Lecture 3: The Canonical Form of Combinatorial Circuits and its Minimisation

This lecture introduces a methodology for designing combinatorial digital circuits. These are circuits whose outputs are defined as functions of their inputs only, and in particular not in terms of their outputs. That is to say the circuits are defined by a set of equations of the form:

$$R_1 = f_1(A, B, C, D) \qquad R_2 = f_2(A, B, C, D) \qquad R_3 = f_3(A, B, C, D)$$

where R_1, R_2 and R_3 are outputs, A, B, C and D are inputs (not equal to R_1, R_2 or R_3) and f_1, f_2 and f_3 are Boolean functions. Circuits where the outputs can appear on the right hand side of the equation as well as the left are called sequential circuits, and we will deal with them later in the course.

There are two main criteria for judging the quality of any digital circuit that is designed. Firstly it should be fast and secondly it should be small. We will now discuss a design methodology for digital circuits that aims to make them as small as possible. We have already seen that circuits can be represented by Boolean equations, and their behaviour can be described by a truth table. Boolean equations are a good representation for formalising what a circuit should do. It is always possible to generate a truth table from a set of combinatorial Boolean equations, and we will use the truth table as the starting point for a design,

As a preliminary we need to define *minterms* and *maxterms*. Suppose a circuit has n inputs designated $A_1, A_2, ..., A_n$. A minterm is simply a Boolean product term in which for each input, say A_k, either A_k or its complement A'_k appears exactly once. For example, if we have a three input circuit with inputs A, B and C then: $A \cdot B' \cdot C'$ and $A \cdot B \cdot C'$ are both minterms, but $A \cdot B'$ and $B' \cdot C'$ are not minterms, since they do not contain all the input variables.

A maxterm is the similarly defined as a Boolean sum term in which for each input, say A_k, either A_k or its complement A'_k appears exactly once. If again we have a three input circuit with inputs A, B and C then: $(A + B' + C')$ and $(A + B + C')$ are both maxterms, but $(A + B')$ and $(B' + C')$ are not.

Any combinational circuit can be expressed in one of two canonical forms. (A canonical form is a standard and agreed upon way of writing a mathematical equation.) The minterm canonical form is a set of Boolean equations, one for each output, in which each equation is a Boolean disjunction (sum) of minterms. The maxterm canonical form is a set of Boolean equations, one for each output, in which each equation is a Boolean product of maxterms

As a simple example consider the three input majority circuit which has output 1 when two or more of its inputs are 1. Let the inputs be A, B, and C and the output be R. The truth table is easily seen to be:

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 $A \cdot B \cdot C$

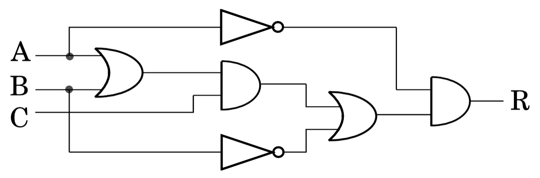
For each input state where $R = 1$ a corresponding minterm is shown. For that given input only it will evaluate to 1. The sum of all these minterms will clearly produce the correct output, and the canonical equation is:

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

The canonical maxterm form can be found by determining the max terms that define all the zeros in the output, and then multiplying them together. If none of the maxterms evaluates to zero then the result will be 1.

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

For most of the course we will use the minterm canonical form as the starting point for our design methodlogy. It is also known as the sum-of-products or the disjunction-of-conjunctions form. In practice we can think of it as a more formal specification of a truth table. However, we can derive the canonical form directly from equations, or a circuit, without drawing up a truth table. For example consider the following badly designed circuit:



Working backwards from the output R we can determine its equation which is:

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

For this, as for any equation we can multiply out the brackets. This yields:

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

and since $A \cdot A'$ is always 0 this simplifies to:

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

We have one minterm $A' \cdot B \cdot C$, but the first term does not contain input C. We solve this problem using augmentation. Since, for any Boolean variable (C+C') = 1, we know that:

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

and so the canonical form is:

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

The strength (and purpose) of the canonical form is that it allows us to devise a simple algorithm for designing any circuit automatically, starting from a set of Boolean equations and finishing with an integrated circuit. We will see next lecture the structure of an integrated circuit, called a programmable logic array, which supports this design methodology. However, we also note that the canonical form is not the smallest representation of most circuits, and consequently not the best implementation according to the size criterion. We therefore now consider how to transform the canonical form into the smallest circuit that implements it. This, theoretically is done by factorisation and simplification (essentially the reverse of the augmentation process that we used in deriving the canonical form). For example if we consider the majority circuit defined by the truth table above:

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

we can see that we can factorise $A \cdot B$ out of the last two terms giving

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

This factorisation has already reduced the number of gates that we will need in the implementation. However, this is not the minimum form of the circuit, which can be derived by first expanding the expression to:

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

and then applying the three possible factorisations to get the minimal form:

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

Factorisations of this kind are in general not easy to derive; one practical visual aid to find them is called the Karnaugh Map. The Karnaugh Map is simply the truth table written out as a two dimensional array. The format for two, three and four variables for the majority circuit are as follows.

		B	
		0	1
A	0	0	1
	1	1	1

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

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

Karnaugh maps for majority circuits

Notice that the order in which the input values are written preserves the rule that adjacent values change in only one variable. Each non-zero square represents a minterm which will appear in the canonical equation. Factorisations can be found by identifying adjacent ones in either the horizontal or the vertical directions (but not the diagonal direction.

Consider the two input map first. This corresponds to the canonical equation:

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

Looking at the map we can see two possible simplifications, or factorisations of the expression which are:

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

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

In practice we would apply both of these to obtain the minimal form which is $X = A + B$. To do this we indicate simplifications by circling the adjacent ones as shown in Figure 1. Each circle will represent one term in the expression (NB we use the term circle loosely here to mean a closed boundary around adjacent squares on the Karnaugh map). The fact that a 1 in the Karnaugh map may appear in more than one circle is irrelevant. In Figure 1 we see that the last row circled together is independent of B, and corresponds to the term A, and similarly the last column corresponds to the term B in the simplest form.

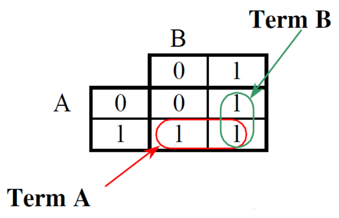


Figure 1: Karnaugh Map of the OR gate

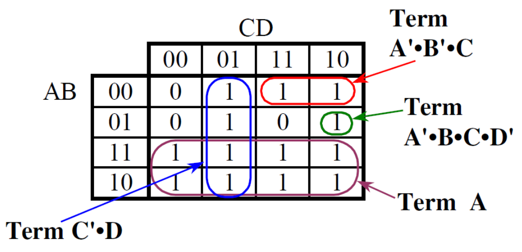
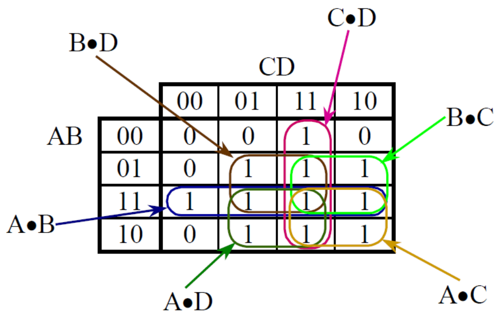


Figure 2: Karnaugh Map Terms

In general a circle on the Karnaugh map of a group of ones represents a factorisation applied to the canonical form. In the Canonical form every one on the Karnaugh map has a corresponding term in the equation. However, if we can circle adjacent ones then only one term, corresponding to the area covered by the circle, need appear in the equation. The areas circled must have side lengths of 1, 2 or 4 - never 3.

Figure 2 shows a four by four Karnaugh map, and indicates some possible circles on it. To find the term that corresponds to a circle we find the inputs that do not vary in that circle. The circle covering the second column is defined by $C = 0$, $D = 1$. The inputs A and B can be either 1 or 0 within the circle. To make sure that the output is a 1 at all four input values within that circle we simply need ensure that $C' \cdot D = 1$. Notice that we always try to find the largest possible circle since this will correspond to the greatest simplification. In Figure 2 we could find a greater simplification by circling together all the ones in the fourth column rather than the single one corresponding to input 0110.

We can now consider the four input majority voter again.



We can see that there are many possible factorisations which can be applied. Consider in particular the square block of four ones in the bottom right hand corner. The top pair corresponds to the minterms $A \cdot B \cdot C \cdot D + A \cdot B \cdot C \cdot D'$ which simplifies to $A \cdot B \cdot C$, and the bottom pair corresponds to $A \cdot B' \cdot C \cdot D + A \cdot B' \cdot C \cdot D'$, which simplifies to $A \cdot B' \cdot C$. There is a further factorisation of these two simplified terms to $A \cdot C$, and this will always be the case for any square block of four, or any row or column of the Karnaugh Map where all the entries are ones. Clearly the bigger the block that we can mark the fewer the variables in each term, and the simpler the expression. Hence we can derive the simplest expression using the six groups of four ones shown above. Inspection tells us which variables appear in the terms. These are just the ones that are always constant in the circled block. From the Karnaugh map we read the simplified expression for the four input majority voter as:

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

The Karnaugh map must be treated as cyclic, so that the last row and column should be considered to be adjacent to the first. We can make this explicit by moving the top row to the bottom, and then the first column to the right hand side end. Consider the four input map shown in Figure 3. Without cycling, there are no apparent simplifications, but when the cycle is made explicit the two possible simplifications become clear.

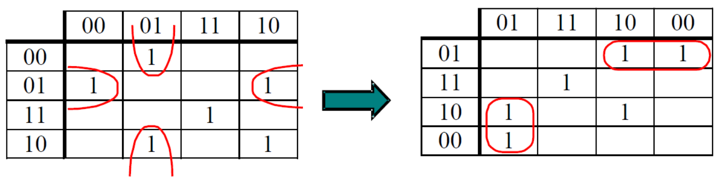


Figure 3: Cyclic Karnaugh Map

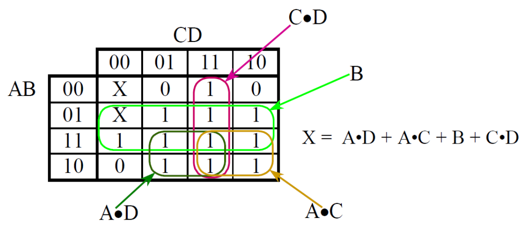


Figure 4: Karnaugh Map with don't cares

In many applications we may know that some possible input states may never occur, and therefore we do not care about the outputs of the circuit. We can make these explicit in the truth table by placing a cross in the output column rather than a zero or one. This has the advantage in minimisation problems that the corresponding points in the Karnaugh map may be treated as either a 1 or a 0 for the purpose of simplification. For example, considering again the four input majority circuit, if we happen to know the input states 0000 and 0100 never occur we can treat them as "don't cares" in the Karnaugh map as shown in Figure 4. Clearly it is advantageous to treat 0100 as a 1 then we can obtain a major simplification with the central block of eight which corresponds to the term B in the second row.

We can apply the principal of duality to all of the preceding material by designing with maxterms. The characteristic of the dual canonical form is that if any maxterms is zero the output is zero. The maxterms are therefore derived from the zeros in the truth table. Karnaugh maps can again be used, but this time it is adjacent zeros that represent possible simplifications. These simplifications follow rules of the form:

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

When circling zeros, don't cares can be applied in the same way, but remembering that simplifications will arise by treating them as zeros not ones. Lastly, it should be noted that the choice of canonical form will often lead to a dramatic simplification. A trivial, but telling example is the design of the OR function.

A	B	C	
0	0	0	Maxterm
0	1	1	Minterm
1	0	1	Minterm
1	1	1	Minterm

The minterm expression which we noted above is $X = A \cdot B + A \cdot B + A \cdot B$ whereas the maxterm expression is $X = (A + B)$.