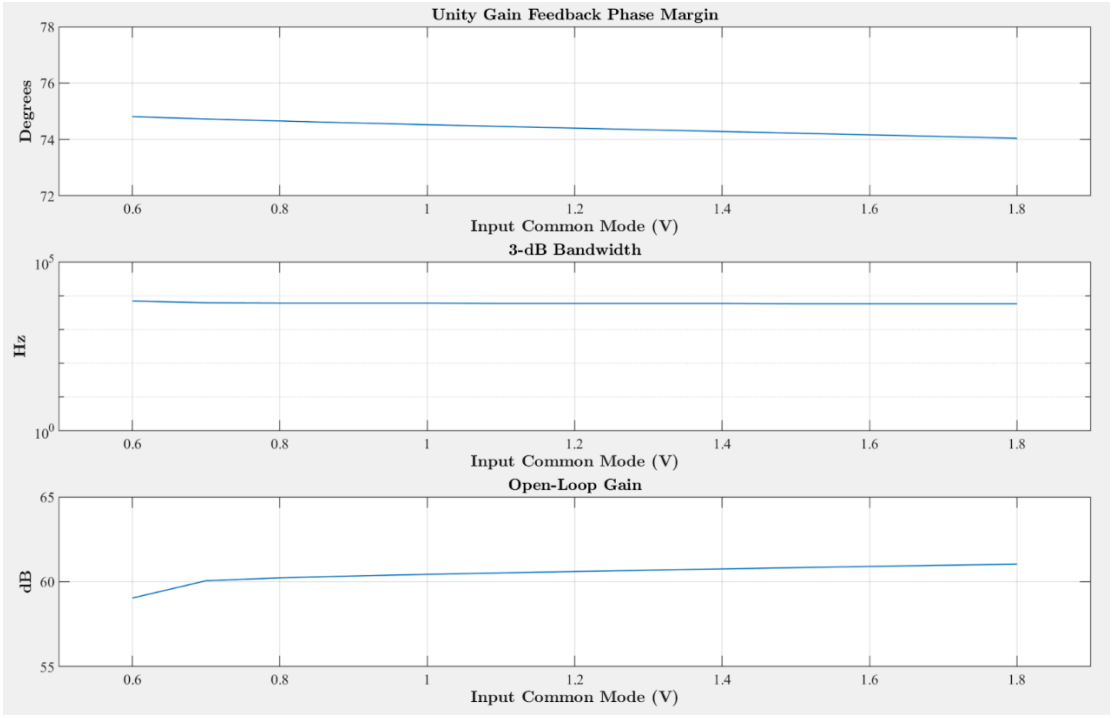


c. Input common-mode sweep AC analysis: implemented in Control Block 2. To analyze, ensure that the AC analysis sweep Boolean is set to true in the MATLAB script and that the file names are correct.

   Expected plots:

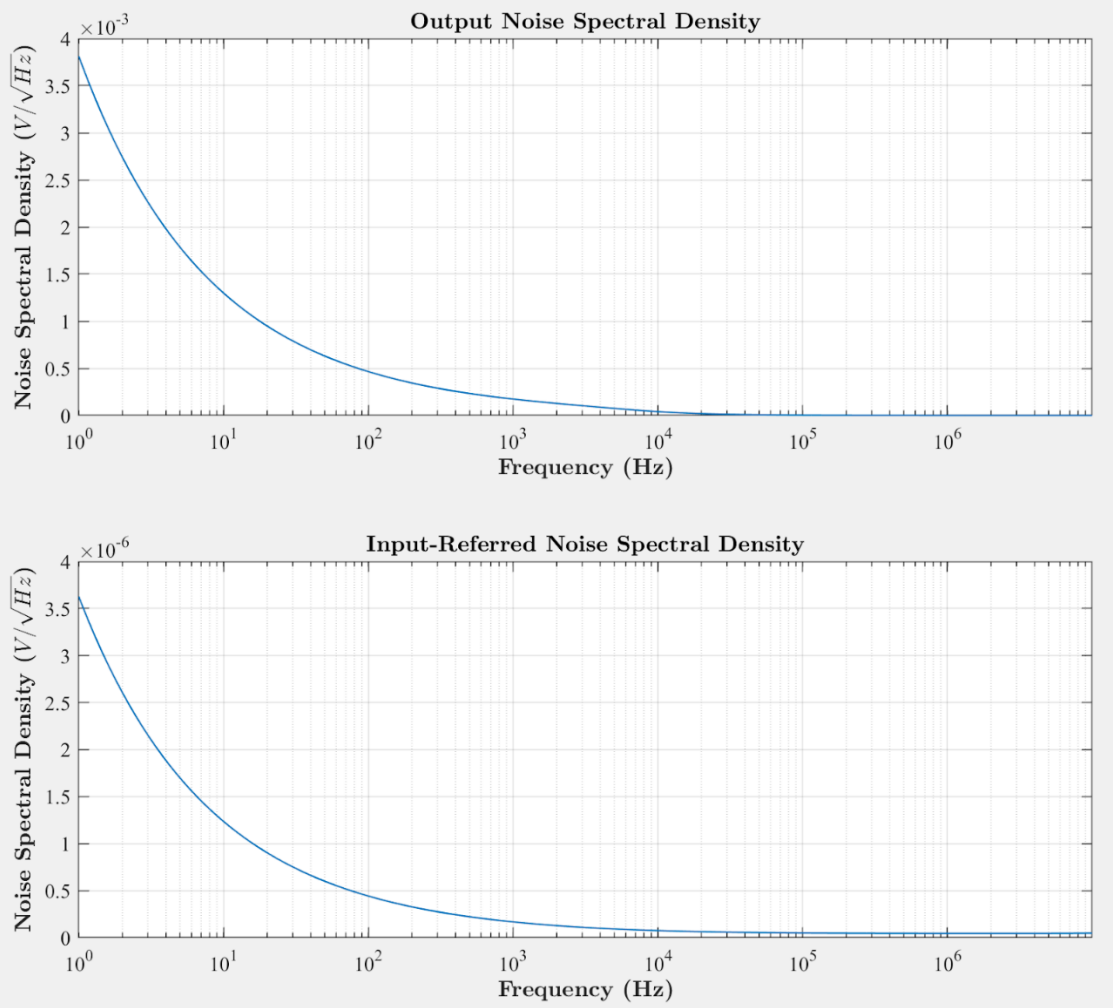






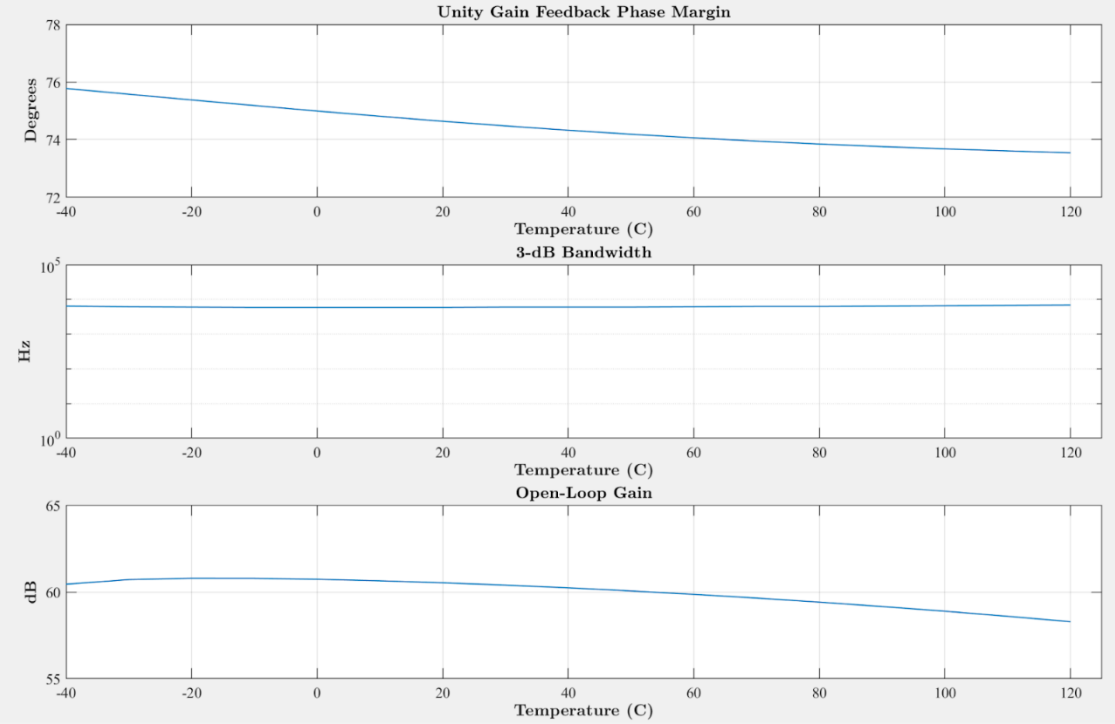
d. Noise analysis: implemented in Control Block 1. Simply comment out the noise analysis lines (mode 1 or mode 2). To analyze, ensure that the noise analysis Boolean is set to true (selects between the two noise analysis modes).

Expected noise spectrums:



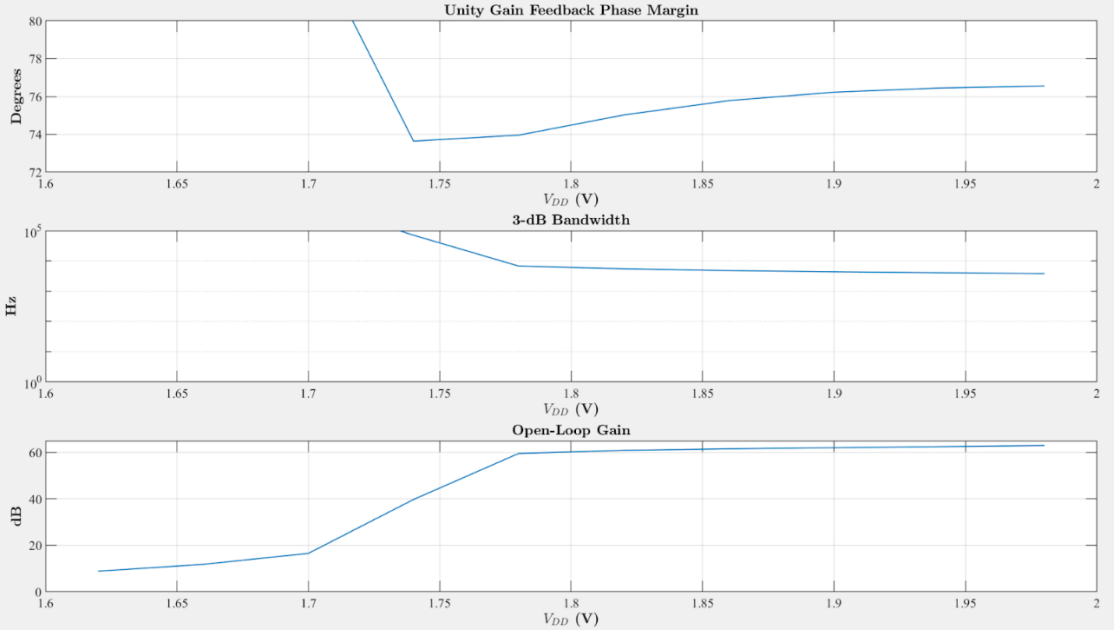
e. Temperature sweep AC analysis: implemented in Control Block 3. To analyze, ensure that the temperature sweep Boolean is set to true in the MATLAB script.

