Evaluation of Aptivia and a Place and Route tool
Examensarbete utfört i Elektroniksystem
vid Linköpings tekniska högskola
av

Anna-Charlotta Klevbrink

LITH-ISY-EX--05/3591—SE
Linköping 2005-08-26
Evaluation of Aptivia and a Place and Route tool
Examensarbete utfört i Elektroniksystem
vid Linköpings tekniska högskola
av
Anna-Charlotta Klevbrink

LITH-ISY-EX--05/3591--SE

Handledare: Henrik Ohlsson

Examinator: Professor Lars Wanhammar
Linköping 2005-08-26
Evaluation of Aptivia and a Place and Route tool

Anna-Charlotta Klevbrink

This master thesis tells about Aptivia, what it contains and how it works (including a manual). As well as the problems with it. It also consists of an evaluation of a Place and Route tool, telling the discovered problems and ideas for solving them. There is also several different descriptions of the code that implements the Place and Route tool.

Aptivia, manual, standard cell, design automation, layout
# TABLE OF CONTENTS

1 Introduction ..............................................................7  
1.1 Background..............................................................7  
1.2 Purpose......................................................................8  

2 Aptivia ...........................................................................9  
Manual 9  
  2.1 Documents ................................................................11  
  2.2 Tests ...........................................................................14  
    2.2.1 Design .................................................................15  
    2.2.2 Includes ..............................................................16  
    2.2.3 Globals .................................................................20  
    2.2.4 Components ........................................................21  
    2.2.5 Analyses ...............................................................22  
    2.2.6 Sim Options .........................................................43  
    2.2.7 Measures .............................................................44  
    2.2.8 Run .......................................................................46  
  2.3 Sweeps/Corners .........................................................54  
    2.3.1 Sweep .................................................................55  
    2.3.2 Monte Carlo .........................................................56  
    2.3.3 Run Options .........................................................57  
    2.3.4 Job Status ............................................................58  
    2.3.5 Results .................................................................59  
  2.4 Spectre DCM Setup ...................................................61  
    2.4.1 Design .................................................................62  
    2.4.2 Function .............................................................63
1 Introduction

In the design of an electronic circuit one must consider all levels including both building components with transistors and making the layout of the transistors. This thesis mainly considers the layout-level of the design.

1.1 Background

When implementing a circuit, standard cells can be used to increase the design efficiency, compared to a full-custom design. In a full-custom design every component is made by hand in the design tool. Standard cells are small components, such as logic gates, that are pre-designed. In standard-cell design each logic gate has a specific design and all designs are equal in height. This makes it possible to place them on rows when placing the design. It is then easy to connect the inputs and outputs as well as VDD and GND that are at the top and bottom of the cell, respectively [1], [2]. In Figure 1.1 is an example of layout of a simple adder that has VDD at the top and GND at the bottom.

![Layout of an adder](image)

*Figure 1.1 Layout of an adder*

The standard cells are collected in a cell library. In a cell library several versions of each cell are included. Versions to be considered are different current-driving capabilities and number of inputs [2].
To simplify the implementation of cell layouts in Cadence, it is possible to write a program in skill for auto generation of the layout. This has been done in an previous master thesis work [3]. Skill is Cadence own high level language for programming and Figure 1.2 shows one example of this. More information about skill (in swedish) is found in [4].

In a cell library the cells that are possible to use are documented in detail. This includes the layout, a description of the functionality, terminal positions, delay and power consumption as a function of load capacitance and the input rise and fall times [2]. To do the characterization of delays and power consumption, the simulation tool Aptivia can be used. Aptiva does all the necessary testing of the given component.

1.2 Purpose

The purpose of this work is to evaluate and write an introduction manual for Aptivia as well as to look into possible ways to make the place and route tool work more accurately. The code for the auto generation program was also sparsely documented and the other part of this work was to document the source code.
2 Aptivia

Aptivia is a simulation tool in Cadence that gives the user a wide range of parameters that specifies a given cell. The big advantage of Aptivia is that the program can perform several tests in one run which saves a lot of time compared to do one at a time manually. It is only needed to specify the tests and the component that is to be tested and Aptivia performs the rest of the work. Cadence claims that Aptivia is an advanced design and simulation environment for their custom design platform and supports extensive exploration of multiple designs against their objective specifications. With Aptivia it is possible to run standard SPICE simulations as well as sweeps, corners and Monte Carlo analysis.

Manual

Before starting Aptivia one must naturally load the module, but also make sure that a new enough version of Cadence is used. One that works fine is 4.4.6MSR11, but if you got access to a newer one make sure that it is MSR11 or above. In the icfb-window in Cadence, Aptivia is found in the Tools-menu but can also be started in a system prompt by typing the command: aptivia.

The first time you start Aptivia you must define a Workspace before getting into the program, the later times you can select an existing one.

When Aptivia is running a lot of help can be found in the user’s guide that is available from the Help-menu when clicking Online Manuals.
After starting Aptivia and, in this case, selecting Test_Aptivia as Workspace the picture given in Figure 2.1 is shown on the screen. In this Workspace some tests of an inverter has been performed, and this will work as an example of the possibilities of Aptivia. Not all features have been used so some figures are empty on project related information.

To the left is all available files for the project Test_Aptivia, and to the right is all parameters. At the upper right side are the Workspace Parameters that are the same for the entire workspace. In this case there is only the temperature. At the lower right side are the Project Parameters that only are defined for that given project. In this case all are connected to the dcm-test that has been performed. (DCM stands for Design Characterization and Modeling) The parameters are the time
interval, the load capacitance, the slope, the temperature, the timestep and the VDD value.

After expanding all plus signs in the files and selecting the Test Console tab, Figure 2.2 is shown. The performed Tests are both visible as files on the left side, and in a netlist at the upper right corner where the information about them also are to be seen. It is possible to change the view from Netlist to Perl, Output, Errors, Cellview (but not in this case) and Other by using the View button.

2.1 Documents

When opening _testDCM_testDCM under Documents, as shown in Figure 2.2, the window shown in Figure 2.3 appears and when scrolling down to the end the text in Figure 2.4 appears.
### Project DUT pin information:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN</td>
<td>Input</td>
<td>Non-inverted</td>
</tr>
<tr>
<td>VDD</td>
<td>Power supply</td>
<td>Controlled by $\text{dcn}<em>\text{vdd}</em>\text{value}$ (default of 1.8 volts)</td>
</tr>
<tr>
<td>VEE</td>
<td>Neg supply</td>
<td>Constant 0 volts</td>
</tr>
</tbody>
</table>

### Project results information:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN_UT_rise_delay</td>
<td>Propagation delay for ‘IN’ to ‘UT’ rise</td>
</tr>
<tr>
<td>IN_UT_rise_slope</td>
<td>Output slope for path ‘IN’ to ‘UT’ rise</td>
</tr>
<tr>
<td>IN_UT_rise_t1</td>
<td>Start of transition for path ‘IN’ to ‘UT’ rise</td>
</tr>
<tr>
<td>IN_UT_rise_t2</td>
<td>End of transition for path ‘IN’ to ‘UT’ rise</td>
</tr>
<tr>
<td>IN_UT_fall_end_val</td>
<td>Ending voltage ‘IN’ to ‘UT’ rise (Not used in model)</td>
</tr>
<tr>
<td>IN_UT_fall_delay</td>
<td>Propagation delay for ‘IN’ to ‘UT’ fall</td>
</tr>
<tr>
<td>IN_UT_fall_slope</td>
<td>Output slope for path ‘IN’ to ‘UT’ fall</td>
</tr>
<tr>
<td>IN_UT_fall_t1</td>
<td>Start of transition for path ‘IN’ to ‘UT’ fall</td>
</tr>
<tr>
<td>IN_UT_fall_t2</td>
<td>End of transition for path ‘IN’ to ‘UT’ fall</td>
</tr>
<tr>
<td>IN_UT_vcc_end_val</td>
<td>Ending voltage ‘IN’ to ‘UT’ fall (Not used in model)</td>
</tr>
<tr>
<td>IN_input_cap</td>
<td>IN input capacitance</td>
</tr>
</tbody>
</table>

