

Digitaltechnik

Wintersemester 2017/2018

9. Vorlesung



TECHNISCHE
UNIVERSITÄT
DARMSTADT





1. Einleitung
2. Zeitverhalten synchroner sequentieller Logik
3. Parallelität
4. Zusammenfassung

Einleitung



TECHNISCHE
UNIVERSITÄT
DARMSTADT

1010110000001100011111010011100000110110
1010001011010110110000011001011111011101
0110011011010110101001001100100010111111
1001111010101010011010001011010010100010
1100011100101111010110110110100101110100
0010001001000011101110011110000001010110
1100000100111000000110100010000001001111
0000010100101111010001001011110011101101
1101110110010111101001011001011110001111
1010000100111101010111101100101011010101
1000110010000111100000011010000100100010
1010010101010000110010011111110000000011
1110011010010010010000111101010110110001
1000100000100000000100001100110011001011
1010000111011111110010111101111110011110
1111110110011100110011000011010010010011



- ▶ Zusammenlegung von Übungsgruppen ab KW 51 geplant
 - ▶ Tutoren-Zuordnung bleibt erhalten
 - ▶ genaue Infos bis Ende KW 50 im Moodle

Rückblick auf die letzten Vorlesungen



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ Sequentielle Logik
 - ▶ Sequentielle Schaltungen
 - ▶ Speicherelemente
 - ▶ Synchrone sequentielle Logik
- ▶ Endliche Zustandsautomaten
 - ▶ Konzept, Notationen und Anwendungsbeispiele
 - ▶ Moore vs. Mealy
 - ▶ Zerlegen von Zustandsautomaten

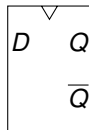
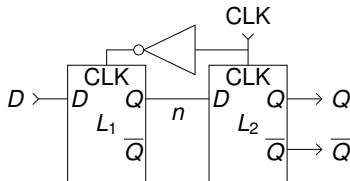
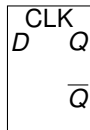
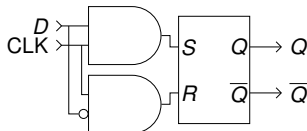
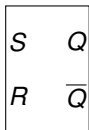
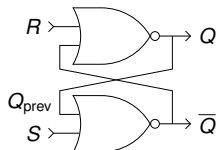


Harris 2013
Kap. 3.1 - 3.4

Wiederholung: Speicherelemente



TECHNISCHE
UNIVERSITÄT
DARMSTADT

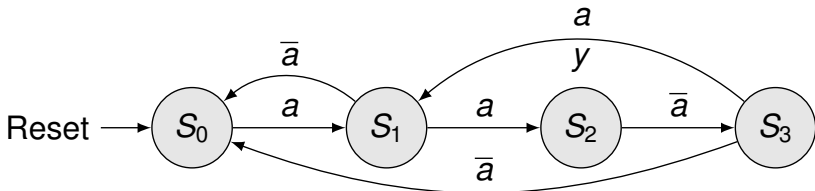
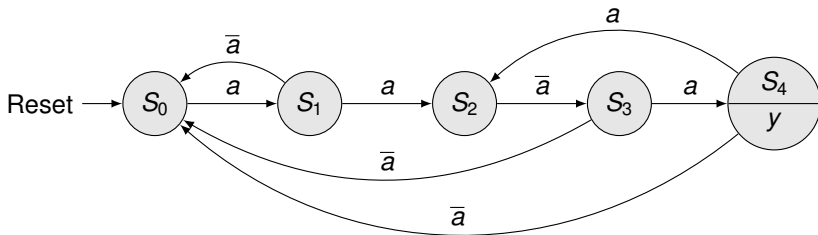


