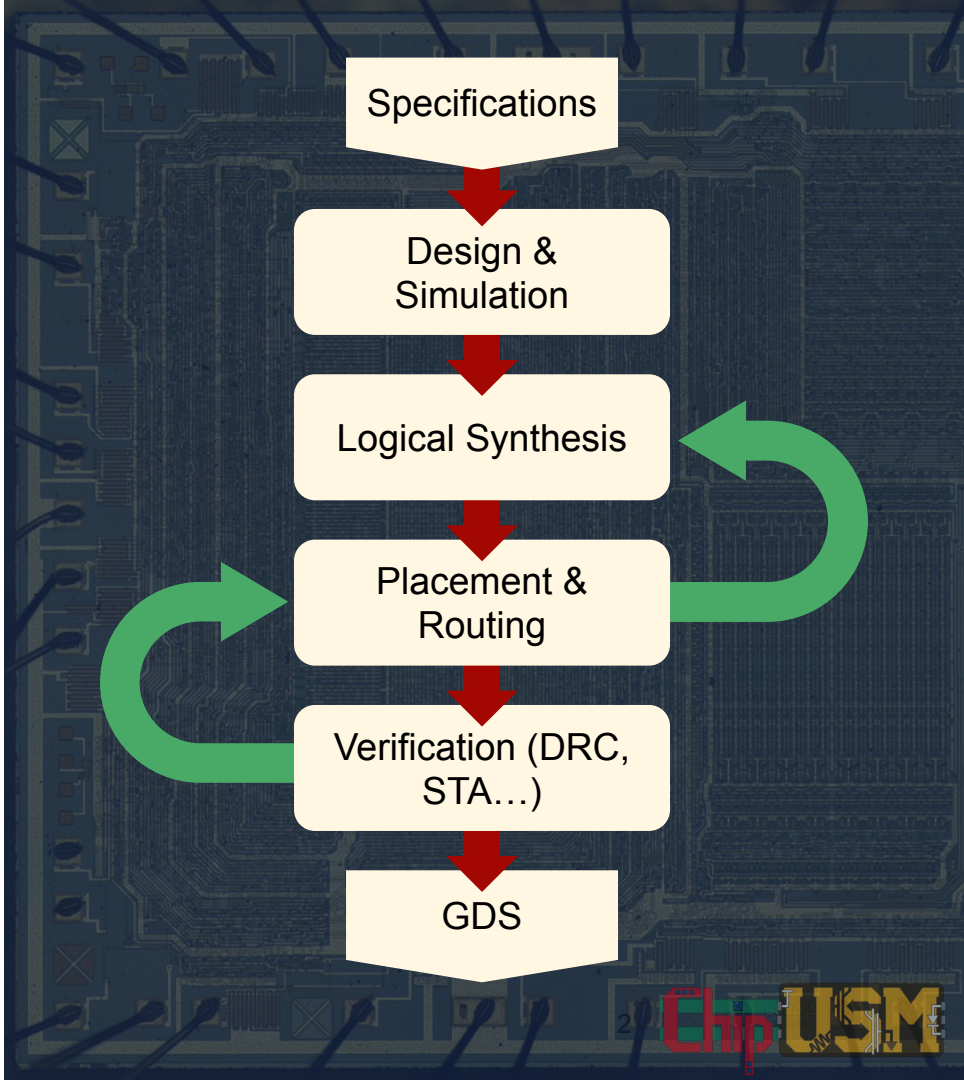
Physical Implementation

Workshop-CANELOS24

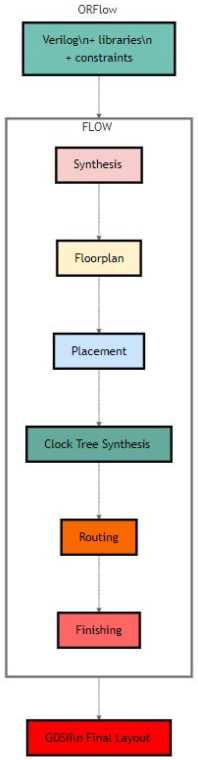
UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Design flow for digital integrated circuits



