Digital Electronics and Microprocessor Systems (ELEC211)

Dave McIntosh and Valerio Selis

dmc@liv.ac.uk

v.selis@liv.ac.uk

Digital 10: ASM: Encoded State and One Hot State

UNIVERSITY OF LIVERPOOL

Outline

- ASM charts v State graphs
- State transition tables
- ASM design / implementation
- Encoded state representation
- One Hot state representation

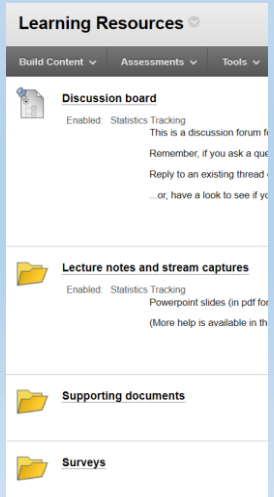
Previous material

Sequential circuits ✓
State tables ✓
Next-state maps ✓
ASM charts ✓
State graphs ✓

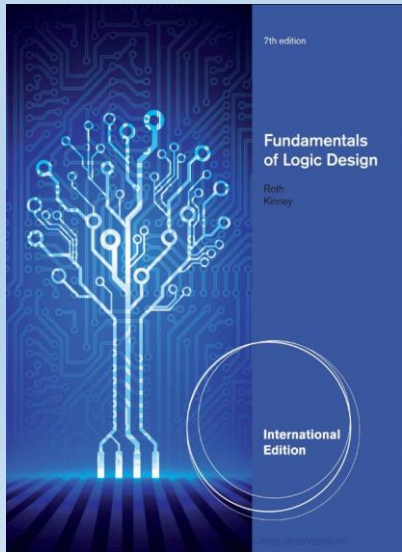
Use VITAL!:

- Stream lectures
- Handouts
- Notes and Q&A each week
- Discussion Board
- Exam resources

www.liv.ac.uk/vital



Course textbook – 7th ed. available as e-book!



UNIVERSITY OF LIVERPOOL

Logic gates & combinational logic

Latches. Concepts of clock, timing, memory. Sequential logic. Timing diagrams, glitches

Combining logic gates to implement multiplexers, decoders; ROM, PLA, PAL

Flip-flops → registers for data storage/transfer

Flip-flops → sequential circuits with output & feedback

Extending latches into flip-flops.

CPLDs / FPGAs from flip-flops, MUX, gates, LUTs. The concept of 'state'

State Machines with output and next-state logic. **Design of state machines**

- State Graphs
- ASM charts
- State (Transition) Tables
- Next-state maps

- State Graphs
- ASM charts
- State (Transition) Tables
- Next-state maps

Notes:

- [Algorithmic] State Machine - really another name for a sequential circuit
- ASM chart ≡ 'SM chart'
- State graph ≡ State diagram
- State Table, Transition table/State transition table

Example of a state table

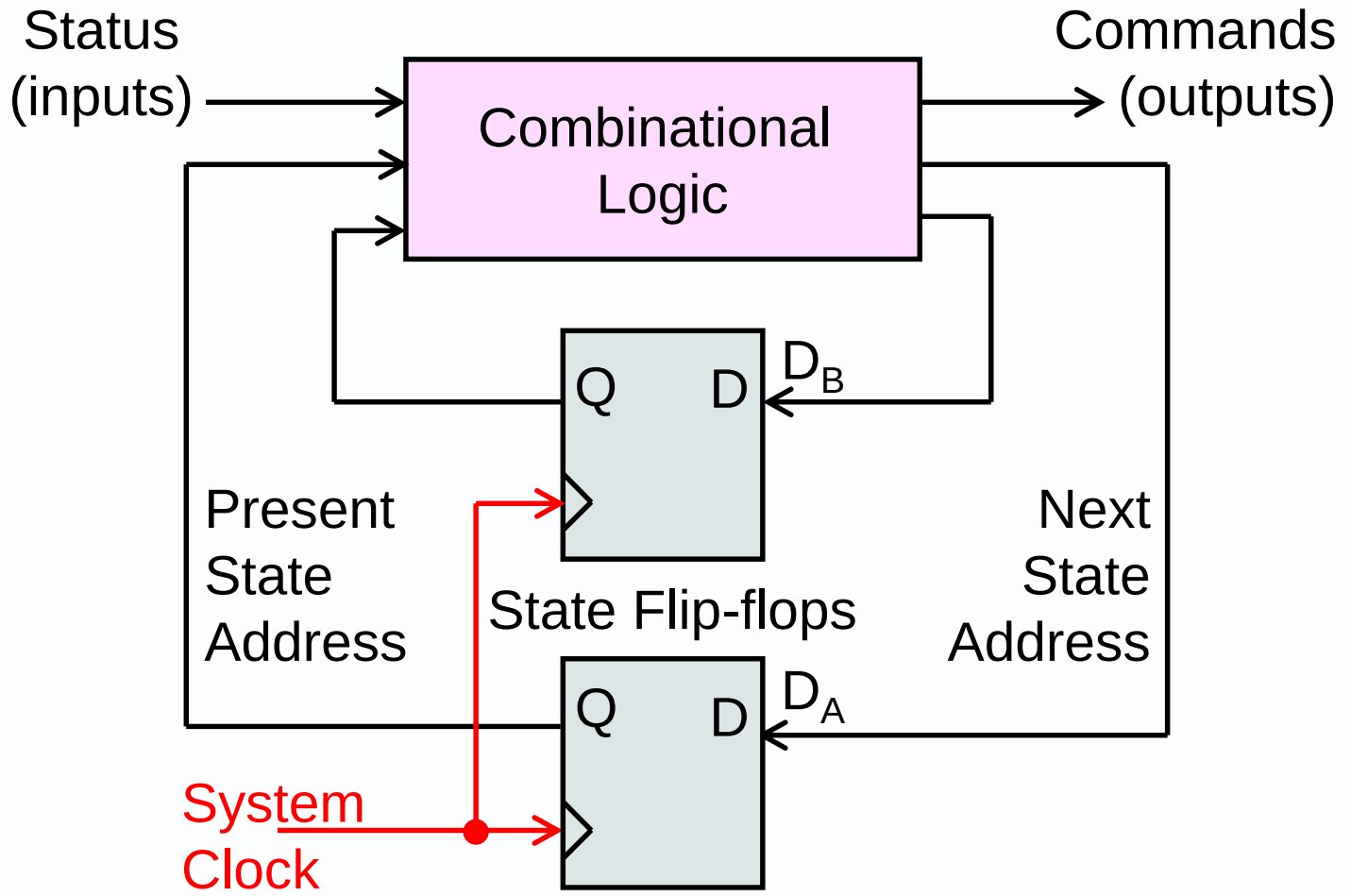
Present State	Input	Next State	Outputs	
	Z		CMD1	CMD2
S_0	0	S_1	1	1
S_0	1	S_2	1	0
S_1	-	S_0	0	0
S_2	-	S_0	0	1

[State] **transition** table

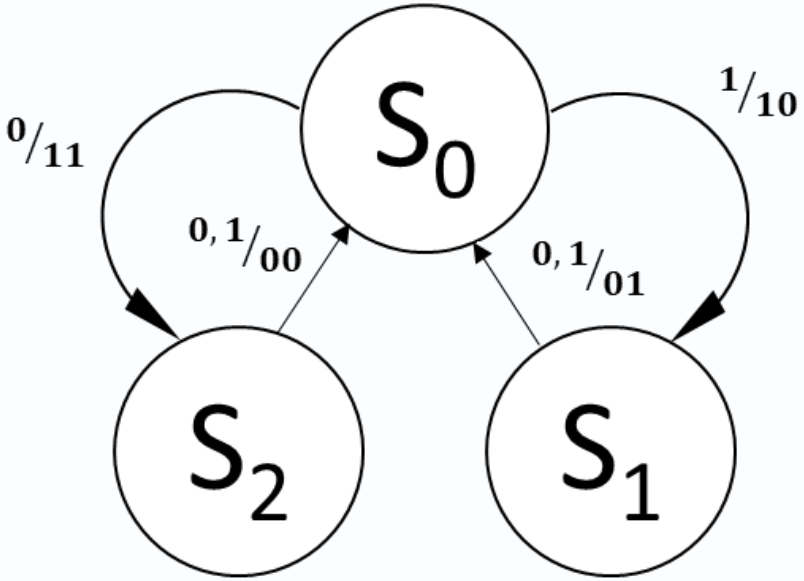
Present State	Input	Next State	Outputs	
BA	Z	D_BD_A	CMD1	CMD2
00	0	10	1	1
00	1	11	1	0
01	-	-	-	-
10	-	00	0	0
11	-	00	0	1

Specifies flip-flop states

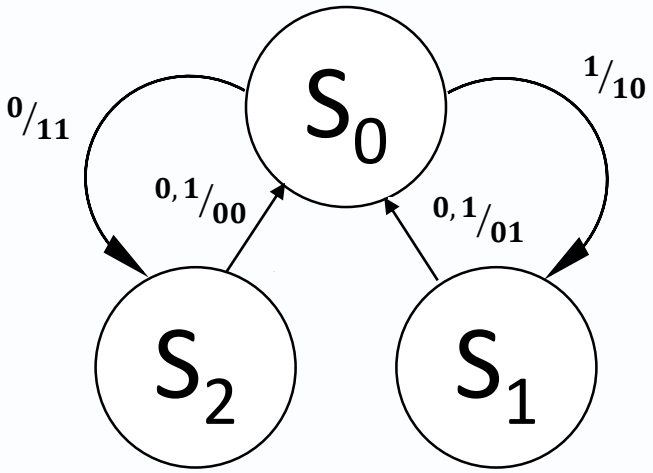
Process Model



Design Implementation



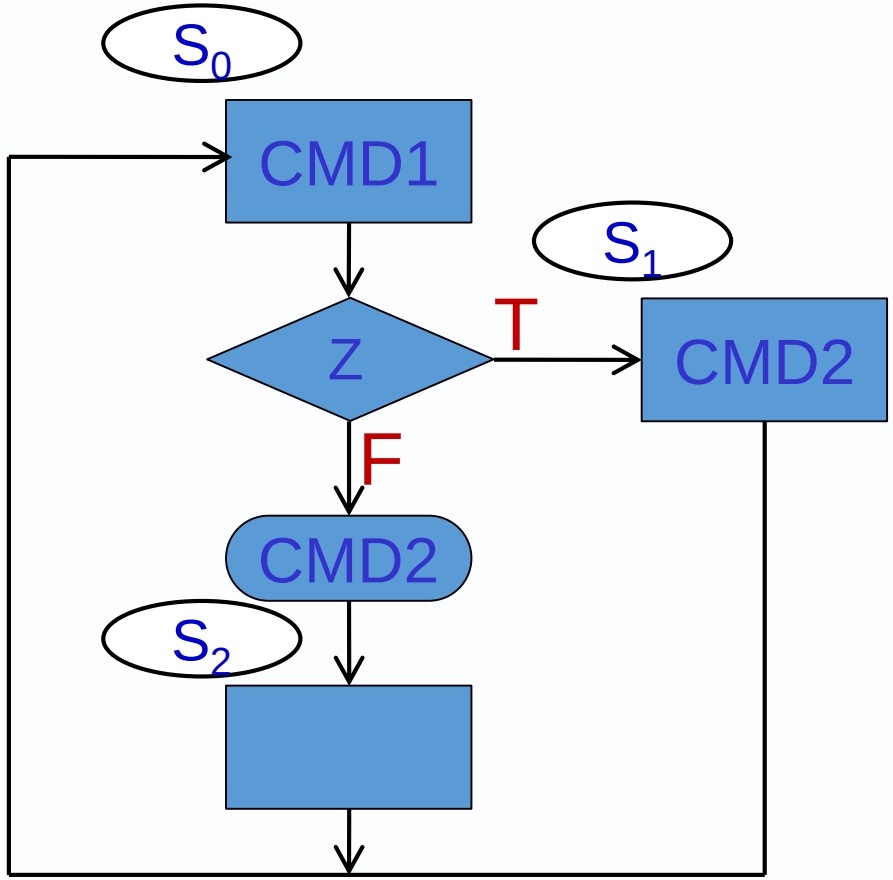
Design Implementation



Input is Z

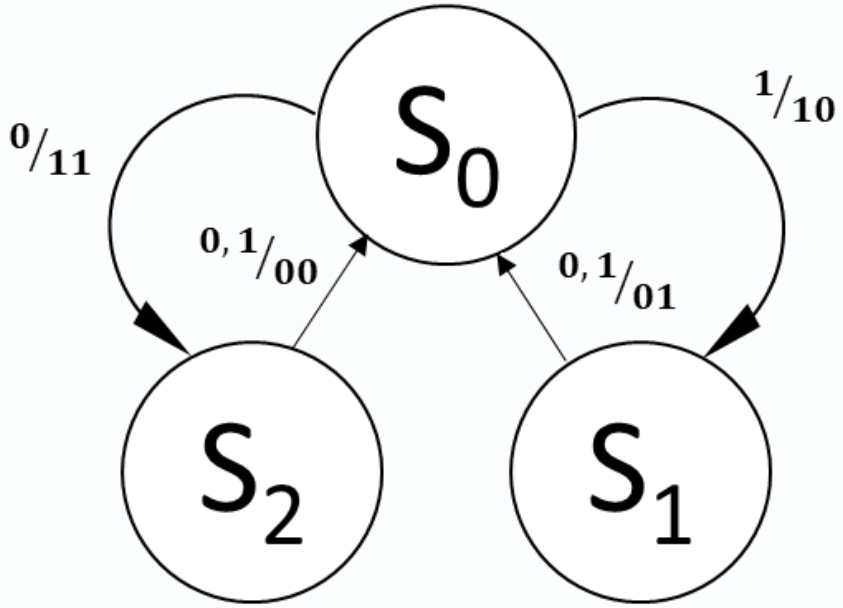
$^{Z}/_{CMD1CMD2}$

Outputs are CMD1, CMD2

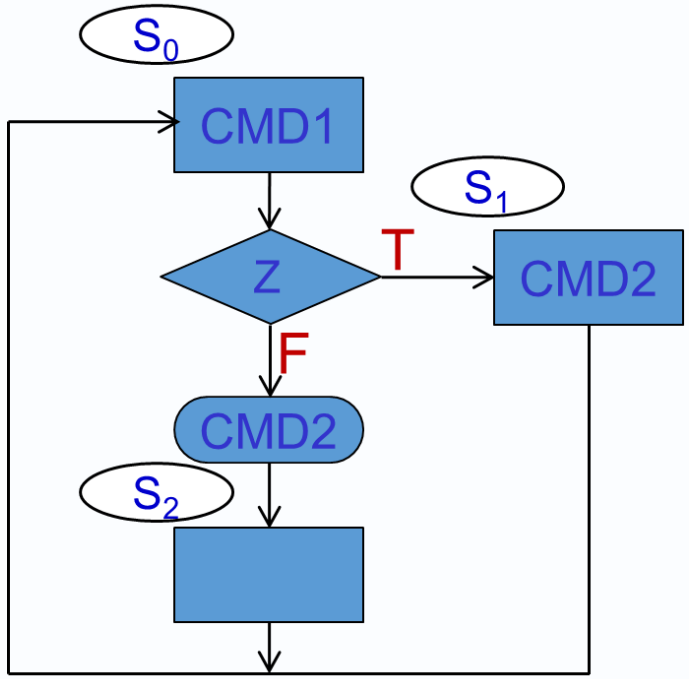


UNIVERSITY OF LIVERPOOL

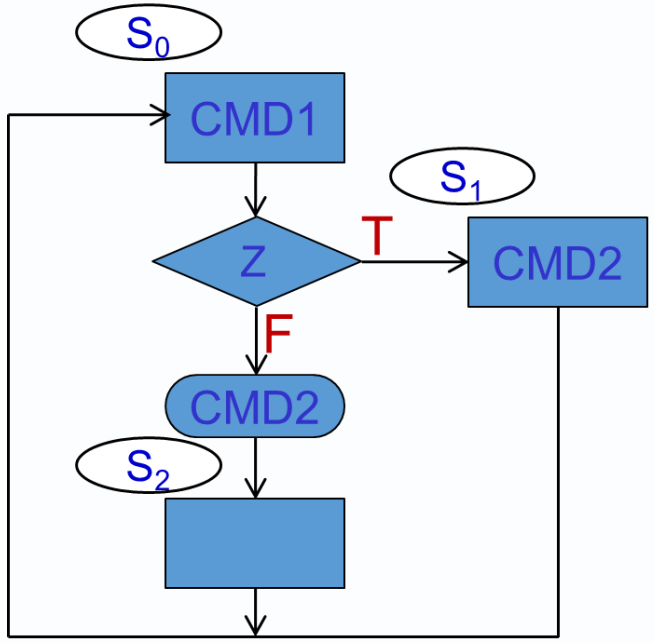
State Graphs v ASM Charts



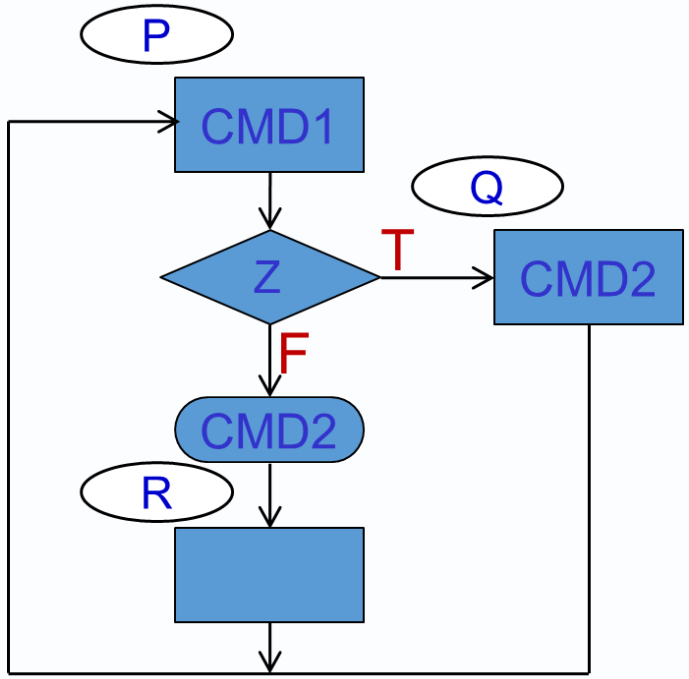
Accounts for all values of input/output
Scaling up is a problem – quickly becomes messy



Clearer depiction of algorithmic structure
Less cluttered (ignore 'don't care' inputs or false outputs
Higher level – allows 'top down' design
Arguably easier to interpret for implementation



