CAPLET: Parallelized Capacitance Extraction Toolkit

Yu-Chung Hsiao, Luca Daniel

February 7, 2013

Contents

Chapter 1

Installation

1.1 caplet_gds2geo.py

1.1.1 Dependencies

- python-gdsii (http://gitorious.org/python-gdsii)

1.1.2 Environment

- Tested with Python 2.7 and python-gdsii 0.2.1.

1.1.3 Installation

Place the gdsii folder of python-gdsii at the same place of caplet_gds2geo.py.

1.2 caplet_geo

1.2.1 Dependencies

- OpenGL

1.2.2 Environment

- Tested with Qt Creator 2.4.1 and Qt 4.7.4 (64bit) on Ubuntu 10.04 LTS

1.2.3 Installation

Type the command make under the folder. After the compilation is done, create two symbolic links: fastcap and capletMPI under the folder of the compiled caplet_geo.

1.3 caplet

1.3.1 Dependencies

- mpi
- pthread
- gfortran
- lapack

- blas

1.3.2 Environment

- Tested with gcc 4.4.3, mpi 1.4.3, Goto BLAS 2, on Ubuntu 10.04 LTS

1.3.3 Installation

- `make` or `make all`: compile both `capletMPI` and `capletOpenMP`.
- `make openmp`: compile `capletOpenMP`.
- `make mpi`: compile `capletMPI`.
- `make clean`: remove all files under `obj` and `tmp`.

Chapter 2

Usage

2.1 caplet_gds2geo.py

- `python caplet_gds2geo.py -l LAYER_FILE GDS2_FILE`

`LAYER_FILE` defines the vertical structures including metal layers and vias. The format is explained through the following example:

```
# Comments
METAL{
# name, layer, datatype, height, thickness
pc    7    0  0.2   0.05
m1   15    0  0.4   0.15
}

VIA{
# layer, datatype, bottom, top
ca   14    0   pc   m1
}
```

This example shows a vertical structure of two layers `pc` and `m1` connected by the `ca` via. Everything after the pound sign is considered as comments. The numbers for 'layer' and 'datatype' are used in GDS2 files, while the names are the corresponding names displayed in layout software. For the metal part, 'height' is the elevation from the surface of diffusion areas, and 'thickness' designates the thickness of the metal layer from the 'height.' In other words, the bottom of a metal layer is at the elevation 'height,' and the top of the metal layer is 'height' plus 'thickness.' For the via part, 'bottom' specifies the name of the metal layer that connects the via from the bottom, and 'top' is the name of the metal layer that connects from the top.

The output `.geo` file is of pure text that includes both layer information and polygons.

2.2 caplet_geo

The graphical interface is shown in Figure 2.1. The control buttons on the left are color schemes, solver types, basis function parameters, and parameters for generating reference solutions. For color schemes, radio boxes of 'Layer' and 'Conductor' determine how colors are painted on the surfaces. Figure 2.1 shows coloring by layer and Figure 2.2 shows coloring by conductor. The 'outline' checkbox determines whether the geometries should be outlined. For solver types, radio boxes selects one of the solver types. For the option part, 'PWC size' suggests the maximum panel size of discretization, 'Arch size' suggests the longest arch extension used to generate instantiable basis functions, and 'Num of Cores' specifies how many cores should be used when the 'Extract' button is pushed. For the

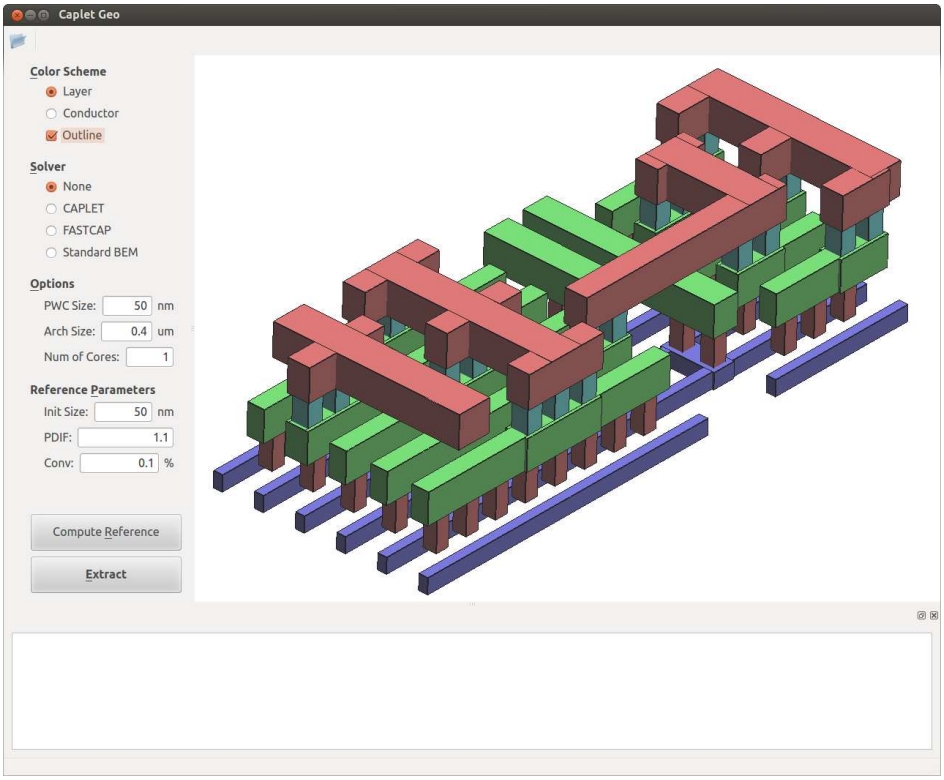

Figure 2.1: GUI with OpenGL visualization: structures colored by layer.

reference parameter part, 'Init size' specifies the initial panel size for the iterative refinement process, 'PDIF' is the ratio for refining panel size for each iteration, and 'Conv' is the level of accuracy between consecutive iterations that is considered converged and to stop the iterative process. Once the iterative process is considered converged, the reference capacitance matrix is obtained. The button 'Compute Reference' is used to trigger the iterative refinement process for generating reference solutions. The button 'Extract' is used to extract the capacitance matrix using the selected solver with specified parameters. The extraction timing information, the capacitance matrix, and the errors compared to the reference solution will be printed in the bottom text area once the extraction is done. Figure 2.3 and 2.4 shows the visualization of piecewise constant basis functions and instantiable basis functions, respectively.

2.3 `caplet`

- For the MPI version, `mpirun -np NC caplet -o OUTPUT INPUT`
- For the OpenMP version, `caplet -o OUTPUT INPUT`

where 'NC' specifies the number of cores to execute, 'INPUT' is the input file name, and 'OUTPUT' is the output capacitance matrix file name. When the input file name ends with `.caplet`, the solver of Galerkin with instantiable basis functions is executed. When the input file name ends with `.fastcap`, the solver of collocation with piecewise constant basis functions is executed.

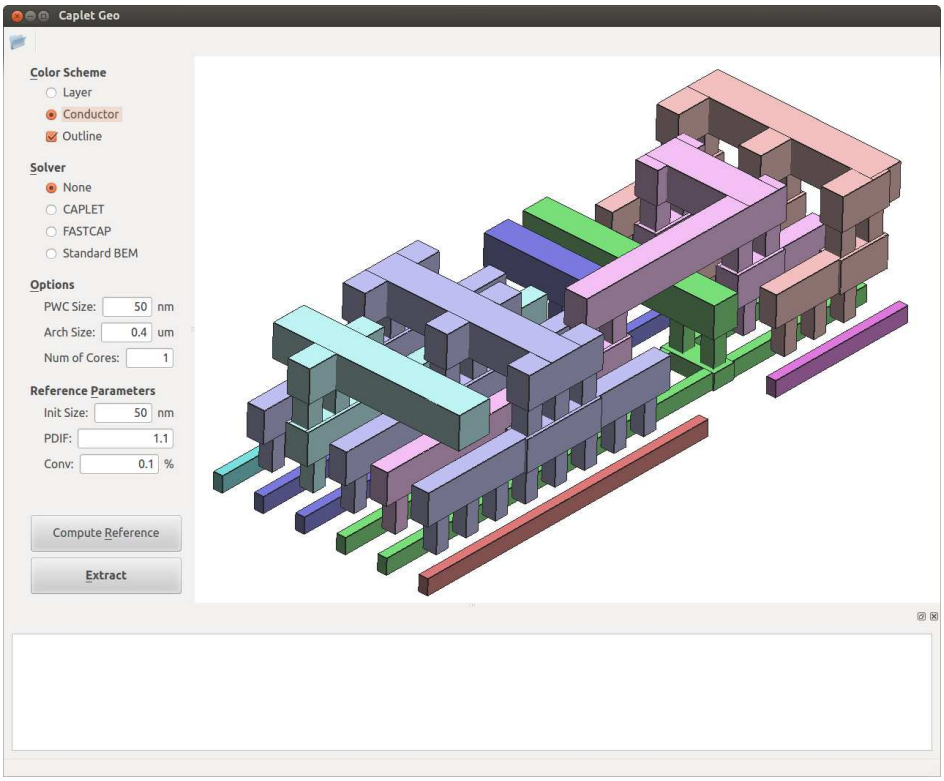

Figure 2.2: GUI with OpenGL visualization: structures colored by conductor.

Figure 2.3: GUI with OpenGL visualization: piecewise constant basis functions.

Figure 2.4: GUI with OpenGL visualization: instantiable basis functions.