SEE 3243/4243
Advanced FSM Implementation

Week 12

- *Using PLA, ROM & counters for implementation*

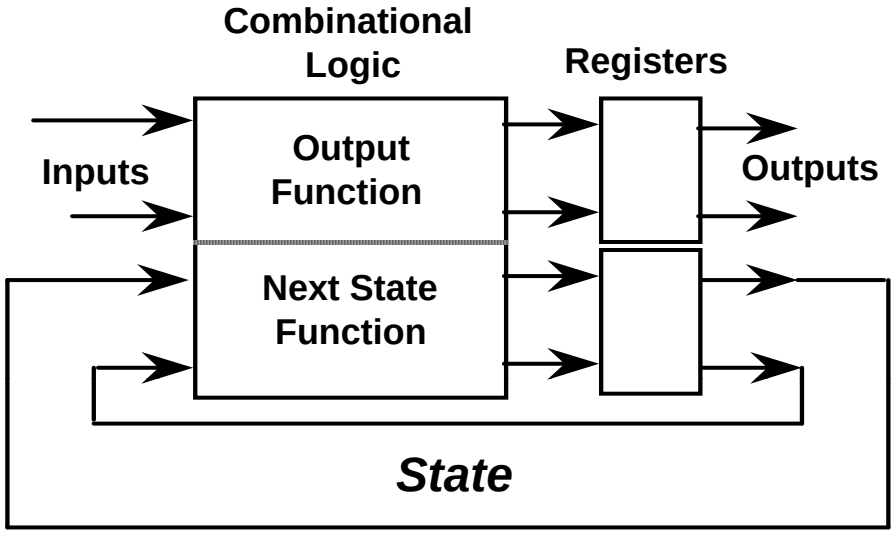
State machine design

- Construct state/output table
- Minimize number of states (optional)
- Assign state variables
- Choose flip-flop type (d or j-k)
- Construct excitation table
- Derive excitation equations
- Derive output equations
- Draw logic diagram

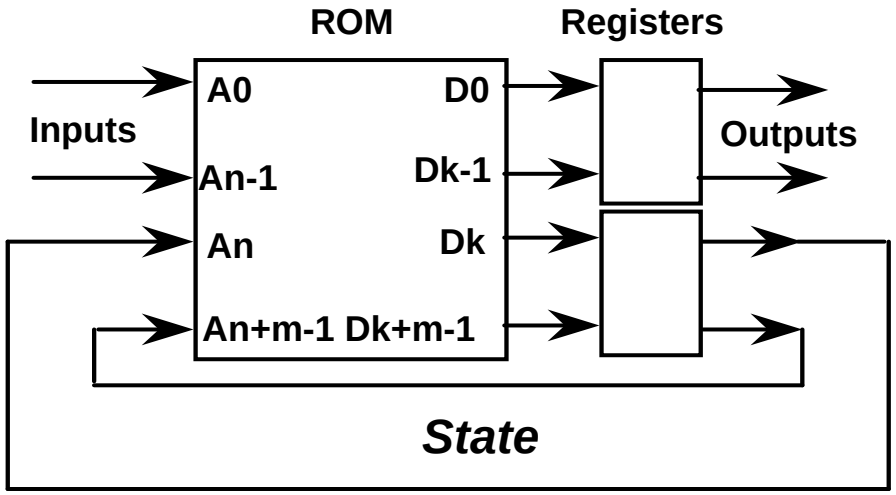
Implementation Strategies

- Discrete logic
- Design with structured Logic
  - Counters
  - Shift Register
  - ROMs
- Programmable logic
  - PALs
  - FGPAs: Altera, Actel, Xilinx

