



University of
Massachusetts
Amherst

Engin112 – Lecture 30-31

FSM Design

Maciej Ciesielski
Department of Electrical and Computer Engineering
11/16-18/2011

Today's Lecture

- **Finite State Machines (FSM)**
 - Review
 - » Mealy vs Moore
 - More examples

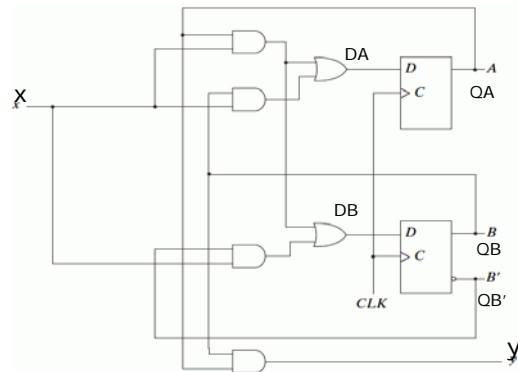
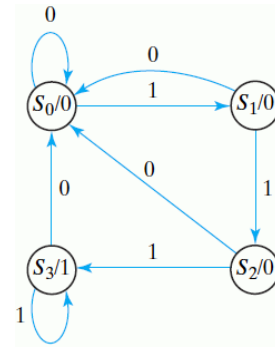
- **State reduction**
 - Reducing equivalent states

FSM Approach to Sequential Circuit Design

Example 1 (review) – sequence detector

- Design a circuit that outputs a 1 when a sequence “111” has been applied to input, and 0 otherwise.”

- Step 1:** derive state diagram
- (Step 2:** reduce number of states)
- Step 3:** encode the states
- Step 4:** create state table
- Step 5:** derive and minimize FF and output equations (use D FFs)
- Step 6:** construct circuit diagram



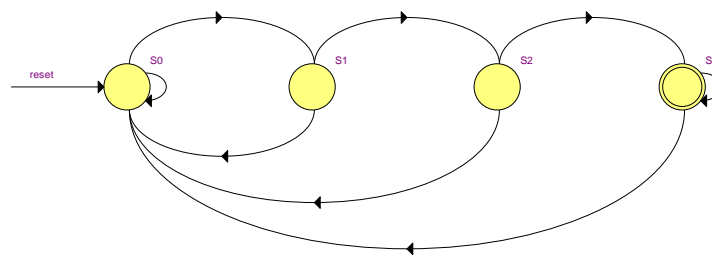
11/16-18/2011

Engin 112 - Intro to ECE

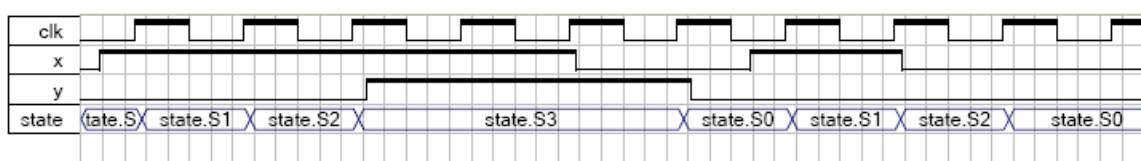
3

Sequence Detector: FSM Synthesis + Simulation

Synthesized Moore FSM (Quartus II)



Simulation results (Quartus II), timing mode



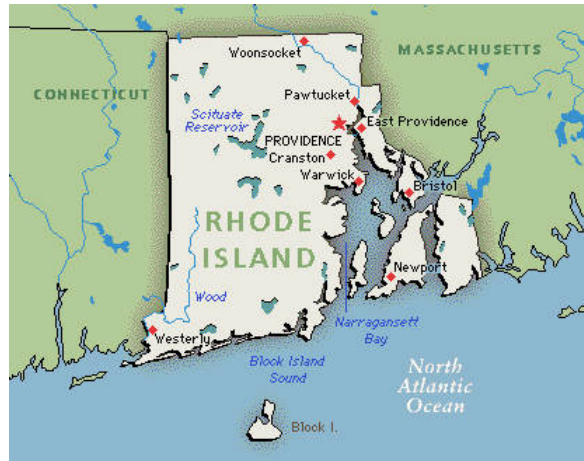
11/16-18/2011

Engin 112 - Intro to ECE

4

State Reduction

- Step 2 in the FSM design process: state reduction
 - Some states are equivalent
 - Can be merged together to minimize the number of states



