Instructional Approach to Writing Parasitic Capacitance Rules Files Using Exact

1. Introduction

The ***Exact*** analysis stage extracts the user-required information necessary for the respective parasitic capacitances by probing the Exact database. This is performed via script files written in ***LISA*** (Language for Interfacing Silvaco Applications). This article demonstrates a systematic approach for writing analysis script files.

2. Full Working LISA script file

Firstly a full workable analysis script file is detailed; explanatory discussion then follows.

```
! ***********BEGIN LISA SCRIPT***********
!Performs numerical fit to determine near body
effect on fringe down capacitance data. Text
!following an exclamation mark is a comment.
Layouts referred to in the script are Parallel
!Plate.pml (termed PP...) and OneArray.pml
(termed OA...).

!Load in the internal database
db = DatabaseLoad(".");

!Create capacitance variables and assign capaci-
tance values to them
extract_name("PP_Ctotal", "B_gnd", "Plate");
extract_name("OA_Ctotal", "B_gnd", "L1p");

!Decide which combinations are to be examined
and included in the tables.
PP_combinations = {1};
OA_combinations = {1};

!create table for parallel plate information
table_PP= select(db, "ParallelPlate", PP_combina-
tions, {"ParallelPlatePlateWidth"},
{"PP_Ctotal"});

!change units of capacitance to fF
column_scalar_op(table_PP, "PP_Ctotal", table_PP,
"PP_Ctotal", "*", 1e15);

!save the table for reference purposes
save_table(table_PP, CSV, "PP_a.csv");

!create table for fringe down capacitance in-
formation.
table_OA= select(db, "OneArray", OA_combinations,
{"OneArrayPlateWidth",
"OneArrayLayer1Space", "OneArrayLayer1Width"},
{"OA_Ctotal"});

!change capacitance units to fF
column_scalar_op(table_OA, "OA_Ctotal", table_OA,
"OA_Ctotal", "*", 1e15);

!save output table for reference purposes
save_table(table_OA, CSV, "OA_a.csv");

!Perform operations on table_OA to obtain fringe
down capacitance.

merge(table_PP, "PP_Ctotal", table_OA);

save_table(table_OA, CSV, "OA_b.csv");

column_vector_op(table_OA, "PP_Ctotal", table_OA,
"OneArrayLayer1Width", table_OA, "OA_Carea", "*");
column_vector_op(table_OA, "OA_Ctotal", table_OA,
"OA_Carea", table_OA, "OA_FDalpha", "-");
column_scalar_op(table_OA, "OA_FDalpha", table_OA,
"OA_FD", "/", 2.0);

save_table(table_OA, CSV, "OA_c.csv");
save_table(table_OA, TONYPLOT, "OA_c.str");

!Equations for fringe down with near body ef-
fect.

equationFD="OA_FD=1.0*K1[0.01]*(1.0-exp(- K2[0.09]*
(${OneArrayLayer1Space}+K3[0.03])))";

res_OA = (calculate_fit(equationFD)(table_
OA)(sum_combinations(OA_combinations))("FD_
OA.rsm")("Downhill-Simplex"));

save_table(res_OA, CSV, "OA_FDcoeff.csv");

! Header notes
write_parameters("eg1.xcl", table_PP, {"\n// Ex-
ample script for Exact2 manual\n"});
write_parameters("eg1.xcl", table_PP, {"\n\n"});
write_parameters("eg1.xcl", table_PP, {"UNIT
LENGTH um\nUNIT CAPACITANCE fF\n\n"});

!Write text for area capacitance expression and
fringe down capacitance expression

write_parameters("eg1.xcl", table_PP, {"CAPACI-
TANCE CROSSOVER PLATE ", "LAYER0", " ", "LAYER1",
"\n\n[\n\n C =", "PP_Ctotal", "*area()\n\n]\n\n"});

write_parameters("eg1.xcl", res_OA, {"CAPACITANCE
CROSSOVER FRINGE ", "LAYER0", " ", "LAYER1", "\
n\n[\n\n C =length()*", "k1", "*(1.0-exp(-", "k2",
"*(distance()+", "k3", ")))\n\n]\n\n"});

!*************END SCRIPT FILE**************
```

Main output from the script file: capacitance rule file eg1.xcl

```
UNIT LENGTH um
UNIT CAPACITANCE fF

CAPACITANCE CROSSOVER PLATE metal1 metal2

[
 C =0.0345313*area()
]

CAPACITANCE CROSSOVER FRINGE metal1 metal2
[
C      =length()*0.0454768*(1.0-exp(-0.444801*(dis-
tance()+0.0874414)))
]
```

