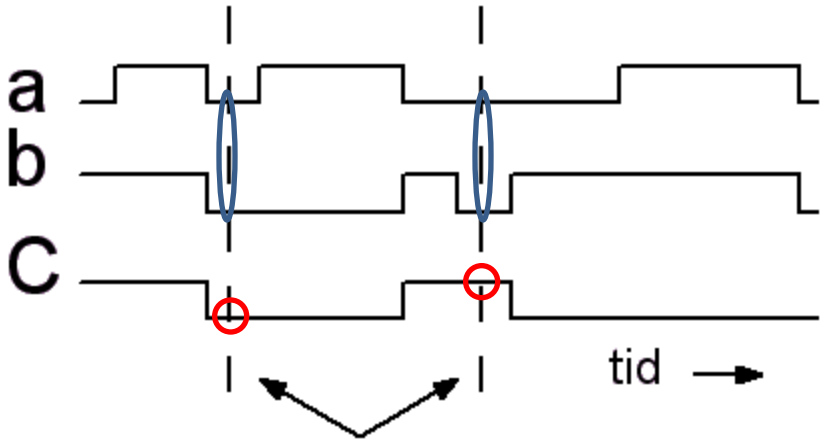
IE1205 Digital Design:

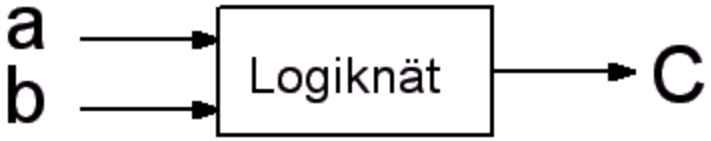
F9: Synkrona tillståndsautomater

Moore och Mealy automater

- F8 introducerade vippor och vi konstruerade räknare, skift-register etc.
- F9-F10 skall vi titta på hur generella tillståndsmaskiner kan konstrueras
- Moore och Mealy automater (uppkallade efter Edward Moore och George Mealy som studerade denna typ av kretsar)

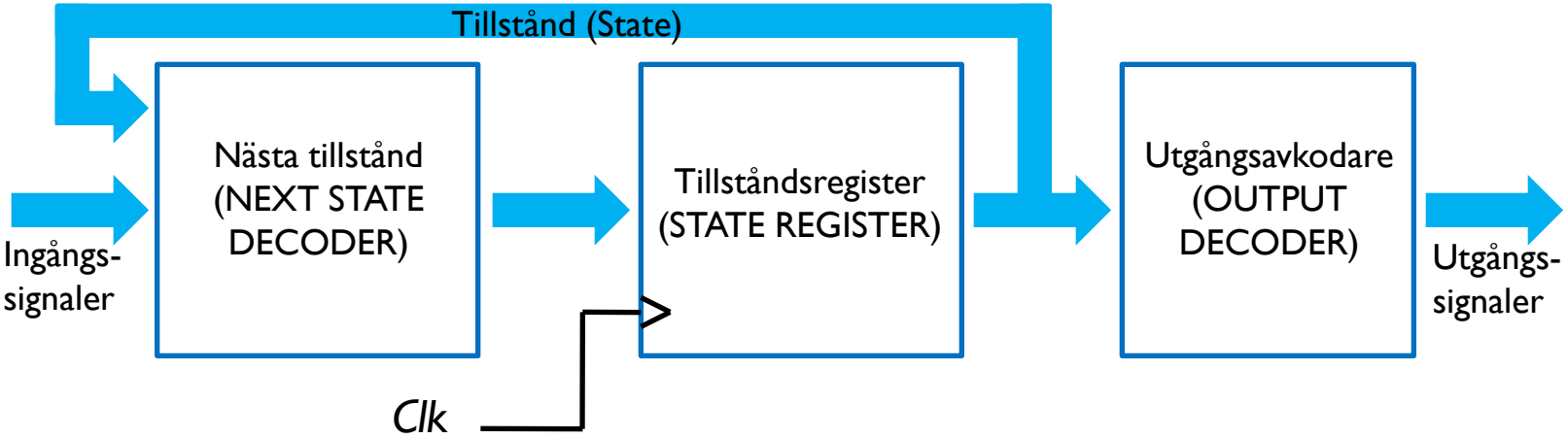
Sekvensnät





Om en och samma insignal kan ge upphov till olika utsignal, är logiknätet ett sekvensnät.
Det måste då ha ett ***inre minne*** som gör att utsignalen påverkas av både nuvarande och föregående insignaler!

Moore-automat



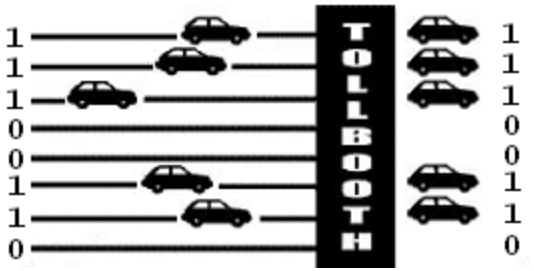
För Moore-automaten beror utsignalerna på insignalerna och det inre tillståndet. Det inre minnet är tillståndsregistret som består av D-vippor.

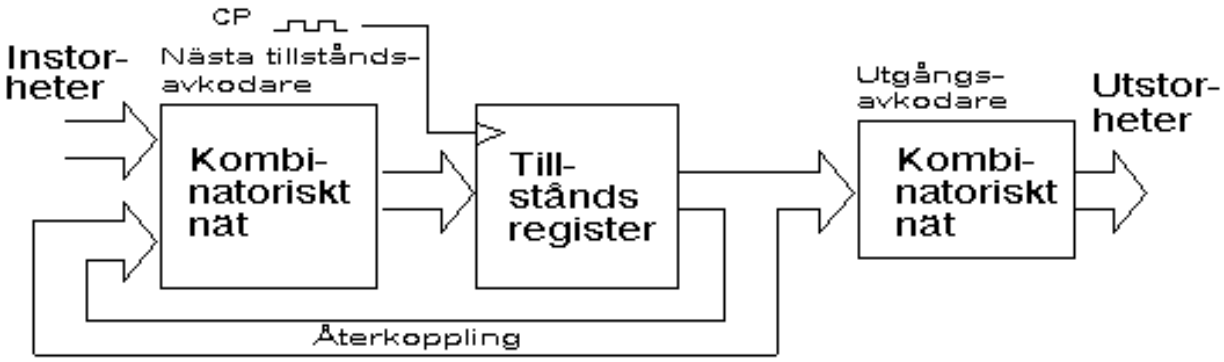
Exempel på tillståndsmaskin:
Kaffe automat: Servera kaffe när kund trycker på knapp, men bara när rätt belopp erlagts.

State 0: Fel belopp, knapp deaktiverad
State 1: Rätt belopp, knapp aktiverad

Tillståndsregistrets D-vippor

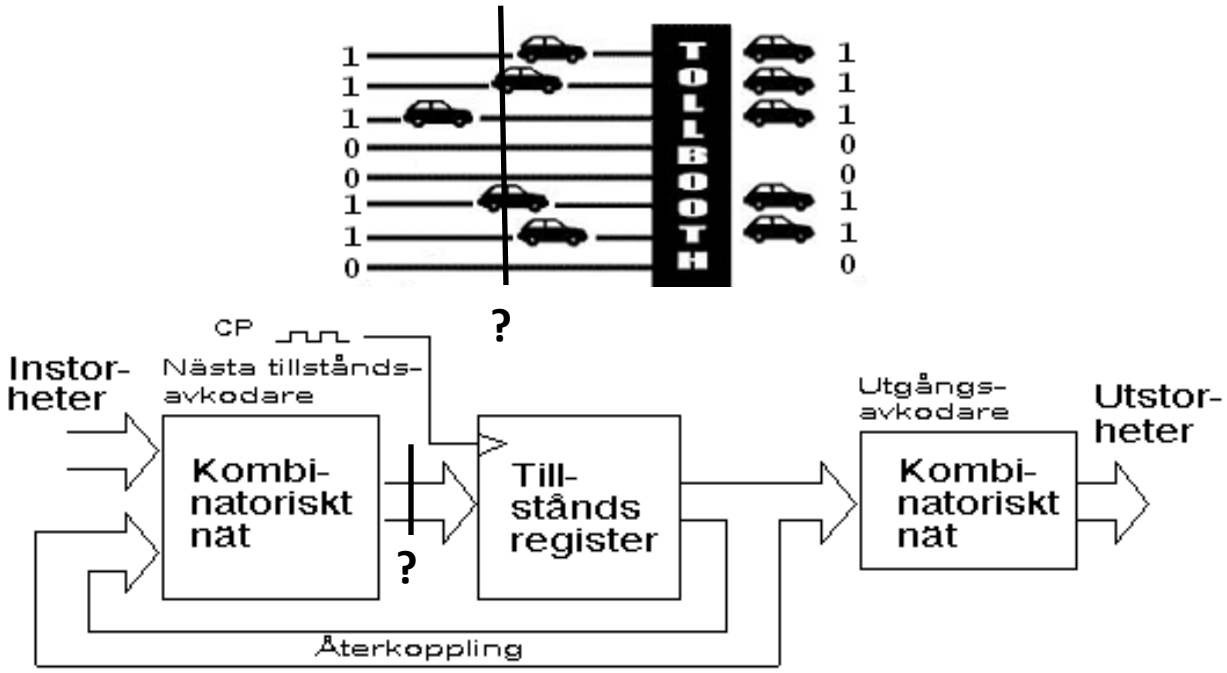
Tillståndsregistrets D-vippor bromsar upp *kapplöpningen* mellan signalerna tills värdet är stabilt. (Jämför med tullstationen).





