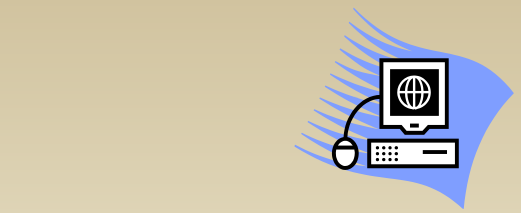
EE6094
CAD for VLSI Design

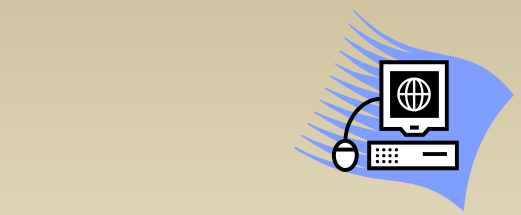
PA3
Analog Placement

Spring 2024
Andy, Yu-Guang Chen
Assistant Professor, Department of EE
National Central University
andyygchen@ee.ncu.edu.tw
Slides Credit: TA Yu-Heng Tsao
Most Materials Courtesy of Prof. Mark, Po-Hung Lin (NYCU)



Outline

- Introduction
- Grading Policy
- Suggestions

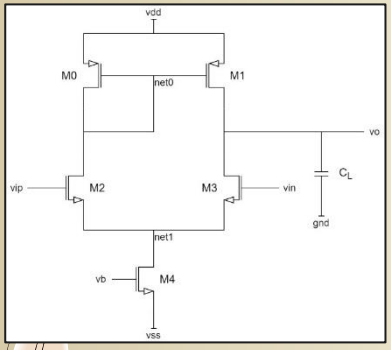


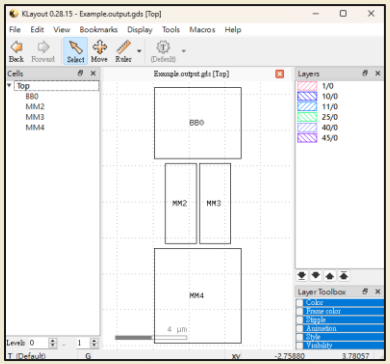
Outline

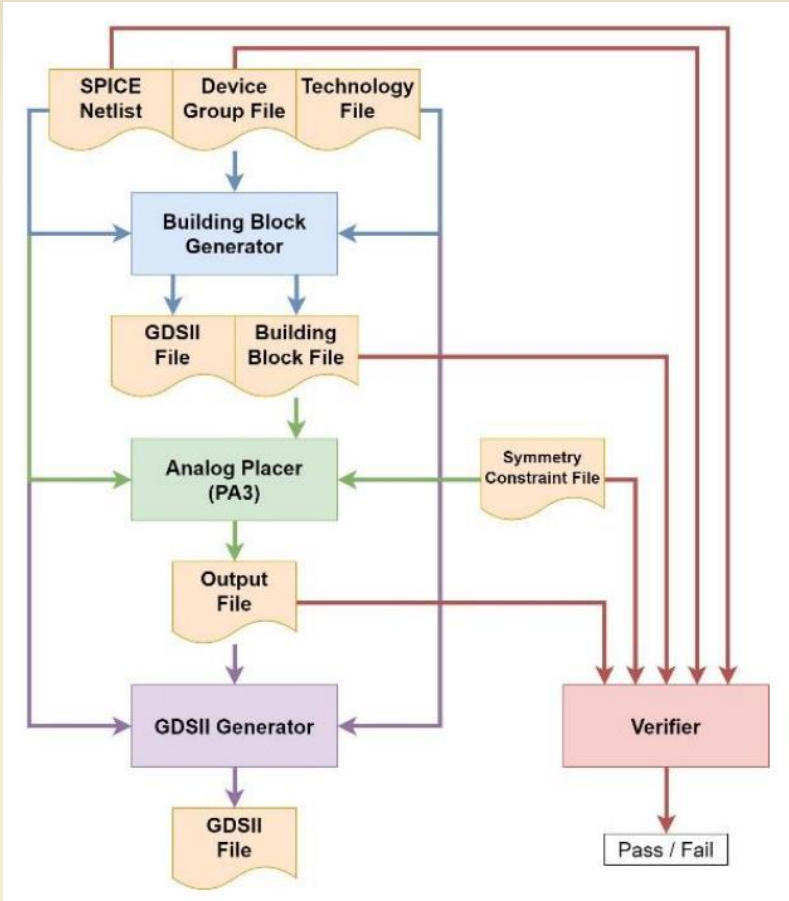
- Introduction
  - Problem Description
  - Workflow
  - Scripts
  - Input Files
  - Intermediate Files
  - Output Files
- Grading Policy
- Suggestions