Physical Implementation with OpenROAD

Source Code (*.v) ➡

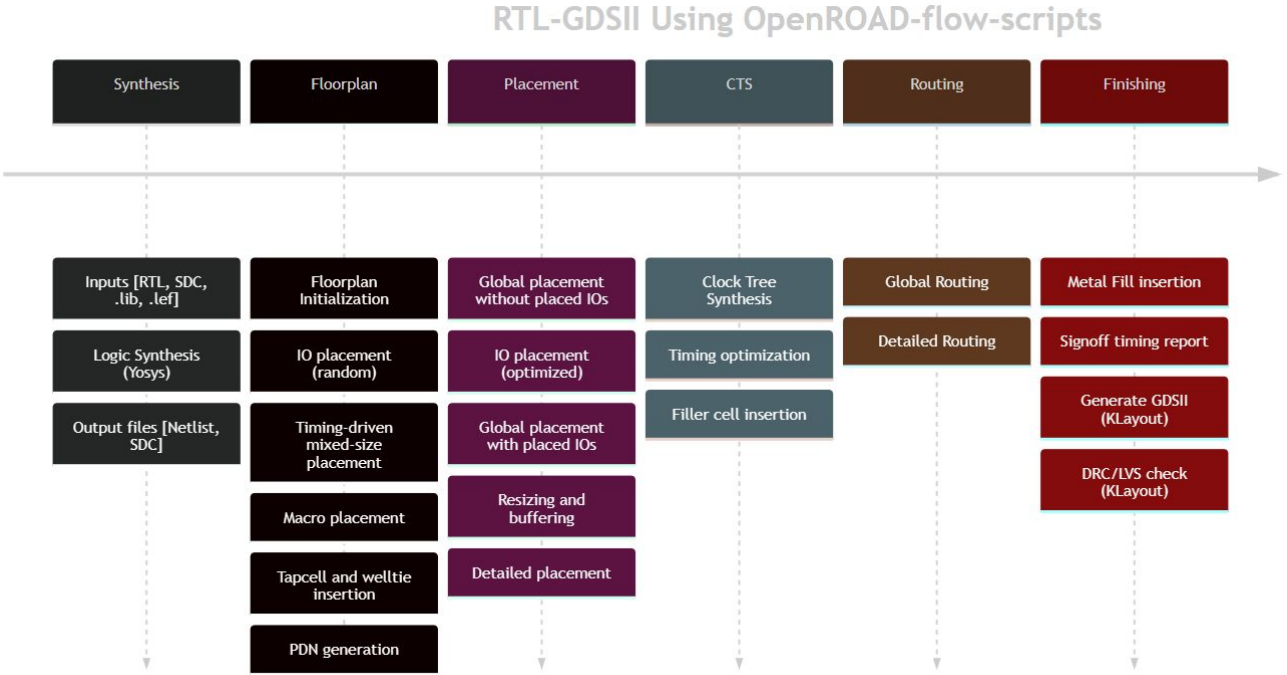


⬅ Technology information (PDK)

Ready for TapeOut ➡

Physical Implementation with OpenROAD

RTL to GDS:



https://openroad.readthedocs.io/en/latest/main/README.html

Use of OpenROAD

Use cases:

- ***Macro Hardening:*** *implementation of a module to later instantiate it.*
- ***SoC Integration:*** *top-level implementation, based on macros.*

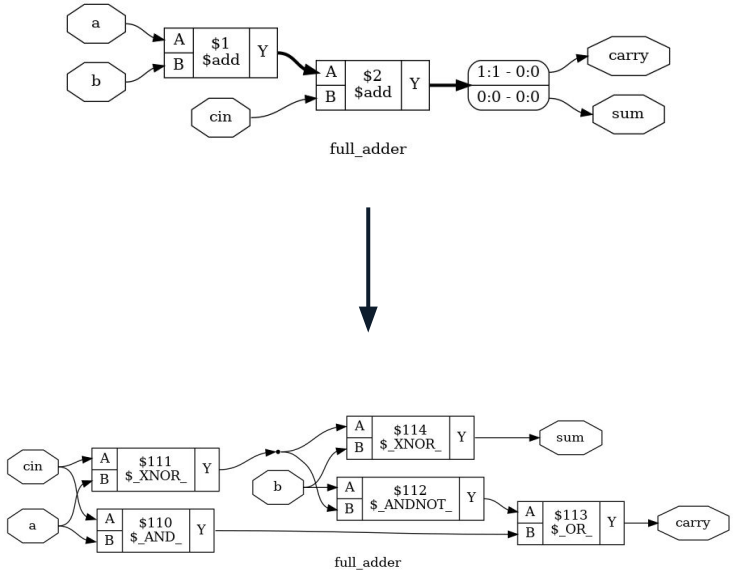
To use OpenROAD, the **Makefile** is executed, which allows:

- Initializing the project.
- Executing the partial or complete flow.

