Karnaugh maps

Olli-Pekka Hämäläinen

Simplification of a logic circuit

- During the previous lecture we learned how to construct logic functions for the circuits in SOP- and POS-form & how to simplify them using Boolean algebra
- We noticed, that if there are less zeros than ones (or vice versa) in a truth table, these standard forms produce a quite simple starting point and the simplification using Boolean algebra is "fairly easy"
- If the truth table is more balanced, algebraic simplification is rather hard task
- In these cases it's a good idea to use a Karnaugh map
  - This tool takes advantage of the human perception
  - Also few other pros (let's get back to this later)
  - Construction is done according to the Gray code

The binary number problem

- So far we've preferred to construct truth tables by writing the rows in binary order
- For example, in a three-variable circuit this order produces a problem: after combination 011 comes combination 100
  - Truth values of each variable change!
  - This is undesirable, because in reality, truth values of variables don't change immediately, but there's a short transition state in between
  - Truth values also don't change fully synchronized
  - So, the change from 011 to 100 can actually happen for example like this:

$$011 \rightarrow 111 \rightarrow 101 \rightarrow 100$$

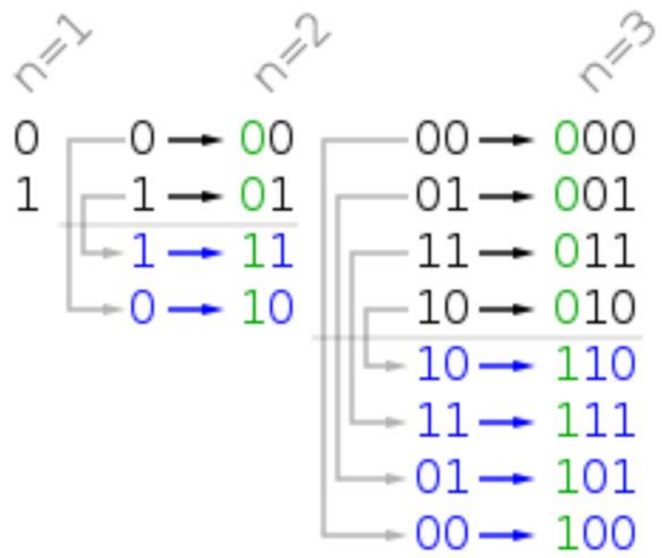
Gray code

- Beforementioned problem can be avoided in such a way that we reorder the combinations to an order where only one bit changes at a time
- This kind of ordering is called reflected binary code (RBC) or Gray code (according to its inventor Frank Gray)

Decimal	Binary	Gray
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101

Gray code

- Gray code follows a simple logic:
  - Start from numbers 0, 1
  - Reflect the numbers of the previous column – so, write the numbers one below another, but in reverse order
  - Write new numbers in front of old ones in such a way that in front of original numbers we put 0 and in front of new numbers we put 1



Karnaugh map (K-map)

- A Karnaugh map in constructed (in its most common form) for 4 variables in such a way that the combinations are ordered in Gray code order – starting from top left corner and in “zig-zag formation”

CD \ AB	00	01	11	10
00				
01				
11				
10				

Karnaugh map (K-map)

- A Karnaugh map in constructed (in its most common form) for 4 variables in such a way that the combinations are ordered in Gray code order – starting from top left corner and in “zig-zag formation”

CD \ AB	00	01	11	10
00	0000	0100	1100	1000
01	0001	0101	1101	1001
11	0011	0111	1111	1011
10	0010	0110	1110	1010

Karnaugh map (K-map)

- A Karnaugh map in constructed (in its most common form) for 4 variables in such a way that the combinations are ordered in Gray code order – starting from top left corner and in “zig-zag formation”

CD \ AB	00	01	11	10
00	0000	0100	1100	1000
01	0001	0101	1101	1001
11	0011	0111	1111	1011
10	0010	0110	1110	1010

Karnaugh map (K-map)

- After this we mark the truth values (0s and 1s) of the combinations in respective cells
- The construction of the K-map is easiest to learn via an example, so let’s consider the example 2 from previous lecture (we didn’t simplify this using Boolean algebra, since it would’ve taken a lot of time)
  - After we’ve done the K-map, we can simplify!
- The truth table for this example problem looked like this:

Karnaugh map (K-map)

- After this we mark the truth values (0s and 1s) of the combinations in respective cells
- The construction of the K-map is easiest to learn via an example, so let's consider the example 2 from previous lecture (we didn't simplify this using Boolean algebra, since it would've taken a lot of time)
  - After we've done the K-map, we can simplify!
- The truth table for this example problem looked like this:

A	B	C	D	F
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Karnaugh map (K-map)

- First let's order the combinations in the map

CD \ AB	00	01	11	10
00	0000	0100	1100	1000
01	0001	0101	1101	1001
11	0011	0111	1111	1011
10	0010	0110	1110	1010

A	B	C	D	F
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Karnaugh map (K-map)

- Then we write the truth values of the logic function F to respective combinations:

CD \ AB	00	01	11	10
00	**1**	**0**	**0**	**0**
01	**1**	**1**	**1**	**1**
11	**1**	**0**	**0**	**0**
10	**1**	**1**	**1**	**1**

A	B	C	D	F
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Simplifying via Karnaugh

- Karnaugh map can be simplified either in SOP form (finding 1s) or in POS form (finding 0s)
- Let’s now consider SOP-simplification (since it’s the most common one)
  - POS simplification can be done using the same principles
- Adjacent 1s (vertically and horizontally; not in diagonal direction) give the opportunity to simplify the function if the following conditions are met:
  - The area formed by the 1s must be a rectangle
  - The area must be a power of 2 (1, 2, 4, 8, …)
  - Areas may overlap each other
  - Map is connected at edges, so the areas can “continue over the boundary”
  - Areas are referred to by abbreviations, which depict the truth values of variables on that area

Karnaugh map (K-map)

- Find the 1-areas:

CD \ AB	00	01	11	10
00	1	0	0	0
01	1	1	1	1
11	1	0	0	0
10	1	1	1	1

Karnaugh map (K-map)

- Find the 1-areas:

AB CD	00	01	11	10	
00	1	0	0	0	
01	1	1	1	1	$\overline{C}D$
11	1	0	0	0	
10	1	1	1	1	

Karnaugh map (K-map)

- Find the 1-areas:

CD \ AB	00	01	11	10	
00	1	0	0	0	
01	1	1	1	1	$\bar{C}D$
11	1	0	0	0	
10	1	1	1	1	$C\bar{D}$

Karnaugh map (K-map)

- Find the 1-areas:

CD \ AB	00	01	11	10
00	1	0	0	0
01	1	1	1	1
11	1	0	0	0
10	1	1	1	1

$\bar{C}D$

$C\bar{D}$

$\bar{A}\bar{B}$

Karnaugh map (K-map)

- The simplest SOP-form is then

$$F = \overline{A}\overline{B} + \overline{C}D + C\overline{D}$$

CD \ AB	00	01	11	10	
00	1	0	0	0	
01	1	1	1	1	$\overline{C}D$
11	1	0	0	0	
10	1	1	1	1	$C\overline{D}$
	$\overline{A}\overline{B}$				

Karnaugh map, 3 variables

- In case of three variables, the lowest 2 rows are left out
- Otherwise the operating principle hasn't changed

C \ AB	00	01	11	10
0	000	010	110	100
1	001	011	111	101

Karnaugh map, 3 variables

- Example: simplify the logic function F by using this truth table

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Karnaugh map, 3 variables

- Move the truth values from table to map
  - Notice: you don't need to remember the order of Gray code by heart very far, if you remember that the ordering of rows and columns follows Gray code!

C \ AB	00	01	11	10
0	000	010	110	100
1	001	011	111	101

Karnaugh map, 3 variables

- Move the truth values from table to map
  - Notice: you don't need to remember the order of Gray code by heart very far, if you remember that the ordering of rows and columns follows Gray code!

C \ AB	00	01	11	10
0	000	010	110	100
1	001	011	111	101

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Karnaugh map, 3 variables

- Move the truth values from table to map
  - Notice: you don't need to remember the order of Gray code by heart very far, if you remember that the ordering of rows and columns follows Gray code!

C \ AB	00	01	11	10
0	**1**	**0**	**1**	**1**
1	**1**	**0**	**0**	**1**

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Karnaugh map, 3 variables

- Find the areas that have 1s:
  - Remember that the map is connected at its borders!

C \ AB	00	01	11	10
0	1	0	1	1
1	1	0	0	1

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Karnaugh map, 3 variables

- Find the areas that have 1s:
  - Remember that the map is connected at its borders!

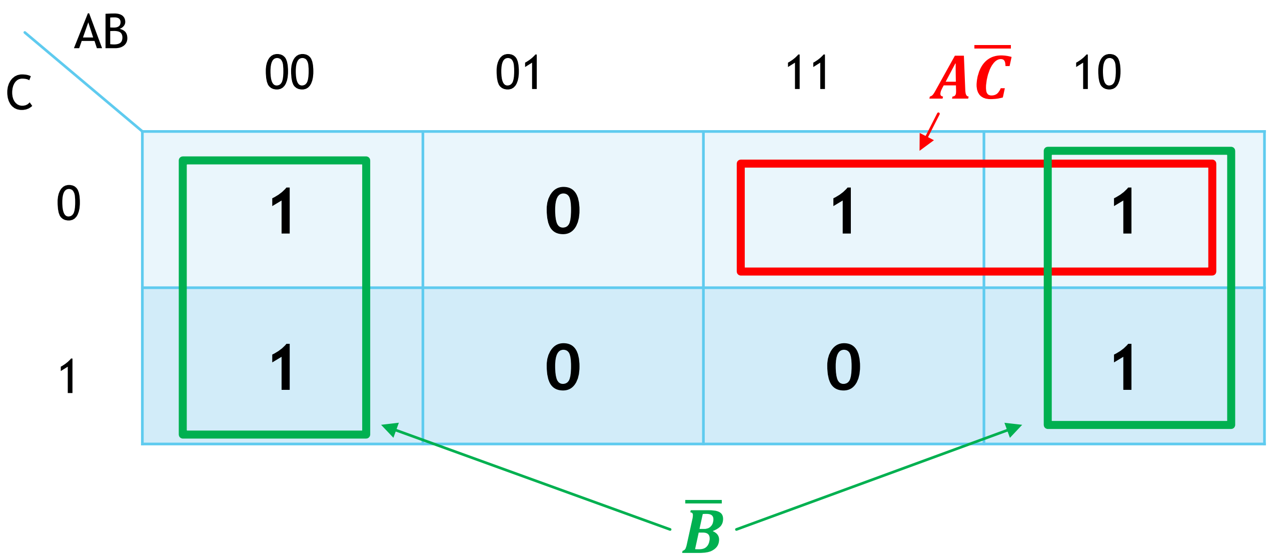
C \ AB	00	01	11	10
0	1	0	1	1
1	1	0	0	1

$A\overline{C}$

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Karnaugh map, 3 variables

- Find the areas that have 1s:
  - Remember that the map is connected at its borders!



A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Karnaugh map, 3 variables

- Find the areas that have 1s:
  - Remember that the map is connected at its borders!
- The simplest logic function is therefore $\boxed{F = \bar{B} + A\bar{C}}$

C \ AB	00	01	11	10
0	1	0	1	1
1	1	0	0	1

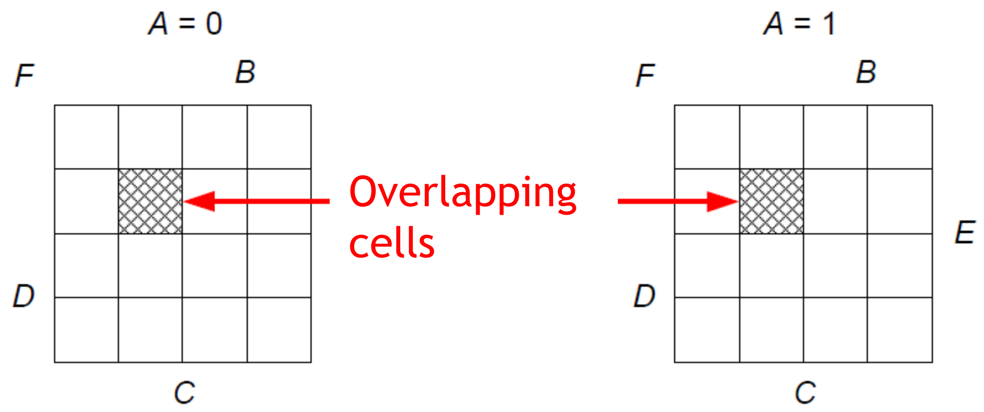
$A\bar{C}$

$\bar{B}$

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Karnaugh map, 5 variables

- In case of 5 variables we need two "overlapping*" 4x4-maps, of which the first one depicts combination BCDE when A = 0 and the latter depicts combination BCDE when A = 1
- In addition to regular adjacence also overlapping cells are considered "adjacent"
- Not nearly as easy as it is for 3 or 4 variables



* Usually these maps are drawn side by side.

Karnaugh map, POS-form

- Karnaugh map can also be used in order to simplify the function in POS-form
- Now we just make a corresponding grouping for combinations which have a truth value of 0
- As a result we get a negation of the logic function (F’) using product-like terms, so now we have to take a negation of this using De Morgan’s II law
- After this the product-form terms can be broken down to sum-form terms using De Morgan’s I law

- An example will (hopefully) make things clearer

Karnaugh map, POS-form

- Define the simplified POS-form function for the truth table shown below.

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Karnaugh map, POS-form

- Let’s move the truth values from table to the map and search for 0s:

C \ AB	00	01	11	10
0	**1**	**1**	**1**	**1**
1	**0**	**1**	**0**	**0**

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Karnaugh map, POS-form

- Let's move the truth values from table to the map and search for 0s:

C \ AB	00	01	11	10
0	1	1	1	1
1	0	1	0	0

AC

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Karnaugh map, POS-form

- Let's move the truth values from table to the map and search for 0s:

C \ AB	00	01	11	10
0	1	1	1	1
1	0	1	0	0

AC

$\bar{B}C$

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Karnaugh map, POS-form

- Let's move the truth values from table to the map and search for 0s:
- The negation of the logic function is $\boxed{\overline{F} = AC + \overline{B}C}$

C \ AB	00	01	11	10
0	1	1	1	1
1	0	1	0	0

AC

$\overline{B}C$

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Karnaugh map, POS-form

- Let's then define the logic function from the negation and simplify it to POS-form:

$$\overline{\boldsymbol{F}} = \boldsymbol{AC} + \overline{\boldsymbol{B}}\boldsymbol{C}$$

Negation from left and right

Karnaugh map, POS-form

- Let’s then define the logic function from the negation and simplify it to POS-form:

$$\overline{\boldsymbol{F}} = \boldsymbol{AC} + \overline{\boldsymbol{B}}\boldsymbol{C}$$

Negation from left and right

$$\boldsymbol{F} = \overline{(\boldsymbol{AC} + \overline{\boldsymbol{B}}\boldsymbol{C})}$$

De Morgan II: $\boxed{\overline{X + Y} = \overline{X} \cdot \overline{Y}}$

Karnaugh map, POS-form

- Let's then define the logic function from the negation and simplify it to POS-form:

$$\overline{\boldsymbol{F}} = \boldsymbol{AC} + \overline{\boldsymbol{B}}\boldsymbol{C}$$

Negation from left and right:

$$\boldsymbol{F} = \overline{(\boldsymbol{AC} + \overline{\boldsymbol{B}}\boldsymbol{C})}$$

De Morgan II: $\boxed{\overline{X + Y} = \overline{X} \cdot \overline{Y}}$

$$= (\overline{\boldsymbol{AC}})(\overline{\overline{\boldsymbol{B}}\boldsymbol{C}})$$

De Morgan I: $\boxed{\overline{X \cdot Y} = \overline{X} + \overline{Y}}$

Karnaugh map, POS-form

- Let's then define the logic function from the negation and simplify it to POS-form:

$$\overline{F} = AC + \overline{B}C$$

Negation from left and right:

$$F = \overline{(AC + \overline{B}C)}$$

De Morgan II: $\boxed{\overline{X + Y} = \overline{X} \cdot \overline{Y}}$

$$= (\overline{AC})(\overline{\overline{B}C})$$

De Morgan I: $\boxed{\overline{X \cdot Y} = \overline{X} + \overline{Y}}$

$$= (\overline{A} + \overline{C})(\overline{\overline{B}} + \overline{C})$$

Double negation:

Karnaugh map, POS-form

- Let's then define the logic function from the negation and simplify it to POS-form:

$$\overline{F} = AC + \overline{B}C$$ Negation from left and right

$$F = \overline{(AC + \overline{B}C)}$$ De Morgan II: $\boxed{\overline{X + Y} = \overline{X} \cdot \overline{Y}}$

$$= (\overline{AC})(\overline{\overline{B}C})$$ De Morgan I: $\boxed{\overline{X \cdot Y} = \overline{X} + \overline{Y}}$

$$= (\overline{A} + \overline{C})(\overline{\overline{B}} + \overline{C})$$ Double negation

$$= (\overline{A} + \overline{C})(B + \overline{C})$$

“Don’t care” conditions

- Sometimes we may have situations, where our truth table includes variable combinations that are not of our interest
- Reasons for disinterest may spring i.e. from
  - The variable combination being unrealistic
  - The variable combination is irrelevant regarding the purpose of the circuit
- This kind of combinations can be marked in the truth table (and hence also to Karnaugh map) as “don’t care” conditions – usually marked by “X”
- This “X” can be 0 or 1 – whichever is more favorable considering the simplification

