Design Verification

Lecture 14 - More Sequential Verification

1. Retime: move FFs forward or backward to
   - reduce the clock cycle time
   - minimize number of FFs

   Example 1

2. Retimed circuit functionally equivalent to its original circuit
   - we can apply sequential verification techniques to check for equivalence
   - problem: few candidate equivalent pairs $\rightarrow$ can't reduce circuit much
   - few candidate pairs because the time-frame a particular gate exhibits the expected value changed

   Example 2

3. Delay compensation
   - Define *responses* of a signal: response values of a given signal on a sequence of input vectors
   - Define *delayed-equivalent pair*: (a, b) is a delayed-equivalent pair if the responses of a is equivalent to a delayed version of b (delay can be negative value)
   - restore the FF's original location so that response of a no longer a delayed version of the repsonse of b
4. Identifying delayed equivalent pairs
   - Define *signature* of a signal a: a signature has N fields, where $N = \#$ of primary inputs. i^{th} field indicates the minimum number of flip-flops on the path from PI $\#i$ to the signal a
   - Computation of signatures linear in size of circuit
   - After signatures are computed, compare pair-wise candidate delayed-equiv pairs

   Example 3

5. After delayed-equivalent pairs have been identified, delay compensation is applied to the retimed circuit
   - move flip-flops across gates whose signals are early/late w.r.t the other circuit
   - may change signatures for other signals
   - must iterate

```
Delay_compensation()
    construct miter circuit
    compute signature of signals in both circuits
    derive candidate equivalent and delayed-equivalent pairs
    sort signals in fanout-first order (closest to PO first)
    iterate
        for a candidate delayed-equivalent pair (a, b)
            delay compensate a move on a or b
            update signature for some signals
    return the modifed circuit
```

6. After delay compensation, perform sequential equivalence checking on the transformed circuit
   - more candidate equivalent pairs possible
   - room for more reduction in incremental verification
7. Verification of 2 circuits generated by different synthesis flows
   - two flows may interpret don't care conditions differently
   - causing a false negative problem

Example 4

8. Unreachable states is one type of don't care conditions
9. Other don't care conditions that can arise in a circuit
   - due to temproal or spatial correlations among signals in circuit
   - impossible value combinations at internal signals

   Example 5

10. The don't care conditions can help us identify more equivalent internal node pairs.
    - step 0: FSM traversal to find a subset of unreachable states and identify other don't care conditions
    - step 1: for a candidate pair (a, b), do constrained (ATPG) search to see if $a \equiv b$, while not violating any don't care constraints
11. We can find unreachable states without considering all FFs
    - perform FSM traversal only on a subset of FFs
    - treat the FFs not in the subset as primary inputs
    - FSM traversal now simpler (fewer ff's)
    - subset of FFs identified by random simulation: all the FFs that do not belong to any candidate FF pair in the miter circuit

Example 6 (subset of FFs)

12. FSM traversal
    - Image computation using the subset of FFs (existential quantification on the FFs not in subset)
    - Once fixed point reached, unreachable states = negation the set of reachable states
13. Represent don't cares symbolically
    - Let BDD_{dc} be the ROBDD for the don't care conditions
      $\longmapsto$ this defines the Boolean don't care space
      $\longmapsto$ $\overline{BDD_{dc}}$ is the Boolean care space
    - Let BDD_{miter} be the ROBDD for the miter output
    - Then, $F = BDD_{miter} \wedge \overline{BDD_{dc}}$ is the output function specified over the *care space* (as opposed to don't care space
    - if $F \equiv 0$, then we know the 2 circuits joined at the miter are equivalent, since over the care space, there is no way to obtain a 1 at the miter output
    - Note that BDD_{dc} can include both unreachable states and impossible internal value combinations

Example 7