Step by Step

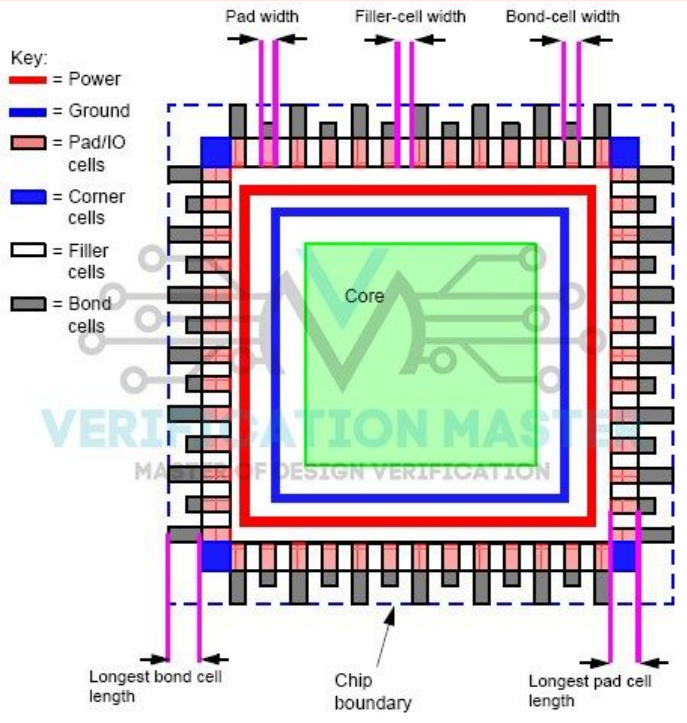
Synthesis

- The first step in the design flow is synthesis, where the hardware description is transformed into the actual system implementation using the logic gates provided by the PDK.
- OpenROAD performs this process using **Yosys**.
- The script used by OpenROAD for the synthesis process is called **synth.tcl**



https://github.com/The-OpenROAD-Project/yosys/blob/master/README.md

Floorplanning

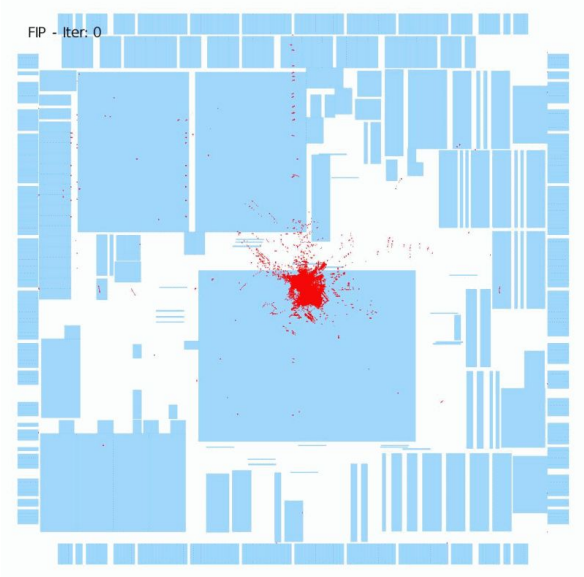


Floorplanning is the process of arranging and positioning functional blocks within a chip's layout to optimize performance, power, and area. Some of the actions performed during floorplanning are:

1. Die area and core area definition
2. IO placement
3. Macro placement
4. PDN (Power Distribution Network) generation
5. PAD insertion