### Project parameter information:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{dcn}_\text{temperature}$</td>
<td>number</td>
<td>Simulation temperature. Temperature used during simulation. Default value=’27’.</td>
</tr>
<tr>
<td>$\text{dcn}<em>\text{vdd}</em>\text{value}$</td>
<td>number</td>
<td>Supply voltage. Voltage connected to pin ‘vdd’. Default value=’1.8’</td>
</tr>
<tr>
<td>$\text{dcn}_\text{slope}$</td>
<td>number</td>
<td>Input stimulus slope time. Must be less than ‘$\text{dcn}_\text{interval}$’.</td>
</tr>
<tr>
<td>$\text{dcn}_\text{interval}$</td>
<td>number</td>
<td>Time between input stimulus changes. Must be long enough to allow outputs to settle. Default value=’100ns’</td>
</tr>
<tr>
<td>$\text{dcn}_\text{timestep}$</td>
<td>number</td>
<td>Transient analysis time-step. Affects simulation accuracy.</td>
</tr>
</tbody>
</table>

*Figure 2.3 Upper part of _testDCM_testDCM-txt*
Figure 2.4 Lower part of _testDCM_testDCM.txt
As seen in Figure 2.3 this file begins with information about the document. At the top is the name of the file, the name of the project and what type of cell it is. In this case we have the file name testDCM, the project name Test_Aptivia, and the cell is an inverter with 1 input. Next comes the information about when it was created, by who, and in which directory.

The second area gives information about the pins, including their Name, Type, and a Description of each pin if there exist one.

The third area from the top contains Project results information with Name and Description. These are the results of the simulation.

The fourth area from the top, divided in Figure 2.3 and Figure 2.4, is the Project parameter information with Name, Type and Description. This are the same project parameters that are visible in the lower right corner of Figure 2.1.

The last area contains the Project content information that is a list of all files in the project and a short description of them. Almost all of them are seen in the left part of Figure 2.2.

2.2 Tests

When opening ak1/test1DCM/schematic/_testDCM_timing, the last one of the Tests in Figure 2.2, a new window opens with the tabs Design, Includes, Globals, Components, Analyses, Sim Options, Measures and Run.
2.2.1 Design

![Figure 2.5 Design](image)

The **Design** tab, as seen in Figure 2.5, gives the **Design location** defined by **Library**, **Cell** and **View**. It also contains the possibility to select a design via **Browse** and **Open Design** to open the design. **Open Design** is in this case light gray and therefore not enabled.

The button **View List** opens the window in Figure 2.6 which contains the **switchViewList** and the **stopViewList**. These ones are used to select lists of switch and stop cell views which contain information that can be simulated.

![Figure 2.6 View List](image)
2.2.2 Includes

Figure 2.7 Includes

Includes consists of four tabs as seen in Figure 2.7. They are: Netlist Files, Other Includes, Perl Commands and Perl Files.

Netlist files consists of three areas with Include directories at the top, Model/other includes in the middle and Specify: fileName sectionName at the bottom.

Other Includes looks the same as Netlist Files but with only two text areas named Stimulus file includes and Definition top includes.

Perl Commands is instead a large text area that states Ending Perl commands above it.
Perl Files contains a larger version of one of the three text areas seen in Figure 2.7 but with the text Ending Perl scripts.

**Templates**

When clicking on the Templates button in Figure 2.7, the window in Figure 2.8 appears.

![Figure 2.8 Workspace Templates](image)

The empty text area seen in Figure 2.8, can contain one or more Templates, and if so, the light gray buttons at the bottom are enabled. The button New opens the window in Figure 2.9 and Import opens the window in Figure 2.10.
As shown in Figure 2.9 it is possible to create a new workspace template by selecting the type (Test includes by reference or Test by copy), for what simulator and give it a name. In this case Spectre is the only available simulator. When the above is done it will be possible to use the OK button.

Figure 2.10 Import Workspace Template

Figure 2.10 shows a window that makes it possible to import a workspace template by selecting type (in Template type) and if to Import from test in project or to Import from other test file. The list beneath Import from
**Test project** contains all tests in the project (the ones that are seen below Tests at the left side of Figure 2.2) and makes it possible to select one of them. When marking **Import from other test file** and then clicking on the button with three dots the window in Figure 2.11 opens, where it is possible to **Select a Test/Template File**.

![Figure 2.11 Select a Test/Template File](image.png)
2.2.3 Globals

In the tab **Globals**, shown in Figure 2.12, the button **Add** is used one time to get the white areas below **Name**, **Default** and **Value Used**. These text areas makes it possible to add **Global design variables**. When using **Remove** the current line will disappear, and in this case leaving the area empty. It is not possible to leave the window when it looks like in Figure 2.12, because it either have to be empty or one or more design variables must be specified.

The disabled area at the bottom of the window is **Update design variables with fixed values** with the boxes **Evaluate operators** (marked)
and Evaluate suffixes plus the buttons Update, View Log, Restore and Clean.

2.2.4 Components

The tab Component, shown in Figure 2.13, contains a list of Components, and on the right side of it some buttons connected to that list. **Add** opens a new window where it is possible to add a component by selecting from a list with different kinds of components. **Delete, Copy**, and **Rename** deletes, makes a copy of, and renamed the selected component, respectively.

![Figure 2.13 Components](image-url)
In the upper right corner is a list of **Pins** with **Net** names, in this case it is only **plus** and **minus**.

In the middle is the **Stimulus** box with used stimulus and to the left three buttons. **Invert** inverts the stimulus and changes 0 into 1 and the other way around. **Repeat** adds one more of the same kind (in this case a 0, as the last one was a 0). **Toggle** adds one more but of the opposite kind (in this case a 1, as last one was a 0).

The last area is **Parameters** and it contains the parameters **High value**, **Low value**, **Interval**, **Rise time**, and **Fall time**. All of them with a text area containing the valid value. The three last ones are using **Project Parameters**. The values of the **Project Parameters** are seen in the lower right corner of Figure 2.1. Among the parameters is also a box that makes it possible to select **Repeat stimulus**.

Below the **Parameters** area is a small text area that gives the **Component type**.

### 2.2.5 Analyses

**Analyses** consists of the **Analysis types** **Transient**, **AC**, **DC**, **Noise**, **Sensitivity**, **XF** and **S-parameter**. Common for all types are the button **Nodesets/ICs**, that opens the window in Figure 2.14. The **Add** button is used one time for the upper part of the window, making the white text areas appear. The white text areas makes it possible to add **Nodes** and **Voltages** both for **Nodesets** and **ICs**. To use them, the corresponding box must be marked.
Figure 2.14 Analyses Nodesets and Initial Conditions
Figure 2.15 Analyses - Transient

Figure 2.15 shows the Analyses tab with the Analyses type Transient selected. The box Enable transient analysis must be marked, otherwise everything that belongs to Transient analysis will be disabled.

Transient analyses consist of Stop time in seconds, which in this case is $dcm\_interval$ times three, and the area Accuracy defaults (errpreset). The options there are Conservative, Moderate, Liberal and Default. Below that area is the Advanced button which opens a new window with the tabs Sim Control, Converge, Output, and Miscellaneous.
Figure 2.16 Sim Control

Figure 2.16 shows the tab **Sim Control** in Advanced Transient Analysis Options. It consists of the areas **Simulation interval parameters**, **Time step parameters**, **Integration method parameters**, and **Accuracy parameters**.

**Simulation interval parameters** has two text areas, one for **start** and one for **outputstart**. Both defined in seconds.

**Time step parameters** also has two text areas. They are **step** and **maxstep**, and both defined in seconds. In this case is step set to $dcm\_timestep$. 
Integration method parameters has only the options default, euler, trap, traponly, gear2, gear2only, and Trapgear2 for method.

Accuracy parameters has the options default, pointlocal, allocal, sigglobal, and allglobal for relref and an text area for iteratio.

Second tab in Advanced Transient Analysis Options is Converge as shown in Figure 2.17. It consists of the areas Initial condition parameters, Convergence parameters, State-file parameters and Newton parameters.

Initial condition parameters consists of the options dc, node, dev, and all for ic, and yes, no, waveless, rampup, and autodc for skipdc, and a text
area for readic. The button with three dots next to that text area makes it possible to select initial condition file.

Convergence parameters consists of two text areas, one for readns and one for cmin. The button with three dots next to the text area for readns makes it possible to select estimation file.

State-file parameters consists of text areas for write, writefinal, and okptperiod. Both buttons with three dots makes it possible to select DC solution file.

