OpenPCells

Technology Files Guide and Format

Patrick Kurth

October 27, 2022

This is the official documentation of the OpenPCells project. It is split in several different files for clarity. This document provides an overview of the creation of technology files. If you are looking for a general overview of the project and how to use it, start with the user guide, which also contains a tutorial for getting started quickly. If you are looking for a guide and documentation on the creation of parametrized cells, consult the celldesign manual. If you want to now more about the technical details and implementation notes, look into the technical documentation.

1 Introduction

The pcells are defined in general layers (such as ”gate” or ”M1” or ”lastmetal”), which have to be translated into a specific technology for cell generation. Furthermore, vias and contacts have to be put into arrays, which requires knowledge about the technology. The technology-defining files (the “techfiles”) describe the required rules and layer mappings. In order to make it easy to start with techfiles they only need to contain what is required by the generated cell. That means that an inductor on only one metal layer without any vias only needs to find the respective mapping for that metal layer.

2 Adding Technologies

A technology definition is a collection of files in one directory. These files get loaded by opc when requested as technology, the directory has to be within on of the paths in the technology search path (see the –techpath command line option). OpenPCells expects a technology to provide at least three files: `config.lua`, `layermap.lua` and `vias.lua`. All these files return one lua table, so a basic (empty technology) has the following content for all of these files:

```
return {}
```

Of course, this does not really make sense, so the following sections explore what (and when) to put into these files.

2.1 Config

The config is pretty simple:

```
return {
    grid = 1, -- in nanometers,
    metals = 10, -- number of metals in the stack
}
```

The ‘grid‘ defines the granularity of the shapes, ‘metals‘ is the total number of available metal (and interconnect) layers in this technology node.

2.2 Layermap

The layermap includes information on the human-readable layer data as well as the stream numbers (virtuoso could also work just with the stream numbers, but often the layers have internal numbers that are NOT the stream numbers). Therefore every entry is a table containing a table for the layer and a table for the purpose:

```
-- example for metal 1
M1 = {
    name = "metal1",
    layer = {
        gds = { layer = 8, purpose = 0 },
        SKILL = { layer = "metal1", purpose = "drawing" },
        svg = { style = "metal1", order = 4, color = "0000ff" },
    }
},
-- example of an unused layer
notused = {},
```

The needed layers depends on the cells that are being used, but the program will also tell you when you are missing something. Therefore, you can also keep running it until it works. The ‘opc‘ technology layer map contains most of the used layers and can be used as a reference.

2.3 Vias

The via rules file defines the via/contact geometries. Multiple entries are possible to accomodate the different ways of drawing vias.

```
viaM1M2 = {
    entries = {
        {
            width = 100, height = 100,
            xspace = 200, yspace = 200,
            xenclosure = 50, yenclosure = 50
        },
        {
```

```
            width = 100, height = 100,
            xspace = 200, yspace = 200,
            xenclosure = 100, yenclosure = 20
        },
    },
},
```

The entry describes the geometry of the actual cuts as well as margin for the surrounding conductors.