オープンソースEDAを活用したLSI開発環境の開拓

日置雅和、堀洋平、片下敏宏、更田裕司、秋田一平

先端半導体研究センター
集積回路設計研究チーム

自己紹介

- 2003年3月　東北大学大学院工学研究科電子工学専攻修了　舛岡富士雄先生（フラッシュメモリの発明者@東芝）
  SGT（GAA）型フラッシュメモリセルの設計と形状効果の研究、博士（工学）
- 2003年4月～　産総研入所　電力再構成可能FPGAであるFlex Power FPGAのアーキテクチャ設計、
  回路設計、チップ評価
- 2012年11月～2013年11月　経産省情報通信機器課（現情報産業課）出向
- 2013年～2018年　SOTB（Silicon On Thin Buried oxide）版Flex Power FPGA設計・評価、
  超伝導量子ビット設計環境整備・簡易PDK作成・設計
- 2018年9月～　産総研・東大 AIチップデザインオープンイノベーションラボラトリ　チーム長
- 2020年4月～　デバイス技術研究部門先端集積回路研究グループ　グループ長
- 2023年10月～　先端半導体研究センター（SFRC）集積回路設計研究チーム　チーム長

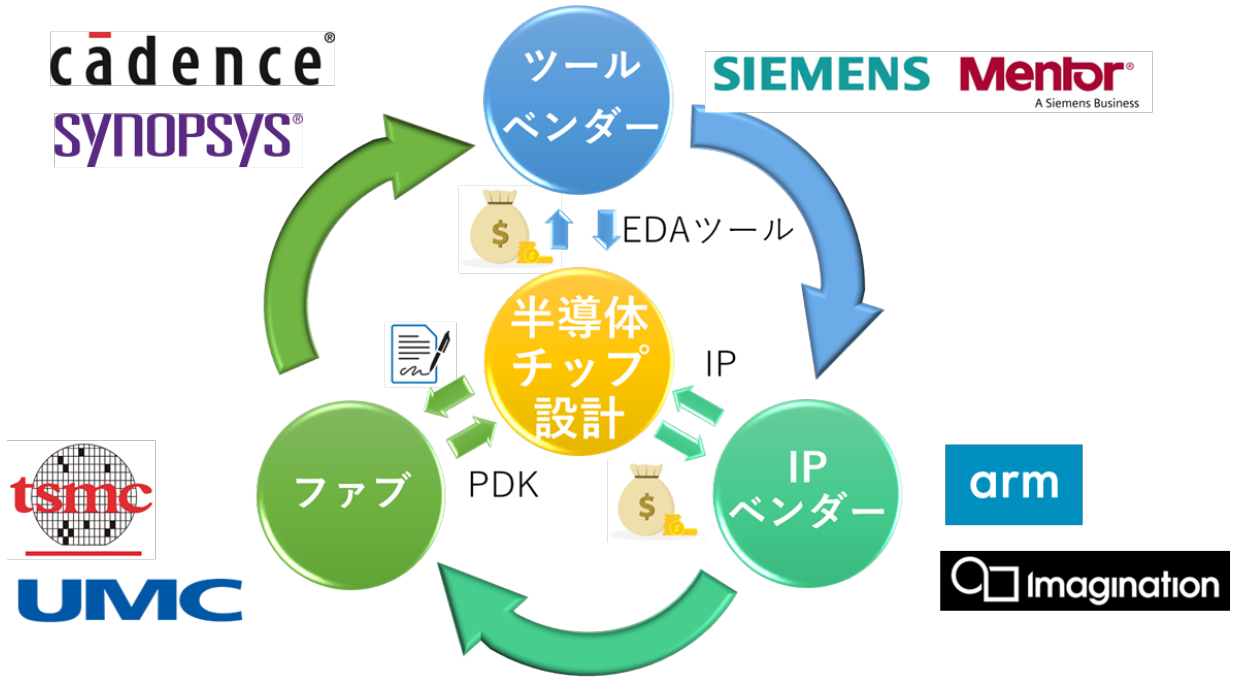
LSI開発のエコシステム

✓現在のLSI開発：既存エコシステムの束縛⇒**新規参入者にとって高い参入障壁**

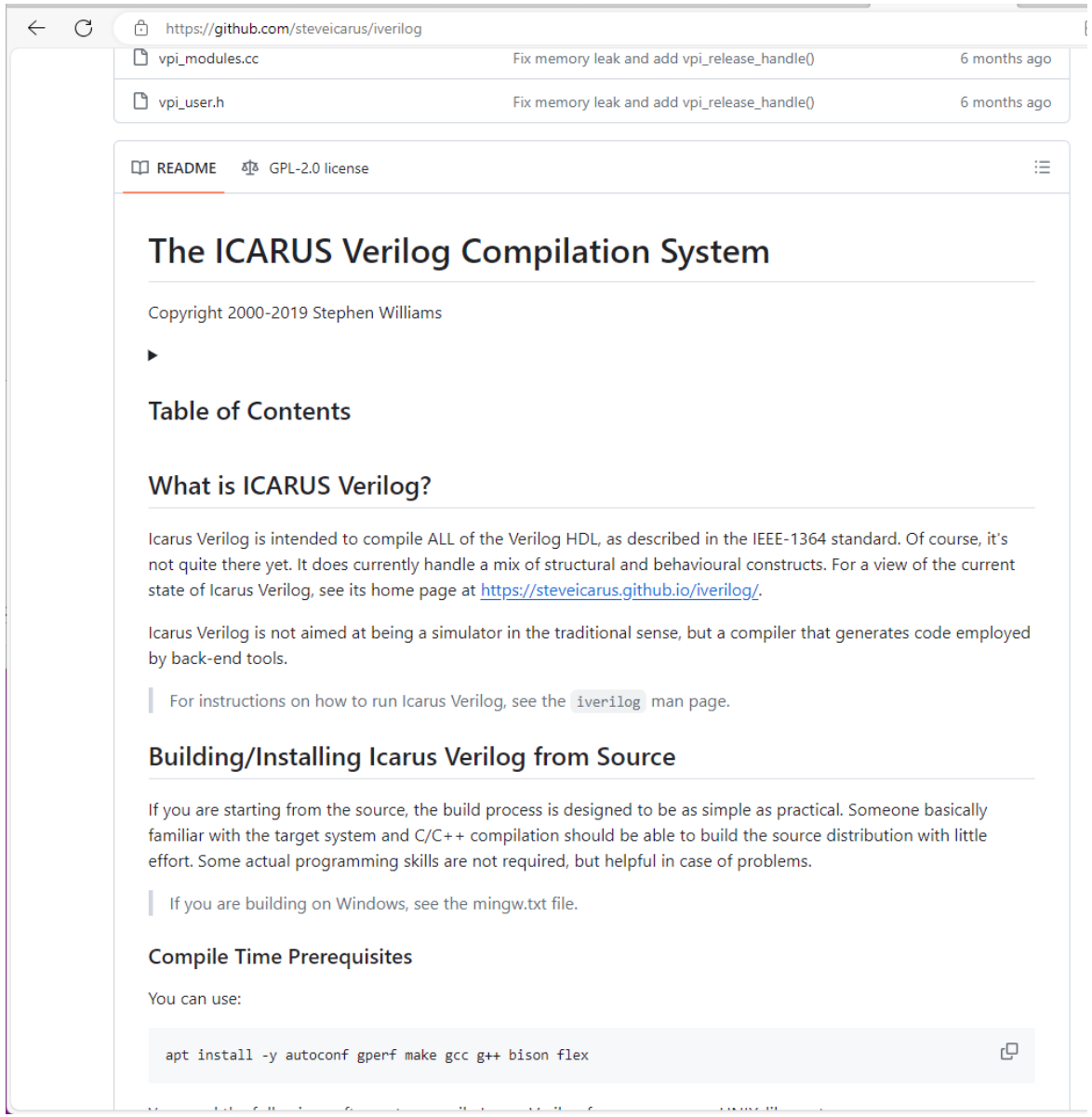
- 高価な**EDAツール**（設計ソフトウェア）
- 高価な**IP**（設計資産：インターフェース、メモリ等）
- 入手に手間のかかる**PDK**（Process Design Kit：デザイン情報）

⇒どれから梯子をかける？

フリーのEDAツールは昔から脈々と開発されていた



ICARUS Verilogシミュレータ（1998年～）



GTKWave　波形表示（10年くらい前～）

産総研
ともに挑む。つぎを創る。

https://gtkwave.sourceforge.net

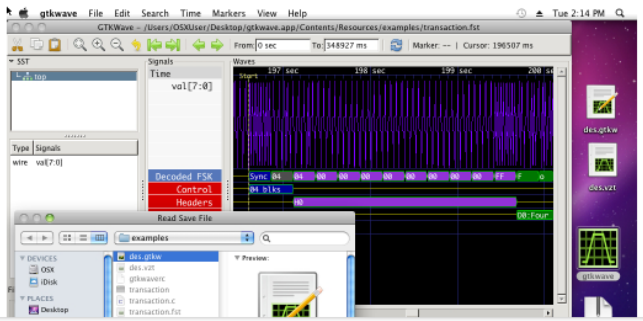
Welcome to GTKWave

GTKWave is a fully featured GTK+ based wave viewer for Unix, Win32, and Mac OSX which reads LXT, LXT2, VZT, FST, and GHW files as well as standard Verilog VCD/EVCD files and allows their viewing. You can grab version 3.3.117 here. Documentation in pdf format can be found here.

For svn access to the experimental, pre-release sourcetree on Sourceforge:
svn checkout svn://svn.code.sf.net/p/gtkwave/code/ gtkwave-code

Native Win32 and OSX binaries are available here, however if you are a Windows user running Cygwin, running under that is recommended instead.
A Mac port can be found both here and here.
Ports to other platforms which GTK supports should be trivial.

An experienced professional shown violating most known rules of electrical safety with GTKWave. Please do not attempt this at home.



Xschem　回路図エディタ＋ネットリスティング（1998～）

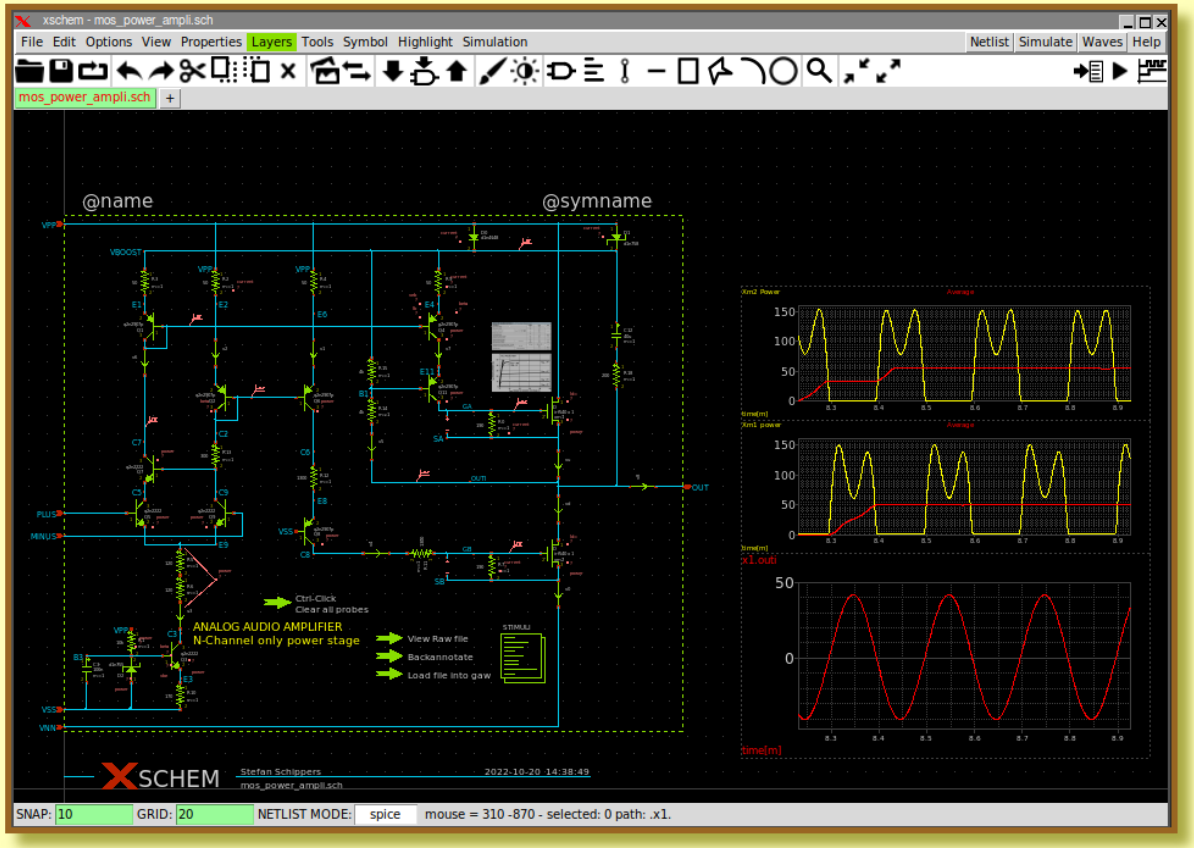
https://xschem.sourceforge.io/stefan/index.html

HOME　XSCHEM MANUAL　GOOGLE

- Solaris sparc
- Windows (with the cygwin layer and cygwin/Xorg X11 server, plus the tcl/tk toolkit and the -dev libraries)

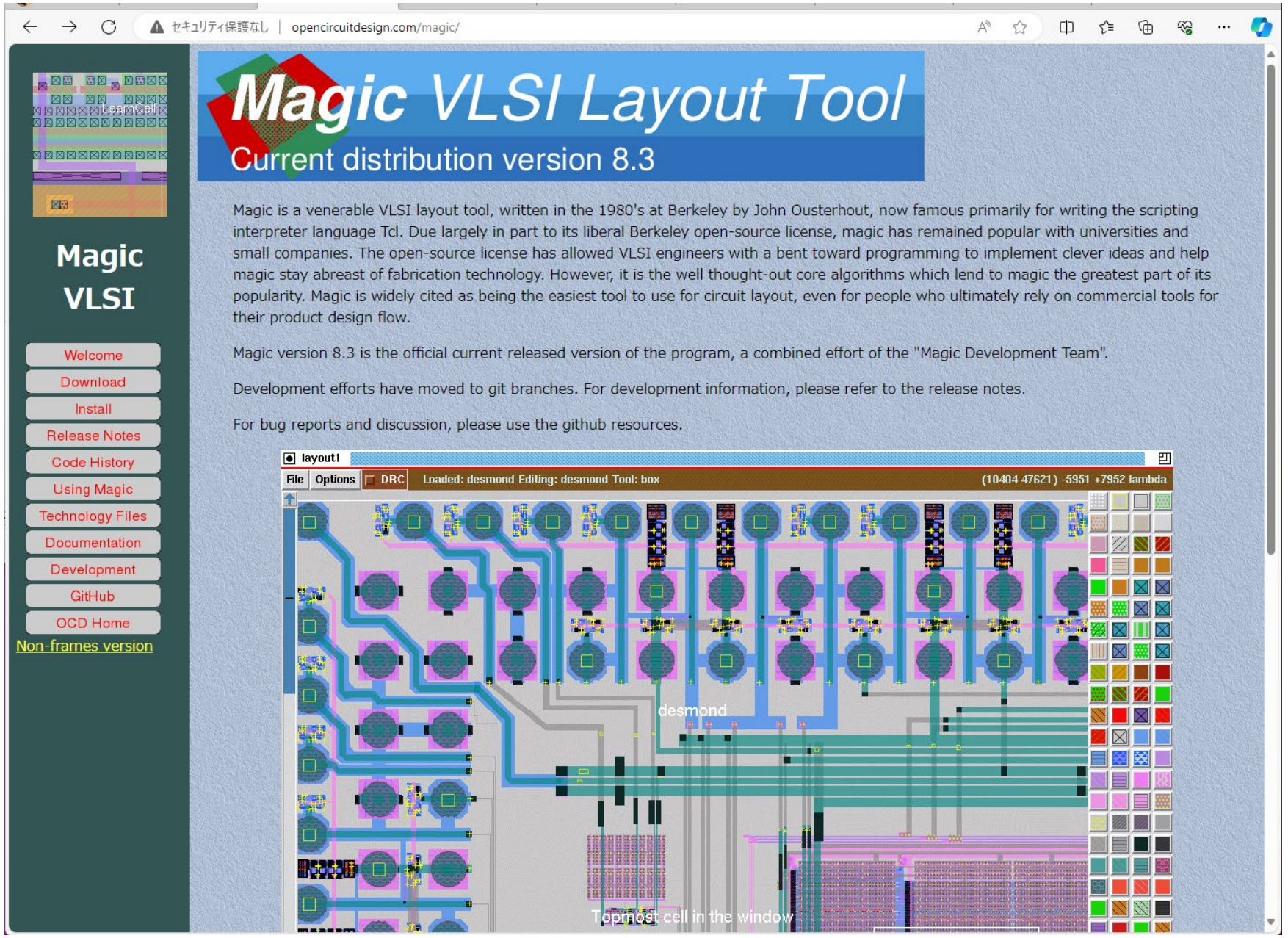
Screenshots

- analog circuit example

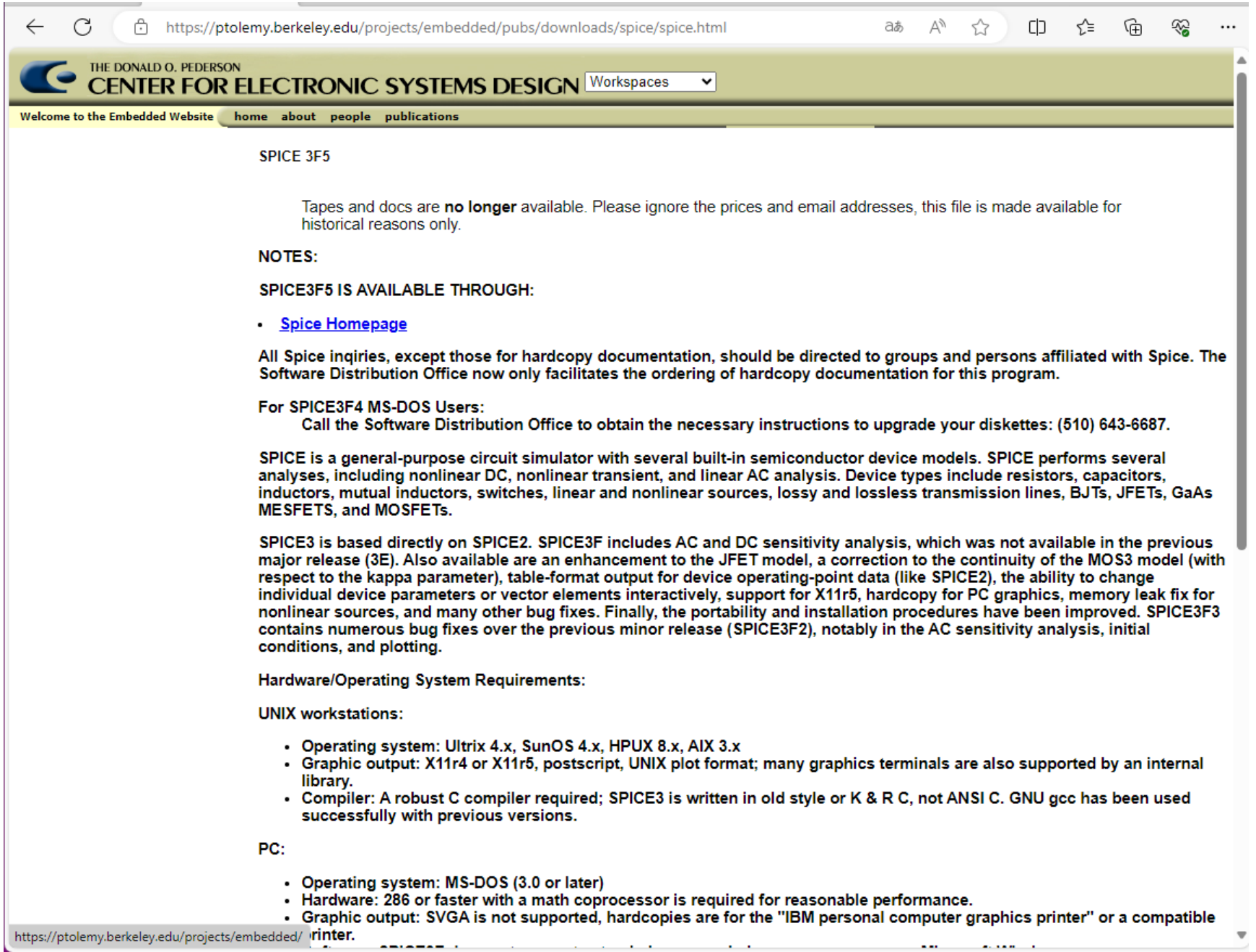


- digital system for VHDL simulation

Magic レイアウトエディタ（1980's～）



SPICE3　アナログ回路シミュレータ（1996～）



https://ptolemy.berkeley.edu/projects/embedded/pubs/downloads/spice/spice.html

THE DONALD O. PEDERSON
CENTER FOR ELECTRONIC SYSTEMS DESIGN　Workspaces

Welcome to the Embedded Website　home　about　people　publications

SPICE 3F5

Tapes and docs are **no longer** available. Please ignore the prices and email addresses, this file is made available for historical reasons only.

NOTES:

SPICE3F5 IS AVAILABLE THROUGH:

- Spice Homepage

All Spice inqiries, except those for hardcopy documentation, should be directed to groups and persons affiliated with Spice. The Software Distribution Office now only facilitates the ordering of hardcopy documentation for this program.

For SPICE3F4 MS-DOS Users:
Call the Software Distribution Office to obtain the necessary instructions to upgrade your diskettes: (510) 643-6687.

SPICE is a general-purpose circuit simulator with several built-in semiconductor device models. SPICE performs several analyses, including nonlinear DC, nonlinear transient, and linear AC analysis. Device types include resistors, capacitors, inductors, mutual inductors, switches, linear and nonlinear sources, lossy and lossless transmission lines, BJTs, JFETs, GaAs MESFETS, and MOSFETs.

SPICE3 is based directly on SPICE2. SPICE3F includes AC and DC sensitivity analysis, which was not available in the previous major release (3E). Also available are an enhancement to the JFET model, a correction to the continuity of the MOS3 model (with respect to the kappa parameter), table-format output for device operating-point data (like SPICE2), the ability to change individual device parameters or vector elements interactively, support for X11r5, hardcopy for PC graphics, memory leak fix for nonlinear sources, and many other bug fixes. Finally, the portability and installation procedures have been improved. SPICE3F3 contains numerous bug fixes over the previous minor release (SPICE3F2), notably in the AC sensitivity analysis, initial conditions, and plotting.

Hardware/Operating System Requirements:

UNIX workstations:

- **Operating system: Ultrix 4.x, SunOS 4.x, HPUX 8.x, AIX 3.x**
- **Graphic output: X11r4 or X11r5, postscript, UNIX plot format; many graphics terminals are also supported by an internal library.**
- **Compiler: A robust C compiler required; SPICE3 is written in old style or K & R C, not ANSI C. GNU gcc has been used successfully with previous versions.**

PC:

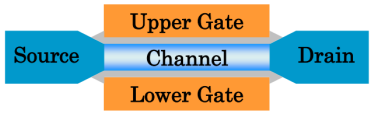
- **Operating system: MS-DOS (3.0 or later)**
- **Hardware: 286 or faster with a math coprocessor is required for reasonable performance.**
- **Graphic output: SVGA is not supported, hardcopies are for the "IBM personal computer graphics printer" or a compatible printer.**

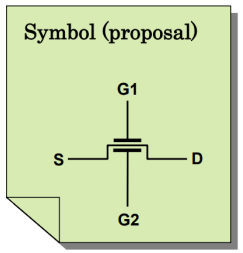
https://ptolemy.berkeley.edu/projects/embedded/

過去の経験１
XMOS(double gate MOS)モデルをspice3f5に組み込む@2004

DG MOSFET

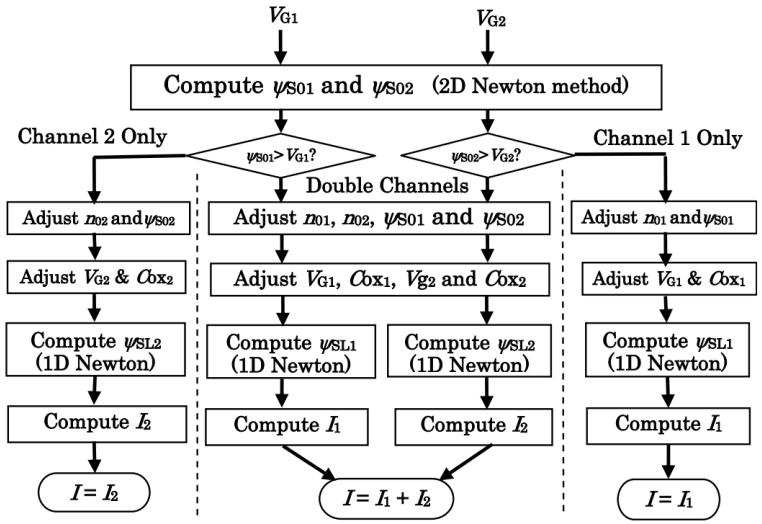
Two gates sandwich the Si channel.
The additional gate can be used as
-the gate electrically tied with the other, or
-another signal input, or
-an alternative for the body bias electrode.

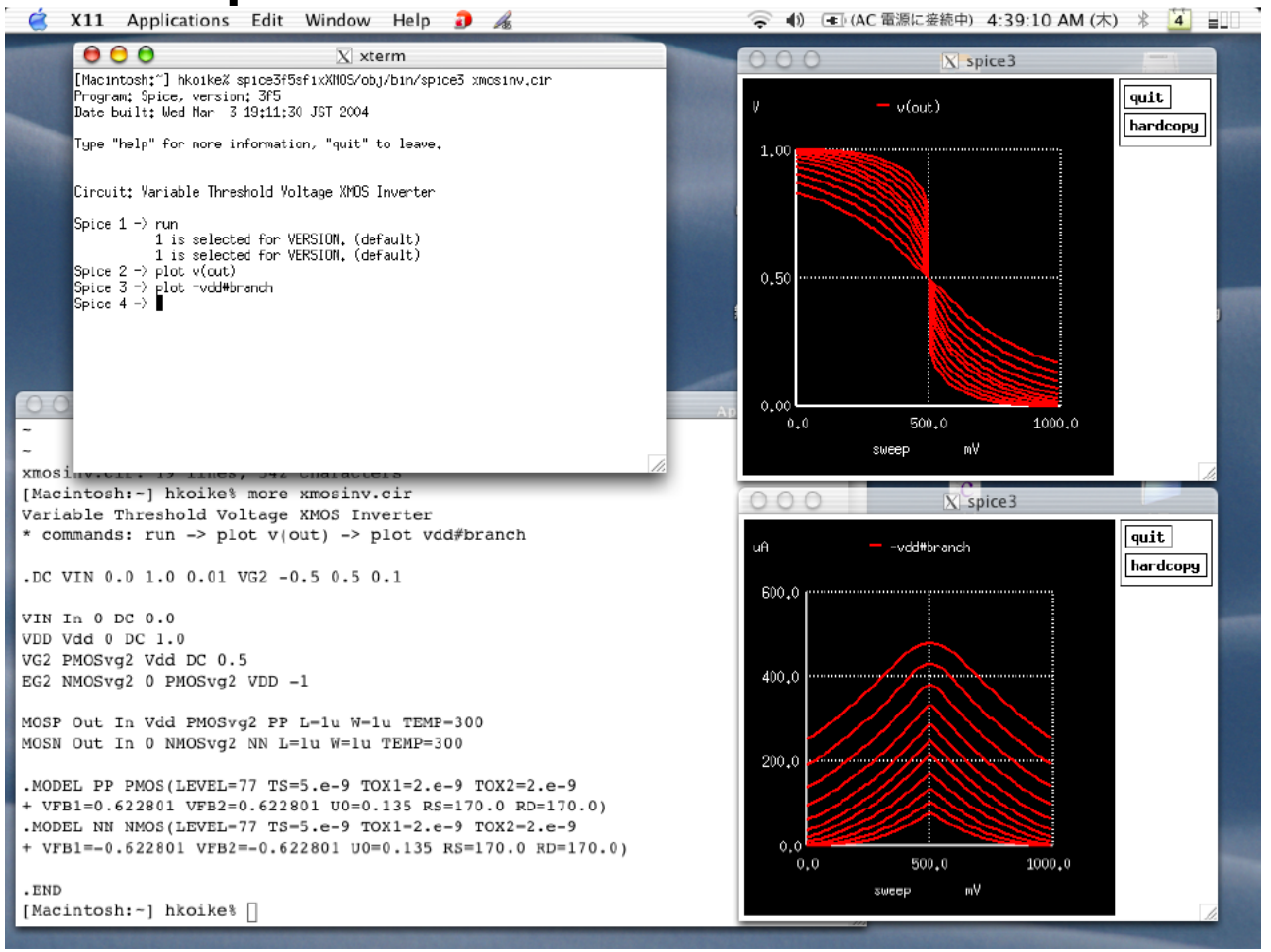




In all usage, minimum short-channel effect is distinct.
When used as electrically tied gates (as FinFETs),
high current drivability and ideal S factor are prominent.
When used as an alternative for the body bias electrode,
high frequency body bias, aggressive forward body bias
and other sophisticated body bias techniques can be easily achieved.
Predominant disadvantage is difficulty of manufacturing,
now overcome in research level, but
long way to go in LSI or production level.