Newton parameters consists of one text area for maxiters and a selection between yes and no for restart.
Third tab in Advanced Transient Analysis Options is Output, as shown in Figure 2.18. Output only consists of the area Output parameters, which consists of selectors for save and oppoint as well as text areas for skipstart, skipstop, skipcount, strobeperiod, strobedelay, flushpoints, flushtime, and flushofftime.

The options for save are selected, lvlpub, lvl, allpub, all, and default. Only lvlpub and lvl makes the text area nestlvl usable.

Oppoint have the options rawfile, logfile, and no
The last tab in Advanced Transient Analysis Options is Miscellaneous, as shown in Figure 2.19. Miscellaneous only consists of Annotation parameters, which consists of the options yes or no for stats and no, title, sweep, status, and steps for annotate.

AC Analysis

When selecting AC as Analysis type and marking Enable AC analysis the window looks like Figure 2.20.
**Figure 2.20 Analyses - AC**

AC analysis consists of **Sweep variable** and **Sweep range**.

Options for **Sweep variable** are **Frequency**, **Design var**, **Temperature**, **Comp param** and **Model param**.

When selecting **Design var** the window looks like in Figure 2.21, with a text area for **Frequency** and a list for **Global name**. This makes it possible to do a simulation at a given frequency for a given global variable.

When selecting **Temperature** as sweep variable only **Frequency** remains with its text area.
Tests

When marking **Comp param** the **Sweep variable** area looks like in Figure 2.22, with text areas for **Frequency**, **Comp name**, and **Param name**.

When selecting **Model param** the options becomes **Frequency**, **Model name**, and **Param name**. As seen, all simulations except **Frequency** needs a frequency as input and can therefore only simulate at one given frequency at a time.
Figure 2.23 Sweep range

Shown in Figure 2.20 and Figure 2.23 is **Sweep range**, which consists of the **Start-stop** or **Center-span** selector with the belonging text areas. For **Start-stop** the text areas are **Start** and **Stop** (shown in Figure 2.20) and for **Center-span** they are **Center** and **Span** (as shown in Figure 2.23) Below them are a list for **Sweep type** with the options **Automatic**, **Linear** and **Logarithmic** and at the bottom a box for **Add specific points**. When that box is marked, as in Figure 2.23, a text area appears where it is possible to type in specific points.

The last thing that is specific for **AC analysis** is the **Advanced** button which opens the window shown in Figure 2.24.
Advanced AC Analysis Options consists of four areas named State-file parameters, Output parameters, Convergence parameters, and Annotation parameters.

State-file parameters consists of readns with a possibility to select an estimation file by clicking on the button with the three dots, and the possibility of selecting yes or no for prevoppoint.

Output parameters consists of selectors for save and oppoint. The options for save are selected, lvlpub, lvl, allpub, all, and default. Only lvlpub and lvl makes the text area nestlvl usable. Oppoint have the options rawfile, logfile, and no. Note the similarity with the top of the corresponding area in Output in Advanced Transient Analysis Options, shown in Figure 2.18.
Convergence parameters only consists of the options yes or no for restart.

Annotation parameters consists of the options no, title, sweep, status, and steps for annotate and yes or no for stats, like in Miscellaneous in Advanced Transient Analysis Options, shown in Figure 2.19

DC Analysis

When selecting DC as Analysis type and marking Enable DC analysis the window will look like Figure 2.25.

![Figure 2.25 Analyses - DC](image)
DC analysis have the option Save DC operating point on the right side of Enable DC analysis. Like AC analysis, DC analysis consists of the areas Sweep variable and Sweep range.

When comparing Figure 2.20 and Figure 2.25 it becomes clear that the only thing that differs is in Sweep variable where the Frequency is missing in DC analysis compared to AC analysis.

The Advanced button in the lower right corner opens the window shown in Figure 2.26.
When comparing Advanced DC Analysis Options in Figure 2.26 with Advanced AC Analysis Options in Figure 2.24 both similarities and differences are found. They have the same four areas, but none of the areas are identical.

State-file parameters consists in Figure 2.26 of an option for force to be none, node, dev or all, and text areas for readns, readforce, write, and writefinal.

Output parameters in Figure 2.26 has instead of the selector for oppoint in Figure 2.24 the options yes or no for print and check, respectively.

Convergence parameters is the area that differs the most. Advanced DC Analysis Options have the options for restart like Advanced AC Analysis Options, but also, above that one the options gmin, source, dptran, none, and all for homotopy, and below text areas for maxiters and maxsteps.

Annotation parameters in the bottom only have the selectors for annotate in DC, but for both annotate and stats in AC.

Noise analysis

When selecting Noise in the list for Analysis type and then marks Enable Noise Analysis, the window will look like in Figure 2.27.
Tests

Noise analysis have, as seen in Figure 2.27, three areas. They are Sweep variable, Sweep range, and Input/Output. The first two are identical to the ones in AC analysis, but Input/Output is a new one. Input/Output consists of a list for Output noise with the options Probe and Voltage, and a list for Input noise with the options Port, Current, and Voltage. When Probe and Port are selected, as in Figure 2.27, the corresponding text areas are Output probe and Input port source, respectively. When
changing to **Voltage** and **Current** it will instead look like in Figure 2.28 where the text areas are **Positive output node**, **Negative output node** and **Input current source**. When selecting **Voltage** for **Input noise**, that text area becomes **Input voltage source**.

![Figure 2.28 Input/Output](image)

Using the **Advanced** button in the lower right corner of Figure 2.27 opens a window that looks like the one in Figure 2.24, but with the text **Advanced Noise Analysis Options** at the top.

**Sensitivity analysis**

When selecting **Sensitivity** from the list with **Analysis types** and then marking **Enable sensitivity analysis** the window will look like in Figure 2.29.
Figure 2.29 Analyses - Sensitivity

Sensitivity analysis consists of the areas For analysis, Outputs, and Sensitivity parameters. For analysis contains boxes for DC operating point, DC, and AC. Outputs only consists of a large text area, and Sensitivity parameters contains the options Tabular or List for Sensformat. Note that there is no Advanced button for Sensitivity analysis.

XF analysis

When selecting XF from the Analysis type list and marking Enable XF analysis, the window looks like Figure 2.30.
**FIGURE 2.30 ANALYSES - XF**

**XF analysis** consists of the three areas **Sweep variable**, **Sweep range**, and **Output**.

**Sweep variable** and **Sweep range** are identical to the ones for **AC analysis**, and **Output** consists of the possibility to select between **Voltage** and **Probe** with corresponding text areas. The options in **Output** work identical to the ones in the **Input/Output** area of **Noise analysis**.
Using the **Advanced** button opens a window that looks like the one in Figure 2.24, except for the **Output parameters** area, which instead looks like in Figure 2.31.

![Figure 2.31 Part of Advanced XF Analysis Options](image)

**Output parameters** consists of the same elements as in Figure 2.24, but with the selection of **source** or **nodes and terminals** for **stimuli**.

**S-parameter analysis**

When selecting **S-parameter** from the list of **Analysis types** and then marking **Enable S-parameter analysis**, the window will look like in Figure 2.32.
S-parameter analysis in Figure 2.32, has the two upper areas equal to the ones in AC analysis in Figure 2.20. The third one, Do noise, only consists of a selection between yes and no.

The Advanced button opens a window that are equal to the one in Figure 2.24, except for the parts visible in Figure 2.33.
Output parameters consist of a text area for file and the same options for oppoint as in Figure 2.24.

As seen in Figure 2.33, there is also an area named Noise parameters with a text area for reftemp defined in C (Celsius).

2.2.6 Sim Options

When clicking on the next tab, Sim Options, the window in Figure 2.34 appears.
Sim Options consists of the two areas Temperature parameters and Nodes/currents save list.

Temperature parameters consists of two text areas, one for temp and one for tnom, plus a selection between vt, tc, and all for tempeffects.

Nodes/currents save list has two boxes at the top, one for Save all pub nodes and one for Save all currents. Below them is a large text area with, in this case, the nodes.

2.2.7 Measures

When selecting the tab Measures, the window in Figure 2.35 appears.
**Figure 2.35 Measures**

**Measures** consists of two parts, at the top is **Measures** and at the bottom **OCEAN Script**. In the upper left corner is a list of variables to measure, the arrows are used to go up and down in the list, respectively. The **Add** button have the three options **Artist Calculator**, **MATLAB Measure** (disabled in this case), and **OCEAN Script**. The area **OCEAN Script** contains a large text area with some code, and a box for selecting **Preserve domain data** or not.
2.2.8 Run

The tab Run consists of four tabs named Options, Status, Results, and Waveforms.

Options

When selecting Options, the window in Figure 2.36 appears.

![Figure 2.36 Options](image)

Options consists of three areas, Simulation options, Job distribution, and Overrides.

Simulation options, in the upper left corner of Figure 2.36, contains boxes for Keep sim directories, Generate netlist, Keep RAW files, Measures-only, and Keep netlist files.
Job distribution contains the option of using Run distributed and an Options button. When clicking on that button, the window in Figure 2.37 opens containing two tabs, one named Distribution and the selected one named Other. Common for both of them are the three buttons at the bottom of the window.

![Options Window](image)

**Figure 2.37 Other**

Other consists of three areas named Raw file location, Job completion, and Defaults and at top boxes for enabling warnings. They are Warn when setting active project and current project is open, Warn when automatic renaming is done, and Warn when sweep steps exceed the suggested limit.

Raw file location contains the selection between Use default and Specify location.
Job completion consists of option boxes for Display alert dialogue, Flash all windows, Play sound, and Show estimated time left for job.

Defaults have a box that makes it possible to Allow MATLAB measures and plotting, a list for Default waveform viewer which in this case only have AWD as option and at the bottom a selection box for Auto-flatten data file nodes in Results tab.

![Figure 2.38 Distribution list](image)

When selecting the Distribution tab, a window that is partly seen in Figure 2.38 appears. The part missing is the row of buttons at the bottom. Distribution have at top the selection between Distribution list and Queue command. In Figure 2.38 and Figure 2.39 is Distribution list and Queue command selected, respectively.

Distribution list has one area named Distributed machine list with a text area for Machine file with a connected selector (the button with three dots), and an area for Machine file contents. The Machine file contents is in this case empty, but data can be added by using the only enabled button and typing information in the text areas beneath Host Name, Local Directory Prefix, and Remote Directory Prefix. Note the scroll bar at the bottom and that the last part of Remote Directory Prefix is missing in the figure.
When selecting **Queue command** the upper part of the window will look like in Figure 2.39. **Queue command options** is the only area, and it contains a text area for **Command** and option boxes for **Extra quotes around arguments**, **Queue command sets working directory**, and **Single script to batch queue command**.

The last area in Figure 2.36, **Overides**, tells the user that it is possible to use overrides to specify an overriding design location to be used when running this test. Below that text is an option box for selecting if or not to **Use override**. When selecting override the text areas for **Library**, **Cell**, and **View** become enabled.

**Status**

The **Status** tab looks like in Figure 2.40, and it contains information about **Status**, **Host**, **Simulator**, date and time for start (named **Start** and **Start time**), **Elapsed time**, **CPU time**, **Memory used**, **Output files size**, and **Run dir**.
It also have buttons to **View Netlist**, **View Perl**, **View Output**, **View Errors**, **View Other**, and **Open Xterm**. The first four buttons opens text files with the corresponding data. **View Other** opens the *run directory* and makes it possible to open another file, and **Open Xterm** opens an *Xterminal*.

**Results**

Next tab in **Run** is **Results**, and one example of that tab is seen in Figure 2.41.
Figure 2.41 Results

The three buttons to the right of the text Measures is from the left Export HTML, Print, and Save as CSV File. In this case the big area in the middle only contains one include. Below that area is the information about Results location, and the buttons Plot, Format, OP Point, and Sensitivity. Plot contains the options Overlay in current plot, Plot in new window, and Clear, then plot. Format opens the window seen in Figure 2.42.
Format selected cells consists of two tabs, one named Format and one named Misc. In the Format tab, seen in Figure 2.42, it is possible to select Format style from a list. The options are General, Scientific, Engineering, Hexadecimal, and Percent. General, Scientific, and Percent only have the text area for Precision. Hexadecimal has no text area at all, and Engineering have both the text area for Precision and a Suffix list with the options Auto, T, G, M, k, m, u, n, p, f, and a where the letters stands for terra, giga, mega, kilo, milli, mico, nano, pico, femto, and atto.

When selecting the tab Misc, the window in Figure 2.43 opens.
Misc have an area for **Number of values to display per cell** with the options of **Limit number to** and **Unlimited number**. The arrows in the text area are used to increase and decrease the number in the area, respectively. When **Unlimited number** is marked the text area becomes disabled.

**Waveforms**

The last tab in **Run** is **Waveforms**, as shown in Figure 2.44.
Waveforms consists of a large text area for **Signals to plot**, a button for **Copy Save List**, and an area for **Plot actions**. **Plot actions** have a selector for **Plot layout** between **Strip plot** and **Overlay plot**, plus option lists for **Plot viewer**, **Plot analysis**, and **Plot window**. **Plot viewer** have, in the case in Figure 2.44, only **AWD** as option. **Plot window** have the options **Overlay in current plot**, **Plot in new window**, and **Clear, then plot**. In **Plot actions** there is also a button for **Plot**, which in this case is disabled.

### 2.3 Sweeps/Corners

When selecting `testDCM_Synopsys_timing_TABLE_sweep` in **Sweeps/Corners**, shown in Figure 2.2, the window in Figure 2.45 opens, containing the tabs **Sweep**, **Monte Carlo**, **Run Options**, **Job Status**, and **Results**.
2.3.1 Sweep

At top the tab **Sweep** has a text area for **Sweep over tests**, containing a list of the tests in the workspace. The ones in Figure 2.45 are the same as the ones included in the open folder **Tests** in Figure 2.2. The one that is marked is the same one that is considered in Section 2.2.

The **Templates** button opens a new window that has striking similarities to the one in Figure 2.8, but this one have the options **Sweep** and **Corner** for the **New** button, and is not involving what simulator that is used, as in Figure 2.9.
Sweep parameters is a list of the parameters that are varied. In the case in Figure 2.45, the Type From is used to specify between which values the sweep are performed, and the size of the steps, by using the text areas after From, To, and By. When clicking on the text From, a list of options opens. The options are List, From, Log, Log_Decade, and Log_Octave.

List only have one large text area, as shown in Figure 2.46, and Log, Log_Decade, and Log_Octave all have To and Pts.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Values</th>
<th>Alterable</th>
</tr>
</thead>
<tbody>
<tr>
<td>$dcn_slope</td>
<td>List</td>
<td></td>
<td>Auto</td>
</tr>
<tr>
<td>$dcn_load_cap</td>
<td>Log</td>
<td>To</td>
<td>Pts</td>
</tr>
</tbody>
</table>

Figure 2.46 Sweep parameters (List and Log)

Alterable has the options Auto, Yes, and No.

2.3.2 Monte Carlo

When selecting the tab Monte Carlo the window in Figure 2.47 appears. If the box Enable at the top is not selected nothing in the tab is enabled. To the left is a list named For simulator, which in this case only has Spectre as option. To the right side are three text areas, which all have
arrows to increase or decrease the number. The text areas are Seed, Start, and Number of Runs. At the bottom is a selector for Statistical variation with the options Process, Mismatch, and All.

### 2.3.3 Run Options

![Figure 2.48 Run Options](image)

The tab Run Options in Figure 2.48 has a lot in common with Figure 2.36, the tab Options in the tab Run in spectre Test Setup. When comparing them it is obviously that they have the same areas, but Run Options has the area Overrides at top instead of below the other two.

Overrides has the same contents in both, except that in Options the text area for Cell is included. There is also a slight difference in the text at the top of the area, in Options the text ends with *when running this test* and in Run Options it ends with *when running a test.*
The only difference in the Job distribution areas is that it in Run Options a text area for Max jobs exists. The button Options opens the window seen in Figure 2.37 to Figure 2.39.

Simulation Options contains in both figures boxes for Keep sim directories, Generate netlist, Keep RAW files, Measures-only, and Keep netlist files. In Run Options there are also boxes for Stop on first failure and Run incrementally.

