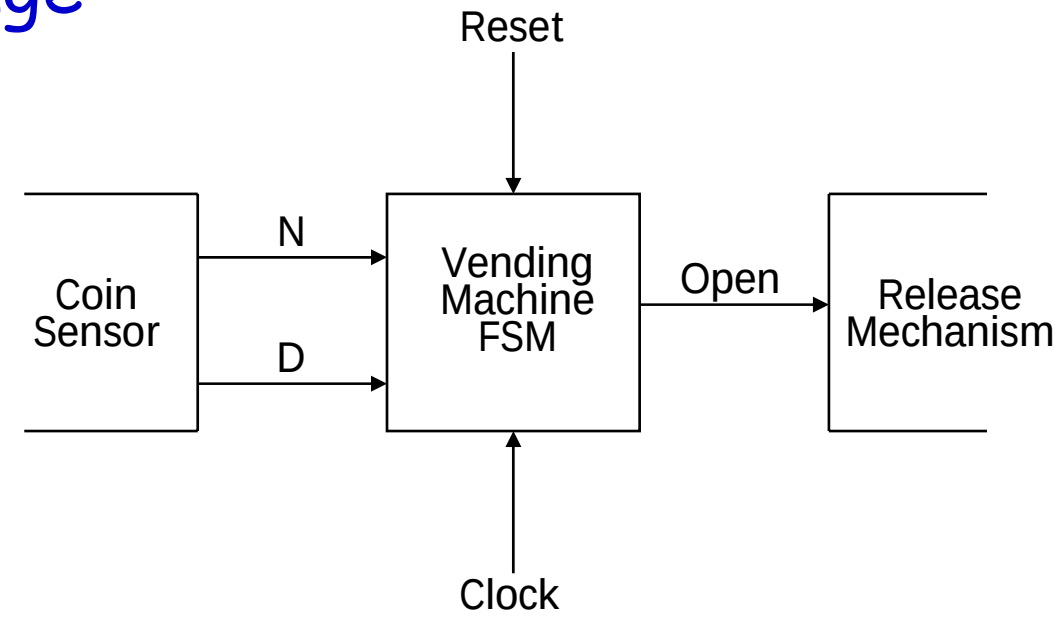
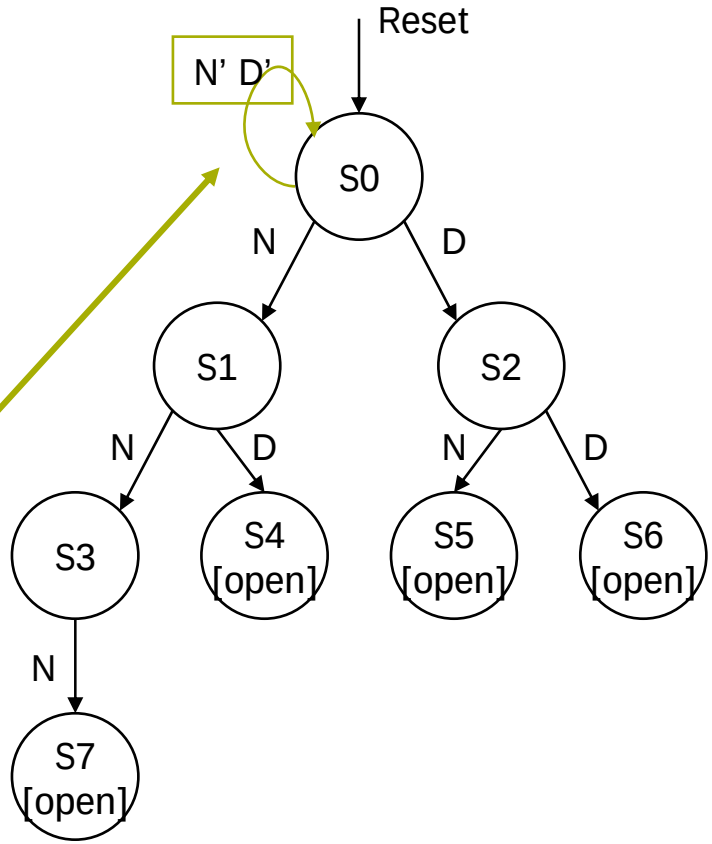
Example: vending machine

- Release item after 15 cents are deposited
- Single coin slot for dimes (10¢), nickels (5¢)
- No change



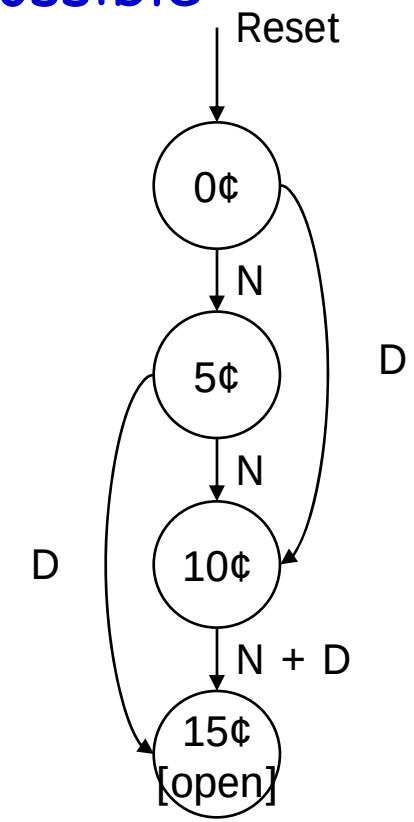
Example: vending machine (cont'd)

- Suitable abstract representation
  - tabulate typical input sequences:
    - 3 nickels (3 X 5¢ =15¢)
    - nickel, dime (5¢ + 10¢ = 15¢)
    - dime, nickel (10¢ + 5¢ = 15¢)
    - two dimes (2 X 10¢ =20¢)
  - draw state diagram:
    - inputs: N, D, reset
    - output: open chute
  - assumptions:
    - assume N and D asserted for one cycle
    - each state has a self loop for N = D = 0 (no coin)



Example: vending machine (cont'd)

- Minimize number of states - reuse states whenever possible



present state	inputs D	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
	0	1	15¢	0
	1	0	15¢	0
	1	1	–	–
15¢	–	–	15¢	1

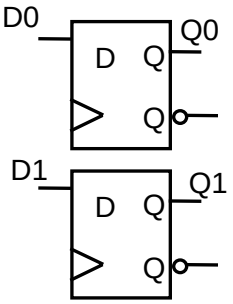
symbolic state table

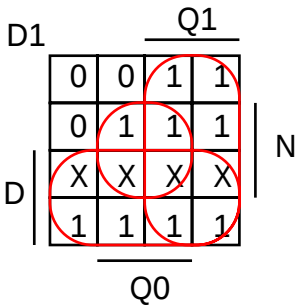
Example: vending machine (cont'd)

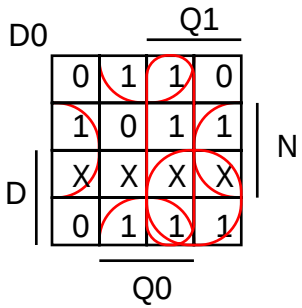
- Uniquely encode states

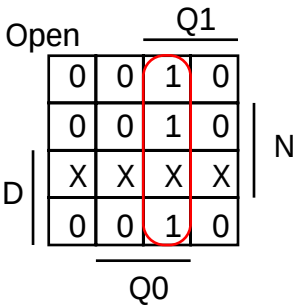
0¢	00
5¢	01
10¢	10
15¢	11

present state Q1	Q0	inputs D	N	next state D1	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



