2025 中国研究生创“芯”大赛·EDA 精英挑战赛

一、赛题名称

FPGA 双输出 LUT 的逻辑映射算法

二、命题单位

深圳市紫光同创电子股份有限公司

三、赛题背景

当前先进的 FPGA（Field Programmable Gate Array，现场可编程逻辑门阵列）架构中通常使用双输出端口的 LUT(Look-Up-Table，查找表)结构（以下简称“双输出 LUT”），而传统的 LUT 映射算法只能将组合逻辑门映射到单输出端口的 LUT，给 FPGA 的综合工具带来一个新的挑战——如何才能更好的将组合逻辑映射到双输出 LUT 上。双输出 LUT 的两个输出端口共用部分输入，且两个输出之间需要满足一定的逻辑关系，难点在于映射过程中如何利用双输出 LUT 的结构并在代价函数中体现双输出 LUT 带来的影响，以及在映射结束后如何拼接 LUT 来最小化网表的总面积。

FPGA 设计流程如图 1 所示。本赛题属于 EDA（Electronic design automation，电子设计自动化）技术中的 FPGA 逻辑综合范畴。本题分基础题目和进阶题目两个部分。基础题目为给定已经映射好的单端口 LUT 网表，找到一个算法将两个 LUT 组合成一个双输出的 LUT，使得网表里最终的 LUT 数量尽可能少；进阶题目为如何修改传统的单端口 LUT 映射算法，直接映射到双输出 LUT，实现最少的 LUT 的数量。两个子题目分别属于逻辑综合的 Technology-dependent Optimization（工艺相关的优化，指利用特定工艺库对网表进行优化）和 Technology Map（工艺映射，指将通用的电路单元映射到特定的工艺库中的单元）两个部分。

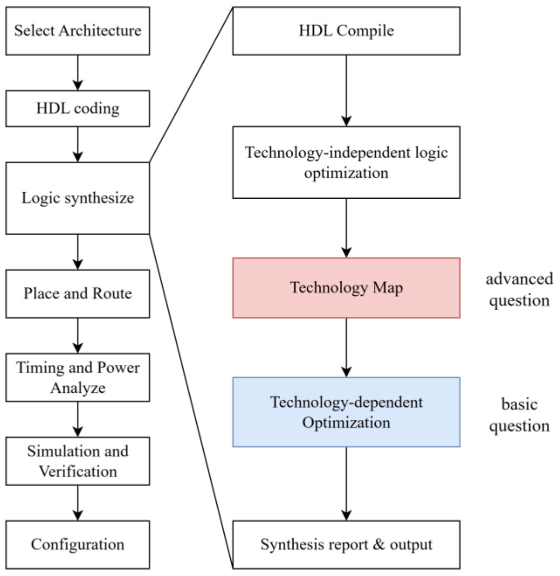

图 1 FPGA 的 EDA 设计流程

四、示例流程

紫光同创提供了一份基础代码。基础代码是基于开源项目 yosys 所编写，用于中国研究生创“芯”大赛-EDA 精英挑战赛。参赛者可以在这份代码基础上进行修改。图 2 给出了使用这份代码的示例流程。

紫光同创提供的代码、脚本、测试用例等存放于yosys/techlibs/pango。源代码文件synth_pango.cc中依据参考文献[1]实现了一个简单的映射算法，功能是将网表中的组合逻辑门映射到紫光同创的 LUT 原语，对文献中未提及的细节做了部分猜想和简化，可以在 yosys 工具下使用 synth_pango 命令来使用该功能。score.cc代码的功能是计算映射后网表的代价函数，可以使用 score 命令来使用该功能。

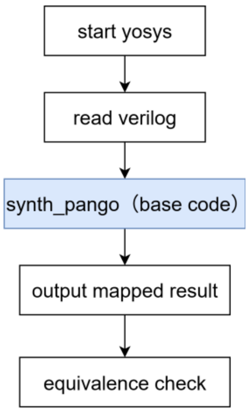

图 2 示例流程图

Yosys项目中包含两个非常关键的数据结构，SigBit和Cell。

SigBit在 yosys 中表示 wire 的其中一个比特，在逻辑综合过程中不关注 wire 的位宽和 wire 之间的 assign 关系，所以将互相连接的 wire 转为唯一的SigBit表示。代码中使用SigBit对象获取其 driver，reader，depth 等数据。

