ASP-DAC 2024
29th Asia and South Pacific Design Automation Conference

iEDA: An Open-source infrastructure of EDA

Xingquan Li, Zengrong Huang, Simin Tao, Zhipeng Huang, Chunan Zhuang, Hao Wang, Yifan Li, Yihang Qiu, Guojie Luo, Huawei Li, Haihua Shen, Mingyu Chen, Dongbo Bu, Wenxing Zhu, Ye Cai, Xiaoming Xiong, Ying Jian, Yi Heng, Peng Zhang, **Bei Yu**, Biwei Xie, Yungang Bao

Jan. 23 2024

01 Introduction

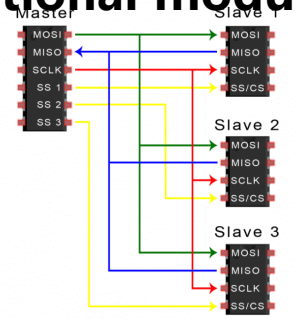
02 iEDA

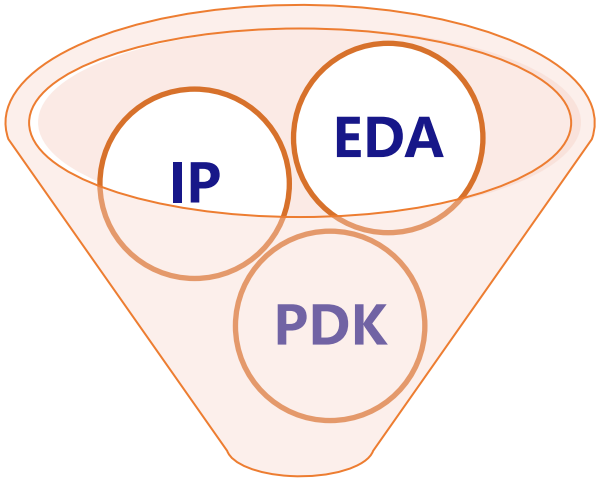
03 iEDA Application

Chip Design Elements

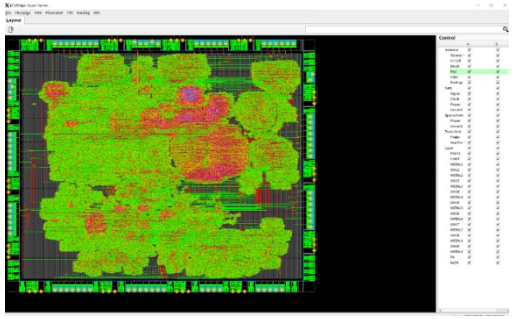
IP: Intellectual Property

Functional module

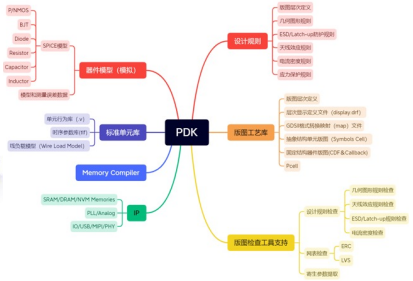




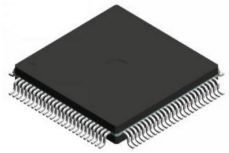
EDA: Electronic design automation software (tool)



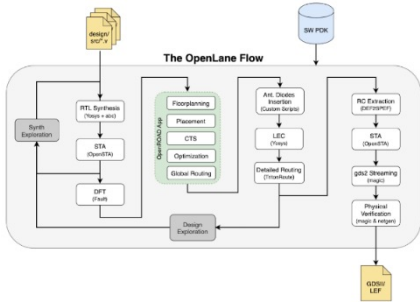
PDK: Process design kits from foundary



Flow



Flow: Chip design flow config and script



Open-source RTL, PDK and Flow

Open-source RTL

Accelerator	Analog	Connectivity	CPU	FPGA	Memory	System
aes/aes_core	AMS_KGD	aib	OpenXiangShan	FABulous	core_axi_cache	Beagle_SDR_GPS
ara	open-pmic	aib-protocols	a2i	fabric_team	HuanCun	bsg_manycore
FFTGenerator	Analog Basic Blocks/LDO	core_ddr3_controller	black-parrot	OpenFPGA	openram	cep
fpu		HDMI	Cores-SweRV	prga	lake	esp
garnet		i2c	core-v-verif			hero
gplgpu		litedram	cva6			litex
core_jpeg		liteeth	cv32e40p			openFASOC
vortex		litepice	ibex			openpiton
VeriGPU		litescope	microwatt			opentitan
tvm-vta		pymtl3-net	neorv32			openwifi-hw
...		verilog-ethernet	picorv32			pulp
		ravenoc	rocket-chip			pulpissimo
		verilog-uart	serv			
			snitch			
			boom			
			Low-RISC			
			OpenXuantie - OpenC910 Core			
			ysyx			

Open-source PDK

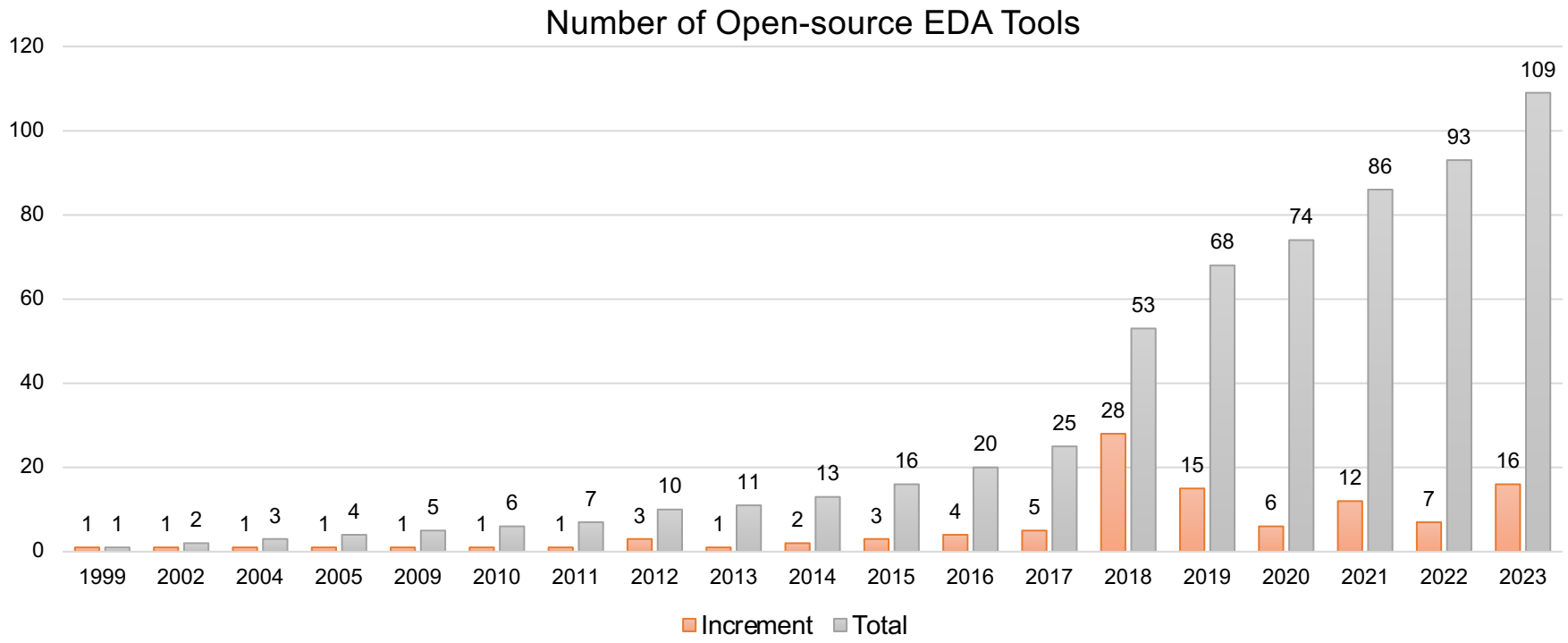
PDK name	Process node	Foundary	Institution
Sky130	130nm	Skywater	Google/Efabless
Sky90	90nm	Skywater	Google/Efabless
gf180	180nm	GlobalFoundries	Google/Efabless
NanGate45	45nm	Fake	Si2
Asap7	7nm	Fake	ARM Ltd

Open-source Flow

Flow	Function	Contributor Institution
Qflow	RTL-GDS	Efabless
VSDFLOW	RTL-GDS	VLSI System Design
OpenRoad	RTL-GDS	UCSD
OpenLane	RTL-GDS	Efabless/UCSD
Ophidian	Netlist-GDS	UFSC
Rsyn	Physical Synthesis	FURG
SiliconCompiler	RTL-GDS	Zero ASIC
SymbiFlow	FPGA design flow	F4PGA
iFlow	**RTL-GDS**	**PCL/ICT/BOSC/…**