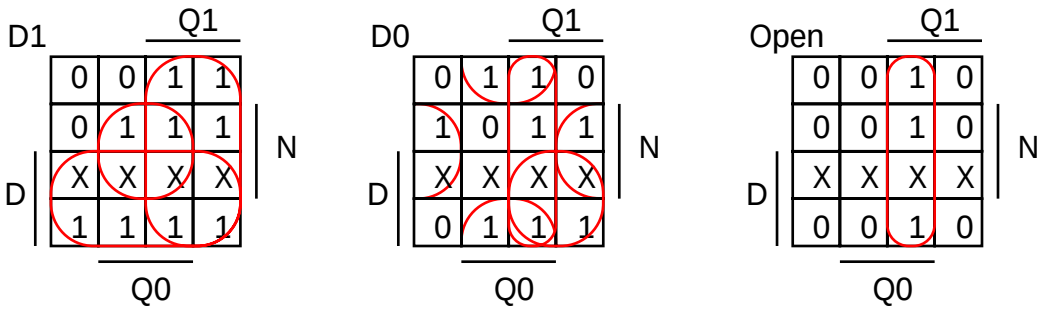


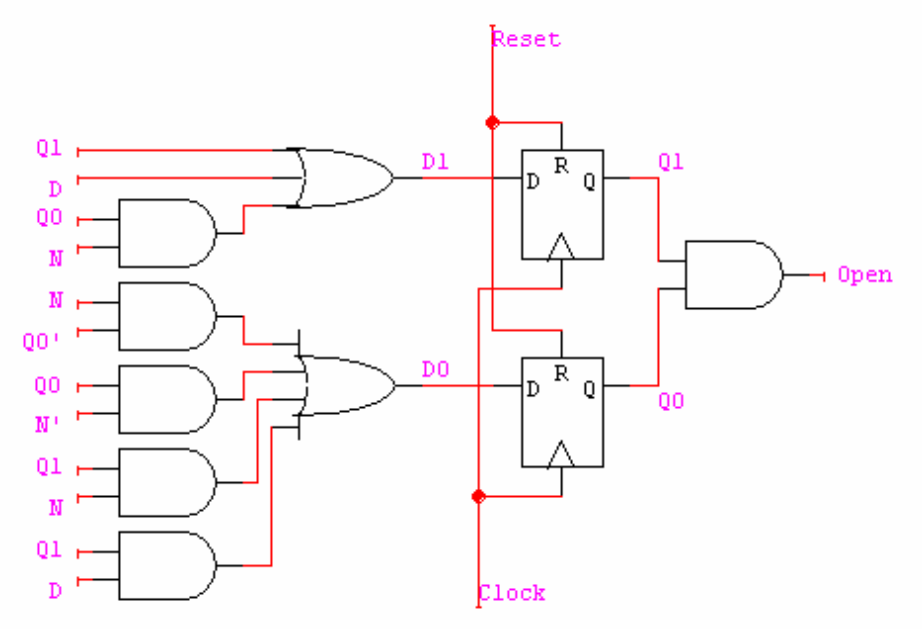




Example: vending machine (cont'd)

