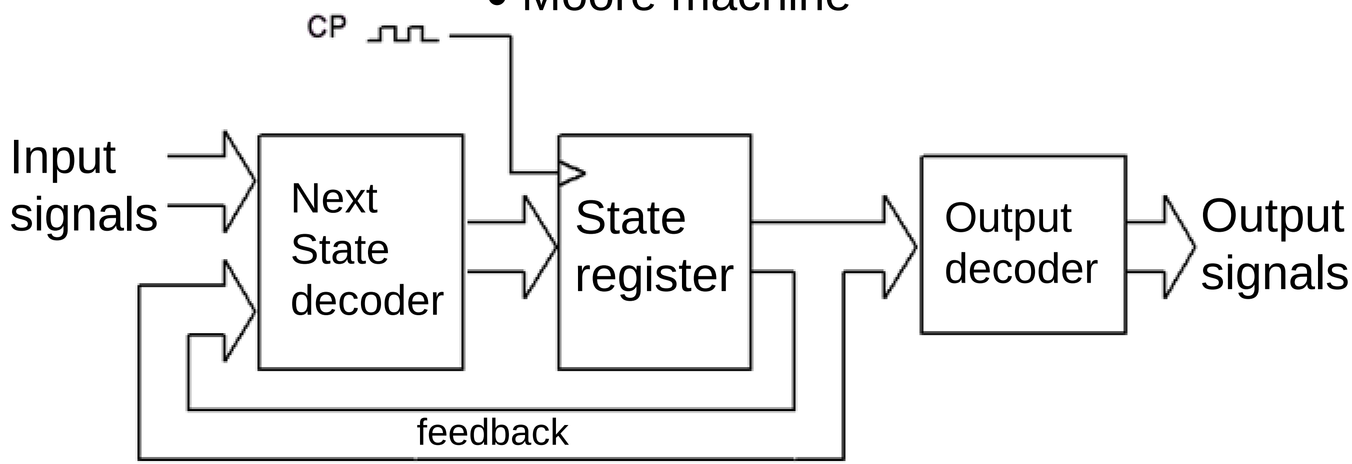
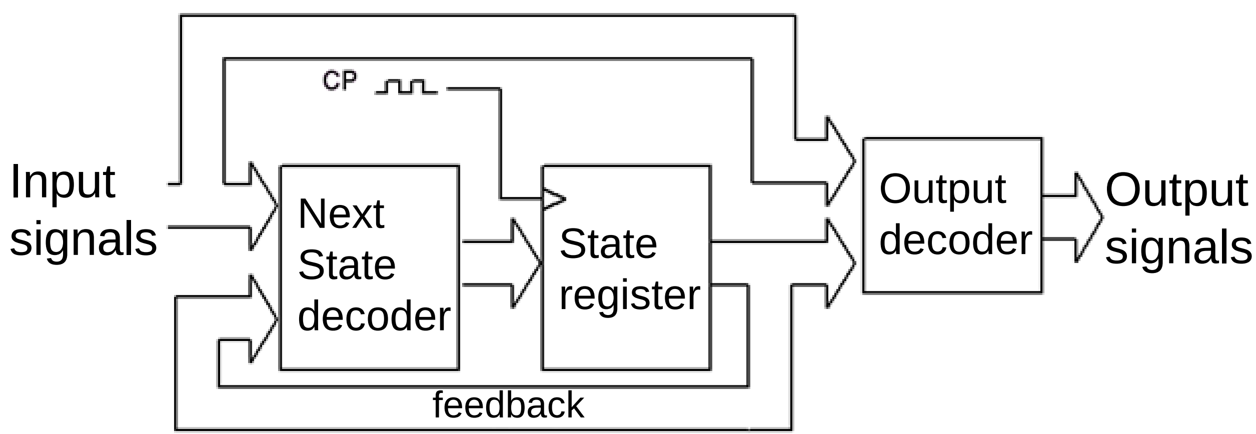
Statemachines

- Moore machine



- Mealy machine

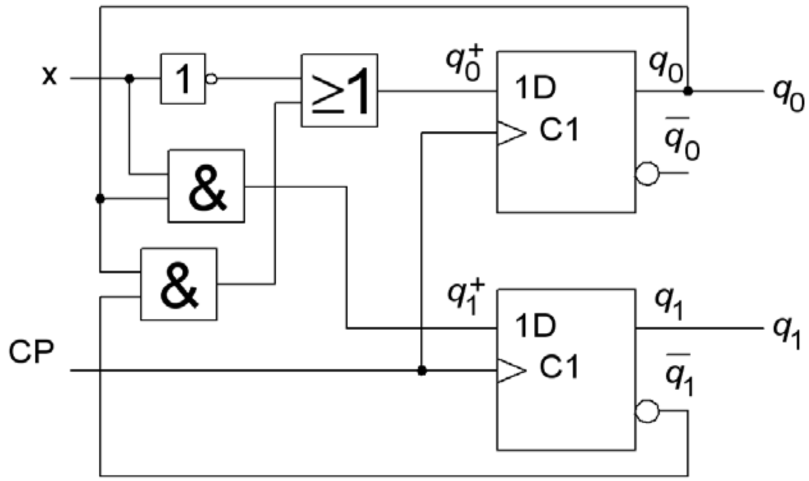


William Sandqvist william@kth.se

Ex 10.1

Determine the state diagram and state table for the sequential circuit.

Which of the models Mealy or Moore fits the circuit?



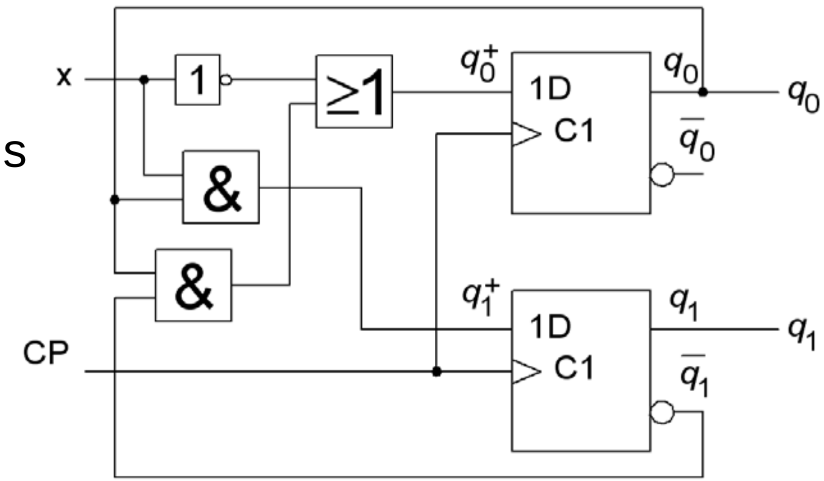
10.1

From the schematic, the following equations can be set up:

$q_1 \quad q_0$ output signals

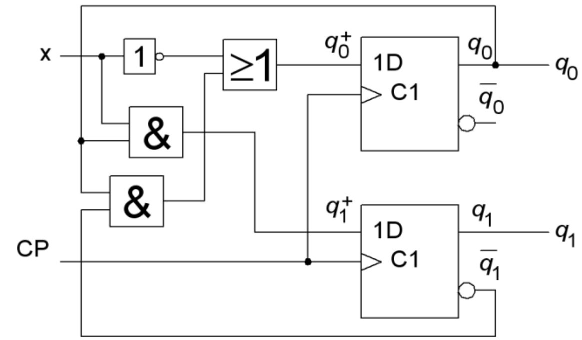
$$q_1^+ = x \cdot q_0$$

$$q_0^+ = \overline{x} + \overline{q_1} \cdot q_0$$



No output decoder is used, the state of the flip-flops are used directly as output. **Moore**-model is used.

10.1



$$q_1^+ = x \cdot q_0 \qquad q_0^+ = \overline{x} + \overline{q_1} \cdot q_0$$

$q_1^+ = x\, q_0$

q_1q_0 \ x	0	1
00		
01		1
11		1
10		

$x\, q_0$

$q_0^+ = \overline{q_1} q_0 + \overline{x}$

q_1q_0 \ x	0	1
00	1	
01	1	1
11	1	
10	1	

$\overline{q_1} q_0$

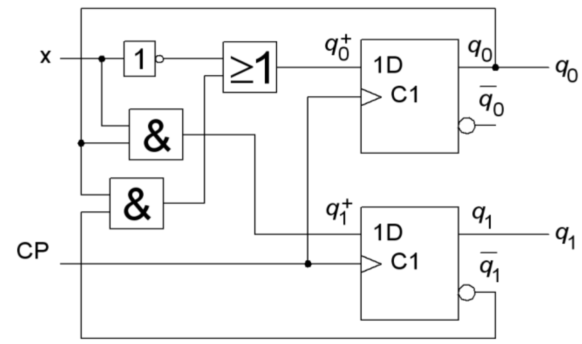
$\overline{x}$

$q_1^+ q_0^+ (q_1 q_0 x)$

q_1q_0 \ x	0	1
00		
01		
11		
10		

10.1

$$q_1^+ = x \cdot q_0 \qquad q_0^+ = \overline{x} + \overline{q_1} \cdot q_0$$



$q_1^+ = x\, q_0$

$q_1q_0 \backslash x$	0	1
00		
01		1
11		1
10		

$x\, q_0$

$q_0^+ = \overline{q_1} q_0 + \overline{x}$

$q_1q_0 \backslash x$	0	1
00	1	
01	1	1
11	1	
10	1	

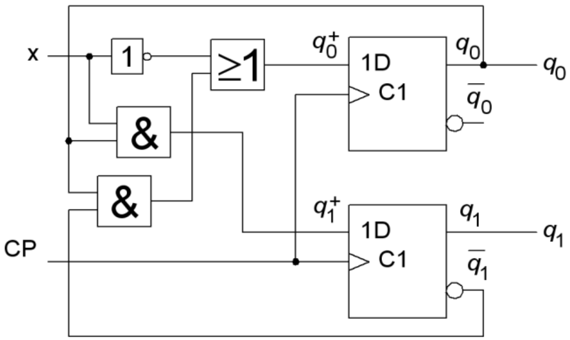
$\overline{q_1} q_0$

$\overline{x}$

$q_1^+ q_0^+ (q_1 q_0 x)$

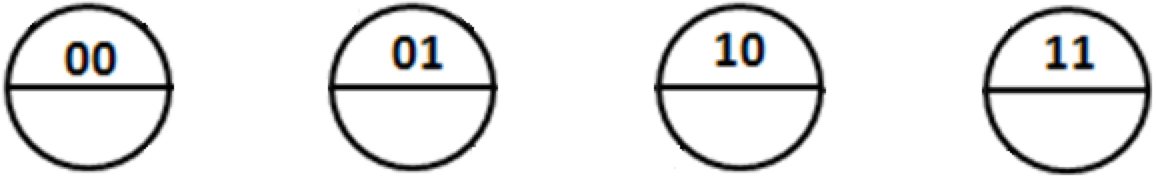
$q_1q_0 \backslash x$	0	1
00	01	00
01	01	11
11	01	10
10	01	00