Increasing Open-souce EDA Tools

- Open-souce in EDA may be a tendency

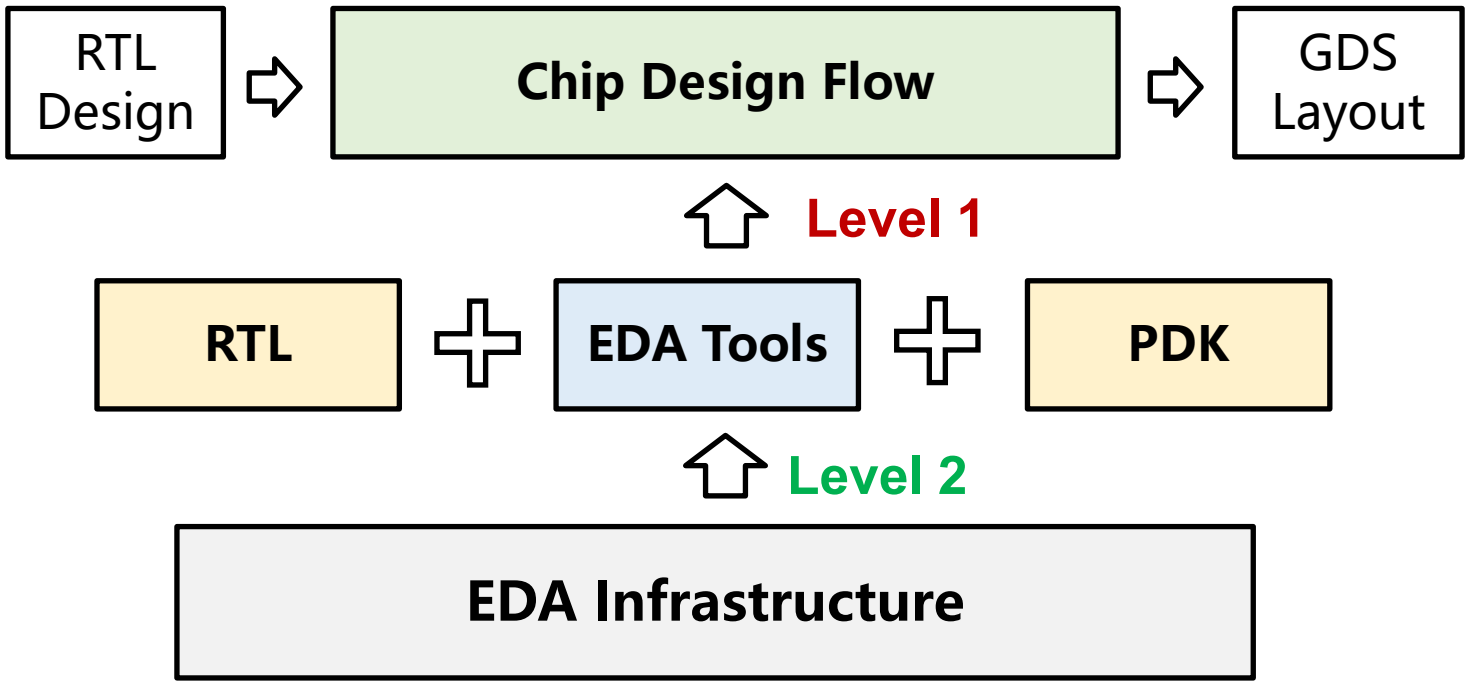


Open-source EDA Tools

Design Module	Design Step	Some Open-source Tools					iEDA
HLS	HLS	LegUp	GAUT	PandA	FCUDA	XLS	
Simulation Verification	Logic Simulation	Verilator/iVerilog	GHDL	FreeHDL	TkGate		
	Circuit Simulation	NGSpice	mixedsim	GnuCap	Qucs	XICE	
	Debug						
Logic Synthesis	Logic Synthesis	Yosys	ABC	EPFL-LS-Lib	LLDHL	UNIVR	iLS
	Tech Map	ABC					iMap
DFT	DFT	Fault					
Formal	Formal						
Physical Design	Partition	PartitionMgr	METIS	KaHyPar	MPPart		iNO
	Floorplan	OpenRoad TritonMacroPlacer	OpenRoad	Parquest			iFP
	PDN	OpenPDN					iPDN
	Placement	RePlace OpenDP	DreamPlace	Graywolf	Capo		iPL
	CTS	TritonCTS					iCTS
	Timing OPT	TritonSizer	Gate-Sizing				iTO
	Routing	FastRoute TritonRoute	CUGR Dr.CU	Qrouter	NTHU-Route	BoxRouter/FGR/ORGE	iRT
	ECO	OpenRoad-eco					iECO
Signoff	STA	OpenSTA	OpenTimer				iSTA
	RCX	OpenRCX	SPEF-Extractor				iRCX
	Power	OpenRoad-pp					iPA
	IR Drop	PDNSim	IREDGe				iIR
Physical Verification	DRC	Klayout	Magic				iDRC
	Antenna	OpenRoad-ant					
	LVS	Netgen					
Layout Synthesis	MPL						
	RET/ILT						
	Mask Generation						

We Need Infrastructure

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



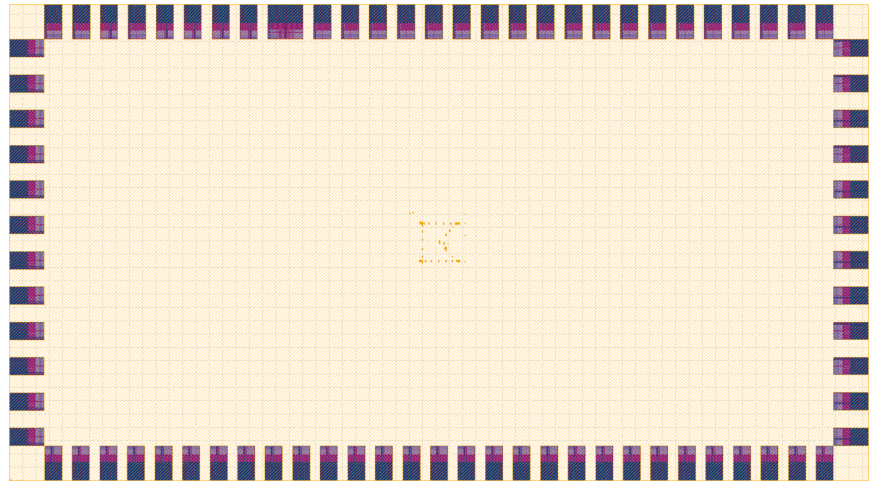
01 Introduction

02 iEDA

03 iEDA Application

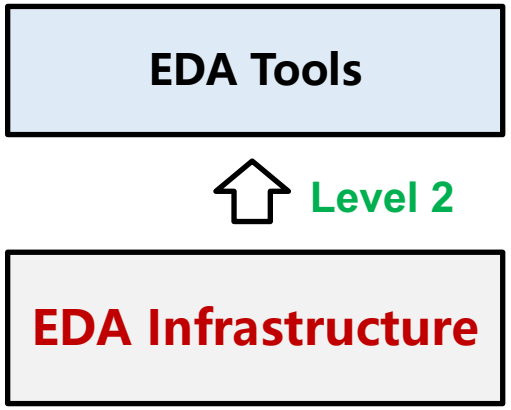
iEDA Introduction

- **iEDA Objective**
  - **EDA Infrastructure**
  - **Explore new** and efficient **EDA R&D method**
  - **High quality and performance EDA tool**