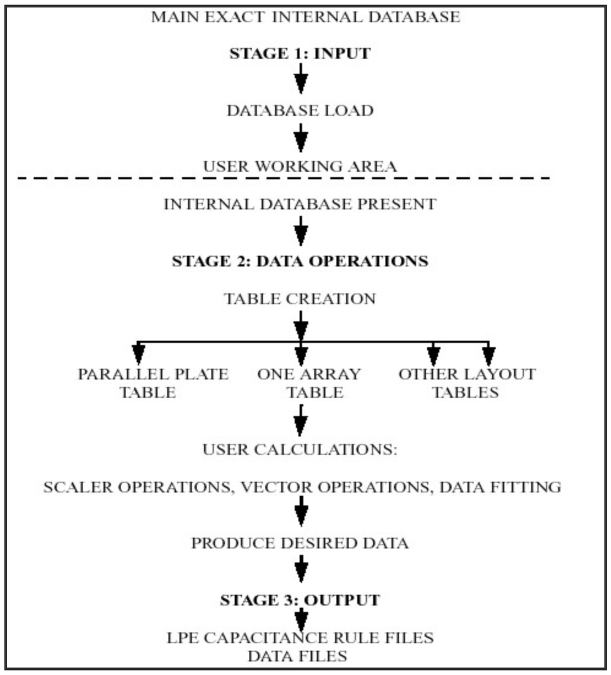


Figure 1. Schematic diagram of the stages within an analysis script

3. Script Discussion

There are three dominant stages in a full workable ***Exact*** script file, (see Figure 1):

Stage 1: input. This creates a table containing all the required data.

Stage 2: data operations. Manipulate the table in order to produce specific capacitance information.

Stage 3: output: This creates capacitance rule files for use in layout parasitic extraction (LPE) tools.

3.1 Stage 1: Input commands.

```
db = DatabaseLoad("/home/.../....etc");
or

db = DatabaseLoad(".");
```

The input stage's main aim is to build a data table or several data tables. Firstly, it is necessary to load in the database where the output from ***Exact*** has been saved. DatabaseLoad performs this task via the user specifying to it the path of the database and the name to use for a variable to store it in. This path must match that in the output stage of the ***Exact*** experiment, see Figure 2. Once the internal database has been loaded into the analysis stage, the next task of the input stage is to create a table. The ***LISA*** command used to create a table is:

```
table_OA= select(db, "OneArray",
OA_combinations, {"OneArrayPlate-
Width", "OneArrayLayer1Space",
"OneArrayLayer1Width"}, {"OA_Ctotal"});
```

The syntax must follow:

1) Name of table to create.

2) Keyword select (arguments inside the parentheses must follow)

a) Database name

b Layout name

c) Specific layout combinations

This is a sequence of integers. The analysis stage is informed what layout-specific combinations to e include in the table. We use OneArray_combinations for this purpose. In this example, OA_combinations = {1} is used since only combination 1 is present. While a string of numbers for the layout combinations argument would suffice, the use of a variable is more intuitive, especially to another user reading the script.

d) Structure information argument

This is a set of comma separated strings which must be contained within braces. The strings form a list of specific structure parameter values that the user requires to be stored in the table.

e) Capacitance list argument

This is a set of comma-separated strings contained within braces. The strings form a list of user-required capacitance values to store in the table after extraction from the database. The key word extract_name is used for this purpose:

```
extract_name("OA_Ctotal", "B_gnd", "L1p");
```

The parameters inside the parentheses (from left to right) are the user-specified capacitance name (target capacitance) to include in the table, the wire 1 name, and the wire 2 name. While the target capacitance name is chosen arbitrarily, the names of the wires must correspond to those of the respective layout.

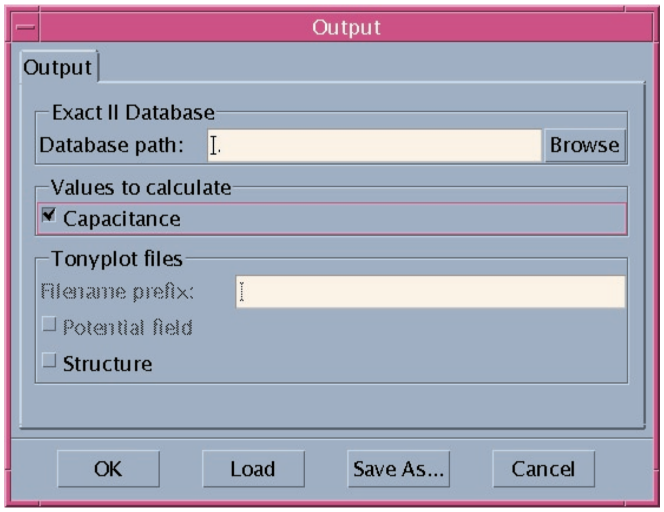


Figure 2. Output GUI from main ***Exact*** experiment.

