THE AMERICAN UNIVERSITY IN CAIRO

School of Sciences and Engineering

CMOS Circuit SPICE Generator Project

By:

Abdallah Saba ID:900204018
Shahd Elmahallawy ID: 900194441
Muhammad El-Mahdi ID:900202967

CSCE 330301- Fundamental Microelectronics

Spring 2022

Description:

A program that takes an input from the user as a boolean expression, and outputs the circuit corresponding to this expression. It checks the input and starts to call the main three functions we have, which are NOT, AND, and OR according to the precedence. Our Code output Y`, so we used the Not gate at the end to invert the output. After all, the output shows the connection between all MOSEFTS in detail.

Main Data Structures &Algorthims:

Data Structures:

Class MOSFET:

It has the attributes that we need to show in the output which is the name, drain, gate, body, source, and type. We pass an object of this class to each function we have.

Vectors:

Vector of CMOS.

Vector of strings.

We mainly depended on string manipulation in this project to adjust the input and validate it.

Algorithms:

We mainly used Brute Force Algorithm and applied it in the following functions:

1. Notgate Function:

   We pass to it the input by reference and two boolean flags. “Flag” is used to handle the case of the buffer when we have two Not gate after each other. For example, if the input is Y= a~~. The other flag is to check if the Not is last in the expression or not. If it is last it generates its output to Y otherwise it generates t a wire. Finally, we used it at the end to invert the output because our functions generate Y`, so we use it to get Y.

2. Andgate Function:

   It takes the input by reference and the two boolean flags. One to check if it is the first. In that case, it generates 4 Mosfets: 2 PMOS, and 2 NMOS. Otherwise, it generates only 2

MOSFETS: 1 PMOS, and 1 NMOS. The other boolean is to check if the last operator in the expression maps the output to be Y.

It generates the Pull-up MOSFETS to be parallel and the Pull Down to be series.

3. Orgate Function:

   The or gate takes the same inputs as the AND gate and the same booleans, but the only difference is that it generates the Pull-up Mosfets to be in series and the Pull Down to be parallel.

4. Adjust Function:

   It takes the input vector of CMOS and the vector of string includes the variables. Then, it loops over the vector of CMOS to finalize the expression as it was used to correct any mistakes in the expression and to connect any redundant points to make them the same, it looped over every point of the MOSFET to make sure it's correct.

5. Get_expression Function:

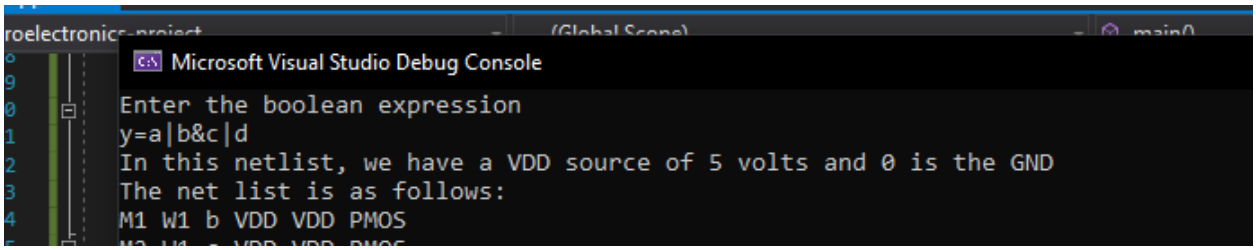
   This function is used to get the boolean expression. Then, we save the indices of each operator we have in vector for each operator. For example, "notindices" save the indices of the operators. Then, we loop over each vector and after we call any gate function we replace what we finish by "@".

6. There were some more functions that we used inside these functions, like getting the missing characters and so on.

7. We used the function (indices of the operators) to where (&,| and ~) are found in our input

User Guide:

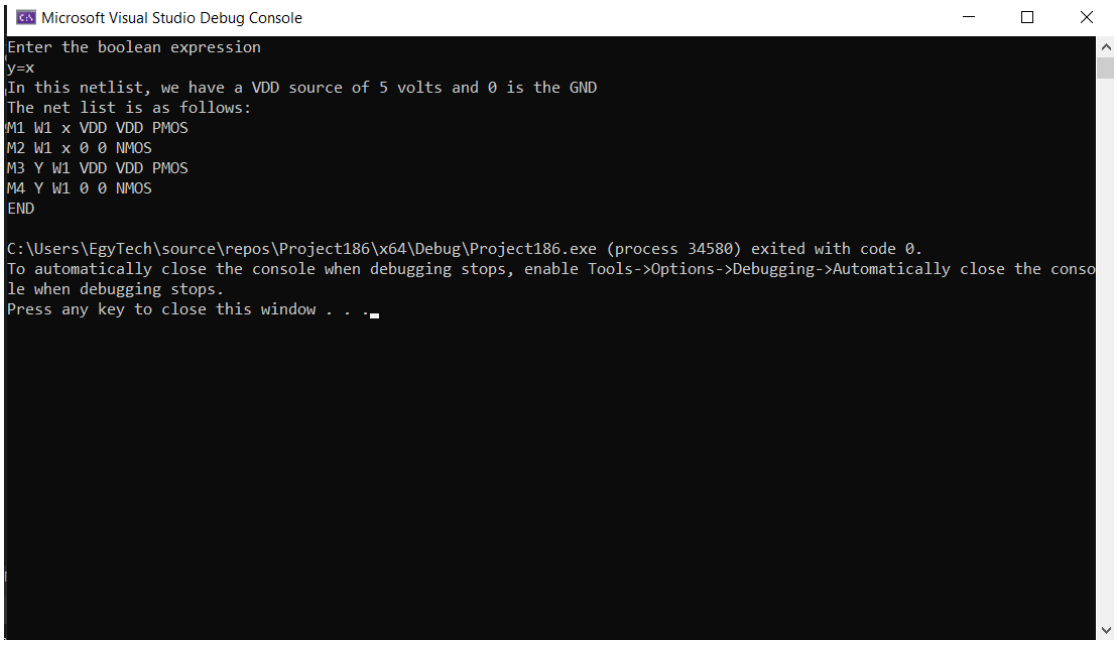
After running the code, the user will be asked to enter the Boolean expression. The user must enter the expression in the form of a letter = expression for example (y=a&b|c&d).The user Otherwise, the code will not run properly. An example of what the user should enter would be the following: After entering the expression, the output SPICE netlist will be displayed after pressing enter. The input can be (y=x (buffer), Y=x~ (inverter) , or any other circuit)



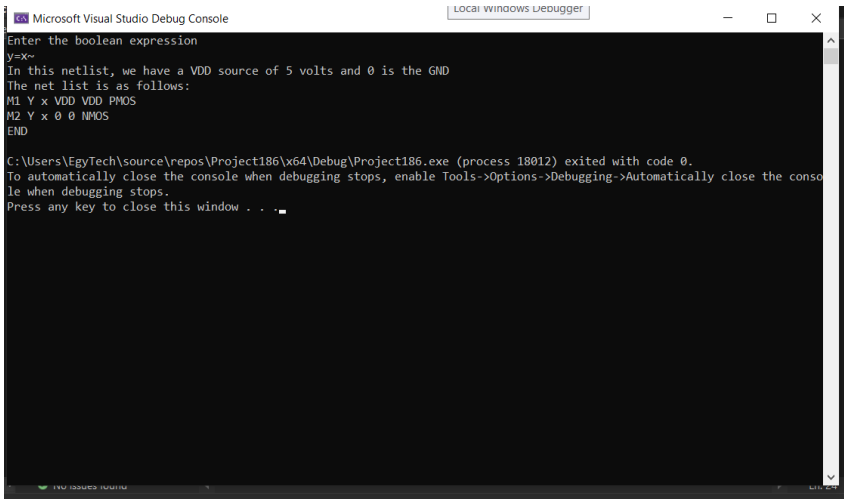
Test Cases:

1. **The first test case is the buffer gate where the user enters y=x, in this case we put 2 not gates together**

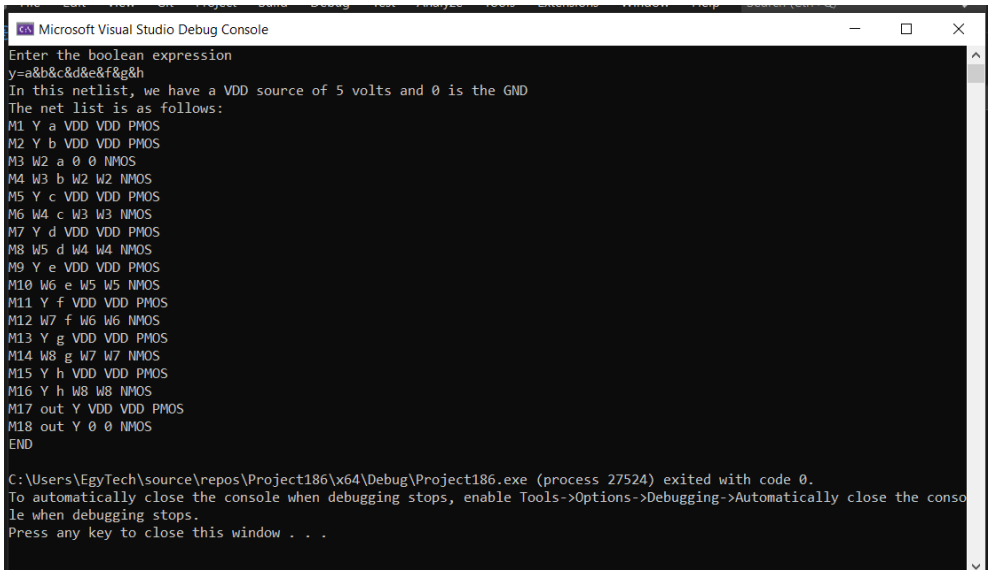
   screenshot :-



2. **The second test case was the simple not gate(y=x~)**