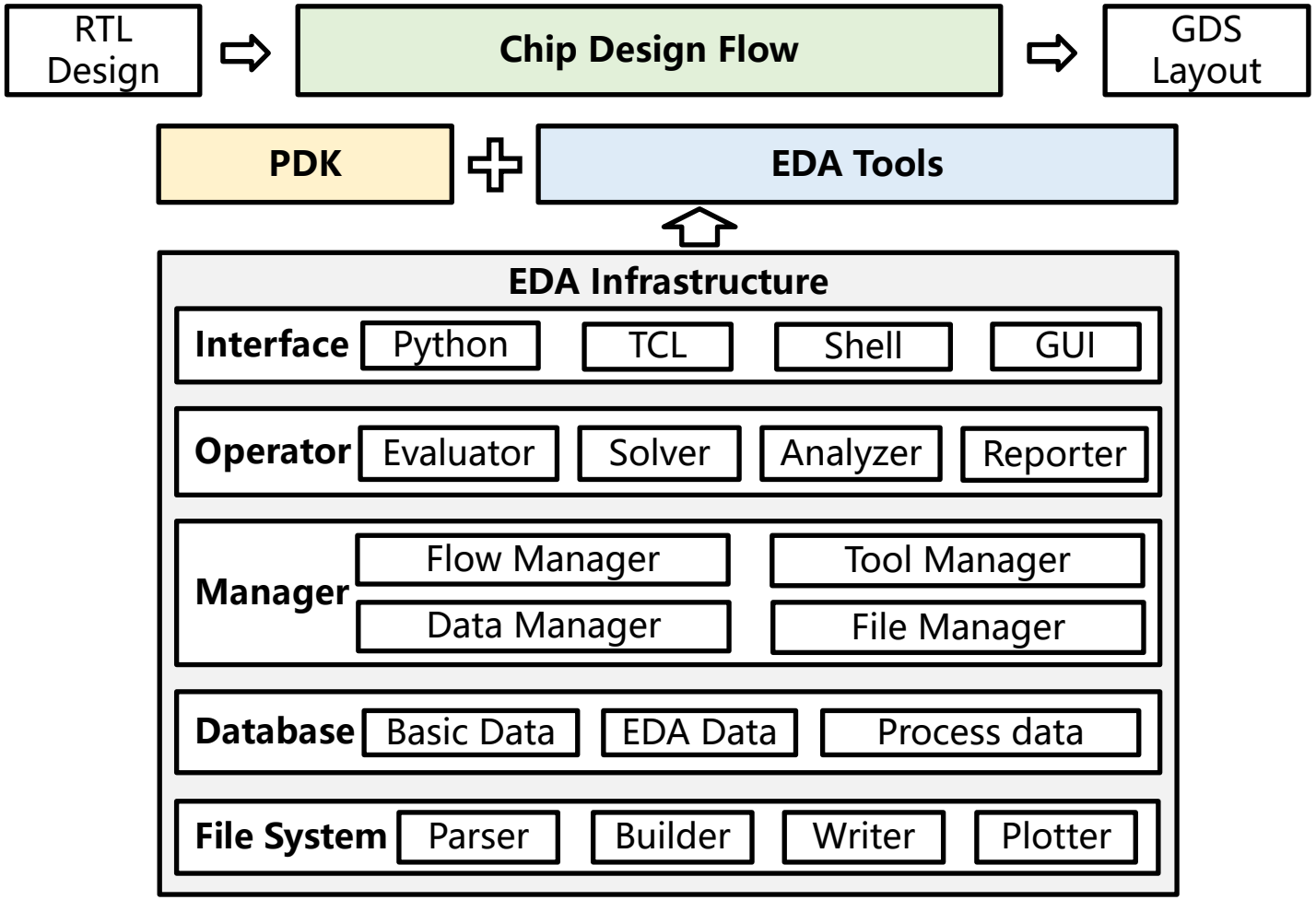
Open-source is not a goal but a way

- **Open-source：(Gitee/Github)**
  - **Gitee: https://gitee.com/oscc-project/iEDA**
  - **GitHub: https://github.com/OSCC-Project/iEDA**



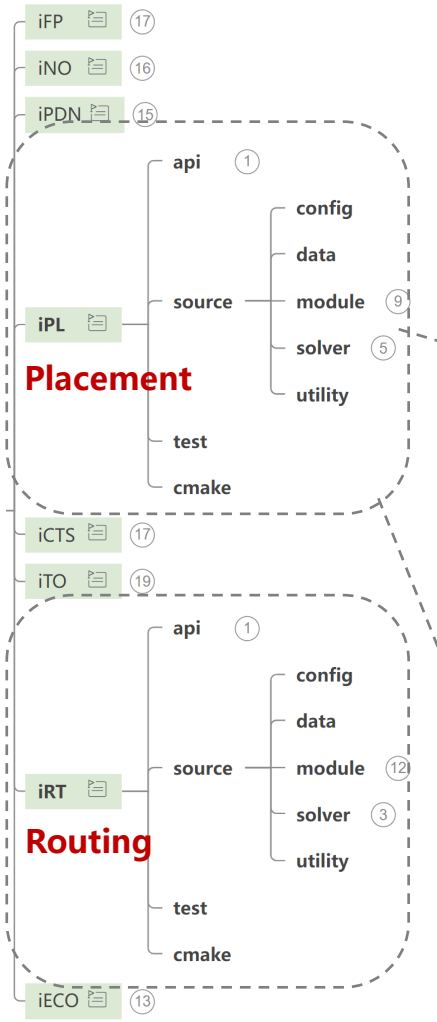
iEDA-Infrastructure

- File System
- Database
- Manager
- Operator
- Interface
- Utility
- Some perf tools

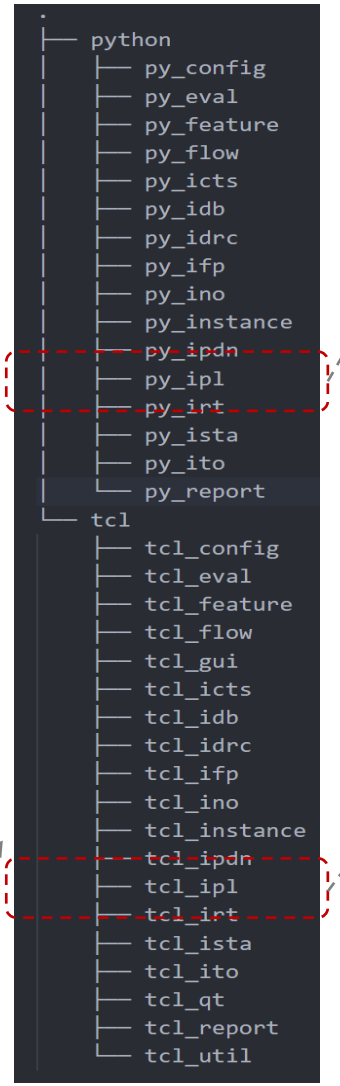


Uniform Software Framework and API

Software Structure



API



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()
```

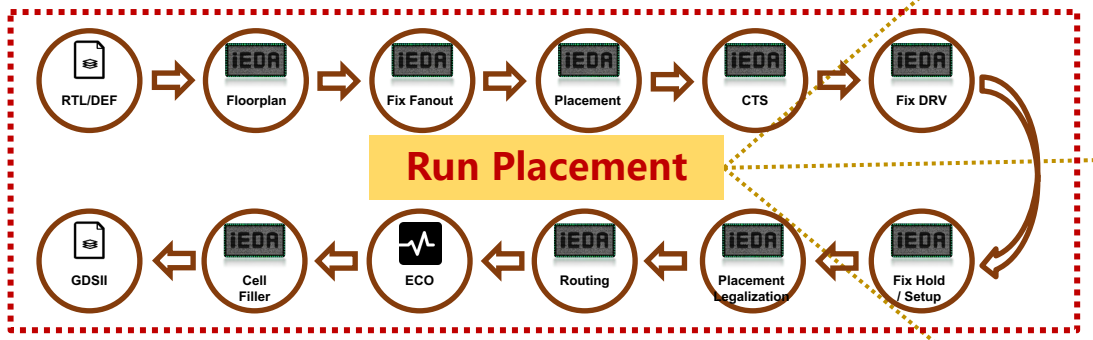
Python

```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

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

TCL API

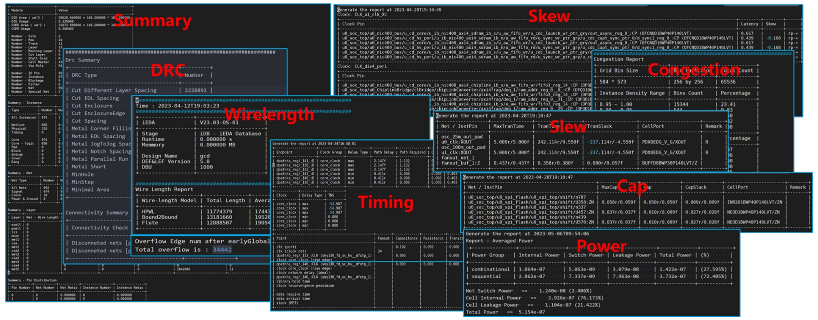
```
#============================================================
##   read def
#============================================================
def_init -path ./result/iTO_fix_fanout_result.def

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

Python API

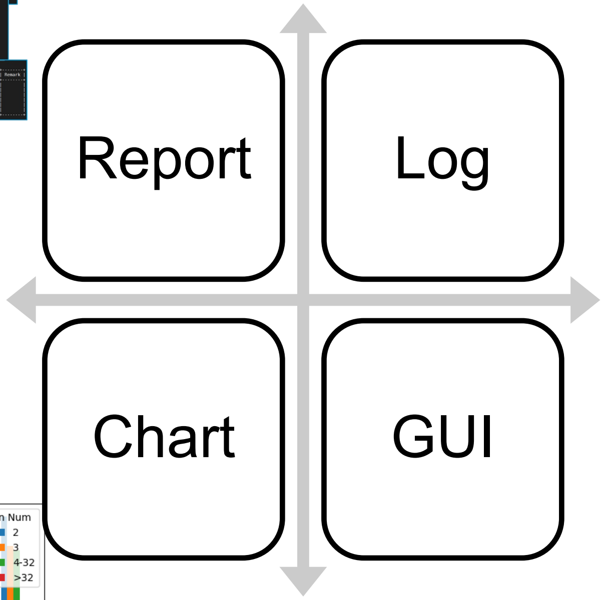
```
def run_placer(self, input_def : str):
    self.read_def(input_def)