Wiederholung: Endliche Automaten

Notation, Überführung



TECHNISCHE
UNIVERSITÄT
DARMSTADT



Wiederholung: Endliche Automaten

Don't Cares



TECHNISCHE
UNIVERSITÄT
DARMSTADT

S	a	S'
S_0	0	S_0
S_0	1	S_1
S_1	0	S_0
S_1	1	S_2
S_2	0	S_3
S_2	1	S_2
S_3	0	S_0
S_3	1	S_4
S_4	0	S_0
S_4	1	S_2

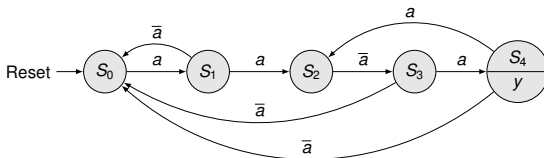
S	s_2	s_1	s_0
S_0	0	0	0
S_1	0	0	1
S_2	0	1	0
S_3	0	1	1
S_4	1	0	0

pattern/moore/state.esp

```

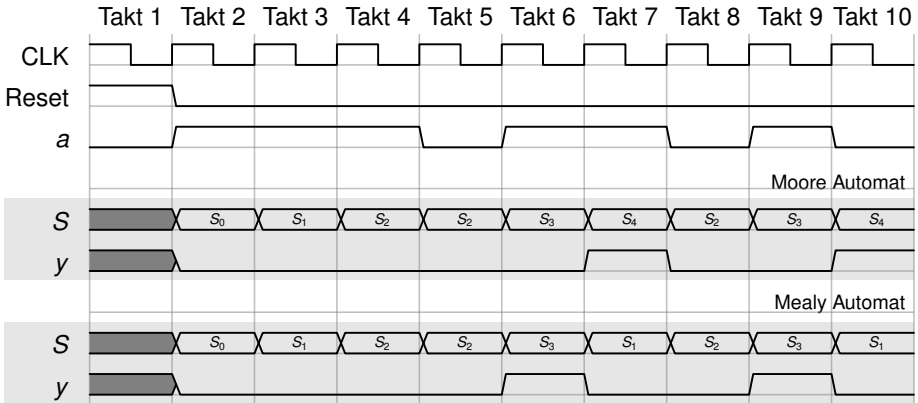
1  .i 4
2  .o 3
3  0000 000
4  0001 001
5  0010 000
6  0011 010
7  0100 011
8  0101 010
9  0110 000
10 0111 100
11 1000 000
12 1001 010
13 1010 ---
14 1011 ---
15 1100 ---
16 1101 ---
17 1110 ---
18 1111 ---