State name	State name
S_0	P
S_1	Q
S_2	R



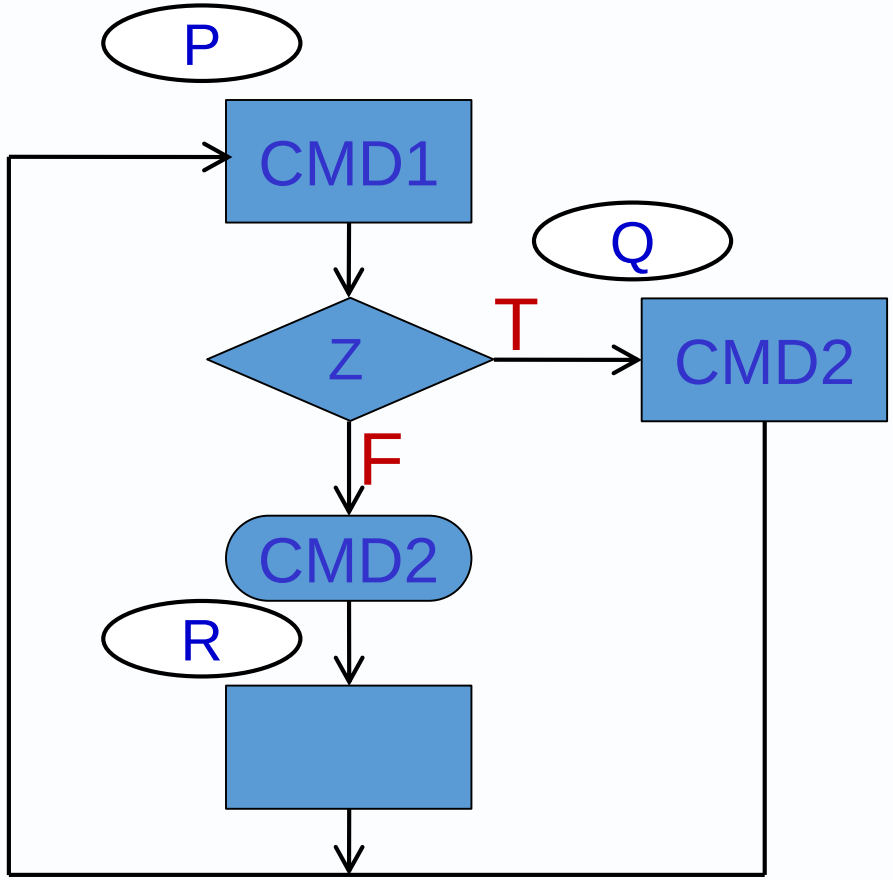
Traditional Design Implementation

Use flip-flops as state memory (either D, J-K, or T types)

There are two ways to represent the present state in flip-flop memory.

Assign a unique binary number to each state (Encoding Method).

Assign one flip-flop to each state (One Hot Method).

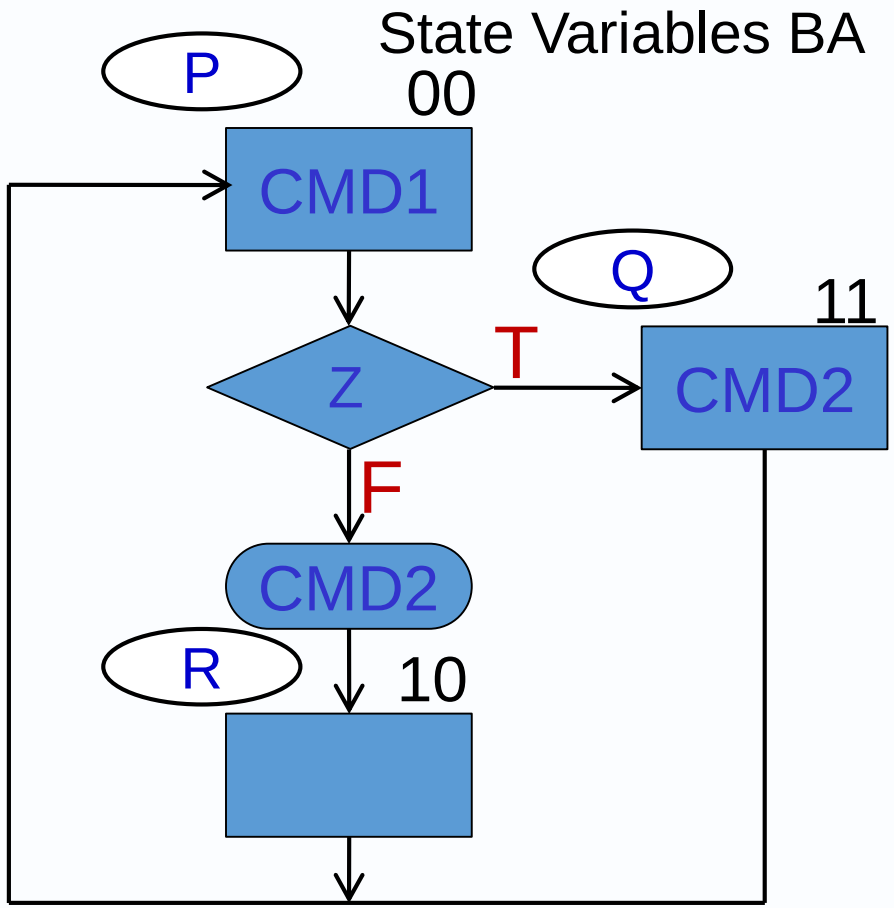


Traditional Design Implementation

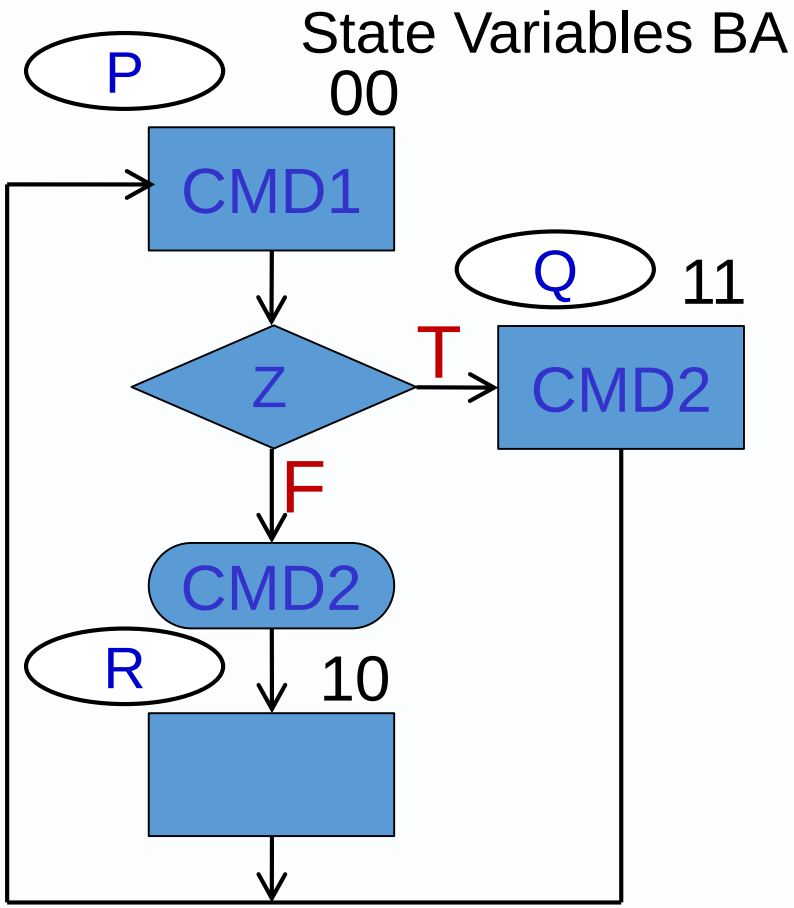
Using the Encoding method, n state variables are required for encoding 2^n states.

The state assignment is arbitrary.

Given the present state, we need to compute the new state code to load into the state flip flops.



