Tutorial 7 - Part 1
iEDA Infrastructure

Zengrong Huang, Xingquan Li

Peng Cheng Laboratory

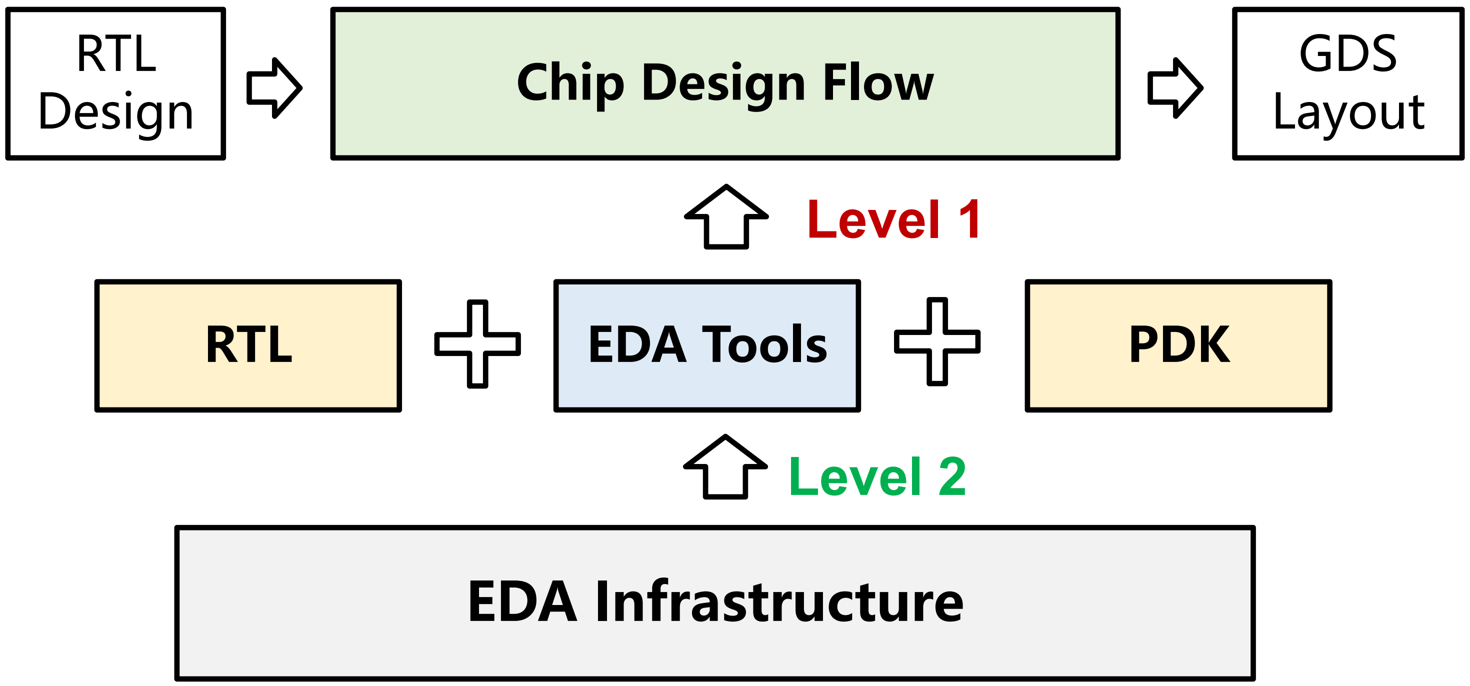
01 iEDA Infrastructure

02 Feature

03 Flow

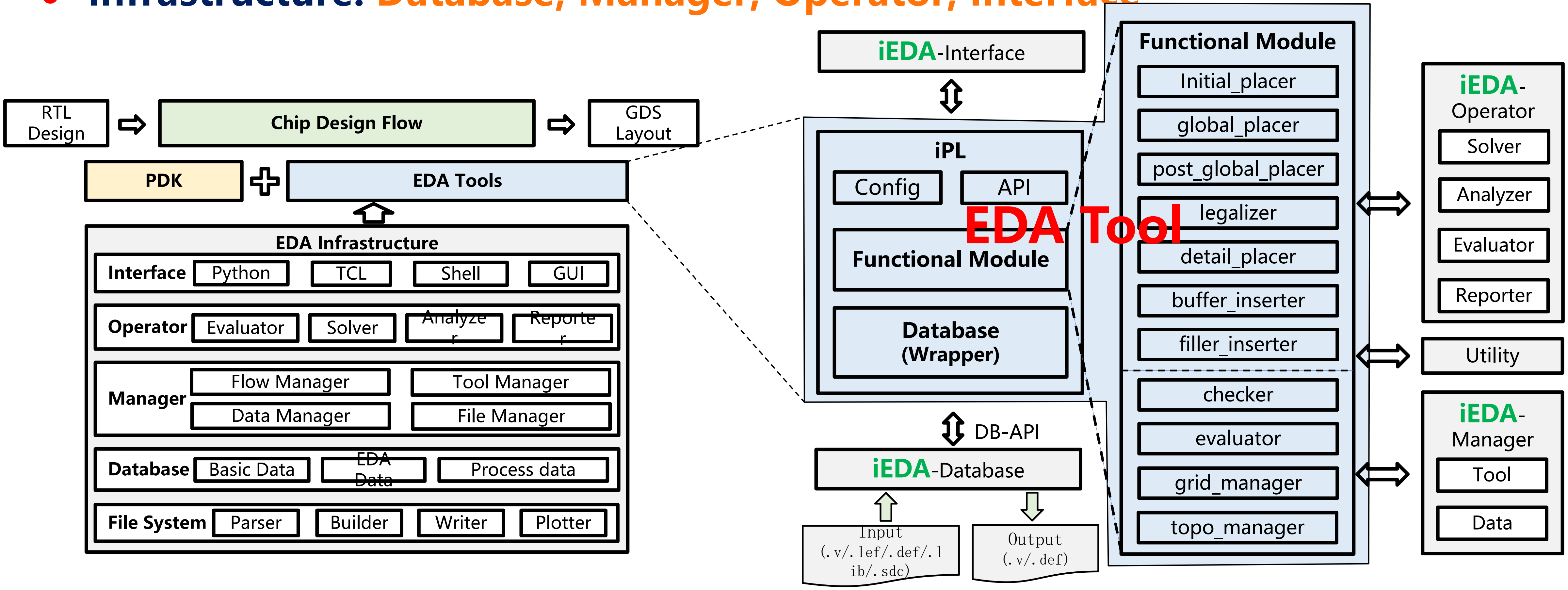
We Need Infrastructure

- **Level 1:** Open-source tools, RTLs, PDKs support chip design
- **Level 2:** Open-source Infrastructure supports EDA development



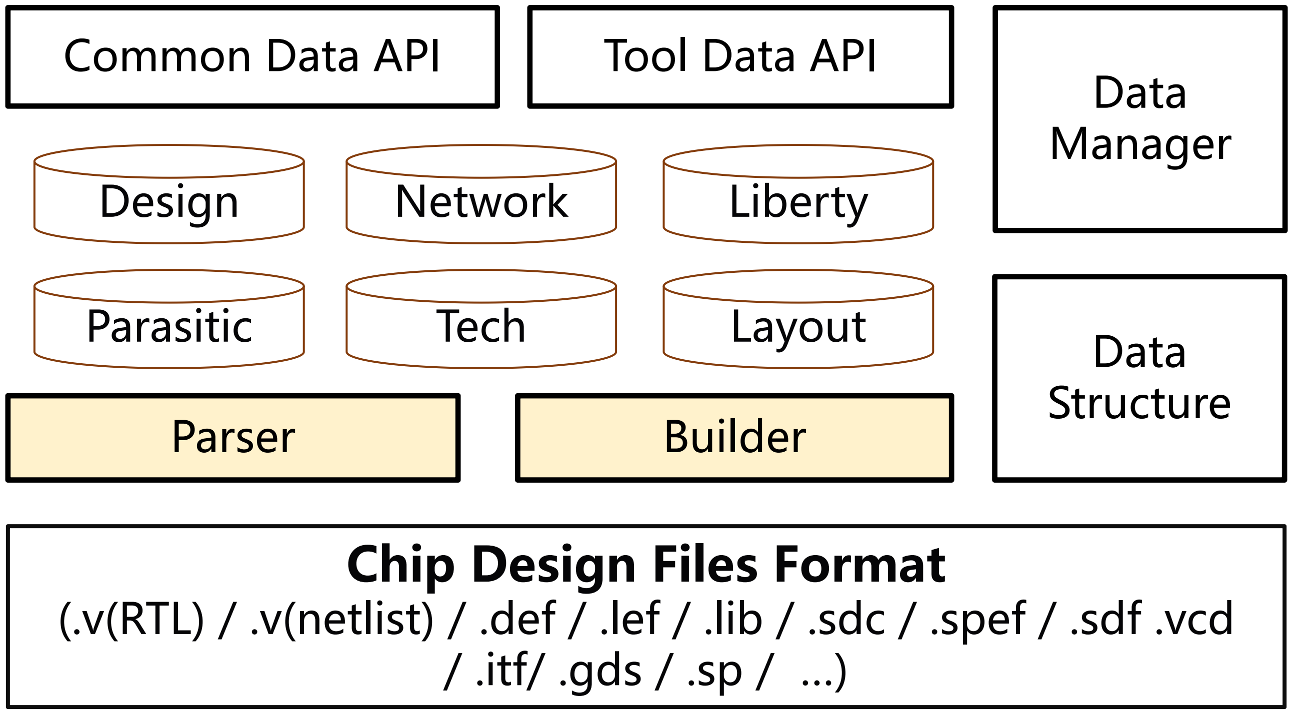
iEDA Infra. Support EDA Tool

- To fast develop high-quality EDA tool, we need a Software Development Kit (SDK)
- iEDA can be used to support developing EDA tool or algorithm
- **Infrastructure: Database, Manager, Operator, Interface**



Parsers and Database

- **Parser: Verilog, SPEF, Liberty, SDF, VCD, SDC, LEF/DEF, ITF, and GDSII**
- **Database: Design, Layout, Tech, Timing, Parasitic, Network**



Managers

Platform Manager

Data

- Config
- ChipData
- Interactive
- Proc Data
- …

Flows

- Initialize
- Input
- Process
- Output

Tools

- Floorplan
- NetOpt
- Placement
- CTS
- TimingOpt
- Legalization
- Routing
- Filler
- DRC
- …

Features

- Summary
- Density
- Wire Length
- Congestion
- Profiles
- …

Reports

- Statistic
- Evaluation
- Flow Results
- Timing
- DRC
- …

Files

- Config
- Design
- Procedure
- Serialize
- …

Interfaces

TCL

- Flows
- DB
- Tools
- Evaluation
- Features
- Reports
- GUI

```
├── tcl_config
├── tcl_contest
├── tcl_eval
├── tcl_feature
├── tcl_flow
├── tcl_gui
├── tcl_icts
├── tcl_idb
├── tcl_idrc
├── tcl_ifp
├── tcl_ino
├── tcl_instance
├── tcl_ipdn
├── tcl_ipl
├── tcl_ipw
├── tcl_irt
├── tcl_ista
├── tcl_ito
├── tcl_report
└── tcl_util
```

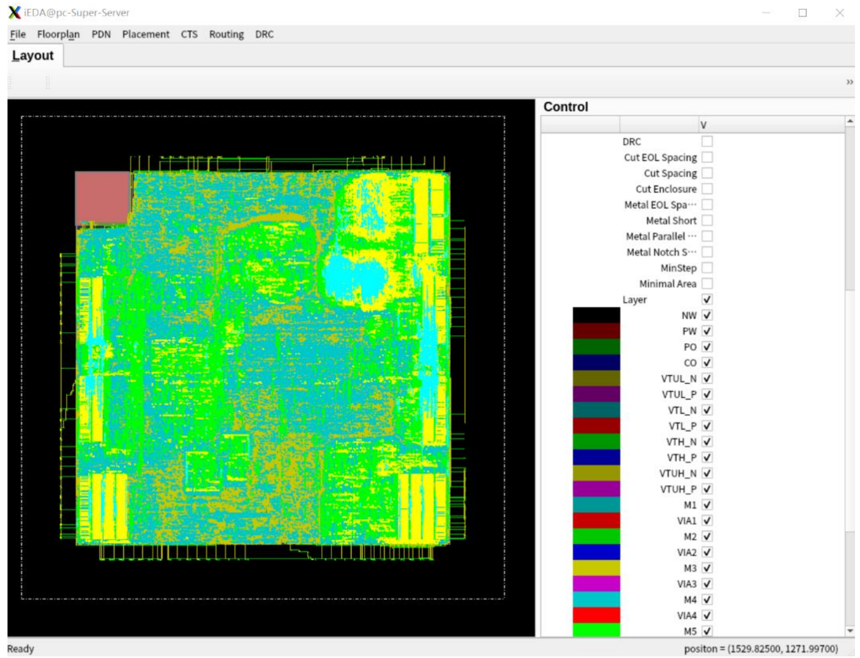
Python

- Flows
- DB
- Tools
- Evaluation
- Features
- Reports

```
├── py_config
├── py_eval
├── py_feature
├── py_flow
├── py_icts
├── py_idb
├── py_idrc
├── py_ifp
├── py_imp
├── py_ino
├── py_instance
├── py_ipdn
├── py_ipl
├── py_ipw
├── py_irt
├── py_ista
├── py_ito
└── py_report
```

GUI

- File Operation
- Layout View
- Layers Control
- Shape Setting
- Instance Options
- Net Options
- PDN Options
- Track Grid
- DRC View
- Clock Tree View



