DAT116 (Mixed-signal system design)
Lab 2: Variability and feedback

Lars Svensson, Lena Peterson
`larssv@chalmers.se, lenap@chalmers.se`

Version 4.1, November 19, 2018

1 Introduction

In this lab session, we will study effects of feedback on parameter variability, using the Cadence system. Instructions are fairly detailed; it is highly recommended that you follow them carefully, as the software is not very forgiving (as most other professional tools—starting with chef's knifes—it gives you plenty of opportunities to do the wrong thing).

2 Preparation

Read Chapter 2, "Top-Down Design", of Kundert, Zinke: "The designer's guide to Verilog-AMS". It is available in the Chalmers E-library.

Rehearse negative feedback. Refer to the lectures and to the reading material.

We assume a certain level of familiarity with the Cadence system, and especially with its circuit simulation capabilities. A highly condensed version of the Cadence documentation is available as the *Cadence Crib Sheet*; circuit simulation is described in Chapter 9. Read this if you don't know Cadence. Even if you know Cadence already, the *Crib Sheet* may be useful to jog your memory. Keep it available throughout the session.

The MATLAB-only labs may be carried out in Windows or in Linux. The Cadence system is available on Linux only, on `heffalump.ita.chalmers.se`.

3 Setup and launch

Connect to `heffalump.ita.chalmers.se` according to instructions given separately.

Cadence needs to keep some configuration files in its working directory. There is a course-specific launch script which will handle this requirement.

- Create a directory in which to keep the working files for this lab, and `cd` to it.
- Then, launch Cadence by issuing this command:

  ```
  /usr/local/cad/course/DAT116/Y2018/stm065 &
  ```

  (If you don't want to cut-and-paste this command every time you launch, you may put the command in a script of your own or make a shell alias.) Wait for the Cadence *console window*, marked **Virtuoso 6.1.6**, to appear.

You will need to create a cell library in which to keep the cells you work with.

- In the console window, select **Tools→Library Manager...** A browser window opens, showing all the cell libraries you have access to.
- In the library manager window, select **File→New→Library...** You will be able to select both the name for the library, and a place in your underlying Unix file system. Name your library distinctively! In this lab PM, we assume that you use the canonical name `foobar`.
- Once you have clicked **OK**, a dialog box asks you to decide on a technology file for the library. Select `Attach to an existing technology library`, click **OK**, select the library `cmos065`, and click **OK** again. Verify that the **Library Manager** window now includes the name of your new library.

4 Using an op-amp macro model

As discussed by Kundert, structured mixed-signal system design progresses mainly from higher levels of abstraction to increased level of detail. In earlier stages of design exploration, it is necessary to evaluate design alternatives without working through all details. In particular, it would not be productive to carry out lengthy transistor-level simulations of an amplifier without first determining the performance requirements given by the application. Thus arises a need for efficient abstract simulation models, or *macro models*, for commonly-used blocks such as amplifiers. We will use such a model to illustrate how negative feedback counteracts circuit performance variation.

- In the **Library Manager** window, select the library `dat116`. The library contains several macro models for operational amplifiers. Select the cell `op1`. You will find a `symbol` view and an `ahdl` view, but no schematic view.
- Double-click to open the `symbol` view and inspect the graphic representation used for the amplifier when included in larger circuits. Close the window when done.
- Double-click to open the `ahdl` view and inspect the description of the amplifier behavior. Verify the lack of a schematic-level description (don't worry too much about the actual content at this time). Close the window when done.

We will now use the opamp macro model together with two resistors in a classic inverting-amplifier configuration.

- Go back to the Library Manager window and select your work library `foobar`. Select **File→New→Cell view...** to create a new schematic view named `lab2`. A schematic-entry window appears.
- In the **lab2** schematic-entry window, press `[i]` to enter the add-instance-mode. Choose **Browse** in the dialog window. In the appearing library browser window (which is easy to mistake for the library manager window—check the title bars!), select the library `dat116` and the cell `op1`. Place an instance of the opamp in **lab2**.
- We will also include two resistors. Go back to the library browser window and select the library `analogLib` and the cell `res`. A parameter window opens and a resistor symbol is visible when the the cursor is moved into the **lab2** schematic window. Ignore the parameter window for now and place two resistances such that they can be easily connected into an inverting-amplifier configuration. (Middle-click to rotate the symbol by 90° if desired.)
- From `analogLib`, also get a sinewave generator `vsin` and two ground connectors `gnd`.
- Exit the add-instance mode with `[ESC]`. Hit `[w]` to enter the wire mode. Connect the components to look something like Figure 1.