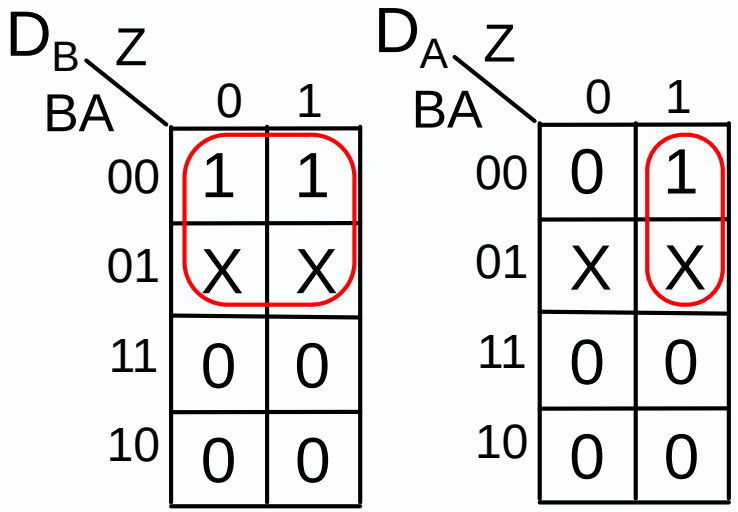
State Transition Table

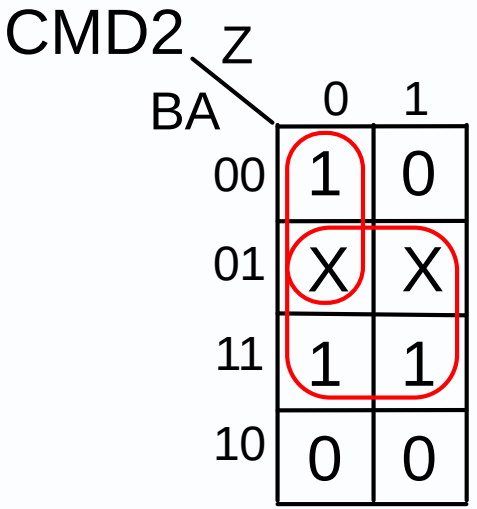


Present State	Input	Next State	Outputs	
BA	Z	$D_B D_A$	CMD1	CMD2
00	0	10	1	1
00	1	11	1	0
01	-	-	-	-
10	-	00	0	0
11	-	00	0	1

Logic Implementation

Present State	Input	Next State	Outputs	
BA	Z	D_BD_A	CMD1	CMD2
00	0	10	1	1
00	1	11	1	0
11	-	00	0	1
10	-	00	0	0
01	-	-	-	-





$D_B = B'$

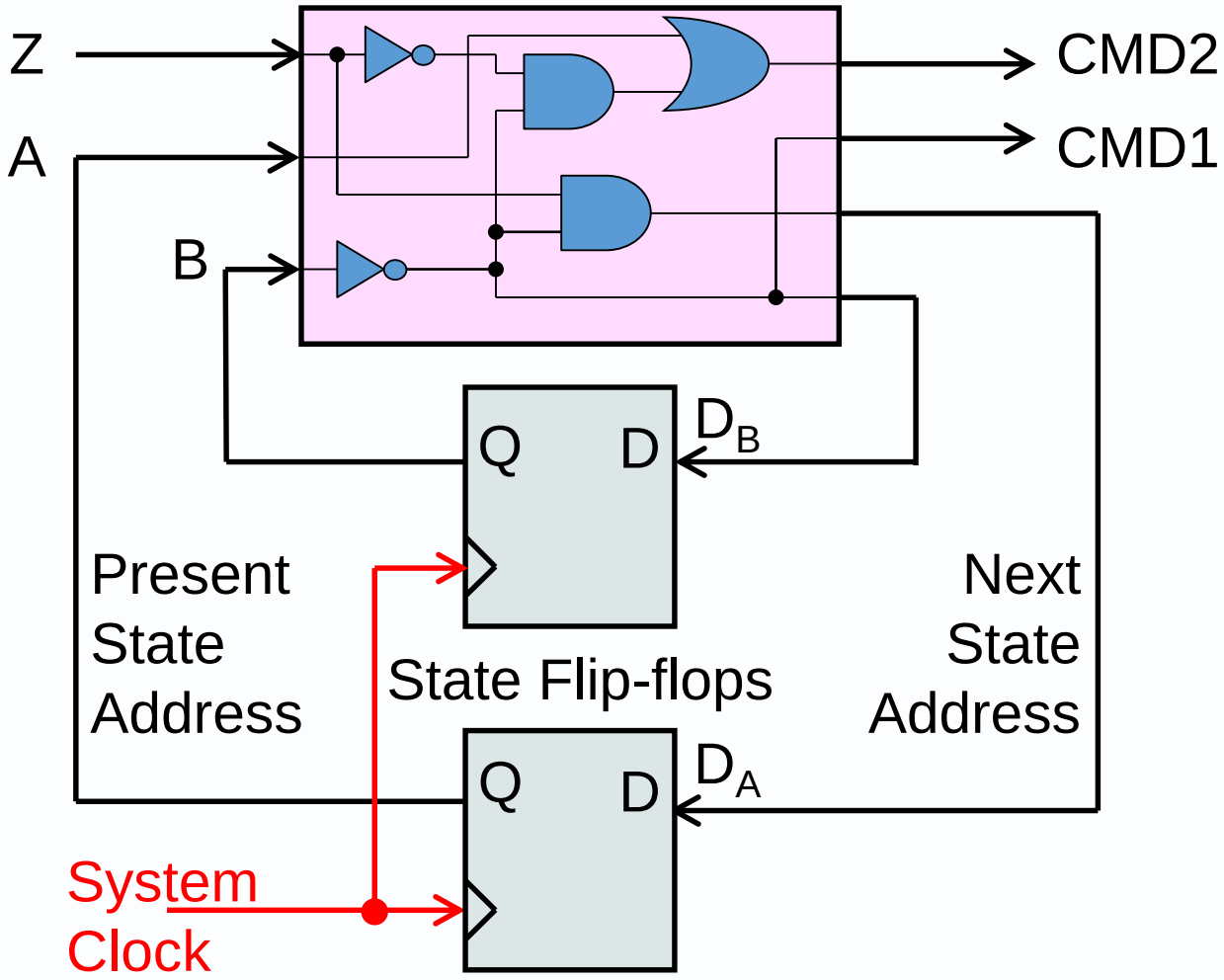
$D_A = Z.B'$

CMD1 = B'

CMD2 = A+ Z'.B'