“Don’t care” conditions, example

- Define a simplified SOP-form for a logic function that fulfills the following truth table.

A	B	C	F
0	0	0	1
0	0	1	X
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	X
1	1	0	1
1	1	1	0

“Don’t care” conditions, example

- Mark the truth values to Karnaugh map and find 1-areas:

C \ AB	00	01	11	10
0	**1**	**1**	**1**	**0**
1	**X**	**1**	**0**	**X**

A	B	C	F
0	0	0	1
0	0	1	X
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	X
1	1	0	1
1	1	1	0

“Don’t care” conditions, example

- Mark the truth values to Karnaugh map and find 1-areas:

C \ AB	00	01	11	10
0	**1**	**1**	**1**	**0**
1	**X**	**1**	**0**	**X**

$B\overline{C}$

A	B	C	F
0	0	0	1
0	0	1	X
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	X
1	1	0	1
1	1	1	0

“Don’t care” conditions, example

- Mark the truth values to Karnaugh map and find 1-areas:

C \ AB	00	01	11	10
0	1	1	1	0
1	X	1	0	X

$\overline{A}$

$B\overline{C}$

Interpreted as 1, so
we get a bigger 1-area

A	B	C	F
0	0	0	1
0	0	1	X
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	X
1	1	0	1
1	1	1	0

“Don’t care” conditions, example

- Mark the truth values to Karnaugh map and find 1-areas:

C \ AB	00	01	11	10
0	1	1	1	0
1	X	1	0	X

$\overline{A}$

$B\overline{C}$

Interpreted as 1, so we get a bigger 1-area

Interpreted as 0, so we don’t need to define a 3rd area

A	B	C	F
0	0	0	1
0	0	1	X
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	X
1	1	0	1
1	1	1	0

“Don’t care” conditions, example

- Mark the truth values to Karnaugh map and find 1-areas:
- Simplified logic function is then $\boxed{F = \overline{A} + B\overline{C}}$

C \ AB	00	01	11	10
0	1	1	1	0
1	X	1	0	X

$\overline{A}$

$B\overline{C}$

Interpreted as 1, so we get a bigger 1-area

Interpreted as 0, so we don’t need to define a 3rd area

A	B	C	F
0	0	0	1
0	0	1	X
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	X
1	1	0	1
1	1	1	0

“Race hazard”

- Logic function’s terms may contain some overlap in which combinations they include
- One could think that this is a bad thing, but in reality, it’s the other way round
- If the terms of the logic function represent areas which are adjacent on the map but contain no overlap, the continuity of the logic function may be compromised
  - Changes are not immediate, remember?
  - This is known as race condition or race hazard
- To tackle the problem, we may want to add redundant terms to our logic function
  - Common especially in problems where the simplicity of the function is not a value on its own
  - Less common when physical gates are used

“Race hazard”

- For example in the following Karnaugh map there’s such a danger:

$B\bar{C}\bar{D}$

$\bar{A}\bar{B}$

CD \ AB	00	01	11	10
00	1	1	1	0
01	1	0	0	0
11	1	0	0	0
10	1	0	0	0

$$F = \bar{A}\bar{B} + B\bar{C}\bar{D}$$

“Race hazard”

- For example in the following Karnaugh map there’s such a danger:

$\overline{A}\,\overline{C}\,\overline{D}$

$B\overline{C}\,\overline{D}$

$\overline{A}\,\overline{B}$

CD \ AB	00	01	11	10
00	1	1	1	0
01	1	0	0	0
11	1	0	0	0
10	1	0	0	0

$$F = \overline{A}\,\overline{B} + B\overline{C}\,\overline{D} + \overline{A}\,\overline{C}\,\overline{D}$$

Dangerous continuity problem can be avoided by adding a redundant term $\overline{A}\,\overline{C}\,\overline{D}$ in the function.

Thank you!