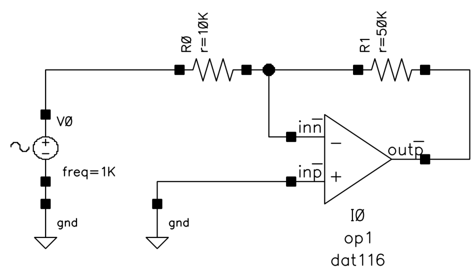

Figure 1: Lab2 circuit diagram

- Exit the wire mode with `[ESC]`. Hit `[X]` to check and save your design. Verify in the console window that no errors occurred.
- You must also enter parameter values for source and resistors. Select each component in turn and press `[q]`. In the **Object Properties** window, set the resistance of the feedback resistor to 50 kΩ, the input resistor to 10 kΩ, and the `Amplitude` and `Frequency` of the sinewave generator to 0.1 V and 10 kHz, respectively[1].
- Check and save design again. If required, fix errors and re-check-and-save. Close the schematic window.

In order to simulate a design which includes VerilogAMS macro models, we need to create a *hierarchy configuration* for the design (as described in Chapter 9 of the *Crib Sheet*).

- Using **File→New→Cell view...** in the **Library Manager**, create a configuration view for the cell `lab2` by selecting the value `config` for the `Type` parameter; the `Open With` and `View Name` parameters should take the correct values automatically. Click **OK**. A **Virtuoso Hierarchy Editor** window appears, together with a **New Configuration** window. The latter is used to quickly initialize the `config` view.
- The `Global Bindings` lists in the middle of the window will provide defaults for view selection when a design is set up for simulation. We will use a template list accessible through **Use Template...** at the bottom of

[1] Do not confuse `Amplitude` and `AC magnitude`. The latter is used in AC analysis, to which you will return later in the lab.

the window. Click that button. A small dialog window appears. Select the template `AMS` and click **OK**. Verify that a number of entries appear in the view lists in the **New Configuration** window. Also verify that in the top pane of the window, the view of the top cell is set to `schematic`.

- You are then finally ready to press **OK** in the **New Configuration** window.

The **Virtuoso Hierarchy Editor** window has now been populated with default bindings for each cell in the design. It is possible to override these bindings by typing into the `View to Use` field for each cell. For now, the defaults are what we want.

- The newly created configuration must now be saved, through the hierarchy editor menus or via `[^S]`.
- Exit the hierarchy editor with `[^Q]`.

The groundwork is now complete for mixed-mode simulation of electrical components using the macro model of an operational amplifier. We will control simulation from the **Virtuoso Analog Design Environment**.

- In the **Library Manager**, double-click the `config` view of `lab2` to open it. A dialog appears, offering to open the configuration view and/or the schematic view. Choose both and click **OK**. Two familiar windows open. Note, however, that the title of the schematic window includes the configuration in force for the cell shown.
- In the schematic window, select **Launch→ADE L** to open the **ADE L** window[2]. A helpful error message informs you that you need to select the correct simulator for your selected configuration. Acknowledge the message by clicking **OK**, select **Setup→Simulator/Directory/Host**, make sure that `Simulator` is set to `ams`, and click **OK** to dismiss the dialog.
- Verify that the `Design` pane in the **ADE L** window specifies the configuration view of the `lab2` cell that you just opened. Otherwise, change it through **Setup→Design. . . .**
- Set up a transient simulation with a duration of 1 ms, using conservative accuracy defaults. Arrange to save and plot all node voltages. (If necessary, refer to Sections 9.3 and 9.4 of the *Crib Sheet* for detailed directions on how to do this.)

[2]The menus also offer two other, more capable, versions of the Analog Design Environment. We stay simple in this lab.

The last preparatory step, *netlisting*, converts all design information to a detailed description of interconnected components which is readable by the simulator. It is important to realize that netlisting works from the *saved* version of *all* cell views. To be sure that you simulate what you intend to simulate, it is a good habit to explicitly check and save all files after a modification, using **File→Check and Save**, `[^X]`, or the corresponding toolbar button.