Cell表示网表中的某个实例化对象，综合过程中仅关注组合逻辑门的 cell，紫光同创提供的用例仅包含与门、或门、异或门、选择器、反相器类型的组合逻辑门，对于其他类型的 cell 可以不做任何处理，原封不动的输出。

五、双输出 LUT 架构介绍

GTP_开头的 cell 类型是紫光同创的 FPGA 原语，yosys/techlibs/pango/pango_sim.v文件中包含了部分原语的仿真模型。

GTP_LUT1~GTP_LUT6 是 n 输入 1 输出的查找表。

GTP_LUT6D 是 6 输入 2 输出的查找表，其仿真模型如图 3，内部结构如图 4。

在资源占用的估算中，GTP_LUT6D 和 GTP_LUT 是相等的，都会占用 FPGA 芯片中的一个查找表硬件资源，但是 GTP_LUT6D 除了能够实现 GTP_LUT6 的功能外，还能够额外实现 1 个 5 输入 1 输出的查找表，Z5 的逻辑表达式是 Z 逻辑表达式的子式，两个逻辑表达式的 I[4:0]是相同的。利用这个特点可以将输入相同或相似的两个

逻辑表达式拼接到一起使用一个GTP_LUT6D来输出两个信号,达到节省面积的目的。

```
// GTP_LUT6D  仿真模型
module GTP_LUT6D
#(
        parameter [63:0] INIT = 64'h0000_0000_0000_0000
) (
        output Z,
        output Z5,
        input I0,
        input I1,
        input I2,
        input I3,
        input I4,
        input I5
);
wire z5a,z5b;
$lut #(.WIDTH(5),.LUT(INIT[31:0])) luta_cell(.A({I4,I3,I2,I1,I0}),.Y(z5a));
$lut #(.WIDTH(5),.LUT(INIT[63:32])) lutb_cell(.A({I4,I3,I2,I1,I0}),.Y(z5b));
$mux #(.WIDTH(1)) u1(.A(z5a),.B(z5b),.S(I5),.Y(Z));
assign Z5 = z5a;
endmodule
```

图 3 GTP_LUT6D 仿真模型

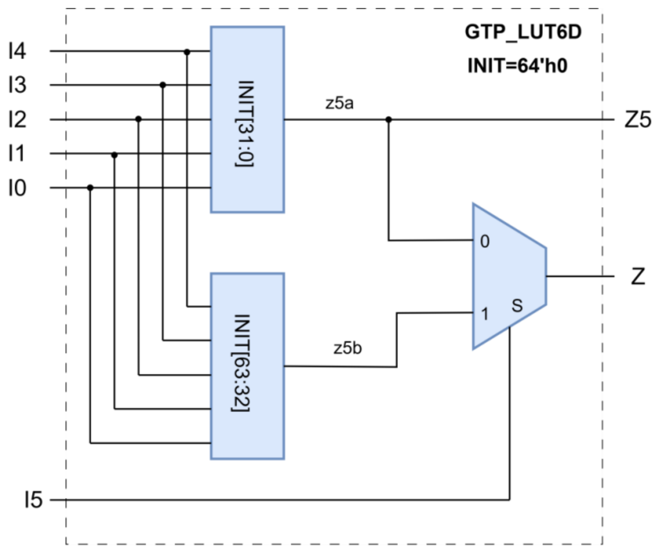

图 4 GTP_LUT6D 内部结构图

六、竞赛内容

竞赛内容包含两个方面：

（1）基础题

给定已经映射好的单端口 GTP_LUT 网表,设计算法和功能将网表中的 GTP_LUT 拼接成为 GTP_LUT6D，尽可能减少总 LUT 数量，即 GTP_LUT 和 GTP_LUT6D 的总数。

例如，网表中如果存在两个 GTP_LUT 的逻辑分别是O1 = A&B&D&E; O2 = A&B&C&F | A&B&D&E，则可以将这两个 GTP_LUT 拼接成为一个 GTP_LUT6D，拼接后网表的总 LUT 数量减少 1。

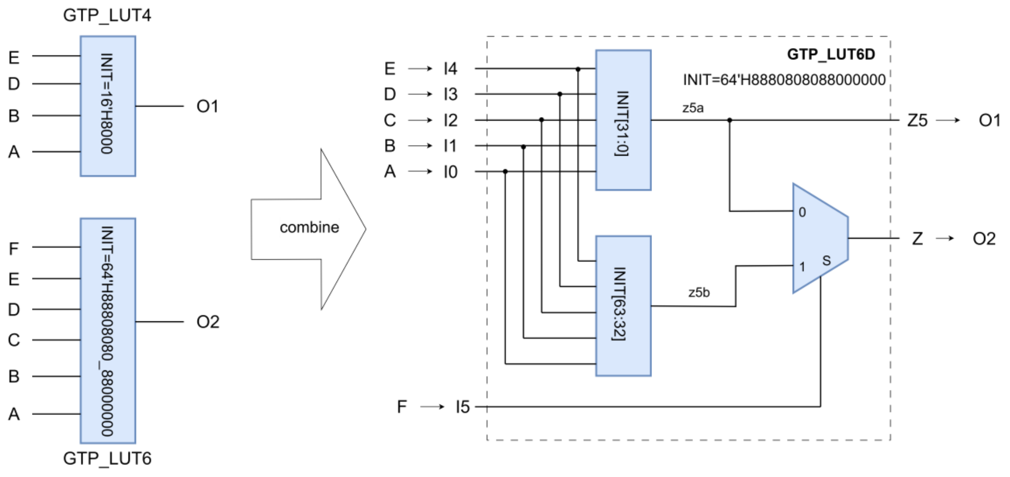

图 5 拼接 LUT 举例说明

（2）进阶题

在 LUT 映射算法层面支持直接映射到 GTP_LUT6D。参赛者可以修改或重新设计映射算法，实现直接映射出 GTP_LUT6D，使用最少的 LUT 总数。

参赛者必需提交程序可执行文件，以及必要的文档说明。为避免程序依赖特定的库导致的不能正常运行，建议提交源代码。

七、用例

紫光同创会提供至少 6 个测试用例和至少 10 个评分用例。赛前会公布测试用例，而评分用例不会在比赛结束前公开。用例之间的规模存在差异，没有难易区分。

基础题和进阶题的用例是相同的，不做区分。若参赛者选择只解决基础题，也必须将组合逻辑门映射为 GTP_LUT，可以使用基础代码中的 synth_pango 功能或其他。

这些用例中的非组合逻辑部分已经被映射到紫光同创 FPGA 的 GTP 类型，参赛者需要处理余下的组合逻辑门，图 6 给出了需要处理的五种组合逻辑门的示例。

所有用例均有如下特征：

（1）Verilog 格式的结构化网表，不含过程块或高级运算符；

（2）只包含一个展平的 module，包含的 cell 均为 yosys 内置类型或 GTP 类型；

（3）非组合逻辑门的 cell（寄存器，RAM，IO 等）视为 black box，参赛者不能对其做任何修改；

（4）组合逻辑门类型均为 yosys 内置的逻辑门类型，且只存在图 6 中的五种类型，$_AND_、 $_OR_、 $_XOR_均为 2 输入逻辑门，$_MUX_和$_NOT_的位宽为 1。

```
\$_OR_    u1 (.A(nt_rst), .B(N145 ), .Y(N145_1 ));
\$_AND_   u2 (.A(N6_inv), .B(N167_4 ), .Y(N107 ));
\$_NOT_   u3 (.A(tready), .Y(tready_inv));
\$_MUX_   u4 (.A(nt_prescale[9]), .B(prescale[12]), .S(N10),.Y(_N145));
\$_XOR_   u5 (.A(bit_cnt[0]), .B(bit_cnt[1]), .Y(N324_inv));
```

图 6 组合逻辑门示例

输出文件必须是 Verilog 格式的结构化网表。同样不区分输出文件是属于基础题或进阶题。

八、评分标准

用例成绩由可执行程序的最终输出文件决定。参赛者所提交的可执行程序的输出

文件，会按照如下公式计算其的代价函数，cost 数值越小代表结果质量越高：

$$\text{cost} = \left(\frac{\text{max logic level}}{20} + 1\right) * \text{num\_of\_luts} * 10 + \text{num\_of\_pins}$$

其中max logic level为输出结果中最大的 GTP_LUT 和 GTP_LUT6D 的串联级数，即信号连续经过 GTP_LUT 或 GTP_LUT6D 的最大个数；num_of_luts为输出结果中 GTP_LUT 和 GTP_LUT6D 的总数量；num_of_pins为输出结果中所有 GTP_LUT 和 GTP_LUT6D 的输入端口的总数量，无论输入端口是否连接了信号，或者连接的信号是否是常量。

用例分数(benckmark_score)是所有ratio的几何平均数x100。每个用例对应的ratio的计算公式如下：

$$\text{benckmark\_score} = \text{geomean}([\text{ratio}_0, \text{ratio}_1, \dots, \text{ratio}_n]) * 100$$

$$\text{ratio}_i = \begin{cases} \max\left(1 + \dfrac{(\text{base}_i - \text{cost})}{\text{base}_i}, 0.7\right), & \text{map\_success} \\ 0.6, & \text{map\_failed} \end{cases}$$

其中base_i是比较基准值，是用例 i 执行 synth_pango 功能（赛前提供的基础代码中的功能）后输出网表的 cost。

下列情况会视为 map failed:

（1） 输出结果中存在未映射的组合逻辑门；

（2） 输入用例和输出结果的逻辑不一致；

demo.ys文件中给出了一个示例，利用 yosys 内置的 equiv 功能来验证两个网表的逻辑一致性。

（3） 输出结果中存在不支持的 cell 类型；

请确保最后输出结果中的Cell均为GTP类型，且没有修改输入用例中存在的GTP cell 及所连信号。

（4） 输出结果中存在 LUT size > 6 的 GTP_LUT；

组合逻辑门只可以映射到 GTP_LUT1~GTP_LUT6，GTP_LUT6D。

（5） 输出结果中存在组合逻辑环路 ；

用例默认不含有组合逻辑环路，输出结果中也不能含有组合逻辑环路。特别提醒 GTP_LUT6D 的输出端口不能连接到自身的输入端口。

（6） 占用内存超 15GB；

（7） 超过 30 分钟未正常结束；

（8） 多次运行程序并输入相同的用例输出结果不一致；

（9） GTP_LUT6D 的 Z5 和 Z 的逻辑没有共享的输入；

例如 Z=I0&I1，Z5=I2|I3。

score.cc代码中根据赛题的描述，实现了 cost 计算功能，可以使用 score 命令计算输出文件的 cost。

相同分数的情况下，（1）比较失败用例数量，数量更少的队伍排名更高；（2）比较用例分数benckmark_score，分数越高排名越高。

九、参考文献

[1] Manohararajah V , Brown S D , Vranesic Z G .Heuristics for Area Minimization in LUT-Based FPGA Technology Mapping[J].IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2006, 25(11):2331-2340.DOI:10.a1109/TCAD.2006.882119.

[2]Yosys,https://yosyshq.readthedocs.io/projects/yosys/en/latest/index.html.

[3] Scholl C .Functional decomposition with application to FPGA synthesis /[J]. IEEE Circuits and Devices Magazine, 2001.DOI:10.1109/MCD.2003.1263462.

[4] Kubica Marcin , Opara Adam , Kania Dariusz .Technology Mapping for L

UT-Based FPGA[M]. 2021.

[5] Feng Wang, Liren Zhu, Jiaxi Zhang, Lei Li, Yang Zhang, Guojie Luo, "Dual-Output LUT Merging during FPGA Technology Mapping," Proceedings of the International Conference on Computer-Aided Design (ICCAD 2020), Virtual Conference, November 2-5, 2020.

[6] Lu Sheng, Shang Liuting, Qu Qian, Jung Sunyong, Liang Qilian, Pan Chenyun. An Efficient Multi-Output LUT Mapping Technique for Field-Programmable Gate Arrays[J].Electronics (2079-9292), 2025, 14(9).DOI:10.3390/electronics14091782.