11/16-18/2011

Engin 112 - Intro to ECE

5

State Reduction

- Equivalent states
 - Have same next state and produce same output under same input
- What do equivalent states “look like”?
 - State diagram: arrows from two states go to same next state and output is the same
 - State table: next state and output entry is same
 - State equations:
 - » $Q_1(t+1) \dots Q_n(t+1)$ and output is the same for two different $Q_1(t) \dots Q_n(t)$ for all inputs
- Which representation is easiest to understand?
 - State table
- Notation:
 - Equivalence class of states:
 - » $\{a,b,c\}$ indicates that states a, b, and c are equivalent

11/16-18/2011

Engin 112 - Intro to ECE

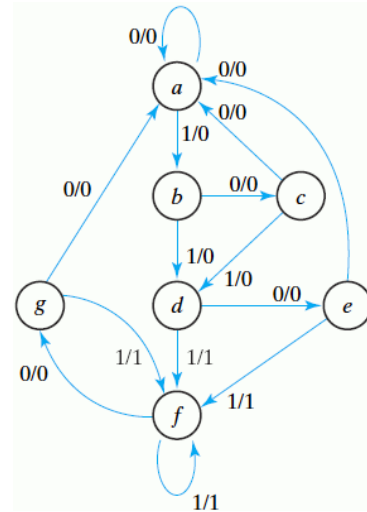
6

State Reduction Example

- Which states are equivalent?

- State table:

present state	next state		output	
	x=0	x=1	x=0	x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	g	f	0	1
g	a	f	0	1



- States e and g are equivalent

- Replace e and g with {e,g} in state table

State Reduction Example

- New state table

- After merging {e, g}

present state	next state		output	
	x=0	x=1	x=0	x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	{e,g}	f	0	1
{e,g}	a	f	0	1
f	{e,g}	f	0	1

- What states are equivalent now?

- States d and f (Note: they were not equivalent initially)
- Replace d and f with {d,f}

State Reduction Example

- New state table:

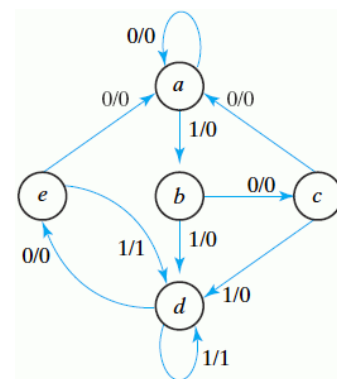
present state	next state		output	
	x=0	x=1	x=0	x=1
a	a	b	0	0
b	c	{d,f}	0	0
c	a	{d,f}	0	0
{d,f}	{e,g}	{d,f}	0	1
{e,g}	a	{d,f}	0	1

- What states are equivalent?
 - No more
 - Rename equivalence classes to states

State Reduction Example

- Final, reduced state table:

present state	next state		output	
	x=0	x=1	x=0	x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1



- Optimization:
 - Skip equivalence classes
 - Rename states directly
 - » E.g., e and g are merged to become “new” e

Try it Yourself!

- Reduce the following FSM:

present state	next state		output	
	x=0	x=1	x=0	x=1
a	b	f	1	1
b	c	e	0	0
c	g	g	1	0
d	f	e	1	1
e	c	e	0	0
f	g	e	0	0
g	g	g	1	0

- How many states do you get?

Solution

present state	next state		output	
	x=0	x=1	x=0	x=1
a	b	f	1	1
b	{c,g}	e	0	0
{c,g}	{c,g}	{c,g}	1	0
d	f	e	1	1
e	{c,g}	e	0	0
f	{c,g}	e	0	0

present state	next state		output	
	x=0	x=1	x=0	x=1
a	{b,f}	{b,f}	1	1
{b,f}	{c,g}	e	0	0
{c,g}	{c,g}	{c,g}	1	0
d	f	e	1	1
e	{c,g}	e	0	0