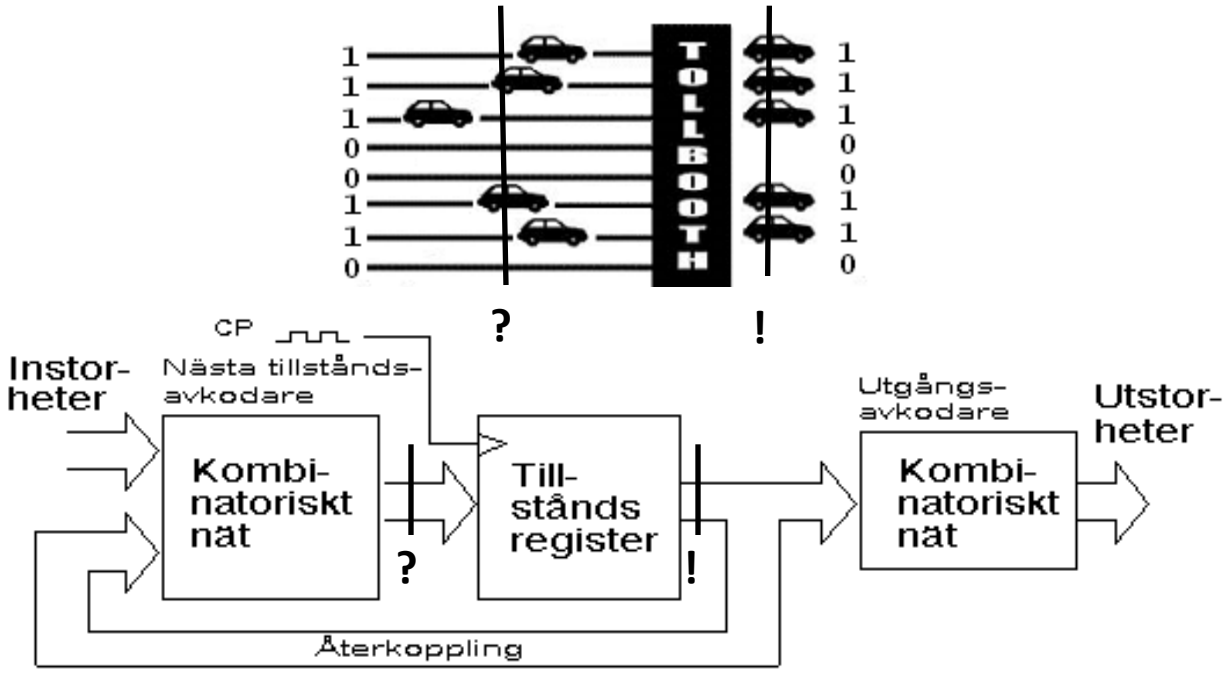
Tillståndsregistrets D-vippor

Tillståndsregistrets D-vippor bromsar upp *kapplöpningen* mellan signalerna tills värdet är stabilt. (Jämför med tullstationen).



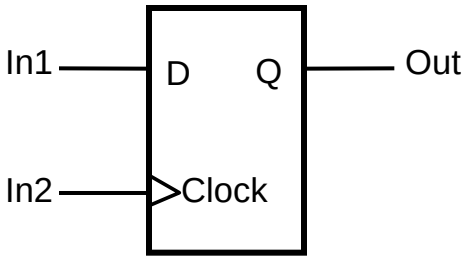
Tillståndsregistrets D-vippor

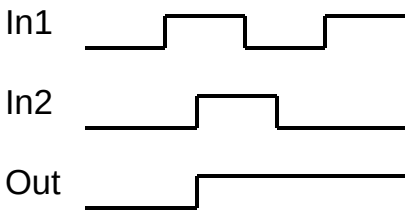
Tillståndsregistrets D-vippor bromsar upp *kapplöpningen* mellan signalerna tills värdet är stabilt. (Jämför med tullstationen).



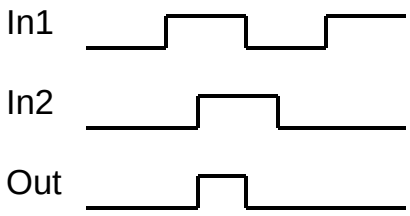
Snabbfråga Vippor

Vilket av följande tidsdiagram är giltigt för en flanktriggad D flip-flop?

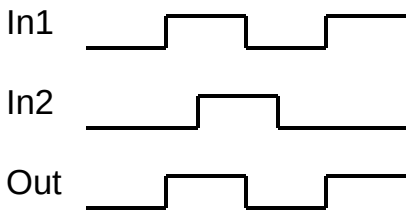




Alt: A



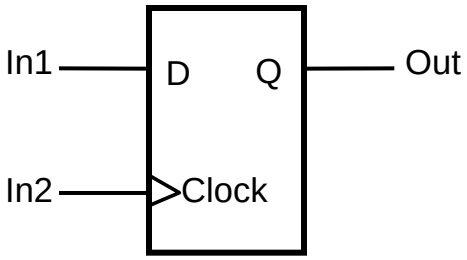
Alt: B

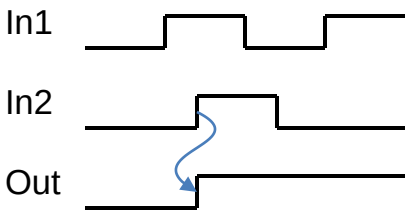


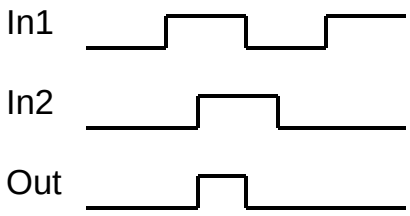
Alt: C

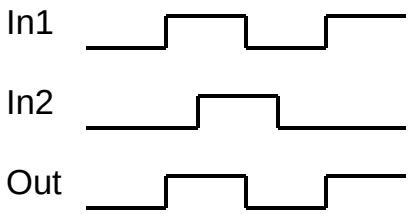
Snabbfråga Vippor

Vilket av följande tidsdiagram är giltigt för en flanktriggad D flip-flop?









Alt: A

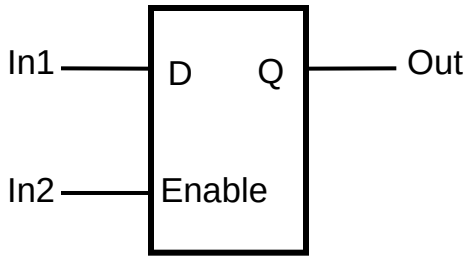
Alt: B

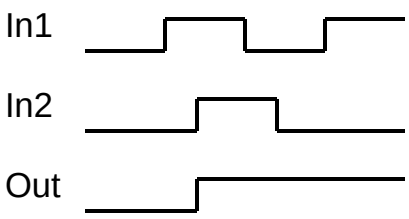
Alt: C

D kopieras till utgången vid flanken,
dvs när Clock går från 0 till 1

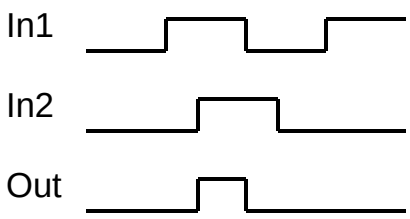
Snabbfråga Vippor

Vilket av följande tidsdiagram är giltigt för en D latch?

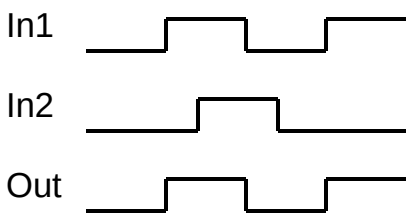




Alt: A



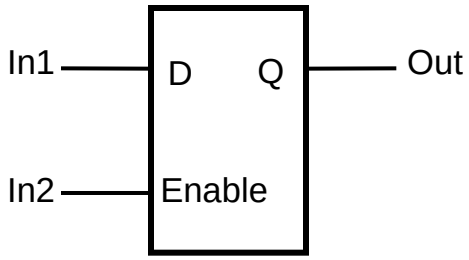
Alt: B

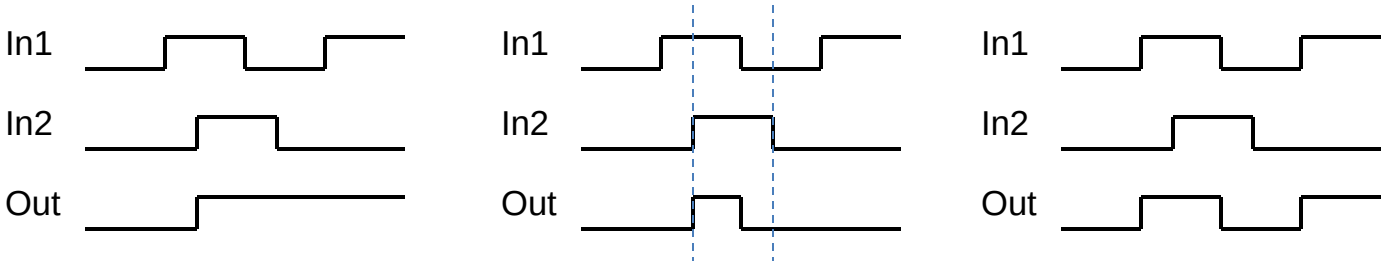


Alt: C

Snabbfråga Vippor

Vilket av följande tidsdiagram är giltigt för en D latch?





Alt: A

Alt: B

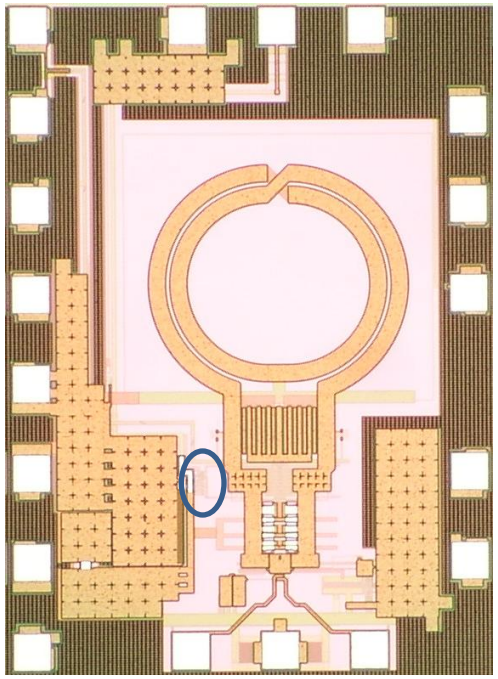
Alt: C

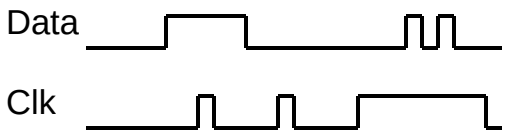
D kopplas till utgången när Enable är 1, låses när Enable är 0

Exempel shift-register

Clk

Data



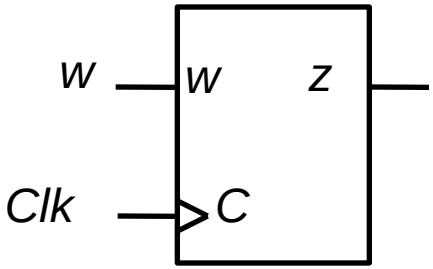


Data klockas in på positiv klockflank
För att signalera slut på data och aktivera kretsen skickas två pulser på data ingången samtidigt som klockan är hög.

Hur kan vi konstruera en krets som detekterar två efterföljande pulser?