[7] Liwei Ni, Zonglin Yang, Jiaxi Zhang, Changhong Feng, Jianhua Liu, Guojie Luo, Huawei Li, Biwei Xie, Xingquan Li, "MEC: An Open-source Fine-grained Mapping Equivalence Checking Tool for FPGA," Proceedings of the International Symposium of EDA (ISEDA 2023), Nanjing, China, pp. 131-136, May 8-11, 2023.

*本赛题指南未尽问题，见赛题 Q&A 文件

2025 China Postgraduate IC Innovation Competition・EDA Elite Challenge Contest

1. Problem

Logic Mapping Algorithm for FPGA Dual Output LUTs

2. Company

Shenzhen Pango Microsystems Co., Ltd.

3. Problem Background

Current advanced FPGA (Field Programmable Gate Array) architectures usually use dual-output LUT (Look-Up-Table) structures (hereinafter referred to as “dual-output LUTs”). The traditional LUT mapping algorithm can only map combinational logic gates to a single output LUT, which brings a new challenge to FPGA synthesis tools, “how to better map combinational logic to dual output LUTs”. The two output ports of a dual-output LUT share part of the input, and the two outputs need to satisfy a certain logical relationship between them. The difficulty lies in how to utilize the structure of the dual-output LUT during the mapping process and reflect the impact of the dual-output LUT in the cost function, as well as how to combine the LUTs to minimize the total area of the netlist at the end of the mapping process.

The FPGA design flow is shown in Figure 1. This contest problem belongs to the FPGA logic synthesis category in EDA (Electronic Design Automation) technology. The problem is divided into two parts: basic problem and advanced problem. The basic question is, given a mapped single-port LUT netlist, to find an algorithm to combine two LUTs into a dual-output LUT, so that the final number of LUTs in the netlist is as small as possible; the advanced question is how to modify the traditional single-port LUT mapping algorithm to map directly to a dual-output LUT to realize the minimum number of LUTs. The two questions belong to the Technology-dependent Optimization (which refers to the optimize netlist using specific technology libraries) and Technology Map (which refers to the mapping of generic circuit cells to cells in specific technology libraries) parts of logic synthesis.

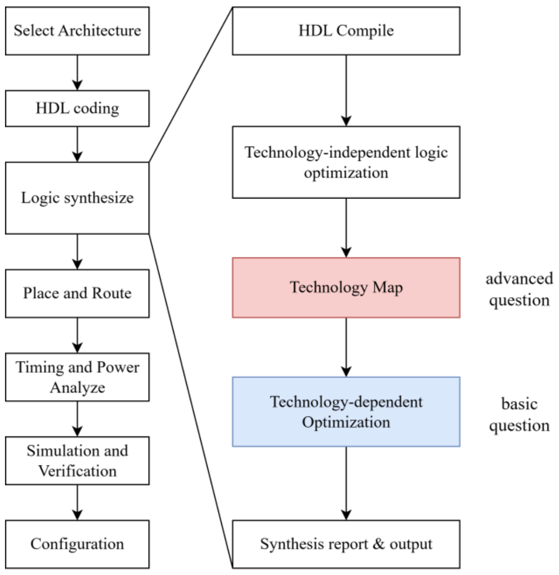

Figure 1 EDA design flow for FPGAs

4. Demo Flow

PANGO has provided a base code. The base code is written based on the open source project YOSYS. Participants can make modifications on the basis of this code. Figure 2 shows a demo flow using this code.

The scripts and test cases provided by PANGO are stored at yosys/techlibs/pango. synth_pango.cc implemented a mapping algorithm based on reference [1], the function is to map the combinational logic gates in the netlist to the LUT primitive of PANGO, with some conjectures and simplifications on the details not mentioned in the literature, which can be used with the synth_pango command under the yosys tool. The function of score.cc is to compute the cost function of the mapped netlist, which can be used by score command under the yosys tool.

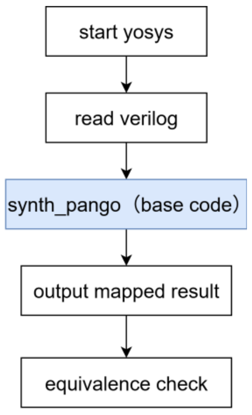

Figure 2 demo flow using base code

Yosys project contains two important data structures, SigBit and Cell.

SigBit is one bit of wire in yosys. Logic synthesis is not concerned with the bit width of the wires and the assign relationship between the wires, so the interconnected wires are converted to the unique SigBit representation. Base code uses SigBit object to get its driver, reader, depth and other data.

The synthesis process only focuses on the cell of combinational logic gates, the cases provided by PANGO only include the combinational logic gates of AND, OR, XOR, Multiplexer, Inverter. Other types of cells must be preserved as is without any processing.

5. Dual Output LUT Architecture

The cell types starting with GTP_ are FPGA primitives of PANGO. yosys/techlibs/pango/pango_sim.v file contains the simulation models of some primitives.

GTP_LUT1~GTP_LUT6 are n-input and 1-output lookup tables.

GTP_LUT6D is a 6-input, 2-output lookup table whose simulation model is shown in Figure 3 and internal structure in Figure 4.

In the estimation of resource consumption, GTP_LUT6D and GTP_LUT are equal, both of them will occupy one lookup table hardware resource in the FPGA chip, but GTP_LUT6D can additionally realize one 5-input and 1-output lookup table in addition to the function of GTP_LUT6, and the logic expression of Z5 is a sub-expression of the Z logic expression, and the I[4:0] of two logic expressions are the same. Therefore, we could combine two logic expressions with the same or similar inputs together using a single GTP_LUT6D to reduce area.

```
// GTP_LUT6D  仿真模型
module GTP_LUT6D
#(
      parameter [63:0] INIT = 64'h0000_0000_0000_0000
) (
      output Z,
      output Z5,
      input I0,
      input I1,
      input I2,
      input I3,
      input I4,
      input I5
);
wire z5a,z5b;
$lut #(.WIDTH(5),.LUT(INIT[31:0])) luta_cell(.A({I4,I3,I2,I1,I0}),.Y(z5a));
$lut #(.WIDTH(5),.LUT(INIT[63:32])) lutb_cell(.A({I4,I3,I2,I1,I0}),.Y(z5b));
$mux #(.WIDTH(1)) u1(.A(z5a),.B(z5b),.S(I5),.Y(Z));
assign Z5 = z5a;
endmodule
```

Figure 3 GTP_LUT6D simulation model

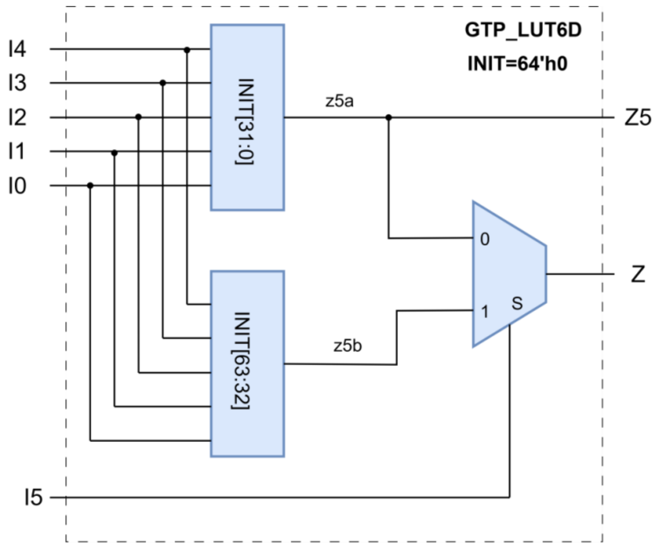

Figure 4 Internal structure of GTP_LUT6D

6. Competition Description

The competition content consists of two questions:

(1) Basic questions

Given a mapped single-port GTP_LUT netlist, design algorithms to combine the GTP_LUTs in the netlist into GTP_LUT6Ds, minimizing the total number of GTP_LUTs and GTP_LUT6Ds. For example, if there are two GTP_LUTs in the netlist whose logic is $O1 = A\&B\&D\&E$ and $O2 = A\&B\&C\&F \mid A\&B\&D\&E$ respectively, the two GTP_LUTs can be combine to become one GTP_LUT6D, and the total number of LUTs in the netlist is reduced by 1 after combing.

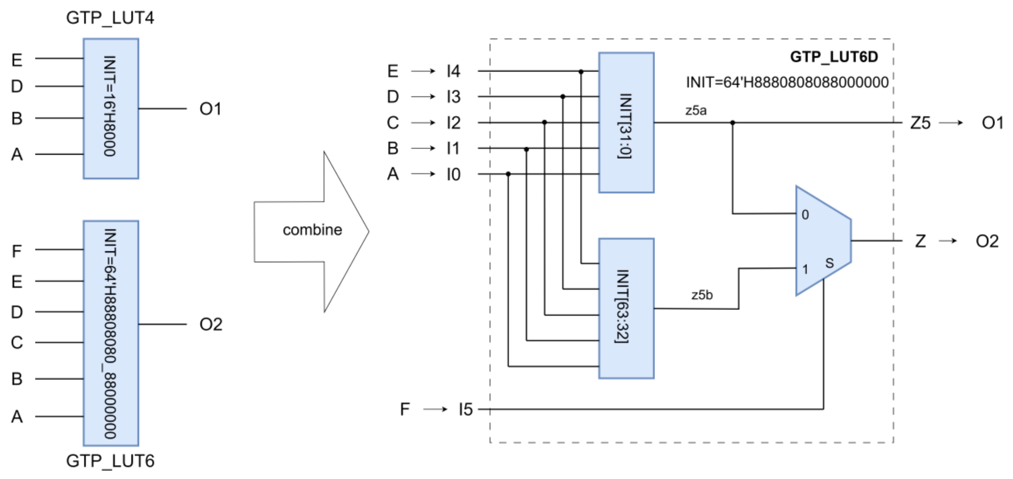

Figure 5 combine LUT Example

(2) Advanced Questions

Design algorithms to direct mapping combinational logic gate to GTP_LUT6D. Participants can modify or redesign the mapping algorithm to realize direct mapping out of GTP_LUT6D using the least number of total LUTs.

Participants must submit an executable program, along with the necessary documentation. It is recommended to submit the source code in case the submitted program fail to run on test server due to lib dependency issue.

7. Benchmarks

At least 6 development test cases and at least 10 evaluation test cases will be provided by PANGO. Development test cases will be published before the competition, while evaluation test cases will not be made public until the end of the competition. There is variance in scale among these test cases, with no distinction in difficulty.

The cases for the basic and advanced questions are the same and are not differentiated. If participant choose to solve only the basic question, also needed map the combinational logic gate to a GTP_LUT, either using the synth_pango function in the base code or otherwise.

The non-combinatorial logic of these cases has already been mapped to the GTP type of the PANGO, and participant only need to deal with the remaining combinatorial logic gates; an example of the five combinatorial logic gates that need to be dealt with is given in Figure 6.

All cases are characterized as follows:

(1) File is structured Verilog format, without procedure blocks or high-level operators;

(2) Include only one flattened module, and only included yosys built-in cell types or GTP cell types;

(3) non-combinational logic gates (registers, RAM, IO, etc.) are considered to be black boxes, and participants cannot make any modifications to them;

(4) The combinational logic gate types are all yosys built-in logic gate types, and only the five types in Figure 6 exist; $_AND_, $_OR_, and $_XOR_ are all 2-input logic gates, and $_MUX_ and $_NOT_ have a bit width of one.

```
\$_OR_    u1 (.A(nt_rst), .B(N145 ), .Y(N145_1 ));
\$_AND_   u2 (.A(N6_inv), .B(N167_4 ), .Y(N107 ));
\$_NOT_   u3 (.A(tready), .Y(tready_inv));
\$_MUX_   u4 (.A(nt_prescale[9]), .B(prescale[12]), .S(N10),.Y(_N145));
\$_XOR_   u5 (.A(bit_cnt[0]), .B(bit_cnt[1]), .Y(N324_inv));
```

Figure 6 combinational gate types

The output file must be structured Verilog. No distinction is made as to whether the output file belongs to a basic or advanced question.

8. Scoring Criteria

Score is determined by the final output file of the executable program. The output file of the executable program submitted by the participant is calculated as a cost function according to the following formula, where the smaller the cost value is, the higher the quality of the result:

$$\text{cost} = \left(\frac{\text{max logic level}}{20} + 1\right) * \text{num\_of\_luts} * 10 + \text{num\_of\_pins}$$

Where max logic level is the maximum number of GTP_LUTs and GTP_LUT6Ds in series in the output file, i.e., the maximum number of the signal through a GTP_LUT or GTP_LUT6D; num_of_luts is the total number of GTP_LUTs and GTP_LUT6Ds in the output result; num_of_pins is the total number of input ports of all the GTP_LUTs and GTP_LUT6Ds in the output file, whether the input ports are connected to signals or not, or whether the connected signals are constants or not.

