Preparatory exercises inside!
Do NOT show up to the lab unprepared!!!

TSEK37 ANALOG CMOS INTEGRATED CIRCUITS

-

Analog Lab

Date and signature of student

Date and signature of the lab assistant

Martin Nielsen-Lönn
November 2016

Ali Fazli & Dai Zhang & Daniel Svärd
September 2012

Purpose of this exercise

During this laboratory exercise you will learn how to build and simulate a simple operational amplifier (OP). OP-amplifiers are very important building blocks in analog circuit design. They can for instance be used in analog filter, bias generation, analog buffer, etc.

The goal of this laboratory exercise is to give you experience of designing a two-pole system and considering stability issues in such systems. Furthermore, you will get hands-on experience with powerful circuit design tools commonly used in industry. Finally the laboratory exercise will give you a glance of all the parameters, and aspects with regard to design for manufacturability (DFM).

The laboratory exercise is divided into four main parts. First you will use the simplified transistor models in the course book to derive the relationship between transistor parameters and the most important performance parameters. **THIS IS DONE IN THE PREPARATORY EXERCISES, AND SHOULD BE DONE PRIOR TO THE FIRST LAB OCCASION!** When you arrive at the lab you will firstly become acquainted with the tools by entering the schematic of the OP-amplifier. The next step is to size the transistors to meet certain performance specifications. This constitutes the second part of the laboratory exercise, which is recommended to complete by the end of the first lab occasion. The third part of the laboratory exercise is to evaluate the designed OP-amplifier when the impact of process, voltage, and temperature variations (PVT) are taken into account. The fourth and final part of the laboratory exercise is to evaluate the final OP-amplifier design in a feedback configuration, and to show the influence of stability and feedback.

Figure 1 shows the two-stage OP-amplifier, which you will analyze in this laboratory exercise. The first stage consists of a differential pair, which drives the output stage. To provide stability compensation a feedback path in the form of R_z and C_c is inserted. A more detailed description of the circuit is found in the main course book *"Design of Analog CMOS Integrated Circuits"*, Behzad Razavi. See preparatory exercises.

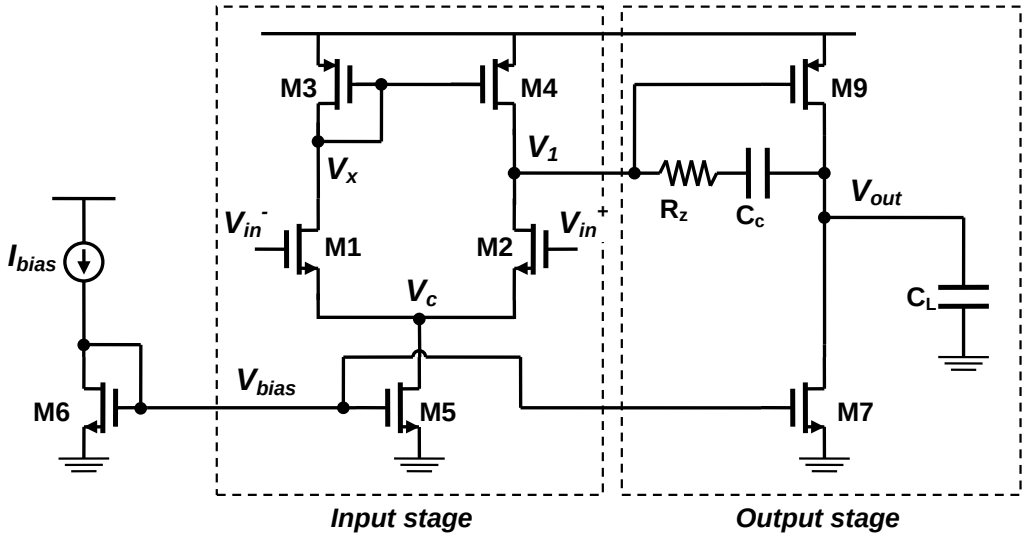

Figure 1 - Two-stage OP-amplifier.

Part I -
Background theory and preparation for laboratory exercise

1 Preparatory exercises

1) In order to be well prepared for this laboratory exercise, read through sections 4.1, 4.2, 5.3, 6.2, and 10.5 in the main course book *"Design of Analog CMOS Integrated Circuits"*, Behzad Razavi.
2) Read through the laboratory manual.
3) Complete the seven following preparatory exercises before the laboratory exercise!
   NOTE: It is MANDATORY to do the exercises BEFORE the laboratory!

1.1 Small signal modeling

Figure 2 shows the simplified small-signal model of a MOS transistor, which you will use in the preparatory exercise.

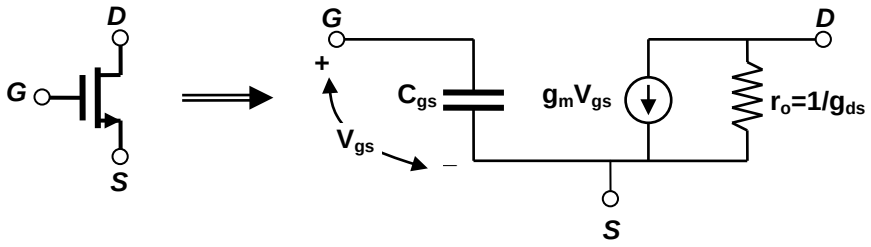

Figure 2 - Small signal model of a MOS transistor.

From classic long-channel transistor theory the transistor drain-source current in saturation is approximately $I_D = \frac{1}{2}\mu C_{ox}\frac{W}{L}\left(V_{gs} - V_{TH}\right)^2(1 + \lambda V_{ds})$, see eq. (2.26) page 25 in the course book. In saturation the gate-source capacitance (C_{gs}) can be approximated with the following expression (see page 32 in the course book):

$$C_{gs} \approx \frac{2}{3}WLC_{ox} \tag{1}$$

Exercise 1

The definition of g_m and g_{ds} is given by equations (2) and (3), respectively.

$$g_m = \left.\frac{\partial I_D}{\partial V_{gs}}\right|_{V_{DS,const}} \tag{2}$$

$$g_{ds} = \frac{\partial I_D}{\partial V_{ds}} \tag{3}$$

From the approximation of I_D derive the small signal parameters g_m and g_{ds} as a function of I_D, W, and L (not V_{gs}):

(4)

(5)

Exercise 2

The differential input to the input stage is defined as: $V_{in,diff} = V_{in}^{+} - V_{in}^{-}$.

Make a small-signal model of the input-stage **neglecting all capacitors in the transistors**. Assume a symmetrical bias point, which means that $I_{D1} = I_{D2} = I_{D3} = I_{D4}$, $g_{m2} = g_{m1}, g_{m4} = g_{m3}, g_{ds2} = g_{ds1}$ and $g_{m3} \gg \left({}^{1}/_{r_{o1}} + {}^{1}/_{r_{o3}}\right)$.

Derive the transfer function for input stage $A_1 = \frac{V_1}{V_{in,diff}}$ as a function of g_m and g_{ds}.

(6)

When designing the stage you will change the bias currents, widths, and lengths of the transistors. From the expression for A_1 above derive the gain as a function of $I_{D,n}$, W_n, and L_n (not V_{gs}).

(7)

Exercise 3

From basic circuit theory we know that any linear circuit network containing passive components, current sources, and voltage sources can be described with either a Norton equivalent or a Thévenin equivalent as in Figure 3.

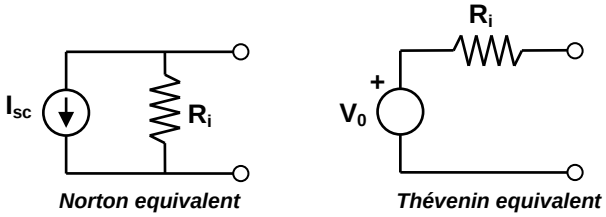

Figure 3 –Norton and Thévenin equivalent circuit models.

The current and voltage sources in Figure 3 are related by $V_0 = I_{sc}R_i$.

Examine the expression in (6). Derive the Thévenin equivalent circuit for the first amplifier stage. Compute the expressions for V_0 and R_i as a function of V_{in}, $g_{m,n}$, and $g_{ds,n}$.

(8)

(9)

Exercise 4

In this exercise you will analyze the complete two-stage amplifier circuit both for amplification and dynamic behavior. Figure 4 shows the simplified model of the two-stage amplifier using the Thévenin equivalent circuit derived in Exercise 3.

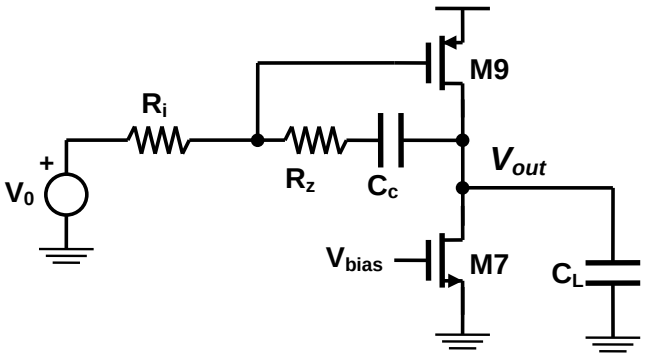

Figure 4 - Second stage driven by the Thévenin equivalent of the first stage.

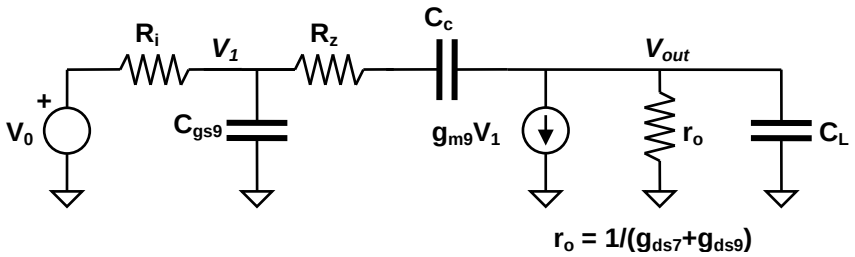

Figure 5 - Small-signal model of the schematic in Figure 4.

Derive the expression for the total transfer function of the small-signal model in Figure 5, with $R_z = 0$. Look in section 10.5 in the course book for the motivation for this simplification.

(10)

In general a two pole system can be expressed as:

$$A(s) = \frac{A_0\left(1-\frac{s}{\omega_z}\right)}{\left(\frac{s}{\omega_{p1}}+1\right)\left(\frac{s}{\omega_{p2}}+1\right)} = \frac{A_0\left(1-\frac{s}{\omega_z}\right)}{\frac{s^2}{\omega_{p1}\omega_{p2}}+\left(\frac{1}{\omega_{p1}}+\frac{1}{\omega_{p2}}\right)s+1} \qquad (11)$$

To be able to stabilize a two pole-system it is required that $\omega_{p1} << \omega_{p2}$. This gives that equation (11) can approximated with:

$$A(s) = \frac{A_{20}\left(1 - \frac{s}{\omega_z}\right)}{\frac{s^2}{\omega_{p1}\omega_{p2}} + \frac{s}{\omega_{p1}} + 1} \tag{12}$$

Use equation (12) to identify A_2, ω_{p1}, and ω_{p2} (**Hint:** Approximate as much as possible according to section 10.5 in the text book):

(13)

(14)

(15)

Exercise 5

Derive the total DC-gain of the two-stage amplifier.

(16)

Exercise 6

Section 10.5 in the course book discusses the stability issues for the amplifier when $R_z = 0$. The numerator of the transfer function reads $(1 - s/\omega_z)$, yielding a phase of $-\tan^{-1}(\omega/\omega_z)$, a negative value because ω_z is positive $\omega_z = g_{m9} / C_C$. As a result, the stability degrades considerably.

To ensure stability we need to push the zero created by the compensation capacitor (C_c) from the right half plane to the left half plane. We can achieve this by introducing R_z. Find the approximate expression for the location of the zero when $R_z \neq 0$ by referring to section 10.5 in the course book or derive the expression yourself.

(17)

To make sure that the zero does not degrade but improve the stability of the system, it should be first in the left half plane $(\omega_z \leq 0)$, and furthermore it should be placed at a frequency which satisfies $|\omega_z| \leq |\omega_{p2}|$, hence it compensates the phase before the arrival of the second pole. Derive a criterion for R_z to ensure this.

(18)

Exercise 7

An important property of an OP-amplifier is the unity-gain frequency (ω_u), which is defined as the frequency where the gain reaches unity according to:

$$|A(s = j\omega_u)| = 1 \tag{19}$$

Section 10.1-3 in the course text book gives a short summary of feedback and stability, which could be helpful for this exercise and in the remaining part of the lab.

Figure 6 shows a two-pole system with a positive phase margin, which is a requirement for a stable system. The position of the unity-gain frequency in relation to the second pole (ω_{p2}) has a large impact on the stability and behavior of the system. If ω_u is larger than ω_{p2} the phase-margin will be less than $45°$, which will cause ringing. In order to have a well behaved system the phase-margin is usually required to be larger than $60°$, which further requires that ω_u is less than ω_{p2}.

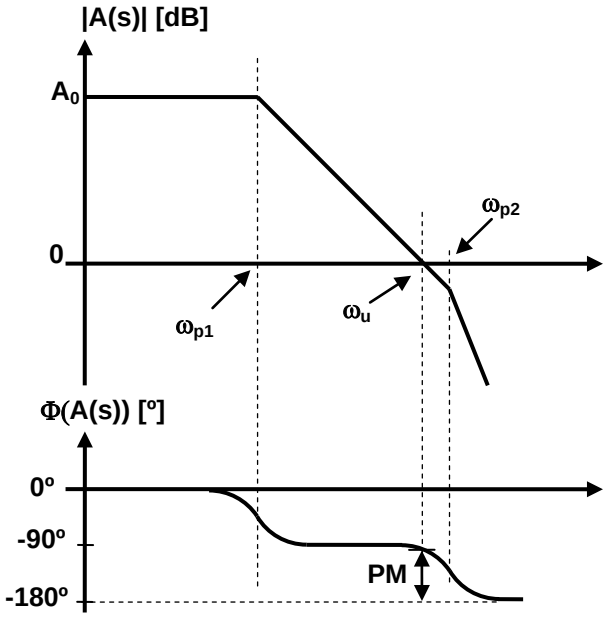

Figure 6 - Bode-plot of a stable two-pole system.

For the type of system shown in Figure 6 the second pole (ω_{p2}) will not impact the amplitude of the transfer-function *A(s)* for frequencies less than ω_u. The unity-gain frequency can therefore be approximated using a one-pole system according to (20).

$$A(s) = \frac{A_0}{\frac{s}{\omega_{p1}} + 1} \tag{20}$$

Use this approximation to derive the expression for ω_u as a function of $g_{m,n}$, $g_{ds,n}$, and C_n.

(21)

The preparatory exercises are completed. Go to the first lab occasion and request your lab assistant to check and approve your solutions.

Part I is completed.

Date and signature of the lab assistant

Part II - Design and simulation of OP-amplifier

2 The simulation tool – Cadence

The software that we are going to use is called Cadence Virtuoso 6.1.6. This is a very powerful and versatile tool commonly used in industry for development of (analog) integrated circuits.

All constructions in Cadence are organized in libraries. *Reference libraries* include basic building blocks such as transistors, resistors, capacitors etc. *Design libraries* include your own designs in which you have instantiated components from the reference libraries. The "Library Manager", which pops up when you start Cadence, shows a tree structure with the names of all the libraries available. You can add or remove libraries from the "Library Manager". Each library includes cells and their different views. A cell is for example an inverter and the view is a schematic, symbol or layout presentation of the cell.

The simulation environment is called "*Analog Design Environment XL"* and is invoked from the menus in the schematic you want to simulate. All the supply and signal sources are inserted directly into the schematic. In this way we save a lot of work compared to writing all the netlists manually.

The results from a simulation are presented as waveforms in a separate window. Post processing of the signals can be made and plotted/printed by the *Calculator* tool that is available.

3 Initialization of the Lab

To complete this lab successfully you have to setup the paths for Cadence and the design-kit in the correct way. The first thing needed is a setup-file that specifies which tools and paths you will use. This setup-file is called "module" and must be loaded every time you want to start Cadence.

The startup procedure for Cadence is:

- Change to the ixtab server by typing:
  `ssh ixtab`
- Load the setup-file by typing:
  `module load TSEK37`
- Go to the directory "TSEK37/lab" by typing:
  `cd ~/TSEK37/lab`
- Start Cadence by typing:
  `ams_cds`

4 Getting Started

4.1 Create a new Design Library

The first thing you see is the window namned “Virtuoso 6.15”, this window is refered to as CIW (Command Interpreter Window). Here you can see different log messages and start other tools.

You are going to save your work in a Design Library. To create a new Design Library, go to the window called “Library Manager”*,* which might pop up when you start Cadence. If it doesn’t appear click Tools > Library Manager in the CIW.

Choose:

File > New > Library...

A window called “New Library” will pop up. In the “Name” field type “OPAmp”, which is the name of your new Design Library. Then click OK, which will open a new window called “Technology File for New Library” in which you choose “Attach to an existing technology library” and press OK. Yet another window will appear named “Attach Design Library to Technology File”. Choose technology library TECH_C35B4 and press OK.

Now you will enter the schematic of the OP-amplifier. To create the OP-amplifier cell in the *OPAmp* library, go to the “Library Manager” and click *OPAmp* in the Library column. Choose:

File > New > Cell View...

A window called “Create New File” pops up. Some times it appears underneath the some other window, check your activity field. In the *Cell Name* field, enter *OPAmp* as the name of the new cell. Check that the *Library name* is *OPAmp*, *View Name* is *schematic*, and *Tool* is *Schematics L* then press **OK**. A window might appear asking if you want to check out another licence, just click yes. An empty schematic window appears. You will now draw the OP-amplifier schematic in this window.

5 Enter the OPAmp schematic

A schematic consists of components, input and output terminals, and their interconnections. You will now learn how to draw a complete OP-amplifier schematic in Cadence. When the OP-amplifier is completed it should look like Figure 7.

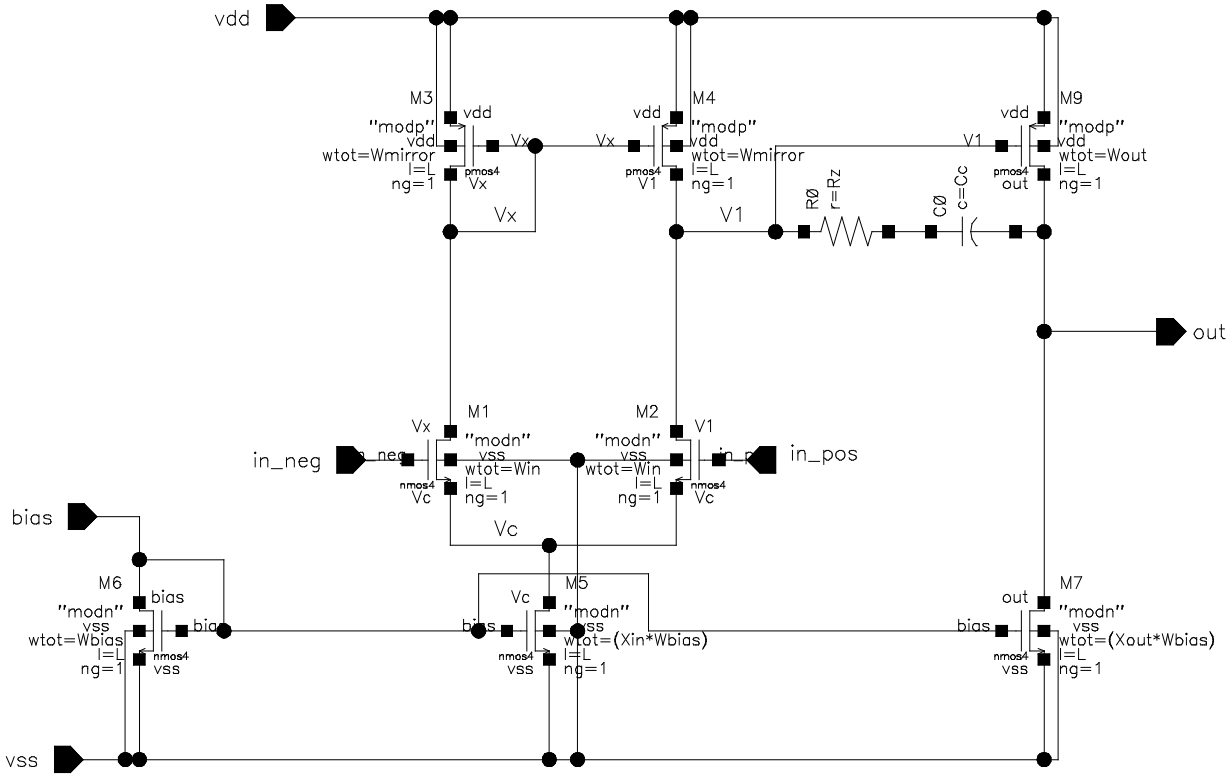

Figure 7 - Cadence schematic of the *OPAmp* cell

5.1 Add instances of components

To add a NMOS transistor to your schematic do:

Create > Instance [i]

Fill in the “Add Instance“ form by using the **Browse** button.