Designexempel ”två i rad”

Sekvensdetektor. Om *w* har varit 1 under två (eller fler) klockcykler i rad skall $z = 1$.



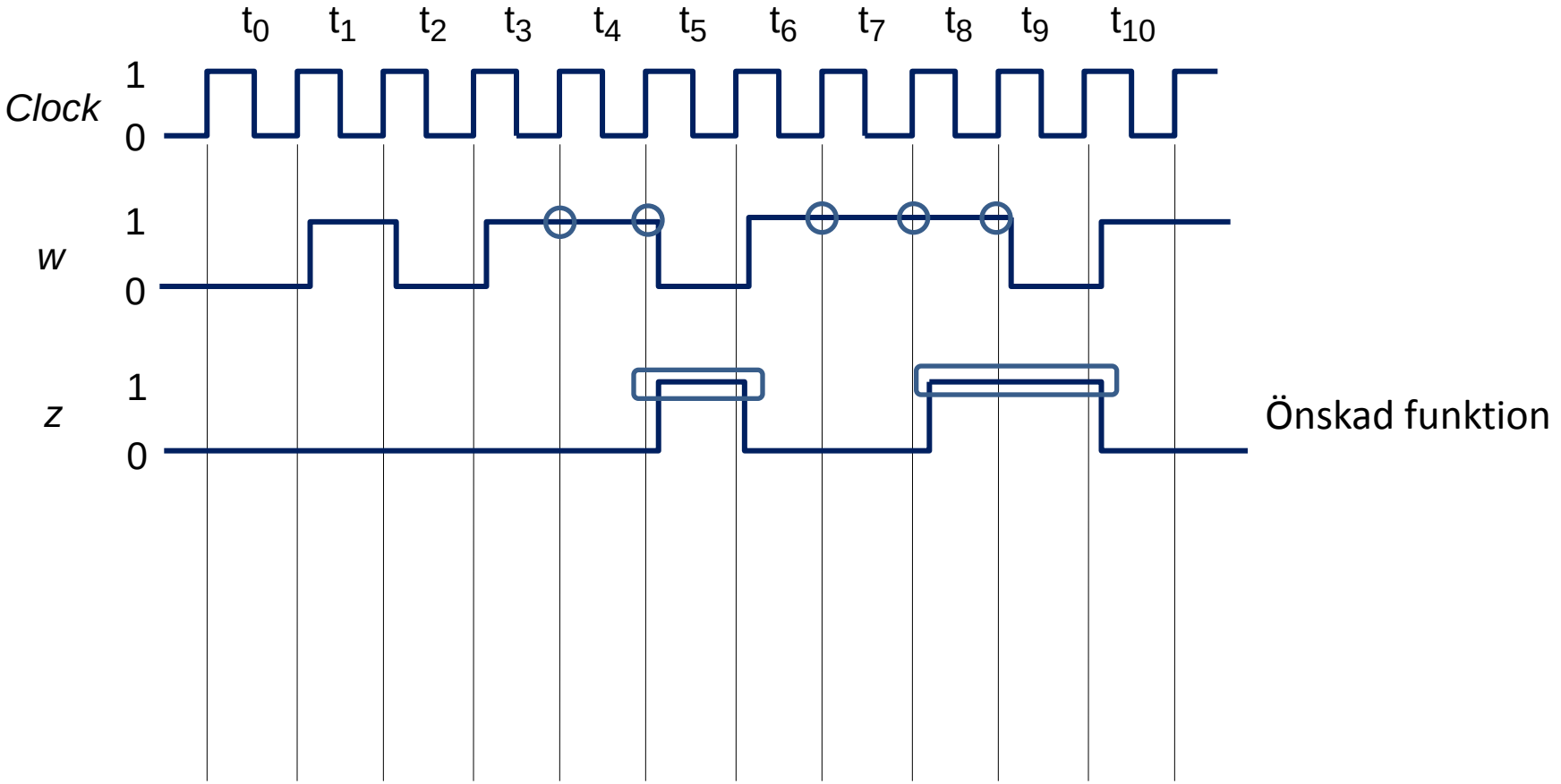
Specifikation

Sekvenskretsen har en ingång *w* och en utgång *z*

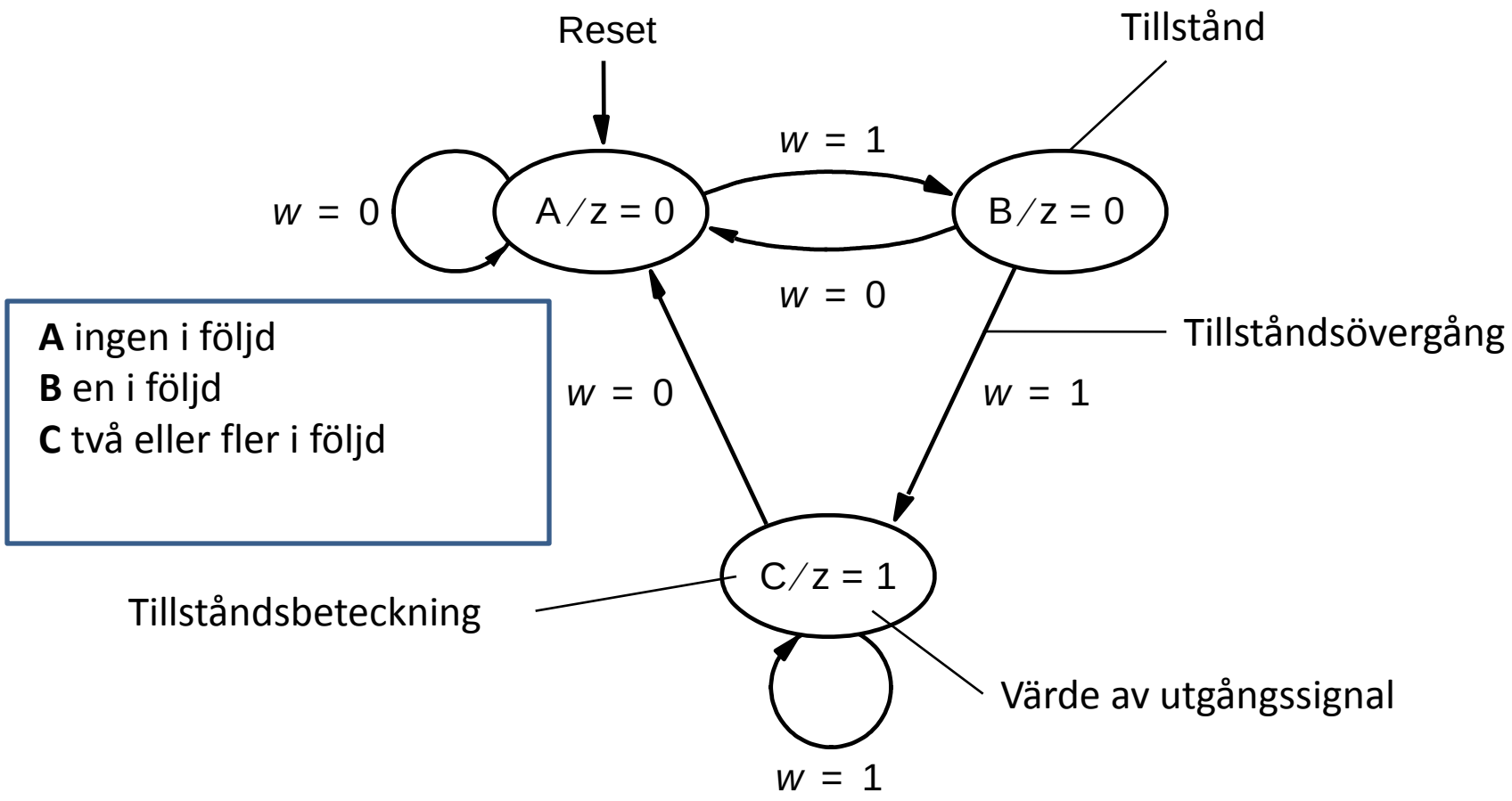
Om ingången *w* har varit 1 under nuvarande och föregående klockcykel så ska utgången *z* sättas till 1

Använd en Moore-automat med D-vippor för att realisera konstruktionen

Tidsdiagram ”två i rad”



Tillståndsdiagram ”två i rad”

