CS 516000 FPGA Architecture & CAD

Programming Assignment (Due: 2024/11/11 22:00)

This project is completely open in terms of how you solve the problem. You may propose your own approach or follow previously proposed approaches in the literature.

Problem Description

We can represent a Boolean network as a directed acyclic graph in which each node represents a logic gate, and in which a directed edge (i,j) exists if the output of gate i is an input of gate j. A primary input (PI) node has no incoming edge, and a primary output (PO) node has no outgoing edge. We use $inputs(u)$ to denote the set of nodes that supply inputs to gate u. Given a subgraph H of a Boolean network, $inputs(H)$ denotes the set of distinct nodes outside H which supply inputs to the gates in H. For a node v in the network, a K-feasible cone at v, denoted C_v, is a subgraph consisting of v and its predecessors such that $|inputs(C_v)| \leq K$ and any path connecting a node in C_v and v lies entirely in C_v. A Boolean network is K-bounded if $inputs(v) \leq K$ for each node v. We assume that each programmable logic block in an FPGA is a K-input one-output lookup-table (K-LUT) that can implement any K-input Boolean function. Thus, each K-LUT can implement any K-feasible cone of a Boolean network. The technology mapping problem is to cover a given Boolean network with K-feasible cones.

In this assignment, you have to write a C/C++ program to generate a technology mapping of the given Boolean network. The objective of the project is to minimize the number of K-LUTs used.

Input Format

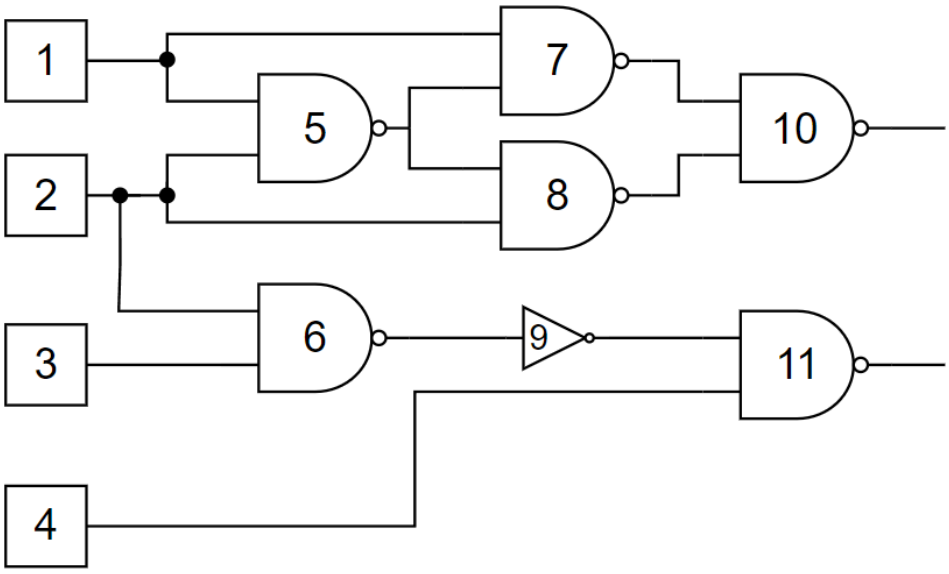
There will be one input file for each testcase which represents a Boolean network.

Format:

The first line of the file contains the name of the Boolean network and three integers N, I, O. N denotes the number of gates in the Boolean network. I denotes the number of primary inputs and O denotes the number of primary outputs. The following I lines will be the IDs of primary input nodes. And the following O lines will be the IDs of primary output nodes. Every line in the rest of the file lists the node ID of a gate and its input node IDs.

Ex:

```
exampleALU 11 4 2      // <name> <# of nodes> <# of PIs> <# of POs>
1                      // I lines of PI node IDs
2
3
4
10                     // O lines of PO node IDs
11
5 1 2                  // Node 5 with inputs 1 and 2
6 2 3                  // Node 6 with inputs 2 and 3
7 1 5                  // Node 7 with inputs 1 and 5
8 2 5                  // Node 8 with inputs 2 and 5
9 6                    // Node 9 with input 6
10 7 8                 // Node 10 with inputs 7 and 8
11 4 9                 // Node 11 with inputs 4 and 9
```



Constraints

- $1 \leq N \leq 10^5$
- $3 \leq K \leq 8$
- Runtime limit: 10 minutes. (If your program fails to generate a feasible mapping result within 10 minutes, it fails the testcase.)

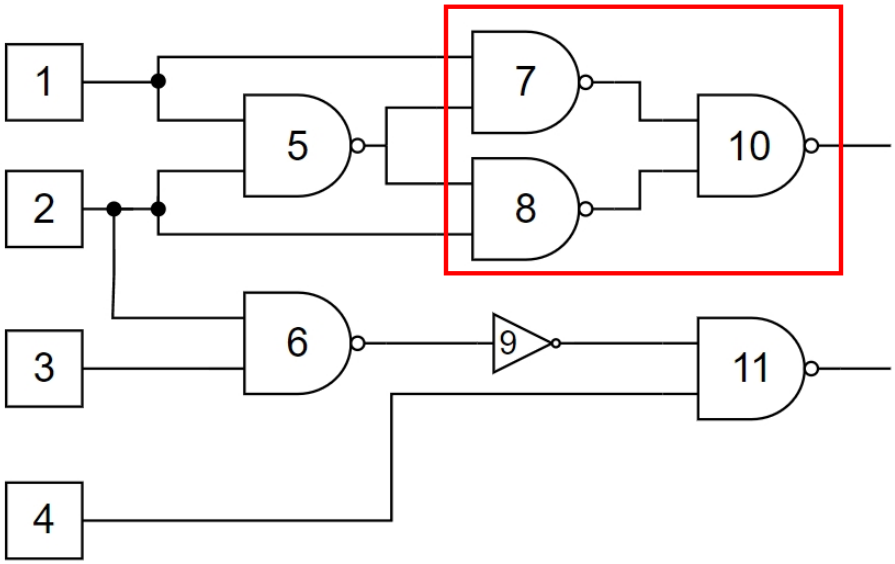
Output Format

Each line in the output fille represents a LUT in the following format:

<Output ID> <1st input ID> <2nd input ID> … <kth input ID>

Ex:

10 1 2 5 represents a 3-input LUT shown below.



Submission

The source codes should be uploaded to eeclass. Please include a Makefile for compiling your codes. The codes are required to be written in C/C++, and be compiled on a Linux platform. In addition, upload a report describing the details of your approach.

Name the executable file “mapper” and make sure your program can be executed by running the command:

$ ./mapper <input file path> <output file path> <K>

For example,

./mapper ./input/testcase1.blif ./output/testcase1.out 4

All codes should be compiled by running the command “make”. You don’t need to submit the executable code, and the command “make clean” should delete all the files generated by the command “make”.

Any plagiarism will result in a 0 grade for the project.

Grading

1. The completeness of your program and report (20%)
2. The solution quality, including hidden testcases (80%)

For each testcase, if your mapping solution is not valid, you will get 0 point on that testcase. Otherwise, the score is based on the total number of LUTs in your solution compared to other students (quality score of different testcases would be calculated independently). Here is the equation for score calculation:

$$100 - 35 * F, \text{ where}$$

$$F = \frac{\#\ of\ LUTs\ in\ your\ solution - \min \#\ of\ LUTs\ among\ all\ students}{\max \#\ of\ LUTs\ among\ all\ students - \min \#\ of\ LUTs\ among\ all\ students}$$