The score is the geometric mean of all the $\text{ratio} * 100$. The formula for the corresponding ratio of each case is as follows:

$$\text{benckmark\_score} = \text{geomean}([\text{ratio}_0, \text{ratio}_1, \ldots, \text{ratio}_n]) * 100$$

$$\mathrm{ratio_i} = \begin{cases} \max\left(1 + \frac{(\mathrm{base_i} - \mathrm{cost})}{\mathrm{base_i}}, 0.7\right), & map\_success \\ 0.6, & map\_failed \end{cases}$$

Here $\mathrm{base_i}$ is the comparative benchmark value, which is the cost of the output netlist after use the synth_pango function (the function in the base code) with input $\mathrm{case_i}$.

Output result will be treated as map failure, if:

(1) there are unmapped combinational logic gates in the output results;

(2) there is logic mismatch between input file and output results;

An example is given in $\mathrm{demo.ys}$, using the built-in equiv function of yosys to verify the logical equivalence of two netlists.

(3) unsupported cell types are present in the output results;

Make sure that the cells in the final output result are all of GTP type, and the signals connect to GTP cells present in the input have not been modified.

(4) exists GTP_LUT with LUT size > 6;

The combinational logic gates can be mapped only to GTP_LUT1 to GTP_LUT6 and GTP_LUT6D.

(5) Combinatorial logic loops are present in the output ;

The cases do not contain combinational logic loops by default and cannot contain combinational logic loops in the output result. Note that the output port of GTP_LUT6D cannot connect to its own input port.

(6) Occupying more than 15 GB of memory;

(7) Not ending normally more than 30 minutes;

(8) Multiple runs of the program with the same input produce different output results;

(9) The logic of Z5 and Z of GTP_LUT6D does not share any input;

For example Z=I0&I1, Z5=I2|I3.

score.cc implement the cost calculation function according to the description before, and you can use the score command to calculate the cost of the output file.

For the same score, (1) compare the number of failed use cases, the team with less number is ranked higher; (2) compare benckmark_score, the higher the score the higher the ranking.

9. References

[1] Manohararajah V , Brown S D , Vranesic Z G .Heuristics for Area Minimization in LUT-Based FPGA Technology Mapping[J].IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2006, 25(11):2331-2340.DOI:10.a1109/TCAD.2006.882119.

[2] Yosys,https://yosyshq.readthedocs.io/projects/yosys/en/latest/index.html.

[3] Scholl C .Functional decomposition with application to FPGA synthesis /[J].IEEE Circuits and Devices Magazine, 2001.DOI:10.1109/MCD.2003.1263462.

[4] Kubica Marcin , Opara Adam , Kania Dariusz .Technology Mapping for LUT-Based FPGA[M]. 2021.

[5] Feng Wang, Liren Zhu, Jiaxi Zhang, Lei Li, Yang Zhang, Guojie Luo, "Dual-Output LUT Merging during FPGA Technology Mapping," Proceedings of the International Conference on Computer-Aided Design (ICCAD 2020), Virtual Conference, November 2-5, 2020.

[6] Lu Sheng, Shang Liuting, Qu Qian, Jung Sunyong, Liang Qilian, Pan Chenyun. An Efficient Multi-Output LUT Mapping Technique for Field-Programmable Gate

Arrays[J].Electronics (2079-9292), 2025, 14(9).DOI:10.3390/electronics14091782.

[7] Liwei Ni, Zonglin Yang, Jiaxi Zhang, Changhong Feng, Jianhua Liu, Guojie Luo, Huawei Li, Biwei Xie, Xingquan Li, "MEC: An Open-source Fine-grained Mapping Equivalence Checking Tool for FPGA," Proceedings of the International Symposium of EDA (ISEDA 2023), Nanjing, China, pp. 131-136, May 8-11, 2023.

* For questions not covered in this guide, please refer to the Q&A document.