Lecture 20

- Logistics
  - HW6 due Wednesday
  - Lab 7 this week (Tuesday exception)
  - Midterm 2 Friday (covers material up to simple FSM (today))
  - Review on Thursday
  - Yoky office hour on Friday moved to Thursday 12-1:20pm online
- Last lecture
  - Counter design
  - Finite state machine – started vending machine example
- Today
  - Continue on the vending machine example
  - Moore/Mealy machines

The "WHY" slide

- Finite State Machine (FSM)
  - This is what we have been waiting for in this class. Using combinational and sequential logics, now you can design a lot of clever digital logic circuits for functional products. We will learn different steps you take to go from word problems to logic circuits.
- Moore/Mealy machines
  - There are two different ways to express the FSMs with respect to the output. Both have different advantages so it is good to know them.

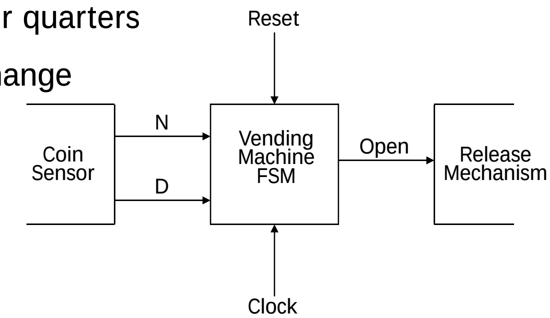
FSM design

- Counter-design procedure
  1. State diagram
  2. State-transition table
  3. Next-state logic minimization
  4. Implement the design

- FSM-design procedure
  1. State diagram
  2. state-transition table
  3. State minimization
  4. State encoding
  5. Next-state logic minimization
  6. Implement the design

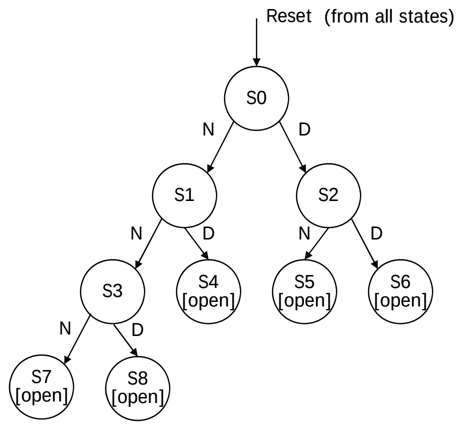
Example: A vending machine

- 15 cents for a cup of coffee
- Doesn't take pennies or quarters
- Doesn't provide any change



- FSM-design procedure
  1. State diagram
  2. state-transition table
  3. State minimization
  4. State encoding
  5. Next-state logic minimization
  6. Implement the design

A vending machine: (conceptual) state diagram



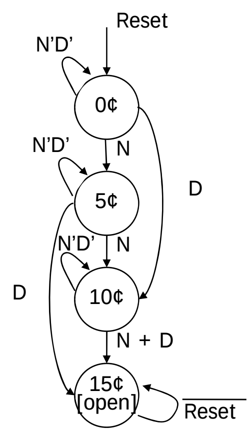
Draw self-loops for N' D' for S0 to S3

Also draw self-loops for 1 for S4 to S8

A vending machine: State transition table

present state	inputs D	N	next state	output open
S0	0	0	S0	0
	0	1	S1	0
	1	0	S2	0
	1	1	X	X
S1	0	0	S1	0
	0	1	S3	0
	1	0	S4	0
	1	1	X	X
S2	0	0	S2	0
	0	1	S5	0
	1	0	S6	0
	1	1	X	X
S3	0	0	S3	0
	0	1	S7	0
	1	0	S8	0
	1	1	X	X
S4	X	X	S4	1
S5	X	X	S5	1
S6	X	X	S6	1
S7	X	X	S7	1
S8	X	X	S8	1

