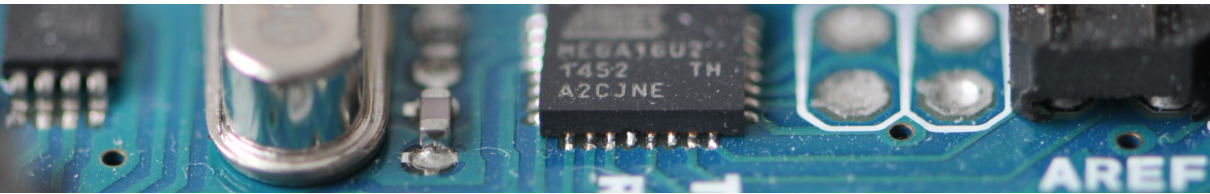
universität innsbruck



Rechnerarchitektur

Sequenzielle Logik I

Univ.-Prof. Dr.-Ing. Rainer Böhme

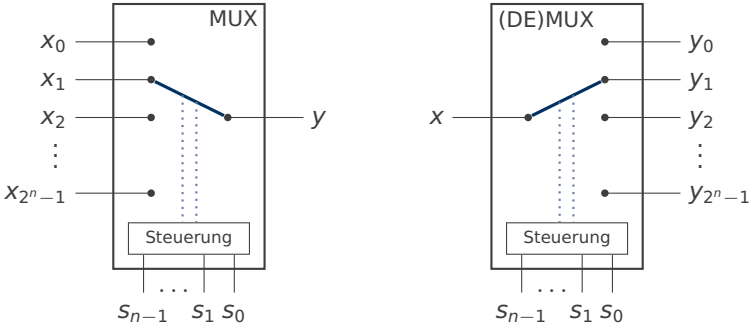
Wintersemester 2020/21 · 28. Oktober 2020

Gliederung heute

1. **Abschluss der Kombinatorischen Logik**
2. Flipflops
3. Schaltwerke

Multiplexer

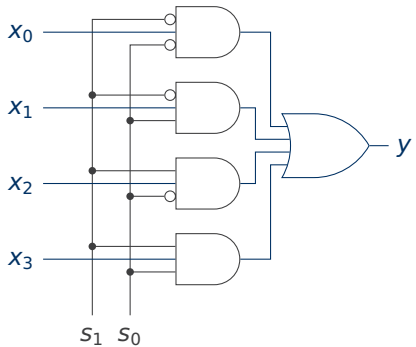
Auswahl einer Datenquelle x_i (bzw. Senke y_j für Demultiplexer) über n Steuerleitungen:



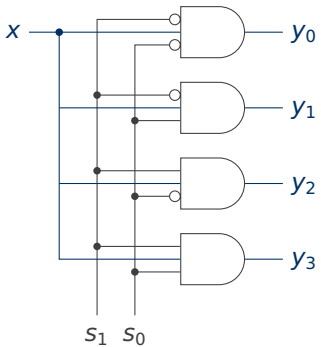
Schaltungstechnische Realisierung

für $n = 2$ Steuerleitungen

MUX



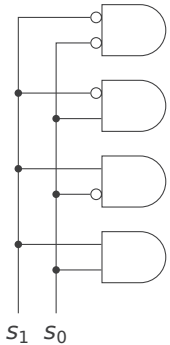
DEMUX



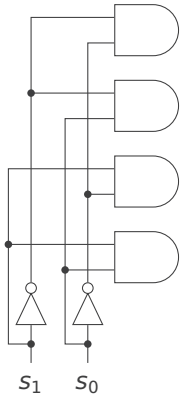
Schaltungstechnische Realisierung

für $n = 2$ Steuerleitungen

Steuerlogik

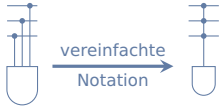


Alternative mit Vorinvertierung

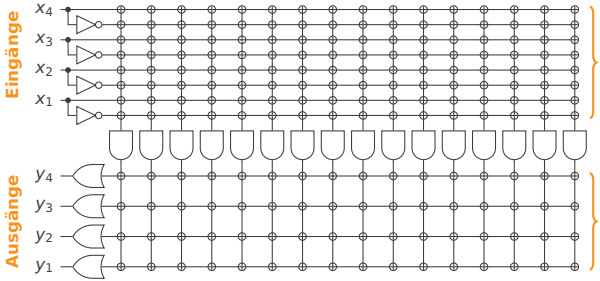


OTP-Logikanordnungen

(engl. one time programmable)



Grundstruktur



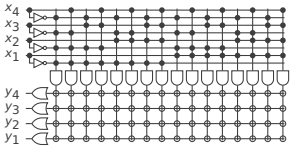
Programmierung der Bauelemente

Hohe Schreibströme brennen Sicherungen (engl. fuses) auf dem Die durch oder neutralisieren Dioden.

Varianten

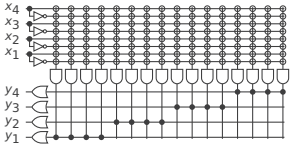
Programmable Read-Only Memory (PROM)

- AND-Matrix fest, OR-Matrix programmierbar
- Direkte Realisierung von Wahrheitstabellen
- PROM mit 2^n m-Bit-Worten implementiert jede Schaltfunktion $f : \{0,1\}^n \to \{0,1\}^m$.



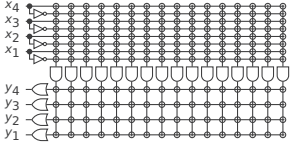
Programmable Array Logic (PAL)

- AND-Matrix programmierbar, OR-Matrix fest
- Realisiert DNFs bis zu einer Obergrenze von Produkttermen pro Summand



Programmable Logic Array (PLA)

- AND-Matrix **und** OR-Matrix programmierbar
- Realisiert DNFs bis zu einer Obergrenze an Produkttermen (gg. durch Anzahl der AND-Gatter)

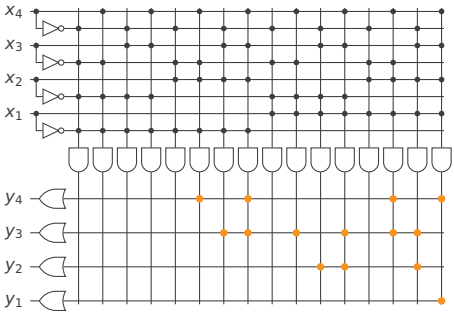


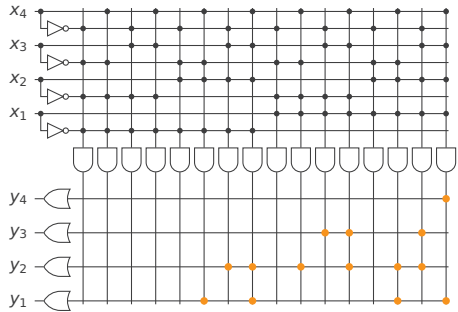
Hörsaalfrage

24 82 94 16

1. Welches PROM realisiert diesen 2-Bit-Multiplizierer?

$$8y_1 + 4y_2 + 2y_3 + y_4 = (2x_1 + x_2) \times (2x_3 + x_4)$$





Zugang: `https://arsnova.uibk.ac.at` mit Zugangsschlüssel **24 82 94 16**. Oder scannen Sie den QR-Kode.

Abgrenzung

Kombinatorische Logik

- Keine Rückkopplung
- Grundelemente sind **Gatter**
- Schaltnetze sind idealisiert **verzögerungsfrei**
- Beschreibung durch **Wahrheitstabellen** oder **Boolesche Ausdrücke**

Sequenzielle Logik

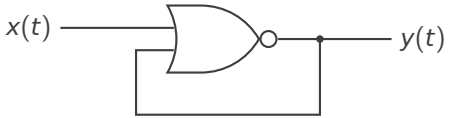
- **Kontrollierte** Rückkopplung
- Grundelemente sind **Flipflops** und Gatter
- Schaltwerke berücksichtigen **Zeitverhalten** unter Annahme konstanter **Gatterlaufzeit** τ
- Beschreibung durch **Zustandstabelle** und **Zustandsdiagramm** oder **Ansteuer-** und **Ausgabegleichungen**

Gliederung heute

1. Abschluss der Kombinatorischen Logik
2. **Flipflops**
3. Schaltwerke

Rückkopplung eines Gatterausgangs

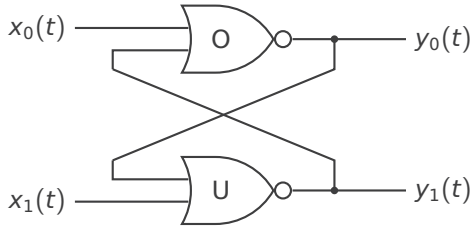
Beispiel: NOR-Gatter



$x(t)$	$y(t+\tau)$	$y(t+2\tau)$	$y(t+3\tau)$
0	$\overline{y(t)}$	$y(t)$	$\overline{y(t)}$
1	0	0	0

$\rightarrow$ **Ausgang oszilliert weitgehend unkontrolliert.**

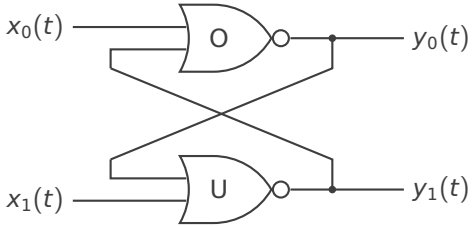
Kreuzweise Rückkopplung zweier Gatterausgänge



$x_0(t)$	$x_1(t)$	$y_0(t+\tau)$	$y_1(t+\tau)$	$y_0(t+2\tau)$	$y_1(t+2\tau)$
0	0	$\overline{y_1(t)}$	$\overline{y_0(t)}$	**?**	**?**
0	1	$\overline{y_1(t)}$	0	1	0
1	0	0	$\overline{y_0(t)}$	0	1
1	1	0	0	0	0

Kreuzweise Rückkopplung zweier Gatterausgänge

Beispiel: NOR-Gatter



Bistabile Kippstufe

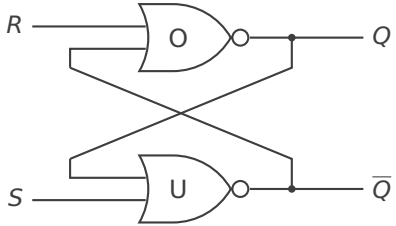
Diese Schaltung kann ein Bit in ihrem **Zustand** Q speichern.

RS-Flipflop

Bezeichnung der zwei Eingänge

- R = “reset” (löschen)
- S = “set” (setzen)

Realisierung mit zwei NOR-Gattern



Folgezustand

charakteristische Tabelle

R	S	Q'
0	0	Q
0	1	1
1	0	0
1	1	nicht erlaubt

Flipflop: bistabile Kippstufe (W)

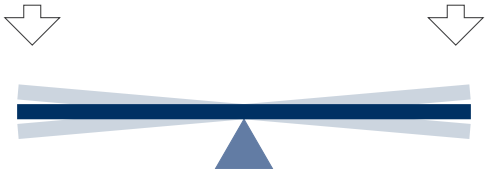
Mechanische Analogie: Wippe

S

Set (setzen)

R

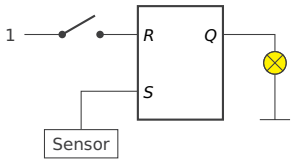
Reset (zurücksetzen)



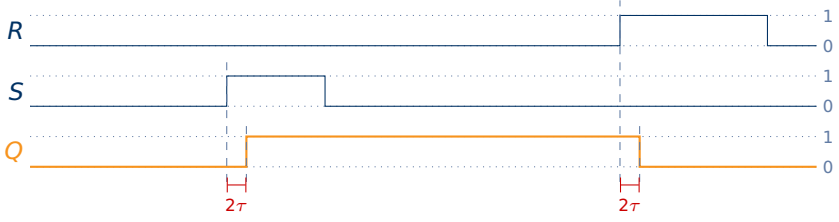
Beispiel für den Einsatz von RS-Flipflops

Einschalten der Warnlampe bei kurzfristiger Temperaturüberschreitung, Taster zum Zurücksetzen

- Speicherung eines flüchtigen Werts
- Initialisierung erforderlich



Darstellung im Zeitverlauf



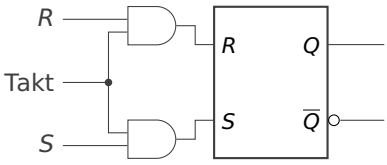
Getaktetes RS-Flipflop

Idee

Vermeidung unbeabsichtigter Zuständsübergänge durch Laufzeiteffekte:

Übernahme der Signale an *R* und *S* **nur wenn** Taktsignal 1 ist.

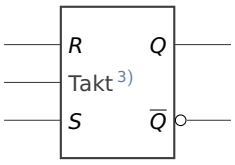
Realisierung



charakteristische Tabelle

R	*S*	Takt	Q'
d [1]	**d**	0	*Q*
0	0	1	*Q*
0	1	1	1
1	0	1	0
1	1	1	— [2]

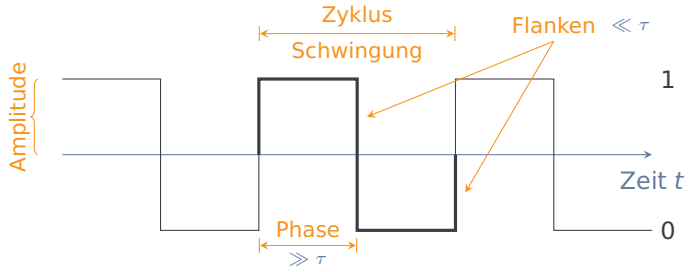
Symbol



[1] don't care, [2] i. d. R. nicht erlaubt, [3] auch: Clk oder *c* für *clock*

Takt

Symmetrisches Rechtecksignal mit konstanter **Taktfrequenz** (Schwingungen pro Zeiteinheit) gemessen in **Hertz** [1 Hz = $1/s$]



Taktgenerierung

Industriell gefertigte **Quarze** schwingen sehr präzise ($>$ 4 MHz).
Niedrigere Frequenzen erreicht man durch (mehrfache) Teilung.

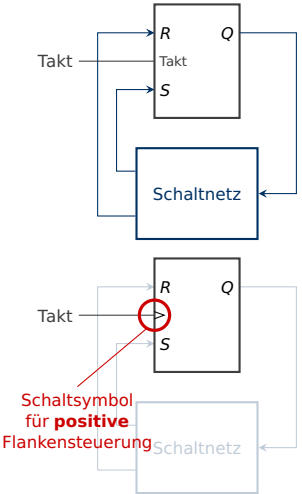
Flankengesteuertes RS-Flipflop

Nachteil getakteter RS-Flipflops

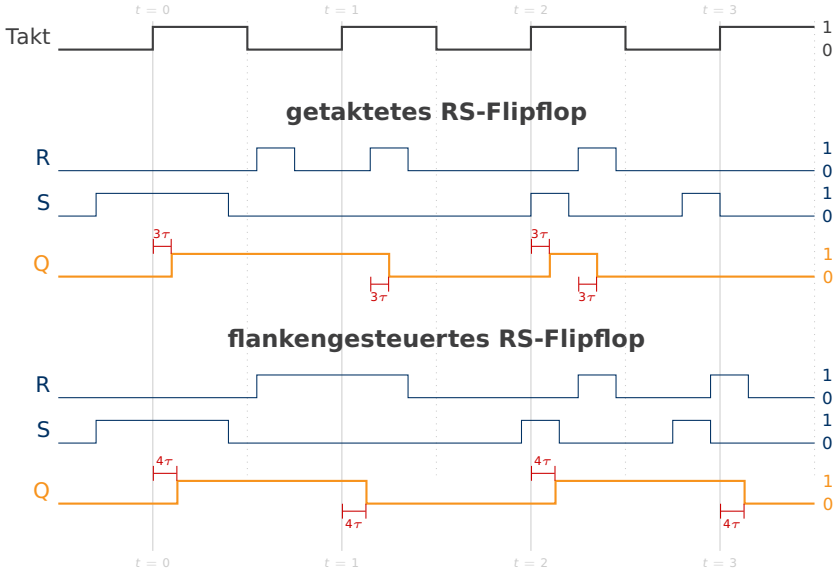
- mehrere Zustandsänderungen in einer Taktphase möglich
- erschwert kontrollierte Rückkopplung über Schaltnetz

Alternative: Flankensteuerung

- Übernahme der Eingabewerte zu einem bestimmten Zeitpunkt
- Varianten für positive (steigende) oder negative (fallende) Flanke



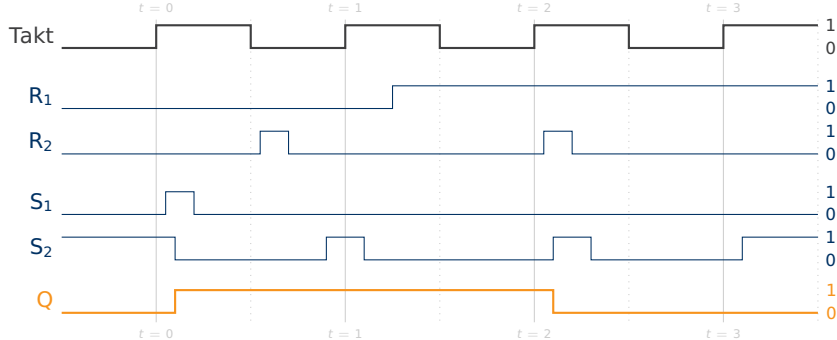
Zeitverhalten ausgewählter Flipflops



Hörsaalfrage

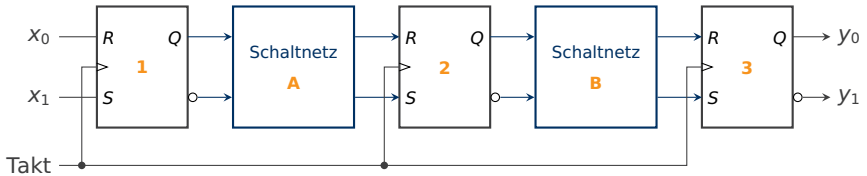
24 82 94 16

2. Welche *R*- und *S*-Signale passen zur (idealen) Ausgabe *Q* eines flankengesteuerten RS-Flipflops?



Zugang: `https://arsnova.uibk.ac.at` mit Zugangsschlüssel **24 82 94 16**. Oder scannen Sie den QR-Kode.

Realisierung einer „Pipeline“



- parallele Speicherung/Verarbeitung unterschiedlicher Daten
- schrittweise **Weitergabe** der Ergebnisse bei Taktflanke
- Ergebnis liegt (im Beispiel) nach drei Takten am Ausgang an.
- **Voraussetzung:** Die maximale Verzögerung aller Schaltnetze – der kritische Pfad – ist kürzer als ein Taktzyklus.

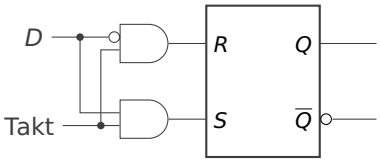
D-Flipflop

(Abk. für Daten oder Delay, engl. *Verzögerung*)

Eigenschaften

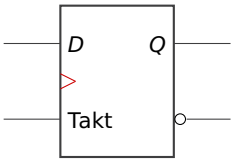
- Ein Datenbit $D = S = \overline{R}$ als Eingang (vermeidet $R = S = 1$).
- Bei Takt $= 1$ wird der Folgezustand $Q' = D$ gesetzt.

Realisierung



D	Takt	Q'
0	0	Q
0	1	0
1	0	Q
1	1	1

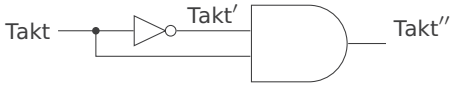
Symbol

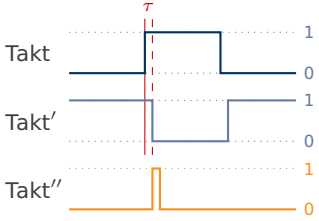


→ D-Flipflops kommen häufig mit Flankensteuerung zum Einsatz.

Flankenerkennung

schaltungstechnische Realisierung für **positive** Flanke

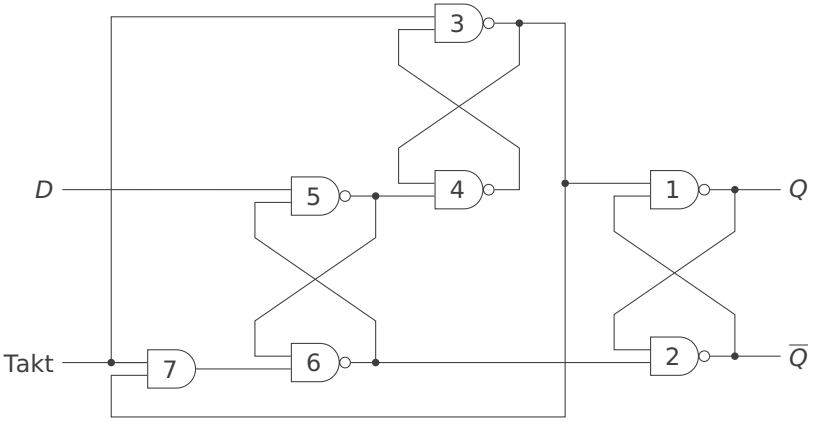




→ Ausnutzung von Laufzeiteffekten

Realisierung mit NAND- und AND-Gattern

positiv flankengesteuertes D-Flipflop

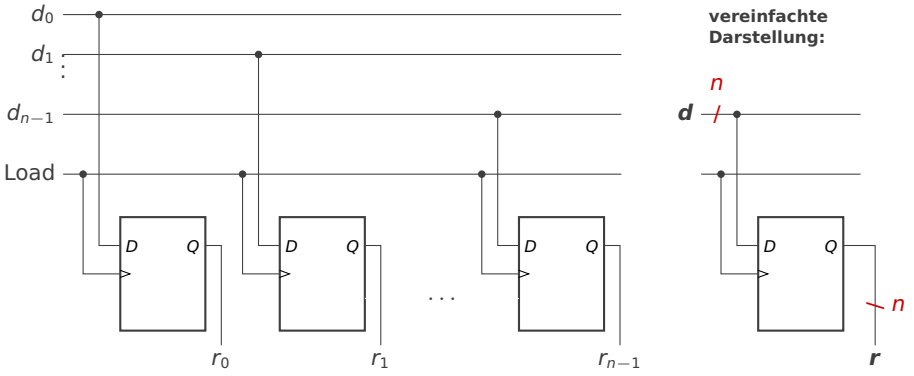


(Funktionsweise nicht prüfungsrelevant)

Gliederung heute

1. Abschluss der Kombinatorischen Logik
2. Flipflops
3. **Schaltwerke**

Beispiel: *n*-Bit-Register



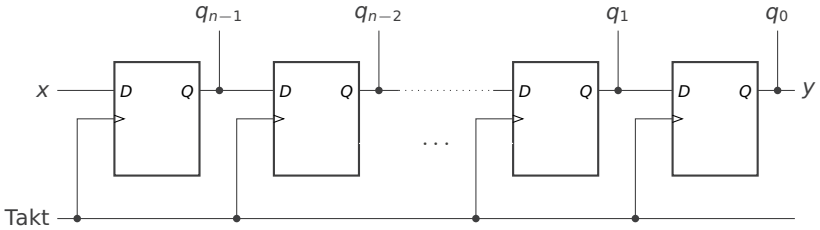
Konstruktion aus flankengesteuerten D-Flipflops

Übernahme der Daten von einem **Datenbus *d*** der Breite *n* Bit bei steigender Flanke des „Load“-Signals.

Beispiel: *n*-Bit-Schieberegister

Anwendungen

- seriell-zu-parallel-Wandlung
- arithmetische Operationen



Konstruktion aus flankengesteuerten D-Flipflops

Verschiebung der Binärwerte pro Takt um eine Position nach rechts.

Syllabus – Wintersemester 2020/21

07.10.20	1.	Einführung
14.10.20	2.	Kombinatorische Logik I
21.10.20	3.	Kombinatorische Logik II
28.10.20	4.	Sequenzielle Logik I
04.11.20	5.	Sequenzielle Logik II
11.11.20	6.	Arithmetik I
18.11.20	7.	Arithmetik II
25.11.20	8.	Befehlssatzarchitektur (ARM) I
02.12.20	9.	Befehlssatzarchitektur (ARM) II
09.12.20	10.	Prozessorarchitekturen
16.12.20	11.	Ein-/Ausgabe
06.01.21	12.	Speicher
13.01.21	13.	Leistung
20.01.21	14.	Wiederholung, Fragestunde
27.01.21		**Klausur (1. Termin)**