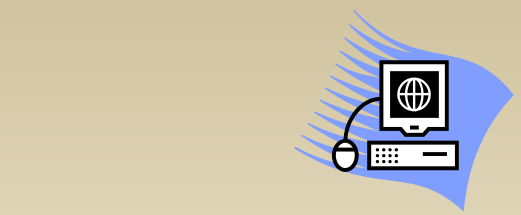
Problem Formulation

- Input
  - A circuit netlist
  - Some constraints
- Output
  - A legal and optimized placement result that describes the location, pattern, and other information of the devices







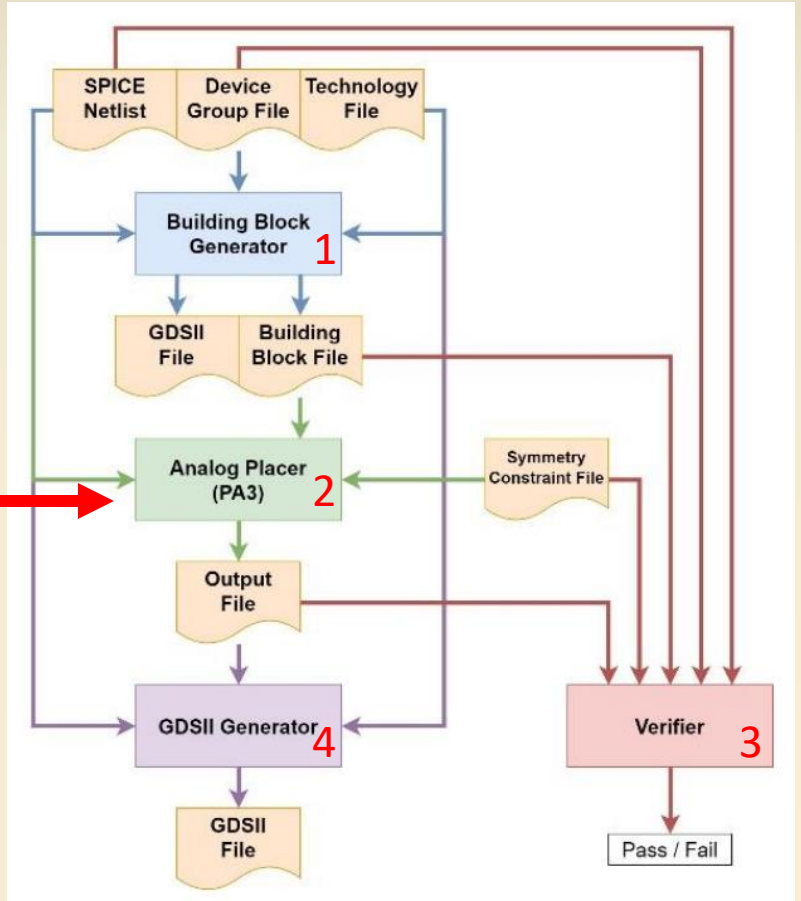


Outline

- Introduction
  - Problem Description
  - Workflow
  - Scripts
  - Input Files
  - Intermediate Files
  - Output Files
- Grading Policy
- Suggestions

Workflow

- We provide three scripts to assist you to solve this problem
  - BuildingBlockGenerator
  - Verifier
  - GdsGenerator
- You are asked to implement your own analog placer
  - B*-tree
  - Sequence Pair