2.3.4 Job Status

![Figure 2.49 Job Status](image)

The tab Job Status, shown in Figure 2.49, is only one large area. At top is a text area containing information about Host, Status, Completed, Count, and Simulator. Parts of this information is the same as in Status in the tab Run in spectre Test Setup, shown in Figure 2.40, but in this case it is possible with information about more than one run.

The text area for Run dir looks the same in both figures, but with different information.
When comparing the buttons it becomes clear that except **Waveform** in **Job Status** the buttons are the same (**View Netlist**, **View Perl**, **View Output**, **View Errors**, **View Other**, and **Open Xterm**) but slightly different placed.

### 2.3.5 Results

![Figure 2.50 Results](image)

The tab **Results** contains only one large text area for **Results**, and three buttons named **MC Plot**, **Plot**, and **View**. Some similarities with **Results** in the tab **Run** in **spectre Test Setup**, shown in Figure 2.41, are found but not as striking as in the two last figures. They both have a button for **Plot** and information about where the results are found.

When clicking on the text in the text area, the window in Figure 2.51 opens containing the results for the sweep. Note that only a part of the results are visible in the window.
Figure 2.51 Results for sweep

The text **File** at the top is a button that opens a list that is divided in four parts. The first one contains the possibilities to **Save as**, **Export to HTML**, and **Print**. The next one have the options **Hide Domain Data**, **Auto Refresh**, and **Refresh**, where the first two are marked. The third part is only **Properties**, and the last one is **Close**.

Beneath **File** are the **Parameters**. The numbers at top give the position in the simulation. The parameters that are varied are **dcm_load_cap** and **dcm_slope**. They are selected and given their range and step size in the area **Sweep parameters** in Figure 2.45.

Next comes the **Results**, which are found in the third area from top in Figure 2.3. Note that they are arranged alphabetically and not as in _testDCM_testDCM.txt_.

At the bottom are the buttons **Waveform**, **Statistics**, **OP Point**, **Format**, **Close**, and **Help**. **Statistics** opens the window shown in Figure 2.52, and **Format** opens the windows shown in Figure 2.42 and Figure 2.43.
Figure 2.52 Results Statistics

Results Statistics contains a statistical summary of the Results for sweep. In the top area are the statistical variables stated. They are Minimum, Maximum, Average, Median, Std. Dev. (Standard Deviation), and Variance. Beneath them is that information for all parts in Results.

The File button contains the same list as the one in Figure 2.51, and the Format button is also identical to the one found there.

2.4 Spectre DCM Setup

When opening testDMC, which is found in Model Generators in Figure 2.2 the window in Figure 2.53 appears. This is the Spectre DCM Setup, which consists of the tabs Design, Function, Create, Verilog-A, Verilog-D, Synopsys, Defaults, Sweeps, and Options.
2.4.1 Design

In the tab **Design**, one selects the **Library** for the design that is to be tested, as well as the **Cell**, the **View**, and the **Pins**.

At the bottom of this window are the buttons for **Generate**, **Generate & Run**, **OK**, **Cancel**, **Apply**, and **Help**. These are visible for all tabs and therefore possible to use at any time. In Figure 2.53, **Apply** is disabled because nothing is changed since the last time it was used.
2.4.2 Function

When selecting **Function**, Figure 2.54 is shown. In this window, it is possible to select what **Function** to be tested, and the **Type** for the **Pins**, as well as voltage levels. In this case an inverter is tested, but there are several digital components to select from and also some analog, converter and PLL as well as one named undefined. When the choice is done, it is most likely that **DCM Help** appears with information about the function and its possible and required pins.
2.4.3 Create

![Figure 2.55 Create](image)

The tab **Create**, shown in Figure 2.55, consists of a list of possible options for the given simulator. In this case it is Spectre and the options are **Characterization**, **Verilog-A model**, **Verilog-D model**, and **Synopsys model**. In this case all of them are marked. **Characterization** needs to be marked if any sweeps are going to be performed, and the other three ones enables the corresponding tab.
2.4.4 Verilog-A

As shown in Figure 2.56, the tab Verilog-A consists of several simulation options. For **Timing model method**, the options are **2D table**, **3D table**, **4D table**, **Input**, and **Output**. When selecting **2D table**, only the Y is available, containing the options **Vdd voltage**, **Temperature**, **Input slope**, and **Output load**. Selecting **3D table** also gives X, and **4D table** also gives W, X and W have the same options as Y.

**Input capacitance model method** also have the options **2D table**, **3D table**, and **4D table**, but it has as well **Single** and **Unused**. Even in this case **2D table** gives only Y, **3D table** gives both Y and X and **4D table** gives all three. The options for Y, X and W are **Vdd voltage**, **Temperature**, and **Output load**.
2.4.5 Verilog-D

For Verilog-D, shown in Figure 2.57, the options for Timing model method are Specify block, Input, and Output. All of them have the same text areas, Default and Verification tolerance.

For Model options one can freely select Precision, and if or not Mixed signal, Calibrate, and Verify are to be considered. In this case Verify is disabled.
2.4.6 Synopsys

The tab Synopsys consists of five areas, but only the first three ones are named. They are Characterization and modelling method, Timing constraint characterization and modelling method, and Input capacitance characterization and modelling method.

For Characterization and modelling method there are several options as shown in Figure 2.58. One can select between Timing only and Timing and Power, and if or not Output load, Temperature, Input slope, Vdd voltage, and Process are to be considered.

The light gray area of the window is Timing constraint characterization and modelling method. It has option boxes for Output load, Temperature, Input slope, Vdd voltage, Related slope, and Process.

Input capacitance characterization and modelling method has the options Temperature, Vdd voltage, and Process.
2.4.7 Defaults

The **Defaults** tab has, as shown in Figure 2.59, two separate areas. The left one is **Options** containing text areas for **Transient time step**, **Transient maximum time-step** (empty in this case), **Stimulus interval**, **Stimulus repeat**, and, in light gray, **High voltage**, **Low voltage**, and **Threshold voltage**, all of them predefined. At the bottom of that area are three options (only one possible at a time) for **Save nodes** and they are **All**, **Design**, and **Measure**.

To the right are **Parameters** containing text areas for **Vdd voltage** (predefined and therefore light gray), **Temperature**, **Input slope**, and **Output load**.
As stated at the top in the Sweeps tab in Figure 2.60, one must select parameters on the Verilog-A or Synopsys tabs to enable sweeps. There are four sweep possibilities; Vdd voltage, Temperature, Input slope and Output load. All of them has the possibility to select List, as for Vdd voltage, or From as for the other ones. List only got one text area, but From got three ones named From, To and By. At the bottom it is possible to select if the Sweep run should be for All or Min/Typ/Max.
2.4.8 Options

In the Option tab, there are two kinds of options, as seen in Figure 2.61. Plan options at the top, with the options Run sweeps incrementally, Stop at first failure, Overwrite DCM schematic, Keep waveform files, Sweeps created by Plan, Keep simulation directories, Distribute plan simulations, and Create plan subdirectories. Below these options is, in light gray, a possibility of Maximum jobs, which can be set to a value but not in this case.

At the bottom is Post-run options with a possibility to select Append to file for batch use, and when marked one can select a File.

2.5 Plans

In Figure 2.2 the two plans in this project are visible. They are _testDCM_run_plans, shown in Figure 2.62, and
_testDCM_Synopsys_model_calibration, shown in Figure 2.63, Figure 2.64, and Figure 2.65.

The simple program _testDCM_run_plans uses _testDCM_synopsys_model_calibration after opening the project and loading the project parameters and workspace parameters.
Figure 2.63 shows that _testDCM_Synopsys_model_calibration_ in the beginning opens the project, and loads project parameters as well as workspace parameters. After that, it creates all sweeps that are listed in Sweeps/Corners in Figure 2.2, starting from the bottom and going up. The first sweep that is created is the one described in Section 2.3, and when looking at Figure 2.45, the Parameters and their Values are the same as on line three and four of the first create sweep section.
Figure 2.64 Middle part of _testDCM_Synopsys_model_calibration

In Figure 2.64, the code runs the sweeps and prints before each one a line telling what it is up to. Next, it calibrates the synopsys model by creating synopsys operation conditions, creating synopsys tables for the one considered in Section 2.3 and (as shown in Figure 2.65) creates kfactors.
Figure 2.65 Lower part of _testDCM_Synopsys_model_calibration

At last in _testDCM_Synopsys_model_calibration it prints that the plan execution is finished.
3 A Layout Generator

The layout generator is a tool for automatic layout design when the design is completed in schematic. Using the information about the transistors that are given in the schematic, the layout generator constructs a layout for the cell.

