CSCE2301 – Digital Design I
Fall 2025

Project 1: Quine-McCluskey Logic Minimization

Objective:

The objective of this project is to make you more familiar with the Quine-McCluskey Logic Minimization algorithm.

Project Description and Requirements:

a) A computer program in ***any programming language of your choice*** (C, C++, C#, Java, Python, etc.) will be developed to accomplish the following tasks:

1. Read in (and validate) a Boolean function using its minterms/maxterms and don't-care terms (as represented below). The inputs are provided by a text file that has 3 lines. The first line contains the number of variables, the second line includes the minterms (indicated by m) or the maxterms (indicated by M) separated by commas, and the third line contains the don't-care terms separated by commas.

   For example, the following could be the content of such file. The input specifies a function of 3 variables, 4 minterms (1, 3, 6, and 7), and 2 don't-care terms (0 and 5):

```
3
m1,m3,m6,m7
d0,d5
```

2. Generate and print all prime implicants (PIs). For each PI show the minterms and don't-care terms it covers as well as its binary representation.
3. Using the PIs generated in part 2, obtain and print all the essential prime implicants EPIs (as Boolean expressions). Also, print the minterms that are not covered by the essential PIs.
4. Solve the PI table and print the minimized Boolean expression of the function. If there is more than one possible solution, print all of them.
5. [Bonus (worth 10%)] Based on the Boolean expression, generate the Verilog module for the function using Verilog Primitives.

b) 10 Test cases to demonstrate all the features of the program

c) PDF report (see details below)

Notes:

- Boolean functions up to 20 variables must be supported.
- Error checking is required to be implemented into your program.
- You need to work in a group of 2 to 3 students.

Deliverables:

You should upload the following to your group GitHub repository before the final interview.

- The application documented source code.
- A readme.txt file explaining how to build and use the application.
- The report (PDF format)

Deadlines and Grading Criteria

- **October 30:** Submit your group formation PDF on Canvas: **-10% if not done on time**.
- **November 2**: Submit five test circuits (test input files) on Canvas (November 3): **10%**
  - Note: You cannot copy or reuse others' circuits.
- **November 6:** A semi-functional implementation (can have bugs): **20%**.
- **November 13:** A fully functional implementation + PDF report: **70%**

    - PDF should be 5-6 pages + cover page (Font: 11 and 1.25 line spacing) should include the following:
        - The program design including any used data structures and algorithms
        - Challenges
        - Testing
        - Instructions to build and use the program
        - Problems/remaining issue in your program (if any)
        - The contributions of each member
- The final demo/interview in the instructor's office (Demo slots to be announced and reserved through Google sheets). This interview can affect the whole project grade.
- Note: Bad code organization and lack of comments: **-10%**.

Important Plagiarism notice: You must write your own code from scratch. Submissions based on others code will receive a grade of **zero** in the entire project (even if you understand every code line and even if the code is heavily re-factored/modified). Examples of such sources include (but is not limited to) code coming from the following sources: other teams, previous course offerings, open-source software, tutors, etc.

Important AI usage notice: As stated in the syllabus, you can use AI-generated code provided the project is far from being entirety AI-generated and provided you fully understand the generated code and have properly tested it. Also, you must document the tools and the exact process you followed (including prompts you used and the responses you received).