OpenLane Tutorial - Hardening the Macro

The process of hardening the macro block is given for the reference. As these macro blocks will be used in the top-level hierarchy some configurations need to be made.

Step 1: Starting the OpenLane environment

Just run the following commands to enter the OpenLane environment (from the installed location of OpenLane)

```
cd Openlane/
make mount
```

Step 2: Creating new designs

The following command creates a new configuration file for your design:

```
./flow.tcl –design <design name> -init_design_config -add_to_designs
```

This will create the following directory structure:

```
designs/<design_name>
├── config.json
├── src
```

Step 3: Create the RTL files

You need to create or copy the RTL files. The recommended location for files is

```
designs/<design name>/src/.v
```

Step 4: Give the parameter configuration

Modify the config.json to include following:

- **DESIGN_IS_CORE** controls the metal levels used for power routing. Set it to **false** to use only lower levels.
- **FP_PDN_CORE_RING** is set to **false** to disable a power ring around the macro block.
- **RT_MAX_LAYER** set to **met4** to limit metal layers allowed for routing.

Step 5: Run the flow on the macro design

Finally, run OpenLane. flow.tcl is the entry point for OpenLane. The command needs to be run from inside the environment of OpenLane as described in quick start.

./flow.tcl -design <design name> -tag full_guide –overwrite

Step 6: Analyzing the flow generated files

You can open the interactive view using the following commands:

./flow.tcl -design <design name> -tag full_guide -interactive

package require openlane

run_synthesis

run_floorplan

run_placement

run_cts

run_routing

run_magic

run_magic_spice_export

run_magic_drc

run_lvs

run_antenna_check

or_gui

The above commands can also be written in a file and passed to flow.tcl:

./flow.tcl –interactive –file <file>

Step 7: Viewing of final layout

The following command generates a new gds or lef view for your design:

klayout designs/<design name>/runs/full_guide/results/final/gds or lef

(or)

magic designs/<design name>/runs/full_guide/results/final/gds or lef

Each run has following structure:

```
├── logs (or) reports (or) results (or) tmp
│   ├── cts
│   ├── floorplan
│   ├── placement
│   ├── routing
│   ├── signoff
│   └── synthesis
├── runtime.yaml
└── warnings.log
```

There are 4 directories logs reports results and tmp. In each of these directories, there are multiple directories. Directories are named according to the stage they belong to.

Finally output of OpenLane can be found in

designs/<design name>/runs/full_guide/results/final

```
├── def
│   └── <design name>.def
├── gds
│   └── <design name>.gds
├── lef
│   └── <design name>.lef
├── mag
│   └── <design name>.mag
├── maglef
│   └── <design name>.mag
├── sdc
│   └── <design name>.sdc
├── sdf
│   └── <design name>.sdf
├── spef
│   └── <design name>.spef
├── spi
│   └── lvs
│       └── <design name>.spice
└── verilog
    └── gl
        └── <design name>.v
```

Example - RAM I/O Tile:

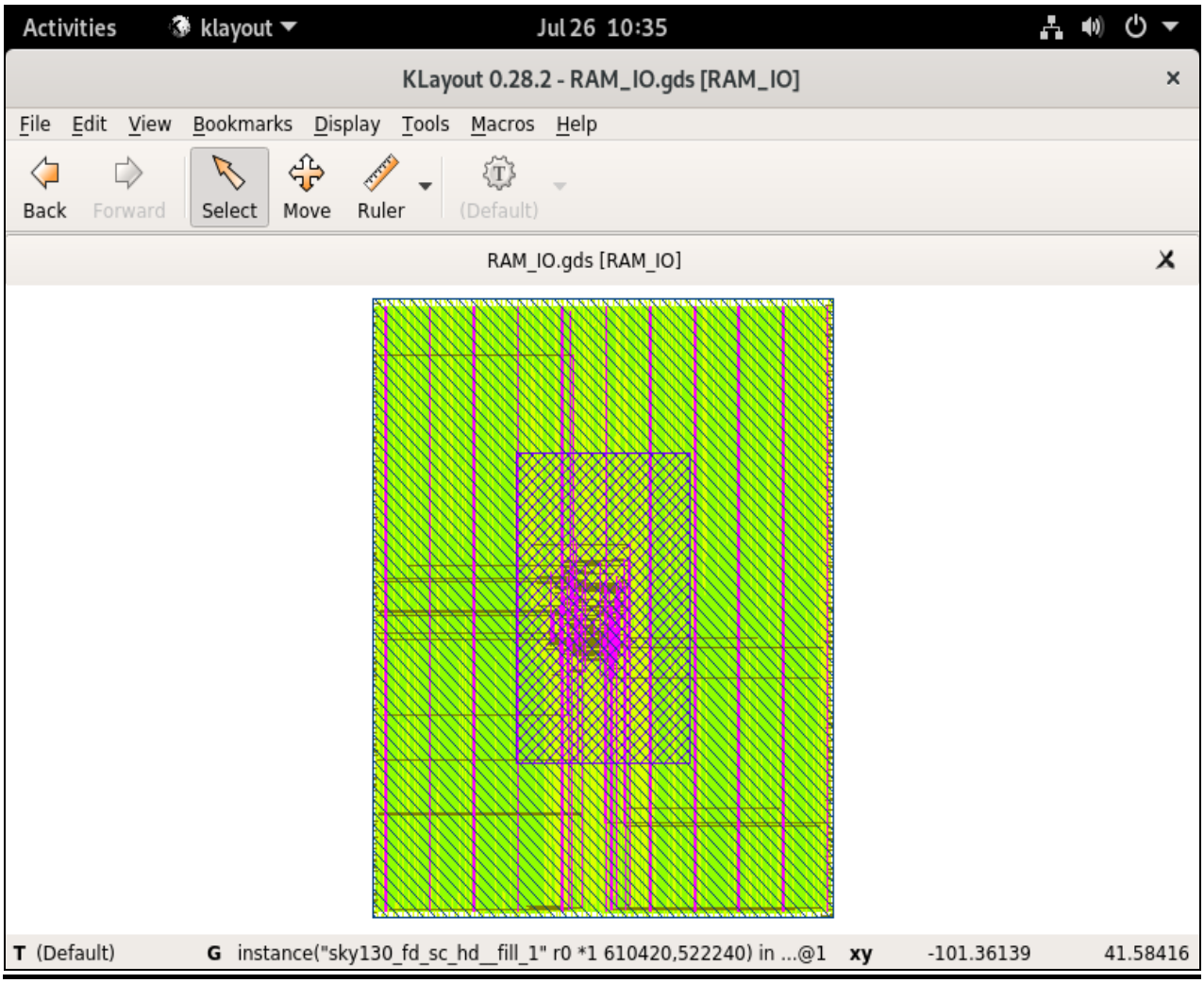
Configuration Variables:

Activities Text Editor Jul 26 10:37

Open config.json ~/sample/OpenLane/designs/ramiotile Save

```
{
"DESIGN_NAME"               : "RAM_IO",
"VERILOG_FILES"             : "dir::src/*.v",
"CLOCK_PERIOD"              : "10.0",
"DESIGN_IS_CORE"            : "false",
"CLOCK_PORT"                : "UserCLK",
"FP_SIZING"                 : "absolute",
"DIE_AREA"                  : "0 0 800 1000",
"CLOCK_NET"                 : "UserCLK",
"SYNTH_NO_FLAT"             : "1"
}
```

GDS View using KLayout:



LEF View using KLayout: