

INDIAN INSTITUTE OF TECHNOLOGY GUWAHATHI
NPTEL
NPTEL ONLINE CERTIFICATION COURSE
An Initiative of MHRD

VLSI Design, Verification & Test

Dr. Arnab Sarkar
Dept. of CSE
IIT Guwahati

You in the last module we had started our discussion on logic synthesis in this module we will continue the discussion further we said that the basic steps in logic synthesis.

(Refer Slide Time: 00:34)



Logic Synthesis Steps

- The following are done:
 - State Reduction
 - State encoding
 - State assignment
 - Choose resource type (flip-flops, gates etc.)
 - Derive flip-flop input equations and output equations
 - Logic minimization


VLSI Design, Verification and Test 4

Our state reduction state encoding state assignment choose resource types in terms of flip-flops and gates and then derive we flock input equations output equations and those and then minimize the logic expressions that are obtained and after that obtain the final gate level circuit the basic this basic design procedure is essentially the same as any digital sequential circuit design procedure that is taught in basic digital electronics courses and in fact the example that I took in the last module war from marsh mallows famous book digital design so although the essential design principles remain the same due to the sheer size of a VLSI circuits here there are many different types of algorithms applied.

(Refer Slide Time: 01:33)

Two-Level Logic Minimization

- Given:
 - A 2-level boolean function in SoP form.
- Objective:
 - Minimize the number of terms (AND-Gates)
 - Minimize the number of literals (Number of inputs to the gates)



VLSI Design, Verification and Test 16

For example one of the places where various kinds of heuristic algorithms are applied is in logic minimization. So when the number of input variables when the Boolean expressions become huge the Karnaugh map based logic simplification does not suffice anymore and specialized computer aided design tools are required to conduct these simplification procedures today we will discuss two level logic minimization.

So we understand that any Boolean expression can be represented by a two-level logic expression here we have two levels one the first level is an ant plane the second level is an airplane so there will be numerous product terms and there will be an or gate which will sum up all the product terms together to realize the final function however we often do not use two logic expressions.

Because many a times these are practically un realizable there could be an arbitrary number of literals in the and gate and we may not practically have an and gate with that many input available and therefore we need to use multi-level logic expressions moreover although the delay which is primarily represented by the number of levels is lowest in two level logic expressions the area consumed is many a times very huge where the area of the corresponding structural representation here is denoted by the number of literals in the Boolean logic expression so many a times multi-level logic expressions are used in place offer to level logic expressions.

However two level logic expressions are still important because within these multi-level logic expressions two level logic expressions are required hence we will study here the simplification procedures for 2 level logic expressions now what are the objectives of two level logic simplification the objective of two level logics implication is to minimize the number of terms so we want to minimize the number of and gates and also to minimize and minimize the number of literals as we said that is the number of inputs to the gates.

There are different methodologies applied for two level logic minimization the first which we all know and we have studied is the Karnaugh map method many of us have all students also studied the queen lat class key method and what we will study today is almost a standard tool which is used in VLSI design in logic synthesis this is called the espresso algorithm it is a heuristic algorithm for 2 level logic minimization.

(Refer Slide Time: 04:58)

Review of Karnaugh Map Method

Minimum Sum of Products Expression from a K-Map
 • Used for function minimization when no. of variables is small

Step 1: Choose a '1' not already covered
 Step 2: Find "maximal" groupings of 1's and X's adjacent to that element.
 Step 3: Repeat Steps 1 and 2 to find all possible maximal groupings
 Step 4: Find the minimal set of groups that cover all 1's

| | | | | |
|----|------|-----|----|-----|
| | b'c' | b'c | bc | bc' |
| a' | 0 | 1 | 0 | 0 |
| a | 1 | 1 | 0 | 1 |

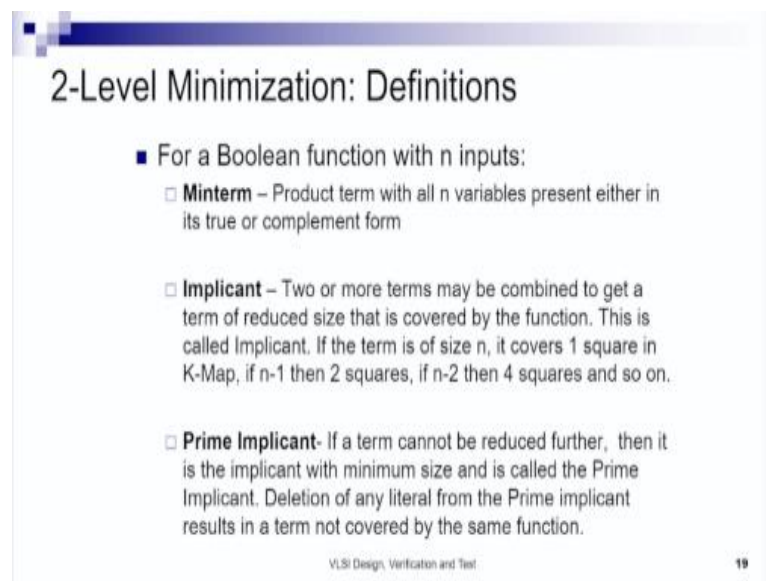
PROBLEM: Number of cells in the K-Map grows exponentially w.r.t no. of input variables; for n inputs, we require 2^n cells