- Choose **Simulation→Netlist and Run** to produce the input file for the simulator and to run it. A window opens to keep you informed of simulation progress. When the job is completed, a graph window opens and shows the resulting signal waveforms. Verify that amplitudes and frequency are what you expect!
- To save the plot for your report, select **File→Save Image...** in the plot window. A window opens to let you select filename and file format, and also set some image options. Save the image and inspect the file. If desired, change the options and repeat.
- Close the plot window.

5 Component variability

Our first goal in this lab session is to demonstrate how negative feedback counteracts parameter variation. The effect is best seen in a Bode plot, which can be produced with an AC simulation.

- In the **ADE L** window, select **Analyses→Choose...** This time, select an AC simulation rather than a transient simulation. Enter a frequency range from 0.1 Hz to 10 MHz, using a logarithmic scale with 5 points per decade.
- Click **OK**. Notice that the `Analyses` pane in **ADE L** now lists *both* the previous transient simulation and the AC simulation you just specified. Uncheck the **Enable** flag for the transient simulation.
- As mentioned above, the AC magnitude of a sine source is specified separately from the amplitude used in transient simulation. In the schematic window, select the sinewave source, press `[q]` to call up the **Edit Object Properties** window, make sure that the `AC magnitude` is set to `1 V`, and click **OK** to accept the values. Check and save the schematic.
- In the **ADE L** window, select **Simulation→Netlist and Run** or press the **Netlist and Run** shortcut button. When simulation is complete, a

graph window appears, showing the signal magnitude at each node as a function of frequency.

The magnitude at the input node is the same as that of the input source and is therefore constant across all frequencies, as expected. The output magnitude should be set by the feedback components.

- Verify visually that the output magnitude agrees with the input magnitude and the feedback resistors (at least at low and medium frequencies).

You may also use *markers* in the graph window to accurately inspect the values at any point of a curve.

- Approach one of the curves with the mouse pointer. A small dynamic window opens to show the x and y values of the indicated point. Press `[a]` to place a marker at the current point.
- Move the mouse pointer to another point on a curve in the diagram and press `[b]`. A second marker is placed, and the x and y differences are shown in the window.
- Grab and move the markers with the mouse. Observe how the display changes. To remove a marker, select it and press `[DEL]`.

The AC analysis is *linear*: all generated signals are calculated to be proportional to the inputs, regardless of its magnitude (which is not the case with a real amplifier with limited supply voltages!). Clearly, an input magnitude of 1 V makes it particularly easy to read out the frequency-dependent gain from the AC-simulation diagrams. Regardless of magnitude, we may also calculate gains by using the built-in *waveform calculator*.

- In the graph window, select **Tools→Calculator...** to open a calculator.

The calculator offers both algebraic and RPN[3] modes of entry. The rest of this lab PM assumes the algebraic mode (which you are probably more used to, unless you use a very old Hewlett-Packard calculator or software emulating such equipment).

- In the calculator window, select **Options→Mode** and make sure **Algebraic Mode** is checked.

[3] Reverse Polish Notation

The simulation has provided values for voltage magnitudes as functions of frequency. To calculate the gain, that is, the ratio of two such magnitudes, we may select two nodes in the schematic and specify algebraic operations in the **Calculator** window. Operation results will be shown in a pane in the waveform viewer.

- In the **Calculator** window, find and click the **vf** radio button (to specify a **v**oltage as a function of **f**requency). Observe the feedback text in the bottom of the *schematic* window.
- In the schematic window, click the wire connected to the output of the amplifier. Observe the expression that appears in the **Calculator** text area.
- In the **Calculator**, enter a slash character, "`/`", to indicate division. Then, select the circuit input node, that is, the wire that connects the sinewave source and the input resistor in the schematic, as above. When the expression in the **Calculator** text area is complete, press `[ESC]` in the schematic window to leave the expression-entry mode.
- Select **New Subwindow** rather than **Append** in the destination menu in the middle of the **Calculator** window. Then, select **Tools→Plot**. The graph pane splits, a new pane appears, and a new diagram appears in the new pane. Verify that it shows the expected circuit gain as a function of frequency.

Bode-style plots use logarithmic axes. We will now edit the calculator expression to produce the decibel values[4] for the gain. As the signals are voltage levels, the decibel values are given by multiplying the 10-logarithm of the voltage-level ratio by 20. The calculator offers a built-in function for this common operation.