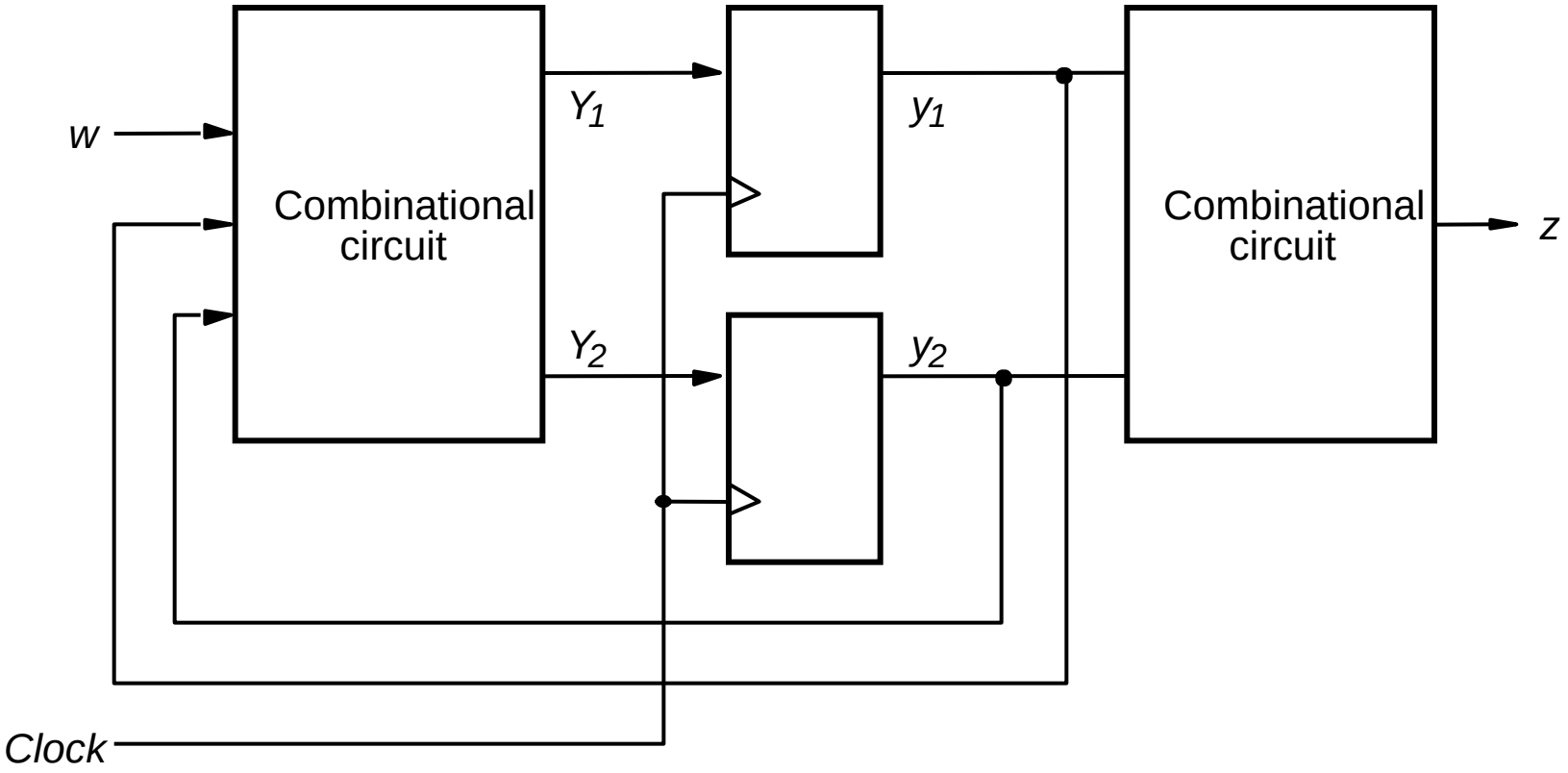


Tillståndstabell

Present state	Next state		Output
	$w = 0$	$w = 1$	z
A	A	B	0
B	A	C	0
C	A	C	1

Tre tillstånd – två vippor behövs
för att minnas tillståndets nummer!

”två i rad” som Moore-automat



Designbeslut

- Designern måste bestämma vilka vippor som ska användas

  D-, T-, eller JK-vippa
- Designern måste välja koden för varje tillstånd

Designbeslut

Denna gång givet:

- D-vippor
- Tillståndsavkodning A = 00, B = 01, C = 10
- Koden 11 ska inte förekommer.
  Vi väljer don’t care.

Kodad tillståndstabell

Present state	Next state		Output
	$w = 0$	$w = 1$	z
A	A	B	0
B	A	C	0
C	A	C	1

A = 00
B = 01
C = 10

	Present state	Next state		Output
		$w = 0$	$w = 1$	z
	$y_2 y_1$	$Y_2 Y_1$	$Y_2 Y_1$	
A	00	00	01	0
B	01	00	10	0
C	10	00	10	1
	11	*dd*	*dd*	*d*

$$Y_2 Y_1 = f(y_2 y_1 w) \qquad z = f(Y_2 Y_1)$$

Nästa tillståndsavkodare

Nästa tillståndsavkodaren består av de två logiknäten som finns som ingångsnät till de två vipporna.
För att kunna minimera logiknäten skriver man in sanningstabellerna i Karnaughdiagram.

$$Y_2 Y_1 = f(y_2 y_1 w) \quad Y_2 = f(y_2 y_1 w) \quad Y_1 = f(y_2 y_1 w)$$

$$z = f(Y_2 Y_1)$$

Kodad tillståndstabell

Present state	Next state		Output
	$w = 0$	$w = 1$	z
A	A	B	0
B	A	C	0
C	A	C	1

A = 00
B = 01
C = 10

	Present state	Next state		Output
		$w = 0$	$w = 1$	z
	$y_2 y_1$	$Y_2 Y_1$	$Y_2 Y_1$	
A	00	00	01	0
B	01	00	10	0
C	10	00	10	1
	11	*dd*	*dd*	*d*

$$Y_1 = f(y_2 y_1 w)$$

Kodad tillståndstabell

Present state	Next state		Output
	$w = 0$	$w = 1$	z
A	A	B	0
B	A	C	0
C	A	C	1

A = 00
B = 01
C = 10

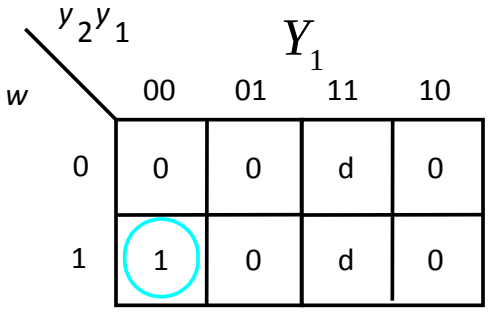
	Present state	Next state		Output
		$w = 0$	$w = 1$	z
	$y_2 y_1$	$Y_2 Y_1$	$Y_2 Y_1$	
A	00	00	01	0
B	01	00	10	0
C	10	00	10	1
	11	*dd*	*dd*	*d*

$$Y_2 = f(y_2 y_1 w)$$

Från kodad tillståndstabell till Karnaughdiagram

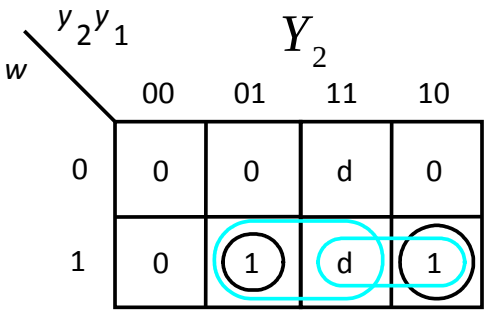
$$Y_2Y_1 = f(y_2y_1w) \quad Y_2 = f(y_2y_1w) \quad Y_1 = f(y_2y_1w)$$

Present state y_2y_1	Next state $w=0$ Y_2Y_1	Next state $w=1$ Y_2Y_1	Output z
00	00	01	0
01	00	10	0
10	00	10	1
11	*dd*	*dd*	*d*



$$Y_1 = w\bar{y}_1\bar{y}_2$$

Present state y_2y_1	Next state $w=0$ Y_2Y_1	Next state $w=1$ Y_2Y_1	Output z
00	00	01	0
01	00	10	0
10	00	10	1
11	*dd*	*dd*	*d*



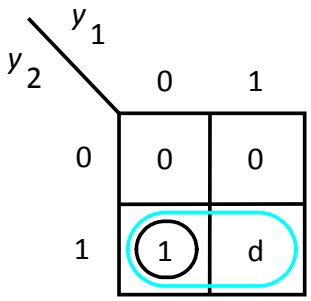
$$Y_2 = wy_1 + wy_2 =$$

$$= w(y_1 + y_2)$$

Utgångsavkodaren

$$z = f(Y_2 Y_1)$$

Present state $y_2 y_1$	Next state $w = 0$ $Y_2 Y_1$	Next state $w = 1$ $Y_2 Y_1$	Output z
00	00	01	0
01	00	10	0
10	00	10	1
11	*dd*	*dd*	*d*