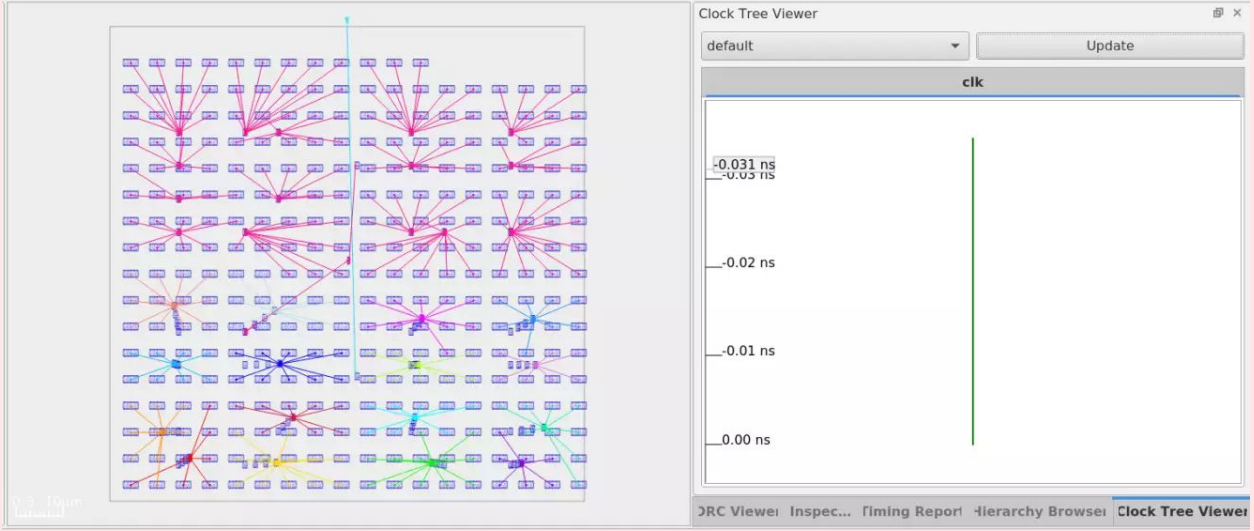
Placement

The place step involves positioning standard cells within the defined floorplan. This step ensures that the cells are placed efficiently to optimize timing, area, and power consumption, while also considering connectivity and routing constraints.



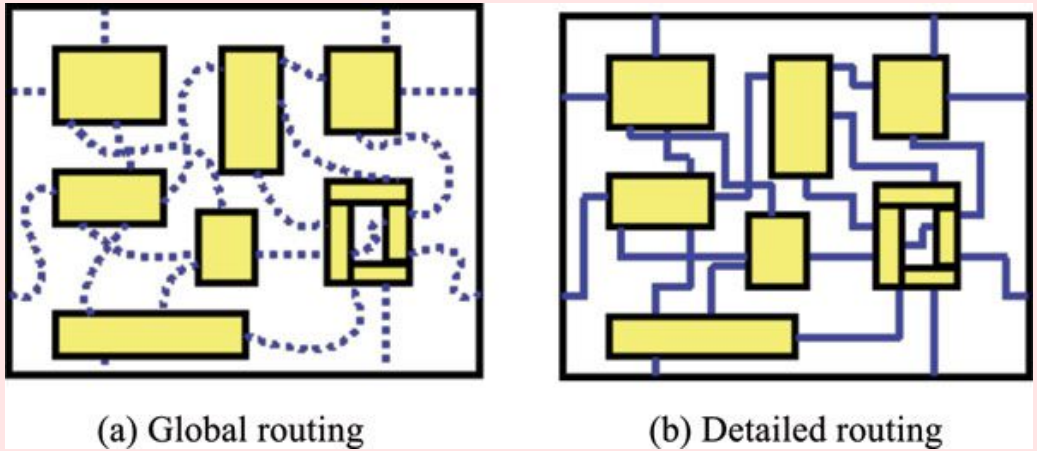
Clock Tree Synthesis

Clock Tree Synthesis (CTS) is the process of creating a clock network in VLSI to ensure the clock signal reaches all elements with minimal skew and delay, ensuring proper timing.