FSM Design with Structured Logic



Block Diagram for Synchronous Mealy Machine

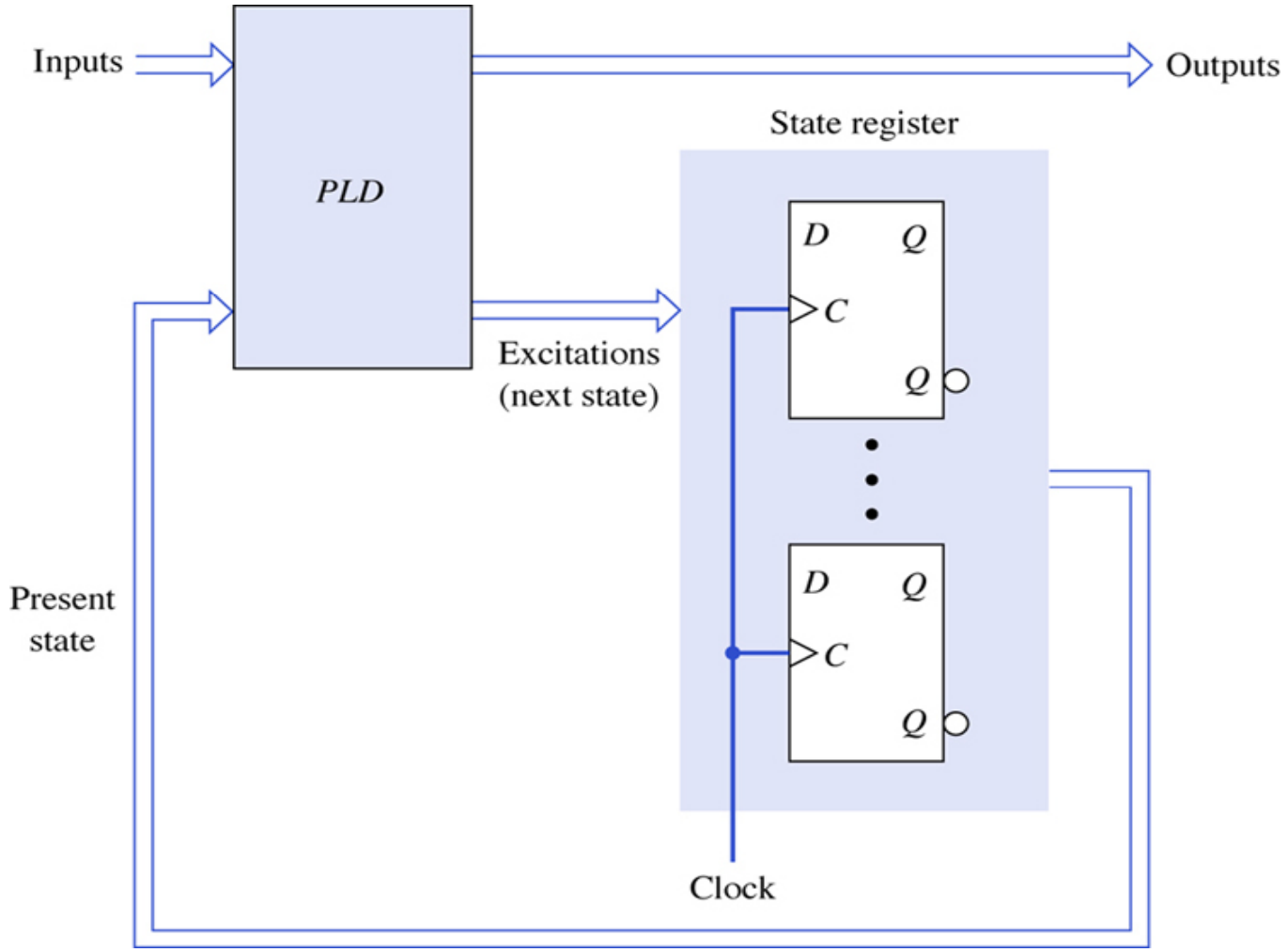


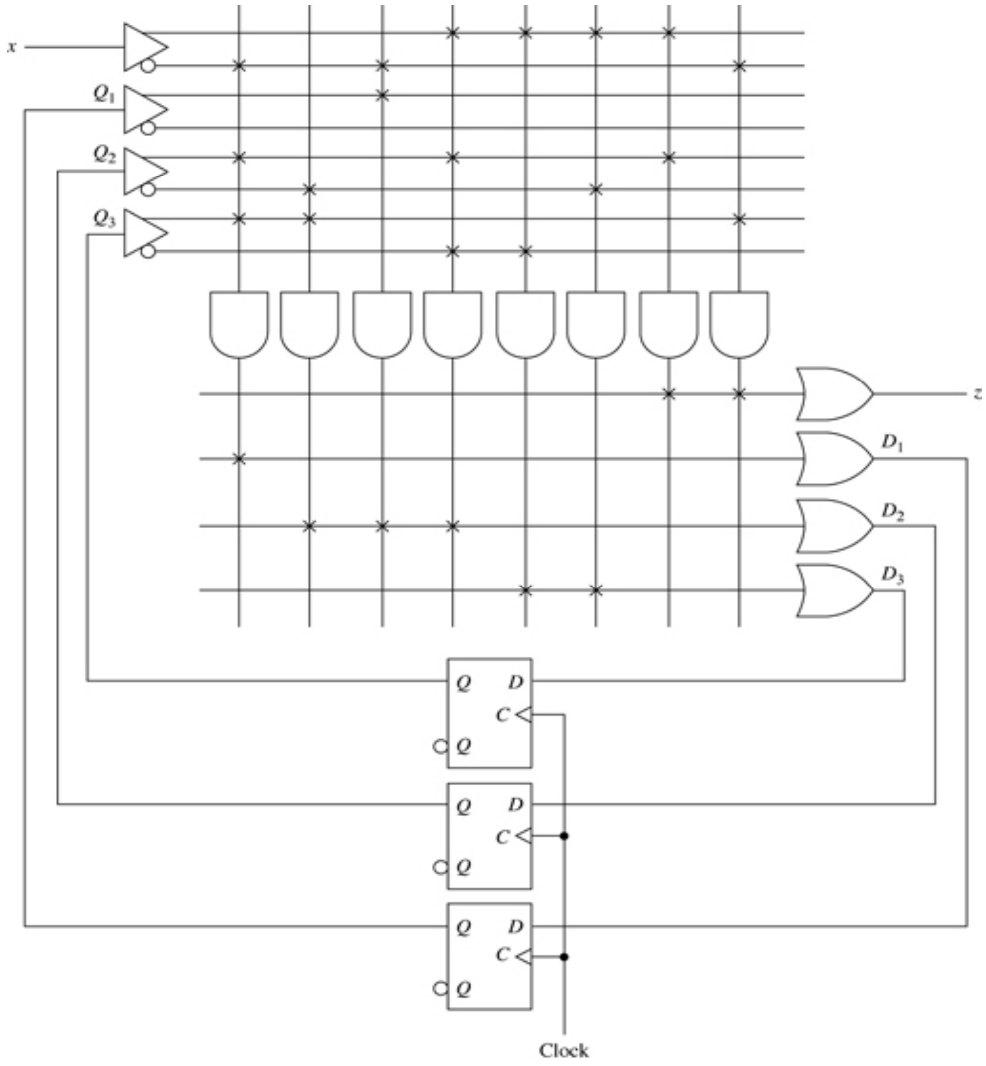
ROM-based Realization

- Inputs & Current State form the address
- ROM data bits form the Outputs & Next State

General structure of a clocked sequential network realization using a PLD and clocked *D* flip-flops.

Figure 7.31





A clocked synchronous sequential network realization using a PLA and clocked *D* flip-flops.

BCD to Excess 3 Serial Converter

ROM-based Design

Example: BCD to Excess 3 Serial Converter

Conversion Process

Bits are presented in bit serial fashion starting with the least significant bit

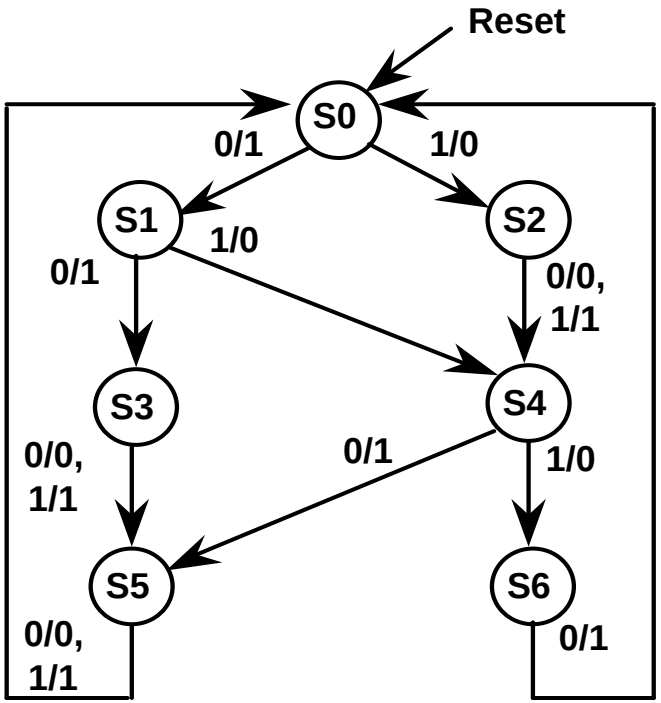
Single input X, single output Z

BCD	Excess 3 Code
0000	0011
0001	0100
0010	0101
0011	0110
0100	0111
0101	1000
0110	1001
0111	1010
1000	1011
1001	1100

BCD to Excess 3 Serial Converter

	Next State		Output	
Present State	X=0	X=1	X=0	X=1
S0	S1	S2	1	0
S1	S3	S4	1	0
S2	S4	S4	0	1
S3	S5	S5	0	1
S4	S5	S6	1	0
S5	S0	S0	0	1
S6	S0	--	1	--

State Transition Table



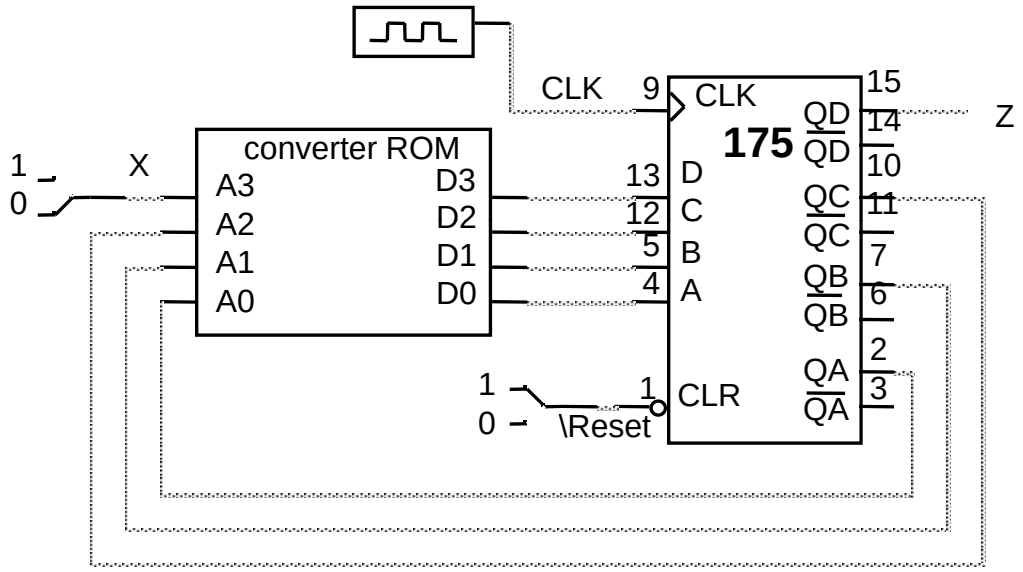
Derived State Diagram

BCD to Excess 3 Serial Converter

ROM-based Implementation

ROM Address				ROM Outputs			
A3	A2	A1	A0	D3	D2	D1	D0
0	0	0	0	1	0	0	1
0	0	0	1	1	0	1	1
0	0	1	0	0	1	0	0
0	0	1	1	0	1	0	1
0	1	0	0	1	1	0	1
0	1	0	1	0	0	0	0
0	1	1	0	1	0	0	0
0	1	1	1	X	X	X	X
1	0	0	0	0	0	1	0
1	0	0	1	0	1	0	0
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	1	0	0	0	1	1	0
1	1	0	1	1	0	0	0
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X