Expected plot:



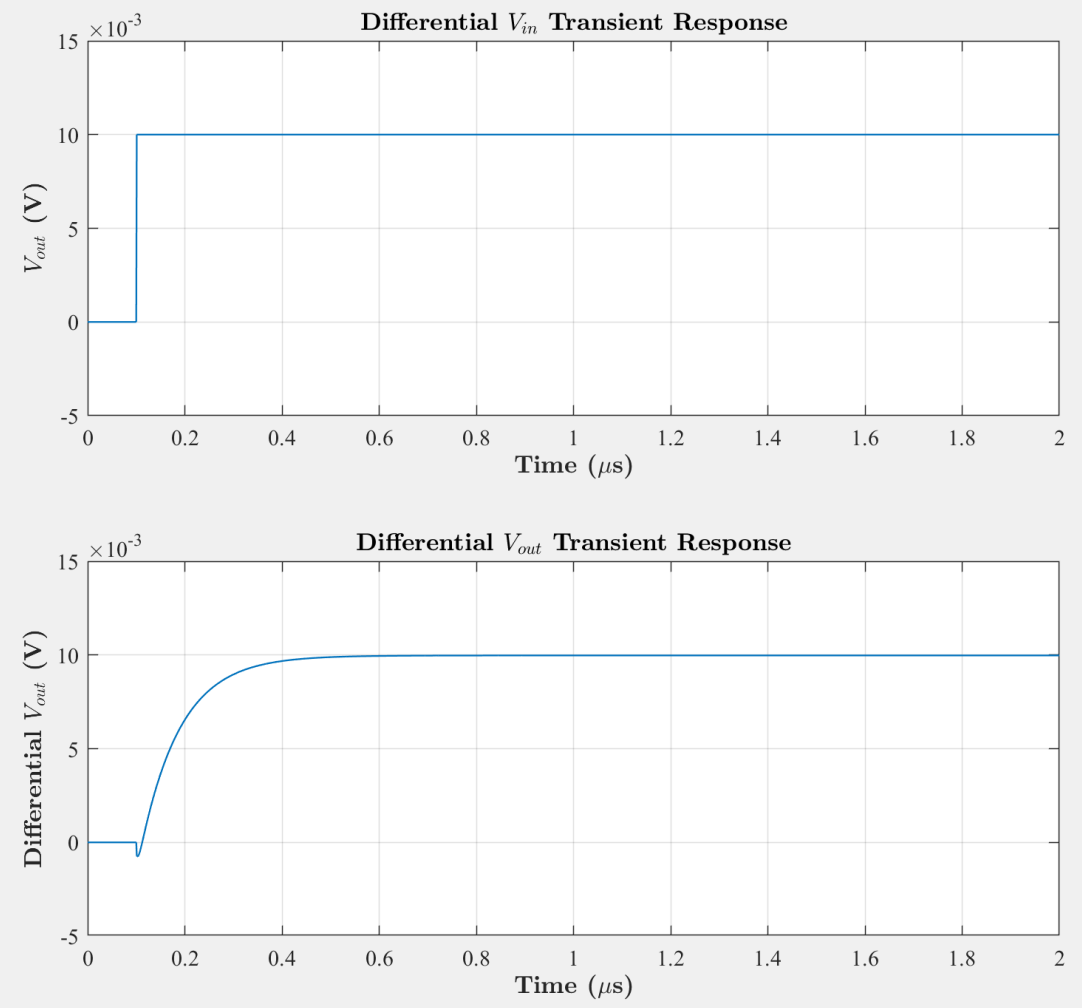
f. Supply sweep: implemented in Control Block 4. To analyze, ensure that the supply sweep Boolean is set to true in the MATLAB script.

Expected plot:



g. Transient step response: implemented in the *diff_ota_tb_commands.spice* file. To analyze, ensure that the transient analysis Boolean is set to true in the MATLAB script.

Expected plot:



Final specifications:

Specification	Design Target	Achieved
Open-loop gain	60 dB	~61 dB
Static closed-loop gain error	0.1%	0.25%
0.1% settling time	< 1 μs	0.573 μs
Output current	10 μA	10 μA
Input common-mode range	0.7 V - 1.5 V	0.6 V - 1.8 V
Bandwidth	5 kHz	~8.0 kHz
Total integrated output noise.	10 mV	14.8 mV
Phase margin	> 70 degrees	~74.5 degrees
PVT Robustness	✓	✓

II. G_m-C Filter Transconductor Stage

Schematic: *designs/gm_c_stage.sch*
Symbol: *designs/gm_c_stage.sym*
SPICE netlist: *simulations/gm_c_stage.spice*
Simulations/tests file: *simulations/gm_c_stage_commands.spice*

- This file includes the parameter values for the final design.

Analysis: "Differential OTA/Op-Amp Analysis" section of *spice_post_process.m*.

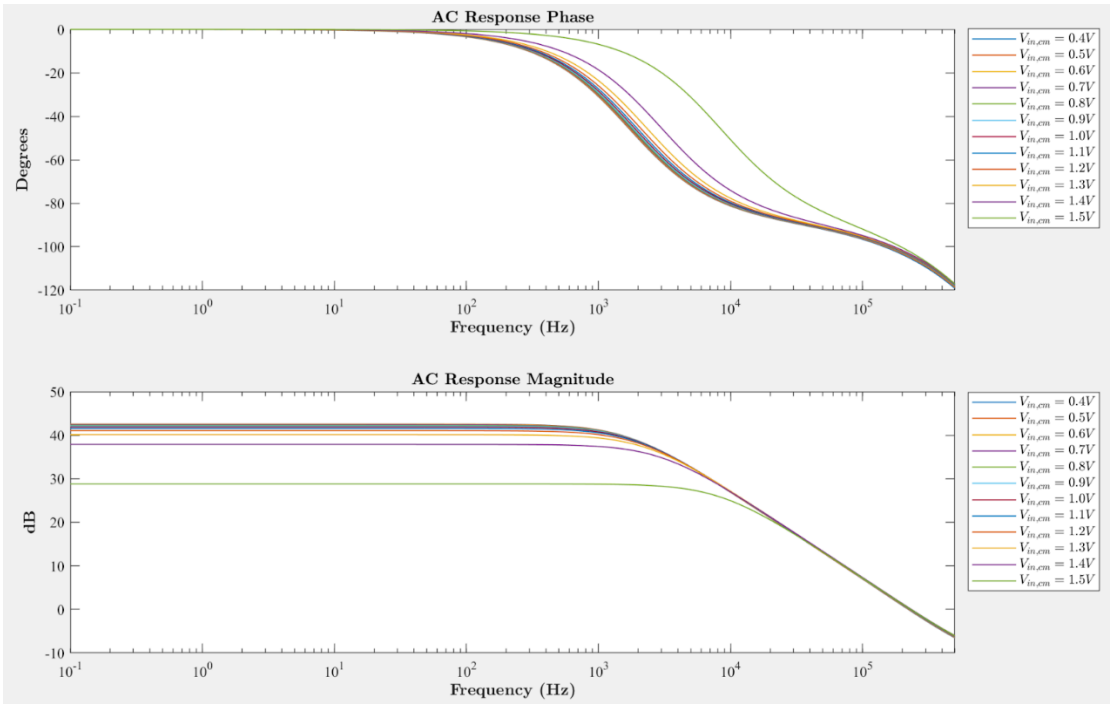
Tests:

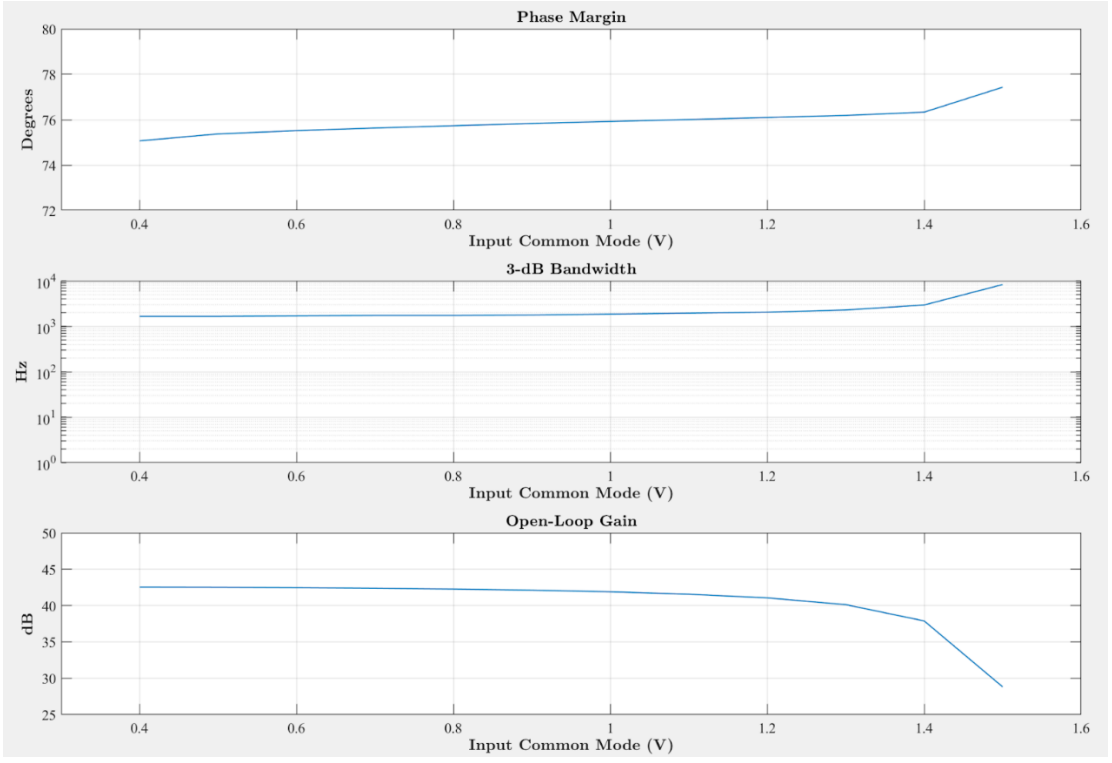
a. Operating point simulation: performed in all of the control blocks.

b. Single-point AC analysis: implemented in Control Block 1. Simply uncomment the AC analysis line. To analyze, ensure that the correct *.raw* file name is uncommented in the MATLAB script and that the AC analysis sweep Boolean is set to false.

c. Input common-mode sweep AC analysis: implemented in Control Block 2. To analyze, ensure that the AC analysis sweep Boolean is set to true in the MATLAB script and that the file names are correct.

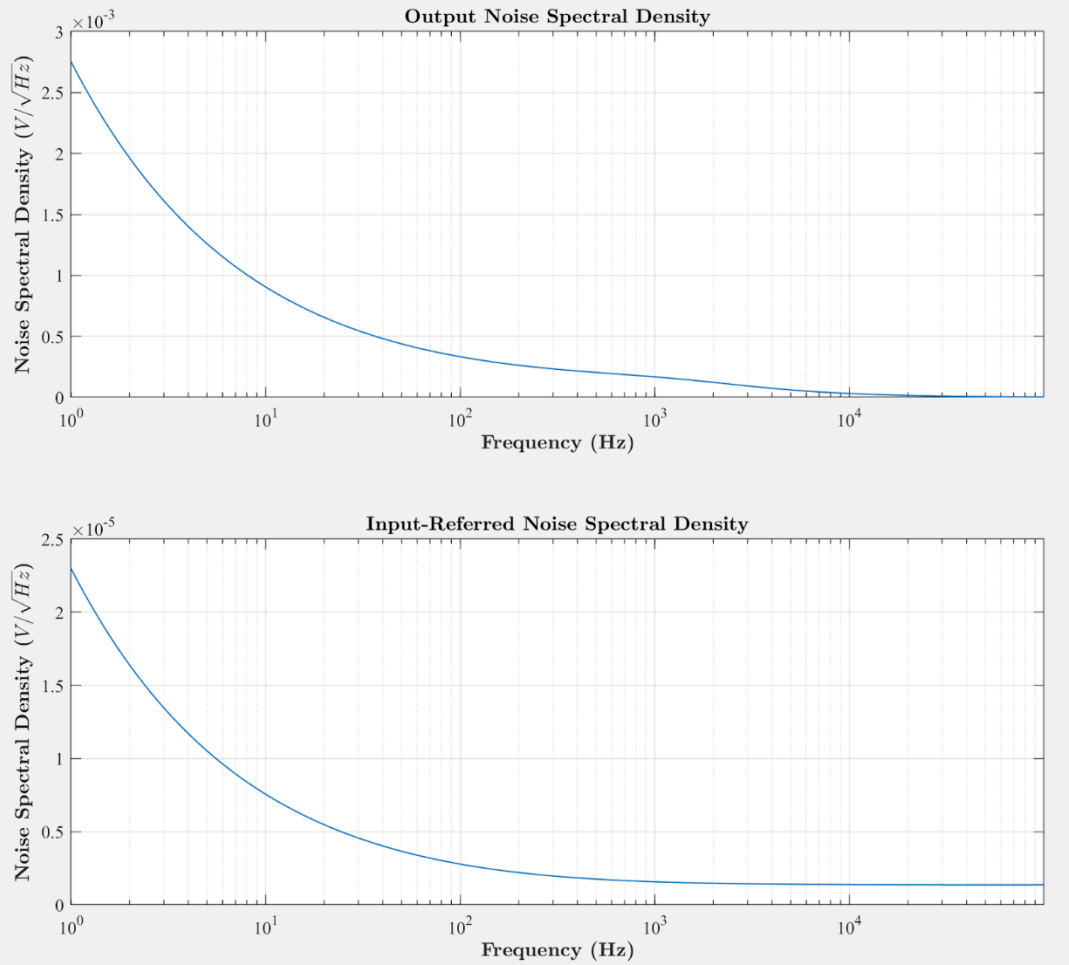
   Expected plots:





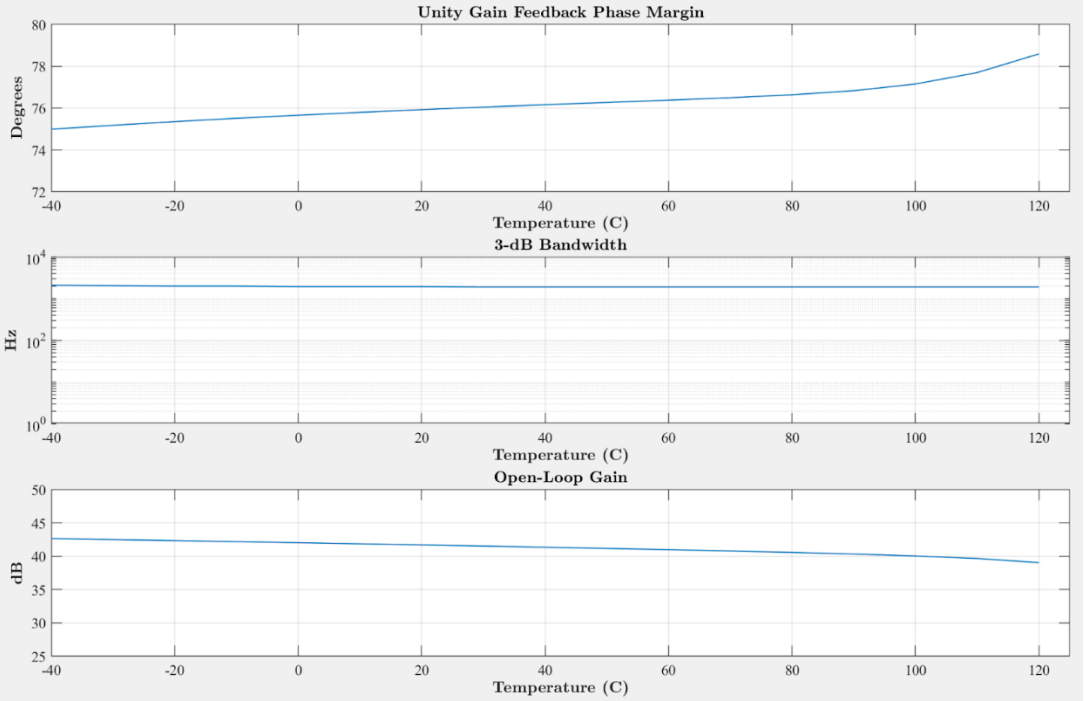
d. Noise analysis: implemented in Control Block 1. Simply comment out the noise analysis lines (mode 1 or mode 2). To analyze, ensure that the noise analysis Boolean is set to true (selects between the two noise analysis modes).

Expected noise spectrums:



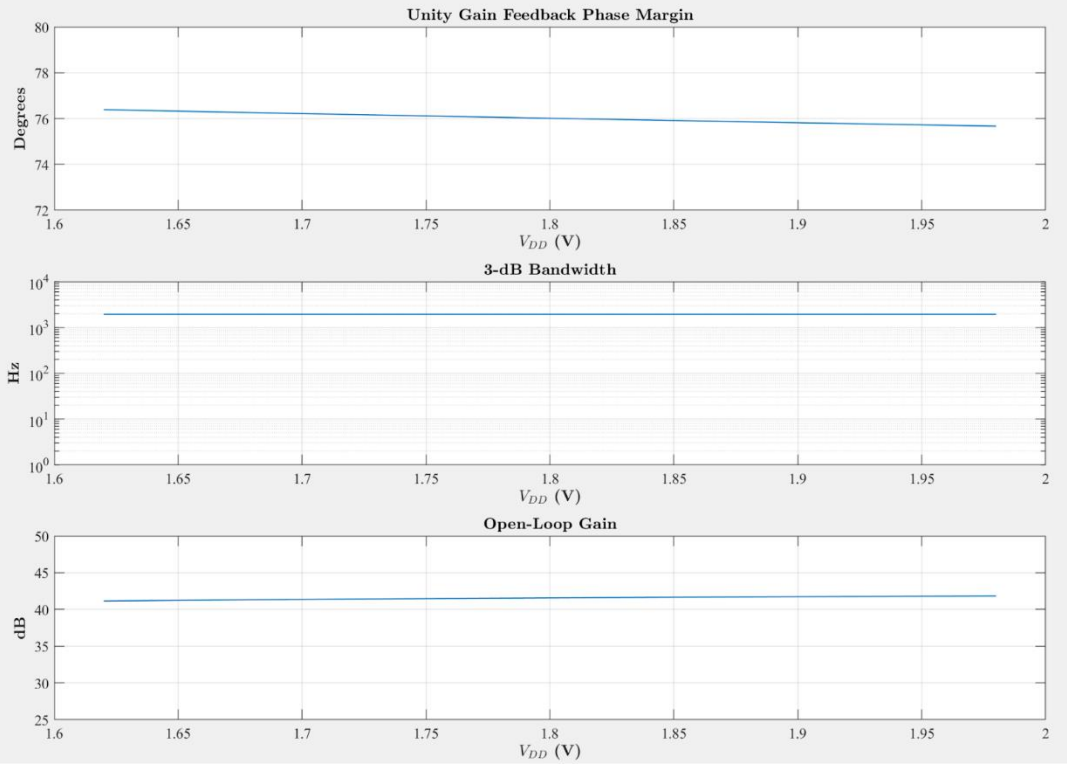
e. Temperature sweep AC analysis: implemented in Control Block 3. To analyze, ensure that the temperature sweep Boolean is set to true in the MATLAB script.

Expected plot:



f. Supply sweep: implemented in Control Block 4. To analyze, ensure that the supply sweep Boolean is set to true in the MATLAB script.

Expected plot:



Final specifications:

Specification	Design Target	Achieved
Open-loop gain	Not critical	~42 dB
Output current	~1 nA	1 nA
Input common-mode range	0.7 V - 1.5 V	0.4 V - 1.5 V
Unloaded bandwidth	1 kHz	~2.0 kHz
Total integrated output noise (up to 100 kHz).	10 mV	11.9 mV
Phase margin	> 70 degrees	~76 degrees
PVT Robustness	✓	✓

III. Input Amplifier/Single-Ended to Differential Converter

Schematic: *designs/input_amplifier.sch*
SPICE netlist: *simulations/input_amplifier.spice*
Simulations/tests file: *simulations/input_amplifier_commands.spice*

- This file includes the parameter values for the final design.

Analysis: "Input Amplifier Analysis" section of *spice_post_process.m*.

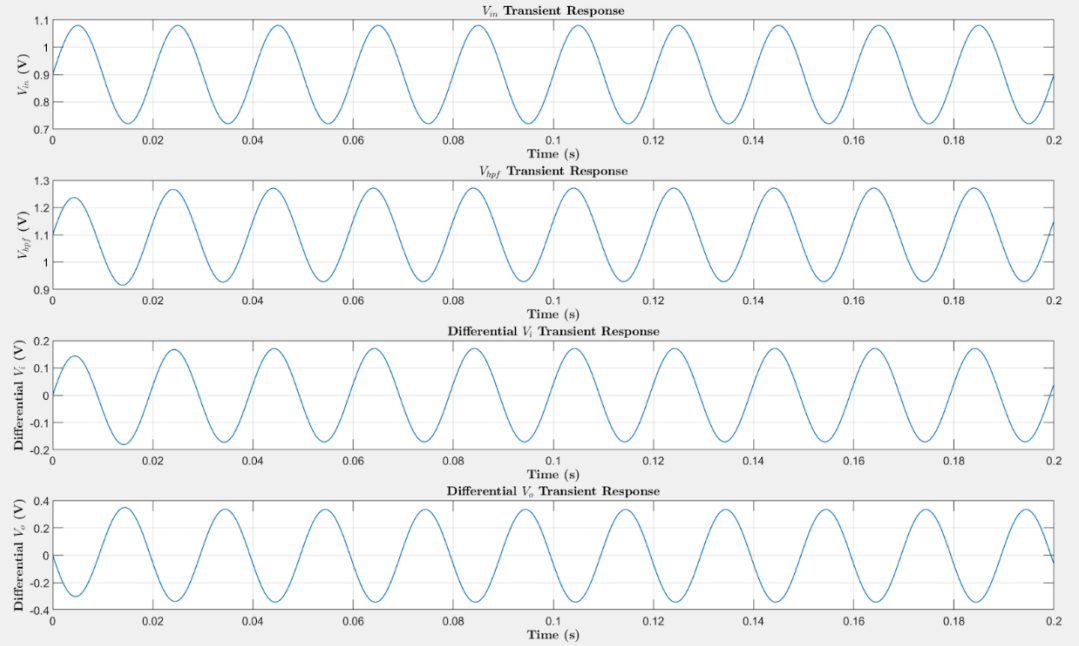
Tests:

a. Operating point simulation.

b. Transient simulation: to adjust the gain of the input amplifier simply vary the digital control signals *gain_ctrl_0* and *gain_ctrl_1*. The input common-mode range can also be adjusted in the commands SPICE file.

- Gain 0.5 configuration: *gain_ctrl_0* = 0 and *gain_ctrl_1* = '*vdd*'.
- Gain 1 configuration: *gain_ctrl_0* = '*vdd*' and *gain_ctrl_1* = '*vdd*'.
- Gain 2 configuration: *gain_ctrl_0* = 0 and *gain_ctrl_1* = 0.
- Gain 4 configuration: *gain_ctrl_0* = '*vdd*' and *gain_ctrl_1* = 0.

Expected plot for the gain 2 configuration:



Final specifications:

The circuit met the requirements of a variable gain, single-ended to differential converter at the input of the chip.

IV. Active Low-Pass G_m-C Biquad Filter

Schematic: *designs/active_lowpass_filter.sch*
SPICE netlist: *simulations/active_lowpass_filter.spice*
Simulations/tests file: *simulations/active_lowpass_filter_commands.spice*

- This file includes the parameter values for the final design.

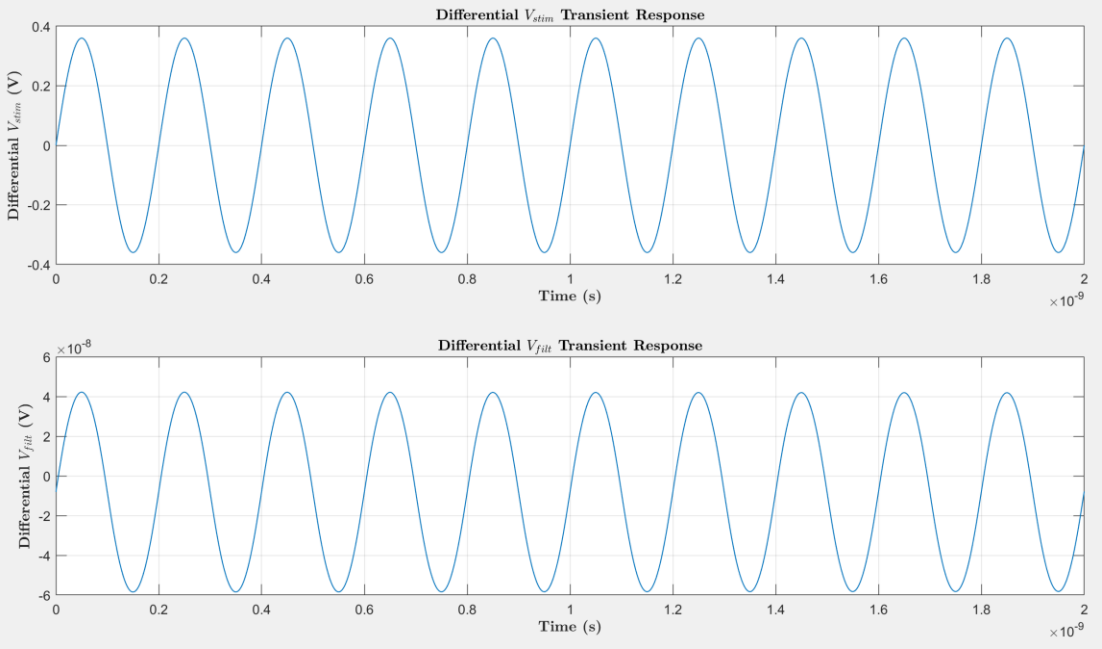
Analysis: "Active Low-Pass Filter Analysis" section of *spice_post_process.m*.

Tests:

a. Operating point simulation.