AIST

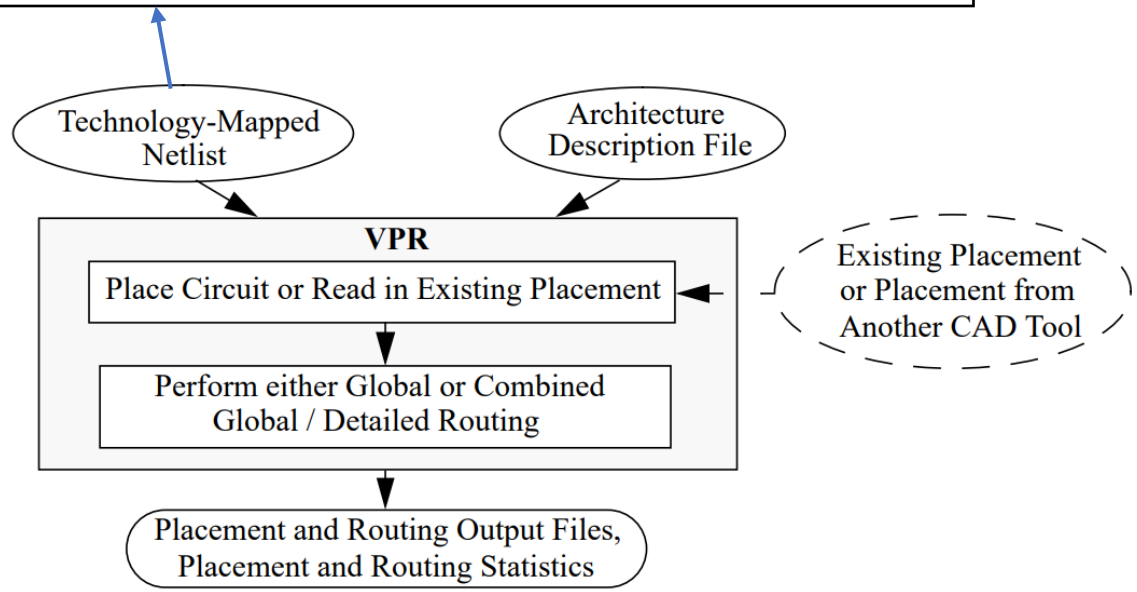




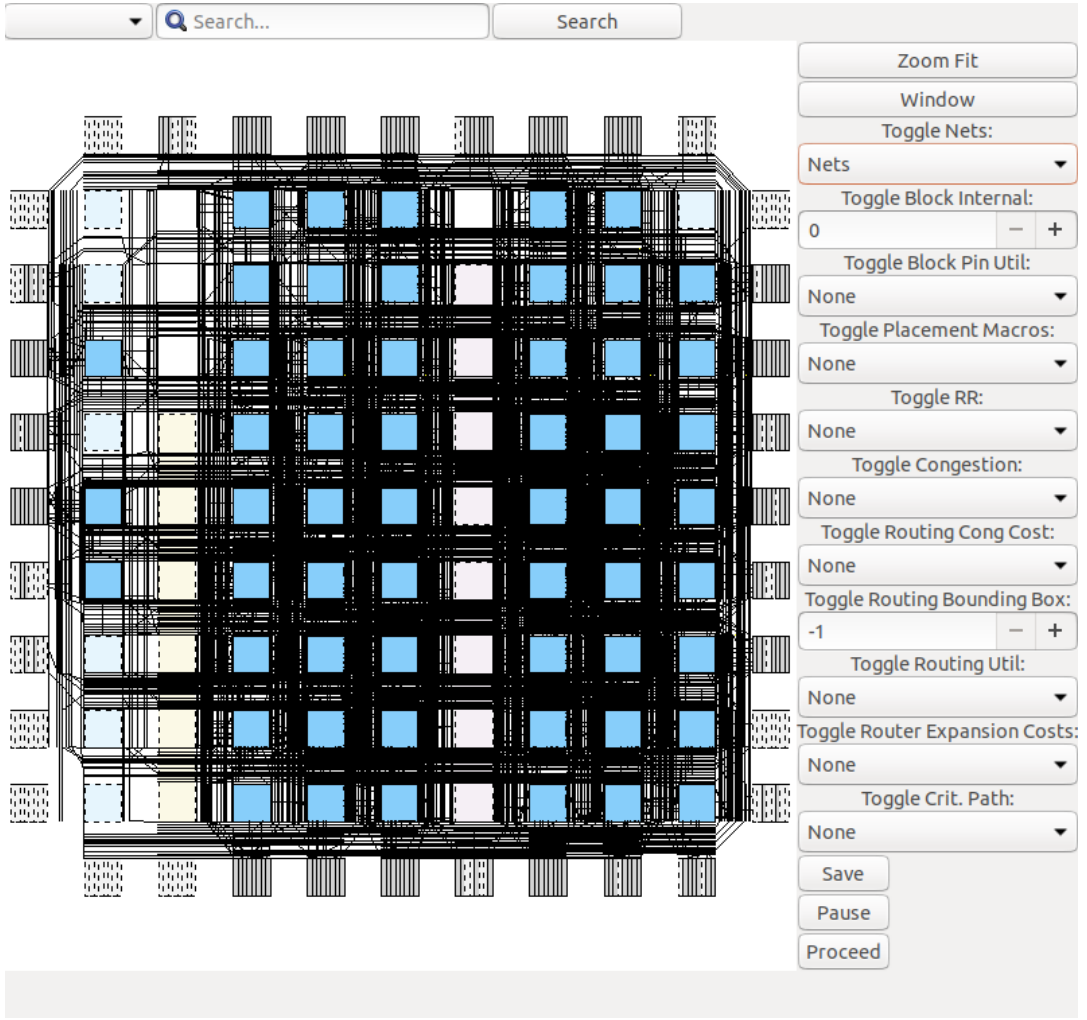
Ref. 中川他@AIST、nanotech2004

過去の経験２
VPR:Versatile Place and Route @ Toronto University（1997～2011？）

BLIF: Berkeley Logic Interchange Format



ビットストリームは吐かない

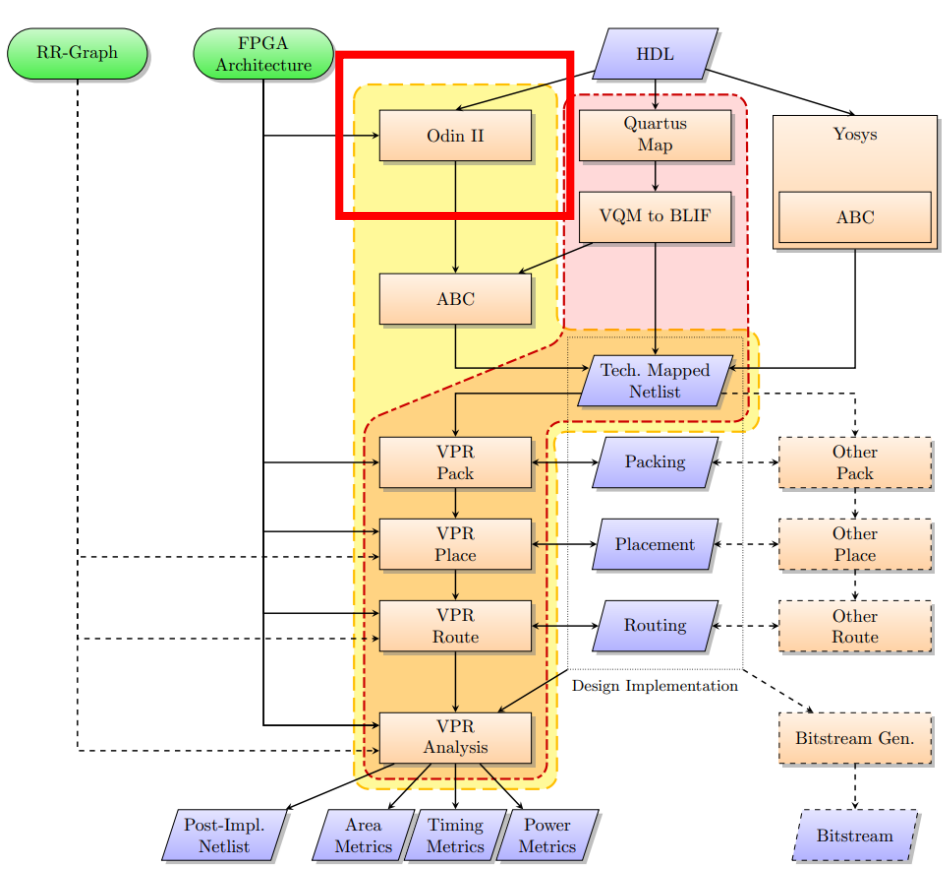


オープンソースEDAツールでアカデミアにおけるFPGA研究を加速

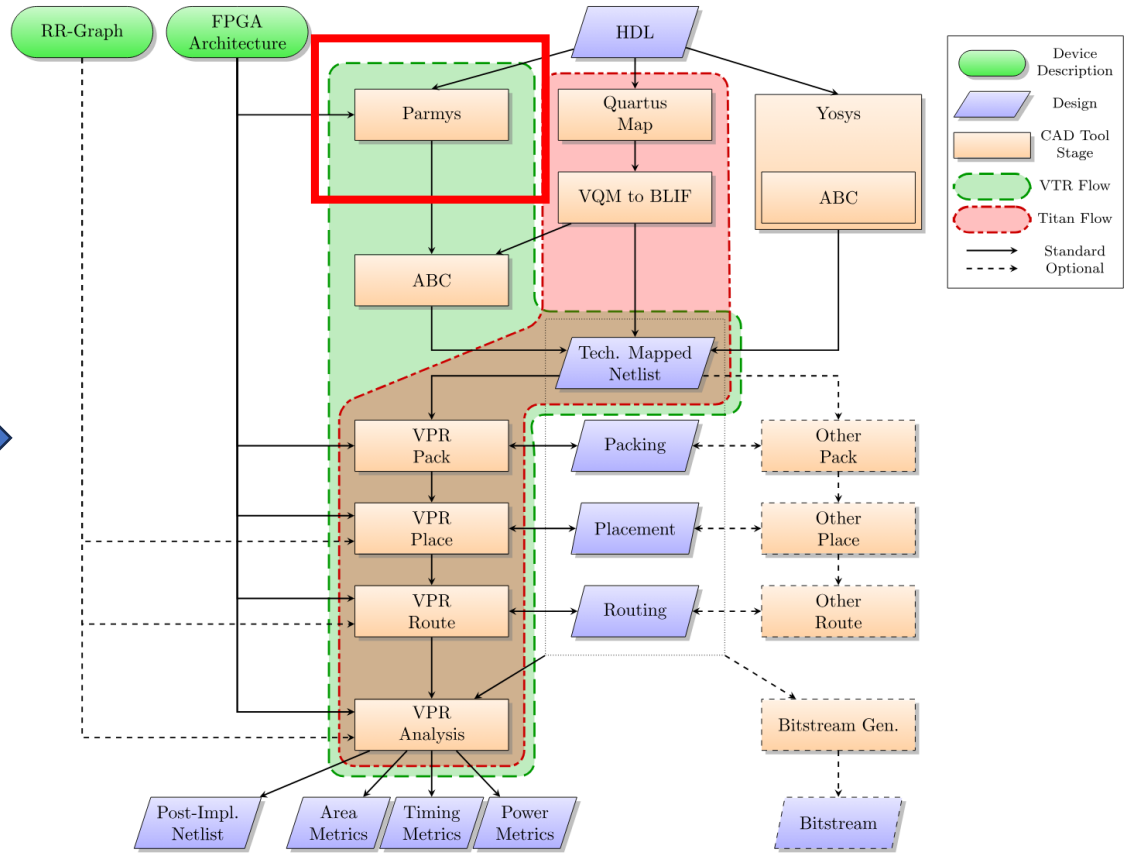
FPGA用CAD VPRの進歩
VTR: Verilog to Routing @ Toronto University（2012～）

Input: BLIF -> Verilog

Older version

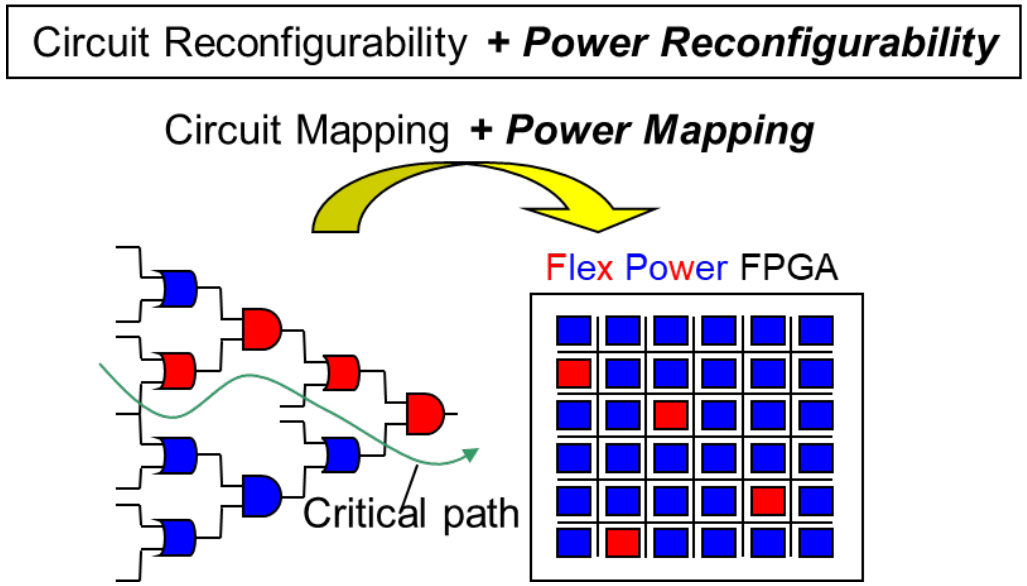


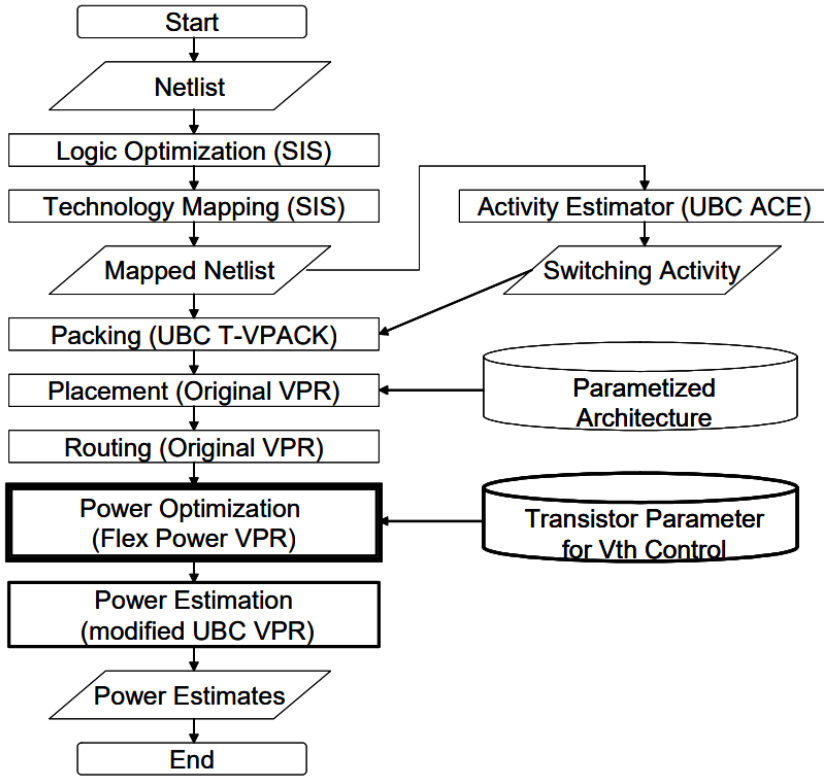
Latest version (VTR8)



過去の経験３
Flex Power FPGA用のCADフローを開発する
（Ref. 河並他@AIST、IEICE2004）

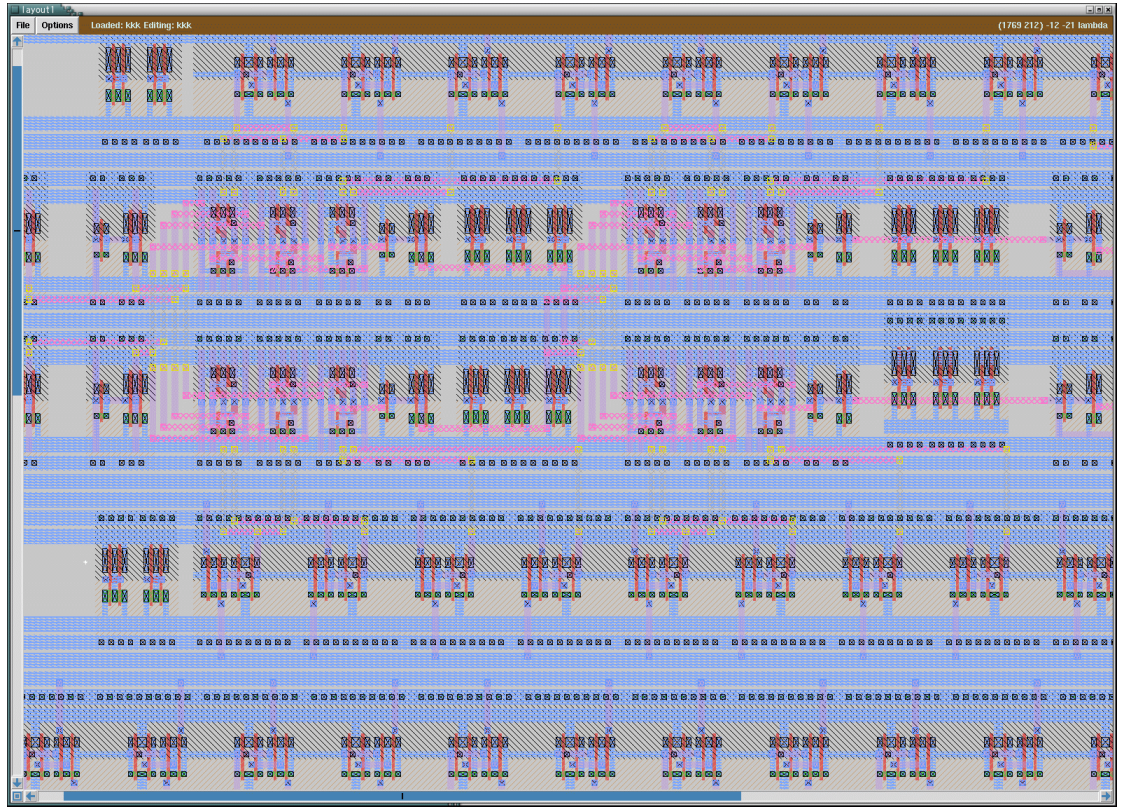
- 電力を再構成可能なFPGA：Flex Power FPGAの研究開発
- Toronto大のVPRにVT Mapperを追加
- タイミングスラックをパワーに変換



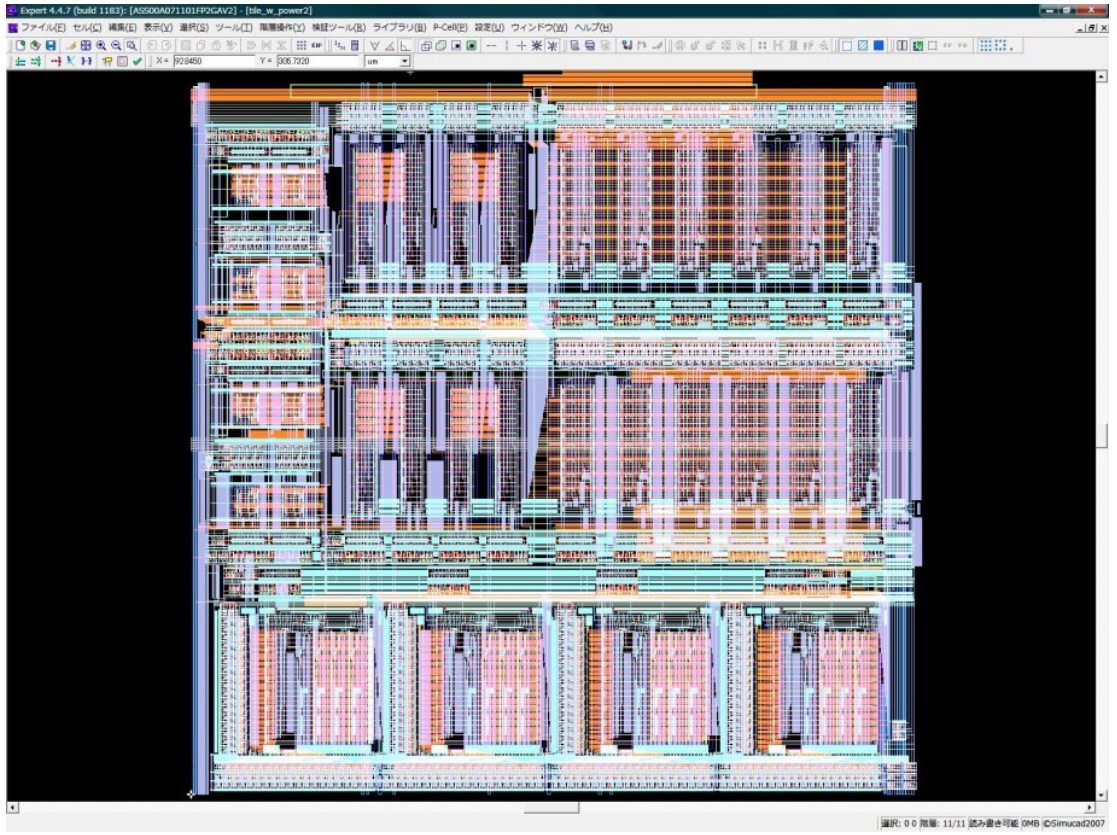


過去の経験４
オープンソースレイアウトエディタMagicを使って Flex Power FPGAレイアウトを予備評価した@2007

Magicを使った面積評価



商用レイアウトツールで設計



商用ツールへ移行
理由：ツール向けPDKがあった（TSMC90nm）

（オープンソースではないですが）
Flex Power FPGA CADの刷新

Flex Power FPGA CAD Flow

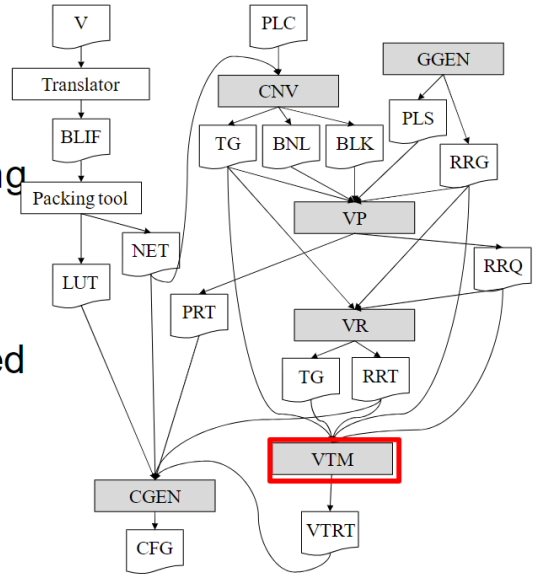
•Original Flex Power FPGA CAD tools are newly developed.

•VTM automatically maps Vt according to the slack analysis

•Successfully Formal-Verified using modest-size application such as microprocessor core

•Interfaces to the standard commercial CAD tools are included

•Why not VPR (4.3)?
- Unidirectional Wire
- 3D Extensibility
- e.t.c.

