# Introduction to RWCap

## Overview

The RWCap is a parallel program suitable for extracting the capacitances among 3-D rectilinear-shaped conductors. The algorithm inside is the floating random walk (FRW) algorithm. The RWCap is free of discretization error, and suitable for the capacitance extraction of large-scale Manhattan VLSI layout.

## Three-Dimensional Field Solver

RWCap solve three-dimensional electrical problems in capacitance.

TODO

# Chapter 1 Usage of QBEMIN

## GDS II Interface

The GDS II Stream format is the most widely used data format for layout representation. A GDS II file stores layout information as polygons, lines, or labels in up to 1,024 different layers.

(to be finished by Wang Yuan or Xiaodong)

## Usage of QBEMIN

QBEMIN can generate the input file of QBEM according to the specified files. Three files are used in QBEMIN as following:

Techology file.

GDSII file interface.

Layer mapping file.

These file format will be described in the next sections.

## Command Lines of QBEMIN

After the compilation, the executive program QBEM will be generated. QBEM can be executed with the command line as follows:

**QBEMIN <*GdsFile*> [*Option*]**

<*GdsFile*> is the GDSII file name. It should be given out.

[***Option***] can be the following:

* **-tech *Techfilename***

<*Techfilename*> is the technology definition file name. It should follow the option <-tech>. If no technology file name specified, QBEMIN will use the default file named “TECHFILE”.

* **-map *Mapfilename***

<*Mapfilename*> is the layer mapping definition file name. It should follow the option <-map>. If no mapping file name specified, QBEMIN will use the default file named “MAPFILE”.

* **-h**

Using “-h” option can display the help information of QBEMIN.

* **-o *OutFilename***

QBEMIN can use the specified file name as the output file with the “-o” option. The default output file name is “qbem\_format”.

Example:

1. Convert the GDSII file named “gdstest.gds” to “qbem\_format”. “TECHFILE” and “MAPFILE” are existed in the current directory. The following command can be used:  
     
    QBEMIN gdstest.gds
2. Convert the GDSII file named “gdstest.gds” to “myqbem”. Using the “Mytech” as the technology file. Using the “Mymap” as the mapping file. The following command can be used:  
     
    QBEMIN gdstest.gds –tech Mytech –map Mymap –o myqbem

## Technology File

Technology file is used to define the semiconductor process. It consists of comments and/or definitions.

* In technology file, all statements are case insensitive.
* The lines are comments if they are leaded by “//”.
* The line leaded by “+” is the continuous line.
* Definition of conductor is in the following format:   
  Conductor %{  
   *Definition*  
  }  
  The syntax of the *definition* is as the following:  
  *Conductor Name*

*+ Thickness (unit = A)*

*+ Thickness variation ( +/- %)*

*+ MinWidth (unit = um) (drawWidth / SiWidth(tw/bw))*

*+ WidthVariation (+/-%)*

*+ MinSpace (unit = um) ( drawSpace / SiSpace(ts/bs))*

*+ SheetResistance(width1 : ohm/square, width2: ohm/square, …..)*

*+ TC(Temperature Coefficient) (delta/C)*

*+ Height (Nominal Distance Between conductor layer and substrate under FOX) (A)*

*+ OverEtch (um)*

Definition of conductor is in the following format:   
Dielectric %{  
 *Definition*  
}  
The syntax of the *definition* is as the following:  
*DielectricLayer Name*

*+ Thickness (unit = A)*

*+ Thickness variation ( +/- %)*

*+ DielectricConstant*

*+ Conformal(Yes/No)*

*+ Height(Nominal Distance Between conductor layer and substrate under FOX) (A)*

*+ Note (Optional)*

## Example of Technology File

Here is one example of technology file which define the semiconductor process as shown in Figure 1.1.