Evaluator: Horizontal Comparison

- Compare and analyze the Q&R
  - **Designs and Flows**
  - Tools and Algorithms

part metrics	flow1	flow2
detail routing HPWL (um)	10879081	11025675
final wirelength (um)	11471595	12071042
setup slack (ns)	-0.492	-0.484
hold slacke (ns)	0.426	0.427
suggest frequency (MHz)	345.804	346.784
power (mW)	0.956	0.966
#DRC	755	643

Flow Comparison

design	aes	aes_core
PDK	sky130	sky130
instance area	408034.7568	371050.9776
IO pin	76	520
instances	45854	42044
nets	30634	28536
core_area	1352765.88	1230601.766
total wire length	2695657	2809505
total vias	280870	271884
setup_slack (max)	14.7	14.73
hold_slack (min)	0.22	0.4
suggest freq (MHz)	188.6792453	189.7533207

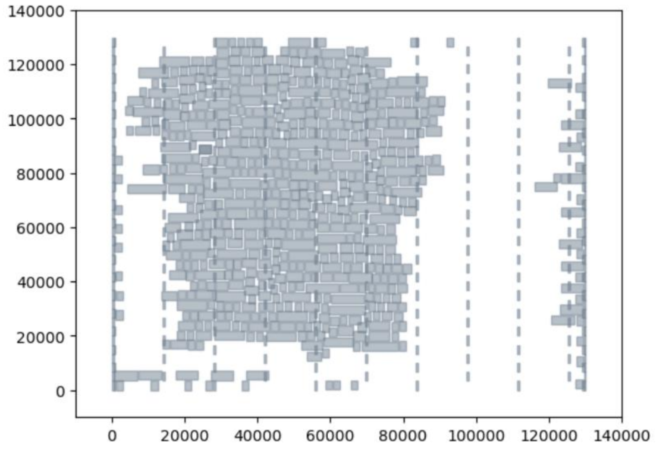
Design Comparison

Evaluator: Horizontal Comparison

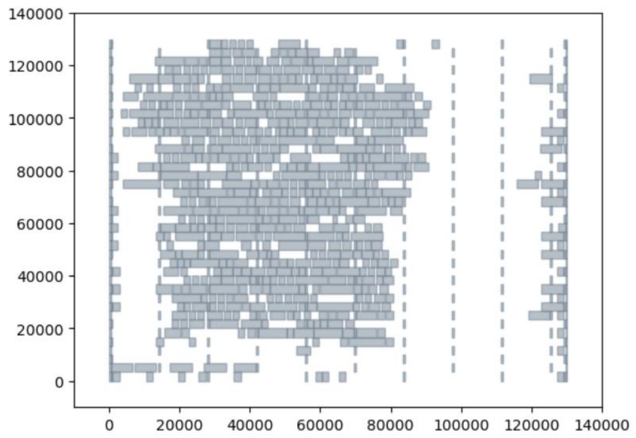
- Compare and analyze the Q&R
  - Designs and Flows
  - **Tools and Algorithms**

part metrics	abacus	tetris
global placement HPWL	10127910	10127910
legalization HPWL	10426323	13168231
detail placement HPWL	9901517	10928985
detail placement STWL	10637190	11674987
maximum STWL	431085	415325
total movement	795829	8705103
maximum movement	5684	218214
average congestion	0.8215	0.8134
total overflow	49	49
peak bin density	1	1
legalization runtime (s)	0.0667	0.0064

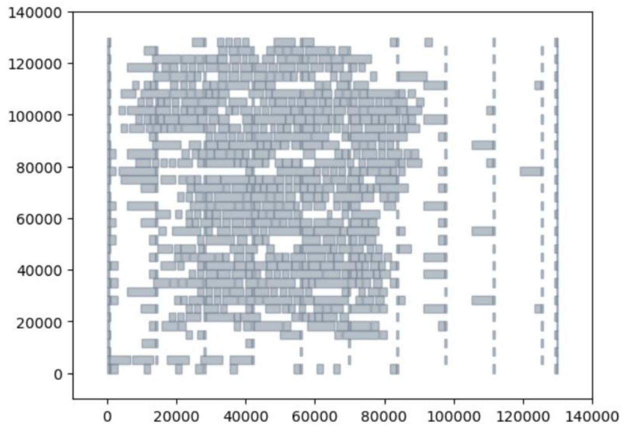
Algorithm Comparison



Input



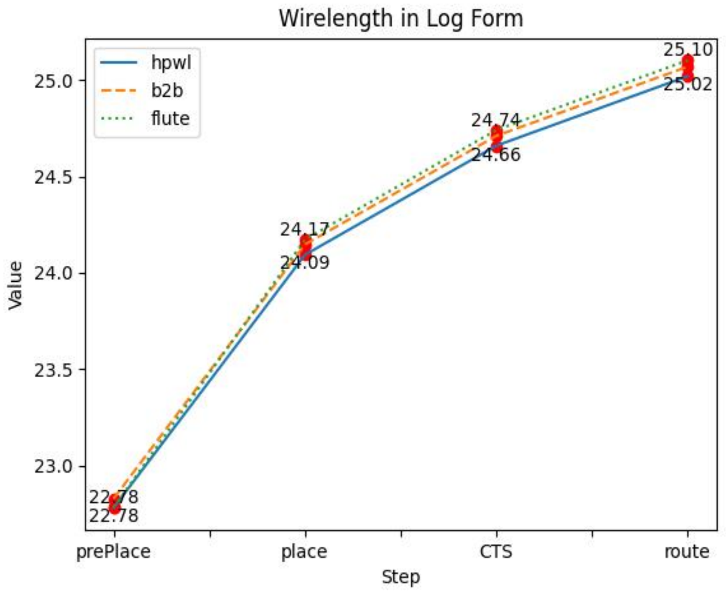
Abacus



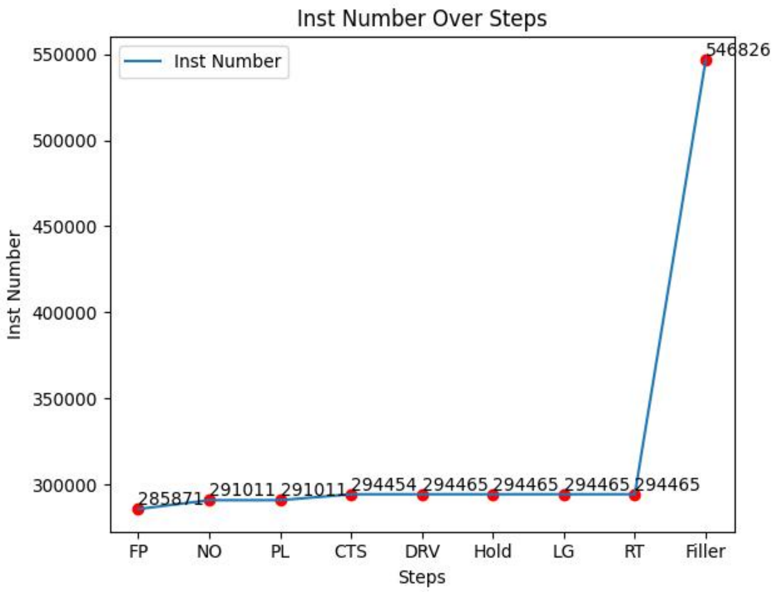
Tetris

Analyzer : Vertical Comparison

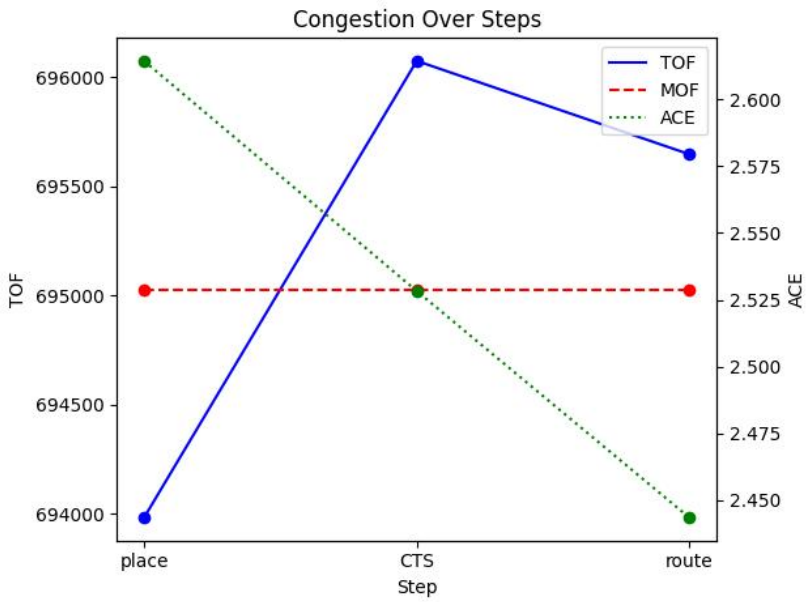
- Analyzing the numerical changes of an indicator **across different stages**
- Differences from: 1) data change, and 2) differences evaluation models
- Usage: evaluating the design quality, analyzing the margin, and optimizing collaboratively



Wirelength



#Instances



Congestion

01 iEDA Infrastructure

02 **Feature**

03 Flow

iEDA Features

For IC Designer

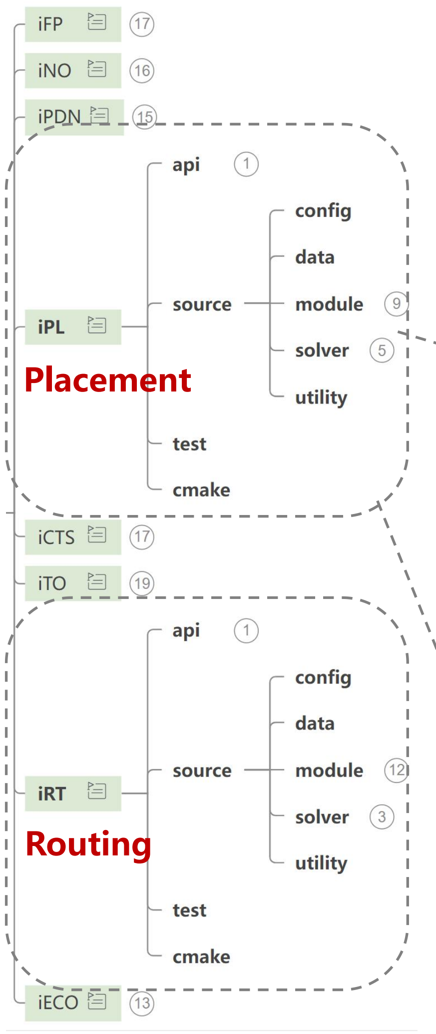
- Establish the backend physical design flow
- Support TCL language
- Provid rich performance evaluations
- Support layout visualization
- Support data snapshots, and toolchain data recovery
- User-friendly, with user guides and community support

For EDA Coder

- Provide Netlist -> GDSII interface
- Offer multi-language version interfaces (C++, Python, TCL, RUST)
- Offer rich performance evaluation interfaces
- Provide kinds of analysis and debugging tools
- Unified development framework, basic infrastructure
- Development manuals, community support

Uniform Software Framework and API

Software Structure



API

```
.
├── python
│   ├── py_config
│   ├── py_eval
│   ├── py_feature
│   ├── py_flow
│   ├── py_icts
│   ├── py_idb
│   ├── py_idrc
│   ├── py_ifp
│   ├── py_ino
│   ├── py_instance
│   ├── py_ipdn
│   ├── py_ipl
│   ├── py_irt
│   ├── py_ista
│   ├── py_ito
│   └── py_report
└── tcl
    ├── tcl_config
    ├── tcl_eval
    ├── tcl_feature
    ├── tcl_flow
    ├── tcl_gui
    ├── tcl_icts
    ├── tcl_idb
    ├── tcl_idrc
    ├── tcl_ifp
    ├── tcl_ino
    ├── tcl_instance
    ├── tcl_ipdn
    ├── tcl_ipl
    ├── tcl_irt
    ├── tcl_ista
    ├── tcl_ito
    ├── tcl_qt
    ├── tcl_report
    └── tcl_util
```

Application

```python
def run_iPL(self):
    ieda.flow_init(config="./iEDA_config/flow_config.json")
    ieda.db_init(config="./iEDA_config/db_default_config.json")
    ieda.db_init(sdc_path = "./sdc/asic_top_SYN_MAX_1.sdc")
    ieda.def_init(path="./result/iTO_fix_fanout_result.def")
    ieda.run_placer(config="./iEDA_config/pl_default_config.json")
    ieda.def_save(path="./result/iPL_result.def")
    ieda.netlist_save(path="./result/iPL_result.v")
    ieda.report_db(path="./result/report/pl_db.rpt")
    ieda.flow_exit()
```

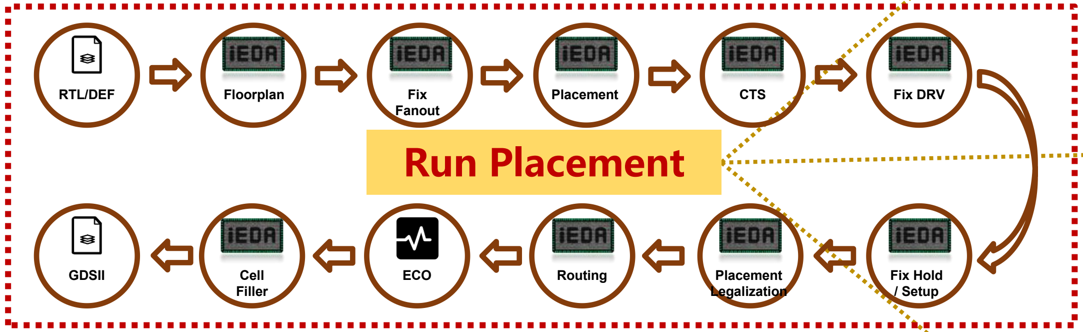
Python

```tcl
flow_init -config ./iEDA_config/flow_config.json
db_init -config ./iEDA_config/db_default_config.json
source ./script/DB_script/db_path_setting.tcl
source ./script/DB_script/db_init_sdc.tcl
source ./script/DB_script/db_init_lef.tcl
def_init -path ./result/iTO_fix_fanout_result.def
run_placer -config ./iEDA_config/pl_default_config.json
def_save -path ./result/iPL_result.def
netlist_save -path ./result/iPL_result.v -exclude_cell_names {}
report_db -path "./result/report/pl_db.rpt"
flow_exit
```