Outline

- Introduction
  - Problem Description
  - Workflow
  - Scripts
    - BuildingBlockGenerator
    - Verifier
    - GdsGenerator
  - Input Files
  - Intermediate Files
  - Output Files
- Grading Policy
- Suggestions

BuildingBlockGenerator

- BuildingBlockGenerator
  - A script that assists in handling merging device constraints and generates all building block patterns for each device
  - To meet minimum spacing design rules, all blocks will be oversized
  - A GDSII file will be generated for visualization
- Usage (Input/Output)
  1. $ BuildingBlockGenerator <netlist_file> <tech_file> <device_group_file> <building_block_file>
  2. Use KLayout to open the generated GDSII file to check the results

Verifier

- Verifier
  - A script for debugging and evaluating placement results
  - Precision is 6 decimal places
- Usage (Input/Output)
  - $ Verifier <expected_aspect_ratio> <netlist_file> <device_group_file> <symmetry_constraint_file> <building_block_file> <output_file>

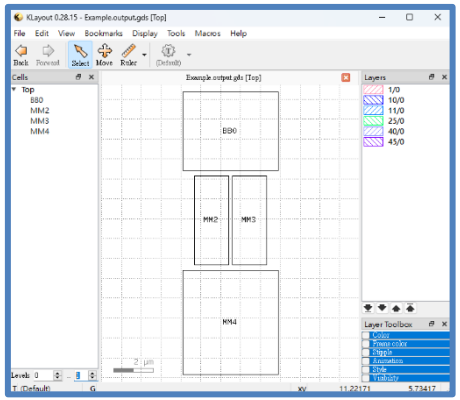
```
**************************************************************************************************
* Check whether the calculation results of the device info. match your own calculation results or not *
**************************************************************************************************
*                          From Device Info.                  From Your Output                   *
*    1. Total HPWL         17.92                              17.92                              *
*    2. Chip Area          72.4548                            72.4548                            *
*    3. Chip Width         4.99                               4.99                               *
*    4. Chip Height        14.52                              14.52                              *
**************************************************************************************************

Correctness:               PASS
Meet Symmetry Constraints: PASS
Cost                       46.534078
```

Screenshot of Verifier execution results

GdsGenerator

- GdsGenerator
  - A script that helps you visualize placement results
- Usage (Input/Output)
  1. $ GdsGenerator <netlist_file> <tech_file> <output_file> <gdsii_file>
  2. Use KLayout to open the generated GDSII file to check the results



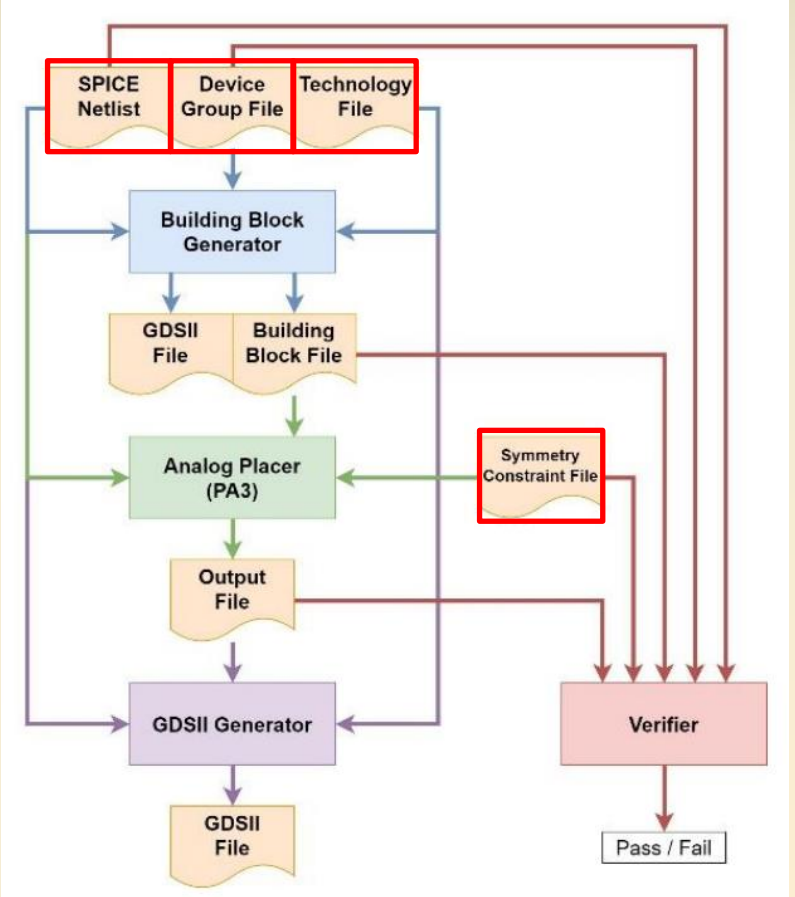
GDSII (KLayout)

```
17.92
72.4548
4.99 14.52
BB0 MM0 MM1 0.0 10.28 (4.99 4.24 4 1)
MM2 0.595 5.54 (1.9 4.74 1 1)
MM3 2.495 5.54 (1.9 4.74 1 1)
MM4 0.0 0.0 (4.99 5.54 4 1)
```

Example.output

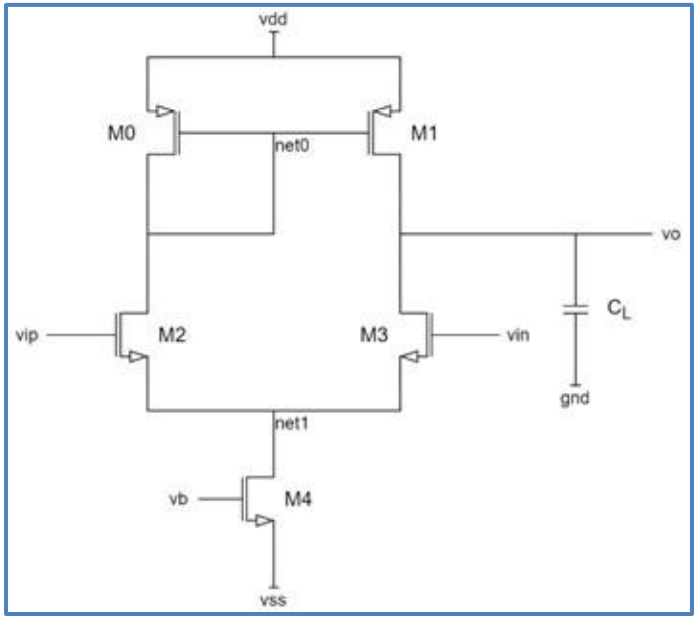
Outline

- Introduction
  - Problem Description
  - Workflow
  - Input Files
    - SPICE Netlist
    - Simplified Technology File
    - Device Groups
    - Symmetry Constraints
  - Intermediate Files
  - Output Files
  - Verifier
- Grading Policy
- Suggestions



SPICE Netlist

- Parse the connectivity (D, G, S) of devices from input netlist



Schematic

```
.SUBCKT EXAMPLE Vb vin vip vo gnd! vdd! vss!
MM0 net0 net0 vdd! vdd! PMOS W=3.5u L=0.5u m=2
MM1 vo net0 vdd! vdd! PMOS W=3.5u L=0.5u m=2
MM2 net0 vip net1 vss! NMOS W=4u L=0.5u m=1
MM3 vo vin net1 vss! NMOS W=4u L=0.5u m=1
MM4 net1 vb vss! vss! NMOS W=4.8u L=0.5u m=4
CL vo gnd! 1p
.ENDS
```

Example.netlist

Simplified Technology File

- Device rules
  - For building block generation
- Layer table
  - For the GDSII file creation

```
deviceRule {
   minPoly2GateExtension = 0.23
   minCo2GateSpace = 0.15
   minCoWidth = 0.23
   minCoSpace = 0.25
   minCo2OdEnclosure = 0.12
   minOdSpace = 0.4
   minPolySpace = 0.28
   minOd2ImpEnclosure = 0.12
   minCo2MetEnclosure = 0.1
}

layerTable {
   Diffusion = 1
   Poly = 25
   Contact = 40
   Pplus = 10
   Nplus = 11
   Metal = 45
}
```