```



Wiederholung: Endliche Automaten

Mealy vs. Moore

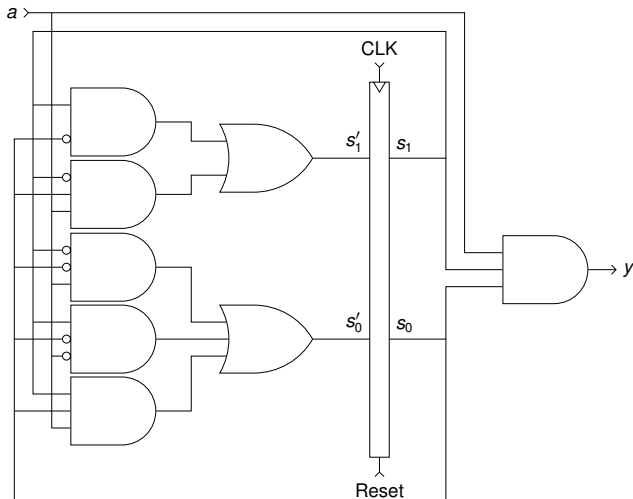


Wiederholung: Endliche Automaten

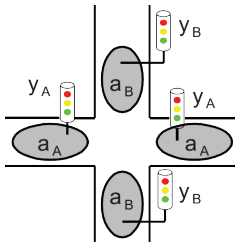
Mealy vs. Moore



TECHNISCHE
UNIVERSITÄT
DARMSTADT

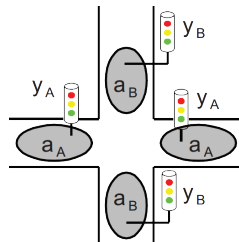
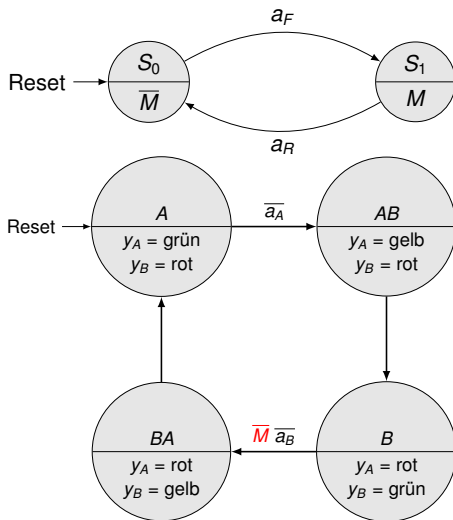


Zerlegung



Wiederholung: Endliche Automaten

Zerlegung



Überblick der heutigen Vorlesung

- ▶ Zeitverhalten synchroner sequentieller Schaltungen
- ▶ Parallelität



Harris 2013
Kap. 3.5-3.6
Seite 133 - 153

Zeitverhalten synchroner sequentieller Logik

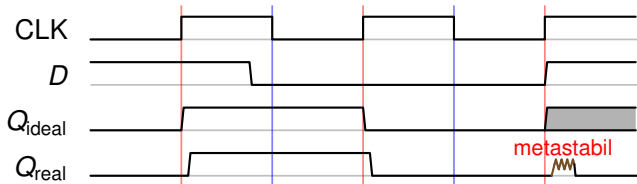
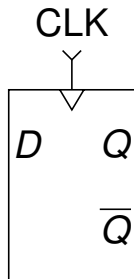


TECHNISCHE
UNIVERSITÄT
DARMSTADT

01111011000111110111110111101000100111110111
0001000001101101010100100000011011010011
01111001001000010010000100010011110001011
1110001100111100011011011111010010101000
1100111100011010110010100000101001101110
0010111001001011110000101101001000011100
0100100010010011000110011011101111011101
111110000011100010100100001000010110011
1011101000001000010001010100010000011110
0100010111110101000111001011111101000100
1001001000110000111011010000011011101101
1000011010011111001110010101110000001011
1010101111011010011010000110001010011100
0100101110011100111100111011001100000100
0010101000110101110110110010100011111110
1100101001001011100010000010110010010011

Zeitverhalten synchroner sequentieller Logik

- ▶ Flip-Flop übernimmt D zur steigenden Taktflanke
- ▶ Was passiert bei zeitgleicher Änderung von D und CLK?
- ▶ bisher vereinfachte Annahme:
 - ▶ Wert unmittelbar vor der Taktflanke wird übernommen
- ▶ Aber:
 - ▶ Was heißt „unmittelbar“?
 - ▶ Wie schnell wird neuer Zustand am Ausgang sichtbar?
 - ▶ Was muss daher bei synchronen sequentiellen Schaltungen beachtet werden?



Analogie zur Fotografie: Bewegungsunschärfe (Bewegung während Belichtung)

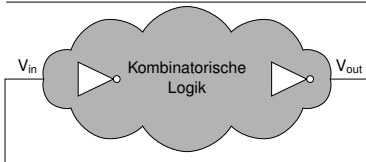


TECHNISCHE
UNIVERSITÄT
DARMSTADT

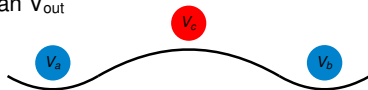
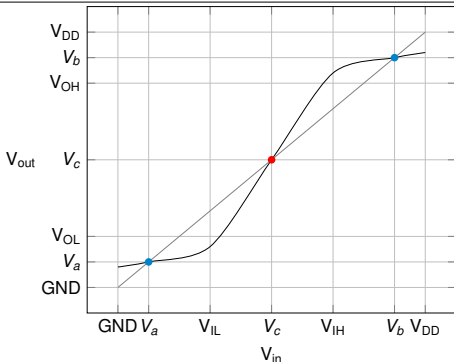


