Lecture 3

The Canonical forms of Combinatorial circuits and their minimisation

Combinatorial Circuits

Combinatorial circuits (aka combinational) are those in which variables on the Right Hand Side of the Boolean equations do not appear on the Left Hand Side:

$$R = (A+B)\cdot C$$

$$P = (Q\cdot S)+(T+C)'$$

but not

$$R = (A+R)\cdot(T+C)'$$

This means there is no cycle in a combinatorial circuit!

Canonical Forms

A canonical form is a standard way of writing an equation: a shape or form of equation.

Canonical forms are useful for analysing algorithms for logic gates. For example they can be used to check whether two Boolean expressions are equal. We will also use them for finding the minimum implementation of a circuit.

Minterms

A minterm is simply a product term (ie a conjunction) containing each input (RHS) variable or its negation:

eg (for 3 variables A, B, C):

$A \cdot B \cdot C$, $A \cdot B' \cdot C'$, $A' \cdot B \cdot C$, $A \cdot B \cdot C'$ are all minterms

but not

$B' \cdot C'$

The Canonical Minterm Form

The canonical minterm form of combinational logic is simply a sum of minterms (ie a disjunction of conjuncts):

$$R = A \cdot B \cdot C + A \cdot B' \cdot C' + A' \cdot B \cdot C$$

Note that, for example

$$R = B \cdot C + B' + A' \cdot B \cdot C$$

is a valid Boolean equation, and defines a valid circuit but is not in canonical form.

The Canonical Maxterm Form

We noted in Lecture 1 that every Boolean equation has a dual form found by swapping the AND and OR functions. The dual of the canonical minterm form, is a different equation which is called the canonical maxterm form.

It is the product of disjunctions:

$$R = (A'+B+C)\cdot(A'+B'+C')\cdot(A'+B+C)$$

This form of the equation is also useful in circuit design and analysis.

The Canonical Forms and the Truth Table

One useful feature of the canonical forms is that they can be used to find the Boolean equation that corresponds to a truth table.

A minterm evaluates to 1 for exactly one of the possible combinations of its input variables, and 0 everywhere else, for example:

$A' \cdot B \cdot C$ = 1 iff A=0, B=1 and C=1
0 otherwise

Example: A Majority Vote Circuit

A	B	C	R	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	Minterm $A' \cdot B \cdot C$
1	0	0	0	
1	0	1	1	Minterm $A \cdot B' \cdot C$
1	1	0	1	Minterm $A \cdot B \cdot C'$
1	1	1	1	Minterm $A \cdot B \cdot C$

To find the Boolean equation corresponding to the truth table we find the Minterms of the lines in the table where the output is 1, and add them together:

$$X = A' \cdot B \cdot C + A \cdot B' \cdot C + A \cdot B \cdot C' + A \cdot B \cdot C$$

Example: A Majority Vote Circuit

A	B	C	R	
0	0	0	0	Maxterm A+B+C
0	0	1	0	Maxterm A+B+C′
0	1	0	0	Maxterm A+B′+C
0	1	1	1	
1	0	0	0	Maxterm A′+B+C
1	0	1	1	
1	1	0	1	
1	1	1	1	

To find the canonical Maxterm form we find a Maxterm for each line of the table where the output is 0, and multiply them together:

$$X = (A+B+C)\cdot(A+B+C')\cdot(A+B'+C)\cdot(A'+B+C)$$

Finding the Canonical form of an Equation

It is possible to find the Canonical form of any Boolean Equation. We use algebraic simplification to get it into a sum of products, and then carry out the process known as augmentation. For example:

$$X = B \cdot C + A' \cdot B \cdot C$$

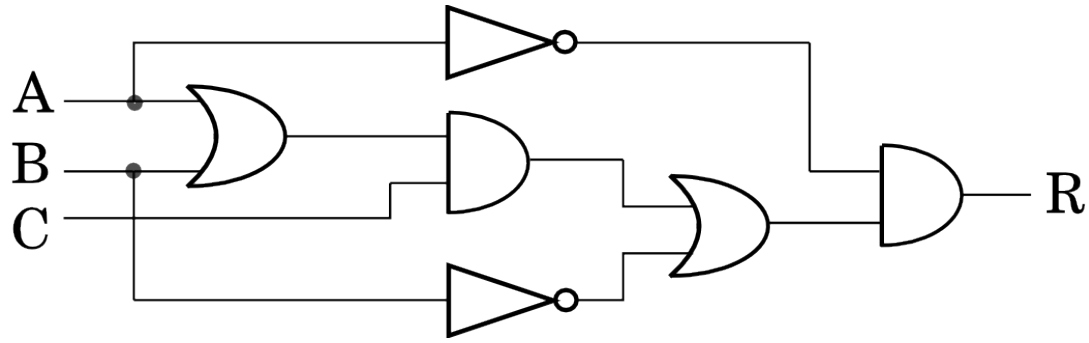
We know that $(A+A') = 1$ and so we can write:

$$X = (A+A') \cdot B \cdot C + A' \cdot B \cdot C$$

$$X = A \cdot B \cdot C + A' \cdot B \cdot C + A' \cdot B \cdot C$$

which is now in Canonical form.

Finding the Canonical form of a Circuit



By inspection the above circuit has equation:

$$R = A' \cdot (B' + C \cdot (A + B))$$

which can be simplified by multiplying out

$$R = A' \cdot B' + A' \cdot C \cdot A + A' \cdot C \cdot B$$

$$R = A' \cdot B' + A' \cdot C \cdot B$$

$$R = A' \cdot B' \cdot (C + C') + A' \cdot C \cdot B \text{ (Augmentation)}$$

$$R = A' \cdot B' \cdot C + A' \cdot B' \cdot C' + A' \cdot C \cdot B$$

Why use the Canonical forms?

Clearly the canonical form is not the smallest circuit representation.

However, it can be obtained automatically from a truth table, and easily from boolean equations and circuits.

The canonical form is used in automated design and analysis of circuits.

Circuit Minimisation

In designing a circuit implementing a Boolean equation we usually aim to find either the smallest or fastest circuit.

Thus, having obtained an equation in canonical form we carry out the process of minimisation.

However, this is no easy task.

Consider again the majority voter

$$X = A' \cdot B \cdot C + A \cdot B' \cdot C + A \cdot B \cdot C' + A \cdot B \cdot C$$

We can use factorisation (the opposite of augumentation) to find simplifications:

$$X = A' \cdot B \cdot C + A \cdot B' \cdot C + A \cdot B \cdot (C' + C)$$

$$X = A' \cdot B \cdot C + A \cdot B' \cdot C + A \cdot B$$

and that is as far as we can go

or is it ??

Further Minimisation is Possible

Suppose we do the following:

$$X = A' \cdot B \cdot C + A \cdot B' \cdot C + A \cdot B \cdot C' + A \cdot B \cdot C$$

$$X = A' \cdot B \cdot C + A \cdot B' \cdot C + A \cdot B \cdot C' + A \cdot B \cdot C + A \cdot B \cdot C + A \cdot B \cdot C$$

we can now find more factorisations

$$X = (A'+A) \cdot B \cdot C + (B'+B) \cdot A \cdot C + (C'+C) \cdot A \cdot B$$

giving

$$X = B \cdot C + A \cdot C + A \cdot B$$

which is the simplest form(!)

The Karnaugh Map

Since we are not all brilliant mathematicians it is unlikely that we will find simplifications of the kind used in the last slide.

However, there is a simple graphical aid to factorisation known as the Karnaugh map.

The Karnaugh map is a truth table written out in a particular form.

Three Input Karnaugh Map

This is the Karnaugh Map for the three input majority voter:

		BC			
		00	01	11	10
A	0	0	0	1	0
	1	0	1	1	1

Notice that it is laid out so that only one of the variables changes in adjacent columns.

Karnaugh Maps for Majority Voters

A \ B	0	1
0	0	1
1	1	1

Two Input OR gate

A \ BC	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Three input majority circuit

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

Four Input Majority Circuit

Factorisations and the Karnaugh map

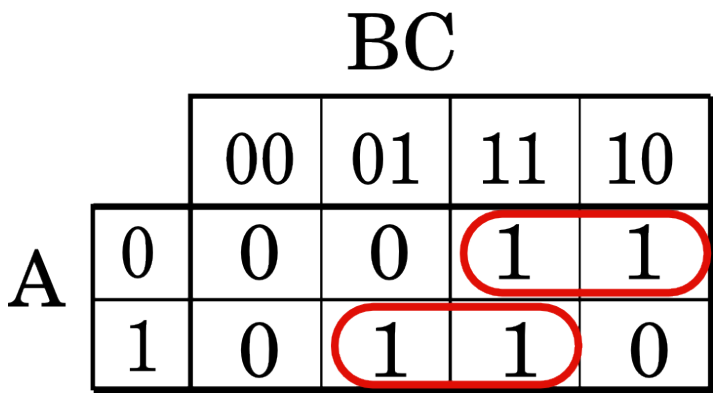
Consider the adjacent 1s circled together on the Karnaugh map below:

		BC			
		00	01	11	10
A	0	0	0	1	0
	1	0	1	1	1

They represent the condition in which both B and C are 1 regardless of A.

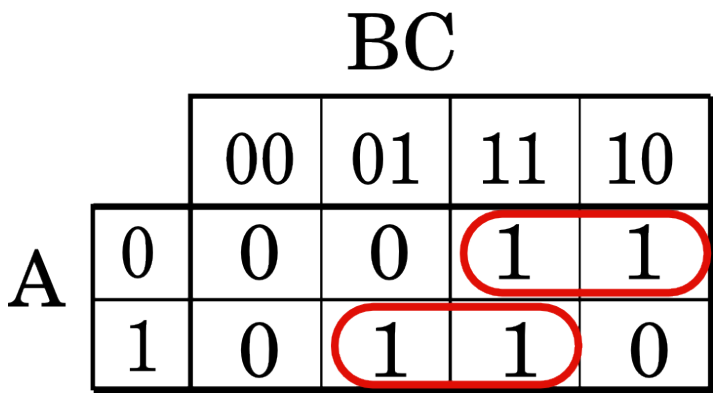
This can be represented by the equation B·C = 1

Problem Break



What conditions are represented by the two circles above?

Problem Break

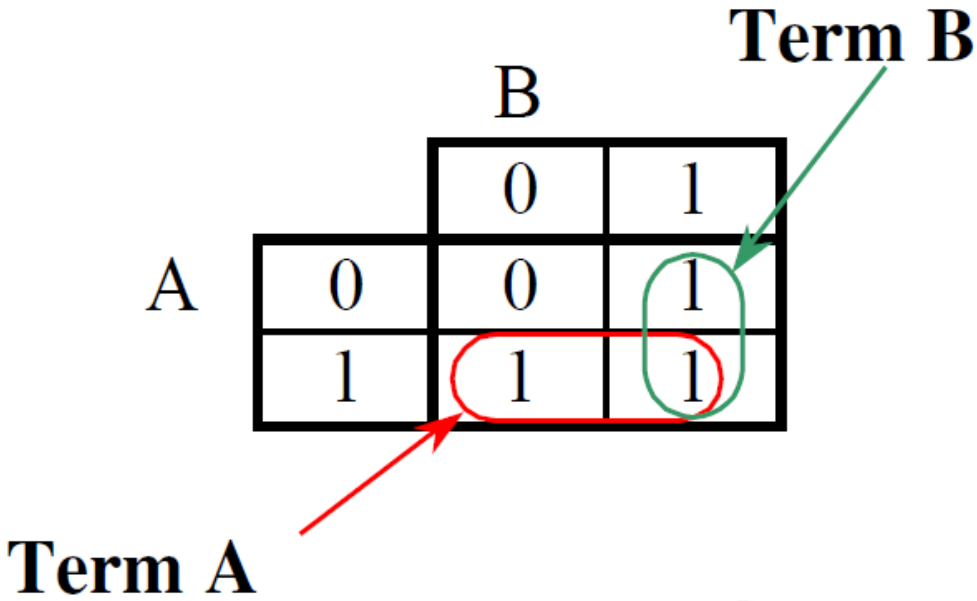


For the upper circle A is always 0 and B is always 1, so the condition is $A' \cdot B = 1$.

For the lower circle A is always 1 and C is always 1, so the term is $A \cdot C = 1$

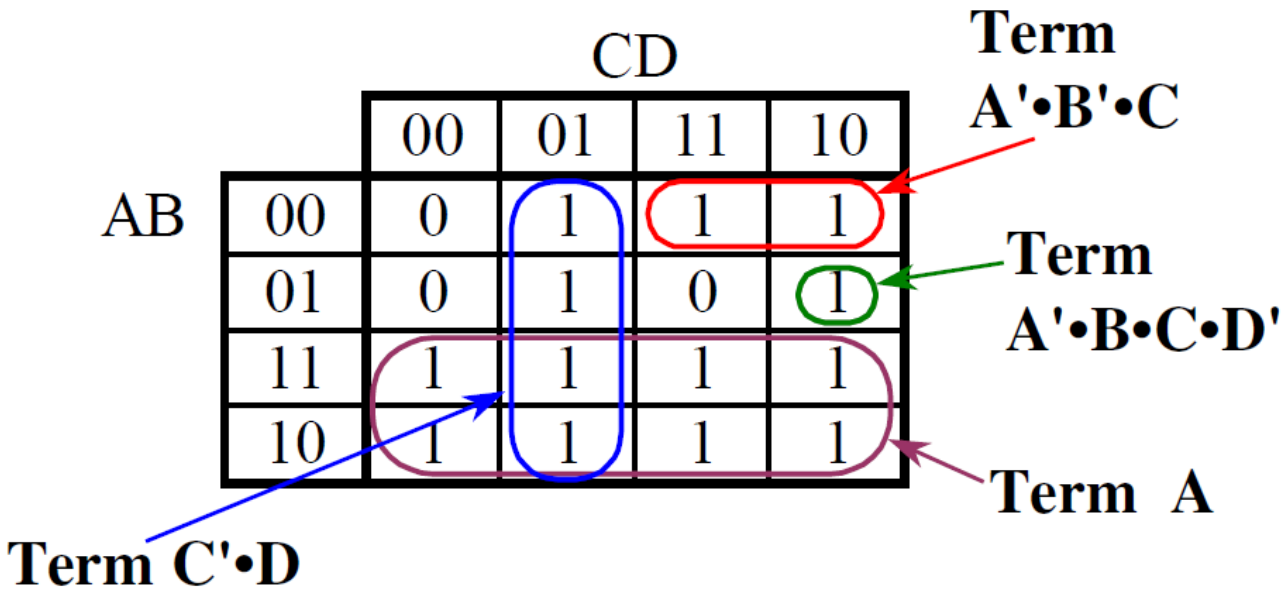
Circles on Karnaugh Maps

Circles are drawn on Karnaugh maps to indicate factorisations of the Canonical equation. Each circle will be represented by one term in the minimised equation



Circles on Karnaugh Maps

The bigger the number of 1s in a circle, the smaller the term in the minimised equation.



Circles on Karnaugh Maps

A circle represents a valid factorisation of the canonical maxterm form if:

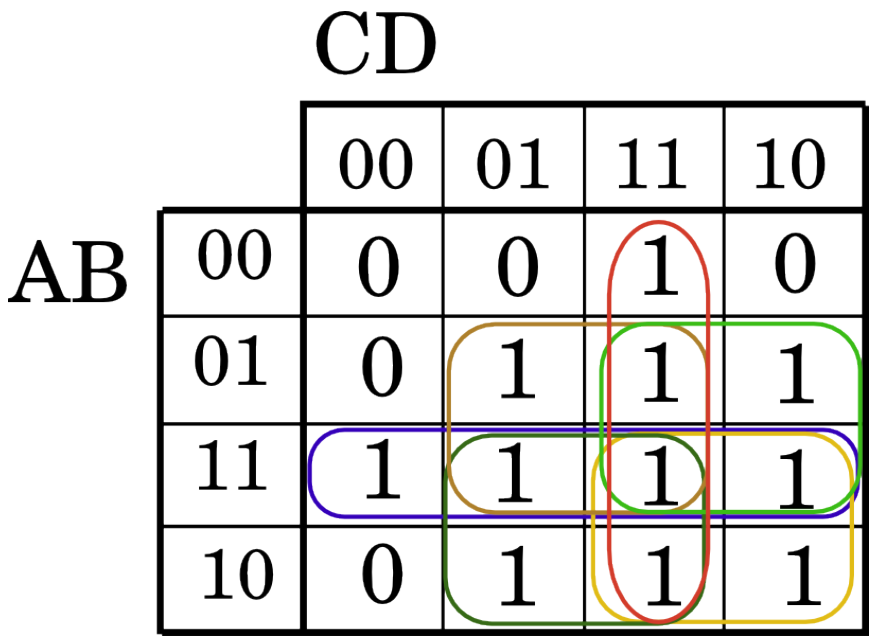
- The circle covers only ones
- The circle extent in both directions is a power of 2: ie 1, 2 or 4 but never 3!

The larger the circles the better.