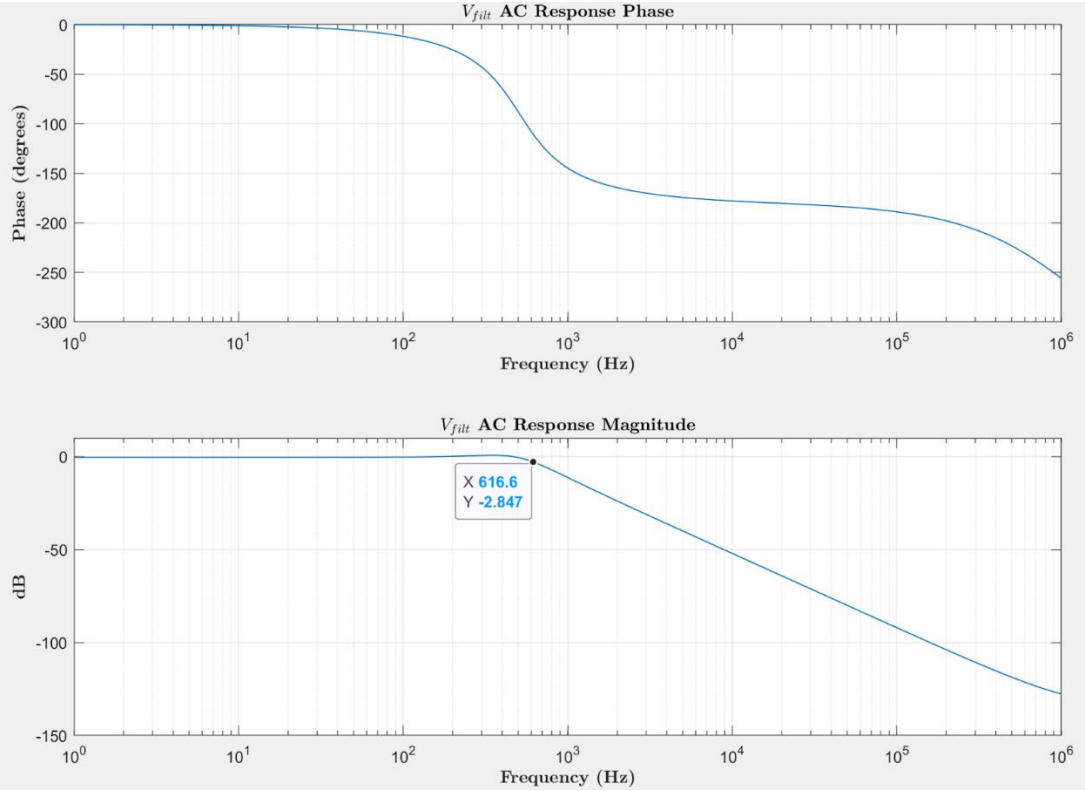
b. Transient simulation: this simply illustrates the expected signal attenuation at frequencies past the cutoff frequency for the biquad filter. The frequency of the input signal can easily be adjusted in the commands SPICE file.

   Expected plot for an input frequency of 5 GHz:



c. AC simulation: this provides information about the filter cutoff frequency.

Expected plot:



Final specifications:

The circuit met the requirements for a cutoff frequency less than 1 kHz with a roll-off of 40 dB typical of a biquad filter.

V. Continuous-Time Comparator

Schematic: *designs/comparator.sch*
SPICE netlist: *simulations/comparator.spice*
Simulations/tests file: *simulations/comparator_commands.spice*

- This file includes the parameter values for the final design.

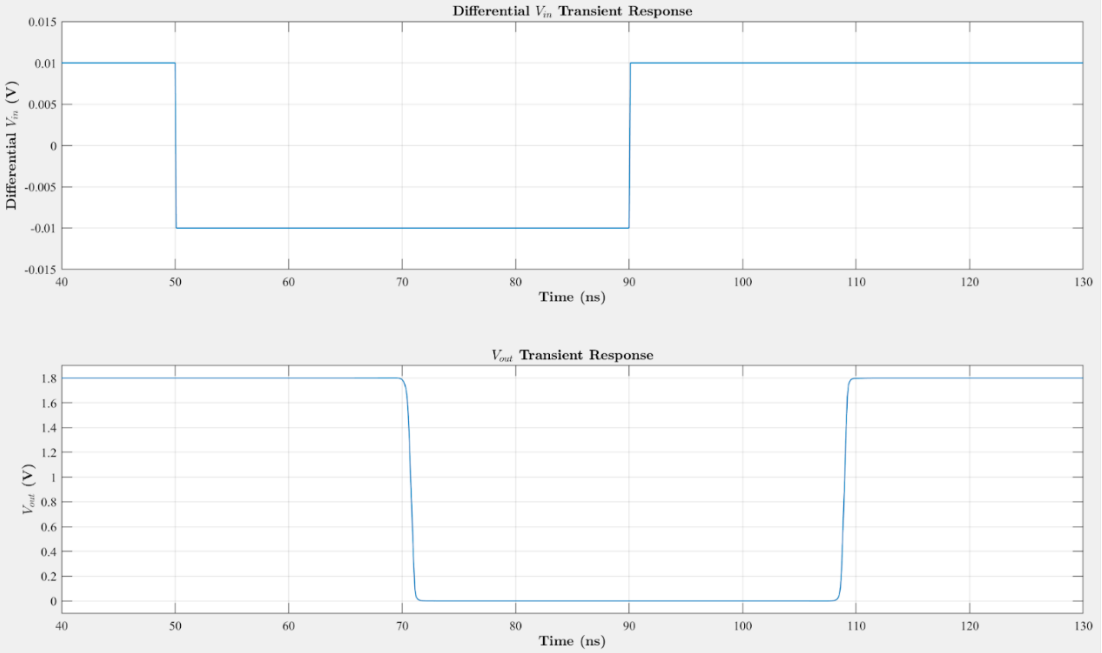
Analysis: "Comparator Analysis" section of *spice_post_process.m*.

<u>Tests:</u>

a. Operating point simulation.

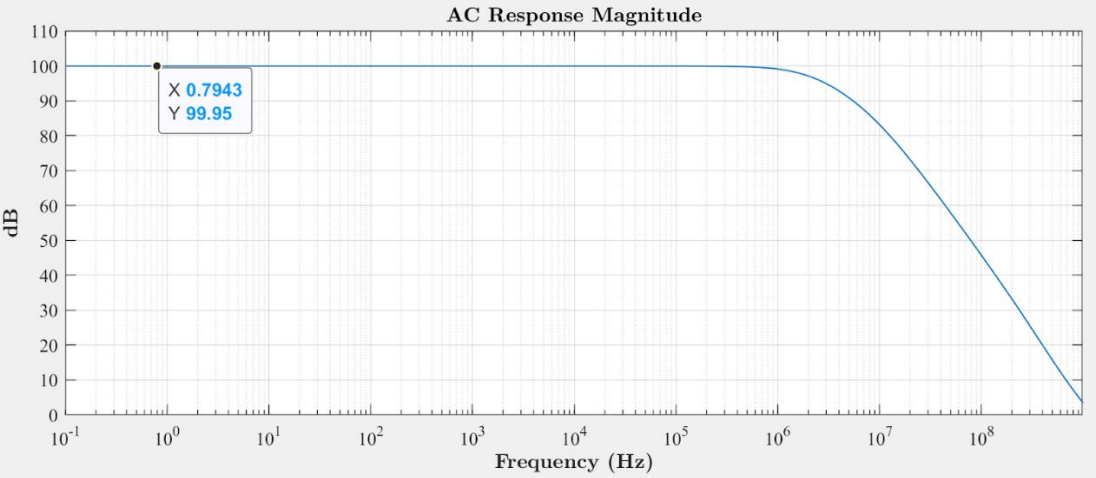
b. Transient simulation: this yields the switching response time for the comparator as well as the rise and fall times.

Expected plot:

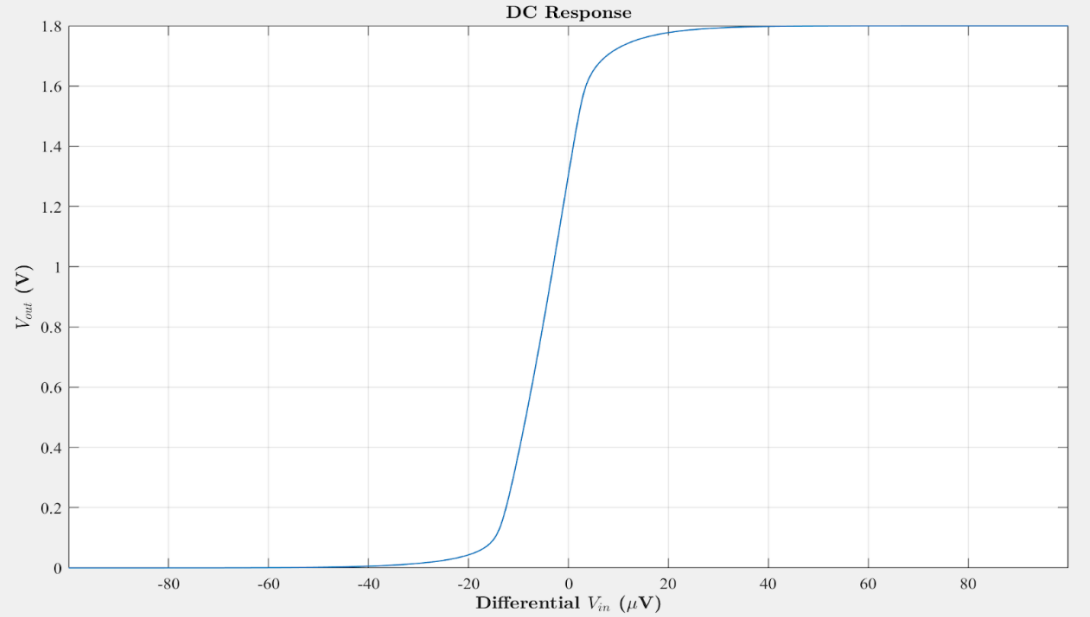


c. AC analysis: this verifies the expected high gain of the comparator.

Expected plot:



d. DC analysis: this yields the input offset voltage of the continuous-time comparator.



Final specifications:

Specification	Design Target	Achieved
Open-loop gain	Very high	~100 dB
Input offset voltage	< 1 mV	~5 μV
Bias current	10 μA	10 μA
Response time	< 1 μs	~20 ns
10-90% rise/fall time	< 1 μs	~1 ns

VI. Single-Ended Output Operational Transconductance Amplifier (OTA)

Schematic: *designs/se_fold_casc_wide_swing_opamp.sch*
Symbol: *designs/se_fold_casc_wide_swing_opamp.sym*
SPICE netlist: *simulations/se_fold_casc_wide_swing_opamp.spice*
Simulations/tests file: *simulations/se_fold_casc_wide_swing_opamp_commands.spice*

- This file includes the parameter values for the final design.
- Used for all simulations except the transient step response.

Transient simulation schematic: *designs/se_ota_tb.sch*
Transient simulation SPICE netlist: *simulations/se_ota_tb.spice*
Transient simulation test file: *simulations/se_ota_tb_commands.spice*

- This file includes the appropriate parameter values.

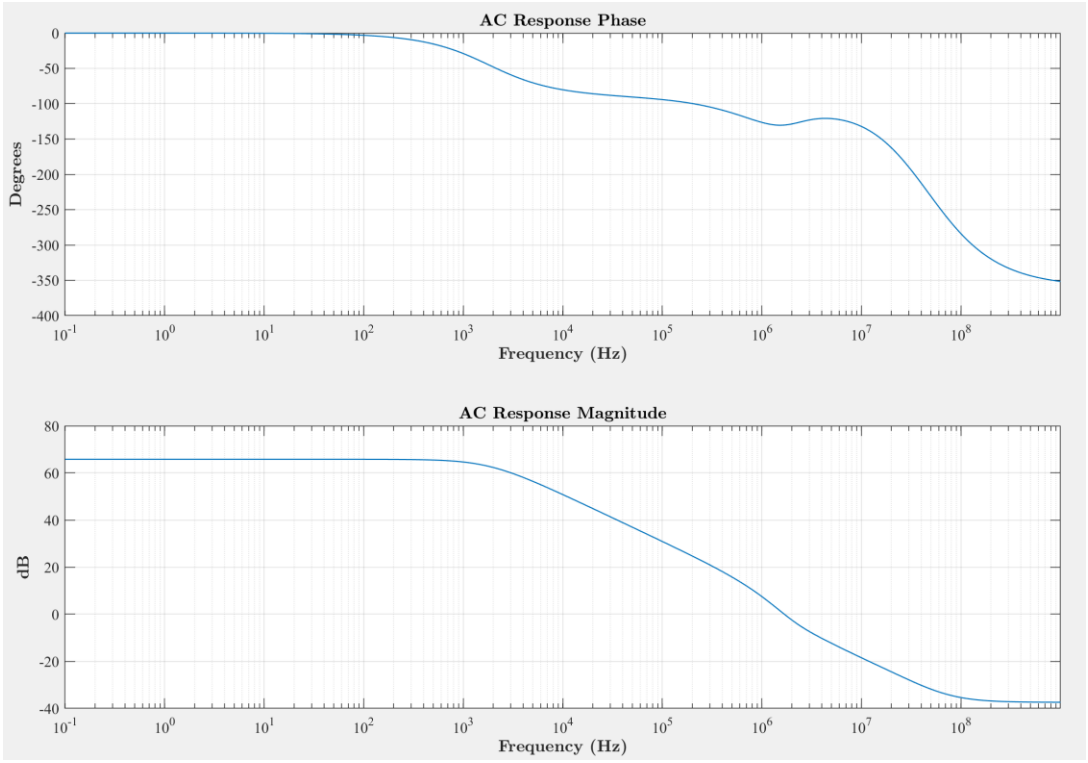
Analysis: "Single-Ended OTA/Op-Amp Analysis" section of *spice_post_process.m*.