- Edit the calculator expression so that the voltage ratio is the argument of the function `dB20`. Select **Replace** rather than **New Subwindow** in the destination menu.
- Make sure that the just-created pane is the selected one (it should be marked with a red border; if not, select it with a mouse click). Then, again select **Tools→Plot** to plot the curve. Observe the old diagram curve being replaced by the new one. Verify that the low-frequency gain has the expected dB value.

[4] `http://en.wikipedia.org/wiki/Decibel#Power_quantities`

- Go back to the **Calculator** window and edit the expression to show the gain from the inverting input of the op-amp to the output, that is, the *raw gain* of the op-amp. Now use the destination **Append** to add the new curve to the graph window without erasing its previous contents, and plot again. A second Bode gain plot should occur with the first one.
- Use marker-based readout from the gain plot to determine the change in overall circuit gain from 10 Hz to 1 kHz.
- Make copies of the plots for your report.

As expected, the circuit gain is set by the feedback resistors as long as the raw op-amp gain is large enough. For high frequencies, the circuit gain falls off with the raw op-amp gain.

Already from these graphs and from the values you read out, it should be clear that the resistive feedback produces very similar gains for very different raw gains (the difference in raw gain is 40 dB over two frequency decades!). The same effect occurs when amplifier parameters are changed, as will be shown next.

- Go back to the schematic window. Select the op-amp symbol and press `[q]` to edit its properties. The parameter `AOLDC` represents the open-loop gain at DC. Change its value to `100` (the unit is decibels) and accept the change by clicking **OK**. Check and save the schematic.

You want to maintain the calculated Bode diagrams you just set up, and have them refreshed when the underlying simulation results change.

- Right-click on one of the Bode diagram waveforms. A context menu appears; select **Send To→ADE**. Repeat for the other Bode diagram waveform. Verify that the calculated waves appear in the **Outputs** pane of the **ADE L** window.
- At the bottom of the **ADE L** window, change `Plot after simulation` to `Refresh`. A dialog box warns you about the behavior of the calculated diagrams. Dismiss the message after reading.

As you have changed a parameter of one of the components, netlisting is necessary before simulating again.

- In ADE-L, select **Simulation→Netlist and Run** or press the **Netlist and Run** shortcut button.

- When simulation is complete, all the curveforms are updated in the graph window. Observe that the raw low-frequency gain has changed in accordance with the parameter alteration. Also observe that the overall circuit gain is virtually unchanged.
- Make copies of the plots for your report.

6 Multi-component variability

Above, we have investigated how variations in one parameter (the raw op-amp gain) can affect circuit behavior. In reality, *all* circuit parameters are subject to variations to some degree, and these variations may emphasize or counteract eachother. Manual assessment of such effects is cumbersome, so design environments provide various methods to semi-automatically explore these parameter spaces. Here, we will investigate the straightforward method of simple parameter sweeps.

6.1 Single-parameter sweep

In our first example, we will sweep the raw gain of the opamp. The first step is to arrange for the gain to be derived from a controllable variable.

- Open the opamp property window again. Replace the numeric value for the gain with a variable name such as `opgain`. Click `OK` to close the property window, and check and save the design.
- In the **ADE L** window, select **Variables→Copy From Cellview**. Observe that the newly introduced variable name shows up in the design variable pane.
- In the design variable pane, double-click on the new variable name. Enter the nominal parameter value `120` (decibels) and click `OK` to close the window. Note that the value shows up in the name-value listing.
- Netlist and run the simulation to verify proper behavior after the modification.

Swept-parameter simulations are controlled from a separate window.

- In **ADE L**, select **Tools→Parametric Analysis...**. A **Parametric Analysis** window appears, showing a table of variables and sweep parameters, with one row already initialized.
- In the text field in the column `Variable Name`, enter `opgain`. Use the other fields to specify a variable value range from 20 to 120 (dB), a `Step Mode` of `Linear Steps`, and a step size of 10 dB. Make sure that the `Sweep?` box is checked.
- In the **Parametric Analysis** window, select **Analysis→Start All**.

Simulation progress will be tracked with a status bar across the **Parametric Analysis** window. Each finished simulation also generates a completion message in the console window.

When all the simulations have been completed, the waveform window shows sets of curves for each plotted node and for each parameter value. To toggle visibility of a certain curve, click on the eye symbol in the signal list; for detailed visibility control per parameter value, click on the `+` sign to expand the list.