Routing

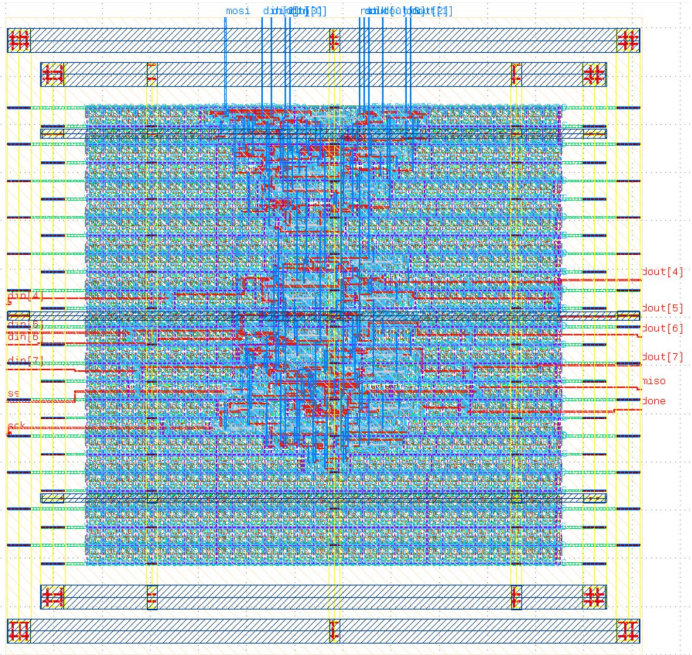
Is the process of connecting the placed components (cells, blocks) with metal wires to ensure proper electrical connections according to the design. It ensures that signals and power are delivered efficiently while minimizing delay and congestion.



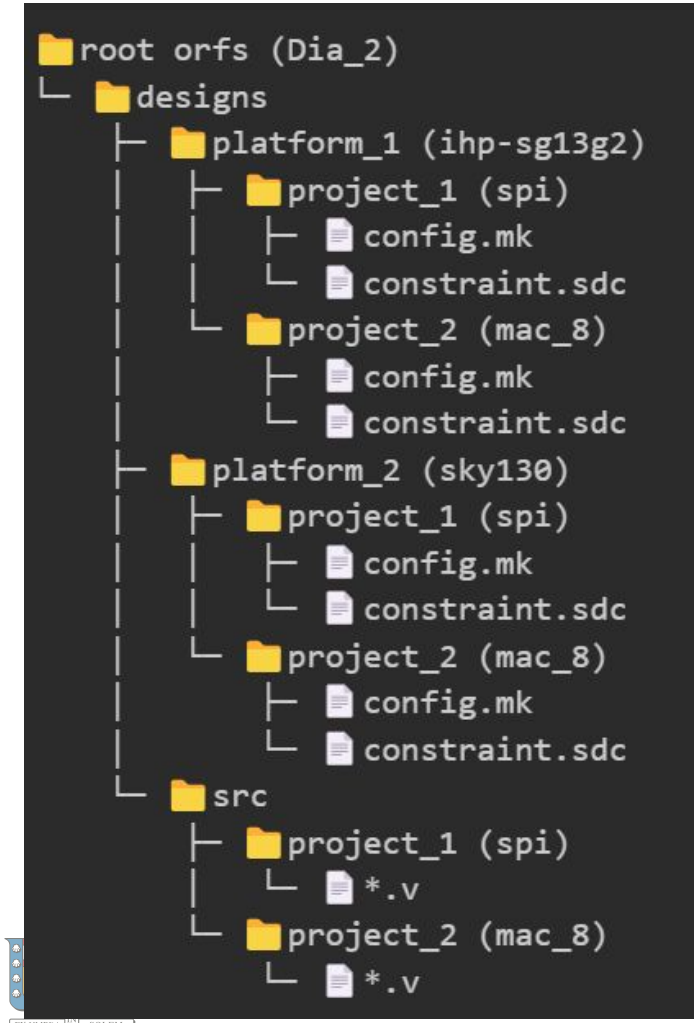
(a) Global routing (b) Detailed routing

Finishing

The final step in the OpenROAD flow is the **finishing process**, which includes filler insertion, GDS generation, and performing DRC/LVS checks to ensure the design meets manufacturing requirements.



Configuring and running an ORFS project



ORFS Directory Structure

The ORFS directory structure consists of two main folders: **designs** and **src**. The **designs** folder is further divided into a folder for each platform (technology) being analyzed. Inside each technology folder, there are project-specific folders. Within each project folder, you will find the configuration files for that project. On the other hand, the **src** folder contains independent folders for each project, which house the corresponding Verilog files.