Library: PRIMLIB

Cell: nmos4

View: symbol

Place the transistor by clicking the left mouse button (*lmb*) when the instance is at the desired location in the schematic. An identical instance appears and can be placed in the same manner.

Before you add the device, the orientation can be changed by the buttons **Rotate**, **Sideways**, and **Upside Down** in the “Add Instance“ form. Exit the add transistor mode by pressing [Esc]. To add a PMOS device, follow the above instructions but in the “Add Instance“ form fill in

Cell: pmos4

The compensation feedback requires one resistance and one capacitor. These components can be found in the library “analogLib”. The cell names are *res* and *cap*, respectively.

5.2 Moving and deleting instances

You can move instances around or change their rotation by using the command:

Edit > Move [M]

and then selecting the object to be moved in the schematic. This will disconnect the instance from all attached wires in the schematic and it will follow the cursor like it did when it was initially placed. By pressing [F3] a small form appears from which the orientation can be changed. When you are finished using the command *Move,* press [Esc].

NOTE: You can always exit any command by pressing [Esc].

The similar command:

Edit > Stretch [m]

can be used to move an instance without disconnecting it from attached wires in the schematic.

A highlighted instance can be removed by:

Edit > Delete

or by pressing the [Delete] button on the keyboard. Deleted instances can be brought back again with:

Edit > Undo [u]

5.3 Add pins

You will need six pins (*in_pos, in_neg, bias, out, vdd, vss*) in your OP-amplifier schematic. Pins connect signals through schematic hierarchies and are necessary for simulation. To add the *in_pos* pin do:

Create > Pin [p]

Fill in the “Add Pin” form

Pin Names: in_pos

Direction: input

and press [enter]. Place the pin by clicking the left mouse button (*lmb*) when it has been moved to the desired location in the schematic. You can change the orientation of the pin before you add it by pressing [F3] and using the buttons **Rotate**, **Sideways** and **Upside Down** in the “Add Pin” form. It is possible to enter several Pin Names at the same time; just separate the individual names with a [space]. The Direction of the pins should be as follows: *in_pos(input)*, *in_neg(input)*, *bias(input)*, *out(output)*, *vdd(input)*, *vss(input)*. Close the “Add Pin” form by pressing the **Cancel** button.

5.4 Add wires

Devices and pins are connected by wires. To add a wire between two points do:

Create > Wire (narrow) [w]

Click with the *lmb* at the starting point. By clicking *lmb* on a point you can change the direction of the wire. Move the cursor and click with the *lmb* at the ending point. You can end the wire in "thin air" by clicking twice at the same position. Exit the Add Wire mode by pressing [Esc].

Finish the *OPAmp* cell according to Figure 7. Do not forget to connect the bulk of each NMOS and PMOS transistor to the *vss* and *vdd* pin respectively!

We have to name wires carrying signals we would like to plot during simulation. A wire can be named by:

Create > Wire Name... [l]

A window called "Add Wire Name" pops up. Enter the wire name in the *Names* field and press [enter] (multiple names are separated by [space]). Place the wire names *Vc*, *Vx*, and *V1* on the appropriate wires using the *lmb* (see Figure 7). In this dialog you can also place multiple labels by separating their names with a space.

5.5 Enter simulation variables

By using variables as component property parameters, instead of specific values, it is possible to directly change the values from the simulator. Otherwise the schematic would have to be modified and saved before the new parameters are visible to the simulator. Moreover we can modify the default instance names given by Cadence to a name we chose. To enter proper component property values for the transistor M1, select it with [*lmb*] and do:

Edit > Properties > Objects... [q]

Fill in the "Edit Object Properties" form (*M* is the unit – meters, see appendix A):

Parameter	Value
Instance Name	M1
Width	Win M
Width Stripe	Win/1 M
Length	L M
Number of gates	1

and press **OK**. Edit the properties for each transistor (M1-M9 in Figure 7) according to the instructions above. Use the parameter values given in Table 1.

Table 1 Transistor parameter values.

Component	Instance name	Width=WidthStripe	Length	No. of gates
M1	M1	Win	L	1
M2	M2	Win	L	1
M3	M3	Wmirror	L	1
M4	M4	Wmirror	L	1
M5	M5	(Xin*Wbias)	L	1
M6	M6	Wbias	L	1
M7	M7	(Xout*Wbias)	L	1
M9	M9	Wout	L	1

Edit the property values for the resistance and capacitance according to Table 2.

Table 2 Resistance and capacitance parameter values.

Component	Instance name	Parameter
Rf	R0	Resistance = Rz
Cc	C0	Capacitance = Cc

5.6 Verify and save your design

When the schematic is completed it must be checked for errors. Start the automatic check program by:

File > Check and Save [Shift+X]

The check program will detect problems such as floating wires or terminals, short circuits, etc. It is good to get an habit of doing a *Check and Save* from time to time to see if you have an errors left.

If a symbol view of the cell also exists, any mismatch between the schematic and the symbol will be reported. You can examine the errors and warnings by:

Check > Find Markers [g]

which will bring up a list of the errors/warnings and what caused them. Remember to check and save your design again if you corrected any errors. If your cell is free from errors, it will be saved into the library.

5.7 Create a OPAmp symbol

Since the *OPAmp* schematic will be used as an instance in the *testbench* schematic we must create a corresponding symbol view for it. The form for symbol generation is brought up by:

Create > Cellview > From Cellview

Check that the field *To View Name* reads *schematicSymbol* and press **OK**. A second form called "Symbol Generation Options" appears in which you can change the location of the pins.

Press **OK** to start up the symbol editor. The outer red rectangle has the pins on the border and contains a green rectangle that will be shown when the symbol is used in a schematic. The red rectangle is the bounding box which, among other things, determine in which area you select the component.

Make sure the symbol view is saved (*File > Check and Save*) before you close down its window. Also close down the *OPAmp* schematic window before you proceed.

6 Enter the testbench schematic

In the previous section you saw most menu options that are regularly used in Cadence when drawing schematics. Create a cell called *OPAmp_testbench* in your *OPAmp* library by using the "Library Manager".

6.1 Place components

We will now enter the *OPAmp_testbench* schematic according to Figure 8. The *OPAmp* cell symbol is located in your Design Library *OPAmp* - use the **Browse** button in the "Add Instance" form. All remaining component instances can be found in *analogLib* library.

Add a ground symbol (*gnd* in *analogLib*) in the testbench schematic according to Figure 8.

Add a sine source (called *vsin* in *analogLib*) and change the property values to the following:

Parameter	Value
Instance Name	Vinp
DC voltage	VinDC V
AC magnitude	(A/2) V
AC phase	phi_pos
Offset voltage	VinDC V
Amplitude	At V
Initial phase for sinusoid	phi_pos
Frequency	F Hz

Create a copy of the first sine source (*Vinp*) by doing

Edit > Copy [c]

and replace all *phi_pos* variable names in the copy with *phi_neg* and change the instance name to *Vinn*.

Now add a DC voltage source (called *vdc* in *analogLib*) with the following property variables:

Parameter	Value
DC voltage	Vdd
AC magnitude	0
AC phase	0

Create a DC current source (called *idc* in *analogLib*) with the property variable
`DC current: Ibias A`

Finally create a capacitance (called *cap* in *analogLib*) with the property variable
`Capacitance: Cl F`

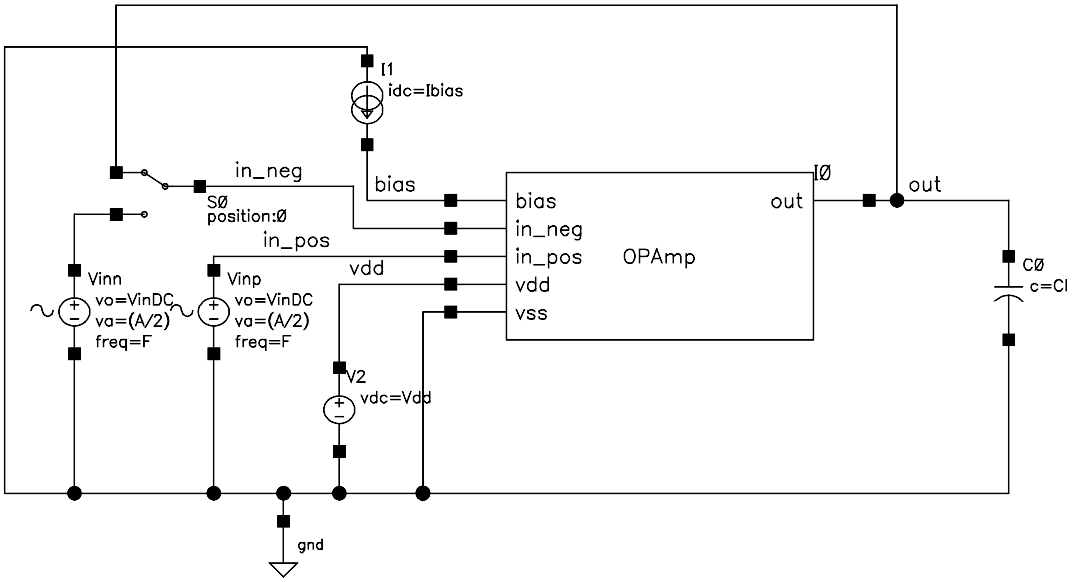

Figure 8 - Cadence schematic of the *OPAmp_testbench* cell.

