K-Map Solver

By:

Ali Ahsan(412637)

Sudais Akbar(403965)

Introduction

- **Overview:**
- **8 Variable kmap solver**
- **14 variable truth table generator**
- Utilizes Quine-McCluskey (QM) algorithm for simplification

Language:

C++ (OOP)

Libraries

- **#include <iostream>**
- **#include <string>**
- **#include <vector>**
- **#include <iomanip> (defines manipulator funcs)**
- **#include <algorithm> (for sorting)**
- **#include <iterator> (for data seq navigation)**
- **#include <sstream> (Reads writes data to string)**
- **#include <cstring> (Retains null and track string length)**
- **#include <cmath>**

How it Works

1.User Input:

1. Enter the number of literals
2. Input minterms or Kmap data

2.Processing:

1. Apply QM algorithm for simplification
2. Converts minterms to binaries
3. Stores in vectors
4. Compares consecutive arrays
5. Eliminates similar bits
6. Converts final array to expressions

3.Output:

1. Display simplified Boolean expression
2. Generate truth table

Functions used:

1.Constructor:

1. Initializes the number of variables and sets up a string of "don't cares" with the same length.

2.getVars():

1. Returns a vector containing letters representing variables (like a, b, c...).

3.decToBin():

1. Converts a decimal number into its binary equivalent.

4.pad():

1. Adds leading zeros to a binary number to make it the desired length.

5.isGreyCode():

1. Checks if two binary strings differ by only one bit.

6.replace_complements():

1. Compares two binary strings and replaces differing bits with dashes ("-").

7.in_vector():

1. Checks if a string exists in a vector.

8.reduce():

1. Attempts to reduce the minterms using the Quine-McCluskey algorithm:
    1. It compares each minterm with others to find pairs that differ by only one bit.
    2. If found, it replaces the differing bit with a dash ("-") and adds it to the new minterms.
    3. Then, it appends the remaining minterms that weren't matched or combined.

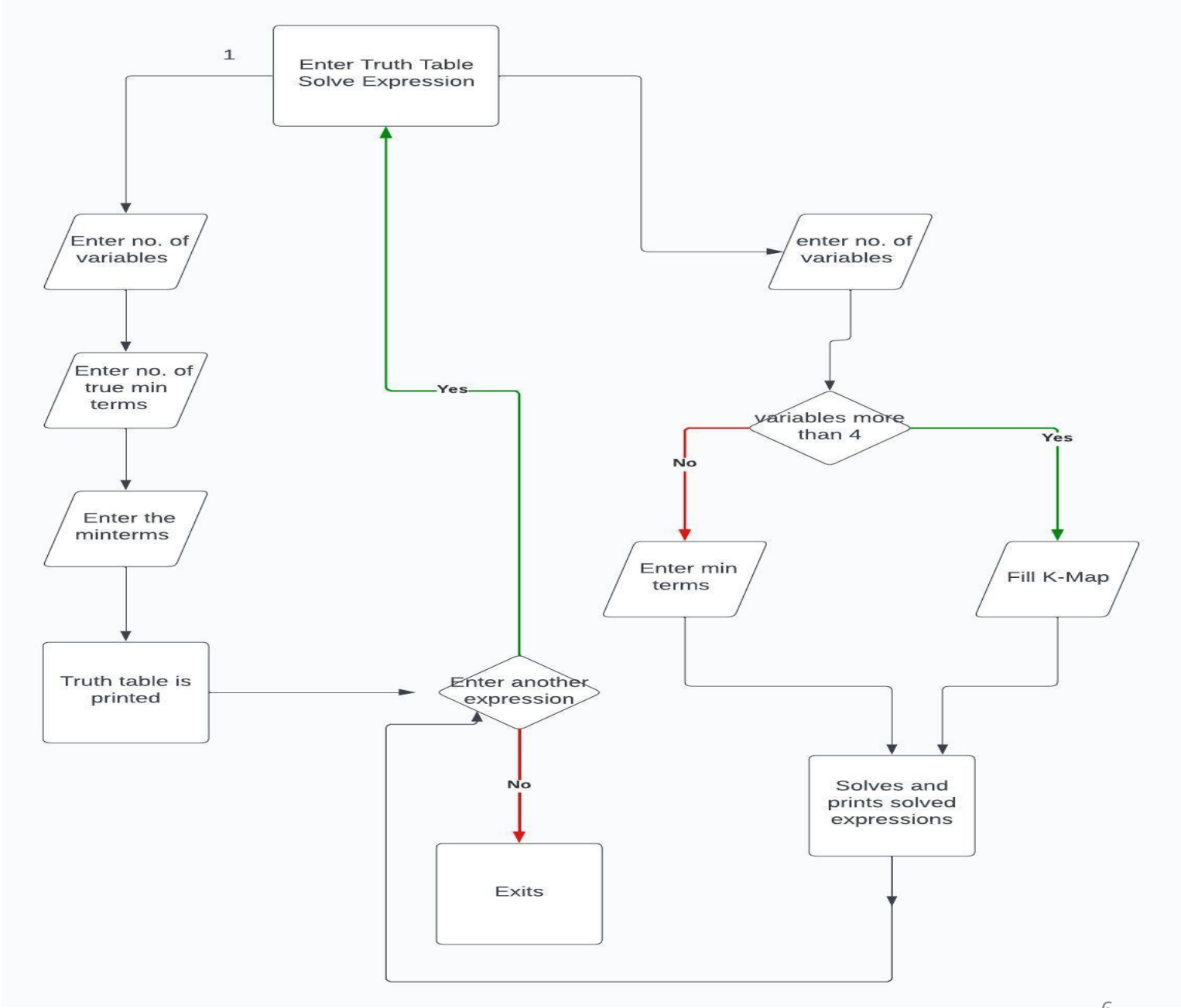
9.getValue():

