Open-Silicon MPW March 2026 Documentation Summary and Submission Guidance

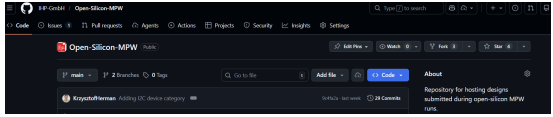
Leibniz Institute
for high
performance
microelectronics

Krzysztof Herman

Open-Silicon-MPW

Purpose of the Repository

`https://github.com/IHP-GmbH/Open-Silicon-MPW`



- Aggregates production-ready IPs as submodules
- dedicated locations for each MPW run (March2026, July2026 ...)
- IP development happens in standalone repositories
- Submodule requests include the IP under certain category
- Four categories: Analog, Digital, RF, Mixed-Signal

High-Level Structure

- Category directories live under `March-2026/<Category>`
- Submodules follow naming: `IHP__<abbr><4digits>`
- Abbreviation must match the ones defined in `IP-Categories.md`

Caution !

Only the categories from `IP-Categories.md` are allowed

Category and Abbreviation Rules

`IHP__OTA1336`

- Subcategories are standardized and mapped to abbreviations
- Abbreviations are used in the top-level IP name and repo path
- New subcategories are requested via GitHub issue

```
.
- dependencies
- doc
- measurements
- OTA1336-main
- README.md
- release
```

IP Categories Overview

Analog

BGR, CMP, OPA, OTA, LDO
VREF, CP, TSENSE, OSC, PMIC

Digital

MCU, MPU, RISCV, MEMCTRL, DMA
SEC, CRYPTO, NOC, GPIO, TIMER
UART, SPI, I2C, I2Cdev

Mixed-Signal

ADC, DAC, PLL, SDM
CDR, SERDES, MSPHY, SoC

RF

LNA, PA, MIX, VCO
RFPLL, FSYN, RFFE, RFSW, BALUN

IP Development Steps (Summary)

- Select category and subcategory abbreviation
- Clone Open-Silicon-MPW to get `gen_structure.py`
- Generate IP repo: `python3 gen_structure.py IHP <subcategory>`
- Implement design and keep structure intact
- Maintain `doc/info.json` and TRL document
- Publish repo and optionally sync shared workflows

Common Repository Layout

- <IP>-main/ with layout, netlist, schematic, verification
- doc/ with Datasheet, Specification, info.json, TRL
- release/version with GDS, netlist, docs
- dependencies and measurements/

```
- dependencies/
- doc/
- measurements/
- OTA1336-main/
- README.md
- release/v0.1.0/
```

Info

The <IP>-main contains mutable design data while release/ should contain immutable release views (data) specific to an iteration in silicon or IP a particular version.

Generation of the IP structure

```
python3 gen_structure.py IHP <subcategory>
```

- Clone Open-Silicon-MPW to get `gen_structure.py`
- Copy the `gen_structure.py` file to the place you want to store your IP
- Generate the structure by executing the script
- Distribute the files and keep structure intact

IP views distribution

Main rule

The repository and data distribution does not imply a workflow flow. Take your data from your flow and distribute it according the directory structure. In the documentation describe your flow.

Design data

The design data are segmented by function (schematic, layout, netlist etc.) and at the very end by tool specific views (klayout, magic, xschem, Qucs-S).

Naming convention

Follow the naming convention of your IP. All the main views of your final cell should have the same name <IP> with the corresponding extension ie. OTA1336.gds OTA1336.sch.

Metadata

Caution!

Fill carefully the `doc/info.json` file.

- assign the license
- provide the link of your repository
- provide the pdk commit as "pdk_version"
- specify process: SG13G2, SG13CMOS5L,
- information about the layout (top cell name, x and y size of the seal ring)
- provide relative paths to the final GDS and spice/CDL netlist used for LVS (it is critical for automatic checks of DRC and LVS)

Technology Readiness Level Templates (Purpose)

- TRL checklists help self-evaluate IP maturity
- Templates provided for Analog, Mixed-Signal, RF, and Digital Hard IP
- self evaluate your design and provide corresponding level in `info.json` file

Physical verification DRC and LVS

Minimal pre check DRC

```
python3 $PDK_ROOT/$PDK/libs.tech/klayout/tech/drc/run_drc.py \
  --precheck_drc --mp=32 --no_offgrid --density_thr=32 \
  --no_angle --disable_extra_rules --no_recommended --path=your.gds
```

Regular DRC

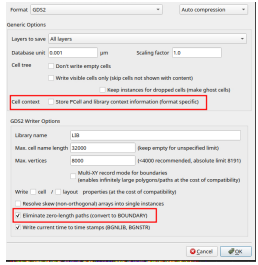
```
python3 $PDK_ROOT/$PDK/libs.tech/klayout/tech/drc/run_drc.py \
   --mp=32 --density_thr=32 --path=your.gds
```

Regular LVS

```
python3 $PDK_ROOT/$PDK/libs.tech/klayout/tech/lvs/run_lvs.py --layout=your.gds
     --layout_netlist=your.cdl
```

Release your final data

Save a copy of your final gds in under `release/v0.1.0/gds/` directory with the following options



Save a copy of your final netlist for LVS check in under `release/v0.1.0/netlist/` directory

Turn your data into github repository

Github repository setup

Go to your github account and create a repository named using the name:
`IHP__<abbr><4digits>`

On your local machine inside your project execute:

```
git init
git add .
git commit -s -m "Initial commit"
git branch -M main
git remote add origin https://github.com/<org>/<repo>.git
git push -u origin main
```

Automatic workflows

Workflow check

The generator script automatically sets up a `check-workflow-sync.yml` script which will synchronize the central workflows with your repository

You need to activate it!

Workflow activation

manually copy the corresponding workflow from .github/workflows-staged/ to .github/workflows/

Submission process

Open an issue on Open-Silicon-MPW github webpage with the title:
IHP__<abbr><4digits> and the following text:

```
## Submodule request for March-2026 Open-Silicon MPW
### .gitmodules snippet
[submodule "March-2026/<Category>/IHP__<subcategory-abbrev><4digits>"]
  path = March-2026/<Category>/IHP__<subcategory-abbrev><4digits>
  url = https://github.com/<org>/<repo>.git
```

The hard deadline for this activity is 30-th March 2026, 8 AM CET time if you can do do it earlier

Thank you for your attention!

IHP GmbH – Innovations for High Performance Microelectronics
Im Technologiepark 25
15236 Frankfurt (Oder)
Tel. +49 (0) 335 5625 380
herman@ihp-microelectronics.com

Leibniz Association