EESD Cadence Crib Sheet
65 nm version

Lars Svensson, Simon Kristiansson, Lena Peterson

Draft of Version 1.01 of August 29, 2013

Contents

1 Introduction **2**
2 Terminology **2**
3 Formatting **4**
4 Setup **5**
5 Cadence windows **5**
6 Interaction models **7**
7 Schematic capture **8**
8 Layout **12**
9 Circuit simulation **16**
10 If things go wrong **21**

1 Introduction

In the computer labs in the EESD program, you will use tools from Cadence Design Systems (CDS), or Cadence for short. Cadence is currently one of three leading suppliers of CAD tools for electronic design (the others being Synopsys and Mentor Graphics). A complete Cadence-based environment comprises tools for most of the tasks an electronics designer can come across.

This document gives a brief summary and some tips on how to work productively with Cadence, with an emphasis on circuit design and simulation[1]. However, it is neither a manual nor a tutorial. It is mainly intended to reduce the amount of time spent browsing the excellent but *very* extensive documentation that accompanies a Cadence installation.

Note that the Cadence environment is configurable to an extreme extent. Thus, commands may have different names in other installations; menus may look different; other tools may be available instead of or as a complement to those mentioned here; and so on. Do not expect another installation to behave in detailed accordance with these descriptions!

This document describes the Cadence setup assumed by the CMOS065-5.2.2 design kit from ST Microelectronics (STM). The semiconductor processes covered by this design kit all share the line width of 65 nm. The main Cadence product version is *Integrated Circuit Design Environment 5.1.41* (which is actually an oldish version of Cadence).

2 Terminology

To facilitate understanding of this document, we will first introduce some of the terminology used.

Design flow A *design flow* is the succession of steps that are executed during a design project. Every step typically requires several iterations where the design is refined and more information is added. For each of these steps, specialized tools are available.

In a "real" industrial project, the design flow is constructed at an early stage

[1]The Cadence framework also offers tools for system- and chip-level design tasks; these are not yet covered by this version of this document.

(often with tools from several different vendors), and this flow is retained throughout the project. Our lab exercises are only intended to illustrate a few of the steps in a typical design flow; we will therefore mostly encounter Cadence tools.

Cell Designs are organized hierarchically into smaller pieces, so called *cells*. The size of a cell is perferably kept small enough for a designer to handle all its details simultaneously. A cell can contain instances of other cells, so called *components*.

View A cell may have different *views*, which present different aspects of the design for inspection or modification. Some common views are schematic, symbol, and layout.

It is very important to understand that the Cadence environment does not automatically maintain consistency of the different views of a cell!

In many cases, a view can be automatically generated from another view by use of a tool; however, in most cases, this tool must be explicitly run by the designer.

Library A library is a collection of cells. You may picture cells as files, and libraries as folders.

Process From the point-of-view of a designer, a semiconductor process is distinguished by the electrical parameters that determine performance of a design, and by the design rules that must be obeyed for the design to be manufacturable on a particular manufacturing line. This collected information is loosely called a "process", or sometimes "technology".

Technology file The process information required by a certain tool is commonly collected in a configuration file, called a "technology file". The information in such a file is almost always specific to that particular tool, and can only rarely be faithfully translated for use by another tool.

Design kit The complexity of commercial chips increases each year. Tools and other software must keep up with this development. Tool vendors update their software often, to fix bugs and to enhance features and performance, unfortunately not always with the compatibility with other tools kept intact. IC manufacturers and foundries, on the other hand, develop and verify design flows, combining software from several different vendors. They then collect all technology files and other configuration files plus cell libraries in a complete package for a certain process. Such a package (often called a

Design kit) is verified to work with certain versions of *each* included tool and *cannot* be expected to work with other versions!

Licences As already mentioned, the Cadence environment is very comprehensive and contains many, more or less independent, tools. Commercial users pay Cadence to use each tool, typically according to the number of simultaneous users. A license system is used to ensure that tools are used in accordance with the Cadence contract. Licenses are managed by a *license server* which is queried before a tool can be started. Thus, if the connection with the license server is lost, the tools generally stop working. This happens occasionally, although redundant servers and other safety measures help minimize the outages.