1. Converts the reduced minterms back into their boolean representation using the variable letters.

10.VectorsEqual():

1. Checks if two vectors of strings are equal by sorting them and comparing each element.

Flow Chart


1
Enter Truth Table
Solve Expression
Enter no. of variables
Enter no. of true min terms
Enter the minterms
Truth table is printed
Yes
Enter another expression
No
Exits
enter no. of variables
variables more than 4
No
Yes
Enter min terms
Fill K-Map
Solves and prints solved expressions

```
============ :Choose an Option: ============
1 => Generate Truth Table
2 => Solve Boolean Expression
============================================
Enter Here: 1
Enter number of Literals (MAX : 14):
14
Enter number of Minterms for which the Output is 1:
1
Enter the True Minterms (RANGE : 0 - 16383)(e.g : 0 1 2 3 - - -) :
1

---------------Truth Table---------------
Minterms | A B C D E F G H I J K L M N | Outputs (Y)
-----------------------------------------
     m0  | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |  0
     m1  | 0 0 0 0 0 0 0 0 0 0 0 0 0 1 |  1
     m2  | 0 0 0 0 0 0 0 0 0 0 0 0 1 0 |  0
     m3  | 0 0 0 0 0 0 0 0 0 0 0 0 1 1 |  0
     m4  | 0 0 0 0 0 0 0 0 0 0 0 1 0 0 |  0
     m5  | 0 0 0 0 0 0 0 0 0 0 0 1 0 1 |  0
     m6  | 0 0 0 0 0 0 0 0 0 0 0 1 1 0 |  0
     m7  | 0 0 0 0 0 0 0 0 0 0 0 1 1 1 |  0
     m8  | 0 0 0 0 0 0 0 0 0 0 1 0 0 0 |  0
     m9  | 0 0 0 0 0 0 0 0 0 0 1 0 0 1 |  0
    m10  | 0 0 0 0 0 0 0 0 0 0 1 0 1 0 |  0
    m11  | 0 0 0 0 0 0 0 0 0 0 1 0 1 1 |  0
    m12  | 0 0 0 0 0 0 0 0 0 0 1 1 0 0 |  0
    m13  | 0 0 0 0 0 0 0 0 0 0 1 1 0 1 |  0
```

```
  \ B |
 A \  | 00   01
------+----+----+
  00  | 1  | 1  |
      +----+----+
  01  | 0  | 0  |
      +----+----+

The Reduced Boolean Expression in (SOP) form is:
A'
```

