Kapitel 5

Timing:

1. Physikalische Eigenschaften
2. **Timing wichtiger Komponenten**
3. Exaktes Timing von ReTI

Albert-Ludwigs-Universität Freiburg

UNI FREIBURG

Dr. Tobias Schubert, Dr. Ralf Wimmer

Professur für Rechnerarchitektur

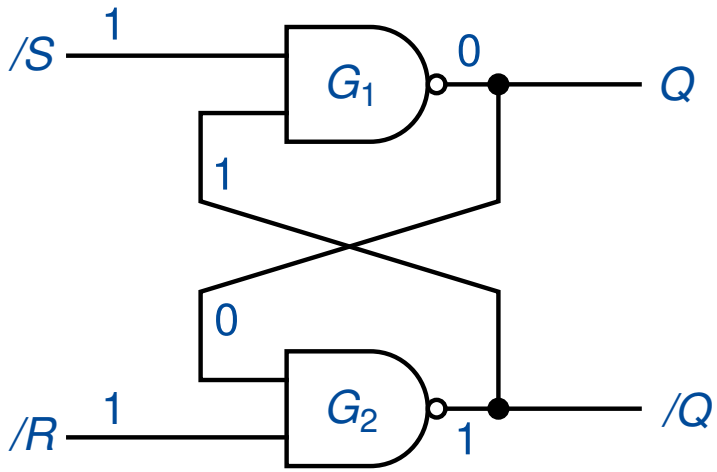
WS 2016/17

Timing - Übersicht

- Timing für ein paar (bereits bekannte) Schaltpläne:
  - RS-Flipflop
  - D-Latch
  - D-Flipflop
- Timing weiterer Komponenten, die bei der Realisierung der ReTI genutzt werden:
  - Kontrolllogik
  - Register mit Clock-Enable
  - ALU
  - Speicher

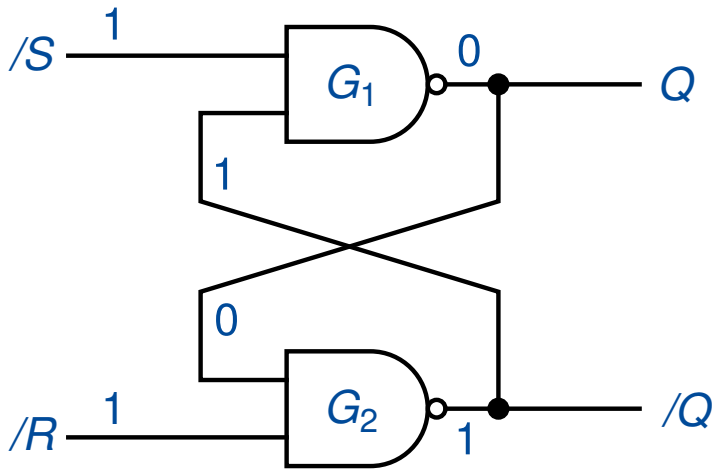
RS-Flipflop

- Zustand $Q = 0 \rightarrow$ Zustand $Q = 1$:



RS-Flipflop

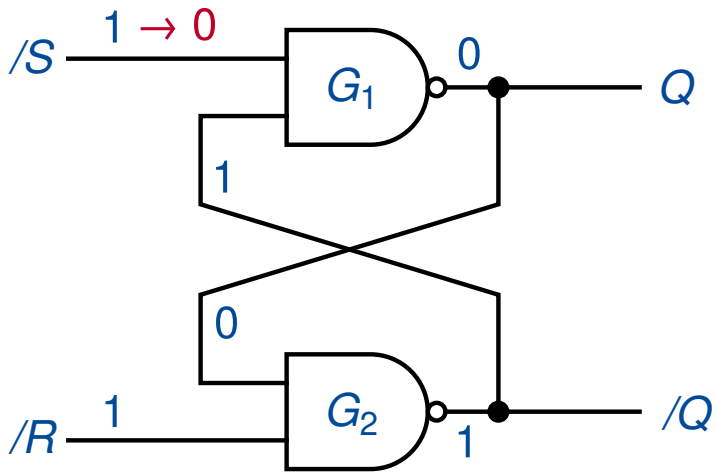
- Zustand $Q = 0 \rightarrow$ Zustand $Q = 1$:



- Senke $/S$ zur Zeit t_0 ab und hebe zu $t_0 + x$ wieder an (einen solchen Signalverlauf nennt man Puls).

RS-Flipflop

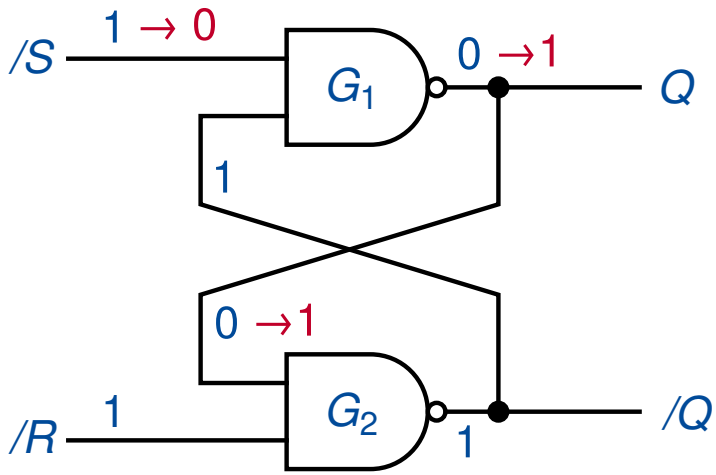
- Zustand $Q = 0 \rightarrow$ Zustand $Q = 1$:



- Senke $/S$ zur Zeit t_0 ab und hebe zu $t_0 + x$ wieder an (einen solchen Signalverlauf nennt man Puls).