- Mapping to logic





D1 = Q1 + D + Q0 N

D0 = Q0' N + Q0 N' + Q1 N + Q1 D

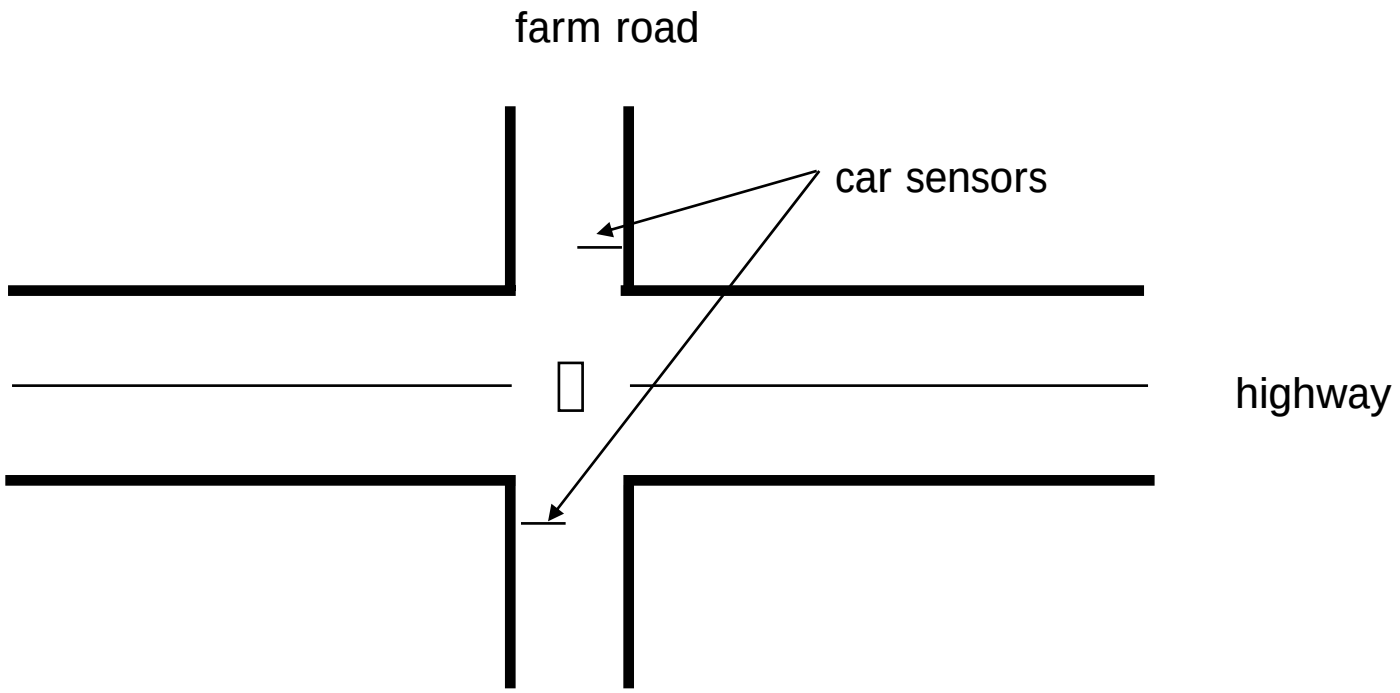
OPEN = Q1 Q0

Example: traffic light controller

- A busy highway is intersected by a little used farmroad
- Detectors C sense the presence of cars waiting on the farmroad
  - with no car on farmroad, light remain green in highway direction
  - if vehicle on farmroad, highway lights go from Green to Yellow to Red, allowing the farmroad lights to become green
  - The farmroad light stays green only as long as a farmroad car is detected but never longer than a set interval
  - when these are met, farm lights transition from Green to Yellow to Red, allowing highway to return to green
  - even if farmroad vehicles are waiting, highway gets at least a set interval as green
- Assume you have an interval timer that generates:
  - a short time pulse (TS) and
  - a long time pulse (TL),
  - in response to a set (ST) signal.
  - TS is to be used for timing yellow lights and TL for green lights

Example: traffic light controller (cont')

- Highway/farm road intersection



Example: traffic light controller (cont')

- Tabulation of inputs and outputs

inputs	description
reset	place FSM in initial state
C	detect vehicle on the farm road
TS	short time interval expired
TL	long time interval expired

outputs	description
H(Green,Yellow, Red)	assert green/yellow/red highway lights
F(Green, Yellow,Red)	assert green/yellow/red Farmroad lights
ST	start timing a short or long interval

- Tabulation of unique states – some light configurations imply others

state	description
S0	highway green (farm road red)
S1	highway yellow (farm road red)
S2	farm road green (highway red)
S3	farm road yellow (highway red)

Example: traffic light controller (cont')

