Design Verification

Lecture 03 - Two-Level Logic Verification

1. Logic Verification: Boolean equivalence check of 2 logic circuits; making sure logic errors are not introduced during synthesis/design

2. Miter Circuit

3. Naive approach: exhaustive simulation: 2^N input vectors needed

4. Formal techniques: perform an implicit search; worst case is exponential, but average case is much smaller!

5. A 2-level design may be described as a set of cubes/implicants. Each cube implies that the output is either true (1) or don't care (X); the cubes together form a **cover** for the output function

   Example 1

6. Unate functions and covers:

   - a function is positive unate in variable x if $f_x \supseteq f_{\bar{x}}$
   - a function is negative unate in variable x if $f_x \subseteq f_{\bar{x}}$
   - a function is unate if $\forall x, f_x \supseteq f_{\bar{x}}$ *or* $f_x \subseteq f_{\bar{x}}$
   - a cover is positive unate in variable x if all its cubes have X or 1 in x's field
   - A logic function f is <u>monotone</u> increasing (decreasing) in x_i if a change in x_i from $0 \to 1$ ($1 \to 0$) causes f to change from $0 \to 1$ ($1 \to 0$) or stay constant.

- A function is <u>unate</u> in x_i if it is monotone increasing or decreasing in x_i.
  Example 2

7. Checking for unateness in covers: Given a cover C for f, if a variable x_i is either '-' or '1'/'0' in each cube, then f is unate in x_i.

   Example 3

8. If unate cover $\Rightarrow$ unate function
   If unate function $\not\Rightarrow$ unate cover

9. TAUTOLOGY

   - a cover is a tautology if it has a row of don't cares (tautology cube)
   - a cover is NOT a tautology if it has a column of 0's or a column of 1's (function depends on at least one variable)
   - a cover is a tautology when it depends on one variable only and both 0 and 1 appear under the variable
     1 - $\Rightarrow f = a + \bar{a} = 1 \rightarrow$ tautology!
     0 -
   - a cover is NOT a tautology if it is unate and no row of don't cares

10. 2-Level Logic Equivalence

    Example 4

11. Co-factor: Given a function f, determine what f would be if a given cube c is true, f_c. Similarly, given the cover C for the function, we can compute C_c.

    Theorem: a cover C contains a cube/implicant α iff C_α is a tautology.

12. <u>Containment Check:</u> $c \subseteq D$ if the cofactor D_c is a tautology.

    Taking co-factor on covers:

    - Step 1: eliminate rows that conflict in values with inputs of c_i
    - Step 2: eliminate rows whose output is 0
    - Step 3: eliminate columns for which c_i is specified

    Example 5

    Example 6 (Containment Check)

Example 7 (Containment Check)

13. Verification algorithm: Given two covers C and D, for each cube $c_i \in C$ such that $c_i \subseteq D$, also for each cube $d_j \subseteq C$.

14. <u>Theorem:</u> A unate cover is a tautology iff the cover can be rewritten as one that contains a row of '-'s

    - If an input column of all 1's or all 0's $\Rightarrow$ NOT a tautology
    - If f can be partitioned into $f = g + h$, where g and h have disjoint covers (i.e., no common variables), then f is a tautology iff either g or h is a tautology.

Example 8

Example 9

15. Shannon expansion: $f = x\ f_x + \bar{x}\ f_{\bar{x}}$

16. If f is monotonically increasing (positive unate) in $x_1 \Rightarrow f = x_1\ f_{x_1} + f_{\bar{x_1}}$
    - if $x_1 = 0 \Rightarrow f(x_1 = 0) = f_{\bar{x_1}}$
    - if $f_{\bar{x_1}} = 1 \Rightarrow f \equiv 1$, *since* $f = x_1\ f_{x_1} + 1 = 1$
    - Similarly for monotonically decreasing variables
    - Thus, if we co-factor the cover with the unate variables, and the result is tautology, then the original cover must be a tautology as well

17. <u>Unate Reduction Theorem:</u>

18. Corollary: Let $C = [A|B]$, where A contains all the unate columns of f: if there is NO rows of '-'s in A, then $f \not\equiv 1$. (Because no T can be formed from unate variables)

19. Algorithm:
    - Step 1: rearrange columns of C such that unate rows are placed first
    - Step 2: if no rows in $A(C = [A|B])$, then NOT tautology, else
    - Step 3: rearrange rows to
    - Step 4: repeat tautology checks only on D

Example 10

20. Recap:
    - Verify $f_1 \equiv f_2$? (If 30 primary inputs $\longmapsto$ 1 billion patterns to explicitly simulate in the worst case. Thus, need implicit enumeration techniques.)
    - For every cube in f_1, check if it's contained in f_2 and vice versa
    - Containment using tautology check.
    - Alternatively: XNOR f_1 and f_2 and check for tautology.
    - Can use multi-level verification (next lectures) on 2-level verification as well