$$z = y_2$$

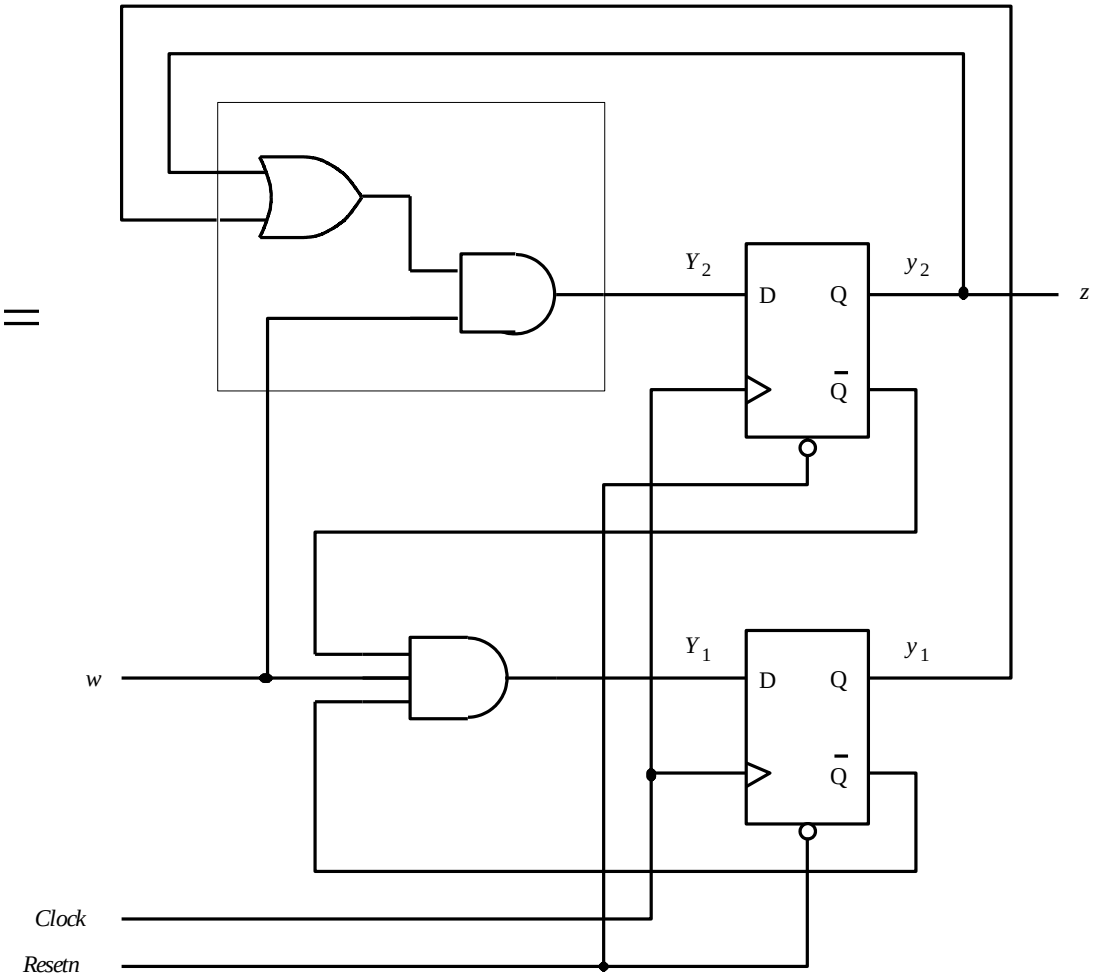
Implementeringen

$$Y_2 = wy_1 + wy_2 =$$

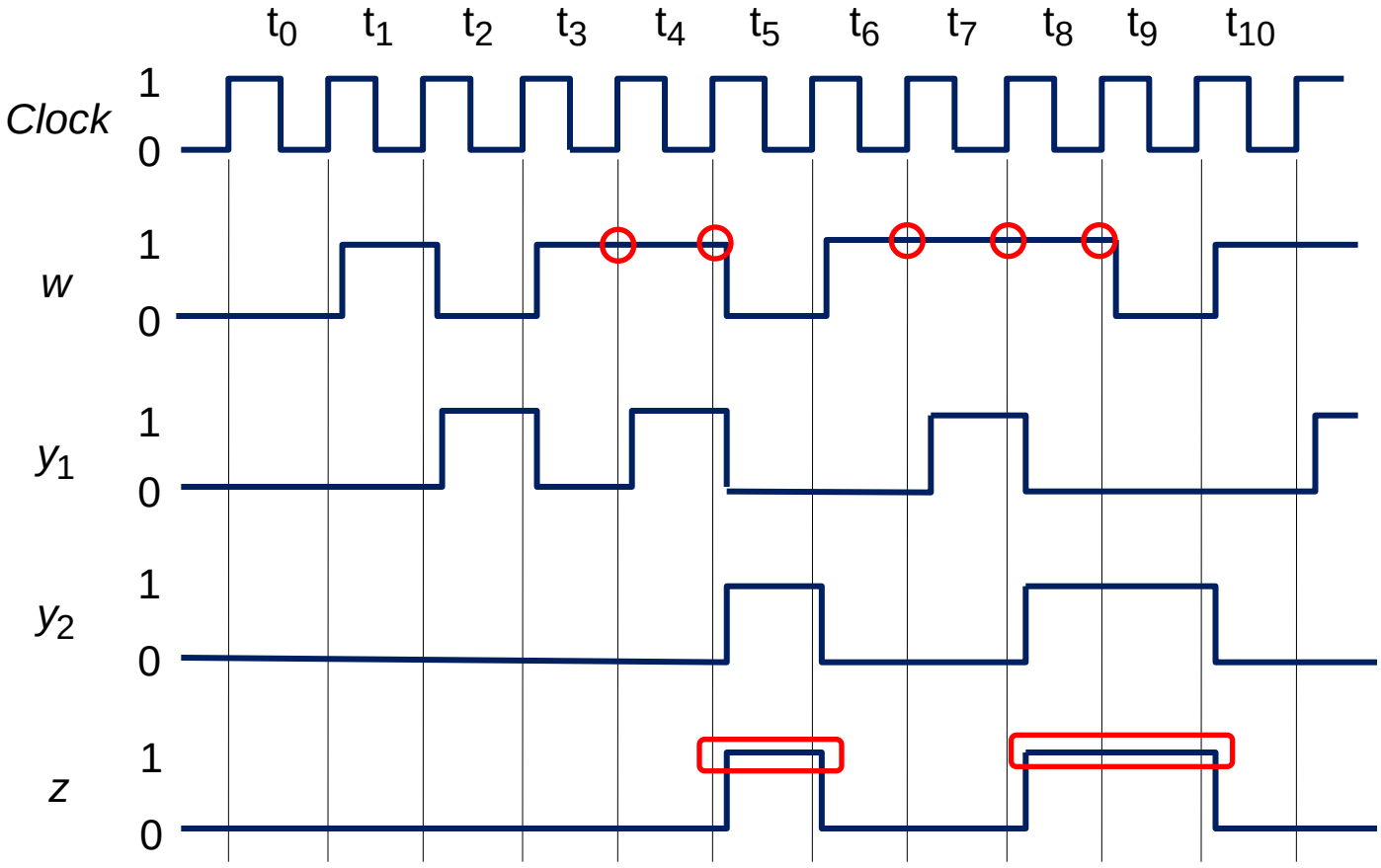
$$= w(y_1 + y_2)$$

$$Y_1 = w\bar{y}_1\bar{y}_2$$

$$z = y_2$$

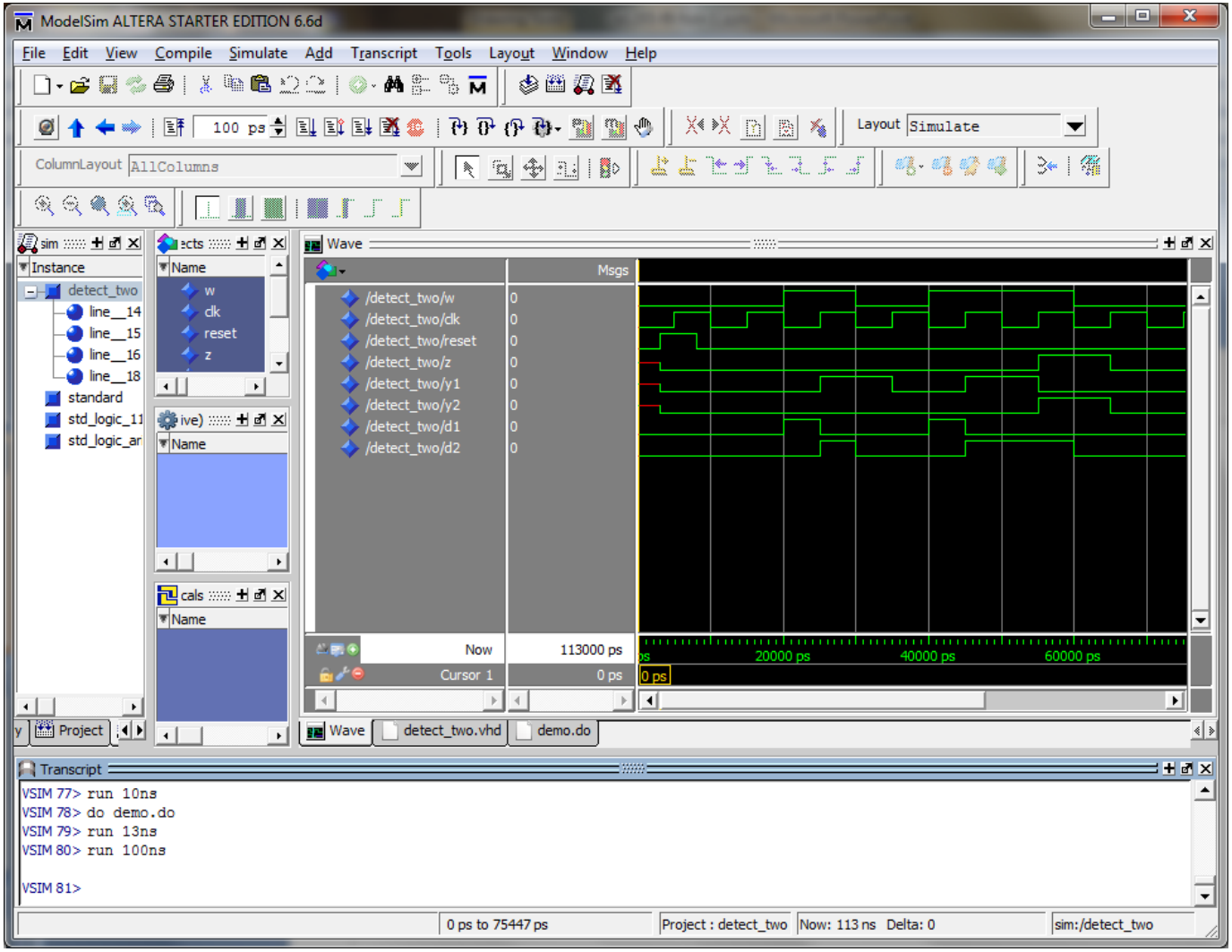


Tidsdiagram ”två i rad”

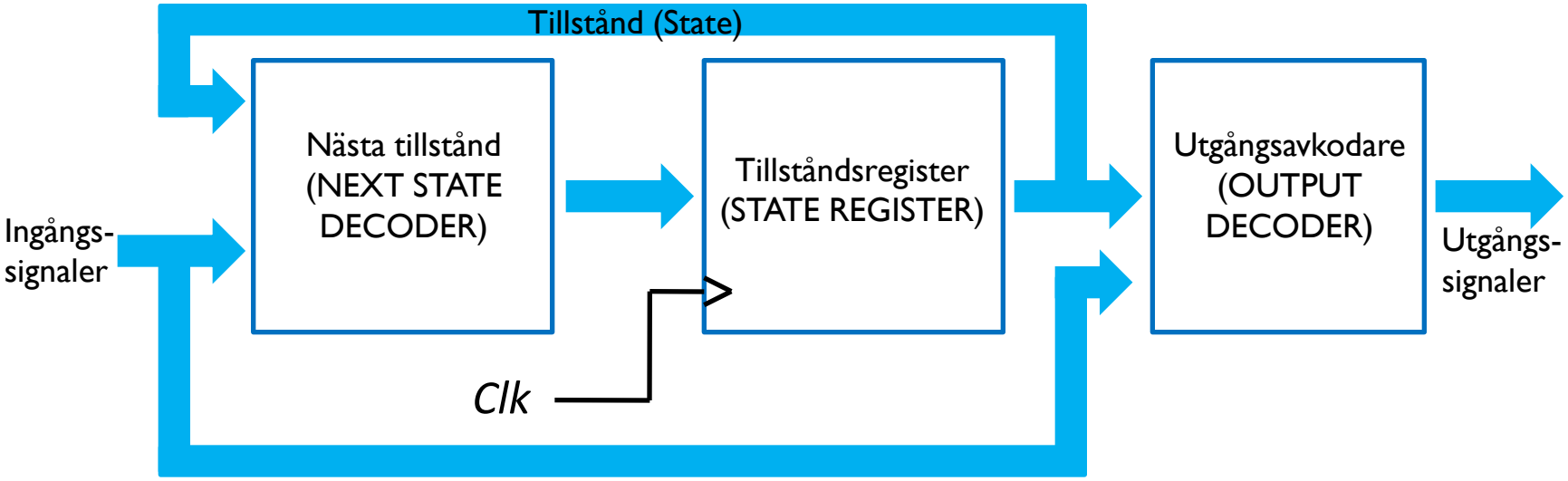


Tillståndsöver-gångar sker bara på den positiva klockflanken!