Process.tf

Device Groups

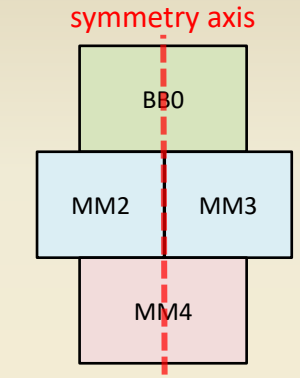
- Guide devices into building blocks so that they move together during the placement process
- Device group file format
  - <building_block_name> <all_device_name> <col_multiple> <row _multiple>

BB0 MM0 MM1 4 1

DeviceMerging.group

Symmetry Constraints

- Self-symmetric module format
  - <symmetry_group_name> <building_block_name>
- Symmetry pair format
  - <symmetry_group_name> <building_block_name_1> <building_block_name_2>



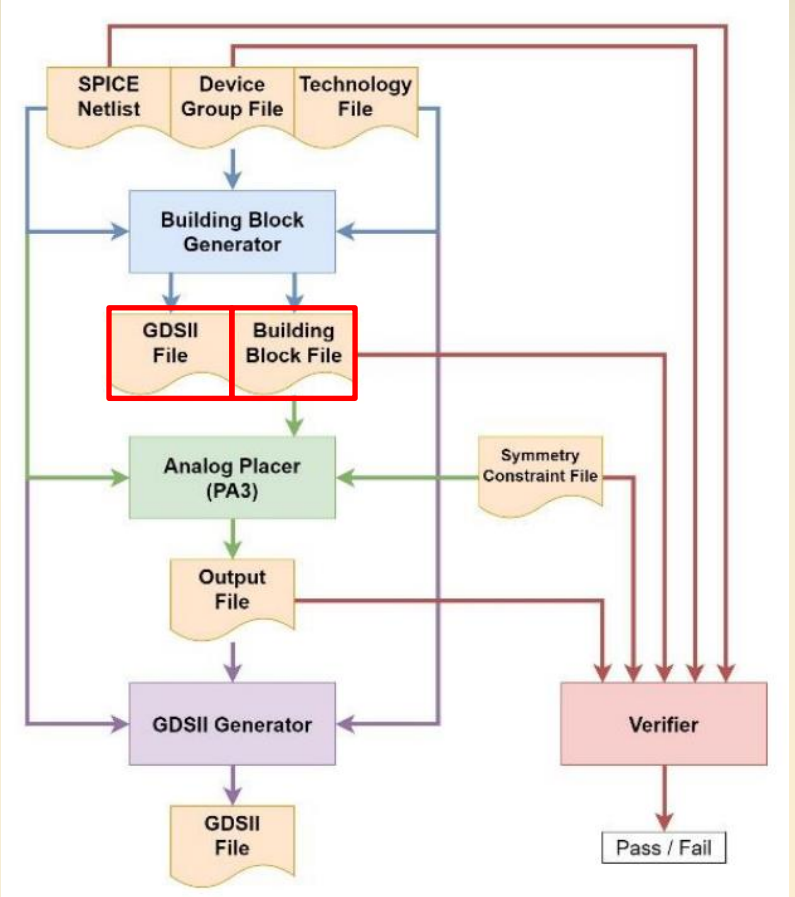
One of feasible placement for Symmetry0

```
Symmetry0 BB0
Symmetry0 MM2 MM3
Symmetry0 MM4
```

TopologicalConstraint.sym

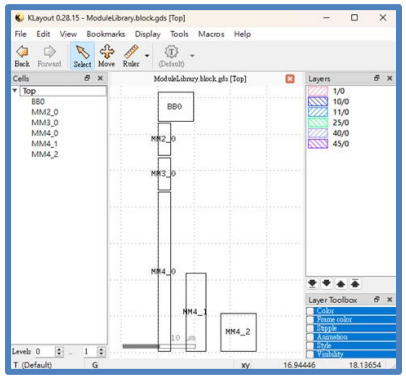
Outline

- Introduction
  - Problem Description
  - Workflow
  - Input Files
  - Intermediate Files
    - Building Block Patterns
  - Output Files
  - Verifier
- Grading Policy
- Suggestions



Building Block Patterns

- Building block file
  - A file generated by the script BuildingBlockGenerator
- Building block file format
  - <building_block_name> <all_device_names> (<width> <height> <col_multiple> <row_multiple>) …
  - <device_name> (<width> <height> <col_multiple> <row_multiple>) …



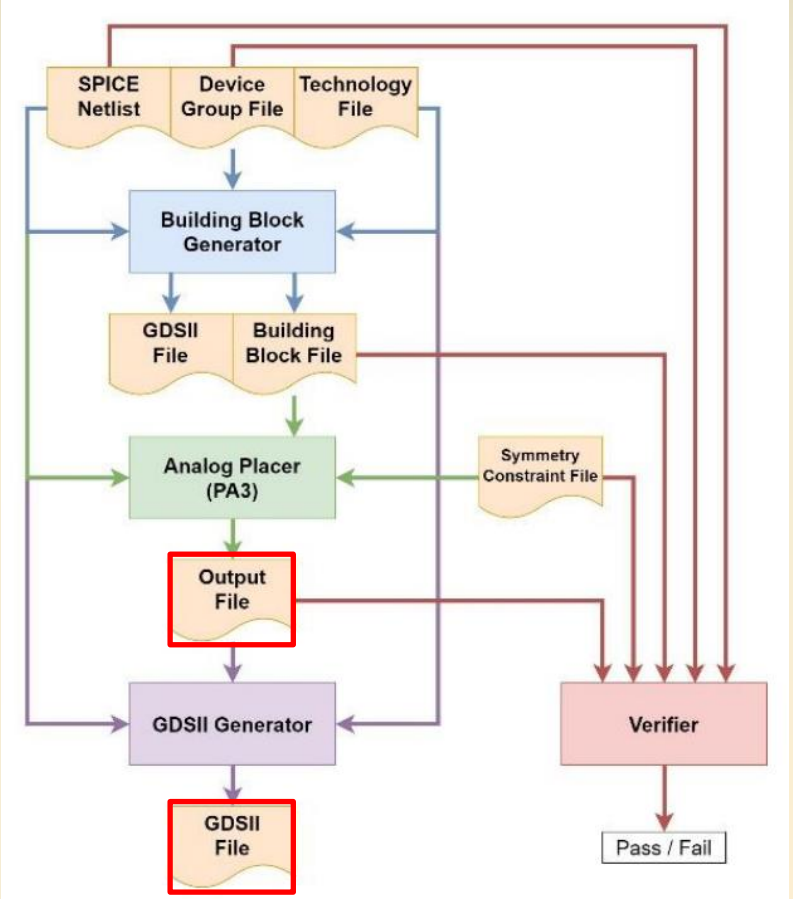
GDSII (KLayout)

```
BB0 MM0 MM1 (4.99 4.24 4 1)
MM2 (1.9 4.74 1 1)
MM3 (1.9 4.74 1 1)
MM4 (1.9 22.16 1 4) (2.93 11.08 2 2) (4.99 5.54 4 1)
```

ModuleLibrary.block

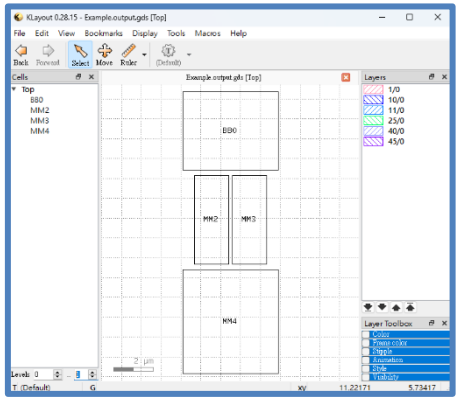
Outline

- Introduction
  - Problem Description
  - Workflow
  - Input Files
  - Intermediate Files
  - Output Files
    - Output File/GDSII File
- Grading Policy
- Suggestions



