Digital Logic and Computer Organization

Sequential Logic —Analysis and Design

Symbolic Analysis

- generate expressions describing outputs and inputs of all flip flops
- generate next state equation for each flip flop
- generate next state truth table/K map for each flip flop and truth table/K map for each output
- generate a state table
- generate state diagram or timing diagram

Sequential Circuit Design

- determine the interface based on the specification (# of inputs, # of outputs)
- derive the state diagram based on the specification
- generate state transition table
- perform state reduction to remove redundant states
- perform state assignment
- choose a flip flop (FF) type to realize the circuit and determine the number of FFs needed to represent the states
- generate K map for each input of each FF based on its excitation table
- generate K map for each output
- generate expressions describing outputs and FFs' inputs

Examples

- Counting (up or down, order)
- Recognize bit patterns
- Arbitrary state transition example

Design Issues

- State Assignment
  - Binary — for 8 states: 000, 001, 010, 011, 100, 101, 110, 111
  - Gray Code — for 8 states: 000, 001, 011, 010, 110, 111, 101, 100
  - Hot-One — for 5 states: 00001, 00010, 00100, 01000, 10000

State Reduction/Minimization

- Equivalent states:
  - Two states are equivalent if they are the same state.
  - Two states are equivalent if they have the same output for all inputs and if they transition to equivalent states on all inputs.
- When two or more states are equivalent, these states except one is/are redundant, and can be eliminated.
- Identify equivalent states:
  - Row comparison
  - Using implication table

Using Implication Table

- Construct the state transition table (present state, next state on all inputs, output on all inputs)
- Construct the blank implication table
    - rows associated with S_1 to S_n, columns associated with S_0 to S_{n-1}
- For each pair S_i and S_j, if their output are not entirely the same, cross out the corresponding cell: state S_i and S_j are not equivalent.
- For each blank cell left associated with pair S_i and S_j, list the implied pairs in the cell. Any implied pair that is identical (e.g., S_0, S_0) or the states themselves (S_i, S_j) is omitted. If the cell remains empty, place a check in the cell: state S_i and S_j are equivalent.
- For each cell with implied pairs, check the cell associated with each implied pair, if any of these cells has a cross, cross out this cell.
- If any cell is crossed out in the previous step, repeat the previous step until no cell is crossed out any more.
- Any cell that is not crossed out in the end means its associated state pairs are equivalent states.