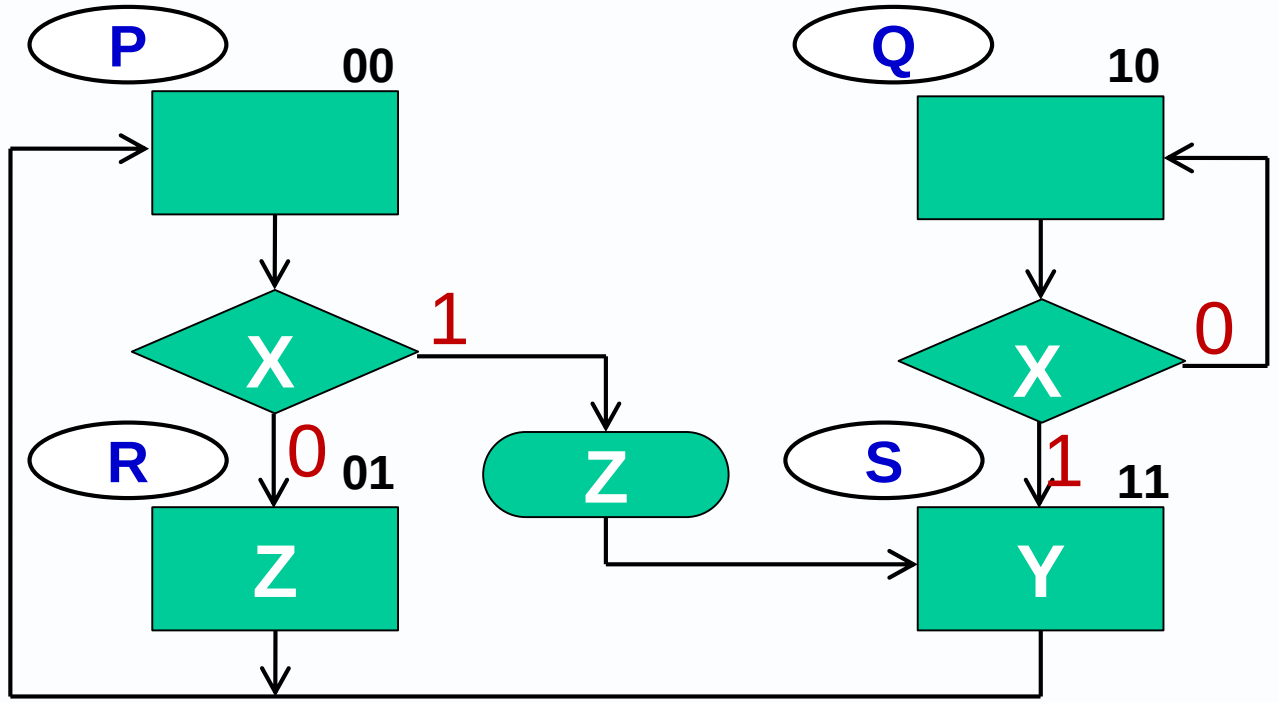
UNIVERSITY OF LIVERPOOL

ASM Implementation



Question

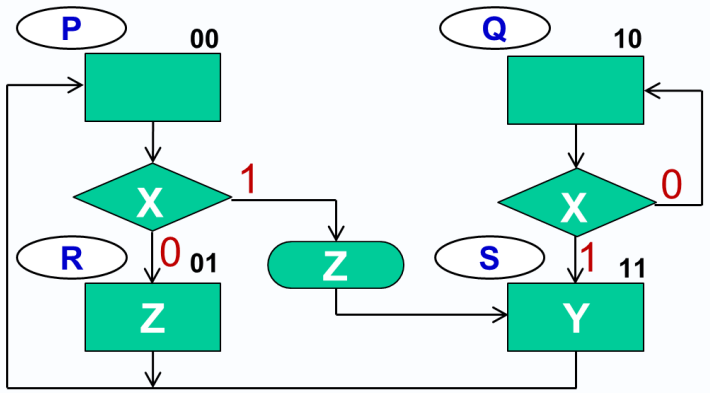
Implement the following ASM using D type flip flops. Use state variables BA.



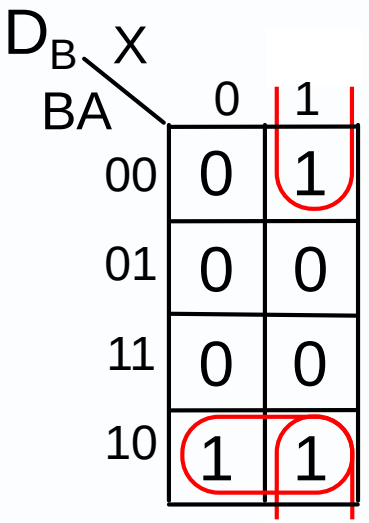
The question requires us to design simplified Boolean equations for the outputs and next state logic

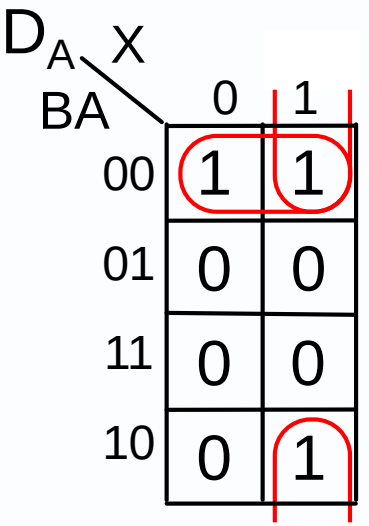
UNIVERSITY OF LIVERPOOL

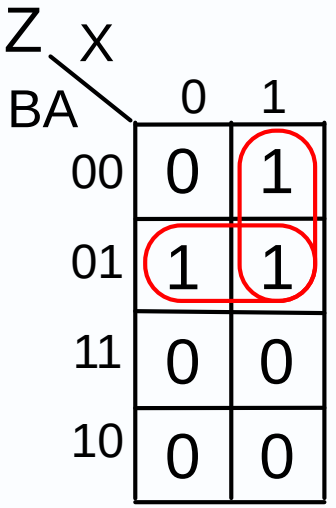
Answer



Present State	Input	Next State	Outputs	
BA	X	D_BD_A	Y	Z
00	0	01	0	0
00	1	11	0	1
01	-	00	0	1
10	0	10	0	0
10	1	11	0	0
11	-	00	1	0