Tests:

a. Operating point simulation.

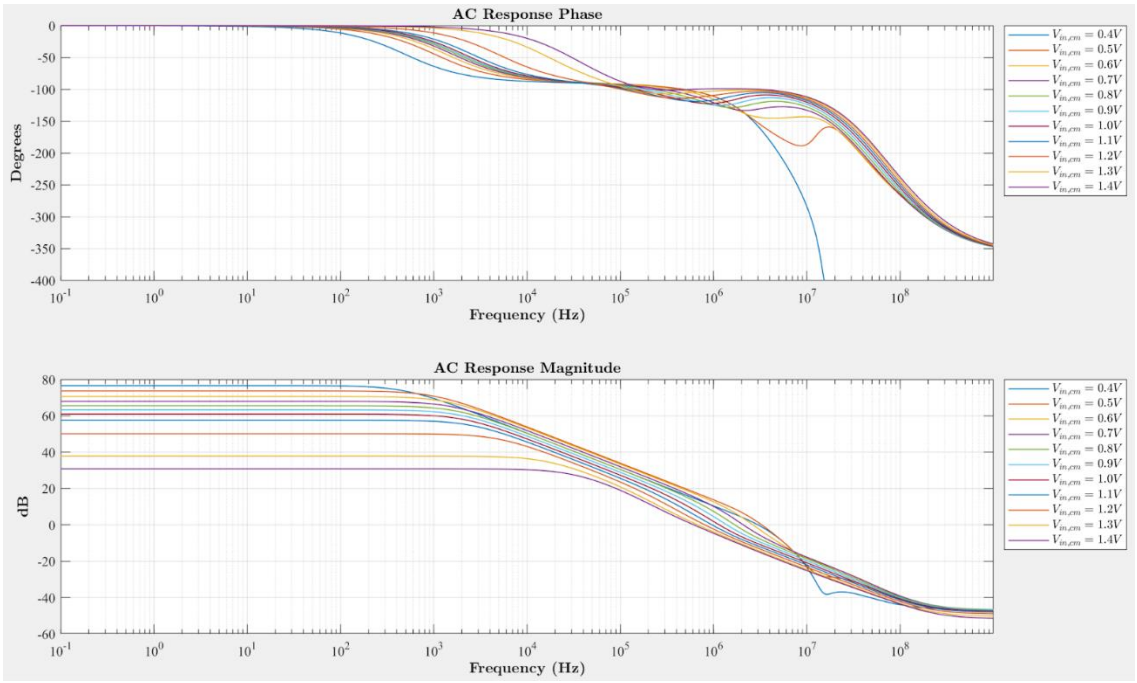
b. Single-point AC analysis: implemented in Control Block 1. Simply uncomment the AC analysis line. To analyze, ensure that the correct *.raw* file name is uncommented in the MATLAB script and that the AC analysis sweep Boolean is set to false.

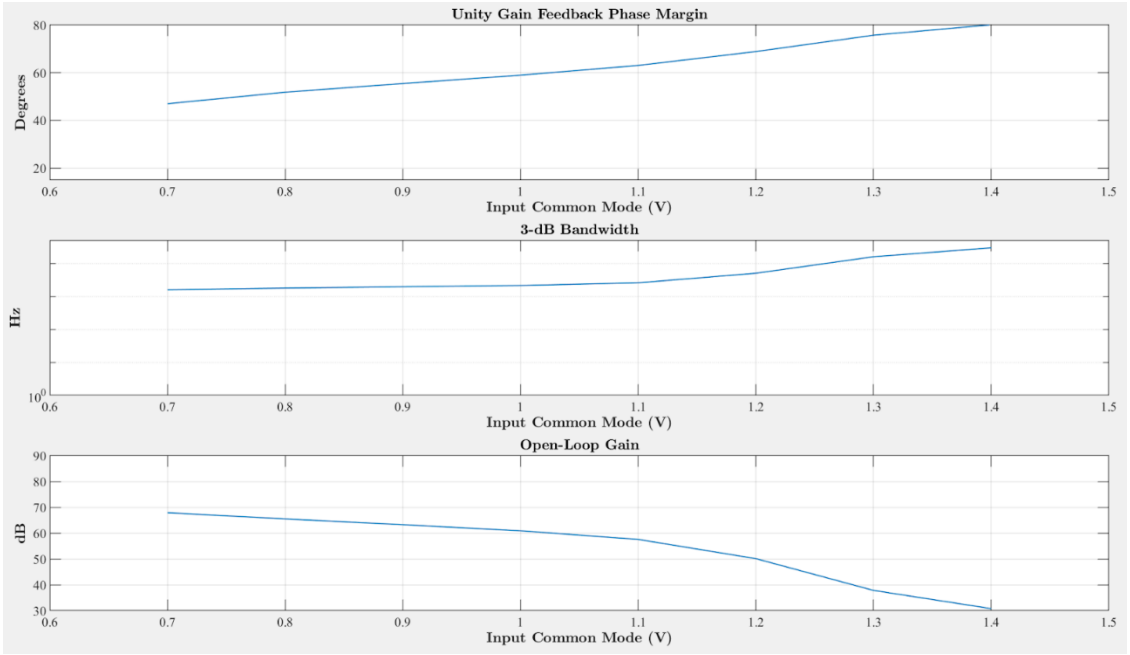
   Expected plot for *vincm* = 1.0 V:



c. Input common-mode sweep AC analysis: implemented in Control Block 2. To analyze, ensure that the AC analysis sweep Boolean is set to true in the MATLAB script and that the file names are correct.

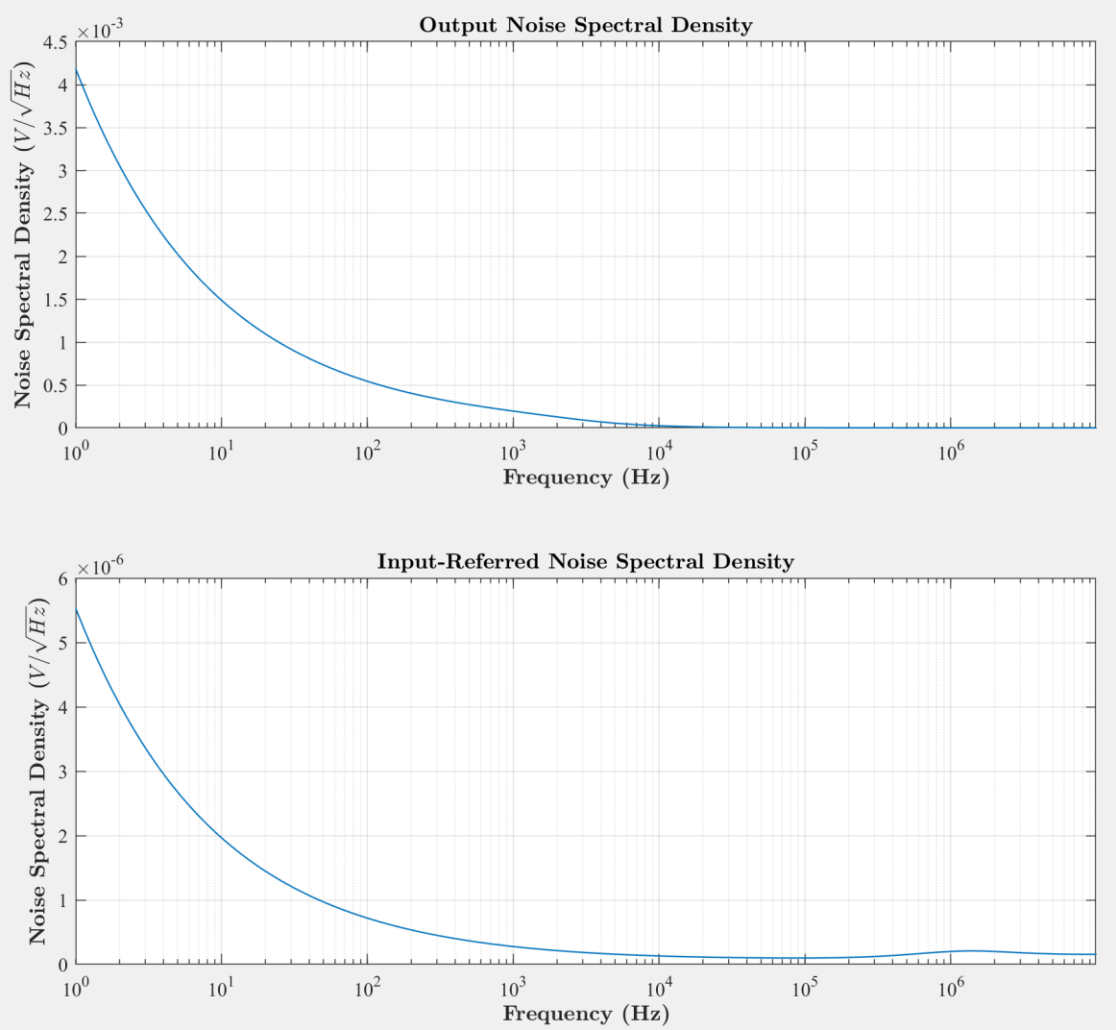
Expected plots:





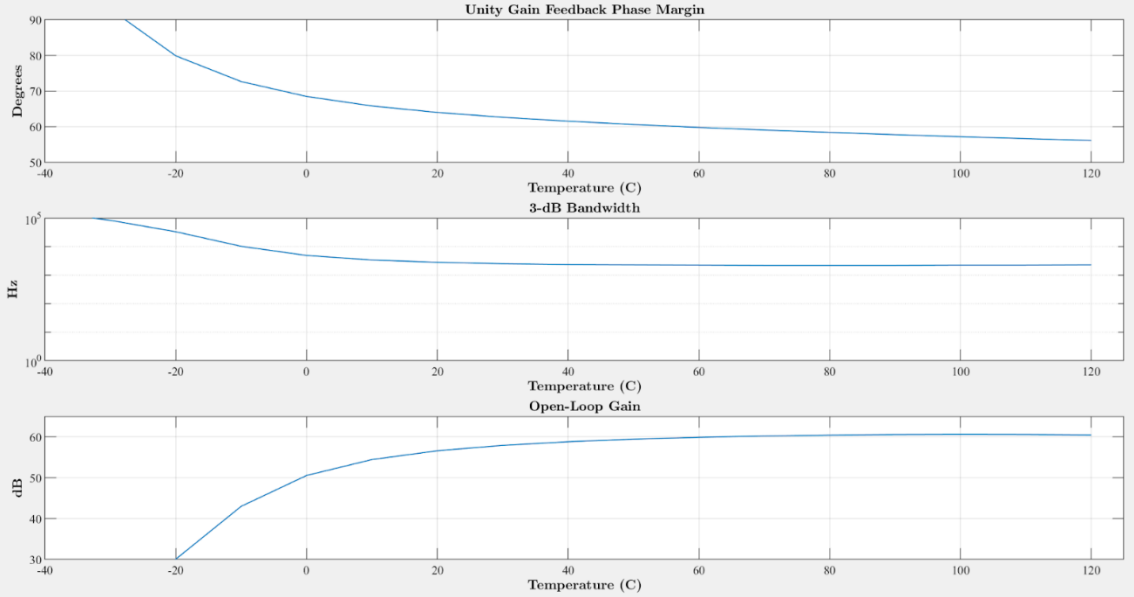
d. Noise analysis: implemented in Control Block 1. Simply comment out the noise analysis lines (mode 1 or mode 2). To analyze, ensure that the noise analysis Boolean is set to true (selects between the two noise analysis modes).

   Expected noise spectrums:



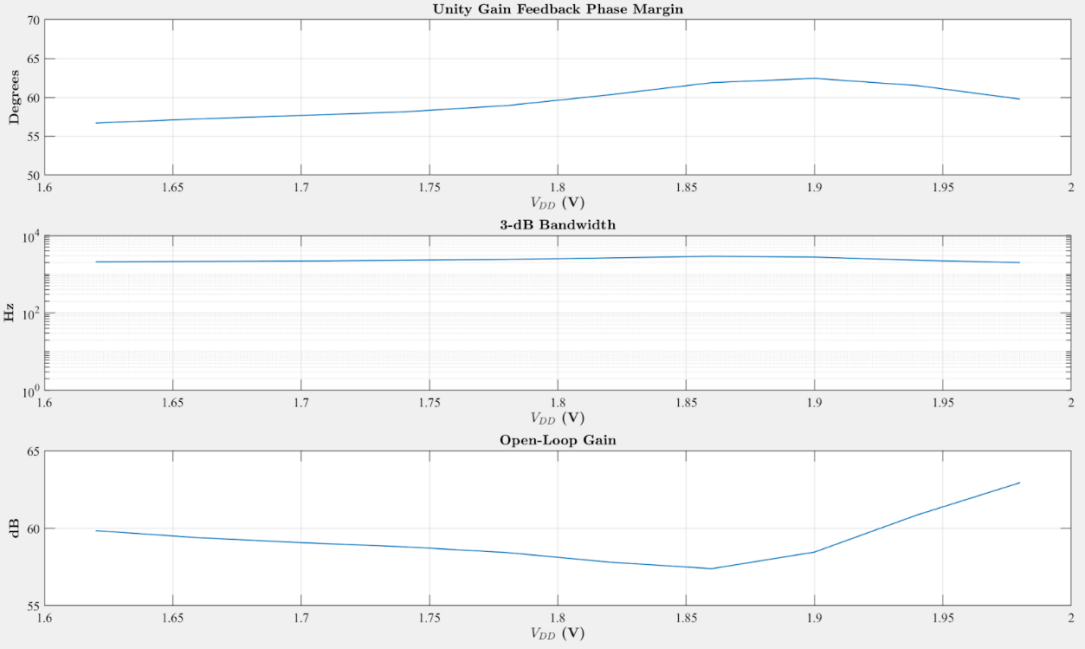
e. Temperature sweep AC analysis: implemented in Control Block 3. To analyze, ensure that the temperature sweep Boolean is set to true in the MATLAB script.

Expected plot:



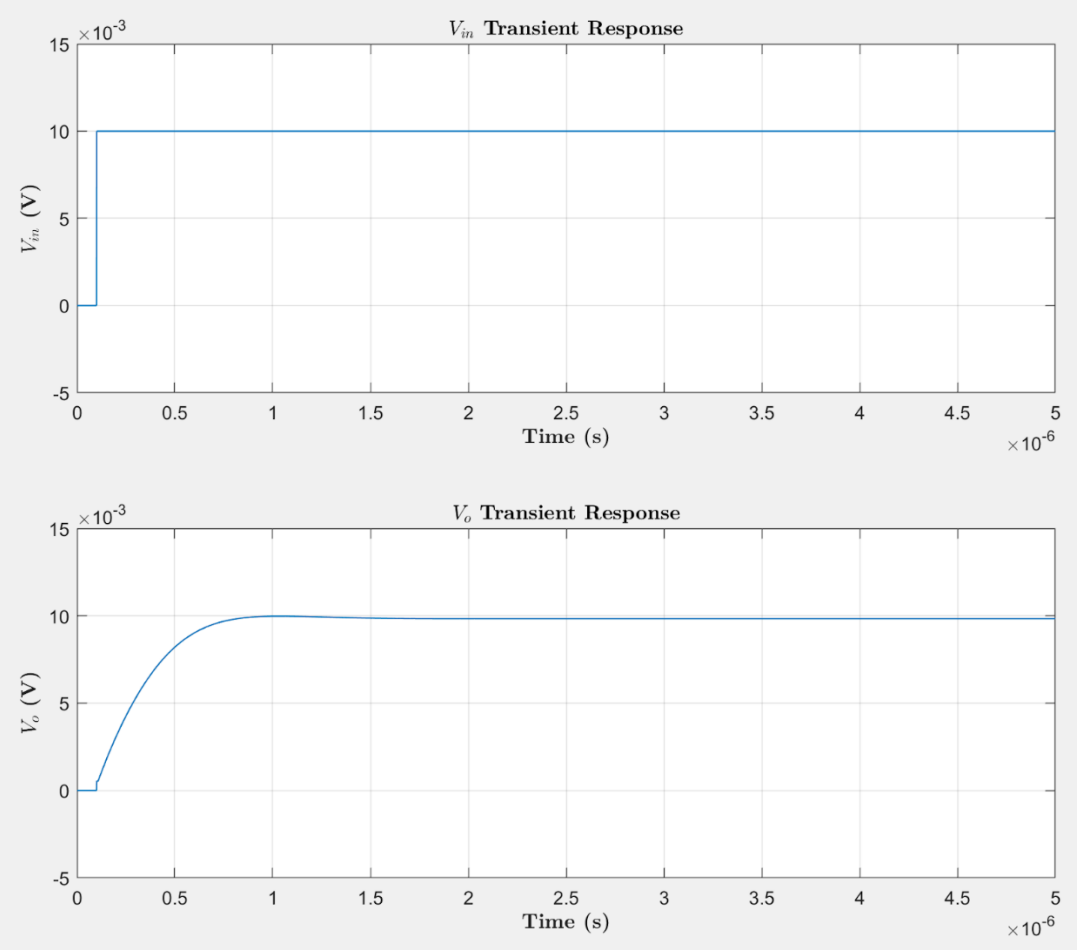
f. Supply sweep: implemented in Control Block 4. To analyze, ensure that the supply sweep Boolean is set to true in the MATLAB script.

Expected plot:



g. Transient step response: implemented in the *se_ota_tb_commands.spice* file. To analyze, ensure that the transient analysis Boolean is set to true in the MATLAB script.

Expected plot:



Final specifications:

Specification	Design Target	Achieved
Open-loop gain	60 dB	~60 dB
Static closed-loop gain error	1.00%	1.60%
0.1% settling time	< 10 μs	1.852 μs
Bias current	10 μA	10 μA
Input common-mode range	0.7 V - 1.4 V	0.7 V - 1.4 V
Bandwidth	1 kHz	~1.5 kHz
Total integrated output noise.	10 mV	15.6 mV
Phase margin	> 60 degrees	~60 degrees
PVT Robustness	✓	✓

VII. Current-Starved Ring Oscillator

Schematic: *designs/cs_ring_osc_tb.sch*
SPICE netlist: *simulations/cs_ring_osc_tb.spice*
Simulations/tests file: *simulations/cs_ring_osc_tb_commands.spice*

- This file includes the parameter values for the final design.

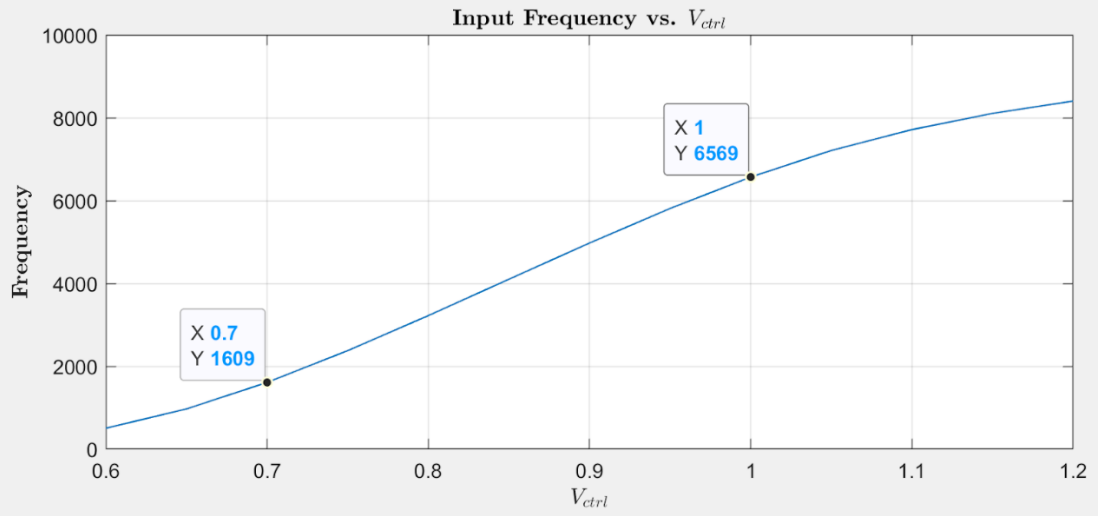
Analysis: "Ring Oscillator Analysis" section of *spice_post_process.m*.

Tests:

a. Operating point simulation.

b. Transient simulation: this ensures a linear frequency range of at least 0.3 V for the applied control voltage. Ensure that the correct lower and upper bounds for the control voltage sweep are set in the both the commands SPICE file and the MATLAB analysis script. Also ensure that the *freq_div* Boolean is set to false in order to select the correct input file set.

   Expected plot of frequency as a function of control voltage:



Final specifications:

The circuit met the requirements for a linear frequency range of at least 0.3 V. As shown by that plot above, the frequency linearly increases for control voltages between 0.7 V and 1.0 V.

VIII. Frequency Divider

Schematic: *designs/cs_ring_osc_div.sch*
SPICE netlist: *simulations/cs_ring_osc_div.spice*

Simulations/tests file: *simulations/cs_ring_osc_div_commands.spice*

- This file includes the parameter values for the final design.

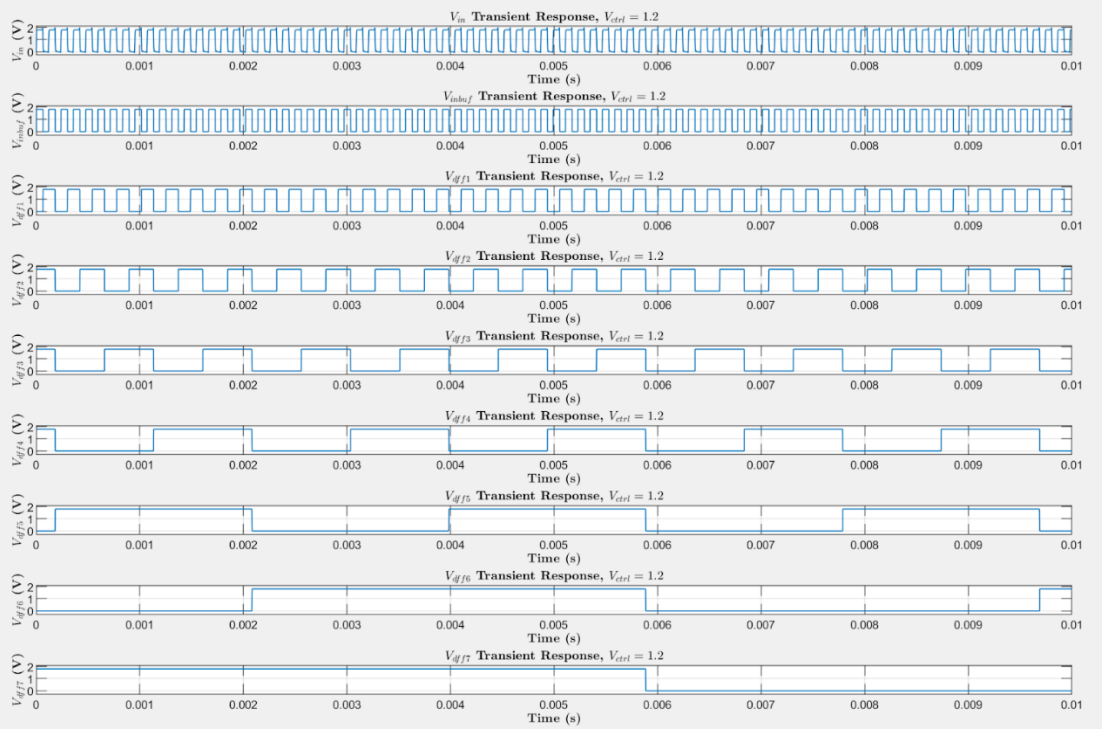
Analysis: "Ring Oscillator Analysis" section of *spice_post_process.m*.

Tests:

a. Operating point simulation.

b. Transient simulation: this ensures that the output clock from the ring oscillator is divided down appropriately at each DFF stage. Ensure that the correct lower and upper bounds for the control voltage sweep are set in the both the commands SPICE file and the MATLAB analysis script. To run just a single control voltage simulation, set the two bounds equal to each other. Also ensure that the *freq_div* Boolean is set to true in order to select the correct input file.

Expected plot:



Final specifications:

The circuit met the requirements for frequency division.

IX. Phase-Frequency Detector / Charge Pump / Low-Pass Filter

Schematic: *designs/pfd_cp_lpf_test.sch*
SPICE netlist: *simulations/pfd_cp_lpf_test.spice*
Simulations/tests file: *simulations/pfd_cp_lpf_test_commands.spice*

- This file includes the parameter values for the final design.

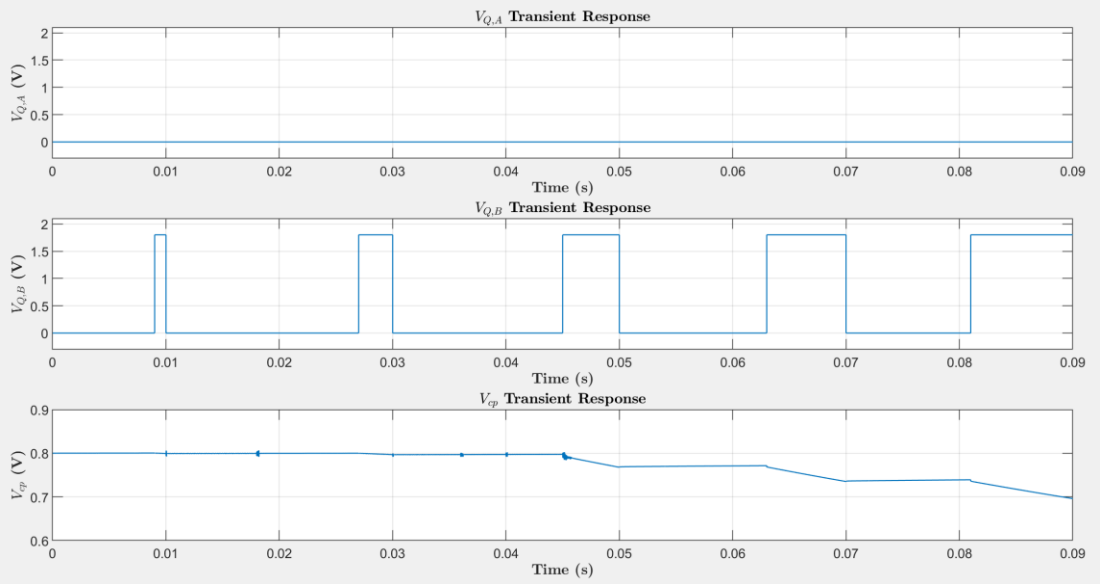
Analysis: "PFD-CP-LPF Analysis" section of *spice_post_process.m*.

Tests:

a. Operating point simulation.

b. Transient simulation: this ensures that the correct integration takes place for a given set of A and B inputs to the phase-frequency detector.

Expected plot:



Final specifications:

The circuit met the target requirements, performing linear integration during the charging periods while maintaining a constant voltage during the static periods. Furthermore, the circuit works correctly down to an output voltage of 0.7 V, the lower bound for the PLL control voltage.

X. Low-Frequency Phase-Locked Loop (PLL)

Schematic: *designs/low_freq_pll.sch*
SPICE netlist: *simulations/low_freq_pll.spice*
Simulations/tests file: *simulations/low_freq_pll_commands.spice*

- This file includes the parameter values for the final design.

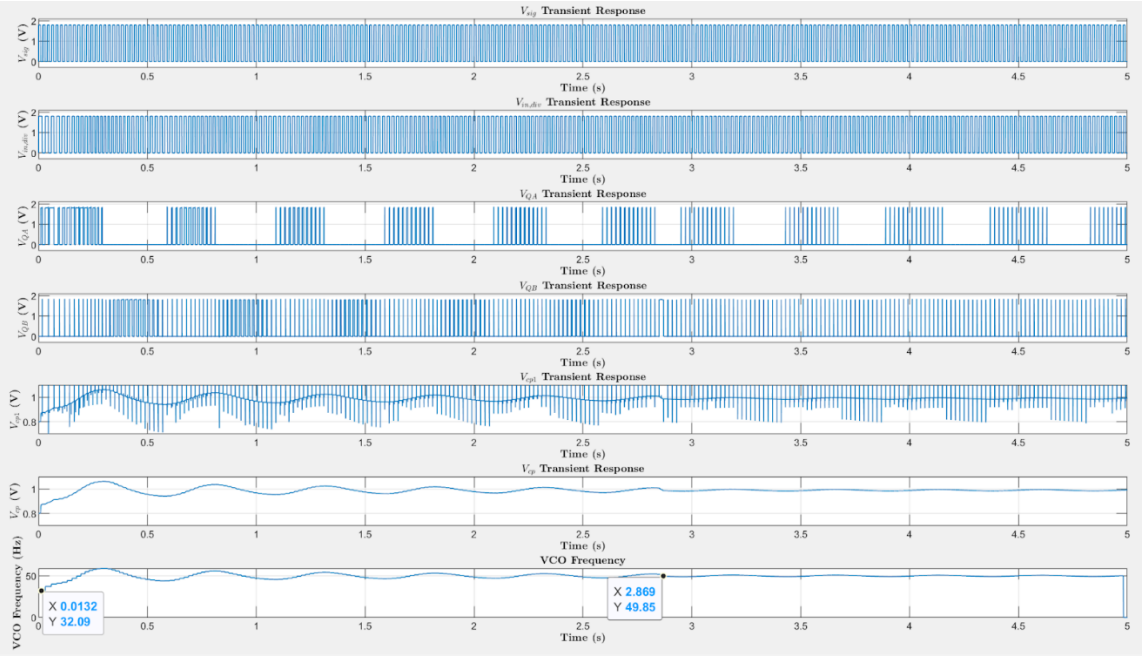
Analysis: "Low-Frequency PLL Analysis" section of *spice_post_process.m*.

Tests:

a. Operating point simulation.

b. Transient simulation: this ensures that the loop settles within a couple seconds for a worst-case step in the input frequency.

Expected plot:



Final specifications:

The circuit met the target requirements, settling within a couple seconds to the desired control voltage. Given confirmation of this worst-case operation, the loop is thus expected to function as expected for gradual changes in the input frequency.

XI. Peak/Envelope Detector Circuit

Schematic: *designs/peak_detector.sch*
SPICE netlist: *simulations/peak_detector.spice*
Simulations/tests file: *simulations/peak_detector_commands.spice*

- This file includes the parameter values for the final design.

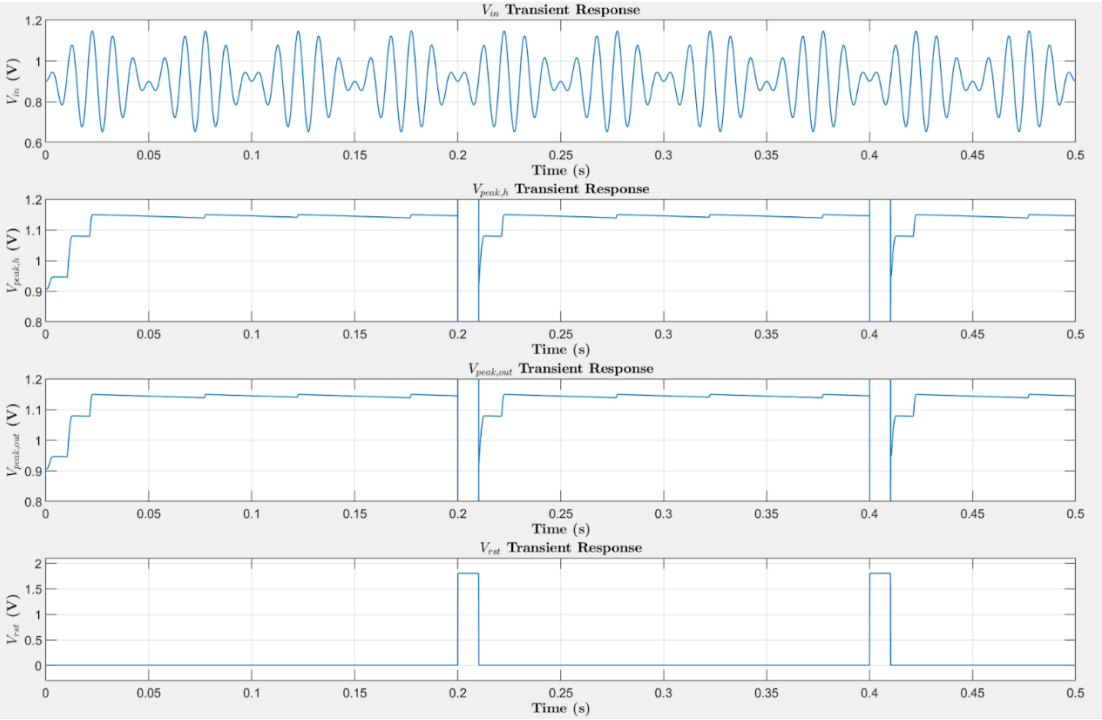
Analysis: “Peak Detector Analysis” section of *spice_post_process.m*.

Tests:

a. Operating point simulation.

b. Transient simulation: this ensures that the peak detector circuit tracks the envelope of a given input signal, thereby effectively measuring its amplitude.

Expected plot:



Final specifications:

The circuit met the target requirements, tracking the input signal peak amplitude without excessive decay.

XII. Sample-and-Hold Circuit

Schematic: *designs/sample_and_hold_open_loop.sch*
SPICE netlist: *simulations/sample_and_hold_open_loop.spice*
Simulations/tests file: *simulations/sample_and_hold_open_loop_commands.spice*

- This file includes the parameter values for the final design.

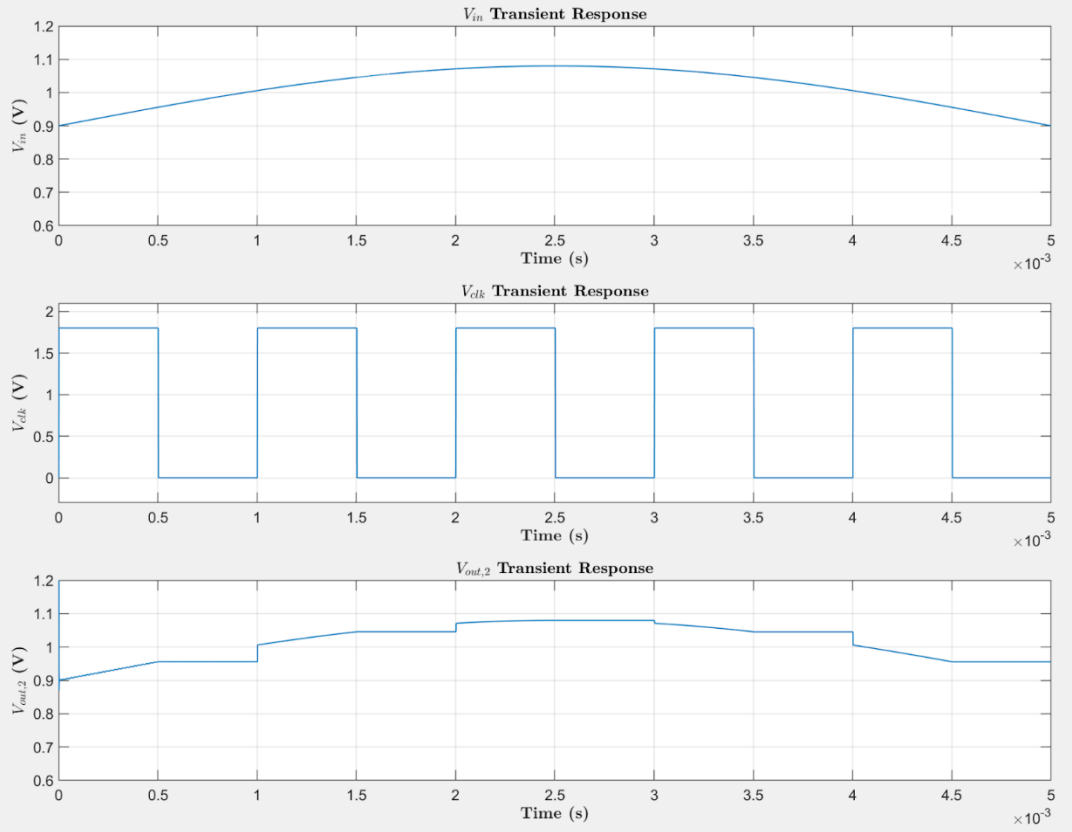
Analysis: "Peak Detector Analysis" section of *spice_post_process.m*.

Tests:

a. Operating point simulation.

b. Transient simulation: this ensures that the sample-and-hold circuit samples the input signal when the clock is high and holds its value when the clock is low.