Demo Modelsim



Mealy-automat

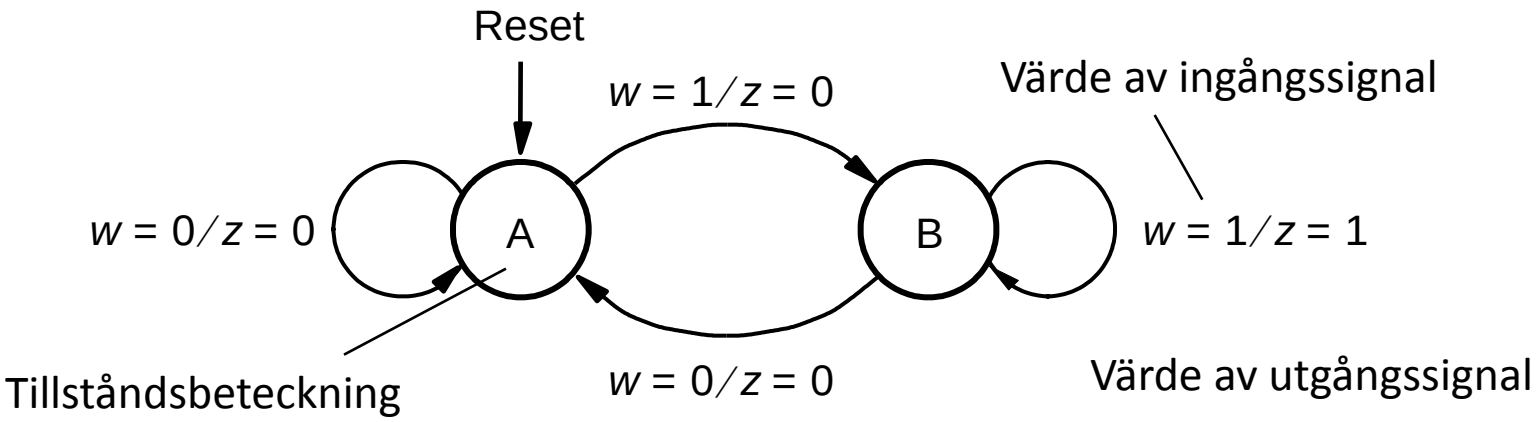


I en **Mealy**-Automat beror utgångssignalerna både på nuvarande tillstånd ***och*** ingångarna

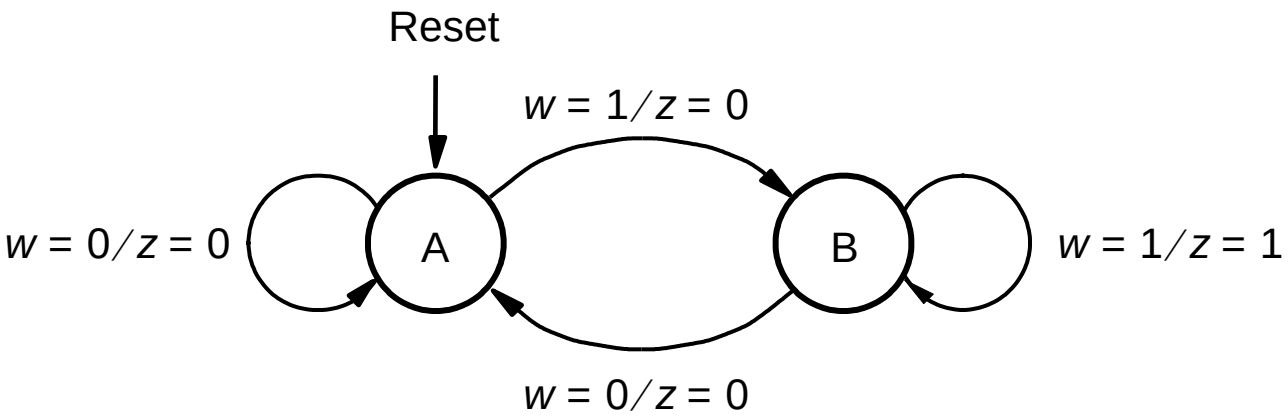
Tillståndsdiagram Mealy

”Två i rad”

- Tillståndsdiagrammet för Mealy-automaten behöver bara två tillstånd
- Utsignalen beror på ***både*** tillstånd och insignaler



Tillståndstabell



Present state	Next state		Output z	
	$w = 0$	$w = 1$	$w = 0$	$w = 1$
A	A	B	0	0
B	A	B	0	1

Två tillstånd – bara en vippa behövs!

Kodad tillståndstabell

Present state	Next state		Output z	
	$w = 0$	$w = 1$	$w = 0$	$w = 1$
A	A	B	0	0
B	A	B	0	1

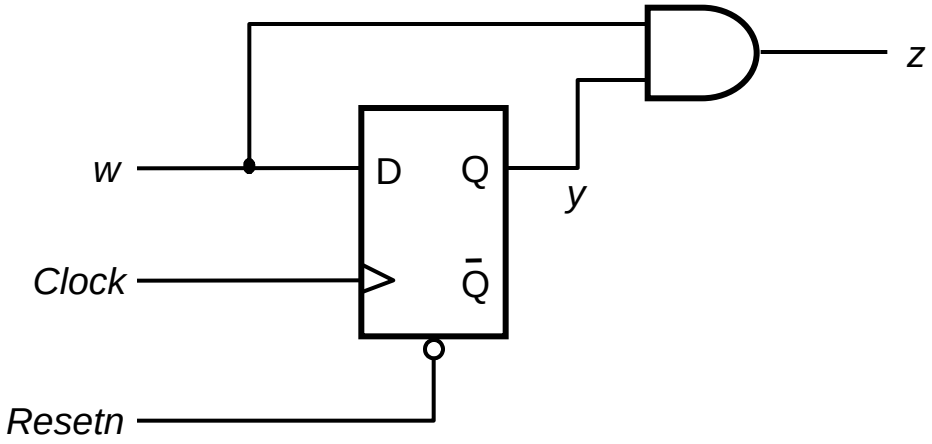
$$Y = f(y\,w) \qquad z = f(y\,w)$$

	Present state	Next state		Output	
		$w = 0$	$w = 1$	$w = 0$	$w = 1$
	y	Y	Y	z	z
A	0	0	1	0	0
B	1	0	1	0	1

Direkt ur tabellen:

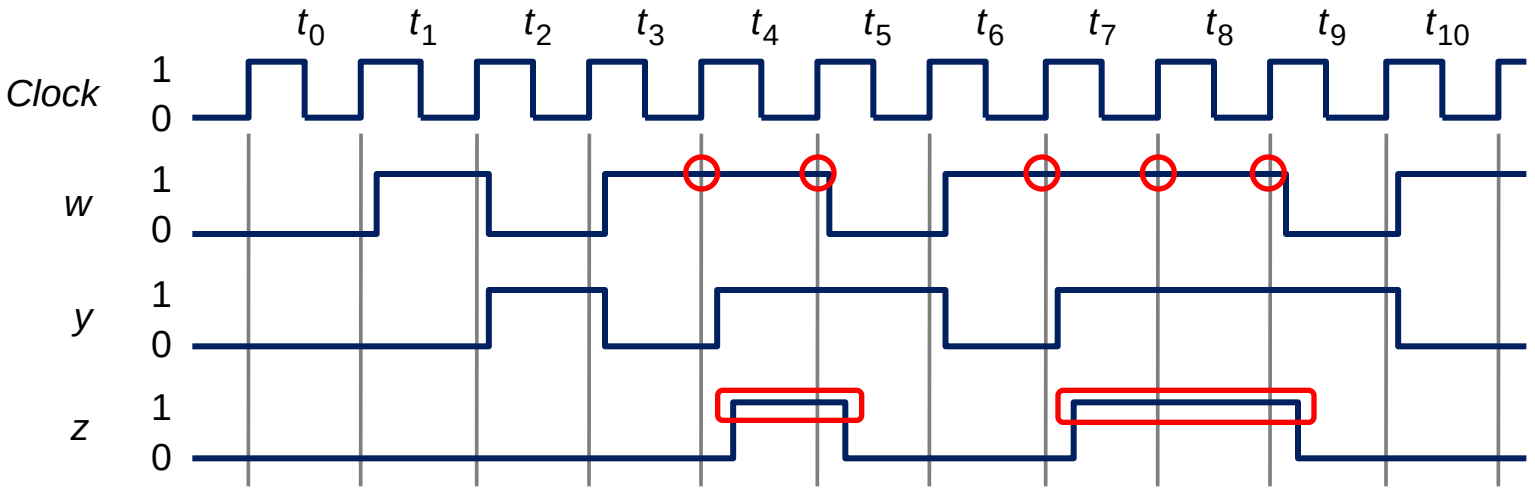
$$Y = w \qquad z = yw$$

Implementeringen



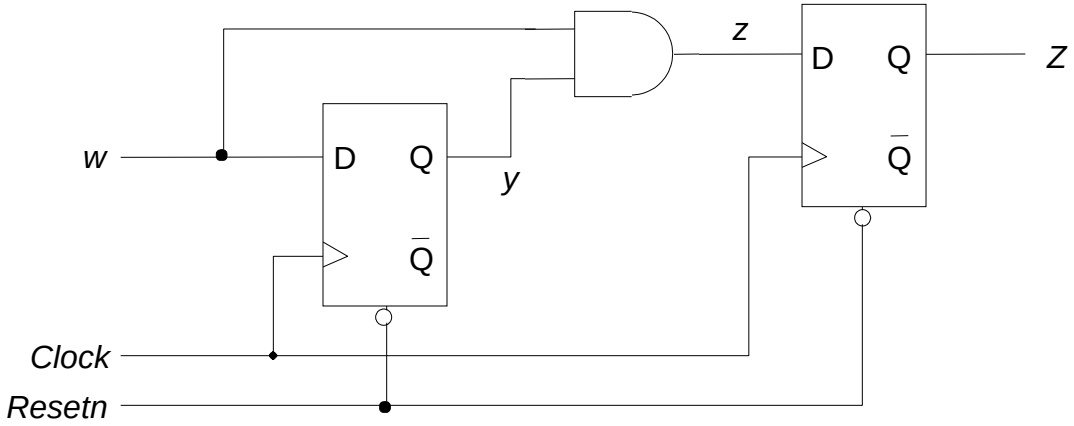
Tidsdiagram

- Utsignalen kan ändrar sig under hela klockperioden eftersom den är en funktion av insignalen
- Jämfört med Moore-automaten så ‘reagerar’ Mealy-automaten tidigare (bitsekvensen detekteras i t_4 jämfört med t_5 i Moore-automaten)



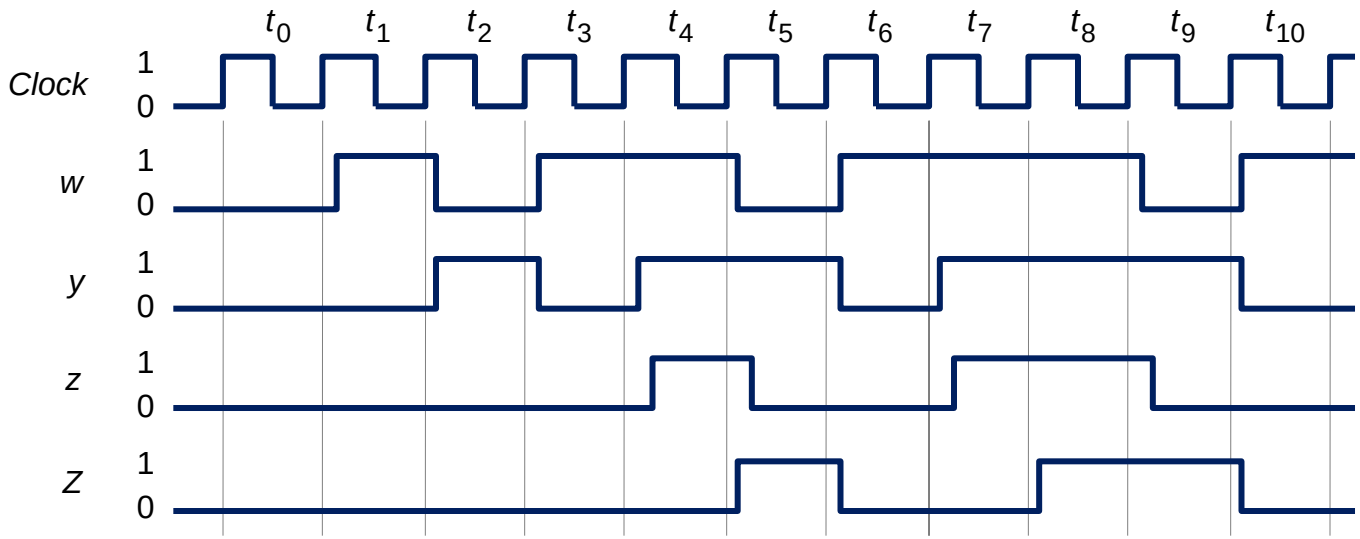
Mealy med utgångsregister

- Nackdelen med Mealy-automaten är att utsignalen kan ändras under hela klockperioden
- Man kan lägga till en register (vippa) på utgången så för att synkronisera utgången med klockflanken



Tidsdiagram med utgångsregister

Med utgångsregister försvinner skillnaden mellan tidsdiagrammen.

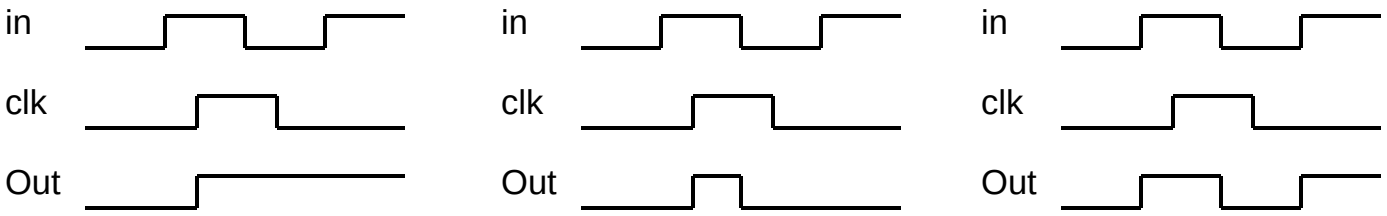


Moore vs Mealy

- Moore-automatens utgångsvärden beror bara på det nuvarande tillståndet
- Mealy-automatens utgångsvärden beror på det nuvarande tillståndet och värden på ingångssignalerna
- Mealy-automaten behöver ofta färre tillstånd
- Mealy-automatens utsignaler är inte synkroniserade med klockan, varför man ofta lägger till ett utgångsregister

Snabbfråga Automater

Vilket/vilka av följande tidsdiagram kan genereras från en Moore Automat?



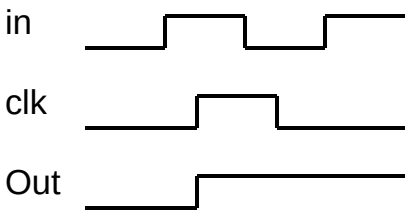
Alt: A

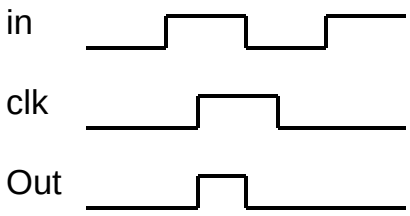
Alt: B

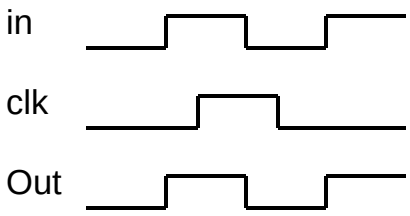
Alt: C

Snabbfråga Automater

Vilket/vilka av följande tidsdiagram kan genereras från en Moore Automat?







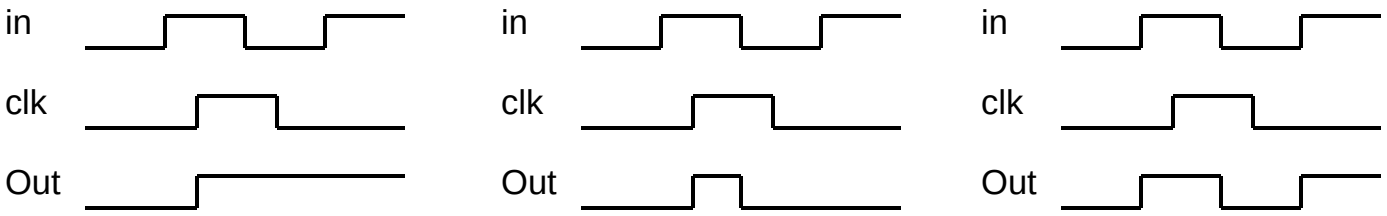
Alt: A

Alt: B

Alt: C

Snabbfråga Automater

Vilket/vilka av följande tidsdiagram kan genereras från en Mealy Automat?



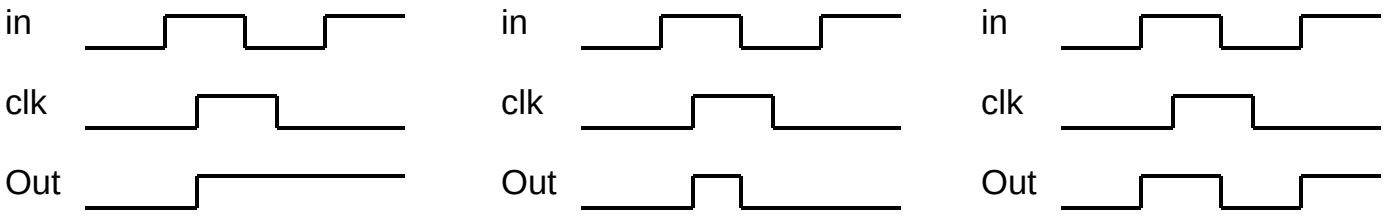
Alt: A

Alt: B

Alt: C

Snabbfråga Automater

Vilket/vilka av följande tidsdiagram kan genereras från en Mealy Automat?



Alt: A

Alt: B

Alt: C

Val av tillståndskodning

Valet av tillståndskodningen kan spela en stor roll för implementeringen eftersom den påverkar logiken för

- Next-state-decoder
- Utgångsavkodare

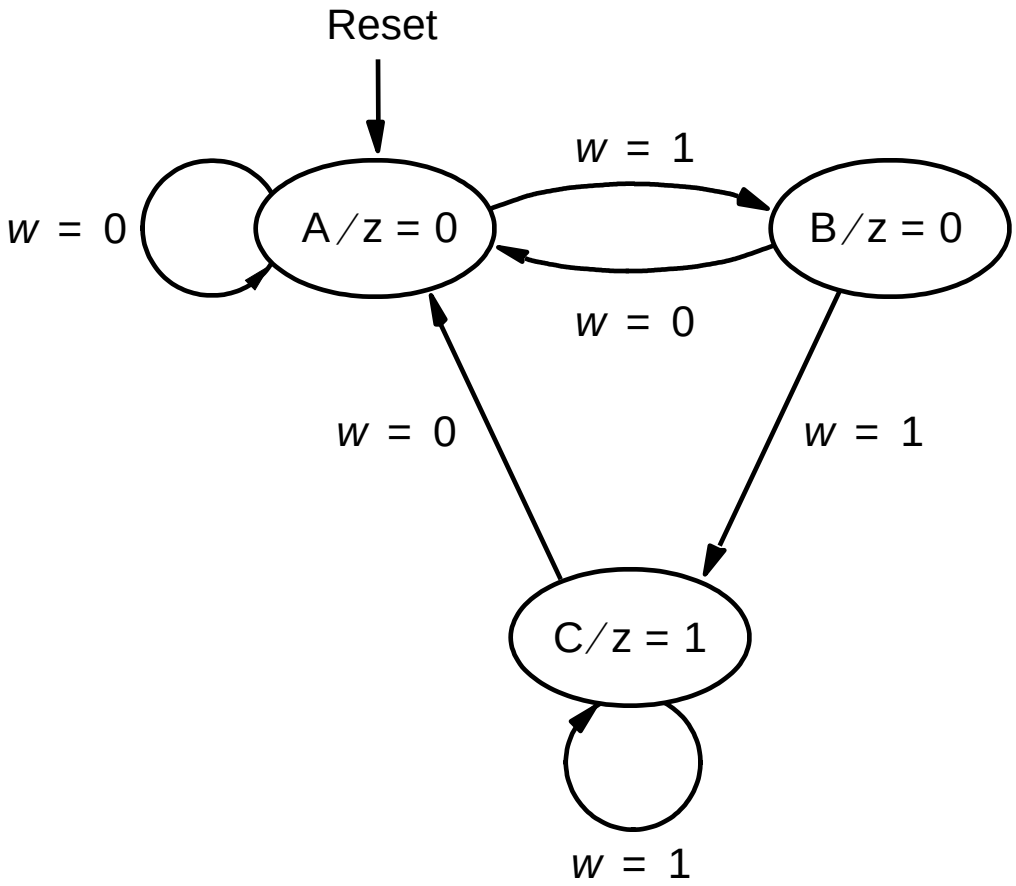
Exempel koder:

- Binär
- Gray
- One hot

One-Hot-kodning

- One-hot-kodningen använder en vippa per tillstånd
- För varje tillstånd är en bit ‘hot’ (1), alla andra bitar är 0,
dvs 0001, 0010, 0100, 1000
- One-hot kodningen minimerar den kombinatoriska logiken men ökar antalet vippor

