VLSI / SOC Testing

Lecture 15

1. Untestable Fault Identification
   - Motivation: ATPG spends lots of time targeting untestable faults
   - Want: quickly identify combinational and sequential untestable faults
   - Fault-dependent approach: analyze per fault
   - Fault-independent approach: based on circuit structure, determine which faults are untestable
2. FIRE:
   - Key concept: identify faults that require conflicting values on a signal in circuit to be testable
   - Algorithm:
     - compute S_0 = all faults that require line $a = 0$ for detection
     - compute S_1 = all faults that require line $a = 1$ for detection
     - any fault in $S_0 \cap S_1$ are untestable

   Example 1:

Example 1 (continued):

3. Effectiveness of FIRE
   - Size of S_0 and S_1 critical
   - S_0 and S_1 depend on the number of implications
   - WANT: as many implications as possible
4. Extended backward implications
   - if z implies an unjustified gate g, then we can learn more on what z can imply on nodes preceding g

Example 2: (Review of backward implication

Example 3:

5. Constant nodes:

6. Extending FIRE to sequential circuits

   - In general, FIRE needs a set of conflicting value assignments $\longmapsto$ illegal/unreachabl states are conflicts

   Example 4:

7. Computing explicit illegal states can be expensive

   - incorporate sequentiality into the implication graph with edge weights
   - sequential conflicts learned!

   Example 5:

8. Managing the size of implication graph important
   - Number of nodes always a constant: $2 \times n$
   - Number of edges can be exponential
   - Need to periodically trim the graph by
     - remove transitive edges (transitive reduction)
     - eliminate equivalent nodes
9. Transitive reduction
   - remove transitive edges to
     - (1) reduce memory/storage requirements for all edges, and
     - (2) reduce the cost of DFS since fewer edges left

   Example 6:

10. Eliminate equivalent nodes
    - identify strongly-connected components (SCCs): nodes in a SCC have paths between every pair

11. Use representative nodes of SCCs only
    - approx. 50% nodes removed
    - many associated edges also removed
12. Using implications for fault-dependent untestable fault identification
    - Associate what necessary values are needed for each fault
    - Recall that in FIRE, S_0 and S_1 are computed as sets of faults untestable when a signal is equal to 0 or 1, respectively
    - Thus, for every fault in S_0, it must require the given signal to equal to 1 to be testable; conversely for faults in S_1
    - At the end of FIRE, every fault would have a list of necessary values $\longmapsto$ check if the list is *consistent*, ie. if the implications for every value in list conflict with each other or not
    - we need not store such lists of necessary values for every fault, only for the faults undetected by random vectors and those missed by FIRE to save storage
13. Algorithm:

For every signal s
    compute S_0
        add $s = 1$ to the lists of every fault in S_0
    compute S_1
        add $s = 0$ to the lists of every fault in S_1

For every undetected fault f
    L_f = list of necessary values for f
    imply each necessary value in L_f
    if conflict occurs
        f is untestable, go to next fault
    else if f can't be excited (implication on fault site = stuck value)
        f is untestable, go to next fault
    else if f can't be propagated (prop path blocked)
        f is untestable, go to next fault

Example 7:

14. In sequential circuits, a fault is present in every time-frame
    $\longmapsto$ thus if a fault is untestable, no vector sequence can detect the multiple fault
    $\longmapsto$ if a fault is combinationally untestable, it is also sequentially untestable

15. Single fault theorem

    - Motivation: using combinational algorithms to identify sequentially untestable faults
    - For each fault f, if it is testable, there must exist a time-frame TF_i during which it is first excited
    - the state for TF_i must be reachable
    - fault f in all time frames less than TF_i are not excited, thus the fault-free value = faulty value

16. Theorem: a target fault that is untestable in $C(n)$ is also untestable in the sequential circuit

    - There does not exist a sequence that can take the circuit from an all-unknown state to one such that the fault could be excited and propagated to at least one FF or PO