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

Multiple Data Analysis and Debug Methods

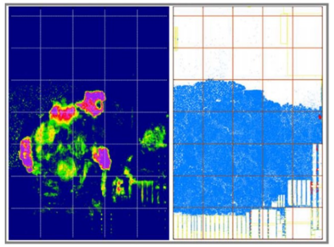
Report

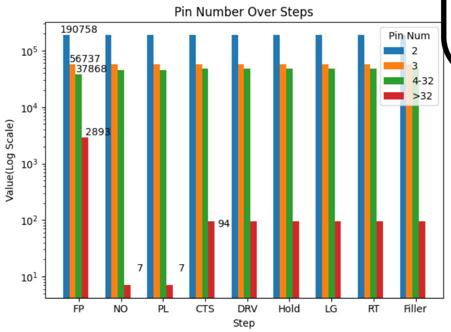


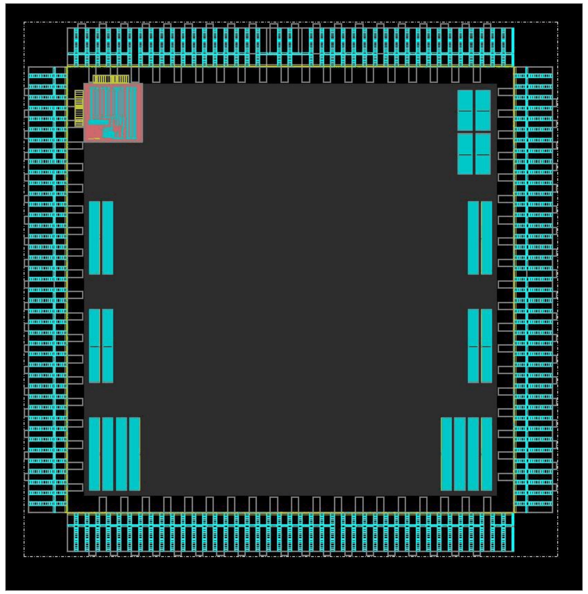
Log



Feature Chart







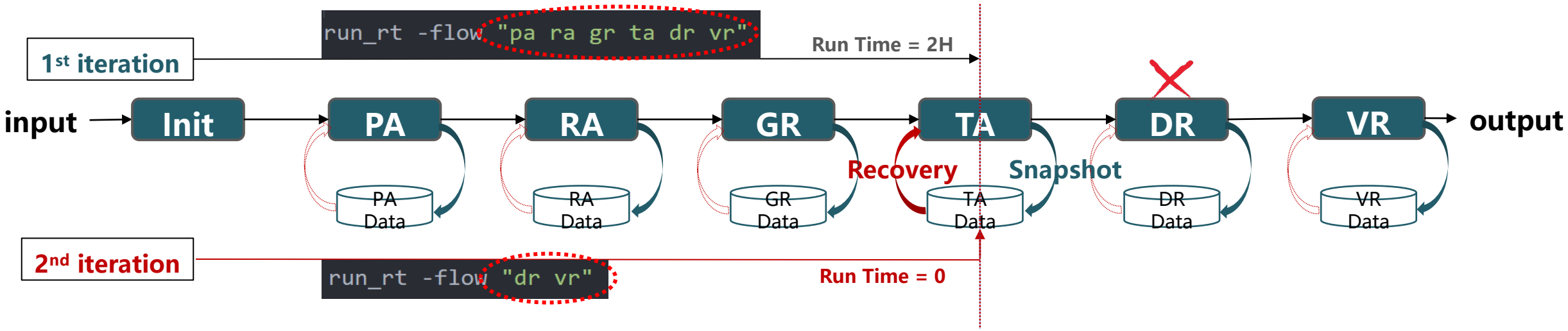
GUI

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



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

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
```

iEDA

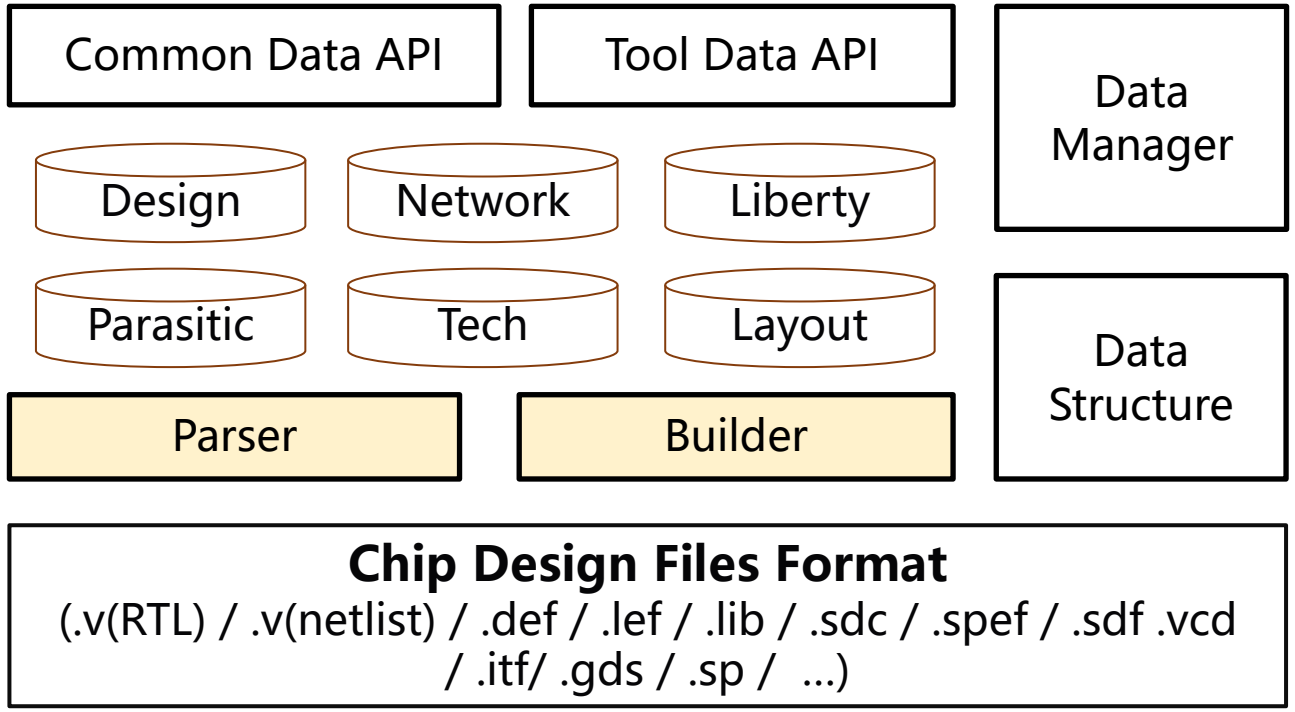
文件 (4734)

- .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