3 Formatting

In this document and in the lab memos for some courses, the following formatting conventions are used to highlight menu choices, keys to be pressed, etc.

- A *menu choice* is denoted with bold font and arrows that denote submenus:

  choose **Tools→Verilog Integration→Verilog-XL**

- *Keys* that are to be pressed, for example keyboard shortcuts, are denoted in typewriter font between brackets:

  push `[p]`

- *Buttons* to be clicked are denoted thus:

  click on **OK**

- Any *text* that is to be typed in, or that is printed by Cadence, is denoted with typewriter font:

  `'View Name' schematic`

  Names of fields are written using single quotes and typewriter font, while the names of different windows and forms are written using bold font:

  the field `'Cell Name'` in the form **Create New File**

4 Setup

Several aspects of the Cadence system (such as what design kit to assume) are controlled mainly by sizable sets of Unix environment variables. Manual maintenance of these variables would be highly involved, since there is no simple principle for their influence on different tools. A special utility, the `module` system, is used here and at many other Cadence installations. The system keeps track of tool and environment dependencies and keeps the setup consistent (updates `PATH` variables, etc). Typical usage is to include some `module` commands in a setup file, such as `.cshrc`.

The `module` system is described in a `man` page (`man module`). The two most frequently used commands are `module list` (which shows what modules are currently loaded) and `module load foobar` (which loads the module `foobar` and any other modules which `foobar` relies upon, thereby typically making some tool available).

Here, the command

```
module add cmos065-5.2.2
```

sets up the tools for the 65-nm processes and enables the command

```
stm_065
```

which launches the system assuming this technology.

5 Cadence windows

The Cadence environment uses many different windows. For our purposes, it is important to be aquainted with the following ones.

5.1 Command Interpreter Window, CIW

This window (also called **icfb**) is the hub of the Cadence environment. This is the first window that appears when Cadence is started (once the splash screen has

disappeared); here is where warnings, errors, and other information from tools are shown. If something goes wrong, this is the first place to look for information. *Note:* Inexperienced users often iconify this window and ignore its contents, which leads to frustration later on, when a problem occurs and the information is hidden in the icon.

Do not iconify the CIW. Feel free to enlarge it.

5.2 Library Manager

The Library Manager is one of the tools available through the CIW.

In the Library Manager, design libraries and the cells they contain are displayed, created, and modified. As mentioned in Section 2, cells and libraries are similar to files and folders in a file system. In this analogy, the Library Manager corresponds to a file-system browser window, but with built-in knowledge of which kinds of cell views there are, and with certain barriers against mixing cells from incompatible processes.

5.3 Tool windows

Most tools open separate windows for the designer to interact and issue commands. For example, a schematic-entry tool provides a drawing area where the designer can place components and connect them; and a simulation window gives opportunities to set parameter values and to select what signals to save. The designer typically spends most of her time in tool windows.

5.4 Input forms

Components typically have many parameters. These can often be given reasonable default values, with no input from the designer. The same applies to parameters for tool programs. Input forms are used to override such default values and to enter required parameter values.

Four buttons appear in most input forms encountered in the Cadence environment: **OK**, **Apply**, **Cancel**, and **Help**.

OK approves the form content and closes the form.

Apply approves the form content but does not close the form.

Cancel cancels any changes made and closes the form. The parameters in the form thus keep their old values.

Help opens a Cadence documentation window. (An installation problem prevents this feature from working properly in the local system. A document with links to the PDF versions of the manuals is available. Hopefully, the existence of this crib sheet will minimize the need for serious searches.)

5.5 Pop-up windows

Pop-up windows are windows with few, or no, possibilities for interaction. A "What's new" window may be displayed at start-up; during a simulation, detailed information is displayed about the progress. These windows are typically only of brief interest and may then be dismissed to reduce desktop clutter.

6 Interaction models

The Cadence environment uses several different models of interaction with the designer, which can be confusing to a beginner. A certain command can often be performed in several different ways:

- A complete set of all commands is available through the tool menus.
- For the more common commands (saving the file etc.), there are shortcut buttons.
- There are also one-keystroke keyboard shortcuts for some common commands.

In addition, many commands can be used both modally and non-modally. Here is one example from the schematic editor tool Virtuoso Composer (see below in Section 7):

- If a component in a schematics window is selected, it can be deleted with the keystroke `[DEL]`.

- If no component is selected, [DEL] will instead put the tool in a "delete mode". In this mode, each component subsequently selected with the mouse will be deleted, until the user exits the mode by hitting [ESC]. (Most users soon make very good use of the [ESC] button...)

7 Schematic capture

7.1 Menus

The Cadence tool for schematic input and editing goes by the fancy name of Virtuoso Composer. As many other Cadence tools, the schematic editor offers menus, button shortcuts, and keyboard shortcuts. The most common ones are collected in the column of buttons at the left edge of the window:

Design→Check and Save	[X]
Design→Save	[S]
Window→Zoom→Zoom In By 2	
Window→Zoom→Zoom Out By 2	
Edit→Strech	[m]
Edit→Copy	[c]
Edit→Delete	[DEL]
Edit→Undo	[u]
Edit→Properties→Objects...	[q]
Add→Instance...	[i]
Add→Wire (narrow)	[w]
Add→Wire (wide)	[W]
Add→Wire Name...	[l]
Add→Pin...	[p]

Some common commands did not fit in the column of button shortcuts:

Window→Fit	[f]
Window→Zoom→Zoom In	[z]

The modal interaction model (see above) is often used; [ESC] is then used to exit a command mode. The current mode is indicated in a text line at the top of the window, together with the number of selected objects. There are also two text lines at the bottom of the window. The upper one of these shows the functions cur-

rently associated with the three mouse buttons; the lower one shows the command prompt and other information. *Look at these lines of text often!*

7.2 Component entry

Component entry is one of the most common tasks in any schematic editor. The command **Add→Instance...** opens the form **Add Instance**, which will remain open for as long as the tool is in component-entry mode. A specific component is specified by library, cell, and view; either by typing in the names, or by using the library browser available by pressing **Browse**.

The form will be updated when the library, component, and view have been selected so that any parameters can be given values. Parameter values have numerical prefixes placed immediately after the number: 10 μm is given as `10u m`.

While in component-input mode, each left-click with the mouse places one instance of the component according to the current specification in the **Add Instance** form. A right-click rotates the component by 90° without placing it. It is not necessary to exit the input mode in order to give another component specification: cell name and cell view can be changed in the form and the new component specification is applied at the next left click. Exit the input mode with `[ESC]`.

If the input form is hidden by accident, it can be recovered with `[F3]`.

7.3 Input of nets

Nets are input in a similar way to components, with the command **Add→Wire (narrow)**; the form is somewhat simpler than for components. There is also a command **Add→Wire (wide)**, which is identical but for the graphical representation of the wire.

In the net-input mode, the designer left-clicks to indicate what component terminals the net should connect. Virtuoso then connects the component terminals, using horizontal and vertical line segments laid out according to an unknown and only partially predictable algorithm. It may be helpful to left-click at some intermediate points though which the wire should pass.

The terminal or net vertex closest to the mouse marker is highlighted with a yellow diamond: ◇. Instead of carefully placing the marker on the terminal and left-

clicking to connect it to the net, one can use the command keystroke [s] ("snap"): the terminal with the yellow diamond will then immediately be connected to the net without any further marker positioning. In addition to speeding up the work, this procedure decreases the risk of a common mistake, namely having a net that stops just short of reaching the intended terminal.

A vertex in a net is marked with a round dot, which symbolizes a solder joint. In Virtuoso, such dots are placed automatically. The tool does not allow more than three wires to be connected at such a dot; this decreases the risk of mistaking a crossing of wires for a vertex with four wires.

Nets that have been started by mistake (a common phenomenon) may be thrown away with [ESC].

Virtuoso allows two component terminals to be connected without an explicit intermediate net, by just letting the two terminals coincide. Don't do this. This practice makes it impossible to select an individual terminal with the mouse, and the invisible net between the terminals cannot be given a name (see Section 7.5). The schematics used in the EESD courses are not complex enough to motivate such area-savings.

7.4 Entering pins

Pins provide the cell with electrical connections to the rest of the world. They define the interface used when the cell is to be used as a component in another cell.

Pins are added to a cell with the command **Add→Pin...**. Correct assignment of signal flow direction (input, output, etc) may help catch mistakes.

7.5 Using net-names

Net names are added with the command **Add→Wire Name...** (or [l]), according to principles similar to those described above. Some net-name conventions are important to know:

- Nets with the same name, in the same schematic, are considered connected. In other words, wire connections do not need to be explicitly shown: two wire stubs with the same name at different ends of a large schematic work

just as well as a long wire across the schematic. *When used with restraint,* this convention can increase the readability of complicated schematics.

- A single net can have only *one* name. If two nets with different names are connected, the command **File→Check and Save** will complain and point out the error.

- A net can symbolize a *bus* with several signals. The name of the net is then often of the form *xxx*< *m*:*n*> (for example `a<3:0>`). A comma-separated list of the form *x1*`,` *x2*`,` *x3*`,...` (e.g. `in1,in2,in3,rst`) can also be used. Mixed forms and even more complicated expressions are also supported, but comprehensibility is degraded accordingly. The complexity of our course examples does not require the use of such expressions.

  The rules controlling what happens when nets of different widths are connected are easy to misunderstand. Avoid such constructs whenever possible.

- When placing individual net names or pins, it may be handy not to have to edit the name string for each of the wires in a wide bus. *Bus expansion* allows the designer to compactly specify the names as described in the previous bullet point; these names will then be used in order, one name for each label placed.

- *Global nets* are available in all cells in a design, without having to be explicitly connected though pins. A net becomes globally available if its name ends with an exclamation mark, `!`. Common examples are `vdd!` and `gnd!`. In simple designs, it is practical to treat the power supplies this way. In larger designs, where several different power supplies may be required, the issue is not as clear-cut.

7.6 Hierarchy

As has been seen, a schematic can contain components that are instances of other schematics. Sometimes the need arises to change the under-lying schematic, or the symbol representing such a component. An obvious way is to open a new window from the **Library Manager** and make the change in this new window. For small changes, however, there is a faster way. The command **Design→Hierarchy →Descend Edit...**, or `[E]`, temporarily changes the selected view in a window to a view of a cell further down in the hierarchy. The command **Design→Hierarchy →Return**, or `[^E]`, returns the designer to the original cell. Several similar commands are found in the menu **Design→Hierarchy**.

7.7 Editing symbols

Symbol views are also created and edited in Virtuoso. The easiest way to create a symbol is to start from an existing schematic, using the command **Design→Create Cellview→From Cellview...** The appearance of the automatically created symbol (a green rectangle with green pins, where the connections are marked by red squares) may seem a bit simplistic. It may, however, be easily modified. The green parts are graphical ornaments without any deeper significance: they can be changed as desired (with commands in the **Edit** menu), and new details can be added (with commands in the **Add→Shape** menu). This is how to create traditional symbol shapes for AND gates, adders, and so on.

The connections, on the other hand, *must* correspond to the pins in the underlying schematic. They can be moved arbitrarily along the perimeter of the symbol with their name tags following. The name tags can also be moved separately.

7.8 Undo

As implied by the name, Virtuoso is primarily designed for virtuosos. For a true virtuoso, every action is First Time Right, so there is never a need to backtrack and correct. For the rest of us, it is important to note that:

The command **Edit→Undo** *(*`[u]`*) can go back only one (1) step!*

Make it a good habit to think before doing large modifications, and save your work regularly.

8 Layout

8.1 Menus

The chip layout editor in the Cadence environment is called Virtuoso Layout. The name indicates a connection to the schematic editor, and many of the commands are similar. In the shortcut menu we find, among others, the following commands:

Design→Save	[F2]
Window→Fit All	[f]
Window→Zoom→Zoom In By 2	[^Z]
Window→Zoom→Zoom Out By 2	[Z]
Edit→Strech	[s]
Edit→Copy	[c]
Edit→Move	[m]
Edit→Delete	[DEL]
Edit→Undo	[u]
File→Properties	[q]
Create→Instance...	[i]
Create→Shape→Path	[p]
Create→Shape→Polygon	[P]
Window→Create Ruler	[k]

(When a layout view is in a read-only state, the shortcut menu is much abbreviated.) Some other useful commands are not present in the short-cut menu:

Window→Zoom→Zoom In	[z]
Window→Redraw	[^R]
Window→Clear All Rulers	[K]

Some seem to be available only from the keyboard:

Abstract View	[^F]
Full View	[F]

8.2 The active layer

The input of layout geometries is made in a workspace that corresponds to the workspace in the schematic editor. This window is named **Virtuoso Layout Suite**. There is also a second window called **LSW** (Layer Selection Window), which is a palette where different layers can be selected to be shown, or to be edited, and where one selects the *active layer,* which is the layer to be affected by subsequent editing commands. Two toggle buttons decide whether instances and pins should be selectable or not. Four buttons decide whether all layers should be selectable and visible at the same time: **AV, NV, AS, NS** are interpreted as "All/None Visible/Selectable". There is also a large button marked **Show Objects**, which when clicked will replace the layer palette with a list of object types, including pins and instances, and whether these should be visible and selectable.

The active layer for input is chosen by a left click in the palette; it is shown at the top in the **LSW**. A right click makes a layer un-selectable.

In most design kits, there are many layers in the layer palette. However, for most tasks only a small subset of these is used. In layout work, the layers of type `drawing` and `pin` are the important ones.

8.3 Input

Input of a layout is in many ways similar to the input work in the schematic tool. In addition to components (that is, instances of other cells), one can draw rectangles, polygons, and other geometrical shapes (these commands are collected in the **Create** menu). The tool also allows the designer to draw geometries such as circles and ellipses; but such geometries are seldom supported in simpler technologies.

As in the schematic tool, components can have parameters reachable via the form **Edit Instance Properties** (`[q]`). In the EESD layout labs, these parameters are mostly used to choose the width of transistors or change layers for pins.

A special **Create** mode, **Create→Path**, can be used to draw longer interconnects. The usage is similar to that of the wire tool in the schematic tool: the interconnect follows the points selected by left clicking in the layout window. (The command allows creation also of non-rectilinear wire shapes, which is however not supported by the technologies considered here.) The interconnect is completed by a double click (click fast or you get an extra segment at the end!) or by pressing return. By default, the interconnect width is set as narrow as allowed in the active layer, but it can be changed in the form **Create Path**, shortcut `[F3]`. In this form, the active layer may also be changed: the menu **Change To Layer** presents as options those layers for which a contact with the present layer is defined.

For completeness, we also mention the command **Create→Multipart Path**. This command can be used to create buses and other structures where several interconnects are to run in parallel. In our lab series, the single interconnect structures will typically suffice.

8.4 Editing

There are many ways to change an existing layout. Elements can be moved, deleted, or changed with respect to form, size or orientation. Commands for such actions are collected in the **Edit** menu.

Select The selection of an object is accomplished by a left mouse click. Additional objects can be selected, in the usual manner, by pressing and holding down the shift key during the selection process.

Move To move a selected object, two points must be specified; the object is moved as far as the distance between the points. It is often simpler, however, to left-click on the selected object and drag it to its new position.

Copy This command is used to create exact copies of one, or several, selected *objects*. In other software, "copy" often means that a selected *area* is copied to a temporary buffer, which may then be pasted into other places. In Virtuoso Layout, the command **Edit→Other→Yank** (`[y]`) handles that task.

Stretch Different objects can be stretched in different ways. A path can only be stretched lengthwise (the width is changed in the **Properties** form, available by `[q]`). For a rectangle, both the vertices and the sides may be moved. Stretchable points light up in yellow when the cursor is close. More flexible changes can be accomplished with the powerful, but somewhat awkward **Edit→Reshape** command.

Merge Several different geometrical shapes, such as rectangles, polygons, and paths, can be merged into one object provided that they are placed edge-to-edge in the same physical layer. The result is in the general case a polygon.

Chop In the same way as **Yank** corresponds to "copy" in many programs, the command **Edit→Other→Chop** (`[C]`) offers a corresponding feature to that commonly known as "cut". This command cuts "parts" of a geometrical object (a rectangle can in this way be transformed into a polygon, or vice versa). Note, however, that **Chop** does *not* save the objects that have been cut for a future paste. Precede this command with **Edit→Other→Yank** (`[y]`) if desired.

Paste This command (**Edit→Other→Paste**, `[Y]`) pastes an object that has been selected by **Edit→Other→Yank**.

8.5 Connectivity

It can be difficult to see without zooming if a connection through several components and/or hierarchical levels really is connected properly. Mistakes are often discovered in the LVS check or in simulation, but they can be discovered earlier and simpler with the command **Connectivity→Mark Net**. In this mode, a left click will highlight the *entire* net connected to the selected wire segment, also through components that are not visible. It is thus possible to quickly see if any circuit block is missing a ground connection, a power supply connection, or a clock connection.

8.6 Hierarchy

As in the schematic tool, it is possible to rapidly switch hierarchical level and perform changes in a subcell. In the layout editor, the command **Design→Hierarchy →Descend Edit** has no keyboard shortcut; `[E]` instead opens the **Layout Editor Options** form. Especially useful in layout work is to perform a change in a subcell "in place", so that the surrounding cell is visible while editing. The command **Design→Hierarchy→Edit In Place**, with the short form `[x]`, can be used for this purpose.

Hierarchical levels can also be created or deleted directly during the layout work. The command **Edit→Hierarchy→Make Cell** creates a new cell containing the selected objects, and replaces the objects with an instance of this new cell. The new cell can then also be used elsewhere. A corresponding command, **Edit→ Hierarchy→Flatten**, is used to delete one, or several, levels of hierarchy.

9 Circuit simulation

The Cadence system collects all simulation of analog circuit and system behavior in a unified user interface: the *Virtuoso Analog Design Environment* (VADE). Many common tasks, such as selecting what input parameters to vary and what output signals to save and plot, can therefore be carried out in a mostly simulator-independent manner.

9.1 Starting ADE

VADE may be started from the **icfb** window (or from a Virtuoso schematic editing window) via **Tools→Analog Environment**. The main VADE tool window includes a menu bar on the top, a shortcut bar on the right, and four panels labeled `Design`, `Analyses`, `Design Variables`, and `Outputs`.

9.2 Session and setup

The `Design` panel identifies the cell view to be simulated, by Library, Cell name, and View. When VADE is started from a Virtuoso window, these fields are filled in automatically. Otherwise, and to change the selection, use **Setup→Design** or the circuit shortcut button.

Setup→Simulator/Directory/Host opens a form which lets the designer choose among the available simulators, and optionally specify an alternative project directory (simulation files can quickly grow very large and are often placed on a file system without quota restrictions). It is also possible to specify another host computer for actually running the simulation, for example one with lots of main memory.

Entering the setup details for a complex simulation may be time-consuming, and recreating a setup the next day may be error-prone. It is possible to save and load simulation setups by **Session→Save State** and **Session→Load State**, respectively.

9.3 Selecting the analysis

The term "analysis" refers to a simulation task and its basic parameters; the most familiar one may be the *transient analysis*, where voltages and currents are calculated as functions of time for a certain duration. The menu choice **Analyses→ Choose** (or a shortcut button) brings up an input form where the analysis types available with the currently selected simulator are listed. A minimum set of parameters may also be required; more parameters can be tweaked in an extra `Options` form.

The `Analyses` panel lists all analyses configured; each of these may be turned on and off individually, in the input form or via the **Analyses** menu.

9.4 Selecting saved/plotted signals

In a large design, it may not make sense to save all voltage and current waveforms for all nodes (as already mentioned, the amount of data generated can be substantial). **Outputs→To Be Saved** lets the designer select the set of signals to be saved. Similarly, signals specified with **Outputs→To Be Plotted** will be displayed in a waveform viewer window when simulation is complete. (The waveform viewer is discussed further in Section 9.7 below.)

Signal selections can be carried out by clicking in a Virtuoso schematic of the view under consideration. The Virtuoso window will be set in a modal state—see above in Section 6—which must be exited with `[ESC]`. Select a net to save/plot its voltage waveform; select a component pin to save/plot the current through it. The selected nets and pins will be highlighted with colors corresponding to those which will be used in the waveform plots.

9.5 Design variables

It is possible to use symbolic names for circuit values in Virtuoso, and then assign values to these parameters at the start of a simulation. The **Variables** menu lets the designer enter name/value pairs manually; to save time and assure that no undefined variables are forgotten, use **Variables→Copy From Cellview**. The name/value pairs are listed in the lower left panel in VADE.

9.6 Netlisting and running

Once the simulation has been fully specified using the forms described above, the job can finally commence. Before simulation proper can be started, however, the *netlisting* task must be carried out: an input file for the simulator must be prepared from the cell hierarchy in the Cadence database.

NOTE: Netlisting works from the saved versions of all cell views, so any Virtuoso changes that were not saved will *not* be reflected in the simulation! In other words, if cell A contains an instance B of cell C and specifies a parameter for B, and if that parameter has been altered, then cell A must be saved to disk for the change to take effect, and netlisting must then be done anew. In contrast, if a variable value is changed in VADE, it is *not* strictly necessary to create a new netlist. However, class examples are usually small enough that little time is lost using

Simulation→Netlist and Run rather than **Simulation→Run**. (Both commands are also available in the shortcut bar.)

9.7 Waveform viewing and post-processing

Several waveform viewers are available in the Cadence system; with VADE, the default one is called WaveScan. Other tools are used to view the results of digital simulations.

When a VADE simulation is complete, a WaveScan window will open to display all signals marked for plotting. By default, the signals will all be drawn in the same coordinate system in a single window. A shortcut button (or **Axis→Strips**) splits the window into horizontal strips, each showing one waveform in its own coordinate system. It is also possible to split a window vertically into panels, and to open extra windows which may contain one or more coordinate systems. Waveforms may then be dragged from one coordinate system to another for easy comparison (grab the waveform label in the top left corner of the coordinate system). A multitude of commands allow zooming and panning, and setting waveform colors.

When the mouse pointer is in the WaveScan window, a cursor appears on the waveform trace closest to the pointer. The cursor coordinates are shown at bottom left. It is possible to select other cursor styles in the **Trace** menu.

The diagrams may be annotated with labels (**Graph→Label→Add...**), which are text strings, and with markers (**Marker→Place...**), which are similar to the waveform cursors but annotate the plot with values in the diagram itself.

The **Tools** menu of the WaveScan window lists three entries: **Browser**, **Calculator**, and **Table**. The Browser offers hierarchical viewing of all parameters and results for the simulation. The Calculator offers one-click calculation and display of sums, differences, and certain commonly-used functions (such as derivatives) applied to the simulation results. (For anything more complex, it is likely a better idea to use MATLAB for post-processing, at least for those of us who do not particularly enjoy SKILL and OCEAN coding. Interface libraries are available from Cadence and are installed on the lab computers.) The Table tool, finally, allows export into CSV format for viewing and manipulation in a spreadsheet program.

9.8 Parametric analysis

Many simulation experiments involve several runs with different values for one or more parameters, such as temperatures, supply voltages, input amplitudes, or transistor sizes. In VADE, such jobs are set up with the form brought up with **Tools→Parametric Analysis**. Named parameters as described in Section 9.5 are each assigned a series of values; as every combination corresponds to one case to be simulated, it is rarely useful to sweep more than two parameters in the same experiment.