3.1 Testing and proposed modifications

During the tests of the layout generator it became clear that some modifications of the code was necessary. A new requirement for the resulting layout was added as well. It is preferred that the transistors are placed close to the middle, instead of close to vdd and vss as the given code does, this to make the length of the connection paths as short as possible. There is also some work left to make the generation of transmission gates to work as supposed.

3.2 General code description

The code consists of eight files, Main.il, which starts the program, Rules.il which contains all the rules that are used, MiscProcedures_new.il, which contains all procedures that do not fit in the other files, GroupSortProcedures.il, which contains procedures that sorts the groups, PathDreawingProcedures.il, which creates the paths, GlobalRoutingProcedures.il, which does the global routing, TransAndContLayoutProcedures.il, which does the layout for the transistors and contacts, and PlacmentProcedures.il, which does the placement.

CreateCell (in Main) is the main procedure, which calls all the others and constructs the cell.

3.2.1 Code in executed order

Main starts by loading all the files and then CreateCell is started. CreateCell begins by opening the cellview followed by attaching the techfile and create a node list. Next, it makes objects of the transistors and performs general grouping. This means that it makes an object of each transistor and puts them in a list called TransistorList. For each transistor object the name (if given) is specified along with the width, the length, the
rotation, and what node that is the first and the last one. While doing that, it uses \texttt{FitToGrid} from \texttt{MiscProcedures\_new} (MP).

If \texttt{KeepFlag} is set to \texttt{nil} the next steps are grouping (that groups the transistors into basic circuits) and sorting algorithm. If \texttt{KeepFlag} is set to \texttt{t} they are not performed.

In grouping the function \texttt{BuildGroup} from MP is used. \texttt{BuildGroup} uses the functions \texttt{FindNext1} and \texttt{FindNext2}, which both are found in MP.

Sorting algorithm uses \texttt{SortGroup} from \texttt{GroupSortProcedures} (GSP), \texttt{GenerateWireList} from MP, \texttt{SimAn} from \texttt{PlacementProcedures} (PP), and \texttt{CalcAll} from \texttt{GlobalRoutingProcedures} (GRP).

\texttt{SortGroup} uses \texttt{Duplicate}, \texttt{SortGroupFindNext}, and \texttt{SortGroupAdvanced}, all from GSP like all functions under them.

\texttt{Duplicate} uses no function, and \texttt{SortGroupFindNext} only calls itself, but \texttt{SortGroupAdvanced} uses \texttt{FindAndBindParallel}, \texttt{FindAndBindSerial}, \texttt{FindAndInsert}, \texttt{FindAndBindSemiParallel}, \texttt{BindAll}, \texttt{RotateTransistors}, and \texttt{Duplicate}.

\texttt{FindAndBindParallel} uses \texttt{Permutations}, \texttt{DuplicateBranches}, \texttt{RotateTransistors} and \texttt{BindParallel}. \texttt{Permutations} uses itself and \texttt{Accumulate}, which also uses itself. \texttt{DuplicateBranches} uses \texttt{DuplicateBranch}, which uses \texttt{Duplicate}. \texttt{RotateTransistors} uses no function, and \texttt{BindParallel} only uses \texttt{RotateTransistors}.

\texttt{FindAndBindSerial} that follows \texttt{FindAndBindParallel} in \texttt{SortGroupAdvanced} only uses \texttt{BindSerial}.

Next in \texttt{SortGroupAdvanced} is \texttt{FindAndInsert}, which uses \texttt{DuplicateBranches}, \texttt{DuplicateBranch}, \texttt{RotateTransistors}, and \texttt{TryToInsertInBranch}. \texttt{DuplicateBranches} uses \texttt{DuplicateBranch}, which uses \texttt{Duplicate} and \texttt{RotateTransistors}, and \texttt{TryToInsertInBranch} uses no function.

After \texttt{FindAndInsert} comes \texttt{FindAndBindSemiParallel}, which uses \texttt{BindSerial} and \texttt{RotateTransistors} (that neighter uses any function), and after that comes \texttt{BindAll}. \texttt{BindAll} calls \texttt{Combine}, \texttt{DuplicateBranch}, \texttt{Duplicate}, \texttt{Permutations}, and \texttt{RotateTransistors}. \texttt{Combine} calls
**General code description**

InsertAll and itself, InsertAll calls itself and Insert, which only calls itself. DuplicateBranch only calls Duplicate, and Permutations calls itself and Accumulate. Accumulate only calls itself.

Next, SortGroupAdvanced calls RotateTransistors and Duplicate, which neither calls any other function.

Next in sorting algorithm is GenerateWireList, which uses NOfP from MP.

After GenerateWireList comes SimAn from PP. SimAn uses CalcCost from PP, which uses CalcAll from GRP, and CalcAll uses NewCost and CheapestTree, both from GRP. After CalcCost comes NotAnnealed, NewTemp, and NumberOfMoves, all three from PP.

After SimAn comes CalcAll from GRP, which uses NewCost and CheapestTree, both from GRP.

The final step in CreateCell is layout, which uses functions from other files.

The first function in layout is MakeSubstrateContacts, which is found in MP. MakeSubstrateContacts uses FitToGrid from MP several times and have a big if-case that checks if Metal 2 is used as a supply layer or not, and performs several activities if it is.

The second function in layout is AddContactInfo from MP.

The third is UpperAndLowerBound, which also is in MP. UpperAndLowerBound uses MinDiffMetalEnc, which only uses FitToGrid (both from MP).

Next in layout is AddTransistors from TransAndContLayoutProcedures (TACLP). AddTransistors uses FitToGrid a lot, but also NotInterfering from MP and MakeTransistors from TACLP. NotInterfering is only a small test function that tests if a path starts or ends at a given node. MakeTransistors uses FitToGrid and CreateTransLayers, both from MP. CreateTransLayers only puts an extra layer on the transistors.

The next function in layout is AssignPins from MP. AssignPins uses AssignPin from MP. AssignPin uses PassingPaths (from MP) and itself.
**PassingPaths** is a function that calculates how many paths that passes a given x-coordinate.

Next in layout comes **CalcStartOrigin** from MP. **CalcStartOrigin** only uses **FitToGrid**.

The next function in layout is **AddTransistors** that has been used before.

After that comes **MakeContacts** from TACLP. **MakeContacts** uses **MakeTransContact** from TACLP, **NotInterfering**, **FitToGrid**, **M_Poly** from TACLP and **FitToGrid** from MP, all of them more than once. **MakeTransContact** uses **FitToGrid** several times, but also **Bin** from MP. **M_Poly** calls no function.

Next function in the layout part of **CreateCell** is **MakeSubstrateContacts** from MP. **MakeSubstrateContacts** uses **FitToGrid** several times.

Next comes **ExtraMetalOnSupplyContacts** from MP, which only uses **FitToGrid**.

**MakeContactList** from MP is the next function, and after that comes **ExtraMetalOnGatePin** from MP. **ExtraMetalOnGatePin** only uses **FitToGrid**.

Next in layout is **CreateAbovePBelowNPaths**, and shortly after that **CreateBelowPAboveNPaths** and **CreateBetweenPaths**, all found in **PathDrawingProcedures** (PDP). All of them uses **FitToGrid** and the two latter ones also uses **GeneratePtsListP** and **GeneratePtsListN**, both of them in PDP. Both **GeneratePtsListP** and **GeneratePtsListN** uses **Bin** and **FitToGrid**, both from MP.

After **CreateBetweenPaths** comes **CalcCellWidth** from MP, and after adding supply wires comes **CreateTransToSupplyPaths** from PDP.

Found last in MP is **CellLayers**, which uses **FitToGrid**. At last **CreateCell** calls itself.

### 3.2.2 Visual code description

Figure 3.1 to Figure 3.12 shows the code, as described above in an visual way. Note the arrow that indicates that a function uses itself, and that if one function uses another more than one time only the first is showed in the figures.
Figure 3.1 The code structure of CreateCell
Figure 3.1 shows the highest levels of the tree structure of the code. **SortGroup** is root in Figure 3.2, **SimAn** in Figure 3.3, and so on ending with **CreateBetweenPaths** in Figure 3.9.