TCL

Multiple Programming Language

✓ Support **C++**、**RUST**、**TCL**、**Python**



C++ API

```cpp
/// run placer
if (PLFConfig::getInstance()->is_run_placer()) {
  if (tmInst->autoRunPlacer(PLFConfig::getInstance()->get_ipl_path())) {
  }
}
```

TCL API

```tcl
#===========================================================
##   read def
#===========================================================
def_init -path ./result/iTO_fix_fanout_result.def

#===========================================================
##   run Placer
#===========================================================
run_placer -config ./iEDA_config/pl_default_config.json
```

Python API

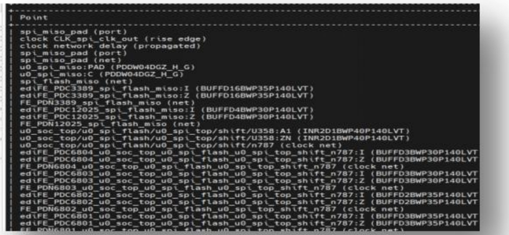
```python
def run_placer(self, input_def : str):
    self.read_def(input_def)

    path = self.path_manager.get_workspace().get_config_ieda(FlowStep.place)
    ieda.run_placer(path)
```

iEDA Code

Evaluation

✓ iSTA/iPA: Timing and Power

- Setup/Hold
- Recoverry/Removal
- Mlticycle
- Clock Gate
- Power
- IR Drop



✓ iDRC: Design Rule Vios

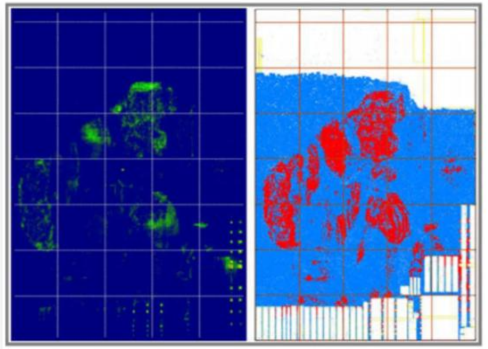
- Wire Connection
- Spacing Check、 Enclosure Check、PRL Check、Metal Short、Minimal Area、MinStep
- DRC report、DRC visulization

Drc Summary

DRC Type	Number
Cut Different Layer Spacing	0
Cut EOL Spacing	0
Cut Enclosure	610489
Cut EnclosureEdge	0
Cut Spacing	264504
Metal Corner Filling Spacing	0
Metal EOL Spacing	8404599
Metal JogToJog Spacing	0
Metal Notch Spacing	1621088
Metal Parallel Run Length Spacing	3253512
Metal Short	6429817
MinHole	94
MinStep	7794204
Minimal Area	744067

✓ iEVAL: Wirelength, Density, Congestion

- Density: Macro, Std Cell, Pin
- Congestion: GlobalNet、RUDYNet、LUTRYDY、
- Wirelength: WLD, HPWL, P2P, Steiner, eGR, GR, DR



✓ DB get: Count Basic Data in DB

- Layout: Area, Utilization, Row
- Cell: Nums, Pins, Area,
- Net: Nums, Fanout, Length
- Path: Nums, Depth
- Layer: Wire Density, Overflow

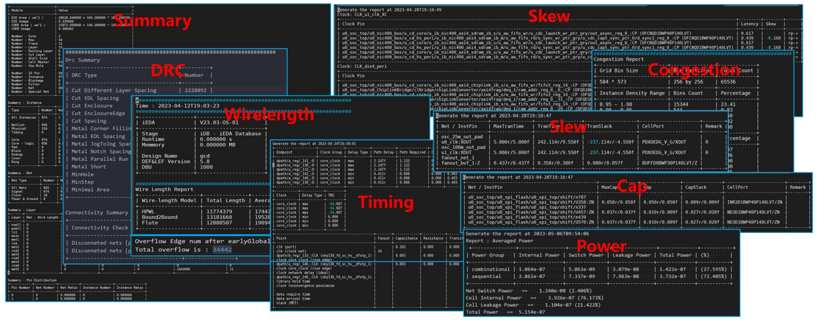
##

Summary

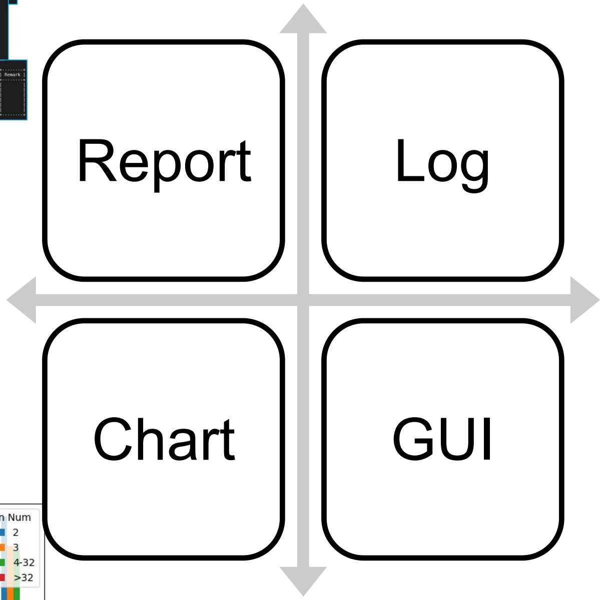
Module	Value
DIE Area (um^2)	2249940.000000 = 1499.960000 * 1500.000000
DIE Usage	0.240103
CORE Area (um^2)	1340542.224000 = 1160.040000 * 1155.600000
CORE Usage	0.402985
Number - Site	9
Number - Row	1284
Number - Track	20
Number - Layer	32
Number - Routing Layer	10
Number - Cut Layer	10
Number - GCell Grid	5
Number - Cell Master	16314
Number - Via Rule	501
Number - IO Pin	110
Number - Instance	362106
Number - Blockage	21
Number - Filler	0
Number - Net	376471
Number - Special Net	5

Multiple Data Analysis and Debug Methods

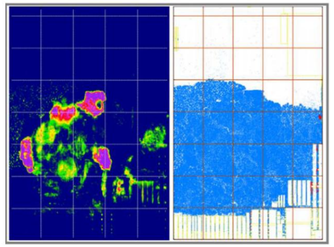
Report

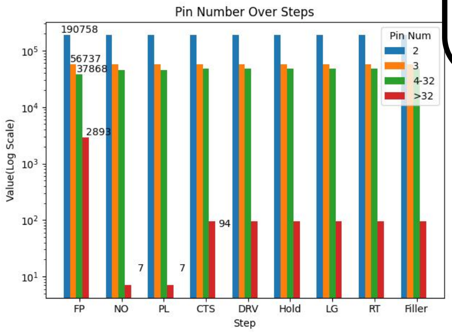


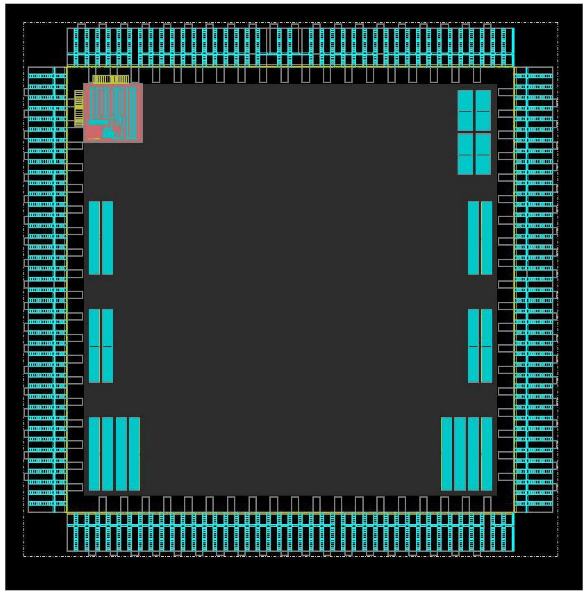
Log



Feature Chart







GUI

Data Snapshot & Recovery

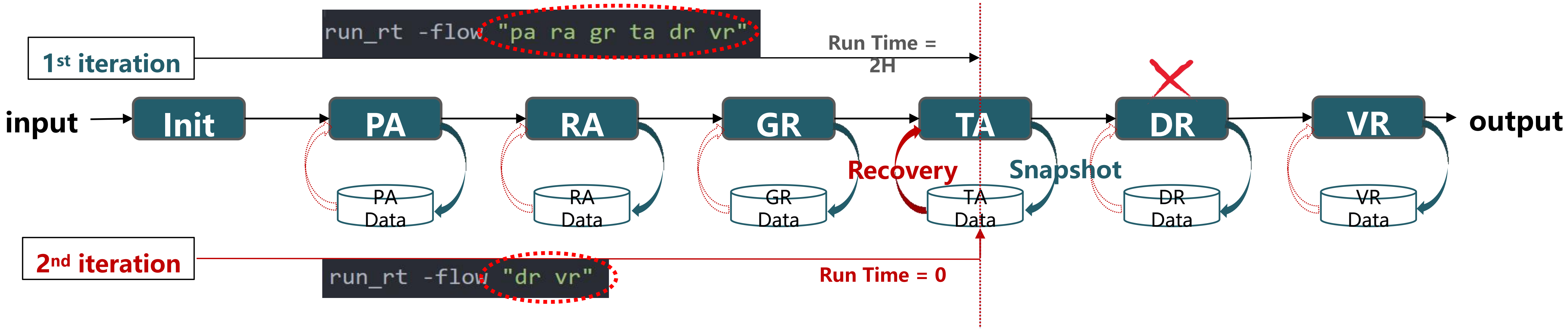
- iEDA adopts **serialization and deserialization** to achieve data snapshot and recovery:

Data Snapshot

- Tool Process Data
- Result Data
- Algorithm Data

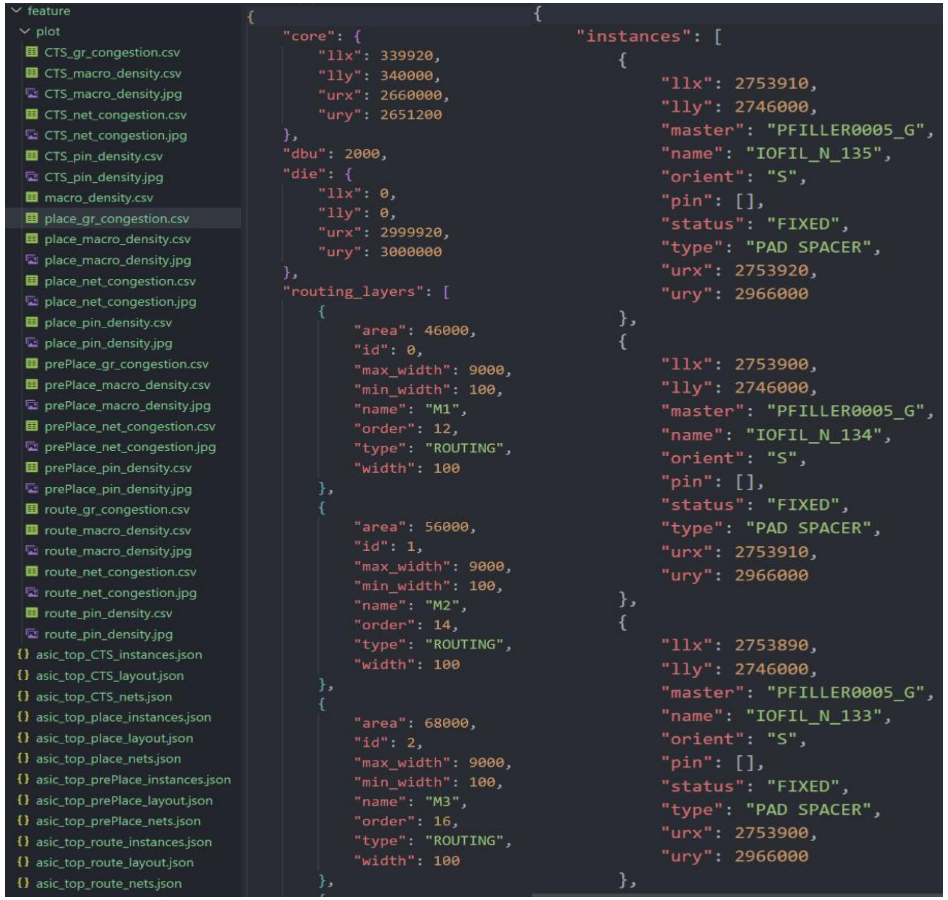
Data Recovery

- Tool Process Data
- Result Data
- Algorithm Data

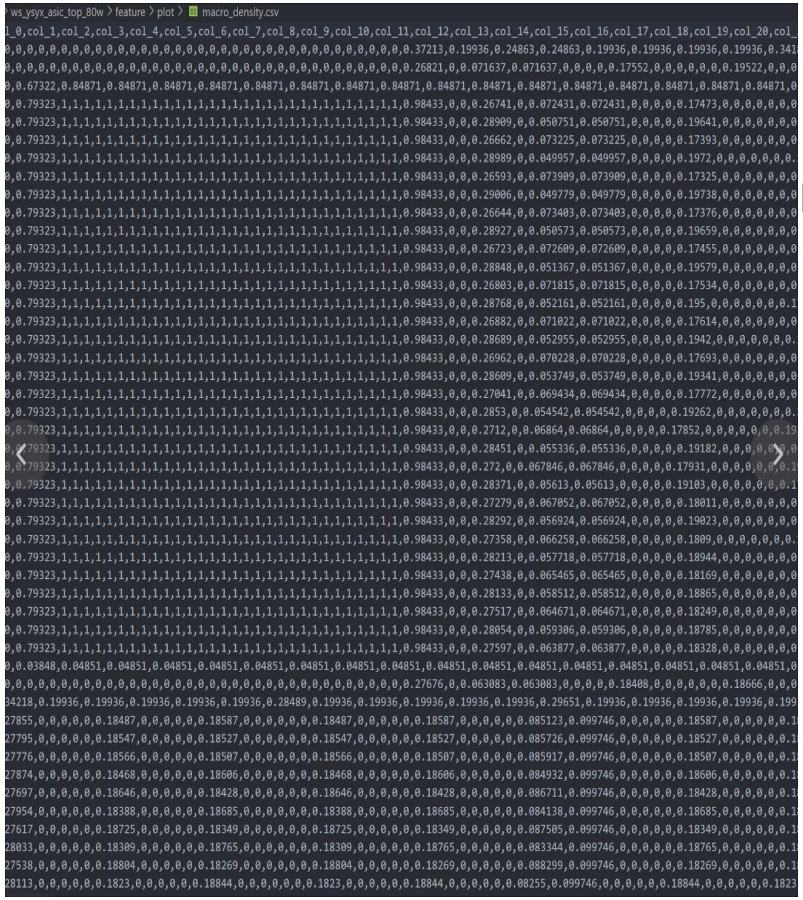


Feature Extraction

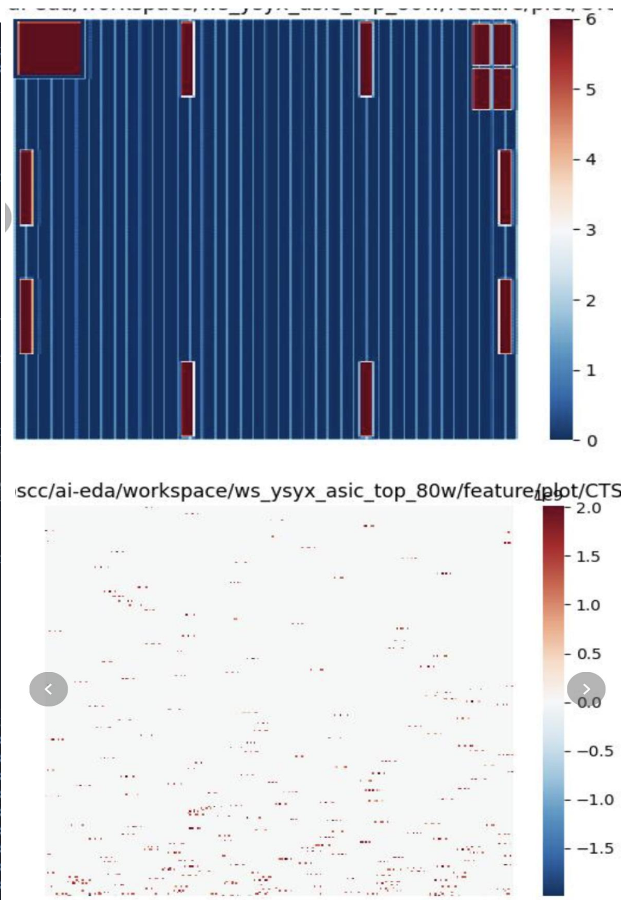
Basic Data



Evaluation Data



Feature Map



Rich API and Documentation

C++ API Doc

API list

API Command	Type	Description
set_num_threads	builder	set the numbers of threads
set_design_work_space	builder	set the directory to output the timing reports
readLiberty	builder	read the liberty files
readDesign	builder	read the design verilog file
readSpef	builder	read the spef file
readSdc	builder	read the sdc file
readAocv	builder	read the aocv files
makeOrFindRCTreeNode	builder	make RC tree internal node
makeOrFindRCTreeNode	builder	make RC tree pin node
incrCap	builder	set the node's cap
makeResistor	builder	make resistor edge of RC tree
updateRCTreeInfo	builder	update the RC info after making the RC tree

buildRCTree		
initRcTree		
initRcTree		
resetRcTree		
buildGraph		
isBuildGraph		
resetGraph		
resetGraphData		
insertBuffer		
removeBuffer		
repowerInstance		
moveInstance		
writeVerilog		
setSignificantDigits	builder	set the significant digits of the timing report
incrUpdateTiming	action	incremental propagation to update the timing data
updateTiming	action	update the timing data

Doc Link：https://gitee.com/ieda-ipd/iEDA/tree/master/docs

User Manual

iEDA

文件 (4734)

- .gitee
- cmake
- docs
  - api
  - paper
  - ppt
  - resources
  - tbd
  - tcl
  - user_guide
    - pic
    - iEDA_user_guide.md
- scripts
- src
- .clang-format
- .clang-tidy
- .gitignore
- CMakeLists.txt
- LICENSE
- README-En.md
- README.md
- build.sh

script

script

```
scripts/design/sky130_gcd/script
├── DB_script                        # Data process flow scripts
│   ├── db_init_lef.tcl              # initialize lef
│   ├── db_init_lib_drv.tcl          # initialize lib only for flow of drv
│   ├── db_init_lib_fixfanout.tcl    # initialize lib only for flow of fix fanout
│   ├── db_init_lib_hold.tcl         # initialize lib only for flow of optimize hold
│   ├── db_init_lib_setup.tcl        # initialize lib only for flow of optimize setup
│   ├── db_init_lib.tcl              # initialize lib for common flow
│   ├── db_init_sdc.tcl              # initialize sdc
│   ├── db_init_spef.tcl             # initialize spef
│   ├── db_path_setting.tcl          # set paths for all processing technology files, including TechLEF, LEF, Lib, sdc and spef
│   ├── run_db_checknet.tcl          # check net connectivity based on data built by DEF (.def) and LEF (.lef & .tlef)
│   ├── run_db_report_evl.tcl        # report wire length and congestion based on data built by DEF (.def) and LEF (.lef & .tlef)
│   ├── run_db.tcl                   # test building data by DEF (.def) and LEF (.lef & .tlef)
│   ├── run_def_to_gds_text.tcl      # transform data from DEF (.def) to GDSII (.gdsii)
│   ├── run_def_to_verilog.tcl       # transform data from DEF (.def) to netlist (.v)
│   ├── run_netlist_to_def.tcl       # transform data from netlist (.v) to DEF (.def)
│   └── run_read_verilog.tcl         # test read verilog file (.v)
├── iCTS_script                      # CTS flow scripts
│   ├── run_iCTS_eval.tcl            # report wire legnth for CTS result
│   ├── run_iCTS_STA.tcl             # report CTS STA
│   └── run_iCTS.tcl                 # run CTS
├── iDRC_script                      # DRC(Design Rule Check) flow scipts
│   ├── run_iDRC_gui.tcl             # show GUI for DRC result
│   └── run_iDRC.tcl                 # run DRC
├── iFP_script                       # Floorplan flow scripts
│   ├── module                       # submodule for Floorplan scripts
│   │   ├── create_tracks.tcl        # create tracks for routing layers
│   │   ├── pdn.tcl                  # create pdn networks
│   │   └── set_clocknet.tcl         # set clock net
│   └── run_iFP.tcl                  # run Floorplan
├── iGUI_script                      # GUI flow scipts
│   └── run_iGUI.tcl                 # run GUI
├── iNO_script                       # NO(Netlist Optimization) flow scipts
│   └── run_iNO_fix_fanout.tcl       # run Fix Fanout
├── iPL_script                       # Placement flow scripts
│   ├── run_iPL_eval.tcl             # report congestion statistics and wire legnth for Placement result
│   ├── run_iPL_filler.tcl           # run standard cell filler
│   ├── run_iPL_gui.tcl              # run gui flow that shows Global Placement Processing result
```

```
scripts
├── design                   #iEDA flows for different designs
│   ├── ispd18               #tbd
│   └── sky130_gcd           #flow of gcd in sky130
│       ├── iEDA
│       ├── iEDA_config      # iEDA parameters configuration files
│       ├── README.md
│       ├── result           # iEDA result output files
│       ├── run_iEDA_gui.py  # Python3 script for running all iEDA flow with GUI layout
│       ├── run_iEDA.py      # Python3 script for running all iEDA flow
│       ├── run_iEDA.sh      # POSIX shell script for running all iEDA flow
```

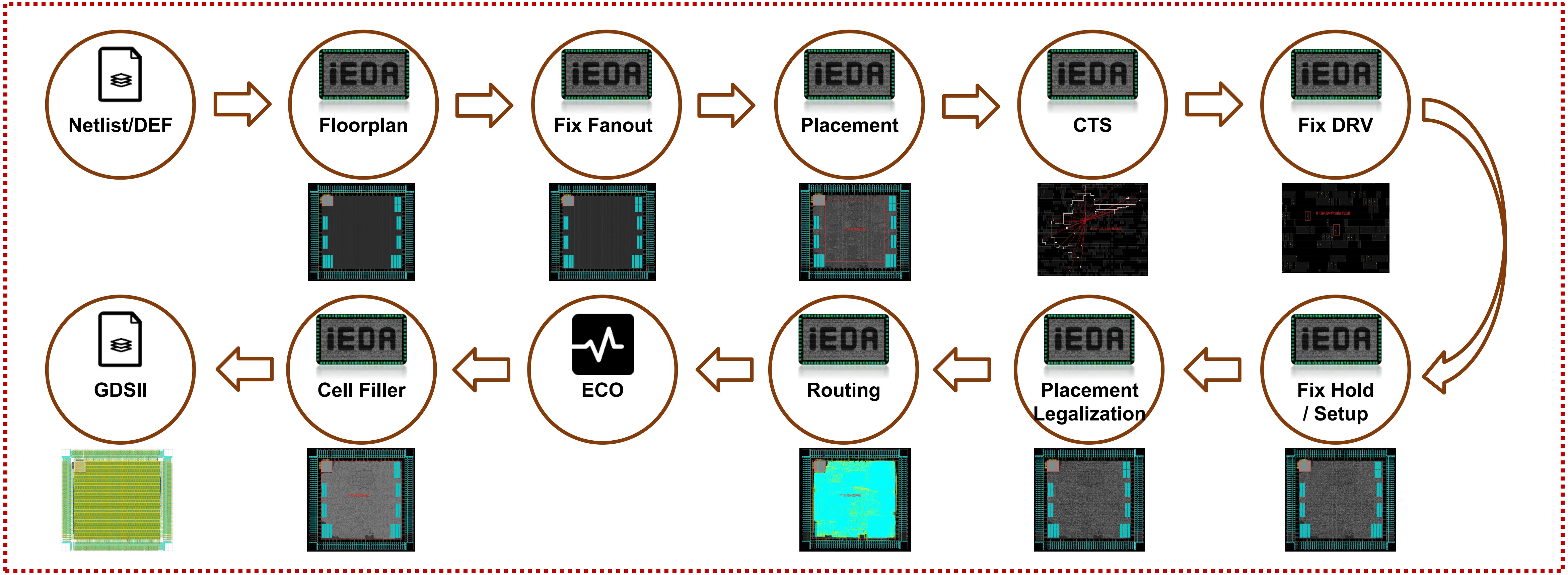
01 iEDA Infrastructure

02 Feature

03 Flow

Flow: Netlist -> GDSII

- ✓ Enabling the Physical Design Flow,
- ✓ Supporting Technologies: 110nm, 28nm, Open-source Technologies (Sky130, Nangate45)



iEDA Physical Design Flow

Environment Config

- **Userguide:** https://gitee.com/oscc-project/iEDA/blob/master/docs/user_guide/iEDA_user_guide.md

Download and Compile iEDA

```
# 下载iEDA仓库
git clone https://gitee.com/oscc-project/iEDA.git iEDA && cd iEDA
# 通过apt安装编译依赖，需要root权限
sudo bash build.sh -i apt
# 编译 iEDA
bash build.sh -j 16
# 若能够正常输出 "Hello iEDA!" 则编译成功
./bin/iEDA -script scripts/hello.tcl
```

拷贝 ./bin/iEDA 到目录 ./scripts/design/sky130_gcd

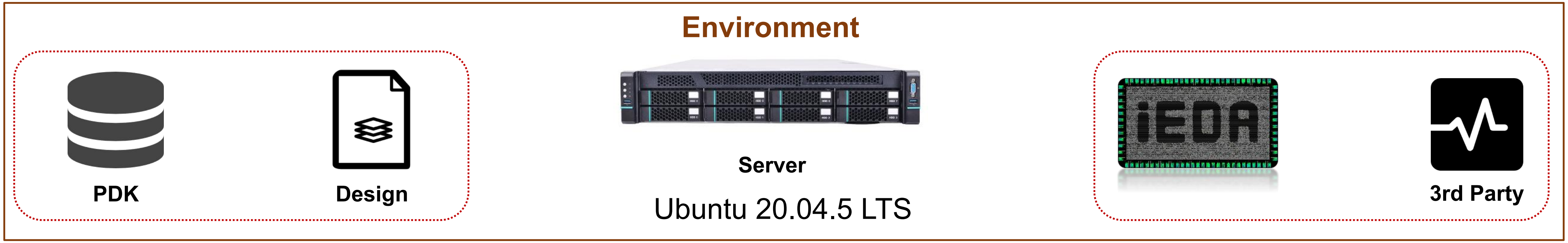
```
# 拷贝 iEDA 到sky130 目录
cp ./bin/iEDA scripts/design/sky130_gcd/.
```