Answer…

$$D_B = B.\bar{A} + X.\bar{A}$$

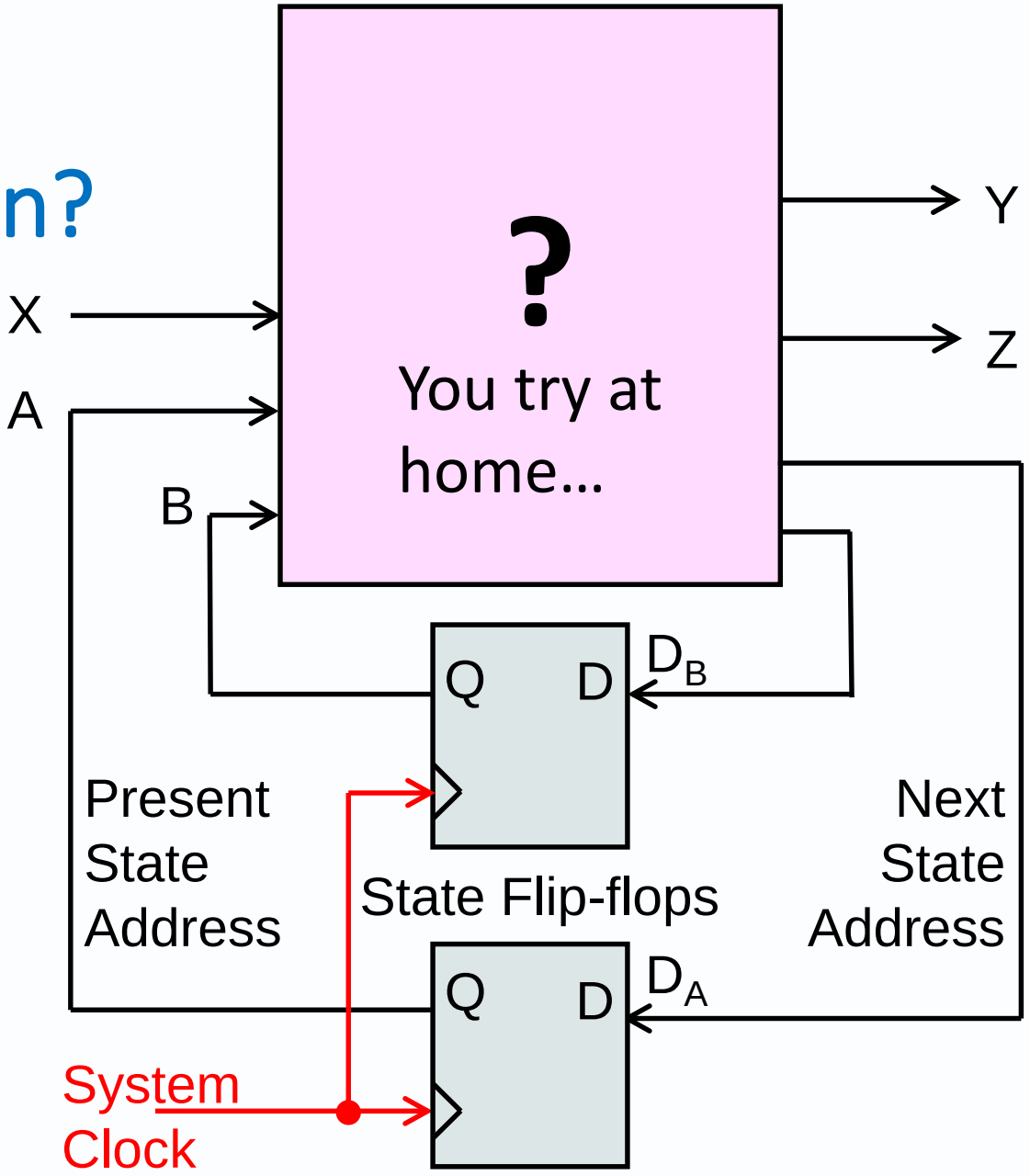
$$D_A = \bar{B}.\bar{A} + X.\bar{A}$$

$$Y = B.A$$

$$Z = \bar{B}.A + X.\bar{B}$$

UNIVERSITY OF LIVERPOOL

ASM Implementation?



Sequential Design: One Hot Method

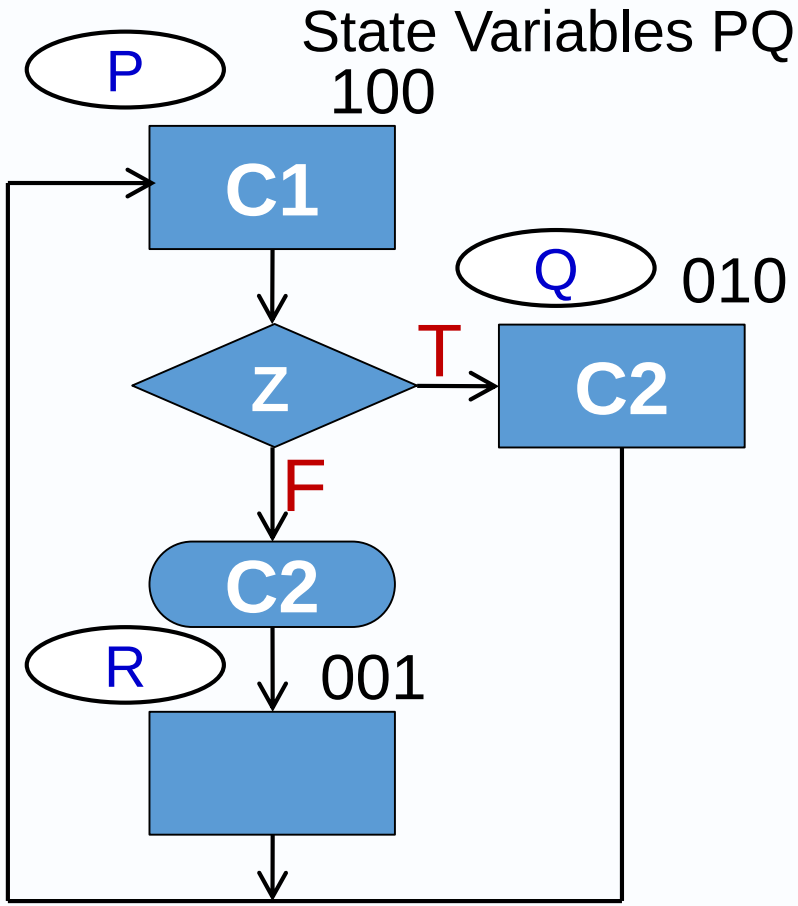
- The ‘one hot’ method uses *n* flip flops for *n* states with 1 ‘hot’ flip flop per state.
- This is different from the ‘encoded’ method where a sequential system with up to 2^n states would use a register with *n* flip-flops.
- A n-to-2^n-line decoder would provide an output corresponding to each of the states for the ‘encoded’ method.
- A decoder is not needed if the one-hot method (one flip-flop per state) is used.

One-Hot Design:
One Flip-Flop per State

Only one of the flip-flops contains a **1** at any time; all others are reset to **0**. The single **1** propagates from one flip-flop to another under the control of decision logic. Each state needs a flip-flop, therefore the system cost looks higher.

The combinational part can be simpler (as decoder is not needed) so the circuit can operate at higher frequencies. This method offers simplicity so that the logic can be designed by inspection of the ASM chart or the state diagram.