Covering all the 1s in the Karnaugh Map

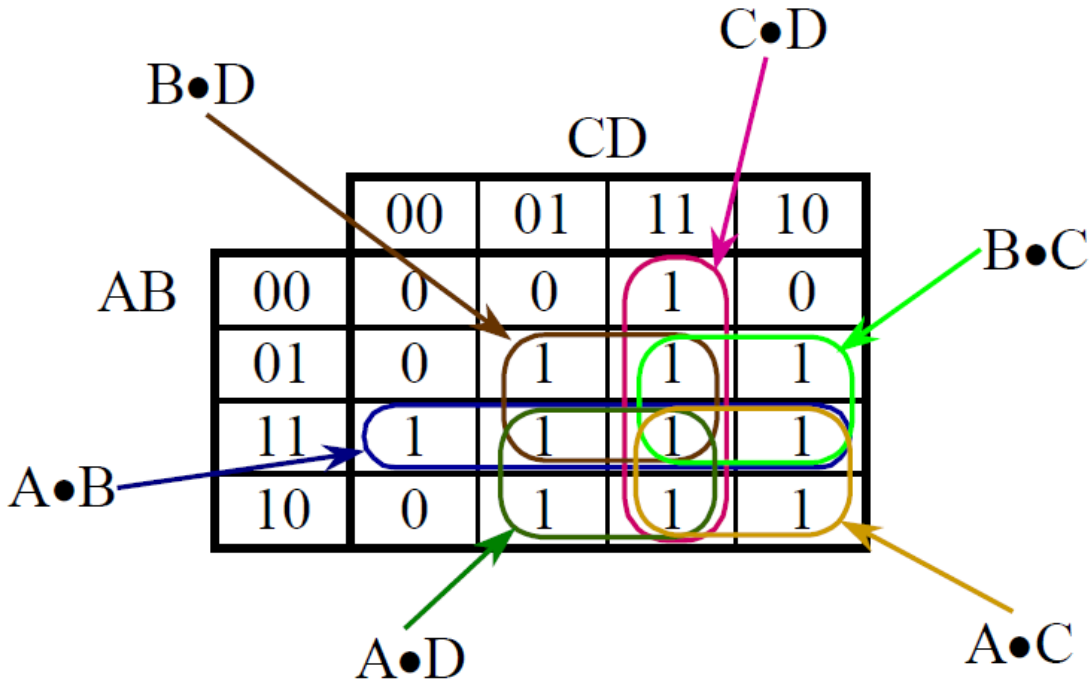
To find a minimised circuit we need to cover all the 1s in the Karnaugh Map with circles, and none of the zeros.

A 1 may be covered by many circles



Finding the Minimised Equation

Each circle contributes one term to the minimised equation:



$$X = A \cdot B + A \cdot D + A \cdot C + B \cdot C + C \cdot D + B \cdot D$$

Cycles in the Karnaugh Map

We noted earlier that the Karnaugh map is organised so that adjacents rows and columns differ by only one variable value.

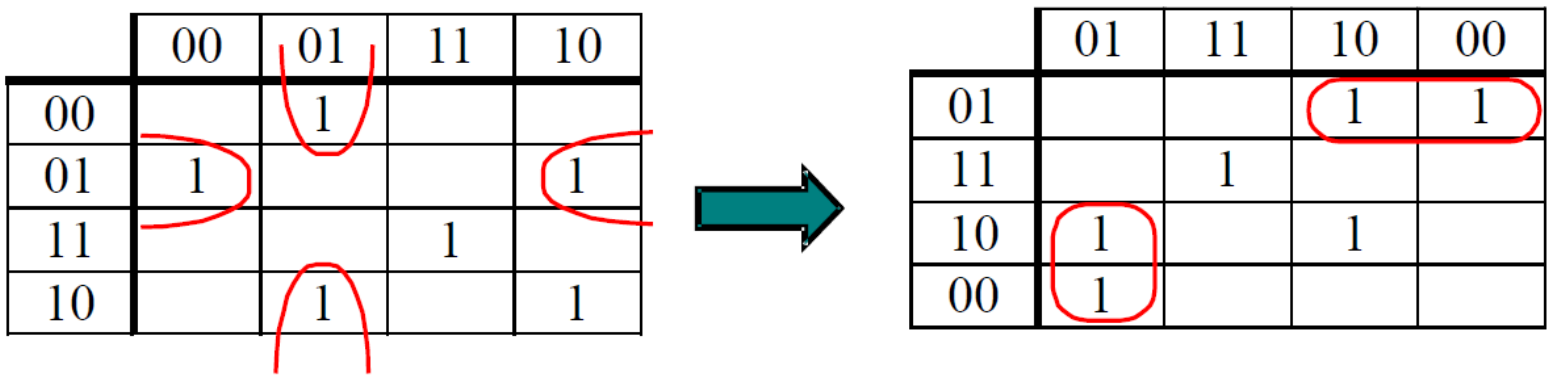
We used the ordering 00 01 11 10, but we could also have used 01 11 10 00 among others. The choice of labelling for the Karnaugh map can alter the ease with which we can spot factorisations.