```
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
```

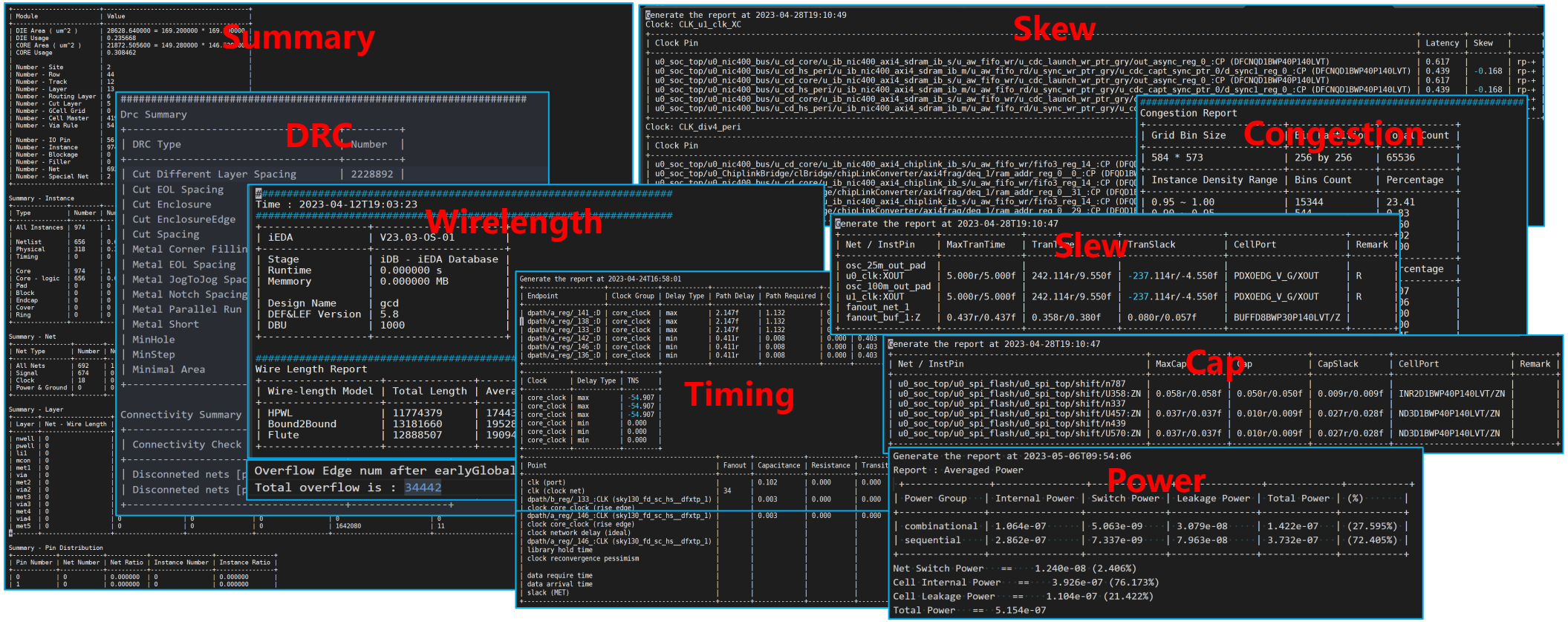
Parser and Database

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



Reporter

- Metrics
  - Timing, power, area, wirelength, congestion, skew, res, cap, slew, fanout, utilization, density, cells, nets, drc (metal, cut, connection), …



Evaluator: Horizontal Comparison

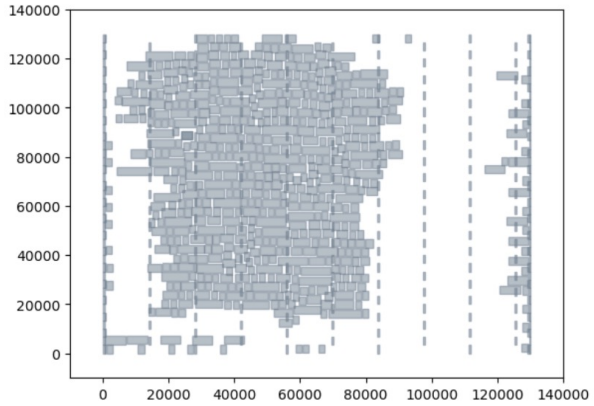
- Compare and analyze the Q&R of designs, tools, algorithms and flows

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

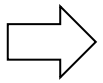
Design Comparison

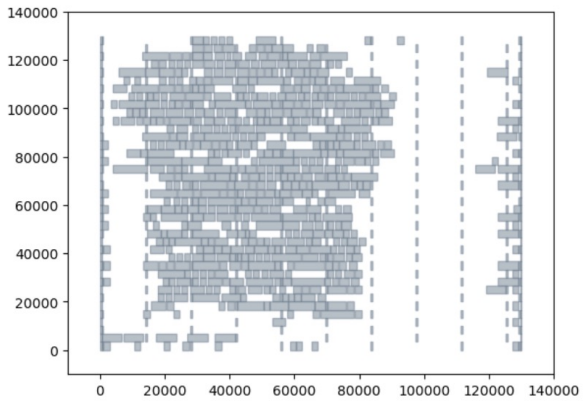
Evaluator: Horizontal Comparison

- Compare and analyze the Q&R of designs, tools, algorithms and flows

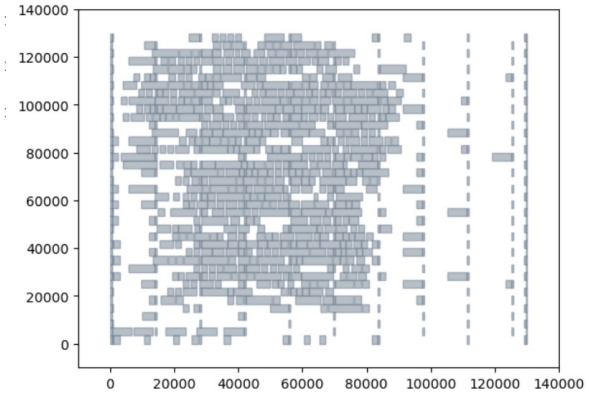


Input





Abacus



Tetris

Algorithm Comparison

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

Evaluator: Horizontal Comparison

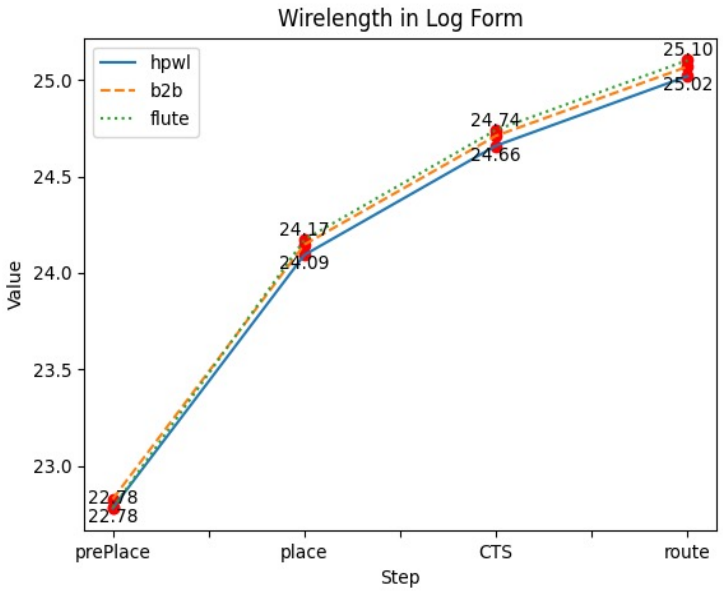
- Compare and analyze the Q&R of designs, tools, algorithms and flows

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

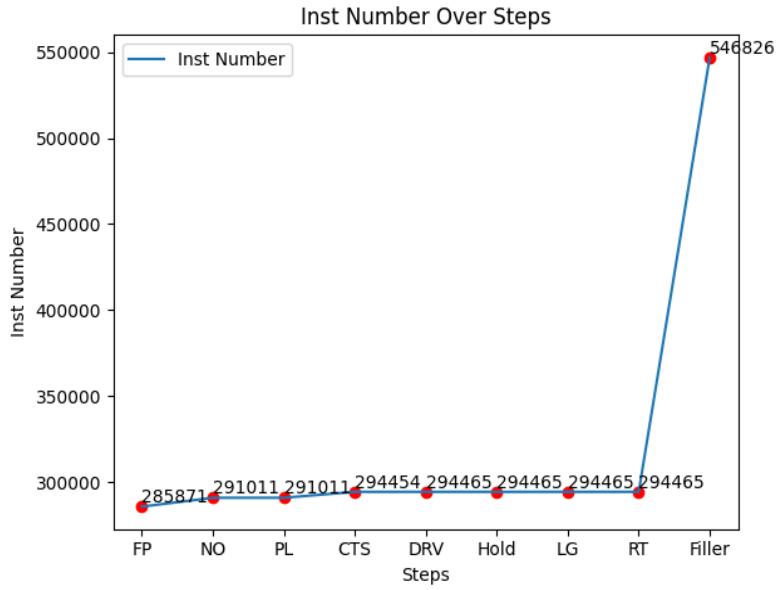
Flow Comparison

Analyzer : Vertical Comparison