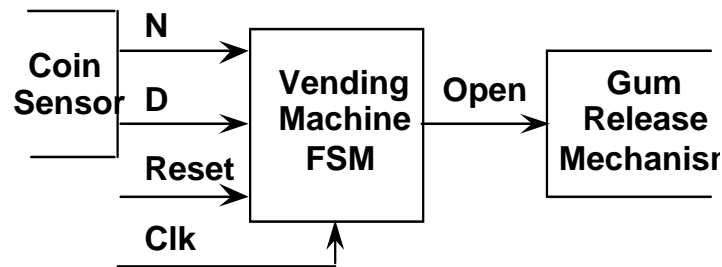
present state	next state		output	
	x=0	x=1	x=0	x=1
a	{b,f,e}	{b,f,e}	1	1
{b,f,e}	{c,g}	{b,f,e}	0	0
{c,g}	{c,g}	{c,g}	1	0
d	{b,f,e}	{b,f,e}	1	1

present state	next state		output	
	x=0	x=1	x=0	x=1
{a,d}	{b,f,e}	{b,f,e}	1	1
{b,f,e}	{c,g}	{b,f,e}	0	0
{c,g}	{c,g}	{c,g}	1	0

present state	next state		output	
	x=0	x=1	x=0	x=1
a	b	b	1	1
b	c	b	0	0
c	c	c	1	0

Example 2: Vending Machine FSM

- Problem specification
 - Deliver package of gum after 15 cents is deposited
 - Single coin slot for dimes (10c) , nickels (5c)
 - No change is given
- Design the FSM using combinational logic and flip flops



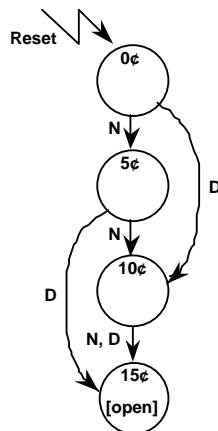
11/16-18/2011

Engin 112 - Intro to ECE

13

Example 2: Vending Machine FSM

- Step 1: Develop state diagram
 - Translate to symbolic state table
 - Take advantage of “don’t care” cases



Reuse states
whenever possible

Present State	Inputs		Next State	Output
	D	N		Open
0¢	0	0	0¢	0
	0	1	5¢	0
	1	0	10¢	0
	1	1	X	X
5¢	0	0	5¢	0
	0	1	10¢	0
	1	0	15¢	0
	1	1	X	X
10¢	0	0	10¢	0
	0	1	15¢	0
	1	0	15¢	0
	1	1	X	X
15¢	X	X	15¢	1

Symbolic State Table

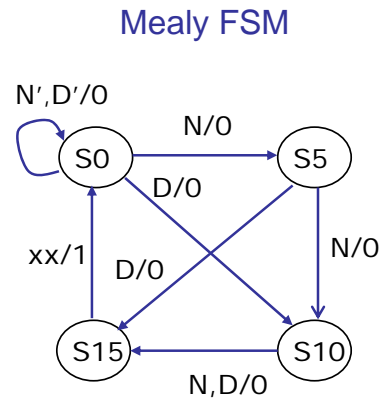
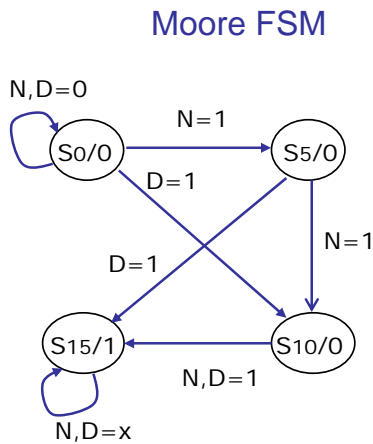
11/16-18/2011

Engin 112 - Intro to ECE

14

Vending Machine: state diagrams

- Step 2: State reduction
 - No equivalent states



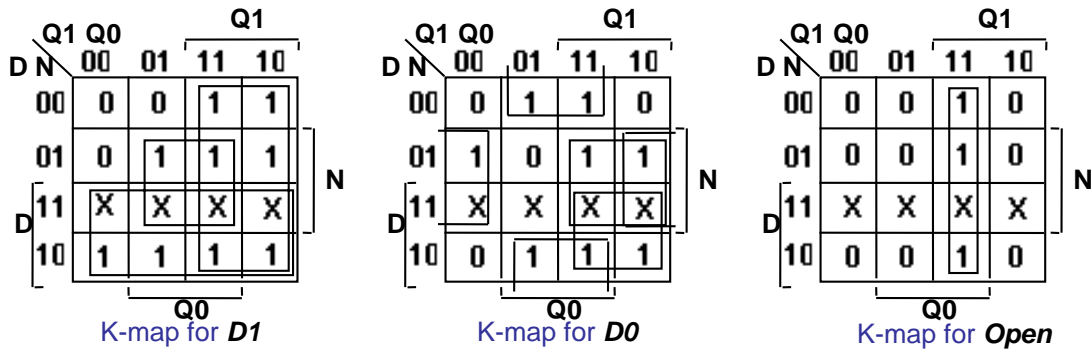
Vending Machine FSM – transition table

- Step 3: State encoding
 - How many FFs are needed?