Once the parameter sweeps have been set up, the multiple simulation runs are launched by **Analysis→Start** in the **Parametric Analysis** form. The parameter sweep setup can be saved with **Tool→Save...** and reloaded with **Tool→Recall...**

9.9 Mixed-mode simulation

The Cadence system supports *mixed-mode simulation:* that is, different parts of a system may be simulated at different levels of abstraction. A typical case might be when a transistor-level circuit description also includes an operational amplifier which still lacks an implementation. Mixed-mode simulation then allows a designer to investigate performance requirements (gain, bandwidth, offset, etc) for the opamp to arrive at a specification before starting the actual implementation work.

Once implementation is underway, there will be a need to simulate the system alternately with the abstract model and with the implementation to verify functionality and performance. In VADE, such alternatives are known as *hierarchy configurations*. A hierarchy configuration defines which view to select for each cell instance during netlisting (Section 9.6), and therefore controls what will be simulated.

A hierarchy configuration is itself stored as a cell view. To create a configuration, select the cell in question in the Library Manager and use **File→New→Cell View...** In the form, select the tool Hierarchy Editor to create a view named `config`.

A large system will contain many cell instances. Picking a view for each one by hand would be tedious and error-prone. A configuration therefore has a default list of view names, the View List, which will be searched for in order in each cell during the recursive traversal of the hierarchy. Additionally, traversal will

be stopped if a view in the Stop List is encountered. In complex design flows, these view lists may be quite long; to aid the designer, template lists are available to suit different simulator setups. Lists and templates are accessible in the **New Configuration** form which is opened when a `config` view is created.

The Hierarchy Editor window shows the bindings for all instances in the design, in a hierarchical or a tabular fashion, and lets the designer modify and save the configuration. *NOTE: Netlisting works from the saved versions of all cell views*, and since the configuration is a cell view, it must be saved before netlisting commences.

To actually run a mixed-mode simulation, it is finally necessary to use a simulator which is capable of handling such descriptions. In the present Cadence installation, that simulator is called `ams` under **Setup→Simulator/Directory/Host**. The `ams` simulator will not run without a `config` view for the top-level cell of the simulation.

10 If things go wrong

They do, sometimes...

10.1 If Cadence is not responding to input

If Cadence is not responding to input, there is most likely a dialog box waiting for input hidden behind another window. How to find such windows depends on the X server and window manager you use. On the lab systems, a window indication at the bottom of the screen shows the windows you have open.

(Any non-trivial Cadence session will spawn multitudes of windows. To minimize the risk of problems such as the one above, it is good to close windows that are not being used—but NEVER EVER the CIW!)

If you cannot find an open dialog box, you may be forced to kill Cadence. There will typically be at least one process per active tool.

10.2 If Cadence crashes

If Cadence crashes, or if you are forced to kill it, it will leave lock files lying around for all cells that were being edited. Before you restart Cadence, you need to remove these files or you will not be allowed to modify the locked cells. There is a command-line utility called `clsAdminTool` which can be used for this purpose. You may also manually remove files with the extension `.cdslck` from directories `library/cell/schematic` and `library/cell/symbol`.

There is also a file-locking daemon which could be causing similar problems. If nothing else helps, you may ask a system administrator to restart the file-locking daemon (`cdsd`).

10.3 Failed to get MPS handle

The Cadence waveform display tool **WaveScan** has been known to interact badly with the Linux daemon `nscd`, resulting in the cryptic error message in the section header. This daemon has therefore been turned off in the lab machines. If you must run Cadence on a machine where the daemon still runs, you may use the older waveform display tool **AWD**: From Analog Design Environment, do **Session→Options...** and select `AWD` rather than `WaveScan`.

10.4 Help does not call up a documentation reader

An installation problem (incompatible Java versions, apparently) prevents this feature from working correctly. A PDF document with links to the relevant documentation is available. We cannot post documentation on the web, for copyright reasons; thus, the links only work when you view the document on the Linux systems.