10.1

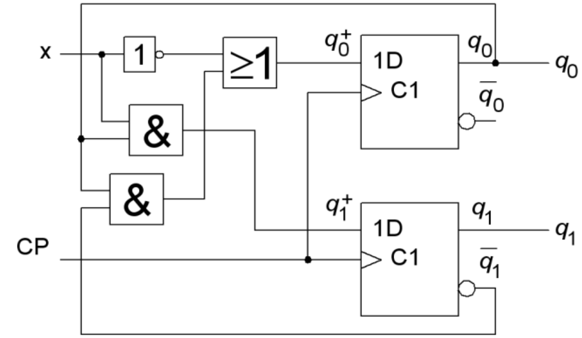


$q_1^+q_0^+(q_1q_0x)$

q_1q_0 \ x	0	1
00	01	00
01	01	11
11	01	10
10	01	00

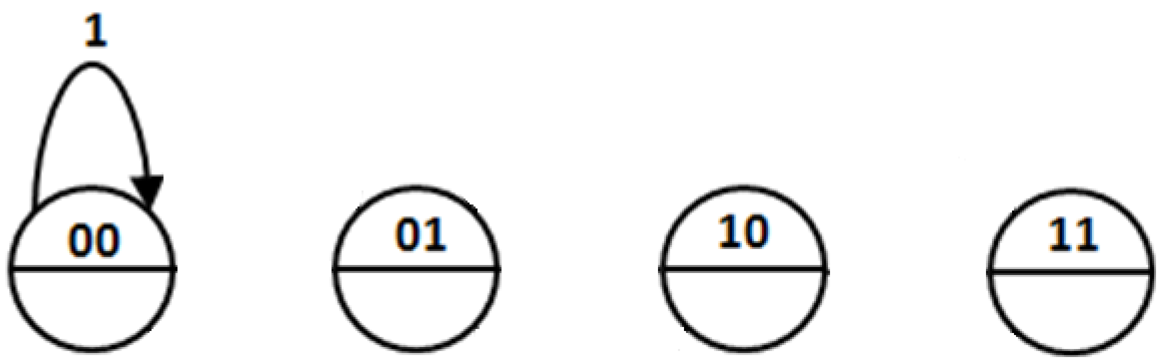


10.1

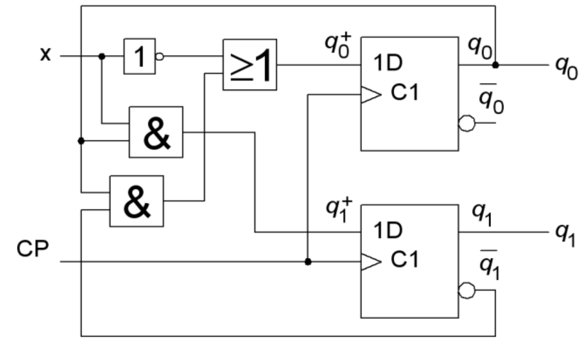


$q_1^+q_0^+(q_1q_0x)$

q_1q_0 \ x	0	1
00	01	00
01	01	11
11	01	10
10	01	00

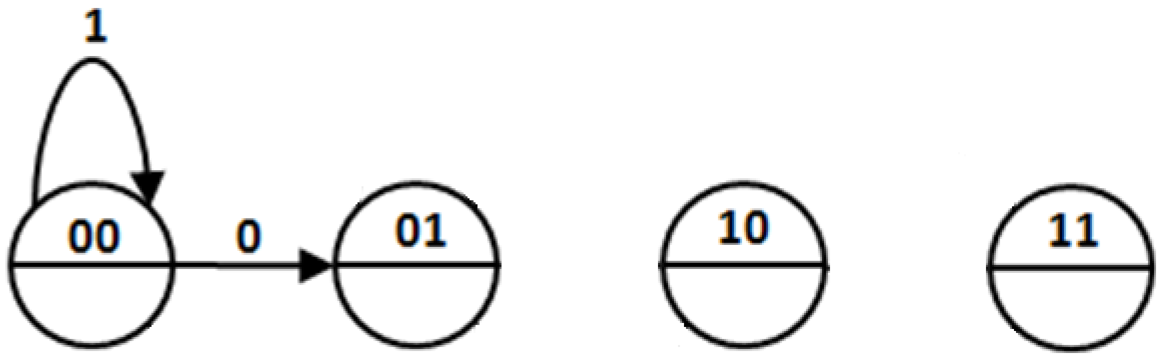


10.1



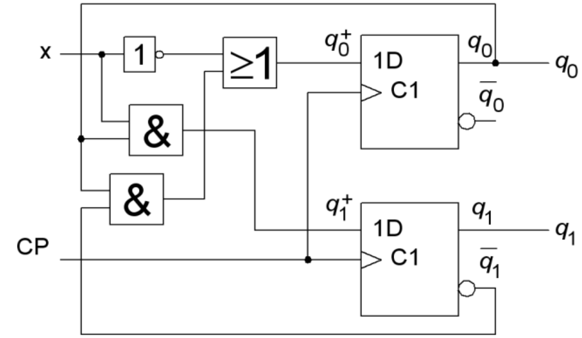
$q_1^+q_0^+(q_1q_0x)$

q_1q_0 \ x	0	1
00	01	00
01	01	11
11	01	10
10	01	00



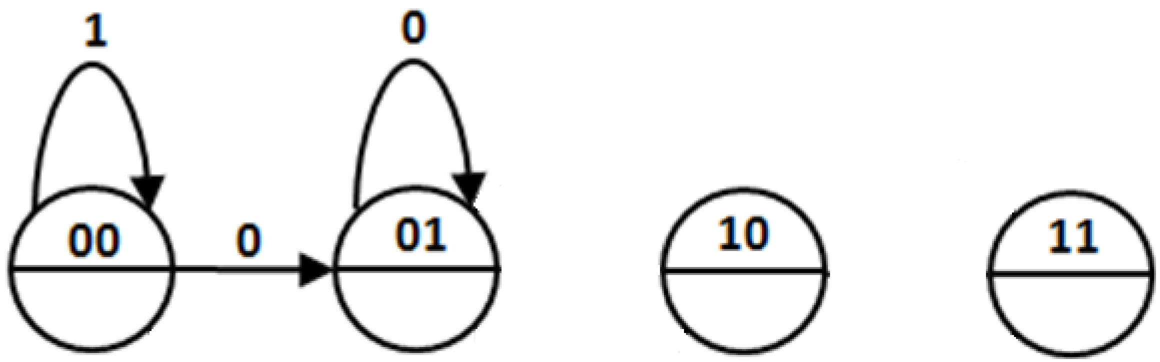
William Sandqvist william@kth.se

10.1

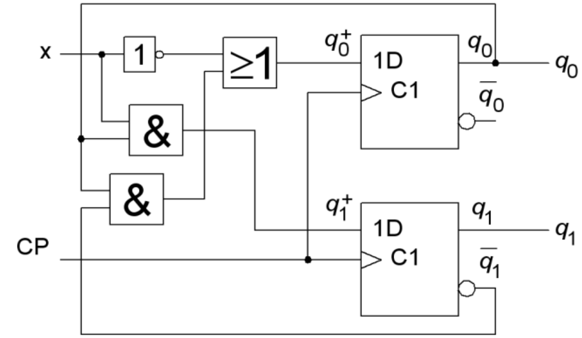


$q_1^+q_0^+(q_1q_0x)$

q_1q_0 \ x	0	1
00	01	00
01	01	11
11	01	10
10	01	00

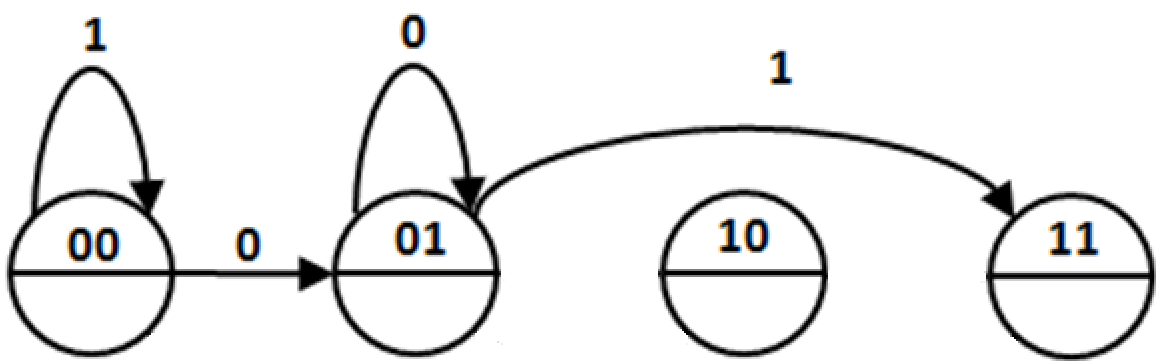


10.1



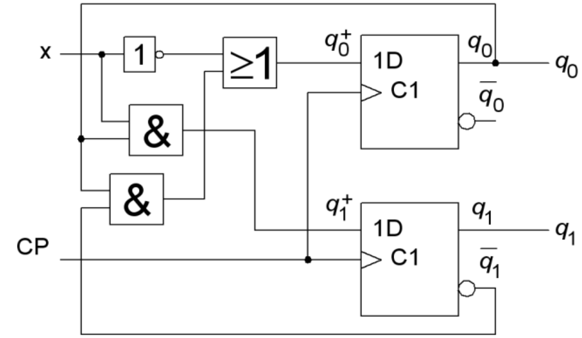
$q_1^+q_0^+(q_1q_0x)$

q_1q_0 \ x	0	1
00	01	00
01	01	11
11	01	10
10	01	00



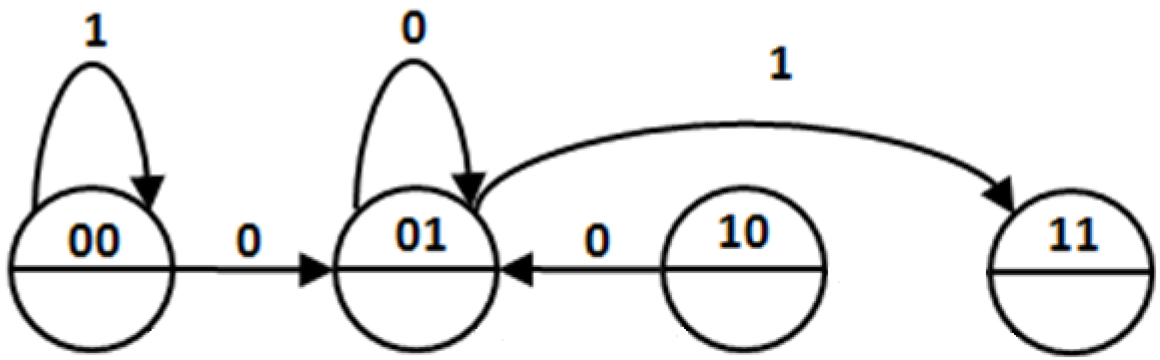
William Sandqvist william@kth.se

10.1



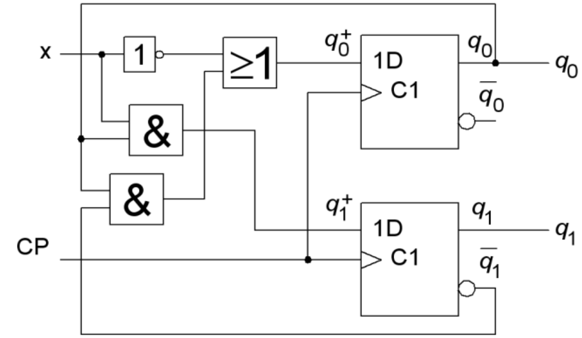
$q_1^+q_0^+(q_1q_0x)$

q_1q_0 \ x	0	1
00	01	00
01	01	11
11	01	10
10	01	00



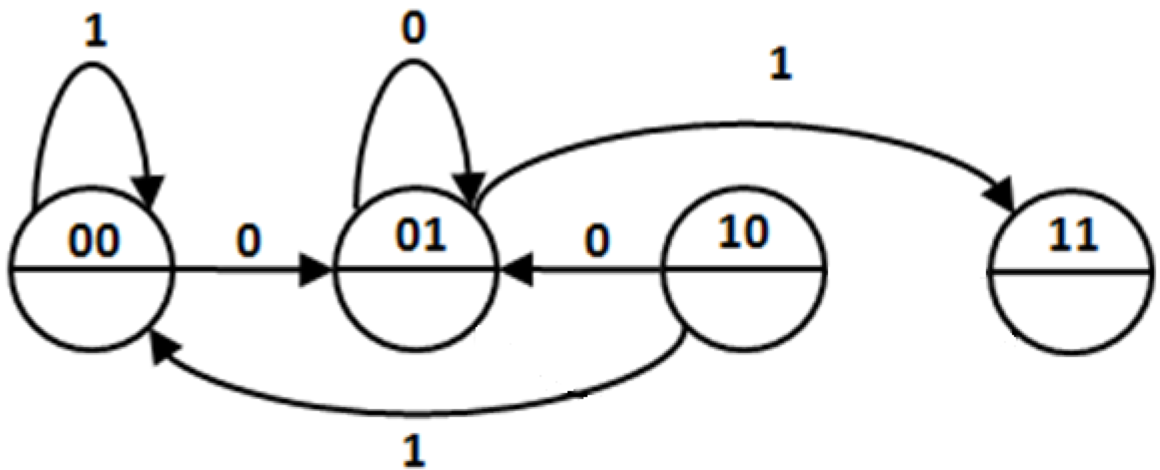
William Sandqvist william@kth.se

10.1

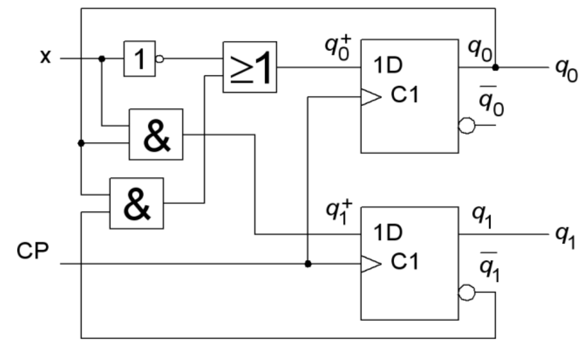


$q_1^+q_0^+(q_1q_0x)$

q_1q_0 \ x	0	1
00	01	00
01	01	11
11	01	10
10	01	00

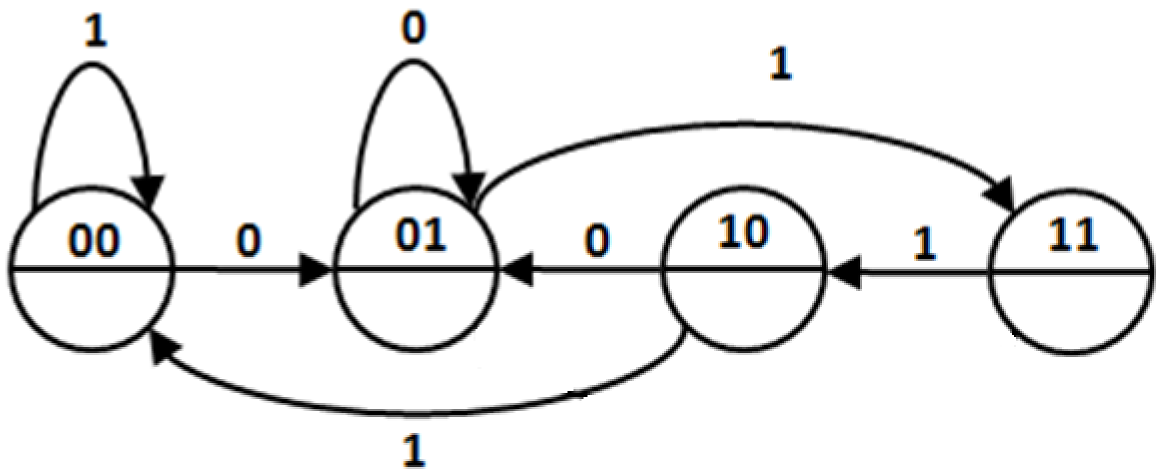


10.1

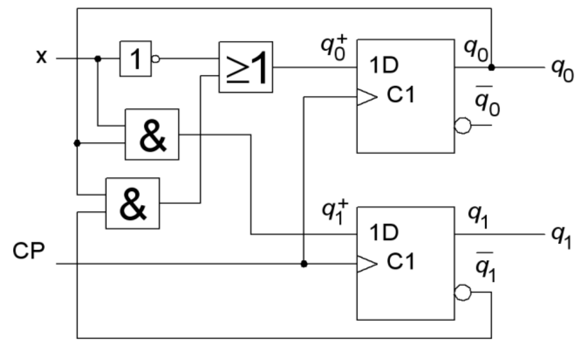


$q_1^+q_0^+(q_1q_0x)$

q_1q_0 \ x	0	1
00	01	00
01	01	11
11	01	10
10	01	00

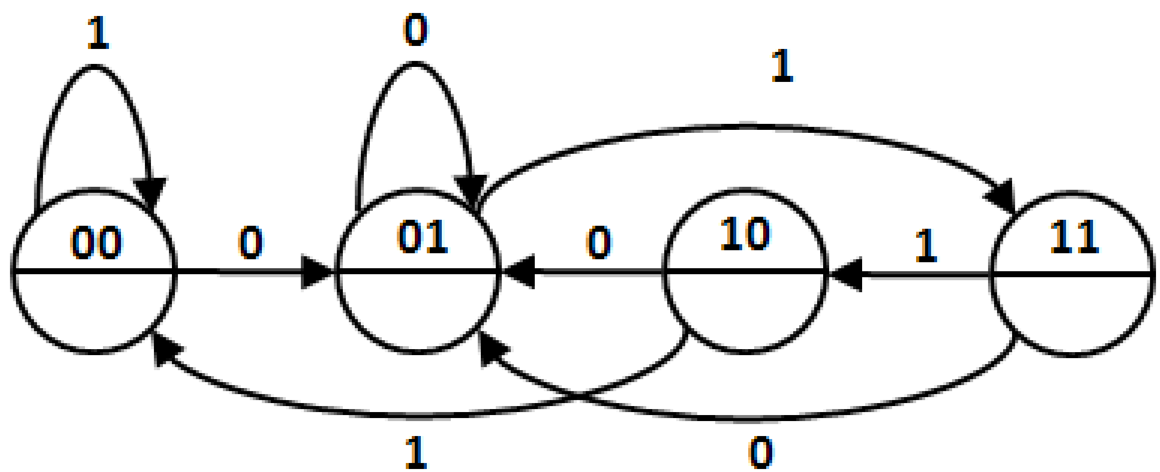


10.1

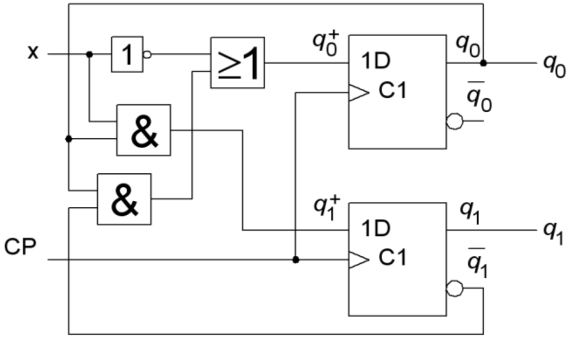


$q_1^+q_0^+(q_1q_0x)$

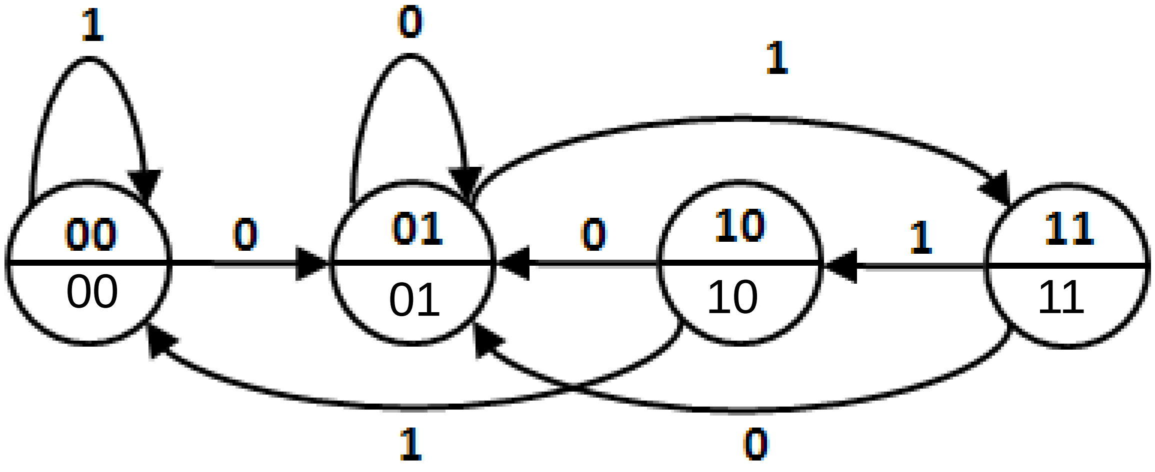
q_1q_0 \ x	0	1
00	01	00
01	01	11
11	01	10
10	01	00



10.1

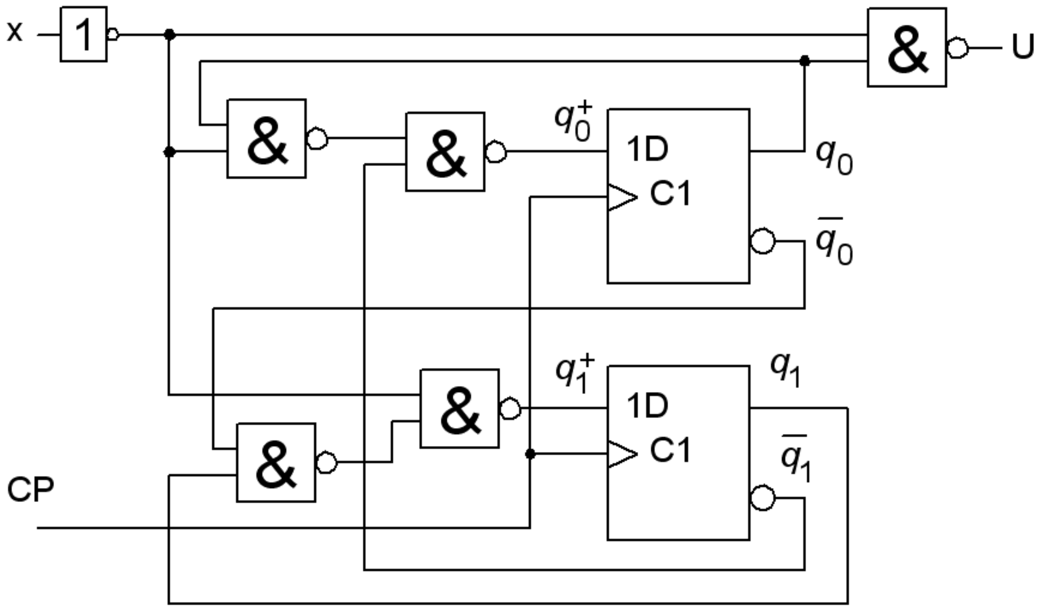


Output is the same as flip-flop state.