Configuring the Design (config.mk)

The **design configuration** is done using the **config.mk** file, where all the main settings and flags for the design process are defined. This file controls various parameters such as technology, paths, and specific tool options used in the flow.

config.mk

Workshop-CANELOS24 > Dia_2 > designs > ihp-sg13g2 > spi > config.mk

```
export DESIGN_NICKNAME = spi
export DESIGN_NAME = spi
export PLATFORM    = ihp-sg13g2

export VERILOG_FILES =  $(DESIGN_HOME)/src/$(DESIGN_NICKNAME)/spi.v
export SDC_FILE      = $(DESIGN_HOME)/$(PLATFORM)/$(DESIGN_NICKNAME)/constraint.sdc

export CORE_UTILIZATION = 20
export PLACE_DENSITY = 0.88
export TNS_END_PERCENT = 100

export SYNTH_MEMORY_MAX_BITS = 16384

# Allow routing on the TopMetal layers, for the padring connections
export MAX_ROUTING_LAYER = TopMetal2

export CDL_FILE = \
    $(IHP_PDK_ROOT)/$(PDK)/libs.ref/sg13g2_stdcell/cdl/sg13g2_stdcell.cdl
```

For more information about the flow variables refer to this link.

Configuring the Design (constraints.sdc)

constraint.sdc

Workshop-CANELOS24 > Dia_2 > designs > ihp-sg13g2 > spi > constraint.sdc

```
set clk_name  core_clock
set clk_port_name clk
set clk_period 10.4
set clk_io_pct 0.2

set clk_port [get_ports $clk_port_name]

create_clock -name $clk_name -period $clk_period $clk_port

set non_clock_inputs [lsearch -inline -all -not -exact [all_inputs] $clk_port]

set_input_delay  [expr $clk_period * $clk_io_pct] -clock $clk_name $non_clock_inputs
set_output_delay [expr $clk_period * $clk_io_pct] -clock $clk_name [all_outputs]

```

The clk_port_name must have the same name that the design clock

In OpenROAD, the **constraint file** defines timing and physical constraints for the design, such as clock definitions, input/output delays, and area constraints. It ensures the design meets performance and timing requirements during synthesis, placement, and routing steps.

Running the flow

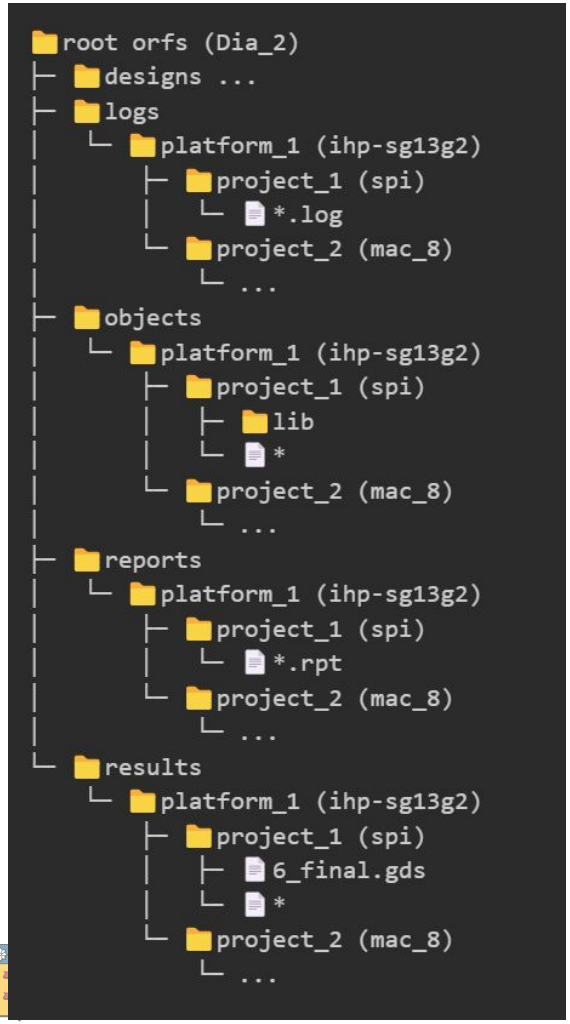
To run the flow, it is first necessary to set some environment variables. Finally, by executing the Makefile located in the OpenROAD **flow** folder, the entire process from synthesis to GDS generation will be performed.