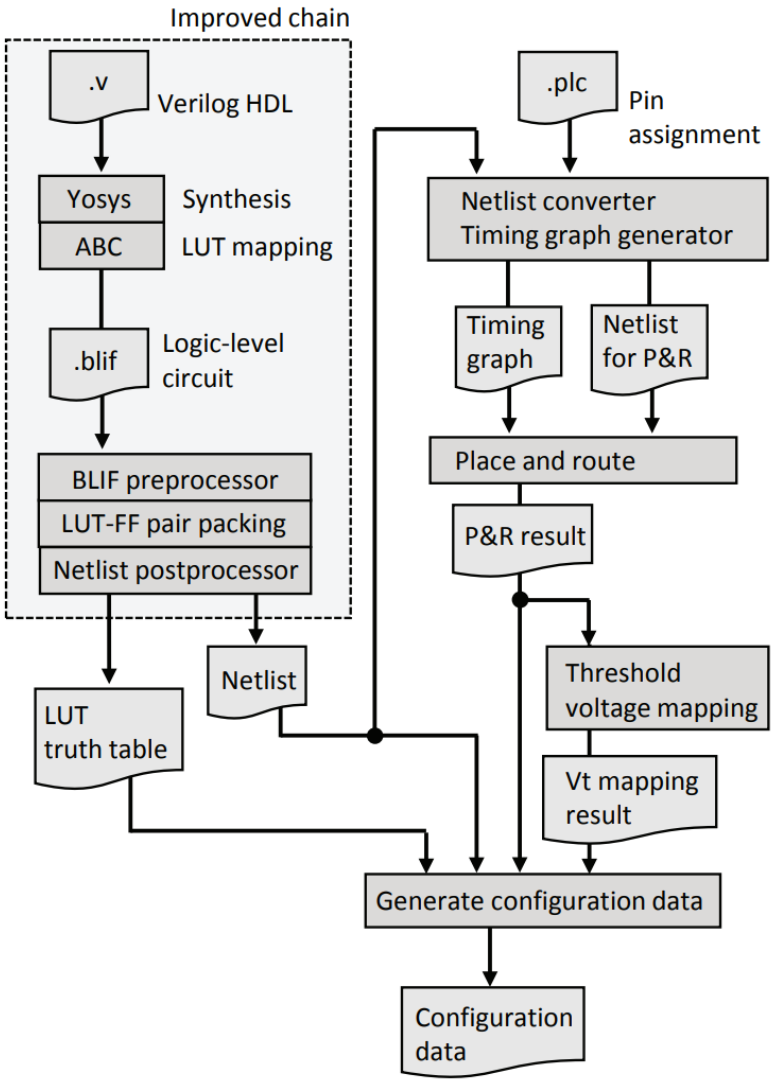


Software	
GGEN	RRG and PLS generator
CNV	NETto BNL transformer TG and BLK generator
VP	Placer
VR	Router
VTM	Vth mapper
CGEN	Config. bit stream generator

File	
V	Verilog RTL netlist
BLIF	Gate level netlist
NET	CLB netlist
LUT	LUT truth table
PLC	Pad location
TG	Timing graph
BNL	VP inputtable netlist
BLK	Fix block location
PLS	Function block location
RRG	Routing resource graph
RRQ	Routing resource request
PRT	Placement result
RRT	Routing result
VTRT	Vt map result
CFG	Config. bit stream

- Flex Power FPGAアーキテクチャの探索を可能とするCADを新たに開発
- Verilog to BLIFはAlteraのQuartusを利用
- ビットストリームまで出力可能
- 実チップが完全動作

（オープンソースではないですが）
Flex Power FPGA CADの改良（Ref. 片下他、IEICE論文誌、2017）



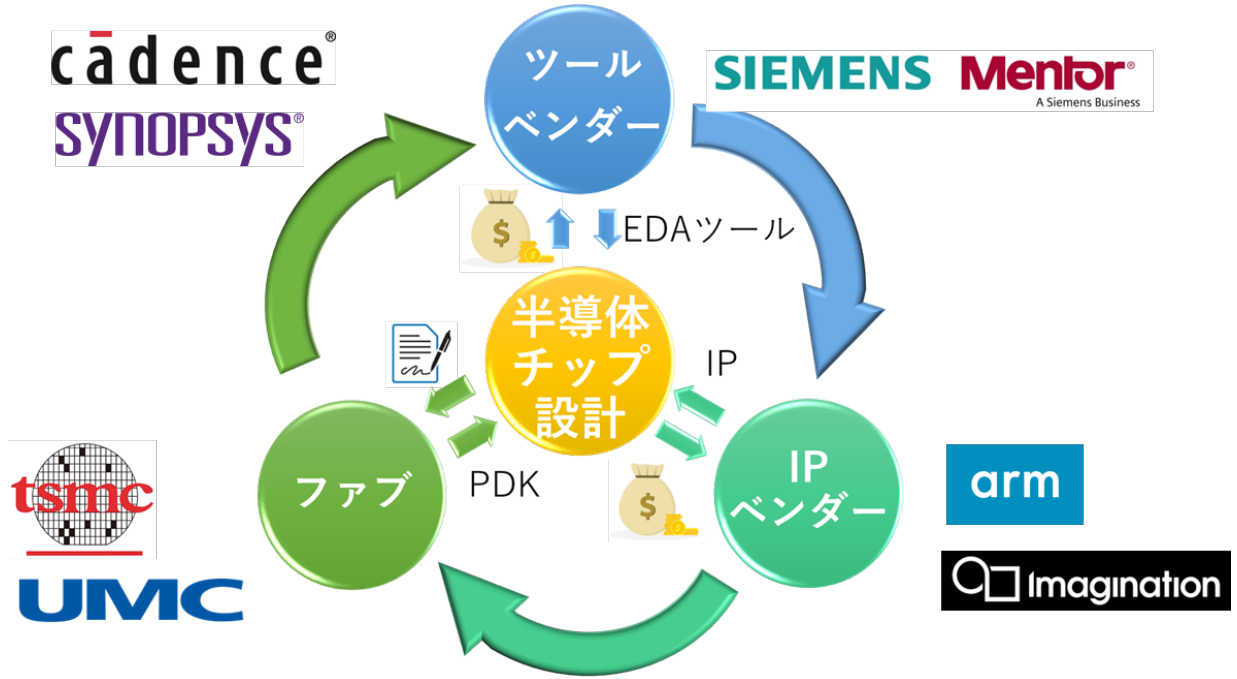
- Verilog to BLIFにオープンソースツールを利用
- 複数個のファイルの入力に対応
- モジュール記述ファイルの生成とLUTマッピング情報を生成するためのポストプロセッサ

（再掲）LSI開発のエコシステム

✓現在のLSI開発：既存エコシステムの束縛⇒**新規参入者にとって高い参入障壁**

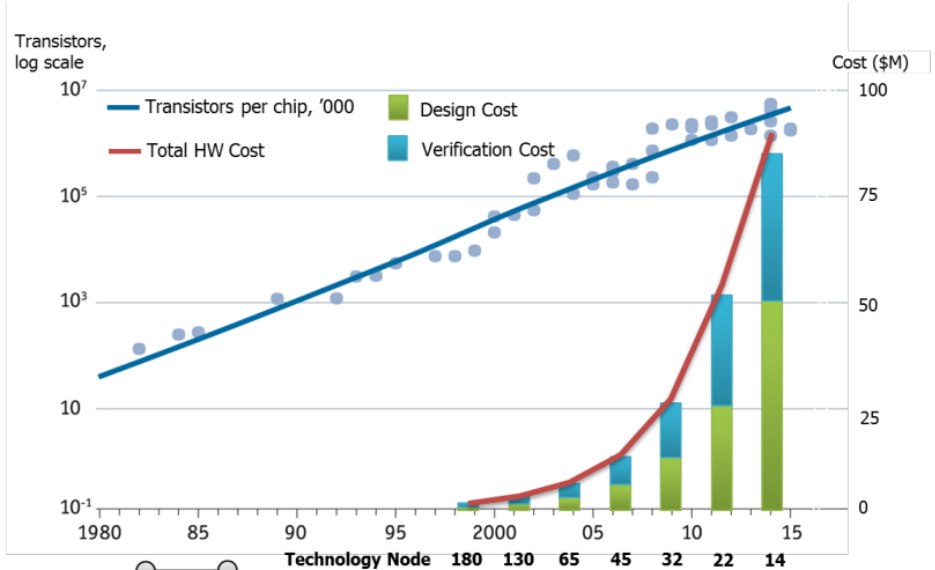
- 高価な**EDAツール**（設計ソフトウェア）
- 高価な**IP**（設計資産：インターフェース、メモリ等）
- 入手に手間のかかる**PDK**（Process Design Kit：デザイン情報）

⇒**ハードウェアイノベーションのバリアを下げるための試み@USA**



ERI: Electronics Resurgence Initiative, 2017
IDEA & POSH Program

Lowering barriers to hardware innovation



New procedures for physical design and verification will lower the design barrier, enabling rapid specialization

Design

Intelligent Design of Electronic Assets (IDEA)

- No human in the loop" 24-hour layout generation for mixed signal integrated circuits, systems-in-package, and printed circuit boards.Machine generated layout of electrical circuits and systems

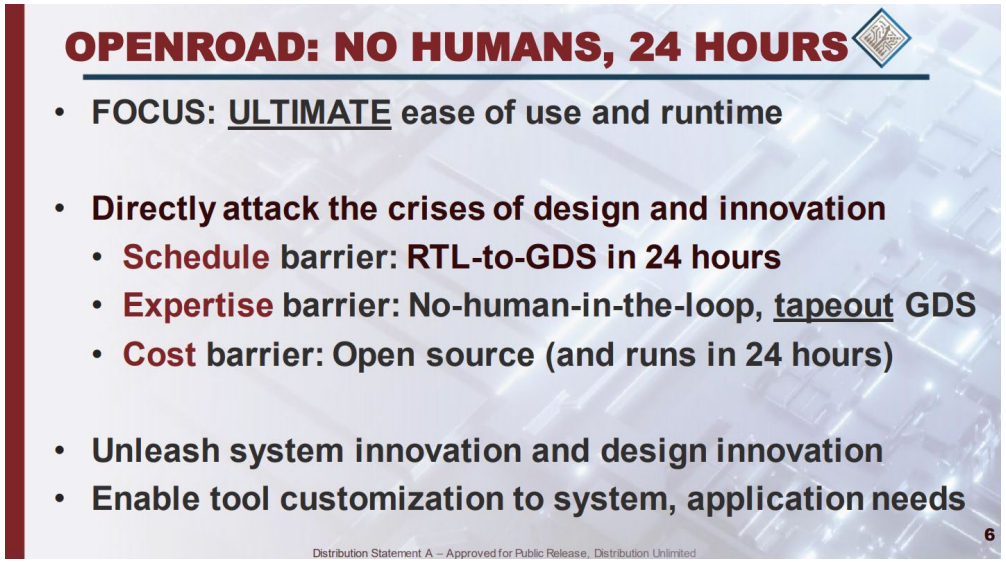
Verification

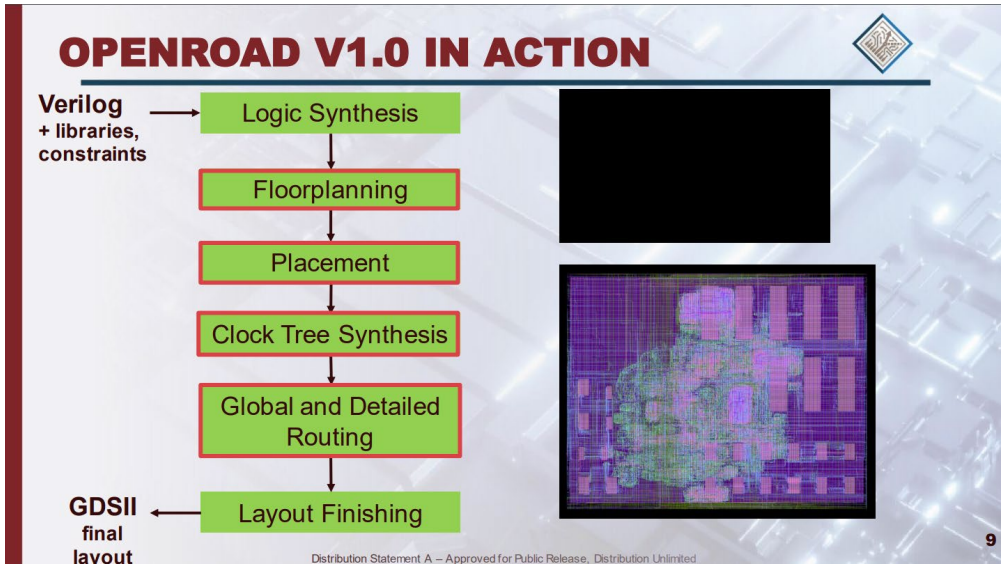
Posh Open Source Hardware (POSH)

- An open source System on Chip (SoC) design and verification eco-system that enables cost effective design of ultra-complex SoCs.

The 1980's DARPA MOSIS effort removed fab cost and fab access barriers and launched the fabless industry. The ERI Design effort will address today's design complexity and cost barriers, creating the environment needed for the next wave of US semiconductor innovation.

IDEA：OpenROADプロジェクト（2018～2023）

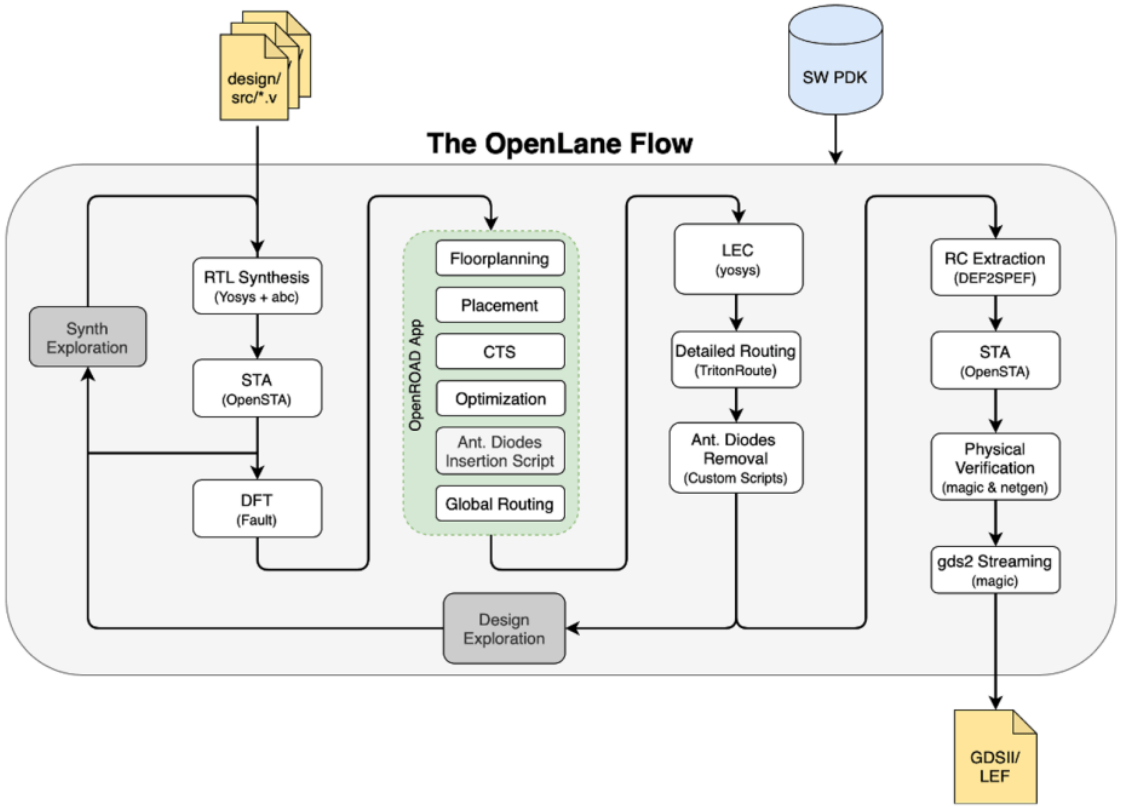




- PL: Prof. ANDREW B. KAHNG@UCSD
- DARPA ERI(Electronic Resurgence Initiative) Summit 2018
  - 24-hours, Non-Human-In-Loop layout design for SOC, Package and PCB with no Power-Performance-Area (PPA) loss, Tapeout-capable tools in source code form -> Open Source
- Funded by DARPA 2019-2023, Now supported by Precision Innovations
- Used in research and commercial apps.
  - OpenROAD-flow-scripts(OpenROAD)
  - OpenLane(Efabless)
  - Silicon Compiler (ZeroASIC)
  - Hammer (UCB)
  - OpenFASoC (IEDA-FASoC)
- PDK Support
  - Open Source: GF180nm, SKY130nm, FreePDK45nm, ASAP7nm
  - Proprietary: GF55nm, GF12nm, Intel22nm, Intel16nm, TSMC65nm

Ref. Kahng@ERI Summit 2020

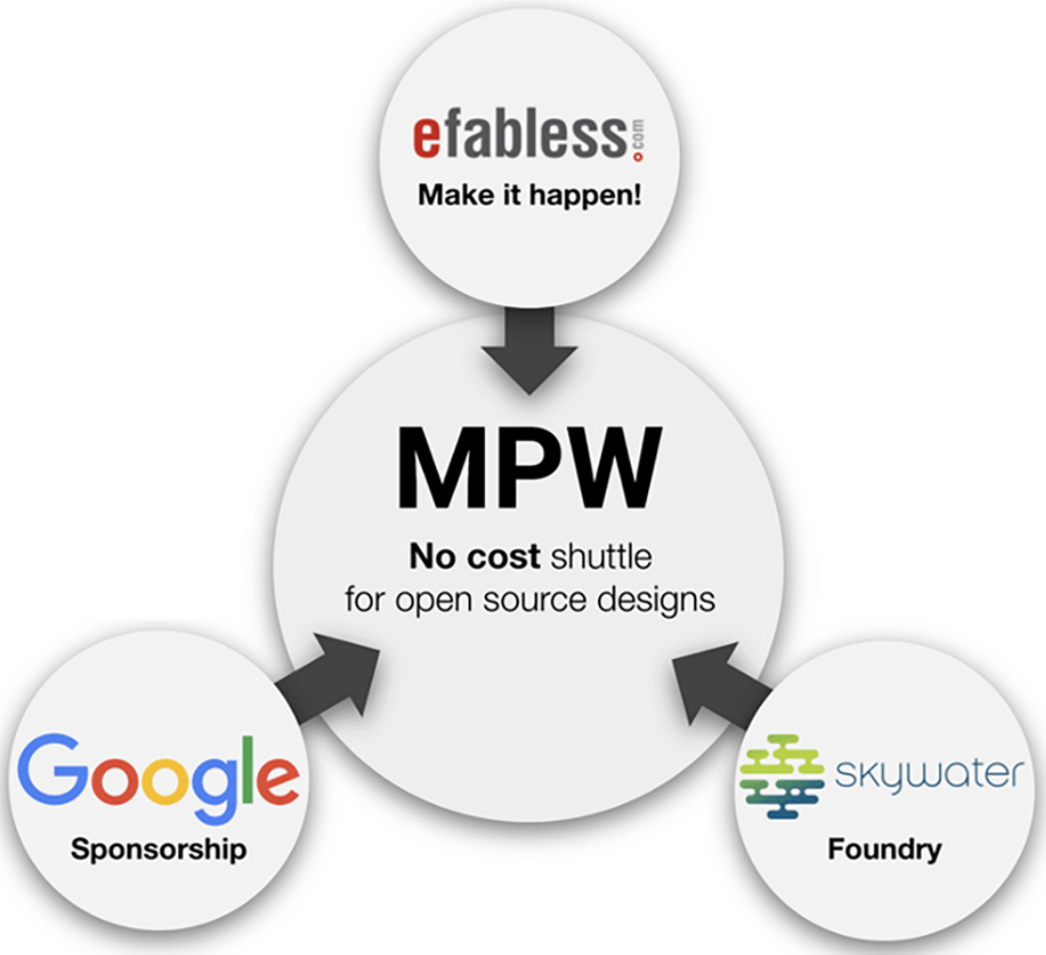
OpenLaneプロジェクト（2020～）



- A 130nm OpenROAD-based Tapeout-Proven Flow (ICCAD 2020)@efabless
- OpenLane ＝OpenROAD + Yosys, Magic, Netgen, CVC, Klayout and custom scripts
- PDK Support
  - Open Source: SKY130nm, GF180nm

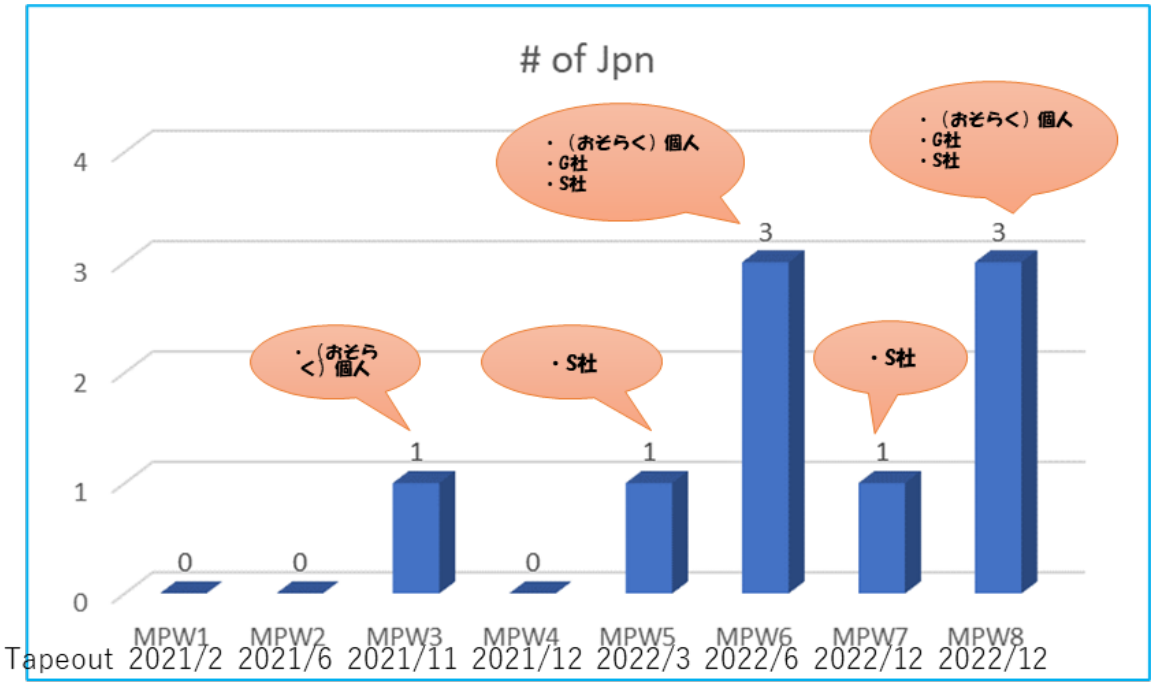
Open MPW Shuttle Program （2020～）

- Google-efabless-SkyWater Open MPW(:Multi Project Wafer) Shuttle Program from 2020
- 無償のシャトルプログラム
- 有償シャトルプログラムあり（ChipIgnite@efabless）
- 設計IPは公開
- efabless: Open Source EDA 環境(OpenLane)
- SkyWater: 130nmプロセスファンドリサービス
- Google: スポンサー、オープンソースPDK管理、高位合成ツールチェーン（XLS）開発など
- Globalfoundriesが2022年に参画



国内の利用状況

- ✓2020年よりGoogle社, skywater社, efabless社がオープンソースEDA環境・PDK/IPを開発し、無償シャトルサービス（情報公開が原則）を開始。全世界で**600件以上**の利用実績。
- ✓2022年よりGlobalFoundriesが参画
- ✓日本国内からの利用例は少なかった（**10件未満**）
- ✓**設計に係る情報や知見が足りない？**



オープンソースEDA環境・試作サービスの国内の利用件数

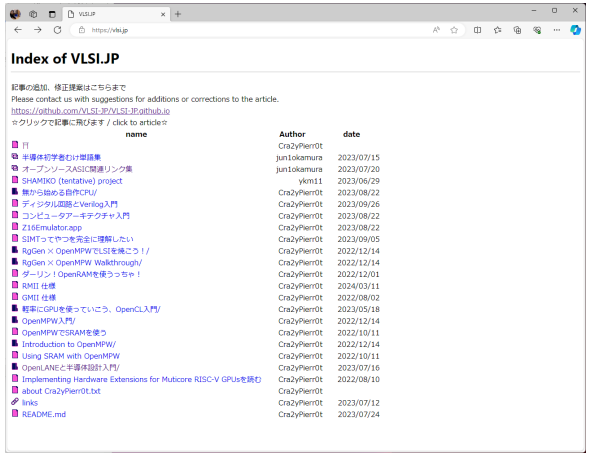
直近のGFMPW-1の日本エントリ：10件⇒TOフェーズ：6件！！

GFMPW-1 Open MPW Shuttle Service

prova	N/A	Gian Domenico Licciardo	Italy	No	N/A	N/A	0
rvcore_chip2	Digital	Kenji Kise	Japan	No	Pass	Pass	1
test	Digital	Miho Yamada	Japan	No	Fail	N/A	2
omotya	Digital	Riku Anan	Japan	Yes	Pass	Pass	3
xls-group-tapeout	Digital	Johan Euphrosine	Japan	No	Fail	N/A	1
micro_irritating_maze	Digital	Noritsuna Imamura	Japan	Yes	Pass	Pass	29
ISHI-KAI_Multiple_Projects_OpenGFMPW-1	Digital	KAI ISHI	Japan	No	Pass	Pass	9
my_gf180	Digital	Riku Anan	Japan	Yes	Pass	Pass	3
JSpi	Digital	Herbert T	Japan	No	Pass	N/A	5
ishikai-gds-test-homelith	Digital	Toru Homemoto	Japan	No	Pass	N/A	4
gitefu_check	Digital	Gitefu	Japan	No	Pass	Pass	1
Ophelia_eFPGA_rerun	Digital	Egor Lukyanchenko	Kazakhstan	Yes	Fail	Pass	2

国内コミュニティ

- ISHI会(website,discord)
- Open Source EDA supporters(discord)
- VLSI.jp(website)
- 個人ブログの記事
- AIST Solutions　AI・半導体プロデュース事業部(slack)



AIST所内プロジェクトの立ち上げ（2023/6～）

産総研
ともに挑む。つぎを創る。

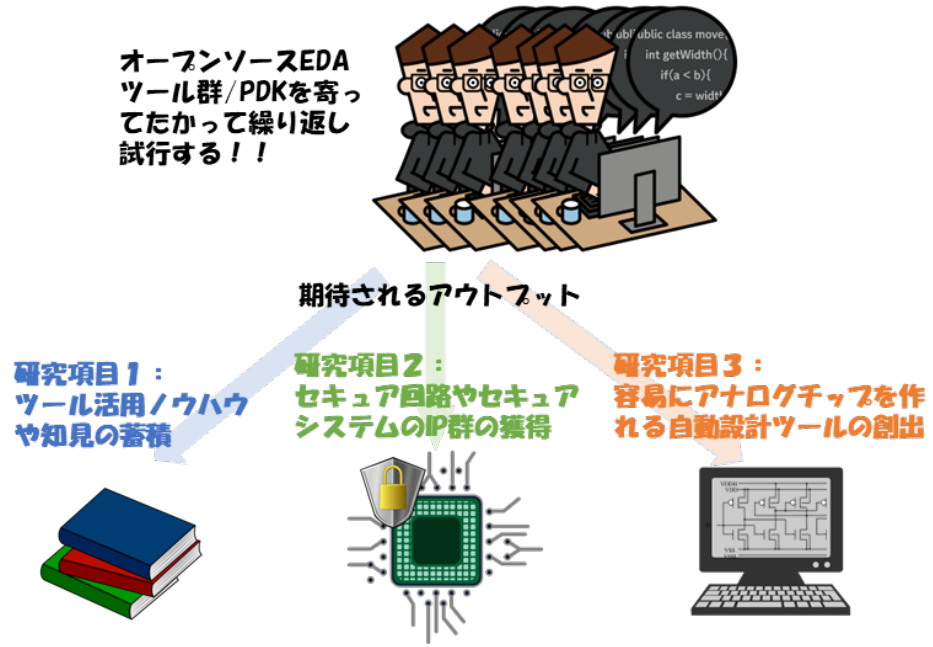
目的：国内のオープンソースEDA/PDKの利用普及に向けた開拓を産総研 がも 担う

研究項目１．オープンソースEDAの利用手法の確立とチップ作成
- オープンソースEDAツールであるOpenLaneの利用環境構築と利用技術の確立
- efabless社の商用シャトル等を用いたLSIチップ試作・機能検証・評価

研究項目２．チップセキュリティ技術を題材としたオリジナルチップ作成の試行
- 開発実績のある暗号・符号回路IPと新規開発の簡易RISC-Vコア・PUF回路を組み合わせたチップの作製・評価
- オープンソースEDA を用いたオリジナルセキュリティSoCの開発

研究項目３．オープンEDA/PDKを活用したアナログ回路自動設計ツールの開発
- 知識のない人でも簡単にアナログ回路を作成できる自動設計ツール開発

✓ 所内LSI研究者の英知を結集
✓ 産総研発回路IP※をフル活用



※代表的な産総研の回路IP
- PUF回路（NEDOプロ@2017-2021）
- 暗号回路（科研費等）
- モーションセンサー（Sensors 2021, Q1）
- 量子ビット読み出し回路（VLSI2022、プレス発表）
- 磁気センサー回路（ISSCC2022、プレス発表）

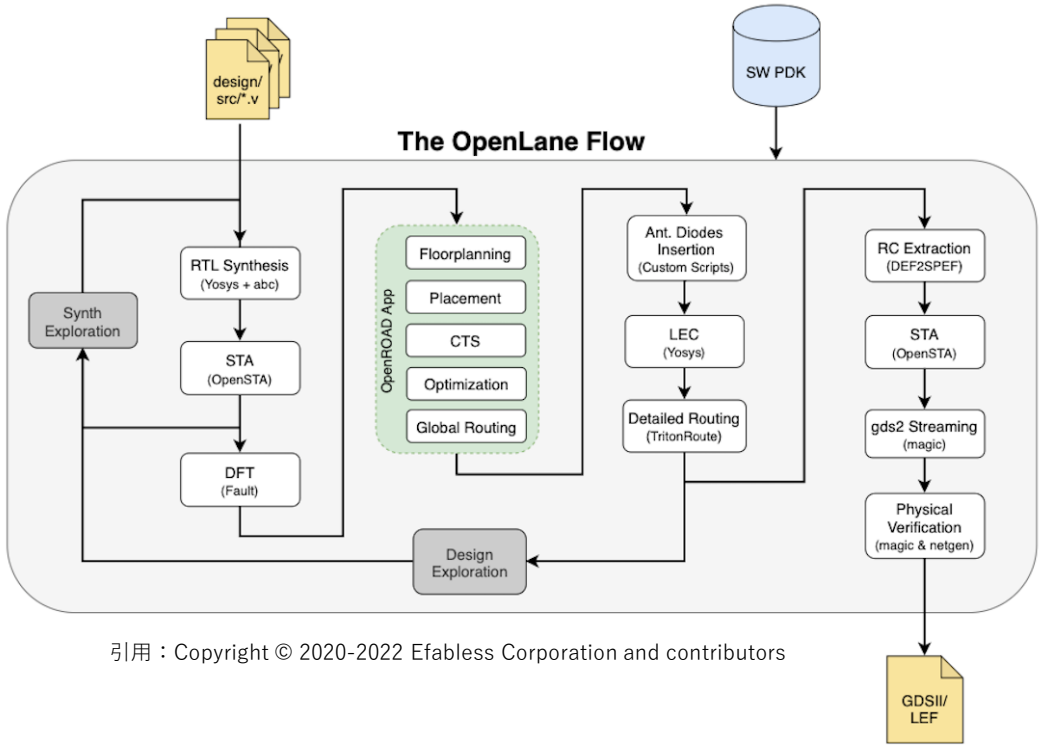
研究項目①：オープンソースEDAの利用手法の確立とチップ作成

- **目的**

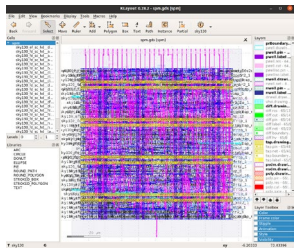
  オープンソースEDAの自動化ツール**OpenLane**の利用環境を構築し、RTL設計からGDSIIデータ出力までの**利用技術を確立**する。
  商用シャトル等を用いて**実際にLSIチップを作製**し、その機能検証・評価を行う。

- **2023年度実施内容と目標**

  OpenLane環境構築と利用ノウハウの蓄積、およびGDS-II等の**チップデータの作成**を実施する。（例：シリパラ変換回路など）
  ツールに付属のチップ制御回路 **Caravel と自作HDLコードを結合**し、**シミュレーションによる動作確認**を実施する。



引用：Copyright © 2020-2022 Efabless Corporation and contributors



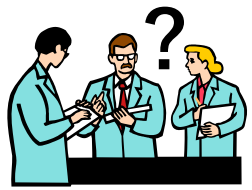
HDLから生成した
チップ作製データの例

他のHDLからチップ作製
データの生成方法は？

チップ作製前の検証方法は？

配置指定や制約設定のほか
他のPDK使用など可能か？

生成したデータにより実際
にチップを作製できるか？



オープンソースEDAの利用技術の蓄積

研究項目②：チップセキュリティ技術を題材としたオリジナルチップ作製の試行

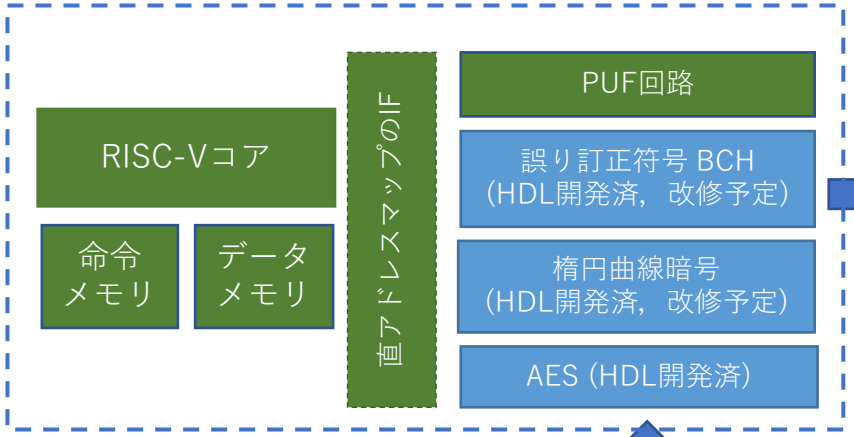
- **目的**

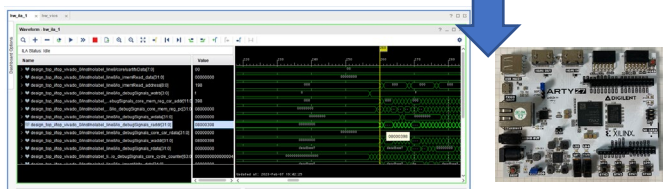
  開発実績のある**暗号・符号回路IP**と新規開発の簡易**RISC-Vコア・PUF回路**を組み合わせたチップの作製と評価を行う。

  真贋判定や認証等のためのオリジナルの**セキュリティSoCをオープンソースEDA を用いて開発**する。

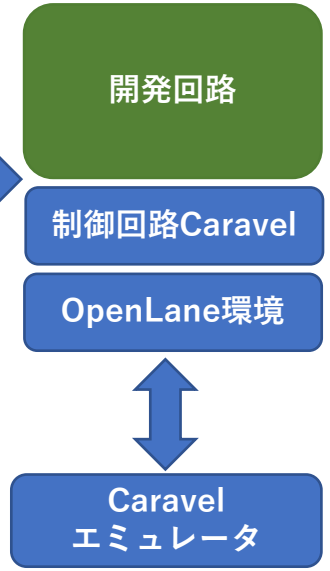
- **2023年度実施内容と目標**

  SoC制御とセキュリティ機能の実現のため、**新規RISC-Vプロセッサ**と**PUF回路の設計・開発**を実施する。

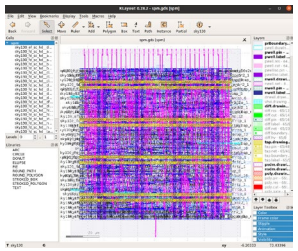




シミュレータやFPGAにより、
チップ作製前の検証を実施



Caravelエミュレータによる
作製チップの動作検証



チップ作製データの生成

オープンソースEDAによるシステムチップ作製技術の蓄積

- 認証システムチップの機能検証・性能評価
- Caravelと自作コアを組み合わせたシステム評価
- 自動化されたフローでのRAM・ROMのテクノロジマッピング手法の調査
- オープンソースEDA環境のシステムLSI設計の有用性評価

研究項目③：オープンEDA/PDKを活用したアナログ回路自動設計ツールの開発

- **目的**

  オープンソースEDAは、アナログ回路設計環境の整備が不十分。また、**アナログ回路設計**は、集積回路・デバイスに関する知識や、レイアウトに関するノウハウなどが必要で、**デジタル回路設計に比べて参入障壁が高い**。

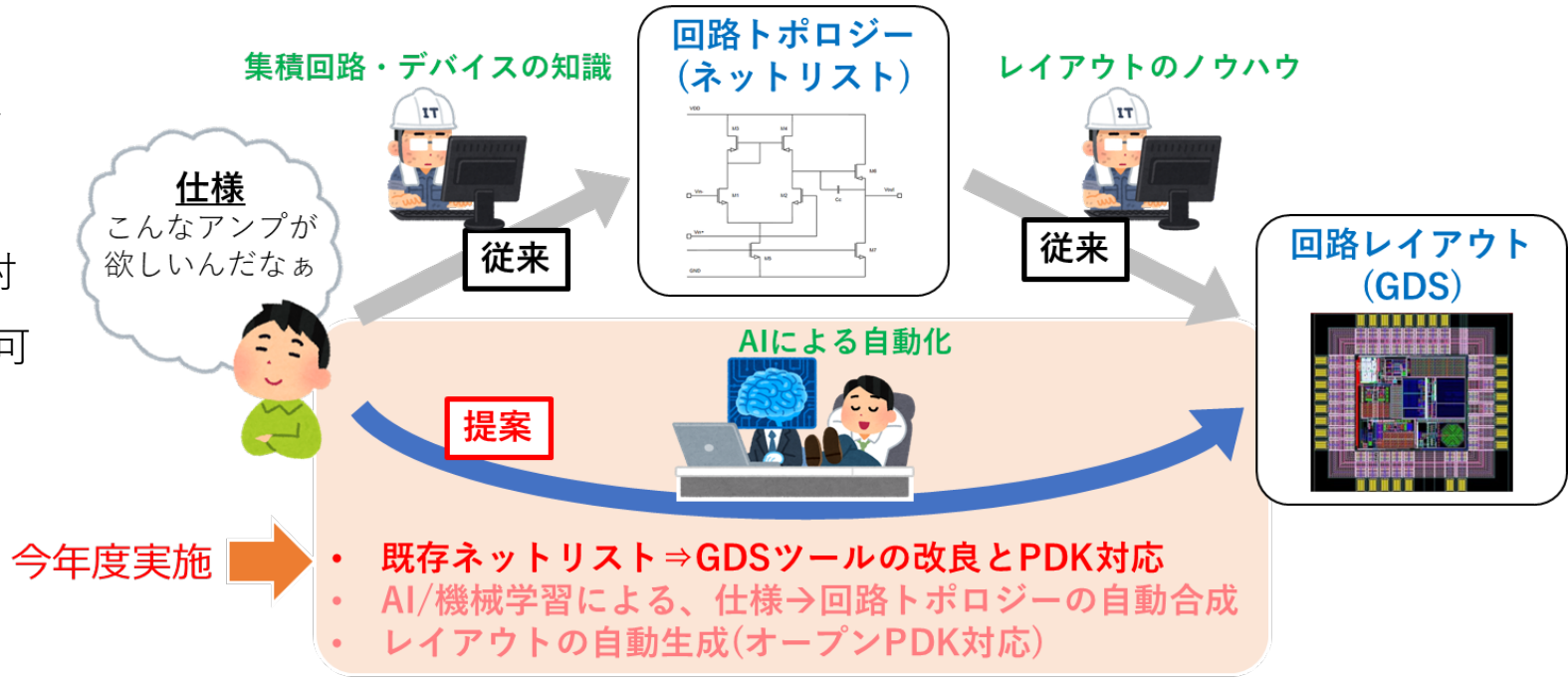
  オープンPDKやAI/機械学習を活用して、**知識のない人でも簡単にアナログ回路を作成できる自動設計ツールを開発**する。
  最終ゴールはアナログ回路の仕様を与えることで、それに相当したGDSファイルを生成するツールの開発を目指す。

- **2023年度実施内容・目標**

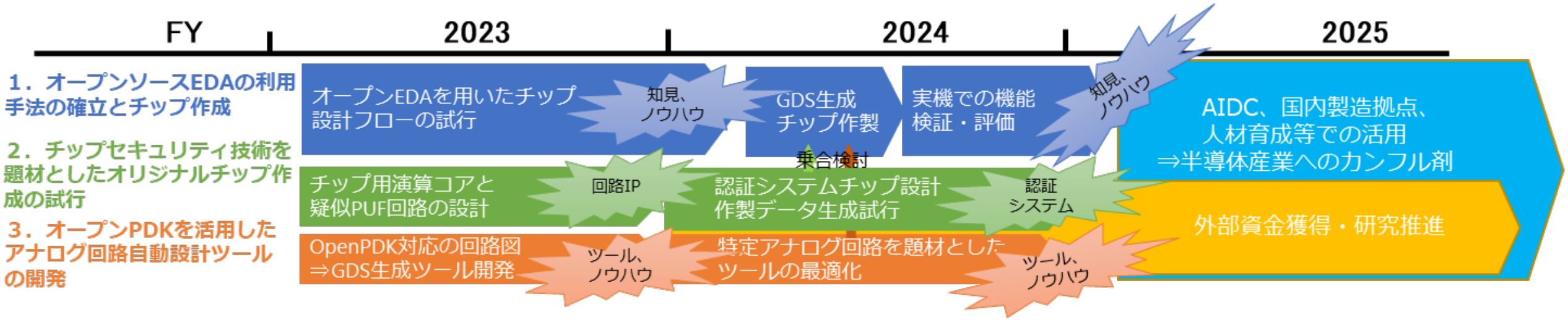
既存のネットリスト→GDS生成ツールを改良して、オープンPDKに対応させる。

- ✓ツールの実装詳細調査
- ✓ツールを修正するか、PDKを変換するか検討

生成されるGDSが、デザインルールを満たし製造可能であることを確認する。



本研究のロードマップ



✓ 期待されるアウトプット

- ツール活用ノウハウや知見の蓄積
- セキュア回路・システムのIP
- アナログ自動設計ツール

- **国内半導体の活性化**
- **人材育成**

✓ 本研究成果の活用

- AIDC等での情報発信
- 国内製造拠点の活性化
- LSI人材育成等

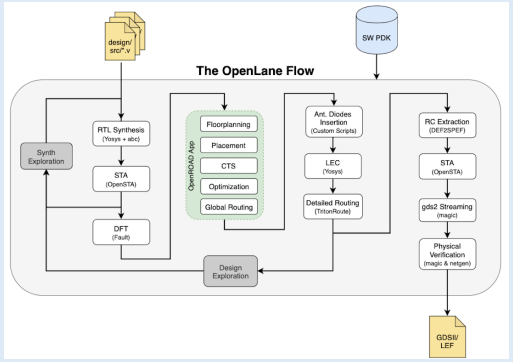
- **SCR用PDK開発**
- **設計フロー構築**

2023年度の進捗：利用環境構築と利用方法の知見の蓄積

Summary

- オープンソースEDAであるOpenLane Flowの環境とシャトルテープアウト用のCaravel統合環境を試行。両環境を構築とフローの実行を完了
- 上記で得られた知見を逐一ドキュメント化

OpenLane flow



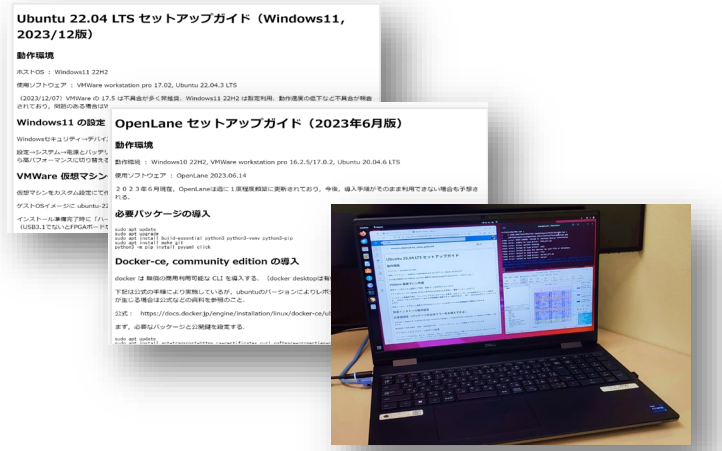
概要

- 複数のプログラムでできたツールチェーン
- 回路記述（RTL）からレイアウト（GDS）までを一気通貫で生成するフロー

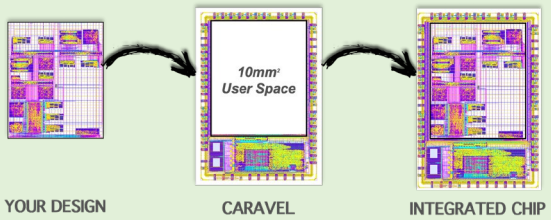
本家マニュアルに未記載の事項

- フローのアップデート方法
- 階層設計パラメータの設定
- 書式チェックの制限

など



Caravel統合環境



概要

- SkywaterとGlobalfoundriesのシャトルにテープアウトできるプログラムとチップフレームマクロのセット

本家マニュアルに未記載の事項

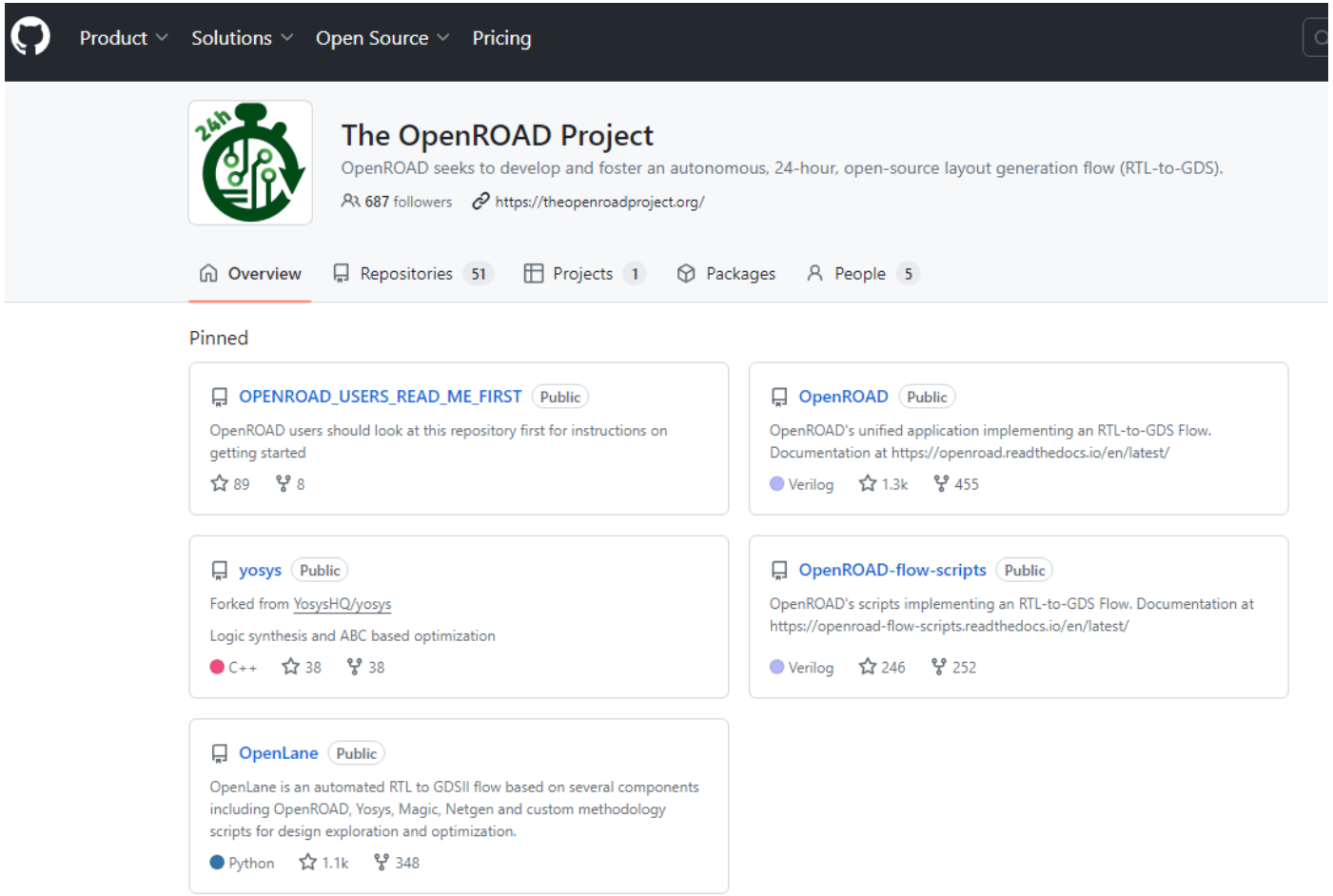
- READMEファイルの変更
- チップIO初期値の再設定
- 周辺回路IPやメモリIPの使い方

など

成果物

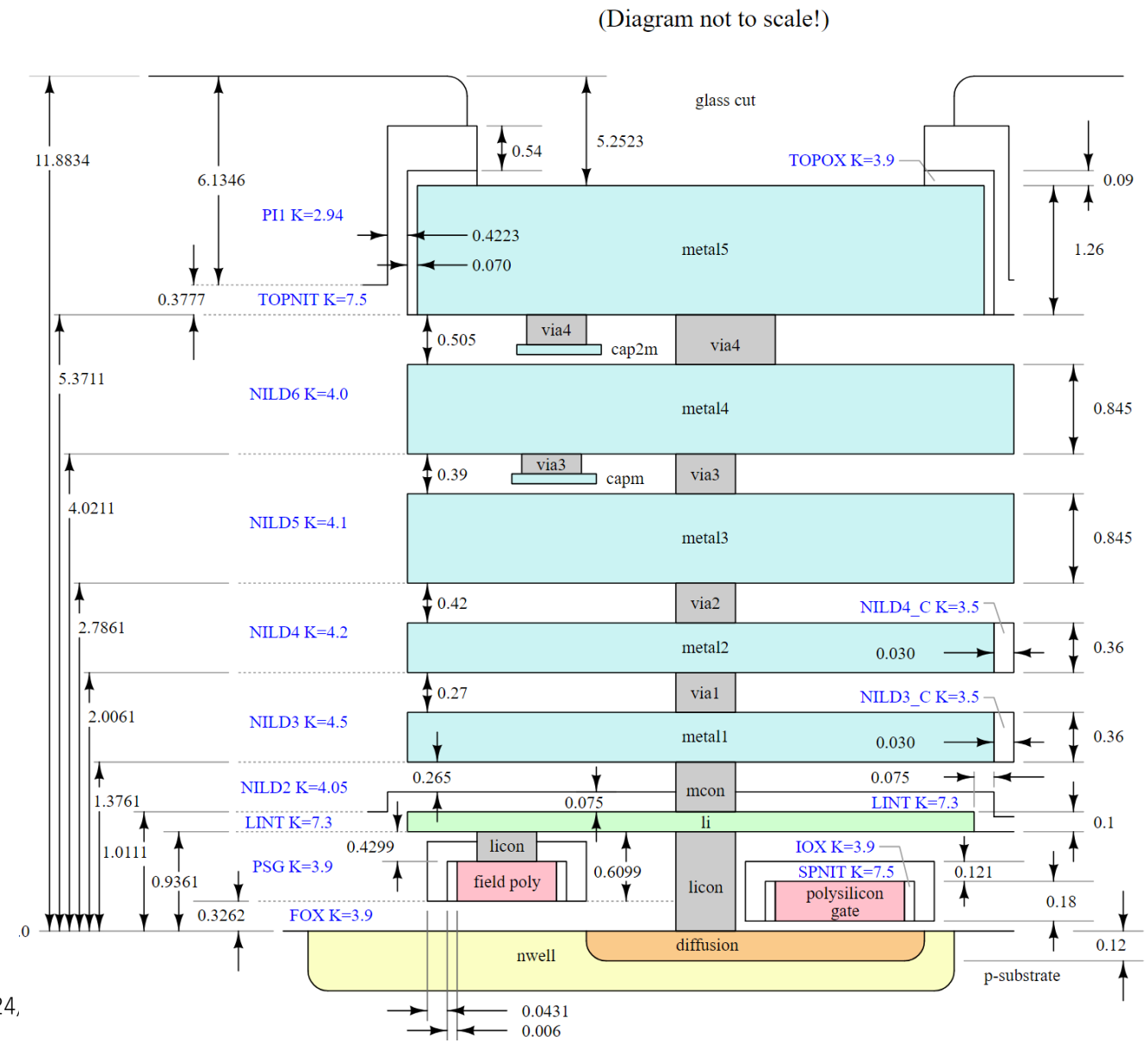
- OpenLane環境とCaravel環境の動作確認済み仮想イメージ
- 利用ドキュメント
  - Linux仮想環境構築
  - OpenLane導入と利用
  - Caravel導入と利用
  - Caravelと自作回路の結合とシミュレーション実行

OpenLaneの導入



- OS
  - Ubuntu 20.04+
  - macOS 11+
  - Windows10, 11+
  - Other Linux
- Download and Installation
  - Get python3.6
  - Get docker
  - git clone openlane
  - cd Openlane
  - Make （PDKも入る）
  - make test（インストールテスト）
  - make mount （OpenLane環境の立ち上げ）
- バーチャルマシン上でも動く（ex. Ubuntu on Windows）：可搬性

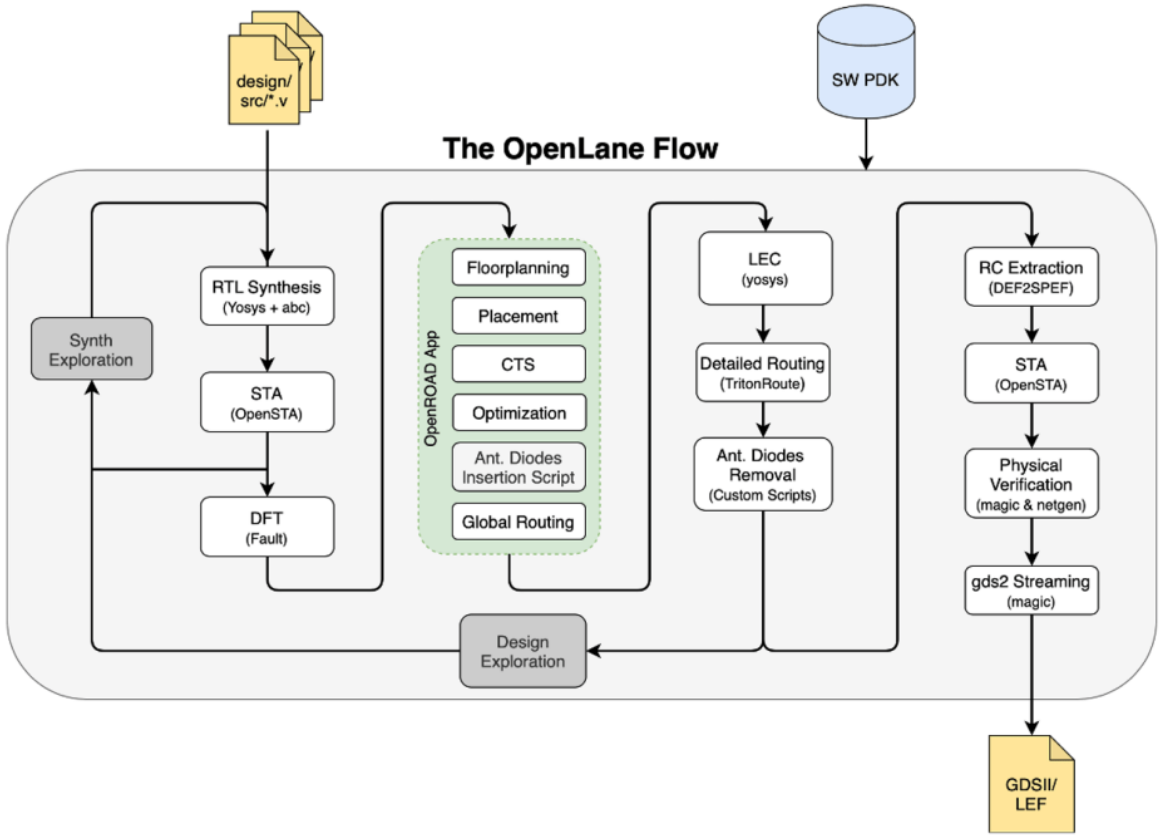
SkyWater 130nmプロセス



- 5 layer Al
- 1 local interconnect
- Poly gate
- Internal: 1.8V, I/O: 5V
- Inductor capable
- Poly resistor
- MIM Cap.

[GitHub - google/skywater-pdk: Open source process design kit for usage with SkyWater Technology Foundry's 130nm node.]

OpenLane Design Stage



LEC: Logic Equivalence Check
SPEF: Standard Parasitic Exchange Format

1. (Lint)
   - Verilator
2. Synthesis
   - Yosys/abc
   - OpenSTA
3. Floorplanning
   - Init_fp
   - ioplacer
   - Pdngen
   - tapcell
4. Placement
   - RePLace
   - Resizer
   - OpenDP
5. CTS
   - TritonCTS
6. Routing
   - FastRoute
   - TritonRoute
   - OpenRCX
7. Tapeout
   - Magic
   - KLayout
8. Signoff
   - Magic
   - Klayout
   - Netgen
   - CVC

実際に実行すると48ステップある（Lint含む）

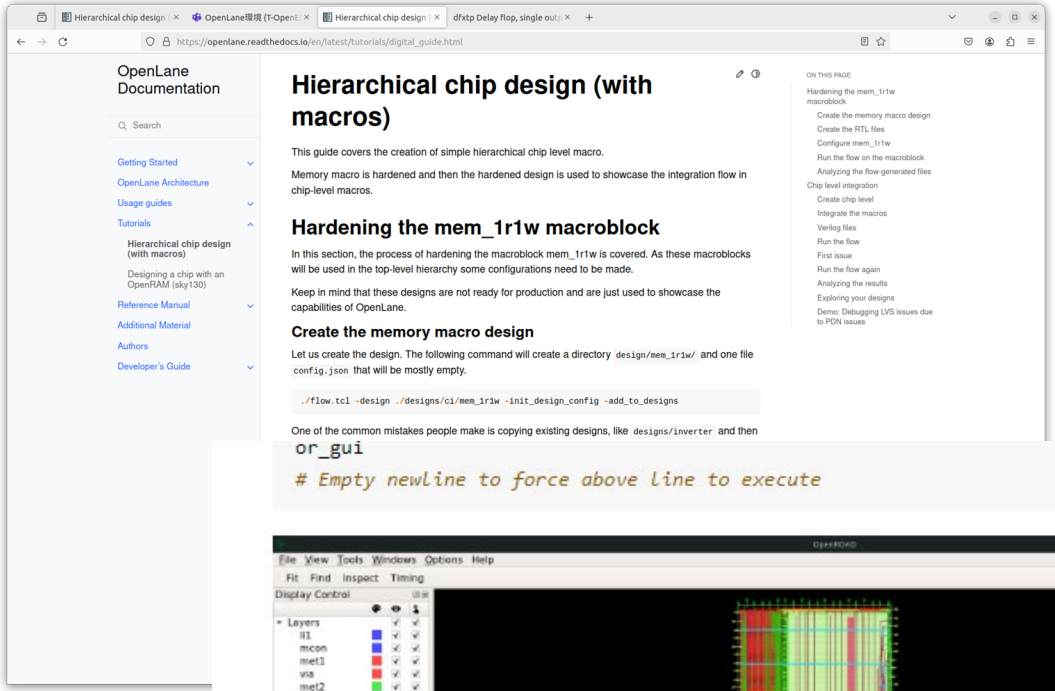
産総研
ともに挑む。つぎを創る。

masa@oseda: ~/OpenLane

```
OpenLane Container (9dbd8b5):/openlane/designs/regfile_2r1w/runs/full_guide/logs$ ls *
cts:
18-cts.errors  18-cts.log  18-cts.warnings  19-cts_sta.errors  19-cts_sta.log  19-cts_sta.warnings  20-resizer.errors  20-resizer.log  20-resizer.warnings

floorplan:
3-initial_fp.errors  3-initial_fp.log  3-initial_fp.warnings  4-io.errors  4-io.log  4-io.warnings  8-tap.errors  8-tap.log  8-tap.warnings  9-pdn.errors  9-pdn.log  9-pdn.warnings

placement:
11-gpl_sta.errors    12-global.warnings  14-gpl_sta.log       16-detailed.errors    17-dpl_sta.warnings  6-gpl_sta.log        9-global_skip_io.errors
11-gpl_sta.log       12-io.errors        14-gpl_sta.warnings  16-detailed.log       4-global.errors      6-gpl_sta.warnings   9-global_skip_io.log
11-gpl_sta.warnings  12-io.log           15-resizer.errors    16-detailed.warnings  4-global.log         7-basic_mp.errors    9-global_skip_io.warnings
12-global.errors     12-io.warnings      15-resizer.log       17-dpl_sta.errors     4-global.warnings    7-basic_mp.log
12-global.log        14-gpl_sta.errors   15-resizer.warnings  17-dpl_sta.log        6-gpl_sta.errors     7-basic_mp.warnings

routing:
21-resizer_design.errors    23-resizer_timing.errors    25-antenna_diodes_1.log      25-global.warnings               27-grt_sta.warnings  29-detailed.warnings
21-resizer_design.log       23-resizer_timing.log       25-antenna_route_1.errors    25-global_write_netlist.errors   28-fill.errors       30-wire_lengths.log
21-resizer_design.warnings  23-resizer_timing.warnings  25-antenna_route_1.log       25-global_write_netlist.log      28-fill.log
22-rsz_design_sta.errors    24-rsz_timing_sta.errors    25-antenna_route_1.warnings  25-global_write_netlist.warnings  28-fill.warnings
22-rsz_design_sta.log       24-rsz_timing_sta.log       25-global.errors             27-grt_sta.errors                29-detailed.errors
22-rsz_design_sta.warnings  24-rsz_timing_sta.warnings  25-global.log                27-grt_sta.log                   29-detailed.log

signoff:
31-parasitics_extraction.min.errors    34-rcx_mcsta.max.errors              37-irdrop.errors     38-lef.errors        41-spice.log                      45-drc.errors
31-parasitics_extraction.min.log       34-rcx_mcsta.max.log                 37-irdrop.log        38-lef.log           41-spice.warnings                 45-drc.log
31-parasitics_extraction.min.warnings  34-rcx_mcsta.max.warnings             37-irdrop.warnings   38-lef.warnings      42-write_powered_def.log          45-drc.warnings
32-rcx_mcsta.min.errors                35-parasitics_extraction.nom.errors  38-gdsii.errors      38-maglef.errors     42-write_powered_verilog.errors   46-drc-klayout.log
32-rcx_mcsta.min.log                   35-parasitics_extraction.nom.log     38-gdsii.log         38-maglef.log        42-write_powered_verilog.log      47-arc.errors
32-rcx_mcsta.min.warnings              35-parasitics_extraction.nom.warnings  38-gdsii.warnings  38-maglef.warnings   42-write_powered_verilog.warnings  47-arc.log
33-parasitics_extraction.max.errors    36-rcx_mcsta.nom.errors              38-gds_ptrs.errors   39-gdsii-klayout.log  44-lvs.lef.log                    47-arc.warnings
33-parasitics_extraction.max.log       36-rcx_mcsta.nom.log                 38-gds_ptrs.log      40-xor.log           44-regfile_2r1w.lef.lvs.json
33-parasitics_extraction.max.warnings  36-rcx_mcsta.nom.warnings             38-gds_ptrs.warnings  41-spice.errors      44-regfile_2r1w.lef.lvs.log

synthesis:
1-synthesis.errors  1-synthesis.log  1-synthesis.warnings  2-sta.errors  2-sta.log  2-sta.warnings  linter.log
OpenLane Container (9dbd8b5):/openlane/designs/regfile_2r1w/runs/full_guide/logs$
```

実際に実行すると48ステップある

チュートリアルの試行：Hierarchcal Chip Design



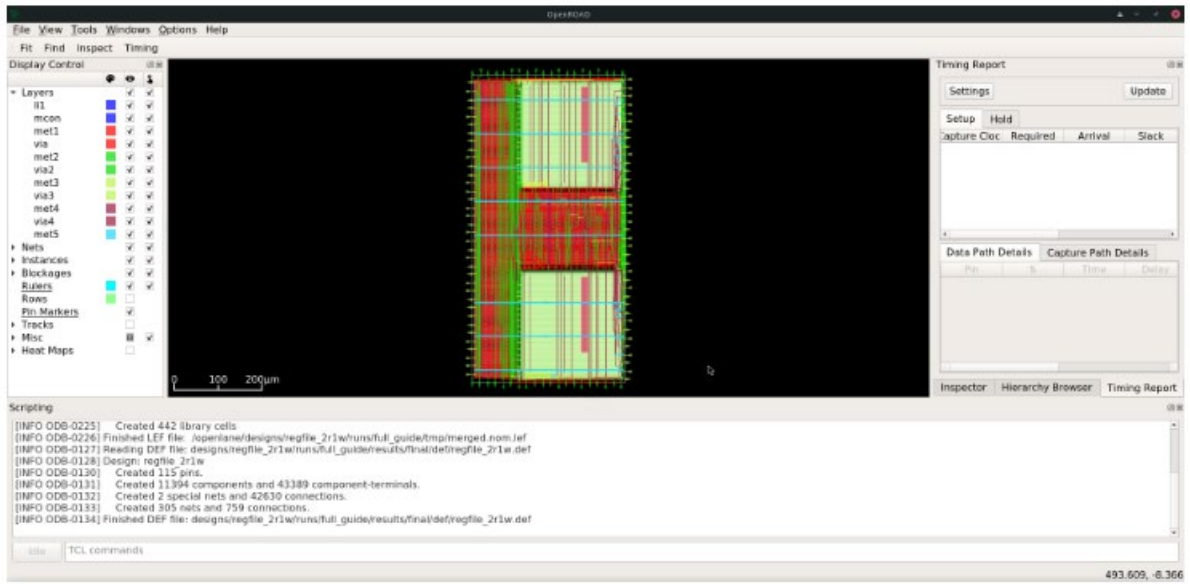


Fig. 2 OpenROAD GUI with loaded final DEF file

手順
1．1Read 1Write Memoryマクロブロックを作成
2．上記マクロブロックを2個並べて2Read 1Write Regfile作成がゴール

どちらもVerilogコード、設定ファイル記述例あり。実行コマンドもあり。

⇒1．は特に問題なく実行が完了する。
　2．でエラーが起こる

Lintチェック（Verilog文法チェック）で止まっている

産総研
ともに挑む。つぎを創る。

マクロブロックのパラメータ継承あたりにエラーあり

masa@oseda: ~/OpenLane

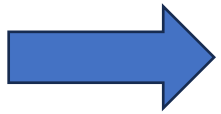
```
[INFO]: Using configuration in 'designs/regfile_2r1w/config.json'...
[INFO]: PDK Root: /home/masa/.volare
[INFO]: Process Design Kit: sky130A
[INFO]: Standard Cell Library: sky130_fd_sc_hd
[INFO]: Optimization Standard Cell Library: sky130_fd_sc_hd
[INFO]: Run Directory: /openlane/designs/regfile_2r1w/runs/full_guide
[INFO]: Removing existing /openlane/designs/regfile_2r1w/runs/full_guide...
[INFO]: Saving runtime environment...
[INFO]: Preparing LEF files for the nom corner...
[INFO]: Preparing LEF files for the min corner...
[INFO]: Preparing LEF files for the max corner...
[WARNING]: PNR_SDC_FILE is not set. It is recommended to write a custom SDC file for the design. Defaulting to BASE_SDC_FILE
[WARNING]: SIGNOFF_SDC_FILE is not set. It is recommended to write a custom SDC file for the design. Defaulting to BASE_SDC_FILE
[INFO]: Running linter (Verilator) (log: designs/regfile_2r1w/runs/full_guide/logs/synthesis/linter.log)...
[ERROR]: 5 errors found by linter
[ERROR]: Step 0 (verilator_lint_check) failed with error:
-code 1 -level 0 -errorcode NONE -errorinfo {
    while executing
"throw_error"
    (procedure "run_verilator" line 86)
    invoked from within
"run_verilator"
    (procedure "run_verilator_step" line 3)
    invoked from within
"run_verilator_step"} -errorline 1
[INFO]: Saving current set of views in 'designs/regfile_2r1w/runs/full_guide/results/final'...
[INFO]: Generating final set of reports...
[INFO]: Created manufacturability report at 'designs/regfile_2r1w/runs/full_guide/reports/manufacturability.rpt'.
[INFO]: Created metrics report at 'designs/regfile_2r1w/runs/full_guide/reports/metrics.csv'.
[INFO]: Saving runtime environment...
[ERROR]: Flow failed.
[INFO]: The failure may have been because of the following warnings:
[WARNING]: PNR_SDC_FILE is not set. It is recommended to write a custom SDC file for the design. Defaulting to BASE_SDC_FILE
[WARNING]: SIGNOFF_SDC_FILE is not set. It is recommended to write a custom SDC file for the design. Defaulting to BASE_SDC_FILE

OpenLane Container (9dbd8b5):/openlane$
```

masa@oseda: ~/OpenLane

```
-code 1 -level 0 -errorcode NONE -errorinfo {
    while executing
"throw_error"
    (procedure "run_verilator" line 86)
    invoked from within
"run_verilator"
    (procedure "run_verilator_step" line 3)
    invoked from within
"run_verilator_step"} -errorline 1
[INFO]: Saving current set of views in 'designs/regfile_2r1w/runs/full_guide/results/final'...
[INFO]: Generating final set of reports...
[INFO]: Created manufacturability report at 'designs/regfile_2r1w/runs/full_guide/reports/manufacturability.rpt'.
[INFO]: Created metrics report at 'designs/regfile_2r1w/runs/full_guide/reports/metrics.csv'.
[INFO]: Saving runtime environment...
[ERROR]: Flow failed.
[INFO]: The failure may have been because of the following warnings:
[WARNING]: PNR_SDC_FILE is not set. It is recommended to write a custom SDC file for the design. Defaulting to BASE_SDC_FILE
[WARNING]: SIGNOFF_SDC_FILE is not set. It is recommended to write a custom SDC file for the design. Defaulting to BASE_SDC_FILE

OpenLane Container (9dbd8b5):/openlane$ more designs/regfile_2r1w/runs/full_guide/logs/synthesis/linter.log
%Error-PINNOTFOUND: /openlane/designs/regfile_2r1w/src/regfile_2r1w.v:39:13: Parameter not found: 'DEPTH_LOG2'
   39 | mem_1r1w #(.DEPTH_LOG2(DEPTH_LOG2), .WIDTH(WIDTH)) lane0(
      |             ^~~~~~~~~~
                    ... For error description see https://verilator.org/warn/PINNOTFOUND?v=5.018
%Error-PINNOTFOUND: /openlane/designs/regfile_2r1w/src/regfile_2r1w.v:39:38: Parameter not found: 'WIDTH'
   39 | mem_1r1w #(.DEPTH_LOG2(DEPTH_LOG2), .WIDTH(WIDTH)) lane0(
      |                                      ^~~~~
%Error-PINNOTFOUND: /openlane/designs/regfile_2r1w/src/regfile_2r1w.v:53:13: Parameter not found: 'DEPTH_LOG2'
   53 | mem_1r1w #(.DEPTH_LOG2(DEPTH_LOG2), .WIDTH(WIDTH)) lane1(
      |             ^~~~~~~~~~
%Error-PINNOTFOUND: /openlane/designs/regfile_2r1w/src/regfile_2r1w.v:53:38: Parameter not found: 'WIDTH'
   53 | mem_1r1w #(.DEPTH_LOG2(DEPTH_LOG2), .WIDTH(WIDTH)) lane1(
      |                                      ^~~~~
%Error: Exiting due to 4 error(s)
       ... See the manual at https://verilator.org/verilator_doc.html for more assistance.
OpenLane Container (9dbd8b5):/openlane$
```

モジュール呼び出しからパラメータを削除

```
mem_1r1w #(.DEPTH_LOG2(DEPTH_LOG2), .WIDTH(WIDTH)) lane0(
    .clk(clk),

    .read_addr(rs1_addr),
    .read(rs1_read),
    .read_data(rs1_rdata),

    .write(write),
    .write_addr(write_addr),
    .write_data(write_data)
);



mem_1r1w #(.DEPTH_LOG2(DEPTH_LOG2), .WIDTH(WIDTH)) lane1(
    .clk(clk),

    .read_addr(rs2_addr),
    .read(rs2_read),
    .read_data(rs2_rdata),

    .write(write),
    .write_addr(write_addr),
    .write_data(write_data)
);


endmodule


U:---  regfile_2r1w.v   All L45    (Verilog)
<C-M-print> is undefined
```

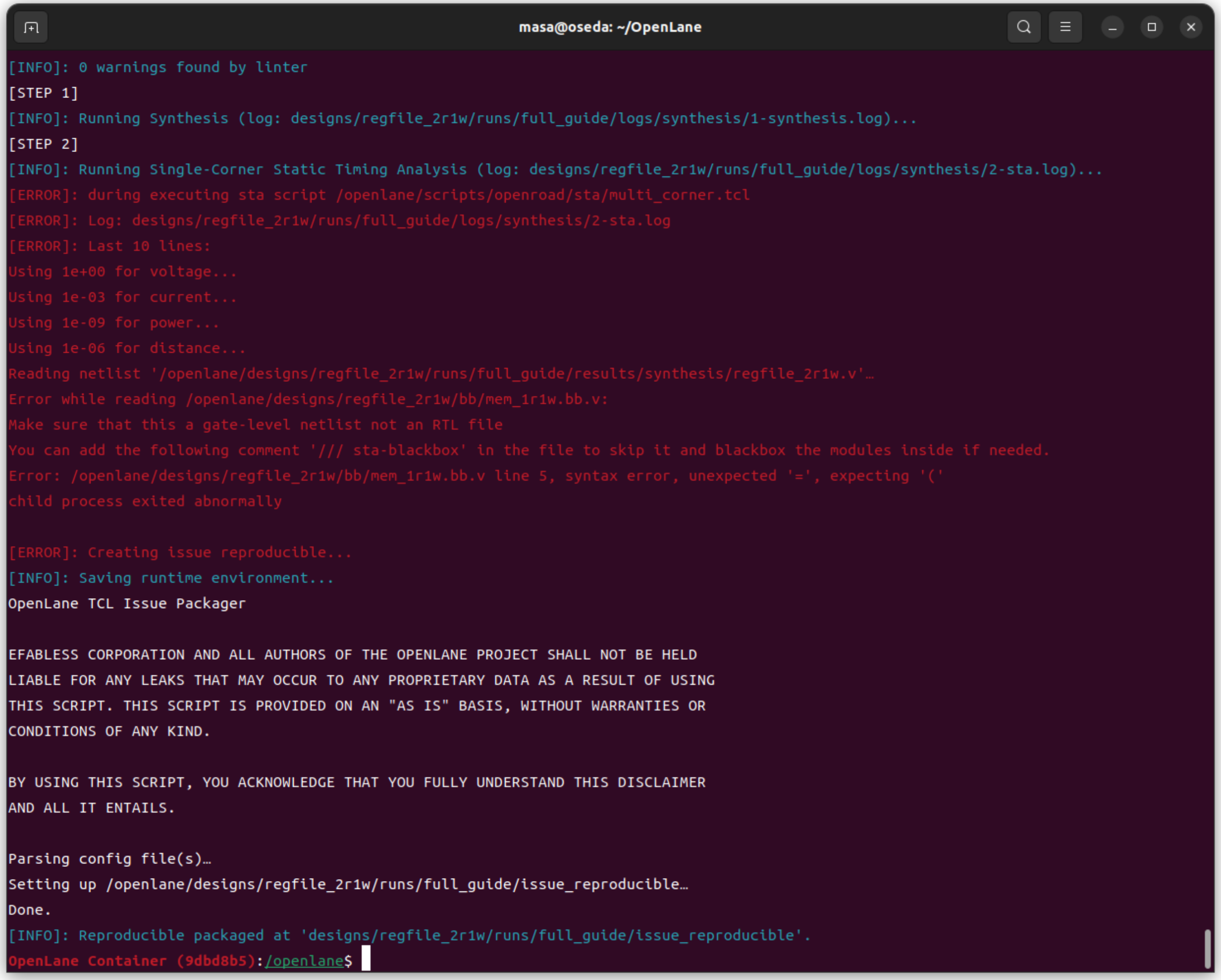


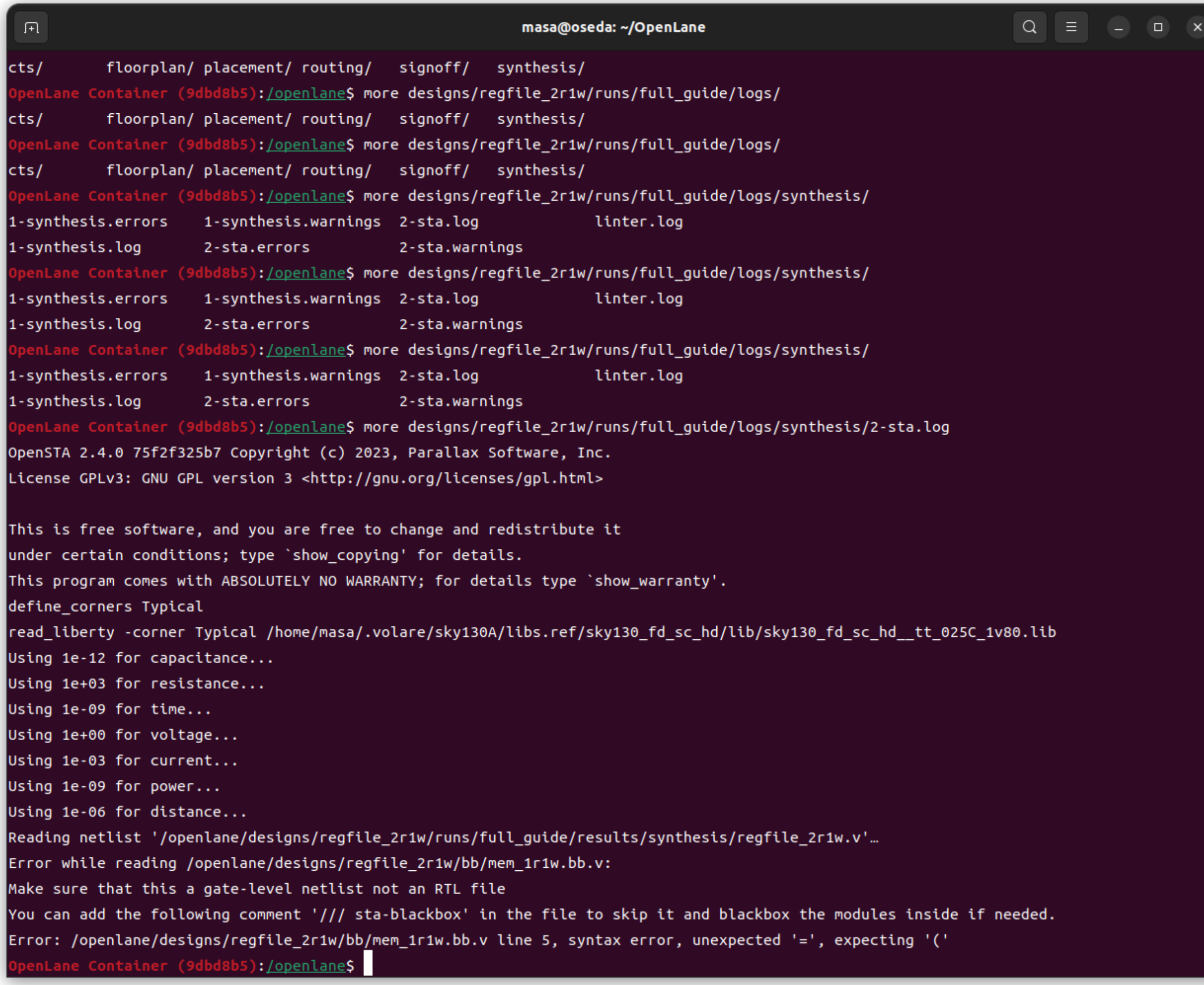
```
mem_1r1w lane0(
    .clk(clk),

    .read_addr(rs1_addr),
    .read(rs1_read),
    .read_data(rs1_rdata),

    .write(write),
    .write_addr(write_addr),
    .write_data(write_data)
);



mem_1r1w lane1(
    .clk(clk),

    .read_addr(rs2_addr),
    .read(rs2_read),
    .read_data(rs2_rdata),

    .write(write),
    .write_addr(write_addr),
    .write_data(write_data)
);


endmodule


U:---  regfile_2r1w.v   All L54    (Verilog)
Wrote /home/masa/OpenLane/designs/regfile_2r1w/src/regfile_2r1w.v
```

今度はSTAで止まっている

産総研
ともに挑む。つぎを創る。

“ブラックボックスファイルがRTLでなくゲートレベルネットリストであることを確認すべし”





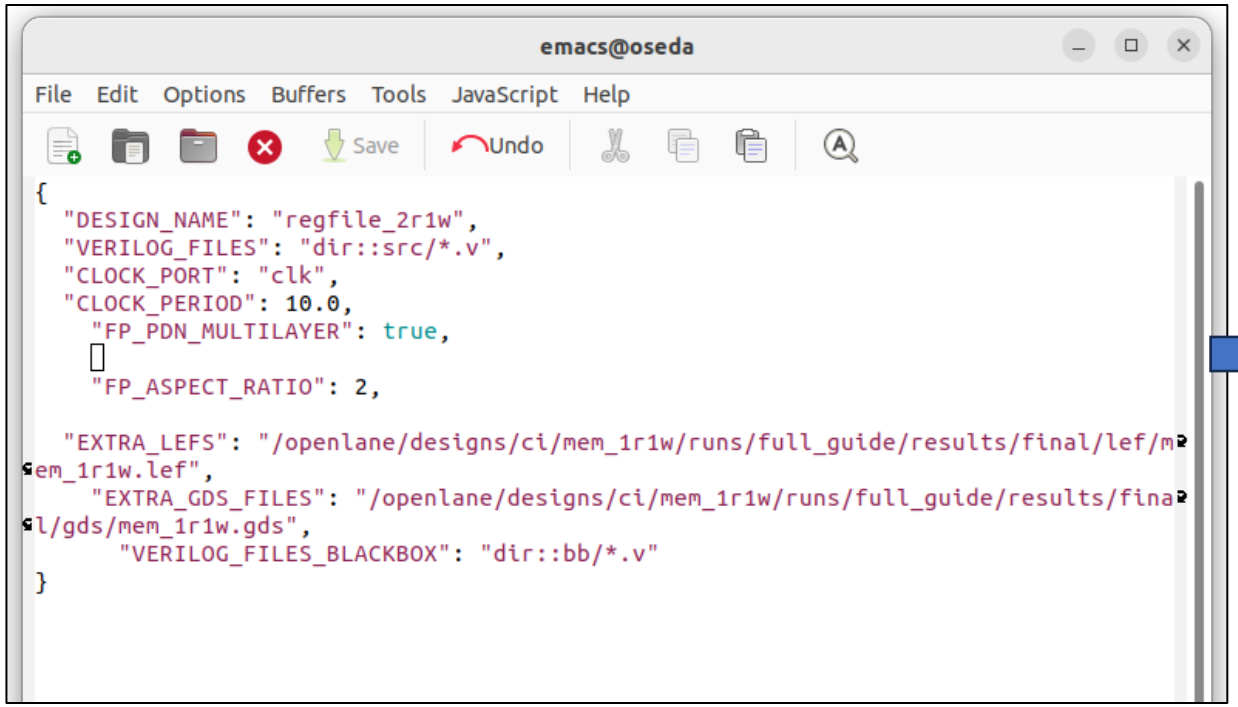
チュートリアルでのブラックボックス作成箇所
RTLの頭を切り抜いただけ、、、

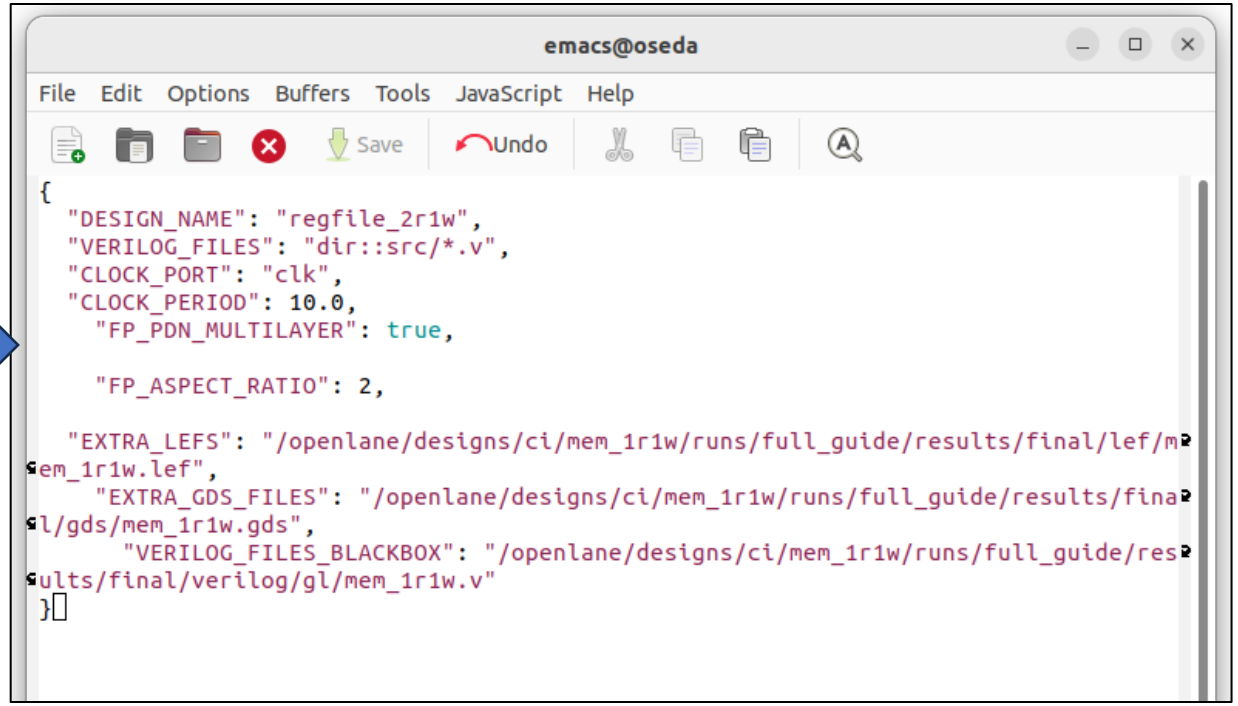
Create the verilog blackbox:

```
(*blackbox*)

module mem_1r1w (clk, read_addr, read, read_data, write_addr, write, write_data);
        parameter DEPTH_LOG2 = 4;
        localparam ELEMENTS = 2**DEPTH_LOG2;
        parameter WIDTH = 32;

        input wire clk;

    input wire [DEPTH_LOG2-1:0] read_addr;
    input wire read;
        output reg [WIDTH-1:0] read_data;


        input wire [DEPTH_LOG2-1:0] write_addr;
        input wire write;
        input wire  [WIDTH-1:0] write_data;

endmodule
```

設定ファイル：Config.jsonを修正

1r1wメモリマクロブロック作成時にゲートレベルネットリストが既に作成されているので
ブラックボックスファイルとしてパスを通す





Flow Complete!

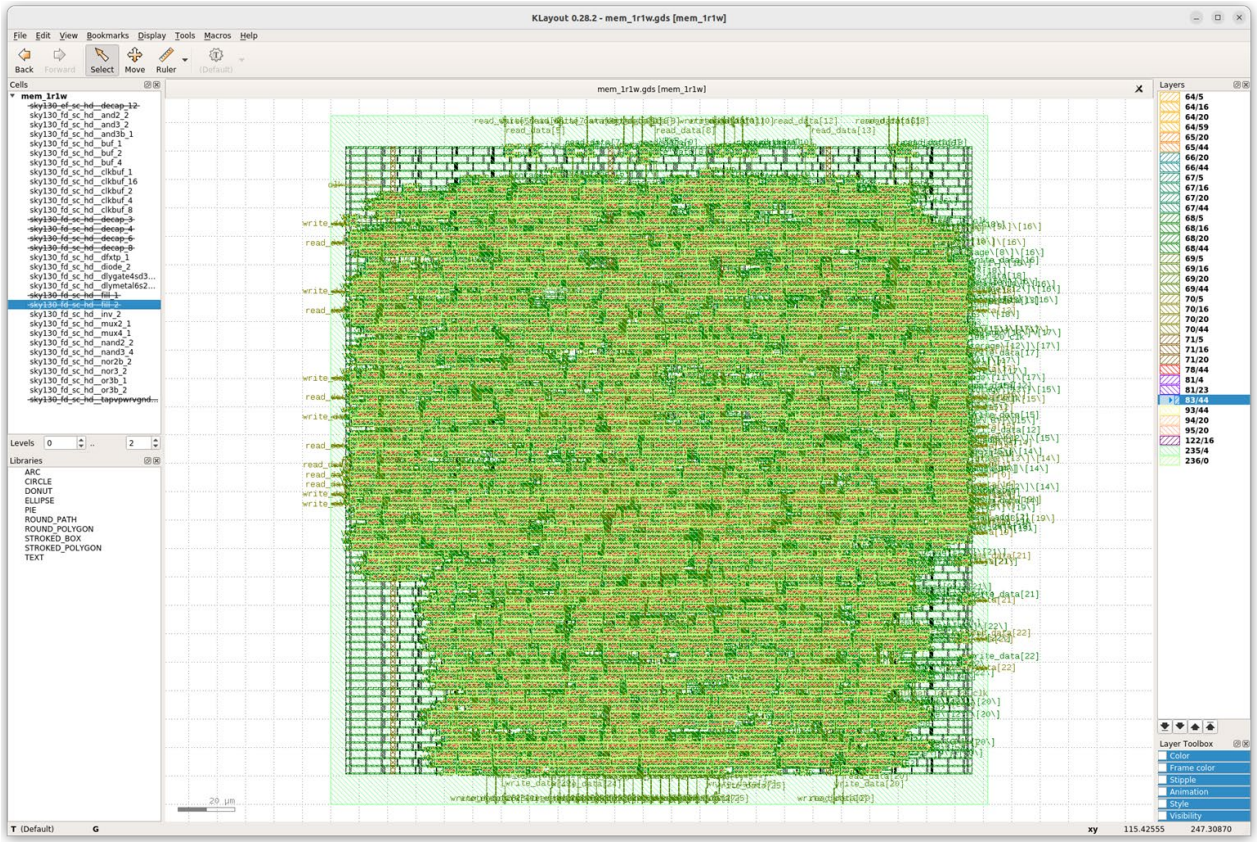
masa@oseda: ~/OpenLane

```
[STEP 42]
[INFO]: Writing Powered Verilog (logs: designs/regfile_2r1w/runs/full_guide/logs/signoff/42-write_powered_def.log, designs/regfile_2r1w
/runs/full_guide/logs/signoff/42-write_powered_verilog.log)...
[STEP 43]
[INFO]: Writing Verilog (log: designs/regfile_2r1w/runs/full_guide/logs/signoff/42-write_powered_verilog.log)...
[STEP 44]
[INFO]: Running LVS (log: designs/regfile_2r1w/runs/full_guide/logs/signoff/44-lvs.lef.log)...
[STEP 45]
[INFO]: Running Magic DRC (log: designs/regfile_2r1w/runs/full_guide/logs/signoff/45-drc.log)...
[INFO]: Converting Magic DRC database to various tool-readable formats...
[INFO]: No Magic DRC violations after GDS streaming out.
[STEP 46]
[INFO]: Running KLayout DRC (log: designs/regfile_2r1w/runs/full_guide/logs/signoff/46-drc-klayout.log)...
[INFO]: No KLayout DRC violations after GDS streaming out.
[STEP 47]
[INFO]: Running OpenROAD Antenna Rule Checker (log: designs/regfile_2r1w/runs/full_guide/logs/signoff/47-arc.log)...
[INFO]: Saving current set of views in 'designs/regfile_2r1w/runs/full_guide/results/final'...
[INFO]: Saving runtime environment...
[INFO]: Generating final set of reports...
[INFO]: Created manufacturability report at 'designs/regfile_2r1w/runs/full_guide/reports/manufacturability.rpt'.
[INFO]: Created metrics report at 'designs/regfile_2r1w/runs/full_guide/reports/metrics.csv'.
[WARNING]: There are max fanout violations in the design at the Typical corner. Please refer to 'designs/regfile_2r1w/runs/full_guide/r
eports/signoff/36-sta-rcx_nom/multi_corner_sta.checks.rpt'.
[INFO]: There are no hold violations in the design at the Typical corner.
[INFO]: There are no setup violations in the design at the Typical corner.
[SUCCESS]: Flow complete.
[INFO]: Note that the following warnings have been generated:
[WARNING]: PNR_SDC_FILE is not set. It is recommended to write a custom SDC file for the design. Defaulting to BASE_SDC_FILE
[WARNING]: SIGNOFF_SDC_FILE is not set. It is recommended to write a custom SDC file for the design. Defaulting to BASE_SDC_FILE
[WARNING]: 4 warnings found by linter
[WARNING]: VSRC_LOC_FILES was not given a value, which may make the results of IR drop analysis inaccurate. If you are not integrating
a top-level chip for manufacture, you may ignore this warning, otherwise, see the documentation for VSRC_LOC_FILES.
[WARNING]: There are max fanout violations in the design at the Typical corner. Please refer to 'designs/regfile_2r1w/runs/full_guide/r
eports/signoff/36-sta-rcx_nom/multi_corner_sta.checks.rpt'.

OpenLane Container (9dbd8b5):/openlane$
```

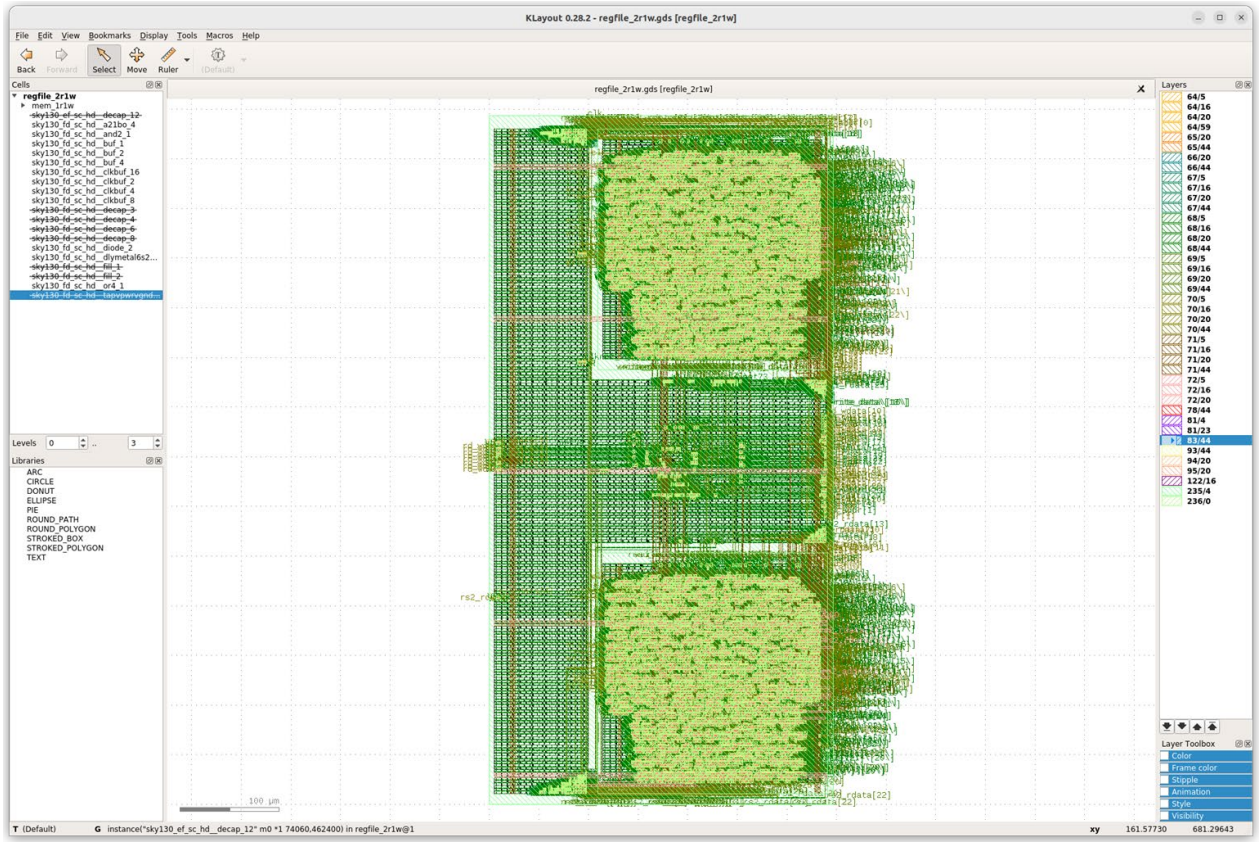
GDSファイルの作成結果（Klayoutで表示）

産総研
ともに挑む。つぎを創る。

Mem 1r1w



regfile 2r1w



- ツールは安定版に移行した
- ドキュメントの更新が追い付いていないのかも？

2023年度の進捗： HWセキュリティ向け回路IPの開発

Summary

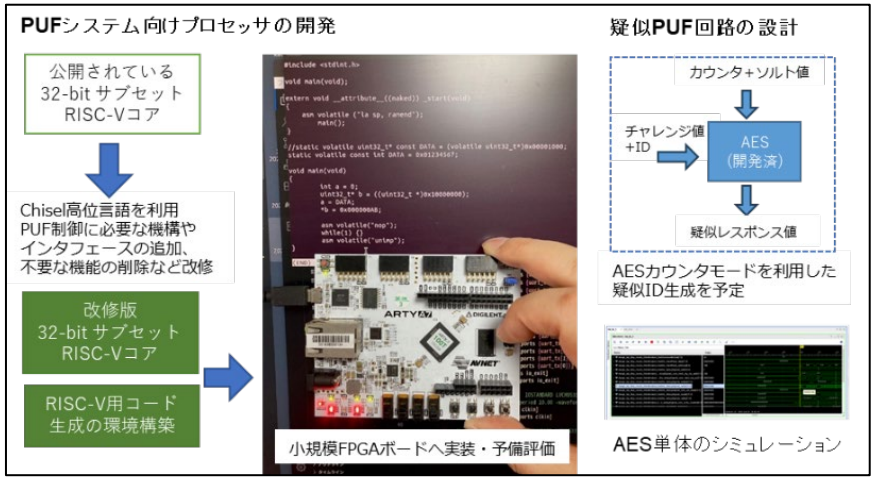
- HWセキュリティシステム用のRISC-V、疑似PUF回路、誤り訂正符号、楕円曲線暗号の回路IP開発を目標
- RISC-Vと疑似PUF回路は回路記述ファイル（RTL）・機能検証・レイアウト生成を完了。
- 誤り訂正符号と楕円曲線暗号はC言語で作成完了。
  ⇒C言語から変換したRTLでOpenLane flowを実行してもレイアウト生成が完了していない。原因究明中。

モチベーション

- オープンソースEDAの利用によって誰でも半導体を作れるようになる
  ⇒偽造半導体の製造も容易になるかもしれない？
- 産総研がいち早くオープンソース半導体開発環境下のHWセキュリティを対策。そのための回路IPを整備

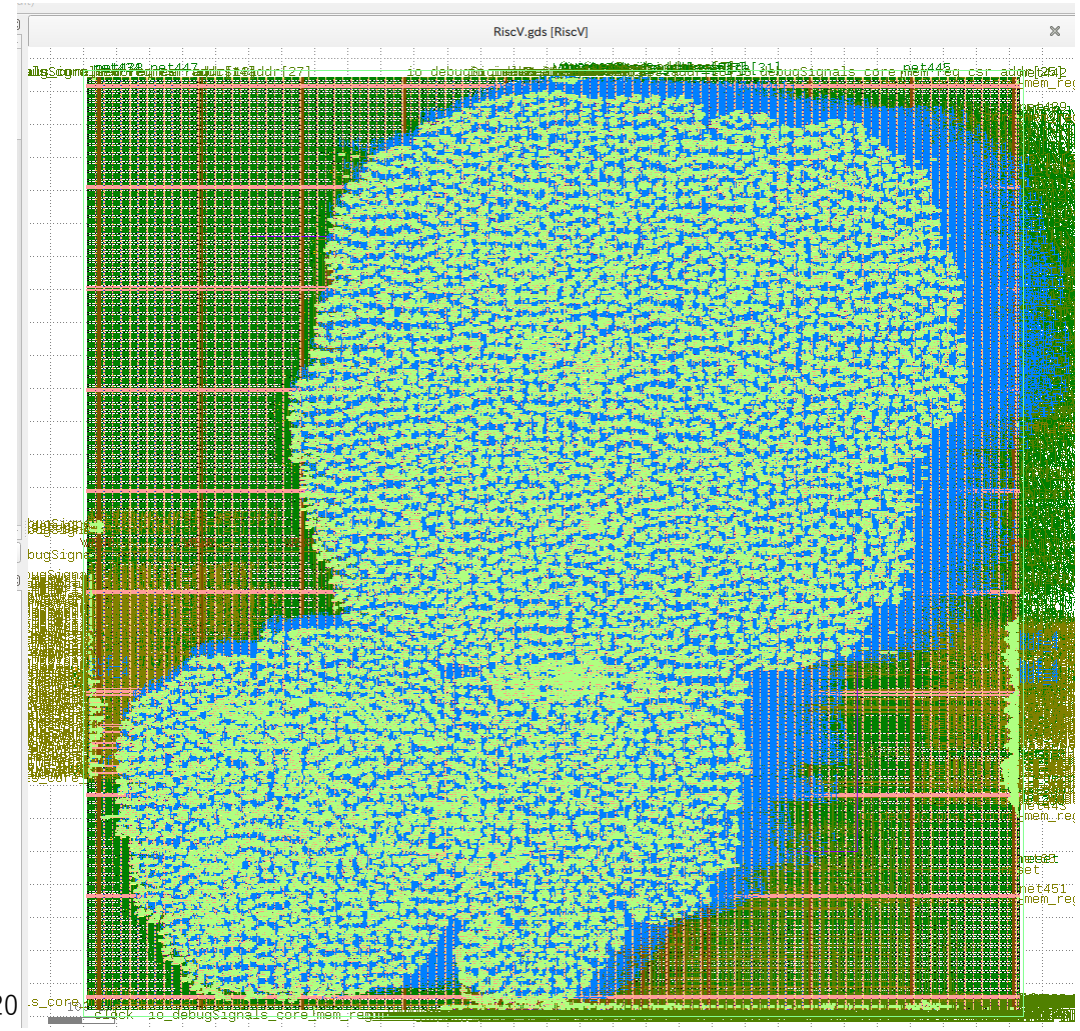
回路IPの開発状況

回路IP	回路記述（RTL）	回路の機能検証	レイアウト生成
RISC-V	○	○（FPGAを利用）	○
疑似PUF回路	○	○（シミュレーション）	○
誤り訂正符号	△（C言語）	×	×
楕円曲線暗号	△（C言語）	×	×

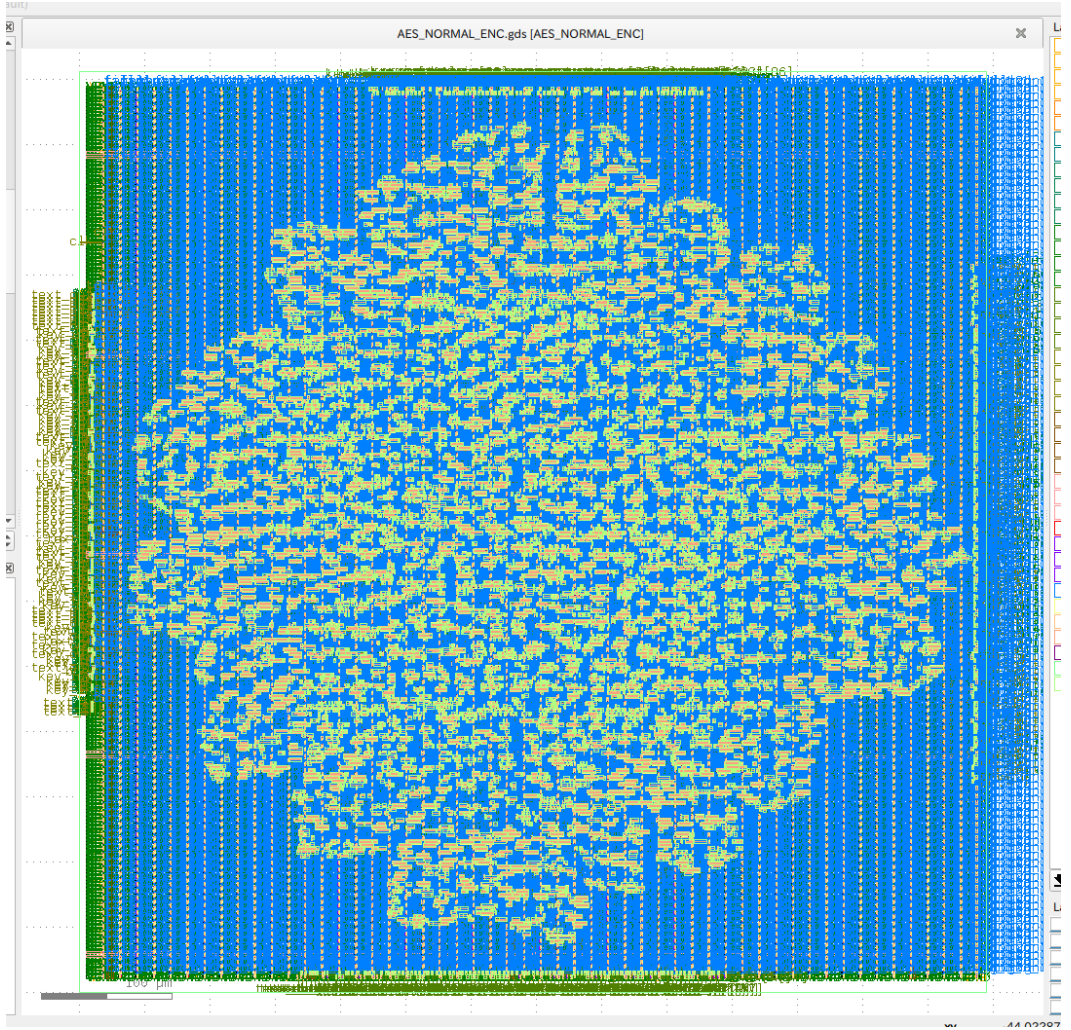


GDS生成結果

RISC-V



疑似PUF



2023年度の進捗： アナログ設計環境の拡張

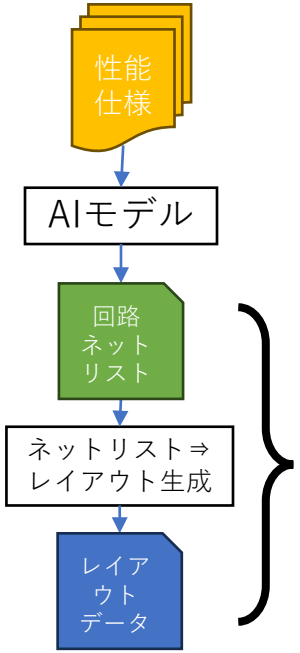
Summary

- オープンソースの回路ネットリスト⇒レイアウト生成ツールを用いてオペアンプを例題に回路ネットリストからskywater130nmのレイアウト生成環境を整備
- 加えてオープンソースEDAで完結する形でデザインルール検証環境（DRC）、ネットリストとレイアウトの整合性検証環境（LVS）も整備

モチベーション

- アナログ回路設計は回路・デバイス・レイアウト等の様々な知識・ノウハウが必要なためデジタルと比較して敷居が高い
- オープンPDKやAI/機械学習を活用して、知識のない人でも簡単にアナログ回路を作成できるようにオープンソースEDAのアナログ設計環境を拡張する

拡張アナログ設計フロー



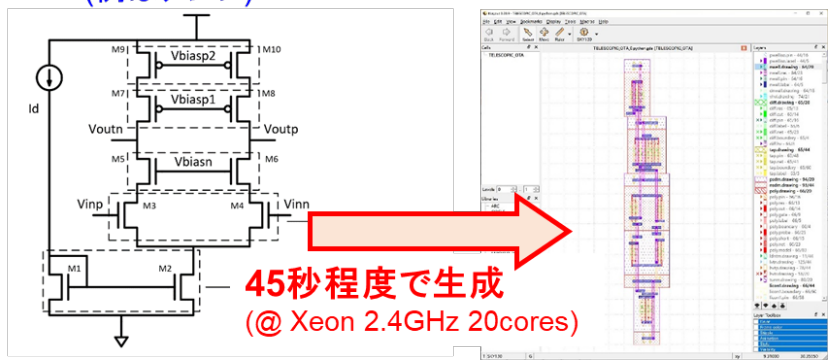
今年度実施内容

- ミネソタ大が開発した回路ネットリスト⇒レイアウト生成オープンソースツール“ALIGN”でskywater130nmオープンソースPDKを利用できるよう設定修正
- オペアンプを例題にALIGNで回路ネットリストからレイアウト生成を完了
- 加えてオープンソースEDAで完結する形でデザインルール検証環境（DRC）、ネットリストとレイアウトの整合性検証環境（LVS）も整備

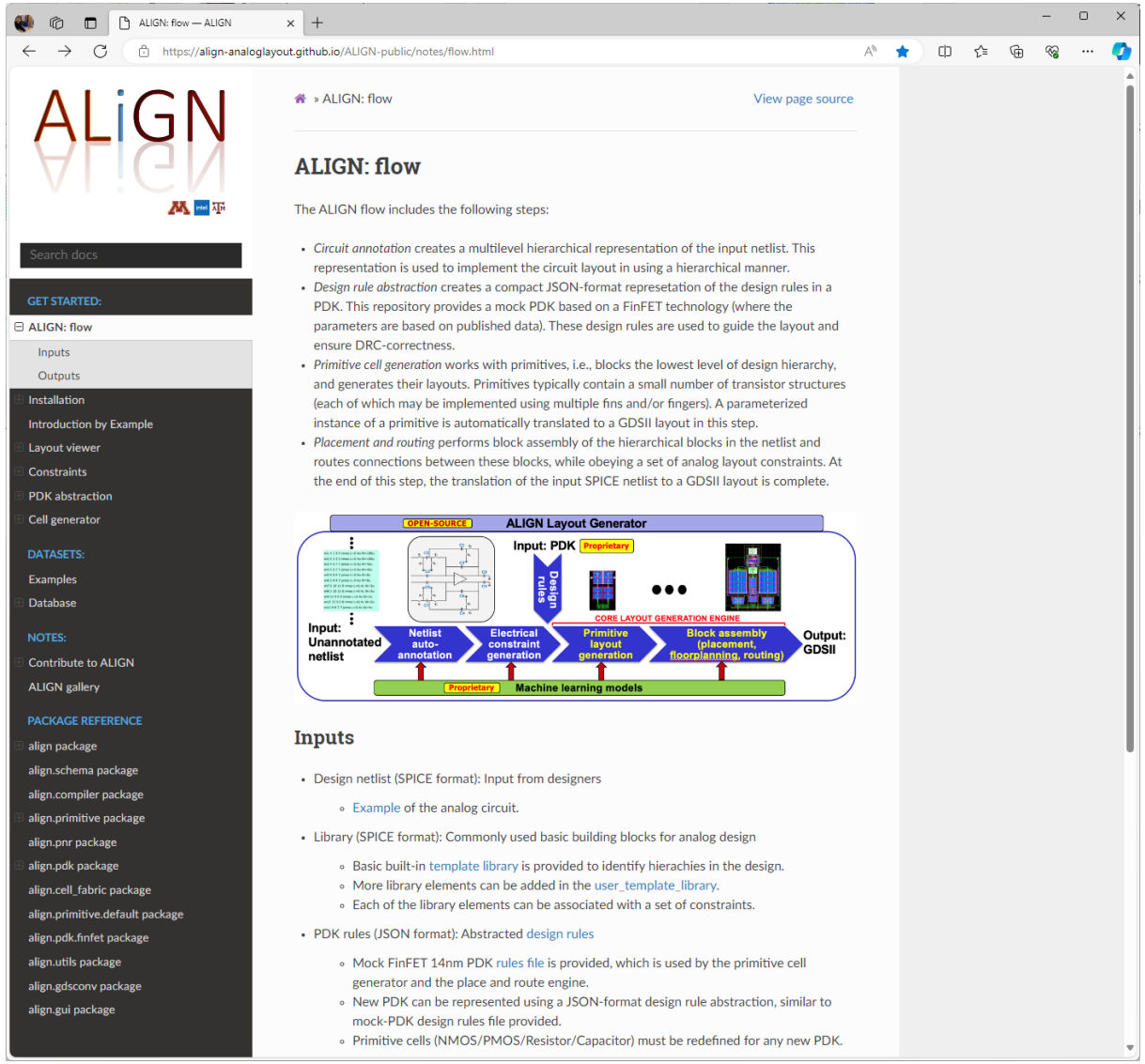
オペアンプを用いた変換例

回路図(ネットリスト)を入力
(例はアンプ)

GDSを出力



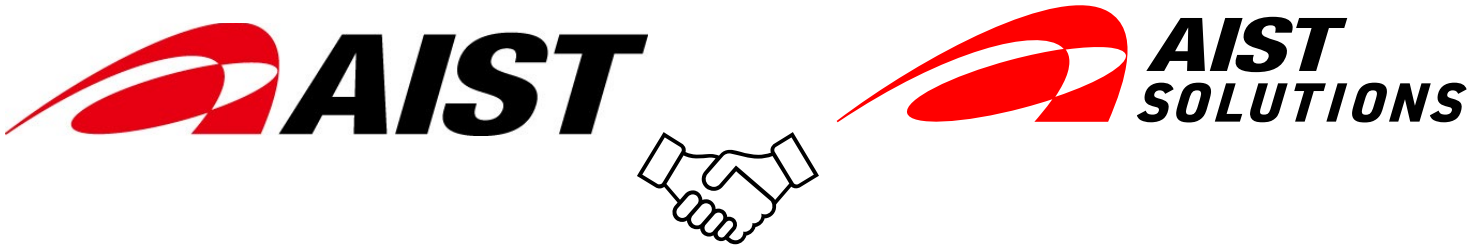
ALIGN: Analog Layout, Intelligently Generated from Netlists



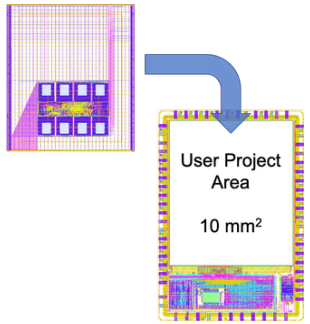
AIST SOLUTIONSとのコラボレーション

本研究で得られた知見や情報を産総研単独で広く公開するのは簡単ではない
- ヒューマンリソース
- コネクション
- メソッド　etc

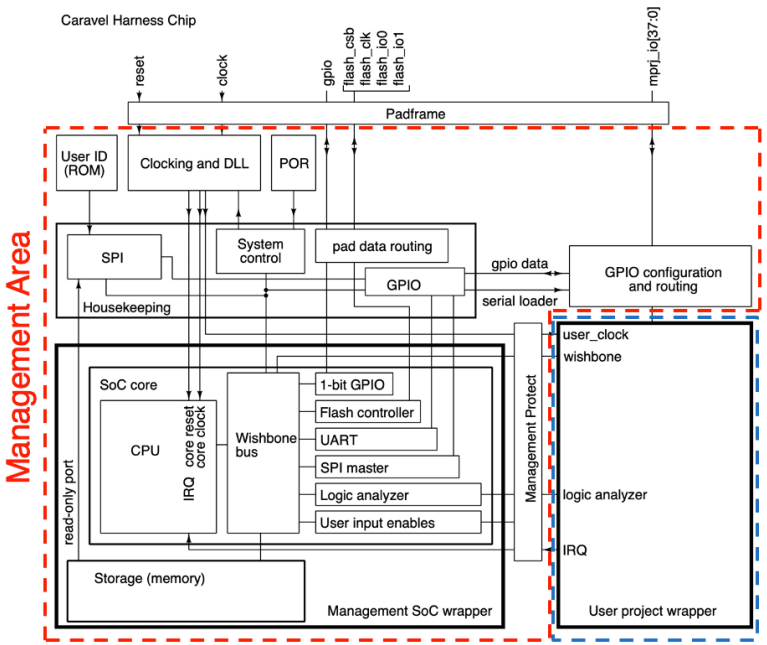
⇒AIST SOLUTIONSで加速
- Open Source Silicon活用検証委員会を立ち上げた
- 委員会活動の一環で本研究で得た知見を公開予定
- efablessのChipIgniteシャトルでチップ試作

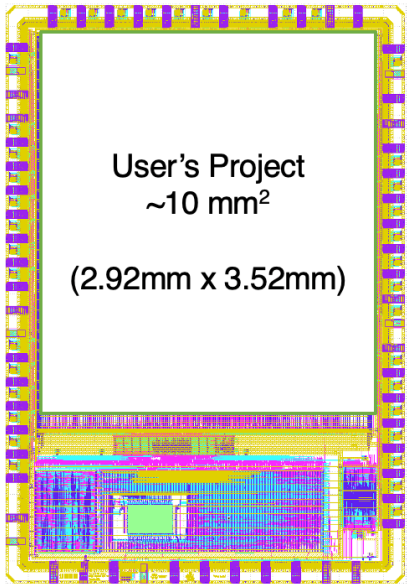


ChipIgniteシャトルサービスChipIgniteシャトルサービス



- ChipIgniteシャトル：efablessによる有償シャトル（$9,750）
- Caravelハーネス、オープンPDK、Openlaneからなる統合環境
- ユーザ回路のVerilog to GDS、ハーネスへの配置、テープアウトチェックまでほぼ自動
- シミュレーション環境もある
- QFNチップ100個、評価ボード付



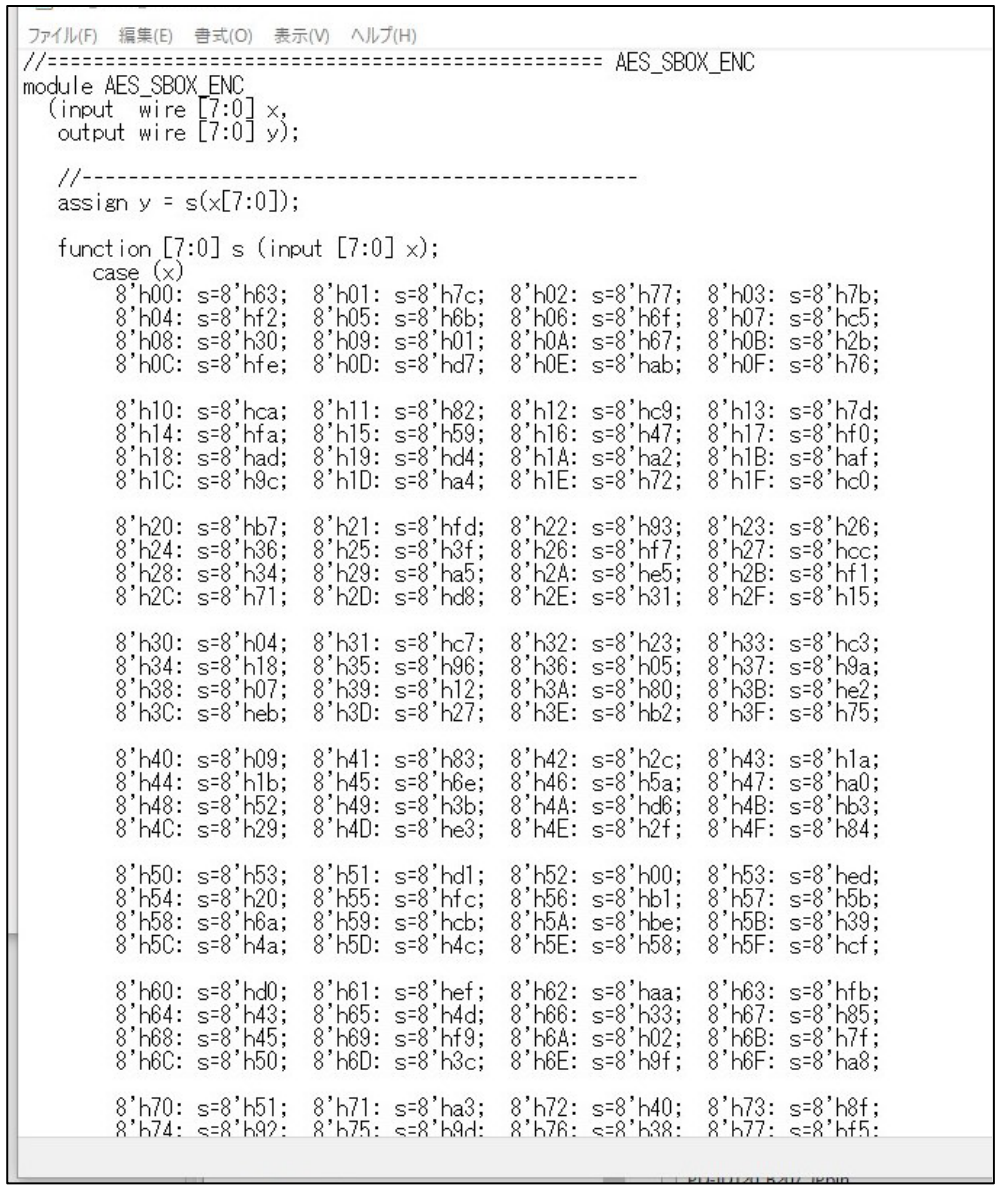


- 38 programmable IO's
- 10 mm2 of User Project Area
- Diagnostic port including IO configuration and Flash pass-thru access for programming
- VexRiscv CPU with debug
- 3 kbytes of RAM
- Flash controller supporting execute-in-place
- SPI, UART and GPIO
- Counter/Timers
- 128 signal logic analyzer for project

ChipIgniteシャトルでのTapeout方針

- efablessから提供されているexampleを活用し、IPを組み込む
- Exampleは必要最小限の変更に留める
  - IPポート名、IPポート数
  - ピン配置
  - もともと存在するモジュールに手を付けない
- IPは公開ベースのものを使用（人為的エラー回避、設計時間の短縮）

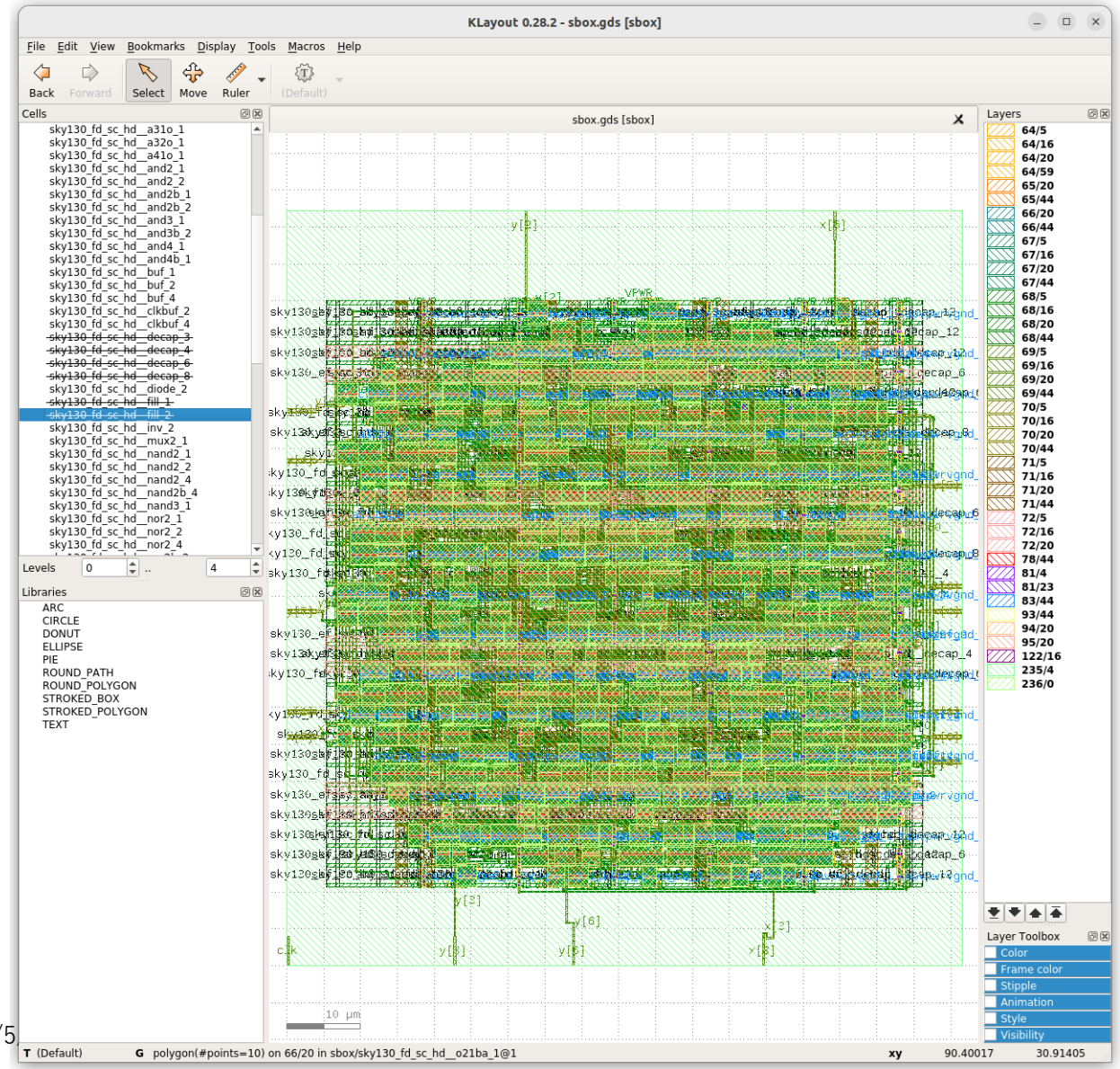
Our design: AES SBOX-like function



```
ファイル(F)  編集(E)  書式(O)  表示(V)  ヘルプ(H)
//================================================ AES_SBOX_ENC
module AES_SBOX_ENC
  (input  wire [7:0] x,
   output wire [7:0] y);

   //----------------------------------------------
   assign y = s(x[7:0]);

   function [7:0] s (input [7:0] x);
      case (x)
        8'h00: s=8'h63;  8'h01: s=8'h7c;  8'h02: s=8'h77;  8'h03: s=8'h7b;
        8'h04: s=8'hf2;  8'h05: s=8'h6b;  8'h06: s=8'h6f;  8'h07: s=8'hc5;
        8'h08: s=8'h30;  8'h09: s=8'h01;  8'h0A: s=8'h67;  8'h0B: s=8'h2b;
        8'h0C: s=8'hfe;  8'h0D: s=8'hd7;  8'h0E: s=8'hab;  8'h0F: s=8'h76;

        8'h10: s=8'hca;  8'h11: s=8'h82;  8'h12: s=8'hc9;  8'h13: s=8'h7d;
        8'h14: s=8'hfa;  8'h15: s=8'h59;  8'h16: s=8'h47;  8'h17: s=8'hf0;
        8'h18: s=8'had;  8'h19: s=8'hd4;  8'h1A: s=8'ha2;  8'h1B: s=8'haf;
        8'h1C: s=8'h9c;  8'h1D: s=8'ha4;  8'h1E: s=8'h72;  8'h1F: s=8'hc0;

        8'h20: s=8'hb7;  8'h21: s=8'hfd;  8'h22: s=8'h93;  8'h23: s=8'h26;
        8'h24: s=8'h36;  8'h25: s=8'h3f;  8'h26: s=8'hf7;  8'h27: s=8'hcc;
        8'h28: s=8'h34;  8'h29: s=8'ha5;  8'h2A: s=8'he5;  8'h2B: s=8'hf1;
        8'h2C: s=8'h71;  8'h2D: s=8'hd8;  8'h2E: s=8'h31;  8'h2F: s=8'h15;

        8'h30: s=8'h04;  8'h31: s=8'hc7;  8'h32: s=8'h23;  8'h33: s=8'hc3;
        8'h34: s=8'h18;  8'h35: s=8'h96;  8'h36: s=8'h05;  8'h37: s=8'h9a;
        8'h38: s=8'h07;  8'h39: s=8'h12;  8'h3A: s=8'h80;  8'h3B: s=8'he2;
        8'h3C: s=8'heb;  8'h3D: s=8'h27;  8'h3E: s=8'hb2;  8'h3F: s=8'h75;

        8'h40: s=8'h09;  8'h41: s=8'h83;  8'h42: s=8'h2c;  8'h43: s=8'h1a;
        8'h44: s=8'h1b;  8'h45: s=8'h6e;  8'h46: s=8'h5a;  8'h47: s=8'ha0;
        8'h48: s=8'h52;  8'h49: s=8'h3b;  8'h4A: s=8'hd6;  8'h4B: s=8'hb3;
        8'h4C: s=8'h29;  8'h4D: s=8'he3;  8'h4E: s=8'h2f;  8'h4F: s=8'h84;

        8'h50: s=8'h53;  8'h51: s=8'hd1;  8'h52: s=8'h00;  8'h53: s=8'hed;
        8'h54: s=8'h20;  8'h55: s=8'hfc;  8'h56: s=8'hb1;  8'h57: s=8'h5b;
        8'h58: s=8'h6a;  8'h59: s=8'hcb;  8'h5A: s=8'hbe;  8'h5B: s=8'h39;
        8'h5C: s=8'h4a;  8'h5D: s=8'h4c;  8'h5E: s=8'h58;  8'h5F: s=8'hcf;

        8'h60: s=8'hd0;  8'h61: s=8'hef;  8'h62: s=8'haa;  8'h63: s=8'hfb;
        8'h64: s=8'h43;  8'h65: s=8'h4d;  8'h66: s=8'h33;  8'h67: s=8'h85;
        8'h68: s=8'h45;  8'h69: s=8'hf9;  8'h6A: s=8'h02;  8'h6B: s=8'h7f;
        8'h6C: s=8'h50;  8'h6D: s=8'h3c;  8'h6E: s=8'h9f;  8'h6F: s=8'ha8;

        8'h70: s=8'h51;  8'h71: s=8'ha3;  8'h72: s=8'h40;  8'h73: s=8'h8f;
        8'h74: s=8'h92;  8'h75: s=8'h9d;  8'h76: s=8'h38;  8'h77: s=8'hf5;
```

- 8bit inputs and outputs
- Defines 8bit functions
- Output a unique value corresponding to an input

SBOXのGDS生成結果（Klayoutで表示）



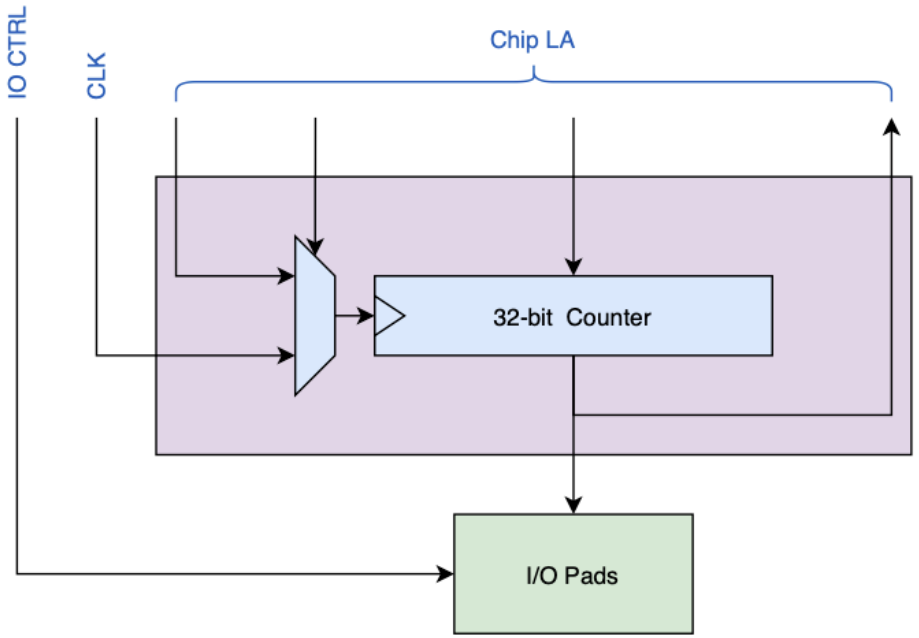
コマンド

./flow.tcl –design sbox

VerilogからGDSまで一気に生成

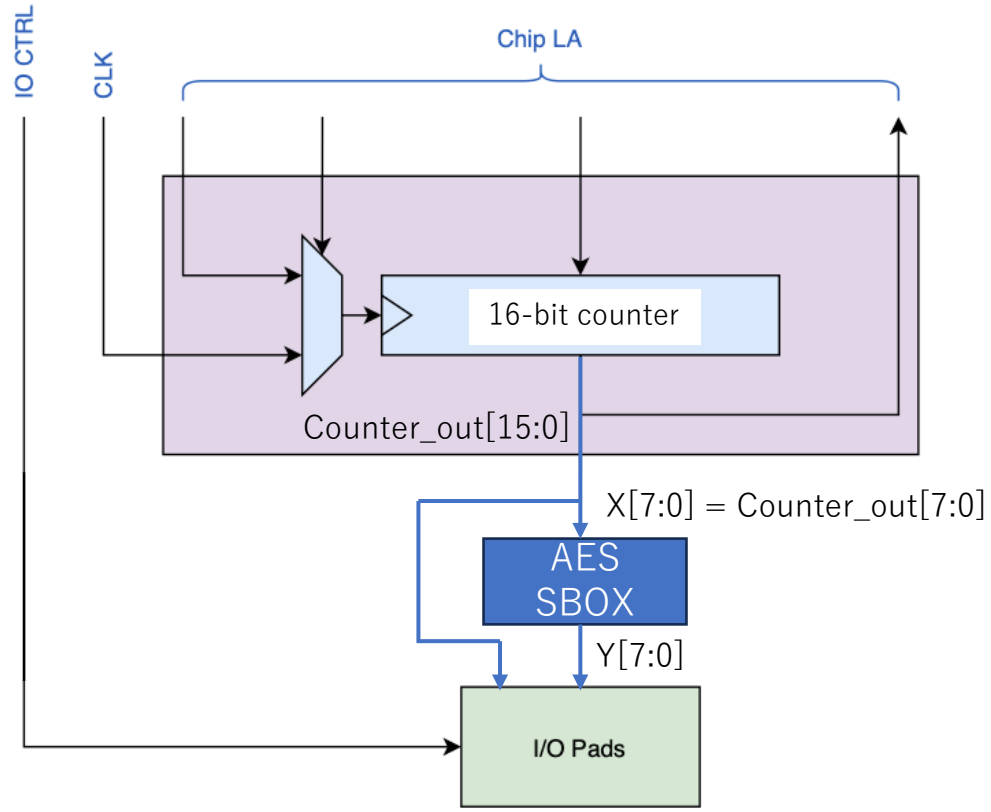
Embedding AES SBOX into the counter example

Original counter circuit example
in a caravel user project

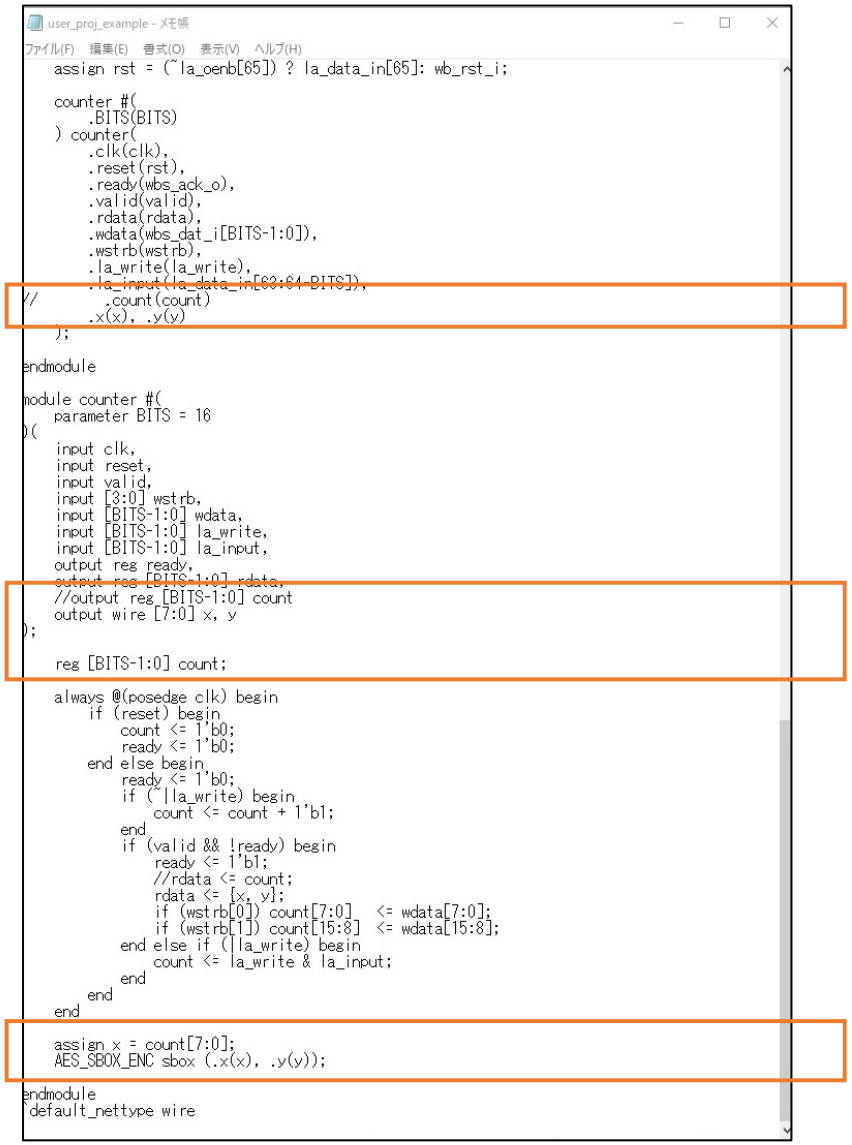


Change

Our design



Change points in User_proj_example.v



```
user_proj_example - メモ帳
ファイル(F)  編集(E)  書式(O)  表示(V)  ヘルプ(H)
    assign rst = (~la_oenb[65]) ? la_data_in[65]: wb_rst_i;

    counter #(
        .BITS(BITS)
    ) counter(
        .clk(clk),
        .reset(rst),
        .ready(wbs_ack_o),
        .valid(valid),
        .rdata(rdata),
        .wdata(wbs_dat_i[BITS-1:0]),
        .wstrb(wstrb),
        .la_write(la_write),
        .la_input(la_data_in[63:64-BITS]),
//        .count(count)
        .x(x), .y(y)
    );

endmodule

module counter #(
    parameter BITS = 16
)(
    input clk,
    input reset,
    input valid,
    input [3:0] wstrb,
    input [BITS-1:0] wdata,
    input [BITS-1:0] la_write,
    input [BITS-1:0] la_input,
    output reg ready,
    output reg [BITS-1:0] rdata,
    //output reg [BITS-1:0] count
    output wire [7:0] x, y
);

    reg [BITS-1:0] count;

    always @(posedge clk) begin
        if (reset) begin
            count <= 1'b0;
            ready <= 1'b0;
        end else begin
            ready <= 1'b0;
            if (~|la_write) begin
                count <= count + 1'b1;
            end
            if (valid && !ready) begin
                ready <= 1'b1;
                //rdata <= count;
                rdata <= {x, y};
                if (wstrb[0]) count[7:0]   <= wdata[7:0];
                if (wstrb[1]) count[15:8]  <= wdata[15:8];
            end else if (|la_write) begin
                count <= la_write & la_input;
            end
        end
    end

    assign x = count[7:0];
    AES_SBOX_ENC sbox (.x(x), .y(y));

endmodule
`default_nettype wire
```

- In counter subcircuit:
  - Comment out “counter” port
  - Add “x/y” ports
- In counter module:
  - Comment out “count” output
  - Add wire type “x/y” output
  - Add register type “count”
  - Assign “x” = “count”
  - Insert sbox subcircuit

Simulation results on a caravel environment

Function definitions in Verilog code of AES SBOX

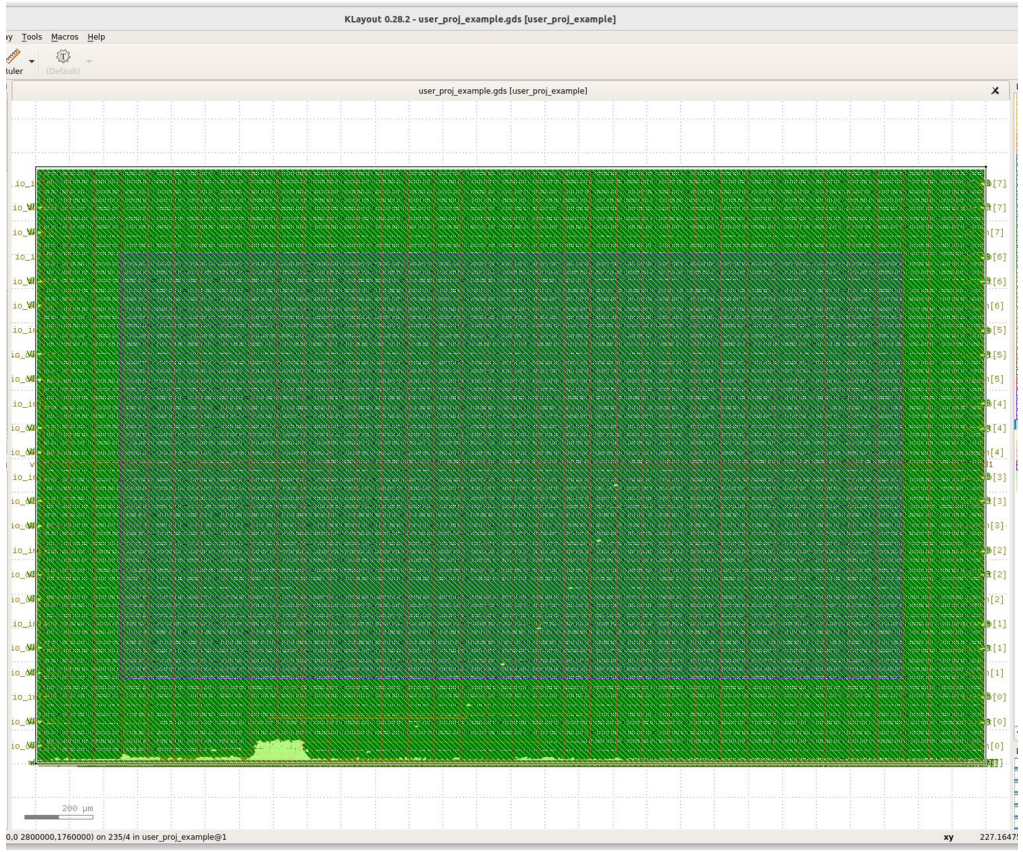
```
8'h90: s=8'h60;  8'h91: s=8'h81;  8'h92: s=8'h4f;  8'h93: s=8'hdc;
8'h94: s=8'h22;  8'h95: s=8'h2a;  8'h96: s=8'h90;  8'h97: s=8'h88;
8'h98: s=8'h46;  8'h99: s=8'hee;  8'h9A: s=8'hb8;  8'h9B: s=8'h14;
8'h9C: s=8'hde;  8'h9D: s=8'h5e;  8'h9E: s=8'h0b;  8'h9F: s=8'hdb;

8'hA0: s=8'he0;  8'hA1: s=8'h32;  8'hA2: s=8'h3a;  8'hA3: s=8'h0a;
8'hA4: s=8'h49;  8'hA5: s=8'h06;  8'hA6: s=8'h24;  8'hA7: s=8'h5c;
8'hA8: s=8'hc2;  8'hA9: s=8'hd3;  8'hAA: s=8'hac;  8'hAB: s=8'h62;
8'hAC: s=8'h91;  8'hAD: s=8'h95;  8'hAE: s=8'he4;  8'hAF: s=8'h79;

8'hB0: s=8'he7;  8'hB1: s=8'hc8;  8'hB2: s=8'h37;  8'hB3: s=8'h6d;
8'hB4: s=8'h8d;  8'hB5: s=8'hd5;  8'hB6: s=8'h4e;  8'hB7: s=8'ha9;
8'hB8: s=8'h6c;  8'hB9: s=8'h56;  8'hBA: s=8'hf4;  8'hBB: s=8'hea;
8'hBC: s=8'h65;  8'hBD: s=8'h7a;  8'hBE: s=8'hae;  8'hBF: s=8'h08;

8'hC0: s=8'hba;  8'hC1: s=8'h78;  8'hC2: s=8'h25;  8'hC3: s=8'h2e;
8'hC4: s=8'h1c;  8'hC5: s=8'ha6;  8'hC6: s=8'hb4;  8'hC7: s=8'hc6;
```

Simulation results

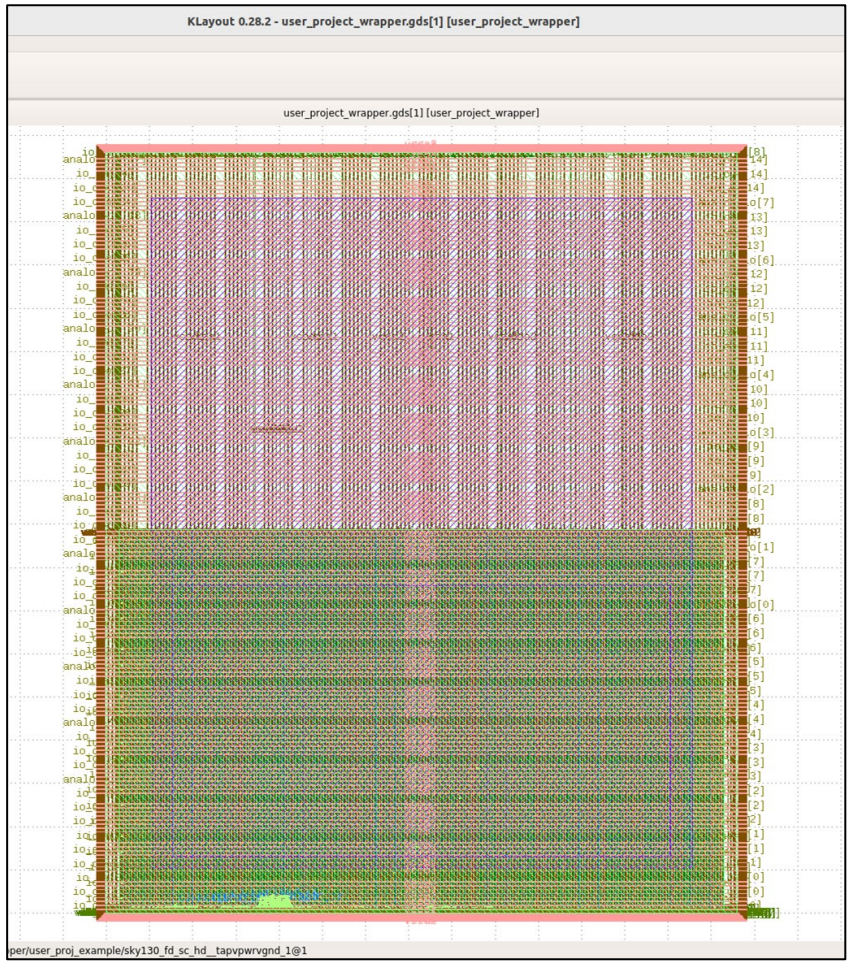
```
vvp  io_ports.vvp
Reading io_ports.hex
io_ports.hex loaded into memory
Memory 5 bytes = 0x6f 0x00 0x00 0x0b 0x13
FST info: dumpfile io_ports.vcd opened for output.
MPRJ-IO state = zz zz : zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz1zzz
MPRJ-IO state = db 9f : zzzzzzzzzzzzzz1001111111011011zzzz1zzz
MPRJ-IO state = e0 a0 : zzzzzzzzzzzzzz1010000011100000zzzz1zzz
MPRJ-IO state = 32 a1 : zzzzzzzzzzzzzz1010000100110010zzzz1zzz
...
MPRJ-IO state = b0 fc : zzzzzzzzzzzzzz1111110010110000zzzz1zzz
MPRJ-IO state = 54 fd : zzzzzzzzzzzzzz1111110101010100zzzz1zzz
MPRJ-IO state = bb fe : zzzzzzzzzzzzzz1111111010111011zzzz1zzz
MPRJ-IO state = 16 ff : zzzzzzzzzzzzzz1111111100010110zzzz1zzz
Monitor: Test 1 Mega-Project IO (RTL) Passed
io_ports_tb.v:208: $finish called at 494862500 (1ps)
mv io_ports.vcd RTL-io_ports.vcd
rm io_ports.vvp
```

User領域のGDSファイル生成

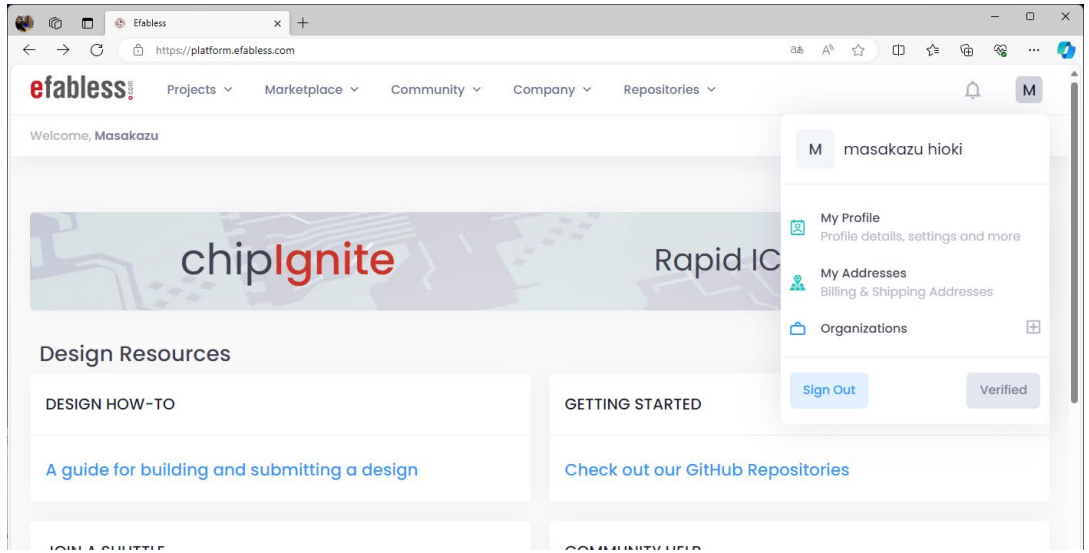
Example counter circuit + our sbox

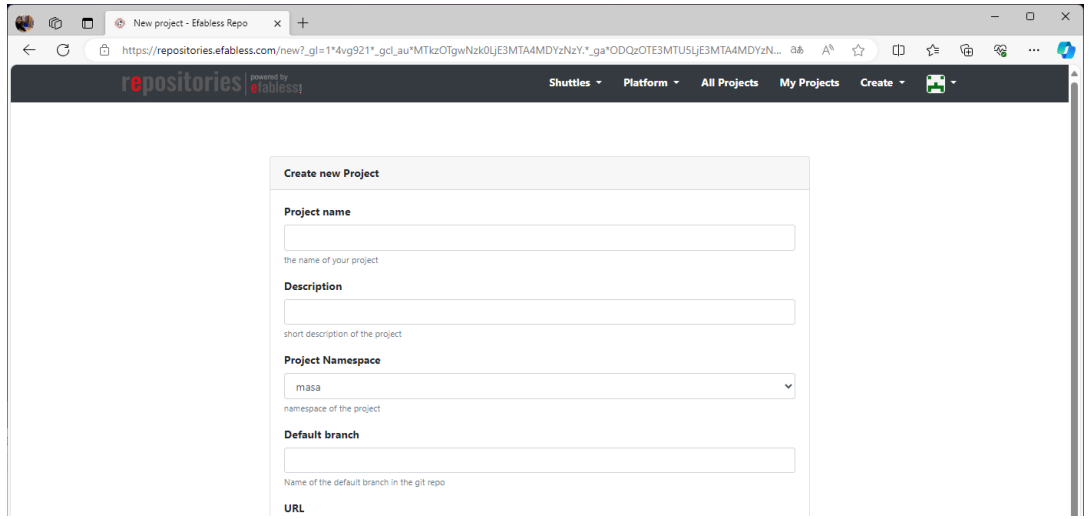


User area gds including our IP



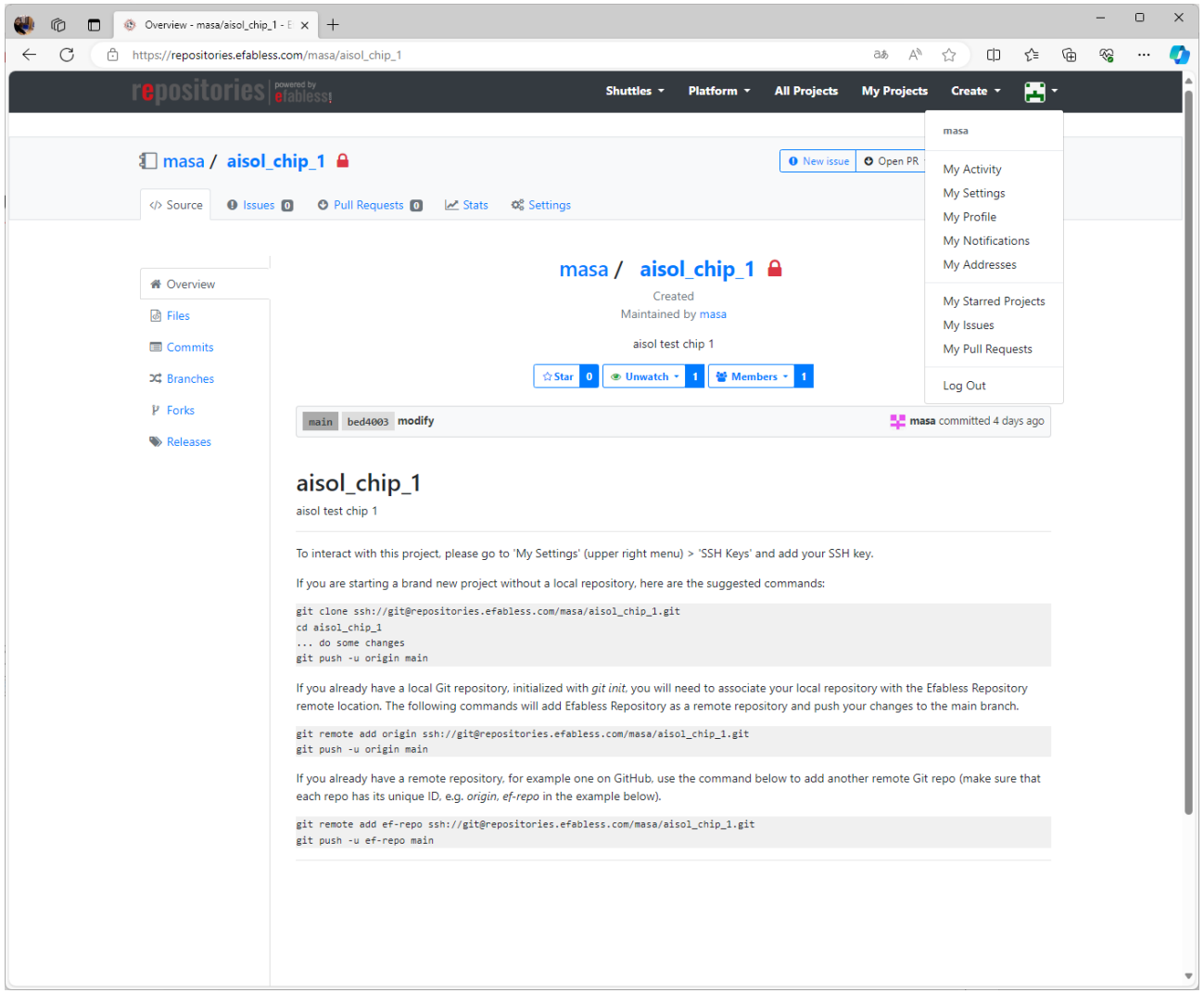
ChipIgniteシャトルにおけるデータサブミットへの道





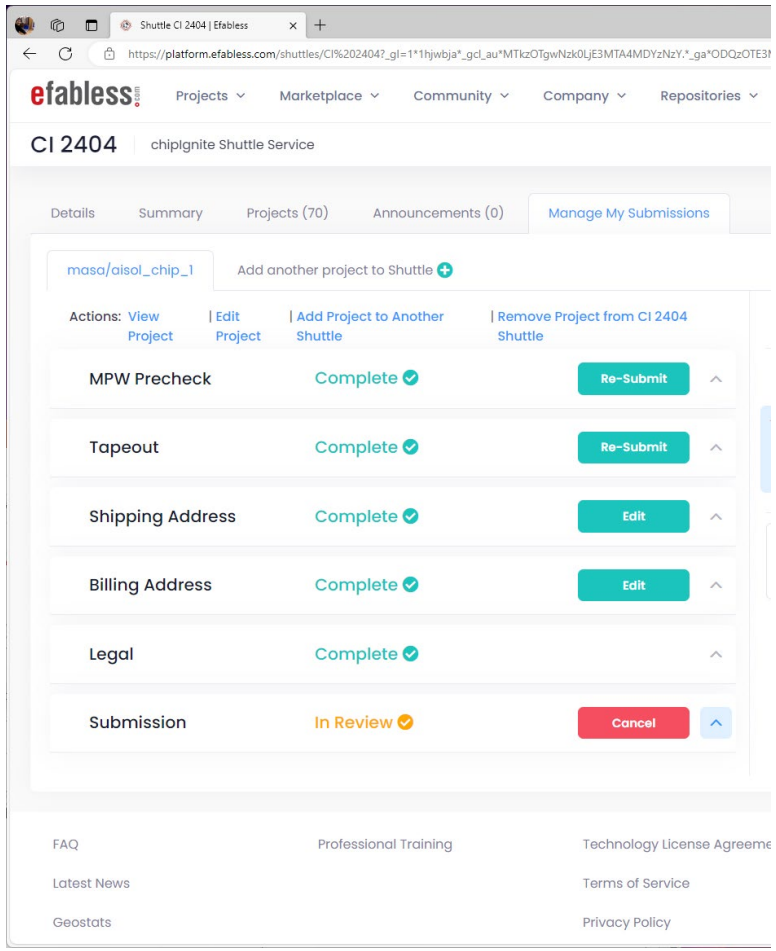
- アカウント作成　今回はaist-solutionsメールアドレスでLinkedInアカウントを作成し、efablessのサイトでアカウント作成
- ブラウザ右上のアイコンをクリックし、profileとbilling/shippingアドレスを作成
- ブラウザ右上のアイコンをクリックし、verificationを実行（数日で完了）
- レポジトリからCreate repositoryを選択しレポジトリ作成
- 今回は、project name, description, privateをチェック、create readmeをチェック、=>create
- gitの画面に移り、SSH key作成と登録、レポジト作成
- 画面上のシャトルをクリック、シャトルCI2404を選択

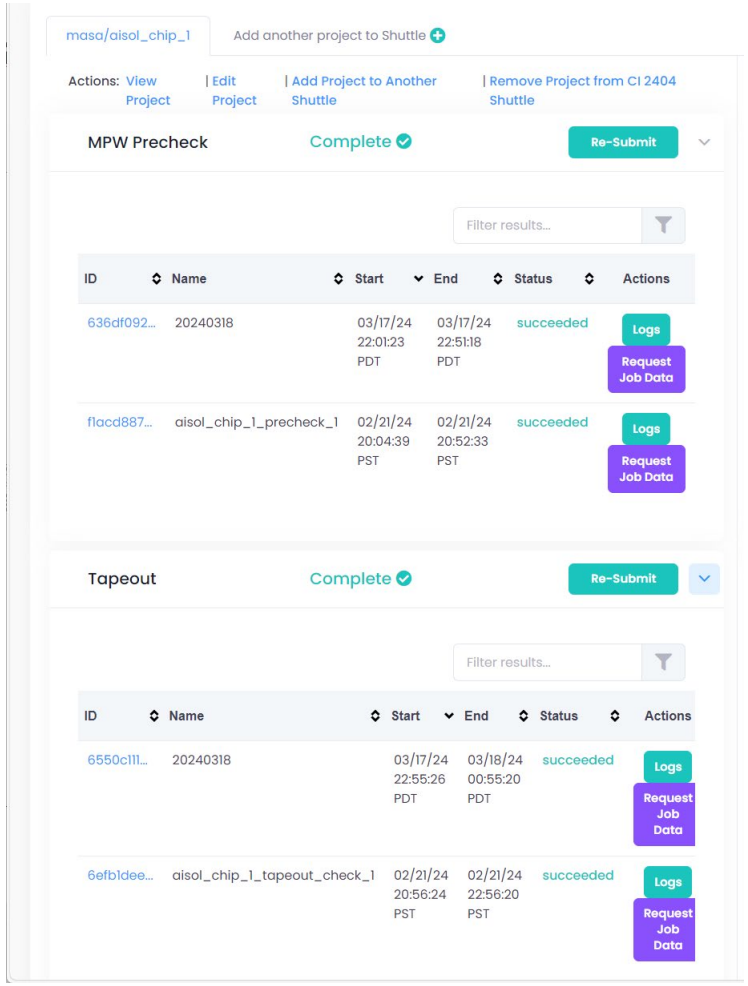
レポジトリ作成



- 右上のアイコンをクリック、Mysettingを選択
- 左のSSH Keysをクリック　ssh keysのページに移動
- 手順に従ってローカルマシンでssh keyを作成
- ページ右上のadd ssh keyをクリック、作成されたid_rsa.pubの中身をボックスにコピペしadd。
- 一番上の手順でレポジトリをクローン
- ローカルレポジトリにファイル追加
- git add, git commit, git push

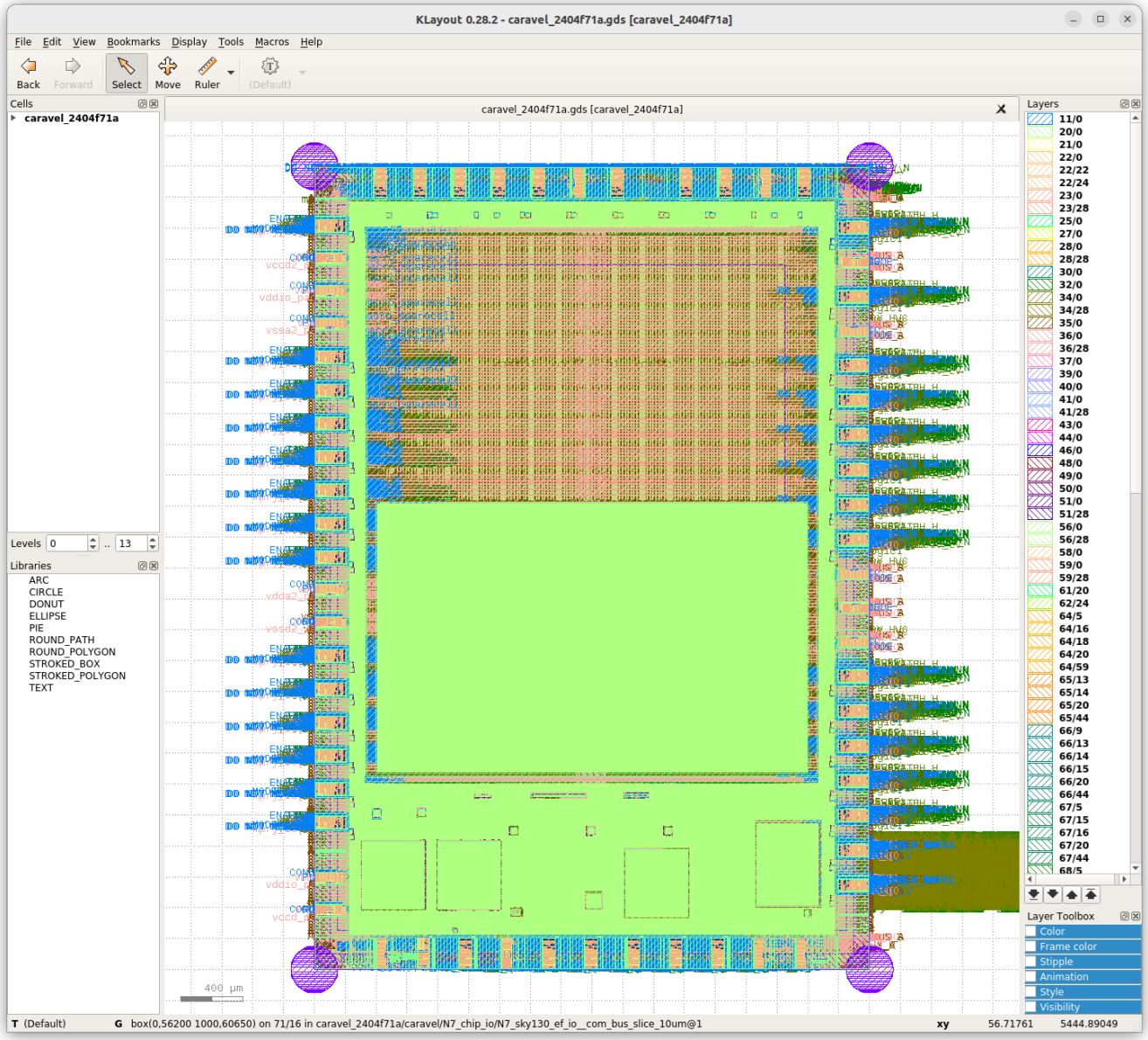
データサブミッションプロセス



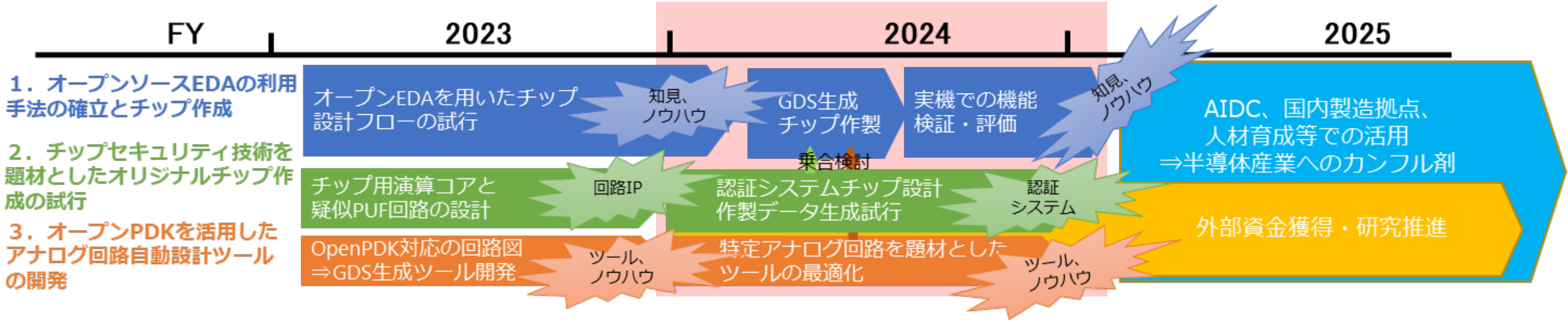


- Manage My Submissionsをクリック、シャトルにレポジトリを追加する。add this project to CI2404 shuttle
- 事前にローカルでプリチェック済みデータをプッシュし、再度プリチェックする。
- MPW precheckをサブミット、ジョブ名を入力・ジョブ実行
- テープアウトはMPW precheckのコンプリートデータを選択してジョブをサブミット
- シッピングアドレス、ビリングアドレスを編集。最初に入力したデフォルトをラジオボタンで選択可能。
- Legal: terms and conditions利用規約とexport compliance輸出管理に同意
- サブミッションで終了
- と思いきや、efablessによるデザインレビューが行われた

ファイナルGDS


KLayout 0.28.2 - caravel_2404f71a.gds [caravel_2404f71a]
File Edit View Bookmarks Display Tools Macros Help
Back Forward Select Move Ruler (Default)
Cells
caravel_2404f71a
Levels 0 .. 13
Libraries
ARC
CIRCLE
DONUT
ELLIPSE
PIE
ROUND_PATH
ROUND_POLYGON
STROKED_BOX
STROKED_POLYGON
TEXT
caravel_2404f71a.gds [caravel_2404f71a]
400 µm
Layers
11/0
20/0
21/0
22/0
22/22
22/24
23/0
23/28
25/0
27/0
28/0
28/28
30/0
32/0
34/0
34/28
35/0
36/0
36/28
37/0
39/0
40/0
41/0
41/28
43/0
44/0
46/0
48/0
49/0
50/0
51/0
51/28
56/0
56/28
58/0
59/0
59/28
61/20
62/24
64/5
64/16
64/18
64/20
64/59
65/13
65/14
65/20
65/44
66/9
66/13
66/14
66/15
66/20
66/44
67/5
67/15
67/16
67/20
67/44
68/5
Layer Toolbox
Color
Frame color
Stipple
Animation
Style
Visibility
T (Default)
G box(0,56200 1000,60650) on 71/16 in caravel_2404f71a/caravel/N7_chip_io/N7_sky130_ef_io__com_bus_slice_10um@1
xy 56.71761 5444.89049

今後の展開



① Caravel統合環境を用いたサンプル回路のテープアウトと実チップでの性能評価
② HWセキュリティチップの個別回路IPの設計完了とそれらの統合、レイアウトデータ生成
③ オペアンプ回路を題材として性能仕様から回路ネットリストを生成するツールの開発
④ 得られた知見の公開
⑤ ChipIgniteシャトルIP試作チップの評価

まとめ

背景と問題点

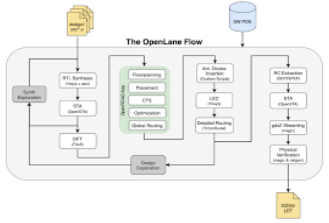
- “誰でも手が届く半導体”に向けたオープンソースEDAの世界的な潮流
- 国内の利用事例が少なく情報が不足気味だったが、増え始めてきた

本研究の目的と想定するアウトカム

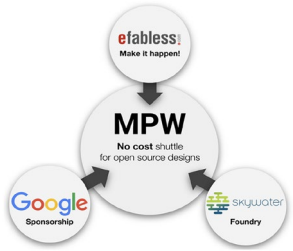
- オープンソースEDAの利用方法等の知見を蓄積・公開し普及促進を図る
- 期待する効果：国内半導体の活性化や人材育成、SCR用PDK開発や設計フロー構築

半導体化のハードルを
下げる取り組み@dapa

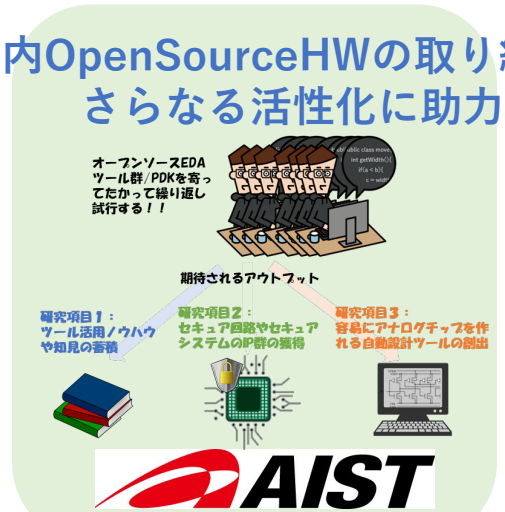
Open Source EDA flow



Open Shuttle Program



国内OpenSourceHWの取り組みの
さらなる活性化に助力



期待する効果

- 国内半導体の活性化
- 人材育成

- SCR用PDK開発
- 設計フロー構築

以降、予備スライド

で、結局オープンソースEDAはどうなのか？

- とても気軽/手軽にVerilogからGDSを生成できる
- オリジナルチップに手が届きそう
- テクノロジは130nmと180nmしかない
  - 以前、90nmのアナウンスはあったが音沙汰ない
  - 先端プロセスにも対応できるみたいなことは書いてあったが、そのためにどれくらいの作業量が必要か未知数
- ツールはマルチスレッドに対応していないので、スピードアップの余地あり
- 階層設計の深さは２まで。
- ハードマクロの余白が意外と大きい印象（検証の余地）

で、結局オープンソースEDAはどうなのか？

- ピン数の多いデザインではピン数律速で物理面積が大きくなる（検証の余地）
- 実用性のあるデザインでPPAを評価したい
- Caravelハーネスに内蔵のロジアナ、wishbornバス、RISC-Vの情報が少ない（実チップで確認）
- 有償シャトルプログラムChipIgnite
  - ユーザーサポートの反応が極めて鈍い
  - デザインレビューは（個人的に）大変だった