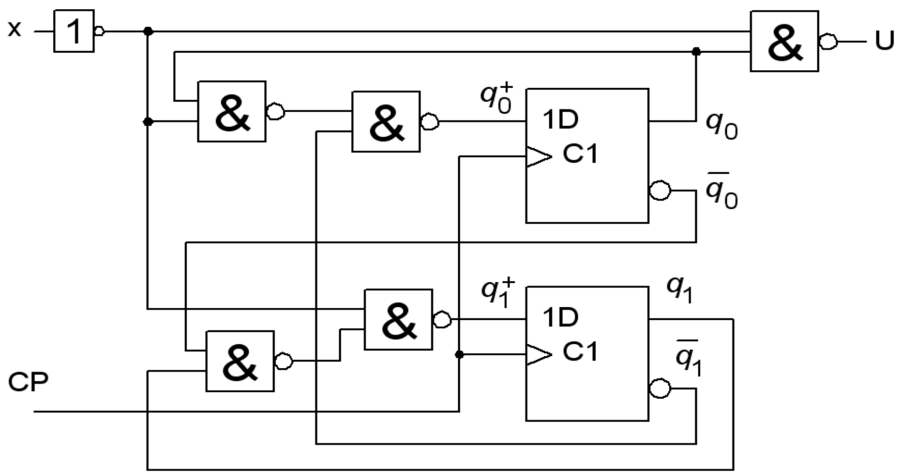
Ex 10.2

Determine the state diagram and state table for the sequence circuit.
Which of the models Mealy or Moore fits on the circuit?



William Sandqvist william@kth.se

10.2



NAND-gates:

$$U = \overline{\overline{x}\cdot q_0} = \{dM\} = x + \overline{q_0}$$

$$q_0^+ = \overline{\overline{q_1}\cdot(\overline{q_0\cdot\overline{x}})} = \{dM\} =$$

$$= q_1 + q_0\overline{x}$$

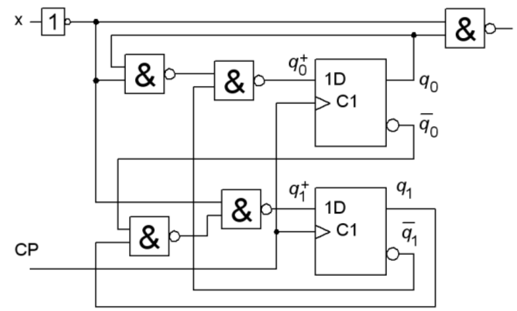
$$q_1^+ = \overline{(\overline{q_1\cdot\overline{q_0}})\cdot\overline{x}} = \{dM\} =$$

$$= x + q_1\overline{q}_0$$

Because *U* is directly depended of *x* so must the Mealy model be used.

De Morgans law gives us the SP form.

10.2



$$U = x + \overline{q_0}$$

$$q_1^+ = x + \overline{q_0} \cdot q_1$$

$$q_0^+ = \overline{x} \cdot q_0 + q_1$$

$q_1^+ = x + \overline{q_0} \cdot q_1$

q_1q_0 \ x	0	1
00		1
01		1
11		1
10	1	1

x

$q_1\overline{q_0}$

$q_0^+ = \overline{x} \cdot q_0 + q_1$

q_1q_0 \ x	0	1
00		
01	1	
11	1	1
10	1	1

$\overline{x}q_0$

q_1

$U = x + \overline{q_0}$

q_1q_0 \ x	0	1
00	1	1
01		1
11		1
10	1	1

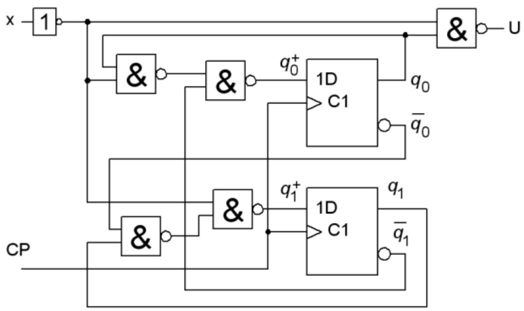
$\overline{q_0}$

x

$q_0^+q_1^+ / U = f(q_0, q_1, x)$

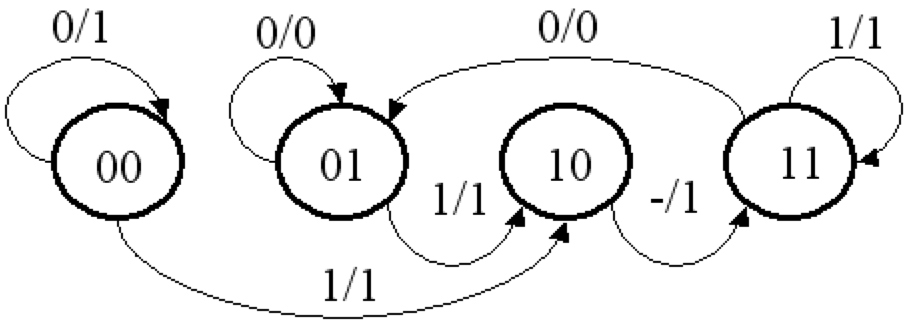
q_1q_0 \ x	0	1
00	00/1	10/1
01	01/0	10/1
11	01/0	11/1
10	11/1	11/1

10.2



$$q_0^+ q_1^+ / U = f(q_0, q_1, x)$$

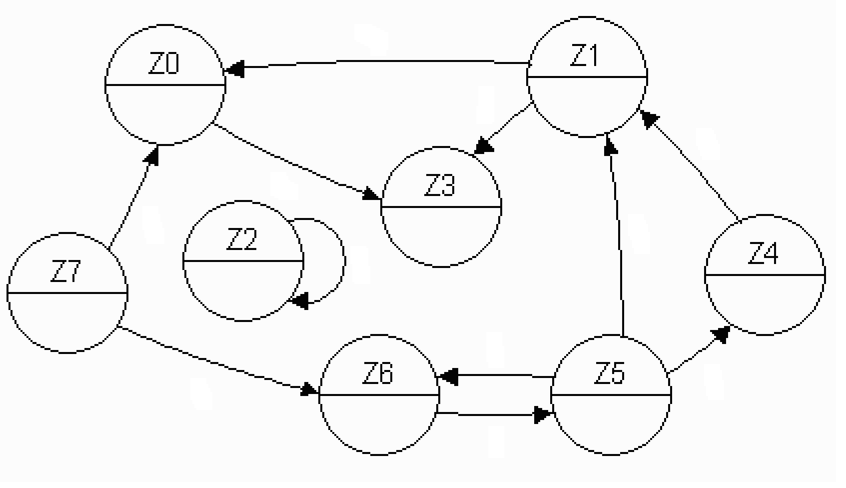
q_0q_1 \ x	0	1
00	00/1	10/1
01	01/0	10/1
11	01/0	11/1
10	11/1	11/1



(Ex 10.4)

Is there any stopping condition, loss condition or isolated states in the state diagram?

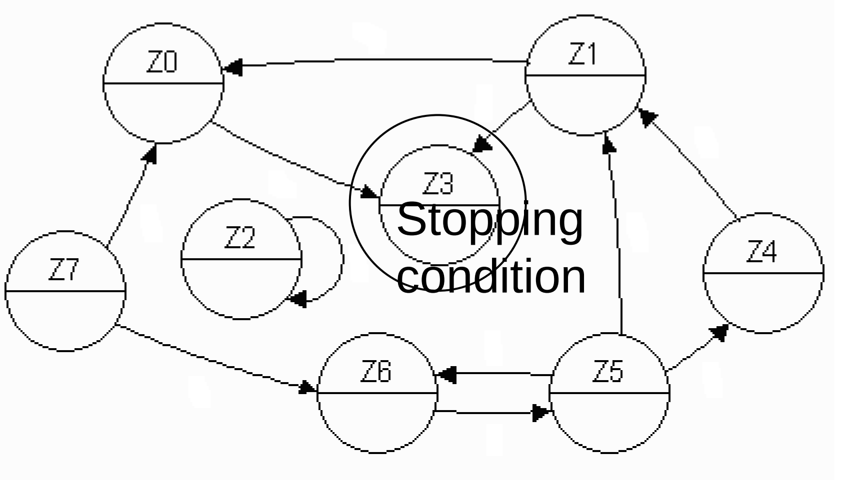
- Stopping condition:
- Loss condition:
- Isolated states:



(Ex 10.4)

Is there any stopping condition, loss condition or isolated states in the state diagram?

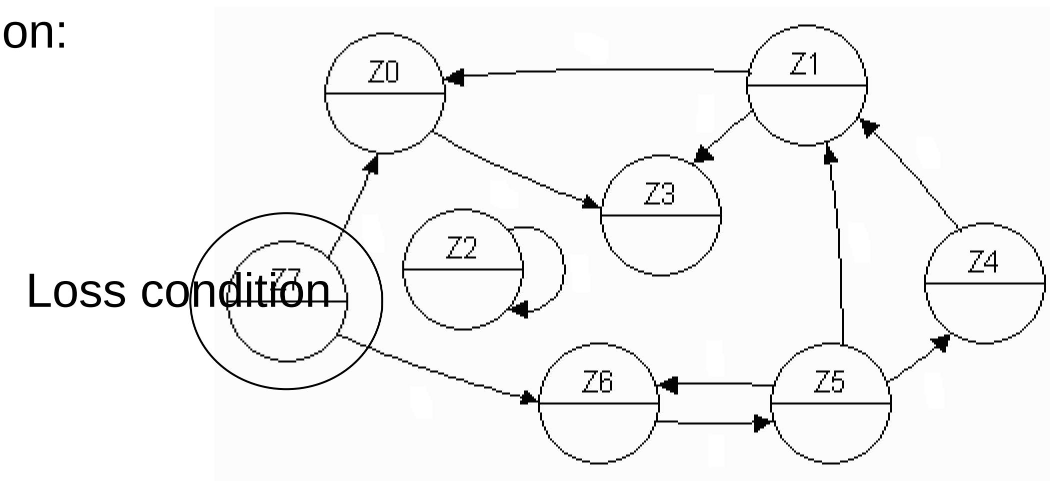
- Stopping condition:
- Loss condition:
- Isolated states:



(Ex 10.4)

Is there any stopping condition, loss condition or isolated states in the state diagram?

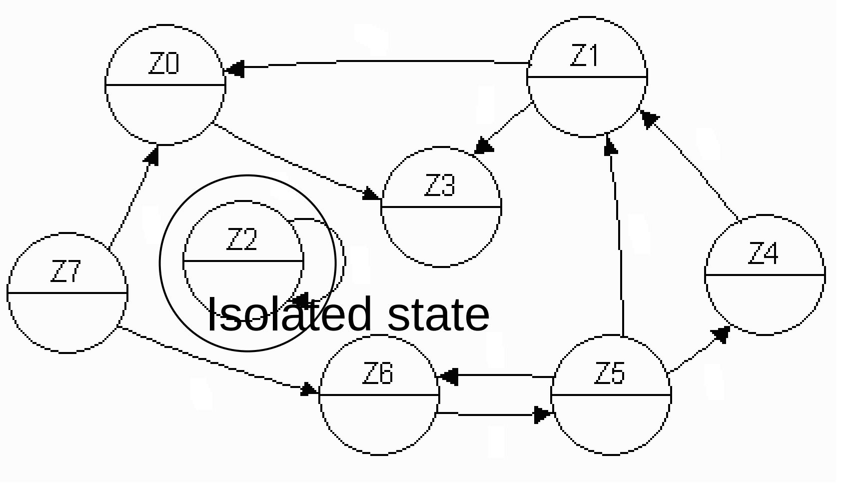
- Stopping condition:
- Loss condition:
- Isolated states:



(Ex 10.4)

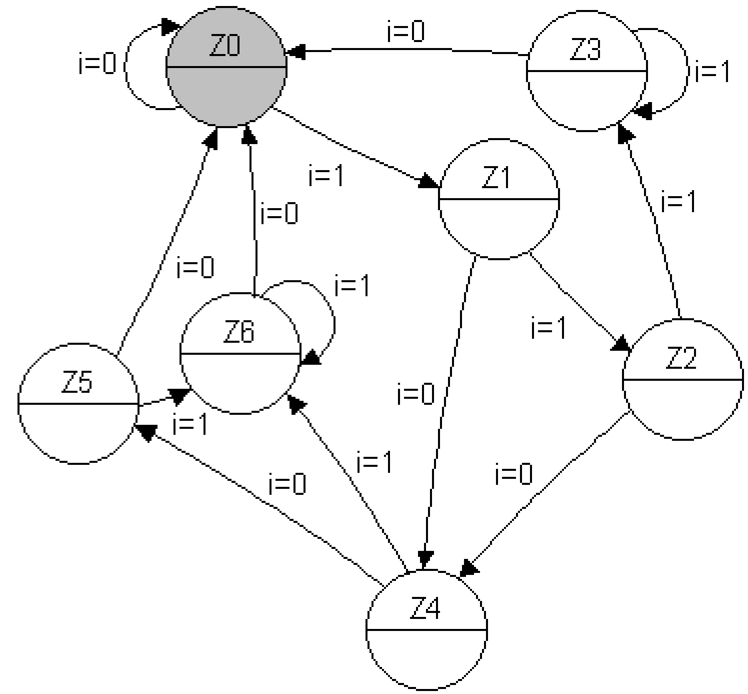
Is there any stopping condition, loss condition or isolated states in the state diagram?

- Stopping condition:
- Loss condition:
- Isolated states:



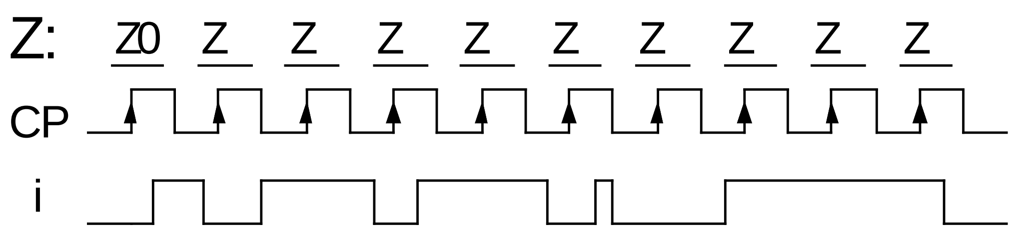
Ex 10.5

To the right is a state diagram for a Moore machine.
(it will detect a double tap).



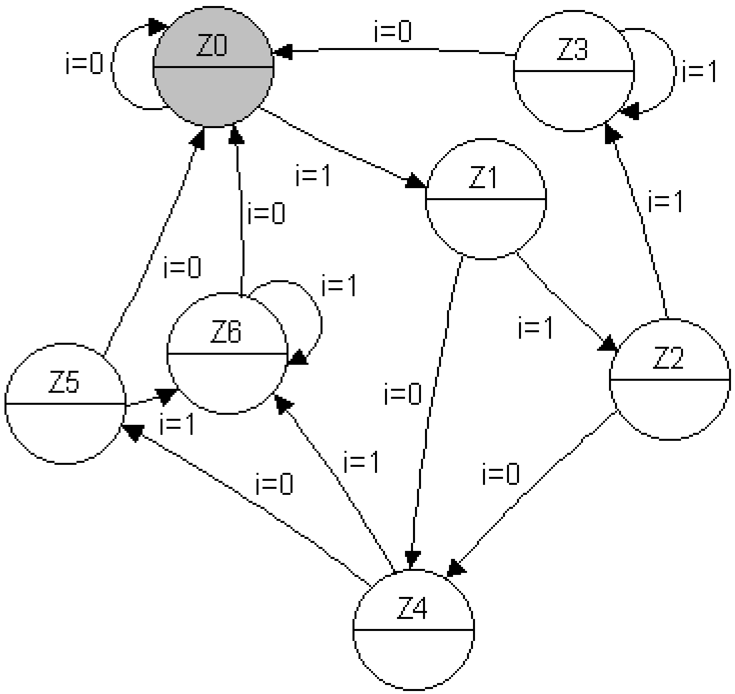
A monkey accidentally get hold of the push-button input signal, and then presses according to the timing diagram below.

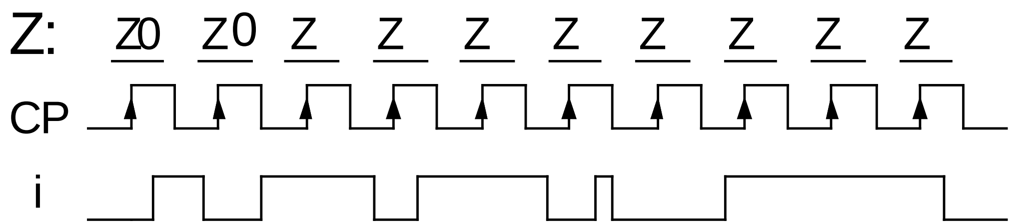
The Moore-machine has flip-flops that are triggered by the positive edge of the clock. Suppose that the initial state is Z0.



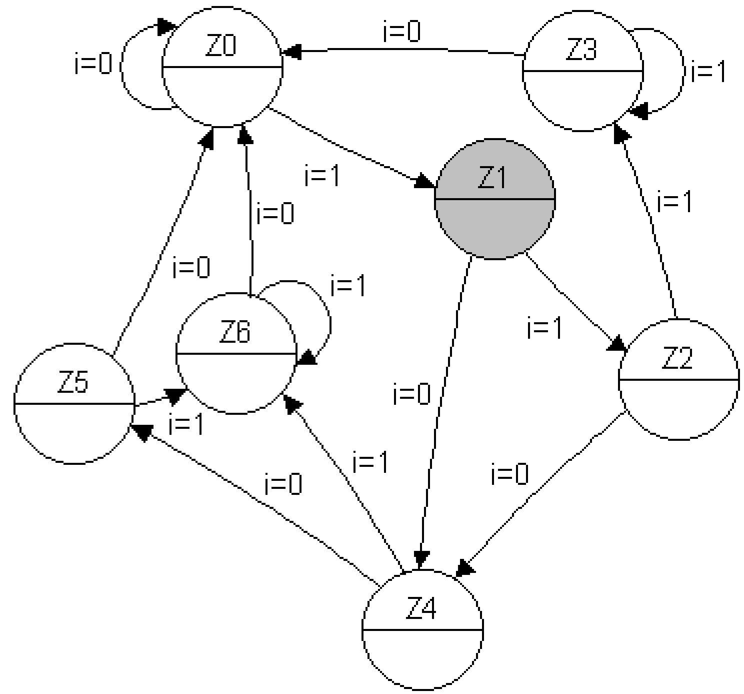
William Sandqvist william@kth.se

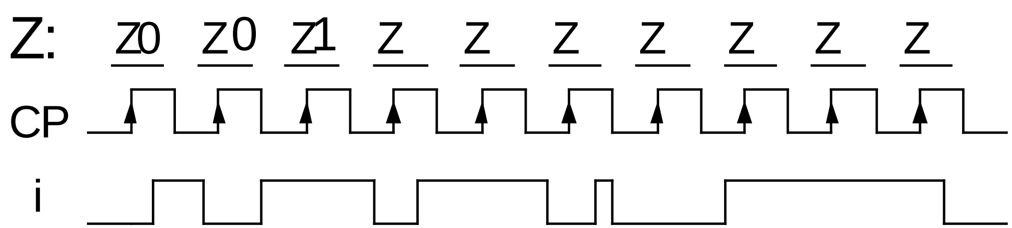
Ex 10.5



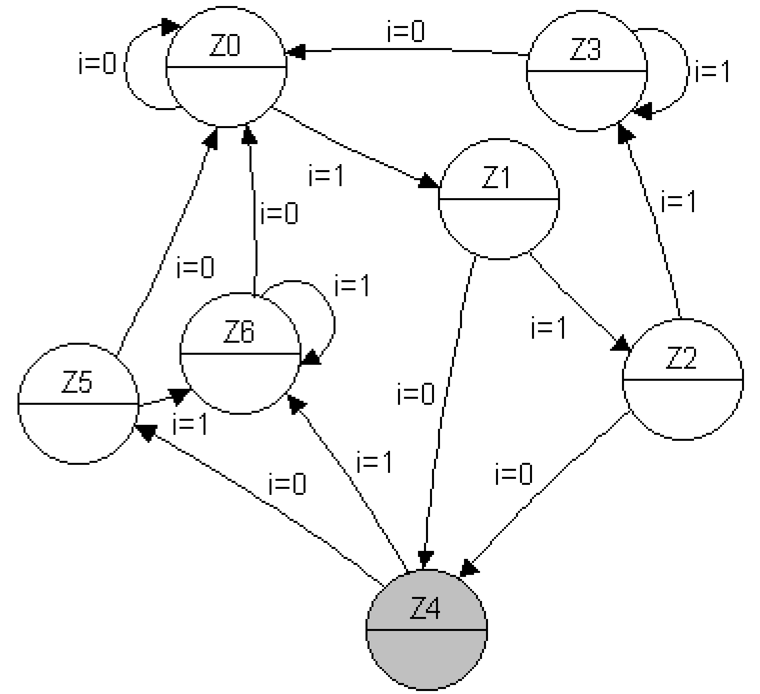


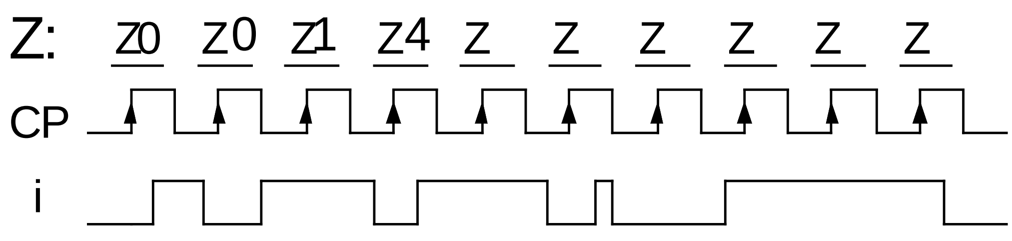
Ex 10.5


Z0
Z1
Z2
Z3
Z4
Z5
Z6
i=0
i=0
i=1
i=1
i=0
i=0
i=1
i=1
i=1
i=0
i=1
i=0
i=1
i=0
i=0

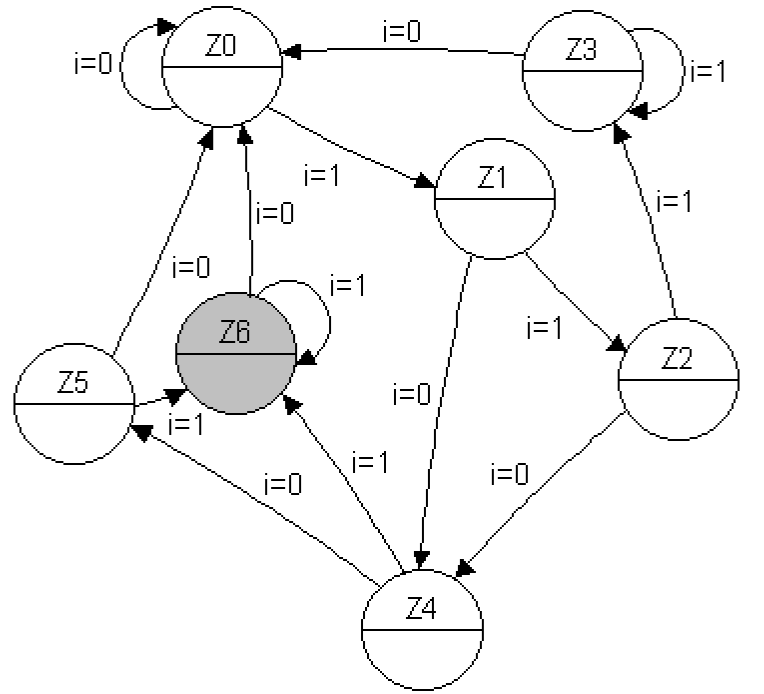

Z:
Z0
Z0
Z1
Z
Z
Z
Z
Z
Z
Z
CP
i

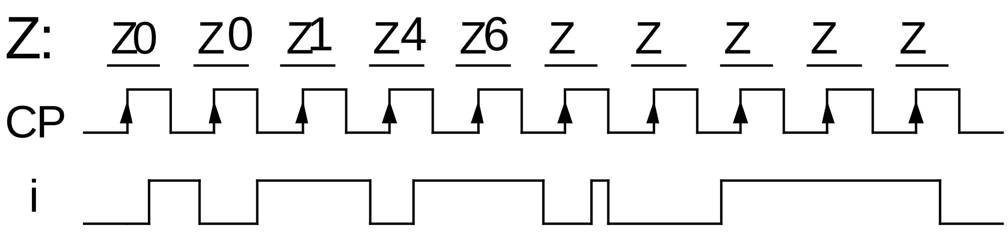
Ex 10.5





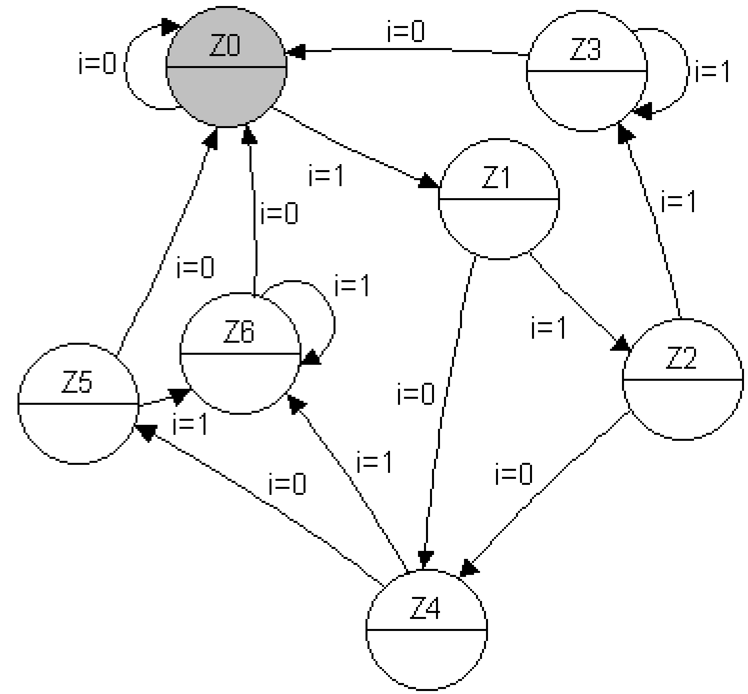
Ex 10.5

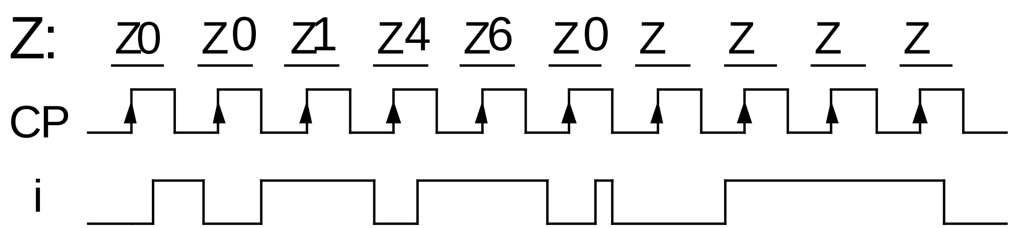




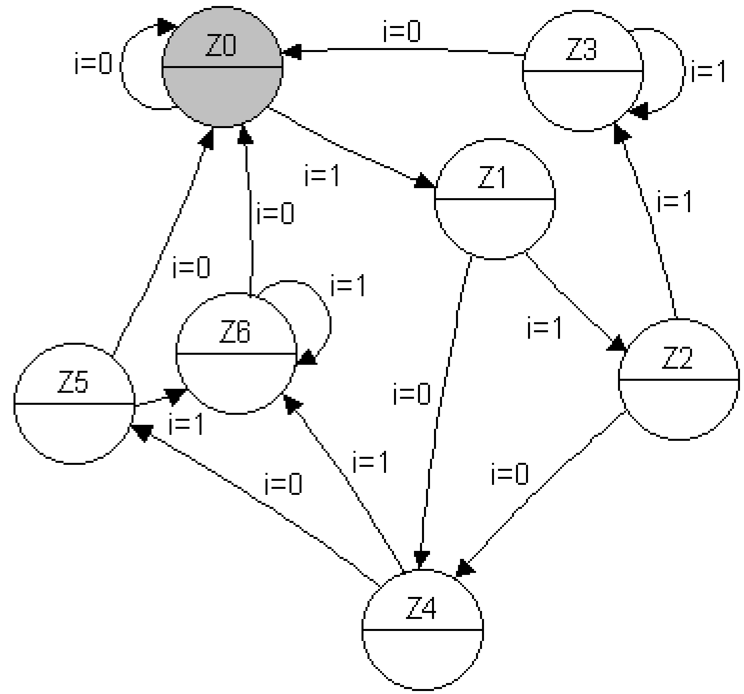
William Sandqvist william@kth.se

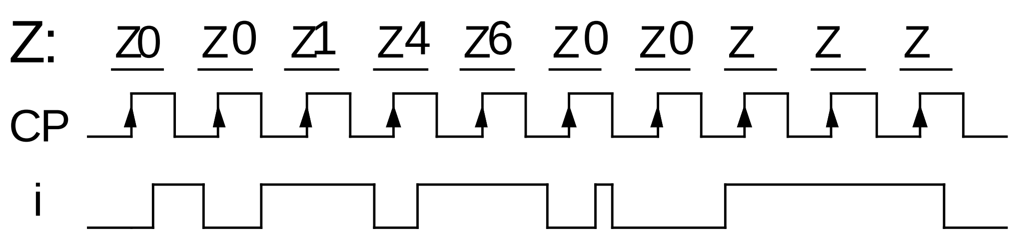
Ex 10.5



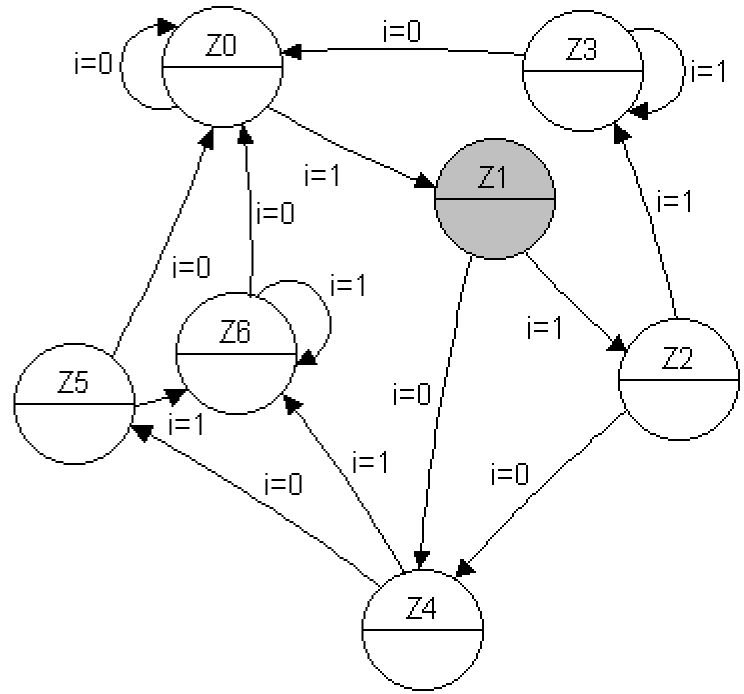


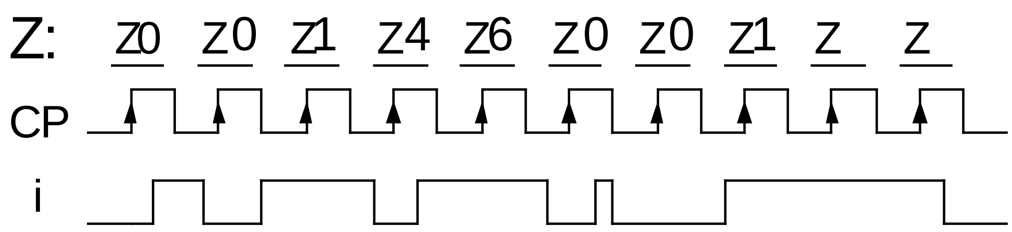
Ex 10.5


Z0
Z1
Z2
Z3
Z4
Z5
Z6
i=0
i=0
i=1
i=1
i=0
i=0
i=1
i=1
i=1
i=0
i=1
i=0
i=1
i=0
i=0

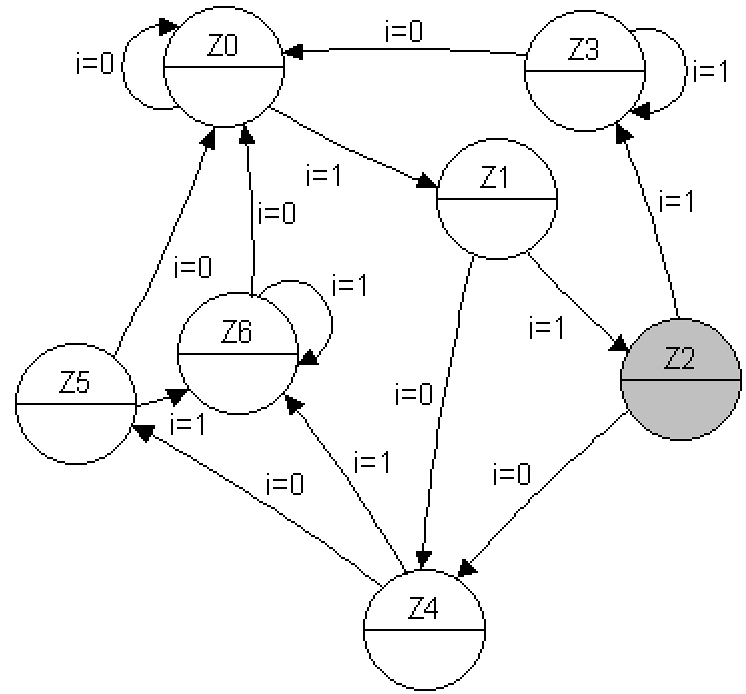

Z: Z0 Z0 Z1 Z4 Z6 Z0 Z0 Z Z Z
CP
i

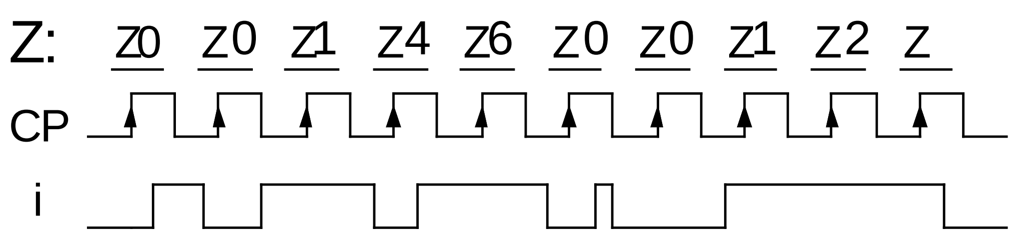
Ex 10.5





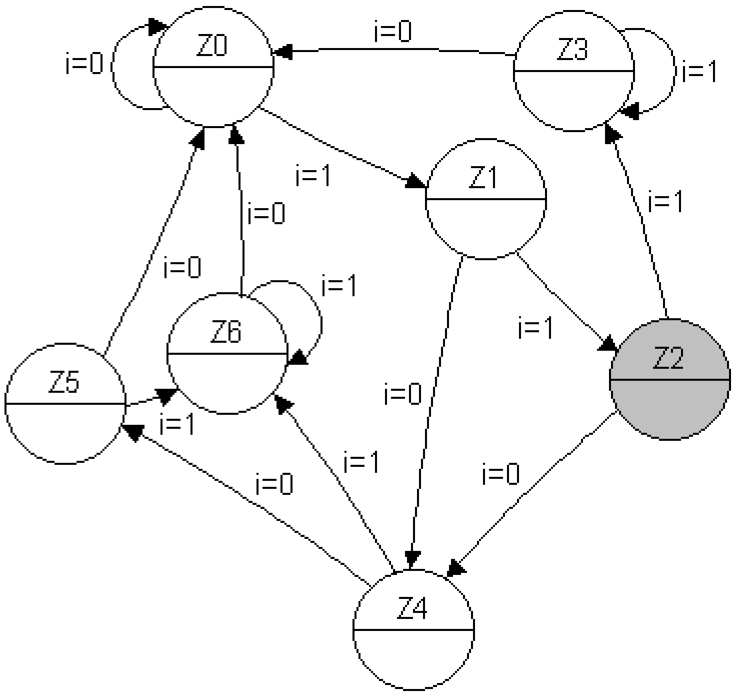
Ex 10.5

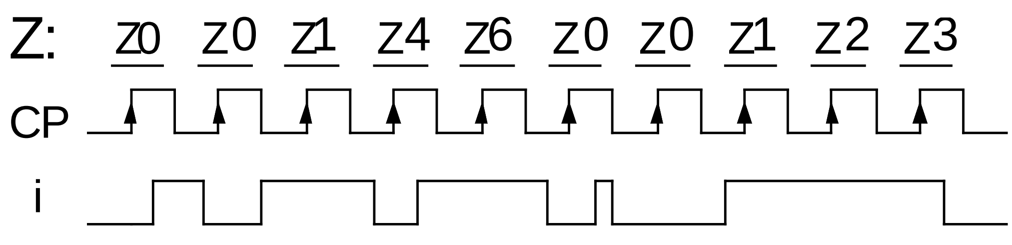




William Sandqvist william@kth.se

Ex 10.5

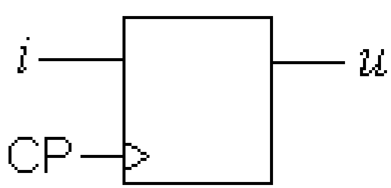


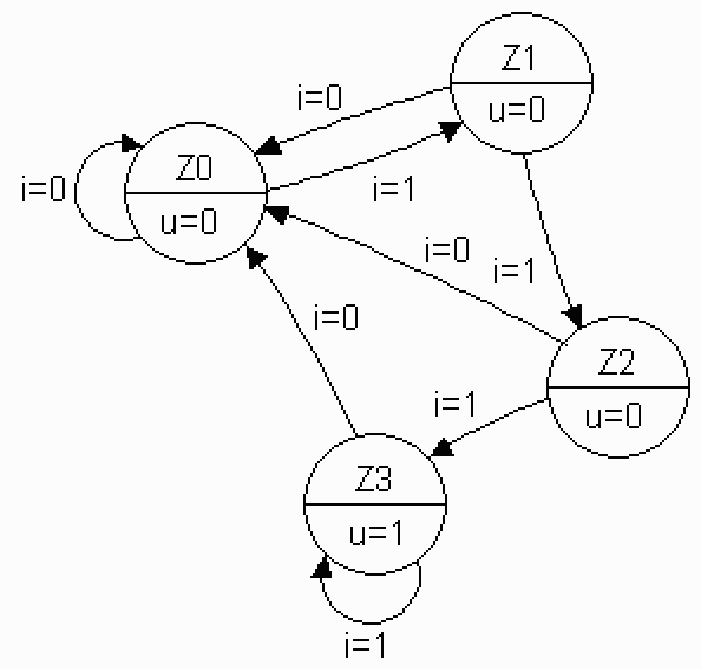


Ex 10.6

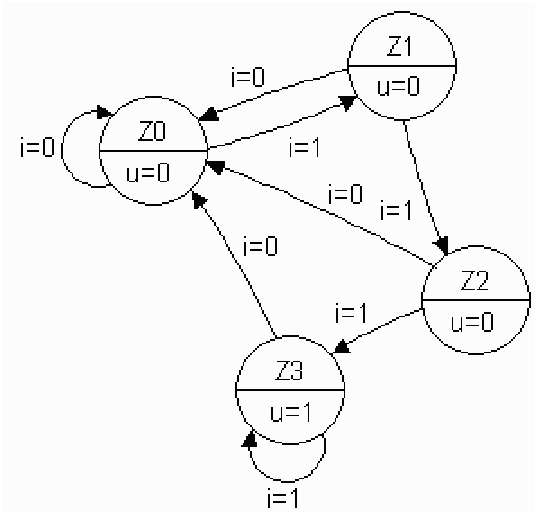
Construct a Moore machine which requires that the input signal is equal to one (i = 1) during three successive clock pulse interval, for the output to be one (u = 1).
As soon as the input signal becomes zero (i = 0) during a clock pulse interval, the circuit output should return to zero (u = 0). See the state diagram.
Choose Gray code for state encoding. (Z0=00, Z1=01, Z2=11, Z3=10).
Use D-flip-flops and AND-OR gates.

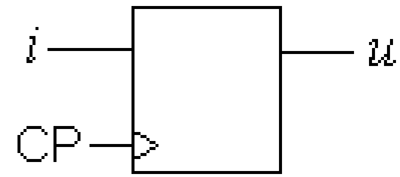
(This is a safety circuit to prevent "false alarms”)





Ex 10.6





From statediagram to coded state table:

$q_1^+ q_0^+ (q_1 q_0 i)$

$q_1 q_0$ \ i	0	1
Z0: 00	00	01
Z1: 01	00	11
Z2: 11	00	10
Z3: 10	00	10

$q_1^+ = i\, q_0 + i\, q_1$

$q_1 q_0$ \ i	0	1
00	0	0
01	0	1
11	0	1
10	0	1

$i\, q_0$

$i\, q_1$

$q_0^+ = i\, \overline{q_1}$

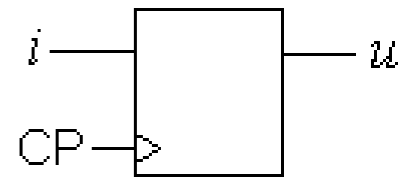
$q_1 q_0$ \ i	0	1
00	0	1
01	0	1
11	0	0
10	0	0

$i\, \overline{q_1}$

$u = q_1 \overline{q_0}$

$q_1 q_0$	
00	0
01	0
11	0
10	1

10.6



$q_1^+ = i\,q_0 + i\,q_1$

q_1q_0 \ i	0	1
00	0	0
01	0	1
11	0	1
10	0	1

$i\,q_0$

$i\,q_1$

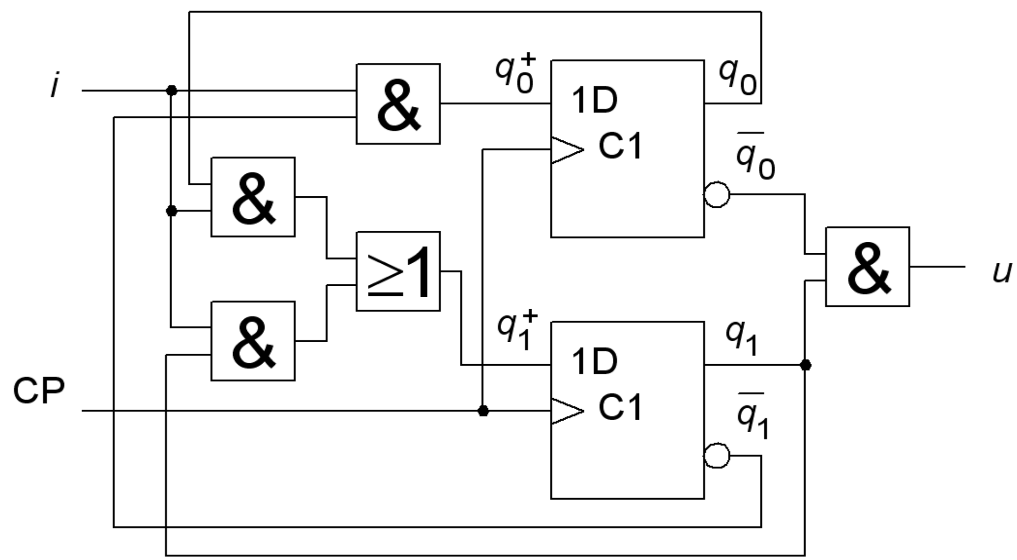
$q_0^+ = i\,\overline{q_1}$

q_1q_0 \ i	0	1
00	0	1
01	0	1
11	0	0
10	0	0

$i\,\overline{q_1}$

$u = q_1\overline{q_0}$

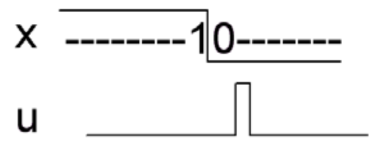
q_1q_0	
00	0
01	0
11	0
10	1

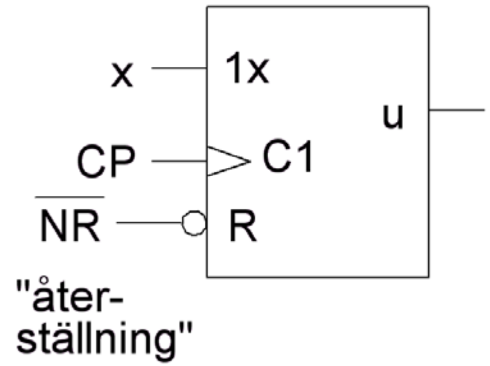


William Sandqvist william@kth.se

Ex 10.7

Construct a sequential circuits that detects when the input signal *x* has a transition 1→0 and then has the output *u* = 1 in the following clock-pulse interval, det nästföljande klockpulsintervallet and then being 0 for the rest of the sequence.





The circuit should be able to "reset" with an asynchronous reset pulse (NR active low), so that it monitors the input signal again.

a) Draw state a diagram of a Moore machine type for the sequence network.

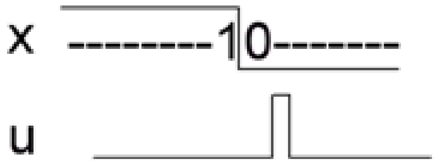
b) Derive the Boolean expressions for the next state and the output for three different state encoding:

1) ”Binarycode”
2) ”Graycode”
3) ”One hot” code

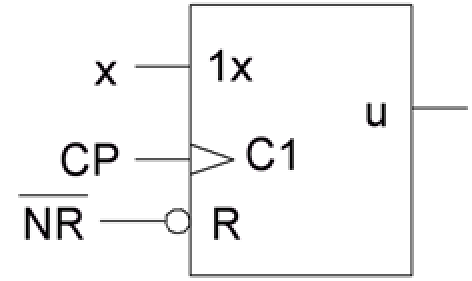
c) Show how the reset signal is connected to NR D-flip-flops PRE and CLR inputs.