Truth Table/ROM I/Os



Circuit Level Realization

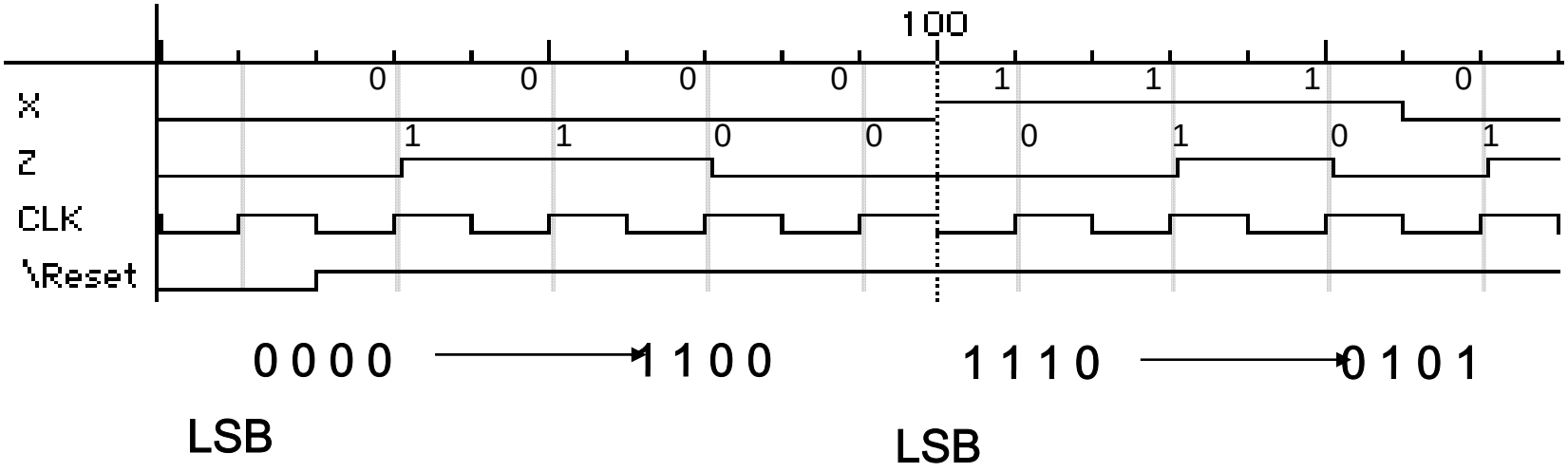
74175 = 4 x positive edge triggered D FFs

In ROM-based designs, no need to consider state assignment

BCD to Excess 3 Serial Converter

LSB

Timing Behavior for input strings 0 0 0 0 (0) and 1 1 1 0 (7)



Implementation Strategies

PLA-based Design

BCD to Excess 3 Converter

S0 = 000
S1 = 001
S2 = 011
S3 = 110
S4 = 100
S5 = 111
S6 = 101

Implementation Strategies

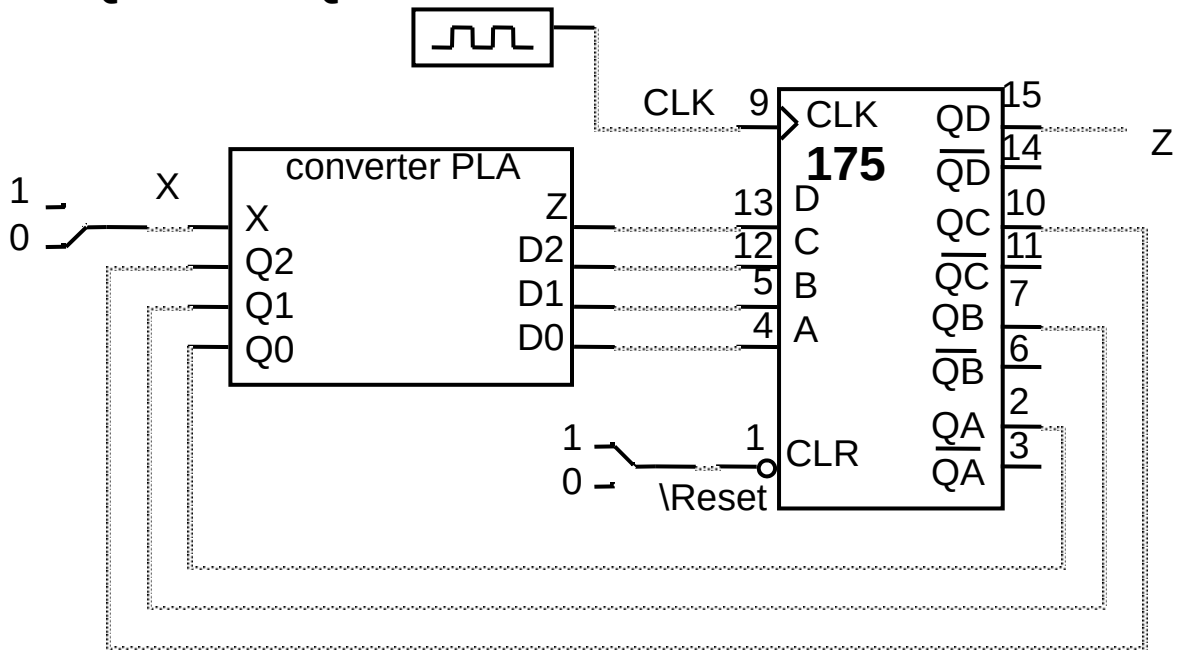
BCD to Excess 3 Converter

$$D2 = \overline{Q2} \cdot Q0 + Q2 \cdot \overline{Q0}$$

$$D1 = \overline{X} \cdot \overline{Q2} \cdot \overline{Q1} \cdot Q0 + X \cdot \overline{Q2} \cdot \overline{Q0} + \overline{X} \cdot Q2 \cdot \overline{Q0} + Q1 \cdot \overline{Q0}$$

$$D0 = \overline{Q0}$$

$$Z = X \cdot Q1 + \overline{X} \cdot \overline{Q1}$$



Implementation Strategies

BCD to Excess 3 Serial Converter

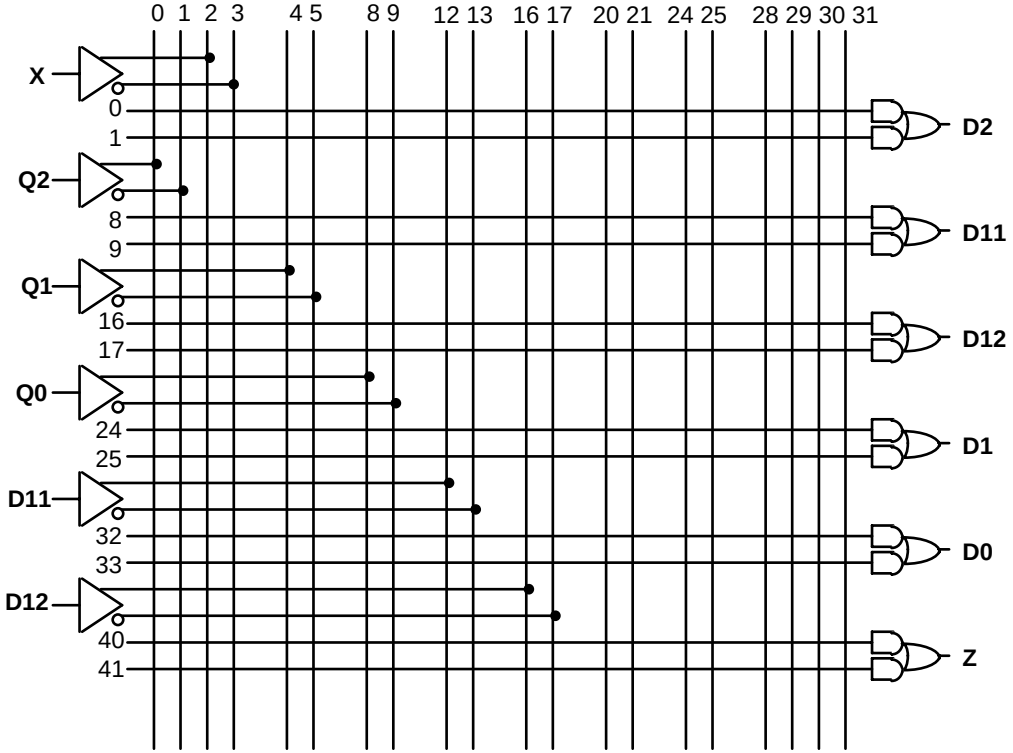
10H8 PAL: 10 inputs, 8 outputs, 2 product terms per OR gate

$$D1 = D11 + D12$$

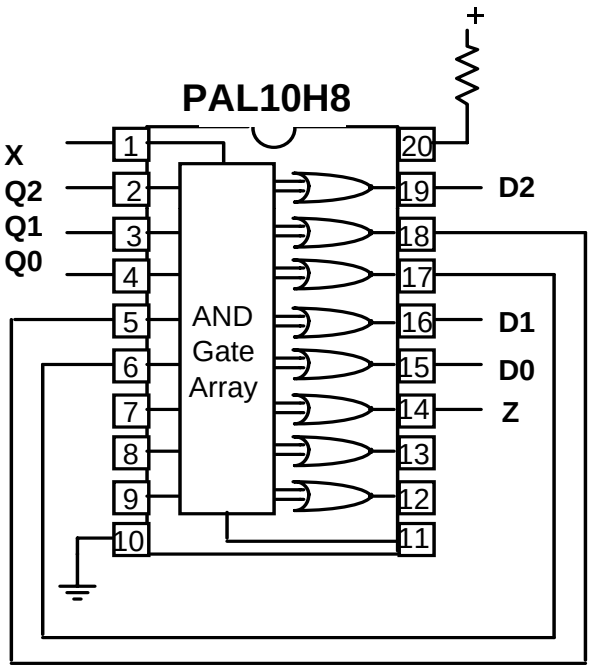
$$D11 = \overline{X} \cdot \overline{Q2} \cdot \overline{Q1} \cdot Q0 \; + \; X \cdot \overline{Q2} \cdot \overline{Q0}$$

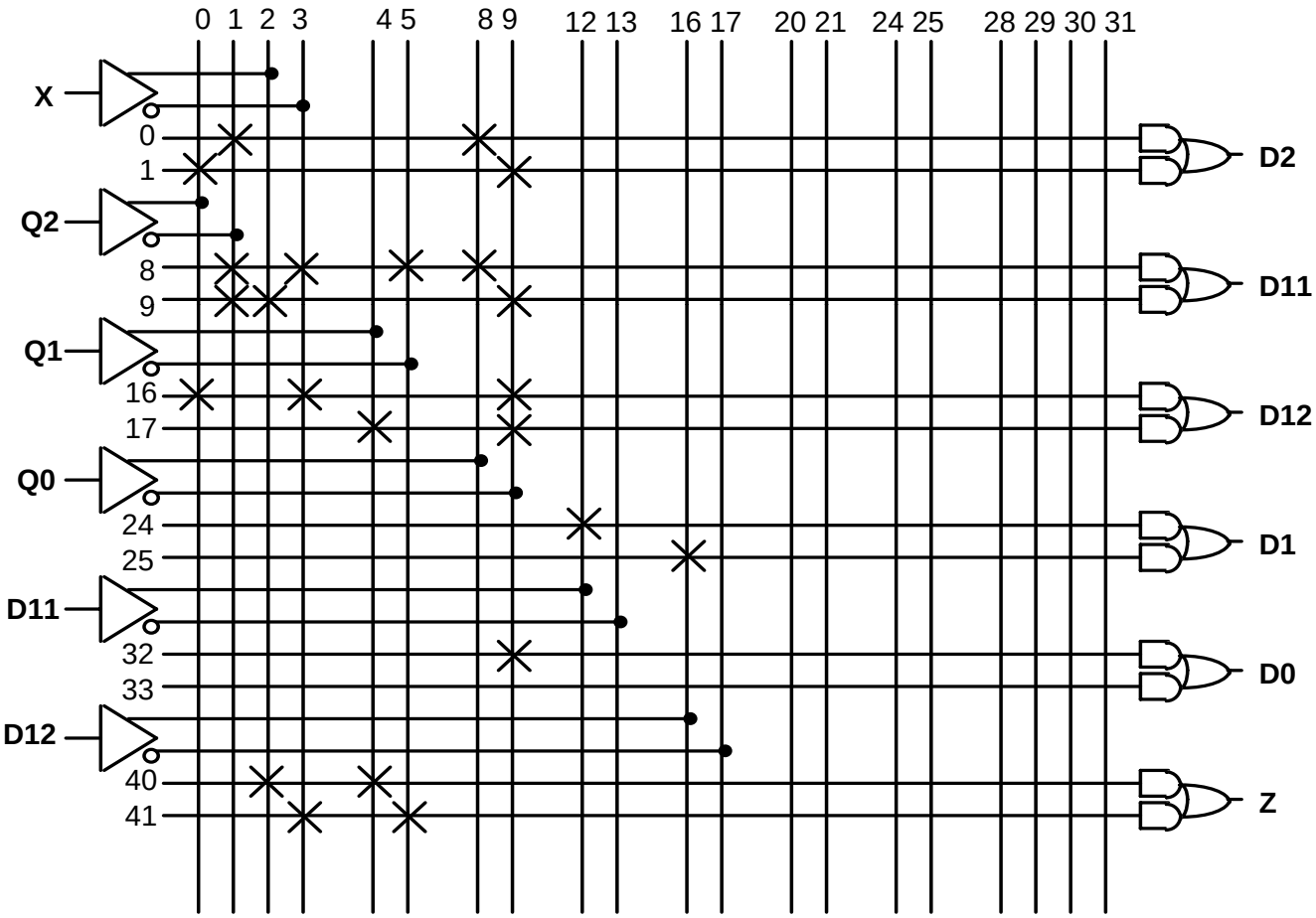
$$D12 = \overline{X} \cdot Q2 \cdot \overline{Q0} \; + \; Q1 \cdot \overline{Q0}$$

0. $\overline{Q2} \cdot \overline{Q0}$
1. $\overline{Q2} \cdot Q0$
8. $X \cdot \overline{Q2} \cdot \overline{Q1} \cdot Q0$
9. $X \cdot \overline{Q2} \cdot Q0$
16. $\overline{X} \cdot Q2 \cdot \overline{Q0}$
17. $Q1 \cdot \overline{Q0}$
24. D11
25. D12
32. $\overline{Q0}$
33. not used
40. $X \cdot Q1$
41. $\overline{X} \cdot Q1$



BCD to Excess 3 Serial Converter





FSM Design with More Sophisticated PLDs

- Programmable Logic Devices = PLD
  - PALs, PLAs = 10 - 100 Gate Equivalents
- Field Programmable Gate Arrays = FPGAs
  - Altera MAX, FLEX, Stratix Families
  - Actel Programmable Gate Array
  - Xilinx Logical Cell Array
  - 100 – 1,000,000(s) of Gate Equivalents!

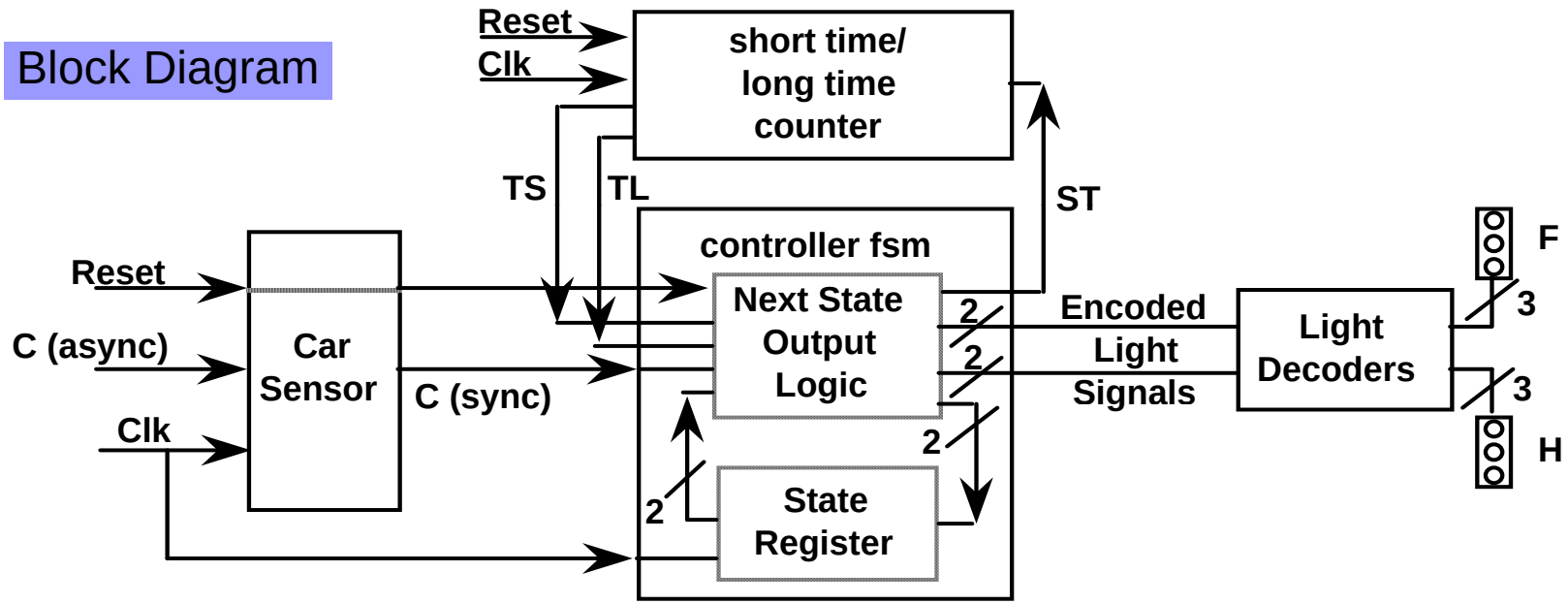
Case Study: Traffic Light Controller

- Decomposition into primitive subsystems
- Controller FSM
  - next state/output functions
  - state register

- Short time/long time interval counter
- Car Sensor
- Output Decoders and Traffic Lights

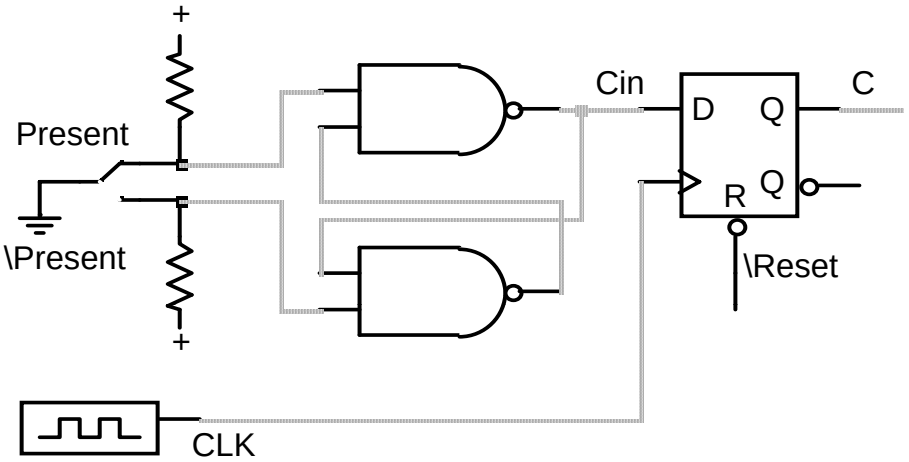
Case Study: Traffic Light Controller

Block Diagram



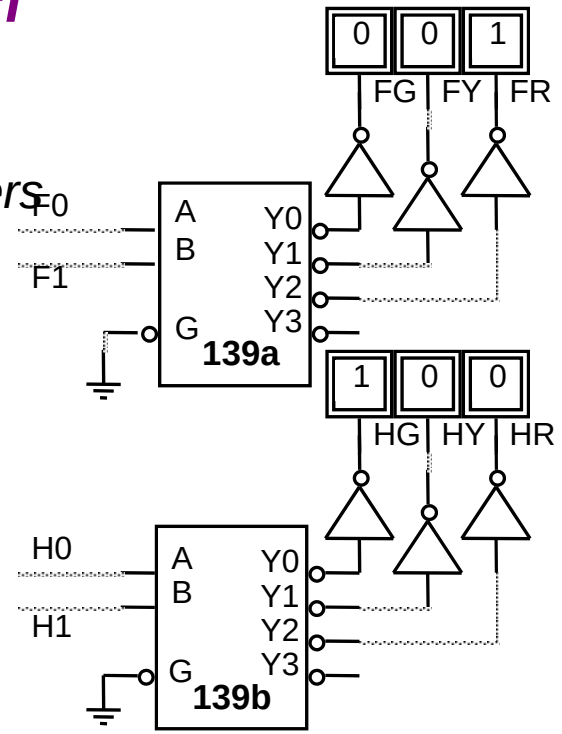
Case Study: Traffic Light Controller

Subsystem Logic

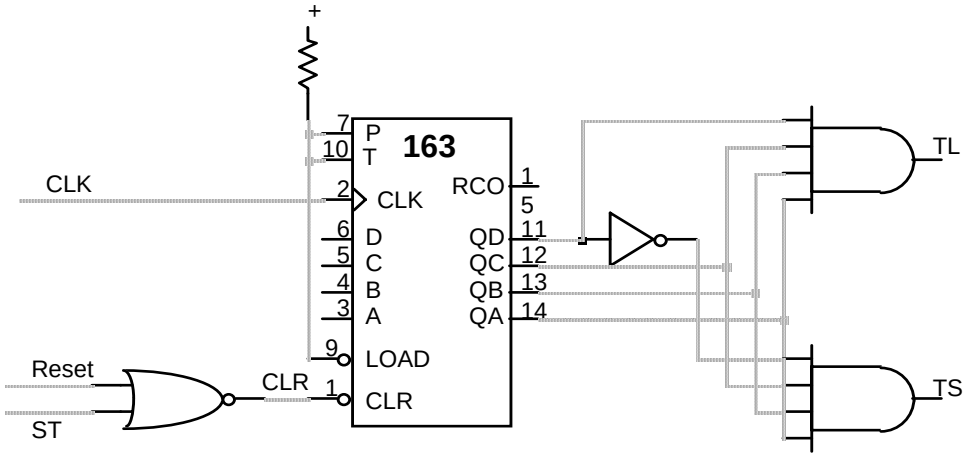


Car Detector

Light Decoders



Interval Timer



Design Case Study: Traffic Light Controller

Next State Logic

State Assignment: HG = 00, HY = 10, FG = 01, FY = 11

P1 = C TL $\overline{Q1}$ + $\overline{TS}$ Q1 $\overline{Q0}$ + $\overline{C}$ $\overline{Q1}$ Q0 + $\overline{TS}$ Q1 Q0

P0 = TS Q1 $\overline{Q0}$ + $\overline{Q1}$ Q0 + $\overline{TS}$ Q1 Q0

ST = C TL $\overline{Q1}$ + $\overline{C}$ $\overline{Q1}$ Q0 + TS Q1 $\overline{Q0}$ + TS Q1 Q0

HL[1] = TS Q1 Q0 + $\overline{Q1}$ Q0 + $\overline{TS}$ Q1 Q0

HL[0] = $\overline{TS}$ Q1 $\overline{Q0}$ + TS Q1 $\overline{Q0}$

FL[1] = $\overline{Q0}$

FL[0] = TS Q1 Q0 + $\overline{TS}$ Q1 Q0

PAL/PLA Implementation:

- 5 inputs, 7 outputs, 8 product terms
- PAL 22V10 -- 11 inputs, 10 prog. IOs, 8 to 14 prod terms per OR

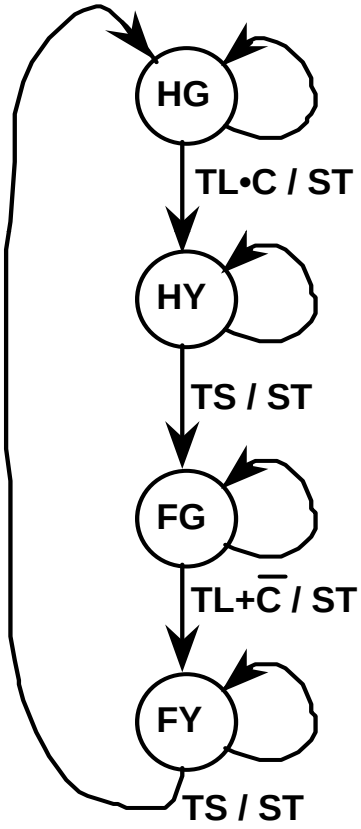
ROM Implementation:

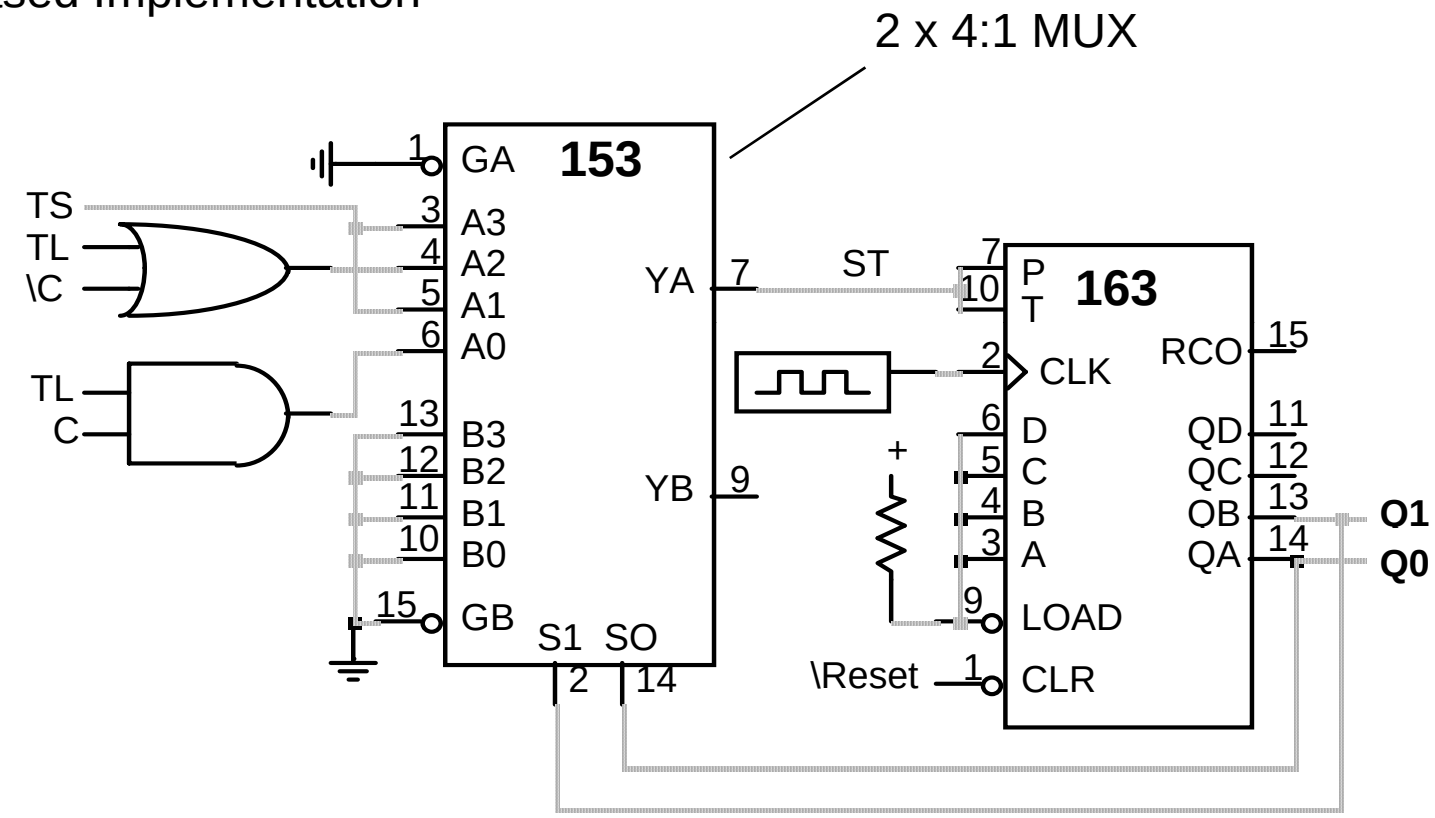
- 32 word by 8-bit ROM (256 bits)
- Reset may double ROM size

Design Case Study: Traffic Light Controller

Next State Logic

Counter-based Implementation





TTL Implementation with MUX and Counter

Can we reduce package count by using an 8:1 MUX?

ST = Count

Summary of FSM Implementation

- Implementation Issues
  - Choice of Flipflops
  - Structured Logic Methods
- ROM based
- PLA/PAL based
- Jump Counter Methods
- Sophisticated PLDs: Altera, Actel, Xilinx