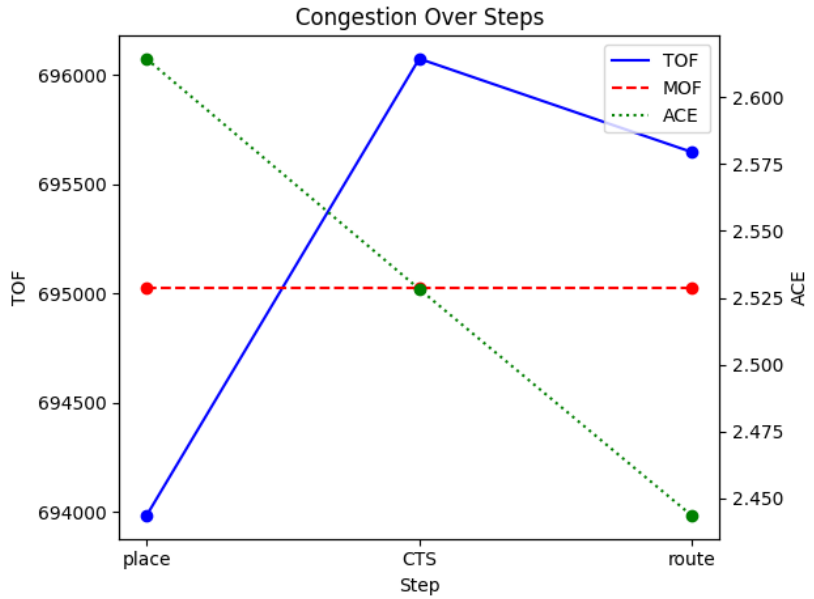
- Analyzing the numerical changes of an indicator **across different stages**
- Differences from: 1) data change, and 2) differences evaluation models
- Usage： evaluating the design quality, analyzing the margin, and optimizing collaboratively



Wirelength



#Instances



Congestion

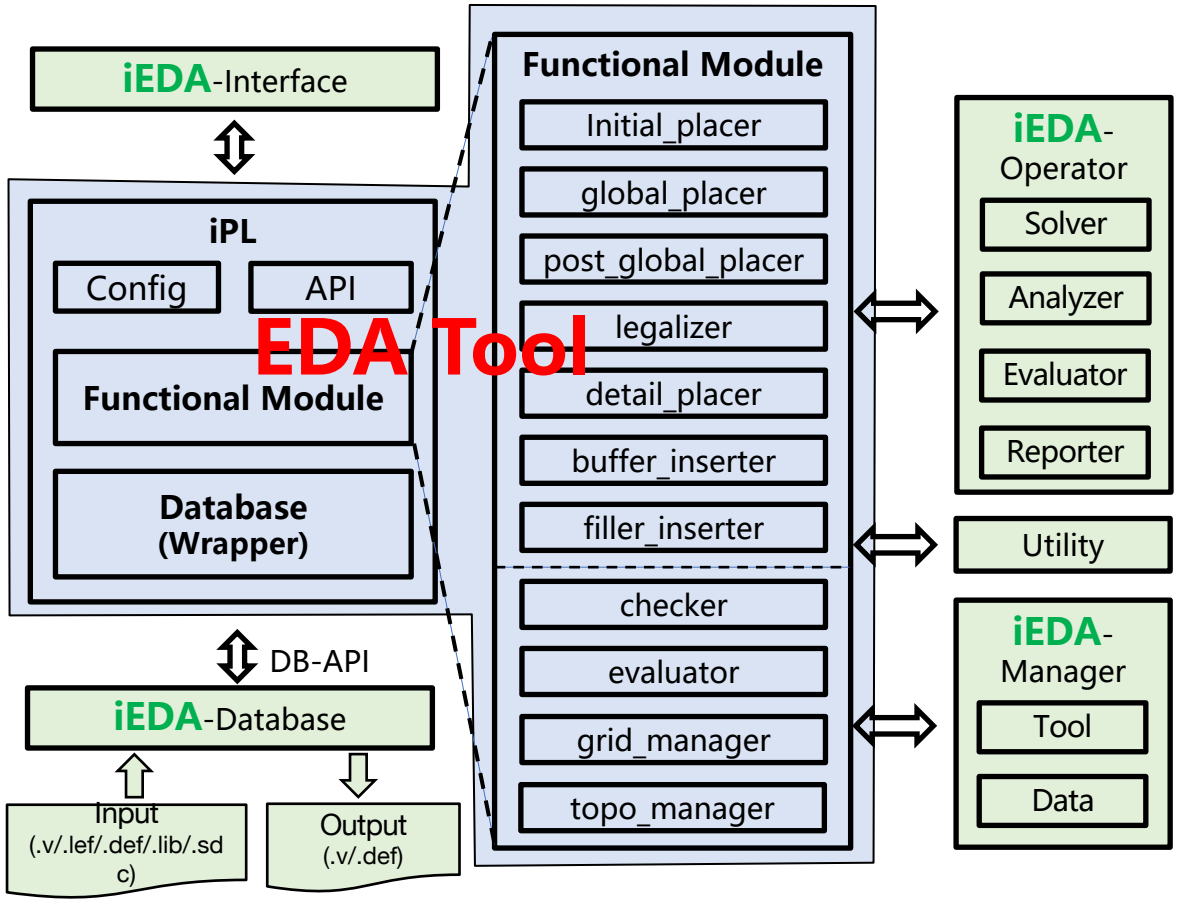
01 Introduction

02 iEDA

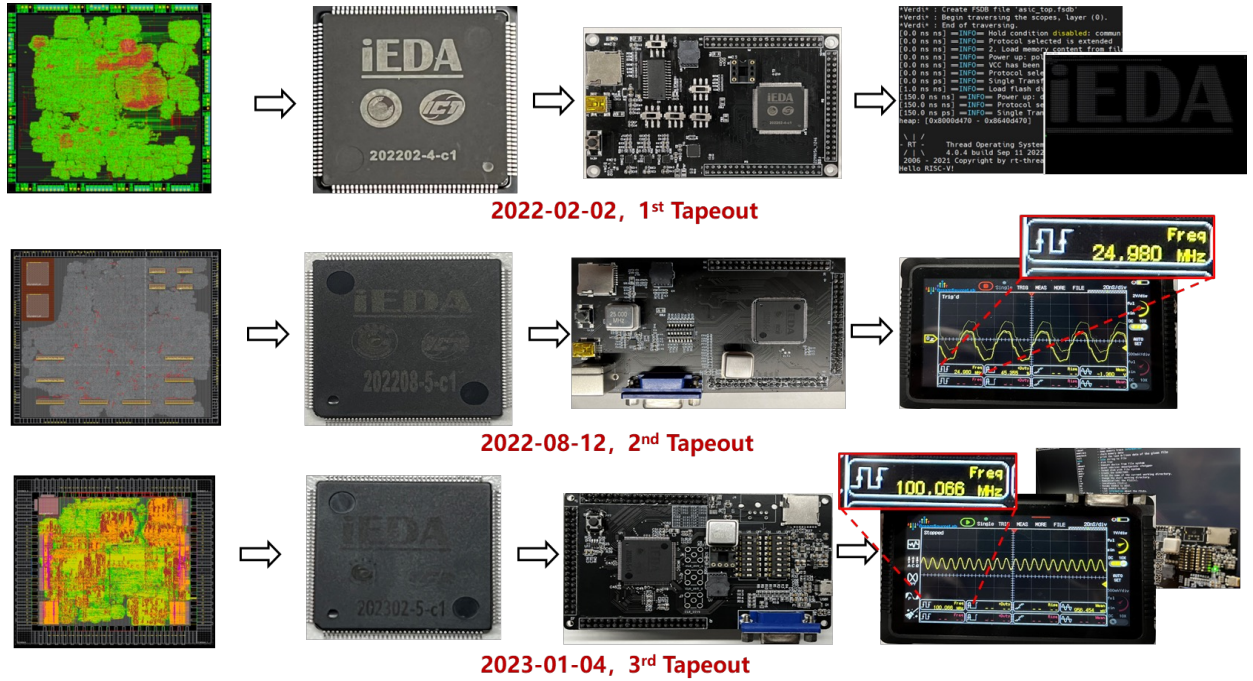
03 iEDA Application

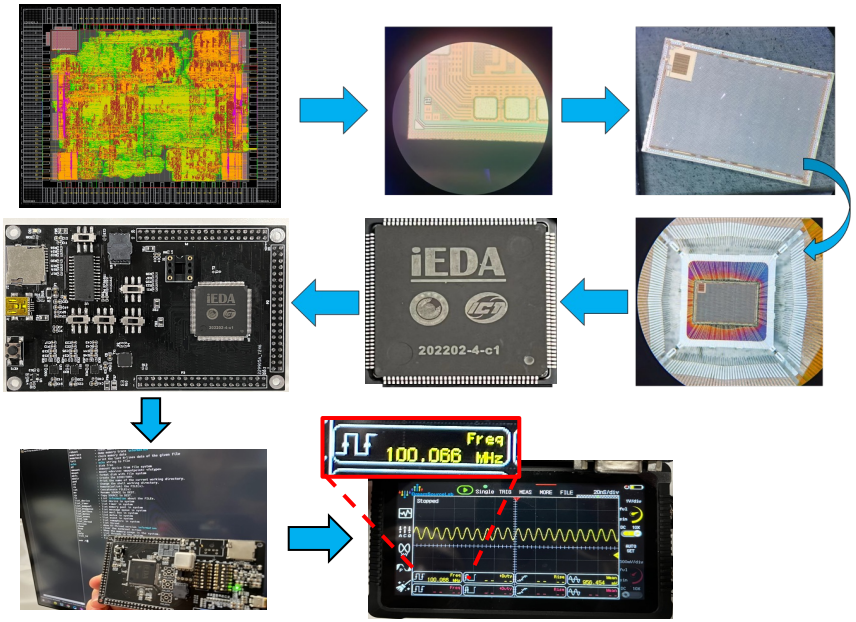
Developing EDA Tool or Algorithm

- 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



Designing Chip





GDS->Tapeout-Package-PCB-Debug

1st Tapeout

- **RTL**：**ysyx(一生一芯)-03**
- **PDK**：110nm
- **Area**：**3mm × 3.5** mm
- **Power**：dynamic = 48mW，leakage = 7 mW
- **Freq.**：**25M**Hz
- **Scale**：>**0.7M** Gates
- **Features**：5 pipeline，Chiplink、UART、SPI，PCB clock，support RT-thread

2nd Tapeout

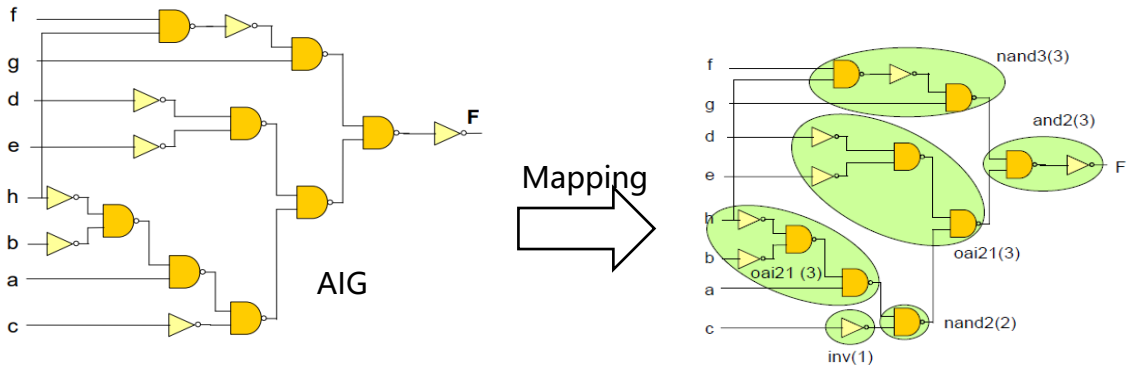
- **RTL**：**ysyx(一生一芯)-04**
- **PDK**：110nm
- **Area**：**4.5mm × 4.5** mm
- **Power**：dynamic = 343mW，leakage = **21** mW
- **Freq.**：**25M**Hz
- **Scale**：>**1.5M** Gates
- **Features**：11 pipelines with cache，IP：UART、VGA、PS/2、SPI、SDRAM、 2 PLLs，support Linux