6.2 DC-point trick

The two stage amplifier provides a very high open-loop gain, which when we simulate will cause slightest offset between the differential inputs to saturate the output. Hence, the DC-point evaluated under such conditions will not be a good linearization point for the following AC-analysis (small-signal).

To surmount this problem we can set the OP-amplifier in a negative feedback configuration during the calculation of the DC-point. When this point is found we break the feedback and make a small-signal analysis using that DC-point, in an open-loop configuration.

A component which can solve this for us is the switch *sp2tswitch* in *analogLib*. This switch can be configured so that different switch conditions are used in the different analysis options. Position “0” means that the output is floating, position “1” means that the top input is connected to the output, and position “2” means that the bottom input is connected to the output.

Add a *sp2tswitch* from *analogLib* and connect it between the *plus* node of *Vinn*, *in_neg* pin on the *OPAmp* symbol, and the output *out* according to Figure 8. Open the property window for the *sp2tswitch* and change according to:

Parameter	Value
Switch position	0
DC position	1
AC position	2
Tran position	1
IC position	1

Switch position tells you under certain analysis which switch is on. During DC and transient analysis, the top switch is on by setting the position as “1”. During AC analysis, the bottom switch is on by setting the position as “2”.

6.3 Connecting it all together

Connect all remaining sources and loads together with the OP-amplifier block according to the schematic in Figure 8. Finally add wire names *bias, in_pos, in_neg* and *out* on the appropriate wires. The testbench schematic should now look like in Figure 8.

Check and save your *OPAmp_testbench* cell.

6.4 Moving in the Design Hierarchy

You can view and edit the content of the *OPAmp* symbol by selecting it and choosing

Edit > Hierarchy > Descend Edit... [Shift+E]

and clicking **OK** in the pop-up window that appears. To return up to the *OPAmp_testbench* schematic do

Edit > Hierarchy > Return... [Ctrl+e]

7 Simulation setup

Start the simulation environment from the *OPAmp_testbench* schematic editor by the command:

Launch > ADE XL

A dialog box named “Launch ADE (G)XL” will appear, select “Create New View” and press “OK”. If a dialog called “Create new ADE (G)XL view” appears make sure that the view field is equal to “adexl” and then click “OK”.

The “Virtuoso Analog Design Environment XL” (ADE XL) window appears which has two panes. On the left a “Data View” pane shows your tests, global parameters and corners and on the right a pane with outputs and results in different tabs. The design is also shown in a tab in the right pane.

7.1 Import component property variables

First all component property variables from the *OPAmp_testbench* schematic has to be added. Do this by right clicking (RMB) on “Global Variables” in the left pane and click “Add Variable”. Add all variables shown in Table 3 with their respective value.

Table 3 Testbench variables

Variable	Value
Vdd	3.3
phi_pos	0
phi_neg	180
L	700n
F	10M
VinDC	1.65
Cl	3p
At	0.5
A	2

These values will be used later in simulation together with values on transistor widths and the bias current specified in section 8.

7.2 Analysis setup

For the initial design work we only consider the small-signal behavior of the amplifier. To setup a small-signal simulation expand the Tests item in the left pane and click “Click to add test”. This will open the “ADE XL Test Editor” with a dialog box called “Choosing Design” over it. In the “Choosing Design” box select the “OPAmp_testbench” and make sure “View Name” is set to “schematic” then press “OK”. Then select:

Analyses > Choose.

Press *Analysis: DC* and mark the two buttons *Save DC Operating Point* and *Enabled*. Save by pressing the *Apply* button. Now click *Analysis: AC* and set

Sweep Variable: Frequency

Sweep Range: Start – Stop

Start: 100

Stop: 1G

Press *Apply* and *OK*. Then you can close the "ADE XL Test Editor" window.

NOTE: A brief summary of the different analyses available is presented in Appendix A.

7.3 Specify output signals to be plotted

To add signals that should be plotted right click (RMB) in the right pane under the tab "Outputs Setup" and select "Add Signal", then click in the field under "Details" and press the button on the right (…). This opens the schematic where you click on the signal you want to plot. Select the *out* wire. In the "Outputs Setup" table the signal has been added as "/out", for the other outputs you can right click the previsously saved output and select "To be plotted…". Now select the *bias*, *in_pos*, and *in_neg* wires and then press [Esc] to exit the selection mode. Go back to the "adexl" tab so see your outputs, the wire names you clicked should now appear in the "Outputs Setup" tab.

7.4 Specify output signals to be saved

By default all voltages in the circuit schematic is saved by the simulator. However, none of the currents are saved. We are interested to know how much current the amplifier uses and therefore we need to specify that we want to save it.

To add signals to be saved right click (RMB) on an existing output in the "Outputs Setup" tab and select "To be saved…".

In your *OPAmp_testbench* schematic click on the *vdd* node of the OPAmp cell to select the current through the *vdd* node to be saved. There should be a yellow circle around the selected node, and a positive current is defined as floating into the circle. Press [Esc] to cancel the selection and then return to the "adexl" tab.

7.5 Enter gain and phase functions using the calculator

We are also interested in plotting the gain (as function of frequency) of the differential stage.

The gain is an analytical expression, which the simulator has to evaluate. Expressions are entered in the waveform calculator. Start the calculator from the ADE window by:

Tools > Calculator

Press **vf** (frequency voltage) on the calculator followed by selecting the wire (voltage) called *out* in the schematic window to make *VF("/out")* pop up in the calculator window. Mathematical operations can be performed on the entered data by the buttons on the calculator; division, addition, subtraction, etc. Enter the expression for the gain:

VF("/out")/(VF("/in_pos")-VF("/in_neg"))

We want the magnitude of the gain in dB, which can be calculated by clicking on *dB20* in the Function Panel of the calculator. If you cannot see *dB20* select *All* in the drop down menu over the functions. The expression should now be:

```
db20(VF("/out")/(VF("/in_pos")-VF("/in_neg")))
```

To import the gain expression to the outputs, go to the ADE window and right click (RMB) in the "Outputs Setup" table and select "Add Expression". Click in the "name" field of the newly created expression and enter *Gain*, then double click (LMB) and press the button on the right with an "<" symbol.

We also want to plot the phase of the transfer function in order to determine phase-margin for the circuit. This can be done in a similar manner by once again entering the gain expression in the calculator according to above, and then select *phase*. The expression should now be:

```
phase(VF("/out")/(VF("/in_pos")-VF("/in_neg")))
```

Import the phase expression to the "Outputs Setup" in the same way as for the *Gain* but with *Phase* as the name.

7.6 Entering expressions for DC-gain, ω_u, PM, and current

You can also use the expressions for the gain and phase to find the performance metrics for the amplifier. The DC-gain is the magnitude of the gain in dB at 0 frequencies. Since we already have an expression for the gain we can use it in this expression. This can be found with the expression:

```
value(Gain,100)
```

This gives the gain at 100 Hz, which is identical to the DC-gain, but for simulation reasons we use 100 Hz as the start frequency of the AC-analysis instead of 0. Add the expression above and name it *DCgain*.

To find the unity-gain frequency we need to find the frequency where the gain crosses 1 (= 0 dB). This can be done by the following expression:

```
cross(Gain,0,1,"falling")
```

Add this expression and name it *UGB*.

The phase-margin is *180°* + the value of the phase at the unity-gain frequency, which can be found with the following expression:

```
phaseMargin(VF("/out")/(VF("/in_pos")-VF("/in_neg")))
```

Enter this expression and call it *PM*. Here we cannot use the Gain expression since the phaseMargin-function only works for linear gain expressions.

The current drawn from the power supply is needed in order to determine the total power dissipation of the amplifier. You can find the DC current drawn from the power supply by the amplifier with the expression:

```
IDC("/I0/vdd")
```

Enter this expression and name it *SupplyCurrent*.
NOTE: “I0” is the instance name of your amplifier and “vdd” refers to the pin of the actual instance.

Also add and expression for the power dissipation in the amplifier called *Power* with the expression and make sure that the node is saved during simulation (section 8.3):

VAR(“Vdd”)*SupplyCurrent

Here we use the VAR-function which includes the design variable *Vdd* into the expression. This is practical if we would like to change the Vdd later.

In the preparatory excercices you derived a requirement for the minimum value of *Rz*, create a new expression called *Rz_min* and enter your derived expression.

7.7 Expressions for pole/zero approximations

From the preparatory exercises we have obtained approximations of the poles and zero locations. We can also add these in order to help in the design of the amplifier. The expression all includes small-signal parameters such as g_m, which can be found by using the OP function in Cadence. The OP function extracts small-signal transistor parameters such as g_m, g_{ds}, and c_{gs} (called *gm*, *gds*, and *cgs* in Cadence) after a simulation run. This function can also be used to extract resistor and capacitance values (called *res* and *cap* in Cadence). You can utilize this to create the expression of the pole/zero approximations. For the zero the approximate expression should look like (note the spaces between the quotation marks):

(1/(2*pi*OP("/I0/C0" "cap")*((1/OP("/I0/M9" "gm"))-OP("/I0/R0" "res"))))

Add the expressions for the zero and the second pole and name them *fz* and *fp2*, respectively. The two expressions for *Ri* and *ro* also has to be added.

NOTE: Some of the OP parameters (read cgs) can be negative. To get a positive value use the function “abs”.

The expressions can be found in the file /site/edu/eks/TSEK37/expressions.txt

8 Design the differential amplifier to meet the specification

Exercise

Design the differential amplifier stage so that it fulfills the specification in Table 3 below. A suitable design strategy with hints is presented in section 8.1. To have time to do all the simulations, do NOT spend too much time trimming the OP-amplifier.

Table 4 Nominal target specification of the OP-amplifier.

Variable name	Value	Comment
V_{dd}	3.3 V	-
Output DC-level	$V_{dd}/2$ = 1.65 V	-
C_L	3 pF	Due to driving capability requirements, cannot be changed
L	≥ 0.6 µm	L ≈ 2xL_{min} reduces the effects of channel length modulation
W_{max}	300 µm	From area requirements
W_{min}	10 µm	-
$I_{bias,max}$	1 mA	-
$P_{tot,max}$	≤ 25 mW	-
A_0 (DC-gain)	≥ 60 dB	-
f_u (Unity-gain frequency)	≥ 200 MHz	-
Phase-margin	≥ 70°	-

8.1 Design strategy

Limiting output offset-voltage

An approximate way to get an output DC-voltage level of $V_{dd}/2$ is to set the current through both transistors in the output stage (M7 and M9) equal.

