SMT-based Simultaneous Standard Cell Place-&-Route (SP&R)

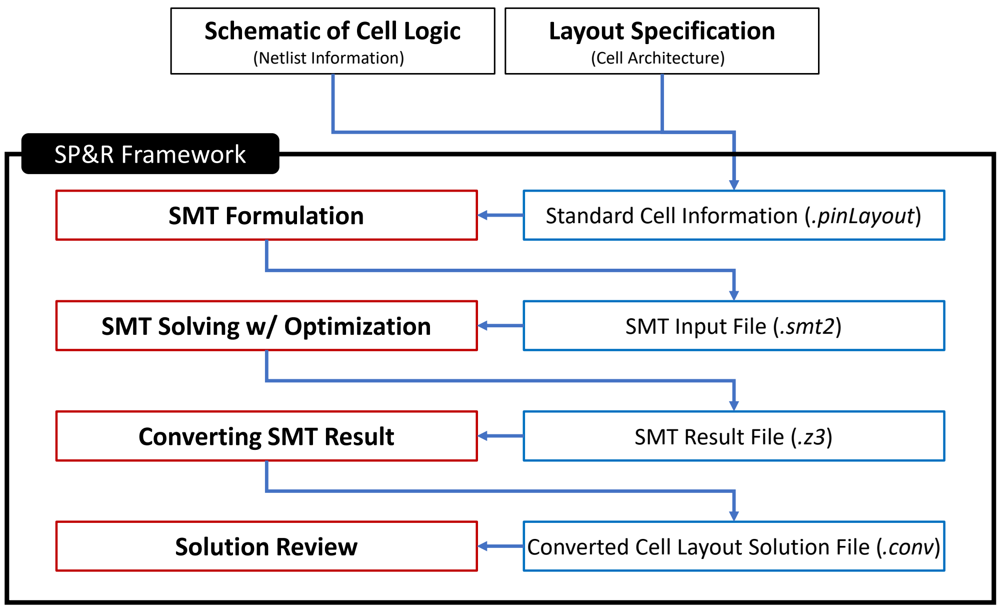
CK Research Group in UC San Diego

Email: ckcheng@ucsd.edu

1. Overview

This manual briefly summarizes the following flows to generate (i) SMT formulation file (*.smt2* file) and (ii) solution files to review the cell layout result. With the given standard cell information inputs (*.pinLayout* file) which are extracted from the ASAP7 PDK library[1], our flow generates the SMT formulation. We provide a solution viewer to validate the transistor placement and in-cell routing result of the SMT formulation. We employ Z3 (*Ver. 4.8.5*) [2] as our SMT solver. Please find more details from our paper [3].

(1) Flow Chart for Our Proposed Framework



(2) Contents in the TAR ball

```
(Current Path) ─ /SPNR/cmd_gen_smt
                      /cmd_conv_solution
                      /pinLayouts/A2O1A1Ixp33.pinLayout … XOR2xp5.pinLayout
                      /inputsSMT
                      /RUN
                      /scripts/convSMTResult_Ver1.0.pl
                              /genSMTInput_SPNR_Ver1.0.pl
                      /solutionsSMT
```

2. Our Tool-Chain Scripts and Commands with User-Specified Options

Our tool-chain scripts are written in *Perl*. SMT solver is Z3 (*Ver. 4.8.5*). For the information of the Z3 solver, please visit the following link: https://github.com/Z3Prover/z3

** Z3 Solver has been frequently updated. We recommend to use the specific version V4.8.5*

(1) Input Standard Cell Information (`.pinlayout`)

We provide 183 standard cell information which are extracted from the ASAP7 PDK library [1]. The list of standard cells is as follows.

A2O1A1Ixp33	AOI211x1	BUFx6f	INVx5	NOR5xp2	OAI32xp33
A2O1A1O1Ixp25	AOI211xp5	BUFx8	INVx6	O2A1O1Ixp33	OAI331xp33
AND2x2	AOI21x1	DFFHQNx1	INVx8	O2A1O1Ixp5	OAI332xp33
AND2x4	AOI21xp33	DFFHQNx2	INVxp33	OA211x2	OAI333xp33
AND2x6	AOI21xp5	DFFHQNx3	INVxp67	OA21x2	OR2x2
AND3x1	AOI221xp5	DFFHQx4	MAJIxp5	OA221x2	OR2x4
AND3x2	AOI222xp33	DFFLQNx1	MAJx2	OA222x2	OR2x6
AND3x4	AOI22x1	DFFLQNx2	MAJx3	OA22x2	OR3x1
AND4x1	AOI22xp33	DFFLQNx3	NAND2x1p5	OA31x2	OR3x2
AND4x2	AOI22xp5	DFFLQx4	NAND2x1	OA331x1	OR3x4
AND5x1	AOI311xp33	DHLx1	NAND2x2	OA331x2	OR4x1
AND5x2	AOI31xp33	DHLx2	NAND2xp33	OA332x1	OR4x2
AO211x2	AOI31xp67	DHLx3	NAND2xp5	OA332x2	OR5x1
AO21x1	AOI321xp33	DLLx1	NAND2xp67	OA333x1	OR5x2
AO21x2	AOI322xp5	DLLx2	NAND3x1	OA333x2	SDFHx1
AO221x1	AOI32xp33	DLLx3	NAND3x2	OA33x2	SDFHx2
AO221x2	AOI331xp33	FAx1	NAND3xp33	OAI211xp5	SDFHx3
AO222x2	AOI332xp33	HAxp5	NAND4xp25	OAI21x1	SDFHx4
AO22x1	AOI333xp33	HB1xp67	NAND4xp75	OAI21xp33	SDFLx1
AO22x2	AOI33xp33	HB2xp67	NAND5xp2	OAI21xp5	SDFLx2
AO31x2	ASYNC_DFFHx1	HB3xp67	NOR2x1p5	OAI221xp5	SDFLx3
AO322x2	BUFx10	HB4xp67	NOR2x1	OAI222xp33	SDFLx4
AO32x1	BUFx12f	ICGx1	NOR2x2	OAI22x1	TIEHIx1
AO32x2	BUFx12	ICGx2	NOR2xp33	OAI22xp33	TIELOx1
AO331x1	BUFx16f	ICGx3	NOR2xp67	OAI22xp5	XNOR2x1
AO331x2	BUFx24	INVx11	NOR3x1	OAI311xp33	XNOR2x2
AO332x1	BUFx2	INVx13	NOR3x2	OAI31xp33	XNOR2xp5
AO332x2	BUFx3	INVx1	NOR3xp33	OAI31xp67	XOR2x1
AO333x1	BUFx4f	INVx2	NOR4xp25	OAI321xp33	XOR2x2
AO333x2	BUFx4	INVx3	NOR4xp75	OAI322xp33	XOR2xp5
AO33x2	BUFx5	INVx4			

(2) SMT Formulation Generation (`genSMTinput_SPNR_Ver1.0.pl`)

[Usage]

```
$ ./scripts/genSMTInput_Ver1.0.pl [inputfile_pinLayout] [MAR] [EOL] [VR] [PRL]
[SHR] [MPO] [DBMode] [FST] [CellPartition] [CrosstalkML] [Localization] [Tolerance]
[BreakingSymmetry] [ObjPartition]
```

* [inputfile_pinLayout] : path for input pinLayout (ex: pinLayouts/AND2x2.pinLayout)

* [MAR] : Minimum Area Rule parameter (integer)

* [EOL] : End-of-Line Rule parameter (integer)

* [VR] : VIA Rule parameter (float)

* [PRL] : Parallem Run-Length Rule parameter (integer)

* [SHR] : Step Heights Rule parameter (integer)

* [MPO] : Minimum Pin Opening parameter (integer)

* [DBMode] : Diffusion Break Mode – 0:single, 1:double, 2:mixed

→ To enable mixed SDB/DDB in crossover region, FET information in crossover region should be specified in pinLayout inputs. Please refer to the sample pinLayout (DFFHQNx1.pinLayout). The FET information in crossover region is described in "i ===SDBCellInfo===" section.

* [FST] : FET Size Transition – 0:disable, 1:enable

* [CellPartition] : Cell Partitioning – 0:disable, 1:enable

→ To enable Cell Partitioning Feature, partitioning information should be specified in pinLayout inputs. Please refer to the sample pinLayout (DFFHQNx1.pinLayout). The partitioning info is described in "i ===PartitionInfo===" section.

* [CrosstalkML] : Metal Length Limit for Crosstalk Mitigation

→ To enable Crosstalk Mitigation Feature, special care net information should be specified in pinLayout inputs. Please refer to the sample pinLayout (DFFHQNx1.pinLayout). The net information for crosstalk mitigation is described in "i ===SpecialNetInfo===" section.

* [Localization] : Localization – 0:disable, 1:enable

* [Tolerance] : Offset Margin for Localization (integer)

* [BreakingSymmetry] : Breaking Symmetry – 0:disable, 1:enable

* [ObjPartition] : Objective Partitioning – 0:disable, 1:enable

* Please refer our paper [3] for further detailed information of each input parameters.

* Cell Partitioning and Breaking Symmetry options can not be used at the same time.

[Example]

Generating the SMT formulation file (*.smt2*) for the AND2x2 standard cell "`AND2x2.pinLayout`" with the design rule parameters used in [3].

```
$ ./scripts/genSMTInput_SPNR_Ver1.0.pl pinLayouts/AND2x2.pinLayout 1 1 1.5 1 1 3 2
1 0 5 1 1 1 1
```

This will create "`AND2x2.smt2`" file in the `inputsSMT` directory. For the .smt2 file format, please visit the following link: https://rise4fun.com/z3/tutorialcontent/guide

* In our work [3], we set different parameters for combinational and sequential logic cells because we only applied the cell partitioning and crosstalk mitigation features to the sequential logic cells. Please refer to the pre-described command list (`cmd_gen_smt`) for the parameters applied to each cell in [3].

(3) RUN SMT Solver (`Z3`)

[Usage]

```
SMT Solving & Storing solution
$ z3 inputsSMT/[inputFile(.smt2)] > RUN/[solutionName(.z3)]
```

[Example]

Running "`AND2x2.smt2`" file and storing the result "`AND2x2.z3`" to the output directory

```
$ z3 inputsSMT/AND2x2.smt2 > RUN/AND2x2.z3
```

(4) Solution Converter (`convSMTResult_Ver1.0.pl`)

[Usage]

```
$ ./scripts/convILPResult_Ver1.0.pl [solPath/solutionName] [inputFile_pinLayout(w/o
file extension)]"
```

[Example]

Converting "`AND2x2.z3`" output file generated from the input pinLayout "`AND2x2.pinLayout`" to the solution output directory

```
$ ./scripts/convSMTResult_Ver1.0.pl RUN/AND2x2.z3 AND2x2
```

This will create "`[solutionName].conv`" file in the `solutionsSMT` directory.

The converted solution files (*.conv*) can be reviewed using an excel-based solution viewer. (`SolutionViewer_3F_6T.xlsm`)

(5) Pre-described Command Lists

There are "`cmd_conv_solution`", "`cmd_gen_smt`" files which consist of command lists to generate and convert the whole standard cells provided in this package. You can refer to these command file to modify the parameters or execute each cell generation or sourcing the list file to execute all cases.

3. References

[1] V. Vashishtha, M. Vangala, and L. T. Clark, "ASAP7 predictive design kit development and cell design technology co-optimization," in 2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 992–998, IEEE, 2017

[2] *Z3*, SMT Solver, https://github.com/Z3Prover/z3.

[3] D. Lee, D. Park, C.-T. Ho, I. Kang, H. Kim, S. Gao, B. Lin, C.-K. Cheng, "SP&R: SMT-based Simultaneous Place- &- Route for Standard Cell Synthesis of Advanced Nodes", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2020