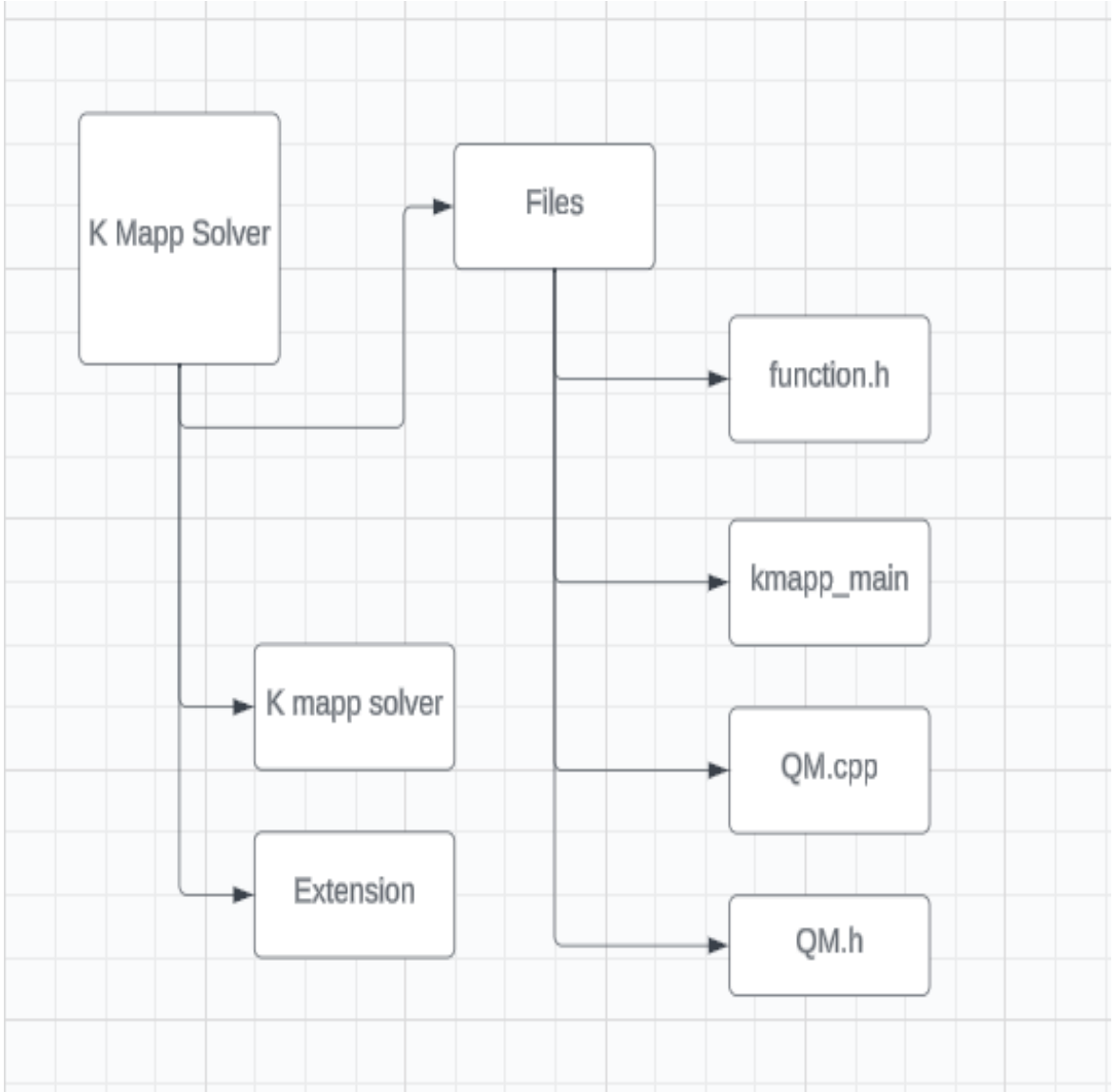
```
  \BC |
 A \  | 00   01   11   10
------+----+----+----+----+
  00  | 1  | 1  | 0  | 1  |
      +----+----+----+----+
  01  | 1  | 1  | 0  | 0  |
      +----+----+----+----+

The Reduced Boolean Expression in (SOP) form is:
B' + A'C'
```

Boolean Expression for different number of literals

```
  \CD |
AB \  | 00   01   11   10
------+----+----+----+----+
  00  | 1  | 1  | 1  | 1  |
      +----+----+----+----+
  01  | 1  | 0  | 1  | 0  |
      +----+----+----+----+
  11  | 0  | 0  | 0  | 0  |
      +----+----+----+----+
  10  | 1  | 1  | 1  | 1  |
      +----+----+----+----+

The Reduced Boolean Expression in (SOP) form is:
B' + A'C'D' + A'CD

Would you like to enter another expression? (y/n)
```

```
===== :Welcome to Karnaugh Map Solver: =====
==========================================
Enter the number of Variables (Literals):
8

Enter the Minterms separated by Commas (RANGE: 0-255) e.g: F(0,1,2,3,---):
0,1,2,8,10,15,20,34,111,150,220,255

The Reduced Boolean Expression in (SOP) form is:
A'B'D'E'F'GH' + A'B'C'D'F'H' + A'B'C'DE'FG'H' + AB'C'DE'FGH' + ABC'DEFG'H' + ABCDEFGH
```

Tree Diagram



Importance of Kmap and QM Algorithm

Karnaugh Maps (Kmaps):

- Visual representation for Boolean functions

Quine-McCluskey (QM) Algorithm:

- Essential for minimizing Boolean expressions
- Effective in reducing complexity and optimizing circuits

Key Features

Capabilities:

- Solves upto 8 variables (256 minterms)
- Can takes bool function or Kmap data
- Truth Table for upto 14 Variables (16384 minterms)

Functionality:

- Application of Quine-McCluskey algorithm
- Simplification of Boolean expressions
- Solving of Karnaugh maps
- Generation of truth tables

Uses

- **Academic Settings:**
  - Students learning Boolean algebra and circuit design
- **Professional Settings:**
  - Engineers designing and optimizing digital circuits
- **Automation:**
  - Integration into software tools for automatic simplification

Benefits

Time Efficiency:

- Rapid simplification of Boolean expressions

- **Accuracy:**
  - Minimize errors in manual simplification

- **Versatility:**
  - Applicable to various fields requiring Boolean expression simplification

Future Enhancements

- **User Interface Improvements:**
  - Intuitive design for better user experience using GUI
- **Additional Algorithms:**
  - Integration of more optimization algorithms
- **Platform Expansion:**
  - Deployment on different platforms and devices

Contributions:

Sudais Akbar Khan:

- Implementation of Kmap Expression Solver using QM Algorithm

Ali Ahsan:

- Implementation of Truth table generation and sorting algorithms

”

Thank You

GitHub link

- https://github.com/aahsan-bee/Boolean-Expression-Solver-using-Quine-McCluskey-Algorithm.git