William Sandqvist william@kth.se

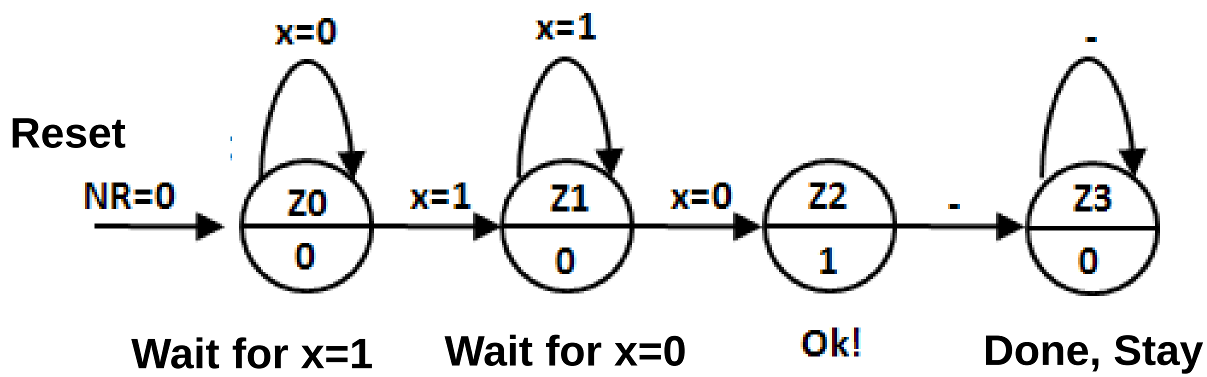
10.7



x
--------1 0-------
u

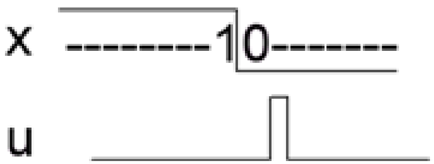


x
1x
u
CP
C1
NR
R

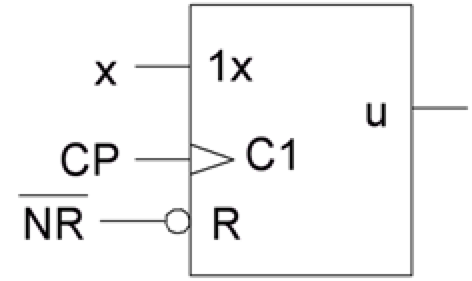


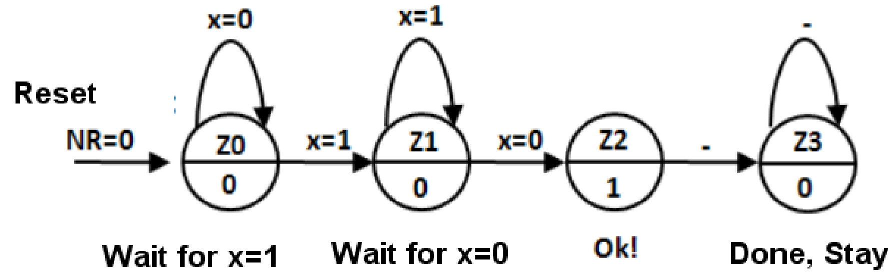
Reset
NR=0
x=0
x=1
x=1
x=0
-
-
Z0
0
Z1
0
Z2
1
Z3
0
Wait for x=1
Wait for x=0
Ok!
Done, Stay

10.7



State code: Binary:





Bin

$q_1^+ q_0^+ (q_1 q_0 x)$

$q_1 q_0$ \ x	0	1
Z0: 00	00	01
Z1: 01	10	01
Z3: 11	11	11
Z2: 10	11	11

$q_1^+ = q_0 \overline{x} + q_1$

$q_1 q_0$ \ x	0	1
00	0	0
$q_0\overline{x}$ 01	1	0
q_1 11	1	1
10	1	1

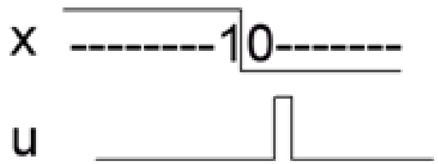
$q_0^+ = q_1 + x$

$q_1 q_0$ \ x	0	1
00	0	1
x 01	0	1
q_1 11	1	1
10	1	1

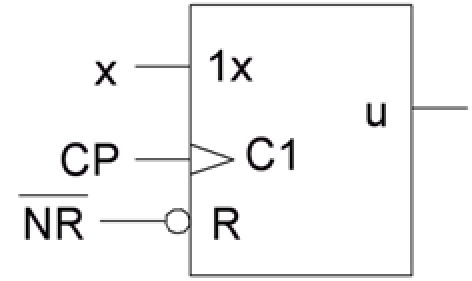
$U = q_1 \overline{q_0}$

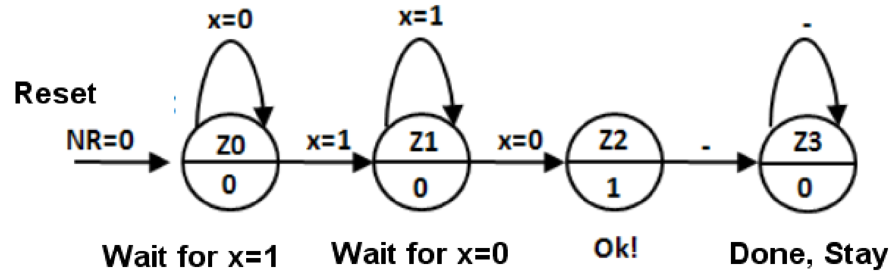
$q_1 q_0$	
00	0
01	0
$q_1\overline{q_0}$ 11	0
10	1

10.7



Statecode Gray:





Gray $q_1^+ q_0^+ (q_1 q_0 x)$

$q_1 q_0$ \ x	0	1
Z0: 00	00	01
Z1: 01	11	01
Z2: 11	10	10
Z3: 10	10	10

$q_1^+ = q_0 \overline{x} + q_1$

$q_1 q_0$ \ x	0	1
00	0	0
01	1	0
11	1	1
10	1	1

$q_0 \overline{x}$ q_1

$q_0^+ = \overline{q}_1 q_0 + \overline{q}_1 x$

$q_1 q_0$ \ x	0	1
00	0	1
01	1	1
11	0	0
10	0	0

$\overline{q}_1 x$ $\overline{q}_1 q_0$

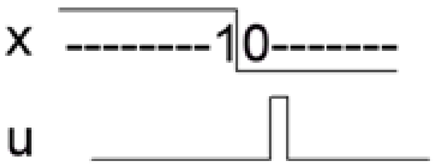
$U = q_1 q_0$

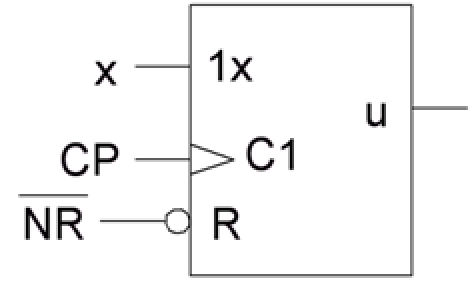
$q_1 q_0$	
00	0
01	0
11	1
10	0

$q_1 q_0$

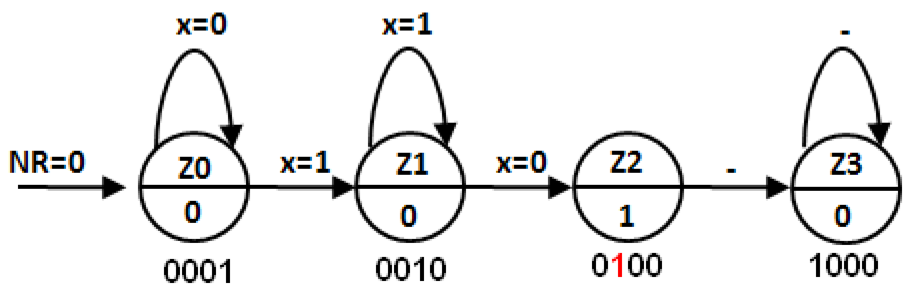
This time the binary state code seems to be the better one.

10.7





State code: One Hot:



$q_3^+ q_2^+ q_1^+ q_0^+ (q_3\, q_2\, q_1\, q_0\, x)$

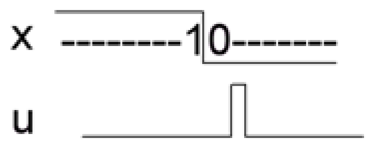
$\overline{x}$

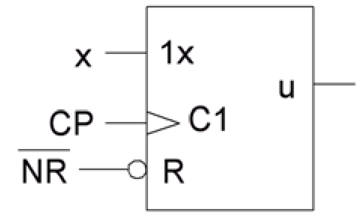
$q_3 q_2 \backslash q_1 q_0$	00	01	11	10
00	0 -	1 0001	3 -	2 0100
01	4 1000	5 -	7 -	6 -
11	12 -	13 -	15 -	14 -
10	8 1000	9 -	11 -	10 -

x

$x_3 x_2 \backslash x_1 x_0$	00	01	11	10
00	16 -	17 0010	19 -	18 0010
01	20 1000	21 -	23 -	22 -
11	28 -	29 -	31 -	30 -
10	24 1000	25 -	27 -	26 -

10.7





State code: One Hot:

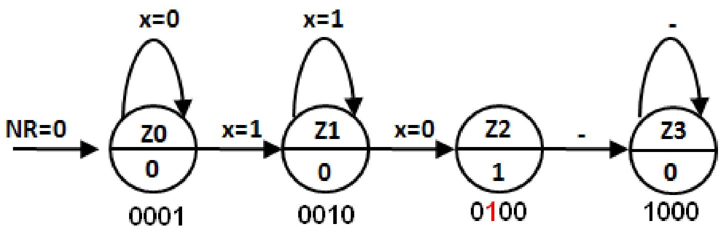
$q_3^+q_2^+q_1^+q_0^+(q_3\,q_2\,q_1\,q_0\,x)$

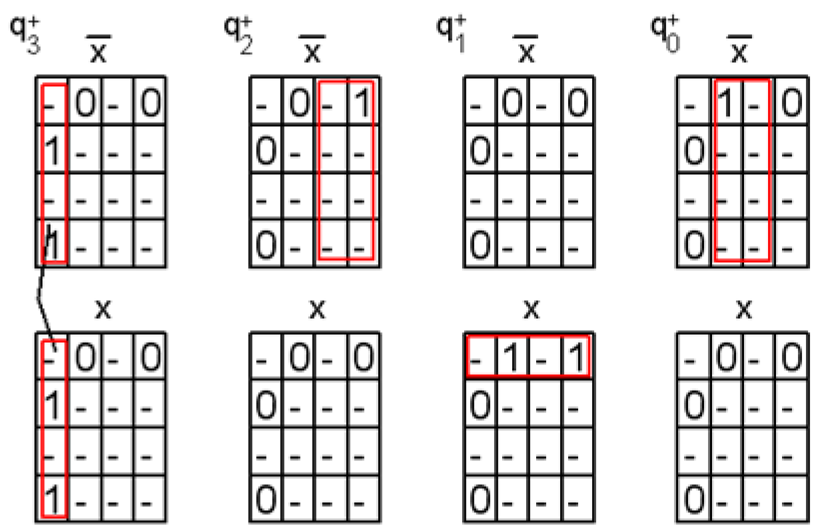
$\overline{x}$

q_3q_2 \ q_1q_0	00	01	11	10
00	-	0001	-	0100
01	1000	-	-	-
11	-	-	-	-
10	1000	-	-	-

x

x_3x_2 \ x_1x_0	00	01	11	10
00	-	0010	-	0010
01	1000	-	-	-
11	-	-	-	-
10	1000	-	-	-





$q_3^+ = \overline{q}_1\overline{q}_0$

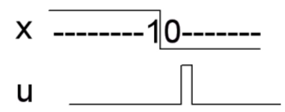
$q_2^+ = q_1\overline{x}$

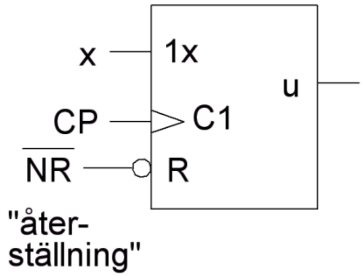
$q_1^+ = \overline{q}_3\overline{q}_2 x$

$q_0^+ = q_0\overline{x}$

$u = q_2$

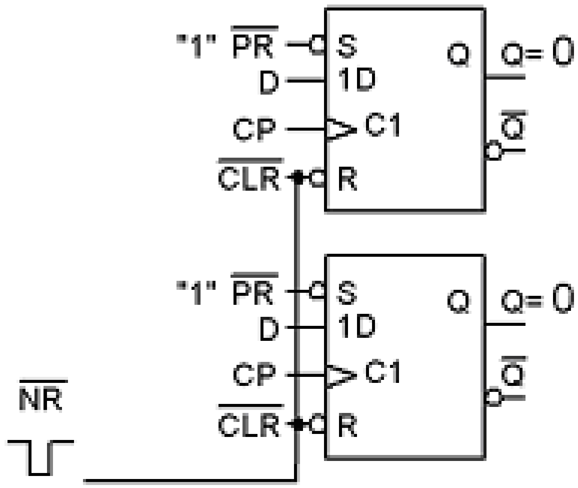
10.7

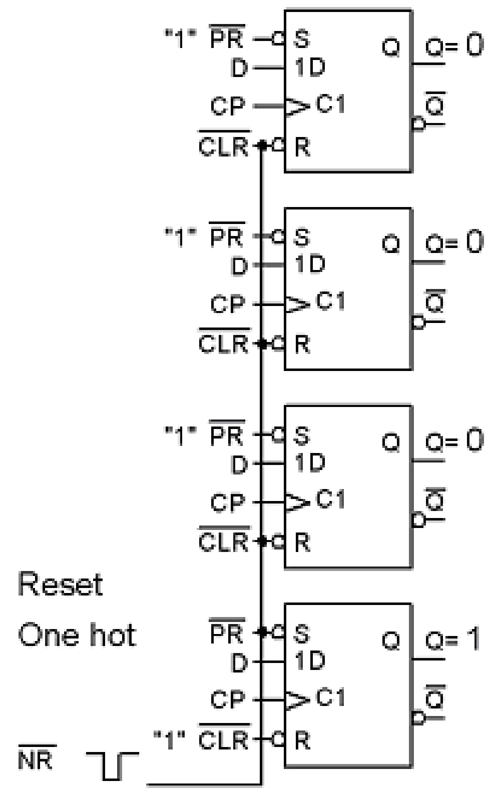




Reset signals.

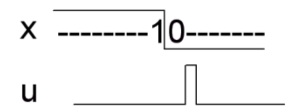
Reset Bin/Gray

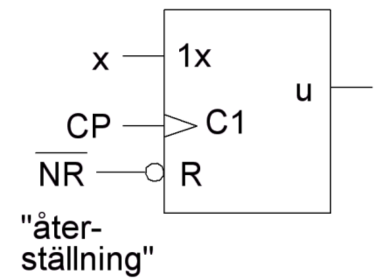




Bin / Gray networks is reseted by the flip-flop’s CLR inputs.
One Hot network is restored by setting the flip-flops to "0001" with the CLR and PR inputs.

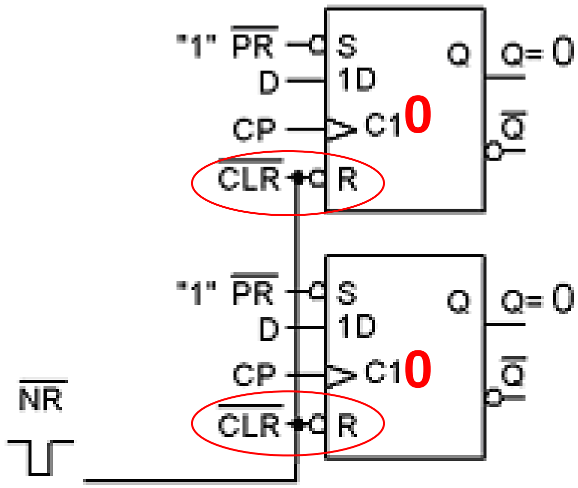
10.7

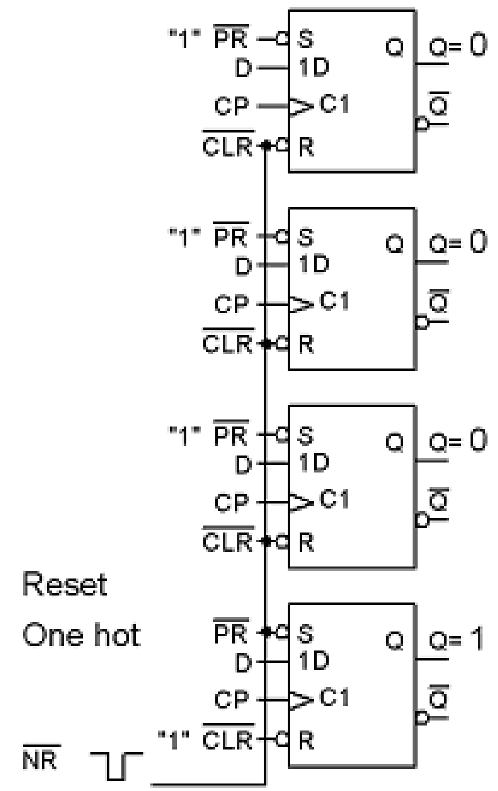




Reset signals.

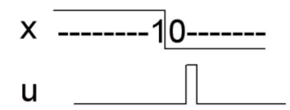
Reset Bin/Gray

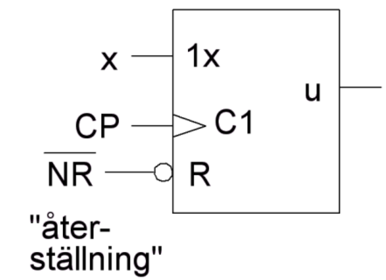




Bin / Gray networks is reseted by the flip-flop’s CLR inputs.