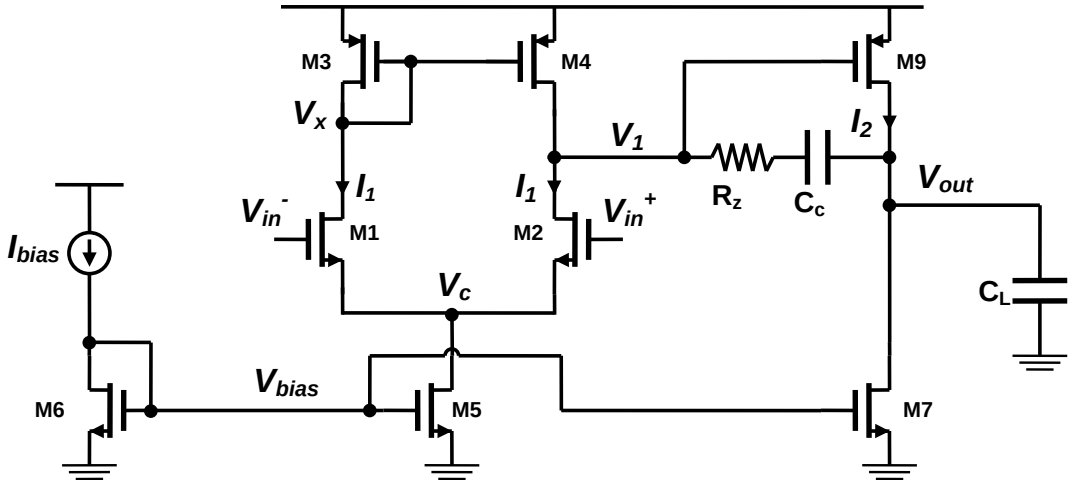

Figure 9 - OP-amplifier at $V_{in}^+ = V_{in}^-$.

Figure 9 shows the OP-amplifier at symmetrical bias points. Both inputs to the OP-amplifier have the same voltage and the current through both M3 and M4 is therefore equal (I_1). The current I_1 is given by:

$$I_1 = \frac{I_{bias}W_5}{2W_6} = \frac{I_{bias}X_{in}}{2} \tag{22}$$

where ratio between the widths of transistor M5 and M6 has been defined as X_{in}. The current through transistor M7 is given by:

$$I_2 = I_{bias}\frac{W_7}{W_6} = I_{bias}X_{out} \tag{23}$$

where X_{out} is defined as the width ratio between M7 and M6.

Further, because of the symmetry V_x=V_1, hence the relation between the width of M9 and M3 is given by:

$$\frac{W_3}{I_1} = \frac{W_9}{I_2}. \tag{24}$$

If we combine equations (22), (23), and (24) we can reduce the complexity of the design work by eliminating W_{mirror}. We thereby get an expression for the width of M3 and M4 (W_{mirror}) according to:

$$W_{mirror} = \left\{W_3 = W_4 = \frac{W_9X_{in}}{2X_{out}} = \frac{W_{out}X_{in}}{2X_{out}}\right\}. \tag{25}$$

To automatically keep the ratio in (25) we can add this expression for *Wmirror* in the "Editing Design Variables" window. Click on *Wmirror* in the "Design Variables" field and update the *Value (Expr)* field to *(Wout*Xin/(2*Xout))*.

Initial values

You can use any values that fulfill the area and power requirements in Table 3. To give you some idea of a starting point you can use the parameter values (see appendix A "Design variables"):

Parameter	Value
$W_{in} = W_{bias} = W_{out}$	10 µm
I_{bias}	100 µA
$X_{in} = X_{out}$	1
C_c	100 fF
R_z	0 Ω

Save and load simulation states

The simulation setup including variables, analysis, and outputs can be saved in a state. To do this select:

File > Save

To open your simulation setup again, open the “adexl” view from the library manager.

Simulation tips

Instead of changing one parameter value and simulate, we can run a set of simulations for a range of parameter values by entering a range for a variable as “start:step:stop” for example “20u:20u:80u” will run 4 simulations with 20u, 40u, 60u and 80u. We strongly recommend that you use this function for the design.

Note: Additional simulation and plotting tips can be found in the Appendix A.

Specifications

In ADE XL you can set specifications for outputs. Switch to the “Outputs Setup” pane and locate your *DCgain* expression. On that line you will find a column called Spec, double click (LMB) on it and select “>” from the drop down list. Then write “60” in the box next to the “>” sign. When the simulation is complete the expressions fullfilling the specification is shown in green, if is is close it is yellow and specifications that is not fullfilled are shown in red.

Add specifications for *Dcgain*, *PM*, *UGB* and *Power*.

Run simulation and view results

When you have set the variables you want start the simulation by clicking the green play button just over the right pane. The right pane will then change to the “Results” tab where you can see the status of each simulation, if you right click (RMB) a simulation and select “Output Log…” you will see how the simulation progress.

When the simulation is complete you will see the result in the table. To plot something you can right click (RMB) that line and select “Plot” or hold down [Ctrl] and select multiple outputs. If you want to plot everything press the small red/blue graph button below the “Results” tab.

Move the second pole

In the derivation of expression (21) for the unity-gain frequency we have assumed that the second pole (ω_{p2}) and the zero (ω_z) are located high enough in frequency not to impact the unity-gain frequency (ω_u). This is also a requirement to obtain a reasonable phase-margin later on in the design step.

The first design step should therefore be to move the second pole up in frequency to at least twice the desired unity-gain frequency. Equation (4) and (15) should give you a hint on how to proceed with this.

Move the zero

For the same reason we want to push the zero higher up in frequency, so the second design stage is to do this. Refer to equation (4) and (17) for hints on how to do this.

Move unity-gain frequency

If the initial unity-gain frequency does not fulfill the specification according to Table 3 you of course need to improve it. The approximation in equation (21) should help you to do this.

Obtain acceptable DC-gain

The DC-gain should according to specification be higher than 60dB and if this is not the case it will need to be improved. Refer to the small-signal DC-gain expression in (16) and the approximations of g_m and g_{ds} in (4) and (5).

Obtain acceptable phase-margin

If the phase-margin is less than 70° it will need to be improved. Refer to Exercise 7 in Part I for hints on how to obtain a better phase-margin.

Iterate

It is not unlikely that when you try to improve for instance the unity-gain frequency you will reduce the phase-margin. Hence, an iterative approach for the design is needed. Iterate the steps above (from “Move the second pole”) until the OP-amplifier meets all the performance requirements in Table 3.

8.2 Final amplifier

When the amplifier meets all of the performance requirements in Table 3 fill in the following table:

Table 5 Results of your OP-amplifier design.

Design parameter	Value	Performance metric	Value
W_{in} (µm)		P_{tot} (mW)	
W_{out} (µm)		I_{bias} (µA)	

W_{bias} (µm)	DC-gain (dB)
W_{mirror} (µm)	Phase-margin (°)
X_{in} *W_{bias} (µm)	Unity-gain frequency (MHz)
X_{out} *W_{bias} (µm)	Appr. 2nd pole (MHz)
L (µm)	Appr. zero (MHz)
R_z (Ω)	
C_c (pF)	
C_L (pF)	

Part II is completed.

Date and signature of the lab assistant

Part III -
Additional performance metrics and impact of process, temperature and voltage variations

The amplifier you designed in part II now meets the specifications in Table 3 (congratulations!!!). However, it does so only for one DC-point and for a single process corner. We are interested to see how this specification holds when we introduce different voltage levels on the inputs and output, change the operational temperature, vary the power supply voltage, and change the process parameters.

9 Input/output range

In Part II we have only linearized around one single DC-point ($V_{in}^{+} = V_{in}^{-} = V_{out} = 1.65\ V$), and we used that to find all the small-signal specifications. If we linearize around a different DC-point we will get different small-signal performance metrics. The range of input and output voltages where the specification still is obtained is defined as the input and output range. This range will be rather small if we strictly use the specification in Table 3, so in order to obtain a reasonable input-output range we allow w_u, A_0, and *PM* to degrade by 5 % compared to specifications in Table 3.

9.1 Modify the testbench

In the *OPAmp_testbench* cell add an instance of a *vdc* symbol from *analogLib* and place it between *out* and the *spt2switch* input as in Figure 10. Specify the following:

`DC voltage: VinDC-VoutDC`

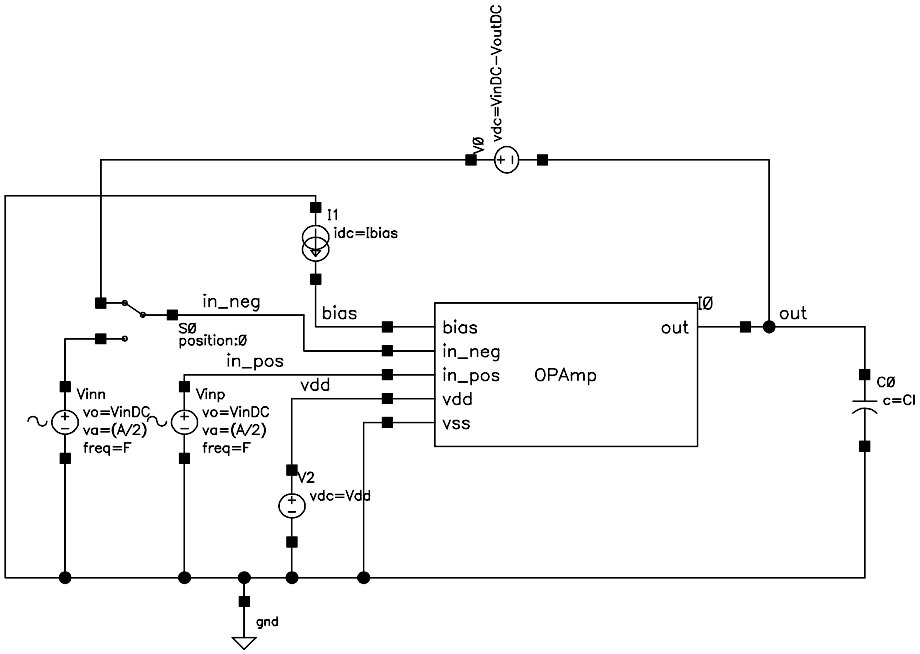

Figure 10 - Cadence schematic of the modified OPAmp_testbench cell.

