Chapter 5 - Process Design Kit (PDK)

Course authors (Git file)

Section 1

What is a PDK?

Wikipedia definition

A process design kit (PDK) is a set of files used within the semiconductor industry to model a fabrication process for the design tools used to design an integrated circuit. The PDK is created by the foundry defining a certain technology variation for their processes. …

… The designers use the PDK to design, simulate, draw and verify the design before handing the design back to the foundry to produce chips. The data in the PDK is specific to the foundry's process variation and is chosen early in the design process, influenced by the market requirements for the chip. An accurate PDK will increase the chances of first-pass successful silicon.

Source: https://en.wikipedia.org/wiki/Process_design_kit

Open-source viewpoint

- Semicondutcor industry started to integrate open-soruce.
- Open-source PDKs created by Semiconductor Fabs were a “missing link” between:
  - Open-source EDA tools a(RTL-to-GDS) and
  - Microchip production (GDS-to-Chip)
- Since there is open-source PDKs, the growth of the open-source ecosystem is measurable.
- Many of the tools have been the classical “one-person maintainted” open-source projects. It is getting better.

In the context of this course

The PDK (ihp-sg13g2) integrates seemless (to the user) into the OpenROAD flow scripts toolchain.

We have seen reference points from the tools onto the PDK in:

- the configuration files
- the structure of the design directories
- some Variables

Naming

PDKs sometimes are refered to as:

- Process design kits
- Process node
- Technology node
- Technology

Section 2

Open-Source PDK and GitHub

Difference from closed source

With publishing a PDK under a open-source license, the development from there on becomes a worldwide visible joint effort. The number of contributors and authors of the PDK can only increase from here on.

Collaborative workflow in GitHub

Some of the main principles of open-source are the permissions to use, study, change and re-distribute the published code and data according to the license. This leads to a open collaboration in which everyone can participate.

GitHub enables a workflow that was designed and build with these principles and opportunities in mind. A good starting point to explore the open collaboration in the IHP PDK are

- Issues (open and closed)
- Pull requests (open and closed)

The topics and discussions that you can read and study there will draw a picture of how the process of open collaboration works for the PDK.

Issues open

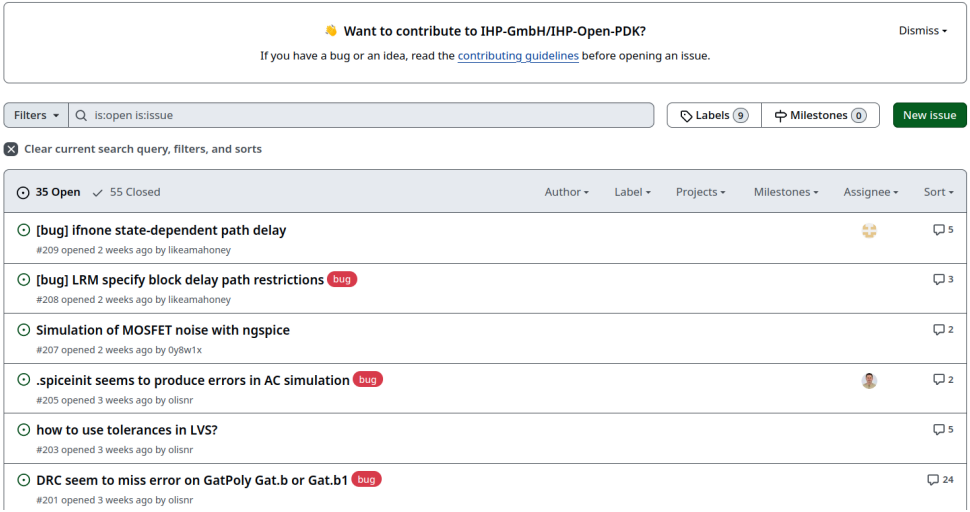


Figure 1: Issues open

Issues closed

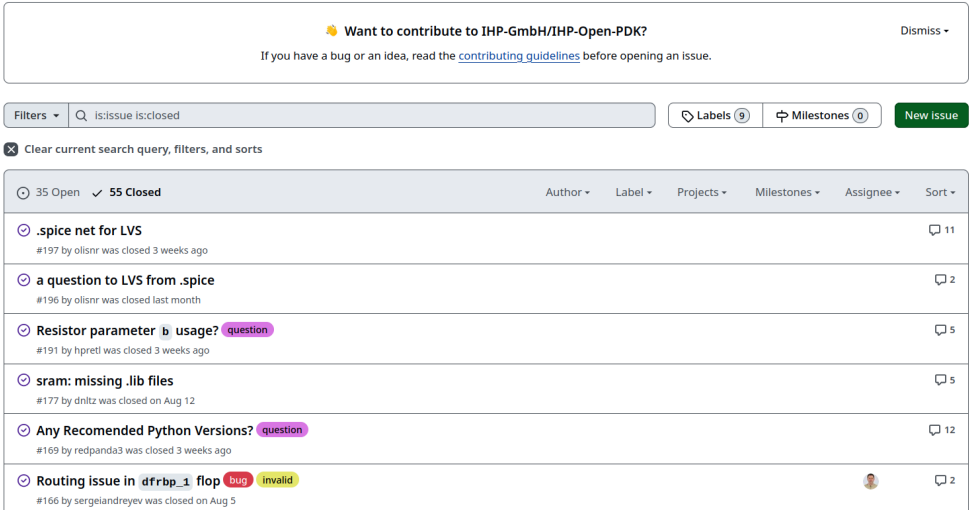


Figure 2: Issues closed

Pull requests closed

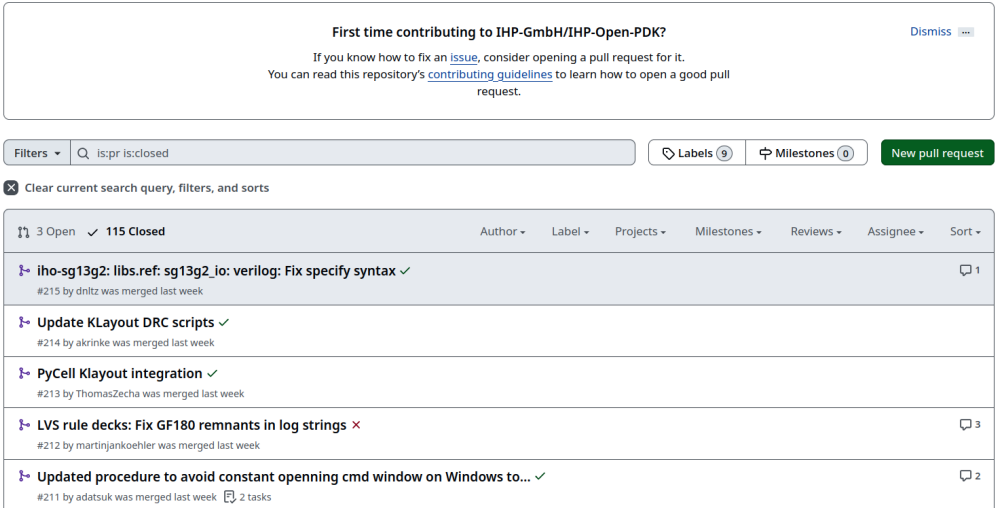


Figure 3: Pull requests closed

License file

The IHp open-source PDK is published with an Apache 2.0 license:

https://github.com/IHP-GmbH/IHP-Open-PDK/blob/main/LICENSE

Apache 2.0 is a permissive open-source license. Read more about different open-source licenses here:

https://choosealicense.com/licenses/

Tip:
Know about the permissions, conditions and limitations of the licenses you are using for your projects!

Section 3

Content of the PDK ihp-sg13g2

The README

The Readme file in the PDKs repository is the starting point for information about the content of the PDK.

https://github.com/IHP-GmbH/IHP-Open-PDK/blob/main/README.md

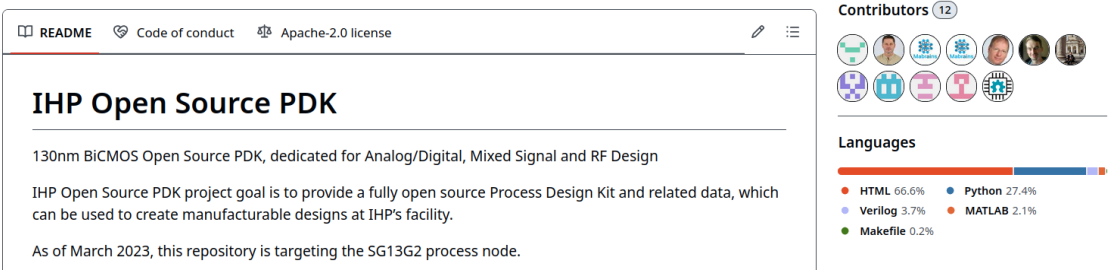


Figure 4: Readme

Project roadmap

A GANTT chart of the roadmap for the open-source PDK is available under this weblink. It shows the projects timeline (2022 - 2026):

https://github.com/IHP-GmbH/IHP-Open-PDK/blob/main/ihp-sg13g2/libs.doc/roadmap/open_PDK_gantt.png

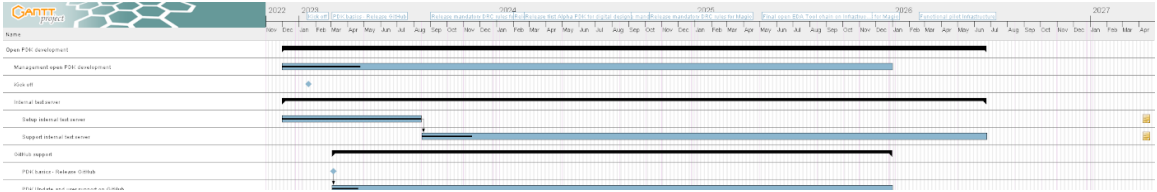


Figure 5: Gantt chart

Tool support

The PDK supports a bunch of tools with the included file formats. A list of the file types and formats follows later in this chapter.

The main tools for this course are

- OpenROAD flow scripts
- OpenROAD
- KLayout

The list of supported tools from the README in the PDK:

Supported EDA Tools

- ngspice
  - Download: https://ngspice.sourceforge.io/download.html
  - Source: https://sourceforge.net/p/ngspice/ngspice/ci/master/tree
- Xyce
  - Download: https://xyce.sandia.gov/downloads/executables
  - Source: https://github.com/Xyce/Xyce
- xschem
  - Source: https://github.com/StefanSchippers/xschem
- Qucs-S
  - Download: https://ra3xdh.github.io
  - Source: https://github.com/ra3xdh/qucs_s
- KLayout
  - Download: https://www.klayout.de/build.html
  - Source: https://github.com/KLayout/klayout
- OpenEMS
  - Source: https://github.com/thliebig/openEMS-Project
- OpenROAD
  - Source: https://github.com/The-OpenROAD-Project/OpenROAD
- OpenROAD-flow-scripts
  - Source: https://github.com/The-OpenROAD-Project/OpenROAD-flow-scripts

Figure 6: Suppoerted tools

Cells in the PDK

There are four different sets of cells (or devices) in the PDK:

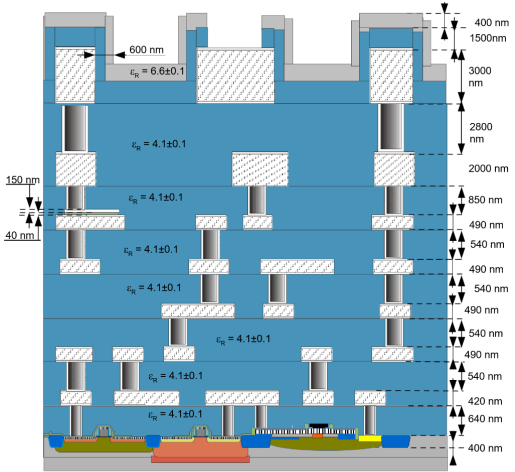
- Base cellset with limited set of standard logic cells
  - CDL, GDSII, LEF, Tech LEF
  - Liberty, SPICE Netlist, Verilog
- IO cellset
  - GDSII, LEF, Liberty (dummy), SPICE Netlist
- SRAM cellset
  - CDL, GDSII, LEF, Liberty, Verilog
- Primitive devices
  - GDSII

Other data in the PDK

- KLayout tool data:
  - layer property and tech files
  - DRC rules (minimal set)
  - PyCells
    - initial version of the wrapper API
    - sample cells
- Pcells (for reference only) libs.tech/pycell
- MOS/HBT/Passive device models for ngspice/Xyce
- xschem: primitive device symbols, settings and testbenches
- OpenEMS: tutorials, scripts, documentation
- SG13G2 Process specification & Layout Rules
- MOS/HBT Measurements in MDM format
- Project Roadmap Gantt chart

Layer stack

IHP sg13g2 Layers in a picture. Source PDF



Design rules

- The DRC (design rules check) with ORFS happens in KLayout.
- The data for the minimal and maximal checks is here:

https://github.com/IHP-GmbH/IHP-Open-PDK/tree/main/ihp-sg13g2/libs.tech/klayout/tech/drc

IHP-Open-PDK / ihp-sg13g2 / libs.tech / klayout / tech / **drc** /

Add file ···

akrinke Update KLayout DRC scripts ✓ 194ef09 · 5 months ago History

Name	Last commit message	Last commit date
..		
MissingRules_maximal.md	Update KLayout DRC scripts	8 months ago
README.md	Update KLayout DRC scripts	8 months ago
README_maximal.md	Update KLayout DRC scripts	8 months ago
README_minimal.md	KLayout DRC: added first version of full DRC rule deck	9 months ago
sg13g2_maximal.lydrc	Update KLayout DRC scripts	5 months ago
sg13g2_minimal.lydrc	Update KLayout DRC scripts	5 months ago

README.md

Minimum Rule Set - README
Maximum Rule Set - README, MissingRules

Figure 8: DRC files lydrc for KLayout

Layout versus Schematic

- The LVS check with ORFS happens in KLayout.
- The data for the LVS is here:

https://github.com/IHP-GmbH/IHP-Open-PDK/tree/main/ihp-sg13g2/libs.tech/klayout/tech/lvs

IHP-Open-PDK / ihp-sg13g2 / libs.tech / klayout / tech / **lvs** /

Add file ···

sergeiandreyev Merge pull request #224 from TinyTapeout/pr-fix-180-ref ··· 6a89d4e · 4 months ago History

Name	Last commit message	Last commit date
..		
images	Updating GUI menus to automatically detect active cell	9 months ago
rule_decks	Merge pull request #212 from martinjankoehler/fix-gf180-remnants	5 months ago
testing	Updating custom writer for LVS runset	7 months ago
README.md	Updating GUI menus to automatically detect active cell	9 months ago
run_lvs.py	Adding some deep/flat tests, Updating FETs/RFFETs derivations, Adding...	9 months ago
sg13g2.lvs	Updating chip derivation, Adding more logs for lvs run, updating lvs ...	9 months ago
sg13g2_full.lylvs	Merge pull request #224 from TinyTapeout/pr-fix-180-ref	4 months ago

Figure 9: LVS files lylvs for KLayout

Section 4

File formats

List of file types and formats

CDL: Circuit design language Link
LEF: Library Exchange Format Link
TechLEF: Technology LEF Link
GDS II: Graphic data system II Link
lib: Liberty timing and power file Link
sym: Symbol file (Xschem) Link
sch: Schematic file (Xschem) Link

lyp: Layer properties file (KLayout)
lyt: Technology mapping file (KLayout)
lydrc: DRC rules file (KLayout)
lylvs: LVS rule deck (KLayout)

Section 5

Example: Cell AO21

Cell AO21: GDS in KLayout

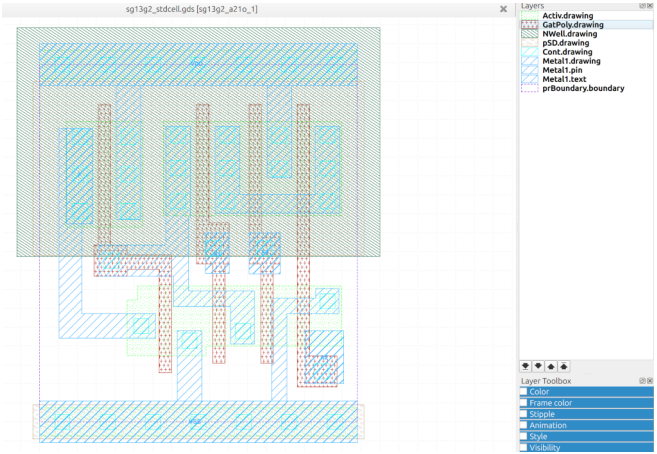


Figure 10: Cell AO21 GDS

Cell AO21: VERILOG HDL language

```
1  // type: AO21
2  `timescale 1ns/10ps
3  `celldefine
4  module sg13g2_a21o_1 (X, A1, A2, B1);
5      output X;
6      input A1, A2, B1;
7
8      // Function
9      wire int_fwire_0;
10
11     and (int_fwire_0, A1, A2);
12     or (X, int_fwire_0, B1);
13
14     // Timing
15     specify
16         (A1 => X) = 0;
17         (A2 => X) = 0;
18         if (A1 == 1'b1 & A2 == 1'b0)
19             (B1 => X) = 0;
20         if (A1 == 1'b0 & A2 == 1'b1)
21             (B1 => X) = 0;
22         ifnone (B1 => X) = 0;
23     endspecify
24 endmodule
25 `endcelldefine
```

Cell AO21: SPICE Netlist

```
* Library name: sg13g2_stdcell
* Cell name: sg13g2_a21o_1
* View name: schematic
* Inherited view list: spectre cmos_sch cmos.sch schematic veriloga ahdl
* pspice dspf
.subckt sg13g2_a21o_1 A1 A2 B1 VDD VSS X
XN0 net1 A1 net2 VSS sg13_lv_nmos w=640.00n l=130.00n ng=1 ad=0 as=0 pd=0 ps=0 m=1
XN1 net2 A2 VSS VSS sg13_lv_nmos w=640.00n l=130.00n ng=1 ad=0 as=0 pd=0 ps=0 m=1
XN2 net1 B1 VSS VSS sg13_lv_nmos w=640.00n l=130.00n ng=1 ad=0 as=0 pd=0 ps=0 m=1
XN3 X net1 VSS VSS sg13_lv_nmos w=740.00n l=130.00n ng=1 ad=0 as=0 pd=0 ps=0 m=1
XP0 net1 B1 net3 VDD sg13_lv_pmos w=1.000u l=130.00n ng=1 ad=0 as=0 pd=0 ps=0 m=1
XP1 net3 A1 VDD VDD sg13_lv_pmos w=1.000u l=130.00n ng=1 ad=0 as=0 pd=0 ps=0 m=1
XP2 net3 A2 VDD VDD sg13_lv_pmos w=1.000u l=130.00n ng=1 ad=0 as=0 pd=0 ps=0 m=1
XP3 X net1 VDD VDD sg13_lv_pmos w=1.12u l=130.00n ng=1 ad=0 as=0 pd=0 ps=0 m=1
.ends
* End of subcircuit definition.
```

Cell AO21: Circuit design language

```
1  ********************************************************************************
2  * Library Name: sg13g2_stdcell
3  * Cell Name:    sg13g2_a21o_1
4  * View Name:    schematic
5  ********************************************************************************
6
7  .SUBCKT sg13g2_a21o_1 A1 A2 B1 VDD VSS X
8  *.PININFO A1:I A2:I B1:I X:O VDD:B VSS:B
9  MN0 net1 A1 net2 VSS sg13_lv_nmos m=1 w=640.00n l=130.00n ng=1
10 MN1 net2 A2 VSS VSS sg13_lv_nmos m=1 w=640.00n l=130.00n ng=1
11 MN2 net1 B1 VSS VSS sg13_lv_nmos m=1 w=640.00n l=130.00n ng=1
12 MN3 X net1 VSS VSS sg13_lv_nmos m=1 w=740.00n l=130.00n ng=1
13 MP0 net1 B1 net3 VDD sg13_lv_pmos m=1 w=1.000u l=130.00n ng=1
14 MP1 net3 A1 VDD VDD sg13_lv_pmos m=1 w=1.000u l=130.00n ng=1
15 MP2 net3 A2 VDD VDD sg13_lv_pmos m=1 w=1.000u l=130.00n ng=1
16 MP3 X net1 VDD VDD sg13_lv_pmos m=1 w=1.12u l=130.00n ng=1
17 .ENDS
```

Cell AO21: LEF

```
MACRO sg13g2_a21o_1
  CLASS CORE ;
  ORIGIN 0 0 ;
  FOREIGN sg13g2_a21o_1 0 0 ;
  SIZE 3.36 BY 3.78 ;
  SYMMETRY X Y ;
  SITE CoreSite ;
  PIN A2
    DIRECTION INPUT ;
    USE SIGNAL ;
    ANTENNAMODEL OXIDE1 ;
      ANTENNAGATEAREA 0.2132 LAYER Metal1 ;
    PORT
      LAYER Metal1 ;
        RECT 2.81 0.405 3.215 0.965 ;
    END
  END A2
  PIN A1
    DIRECTION INPUT ;
    USE SIGNAL ;
    ANTENNAMODEL OXIDE1 ;
      ANTENNAGATEAREA 0.2132 LAYER Metal1 ;
    PORT
      LAYER Metal1 ;
        RECT 2.215 1.565 2.545 2 ;
    END
  END A1
```

Cell AO21: Schematic in XScheme

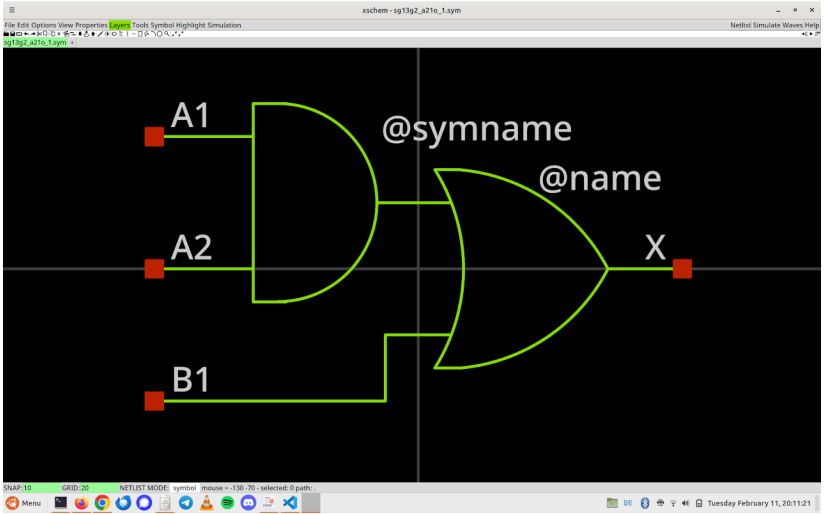


Figure 11: AO21 in XScheme

Liberty files

- Liberty files (.lib) contain information about timing, power and temperature of the cells.
- The ihp130-sg13g2 PDK contains six different Liberty files for the standard cells.
- These six files are categorized by nominal voltages and nominal temperatures:

Liberty file	Voltage	Temperature
sg13g2_stdcell_fast_1p32V_m40C.lib	1.32 V	-40° C
sg13g2_stdcell_fast_1p65V_m40C.lib	1.65 V	-40° C
sg13g2_stdcell_slow_1p08V_125C.lib	1.08 V	125° C
sg13g2_stdcell_slow_1p35V_125C.lib	1.35 V	125° C
sg13g2_stdcell_typ_1p20V_25C.lib	1.20 V	25° C
sg13g2_stdcell_typ_1p50V_25C.lib	1.50 V	25° C

- The PDK contains .lib files for RAM macros and io cells too.

Cell AO21: Liberty description

- The cell AO21 (first cell in the standard cell library list) has 622 lines of data in the lib file.
- The cell description contains:
  - First comes the leakage power for the whole truth table on the inputs.
  - After that each pin gets described with its timing and power characteristics
  - The order of the pins starts with the output. Order: X, A1, A2, B1

Cell AO21: Liberty file extract:

```
  cell (sg13g2_a21o_1) {
    area : 12.7008;
    cell_footprint : "AO21";
    cell_leakage_power : 158.343;
    leakage_power () {
      value : 163.606;
      when : "!A1&!A2&!B1";
    }

...

    pin (X) {
      direction : "output";
      function : "((A1*A2)+B1)";

...

    pin (B1) {
      direction : "input";
      max_transition : 2.5074;

...
```

Cell A021: Liberty file - Output Pin X

```
    pin (X) {
      direction : "output";
      function : "((A1*A2)+B1)";
      min_capacitance : 0.001;
      max_capacitance : 0.3;
      timing () {
        related_pin : "A1";
        sdf_cond : "B1 == 1'b0";
        timing_sense : positive_unate;
        timing_type : combinational;
        when : "!B1";
        cell_rise (TIMING_DELAY_7x7ds1) {
          index_1 ("0.0186, 0.0966, 0.174, 0.3294, 0.6408, 1.263, 2.5074");
          index_2 ("0.001, 0.0234, 0.039, 0.0648, 0.108, 0.18, 0.3");
          values ( \
            "0.0807687, 0.15146, 0.195473, 0.267231, 0.386899, 0.586296, 0.918436", \
            "0.116643, 0.187459, 0.231451, 0.303439, 0.423359, 0.623019, 0.954578", \
            "0.142067, 0.214006, 0.258032, 0.330026, 0.450148, 0.649615, 0.981533", \
            "0.179515, 0.252822, 0.297045, 0.368987, 0.488854, 0.688005, 1.02054", \
            "0.23237, 0.310828, 0.354629, 0.426105, 0.545922, 0.745166, 1.07728", \
            "0.304206, 0.395567, 0.439475, 0.510596, 0.629839, 0.829268, 1.1608", \
            "0.396364, 0.507762, 0.554141, 0.625639, 0.74573, 0.944857, 1.27636" \
          );
        }
```

Cell A021: Liberty file - Output Pin X

```
1     pin (X) {
2       direction : "output";
```

This is an output pin.

```
1       function : "((A1*A2)+B1)";
```

The boolean function is X = (A1 AND A2) OR B1

```
1       min_capacitance : 0.001;
2       max_capacitance : 0.3;
```

The minimum and maximum load capacitance. You must not attach more than 0.3 pF to the output X of the gate.

Cell A021: Liberty file - Output Pin X

```
1       timing () {
2         related_pin : "A1";
```

This only relates to the timing arc from input A1. The timings are different for all input pins.

```
1         sdf_cond : "B1 == 1'b0";
```

This timing is only relevant if the input B1 is 0. The reason is that if B1 is 1, then the output is 1 anyway and A1 has no influence on the output X because of the boolean function. This is for SDF export for gate-level simulation. See “when” below for STA tools.

Cell A021: Liberty file - Output Pin X

```
1         timing_sense : positive_unate;
```

The output changes in the same direction as the input. When the input rises, then the output rises. Alternatives could be “negative_unate” for inverting behaviour like inverters and NAND or “non_unate” for XOR.

```
1         timing_type : combinational;
```

This is combinational gate and not a sequential one like a D-Flipflop.

```
1         when : "!B1";
```

Same as “sdf_cond”, but for static-timing analysis (STA) tools.

Cell A021: Liberty file - Timing table - Output Pin X

```
1  cell_rise (TIMING_DELAY_7x7ds1) {
2    index_1 ("0.0186, 0.0966, 0.174, 0.3294, 0.6408, 1.263, 2.5074");
3    index_2 ("0.001, 0.0234, 0.039, 0.0648, 0.108, 0.18, 0.3");
4    values ( \
5      "0.0807687, 0.15146, 0.195473, 0.267231, 0.386899, 0.586296, 0.918436", \
6      "0.116643, 0.187459, 0.231451, 0.303439, 0.423359, 0.623019, 0.954578", \
7      "0.142067, 0.214006, 0.258032, 0.330026, 0.450148, 0.649615, 0.981533", \
8      "0.179515, 0.252822, 0.297045, 0.368987, 0.488854, 0.688005, 1.02054", \
9      "0.23237, 0.310828, 0.354629, 0.426105, 0.545922, 0.745166, 1.07728", \
10     "0.304206, 0.395567, 0.439475, 0.510596, 0.629839, 0.829268, 1.1608", \
11     "0.396364, 0.507762, 0.554141, 0.625639, 0.74573, 0.944857, 1.27636" \
12   );
13 }
```

This is a two dimensional table with CellDelay=f(input slew,load capacitance). For an input slew time of 0.0966ns and a load capacitance of 0.039pF, we have cell delay of 0.231451ns. The tool interpolates between those points in the table for intermediate values. Note that the delay varies widely between 0.08ns and 1.27ns (=16x).

Section 6

Ruleset documents

Layout rules document

https://github.com/IHP-GmbH/IHP-Open-PDK/tree/main/ihp-sg13g2/libs.doc/doc

Figure 12: Layour rules document

Process specification document

https://github.com/IHP-GmbH/IHP-Open-PDK/tree/main/ihp-sg13g2/libs.doc/doc

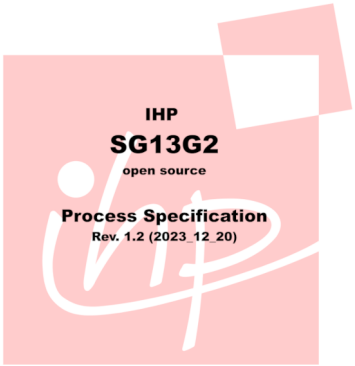


Figure 13: Process specification

Section 7

Questions Timing

Questions Timing

Propagation Delay AO21_2

- What is the propagation delay of cell AO21_2?
- Compare the propagation delay between AO21 and AO21_2

Limiting path on in counter design

- What is the critical path in the counter design?
- Add additional registers at the primary inputs and outputs of the counter and only investigate the internal register to register paths
- What happens when you increase the counter width?
- What is the maximum clock frequency?

Section 8

Questions Pipelining

Questions Pipelining

Pipelining with feedback loop

- Why is it difficult to increase the clock frequency of a counter with pipelining?