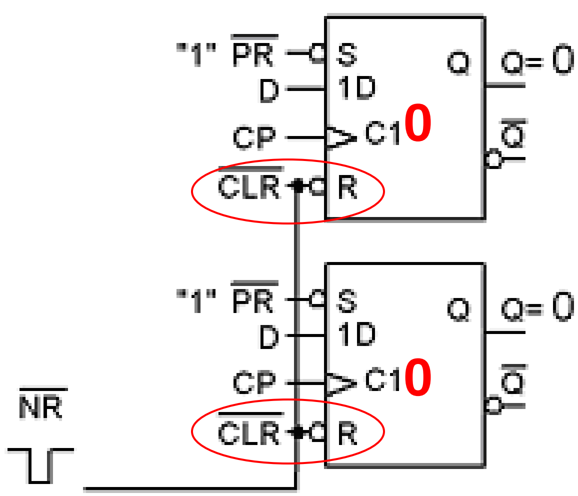
10.7

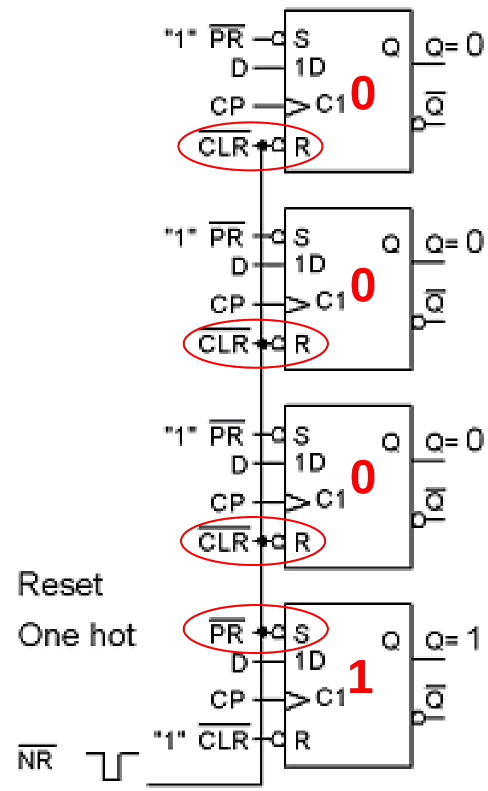




Reset signals.

Reset Bin/Gray

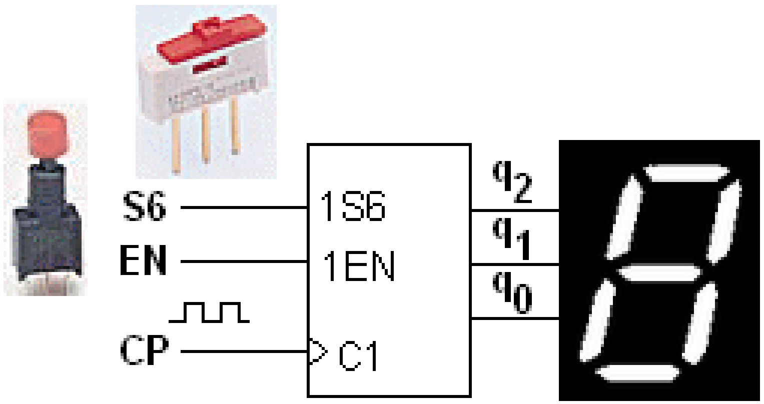


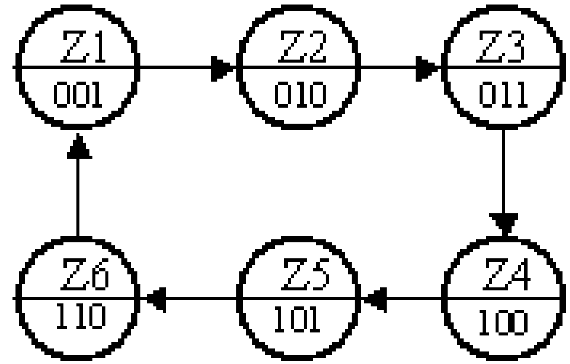


Bin / Gray networks is reseted by the flip-flop's CLR inputs.
One Hot network is restored by setting the flip-flops to "0001" with the CLR and PR inputs.

William Sandqvist william@kth.se

Ex 10.8



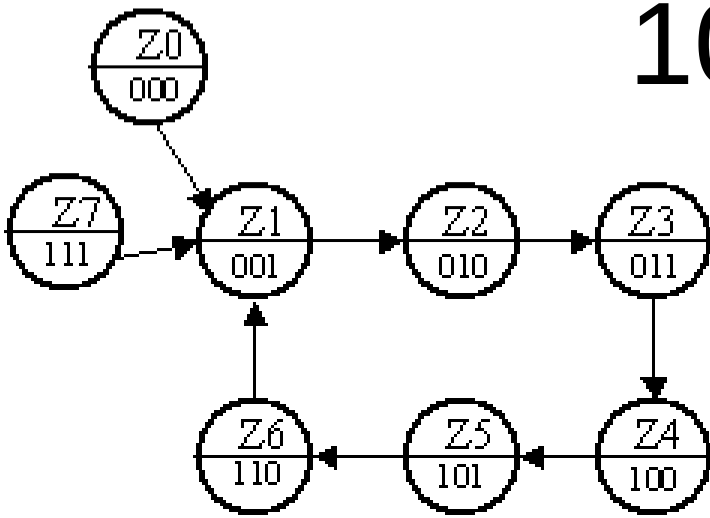


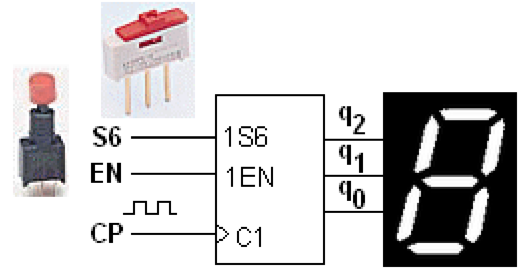
Design a counter that counts {… 1, 2, 3, 4, 5, 6, 1 …}. The counting sequence, $q_2q_1q_0$, is to be shown with a 7-segment display, as a roll of the dice.

a) State the expressions for the next state..
b) Complete the expressions with a variable EN which will ” freeze” the state when EN = 0 (unpressed button). The counter shold count for EN = 1 (pressed button).
c) Complete the expressions with a variable S6 which forces the couter to state “6” when S6 = 1 (hidden button pressed). This is the “cheat-button”. S6 takes precedence over one.

William Sandqvist william@kth.se

10.8





We let the two unused states Z0 and Z7 as a precaution go to Z1.

$q_2^+ q_1^+ q_0^+ (q_2 q_1 q_0)$

$q_2 q_1$ \ q_0	0	1
00	001	010
01	011	100
11	001	001
10	101	110

q_2^+

0	0
0	1
0	0
1	1

q_1^+

0	1
1	0
0	0
0	1

q_0^+

1	0
1	0
1	1
1	0

$$q_2^+ = q_2\overline{q}_1 + \overline{q}_2 q_1 q_0$$

$$q_1^+ = \overline{q}_1 q_0 + \overline{q}_2 q_1 \overline{q}_0$$

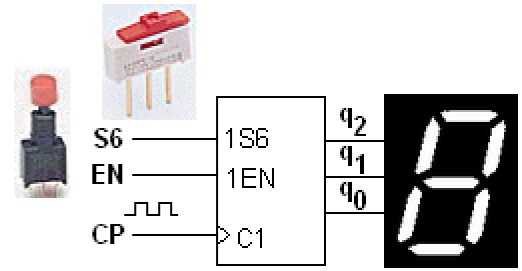
$$q_0^+ = \overline{q}_0 + q_2 q_1$$

$$q_2^+ = q_2\overline{q}_1 + \overline{q}_2 q_1 q_0$$

$$q_1^+ = \overline{q}_1 q_0 + \overline{q}_2 q_1 \overline{q}_0$$

$$q_0^+ = \overline{q}_0 + q_2 q_1$$

10.8



Equations with EN
(EN=0 → next state same

$$(q_2^+)' = EN \cdot (q_2^+) + \overline{EN} \cdot (q_2)$$

$$(q_1^+)' = EN \cdot (q_1^+) + \overline{EN} \cdot (q_1)$$

$$(q_0^+)' = EN \cdot (q_0^+) + \overline{EN} \cdot (q_0)$$

Equations with S6
(S6 = 1 → next state is 110) :

$$(q_2^+)'' = (q_2^+)' + S6$$

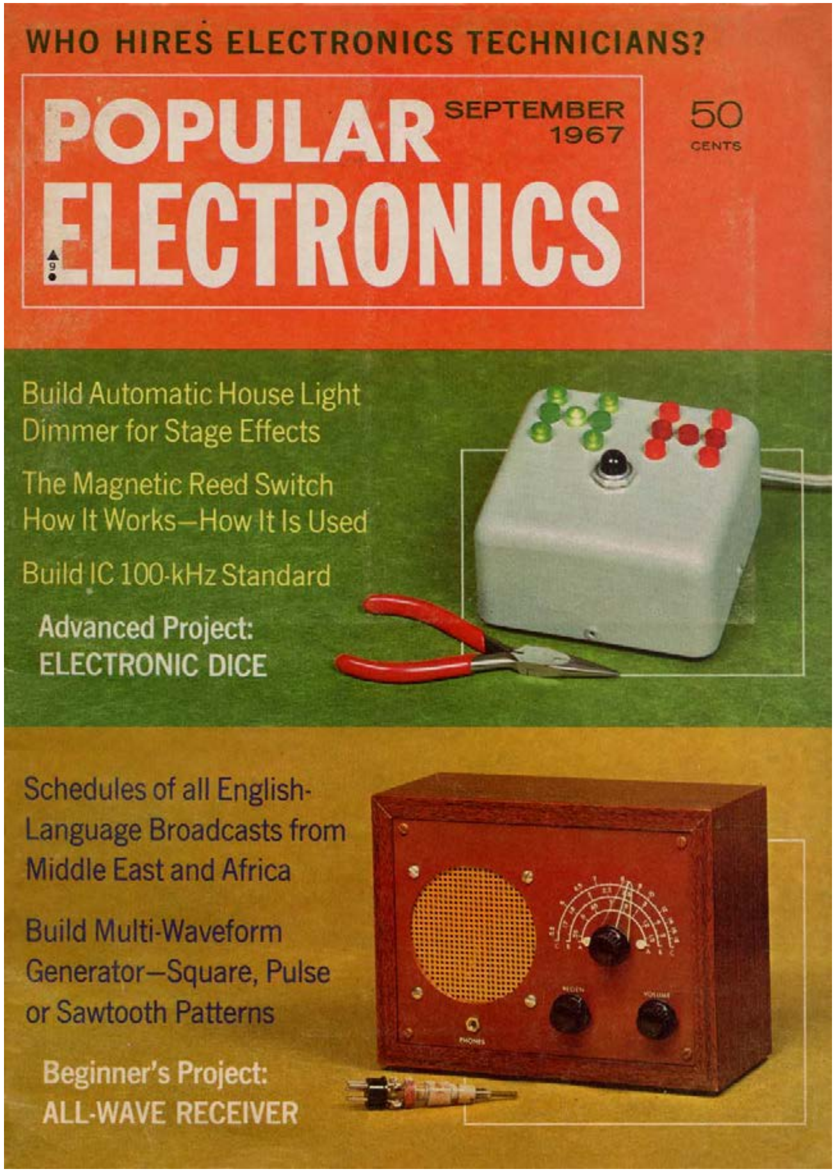
$$(q_1^+)'' = (q_1^+)' + S6$$

$$(q_0^+)'' = (q_0^+)' \cdot \overline{S6}$$

1967 was the construction of an electronic dice an ”advanced project”.

Today it is the analog technology that is advanced!

The construction of an all-band receiver was an entry-level project!



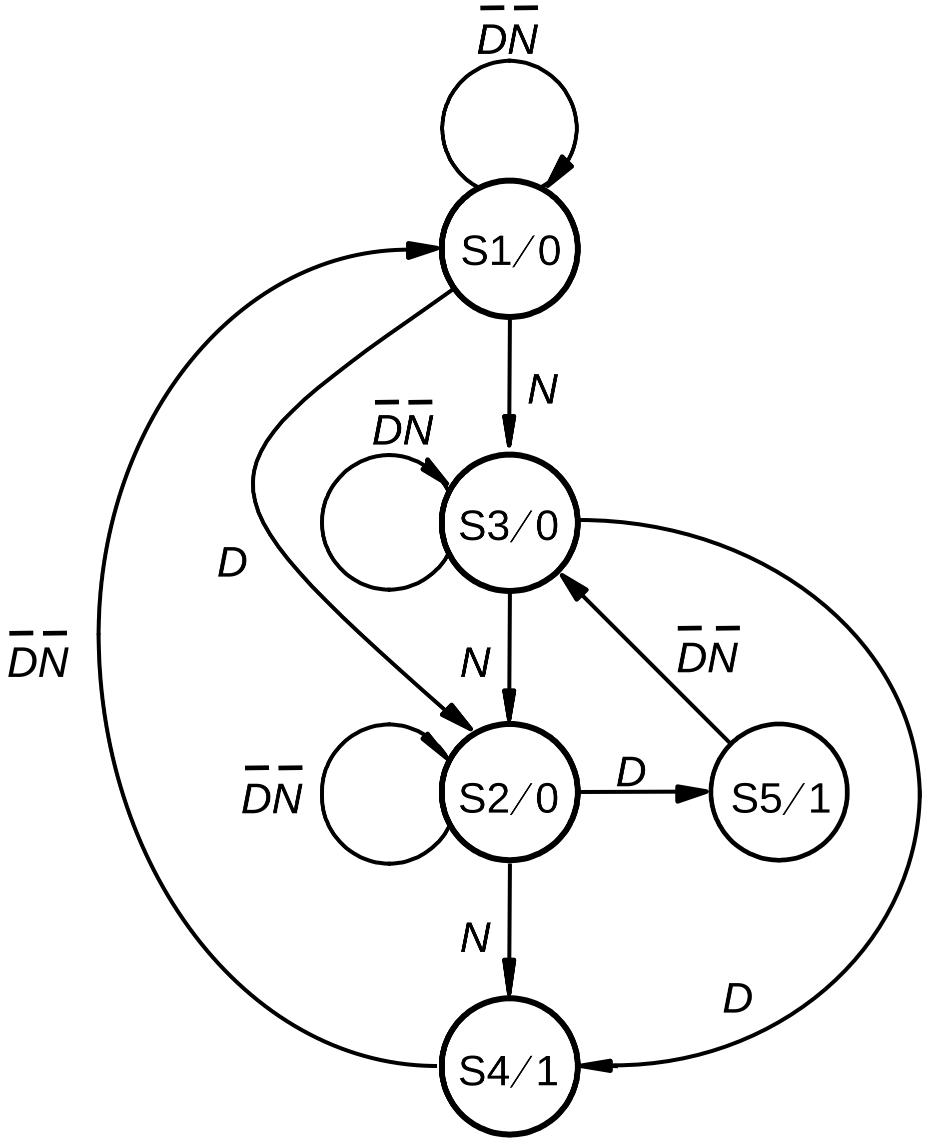


Figure 8.57. Minimized state diagram for Example 8.6.