VLSI Design, Verification and Test 18

A brief review of the Karnaugh map method is as follows we choose a one which is not already covered then we find maximal groupings of ones and do not cares adjacent to that element we repeat steps one and two to find all possible maximal coverings from these all possible maximal coverings we choose the minimum set that covers all the ones.

This is the way Karnaugh map works what is the problem which is the basic problem of the Karnaugh map method the number of cells in the k-map grows exponentially with respect to the number of input variables for any inputs were quire 2 to the power n cells more over moreover it is difficult to visually represent a function which is which consists of more than say four variables because the number of cells becomes very huge and therefore simplification you using this method is difficult for a large number of variables this minimization procedure is used usually when the number of variables is small.

(Refer Slide Time: 06:13)



2-Level Minimization: Definitions

- For a Boolean function with n inputs:
 - **Minterm** – Product term with all n variables present either in its true or complement form
 - **Implicant** – Two or more terms may be combined to get a term of reduced size that is covered by the function. This is called Implicant. If the term is of size n , it covers 1 square in K-Map, if $n-1$ then 2 squares, if $n-2$ then 4 squares and so on.
 - **Prime Implicant**- If a term cannot be reduced further, then it is the implicant with minimum size and is called the Prime Implicant. Deletion of any literal from the Prime implicant results in a term not covered by the same function.

VLSI Design, Verification and Test 19

Before proceeding further we will take we will understand a few basic definitions first the definition of a min term what is a min term a min term is a product term in the sum of products expression with all n variables present either in true or complement form so a min term is a product term where we have all the variables of the function present.

Either in its a true form or in its complement form what is in implicant two or more product terms which could be min terms as well may be combined to get a product term of reduced size that covers the that covers both the product terms that we that were originally there so two or more product terms may be combined right to produce a product term of reduced size such that this new product term covers both the original product terms right.

So this is called up this is called an implicant if the term is of size n it covers one square in the Karnaugh map right if it is the mean term it will cover only one term in the carnival if it contains n minus if $n - 1$ if the term is of size n minus 1 it will cover two squares in the card owner if it has n minus 2 right either if the term is of size $n - 2$ it will cover four covers in the carnival and so on.

Now we come to the definition of a prime implicant if a turn cannot be reduced further then it is the implicant with minimum size why cannot it be reduced further because if you try to reduce it further it will cover of whenever we reduce a product term it covers a few more cells in the Karnaugh map and let us say for this case if we try to reduce this product term.

It will cover a few cells not containing once or don't cares so it will cover it will contain a cover why it will contain a covered with some zeros in it right which is not which is not what is desirable which will which is not what is desirable that is error so if a turn cannot be reduced further because it will then cover some of the zeros in the Karnaugh map then it is the implicant with the minimum size that is possible right and is called a prime implicant.

So deletion of any literal from the prime implicant results in a term not covered by the same function so if we if we covered if you try to reduce a prime implicant what will happen is that the function will produce a 1 in some cases where it should not right so this is what is told when we say prime implicant this deletion of any literal from the prime implicant results in a term not covered by the same function.

Now we come to the definition of an essential prime implicant a prime implicant is said to be essential if it covers m in terms which are not covered by others let us consider the Karnaugh map where what do we want to do we want to cover the set of one is by a set of prime implicants and we want to have the minimum number of prime implicants when can we minimize the number of prime implicant when we have a cover of essential prime implicants only write a prime implicant is said to be essential if it covers m in terms which are not covered by others.

This prime implicant possibly contains a set of one is which is covered by other prime implicants but this prime implicant becomes an essential prime implicant when it covers at least 11 in the Karnaugh map that is not covered by any other prime implicant then this

becomes an essential prime implicant ok now we define what is an on-set min terms which are one when the expression is one.

So on set represents the set of min terms which are one when the function itself is one ok offset min terms which are zero when the expression or the Boolean function is one BC said min terms representing don not care terms so their values do not affect the output of the function meaning that we do not observe the output of the function with respect to these min terms right.

Therefore they are mean they are do not care so then what is the basic idea of 11 logic minimization the generic idea of two level logic minimization is to find prime implicants and trying to cover them using a minimal number of covers this is what we said just long the generic idea is to is to find prime implicants and try to cover them using a minimal number of covers that is why we want to find out essential prime implicants because it contains at least one that is not covered by any other prime implicant right obviously a minimum cover will be a cover over essential prime implicants.

(Refer Slide Time: 12:06)

Example using K-Map Method

Essential Primes with Min Cover

VLSI Design: Verification and Test 21

So if it now we take an example a small example using the Karnaugh map method so what do we do let us say we have we have a function which has these set of 10 and do not cares which we represent you now firstly what we do we try to find the cover over primes so here is one

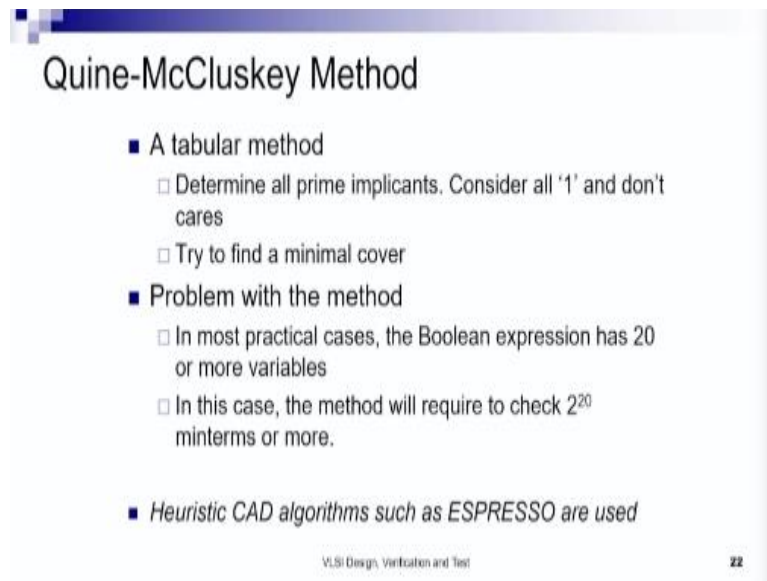
prime implicant which cannot be extended further here is another prime implicant which cannot be extended further here is another prime implicant right again there are other prime implicants.

For example here so all the set of Min term so which cannot be expanded further right by using only the ones and do not care form the prime implicants now we have other prime implicants as well for example this one is a prime implicant this one is another prime implicant right likewise we find out all the prime implicants and then from these from this set of prime implicants which we have been able to find out we try to find out the essential prime implicants for example here when these three prime end.

These three prime implicants here oh sorry here this one and this one these three prime implicants are sufficient to cover all the ones in the function and therefore it is a covered over essential prime implicants and love we do in through Karnaugh map in this way however as we told before Karnaugh map is inefficient because when the number of variables becomes large representing individually through a map is very difficult when we have n variables we need 2 to the power n cells right.

So more than five variables it is very difficult for six variables we need 2 to the power 6 equals 264 cells for seven variables we will require 128 cells and so on In the Queen McCluskey method.

(Refer Slide Time: 14:34)



Quine-McCluskey Method

- A tabular method
 - Determine all prime implicants. Consider all '1' and don't cares
 - Try to find a minimal cover
- Problem with the method
 - In most practical cases, the Boolean expression has 20 or more variables
 - In this case, the method will require to check 2^{20} minterms or more.
- *Heuristic CAD algorithms such as ESPRESSO are used*

VLSI Design, Verification and Test 22

We do not represent it visually in the form of a map it is a tabular method so what we do here basically I will not I am NOT going to discuss the entire method here I will just explain what are the basics fundamentals to this method we determine all prime implicants like in Karnaugh map but in a tabular form and we consider all the ones and do not cares while determining the prime implicants similarly and then we try to find a minimum cover but using a tabular method we have similar problems as we have in Karnaugh map method because in most VLSI circuits that we have in many cases we have at least 20 or more variables in the Boolean expression in the Boolean function.

The number of input variables is 20 or more in this case we have to we have to check 2 to the power 20 min terms because queen McCluskey method is based on checking in terms we have to find min terms and then cover min term through to form prime implicants and therefore we have to check at least 2 to the power 20min terms which is huge so therefore we use a better heuristic CAD algorithms one of the standard tools that we use today is the espresso algorithm we come to the end of this module with this discussion.

Centre For Educational Technology
IIT Guwahati
Production

Head CET

Prof. Sunil Khijwania

CET Production Team

Bikask Jyoti Nath

CS Bhaskar Bora

Dibyajyoti Lahkar

Kallal Barua

Kaushik Kr. Sarma

Queen Barman

Rekha Hazarika

CET Administrative Team

Susanta Sarma

Swapan Debnath