”två i rad” tillståndsdiagram

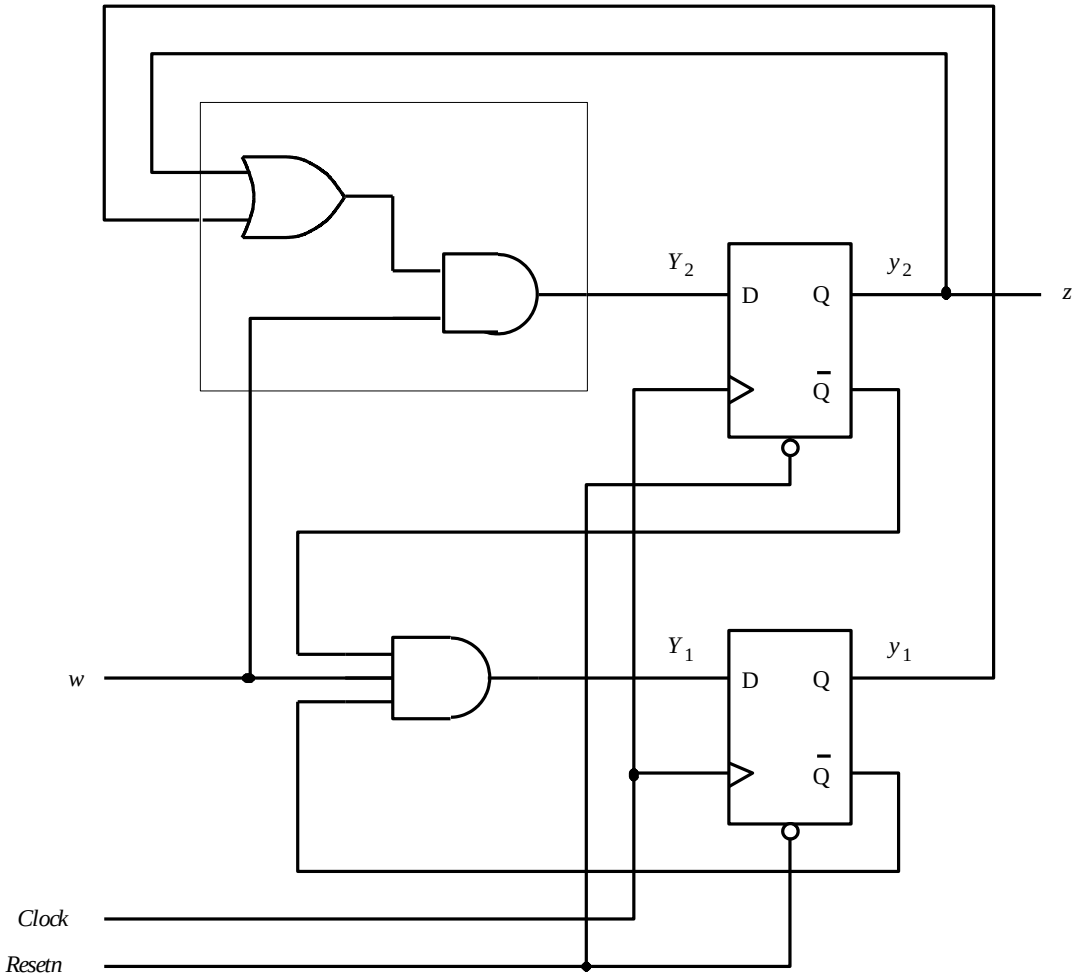


Tillståndskod = Binärkod

	Present state	Next state		Output
		$w = 0$	$w = 1$	
	$y_2 y_1$	$Y_2 Y_1$	$Y_2 Y_1$	z
A	00	00	01	0
B	01	00	10	0
C	10	00	10	1
	11	*dd*	*dd*	*d*

Realisering (Binärkod)

2 D-vippor
2 AND-grindar
1 OR-grind



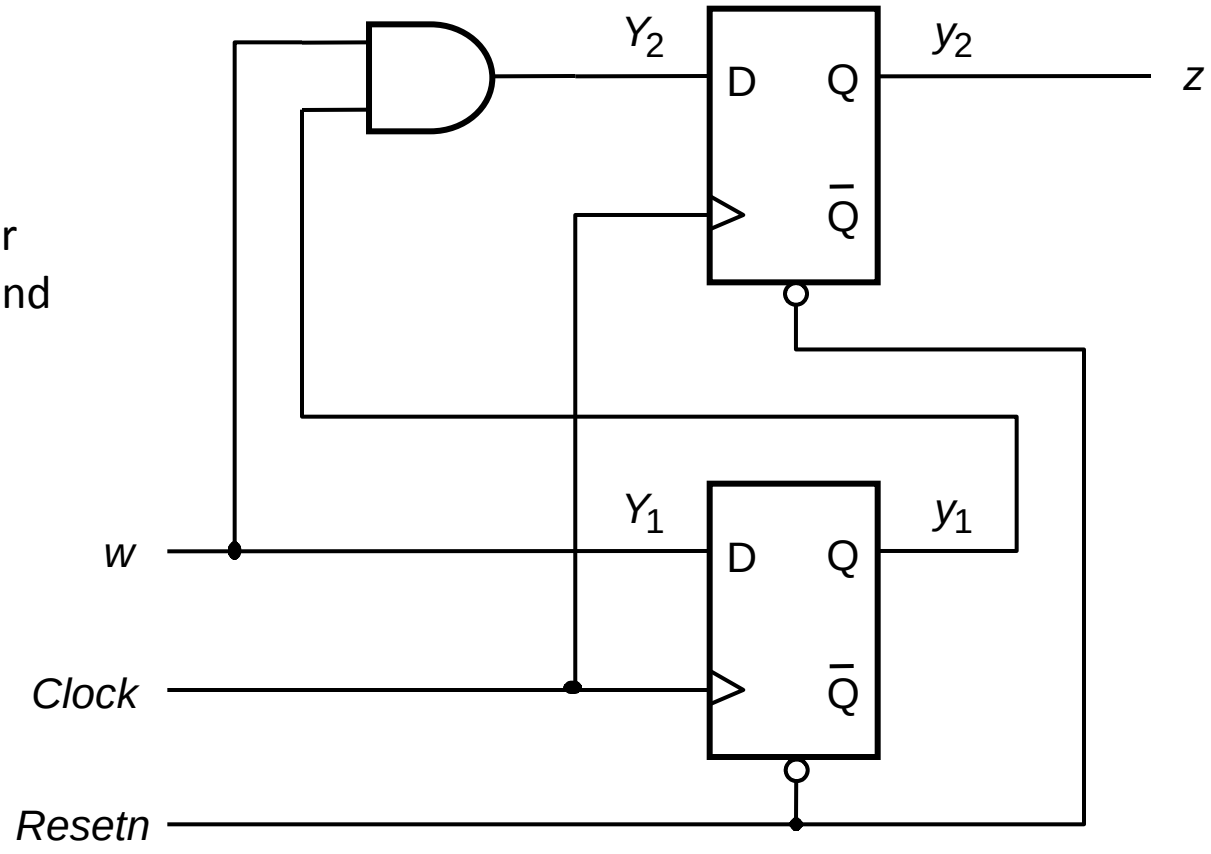
Tillståndskod = Graykod

- I Gray-koden ändras bara en bit åt gången, dvs 00, 01, 11, 10
- Gray-koden är bra för räknare

	Present state	Next state		Output
		$w = 0$	$w = 1$	
	y_2y_1	Y_2Y_1	Y_2Y_1	z
A	00	00	01	0
B	01	00	11	0
C	11	00	11	1
	10	*dd*	*dd*	*d*

Realisering (Graykod)

2 D-vippor
1 AND-grind

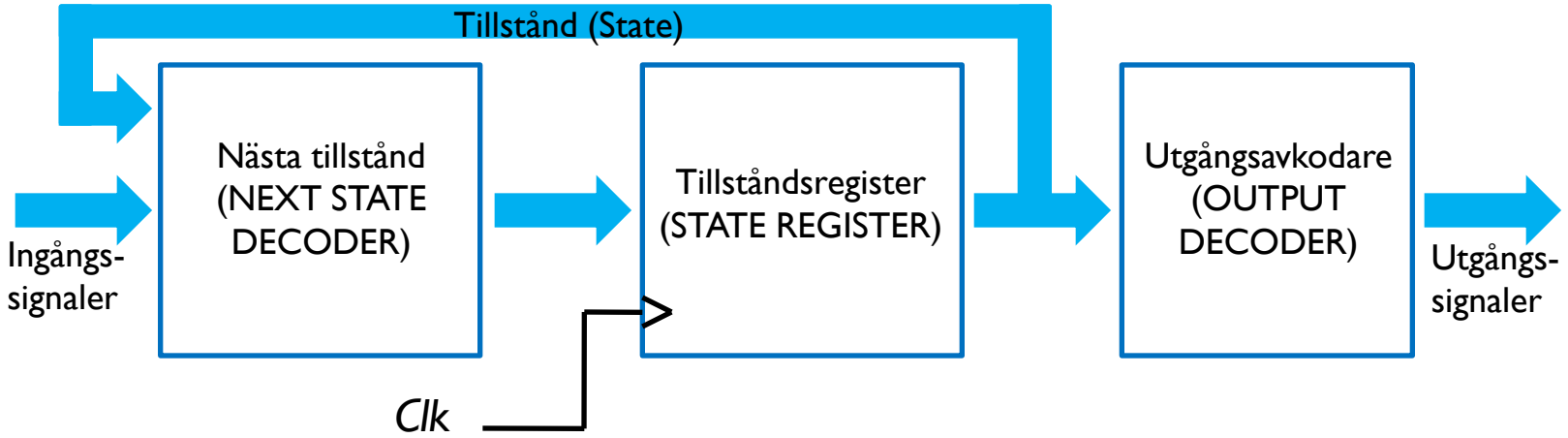


Vilken kod ska man välja?

I detta exempel gav Gray kod enklare realisering, men detta gäller inte för alla tillståndsmaskiner

- Det finns inte en kod som är den bästa i alla lägen, utan det beror helt på tillståndsdiagrammet
- Man kan även ha ‘egna koder’ som passar till konstruktionen, t ex 00, 11, 10, 01

Tillståndsmaskiner



- Moore maskin: Utgång beror bara på tillstånd
- Mealy maskin: Utgång beror på tillstånd och insignal
- Implementering beror på kodning
- Nästa föreläsning: Tillståndsminimering och Asynkrona tillståndsmaskiner