3.2 Stage 2: Data Operations

The goal of this stage is to calculate specific capacitance effects with the generated table. This manipulation is necessary since the capacitance calculated between any two wires is the total capacitance between them. However, a user may wish to ascertain how much capacitance pertains to fringe down capacitance..

Figure 3(a) shows the ParallelPlate.pml test structure. Figure 3(b) shows the OneArray.pml test structure, representing three single conductors over a ground plane. The total capacitance calculated between L1p and B_gnd conductors in Figure 3(b) includes fringe capacitance, termed OA_FD, and area capacitance, termed OA_Carea. The total capacitance between L1p and B_gnd is writable as:

```
OA_Ctotal=OA_Carea+2 x OA_FD,                2.1
```

where the fringe capacitance is written as:

```
OA_FD=(OA_Ctotal-OA_Carea)/2.                2.2
```

OA_Carea in equation 2.2 is obtained from using the test structure of figure 3(a), where

```
PP_Ctotal = PP_ Carea.                       2.3
```

PP_Carea must be scaled by the width of L1p to obtain OA_Carea, therefore:

```
OA_Carea=OneArrayLayer1Width x PP_ Carea.2.4
```

After obtaining the area capacitance component of the total capacitance between L1p and B_gnd, the fringe capacitance component is easily calculated from Equation 2.2.

It is evident from the description above that a user must:

- identify what capacitance effect to examine
- identify which test structures are required for effect examination
- identify what respective user calculations are required

A demonstration of the ***LISA*** commands used to obtain the fringe capacitance is detailed below:

```
column_vector_op(table_OA, "PP_Ctotal", table_OA,
  "OneArrayLayer1Width", table_OA, "OA_Carea",
  "*");
```

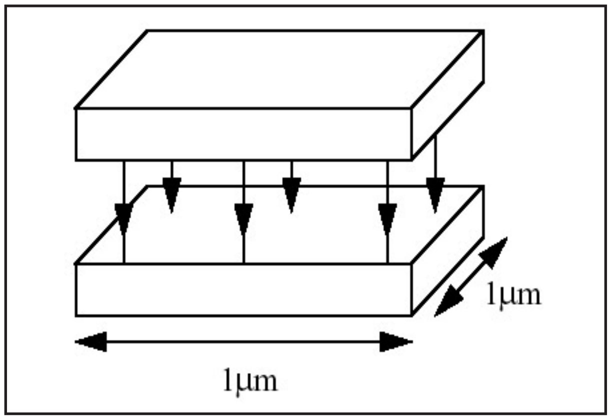


Figure 3(a). ParallelPlate.pml test structure. Capacitance effects are indicated by bold arrows.

```
column_vector_op(table_OA, "OA_Ctotal",
  table_OA, "OA_Carea", table_OA, "OA_FD_
  alpha", "-");
column_scalar_op(table_OA, "OA_FD_alpha",
  table_OA, "OA_FD", "/", 2.0);
```

The above commands highlight scalar operations and vector operations that exist in a ***LISA*** analysis script.

3.2.1 Scalar Operation

A Scalar operation involves a number operating on a quantity in the table. The for-mat for a scalar operation must follow:

1) Key word: column_scalar_op.

2) Inside the parentheses: SOURCE ADDRESS: specific table name, column in table.

3) Inside the parentheses: ADDRESS TO WHICH RESULTS ARE WRITTEN: specific table name, column in table.

4) Inside the parentheses: operation to perform

5) Inside the parentheses: number to use

3.2.2 Vector operation.

A vector operation involves a quantity in a table operating on a quantity in a table. The column_vector_op format must follow:

1) Key word: column_vector_op

2) Inside the parentheses: SOURCE 1 ADDRESS: specific table name, column in table

3) Inside the parentheses: SOURCE 2 ADDRESS: specific table name, column in table

4) Inside the parentheses: ADDRESS TO WHICH RESULTS ARE WRITTEN: specific table name, column in table

5) Inside the parentheses: operations to perform

3.2.3 Data fitting.

In addition to the capacitance effects in Figure 3(b), there are additional effects in test structure OneArray.pml (Figure 4). A comparison of the fringe capacitance in Figures 3(b) and 4 shows that some of the would be