A vending machine: State minimization



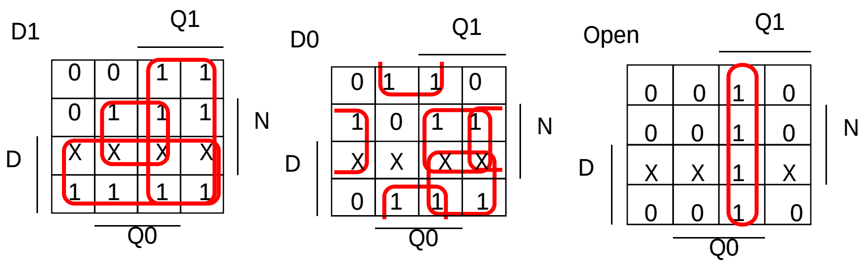
present state	inputs D	inputs N	next state	output open
0¢	0	0	0¢	0
	0	1	5¢	0
	1	0	10¢	0
	1	1	–	–
5¢	0	0	5¢	0
	0	1	10¢	0
	1	0	15¢	0
	1	1	–	–
10¢	0	0	10¢	
	0	1	15¢	0
	1	0	15¢	0
	1	1	–	–
15¢	–	–	15¢	1

symbolic state table

A vending machine: State encoding

present state Q1	present state Q0	inputs D	inputs N	next state D1	next state D0	output open
0	0	0	0	0	0	0
		0	1	0	1	0
		1	0	1	0	0
		1	1	–	–	–
0	1	0	0	0	1	0
		0	1	1	0	0
		1	0	1	1	0
		1	1	–	–	–
1	0	0	0	1	0	0
		0	1	1	1	0
		1	0	1	1	0
		1	1	–	–	–
1	1	–	–	1	1	1

A vending machine: Logic minimization

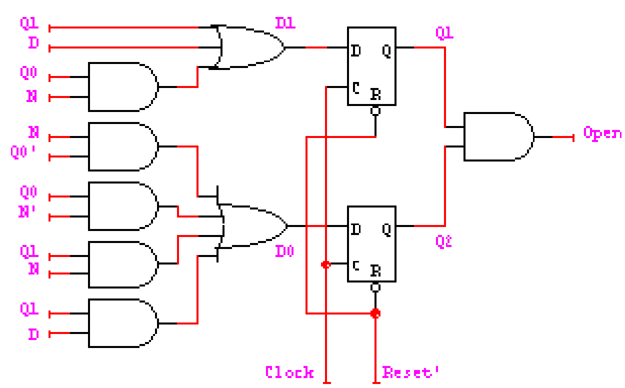


D1 = Q1 + D + Q0 N

D0 = Q0’ N + Q0 N’ + Q1 N + Q1 D

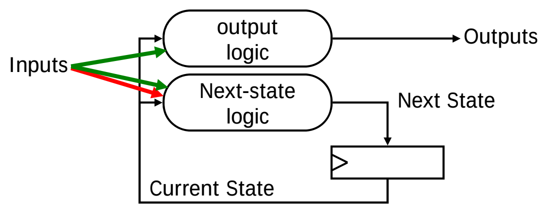
OPEN = Q1 Q0

A vending machine: Implementation

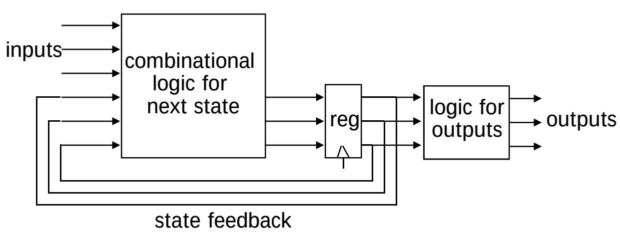


Generalized FSM model: Moore and Mealy

- Combinational logic computes next state and outputs
  - Next state is a function of current state and inputs
  - Outputs are functions of
    - Current state (**Moore** machine)
    - Current state and inputs (**Mealy** machine)



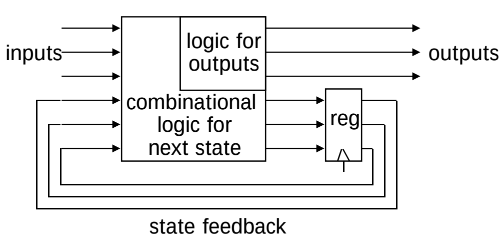
Moore versus Mealy machines



Moore machine

Outputs are a function of current state

Outputs change synchronously with state changes



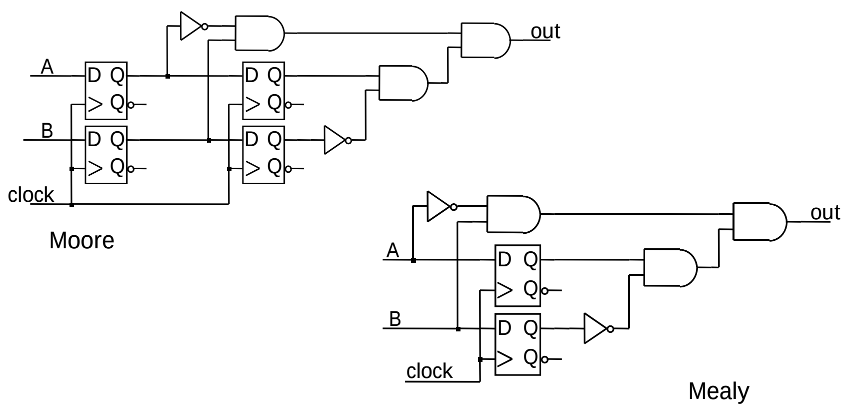
Mealy machine

Outputs depend on state and on inputs

Input changes can cause immediate output changes (**asynchronous**)

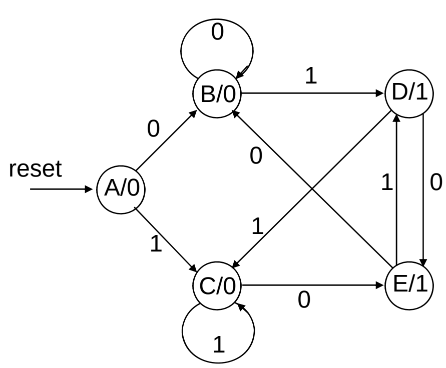
Example 10 -> 01: Moore or Mealy?

- Circuits recognize AB=10 followed by AB=01
  - What kinds of machines are they?



Example 01/10 detector: a Moore machine

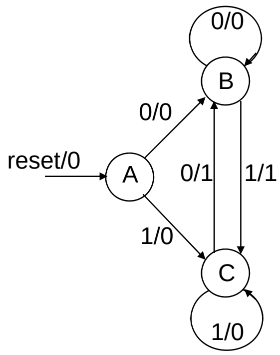
- Output is a function of state only
  - Specify output in the state bubble



reset	input	current state	next state	current output
1	–	–	A	0
0	0	A	B	0
0	1	A	C	0
0	0	B	B	0
0	1	B	D	0
0	0	C	E	0
0	1	C	C	0
0	0	D	E	1
0	1	D	C	1
0	0	E	B	1
0	1	E	D	1

Example 01/10 detector: a Mealy machine

- Output is a function of state and inputs
  - Specify outputs on transition arcs



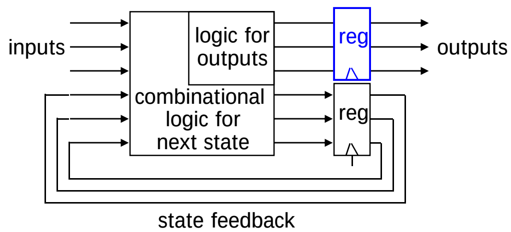
reset	input	current state	next state	current output
1	–	–	A	0
0	0	A	B	0
0	1	A	C	0
0	0	B	B	0
0	1	B	C	1
0	0	C	B	1
0	1	C	C	0

Comparing Moore and Mealy machines

- Moore machines
  - \+ Safer to use because outputs change at clock edge
  - – May take additional logic to decode state into outputs
- Mealy machines
  - \+ Typically have fewer states
  - \+ React faster to inputs — don't wait for clock
  - – Asynchronous outputs can be dangerous
- We often design synchronous Mealy machines
  - Design a Mealy machine
  - Then register the outputs

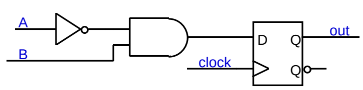
Synchronous (registered) Mealy machine

- Registered state and registered outputs
  - No glitches on outputs
  - No race conditions between communicating machines

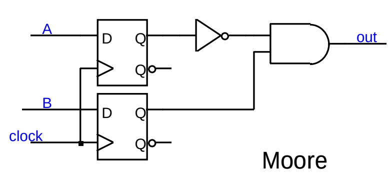


Example 0 -> 1: Moore or Mealy?

- Recognize A,B = 0,1
  - Mealy or Moore?



Registered Mealy
(actually Moore)



Moore