# VLSI / SOC Testing

## Lecture 20

1. Hybrid techniques for partial scan
   - Combining testability-based, structure-based, ATPG-based, etc.
2. Combining structure-based and ATPG-based
   - obtain subset of valid/reachable states with logic sim of $n$ random vectors
   - map states onto the cycles in S-graph
     $\longmapsto$ this gives a new metric of testability of FFs in the cycle
   - Define: density of encoding: for a group of n FFs:
     density =
   - Define: partial valid state: a subset of $k$ FFs in circuit such that the state represented by this $k$-tuple is valid
   - testability of each FF in a cycle =
   - testability of each FF for all cycles it is involved =
     T(FF) =
     $\longmapsto$ larger T() means harder to test

   **Example 1:**

3. Once a FF is selected for scan, the T() values can change, since the partial valid states are now different

   **Example 2:**

4. Another way to combine structure-based + ATPG based
   - Goal-1: Should we break some FFs that are involved in only self-loops?
   - Goal-2: Should we break large cycles with more than one scan FF? breaking cycle into small segments
   - Example of circuits without cycles: counters
   - Instead of valid states, use aborted states
5. Testability-based + ATPG-based
   - aborted states from ATPG + testability measures
   - find the best scan FFs that can reach most # of aborted states
   - Define: ADP (aborted-fault detectability potential):

**Example 3:**

6. Algorithm:

   while not done
       perform a quick ATPG to gather some aborted states
       compute ADP($FF_i$) $\forall$ unscanned $FF_i$s
       pick a subset of best FFs for scan

7. Making aborted states easier to reach

   - Collect easy-to-reach states as aborted states
   - Want: make aborted states reachable from easy-to-reach states

   **Example 4:**

8. Algorithm:

   while not done
       collect easy-to-reach states
       quick run of ATPG to collect aborted states
       for each aborted state $S$
           if $S$ was made to be reachable, how many other aborted states will also be reachable?
       Pick best aborted state and scan FFs such that it can be reached from an easy-to-reach state

9. High-Level DFT

   - Motivation: insert DFT early on in the design cycle (don't need to wait till gate-level is available)
   - Simple approach 1: avoid data-flow loops in RTL
     $\longmapsto$ break the loops either by (1) scanning, (2) add mux to direct load register, or (3) avoid sharing the same register
   - Simple approach 2: Compute density of registers in circuit
     $\longmapsto$ any register with low density means hard to control

   **Example 5:**

10. Behavioral modification to make circuit more testable

    - Define: locus of execution: a node within the control-data-flow graph is the locus for the system if it is currently executing
    - Define: K-controllability: a decision/branch within the CDFG is K-controllable if the direction of the branch can be controlled by the PI K clocks before the locus reaches that decision node
    - Define: non-controllable: a decision/branch not K-controllable, or K cannot be pre-determined
    - Goal: make every node K-controllable, with a small K

**Example 6:**

11. Nested Loops

    - may need multiple test pins inserted

12. Non-scan high-level DFT

    - Designs contain controller and data-path, interacting via control and status signals
    - Generally, data-path modules easy to test, but the control and status signals hard to control/observe
      $\longmapsto$ module alone is easy to test, but composing a test sequence that involves other controller and data-path modules is hard
    - add controllability/observability to these signals at the high level
      - via muxes, extra test pins, etc.
      - specifying particular values at these control/status signals become easier
    - Advantage over gate-level DFT:
      - no scan needed, thus can test at speed, and reduce test application time
      - at gate-level, many more signals to analyze, may place DFT on extra signals or miss some critical ones
    - Disadvantages: may incur higher area overhead if there are many control/status signals to add DFT to

13. From another angle, we'd like to add DFT such that the resulting ATPG problem becomes as easy as full-scan

    - Break all FF cycles at high-level: make sure registers that form cycles are broken with test pins
    - Circuit becomes like a pipelined one, without need of scan-shifting

    **Example 7:**

14. Once the registers do not form cycles, we can further analyze value reachability, to enhance controllability of registers not directly connected to PIs

15. To reduce area overhead, utilizable paths can be exploited

    **Example 8:**