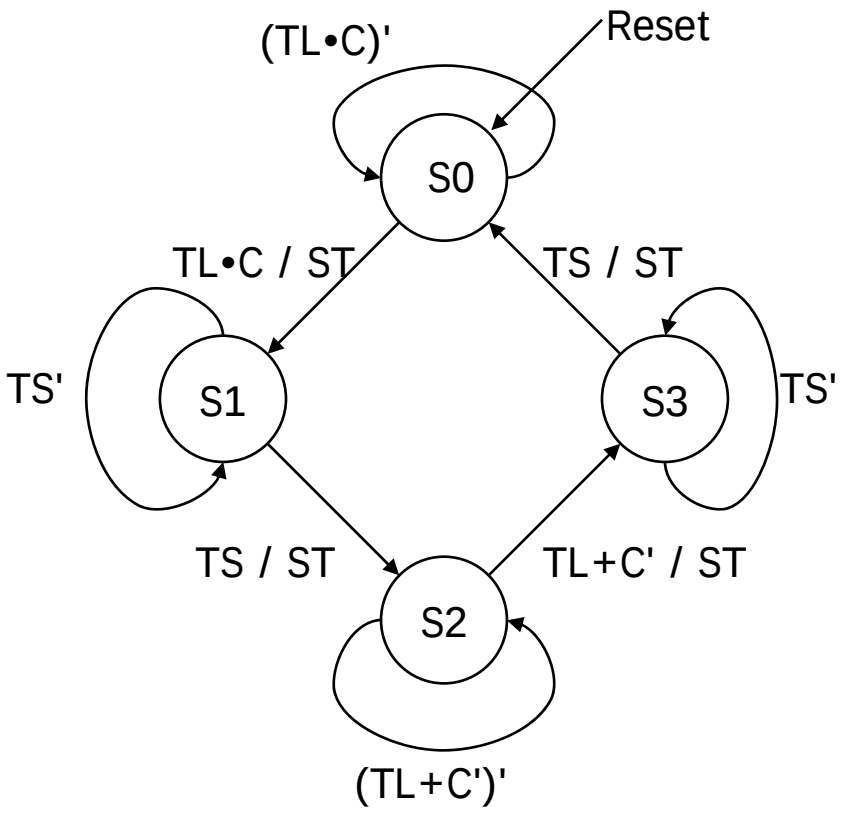
- State diagram

S0: HG

S1: HY

S2: FG

S3: FY



Example: traffic light controller (cont')

- Generate state table with symbolic states
- Consider state assignments

output encoding – similar problem to state assignment
(Green = 00, Yellow = 01, Red = 10)

Inputs C	TL	TS	Present State	Next State	Outputs ST	H	F
0	–	–	HG	HG	0	Green	Red
–	0	–	HG	HG	0	Green	Red
1	1	–	HG	HY	1	Green	Red
–	–	0	HY	HY	0	Yellow	Red
–	–	1	HY	FG	1	Yellow	Red
1	0	–	FG	FG	0	Red	Green
0	–	–	FG	FY	1	Red	Green
–	1	–	FG	FY	1	Red	Green
–	–	0	FY	FY	0	Red	Yellow
–	–	1	FY	HG	1	Red	Yellow

SA1:	HG = 00	HY = 01	FG = 11	FY = 10
SA2:	HG = 00	HY = 10	FG = 01	FY = 11

Example: traffic light controller (cont')

SA1: HG = 00, HY = 01, FG = 11, FY = 10 (Green = 00, Yellow = 01, Red = 10)

