Design Verification

Lecture 24 - Diagnosis II

1. We want to further prune the candidate list, thus improving diagnostic resolution
   - Key concepts: eliminate as many false candidates as possible without much cost
2. Flip-fanout-bits method:
   - idea: want to magnify factors that distinguish the true candidate and false candidate regions
   - if region A is a true candidate, then at least one output of the region must be contaminated
     $\longmapsto$ thus, by flipping one bit while keeping the rest X's, we want to see if the erroneous outputs still remain
   - note: a region is falsified only if it fails on every flip

   Example 1

3. Distinguishing X's method:
   - idea: want to prune the propagation of X's
   - if the distinguishing X's do not propagate to any erroneous output, it is a false candidate

Example 2

Example 3

4. Combined method:
   - trivial combination (cascading of flip-fanout + distinguishing X's) won't help
   - alternative: flip one fanout, and keep the rest as dist. X's

Example 4

5. Diagnosing sequential circuits

- issue 1: when the implementation circuit fails on a vector v_i, there is a sequence that first took it to a necessary state
- issue 2: the erroneous sequence may potentially be very long
- issue 3: the error may have been *excited* several time frames prior to actual detection
  $\longmapsto$ need to isolate the time-frame at which the error is excited and propagated to a FF
- approach: instead of simulation the sequence $T[v_0, ..., v_i]$ to find the first vector that the error was excited, simply simulate a subsequence to see if error can still be detected
  $\longmapsto$ Note: the vector with which the error is detected may change

Example 5

6. Region based diagnosis approach can be applied to hierarchical diagnosis
   - start with large regions and only go down in hierarchy on those candidate regions

7. Static approaches to diagnosis
   - idea: record signatures of bit-flips as a dictionary
     $\longmapsto$ don't need to store complete responses for each error/fault, but only record the responses of erroneous outputs
   - using the dictionary, perform diagnosis inductively
   - issue 1: dictionary is built by fault simulation without fault dropping, thus the cost may be high
   - issue 2: for large circuits, dictionary may be large
   - issue 3: dictionary only correspond to a given underlying error/fault model
   - to relieve dictionary construction step, we can drop detected faults early, at a sacrifice of lower diagnostic resolution
     $\longmapsto$ the errors detected by same vector on the same erroneous output no longer distinguishable by the vector set $\longmapsto$ note: the errors detected by same vector but on different erroneous outputs still distinguishable
   - to relieve dictionary size problem, we can store only relevant faults

   Example 6

Example 7

8. Adaptive diagnosis

   - after an erroneous vector is applied, pick the best suitable next vector to apply based on results obtained so far

Example 8

9. The response from actual error may not match any responses in dictionary
   - this is because the dictionary is constructed with respect to a specific error/fault type
   - thus, we must match the closest responses and deduce on that
10. Diagnostic test generation
    - to enrich the erroneous test vector set so that more errors/faults can be distinguished
    - constrained ATPG needed to target a pair of errors (a,b): detect a while not detect b, and vice versa