K-map Minimizer

Digital Design 1

Dr. Mona Farouk

Mariam Fawzy 900192318

Basant Abdelaal 900192802

Program Design:

The program consists of functions to achieve 3 goals:

1) Validation of input
2) Creating K-map
3) Calculating the simplified function

1. Validation of Input:

The user enters two inputs: 1. the number of variables in the function 2. the minterms.

Validation of the number of variables:

- Should be between 2 and 4 variables (inclusive).

Validation of the minterms:

- Should be non-negative values.
- Should be less than $2^{\text{number of variables}}$.

2. Creating K-map:

This is done through two functions: 1. buildKmap   2. get_row_col (helper function)

buildKmap:

It simply initializes a table for the K-map with the proper size according to the number of variables and fills it with zeros.

And then, loops over the minterms and calculates the position in the K-map using get_row_col and changes its value to one.

get_row_col (helper function):

It calculates the position of the minterms with equations that make use of its binary representation.

3. Calculating the simplified function:

This is done through steps:

1. Getting all the prime implicants

2. Extracting the essential prime implicants

3. Minimizing the remaining prime implicants

Getting all the prime implicants

1. we get all the binary representation of the possible minterms in “string” format using the helper function getBinaryRep.

2. We try to get the largest combination of adjacent minterms to form prime implicants.

3. So, we start by getting all the combination starting from $2^{\text{number of variables}}$ to 1 using next_permutation

4. We only add the prime implicant if its minterms are not covered by a larger prime implicant by the help of keeping track of included minterms using visited array vis.

Extracting the essential prime implicants:

1. We created a table of the prime implicants (rows) and the minterms (columns) where its value is 1 whenever this minterm is included in that prime implicant.

2. Then, for each minterm, we looped over the prime implicants. So, if it is only present in one prime implicant, it is an essential prime implicant.

Minimizing the remaining prime implicants

1. To minimize the remaining prime implicants, we tried to generate the least combination of them, starting from 0 to all the non-essential prime implicants, using next_permutation.

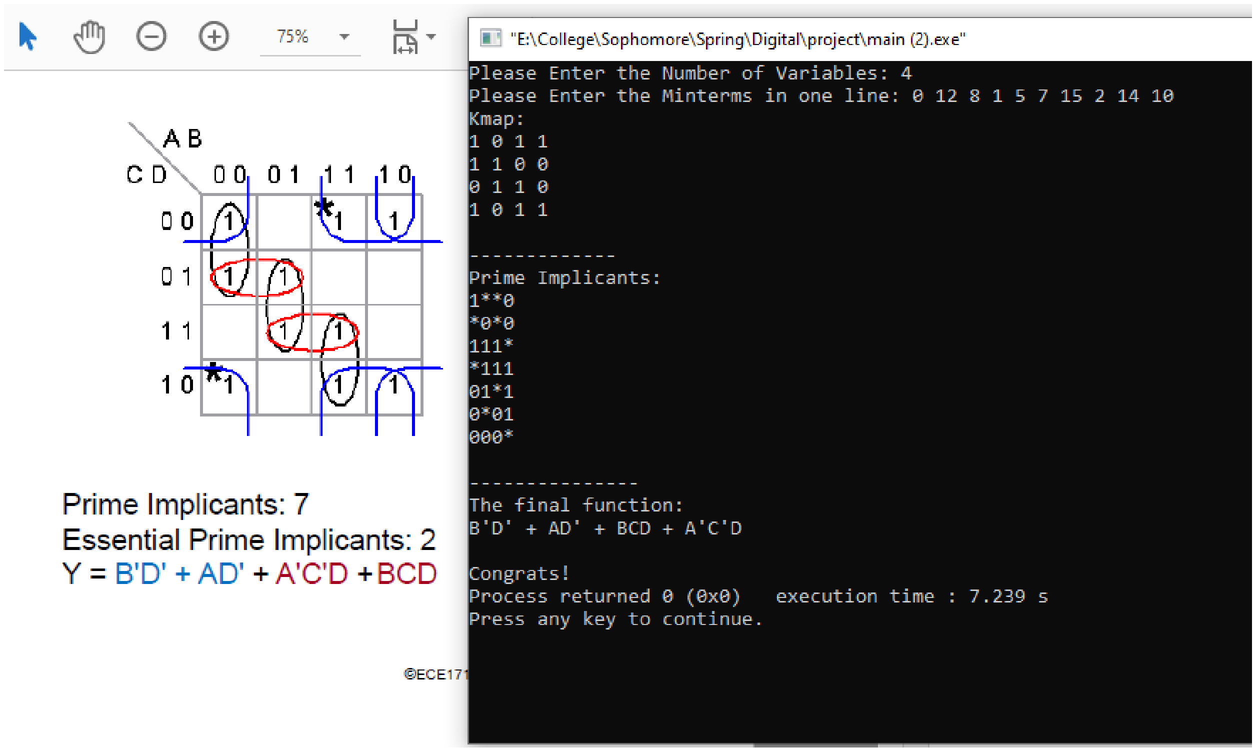
2. Once it covered all the minterms along with the essential prime implicants, terminate as we got the simplified function.

Problems in your program:

We covered all the number of variables from 2 to 4 and provided all the prime implicants and essential ones with no problems.

Instructions for how to build and use the program:

- To build the program just run the cpp file on any ide.
- The user interface is very intuitive, just enter the number of variables and then the minterms (in a single line) as in the example below:



Prime Implicants: 7
Essential Prime Implicants: 2
Y = B'D' + AD' + A'C'D + BCD

"E:\College\Sophomore\Spring\Digital\project\main (2).exe"

```
Please Enter the Number of Variables: 4
Please Enter the Minterms in one line: 0 12 8 1 5 7 15 2 14 10
Kmap:
1 0 1 1
1 1 0 0
0 1 1 0
1 0 1 1

-------------
Prime Implicants:
1**0
*0*0
111*
*111
01*1
0*01
000*

---------------
The final function:
B'D' + AD' + BCD + A'C'D

Congrats!
Process returned 0 (0x0)   execution time : 7.239 s
Press any key to continue.
```