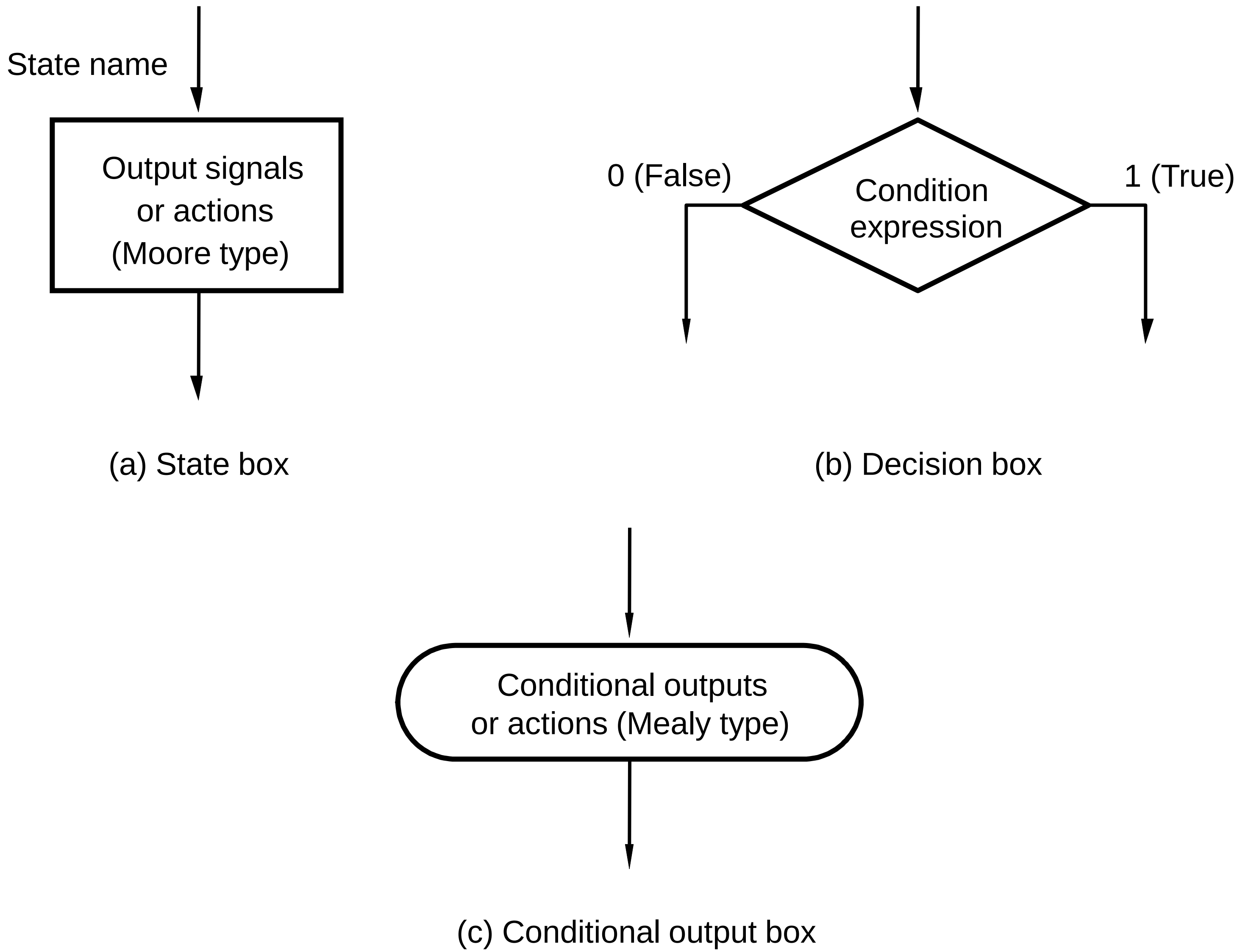
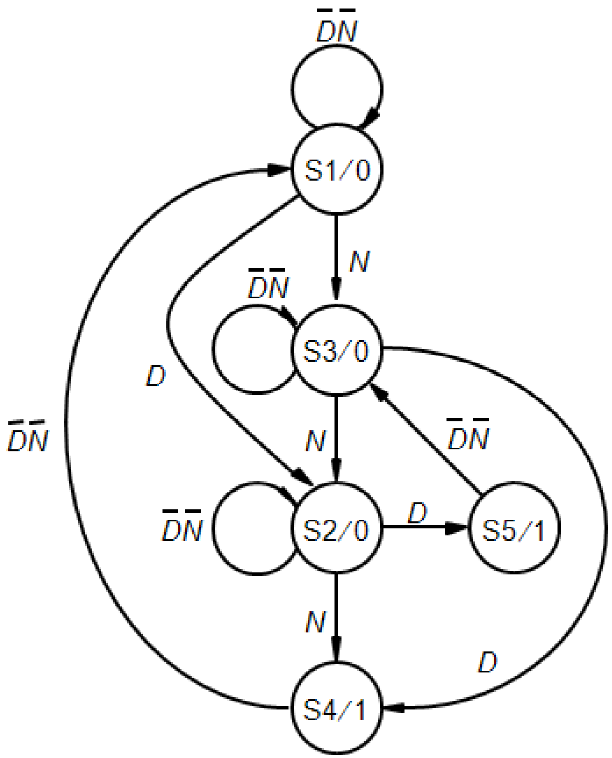
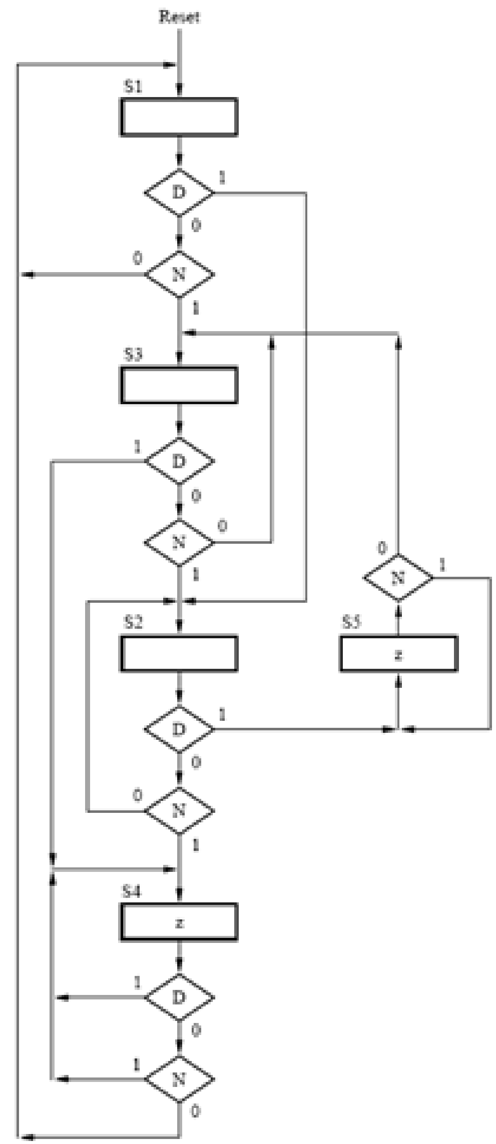


Figure 8.86. Elements used in ASM charts.

BV 8.36

Represent the FSM in Figure 8.57
in form of an ASM chart.



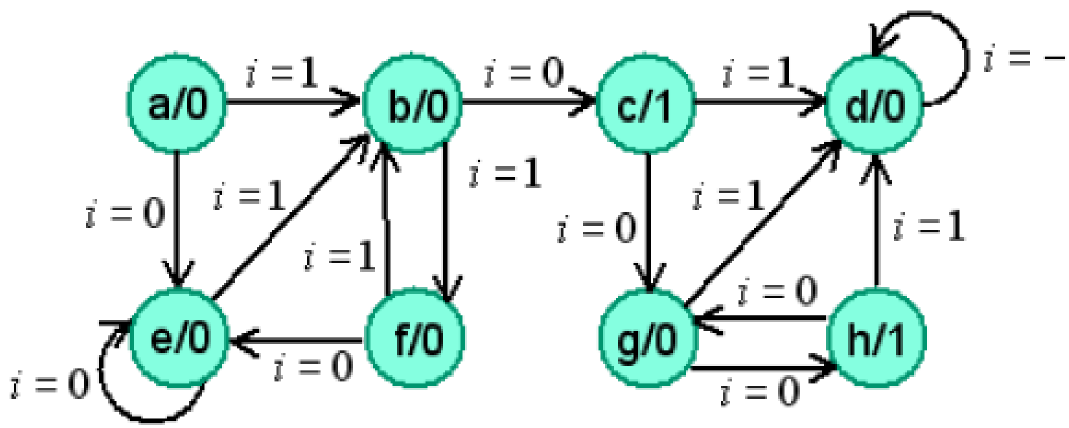


William Sandqvist william@kth.se

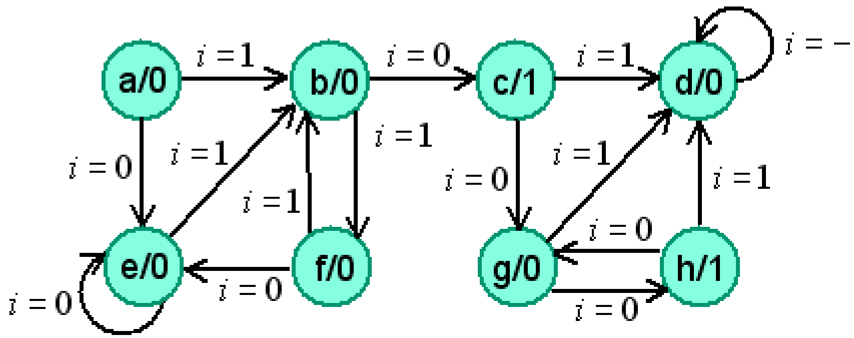
Ex 10.10 stateminimization

This statediagram is for a synchronous sequencial circuit.

- State table
- Minimize states
- Minimal statediagram



Ex 10.10 stateminimization



Write down the state table

now	next		out
$i =$	0	1	z
a	e	b	0
b	c	f	0
c	g	d	1
d	d	d	0
e	e	b	0
f	e	b	0
g	h	d	0
h	g	d	1

Two states can not be equivalent if the output is different or if subsequent state output is different.

Ex 10.10 stateminimization

now	next $i=0$	next $i=1$	out z
a	e	b	0
b	c	f	0
c	g	d	1
d	d	d	0
e	e	b	0
f	e	b	0
g	h	d	0
h	g	d	1

Groups with the same ouput:

$$P_1 = \boxed{(a,b,d,e,f,g)}(c,h)$$

Examine subsequent state:

$$a_{i=0} \rightarrow (a,b,d,\mathrm{e},f,g) \quad a_{i=1} \rightarrow (a,\mathrm{b},d,e,f,g)$$

$$\boxed{b_{i=0} \rightarrow (\mathrm{c},h) \quad b_{i=1} \rightarrow (a,b,d,e,\mathrm{f},g)}$$

$$d_{i=0} \rightarrow (a,b,\mathrm{d},e,f,g) \quad d_{i=1} \rightarrow (a,b,\mathrm{d},e,f,g)$$

$$e_{i=0} \rightarrow (a,b,d,\mathbf{e},f,g) \quad e_{i=1} \rightarrow (a,\mathbf{b},d,e,f,g)$$

$$f_{i=0} \rightarrow (a,b,d,\mathbf{e},f,g) \quad f_{i=1} \rightarrow (a,\mathbf{b},d,e,f,g)$$

$$\boxed{g_{i=0} \rightarrow (c,\mathbf{h}) \quad g_{i=1} \rightarrow (a,b,\mathbf{d},e,f,g)}$$

(b, g) forms a group of them self.

$$P_2 = (a,d,e,f)\boxed{(b,g)}(c,h)$$

Ex 10.10 stateminimization

now	next		out
$i =$	0	1	z
a	e	b	0
b	c	f	0
c	g	d	1
d	d	d	0
e	e	b	0
f	e	b	0
g	h	d	0
h	g	d	1

$$P_2 = (\boxed{a, d, e, f})(b, g)(c, h)$$

Examine subsequent state :

$$a_{i=0} \rightarrow (a, d, \mathbf{e}, f) \quad a_{i=1} \rightarrow (\mathbf{b}, g)$$

$$\boxed{d_{i=0} \rightarrow (a, \mathbf{d}, e, f) \quad d_{i=1} \rightarrow (a, \mathbf{d}, e, f)}$$

$$e_{i=0} \rightarrow (a, d, \mathbf{e}, f) \quad e_{i=1} \rightarrow (\mathbf{b}, g)$$

$$f_{i=0} \rightarrow (a, d, \mathbf{e}, f) \quad f_{i=1} \rightarrow (\mathbf{b}, g)$$

(d) Forms a group of it self.

$$P_3 = (a, e, f)(b, g)(\boxed{d})(c, h)$$

Ex 10.10 stateminimization

now	next		out
$i =$	0	1	z
a	e	b	0
b	c	f	0
c	g	d	1
d	d	d	0
e	e	b	0
f	e	b	0
g	h	d	0
h	g	d	1

$$P_3 = (a, e, f)(\boxed{b, g})(d)(c, h)$$

Examine subsequent state :

$$b_{i=0} \rightarrow (\mathbf{c}, h) \quad b_{i=1} \rightarrow (a, e, \mathbf{f})$$

$$g_{i=0} \rightarrow (c, \mathbf{h}) \quad g_{i=1} \rightarrow (\mathbf{d})$$

(b) (g) forms own groups.

$$P_4 = (a, e, f)(\boxed{b})(d)(\boxed{g})(c, h)$$

Ex 10.10 stateminimization

now	next $i=0$	next $i=1$	out z
a	e	b	0
b	c	f	0
c	g	d	1
d	d	d	0
e	e	b	0
f	e	b	0
g	h	d	0
h	g	d	1

$$P_4 = (a, e, f)(b)(d)(g)(\boxed{c, h})$$

Examine subsequent state :

$$\boxed{\begin{array}{ll} c_{i=0} \rightarrow (\mathbf{g}) & c_{i=1} \rightarrow (\mathbf{d}) \\ h_{i=0} \rightarrow (\mathbf{g}) & h_{i=1} \rightarrow (\mathbf{d}) \end{array}}$$

$P_5 = P_4$ Done!

Ex 10.10 stateminimization

$$P_4 = (a, e, f)(b)(d)(g)(c, h)]$$

now $i =$	next 0	next 1	out z
a	e	b	0
b	c	f	0
c	g	d	1
d	d	d	0
e	e	b	0
f	e	b	0
g	h	d	0
h	g	d	1

Changing names

$(a, e, f) \Rightarrow a$

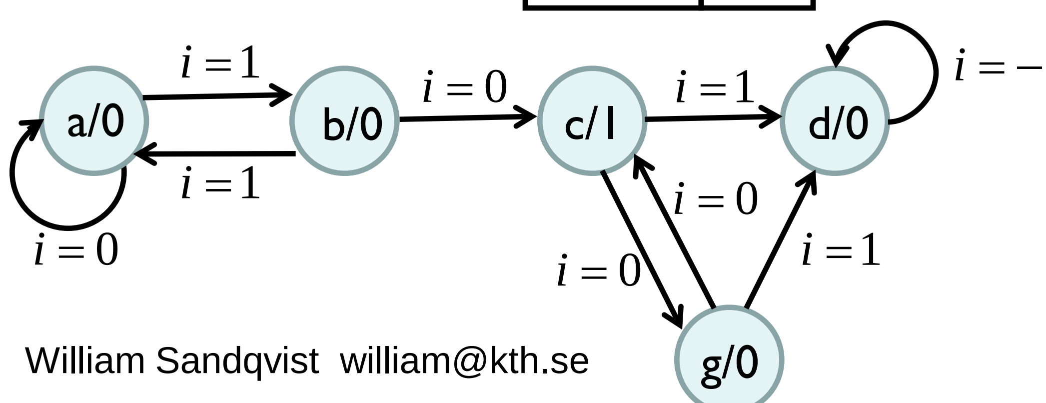
$(b) \Rightarrow b$

$(c, h) \Rightarrow c$

$(d) \Rightarrow d$

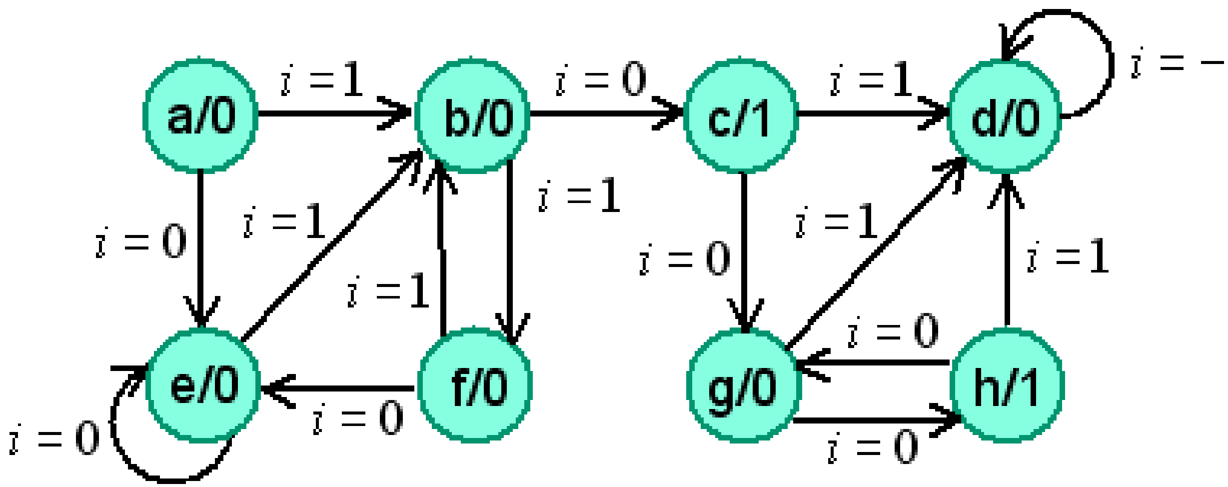
$(g) \Rightarrow g$

now $i =$	next 0	next 1	out z
a	a	b	0
b	c	a	0
c	g	d	1
d	d	d	0
g	c	d	0



Ex 10.10 stateminimization

Before:



After: