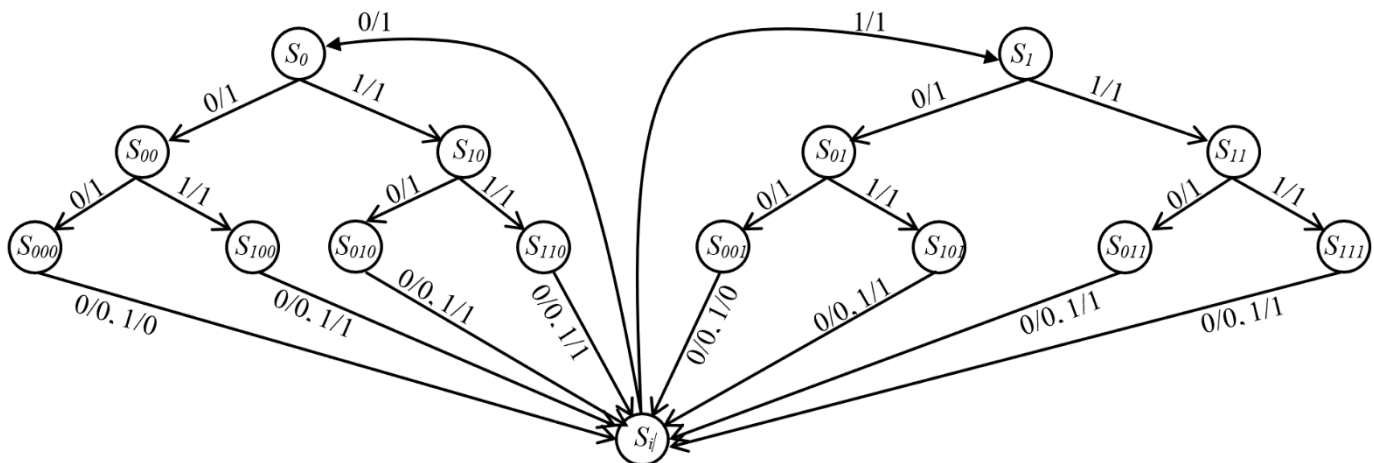


1. Mealy model example

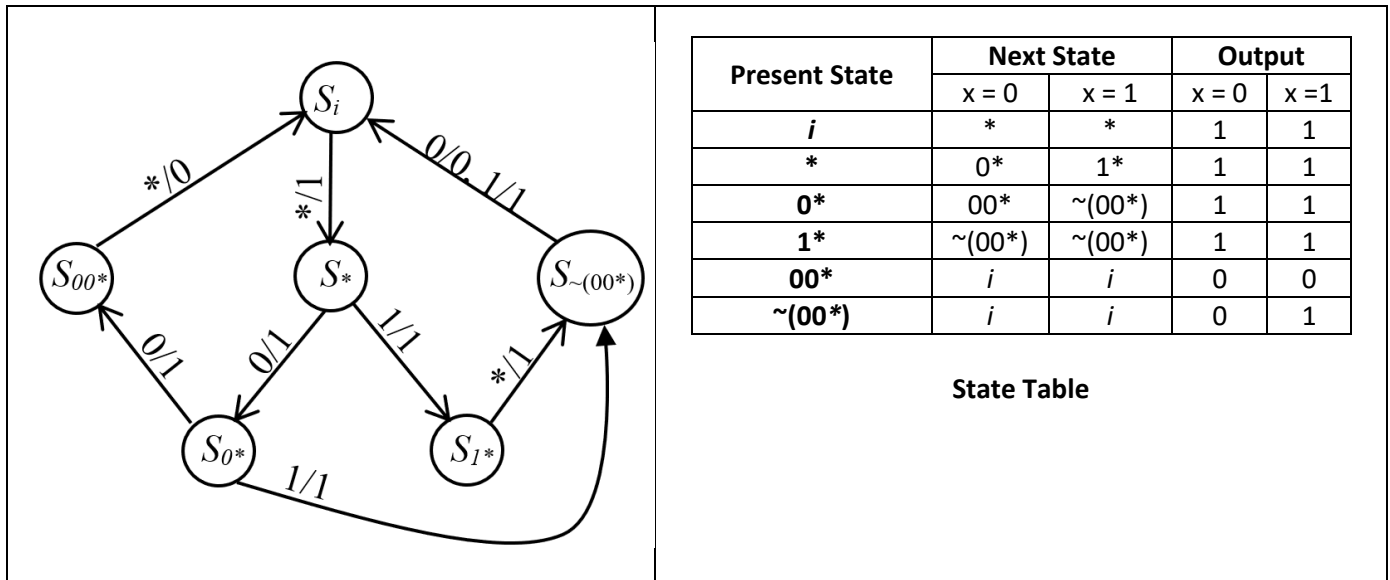
- a. Design a circuit for checking binary coded decimal (BCD)
 - i. BCD uses a 4-bit number to represent a single decimal digit
 - ii. Circuit will take in 4 bits, indicate whether BCD is valid, then take in another 4 bits
 - iii. First bit taken in is least significant, last bit is most significant
 1. This means newest numbers get placed on left side in our diagram below
 - iv. Circuit outputs 0 if the BCD is valid (binary number formed is between 0 and 9) and 1 otherwise
- b. State transition diagram
 - i. Left branch means 0 input
 - ii. Right branch means 1 input