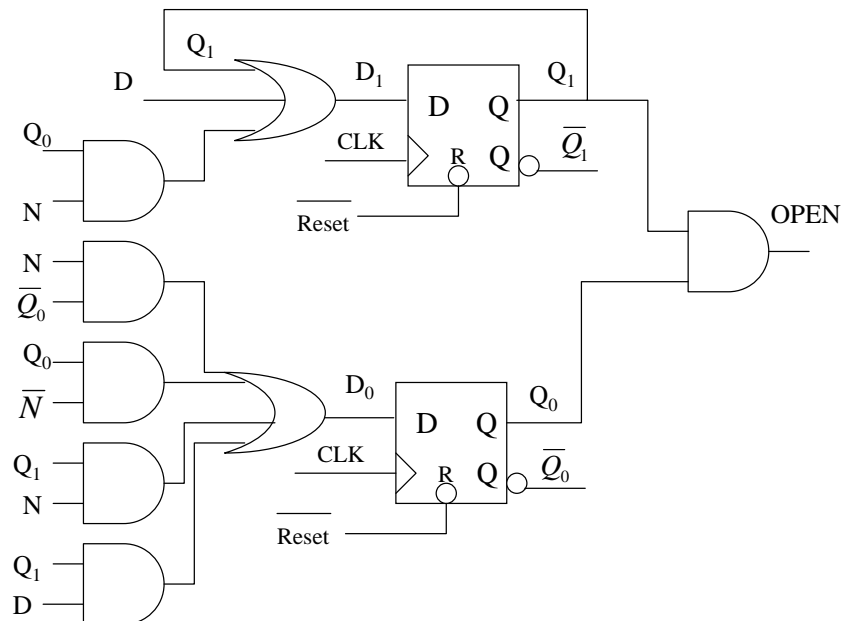
Present State		Inputs		Next State		Output
Q_1	Q_0	D	N	D_1	D_0	Open
0	0	0	0	0	0	0
		0	1	0	1	0
		1	0	1	0	0
		1	1	X	X	X
0	1	0	0	0	1	0
		0	1	1	0	0
		1	0	1	1	0
		1	1	X	X	X
1	0	0	0	1	0	0
		0	1	1	1	0
		1	0	1	1	0
		1	1	X	X	X
1	1	0	0	1	1	1
		0	1	1	1	1
		1	0	1	1	1
		1	1	X	X	X

Vending Machine FSM – K maps

- Step 4: Determine flip-flop implementation



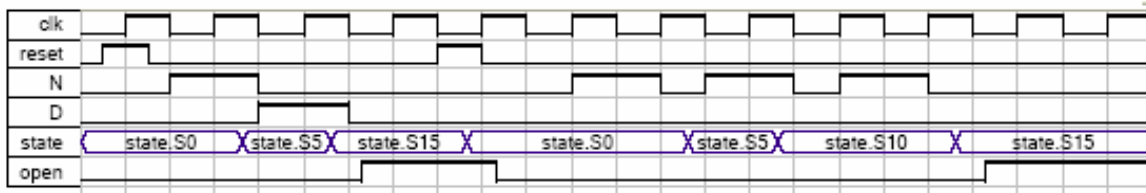
Vending Machine FSM: Minimized Implementation



Vending machine FSM implementation based on D flip-flops (Moore).

Vending Machine: Simulation Results

Moore FSM



Mealy FSM



Summary

- Finite state machines form the basis of many digital systems
- Designs often start from clear specifications
- Design process
 - Develop state diagram and state table
 - Optimize using combinational design techniques
 - » Output logic
 - » Next state logic
 - Mealy or Moore implementations possible
 - » Can model using hardware description languages (HDL)
 - Verilog HDL