Quelle: <http://lightwatching.de>

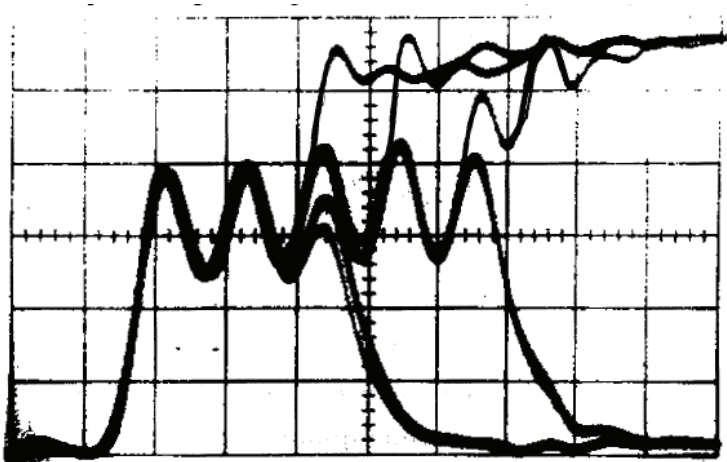
Metastabilität durch Rückkopplung kombinatorischer Logik



- ▶ Rückkopplung stabil für $V_{out} = V_{in}$
 - ▶ Transferfunktion schneidet Hauptdiagonale in
 - V_a repräsentiert 0
 - V_b repräsentiert 1
 - V_c im „verbotenen“ Spannungsbereich
 - ▶ V_c ist „metastabil“, da
 - ▶ kleine Änderung an $V_{in} \rightarrow$ große Änderung an V_{out}
- ⇒ geht nach zufälliger Verzögerung in einen stabilen Zustand über



Metastabilität an Flip-Flops



Quelle: Thomas. J. Chaney, Washington University

Zeitanforderungen an DFF Eingangssignale



TECHNISCHE
UNIVERSITÄT
DARMSTADT

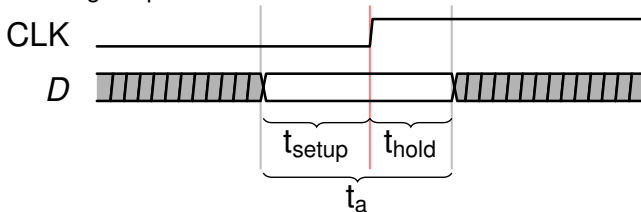
- ▶ Dateneingang D muss in Abtast-Zeitfenster um Taktflanke stabil sein, um Metastabilität zu vermeiden