	00	01	11	10
00		1		
01	1			1
11			1	
10		1		1

	01	11	10	00
01			1	1
11		1		
10	1		1	
00	1			

Circling adjacent edge points on the Karnaugh Map

We can also think of the Karnaugh map opposite edges as being adjacent, and circle the adjacent edge squares:



Don't Cares

It often happens in digital design that we know certain input combinations to a circuit will not occur.

We call these combinations "don't cares" because we don't care what the circuit does if it receives one.

Furthermore, we can use "don't cares" to design a smaller circuit implementation.

On the Karnaugh Map we denote a don't care by $\times$.

Don't Cares

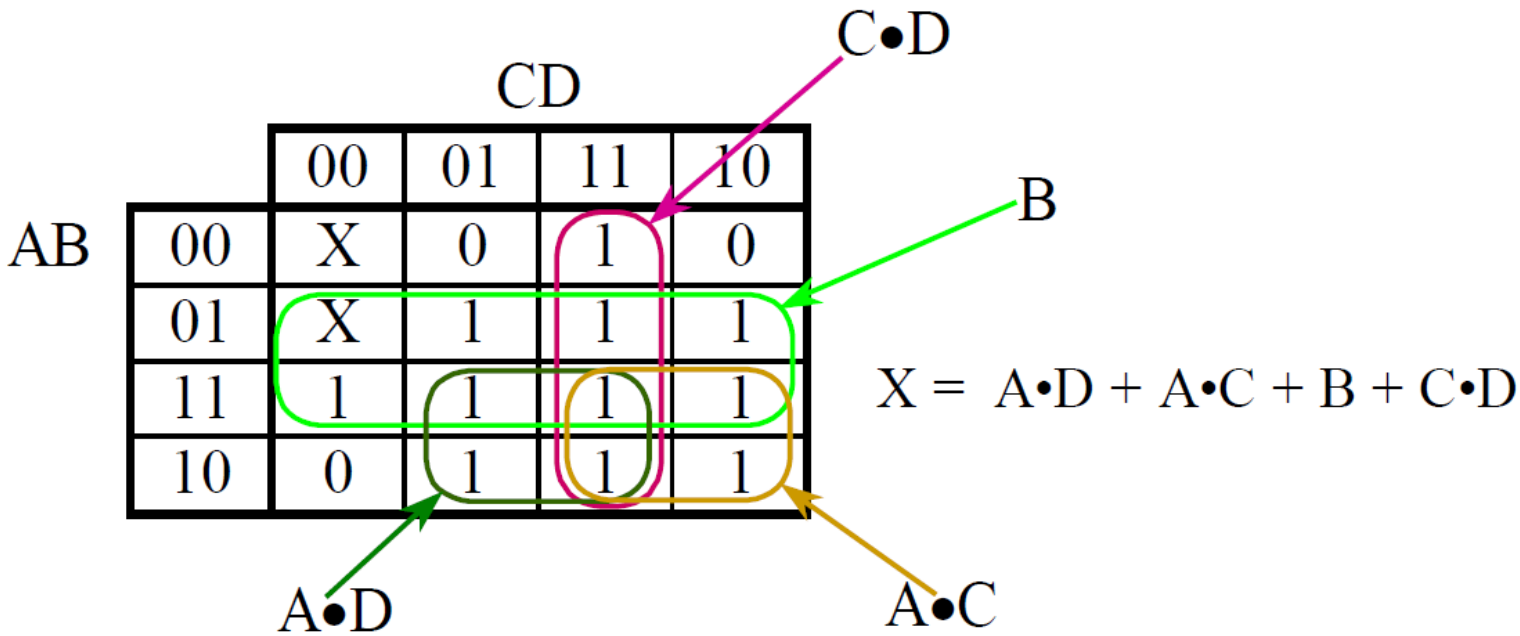
Suppose we are designing a majority voter and we know that inputs 0000 and 0100 will never occur.

The Karnaugh map becomes:

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

Don't Cares

We can choose to make the don't cares either 1 or 0 in order to make our circles as big as possible.



This can often make considerable simplifications to the final design.