- Zoom into the diagram showing the original voltage curves (that is, not the log-scale Bode curves). The low-frequency output magnitude approaches its ideal value for increasing values of the `opgain` parameter. At what gain value is the low-frequency output magnitude within 10% of its ideal value? What about 2%? Relate your observed values to the results of simple calculations.
- Make plots for your report.

6.2 Multi-parameter sweep

We will now extend the simulation setup to sweep several parameters.

- Return to the schematic-entry window. Open the property window for the input resistor and enter a parameter name such as `rin` for its resistance value. Accept the change by clicking `OK`. In the same way, connect the value of the feedback resistance to the parameter `rfb`. Check and save the schematic as before.
- In the **ADE L** window, copy the variable names from the cell view to the `Design Variables` pane, and enter nominal values which correspond to the original values used in the circuit.

- Return to the **Parametric Analysis** window. Mark the `opgain` row and click the red × button to delete the row.

- Add a new row to the table, using the button to the left of the delete button. Enter specifications for `rin` with `Range Type` set to `Center/Span%` . Set the center values to the nominal value given in **ADE L**, and the span to 20% (i.e., ± 10%). Use 6 steps.

- Repeat these stages for the `rfb` parameter.

- Make sure that both `Sweep?` boxes are checked. Select **Analysis→Start All** to run the sweeps. The **Parametric Analysis** window will show, in its status line at the top of the window, how many runs remain before completion.

After completion, the plot window shows a large number of magnitude curves for the selected circuit nodes. What are the largest and smallest values used for the two resistors? What maximum and minimum low-frequency output magnitude values did you expect, and why? Do the simulation results match your expectations?

- Make plots for your report.

6.3 Parameter co-variation

The sweep mechanism may also be used to model (crudely) the effects of parameter co-variation. Typically, some sources of variation will influence several components in the same way, while others will have uncorrelated influences. Here, we will use one extra parameter to model the "global" variations.

- In the schematic window, open the component property window for the input resistor and change the resistance value from `rin` to the expression `rglob*rin`. Make the corresponding change for the feedback resistor. Check and save the design.

- In the **ADE L** window, copy all variables from the cellview. Enter the nominal value `1` for `rglob`.

- In the **Parametric Analysis** window, add another variable to the sweep pane, as before. Let `rglob` vary by 20% around its center value 1. Reduce the span of the other resistance parameters to 2%.

The total number of runs should be on the same order as in the previous example. How can you select the number of steps for each of the sweeps such that the total simulation time is still reasonable?

- Enter the numbers of steps for each sweep, and start the analysis.

After completion, the plot window shows a number of output magnitude curves. What are the largest and smallest values used for the two resistors? What maximum and minimum low-frequency output magnitude values did you expect, and why? Do the simulation results match your expectations?

As you have noticed, even very simple simulations of very small circuits may take significant time when several parameters must be varied. Time is of the essence in industrial design projects, and several approaches are therefore used to speed up these verifications. The individual simulation runs are mutually independent and may therefore be run in parallel on several machines (typically arranged in a compute cluster with dozens or hundreds of processors).

Additionally, running a full set of simulations based on *all* combinations of *all* parameters may not be a cost-effective way to useful information. It is common practice to take random samples of the multidimensional parameter space until a certain statistical confidence has been reached. We will revisit such *Monte Carlo simulation* in a future lab.

7 Wrap-up

After completing this lab session, you are supposed to be able to do the following:

- Create new Cadence libraries, create new cells, and use instances of cells from other libraries in your new cells
- Perform mixed-model circuit and system simulations, including macro models, using the Cadence hierarchy editor
- Use the Cadence Calculator to produce plots defined by simple expressions involving simulation results
- Use single- and multidimensional parameter sweeps in simulations to assess circuit-behavior sensitivity to parameter variations

Reflection questions:

- Assume for a moment that *perfectly accurate* resistors were available for use in feedback networks; that is, the resistor tolerance is 0%. Then, any deviation from the value set by the feedback resistor ratio is caused by insufficient raw op-amp gain. What minimum raw gain would be needed to get within 1% of an intended overall gain of 10? What if the intended gain is 100 instead?

- A high *and predictable* gain may require several gain stages, even when a single stage could provide raw gain in excess of the overall gain needed. Why is this?

- How should gain be *distributed* across two or more stages to get the most accurate overall gain? Consider both your findings in the previous items (which assumed perfectly accurate resistor values) and Hastings's discussion of matching of unequal resistors. Assume that the total silicon area is fixed (or, equivalently, that the total resistance is fixed).