Check and save the testbench.

9.2 Simulate input and output range

Go to the “adexl” view and create a new variable called *VoutDC* and specify *Value* to 1.65 and press *Apply*.

Now change the value for the variable *VinDC* to “0:0.165:3.3” to sweep *VinDC*. Start the simulation.

When the simulation is completed fill in the following table:

Table 6 Results of input range simulation.

Performance metric	From (V)	To (V)
f_u > 190 MHz (200 MHz-5%)		
DC-gain > 57 dB (60 dB-5%)		
PM > 66.5º (70º-5%)		
Input range		

To simulate the output range do another parametric analysis for *VoutDC* from 0 to 3.3 V (“0:0.165:3.3”). When this simulation is done fill in the values in the following table:

Table 7 Results of output range simulation.

Performance metric	From (V)	To (V)
f_u > 190 MHz (200 MHz-5%)		
DC-gain > 57 dB (60 dB-5%)		
PM > 66.5° (70°-5%)		
Output range		

Common mode rejection ratio (CMRR)

Differential amplifiers are used because of their excellent ability to suppress common-mode signals, i.e. signals that are common to both positive and negative inputs. This makes them resistant e.g. to noise since it often affects both inputs the same way. **Common Mode Rejection Ratio (CMRR)** is a measure of how well common-mode variations are suppressed and is defined as:

$$CMRR = \left|\frac{A_{differential}}{A_{common-mode}}\right| = A^{dB}_{differential} - A^{dB}_{common-mode} \quad (26)$$

where $A_{differential}$ and $A_{common-mode}$ are the maximum values versus frequency.

9.3 Simulate CMRR

Go to the "adexl" view and change the value of *phi_neg* in the "Design Variables" from 180 to 0. This will ensure that we only have common mode input signals. Create a new expression named *Gain_CM* with the expression:

db20(VF("/out")/(0.5*(VF("/in_pos")+VF("/in_neg"))))

Run a simulation.

Fill in the common-mode gain at low frequencies and CMRR below.

Low-freq. common-mode gain (dB)	
CMRR (dB)	

10 Power supply rejection ratio (PSRR)

Power Supply Rejection Ratio (PSRR) is a measure of how well power supply variations are suppressed. It is defined as the minimum of the positive and negative PSRR:

$$\begin{aligned} PSRR &= \min\left[\left|\frac{A_{differential}}{A_{Vdd}}\right|, \left|\frac{A_{differential}}{A_{gnd}}\right|\right] \\ &= \min\left[\left|A^{dB}_{differential} - A^{dB}_{Vdd}\right|, \left|A^{dB}_{differential} - A^{dB}_{gnd}\right|\right] \end{aligned} \quad (27)$$

Prepare for the PSRR measurement by adding an extra DC-source (called *vdc* in a*nalogLib*) on the negative supply as shown in Figure 11.

Modify The DC source connected to the positive supply (*Vdd*) with the property variable *AC magnitude = Asupp V.* Set the *AC magnitude* of the *DC*-source connected to the negative supply to *Asupn V.* Check and save your modified schematic. Import the new property variables to the simulator.

Define a new expression: VoutdB=db20(VF("/out"))
Run an AC analysis with *A* = 0, *Asupp* = 1 and *Asupn* = 0.

Fill in the A_{Vdd} at the highest point and positive PSRR below.

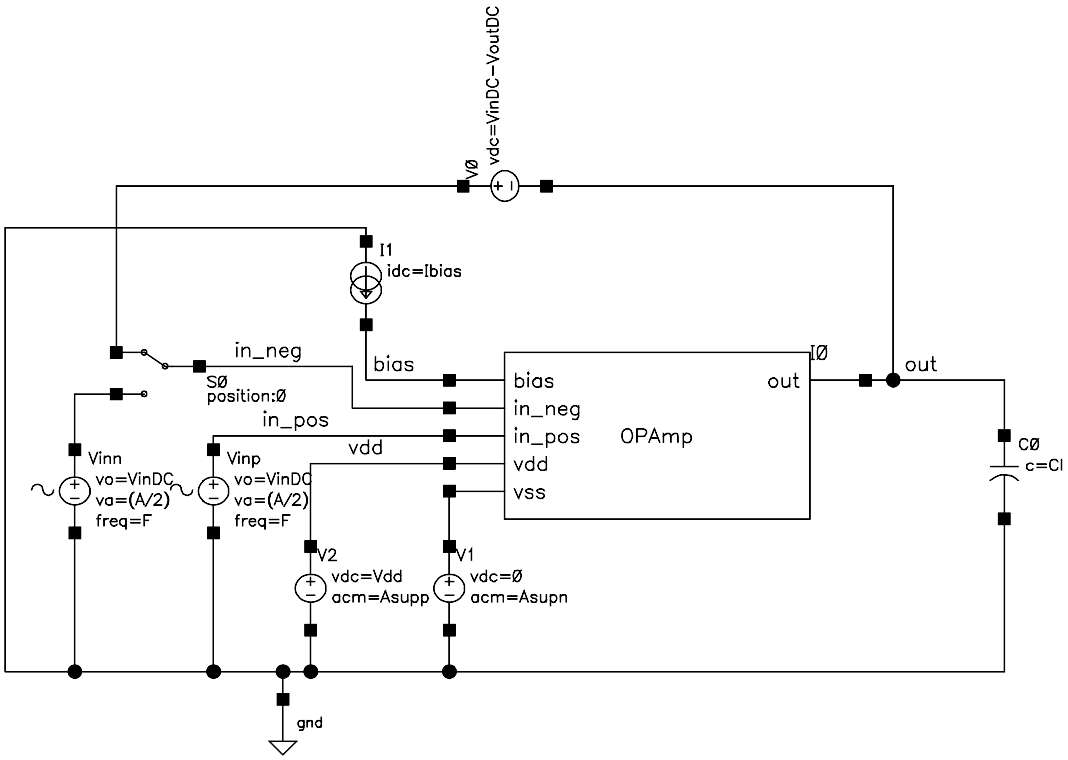

Figure 11 - Extra sine sources added to the positive and negative supplies.

Measurement	Value
A_{vdd}	
Positive PSRR	

In the same way, run an AC analysis with *A* = 0, *Asupp* = 0 and *Asupn* = 1 to find the maximum negative PSRR over the frequency range.

Measurement	Value
A_{gnd}	
Negative PSRR	

Measurement	Value dB
PSRR	

11 Global variations impacting the design

11.1 Temperature variations

Usually when designing circuits, the specifications given should be met for a certain temperature range. When the temperatures change the behavior of the transistors and other components in the circuits also changes. This causes shifts in the performance from the nominal values.

In the simulator you can change the simulation temperature by adding a new variable with the name temperature and the desired temperature as the value.

Change the simulation setup back to differential gain at nominal input and output values (*Asupp = 0, Asupn = 0, VinDC = 1.65, VoutDC = 1.65, phi_pos = 0, phi_neg = 180*, and *A = 2*).

Temperature (°C)	DC-gain (dB)	Unity-gain freq. (MHz)	Phase-margin (°)	Supply current (mA)
-40				
0				
27				
85				
125				

11.2 Voltage variations

Set the simulation temperature back to 27 ºC. Change the design variable *Vdd* and run simulations and fill in the table below.

Power supply (V)	DC-gain (dB)	Unity-gain freq. (MHz)	Phase-margin (º)	Supply current (mA)
2.97				
3.30				
3.63				

11.3 Load variation

Set the supply voltage back to nominal 3.3 V. Change the design variable CL and run simulations and fill in the table below.

Load capacitance (pF)	DC-gain (dB)	Unity-gain freq. (MHz)	Phase-margin (°)	Supply current (mA)
0				
3				

11.4 Process variation

The manufacturing plants are not capable of producing exactly the same transistors every time. The parameters of the transistors will therefore vary between different chips. To let the designers take this into account, the manufacturer has provided a set of transistor models which corresponds to typical transistors, slow transistors and fast transistors. We use two types of transistors, NMOS and PMOS, which can be slow and fast independently. This gives us five different corners, which are:

Table 8 Corners

Abbrevation	Meaning	NMOS	PMOS
cmoswp	Worst power	Fast	Fast
cmosws	Worst speed	Slow	Slow
cmostm	Typical mean	Typical	Typical
cmoswo	Worst one	Fast	Slow
cmoswz	Worst zero	Slow	Fast

As you can see the names refer to digital circuits.

In “adexl” you can add corners by expanding the “Corners” list in the left pane and click (LMB) “Click to add corner”. The dialog box “Corners Setup” will open showing you the different corners which has been setup. As for now only one corner is shown, “Nominal”, which uses the typical mean transistors. Click the thermometer button labeled “Add new corner”. A new corner named “C0” will appear, change the name to “WP” and then click (LMB) “Click to add” under “Model Files”. In the dialog box that appears click (LMB) “Import from Tests” and then “OK”. Now change the “<section>” for the “cmos53.scs” model file to “cmoswp” and tick the box next to the text “cmoswp”. Repeat the procedure for each corner in Table 8 except “cmostm”. When you are done, click “OK” to save your changes and close the dialog box.

Run the simulations and fill in the following table:

Corner	DC-gain (dB)	Unity-gain freq. (MHz)	Phase-margin (°)	Supply current (mA)
cmoswp				
cmosws				
cmostm				
cmoswo				
cmoswz				

11.5 Worst-case simulations

Look at the results for temperature, voltage and process variation and identify the worst-case condition for each of the four metrics, DC-gain, unity.-gain frequency, phase-margin, and supply current.

Simulate the worst case combination for the four metrics and fill in the table below.

Worst case for:	Corner	Temp. (°C)	Power supply (V)	Load cap (pF)	DC-gain (dB)	UGB (MHz)	Phase-margin (°)	Supply current (mA)
DC-gain								
UGB								
PM								
Supply current								

Are these the worst-case scenarios?

Restore the default values on supply voltage, temperature, process-corner, and load (3.3 V, 27 ºC, *cmostm*, 3 pF).

Save the current state. You will need it later in the lab!

Statistical variations impacting the design

In the previous section we analyzed variation that impacts all transistors on the entire chip in the same way. Some certain performance metrics are impacted more by differences between transistors in the same circuit. As an example the two input transistors in a differential stage are expected to behave equally. However, if they differ in any parameter we will obtain an offset at the input.

11.6 Monte-Carlo analysis

Monte-Carlo analysis is a statistical simulation tool with which random process variation can be analyzed. What the tool does is to run a large number of simulations where the process parameters for a circuit are varied randomly. The statistical distributions of the output data can then be analyzed.

ADE XL

To run Monte-Carlo analysis we have to change from ADE L to ADE XL. As seen by the name it is a bigger version of ADE and it contains much more functionallity. Fortunaly Cadence has a simple way to migrate your ADE L test setup to ADE XL.

*Press Launch->ADE XL, select Create New View and click **OK**. A popup window will appear asking you where to save the adexl view, the default location is fine so just press **OK**.*

As you can see the view is a bit different from ADE L. We will just explain the most important parts here but feel free to look in the manual and experiment a bit after the lab. To the left you have a Data View pane showing the tests you have created, your Global Variables (Design Variables in ADE L), Parameters (not used for now) and Corners. To the right you have three tabs: Outputs Setup, Results and Diagnostics. They are quite self explainatory.

Some of the advantages of ADE XL is that you can define different tests which allows you to run different types of simulations on the same or different circuits at the same time. More flexible corner control, you can run all corners at the same time. It saves previous results so you easily can compare them. And much much more.

Prepare simulation setup

Change the value of *phi_neg* from 180 to 0 by expanding the *Global Variables* list on the left and click on the value for *phi_neg*. We now want to measure the common mode gain of the amplifier. We are interested in the offset voltage between the positive and negative inputs. Add an expression for the offset voltage to the plotted outputs. The expression should be:

VDC("/in_pos")-VDC("/in_neg")

and call it *Offset*. You do this by click the **RMB** on the *Outputs Setup* list and press *Add Expression*. A new expression is added to the list and here you can enter the name and the expression. Also add one expression for the low-frequency common mode gain. The expression should be:

value(db20(VF("/out")/(0.5*(VF("/in_pos")+VF("/in_neg")))),100)

and name it *CMgainDC*.

The Monte-Carlo tool runs a large number of simulations so we would like to reduce the simulation time for each run as much as possible. Begin by disable all outputs except *Offset, CMgainDC* by deselecting the *Plot* checkbox in the *Outputs Setup* pane. Run a simulation by clicking *Run->Single Run, Sweeps and Corners* or the green play button and check that the values are reasonable, i.e. compare with the common mode DC-gain (phi_neg = 0) of the result in section 9.3 and the offset voltage should be in the µV-range.

If you get a simulation error stating that "Value of 'leff' should be greater than zero" you have to set the length and width of the corresponding component to for example 1 mm. Since these are virtual components the size does not affect the operation, this is simply an error in the design kit we are using.

Monte-Carlo simulation setup

The Monte-Carlo analysis needs different types of transistor models, which includes the statistical variations and definitions for the devices. You need to manually change transistor models by choosing:

Corners > Click to add corner

Choose *Add new corner* (the icon looks like a thermometer) and give it an appropriate name like *MC*. Now you can select *cmosmc* from the dropdown menu next to *cmos53.scs* and enable it by pressing the checkbox in front of the name. Press ***OK*** and disable the *Nominal, WP, WS, WO, and WZ* corners by deselecting the checkbox in front of its name.

Now open the Monte-Carlo tool by clicking:

Run -> Monte Carlo Sampling…

The *Monte Carlo* window appears in which the Monte-Carlo simulation will be setup. In the window there are some fields that can be changed. In the *Number of Points* field specify 100 simulations. The *Method* option should be set to *Process Only*.

Click ***OK*** to start the Monte Carlo simulations.

Open the histogram plots by selecting the *Offset* and *CmgainDC* outputs and right clicking (RSB) and choosing *Histogram*. Inspect the histogram plots that appear and fill in the "Process variation only" part of the table below. In this run all model parameters have been changed in the same way for all transistors. To analyze the variation between transistors (i.e. mismatch), set the *Method* option to *All*. Run the Monte-Carlo simulations again and fill in the remaining part of the table below.

	Process variation only		Process and mismatch	
	Input offset voltage (V)	**DC common mode gain (dB)**	**Input offset voltage (V)**	**DC common mode gain (dB)**
Mean value				
Standard deviation (Sigma)				

Comment on major differences between the results from the two different simulation runs.

Part III is completed.

Date and signature of the lab assistant

Part IV -
Unity-gain buffer using the OP-amplifier in feedback configuration

Feedback is a powerful technique that finds wide applications in electronic circuits. For example, negative feedback allows high-precision signal processing and positive feedback makes it possible to build oscillators.

We have already used feedback to set the correct output DC-point of the amplifier by using the *sp2tswitch*-block. In this section you will look more into some properties of your OP-amplifier in a feedback configuration. The simplest feedback system is a unity-gain buffer, for which the output is feed back to the negative input of the OP-amplifier (see Figure 12). This circuit is often used as a buffer when a weak input signal should drive a large load.

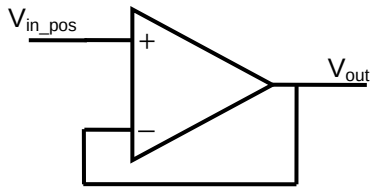

Figure 12 - Unity-gain buffer using an OP-amplifier in feedback configuration

12 Open and closed loop bandwidth

So far we have only analyzed the open-loop frequency characteristic of the OP-amplifier. We will now try to illustrate the difference the open and close loop frequency characteristics.

12.1 Open loop

Change the corner to *Nominal* and *phi_neg* to *180*. In the drop down list over the right pane make sure *Single Run, Sweepts and Corners* are selected.

Check that the AC analysis is active.

Run the simulation and view the frequency response of the input and output signals. Find the 3 dB bandwidth of the output signal by identifying the frequency for which the amplitude has reduced by 3 dB compared to the low-frequency amplitude or in the linear scale when the amplitude have reduced by $1/\sqrt{2}$.

3 dB bandwidth: _____Hz

Now find the unity-gain bandwidth by identifying the frequency for which the output signal is equal to the differential input signal (i.e. *A*).

Unity-gain bandwidth: _____Hz

12.2 Closed loop

Now go the testbench and change the parameters for the *sp2tswitch*. Change the position for the *AC position* from 2 to 1. This will set the switch so that also the AC-analysis is run in closed-loop configuration.

Check and save the schematic and go back to *ADE XL*.

Define a new output called *GainCL* with the expression:

db20(VF("/out")/VF("/in_pos"))

Netlist and run the simulation again.

Find the 3 dB bandwidth for the closed-loop system.

3 dB bandwidth: _____Hz

To explain how these numbers fit together we need to derive a transfer function of the closed-loop system.

First let the open-loop system be approximated with a first-order system with the transfer function in expression (28).

$$A_{ol}(s) = \frac{A_0}{\frac{s}{\omega_{p1}} + 1} \tag{28}$$

The linear model of the closed loop unity-gain buffer is defined according to Figure 13.

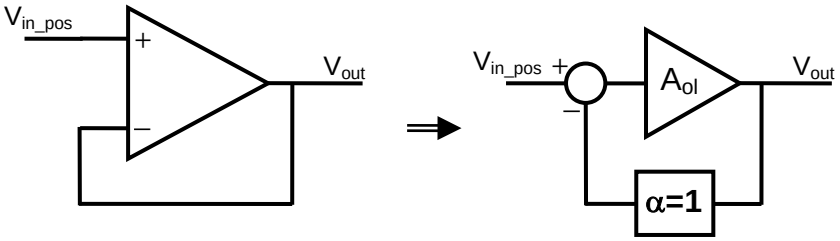

Figure 13 - Equivalent linear model of the unity-gain buffer.

The system can be described by the following equations:

$$V_{out} = A_{ol}\left(V_{in,pos} - \alpha V_{out}\right) \Rightarrow V_{out} = \frac{A_{ol}}{1 + \alpha A_{ol}} V_{in,pos}$$

$$A_{cl} = \frac{A_{ol}}{1 + \alpha A_{ol}} = \{eq.28\} = \frac{A_0}{\frac{s}{\omega_{p1}} + 1} \frac{1}{1 + \alpha\left(\frac{A_0}{\frac{s}{\omega_{p1}} + 1}\right)}$$

$$A_{cl} = \frac{A_0}{1+\alpha A_0} \frac{1}{1+\frac{s}{(1+\alpha A_0)\omega_{p1}}}. \tag{29}$$

With the help of the final closed-loop approximation in equation (29), comment on the three different bandwidth values obtained.

13 Closed loop large signal behavior

Edit the device properties of the *sp2tswitch* in the testbench schematic. Verify that the *Tran position* is set to *1*. This means that when a time-domain simulation (transient) is run the OP-amplifier will have a feedback from the output to the negative input. The feedback factor is in this case one, which means that the gain of the amplifier is unity.

Setup a transition simulation by clicking (LSB) on *Click to add test* under *Tests* in the left pane (*Data View*). Click *OK* in the dialog box that appears and choose:

Analysis > Choose…

Click on *Tran* and specify the *Stop Time* to *1u* (1 µs), *Accurate Defaults (errpreset)* to *conservative*, and enable the simulation by activating the *Enable* button and click *OK*. Close the *Test Editor* window and make sure the checkbox in front of it is enabled in the *Data View*. Click once on the name of the newly created test and change it to *Tran*.

Go to the *Outputs Setup* tab and click on the down arrow next to the button with a green plus sign and a probe in the top left corner. Choose *Tran* and S*ignal*. A new signal output has been added in the list where the first column, *Test*, is *Tran* since it belongs to the new test we just created. Right click (RMB) on it and choose *To be plotted.*

Select the *in_pos* and *out* nets in the testbench schematic and end by pressing [Esc]. Then delete the empty signal output by right clicking (RMB) and choosing *Delete Output*.

Check that the newly created test is enabled and the old one is disabled in the *Data View*. Start the transient simulation.