ASM – One Hot Implementation



State Transition Table

Present State	Input	Next State	Outputs	
PQR	Z	$D_PD_QD_R$	C1	C2
001	-	100	0	0
010	-	100	0	1
100	0	001	1	1
100	1	010	1	0

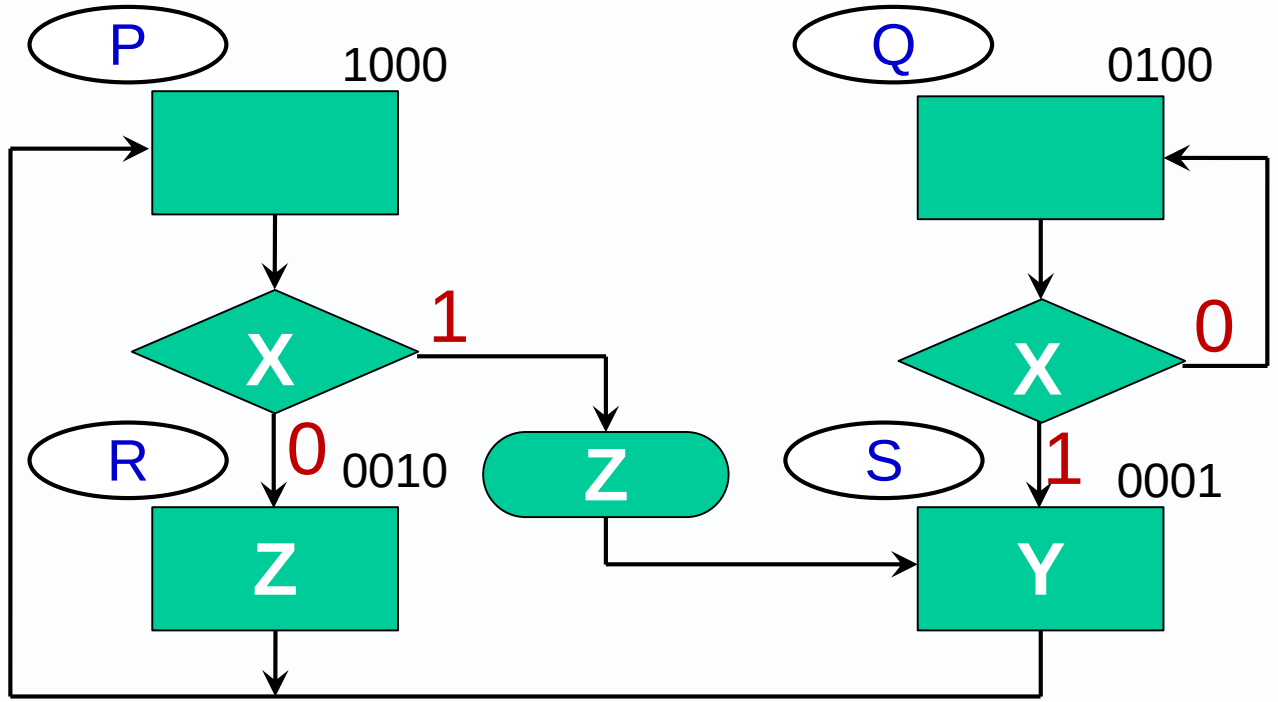
Next state logic $D_P = Q + R$ $D_Q = Z.P$ $D_R = Z'.P$

Outputs $C1 = P$ $C2 = Q + Z'.P$

We have designed simplified Boolean equations for the outputs and next state logic

Question

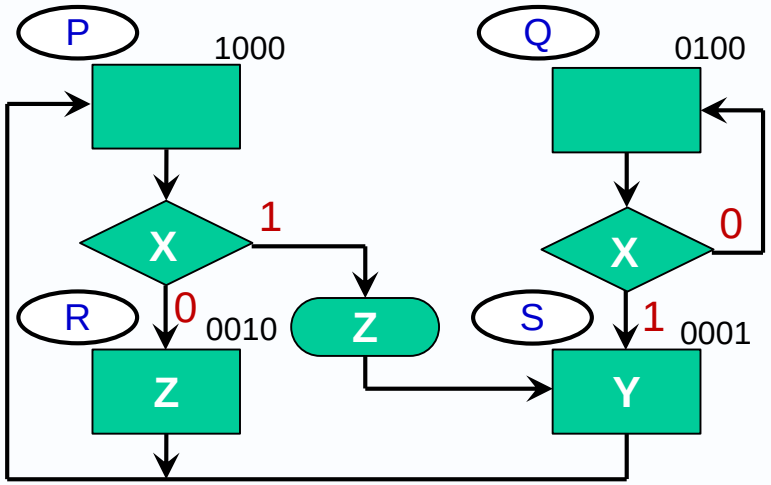
- Implement the following ASM using D type flip flops. Use the one hot method.



The question requires us to design simplified Boolean equations for the outputs and next state logic

UNIVERSITY OF LIVERPOOL

Answer



State Transition Table

Present State	Input	Next State	Outputs	
PQRS	X	$D_P D_Q D_R D_S$	Y	Z
1000	0	0010	0	0
1000	1	0001	0	1
0100	0	0100	0	0
0100	1	0001	0	0
0010	-	1000	0	1
0001	-	1000	1	0

Next state logic

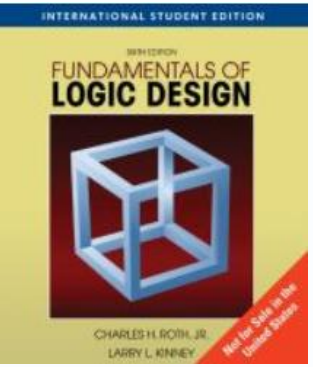
$D_P = R + S$ $D_Q = X'.Q$

$D_R = X'.P$ $D_S = X.(P + Q)$

Outputs

$Y = S$ $Z = R + X.P$

Summary and suggested reading



Chapter 14 State graphs & tables

Section 19.1-2 [A]SM Charts

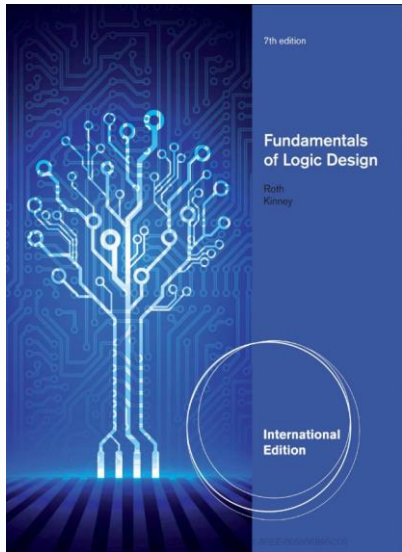
Section 19.3 ASM realisation

Next lecture:

More on ASM design
Serial communications.

Roth and Kinney *Fundamentals of Logic Design*

….. 7th ed. is available as an e-book!



UNIVERSITY OF LIVERPOOL