t_{setup} Zeitintervall vor Taktflanke, in dem D stabil sein muss

t_{hold} Zeitintervall nach Taktflanke, in dem D stabil sein muss

t_a Abtastzeitfenster: $t_a = t_{\text{setup}} + t_{\text{hold}}$

- ▶ Größenordnung: 10 ps

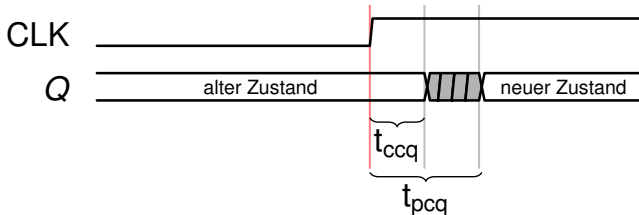


Zeitcharakteristik der DFF Ausgangssignale

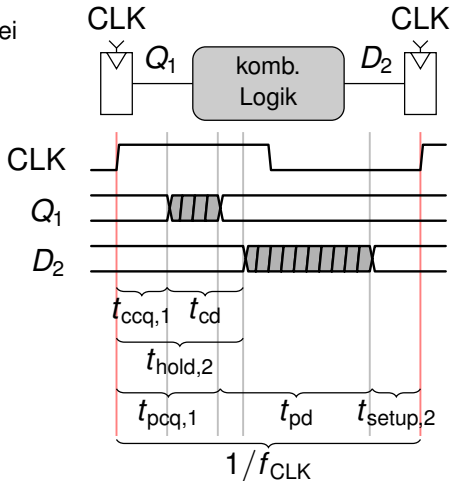


TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ Verzögerung des Registerausgangs relativ zur steigenden Taktflanke
 - t_{ccq} Kontaminationsverzögerung, bis Q (frühestens) umschaltet
 - t_{pcq} Laufzeitverzögerung, bis Q (spätestens) stabil
- ▶ Größenordnung: 10 ps



- ▶ kombinatorische Logik zwischen zwei Registern hat max/min Verzögerung
 - ▶ abhängig von Verzögerungen der Gatter und des *ersten* Registers
- ⇒ Timing-Bedingungen des *zweiten* Registers müssen erfüllt werden
- ▶ $t_{ccq,1} + t_{cd} > t_{hold,2}$
 - ▶ $t_{pcq,1} + t_{pd} + t_{setup,2} \leq \frac{1}{f_{CLK}}$
- ⇒ maximale Taktrate wird durch *kritischen Pfad* bestimmt
- ▶ $f_{CLK} \leq \frac{1}{t_{pcq} + t_{pd} + t_{setup}}$



Beispiel: Analyse der Timing-Bedingungen



TECHNISCHE
UNIVERSITÄT
DARMSTADT

▶ Timing-Vorgaben:

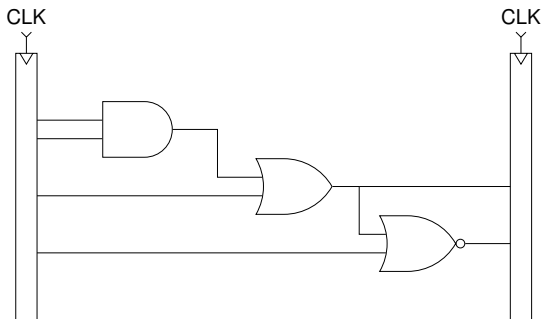
- ▶ $t_{ccq} = 30 \text{ ps}$
- ▶ $t_{pcq} = 50 \text{ ps}$
- ▶ $t_{setup} = 60 \text{ ps}$
- ▶ $t_{hold} = 70 \text{ ps}$
- ▶ $t_{cd,Gatter} = 25 \text{ ps}$
- ▶ $t_{pd,Gatter} = 35 \text{ ps}$

▶ kombinatorischer Pfad:

- ▶ $t_{cd} = 25 \text{ ps}$
- ▶ $t_{pd} = 3 \cdot 35 \text{ ps} = 105 \text{ ps}$

▶ Timing-Bedingungen:

- ▶ $f_{CLK} \leq \frac{1}{t_{pcq} + t_{pd} + t_{setup}} = \frac{1}{215 \text{ ps}} = 4,65 \text{ GHz}$
- ▶ $t_{ccq} + t_{cd} = 55 \text{ ps} < t_{hold}$ ⚡



Beispiel: Beheben der verletzten Hold-Zeit Anforderung



TECHNISCHE
UNIVERSITÄT
DARMSTADT

► Timing-Vorgaben:

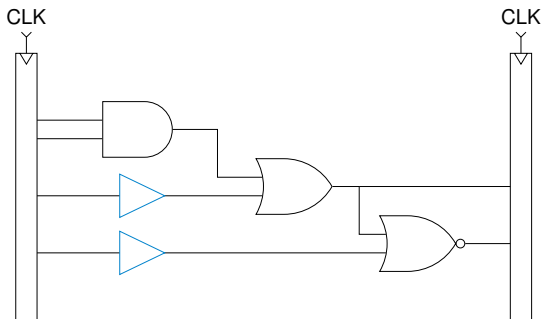
- $t_{ccq} = 30 \text{ ps}$
- $t_{pcq} = 50 \text{ ps}$
- $t_{setup} = 60 \text{ ps}$
- $t_{hold} = 70 \text{ ps}$
- $t_{cd,Gatter} = 25 \text{ ps}$
- $t_{pd,Gatter} = 35 \text{ ps}$