Verify that the input and output signals are similar.

Change the transient amplitude (design variable *At*) to 2 V and run the simulation again.

As you can see in the results the output voltage no longer follow the input due to limitations in the amplifier. The minimum and maximum output voltage can be determined from the simulation result and this gives us a second way of determining the output range. Compare the output range from the transient simulations with the output range you obtained in section 9.2. Comment on the difference. Which of the two output range results do you think is valid?

13.1 Slew-rate

The slew-rate is defined as the maximum voltage change per time unit by which the amplifier can change the output voltage. To simulate the slew-rate replace the sine-source connected to the positive input with a pulse source (*vpulse* in *analogLib*). Set the parameters of the pulse source according to:

Parameter	Value
AC magnitude	A/2
AC phase	phi_pos
DC voltage	VinDC
Voltage 1	1
Voltage 2	2
Period	(1/F)
Rise time	100p
Fall time	100p
Pulse width	(1/(2*F)-100p)

Check and save the schematic.

Go back to *ADE XL* and start the transient simulation.

In the waveform window you can see the square wave on the input signal with very fast rise and fall times. The output of the amplifier can not handle these fast edges and the output rise and fall times are therefore slower. Zoom in on one of the rising or falling edges by press-and-hold the right mouse button and draw a rectangle around the area you want to see (you can always zoom out by clicking *f* or *View > Fit [F]*).

Use the cross-hair markers A and B by pressing *a* and *b* and place the markers on the output voltage waveform to measure the slope. Fill in the positive and negative slew-rates in the table below.

Positive Slew-rate (V/s)	Negative Slew-rate (V/s)

We will return to these results in section 13.3.

13.2 Phase margin and closed loop stability

In part II we had a lower required limit on the phase-margin of 70°. In this section we want to illustrate why this phase margin requirement was imposed. By changing the resistance value (R_z) in the internal feedback in the OP-amplifier the phase margin is

impacted. This will impact the time domain behavior of the amplifier in a closed loop configuration.

In order to see the relationship between the phase-margin, which is a small-signal measure simulated for open-loop and the large-signal behavior of the amplifier in closed-loop we once again utilize the *sp2tswitch*.

Edit the properties of the *sp2tswitch* in the testbench schematic, and set the *AC position* to 2 and *Tran position* to 1.

Make sure you have this expressions in your *Outputs*:

Name: Gain

Expression: db20(VF("/out")/(VF("/in_pos")-VF("/in_neg")))

Name: Phase

Expression: phase(VF("/out")/(VF("/in_pos")-VF("/in_neg")))

Name: PM

Expression: phaseMargin(VF("/out")/(VF("/in_pos")-VF("/in_neg")))

Now enable the old test with AC analysis and DC analysis by checking the checkbox in front of it in the *Data View*.

Run a sweep where you change the resistance R_z from 0 to 2 times the value you got for R_z in Part II.

Comment on the relationship between phase margin and ringing in the time domain response.

Comment on the relationship between the unity-gain frequency and the ringing frequency in the time domain response.

13.3 Non-linearity

Nonlinearity is an important property to check for the amplifier. All non-linear phenomena in the circuit will cause distortion of the output from the ideal linear behavior. Hence, nonlinearities should in general be kept as low as possible.

One linearity measure that is commonly used is *Total Harmonic Distortion* or THD. The THD is usually quantified by summing the amplitude of all of the harmonics and normalizing the result to the amplitude of the fundamental as shown in (30).

$$THD = \frac{\sqrt{V_2^2 V_3^3 + \cdots + V_n^2}}{V_1} \tag{30}$$

To simulate nonlinearities go to the testbench schematic and change back from a pulse source to a sine-source on the input voltage node. The parameters of the sine-source should be as previously defined in section 6, which is:

Parameter	Value
Instance Name	Vinp
AC magnitude	(A/2) V
AC phase	phi_pos
DC voltage	VinDC V
Offset voltage	VinDC V
Amplitude	At V
Initial phase for sinusoid	phi_pos
Frequency	F Hz

Go back to *ADE XL* and add the following output expressions for the *Tran* test:

Name : THD_out

Expression : thd(VT("/out"),0,1u,2048,VAR("F"))

Name : THD_in

Expression : thd(VT("/in_pos"),0,1u,2048,VAR("F"))

The expressions above will give the THD of the output and input voltages. Cadence calculated the THD with a 2048 point FFT of the transient signal between 0 and 1µs and the fundamental tone is at a frequency of 10MHz.

Change *Rz* to the one you determined earlier.

To visualize the different tones we want to see the frequency spectra of the output and input voltages. Add the following DFT expressions, where the magnitude response has been determined using a 2048-point FFT between 0 and 1µs using a rectangular windowing function.

Name : DFT_out

Expression : abs(dft(VT("/out"),0,1u,2048,"Rectangular",1,1))

Name : DFT_in

Expression : abs(dft(VT("/in_pos"),0,1u,2048,"Rectangular",1,1))

Set the transient input voltage (*At*) to 2 V.

Run the transient simulation and view the results in the waveform window.

Look at the DFT of the input and output signals. At which frequency can you find the highest distortion term in the output spectrum? Ignore the tone close to DC since this is the DC offset.

Distortion frequency:_____Hz

What is the output THD? The expression for the THD gives the result in percent.

Output THD:_____%Run a sweep where you change the amplitude of the input signal (*At*). Find the maximum input amplitude for which the output THD is lower than 1 %.

What is the maximum input voltage amplitude?

Input voltage amplitude:_____V Now change the frequency of the input signal (*F*) from 10 MHz to 100 MHz and set the amplitude (*At*) to the maximum input voltage you found in the previous simulation.

Run a transient simulation and view the output voltage waveform. What is the output THD value?

Output THD:_____%Why has the THD increased compared to the result found at 10 MHz?

Hint:

Look at the transient response and relate to the slew-rate discussion in section 13.1.

Part IV is completed.

Date and signature of the lab assistant

Appendix A -
How to enter design variables and simulation analyses, run simulations and view results

13.4 Design variables and its units

When setting the design variables it is important to type in the unit in a correct way. Consider the following desired variable values:

Parameter	Value
W	10 µm
I	100 µA
C	100 fF
R	0 Ω

To set the variables correctly in the "object properties" the variables has to be set accordingly:

Parameter	Value
W	10u
I	100u
C	100f
R	0

13.5 Setup a simulation analysis

During circuit simulation, the designer can choose to do a variety of simulation analyses such as: *DC Analysis, AC Analysis, Transient Analysis and Parametric Analysis*.

In the *ADE XL Test Editor* window choose the type of analysis to perform by

Analyses > Choose

and select the desired analyses one by one in the "Choosing Analyses" dialog box. Several options have to be set for each chosen analysis. After setting up each analysis type (as explained below), **Apply** must be pressed.

13.6 DC analysis

It is often interesting to run a DC analysis to be able to view the DC operating points of the transistors (**vds** and **vdsat** to check if a transistor is saturated). To make this possible, the *Save DC Operating Point* box must be checked. The DC operating points can be viewed after the simulation by choosing

Results > Print > DC Operating Point

and then clicking on transistors in the schematic. A new window with the operating point information is opened. The same way, it is also possible to find the DC node voltages by choosing

Results > Print > DC Node Voltages

In the *Choosing Analyses* window check the *Design Variable* box in the *Sweep Variable* section to find the DC values for a certain range of a design variable. The design variable to be swept is chosen by pressing the **Select Design Variable** button and making a choice in the "Select Design Variable" pop-up window.

Start and Stop values for the design variable are entered in the *Sweep Range* section. When a simulation is run, the selected outputs are plotted vs. the chosen design variable. This is a useful simulation e.g. to set the output DC level.

13.7 AC Analysis

An AC analysis is useful to find the gain and bandwidth of a design. The standard way to use an AC analysis is to sweep the frequency from a low to a high value which results in a transfer function from input to output. Start and stop frequencies are entered in the *Sweep Range* section.

13.8 Transient Analysis

A transient analysis is maybe the most useful analysis. It simulates time domain data from time 0 to the stop time entered in the "Choosing Analyses" window.

13.9 Parametric Analysis

When a Parametric Analysis is executed, several simulations are run after each other while a component variable is varied through an interval. Choose

Tools > Parametric Analysis

in the *ADE L* to open the "Parametric Analysis" form.

The name of the variable to be swept is entered as *Variable Name*, start and stop values as *From* and *To* respectively, and the total number of steps desired as *Total Steps*. The parametric sweep is started by choosing

Analysis > Start

This results in a number of output waveforms, which will be plotted in the *Virtuoso Visualization & Analysis XL* window or VIVA for short.

13.10 Run a simulation

To make the simulator extract a netlist of your design and run a simulation, choose

Run > Single run, Sweepts and Corners

Note that the design has to be checked and saved after all changes for this command to work! Otherwise, Cadence will fail to generate a netlist and an error will occur.

A started simulation can be stopped by

Run > Stop Simulation

14 Viewing Results

Outputs you have selected are plotted in the *VIVA* window. If the cursor is moved along a curve, the simulated values of that signal at those points are displayed in the upper left corner. Zooming is controlled from the *View* menu. Style and range of axes in the selected sub-window can be changed by **RMB** clicking an axis. A sub-window is selected by clicking in it. To plot waveforms in a certain sub-window in separate plots choose

Graph > Split Current Strip

The waveforms are plotted together again by choosing

Graph > Combine All Analog Traces

Markers for making measurements can be displayed by

Marker > Create Marker

Or the ***a*** and ***b*** keys.

Voltage values for the marker points and the difference between Marker A and B are displayed between the markers.

Appendix B -
Hot-keys in schematic and layout views

General Commands:

cancel command [Esc]

redraw [F6]

undo [u]

zoom in [ ]]

zoom out [[ ]

zoom [ctrl]+mouse wheel

zoom to fit [f]

Add commands:

add instance of cell [i]

add pin [p]

add wire (narrow) [w]

add wire name [l]

Edit commands:

copy [c]

delete [delete]

move [M]

properties [q]

rotate [r]

stretch [m]

change orientation [F3]

decend into instance [e]

decend into instance to edit [E]

return from instance [B]

check and save [X]

find markers [g]

Plot Commands:

marker A [a]

marker B [b]