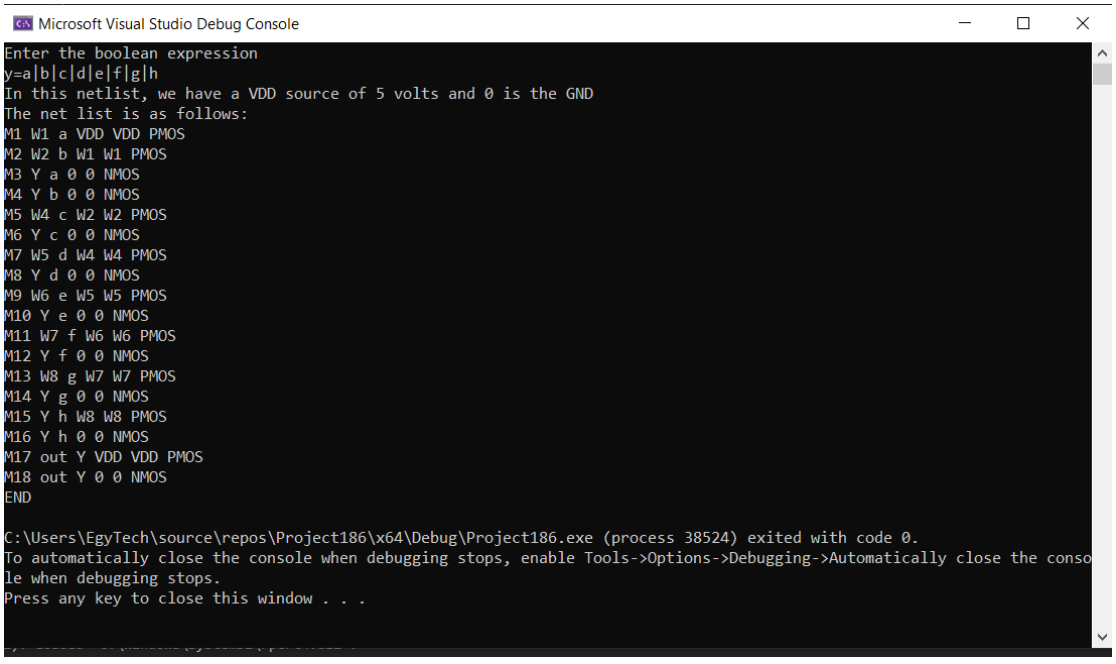


3. The third test case was (y=a&b&c&d&e&f&g&h)



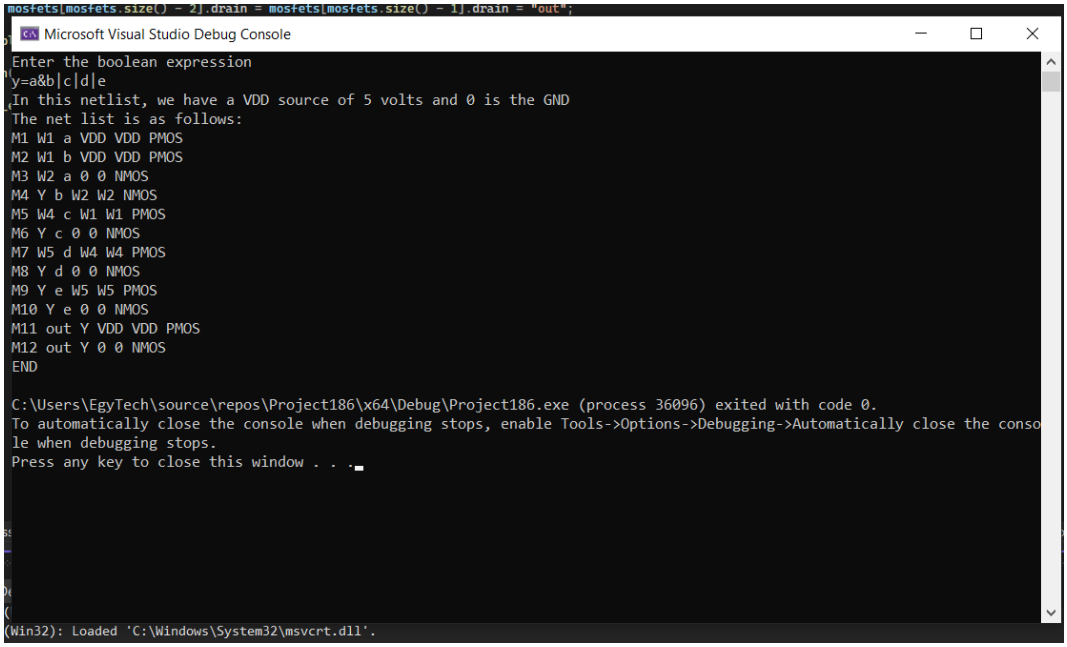
4. The fourth test case was (y=a|b|c|d|e|f|g|h)