► kombinatorischer Pfad:

- $t_{cd} = 50 \text{ ps}$
- $t_{pd} = 3 \cdot 35 \text{ ps} = 105 \text{ ps}$

► Timing-Bedingungen:

- $f_{CLK} \leq \frac{1}{t_{pcq} + t_{pd} + t_{setup}} = \frac{1}{215 \text{ ps}} = 4,65 \text{ GHz}$
- $t_{ccq} + t_{cd} = 80 \text{ ps} > t_{hold} \checkmark$

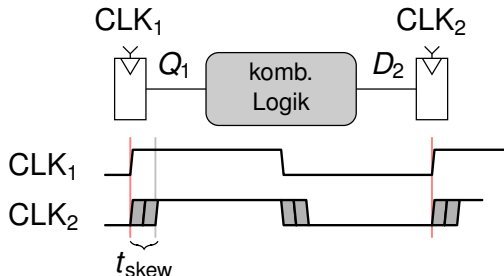


Taktverschiebung (clock skew)



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ Takt kommt nicht bei allen Registern gleichzeitig an
 - ▶ unterschiedliche Verdrahtungswege auf dem Chip (clock tree),
 - ▶ Logik in Taktsignal (bspw. gated clock)
- ▶ t_{skew} ist max. Differenz der Taktankunftszeit zwischen zwei Registern



Timing-Bedingungen mit Taktverschiebung



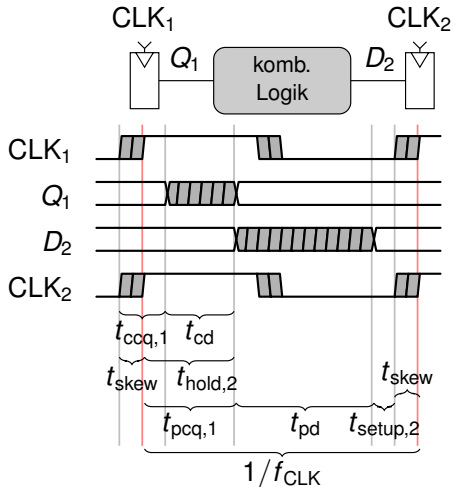
TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ Timing-Bedingungen müssen auch im worst-case eingehalten werden:

- ▶ $t_{ccq,1} + t_{cd} > t_{skew} + t_{hold,2}$
 - ▶ $t_{pcq,1} + t_{pd} + t_{setup,2} + t_{skew} \leq \frac{1}{f_{CLK}}$

⇒ idR. wird Timing durch t_{skew} enger

- ▶ Taktfrequenz kann durch t_{skew} auch steigen, wenn CLK_2 sicher nach CLK_1 schaltet



- ▶ asynchronen Eingänge:

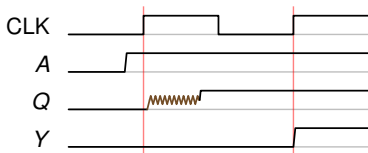
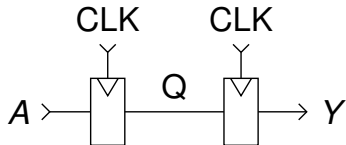
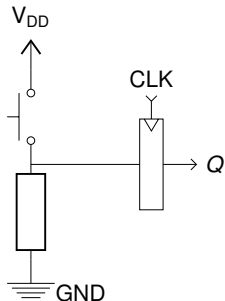
- ▶ Benutzereingaben
- ▶ Kommunikationssignale von externen ICs

