

SYNTHESIS OF SEQUENTIAL CIRCUITS

part 1

A sequential circuit is one in which the output depends on both input and state. They are typically represented as a finite state machine(FSM), a series of states and transitions. The two FSMs covered by this course are the Mealy type, in which the current output depends on current state and current input, and Moore type, in which current output depends only on current state.

A FSM can be described by a 6-tuple $\langle I, S, \delta, S_0, O, \lambda \rangle$

-I: input alphabet

-S: finite, non-empty set of states

- δ : next-state function $S \times I \rightarrow S$

- S_0 : set of initial states

-O: output alphabet

- λ : output function $S \times I \rightarrow O$ [Mealy type] or $S \rightarrow O$ [Moore type]

Example: FSM 6-tuple with following set values

$I = \{0,1\}$ $S = \{S1, S2, S3\}$ $O = \{0,1\}$ $S_0 = \{S1\}$

$\delta(S1,0) = S1$ $\lambda(S1,0) = 0$

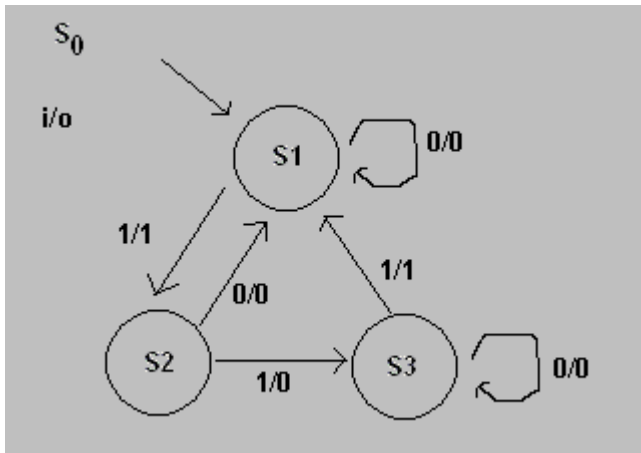
$\delta(S1,1) = S2$ $\lambda(S1,1) = 1$

$\delta(S2,0) = S1$ $\lambda(S2,0) = 0$

$\delta(S2,1) = S3$ $\lambda(S2,1) = 0$

$\delta(S3,0) = S3$ $\lambda(S3,0) = 0$

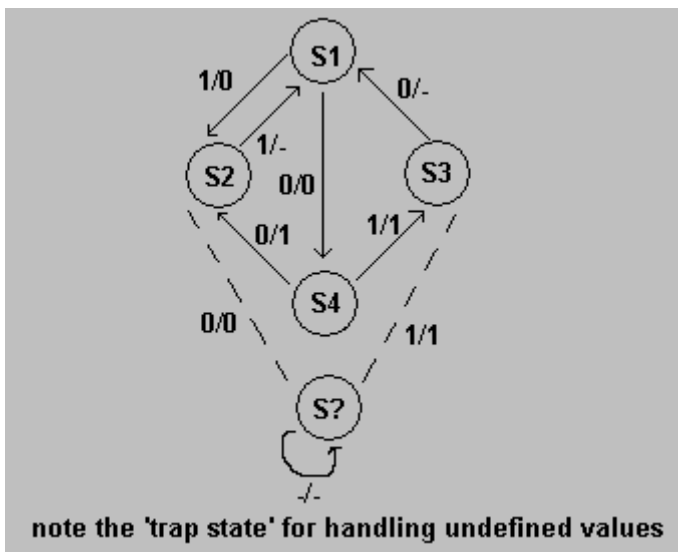
$\delta(S3,1) = S1$ $\lambda(S3,1) = 1$



We can also represent a FSM using a table

ex.

I	PS	WS	O
0	S1	S4	0
1	S1	S2	0
0	S2	-	0
1	S2	S1	-
0	S3	S1	-
1	S3	-	1
0	S4	S2	1
1	S4	S3	1



This is an incompletely-defined state machine because not all transitions and output values are directly specified.

Finite Automata: another way to describe a FSM, using a 5-tuple $\langle I, S, \delta, S_0, F \rangle$

F: set of accepting states

DFA: $\delta: S \times I \rightarrow S$ [one state]

DFA: $\delta: S \times I \rightarrow 2^S$ [many possible states]

MINIMIZATION

We want to minimize cost, so optimize the FSM. Want least number of states while preserving functionality; equivalent states can be combined.

How do we check equivalence and reduce state?

Defn #1: Consider two states s and t of an FSM and a k -string (finite sequence of input) $x = \langle x_0, x_1, \dots, x_{k-1} \rangle$. Suppose string x produces one run of states $S = \langle s_0, s_1, \dots, s_k \rangle$ with $S_0 = s$ and another run $T = \langle t_0, t_1, \dots, t_{k-1} \rangle$ with $T_0 = t$. Let $z^s = \langle z_0^s, z_1^s, \dots, z_k^s \rangle$ and $z^t = \langle z_0^t, z_1^t, \dots, z_k^t \rangle$ be the corresponding output strings. String x is a length k distinguishing sequence for states s and t iff $z_{k-1}^s \neq z_{k-1}^t$.

Defn #2: Two states s and t are k -equivalent $s \equiv_k t$ iff there does not exist a distinguishing sequence of length k or less for these states.

Theorem: Two states are $k+1$ -equivalent, $s \equiv_{k+1} t$ iff $s \equiv_k t$ and $\forall x \in X, s_x \equiv_k t_x$ where s_x and t_x are successors of s and t .