RS-Flipflop

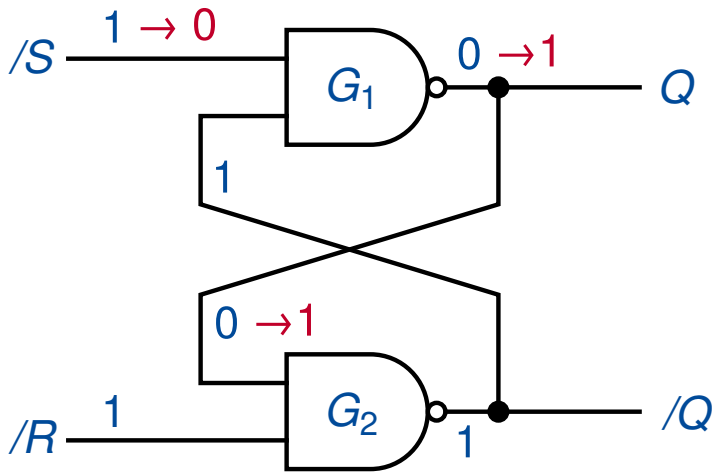
- Zustand $Q = 0 \rightarrow$ Zustand $Q = 1$:



- Senke $/S$ zur Zeit t_0 ab und hebe zu $t_0 + x$ wieder an (einen solchen Signalverlauf nennt man Puls).

RS-Flipflop

- Zustand $Q = 0 \rightarrow$ Zustand $Q = 1$:

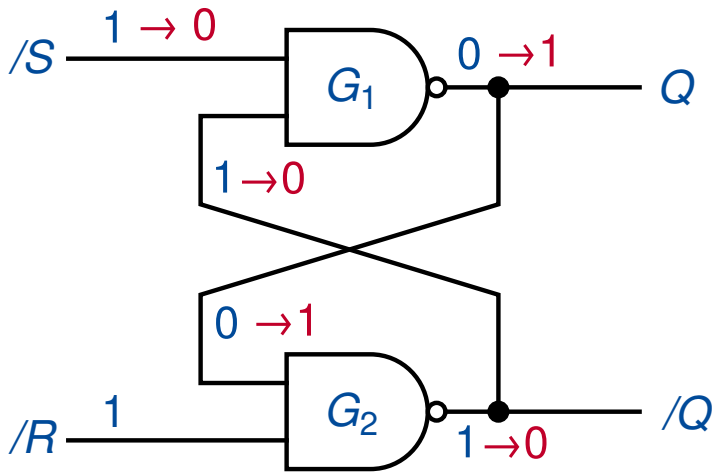


- Senke $/S$ zur Zeit t_0 ab und hebe zu $t_0 + x$ wieder an (einen solchen Signalverlauf nennt man Puls).

- Nach Zeit $t_{P/SQ}$ ist $Q = 1$.

RS-Flipflop

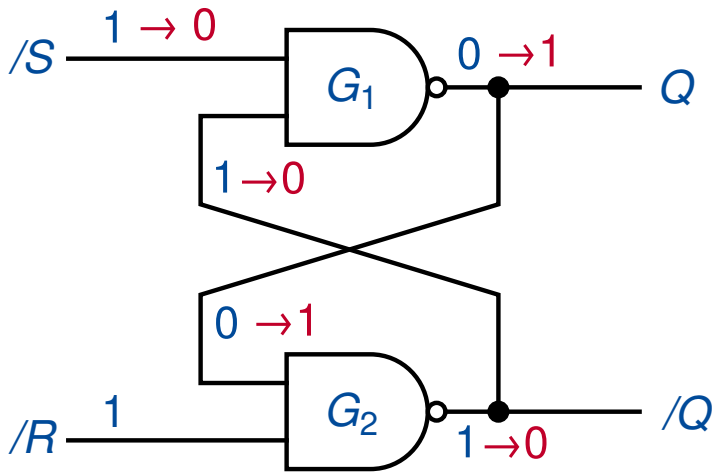
- Zustand $Q = 0 \rightarrow$ Zustand $Q = 1$:



- Senke $/S$ zur Zeit t_0 ab und hebe zu $t_0 + x$ wieder an (einen solchen Signalverlauf nennt man Puls).
- Nach Zeit $t_{P/SQ}$ ist $Q = 1$.

RS-Flipflop

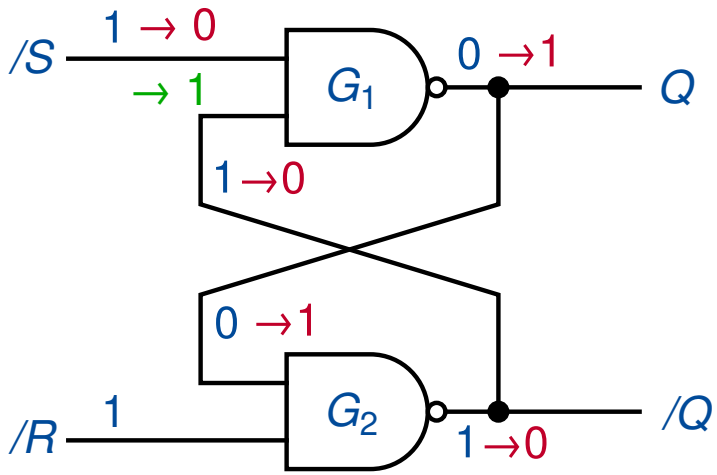
- Zustand $Q = 0 \rightarrow$ Zustand $Q = 1$:



- Senke $/S$ zur Zeit t_0 ab und hebe zu $t_0 + x$ wieder an (einen solchen Signalverlauf nennt man Puls).
- Nach Zeit $t_{P/SQ}$ ist $Q = 1$.
- Nach Zeit $t_{P/S/Q}$ ist $/Q = 0$.

RS-Flipflop

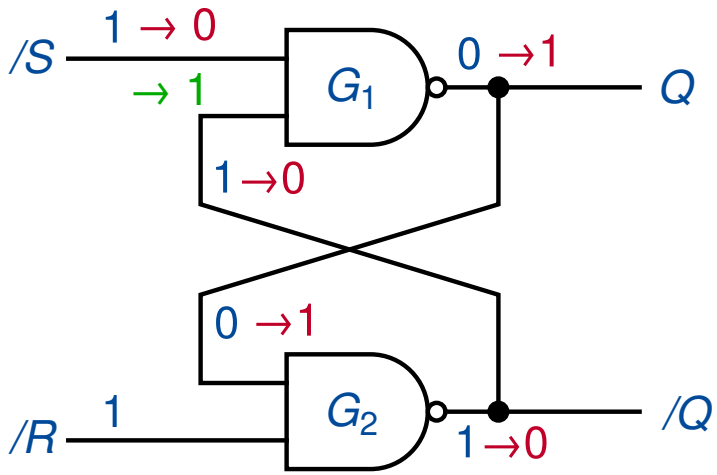
- Zustand $Q = 0 \rightarrow$ Zustand $Q = 1$:



- Senke $/S$ zur Zeit t_0 ab und hebe zu $t_0 + x$ wieder an (einen solchen Signalverlauf nennt man Puls).
- Nach Zeit $t_{P/SQ}$ ist $Q = 1$.
- Nach Zeit $t_{P/S/Q}$ ist $/Q = 0$.

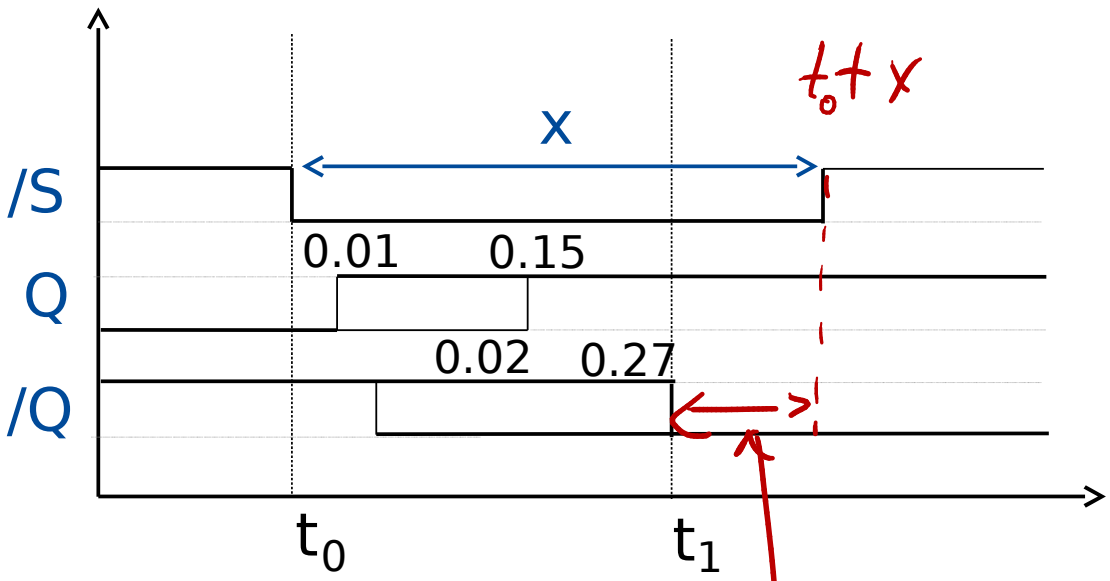
RS-Flipflop

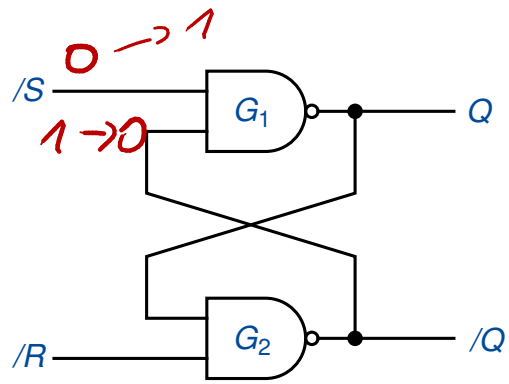
- Zustand $Q = 0 \rightarrow$ Zustand $Q = 1$:



- Senke $/S$ zur Zeit t_0 ab und hebe zu $t_0 + x$ wieder an (einen solchen Signalverlauf nennt man Puls).
- Nach Zeit $t_{P/SQ}$ ist $Q = 1$.
- Nach Zeit $t_{P/S/Q}$ ist $/Q = 0$.
- Wähle x so, dass kein Spike entsteht.

Übergang - graphisch





NAND	t^{min}	t^{max}
τ_{PLH}	0.01	0.15
τ_{PHL}	0.01	0.12

G_1

G_2

muss spikefreies Umschalten von G_1 gewährleisten !!!

Spikefreier Übergang

- Nach den Regeln des spikefreien Umschaltens von Gattern entsteht kein Spike, falls:

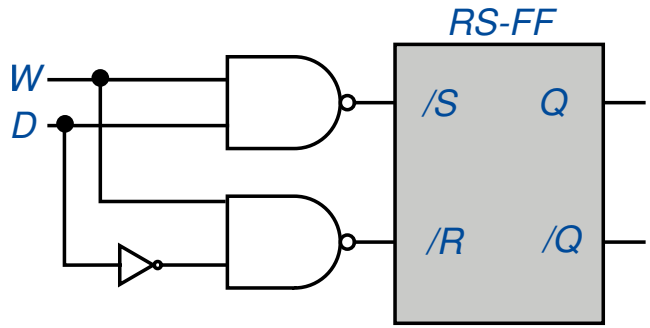
$$(t_0 + x) - (t_0 + 0.27) \geq 0.41 \Leftrightarrow x \geq 0.68ns$$

- Wechsel von Zustand $Q = 1$ zu Zustand $Q = 0$ aus Symmetriegründen analog.

Symbole und Bezeichnungen

Symbol	Bezeichnung	$t^{\min}$	$t^{\max}$
x	Pulsweite	0.68	
$\tau_{P/SQ}$	Verzögerungszeit von $/S$ bis Q	0.01	0.15
$\tau_{P/S/Q}$	Verzögerungszeit von $/S$ bis $/Q$	0.02	0.27
$\tau_{P/RQ}$	Verzögerungszeit von $/R$ bis Q	0.02	0.27
$\tau_{P/R/Q}$	Verzögerungszeit von $/R$ bis $/Q$	0.01	0.15

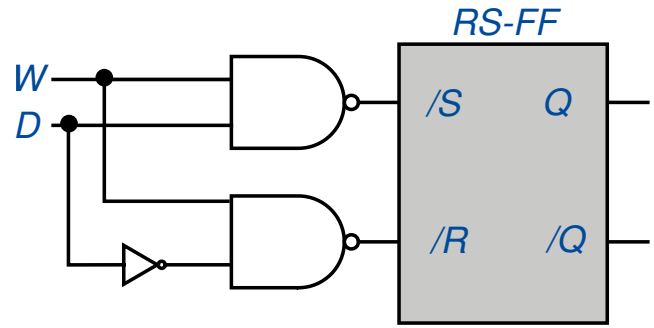
D-Latch

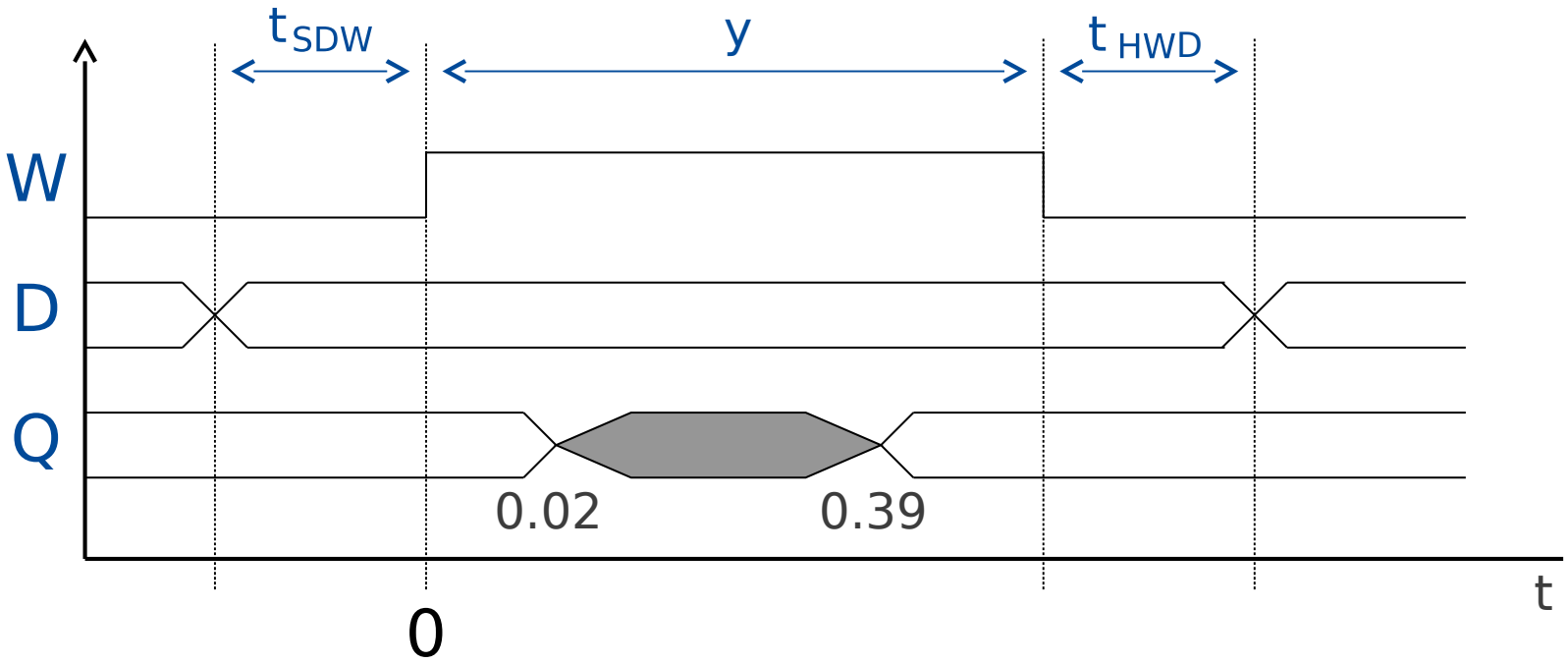


- W ist *active high*.
  - $W = 0 \Rightarrow /S, /R$ inaktiv
  - $W = 1 \Rightarrow \begin{cases} /S \text{ aktiv}, & \text{falls } D = 1 \\ /R \text{ aktiv}, & \text{falls } D = 0 \end{cases}$

- Wie beim RS-Flipflop (minimale Pulsweite!) muss man auch beim D-Latch bestimmte Forderungen an den zeitlichen Verlauf der Signale stellen, um Spikefreiheit zu garantieren.

Timing-Diagramm





Timing-Bedingungen für das D-Latch

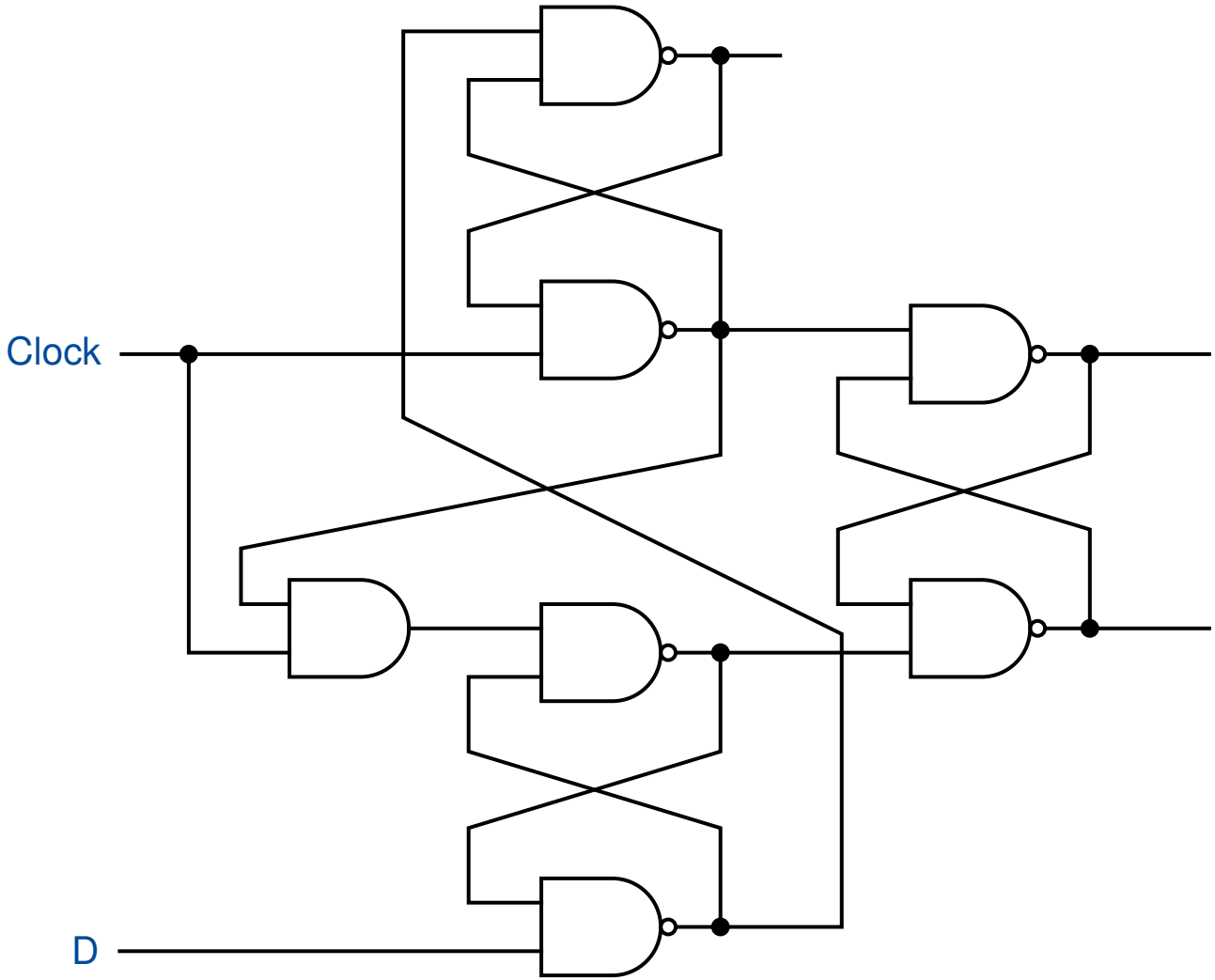
- W muss beim Schreiben lange genug 1 sein, um minimale Pulsweite x des RS-FFs zu garantieren.
- Vor $W : 0 \rightarrow 1$ werden Daten für Zeit t_{SDW} stabil gehalten, um Spikes auf $/S$, $/R$ zu vermeiden (der kritischste Fall ist das Verhindern von Spikes auf $/R$ bei Schreiben von 1).
- Nach $W : 1 \rightarrow 0$ werden Daten für Zeit t_{HWD} stabil gehalten, um Spikes auf $/S$, $/R$ zu vermeiden (der kritischste Fall ist das Verhindern von Spikes auf $/S$ beim Schreiben von 0).

Man rechnet nach:

Der Schreibvorgang beim D-Latch funktioniert mit den Parameterwerten aus der Tabelle (siehe Übung).

Symbol	**Bezeichnung**	t^{min}	t^{max}
y	Pulsweite des Schreibimpulses	0.79	
t_{SDW}	Setupzeit von D bis W	0.49	
t_{HDW}	Holdzeit von W nach D	0.41	
τ_{PWQ}	Verzögerungszeit von W bis Q	0.02	0.39
(τ_{PDQ}	Verzögerungszeit von D bis Q	0.02	0.54)

Mögliche Realisierung: D-Flipflop

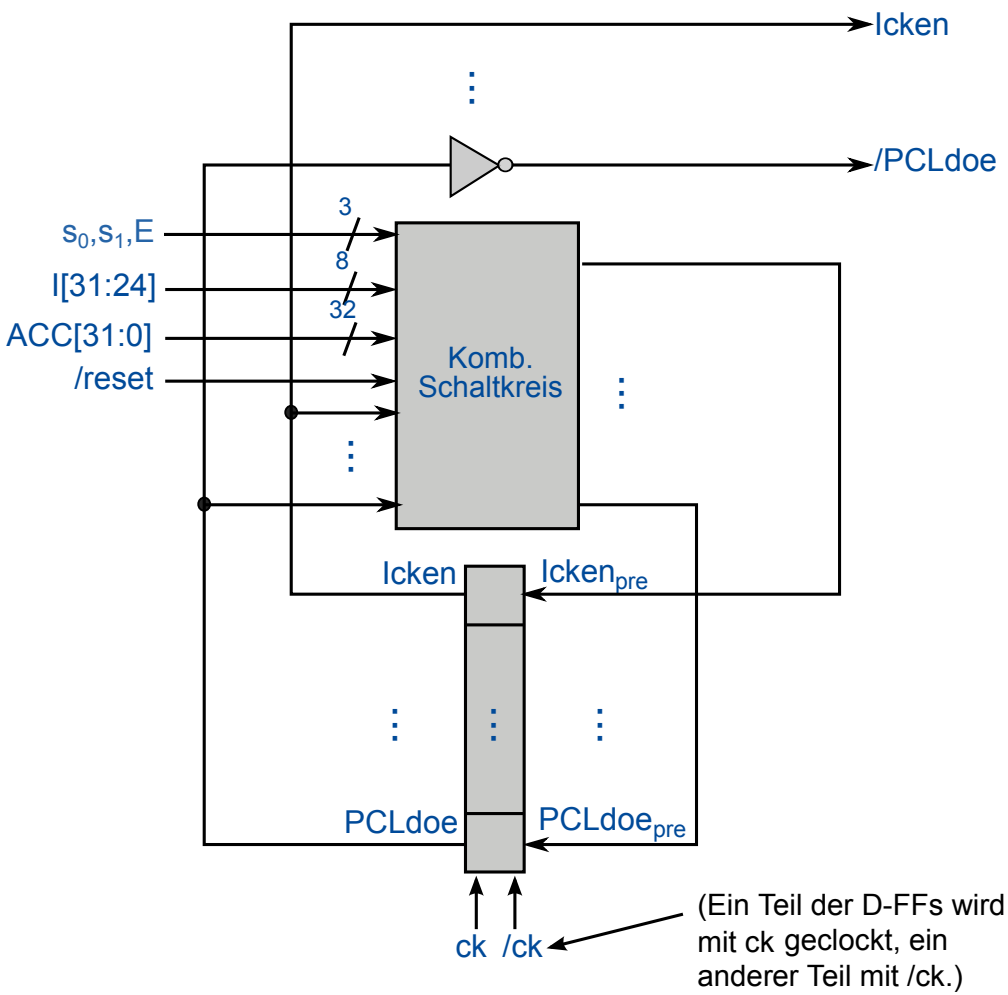


Timing: D-Flipflop

- Vorgehen analog zu RS-Flipflop und D-Latch.
- Wir verzichten daher auf weitere Details.
- Die NanGate-Bibliothek enthält bereits ein D-FF mit folgenden charakteristischen Zeiten (in *ns*):

Symbol	**Bezeichnung**	t^{min}	t^{max}
t_{SDC}	Setupzeit von *D* bis *ck*	0.08	
t_{HCD}	Holdzeit von *D* nach *ck*	0.14	
τ_{PCQ}	Verzögerungszeit von *ck* bis *Q*	0.12	0.26

Aufbau der Kontrolllogik, zur Erinnerung



- Generierung der Kontrollsignale (OE von Treibern, ALU-Ansteuerung, ...).
- Ist ein Kontrollsignal *active low*, dann bezeichnen wir es z.B. mit $/x$. Das Ausgangssignal $/x$ ergibt sich dann durch Negation des Ausgangssignals x eines entsprechenden FFs mit Eingangssignal x_{pre}.
- Ist ein Kontrollsignal *active high*, dann bezeichnen wir es z.B. mit x. Das Ausgangssignal x entspricht dem Ausgangssignal eines FFs mit Eingangssignal x_{pre}.

Kontrolllogik

- Die Dauer eines Taktes bezeichnen wir als Zykluszeit t_c.
- Active-High-Ausgangssignale der Kontrolllogik, bei denen das FF mit *ck* gesteuert ist, sind gegenüber der steigenden Flanke von *ck* um Zeit $\tau^+_{p,ah}$ verzögert (resultiert aus D-FF-Verzögerung).
- Active-Low-Ausgangssignale der Kontrolllogik, bei denen das FF mit *ck* gesteuert ist, sind gegenüber der steigenden Flanke von *ck* um Zeit $\tau^+_{p,al}$ verzögert (resultiert aus D-FF-Verzögerung + Inverterverzögerung).
- Active-High-Ausgangssignale der Kontrolllogik, bei denen das FF mit $/ck$ gesteuert ist, sind gegenüber der letzten steigenden Flanke von *ck* um Zeit $\tau^-_{p,ah} = \tau^+_{p,ah} + t_c/2 + \tau_{PLH,Inv}$ verzögert.
- Active-Low-Ausgangssignale der Kontrolllogik, bei denen das FF mit $/ck$ gesteuert ist, sind gegenüber der letzten steigenden Flanke von *ck* um Zeit $\tau^-_{p,al} = \tau^+_{p,al} + t_c/2 + \tau_{PLH,Inv}$ verzögert.

INV	$t^{\min}$	$t^{\max}$
τ_{PLH}	0.01	0.15
τ_{PHL}	0.00	0.08

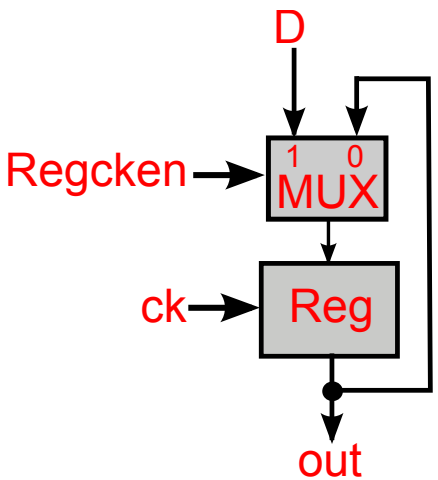
Timing: Kontrolllogik

- Mit geeigneter Implementierung des kombinatorischen Teiles erhält man folgende charakteristische Zeiten.

Symbol	Bezeichnung	t^{min}	t^{max}
$\tau^{+}_{p,ah}$	Verzögerungszeit *ck* bis *Q*, active high	0.12	0.26
$\tau^{+}_{p,al}$	Verzögerungszeit *ck* bis *Q*, active low	0.12	0.41
$\tau^{-}_{p,ah}$	Verzögerungszeit *ck* bis *Q* (von $/ck$ angesteuert, active high)	$t_c/2+0.13$	$t_c/2+0.41$
$\tau^{-}_{p,al}$	Verzögerungszeit *ck* bis *Q* (von $/ck$ angesteuert, active high)	$t_c/2+0.13$	$t_c/2+0.56$
t^{+}_{SDC}	Setupzeit von *D* bis *ck*	0.88	
t^{-}_{SDC}	Setupzeit von *D* bis $/ck$	0.88	
t^{+}_{HCD}	Holdzeit von *D* nach *ck*	0.06	
t^{-}_{HCD}	Holdzeit von *D* nach $/ck$	0.06	

Register mit Clock-Enable

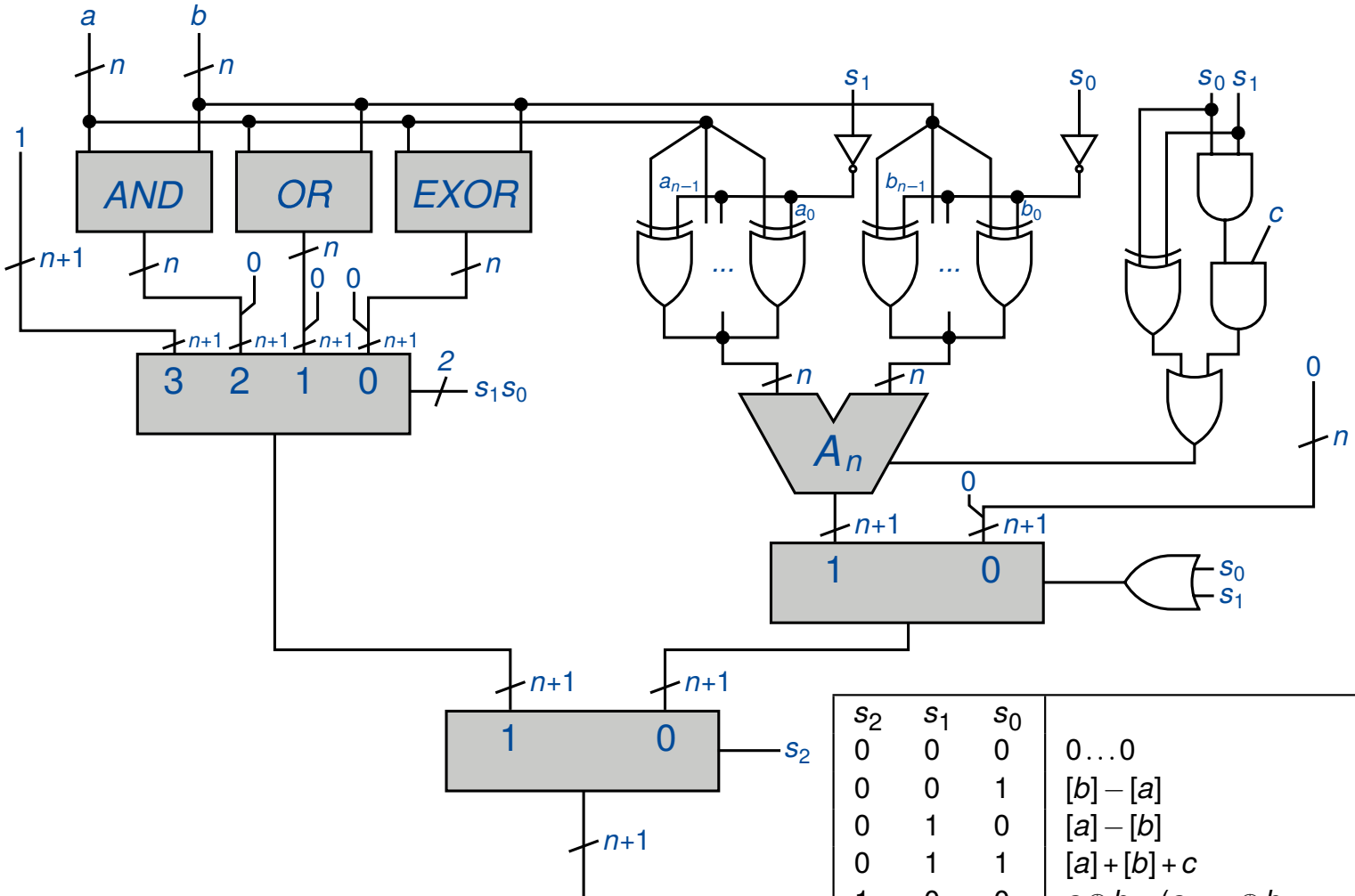
- Bei der Implementierung benötigen wir noch einen Treiberbaum der Tiefe 2, um *Regcken* auf 32 1–Bit–Multiplexer zu verteilen.



Symbol	Bezeichnung	t^{min}
t_{SDC}	Setup-Zeit von *D* vor *ck*	0.23
t_{HDC}	Hold-Zeit von *D* nach *ck*	0.11
t_{SEC}	Setup-Zeit von *Regcken* vor *ck*	0.46
t_{HEC}	Hold-Zeit von *Regcken* nach *ck*	0.08

- t_{SDC} ergibt sich aus Setupzeit D-FF + maximale Verzögerungszeit Multiplexer (Daten bis Ausgang) (0.08 + 0.15).
- t_{HDC} ergibt sich aus Holdzeit D-FF - minimale Verzögerungszeit Multiplexer (Daten bis Ausgang) (0.14 - 0.03).
- t_{SEC} ergibt sich aus Setupzeit D-FF + maximale Verzögerungszeit Multiplexer (Select bis Ausgang) + 2 x maximale Verzögerungszeit Treiber (0.08 + 0.16 + 2 x 0.11).
- t_{HEC} ergibt sich aus Holdzeit D-FF - minimale Verzögerungszeit Multiplexer (Select bis Ausgang) - 2 x minimale Verzögerungszeit Treiber (0.14 - 0.02 - 2 x 0.02).

Schaltrealisierung der ALU



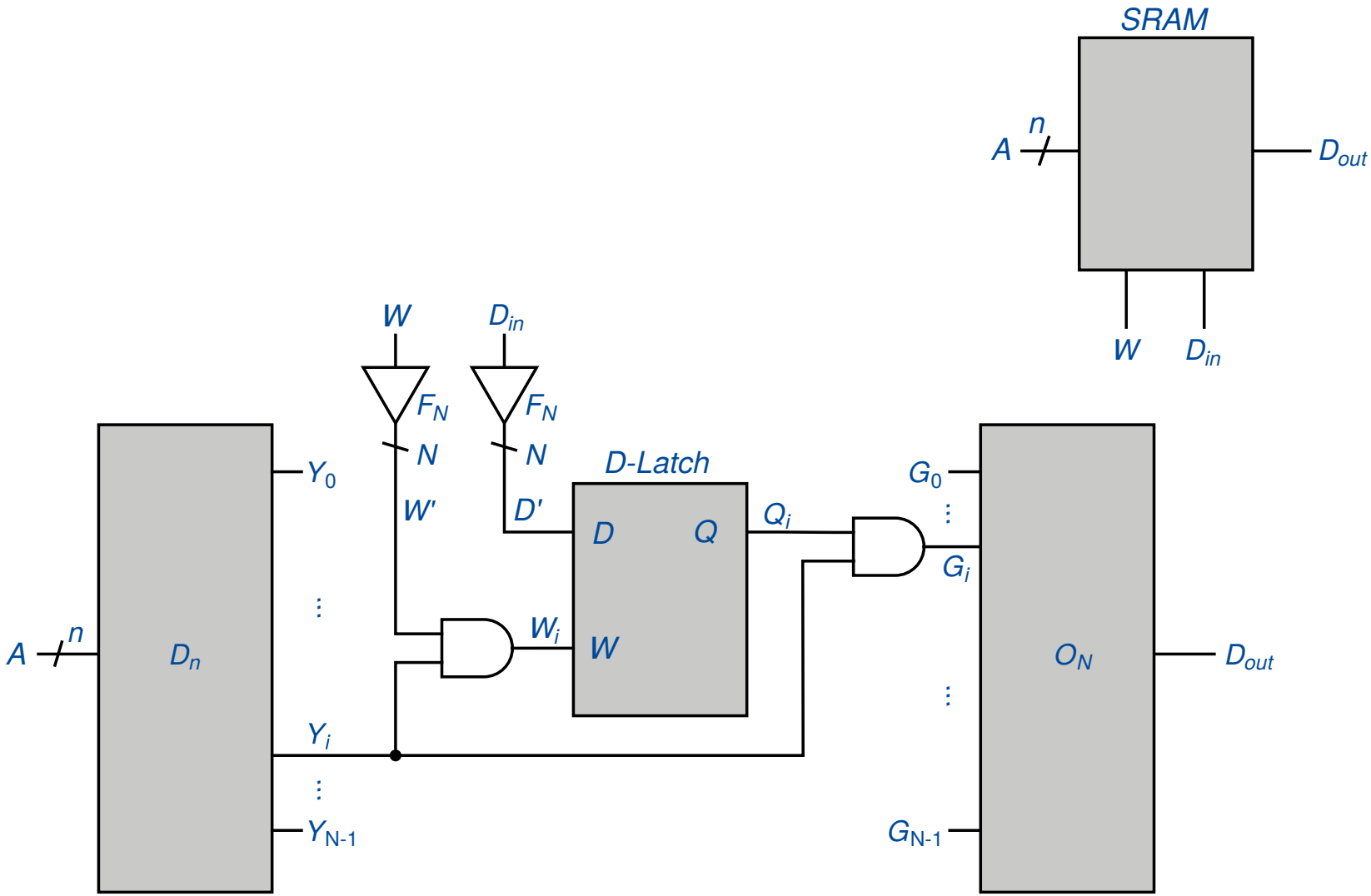
s_2	s_1	s_0	
0	0	0	$0 \ldots 0$
0	0	1	$[b] - [a]$
0	1	0	$[a] - [b]$
0	1	1	$[a] + [b] + c$
1	0	0	$a \oplus b = (a_{n-1} \oplus b_{n-1}, \ldots, a_0 \oplus b_0)$
1	0	1	$a \vee b = (a_{n-1} \vee b_{n-1}, \ldots, a_0 \vee b_0)$
1	1	0	$a \wedge b = (a_{n-1} \wedge b_{n-1}, \ldots, a_0 \wedge b_0)$
1	1	1	$1 \ldots 1$

Timing: ALU

- Annahme: ALU mit 32-Bit-Addierer (Conditional Sum).
- Man zeigt:
  - Längster Pfad über ALU läuft durch den Addierer.
  - Annahme:
    - Die Funktion–Select–Bits sind mindestens t_{select} = 0.28 ns vor den Operanden gültig.
    - Dann ist garantiert, dass der kritische Pfad nicht durch die select–Eingänge bestimmt wird.
  - Zeitverhalten der ALU:

Symbol	**Bezeichnung**	t^{min}	t^{max}
t_{select}		0.28	
t_{ALU}	Verzögerungszeit von a, b bzw. c_{in} bis Ausgang		3.25

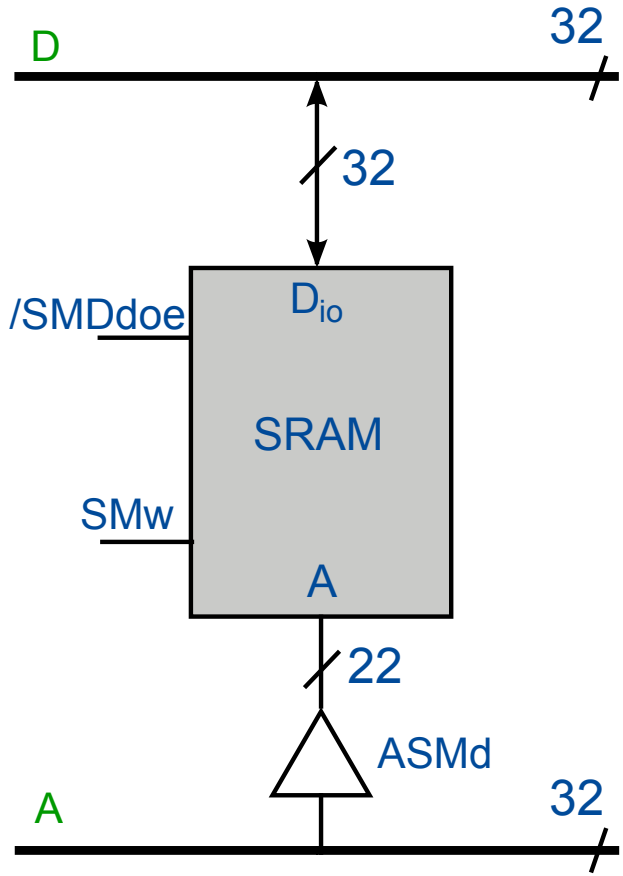
SRAM


SRAM
n
A
D_{out}
W
D_{in}
W
D_{in}
F_N
F_N
N
N
D-Latch
Y_0
W'
D'
D
Q
Q_i
G_0
G_i
A
n
D_n
W_i
W
O_N
D_{out}
Y_i
Y_{N-1}
G_{N-1}

Timing: SRAM

- Auch hier wäre das Vorgehen analog zu den bereits vorgestellten Analysen möglich. Zuvor muss man sich noch Gedanken machen um das Timing von:
  - Dekodierer
  - Treiberbäume
  - OR-Baum
- Eine detaillierte Timinganalyse ist aufwändig.
- Für die folgenden Timinganalysen orientieren wir uns an dem kommerziell angebotenen SRAM CY7C1079DV33 der Firma Cypress Semiconductor (siehe folgende Folien).

Interface zu CY7C1079DV33



Timing: CY7C1079DV33

- Aus dem Datenblatt entnimmt man:

Symbol	Bezeichnung	t^{min}	t^{max}
t_{acc}	Lesezugriffszeit		12.0
t_{OED}	Zeit von $/SMDdoe = 0$ bis D		7.0
t_{OEZ}	Zeit von $/SMDdoe = 1$ bis high-Z		7.0
t_{wc}	Schreibzykluszeit	12.0	
t_{SAW}	Setupzeit von A bis W	0.0	
t_{SAEW}	Setupzeit von A bis Ende W	9.0	
t_{HWA}	Holdzeit von A nach W	0.0	
w	Schreibpulsweite	9.0	
t_{SDEW}	Setupzeit von D bis Ende W	7.0	
t_{HWD}	Holdzeit von D nach W	0.0	