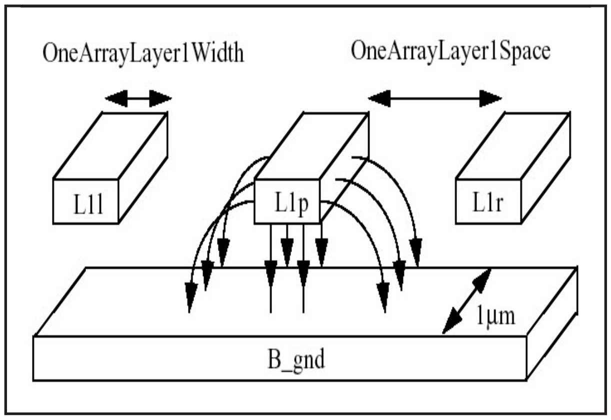


Figure 3(b). OneArray.pml layout. Capacitance effects are indicated by the bold arrows.

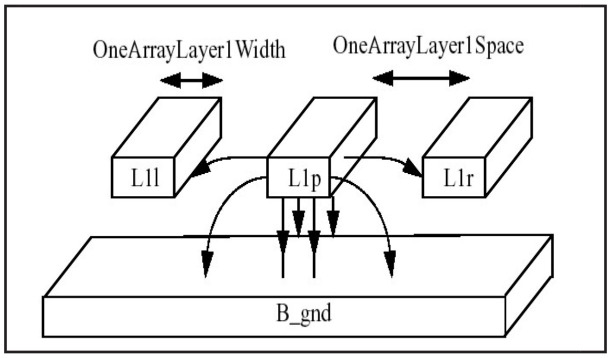


Figure 4. Test structure OneArray.pml. Capacitance effects are indicated by the bold arrows.

fringe down field lines from L1p now terminate on L1l and L1r. Since this significant only when the two conductors are sufficiently near to one another, this is termed a near body effect.

The requirement is to obtain an analytical expression for the fringe capacitance between L1p and B_gn, while taking the near body effect into account. These well-known expressions are typically obtained from the specific LPE manual, but require the process-specific capacitance coefficients obtained through ***Exact's*** fitting routines.

In this example, the expression takes the form:

```
Cfringe=K1*(1.0-exp(-K2*(distance+K3)))
```

where the coefficients K1, K2 and K3 are calculated by the fitting routine. For example, in the DOE of the ***Exact*** experiment, OneArrayLayer1Space varies from 0.1 to 5 microns (Figure 5). The capacitance between wires L1p and B_gnd is calculated over this range. Since Exact calculates the total capacitance between any two conductors, it is first necessary to obtain the values of fringe down capacitance as a function of near body distance. Once done, a column containing this capacitance information as well as column containing conductor spacing information appears in the table.

To then perform a numerical fit on the data, used the equation that is described in the ***LISA*** script. For example:

```
equationFD=”OA_FD=1.0*K1[0.01]*(1.0-exp( K2[0.0
9]*(${OneArrayLayer1Space}+K3[0.03])))”;
```

OA_FD and OneArrayLayer1Space are variables in the table that identifies, respectively, the fringe down capacitance data and the spacing between the two conductors. The variable equationFD holds the description of the equation. The conditions for fitting routine calculated coefficients are set via []. Once the equation is described, the numerical is performed with:

```
res_OA = (calculate_fit(equationFD)(table_
OA)(sum_combinations(OA_combinations))(“FD_
OA.rsm”)(“Downhill-Simplex”));
```

In this command, res_OA is used to store the values of the calculated coefficients. Within the parentheses of the key word calculate_fit are the following:

1) The name of the fitting routine equation described in the ***LISA*** script

2) The name of the table that contains the data

3) The number of combinations for the fit

4) The name of the response surface model, which in this case is FD_OA.rsm. Users can choose to not save the RSM file by giving a file name of “”.

5) The name of the chosen fitting method

On completion of the fitting routine, the user saves the file containing the coefficients by using the following ***LISA*** syntax

```
save_table(res_OA, CSV, “OA_FDcoeff.csv”);
```

A extract from this file is detailed below:

```
AVG_ERROR,K1,K2,K3,LAYER0,LAYER1,MAX_ERROR

0.963929,0.0454768,0.444801,0.0874414,metal1,metal2
,6.01101
```

The analytical expression that describes the fringe down capacitance with near body effect is plotted against the actual calculated data using these coefficients (Figure 6).

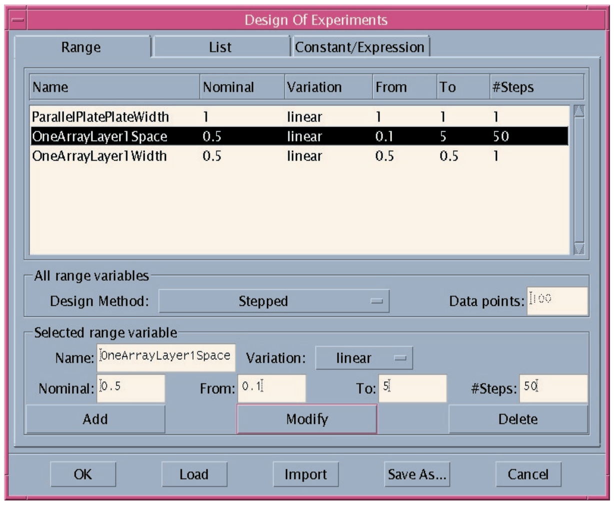
3.3 Stage 3: Output

After completing the data manipulation, it may be necessary to write out capacitance rule files to use with a layout parasitic extraction (LPE) tool. The ***LISA*** command form for writing to these files remains the same for any LPE tool. Users must consult the respective LPE manual for the required syntax. An example ***LISA*** command for writing out a string is:

```
write_parameters(“eg1.xcl”, table_PP, {“\n/
/ Example string\n”});
```

Each write_parameters command follows:

1. Key word write_parameters



Name	Nominal	Variation	From	To	#Steps
ParallelPlatePlateWidth	1	linear	1	1	1
OneArrayLayer1Space	0.5	linear	0.1	5	50
OneArrayLayer1Width	0.5	linear	0.5	0.5	1

All range variables
Design Method: Stepped
Data points: 100

Selected range variable
Name: OneArrayLayer1Space
Variation: linear
Nominal: 0.5
From: 0.1
To: 5
#Steps: 50

Figure 5. Design of Experiments layout GUI.

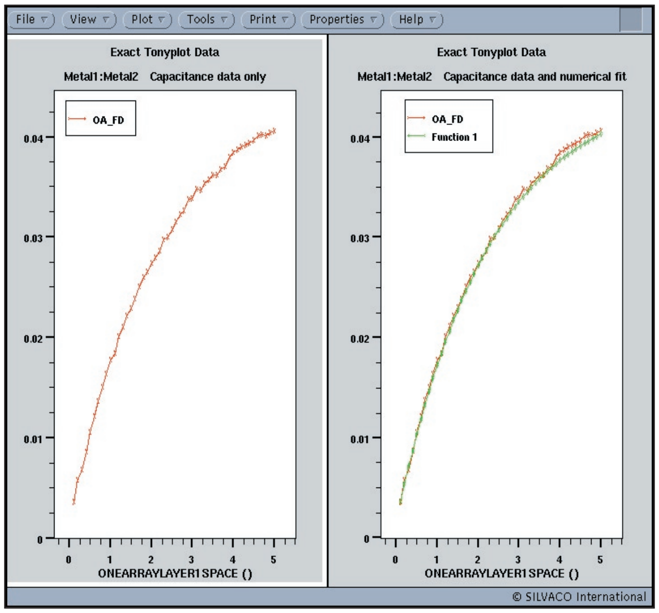


Figure 6: Calculated capacitance data (left figure) and calculated capacitance data with numerical fit (right figure).

The arguments inside the parentheses follow:

a) The file to write the text string to

b) The table containing the data referred to in the text string, if applicable

c) The actual string within braces

Another example for writing out a string is given:

```
write_parameters("eg1.xcl", res_OA, {"CAPACITANCE
CROSSOVER FRINGE ", "LAYER0", " ", "LAYER1", "\
n\n[\n\n C =length()*", "k1", "*(1.0-exp(-", "k2",
"*(distance()+", "k3", ")))\n\n]\n\n"});
```

The expression contains the fitting routine's calculated coefficients, so the coefficient values must appear in the text. This output requirement is obtained by parsing the entire string with segments of the text and coefficients contained within braces. Each segment must reside within inverted commas and is separated from other segments by commas. If the format is not strictly followed, then nothing will appear in the string output.

Output operations are performed throughout the ***LISA*** script. In addition, simpler output commands exist (to output a table, for example):

```
save_table(table_PP, CSV, "PP_a.csv"); or directly
in Tonyplot format by:
```

```
save_table(table_OA, TONYPLOT, "OA_c.str");
```

4. Conclusion

Exact's analysis stage is algorithm-intensive and benefits greatly from proper scripting techniques. This article should help users to easily write scripts for a specific process technology.