Expected plot:



Final specifications:

The circuit met the target requirements, tracking the input signal precisely during the high clock periods and holding its value without leakage during the low clock periods.

XIII. Latched Comparator

Schematic: *designs/latched_comparator.sch*
SPICE netlist: *simulations/latched_comparator.spice*
Simulations/tests file: *simulations/latched_comparator_commands.spice*

- This file includes the parameter values for the final design.

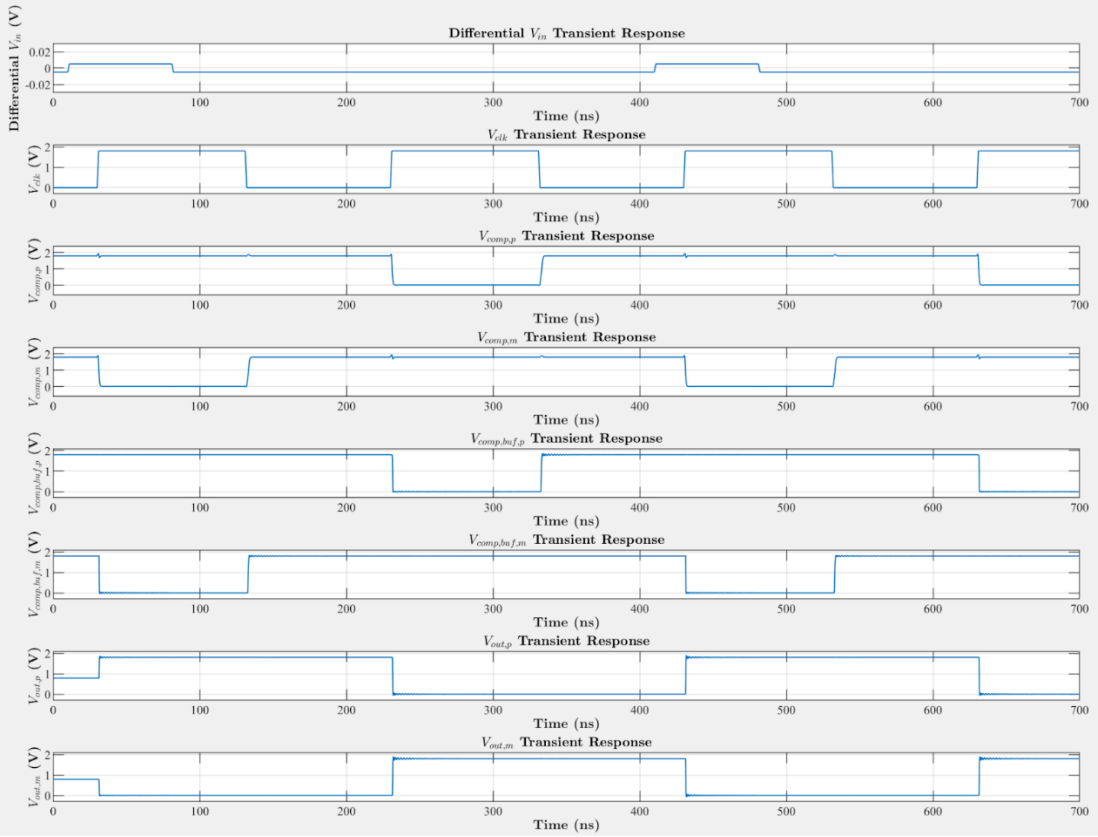
Analysis: "Latched Comparator Analysis" section of *spice_post_process.m*.

Tests:

a. Operating point simulation.

b. Transient simulation: this yields the switching response time for the comparator as well as the rise and fall times.

   Expected plot:



Final specifications:

Specification	Design Target	Achieved
Input offset voltage	< 1 mV	< 0.1 mV
Bias current	10 μA	10 μA
Falling edge *clk* → *Q* delay	< 1 μs	0.84 ns
Rising edge *clk* → *Q* delay	< 1 μs	1.05 ns
10-90% rise/fall time	< 1 μs	< 1 ns
Input common-mode range upper bound	> 1.2 V	1.45 V

XIV. 8-Bit Binary-Weighted Capacitive DAC

Schematic: *designs/dac_8bit.sch*
SPICE netlist: *simulations/dac_8bit_tran.spice*
Simulations/tests file: *simulations/dac_8bit_commands.spice*

- This file includes the parameter values for the final design.

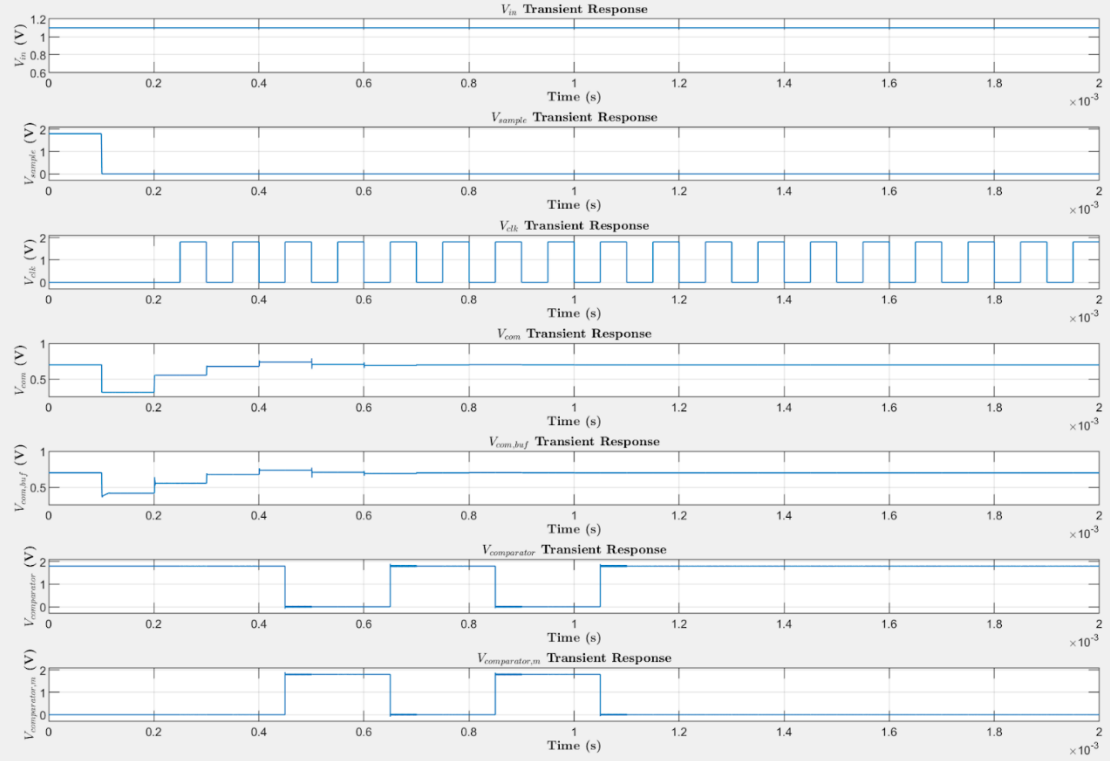
Analysis: “DAC Analysis” section of *spice_post_process.m*.

Tests:

a. Operating point simulation.

b. Transient simulation: this ensures that DAC input bits converge to the correct digital sequence for a given input analog voltage and that the DAC output voltage matches that at the positive terminal of the latched comparator.

Expected plot:



Final specifications:

The circuit met the target requirements, converging to the low voltage of 0.7 V by the bit in the input sequence and producing the correct sequence at the latched comparator output.

XV. SAR ADC Control Logic

Verilog code: *verilog/sar_adc_controller.v*
Verilog testbench: *verilog/sar_adc_controller_tb.v*
Makefile: *verilog/Makefile*

To run the test for the SAR ADC controller unit, simply invoke:

- *make run_adc*

The SAR ADC controller unit implements the timing logic for the ADC conversion sequence, taking in inputs from the simulated latched comparator and generating the output timing signals. The testbench for the SAR ADC controller performs a specified number of tests (default 50) and for each test, randomizes the 8-bit input value while generating the appropriate latched comparator output. It then checks that the final output value from the ADC matches the input sample.

Final specifications:

The controller met the target logic requirements.

XVI. IIR Notch Filter Estimator

Unit and test development is ongoing.

Verilog code: *verilog/iir_notch_filter.v*
Verilog helper files:
- *cordic_polar_to_rect.v*
- *cordic_rect_to_polar.v*
- *fixed_pt_div.v*

Verilog testbench: *verilog/iir_notch_filter_tb.v*
Makefile: *verilog/Makefile*

To run the test for the IIR filter estimator unit, simply invoke:

- *make run_iir*

XVII. Amplitude Processing Top-Level, End-to-End Circuit

Schematic: *designs/amplitude_processing_top_level.sch*
SPICE netlist: *simulations/amplitude_processing_top_level_tran.spice*
Simulations/tests file: *simulations/amplitude_processing_top_level_commands.spice*
- This file includes the parameter values for the final design.

Analysis: "Top-Level Amplitude Processing Analysis" section of *spice_post_process.m*.

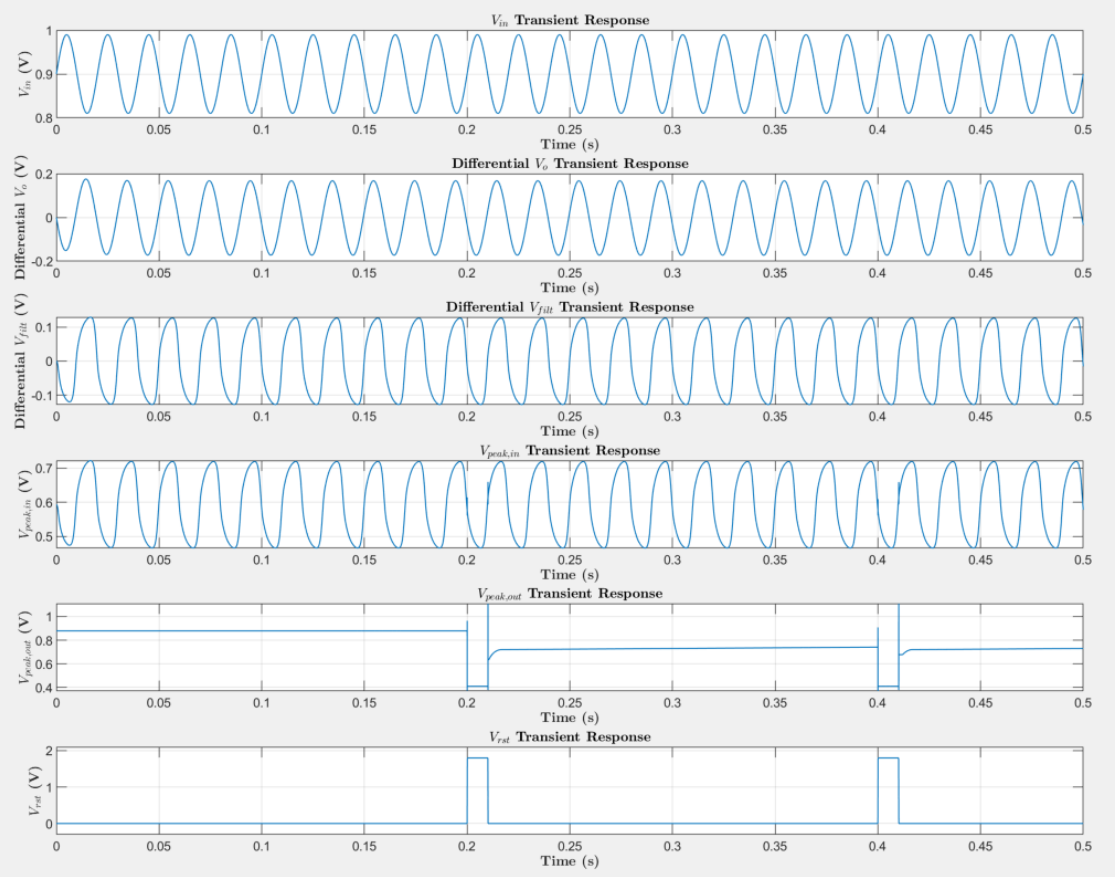
Tests:

a. Operating point simulation.

b. Transient simulation: this test runs a top-level simulation of the system from the signal input to the chip to the amplitude detection SAR ADC input. This path consists of the input amplifier, active low-pass biquad filter, peak-detector, and

sample-and hold circuit. The output is expected to sample the correct peak values of the input signal.

Expected plot for a gain-of-2 configuration input amplifier.



Final specifications:

The circuit met the target requirements, converging to the correct sampled peak value for a given amplitude input signal. Altering the input signal amplitude yields a different sampled peak value, as desired.