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

*//========================================//*

*// Example of Technology File //*

*//========================================//*

*TECHNOLOGY=TECHTEST*

*// Dielectric Layer Definition*

*Dielectric%{*

*Substrate*

*+ Thickness(A) = 0.0*

*+ Height(A)=0.0*

*+ Dielectric=3.9*

*}*

*Dielectric%{*

*Layer0*

*+ Thickness(A)=5500*

*+ Dielectric=4.2*

*+ Height(A)=5500*

*}*

*Dielectric%{*

*Layer1*

*+ Thickness(A)=4000*

*+ Dielectric=3.9*

*+ Height(A)=9500*

*}*

*Conductor%{*

*Metal1*

*+ Thickness(A)=2800*

*+ MinWidth(um)=0.2 (0.24/0.2)*

*+MinSpace(um)=0.21(0.17/0.21)*

*+SheetResistance(ohm/square)=0.098*

*+Height(A)=8300*

*}*

*Dielectric%{*

*Layer2*

*+ Thickness(A)=4900*

*+ Dielectric=3.71*

*+ Height(A)=14400*

*}*

*Conductor%{*

*Metal2*

*+ Thickness(A)=3700*

*+ MinWidth(um)=0.2 (0.24/0.2)*

*+MinSpace(um)=0.21(0.17/0.21)*

*+SheetResistance(ohm/square)=0.057*

*+TC(delta/C)=0.0038*

*+Height(A)=13200*

*}*

*Dielectric%{*

*Layer3*

*+ Thickness(A)=5000*

*+ Dielectric=8.1*

*+ Height(A)=19400*

*}*

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~



## Mapping File

# Chapter 2 Using RWCap

## Command Line

RWCap use Makefile to organize code compilation. You can compile the whole project with **make** command at the project root directory. Also we provide a pack tool to generate a release package of RWCap as follows:

**tool/packer/pach.sh**

Then, a release package rwcap-solver.tar.gz will generated in current dircetory. Unzip the package, and you will see something like this:

**rwcap-sovler/**

**├── bin/**

**├── lib32/**

**├── lib64/**

**├── tables/**

**└── rwcap**

The **rwcap** is the entry of this program. It has two running modes.

1st mode:

**$ ./rwcap –gft <data\_dir>**

Register the folder "data\_dir" storing the data generated by TechGFT.

2nd mode:

**$ rwcap -f <file\_name> [-p <rel\_self\_cap> | -t <num\_walks> ] [-n <num\_thread> ] [-w <min\_width> ] [-o <file\_name>]**

Execute the capacitance extraction.  
"-f" specifies the input file name.   
"-p" specifies the threshold of relative error of Self capacitance.   
"-t" specifies the number of walks for termination.   
"-n" specifies the number of threads for parallel computing.   
"-w" specifies the minimum wire width of the process technology.

"-o" specifies the output file name.

Under **tables/** there are the GFTs of single dielectric. You can also put multi-dielectric GFTs into it. It is the default folder for rwcap to find the GFTs.

**lib32 / lib64** contains the shared libs that RWCap depends.

## RWCap Input File Format

3D format file is used to describe 3D structures for accurate extraction. RWCap uses the 3D format file as the input file.

The syntax of the input file is explained in the next section, If no options are specified, RWCap reads *<file>*, makes calculations according to the input, and writes the results to the output file named *<file>.out*.

The input file statements consist of comments and/or commands.

* Lines beginning with /\* and ending with \*/ are comment lines and can be ignored.
* All commands may be defined by the set description statements. The set description statements use the signs similar with HTML. It should begin with <> and end with [/].
* The following commands are available in RWCap which can be classified into 3 groups:
  1. Identified commands: **QBEM.**All the other commands should be included in this command.
  2. Primary commands**: WINDOW, MEDIUM, CONDUCTOR, TASK.**
  3. Basic geometry definition commands: **BLOCK, POLY.**The geometry definition commands can be included in the primary commands “**MEDIUM, CONDUCTOR”.**
  4. Task definition commands: **CAPACITANCE**The task definition commands can only be included in command **TASK.**
* The input to RWCap is case insensitive, meaning two instructions when written with all lowercase or all uppercase letters are same.

The syntax and usage of each command are described in this section.

## Identified Command

## QBEM

The **QBEM** command is the symbol of the RWCap input format file. The file should start with <qbem>, and end with [/qbem].

## Primary Commands

## WINDOW

The **WINDOW** command sets the simulation cube size in three-dimension. A reflective (or, Neumann) boundary condition is applied on the six surfaces of the simulation window.

<**window**>

**v1** <*vector*>

**v2** <*vector*>

[/**window]**

|  |  |  |
| --- | --- | --- |
| Parameter | DataType | Definition |
| V1 | vector | Coordinates of the left bottom of the window |
| V2 | vector | Coordinates of the right top of the window |

## MEDIUM

The **MEDIUM** command describes one dielectric in three-dimension when compute the capacitance. It can contain one or more basic geometry description statements.

<**medium**>

**<***BASIC GEOMETRY DESCRIPTION***>**

[**<***BASIC GEOMETRY DESCRIPTION***>]**

**…**

**diel** <*VALUE>*

[/**medium]**

|  |  |  |
| --- | --- | --- |
| Parameter | DataType | Definition |
| Basic Geometry | definition | Geometry description statements, such as : BLOCK, POLY. |
| diel | value | Permittivity of dielectric |

## CONDUCTOR

The **CONDUCTOR** command describes one conductor in three-dimension. It can contain one or more basic geometry description statements.

<**conductor**>

**name** *<NAME>*

**<***BASIC GEOMETRY DESCRIPTION***>**

[**<***BASIC GEOMETRY DESCRIPTION***>]**

**…**

[/**conductor]**

|  |  |  |
| --- | --- | --- |
| Parameter | DataType | Definition |
| name | character | Name of the conductor. **NAME** must begin with an alphabetic character, but may include up to 100 alphanumeric characters. |
| Basic Geometry | definition | Geometry description statements, such as : BLOCK, POLY. |

## TASK

The **TASK** command defines the specific computational task to be carried on. Four different tasks can be executed including: CAPACITANCE, CHARGE, RESISTANCE and CURRENT.

<**task**>

<*TASK DEFINITION*>

[*TASK DEFINITION*]

…

[/**task**]

|  |  |  |
| --- | --- | --- |
| Parameter | DataType | Definition |
| Task Definition | definition | Task definition statements, such as : CAPACITANCE, RESISTANCE, CHARGE and CURRENT. |

## Task Definition Commands

## CAPACITANCE

The **CAPACITANCE** command defines task to compute the self capacitance on the specific conductor and all the coupling capacitance between it and all other conductors.

<capacitance>

<*CONDUCTOR NAME*>

[<*CONDUCTOR NAME*>]

…

[**ALL**]

[/capacitance]

|  |  |  |
| --- | --- | --- |
| Parameter | DataType | Definition |
| Conductor NAME | character | The NAME of the specific conductor for capacitance computation. If the NAME is “ALL” means all the capacitances will be extracted. |

## Basic Geometry Definition Commands

## BLOCK

The **BLOCK** command defines a rectangular box. Here *vector* is the collection of three numbers X,Y,Z, separated by commas. These numbers are coordinates of a vector on a global coordinate system (See Figure 1).

<**block**>

**basepoint** <*VECTOR*>

**v1** *<VECTOR>*

**v2** *<VECTOR>*

**hvector** <*VECTOR*>

**[/block]**

|  |  |  |
| --- | --- | --- |
| Parameter | DataType | Definition |
| basepoint | Vector | Coordinate of the base point of the block.  (**default units:** microns) |
| V1 | Vector | Direction vector of the length axis. This vector is defined in the *local coordinate system* according to the base point. The magnitude of this vector is the length of the block.  (**default units:** microns) |
| V2 | vector | Direction vector of the width axis. This vector is defined in the *local coordinate system* according to the base point. The magnitude of this vector is the width of the block.  (**default units:** microns) |
| Hvector | Vector | Direction vector of the height axis. This vector is defined in the *local coordinate system* according to the base point. This vector is always perpendicular to the surface determined by V1 and V2. The magnitude of this vector is the height of the block.  (**default units:** microns) |

Figure 1 clarifies the meanings of these parameters.



Examples:

The following commands generate a block 3 units long in the X direction, 2 units long in the Y direction and 2.5 units long in the Z direction with the base point at (1,1,2).

<block>

basepoint (1,1,2)

v1 (3,0,0)

v2 (0,2,0)

hvector (0,0,2.5)

[/block]

## POLY

The **POLY** command defines a polyhedron with **n** vertices in the polygon base (bottom surface) and by extrusion of this base (See Figure 2). The <*POINT*> is the collection of two numbers X,Y, separated by commas.

<**poly**>

**basepoint** <*VECTOR*>

**v1** *<VECTOR>*

**v2** *<VECTOR>*

**hvector** <*VECTOR*>

<**coord**>

<*POINT*> <*POINT*> …

[/**coord]**

**[/poly]**

|  |  |  |
| --- | --- | --- |
| Parameter | DataType | Definition |
| basepoint | Vector | Coordinate of the base point of the block.  (**default units:** microns) |
| V1 | Vector | This vector is used to define the X axis of the *local coordinate system* according to the base point.  (**default units:** microns) |
| V2 | vector | This vector is used to define the Y axis of the *local coordinate system* according to the base point.  (**default units:** microns) |
| Hvector | Vector | Direction vector of the extrusion. This vector is in the *local coordinate system*. This vector is arbitrary oriented. The magnitude of this vector is the height of the extrusion.  (**default units:** microns) |
| Coord | point | x,y coordinates of the vertices of the polygon base expressed in the *local coordinate system* with origin **v1** and **v2.** The polygon base lies in the xy-plane of this local coordinate system. A minimum of 3 coordinate points should be given.  (**default units:** microns) |

Figure 2 clarifies the meanings of these parameters.



Examples:

The following commands generate a polyhedron. The polygon is defined by four vertices, with a height of 2.5. The origin of the local coordinate system is the same as the origin of the global coordinate system; the direction perpendicular to the plane of the polygon is along the global Z axis.

<poly>

basepoint (1,1,2)

v1 (1,0,0)

v2 (0,1,0)

hvector (0,0,2.5)

<coord>

(2,2) (2,4) (4,4,) (4,2)

[/coord]

[/block]

## Constraints of the Input File

The structures described in the input file will have the following constraints:

* All the conductors should not be overlap with others.
* The 3D basic geometries defined in the same command set of MEDIUM can be overlap with each other.
* The geometries defined in different command sets of MEDIUM can only be inclusive or separate. The overlap of them is not allowed.

## Example of the Input File

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

<qbem>

<window>

v1 (10.7500, 12.2000, -0.0100)

v2 (11.5500, 12.9250, 9.9330)

[/window]

<medium>

<block>

basepoint (10.7500, 12.2000, -0.0100)

v1 (0.8000, 0, 0)

v2 (0, 0.7250, 0)

hvector (0, 0, 0.3450)

[/block]

diel 3.9000

[/medium]

<medium>

<block>

basepoint (10.7500, 12.2000, 0.3350)

v1 (0.8000, 0, 0)

v2 (0, 0.7250, 0)

hvector (0, 0, 0.7500)

[/block]

diel 3.9000

[/medium]

<medium>

<block>

basepoint (10.7500, 12.2000, 1.0850)

v1 (0.8000, 0, 0)

v2 (0, 0.7250, 0)

hvector (0, 0, 1.5980)

[/block]

diel 3.9000

[/medium]

<medium>

<block>

basepoint (10.7500, 12.2000, 2.6830)

v1 (0.8000, 0, 0)

v2 (0, 0.7250, 0)

hvector (0, 0, 1.5000)

[/block]

diel 3.9000

[/medium]

<medium>

<block>

basepoint (10.7500, 12.2000, 4.1830)

v1 (0.8000, 0, 0)

v2 (0, 0.7250, 0)

hvector (0, 0, 5.7500)

[/block]

diel 7.5000

[/medium]

<conductor>

name groudplane

<block>

basepoint (10.7500, 12.2000, -0.0100)

v1 (0.8000, 0, 0)

v2 (0, 0.7250, 0)

hvector (0, 0, 0.0100)

[/block]

[/conductor]

<conductor>

name conductor1\_61\_1

<block>

basepoint (10.9500, 12.4000, 0.3350)

v1 (0.6000, 0, 0)

v2 (0, 0.1250, 0)

hvector (0, 0, 0.2500)

[/block]

<block>

basepoint (10.9500, 12.2000, 0.3350)

v1 (0.4000, 0, 0)

v2 (0, 0.2000, 0)

hvector (0, 0, 0.2500)

[/block]

<block>

basepoint (10.7500, 12.5250, 0.3350)

v1 (0.8000, 0, 0)

v2 (0, 0.2750, 0)

hvector (0, 0, 0.2500)

[/block]

<block>

basepoint (10.7500, 12.8000, 0.3350)

v1 (0.5750, 0, 0)

v2 (0, 0.1250, 0)

hvector (0, 0, 0.2500)

[/block]

[/conductor]

<task>

<capacitance>

conductor1\_61\_1

[/capacitance]

[/task]

[/qbem]

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

## RWCap Output File

RWCap can writes the results to the output file named *<file>.out* when the input file *<file>* is computed. The output file is organized as the following formats.

|  |  |  |
| --- | --- | --- |
| Key Word | **Data Type** | Description |
| Command | Character | Show the name of the executive program. |
| Master | Character | Show the master conductor name and the capacitance on master conductor |
| CAPACITANCE | Capacitance List | Capacitance report. |
| Total Cost | Double (unit: s) | Total time used. |

### Capacitance List

The capacitance list in the output file will report the compute result in the following format:

**Capacitance on <***ConductorName***> = <***Value***>**

The capacitance on the conductor whose name is *<ConductorName>* is the value between the master conductor and the *<ConductorName>*. The master conductor is defined in the command set **CAPACITANCE** of the input file. *<ConductorName>* is the name of the conductor defined in the command CONDUCTOR of the input file. <*Value*> is the capacitance value with the unit Farad.

Example：

Capacitance on substrate = -1.2345e-17 F

Capacitance on conductor1\_2\_Metal1 = -1.2345e-17 F

Capacitance on conductor2\_3\_Metal2 = 2.4689e-17 F

**Source Code Introduction**

**Doxygen** is used to generate document for this document.

**TODO** 描述doxygen 生成文档方法