Screenshot:-



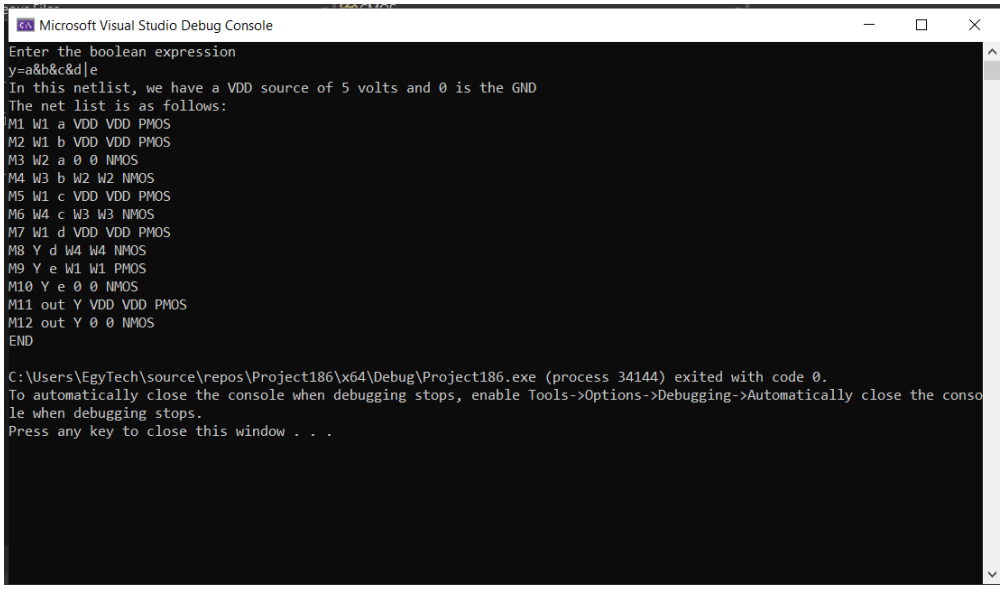
5. y= a&b|c|d|e

screenshot :-

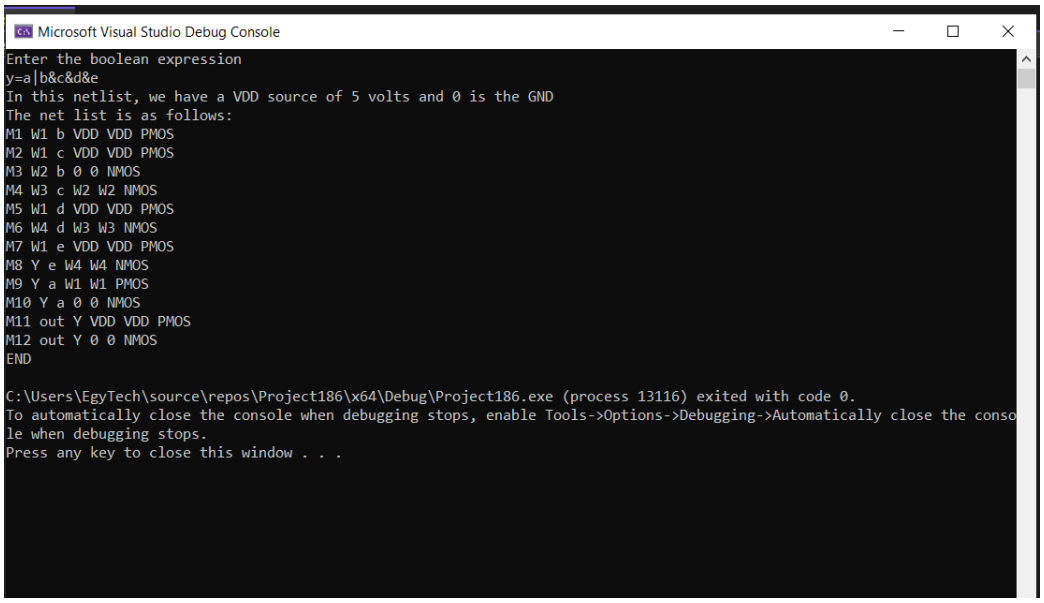


6. y=a&b&c&d|e

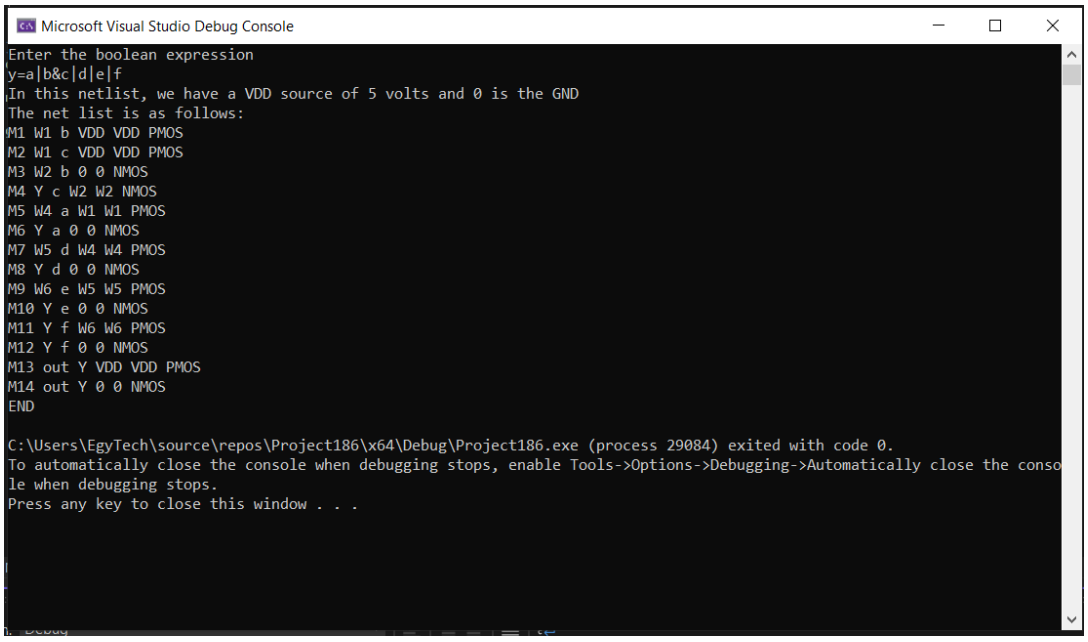
screenshot :-



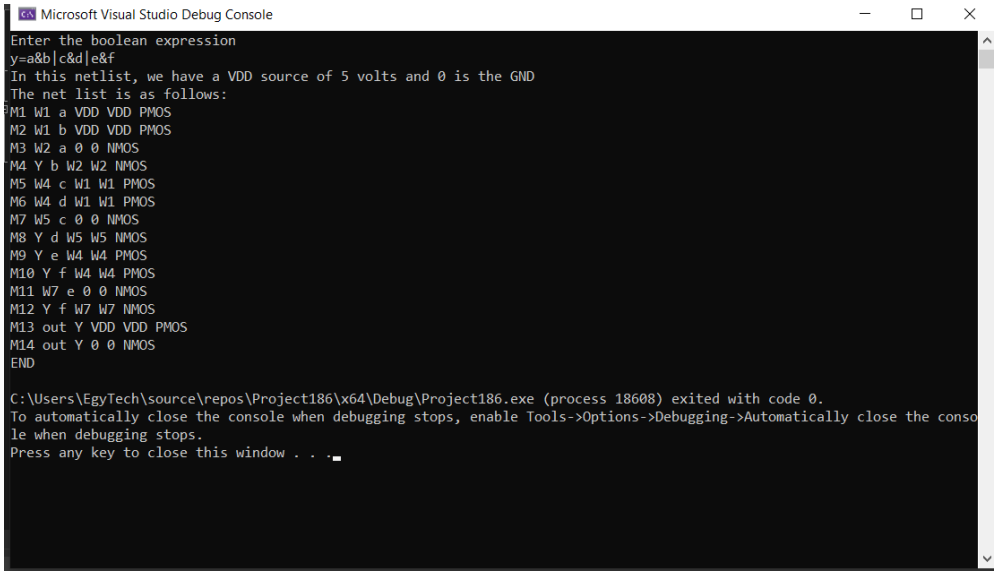
7. y=a|b&c&d&e



8. y=a|b&c|d|e|f



9. y=a&b|c&d|e&f



Contributions:

In this project, we all worked together equally as it was done through on-campus meetings where we used to brainstorm together, write the codes together and solve the bugs that we used to see in the code. Since we are all dorm residents, we met in the library and worked together on the project from A to Z and no one worked on something alone as we liked to share ideas together as if someone worked on a part alone, it was a bit difficult to adjust his ideas to the rest of the code.

Presentation link:-

https://drive.google.com/file/d/1WrV3fAmrGaFUtIyp5uXfe69ahdqg41hY/view?usp=sharing