2^5=32 states

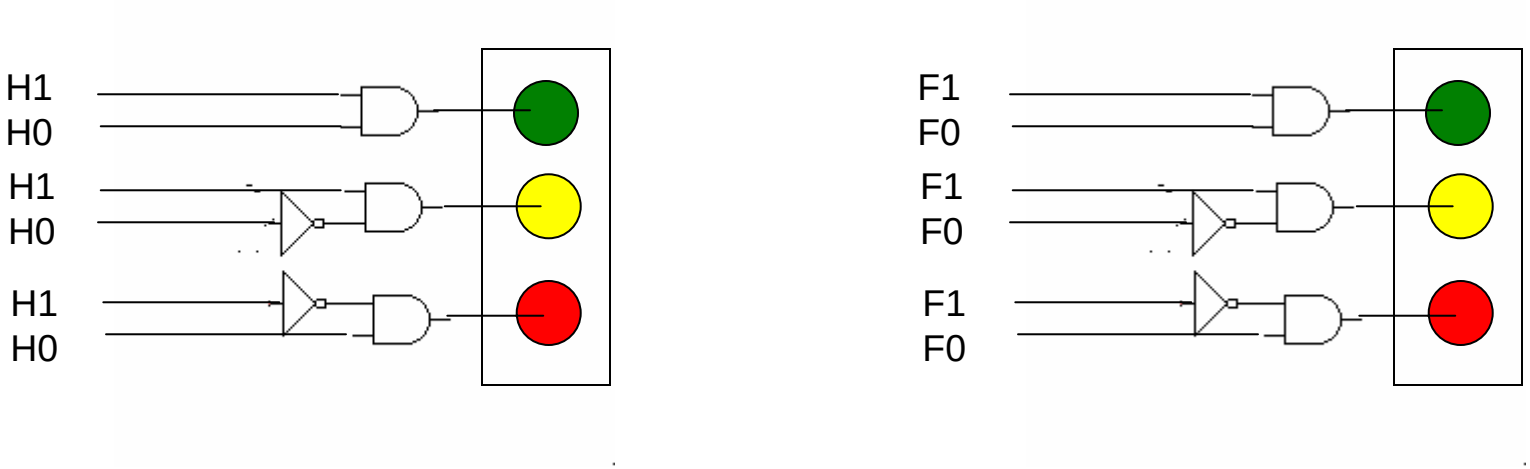
	Inputs			Present State	Next State	Outputs		
	C	TL	TS			ST	H	F
4	0	–	–	HG	HG	0	Green	Red
4	–	0	–	HG	HG	0	Green	Red
2	1	1	–	HG	HY	1	Green	Red
4	–	–	0	HY	HY	0	Yellow	Red
4	–	–	1	HY	FG	1	Yellow	Red
2	1	0	–	FG	FG	0	Red	Green
4	0	–	–	FG	FY	1	Red	Green
4	–	1	–	FG	FY	1	Red	Green
4	–	–	0	FY	FY	0	Red	Yellow
4	–	–	1	FY	HG	1	Red	Yellow

Inputs			Present State		Next State		Outputs				
C	TL	TS	PQ1	PQ0	NQ1	NQ0	ST	H1	H0	F1	F0
0	0	0	0	0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	0	0	1	0
0	1	1	0	0	0	0	0	0	0	1	0

Example: Output to control the lights

Green = 00, Yellow = 01, Red = 10

H1H0 and F1F0
Use low to turn on the lights



ST' —●

Indicating the time duration has expired
Light is changing soon

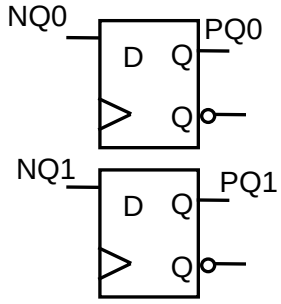
Logic for different state assignments

- SA1: HG = 00 HY = 01 FG = 11 FY = 10

Q1
Q0

NQ1 = C·TL'·PQ1·PQ0 + TS·PQ1'·PQ0 + TS·PQ1·PQ0' + C'·PQ1·PQ0 + TL·PQ1·PQ0
NQ0 = C·TL·PQ1'·PQ0' + C·TL'·PQ1·PQ0 + PQ1'·PQ0

ST = C·TL·PQ1'·PQ0' + TS·PQ1'·PQ0 + TS·PQ1·PQ0' + C'·PQ1·PQ0 + TL·PQ1·PQ0
H1 = PQ1 H0 = PQ1'·PQ0
F1 = PQ1' F0 = PQ1·PQ0'



- SA2: HG = 00 HY = 10 FG = 01 FY = 11

NQ1 = C·TL·PQ1' + TS'·PQ1 + C'·PQ1'·PQ0
NQ0 = TS·PQ1·PQ0' + PQ1'·PQ0 + TS'·PQ1·PQ0

ST = C·TL·PQ1' + C'·PQ1'·PQ0 + TS·PQ1
H1 = PQ0 H0 = PQ1·PQ0'
F1 = PQ0' F0 = PQ1·PQ0

Sequential logic implementation summary

- Models for representing sequential circuits
  - abstraction of sequential elements
  - finite state machines and their state diagrams
  - inputs/outputs
  - Mealy, Moore, and synchronous Mealy machines
- Finite state machine design procedure
  - deriving state diagram
  - deriving state transition table
  - determining next state and output functions
  - implementing combinational logic
- Implementation of sequential logic
  - state minimization
  - state assignment
  - support in programmable logic devices