3rd Tapeout

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

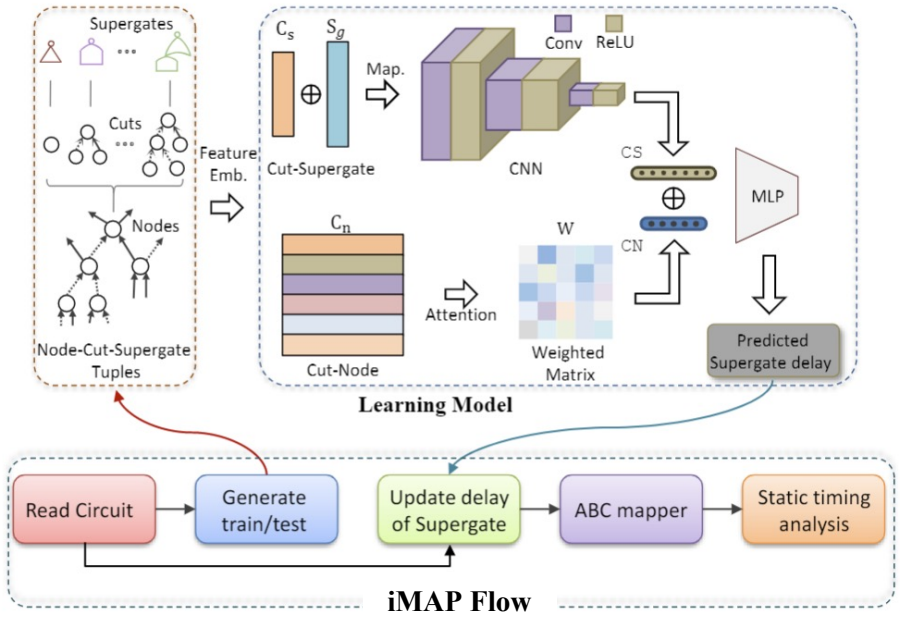
Research：Learning to Optimize Tech Mapping

- **Based on iMAP，we propose a ML method to predict cut delay**



Circuits	Estimated Results Area(μm^2)	LI-Delay(ps)	Actual Results Area(μm^2)	Delay(ps)	Δ Delay
adder	898.31	2,613.78	898.13	3,770.65	44%
bar	2,681.62	152.96	2,680.39	1,114.9	629%
log2	26,556.98	3,891.66	26,561.26	6,797.77	75%
cavlc	463.27	185.07	463.29	93.2	50%
int2float	158.61	174.27	158.63	91.7	47%
ctrl	106.92	98.53	106.84	89.9	9%

LI-Delay refers to the load-independent delay estimation in ABC [1]. The actual delay is computed by the non-linear delay model.



- AiMap: Learning to Improve Technology Mapping for ASICs via Delay Prediction, in Proc. of ICCD'23.

Talent Training： Curriculum Training Platform

- **Min Wirelength Model**

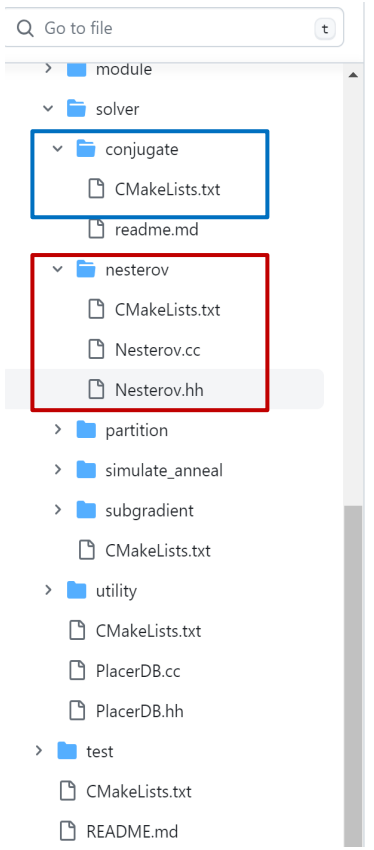
$$\min_{\boldsymbol{v}} \quad W(\boldsymbol{v})$$
$$s.t. \quad \rho_b(\boldsymbol{v}) \le \rho_0, \qquad \forall b \in B$$

- 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 $k = 0,1,2,\ldots$ until $\|r_k\| = 0$

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




```
38  class Nesterov
39  {
40   public:
41    Nesterov();
42    Nesterov(const Nesterov& other) = delete;
43    Nesterov(Nesterov&& other) = delete;
44    ~Nesterov() = default;
45
46    // getter.
47    int get_current_iter() const { return _current_iter; }
48    const std::vector<Point<int32_t>>& get_current_coordis() const { return _current_coordis; }
49    const std::vector<Point<float>>& get_current_grads() const { return _current_gradients; }
50    const std::vector<Point<float>>& get_next_grads() const { return _next_gradients; }
51    const std::vector<Point<int32_t>>& get_next_coordis() const { return _next_coordis; }
52    const std::vector<Point<int32_t>>& get_next_slp_coordis() const { return _next_slp_coordis; }
53    float get_next_steplength() const { return _next_steplength; }
54
55    // for RDP
56    const std::vector<Point<float>>& get_next_gradients() const { return _next_gradients; }
57    float get_next_parameter() const { return _next_parameter; }
58    void set_next_coordis(const std::vector<Point<int32_t>>& next_coordis) { _next_coordis = next_co
59    void set_next_slp_coordis(const std::vector<Point<int32_t>>& next_slp_coordis) { _next_slp_coord
60    void set_next_gradients(const std::vector<Point<float>>& next_gradients) { _next_gradients = nex
61    void set_next_parameter(float next_parameter) { _next_parameter = next_parameter; }
62    void set_next_steplength(float next_steplength) { _next_steplength = next_steplength; }
63
64    // function.
65    void initNesterov(std::vector<Point<int32_t>> previous_coordis, std::vector<Point<float>> previo
66                      std::vector<Point<int32_t>> current_coordis, std::vector<Point<float>> current
67    void calculateNextSteplength(std::vector<Point<float>> next_grads);
68
69    void runNextIter(int next_iter, int32_t thread_num);
70    void runBackTrackIter(int32_t thread_num);
71
```

- Assignment: please implement CG method by C++ or Python, and test it on “iEDA/iPL”

Talent Training: Support Contest

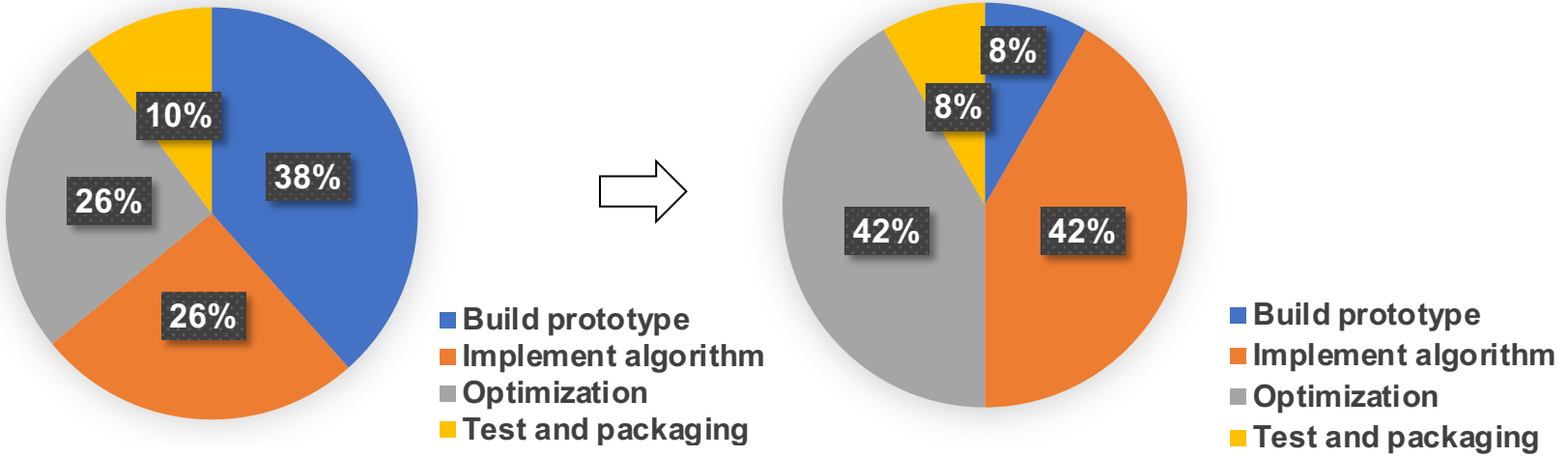
iEDA

文件 (4826)

- .gitee
- cmake
- contest
- docs
- scripts
- src
- .clang-format
- .clang-tidy
- .gitignore
- CMakeLists.txt
- LICENSE
- README-En.md
- README.md
- build.sh

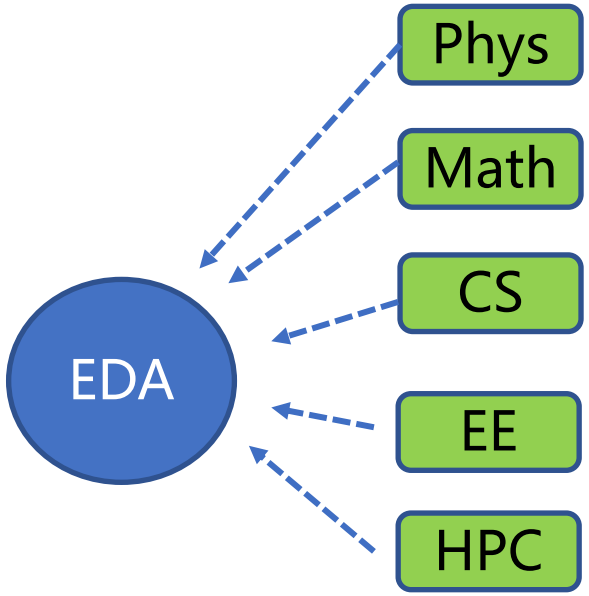
1st Place at 2022 ICCAD@ Contest

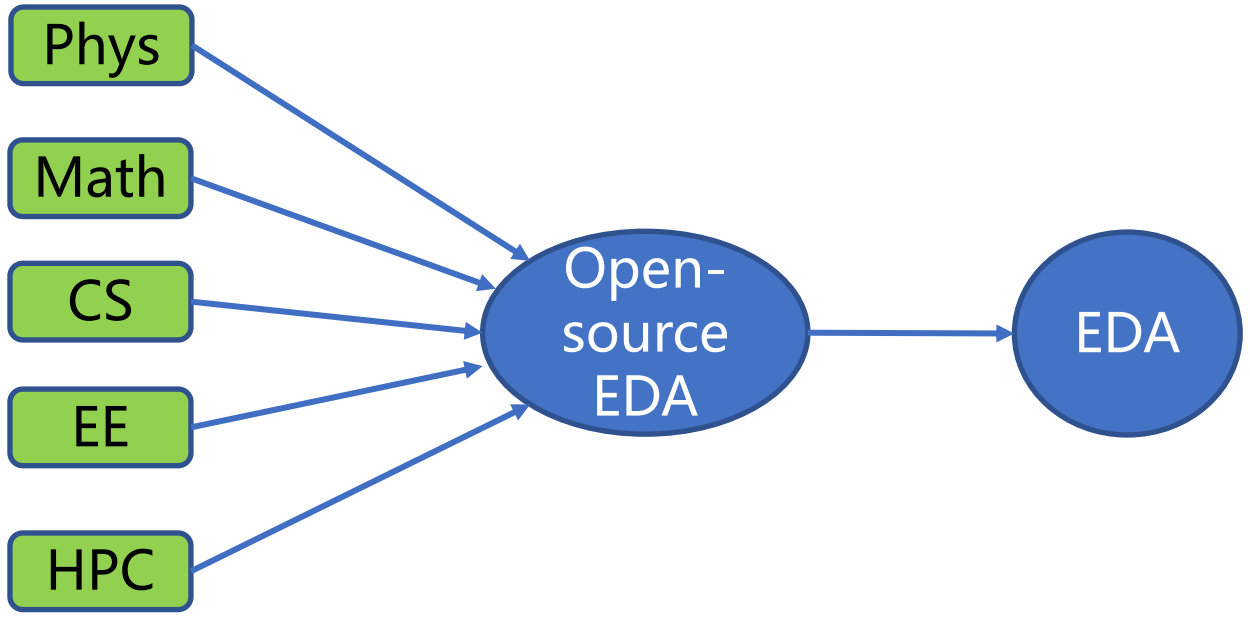
2st Place at 2023 ICCAD@ Contest



Proportion of time spent

Closed vs. Open





Closed-source EDA

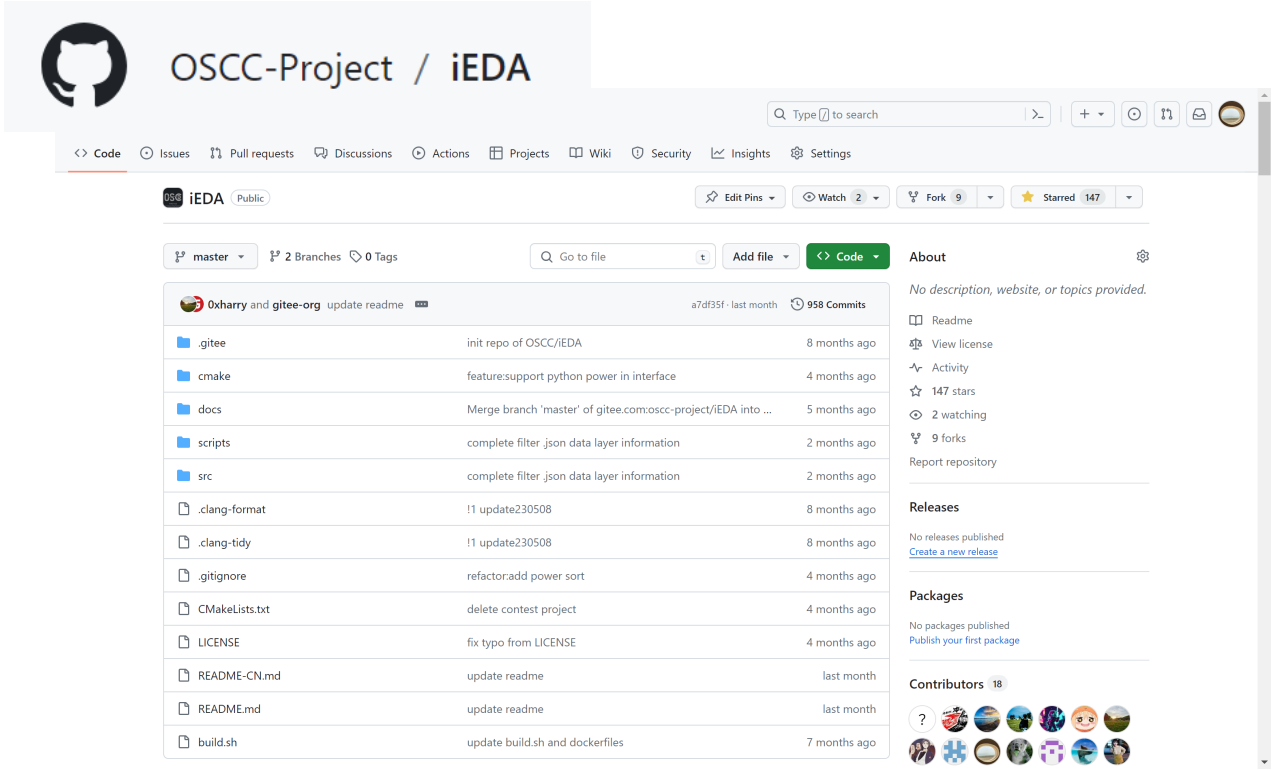
Open-source EDA is a bridge

Conclusions

- **iEDA**
  - EDA Infrastructure, explore EDA method, develop High quality EDA tool
  - Uniform software, multiple language, data analysis and debug, data recovery, evaluation (vertical and horizontal)
- **iEDA Application**
  - Develop EDA tool
  - Design chip and tape-out
  - EDA research
  - Curriculum Training
  - EDA Contest

Future Works

- **Upcoming Solvers**
  - ILP solver,
  - SAT solver,
  - Partition solver,
  - Steiner tree solver,
  - Geometric calculator,
  - Non-linear program optimizer
- **Analyzer:** analyze and visualize process data
- **Performance:** improve data library efficiency
- **EDA Parsers:** implemented by RUST



Thanks

Xingquan Li
lixq01@pcl.ac.cn