/opt/OpenROAD-flow-scripts/flow$ make

export.me M

Workshop-CANELOS24 > Dia_2 > designs > ihp-sg13g2 > spi > export.me

```
1   ##### set the pdk to ihp
2   set_pdk ihp-sg13g2
3
4   #design specific
5   export ORFS_ROOT=$ORFS_DIR
6   export OPENROAD_EXE=$(command -v openroad)
7   export YOSYS_CMD=$(command -v yosys)
8   export WORK_HOME=$(pwd)
9   export IHP_PDK_ROOT=$PDK_ROOT
10  export DESIGN_HOME=$WORK_HOME/designs
11  export DESIGN_CONFIG=${DESIGN_HOME}/ihp-sg13g2/spi/config.mk
12  #design specific
13
```



Analyzing the Results

The OpenROAD flow generates various output files for design verification, including:

- **Logs**: Detailed logs for each stage, tracking errors, warnings, and execution details.
- **Reports**: Summaries of key metrics like timing, area, and power.
- **Objects**: Intermediate design files, such as libraries.
- **Results**: Final files, including the GDS, ready for fabrication and final checks.

Practical exercise

Run the example project

The example is available in the Dia_2 folder of the repository.

The example design is an **SPI (Serial Peripheral Interface)**, a high-speed communication protocol used for short-distance data transfer between devices, such as microcontrollers and peripherals. It operates with four main signals: MOSI, MISO, SCLK, and SS, enabling full-duplex communication.

1. To run the example in ORFS, some environment variables need to be set. To do this, execute the following command from the **Dia_2** folder.

```
/Dia_2$ source designs/ihp-sg13g2/spi/export.me
```

2. Then, start the flow with the following command.

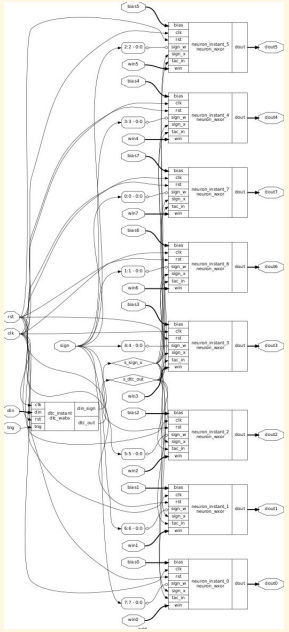
```
/Dia_2$ make -B -C $ORFS_ROOT/flow
```

3. Finally, the final GDS can be viewed using **Klayout**.

```
/Dia_2$ klayout -e results/ihp-sg13g2/mac_8/base/6_final.gds
```

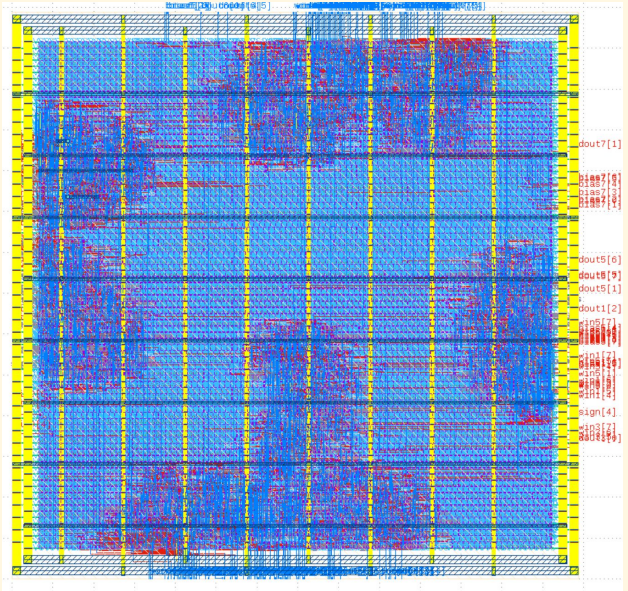
Configuring and running our own project in ORFS.

In this activity, you will implement the 8-neuron grid using the IHP-SG13G2 technology, utilizing the digital flow of the OpenROAD Flow Script. You can use the SPI example as a reference.



grid.v

ORFS



results/../6_final.gds

¿Preguntas?