Design

- ✓ Netlist

(28nm) Library

- ✓ TechLEF
- ✓ Std Cell LEF
- ✓ liberty
- ✓ sdc
- ✓ (spef)



TCL Script

```
├── design                  #iEDA flows for different designs
│   ├── ispd18              #tbd
│   └── sky130_gcd          #flow of gcd in sky130
│       ├── iEDA
│       ├── iEDA_config     # iEDA parameters configuration files
│       ├── README.md
│       ├── result          # iEDA result output files
│       ├── run_iEDA_gui.py # Python3 script for running all iEDA flow with GUI layout
│       ├── run_iEDA.py     # Python3 script for running all iEDA flow
│       ├── run_iEDA.sh     # POSIX shell script for running all iEDA flow
│       └── script          # TCL script files
├── foundry
│   ├── README.md
│   └── sky130              # SkyWater Open Source PDK
│       ├── lef             # lef files
│       ├── lib             # lib files
│       ├── sdc             # sdc files
│       └── spef            # folder for spef files if needed
└── hello.tcl               # Test running iEDA
```

```
├── DB_script                           # Data process flow scripts
│   ├── db_init_lef.tcl                 # initialize lef
│   ├── db_init_lib_drv.tcl             # initialize lib only for flow of drv
│   ├── db_init_lib_fixfanout.tcl       # initialize lib only for flow of fix fanout
│   ├── db_init_lib_hold.tcl            # initialize lib only for flow of optimize hold
│   ├── db_init_lib_setup.tcl           # initialize lib only for flow of optimize setup
│   ├── db_init_lib.tcl                 # initialize lib for common flow
│   ├── db_init_sdc.tcl                 # initialize sdc
│   ├── db_init_spef.tcl                # initialize spef
│   ├── db_path_setting.tcl             # set paths for all processing technology files, including TechLEF, LEF, Lib, sdc and sp
│   ├── run_db_checknet.tcl             # check net connectivity based on data built by DEF (.def) and LEF (.lef & .tlef)
│   ├── run_db_report_evl.tcl           # report wire length and congestion based on data built by DEF (.def) and LEF (.lef &
│   ├── run_db.tcl                      # test building data by DEF (.def) and LEF (.lef & .tlef)
│   ├── run_def_to_gds_text.tcl         # transform data from DEF (.def) to GDSII (.gdsii)
│   ├── run_def_to_verilog.tcl          # transform data from DEF (.def) to netlist (.v)
│   ├── run_netlist_to_def.tcl          # transform data from netlist (.v) to DEF (.def)
│   └── run_read_verilog.tcl            # test read verilog file (.v)
├── iCTS_script                         # CTS flow scripts
│   ├── run_iCTS_eval.tcl               # report wire legnth for CTS result
│   ├── run_iCTS_STA.tcl                # report CTS STA
│   └── run_iCTS.tcl                    # run CTS
├── iDRC_script                         # DRC(Design Rule Check) flow scipts
│   ├── run_iDRC_gui.tcl                # show GUI for DRC result
│   └── run_iDRC.tcl                    # run DRC
├── iFP_script                          # Floorplan flow scripts
│   ├── module                          # submodule for Floorplan scripts
│   │   ├── create_tracks.tcl           # create tracks for routing layers
│   │   ├── pdn.tcl                     # create pdn networks
│   │   └── set_clocknet.tcl            # set clock net
│   └── run_iFP.tcl                     # run Floorplan
├── iGUI_script                         # GUI flow scipts
│   └── run_iGUI.tcl                    # run GUI
├── iNO_script                          # NO(Netlist Optimization) flow scipts
│   └── run_iNO_fix_fanout.tcl          # run Fix Fanout
├── iPL_script                          # Placement flow scripts
│   ├── run_iPL_eval.tcl                # report congestion statistics and wire legnth for Placement result
│   ├── run_iPL_filler.tcl              # run standard cell filler
```

TCL Script

Flow

```
#=======================================================================
##   run floorplan
#=======================================================================
os.system('./iEDA -script ./script/iFP_script/run_iFP.tcl')
#=======================================================================
##   run NO -- fix fanout
#=======================================================================
os.system('./iEDA -script ./script/iNO_script/run_iNO_fix_fanout.tcl')
#=======================================================================
##   run Placer
#=======================================================================
os.system('./iEDA -script ./script/iPL_script/run_iPL.tcl')
os.system('./iEDA -script ./script/iPL_script/run_iPL_eval.tcl')
# =======================================================================
#   run CTS
# =======================================================================
os.system('./iEDA -script ./script/iCTS_script/run_iCTS.tcl')
os.system('./iEDA -script ./script/iCTS_script/run_iCTS_eval.tcl')
os.system('./iEDA -script ./script/iCTS_script/run_iCTS_STA.tcl')
#=======================================================================
##   run TO -- fix_drv
#=======================================================================
os.system('./iEDA -script ./script/iTO_script/run_iTO_drv.tcl')
os.system('./iEDA -script ./script/iTO_script/run_iTO_drv_STA.tcl')
#=======================================================================
#   run TO -- opt_hold
#=======================================================================
os.system('./iEDA -script ./script/iTO_script/run_iTO_setup.tcl')
os.system('./iEDA -script ./script/iTO_script/run_iTO_hold.tcl')
os.system('./iEDA -script ./script/iTO_script/run_iTO_hold_STA.tcl')
#=======================================================================
#   run PL Incremental Flow
#=======================================================================
os.system('./iEDA -script ./script/iPL_script/run_iPL_legalization.tcl')
os.system('./iEDA -script ./script/iPL_script/run_iPL_legalization_eval.tcl')
# =======================================================================
# #   run Router
# =======================================================================
os.system('./iEDA -script ./script/iRT_script/run_iRT.tcl')
os.system('./iEDA -script ./script/iRT_script/run_iRT_eval.tcl')
os.system('./iEDA -script ./script/iRT_script/run_iRT_STA.tcl')
os.system('./iEDA -script ./script/iRT_script/run_iRT_DRC.tcl')
#=======================================================================
##   run Filler
#=======================================================================
os.system('./iEDA -script ./script/iPL_script/run_iPL_filler.tcl')
#=======================================================================
##   run def to gdsii
#=======================================================================
os.system('./iEDA -script ./script/DB_script/run_def_to_gds_text.tcl')
```

Placement

```
#=======================================================================
##   init flow config
#=======================================================================
flow_init -config ./iEDA_config/flow_config.json
#=======================================================================
##   read db config
#=======================================================================
db_init -config ./iEDA_config/db_default_config.json
#=======================================================================
##   reset data path
#=======================================================================
source ./script/DB_script/db_path_setting.tcl
#=======================================================================
##   reset sdc
#=======================================================================
source ./script/DB_script/db_init_sdc.tcl
#=======================================================================
##   read lef
#=======================================================================
source ./script/DB_script/db_init_lef.tcl
#=======================================================================
##   read def
#=======================================================================
def_init -path ./result/iTO_fix_fanout_result.def
#=======================================================================
##   run Placer
#=======================================================================
run_placer -config ./iEDA_config/pl_default_config.json
#=======================================================================
##   Save def
#=======================================================================
def_save -path ./result/iPL_result.def
#=======================================================================
##   Save netlist
#=======================================================================
netlist_save -path ./result/iPL_result.v -exclude_cell_names {}
#=======================================================================
##   report
#=======================================================================
report_db -path "./result/report/pl_db.rpt"
#=======================================================================
##   Exit
#=======================================================================
flow_exit
```

Parameter Config

- Config

- iEDA_config
  - cts_default_config.json
  - db_default_config.json
  - drc_default_config.json
  - flow_config.json
  - fp_default_config.json
  - no_default_config_fixfanout...
  - pl_default_config.json
  - rt_default_config.json
  - to_default_config_drv.json
  - to_default_config_hold.json
  - to_default_config_setup.json

iEDA / scripts / design / sky130_gcd / iEDA_config / pl_default_config.json

Code Blame 82 lines (82 loc) · 2.38 KB

```
        "is_max_length_opt": 0,
        "max_length_constraint": 1000000,
        "is_timing_effort": 0,
        "is_congestion_effort": 0,
        "ignore_net_degree": 100,
        "num_threads": 1,
        "GP": {
            "Wirelength": {
                "init_wirelength_coef": 0.25,
                "reference_hpwl": 446000000,
                "min_wirelength_force_bar": -300
            },
            "Density": {
                "target_density": 0.8,
                "bin_cnt_x": 128,
                "bin_cnt_y": 128
            },
            "Nesterov": {
                "max_iter": 2000,
                "max_backtrack": 10,
                "init_density_penalty": 0.00008,
                "target_overflow": 0.1,
                "initial_prev_coordi_update_coef": 100,
                "min_precondition": 1.0,
                "min_phi_coef": 0.95,
                "max_phi_coef": 1.05
            }
        },
        "BUFFER": {
            "max_buffer_num": 10000,
            "buffer_type": [
                "sky130_fd_sc_hs__buf_1"
            ]
        },
        "LG": {
            "max_displacement": 1000000,
            "global_right_padding": 0
```

iEDA / scripts / design / sky130_gcd / iEDA_config / pl_default_config.json

Code Blame 82 lines (82 loc) · 2.38 KB

```
            "max_displacement": 1000000,
            "global_right_padding": 0,
            "enable_networkflow" : 0
        },
        "Filler": {
            "first_iter": [
                "sky130_fd_sc_hs__fill_8",
                "sky130_fd_sc_hs__fill_4",
                "sky130_fd_sc_hs__fill_2",
                "sky130_fd_sc_hs__fill_1"
            ],
            "second_iter": [
                "sky130_fd_sc_hs__fill_8",
                "sky130_fd_sc_hs__fill_4",
                "sky130_fd_sc_hs__fill_2",
                "sky130_fd_sc_hs__fill_1"
            ],
            "min_filler_width": 1
        },
        "MP": {
            "fixed_macro": [],
            "fixed_macro_coordinate": [],
            "blockage": [],
            "guidance_macro": [],
            "guidance": [],
            "solution_type": "BStarTree",
            "SimulateAnneal": {
                "perturb_per_step": 100,
                "cool_rate": 0.92
            },
            "Partition": {
                "parts": 66,
                "ufactor": 100,
                "new_macro_density": 0.6
            },
            "halo_x": 0,
            "halo_y": 0,
            "output_path": "${RESULT_DIR}/pl/"
```

How to Run Netlist -> GDSII Flow by iEDA

./run_iEDA.py

.V → .gdsii

Flow	Script	Config	Design Input
布图规划 (Floorpan)	./iEDA -script ./script/iFP_script/run_iFP.tcl		./result/verilog/gcd.v
网表优化（Fix Fanout）	./iEDA -script ./script/iNO_script/run_iNO_fix_fanout.tcl	./iEDA_config/cts_default_config.json	./result/iFP_result.def
布局 (Placement)	./iEDA -script ./script/iPL_script/run_iPL.tcl	./iEDA_config/pl_default_config.json	./result/iTO_fix_fanout_resu
布局结果评估 (评估线长和拥塞)	./iEDA -script ./script/iPL_script/run_iPL_eval.tcl		./result/iPL_result.def
时钟树综合 (CTS)	./iEDA -script ./script/iCTS_script/run_iCTS.tcl	./iEDA_config/cts_default_config.json	./result/iPL_result.def
时钟树综合结果评估 (评估线长)	./iEDA -script ./script/iCTS_script/run_iCTS_eval.tcl		./result/iCTS_result.def
时钟树综合时序评估 (评估时序)	./iEDA -script ./script/iCTS_script/run_iCTS_STA.tcl		./result/iCTS_result.def
修复DRV违例 (Fix DRV Violation)	./iEDA -script ./script/iTO_script/run_iTO_drv.tcl	./iEDA_config/to_default_config_drv.json	./result/iCTS_result.def
Fix DRV结果评估 (评估时序)	./iEDA -script ./script/iTO_script/run_iTO_drv_STA.tcl		./result/iTO_drv_result.def
修复Hold违例（Fix Hold Violation)	./iEDA -script ./script/iTO_script/run_iTO_hold.tcl	./iEDA_config/to_default_config_hold.json	./result/iTO_drv_result.def
Fix Hold结果评估（评估时序）	./iEDA -script ./script/iTO_script/run_iTO_hold_STA.tcl		./result/iTO_hold_result.def
单元合法化（Legalization）	./iEDA -script ./script/iPL_script/run_iPL_legalization.tcl	./iEDA_config/pl_default_config.json	./result/iTO_hold_result.def
合法化结果评估（评估线长和拥塞）	./iEDA -script ./script/iPL_script/run_iPL_legalization_eval.tcl		./result/iPL_lg_result.def
布线（Routing）	./iEDA -script ./script/iRT_script/run_iRT.tcl		./result/iPL_lg_result.def

Report

✓ Statistics

- iFP
- iNO
- iPL
- iCTS
- iTO
- iRT

##

Summary

Module	Value
DIE Area (um^2)	2249940.000000 = 1499.960000 * 1500.000000
DIE Usage	0.237865
CORE Area (um^2)	1340542.224000 = 1160.040000 * 1155.600000
CORE Usage	0.399228
Number - Site	9
Number - Row	1284
Number - Track	20
Number - Layer	32
Number - Routing Layer	10
Number - Cut Layer	10
Number - GCell Grid	5
Number - Cell Master	16314
Number - Via Rule	492
Number - IO Pin	110
Number - Instance	362883
Number - Blockage	21
Number - Filler	0
Number - Net	377248
Number - Special Net	5