Output File/GDSII File

- Describe placement results
- Output file format
  - Line 1 to 3
    - Total HPWL, chip area, chip width, chip height
  - Line 4 to end:
    - <building_block_name> <all_device_names> <x> <y> (<width> <height> <col_multiple> <row_multiple>)



GDSII (KLayout)

```
17.92
72.4548
4.99 14.52
BB0 MM0 MM1 0.0 10.28 (4.99 4.24 4 1)
MM2 0.595 5.54 (1.9 4.74 1 1)
MM3 2.495 5.54 (1.9 4.74 1 1)
MM4 0.0 0.0 (4.99 5.54 4 1)
```

Example.output

Outline

- Introduction
- Grading Policy
  - Correctness (30%)
  - Quality (40%)
  - Readability (5%)
  - Report (10%)
  - Demo Session (25%)
- Suggestions

Correctness

- Correctness
  - Each test case will be scored independently
  - If the “Correctness” field is “PASS”, you will receive all correctness points

```
*************************************************************************************************
* Check whether the calculation results of the device info. match your own calculation results or not *
*************************************************************************************************
*                            From Device Info.                  From Your Output                  *
*    1. Total HPWL           17.92                              17.92                             *
*    2. Chip Area            72.4548                            72.4548                           *
*    3. Chip Width           4.99                               4.99                              *
*    4. Chip Height          14.52                              14.52                             *
*************************************************************************************************

Correctness:                PASS
Meet Symmetry Constraints:  PASS
Cost                        46.534078
```

Screenshot of Verifier execution results

Outline

- Introduction
- Grading Policy
  - Correctness (30%)
  - Quality (40%)
  - Readability (5%)
  - Report (10%)
  - Demo Session (25%)
- Suggestions

Quality (1)

- Part 1
  - Each test case will be scored independently
  - If the “Correctness” field and the “Meet Symmetry Constraints” field are “PASS”, you will get all points

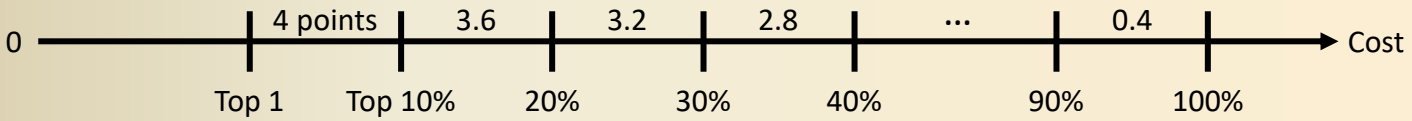
```
*************************************************************************************************
* Check whether the calculation results of the device info. match your own calculation results or not *
*************************************************************************************************
*                            From Device Info.                  From Your Output                  *
*    1. Total HPWL           17.92                              17.92                             *
*    2. Chip Area            72.4548                            72.4548                           *
*    3. Chip Width           4.99                               4.99                              *
*    4. Chip Height          14.52                              14.52                             *
*************************************************************************************************

Correctness:                PASS
Meet Symmetry Constraints:  PASS
Cost                        46.534078
```

Screenshot of Verifier execution results

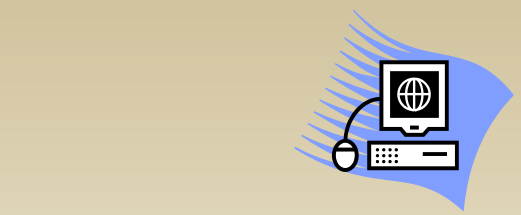
Quality (2)

- Optimization goal
  - The smaller the wire length, the better
  - The smaller the chip area, the better
  - The more the aspect ratio exceeds the expected value, the worse it is
  - For specific details, please refer to the cost function definition in the document
- Part 2
  - Each test case will be scored independently
  - If the “Correctness” field is “PASS”, the optimization points will be determined by the ranking of cost



Outline

- Introduction
- Grading Policy
  - Correctness (30%)
  - Quality (40%)
  - Readability (5%)
  - Report (10%)
  - Demo Session (25%)
- Suggestions



Readability

- With comments
- With meaningful names of functions and variables
- Organizing functions into separate files based on their functionality

Outline

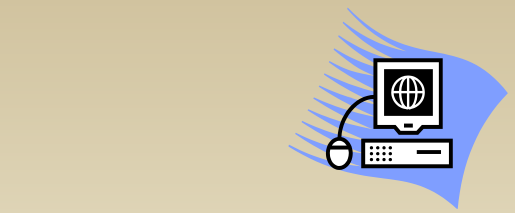
- Introduction
- Grading Policy
  - Correctness (30%)
  - Quality (40%)
  - Readability (5%)
  - Report (10%)
  - Demo Session (25%)
- Suggestions

Report

- In your report, you have to include at least
  - Compilation and execution
  - Completion
    - Screenshot of Verifier execution results
    - Screenshot of viewing placement results using KLayout
  - Concepts
    - Algorithms and data structures
    - Perturbation strategy and operations
    - Your cost function and how to determine whether to move to next state or not
  - Hardness
  - Suggestions
- We don’t restrict the report format and length
- English version is a plus
- The grading of the report will compare yours with others

Outline

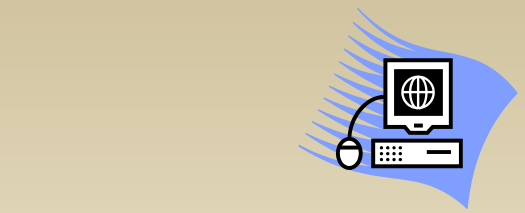
- Introduction
- Grading Policy
  - Correctness (30%)
  - Quality (40%)
  - Readability (5%)
  - Report (10%)
  - Demo Session (25%)
- Suggestions



Demo Session

- You must show up in the demo session
- Be familiar with your code
- Be familiar with the concepts in your report

Outline



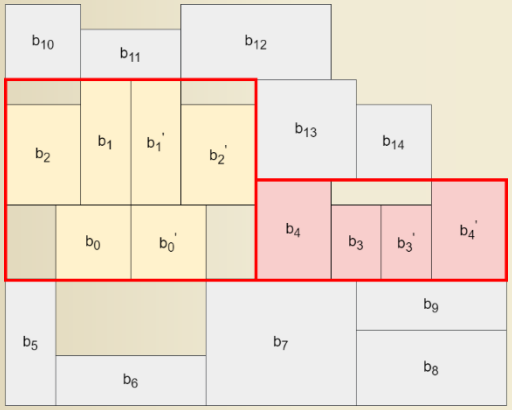
- Introduction
- Grading Policy
- Suggestions

Suggestions

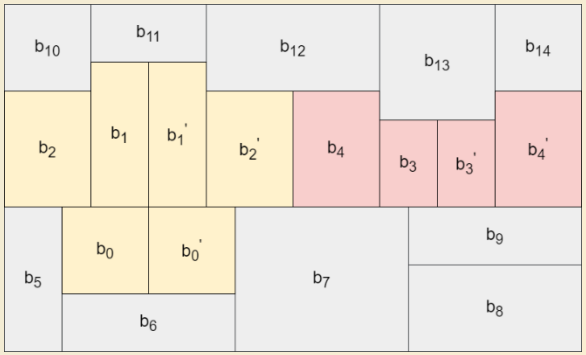
- Algorithm

	B*-tree	Sequence Pair
Difficulty	High	Low
Time Complexity	Low	High
Solution Space	Small	Large

- Symmetry constraints



Basic



Advanced

Q&A

- Email
  - adcarry00334@gmail.com

Andy, Yu-Guang Chen
Assistant Professor, Department of EE, NCU
Email: andyygchen@ee.ncu.edu.tw
FB: Yu-Guang Chen
IG: ncu.eda.andy
Google account: andyygchen.ncu
Slides credit: TA Yu-Heng Tsao