ECE380 Digital Logic

State Minimization

State minimization

- For simple FSMs, it is easy to see from the state diagram that the number of states used is the minimum possible
  - FSMs for counters are good examples
- For more complex FSMs, it is likely that an initial state diagram may have more states than are necessary to perform a required function
- Minimizing states is of interest because fewer states implies fewer flip-flops to implement the circuit
  - Complexity of combinational logic may also be reduced
- Instead of trying to show which states are equivalent, it is often easier to show which states are definitely **not** equivalent
  - This can be exploited to define a minimization procedure

State equivalence

- **Definition**: *Two states S_i and S_j are equivalent if and only if for every possible input sequence, the same output sequence will be produced regardless of whether S_i or S_j is the initial state*
- If an input w=0 is applied to an FSM in state S_i and the FSM transitions to state S_u, then S_u is termed a ***0-successor*** of S_i
- Similarly, if w=1 and the FSM transitions to S_y, then S_y is a ***1-successor*** of S_i
- The successors of S_i are its ***k-successors***
- With one input, ***k*** can only be 0 or 1, but if there are multiple inputs then ***k*** represents all the valuations of the inputs

Partitioning minimization

- From the equivalence definition, if S_i and S_j are equivalent then their corresponding ***k-successors*** are also equivalent
- Using this, we can construct a minimization procedure that involves considering the states of the machine as a set and then breaking that set into ***partitions*** that comprise subsets that are definitely not equivalent
- **Definition**: *A partition consists of one or more blocks, where each block comprises a subset of states that may be equivalent, but the states in one block are definitely not equivalent to the states in other blocks*

Partition minimization example

- Consider the following state table

Present state	Next state		Output
	$w = 0$	$w = 1$	z
A	B	C	1
B	D	F	1
C	F	E	0
D	B	G	1
E	F	C	0
F	E	D	0
G	F	G	0

An initial partition contains all the states in a single block

P_1=(ABCDEFG)

Partition minimization example

- The next partition separates the states that have different outputs
  - P_2=(ABD)(CEFG)
- Now consider all 0- and 1-successors of the states in each block
  - For (ABD), the 0-successors are (BDB)
    - Since these are all in the same block we must still consider A, B, and D equivalent
  - The 1-successors of (ABD) are (CFG)
    - We must still consider A, B, and D equivalent
  - Now consider the (CEFG) block

Partition minimization example

- P_2=(ABD)(CEFG)
- For (CEFG), the 0-successors are (FEFF) which are all in the same block in P_2
  - C, E, F, and G must still be considered equivalent
- The 1-successors of (CEFG) are (ECDG)
  - Since these are not in the same block in P_2 then at least one of the states in (CEFG) is not equivalent to the others
    - State F must be different from C, E and G because its 1-successor,D, is in a different block than E, C, and G
- Therefore, P_3=(ABD)(CEG)(F)
- At this point, we know that state F is unique since it is in a block by itself

Partition minimization example

- P_3=(ABD)(CEG)(F)
- The process repeats yielding the following
  - 0-successors of (ABD) are (BDB)
    - A, B and D are still considered equivalent
  - 1-successors of (ABD) are (CFG), which are not in the same block
    - B cannot be equivalent to A and D since F is in a different block than C and G
  - The 0- and 1-successors of (CEG) are (FFF) and (ECG)
    - C, E, and G must still be considered equivalent
- Thus, P_4=(AD)(B)(CEG)(F)

Partition minimization example

- If we repeat the process to check the 0- and 1-successors of the blocks (AD) and (CEG), we find that
  - P_5=(AD)(B)(CEG)(F)
- Since P_5=P_4 and no new blocks are generated, it follows that states in each block are equivalent
  - A and D are equivalent
  - C, E, and G are equivalent

Partition minimization example

- The state table can be rewritten, removing the rows for D, E and G and replacing all occurrences of D with A and all occurrences of E or G with C
- The resulting state table is as follows

Present state	Nextstate		Output z
	w = 0	w = 1	
A	B	C	1
B	A	F	1
C	F	C	0
F	C	A	0

Design Example

- Determine which states (if any) are equivalent in the following Moore state diagram