![Figure 3.2 SortGroup](image)

**Figure 3.2 SortGroup**

Figure 3.2 shows the code structure of **SortGroup**, and **FindAndBindParallel**, **FindAndInsert**, and **BindAll** are seen in Figure 3.10, Figure 3.11 and Figure 3.12.

![Figure 3.3 SimAn](image)

**Figure 3.3 SimAn**
Figure 3.3 shows all functions from SimAn and down,

UpperAndLowerBound → MinDiffMetalEnc → FitToGrid

*Figure 3.4 UpperAndLowerBound*

Figure 3.4 shows the part of the function tree that has UpperAndLowerBound as root.

AddTransistors

FitToGrid

NotInterfearing

MakeTransistor

FitToGrid

CreateTransLayers

*Figure 3.5 AddTransistors*

In Figure 3.5 AddTransistors and the functions beneath are shown.

AssignPins

AssignPin

PassingPaths

*Figure 3.6 AssignPins*

In Figure 3.6, the part of the tree that has AssignPins as root, is shown.
Figure 3.7 MakeContacts

Figure 3.7 shows the tree from MakeContacts and down.

Figure 3.8 CreateBelowPAboveNPaths

Figure 3.8 has CreateBelowPAboveNPaths as root.

Figure 3.9 CreateBetweenPaths

Figure 3.9 is identical to Figure 3.8, except for the root.
Figure 3.10 FindAndBindParallel

Figure 3.10 is the first one that is a continuation of Figure 3.2.

Figure 3.11 FindAndInsert

Figure 3.11 shows all functions that are beneath \textbf{FindAndInsert}.

Figure 3.12 BindAll

Figure 3.12 is the final figure representing the code. Note the similarities between the final three ones that all comes from \textbf{SortGroupAdvanced}.
### 3.3 The functions in the code

Table 3.1 lists all functions in the code and in which file they are found, as well as which functions they are used by and uses.

<table>
<thead>
<tr>
<th>Name</th>
<th>Found in</th>
<th>Used by</th>
<th>Uses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accumulate</td>
<td>GroupSortProcedures</td>
<td>Accumulate, Permutations</td>
<td>Accumulate</td>
</tr>
<tr>
<td>AddContactInfo</td>
<td>MiscProcedures_new</td>
<td>CreateCell</td>
<td>None</td>
</tr>
<tr>
<td>AddTransistors</td>
<td>TransAndContLayoutProcedures</td>
<td>CreateCell</td>
<td>FitToGrid, MakeTransistors, NotInterfering</td>
</tr>
<tr>
<td>AssignPin</td>
<td>MiscProcedures_new</td>
<td>AssignPins</td>
<td>AssignPin, PassingPaths</td>
</tr>
<tr>
<td>AssignPins</td>
<td>MiscProcedures_new</td>
<td>CreateCell</td>
<td>AssignPin</td>
</tr>
<tr>
<td>Bin</td>
<td>MiscProcedures_new</td>
<td>ExtraMetalOnGatePin, GeneratePtsListP, GeneratePtsListN, MakeTransContacts</td>
<td>None</td>
</tr>
<tr>
<td>BindAll</td>
<td>GroupSortProcedures</td>
<td>SortGroupAdvanced</td>
<td>Combine, Duplicate, DuplicateBranch, Permutations, RotateTransistors</td>
</tr>
<tr>
<td>BindParallel</td>
<td>GroupSortProcedures</td>
<td>FindAndBindParallel</td>
<td>RotateTransistors</td>
</tr>
<tr>
<td>BindSerial</td>
<td>GroupSortProcedures</td>
<td>FindAndBindSemiparallel, FindAndBindSerial</td>
<td>None</td>
</tr>
<tr>
<td>BuildGroup</td>
<td>MiscProcedures_new</td>
<td>CreateCell</td>
<td>FindNext1, FindNext2</td>
</tr>
</tbody>
</table>
The functions in the code

