Academie voor Technology, Innovation & Society Delft
Academie voor ICT & Media

Inleiding Digitale Techniek

Week 6 – SR-latch, gated latches, flipflops, register

Jesse op den Brouw

INLDIG/2018-2019

DE HAAGSE HOGESCHOOL

Geheugen

- Tot nu toe zijn alleen *combinatorische* schakelingen behandeld.
- Bij deze schakelingen zijn de uitgangen alleen afhankelijk van de ingangen.
- Schakelingen die geheugenwerking hebben worden *sequentiële* schakelingen genoemd.
- De uitgangen van deze schakelingen zijn afhankelijk van de ingangen én de inhoud van het geheugen.
- De inhoud van het geheugen kan weer aangepast worden.

Geheugen

- Er zijn verschillende soorten geheugenelementen:
- Een (direct werkende) SR-latch
- Een gated latch (SR- en D-)
- Flankgevoelige flipflop (D- en JK-)
  - master-slave, data-lock/voltage-triggered
- Pulse-triggered flipflop (SR- en JK) (wordt niet meer gemaakt).

SR-latch

- Een geheugenelement heeft de volgende acties:

  Een actie waarin het geheugenelement gezet (set) wordt.

  Een actie waarin het geheugenelement gewist (reset) wordt.

  Een actie waarin de laatst ingebrachte stand van het geheugenelement onthouden wordt.

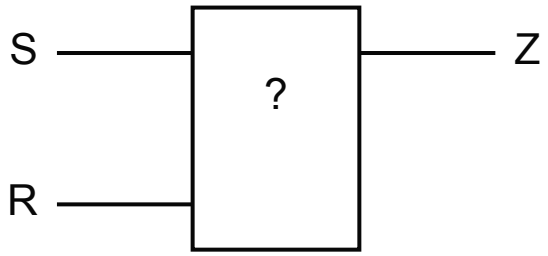
- Deze drie acties kunnen met twee (stuur-)bits gecodeerd worden.

SR-latch

- De namen van deze twee signalen zijn S (set) en R (reset), deze namen hebben een directe relatie met de acties.
- De uitgang wordt in de regel Z genoemd, sommige boeken gebruiken Q.
- Het geheugenelement is te tekenen als een black box.
- Van dit te ontwerpen geheugenelement kan een functietabel worden opgesteld.

SR-latch

- Hieronder de black box en de functietabel.



S	R	functie
0	0	onthouden
0	1	reset
1	0	set
1	1	niet gebruikt

- NB: voor één van de acties moet altijd een *combinatie* van S en R worden aangeboden.

- NB: de combinatie SR = 11 wordt (voorlopig) niet gebruikt.

SR-latch

- Bij de set- en resetactie moet de stand van het geheugenelement worden aangepast, dus de oude waarde wordt vervangen door een nieuwe waarde.

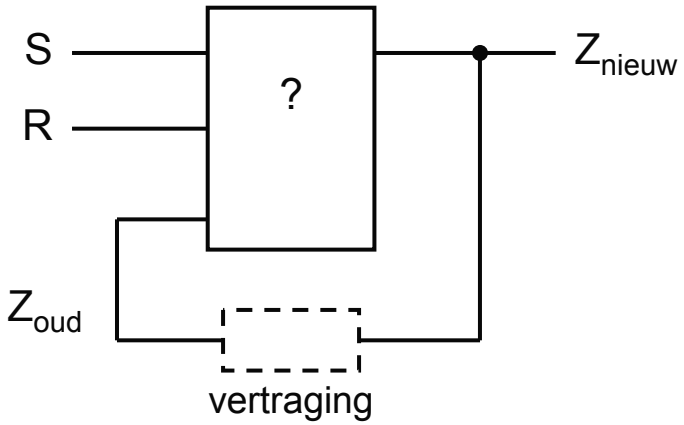
- Bij de onthoudactie moet de stand behouden blijven, dus de nieuwe waarde is hetzelfde als de oude waarde.

- Er wordt dan ook wel gesproken van oude waarde Z_{oud} en de nieuwe waarde Z_{nieuw}.

- De te ontwerpen schakeling heeft dus eigenlijk drie ingangen S, R en Z_{oud} en één uitgang Z_{nieuw}.

SR-latch

- We kunnen de schakeling als volgt modelleren:



- De vertraging zorgt ervoor dat de waarde van Z_{oud} nog even beschikbaar blijft, als de nieuwe waarde op Z_{nieuw} gegenereerd wordt.

SR-latch

- Rechts is de waarheidstabel.
- Bij SR = 00 moet de nieuwe waarde de oude waarde volgen.
- Bij SR = 01 moet de nieuwe waarde op 0 worden gezet.
- Bij SR = 10 moet de nieuwe waarde op 1 worden gezet.
- Bij SR = 11 wordt de nieuwe waarde als don't care gespecificeerd.

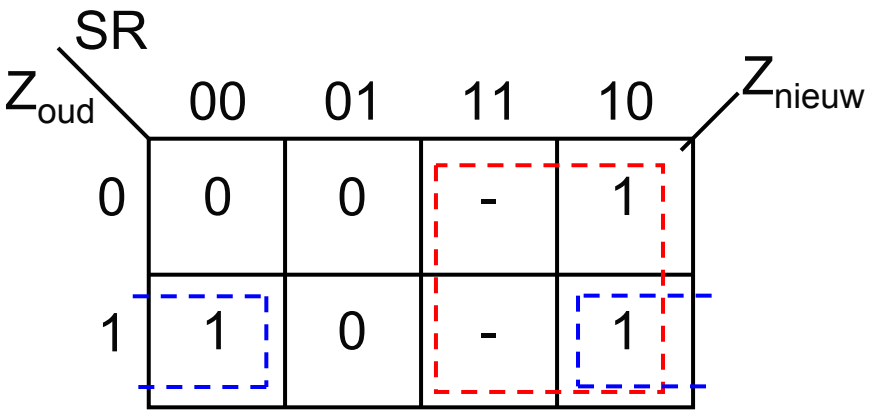
S	R	Z_{oud}	Z_{nieuw}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	-
1	1	1	-

DE HAAGSE HOGESCHOOL

SR-latch

- Van deze waarheidstabel kan een Karnaughdiagram worden op gesteld:
- De bijbehorende functie is:

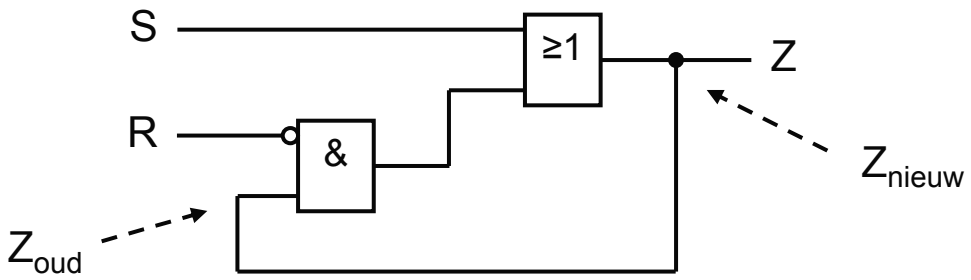
$$Z_{nieuw} = S + \overline{R} \cdot Z_{oud}$$



- De functie laat duidelijk zien dat Z_{nieuw} een functie is van Z_{oud}.
- De ingangscombinatie SR = 11 is na uitwerking gebruikt als set-operatie. Het geheugenelement heeft een *overheersende set*.

SR-latch met overheersende set

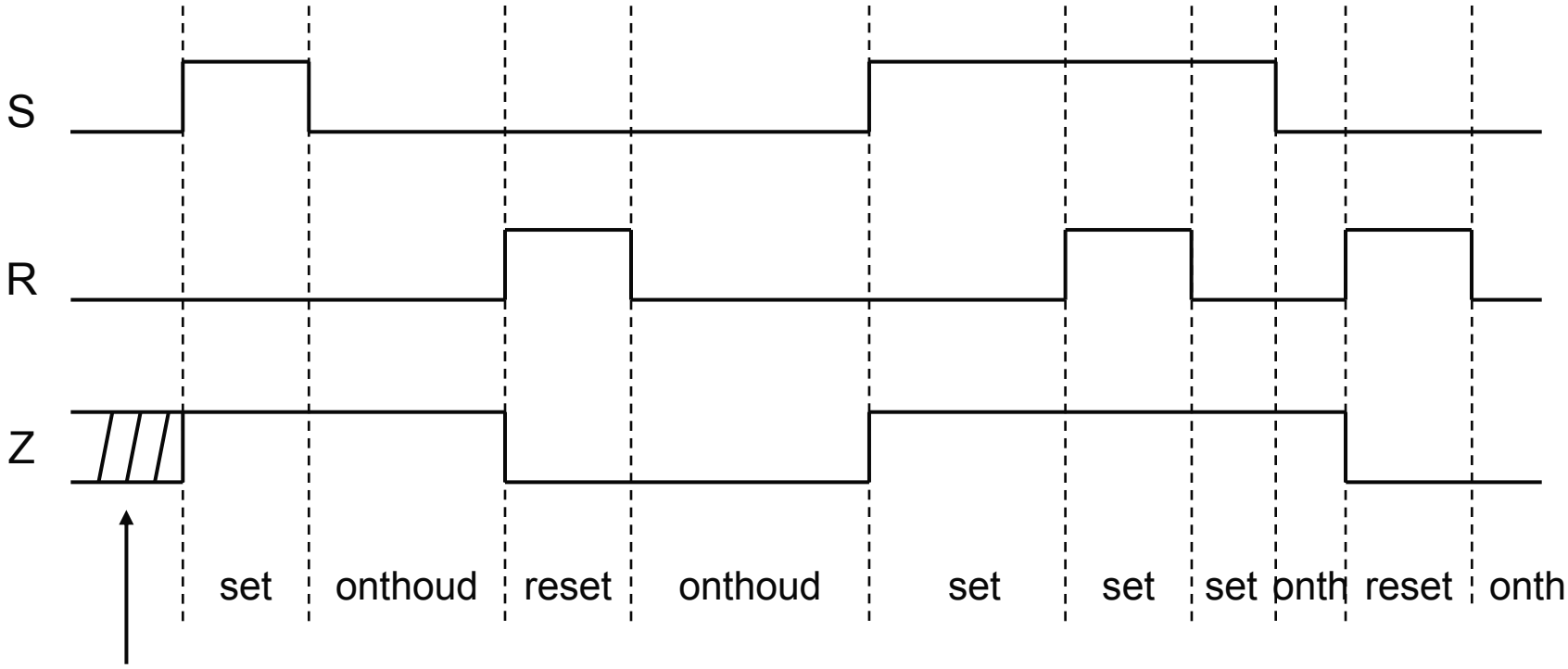
- Nu kan het schema getekend worden. Merk op dat de uitgang Z wordt teruggekoppeld als een ingang.



- Kenmerkend voor deze *SR-latch* is de *lus* in de schakeling.

Signaaldiagram SR-latch met overheersende set

- In het *signaaldiagram* is het gedrag van het SR-latch goed weer te geven.



SR-latch met overheersende set

- De functie voor het SR-latch kan worden omgewerkt tot een NAND-NAND-vorm:

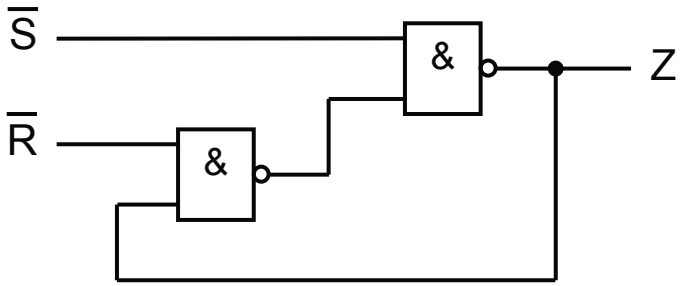
$$Z_{nieuw} = S + \overline{R} \cdot Z_{oud} = \overline{\overline{S} \cdot (\overline{\overline{R} \cdot Z_{oud}})}$$

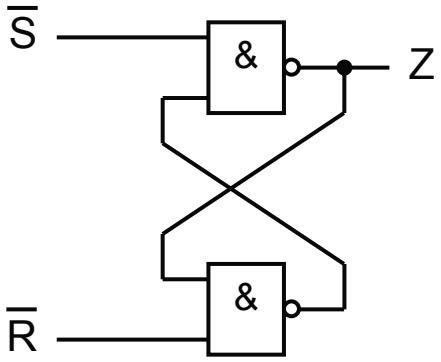
- De schakeling is nu te bouwen met 2 NANDs.

- Voor de stuursignalen zijn wel beide inversen nodig.

SR-latch met overheersende set

- De schakeling is als volgt te tekenen.

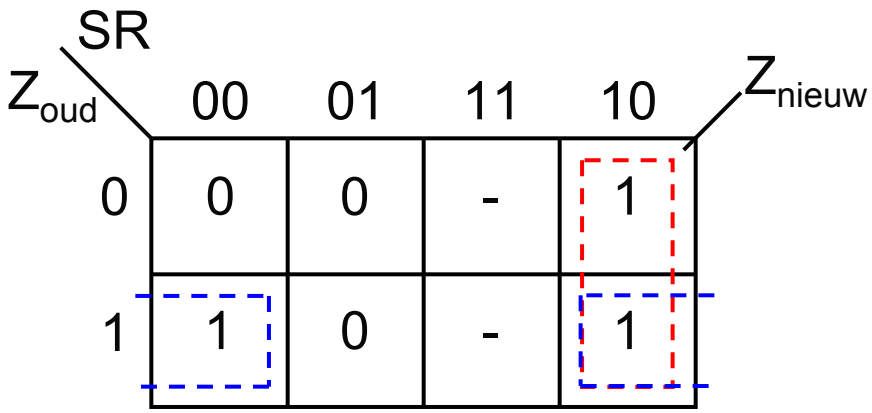




- De tweede variant wordt doorgaans gebruikt in de literatuur.

SR-latch met overheersende reset

- Het Karnaughdiagram van het SR-latch wordt nog eens opnieuw uitgewerkt:
- De bijbehorende functie is nu:

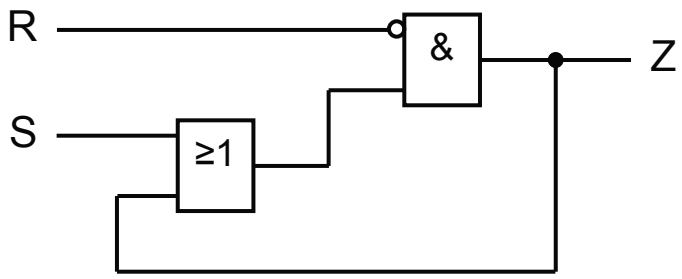


$$Z_{nieuw} = \overline{R} \cdot S + \overline{R} \cdot Z_{oud} = \overline{R} \cdot (S + Z_{oud})$$

- De ingangscombinatie SR = 11 is na uitwerking gebruikt als resetoperatie. Het geheugenelement heeft een *overheersende reset*.

SR-latch met overheersende reset

- Nu kan het schema getekend worden.



- In feite zijn de poorten van plaats verwisseld. Let op dat R nu boven staat en S onder.

SR-latch met overheersende reset

- De functie voor de SR-latch kan worden omgewerkt tot een NOR-NOR-vorm:

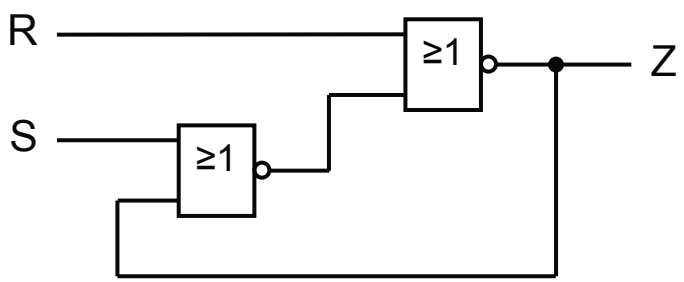
$$Z_{nieuw} = \overline{R} \cdot (S + Z_{oud}) = \overline{R + (\overline{S + Z_{oud}})}$$

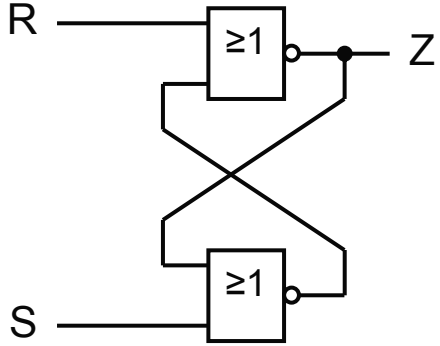
- De schakeling is nu te bouwen met 2 NORs.

- Voor de stuursignalen zijn *geen* inversen nodig.

SR-latch met overheersende reset

- De schakeling is als volgt te tekenen.





- De tweede variant wordt doorgaans gebruikt in de literatuur, maar soms zó getekend dat S weer boven staat.

SR = 11

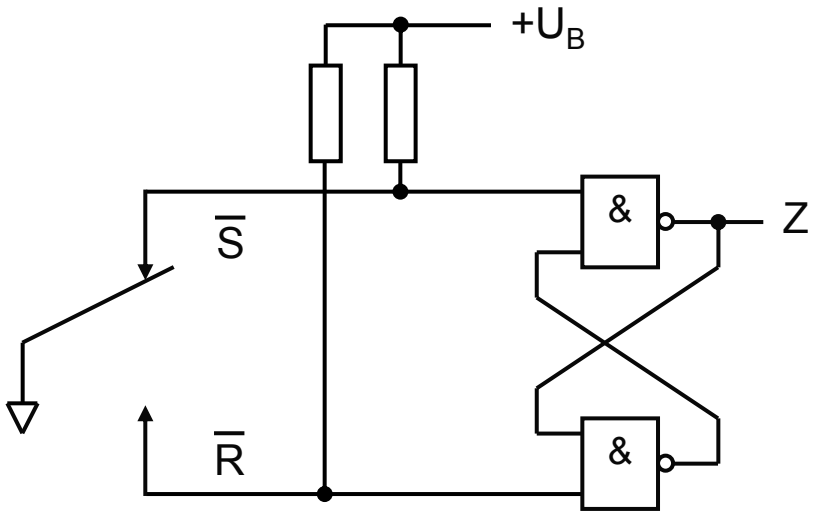
- In veel boeken wordt de combinatie SR = 11 de *verboden stand* genoemd.

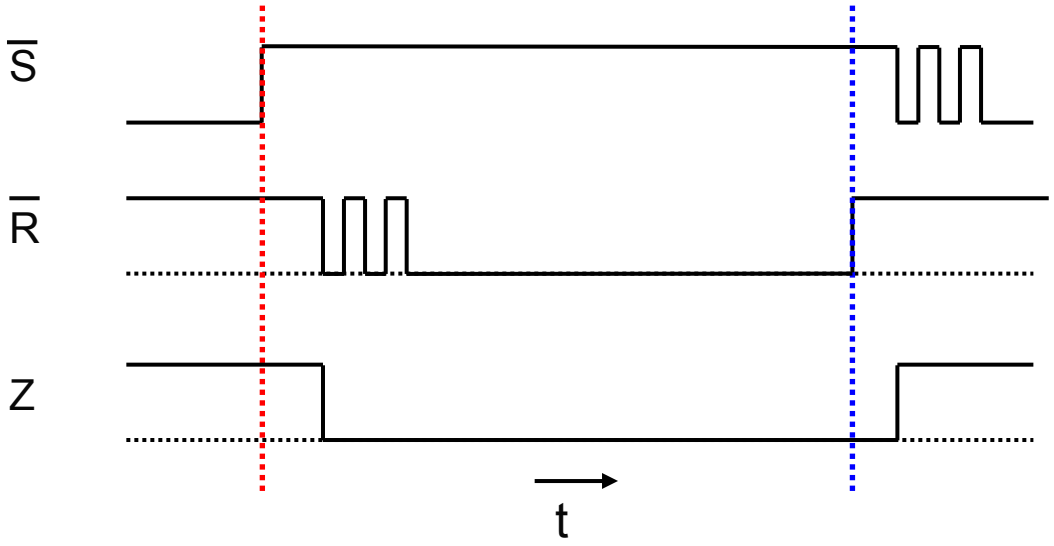
- Deze combinatie is dan wel niet verboden, maar wordt in de praktijk nauwelijks gebruikt.

- Daarnaast levert het problemen op als van SR = 11 naar SR = 00 gegaan wordt. Dit is in de praktijk nooit goed te realiseren. Eén van de twee stuursignalen verandert het eerst, zodat onbedoeld een set- of resetoperatie gestart wordt.

Schakelaar ontdenderen

- Mechanische schakelaars hebben last van denderen. Deze schakeling zorgt dat dit denderen wordt opgeheven.

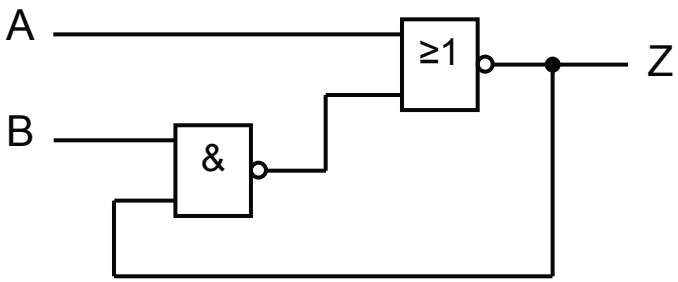




Opgaven

- Op de vorige slides zijn het SR-latch met overheersende set en reset behandeld. De don't cares waren dan ingevuld als 1-en en als 0-en. Het zijn twee don't cares, dus er zijn vier combinaties mogelijk. Ontwerp SR-latches voor de twee overgebleven combinaties. Zijn de ontwerpen zinvol?

- Gegeven de onderstaande schakeling. Realiseert deze schakeling een goedwerkend geheugenelement?



Opgaven

- Is het mogelijk om een SR-latch te maken met de onderstaande SR-combinaties? Motiveer je antwoord.

SR = 11	onthouden
SR = 10	niet gebruikt
SR = 01	reset
SR = 00	set

- Vul het eerder gegeven timingdiagram in maar nu voor een SR-latch met overheersende reset.

Gated latch

- SR-latches zijn transparant voor hun ingangssignalen.
- Eventuele veranderingen van de ingangen heeft direct een gevolg voor de waarde van de uitgang.
- Meestal worden de signalen voor S en R door combinatorische schakelingen opgewekt.
- Deze schakelingen kunnen *hazards** op de uitgangen vertonen, waardoor onbedoelde combinaties voor S en R worden opgewekt.

* Onbedoelde tijdelijke uitgangsverandering als gevolg van ingangsveranderingen en verschillen in looptijden in een combinatorische schakeling.

Gated latch

- Beter is het om de ingangen van een SR-latch ongevoelig te maken voor deze verandering.
- De SR-latch wordt uitgebreid met een enable-signaal EN en wordt een gated SR-latch genoemd.
- Als het enable-signaal 0 is, houdt de latch de huidige waarde vast (onthouden).
- Als het enable-signaal 1 is, heeft de latch de werking van een gewone SR-latch.

Gated latch

- Bij een SR-latch op basis van AND en OR is ingangscombinatie SR = 00 de onthoudactie.

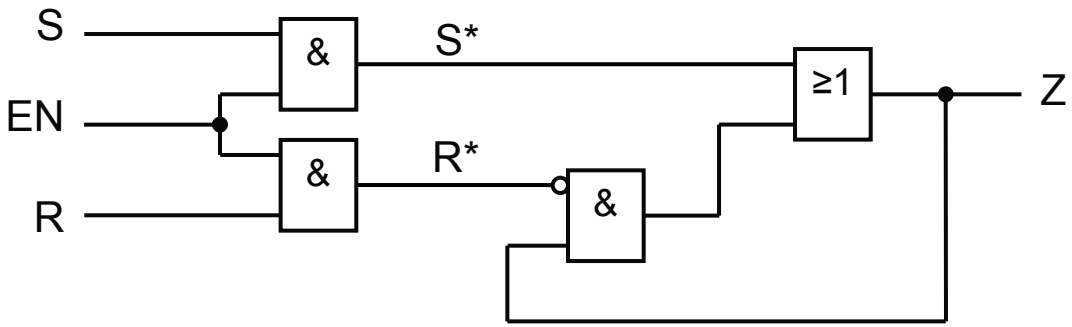
- Als het enable-signaal 0 is moet de aangeboden ingangscombinatie voor SR geblokkeerd en vervangen worden door de ingangscombinatie voor de onthoudactie.

- Dit kan gerealiseerd worden door een AND-poort. De functies worden dan

$$S^* = S \cdot EN$$
$$R^* = R \cdot EN$$

Gated SR-latch

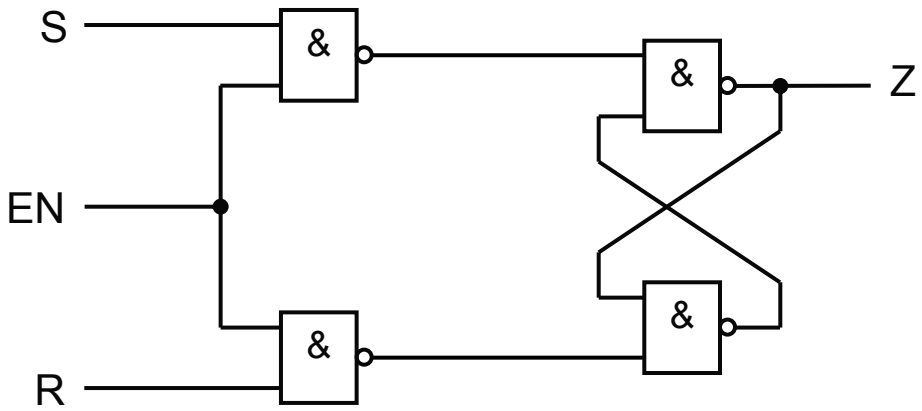
- De schakeling ziet er dan als volgt uit.



- Tijdens EN = 0 is de latch niet meer transparant.
- Ingangsveranderingen tijdens EN = 0 worden uitgesteld tot in de periode dat EN = 1.

Gated SR-latch

- Door gebruik te maken van NANDs kan de schakeling als volgt worden getekend.



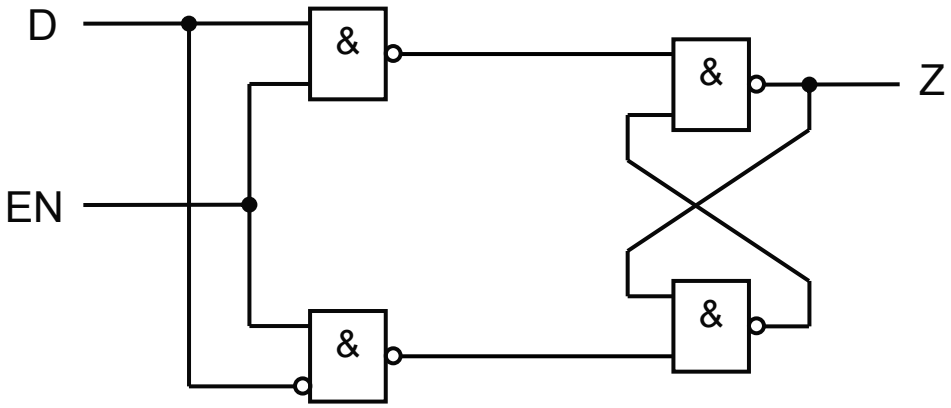
- Nu zijn de inversen van S* en R* niet nodig!

Gated latch

- De nu ontwikkelde latch heeft eigenlijk vijf onthoudstanden: SR = 00 tijdens EN = 1 en EN = 0 (waarbij SR = don’t care).
- De combinatie SR = 00 kan dus geschrapt worden, want die wordt overgenomen door EN = 0.
- De combinatie SR = 11 wordt toch niet gebruikt en kan dus ook geschrapt worden.
- Er blijven twee combinaties over: SR = 01 voor reset en SR = 10 voor set (tijdens EN = 1).
- Nu blijkt dat S en R complementair zijn.

Gated D-latch

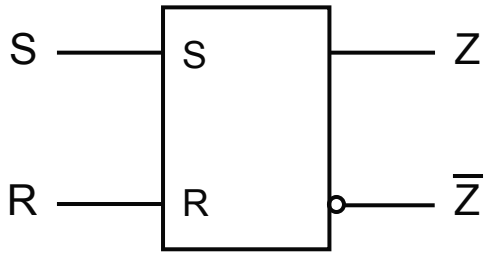
- De waarde van R is te verkrijgen door de inverse van S te nemen. De namen S en R verdwijnen (er is immers nog maar één ingang over) en wordt nu D.

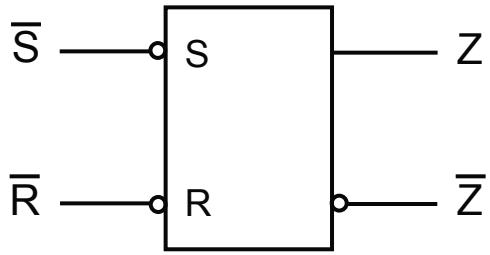


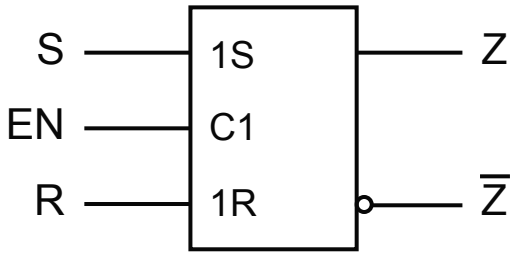
- De hier gepresenteerde schakeling heet *gated D-latch*.

Symbolen

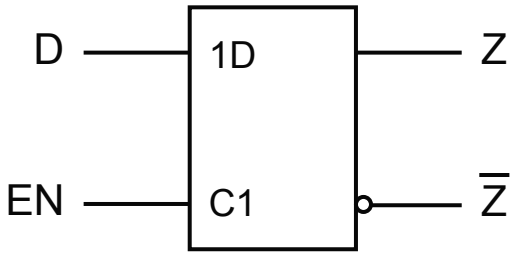
- Hieronder de symbolen voor de diverse latches:



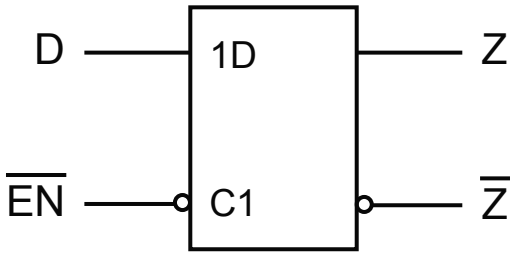




werking S en R
afhankelijk van C



werking D
afhankelijk van C

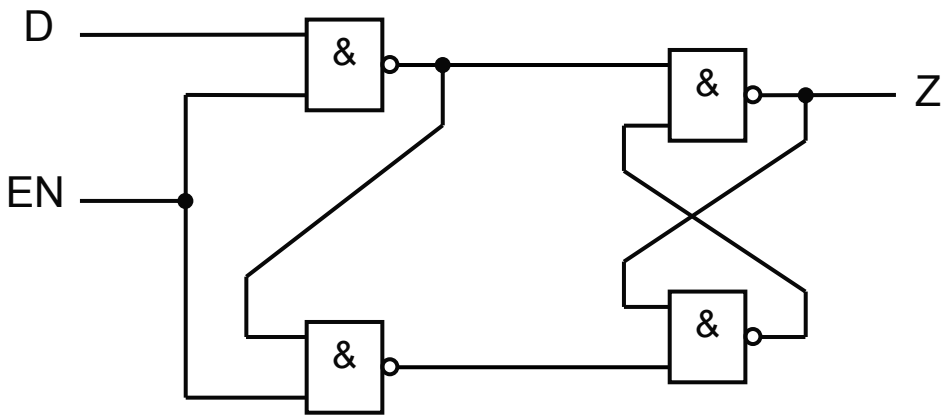


werking D
afhankelijk van C

Opgaven

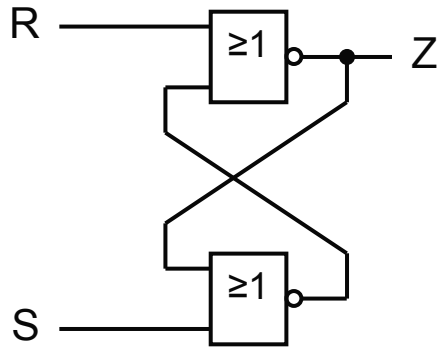
- In het symbool van een SR-latch komt de uitgang $\overline{Z}$ voor. Dat suggereert dat dit de inverse is van Z. Is dat voor elke combinatie van S en R?

- Onderstaande schakeling lijkt op een gated D-latch. Is dit een correct werkende gated D-latch? Motiveer je antwoord.



Timing SR-latch

- De duur van de set-en resetopdracht is uit te drukken in poorttijden. Vanuit S of R is te zien dat een signaal minstens drie poorttijden stabiel moet blijven om de lus in te stellen.

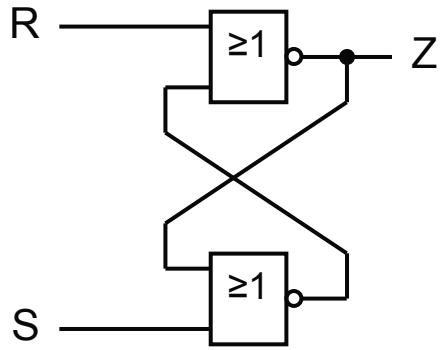


$t_{w(min)}$(setopdracht) = 3·$t_{P(max)}$(NOR) (w staat voor width)

$t_{w(min)}$(resetopdracht) = 3·$t_{P(max)}$(NOR)

Timing SR-latch

- Een SR-latch heeft net als een poort een minimum en maximum vertragingstijd. De kortste tijd loopt van R naar Z, de langste tijd loopt van S naar Z.



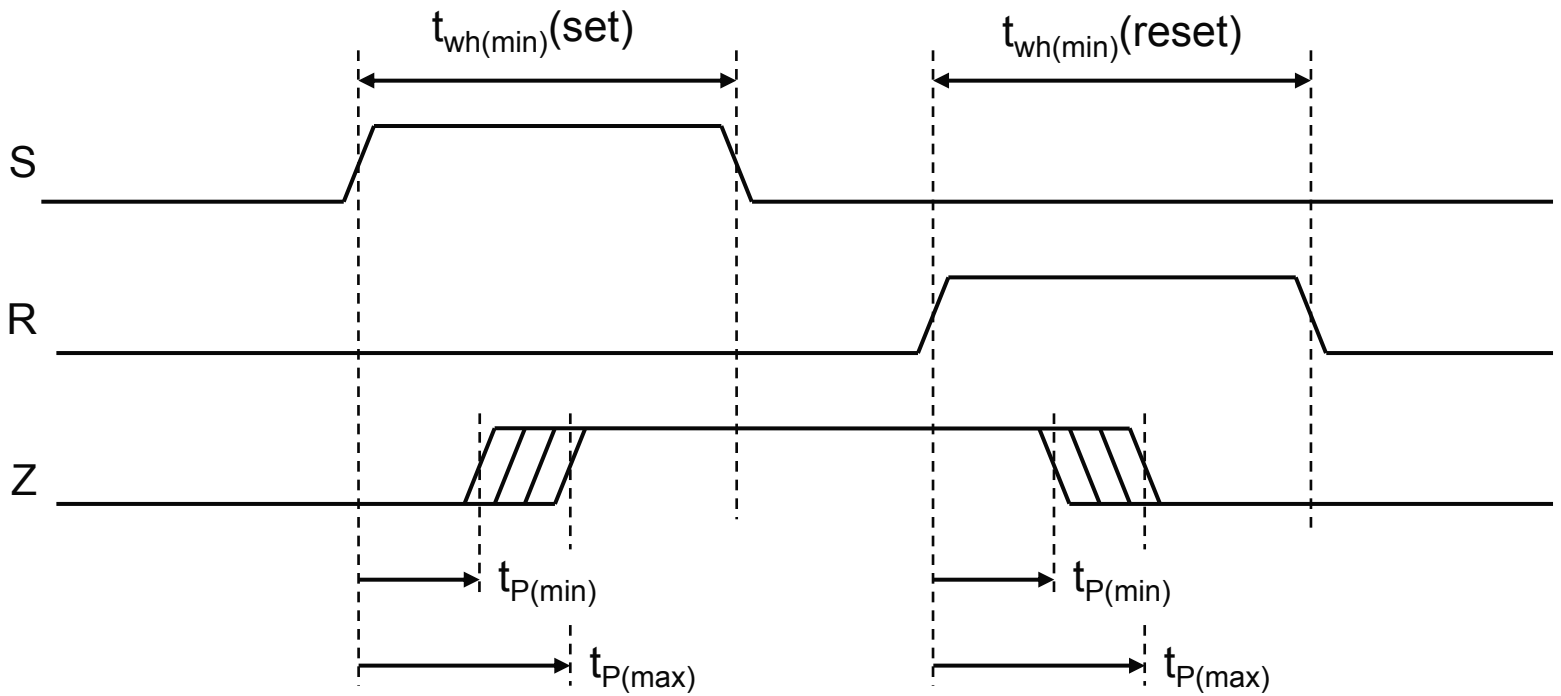
$t_{P(min)}(\text{latch}) = t_{P(min)}(\text{NOR}) = t_{P(min)}(\text{R-to-Z})$
$t_{P(max)}(\text{latch}) = 2 \cdot t_{P(max)}(\text{NOR}) = t_{P(max)}(\text{S-to-Z})$

Dit geldt alleen als de duur van de set- of resetopdracht gerespecteerd wordt.

DE HAAGSE HOGESCHOOL

Timing SR-latch

- Het geheel in een tijddiagram

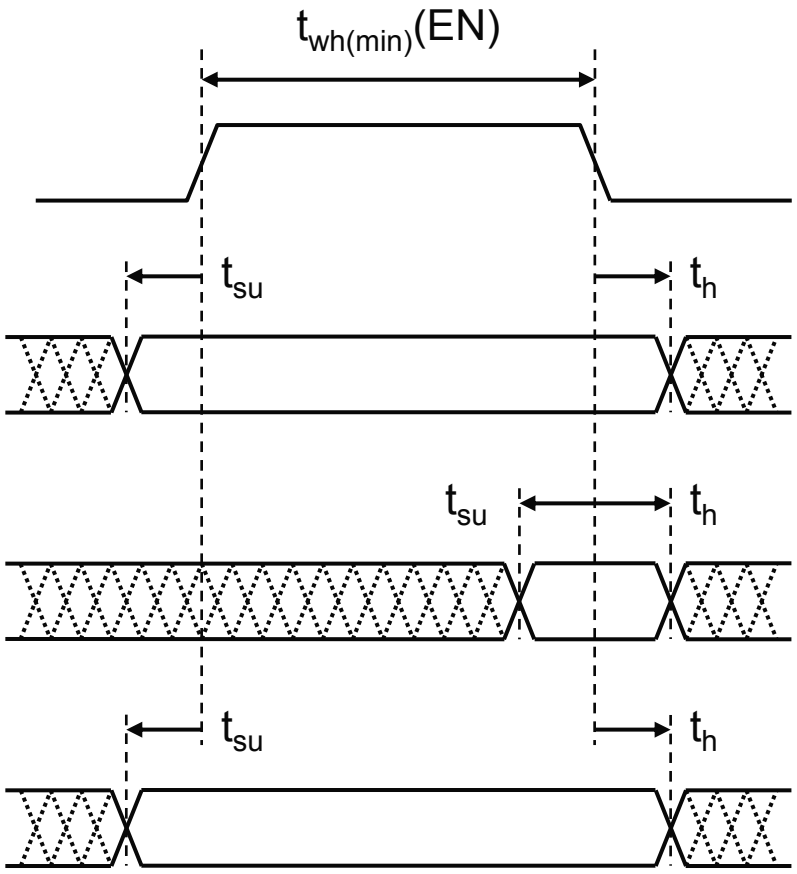


Timing gated SR-latch

- De timing van een SR-latch is erg complex.
- We onderscheiden twee situaties: SR = 00 en SR = 01/10 (SR = 11 wordt buiten beschouwing gelaten).
- Bij SR = 00 onthoudt de latch. Tijdens EN = 1 mogen S en/of R dan niet veranderen, anders wordt (onbedoeld) een set- op reset-opdracht gegeven. Denk hierbij aan glitches en spikes van de sturende logica. Rond het opengaan en sluiten van de latch moeten S en R stabiel zijn zodat de latch zich netjes kan instellen.
- Bij SR = 01 of SR = 10 wordt de latch gereset of geset, dat is een bedoelde operatie. Tijdens EN = 1 mogen S en R zich instellen. Rond het sluiten van de latch moeten S en R stabiel zijn zodat de latch zich netjes kan instellen.

Timing gated SR-latch

- Timing van de instelsignalen.



Timing gated SR-latch

- Voor de timing van de uitgang Z onderscheiden we twee situaties:

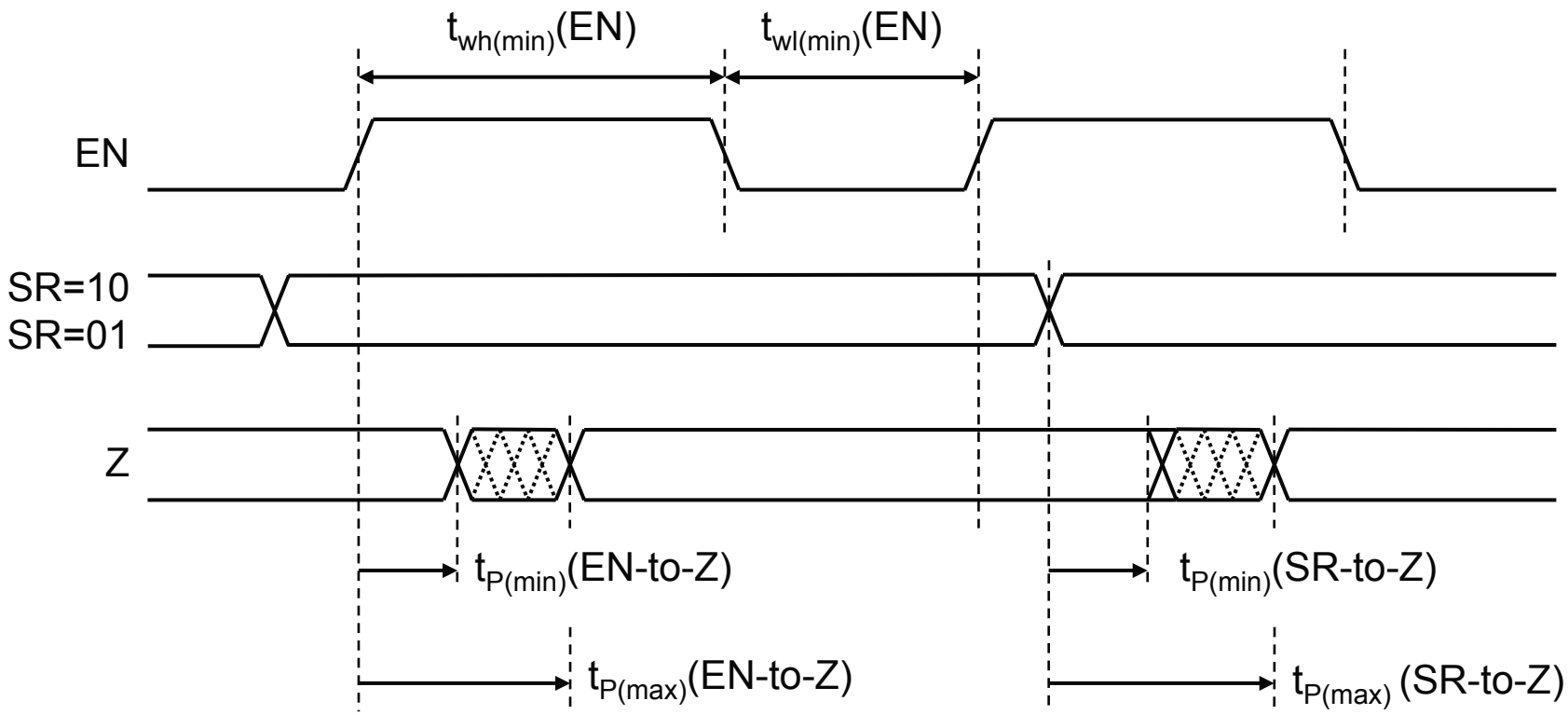
- SR is stabiel en EN gaat van 0 naar 1
  - De timing van de uitgang is gerelateerd aan EN

- SR verandert tijdens EN = 1
  - De timing van de uitgang is gerelateerd aan de combinatie van SR.

- Dat levert vier nieuwe vertragingstijden op.

Timing gated SR-latch

- Timing van de uitgang voor SR = 01 en SR = 10*.



* SR=00 levert geen verandering op de uitgang.

Timing gated D-latch

- De timing van een gated D-latch is complex.
- Er zijn twee situaties te onderscheiden:

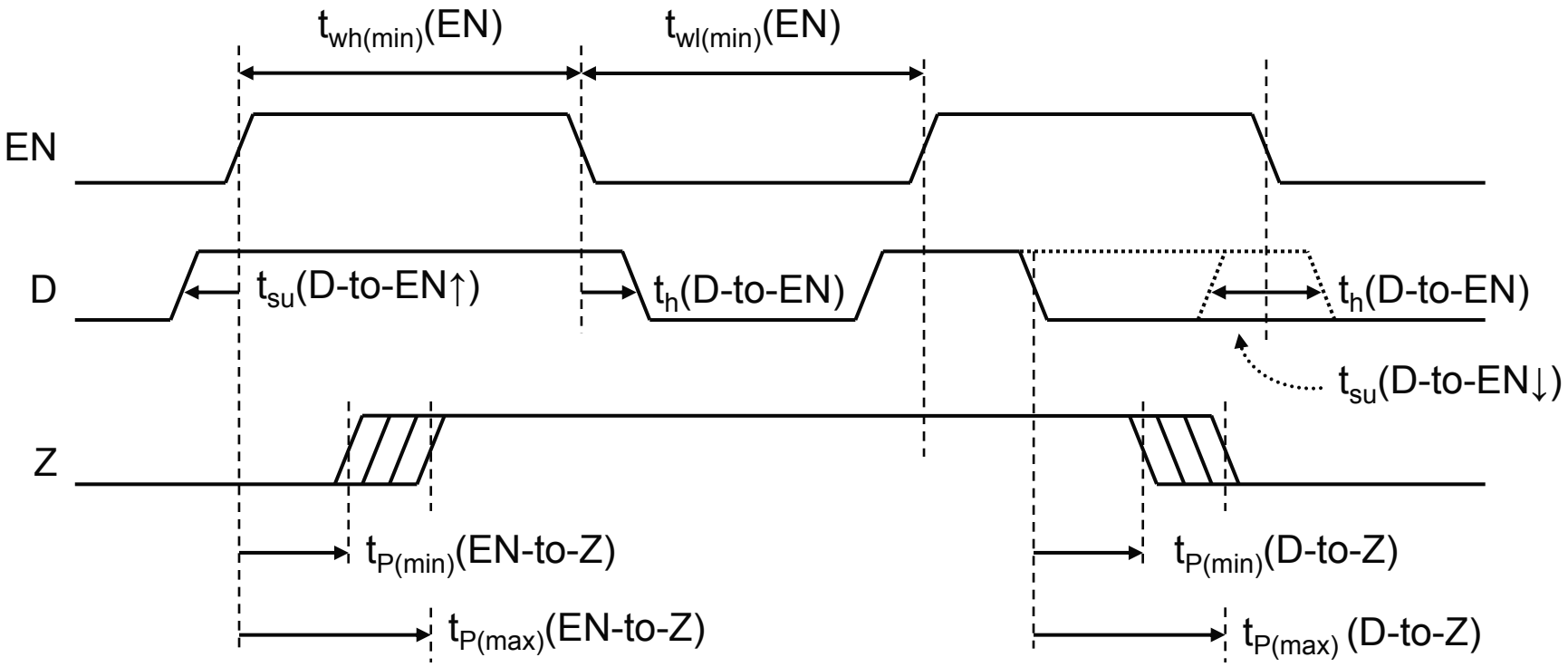
  D heeft een stabiele waarde tijdens EN = 1. In de praktijk moet D stabiel zijn van net iets voor EN 0→1 tot net iets na EN 1→0. De latch neemt de waarde van D over op het moment dat EN van 0 naar 1 gaat.

  EN = 1 en D verandert (de latch is transparant). De latch neemt de waarde van D over als D verandert.

- Dit levert twee sets vertragingstijden op.

Timing gated D-latch

- Het geheel in een timingdiagram.



Timing gated D-latch

- Er zijn twee nieuwe tijden bijgekomen.
- De *setuptijd* t_{su} is de tijd waarbij de waarde van de D-ingang stabiel moet zijn vóórdat EN van 0 naar 1 of van 1 naar 0 gaat.
- De *holdtijd* t_h is de tijd waarbij de waarde van de D-ingang stabiel moet blijven nádat EN van 1 naar 0 gaat.
- In het gebied t_{su}-t_h moet D dus stabiel blijven.
- Gaan we ervan uit dat D stabiel is tijdens EN = 1, dan heeft de gated D-latch een *pulse-triggered* timing.

Timing gated D-latch

- De tijden op een rijtje

$t_{P(min)}$(EN-to-Z)	De minimale vertragingstijd van de latch gezien vanuit EN.
$t_{P(max)}$(EN-to-Z)	De maximale vertragingstijd van de latch gezien vanuit EN.
$t_{P(min)}$(D-to-Z)	De minimale vertragingstijd van de latch gezien vanuit D.
$t_{P(max)}$(D-to-Z)	De maximale vertragingstijd van de latch gezien vanuit D.
t_{su}(D-to-EN↑)	De setuptijd van de D-ingang t.o.v. opgaande flank EN.
t_{su}(D-to-EN↓)	De setuptijd van de D-ingang t.o.v. neergaande flank EN.
t_{h}(D-to-EN)	De holdtijd van de D-ingang t.o.v. EN.
$t_{w(min)}$(EN)	De minimale pulsduur van EN.

D-flipflop

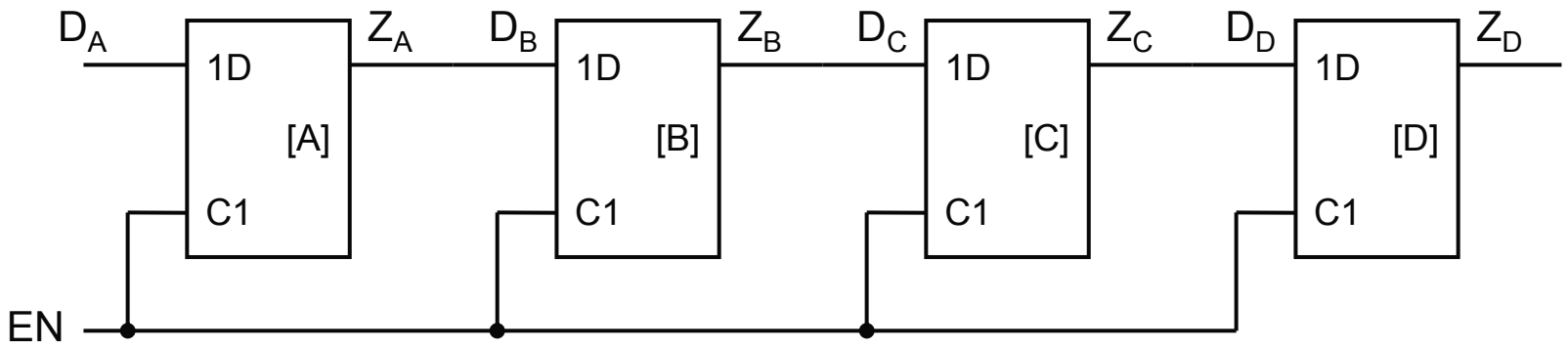
- De gated D-latch heeft als nadeel dat D stabiel moet worden gehouden zolang EN = 1.

- Op basis van deze latch is het niet makkelijk om data van één latch in een andere over te brengen.

- Een veel voorkomende schakeling is een *schuifregister*. Hierbij worden een aantal identieke geheugenelement zó geschakeld dat de data geschoven kan worden onder besturing van een enablesignaal.

D-flipflop

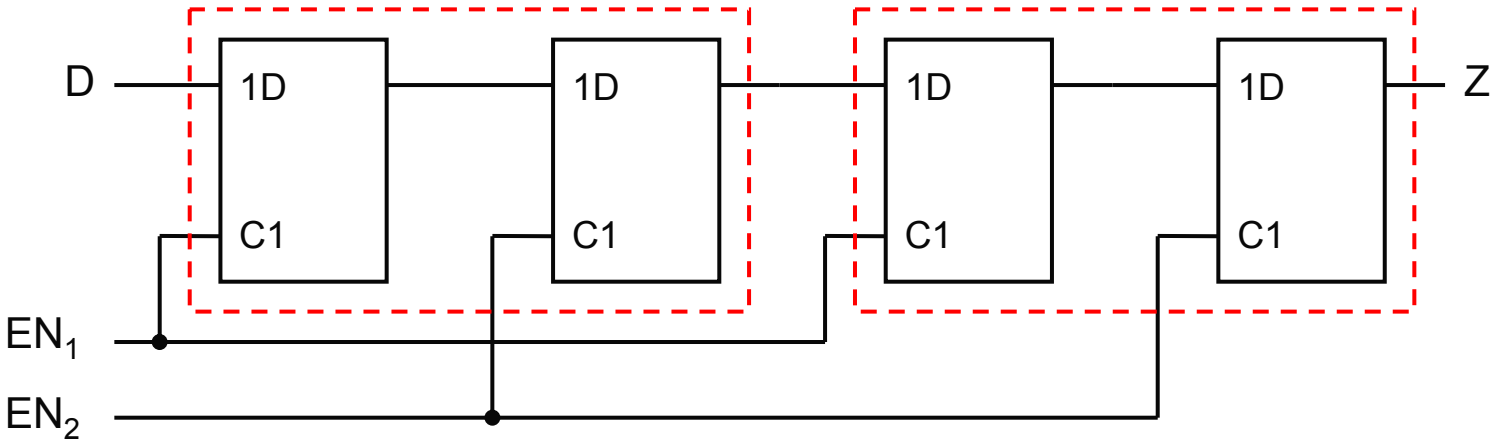
- Onderstaande is een poging tot een schuifregister.



- Dit gaat fout als EN van 0 naar 1 gaat. *Alle* latches zijn dan *transparant* en data op D_A wordt direct naar Z_D doorgevoerd.

D-flipflop

- Dit transparant zijn kan worden tegengehouden door twee latches afwisselend te plaatsen, elk met een eigen EN. Er zijn dan ook twee EN-signalen nodig.



- Voorwaarde voor juiste werking: EN1 en EN2 mogen elkaar niet overlappen.

D-flipflop

- Als EN1 = 0 en EN2 = 0 onthouden alle latches. Geen enkele latch is transparant.

- Als EN1 = 1 en EN2 = 0 zal de eerste latch data overnemen van ingang D en de derde latch data overnemen van de tweede latch. De eerste en derde latch zijn transparant. De inhoud van de tweede en vierde latch zijn ongewijzigd.

- Als EN1 = 0 en EN2 = 1 zal de tweede latch data overnemen van de eerste latch en de vierde latch data overnemen van de derde latch. De inhoud van de eerste en derde latch zijn ongewijzigd.

- Voor het maken van een 4-bit schuifregister zijn dus acht latches nodig.

D-flipflop

- Een geheugenelement op basis van twee latches die niet tegelijk transparant zijn wordt een *flipflop* genoemd, in dit geval een D-flipflop.

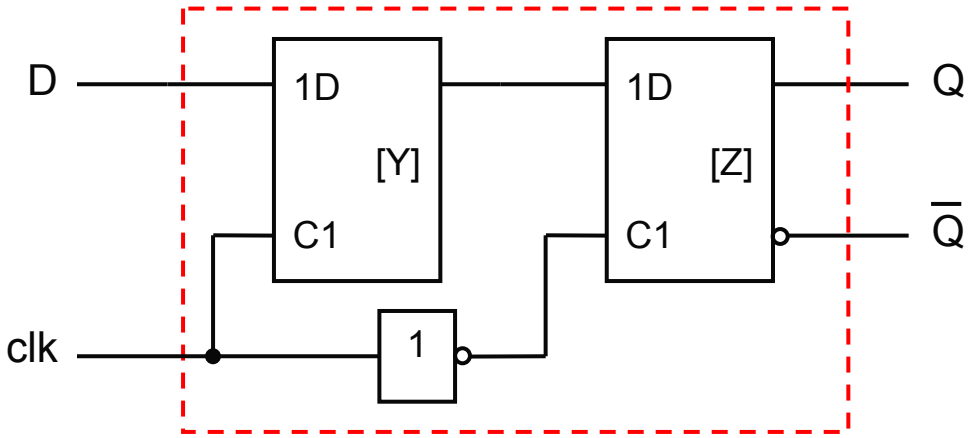
- Door de constructie van de flipflop met twee latches wordt dit meester-en-slaaf (*master-slave*) genoemd*).

- De schakeling kan nog wat eenvoudiger worden door het gebruik van een NOT-poort zodat $EN_2 = \overline{EN_1}$

nb: maar welke is nou de master en welke de slave?

Opbouw master-slave D-flipflop

- Hieronder de gerealiseerde D-flipflop. De meeste (losse) flipflops hebben ook een inverse uitgangswaarde beschikbaar.

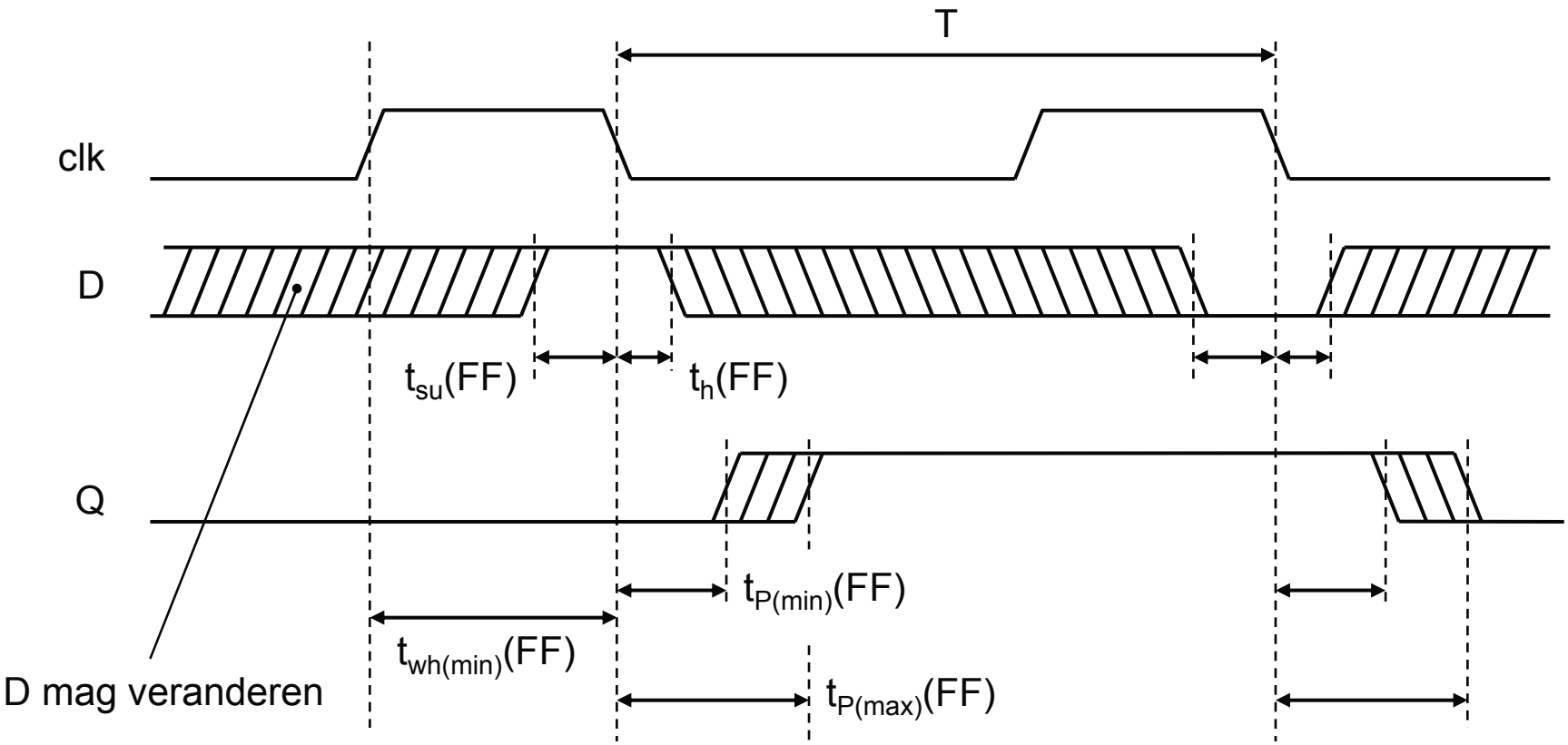


- Voorwaarde voor betrouwbare werking:

  $t_{P(max)}(NOT)+t_h(Z\text{-latch}) < t_{P(min)}(Y\text{-latch})$

Timing master-slave D-flipflop

- Hieronder de timing van de D-flipflop.

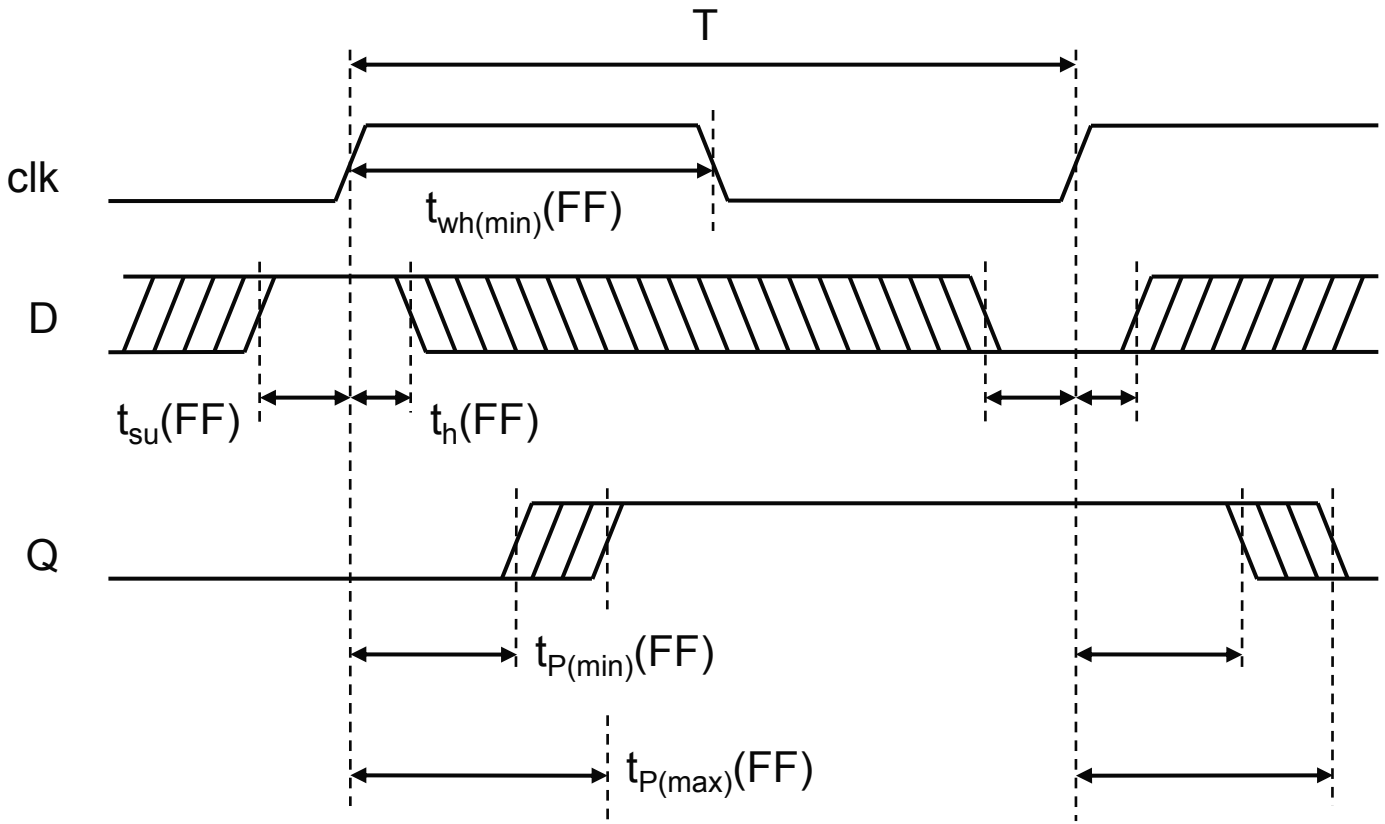


Edge triggered timing

- Te zien is dat de D-ingang een stabiele waarde moet hebben rond de neergaande flank.
- Het timing-voorschrift wordt *flankgestuurde* of *edge-triggered timing* genoemd.
- Bij een edge-triggered timing refereren alle timing-parameters aan dezelfde flank van het kloksignaal. Dit wordt de actieve flank genoemd.
- Het tijdinterval waarin de data klaar moet staan is in de regel zeer klein, in de orde van enkele ns.

Timing positive edge-triggered D-flipflop

- Hieronder de timing van de positive edge-triggered D-flipflop.



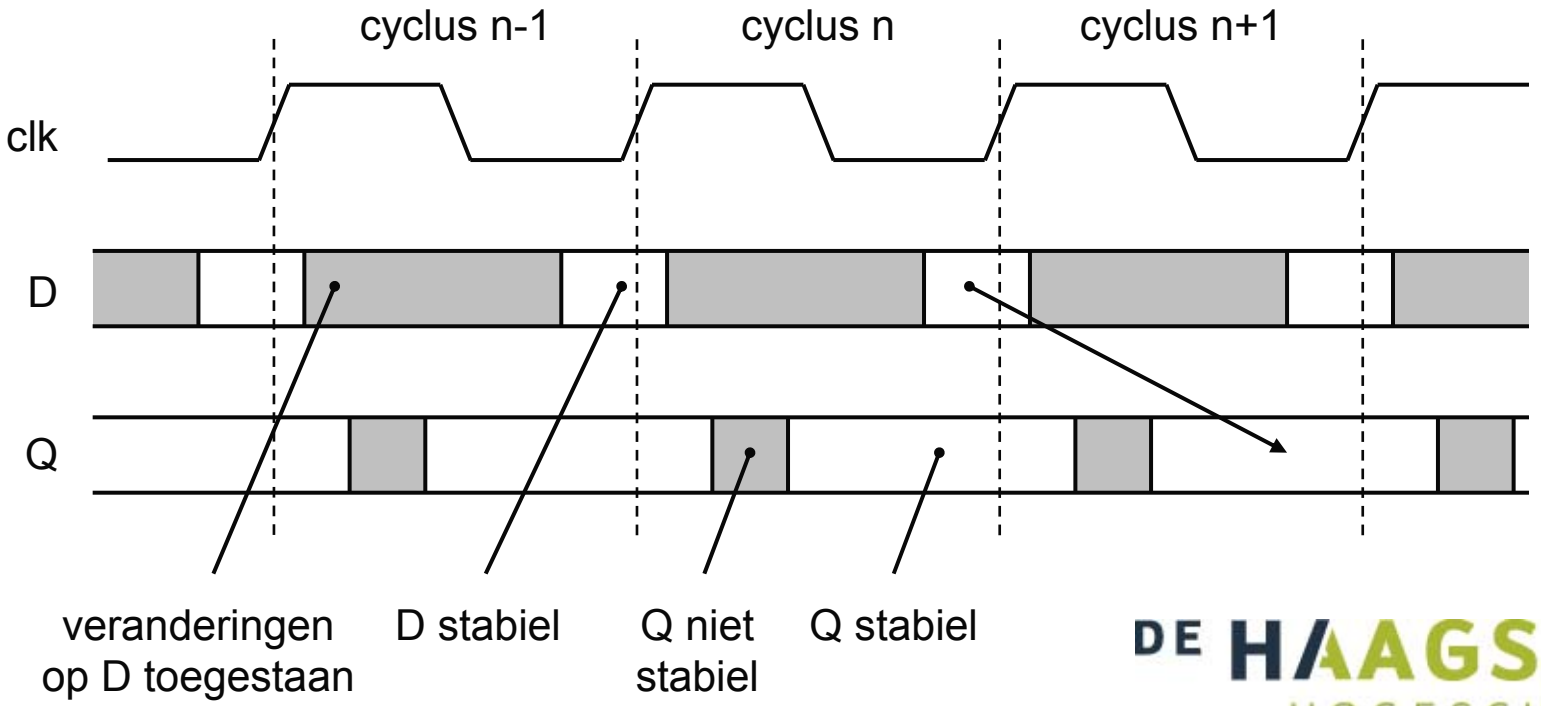
Timing edge-triggered D-flipflop

- De tijden op een rijtje

$t_{P(min)}(FF)$	De minimale vertragingstijd van de uitgang van de flipflop t.o.v. de actieve klokflank.
$t_{P(max)}(FF)$	De maximale vertragingstijd van de uitgang van de flipflop t.o.v. de actieve klokflank.
$t_{su}(FF)$	De setuptijd van de D-ingang van de flipflop t.o.v. de actieve klokflank.
$t_{h}(FF)$	De holdtijd van de D-ingang van de flipflop t.o.v. de actieve klokflank.
$t_{wh(min)}(FF)$	De minimale pulsbreedteduur van het kloksignaal.
T	De periodeduur van het kloksignaal.

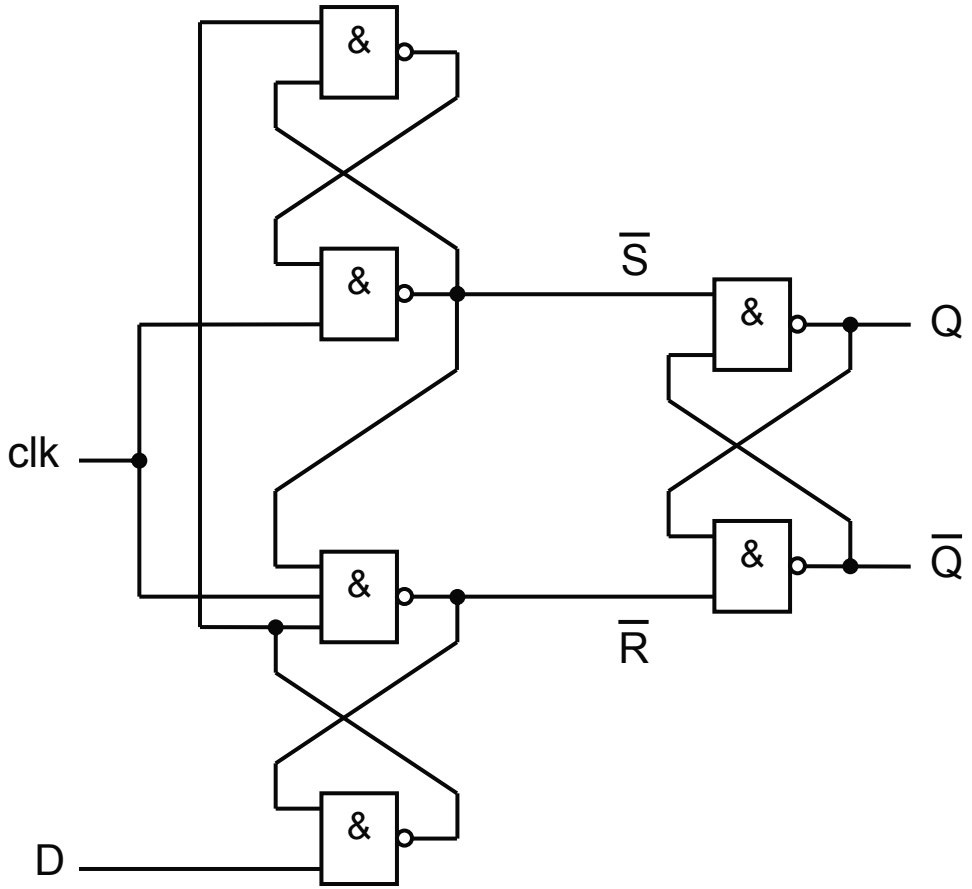
Functie edge-triggered D-flipflop

- In de functie van de D-flipflop wordt het kloksignaal *clk* niet meer expliciet meegenomen. Om aan te geven dat een huidige waarde van D pas ná de klokflank op Q beschikbaar is, wordt gebruik gemaakt van de volgende schrijfwijze: $Q^{n+1} = D^n$



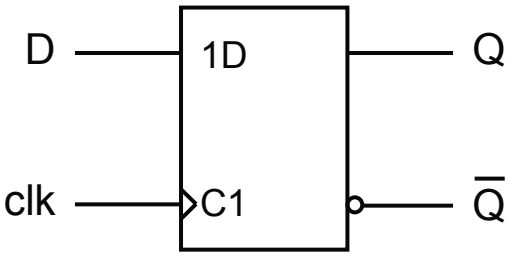
Edge-triggered flipflop

- Er zijn ook edge-triggered flipflop te construeren bestaande uit drie SR-latches. Deze werken niet op de bekende master-slave wijze.
- Het gaat hier om triggering op een bepaalde spanning.
- Hiernaast is de edge-triggered flipflop te zien die uitgevoerd is in de bekende TTL-serie.



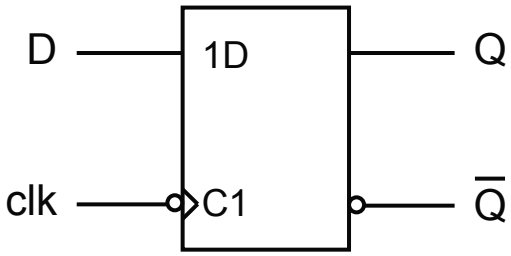
Symbolen edge-triggered D-flipflop

- Hieronder de symbolen van de D-flipflop.



positive edge-triggered
D-flipflop

werking D afhankelijk
van de opgaande flank
van C



negative edge-triggered
D-flipflop

werking D afhankelijk
van de neergaande flank
van C

Flipflop ontwerpen

- Het ontwerpen van een betrouwbare flipflop is een lastige aangelegenheid en moet gedaan worden door ervaren ontwerpers.

- Het zelf ontwerpen van een flipflop met losse poorten of in een beschrijvingstaal als VHDL is niet aan te raden.

- Bij gebruik van *reconfigureerbare logica* (FPGA, CPLD) is het helemaal niet gewenst, de fabrikant heeft al kant-en-klare flipflops aangebracht. Door gebruik van *library modules* kan een flipflop gerealiseerd worden.

Flipflop

- Voor de logische werking van een schakeling maakt het niet uit op welke flank een flipflop z'n data inklokt.
- Het door elkaar gebruiken van positive en negative edge-triggered flipflops is meestal niet de bedoeling.
- Het gebruik van flipflops met verschillende timing-parameters levert problemen op.
- Binnen één IC is dat echter niet het geval.

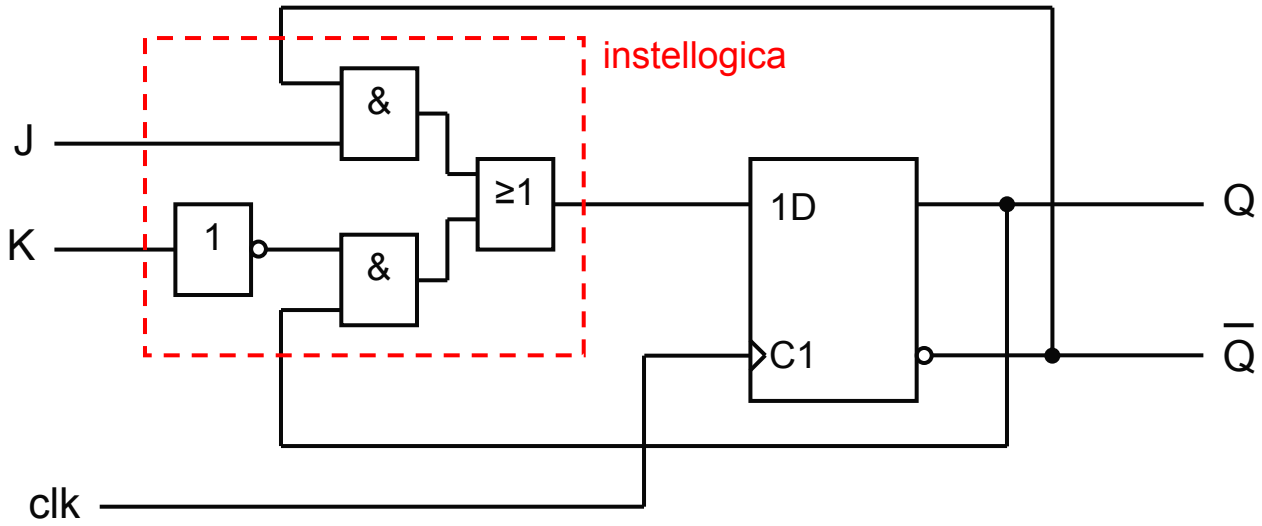
JK-flipflop

- In de praktijk komt nog een andere flipflop voor, de JK-flipflop.
- Deze flipflop heeft twee stuuringangen J en K en een klokingang.
- Rechts is de functietabel gegeven.
- Het idee is dat bij JK = 11 de uitgang van stand verandert. Zo zijn er eenvoudig tweedelers te maken.
- De functie is $Q^{n+1} = J^n \cdot \overline{Q^n} + \overline{K^n} \cdot Q^n$

J	K	Q^{n+1}
0	0	Q^n
0	1	reset
1	0	set
1	1	$\overline{Q^n}$

JK-flipflop

- Oude ontwerpen gingen uit van SR-flipflops. Deze hadden echter slechte timingeigenschappen.
- Tegenwoordig worden JK-flipflops gemaakt op basis van D-flipflops en *instellogica*.



Asynchrone reset en set

- Bij het opkomen van de voedingsspanning is het onduidelijk wat de stand van een flipflop wordt.
- Vandaar dat flipflops worden voorzien van twee *asynchrone* ingangen die de stand onafhankelijk van de klok vastleggen.
- De *reset*-ingang zorgt ervoor dat de stand van de flipflop logisch 0 wordt.
- De *set*-ingang zorgt ervoor dat de stand van de flipflop logisch 1 wordt.

Asynchrone reset en set

- Deze asynchrone reset- en set-ingangen zijn storingsgevoelig en moeten goed afgeschermd worden.

- Deze ingangen moeten alleen gebruikt worden voor *power-on reset en set*, nooit sturen met combinatorische logica.

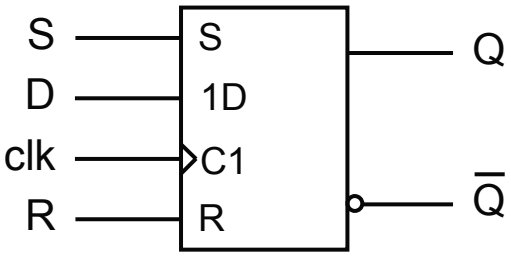
- Voor reset en set gelden aanvullende timing eisen zoals pulsduur op ingangen en de *recovery time* en *removal time*. Dit is de tijd dat de twee ingangen inactief moeten zijn voordat de actieve klokflank passeert.

recovery time: de tijd tussen het moment dat het asynchrone signaal inactief wordt en de actieve klokflank, werkt als een setup-tijd voor asynchrone signalen.

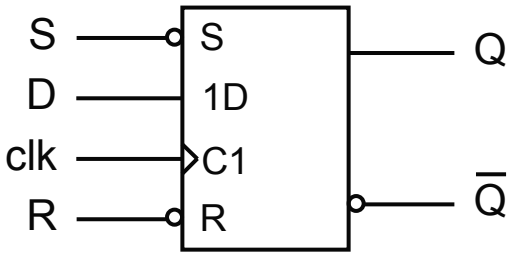
removal time: de tijd tussen de actieve klokflank en het moment dat het asynchrone signaal inactief wordt, werkt als een hold-tijd voor asynchrone signalen.

Asynchrone reset en set

- Hieronder de symbolen van de D-flipflop met asynchrone reset en set.



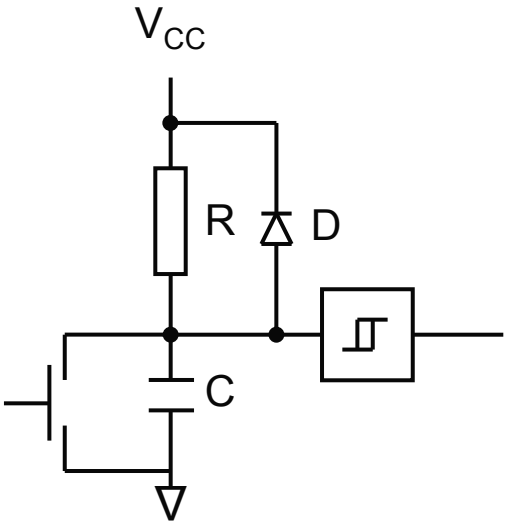
positive edge-triggered D-flipflop with asynchronous active high set and reset



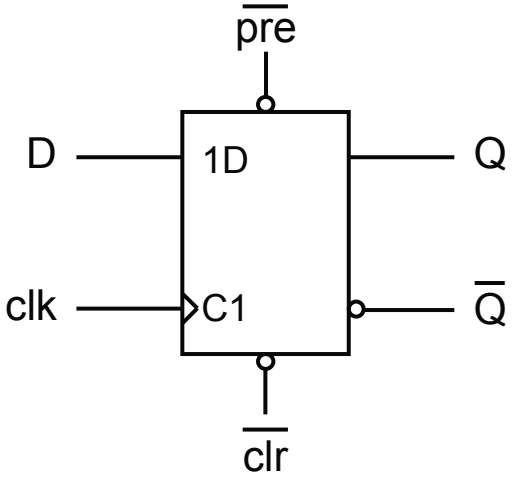
positive edge-triggered D-flipflop with asynchronous active low set and reset

Asynchrone reset en set

- In Amerikaanse literatuur wordt doorgaans het onderstaande symbool gebruikt. Vaak zijn preset- en clear-ingangen zijn *actief laag*.



power-on reset (active low) met gebruikersreset en Schmitt-trigger-uitgang.



positive edge-triggered D-flipflop met asynchrone active low preset en clear

D-flipflop met enable

- Soms moet de stand van een flipflop behouden blijven “over de klokflanken heen”, dus de stand verandert niet ondanks dat er een actieve klokflank passeert.

- Dat kan op twee manieren:

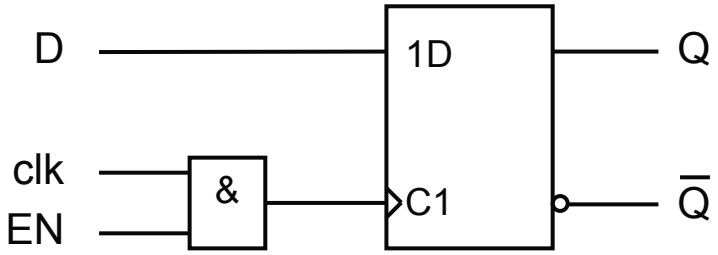
  Schakel de klok uit zodat geen klokflank passeert.

  Zorg ervoor dat de flipflop zijn eigen stand opnieuw inklokt.

- Dit kan beide met een enable-signaal.

D-flipflop met enable

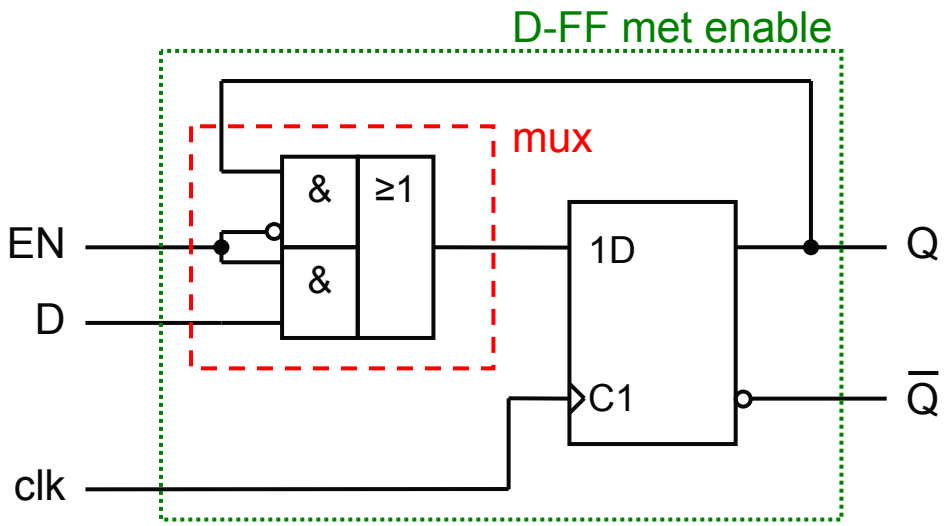
- Oplossing: schakel de klok uit.



- Het enable-signaal mag niet veranderen tijdens clk = 1!
- Het resultaat is een flipflop met pulse-triggered eigenschappen.
- **Dit is dus ab-so-luut verboden en een slecht ontwerp.**

D-flipflop met enable

- Oplossing: flipflop klokt zijn eigen waarde óf een nieuwe waarde in door middel van een multiplexer.

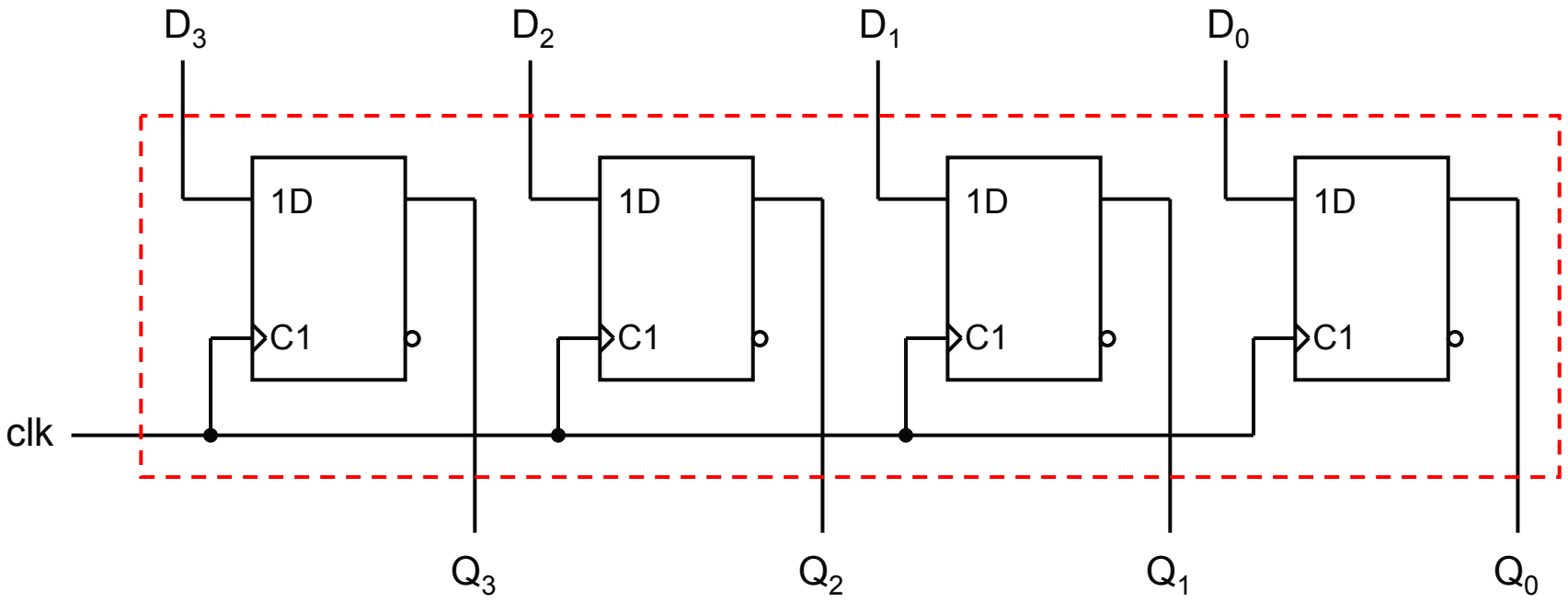


$$Q^{n+1} = \overline{EN^n} \cdot Q^n + EN^n \cdot D^n$$

- Er is wel meer logica nodig dan in het vorige ontwerp, maar het resultaat is een edge-triggered flipflop.

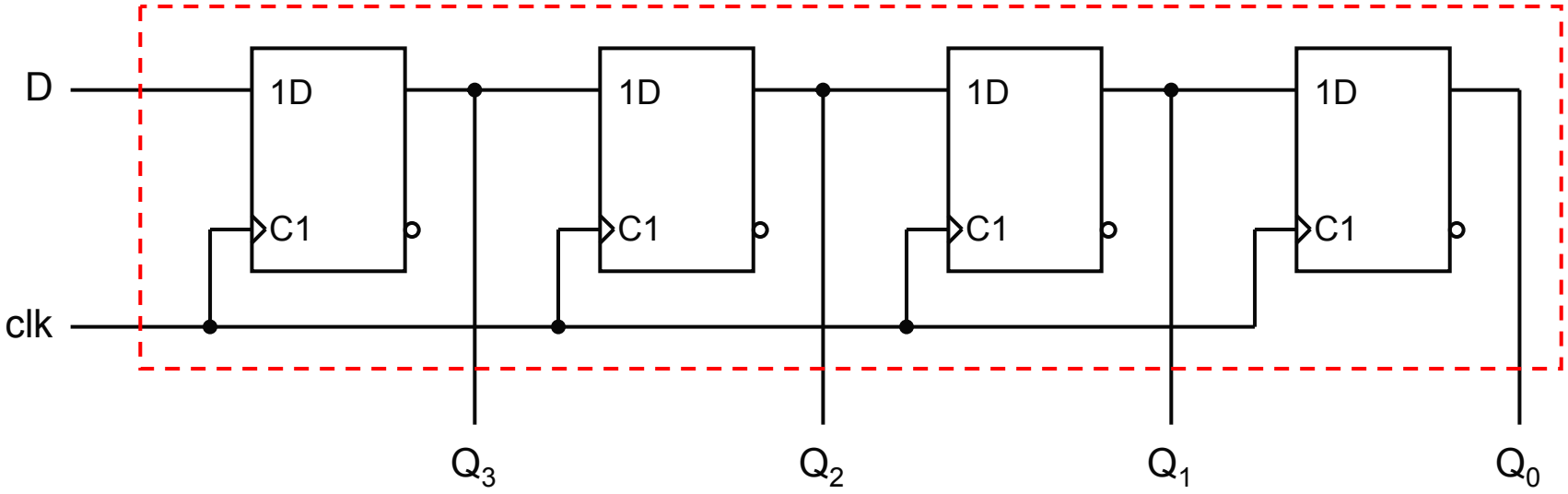
Register

- Een register is een groep (D-)flipflops die alle getriggerd worden op hetzelfde kloksignaal, soms met enable-faciliteit.



Schuifregister

- Een schuifregister is een groep (D-)flipflops waarvan de data-uitgang van een flipflop is verbonden met de data-ingang van de naastgelegen flipflop. Schuifregisters spelen een belangrijke rol in seriële communicatie. Hieronder een Serial-In-Parallel-Out (SIPO)-schuifregister.



VHDL-beschrijving D-flipflop

```
library ieee;
use ieee.std_logic_1164.all;

entity dff is
    port (CLK : in std_logic;
          D : in std_logic;
          Q : out std_logic
         );
end dff;

architecture behavioral of dff is
begin
    -- process alléén afhankelijk van CLK!
    P0: process (CLK) is
    begin
        -- CLK is veranderd én CLK is '1', dan opgaande flank!
        if CLK'event and CLK = '1' then
           Q <= D;
        end if;         -- geen else!
    end process P0;
end behavioral;
```

Hiernaast is de VHDL-beschrijving van een D-flipflop. Let vooral op de constructie voor het beschrijven van de klokflank.

Opgaven

- Een *tweedeler* is een schakeling die op commando van een enablepuls geacht wordt één maal van uitgangswaarde te veranderen. Als de uitgang 1 is wordt dit 0 en omgekeerd. Is een tweedeler te maken met één latch en poorten? En met twee latches en poorten?

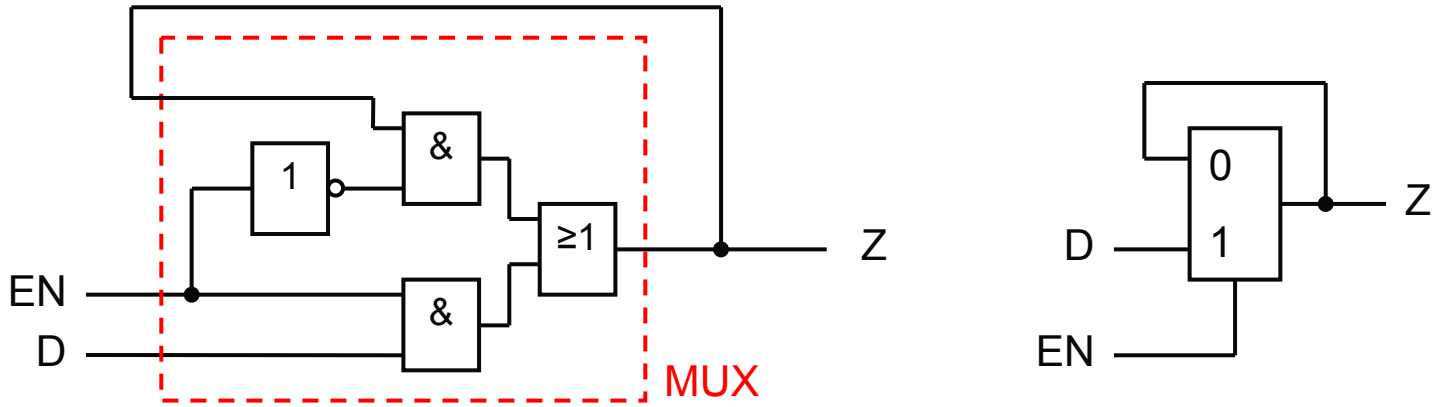
- Op één van de slides staat dat moet gelden

$$t_{P(max)}(\text{NOT}) + t_h(\text{Z-latch}) < t_{P(min)}(\text{Y-latch})$$

Toon dit aan met een timingdiagram en verklaar deze.

Opgaven

- Een gated D-latch voldoet aan de volgende functie:
  $Z_{nieuw} = \overline{EN} \cdot Z_{oud} + EN \cdot D$
  en kan worden gebouwd met behulp van een *multiplexer*.



Toon aan dat dataoverdracht *niet* betrouwbaar verloopt. Hint: bekijk de situatie Z=1, D=1 en EN 1→0.

Opgaven

- Probeer een double edge-triggered D-flipflop te ontwerpen. Dit is een flipflop die op *beide* flanken reageert.

- Een T-flipflop is een flipflop die van stand (waarde) verandert onder besturing van een stuursignaal T. Het (steeds weer) veranderen van stand wordt “toggelen” genoemd. Ontwerp deze T-flipflop, er mag uiteraard *niet* geschakeld worden in de kloklijn.

- Eerder is een ontwerp van een schuifregister aan bod gekomen. Deze schakeling schuift (althans op tekening) naar rechts. Ontwerp nu een schuifregister dat zowel rechtsom als linksom kan schuiven (gebruik een stuursignaal).

Opgaven

- De functie van een JK-flipflop is $Q^{n+1} = J^n \cdot \overline{Q^n} + \overline{K^n} \cdot Q^n$
  Toon dit aan met behulp van een Karnaughdiagram.

- Van een latch zijn gegeven t_{su}(EN-to-Z) = 15 ns, t_h(EN-to-Z) = 10 ns. De klok loopt op 5 MHz en heeft een *duty cycle* van 50%. Wat is de tijd dat D stabiel moet blijven?

- Maak van een edge-triggered D-flipflop met asynchrone preset en clear een SR-latch.

Literatuur

- Digitale Techniek, van probleemstelling tot realisatie deel 2, A.P. Thijssen, ISBN 978-90-407-1838-0
- Switch bounce – https://www.youtube.com/watch?v=xadoeJEhTJU
- 7474 positive edge triggered D-flipflop – http://www.ti.com/lit/ds/symlink/sn54s74.pdf
- Dictaat hoofdstuk 6.

Academie voor Technology, Innovation & Society Delft

Academie voor ICT & Media

De Haagse Hogeschool, Delft
+31-15-2606311
J.E.J.opdenBrouw@hhs.nl
www.dehaagsehogeschool.nl

DE HAAGSE HOGESCHOOL