2. Minimizing a Mealy model

- a. Apply the Partitioning Minimization Procedure to reduce the number of states in the Mealy model
- b. Except for state i , we will refer to the states simply by their bit patterns
- c. Since this is a Mealy model, states do not have output values
 - i. The k -successors for a state are the output value created by the combination of the state and the possible inputs
- d. $P_1 = (i, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, 101, 110, 111)$
- e. $P_2 = (i, 0, 1, 00, 01, 10, 11) (000, 001) (100, 010, 110, 101, 011, 111)$
 - i. The first block all have k -successors of 0/1, 1/1
 - ii. The second block all have 0/0, 1/0
 - iii. The last block all have 0/0, 1/1
 - iv. Don't have a fourth block based on the other possible k -successor combination, 0/1, 1/0
 1. No states that have that combination
- f. $P_3 = (i, 0, 1) (00, 01) (10, 11) (000, 001) (100, 010, 110, 101, 011, 111)$
 - i. The first block all have k -successors in the first block of P_2
 - ii. $(00, 01)$ have 0-successors in $(001, 001)$, and 1-successors in $(100, 010, 110, 101, 011, 111)$
 - iii. $(10, 11)$ have both of their k -successors in $(100, 010, 110, 101, 011, 111)$
 - iv. $(000, 001)$ have the single state, i , as their k -successor
 1. We can presume that that block will never split
 2. Ignore it until the end of the procedure
 - v. Similarly, $(100, 010, 110, 101, 011, 111)$ all have a single state, i , as their k -successor
 1. That block will never split either
- g. $P_4 = (i) (0, 1) (00, 01) (10, 11) (000, 001) (100, 010, 110, 101, 011, 111)$
 - i. $(i, 0, 1)$ of P_3 must split
 1. i leads to $(i, 0, 1)$ for both of its k -successors

2. 0 and 1 both lead to (00, 01) for its 0-successor and (10, 11) for its 1-successor
 - h. No need for another partition because all elements in each block of P_4 lead to identical blocks of P_4
 - i. Therefore, P_4 is our final partition, and we will use it to make the state diagram below
3. Implementing the FSM
 - a. We let * stand for either a 0 or 1 at a given position
 - b. $\sim(00^*) = (100, 010, 110, 101, 011, 111)$ in our notation
 - i. Call it so because these states are not $(\sim) 00^*$
 - c. State transition diagram and state table below



4. Assigning binary codes
 - a. Will need 3 flip flops to represent 6 states
 - b. Can assign binary codes randomly, but careful assignment reduces the combinational logic
 - c. Rule of thumb for state binary code assignments
 - i. Try to assign adjacent (Hamming distance of 1) code words to a state and the state that follows it
 - ii. If two present states have the same next state, assign those present states adjacent code words
 - iii. Creating a K-map will help
 - iv. Initial state *i* will always be all 0s
 - d. Assign from the above
 - i. Place *i* at 000, will always do this
 1. Place * next to *i* at 010 since * is the successor to *i*
 2. Place 0* and 1* adjacent to * at 110 and 011 respectively
 3. 00* needs to be adjacent to *i* and 0*, which leaves 100 as the only place
 4. Would like $\sim(00^*)$ to be adjacent to 0*, 1*, and *i*, but that isn't possible
 - a. These are rules of thumb, not fixed laws
 5. Can place $\sim(00^*)$ at 001 to be adjacent to *i* and 1*, though
 - e. Note that there may potentially be more than one valid code assignment that minimizes distance

		AB			
		00	01	11	10
C	0	<i>i</i>	*	0*	00*
	1	$\sim(00^*)$	1*		