<table>
<thead>
<tr>
<th>Name</th>
<th>Found in</th>
<th>Used by</th>
<th>Uses</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalcAll</td>
<td>GlobalRouting-Procedures</td>
<td>CalcCost, CreateCell</td>
<td>CheapestTree, NewCost</td>
</tr>
<tr>
<td>CalcCellWidth</td>
<td>MiscProcedures_new</td>
<td>CreateCell</td>
<td>None</td>
</tr>
<tr>
<td>CalcCost</td>
<td>PlacementProcedures</td>
<td>SimAn</td>
<td>CalcAll</td>
</tr>
<tr>
<td>CalcStartOrigin</td>
<td>MiscProcedures_new</td>
<td>CreateCell</td>
<td>FitToGrid</td>
</tr>
<tr>
<td>CellLayers</td>
<td>MiscProcedures_new</td>
<td>CreateCell</td>
<td>FitToGrid</td>
</tr>
<tr>
<td>CheapestTree</td>
<td>GlobalRouting-Procedures</td>
<td>CalcAll</td>
<td>None</td>
</tr>
<tr>
<td>Combine</td>
<td>GroupSortProcedures</td>
<td>BindAll, Combine</td>
<td>Combine, InsertAll</td>
</tr>
<tr>
<td>CreateAboveP-BelowNPaths</td>
<td>PathDrawing-Procedures</td>
<td>CreateCell</td>
<td>FitToGrid</td>
</tr>
<tr>
<td>CreateBelowPAboveNPaths</td>
<td>PathDrawing-Procedures</td>
<td>CreateCell</td>
<td>FitToGrid, GeneratePtsListP, GeneratePtsListN</td>
</tr>
<tr>
<td>CreateBetweenPaths</td>
<td>PathDrawing-Procedures</td>
<td>CreateCell</td>
<td>FitToGrid, GeneratePtsListP, GeneratePtsListN</td>
</tr>
</tbody>
</table>
Table 3.1: The Functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Found in</th>
<th>Used by</th>
<th>Uses</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateCell</td>
<td>Main</td>
<td>None</td>
<td>AddContactInfo, AddTransistors, AssignPins, BuildGroup, CalcAll, CalcCellWidth, CalcStartOrigin, CellLayers, CreateAbovePBelowNPaths, CreateBelowPAboveNPaths, CreateBetweenPaths, CreateTransToSupplyPaths, ExtraMetalOnGatePin, ExtraMetalOnSupplyContacts, GenerateWireList, MakeContactList, MakeContacts, MakeSubstrateContact, SimAn, SortGroup, UpperAndLowerBound</td>
</tr>
<tr>
<td>CreateTransLayers</td>
<td>MiscProcedures_</td>
<td>MakeTransistor</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>new</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CreateTransToSupplyPaths</td>
<td>PathDrawing-</td>
<td>CreateCell</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Procedures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Duplicate</td>
<td>GroupSortProce-</td>
<td>BindAll, Duplicate-</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>dures</td>
<td>Group, SortGroup,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SortGroupAdvanced</td>
<td></td>
</tr>
</tbody>
</table>
Table 3.1: The Functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Found in</th>
<th>Used by</th>
<th>Uses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duplicate-Branch</td>
<td>GroupSortProce-</td>
<td>BindAll, DuplicateBranches, FindAndInsert</td>
<td>Duplicate</td>
</tr>
<tr>
<td></td>
<td>dures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Duplicate-Branches</td>
<td>GroupSortProce-</td>
<td>FindAndBindParallel, FindAndInsert</td>
<td>DuplicateBranch</td>
</tr>
<tr>
<td></td>
<td>dures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ExtraMetalOn-GatePin</td>
<td>MiscProcedures_</td>
<td>CreateCell</td>
<td>Bin, FitToGrid</td>
</tr>
<tr>
<td></td>
<td>new</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ExtraMetalOn-SupplyCon-</td>
<td>MiscProcedures_</td>
<td>CreateCell</td>
<td>FitToGrid</td>
</tr>
<tr>
<td>tactss</td>
<td>new</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FindAndBind-</td>
<td>GroupSortProce-</td>
<td>SortGroupAdvanced</td>
<td>BindParallel,</td>
</tr>
<tr>
<td>Parallel</td>
<td>dures</td>
<td></td>
<td>DuplicateBranches, Permuation,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RotateTransistors</td>
</tr>
<tr>
<td>FindAndBind-</td>
<td>GroupSortProce-</td>
<td>SortGroupAdvanced</td>
<td>BindSerial, RotateTransistors</td>
</tr>
<tr>
<td>SemiParallel</td>
<td>dures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FindAndBind-</td>
<td>GroupSortProce-</td>
<td>SortGroupAdvanced</td>
<td>BindSerial</td>
</tr>
<tr>
<td>Serial</td>
<td>dures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FindAndInsert</td>
<td>GroupSortProce-</td>
<td>SortGroupAdvanced</td>
<td>DuplicateBranch,</td>
</tr>
<tr>
<td></td>
<td>dures</td>
<td></td>
<td>DuplicateBranches,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RotateTransistors,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TryToInsertInBranch</td>
</tr>
<tr>
<td>FindNext1</td>
<td>MiscProcedures_</td>
<td>BuildGroup, FindNext1</td>
<td>FindNext1</td>
</tr>
<tr>
<td></td>
<td>new</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FindNext2</td>
<td>MiscProcedures_</td>
<td>BuildGroup, FindNext2</td>
<td>FindNext2</td>
</tr>
<tr>
<td></td>
<td>new</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 3.1: The Functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Found in</th>
<th>Used by</th>
<th>Uses</th>
</tr>
</thead>
<tbody>
<tr>
<td>FitToGrid</td>
<td>MiscProcedures_ new</td>
<td>AddTransistors, CalcStartOrigin, CellLayers, CreateAbovePBelowNPaths,</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CreateBelowPAboveNPaths, CreateBetweenPaths, ExtraMetalOnGatePin,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ExtraMetalOnSupplyContacts, GeneratePtsListN, GeneratePtsListP, MakeContacts, MakeSubstrateContacts, MakeTransistors, MakeTransistors, MinDiffMetalEnc</td>
<td></td>
</tr>
<tr>
<td>GeneratePtsListN</td>
<td>PathDrawingProcedures</td>
<td>CreateBelowPAboveNPaths, CreateBetweenPaths</td>
<td>Bin, FitToGrid</td>
</tr>
<tr>
<td>GeneratePtsListP</td>
<td>PathDrawingProcedures</td>
<td>CreateBelowPAboveNPaths, CreateBetweenPaths</td>
<td>Bin, FitToGrid</td>
</tr>
<tr>
<td>GenerateWireList</td>
<td>MiscProcedures_ new</td>
<td>CreateCell</td>
<td>NOfP</td>
</tr>
<tr>
<td>Insert</td>
<td>GroupSortProcedures</td>
<td>Insert, InsertAll, InsertAll</td>
<td>Insert</td>
</tr>
<tr>
<td>InsertAll</td>
<td>GroupSortProcedures</td>
<td>Combine, InsertAll</td>
<td>Insert, InsertAll</td>
</tr>
</tbody>
</table>
### Table 3.1: The Functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Found in</th>
<th>Used by</th>
<th>Uses</th>
</tr>
</thead>
<tbody>
<tr>
<td>MakeContactList</td>
<td>MiscProcedures_new</td>
<td>CreateCell</td>
<td>None</td>
</tr>
<tr>
<td>MakeContacts</td>
<td>TransAndContLayoutProcedures</td>
<td>CreateCell</td>
<td>FitToGrid, MakeTransContact, M_Poly, NotInterfering</td>
</tr>
<tr>
<td>MakeSubstrateContacts</td>
<td>MiscProcedures_new</td>
<td>CreateCell</td>
<td>FitToGrid</td>
</tr>
<tr>
<td>MakeTransistor</td>
<td>TransAndContLayoutProcedures</td>
<td>MakeContacts</td>
<td>Bin, FitToGrid</td>
</tr>
<tr>
<td>MinDiffMetalEnc</td>
<td>MiscProcedures_new</td>
<td>UpperAndLowerBound</td>
<td>FitToGrid</td>
</tr>
<tr>
<td>M_Poly</td>
<td>TransAndContLayoutProcedures</td>
<td>MakeContacts</td>
<td>None</td>
</tr>
<tr>
<td>NewCost</td>
<td>GlobalRoutingProcedures</td>
<td>CalcAll</td>
<td>None</td>
</tr>
<tr>
<td>NewTemp</td>
<td>PlacementProcedures</td>
<td>SimAn</td>
<td>None</td>
</tr>
<tr>
<td>NOfP</td>
<td>MiscProcedures_new</td>
<td>GenerateWireList</td>
<td>None</td>
</tr>
<tr>
<td>NotAnnealed</td>
<td>PlacementProcedures</td>
<td>SimAn</td>
<td>None</td>
</tr>
<tr>
<td>NotInterfering</td>
<td>MiscProcedures_new</td>
<td>AddTransistors, MakeContacts</td>
<td>None</td>
</tr>
<tr>
<td>NumberOfMoves</td>
<td>PlacementProcedures</td>
<td>SimAn</td>
<td>None</td>
</tr>
</tbody>
</table>
### Table 3.1: The Functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Found in</th>
<th>Used by</th>
<th>Uses</th>
</tr>
</thead>
<tbody>
<tr>
<td>PassingPaths</td>
<td>MiscProcedures_new</td>
<td>AssignPin</td>
<td>None</td>
</tr>
<tr>
<td>Permutations</td>
<td>GroupSortProcedures</td>
<td>BindAll, FindAndBindParallel</td>
<td>Accumulate, Permutations</td>
</tr>
<tr>
<td>RotateTransistors</td>
<td>GroupSortProcedures</td>
<td>BindAll, BindParallel, FindAndBindParallel, FindAndBindSemiParallel, FindAndInsert, SortGroupAdvanced</td>
<td>None</td>
</tr>
<tr>
<td>SimAn</td>
<td>PlacementProcedures</td>
<td>CreateCell</td>
<td>CalcCost, New-Temp, NotAnnealed, NumberOfMoves</td>
</tr>
<tr>
<td>SortGroup</td>
<td>GroupSortProcedures</td>
<td>CreateCell</td>
<td>Duplicate, SortGroupAdvanced, SortGroupFindNext</td>
</tr>
<tr>
<td>SortGroupAdvanced</td>
<td>GroupSortProcedures</td>
<td>SortGroup</td>
<td>BindAll, Duplicate, FindAndBindParallel, FindAndBindSemiParallel, FindAndBindSerial, FindAndInsert, RotateTransistors</td>
</tr>
<tr>
<td>SortGroupFindNext</td>
<td>GroupSortProcedures</td>
<td>SortGroup</td>
<td>SortGroupFindNext</td>
</tr>
<tr>
<td>TryToInsertInBranch</td>
<td>GroupSortProcedures</td>
<td>FindAndInsert</td>
<td>None</td>
</tr>
<tr>
<td>UpperAndLowerBound</td>
<td>MiscProcedures_new</td>
<td>CreateCell</td>
<td>MinDiffMetalEnc</td>
</tr>
</tbody>
</table>
3.4 Using the code

For information about how to get started, see Appendix B in [3].

It is important to set `KeepFlag` to `nil` the first time you create a layout for a cell, otherwise the transistors will not get any rotation at all. When redrawing a layout, `KeepFlag` can be used to keep the previous rotation.

3.5 Present state of the code

No updates have been applied to the code. The only way to put the transistors closer to middle (that I can find) is to use a variable for the distance and change it in steps, until it is not possible to route the connection lines anymore. Then the previous results can be used. This method takes a long time because the program must be restarted for every rerouting.
4 Conclusions

This thesis mainly consists of a manual for Aptivia and, several different descriptions of the code that implements a layout generator.

The major problem with Aptivia was that it does not work well with older versions of Cadence, but as soon as it was tested with a new enough version it has worked without any problem. When Aptivia was running, some of its functions were tested, and the major part of it is documented in a manual. (See chapter 2.)

The layout generator was at first tested, and then documented as detailed as possible because there existed hardly any documentation at all. The code itself is commented. That was the starting point for me to understand it.

The transmission gates are still left to be redone and have not been considered. How to get all transistors as close as possible to the middle has been looked upon, but no good solution was found.
5 References


På svenska

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extra-ordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut ens- taka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

In English

The publishers will keep this document online on the Internet - or its possible replacement - for a considerable time from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

© Anna-Charlotta Klevbrink