✓ Design rule violations

- iRT

Drc Summary

DRC Type	Number
Cut Different Layer Spacing	0
Cut EOL Spacing	0
Cut Enclosure	610489
Cut EnclosureEdge	0
Cut Spacing	264504
Metal Corner Filling Spacing	0
Metal EOL Spacing	8404599
Metal JogToJog Spacing	0
Metal Notch Spacing	1621088
Metal Parallel Run Length Spacing	3253512
Metal Short	6429817
MinHole	94
MinStep	7794204
Minimal Area	744067

✓ Evaluation

- Placement
- CTS
- Physical Incremental
- Routing

Congestion Report

Grid Bin Size	Bin Partition	Total Count
9063 * 9029	256 by 256	65536

Instance Density Range	Bins Count	Percentage
0.95 ~ 1.00	5372	8.20
0.90 ~ 0.95	34	0.05
0.85 ~ 0.90	29	0.04
0.80 ~ 0.85	329	0.50
0.75 ~ 0.80	0	0.00

Pin Count Range	Bins Count	Percentage
205 ~ 228	22	0.03
182 ~ 205	129	0.20
160 ~ 182	514	0.78
137 ~ 160	1555	2.37
114 ~ 137	252	0.38

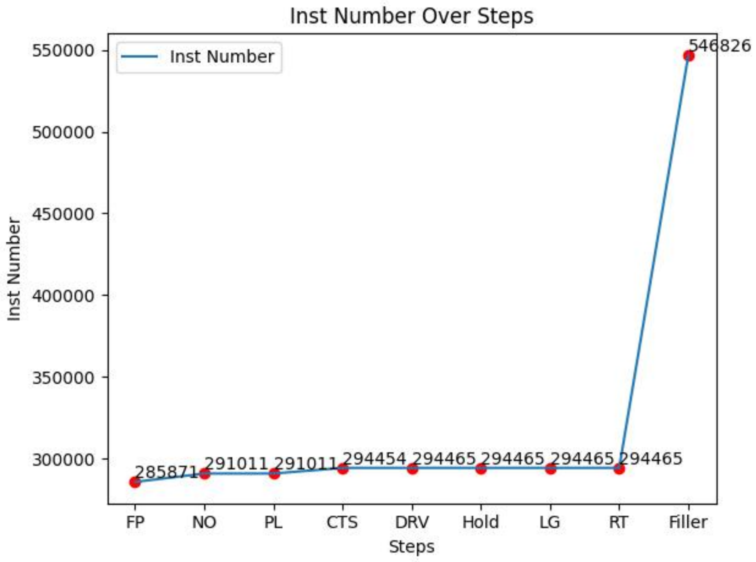
✓ Timing & Power

- Setup, Hold, Violations, Power
- Placement, CTS，Fix DRV，Fix Setup/Hold，Routing

Point	Fanout	Capacitance	Resistance	Transition	Delta Delay	Derate	Incr	Path
spi_miso_pad (port)		8.0000000	0.0010000	0.0000000		1.0000000	0.0000000	0.0000000r
clock CLK_spi_clk_out (rise edge)							0	0
clock network delay (propagated)							0.0000000	0.0000000
spi_miso_pad (port)		8.0000000	0.0010000	0.0000000		1.0000000	29.1000000	29.1000000r
spi_miso_pad (net)	2				0.0000000			
u0_spi_miso:PAD (PDDW04DGZ_H_G)		1.0906020	0.0000000	0.0000000		1.0000000	0.0000000	29.1000000r
u0_spi_miso:C (PDDW04DGZ_H_G)		0.0250522	0.0000000	0.0172830		1.0765001	0.8057400	29.9057400r
spi_flash_miso (net)	1				0.0000000			
ediFE_PDC3389_spi_flash_miso:I (BUFFD16BWP35P140LVT)		0.0028179	2612.1424828	0.0172830		1.0000000	0.0000000	29.9057400r
ediFE_PDC3389_spi_flash_miso:Z (BUFFD16BWP35P140LVT)		0.0197853	0.0000000	0.0106300		1.0765001	0.0217290	29.9274690r
FE_PDN3389_spi_flash_miso (net)	1				0.0000000			
ediFE_PDC12025_spi_flash_miso:I (BUFFD4BWP30P140LVT)		0.0011448	1760.8977135	0.0106300		1.0000000	0.0000000	29.9274690r
ediFE_PDC12025_spi_flash_miso:Z (BUFFD4BWP30P140LVT)		0.0236758	0.0000000	0.0271790		1.0765001	0.0263400	29.9538090r
FE_PDN12025_spi_flash_miso (net)	1				0.0000000			
u0_soc_top/u0_spi_flash/u0_spi_top/shift/U358:A1 (INR2D1BWP40P140LVT)		0.0005121	1727.8863792	0.0194320		1.0000000	0.0129420	29.9667510r
u0_soc_top/u0_spi_flash/u0_spi_top/shift/U358:ZN (INR2D1BWP40P140LVT)		0.0012539	0.0000000	0.0186460		1.0765001	0.0294440	29.9961950r
u0_soc_top/u0_spi_flash/u0_spi_top/shift/n787 (clock net)	1				0.0000000			
ediFE_PDC6804_u0_soc_top_u0_spi_flash_u0_spi_top_shift_n787:I (BUFFD3BWP30P140LVT)		0.0005980	92.1830030	0.0057640		1.0000000	0.0000140	29.9962090r
ediFE_PDC6804_u0_soc_top_u0_spi_flash_u0_spi_top_shift_n787:Z (BUFFD3BWP30P140LVT)		0.0329341	0.0000000	0.0510340		1.0765001	0.0428270	30.0390360r
FE_PDN6804_u0_soc_top_u0_spi_flash_u0_spi_top_shift_n787 (clock net)	29				0.0000000			
ediFE_PDC6803_u0_soc_top_u0_spi_flash_u0_spi_top_shift_n787:I (BUFFD3BWP30P140LVT)		0.0005980	701.9384875	0.0107370		1.0000000	0.0085510	30.0475870r
ediFE_PDC6803_u0_soc_top_u0_spi_flash_u0_spi_top_shift_n787:Z (BUFFD3BWP30P140LVT)		0.0327282	0.0000000	0.0507560		1.0765001	0.0440820	30.0916690r
FE_PDN6803_u0_soc_top_u0_spi_flash_u0_spi_top_shift_n787 (clock net)	24				0.0000000			
ediFE_PDC6802_u0_soc_top_u0_spi_flash_u0_spi_top_shift_n787:I (BUFFD2BWP35P140LVT)		0.0006282	900.9526939	0.0136380		1.0000000	0.0107520	30.1024210r
ediFE_PDC6802_u0_soc_top_u0_spi_flash_u0_spi_top_shift_n787:Z (BUFFD2BWP35P140LVT)		0.0117069	0.0000000	0.0301890		1.0765001	0.0326970	30.1351180r
FE_PDN6802_u0_soc_top_u0_spi_flash_u0_spi_top_shift_n787 (clock net)	10				0.0000000			
ediFE_PDC6801_u0_soc_top_u0_spi_flash_u0_spi_top_shift_n787:I (BUFFD3BWP35P140LVT)		0.0006183	538.5603962	0.0071530		1.0000000	0.0030720	30.1381900r
ediFE_PDC6801_u0_soc_top_u0_spi_flash_u0_spi_top_shift_n787:Z (BUFFD3BWP35P140LVT)		0.0246096	0.0000000	0.0415460		1.0765001	0.0401200	30.1783100r
FE_PDN6801_u0_soc_top_u0_spi_flash_u0_spi_top_shift_n787 (clock net)	21				0.0000000			

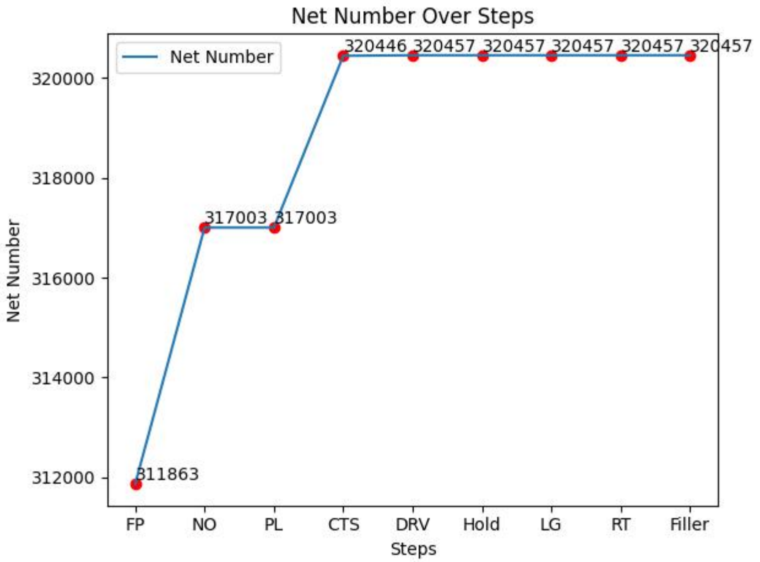
Data Analysis

Instance number



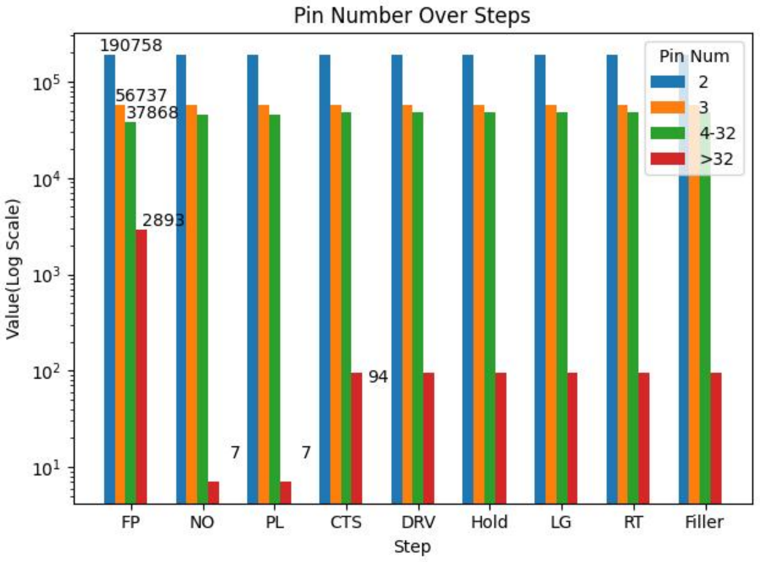
- ✓ Floorplanning -> Routing, where #Inst increased by **8594**, generated by the Fix Fanout, CTS, DRV, and Fix Setup/Hold.
- ✓ In the Filler stage, **252361** Instances is added.

Net number



- ✓ The total increase #nets in backend physical design flow is **8324**, primarily contributed by netlist optimization and clock tree synthesis stages, which are 5140 and 3284, respectively.

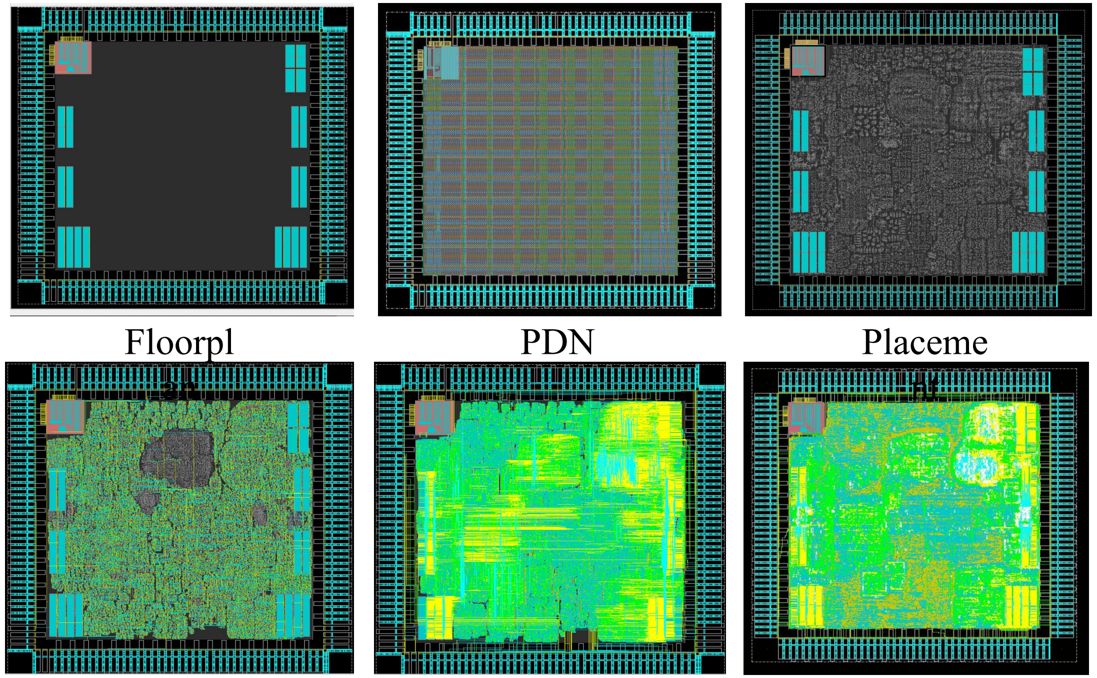
Pin number

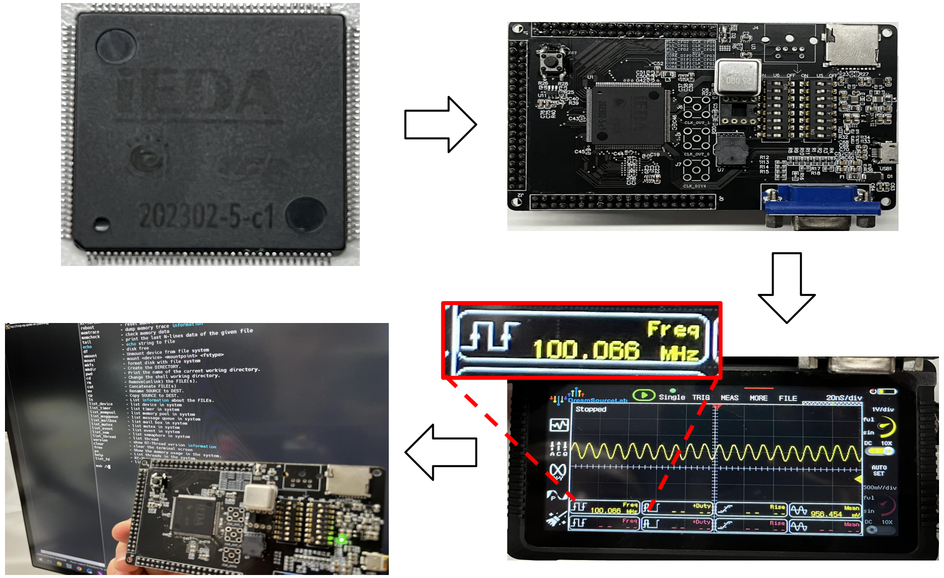


- ✓ Most of the nets consist of **2** pins and **3** pins.
- ✓ The number of nets with excessive fanout (Pins in Net) was optimized in the NO stage, reducing from **2893 to 7**.
- ✓ In the CTS stage, **87** new clock nets with excessive fanout were generated (Pin Number > 32).

Example Design: ysyx-04-01

- RTL：ysyx(一生一芯)-04
- PDK：28nm
- Area：1.5mm × 1.5 mm
- Power：dynamic = 317mW，leakage = 29 mW
- Freq.：200MHz
- Scale：>1.5M Gates
- Features：11 pipelines with cache，IP： UART、VGA、PS/2、SPI、SDRAM、 2 PLLs，support Linux



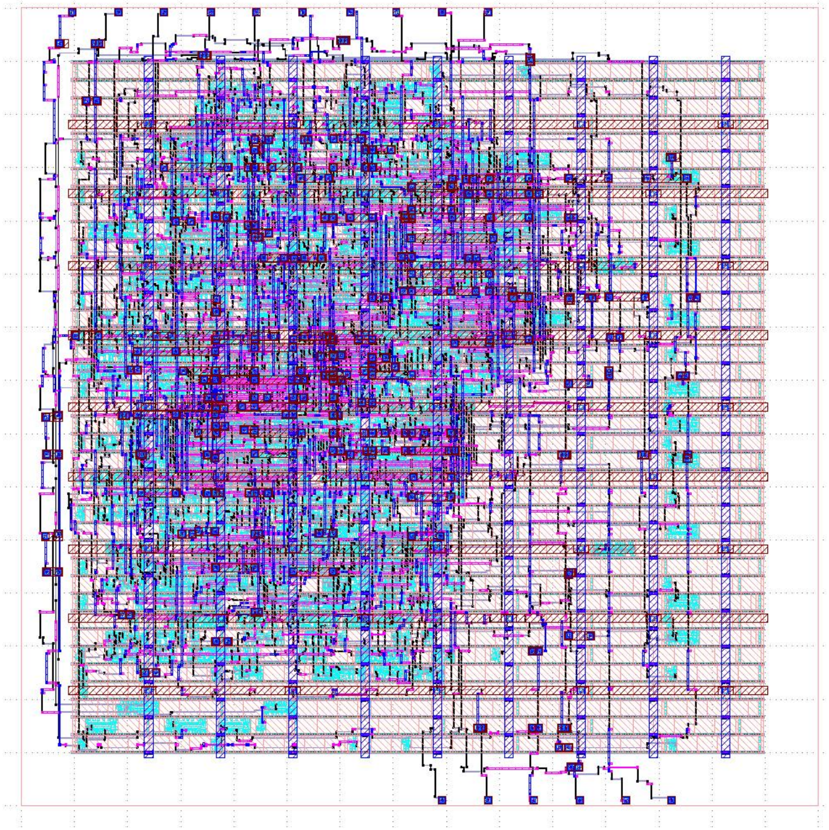


part metrics	iPL (place)	iCTS	iTO	iRT (route)
#inst	1043440	1057291	1057549	1057549
#net	1015532	1029383	1029641	1029641
utilization	0.563929	0.570644	0.570768	0.570768
HPWL	34108823398	35042653984	35044866877	50157263995*
STWL	46195026227	46580611921	46581568292	
frequency	245.245	238.226	241.386	224.254
#DRC	0	0	0	233335

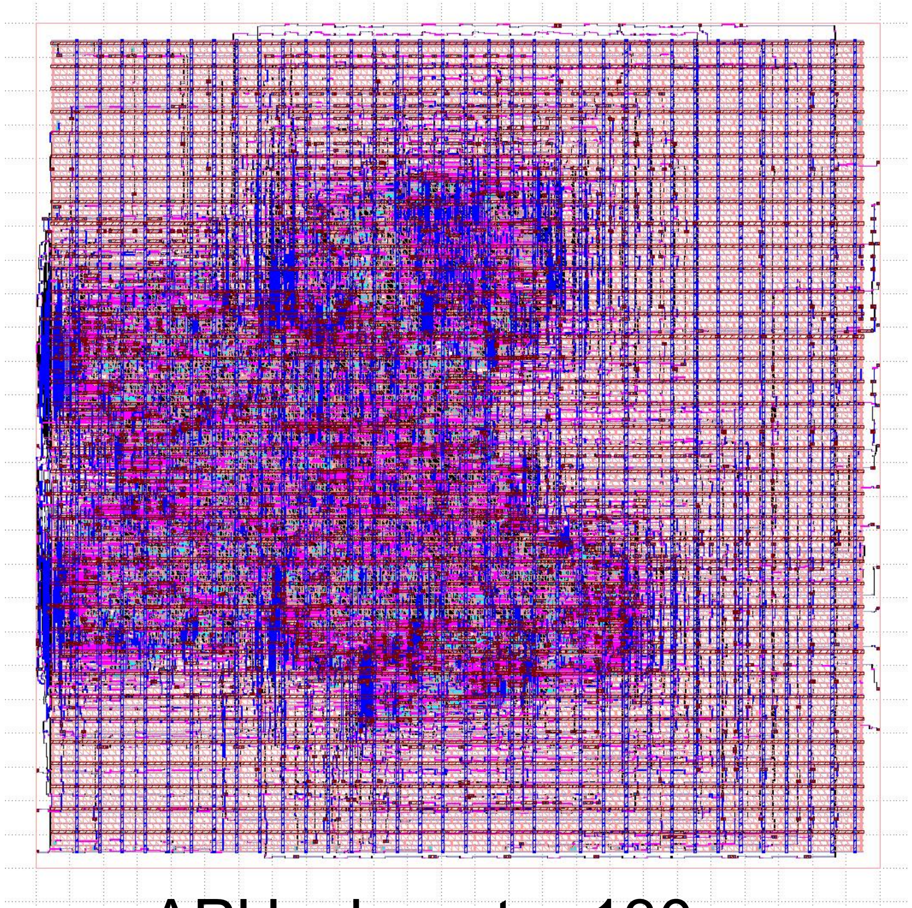
* Total wirelength after routing

Example Design: from other users

- gcd & APU



gcd, skywater 130nm
Area：0.15mm × 0.15 mm



APU, skywater 130nm
Area：0.45mm × 0.45 mm

iEDA & OpenROAD

Flow				placement		routing			sta		
metrics	design	PDK	时钟_MHz	GP_original HPWL（um)	DP HPWL(PL_eval)	droute run time（s)	total wire length	total vias	setup_slack（max)	hold_slack（min)	suggest freq(MHz)
openroad	APU	sky130	50	108025.8	304807.6	900	348193	42309	14.28	0.44	174.8251748
	BM64	sky130	50	950721.2	1151209.6	660	1298431	118906	14.22	0.42	173.0103806
	PPU	sky130	50	799473.6	1325241.3	14220	1666739	173502	15.46	0.38	220.2643172
	aes	sky130	50	1736825.9	2234547.8	1920	2695657	280870	14.7	0.22	188.6792453
	aes_core	sky130	50	1915862.8	2353877.1	2040	2809505	271884	14.73	0.4	189.7533207
	blabla	sky130	50	2162081.7	2401241.7	1320	2651252	226526	9.84	0.38	98.42519685
	caravel_upw	sky130	50	35240.4	60479.9	180	66621	7595	17.82	0.4	458.7155963
	gcd	sky130	50	10958.5	23153.9	60	25345	3798	16.73	0.44	305.8103976
	picorv32a	sky130	50	955226.7	1458821.9	2040	1659581	177160	8.78	0.38	89.12655971
	s44	sky130	50	3153.7	6408.8	1020	7149	1220	19.25	0.42	1333.333333
	salsa20	sky130	50	2014421.6	2231403.5	1140	255535	262945	9.93	0.41	99.30486594
iEDA	APU	sky130	50	101311.635	108052.855	59.15	153766.992	39682	15.81	0.364	238.638
	BM64	sky130	50	724102.907	734325.379	167.71	814055.014	183714	15.73	0.386	234.199
	PPU	sky130	50	798854.543	814966.702	236.65	1133497.302	141243	16.185	0.354	262.124
	aes	sky130	50	1787923.281	1804659.776	456.84	2447231.105	284325	15.856	0.261	241.341
	aes_core	sky130	50	1869162.753	1880282.585	417.38	2360070.252	261450	15.934	0.366	245.913
	blabla	sky130	50	1915513.912	1935383.056	305.18	2004497.857	192730	12.203	0.341	128.25
	caravel_upw	sky130	50	34144.56	35161.52	1.17	50311073	15885	18.313	0.353	592.825
	gcd	sky130	50	11282.203	11774.379	0.78	16126254	9534	17.735	0.403	441.47
	picorv32a	sky130	50	928885.74	951888.682	236.62	1093014.143	163736	13.825	0.341	161.933
	s44	sky130	50	3008.324	3292.983	0.27	4641801	2764	19.421	0.388	1727.05
	salsa20	sky130	50	1895922.266	1938562.099	429.95	2331960.542	267963	12.629	0.384	135.676
Ratio	APU	sky130	50	1.066272398	2.820912043	15.21	2.264419662	1.0662013	0.903225806	1.208791209	0.732595709
	BM64	sky130	50	1.312964208	1.56771049	3.94	1.59501628	0.647234288	0.904005086	1.088082902	0.738732363
	PPU	sky130	50	1.000774931	1.626129383	60.09	1.470439318	1.22839362	0.955205437	1.073446328	0.840305799
	aes	sky130	50	0.97142082	1.23821001	4.20	1.101513051	0.987848413	0.927093845	0.842911877	0.781795241
	aes_core	sky130	50	1.024984473	1.251874117	4.89	1.190432784	1.039908204	0.924438308	1.092896175	0.771627855
	blabla	sky130	50	1.128721481	1.240706171	4.33	1.322651451	1.175354122	0.806359092	1.114369501	0.767447929
	caravel_upw	sky130	50	1.032094132	1.720059315	154.50	0.001324182	0.478124016	0.973079233	1.133144476	0.773779102
	gcd	sky130	50	0.971308529	1.966464643	77.09	0.001571661	0.398363751	0.943332394	1.091811414	0.692709352
	picorv32a	sky130	50	1.028357589	1.532555148	8.62	1.518352723	1.081985635	0.635081374	1.114369501	0.55039158
	s44	sky130	50	1.048324582	1.946198933	3,784.06	0.001540135	0.441389291	0.991195098	1.082474227	0.772029376
	salsa20	sky130	50	1.062502211	1.15106114	2.65	0.10957947	0.981273534	0.786285533	1.067708333	0.731926545
Average				1.058884123	1.641989218	374.51	0.961530974	0.866006925	0.88630011	1.082727813	0.741212805

Functional Shelf

- **Level 1:** Open-source tools, RTLs, PDKs support chip design
- **Level 2:** Open-source Infrastructure supports EDA **Tool** and **Algorithm** development

iEDA Solvers

QP/LP/NLP | SP/MCF | partition | Fit | steiner | A* | SA/GA | FEM/FFT

iEDA Tools

iFP Floorplan	**iPDN** Power Deliver Net	**iNO** Netlist Opt	**iCTS** Clock Tree Syn	**iPL** Placement	**iTO** Timing Opt
iRT Routing	**iDRC** Design Rule Check	**iSTA** Timing Analysis	**iPA** Power Analysis	**iRCX** RC Extraction	**iMP** Macro Placement

iEDA Infrastructure

Database | DataManager | Report | Scripts | File Manager

GUI | TCL | Python | Evaluation | Feature Manager

R & D EDA Tools or Algorithms

- **Min Wirelength Model**

$$\min_{\boldsymbol{v}} \quad W(\boldsymbol{v})$$
$$s.t. \quad \rho_b(\boldsymbol{v}) \le \rho_0, \ \ \forall b \in B$$

- where v is cell location, $W(\boldsymbol{v})$ is wirelength, $\rho_b(\boldsymbol{v})$ is the area density in $b \in B$, ρ_0 is density thredhold.

- **Nesterov Method or Conjugate Gradient**

1. Given x_0, $r_0 = Ax_0 - b$, $p_0 = -r_0$
2. For $k = 0,1,2,\dots$ until $||r_k|| = 0$

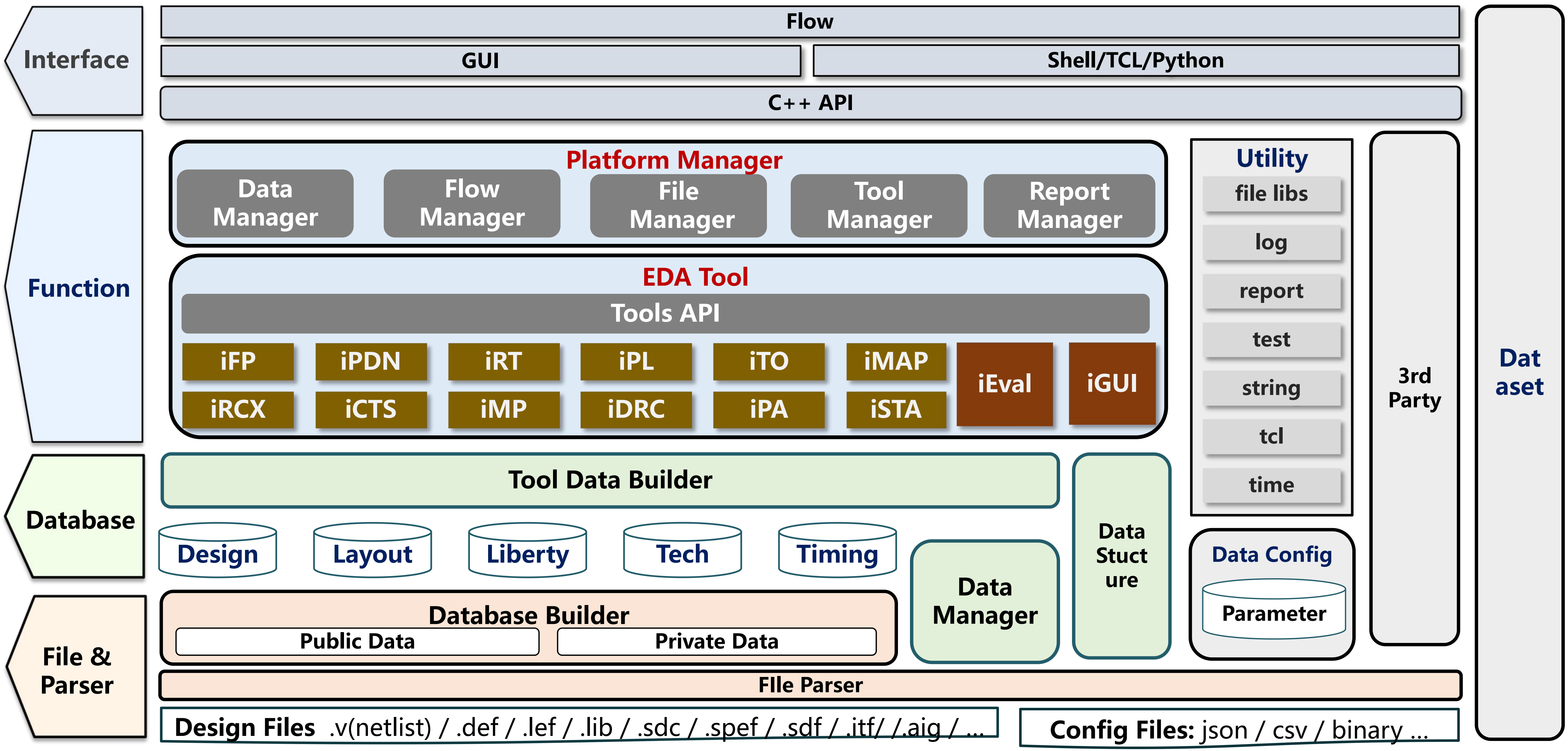
$$\begin{aligned}
\alpha_k &= r_k^T r_k / p_k^T A p_k \\
x_{k+1} &= x_k + \alpha_k p_k \\
r_{k+1} &= r_k + \alpha_k A p_k \\
\beta_{k+1} &= r_{k+1}^T r_{k+1} / r_k^T r_k \\
p_{k+1} &= -r_{k+1} + \beta_{k+1} p_k
\end{aligned}$$

- Assignment: please implement CG method by C++ or Python, and test it on "iEDA/iPL", submit by PR to iEDA repo.

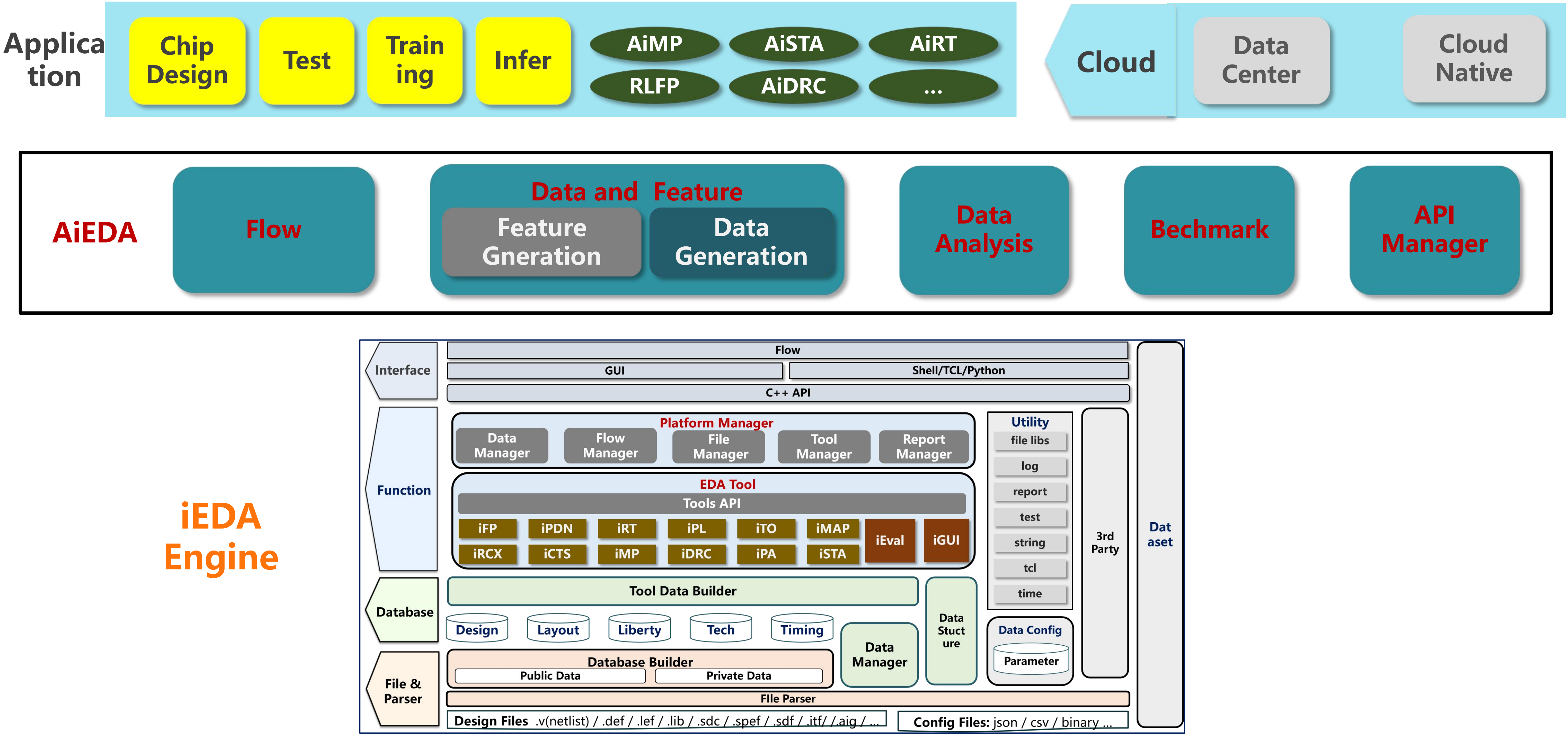
```
Go to file
module
solver
  conjugate
    CMakeLists.txt
  readme.md
  nesterov
    CMakeLists.txt
    Nesterov.cc
    Nesterov.hh
  partition
  simulate_anneal
  subgradient
  CMakeLists.txt
utility
CMakeLists.txt
PlacerDB.cc
PlacerDB.hh
test
CMakeLists.txt
README.md
```

```cpp
class Nesterov
{
 public:
  Nesterov();
  Nesterov(const Nesterov& other) = delete;
  Nesterov(Nesterov&& other) = delete;
  ~Nesterov() = default;

  // getter.
  int get_current_iter() const { return _current_iter; }
  const std::vector<Point<int32_t>>& get_current_coordis() const { return _current_coordis; }
  const std::vector<Point<float>>& get_current_grads() const { return _current_gradients; }
  const std::vector<Point<float>>& get_next_grads() const { return _next_gradients; }
  const std::vector<Point<int32_t>>& get_next_coordis() const { return _next_coordis; }
  const std::vector<Point<int32_t>>& get_next_slp_coordis() const { return _next_slp_coordis; }
  float get_next_steplength() const { return _next_steplength; }

  // for RDP
  const std::vector<Point<float>>& get_next_gradients() const { return _next_gradients; }
  float get_next_parameter() const { return _next_parameter; }
  void set_next_coordis(const std::vector<Point<int32_t>>& next_coordis) { _next_coordis = next_co
  void set_next_slp_coordis(const std::vector<Point<int32_t>>& next_slp_coordis) { _next_slp_coord
  void set_next_gradients(const std::vector<Point<float>>& next_gradients) { _next_gradients = nex
  void set_next_parameter(float next_parameter) { _next_parameter = next_parameter; }
  void set_next_steplength(float next_steplength) { _next_steplength = next_steplength; }

  // function.
  void initNesterov(std::vector<Point<int32_t>> previous_coordis, std::vector<Point<float>> previo
                    std::vector<Point<int32_t>> current_coordis, std::vector<Point<float>> current_
  void calculateNextSteplength(std::vector<Point<float>> next_grads);

  void runNextIter(int next_iter, int32_t thread_num);
  void runBackTrackIter(int32_t thread_num);

```

iEDA Infra.: Evolution – System



iEDA Infra.: Evolution – Engine

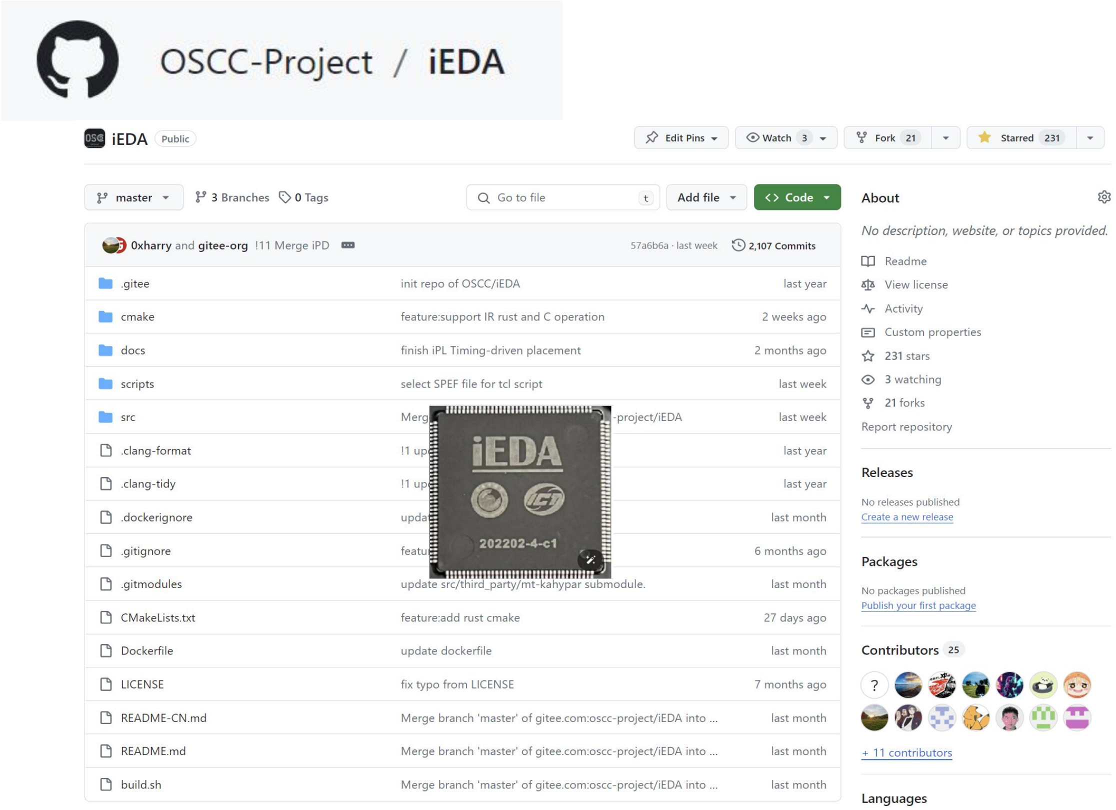


Conclusions and Future works

- **Conclusions**
  - The iEDA infrastructure
  - Its features
  - How to use to design chip from netlist to GDS
- **Future works**
  - More Solvers
  - High-performance
  - More APIs

iEDA Tutorial Agenda

- **Part 0:** iEDA Overview **(Xingquan Li)**
- **Part 1:** iEDA Infrastructure **(Zengrong Huang)**
- **Part 2: iPL: Placement Tool and Its Technology (Shijian Chen)**
- **Part 3:** iCTS: Clock Tree Synthesis Tool and Its Technologies **(Weiguo Li)**
- **Part 4:** iRT: Routing Tool and Its Technologies **(Zhisheng Zeng)**
- **Part 5:** iSTA: Static Timing Analysis Tool and Its Technologies **(Simin Tao/He Liu)**
- **Part 6:** iPA: Power Analysis Tool and Its Technologies **(Siming Tao)**



Thanks

Xingquan Li
lixq01@pcl.ac.cn