Assembly - Assignment I
Simplifying Boolean expression using Kmap

Manideep Parusha - FWC22004

CONTENTS

***Abstract*—This manual describes the rules for the simplification of K-maps and steps to develop the simplified expression which is furthuer implemented on hardware for functionality for a given Boolean Function.**

I. PROBLEM

Reduce the following Boolean expression in the simplest form using Kmap. The Expression with Sum of Products (SoP) is as follows:

$$F(P,Q,R,S) = \sum(0,1,2,3,5,6,7,10,14,15)$$

II. SOLUTION

A. Truth Table

Truth table for the given Boolean function:

	P	Q	R	S	F(P,Q,R,S)
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	1
15	1	1	1	1	1

TABLE I
TRUTH TABLE

B. K-map

A Karnaugh map (K-map) is a visual method used to simplify the algebraic expressions in Boolean functions without having to resort to complex theorems or equation manipulations. It involves fewer steps than the algebraic minimization technique to arrive at a simplified expression. K-map simplification technique always results in minimum expression if carried out properly. Rules to follow while simplifying a K-map mention in further sections.

K-map for the above mentioned truth table:

RS \ PQ	00	01	11	10
00	1	1	1	1
01	0	1	1	1
11	0	0	1	1
10	0	0	0	1

TABLE II
K-MAP FROM THE TRUTH TABLE

C. Rules to simplify K-maps

The Karnaugh map uses the following rules for the simplification of expressions by grouping together adjacent cells containing ones

1) Groups may not include any cell containing a zero
2) Groups may be horizontal or vertical, but not diagonal
3) Groups must contain 1, 2, 4, 8, or in general 2^n cells
4) Each group should be as large as possible.
5) Each cell containing a one must be in at least one group
6) Groups may overlap
7) Groups may wrap around the table. The leftmost cell in a row may be grouped with the rightmost cell and the top cell in a column may be grouped with the bottom cell
8) There should be as few groups as possible, as long as this does not contradict any of the previous rules

D. Simplification

PQ \ RS	00	01	11	10
00	1	1	1	1
01	0	1	1	1
11	0	0	1	1
10	0	0	0	1

TABLE III
K-MAP 1

Resulting expression from the groups in the above K-map: P'Q'

PQ \ RS	00	01	11	10
00	1	1	1	1
01	0	1	1	1
11	0	0	1	1
10	0	0	0	1

TABLE IV
K-MAP 2

Resulting expression from the groups in the above K-map: P'Q' + P'S

PQ \ RS	00	01	11	10
00	1	1	1	1
01	0	1	1	1
11	0	0	1	1
10	0	0	0	1

TABLE V
K-MAP 3

Resulting expression from the groups in the above K-map: P'Q' + P'S + RS'

PQ \ RS	00	01	11	10
00	1	1	1	1
01	0	1	1	1
11	0	0	1	1
10	0	0	0	1

TABLE VI
K-MAP 4

Resulting expression from the groups in the above K-map: P'Q' + P'S + RS' + QR

As we have grouped all the 1's in the K-map, we can consider the obtained expression to be final. In the K-map above, we can group atmost four 1's only. So we cannot further simplify the obtained expression from the K-map.

Hence, the simplified Boolean expression using Kmap will be:

F(P,Q,R,S) = P'Q' + P'S + RS' + QR

III. Hardware Implementation

A. Components required

Component	Qty.
Arduino UNO	1
Jumper wires	4

TABLE VII
COMPONENTS LIST

The hardware we are using for implementation is the Arduino UNO board, which has AtMega328p micro-controller on-board. It is an AVR micro-controller with 8-bit ALU, works at 16MHz clock frequency, 3 ports including 1 analogue port. We are using 4 pins(2,3,4,5) of PORTD for input and 1 pin(13) of PORTB for output.

B. Connections

The following table shows the inputs to be given to the microcontroller for implementing the above simplified Boolean expression.

Arduino	2	3	4	5
Inputs	P	Q	R	S

TABLE VIII
INPUT CONNECTIONS

P, Q, R, S are the inputs which can be set or reset by manually connecting them to 5V or GND respectively. In this way we can make 16 combinations of the input sequence and they can be tested for functionality. The output is from from Pin 13, which drives the on-board LED of Arduino UNO. When the output is '1', the LED is 'ON' and when the output is '0', the LED will be 'OFF'.

C. Code

The code for implementing the logic of the Boolean expression in 'Assembly' language using AVR assembler can be downloaded by the command mentioned below

```
wget https://raw.githubusercontent.com/parusamanideep
/FWC/main/assembly/codes/kmap.asm
```

The logic in the assembly code is shown below which computes the Boolean Expression simplified above.

```
mov result, P
eor result, mask
mov temp,Q
eor temp, mask
and result, temp
mov temp, P
eor temp, mask
and temp, S
or result, temp
mov temp, S
eor temp, mask
and temp, R
or result, temp
mov temp, Q
and temp, R
or result, temp
```

Few AVR instructions used in the code are listed below with there usage explained.

Instruction	use
LDI	Load register with value
MOV	Move data from register to register
AND	Perform AND operation
OR	Perform OR operation
EOR	Perform XOR operation

TABLE IX
AVR INSTRUCTIONS USED IN CODE

IV. CONCLUSION

The Simplified Boolen Expression is implemented on Arduino and it's functionality can be tested by connecting P, Q, R, S inputs to 5V and GND to give the Pin '1' or '0' value.
Total number of combination of inputs that can be formed are 16 and the output can be verified by comparing the truthtable and the output obtained for the given inputs.