⇒ Timing-Bedingungen können nicht garantiert werden

- ▶ zweifaches Shiftregister für Synchronisation

- ▶ erstes Flip-Flop kann metastabil werden
- ▶ kippt idR. vor nächster Taktflanke in stabilen Zustand

⇒ zweites Flip-Flop wird nicht metastabil



Parallelität



TECHNISCHE
UNIVERSITÄT
DARMSTADT

1010101110011001101110101000111100011001
1110111100011011000001100001111110000110
0011010010011100011101010110111100001101
1010111001011000101111011000100111001011
1110111111111001000111010111100010100110
1111001000101011010110110111010000101011
1111000110000101010100010110000010110111
000111100001010001001111110110011100001
1001000011100110000010001100010101100001
1010011000100100101011111011110001000010
1101100011010001100101010011111001100101
0010010100010001000110011111100100011110
0011110000000100000001111100100000110111
0010000010011111111101010011011111101010
1001010000000101111111011110111000100010
11001111110100010101100111001000101000101



- ▶ räumliche Parallelität
 - ▶ mehrere Aufgaben durch vervielfachte Hardware gleichzeitig bearbeiten
 - ▶ zeitliche Parallelität
 - ▶ Aufgabe in mehrere Unteraufgaben aufteilen
 - ▶ Unteraufgaben parallel ausführen
 - ▶ Beispiel: Fließbandprinzip bei Autofertigung („Pipelining“)
 - ▶ nur eine Station für pro Arbeitsschritt
 - ▶ alle unterschiedlichen Arbeitsschritte für mehrere Autos parallel ausgeführt
- ⇒ zeitliche Parallelität



Datensatz: Vektor aus Eingabewerten, zu denen ein Vektor aus Ausgabewerten berechnet wird

Latenz: Zeit von der Eingabe eines Datensatzes bis zur Ausgabe des zugehörigen Ergebnisses

Durchsatz: Anzahl von Datensätzen, die pro Zeiteinheit bearbeitet werden können

⇒ Parallelität erhöht Durchsatz

Beispiel: Plätzchen backen



TECHNISCHE
UNIVERSITÄT
DARMSTADT

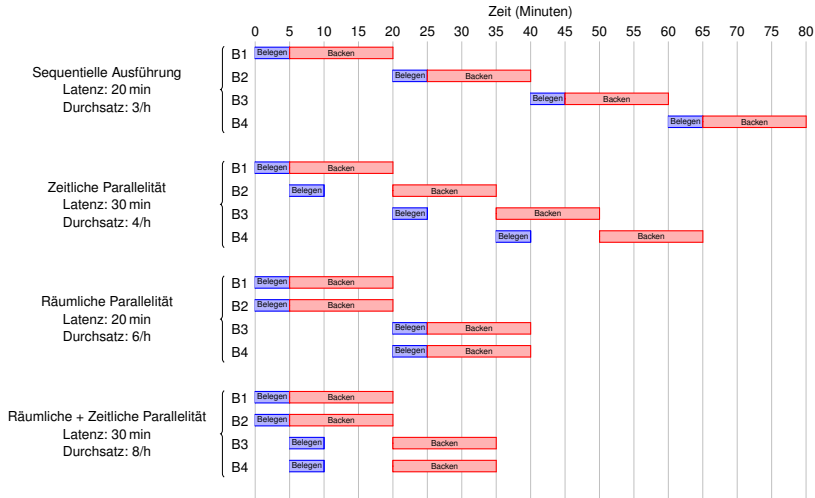
- ▶ Annahmen:
 - ▶ genug Teig ist fertig
 - ▶ 5 Minuten zum Belegen eines Bleches
 - ▶ 15 Minuten Backzeit
- ▶ *sequentiell*: ein Blech nach dem anderen belegen und backen
- ▶ *zeitlich parallel*: nächstes Blech belegen, während erstes noch im Ofen ist
- ▶ *räumlich parallel*: zwei Bäcker, jeweils mit eigenem Ofen
- ▶ räumliche und zeitliche Parallelität kombiniert



Beispiel: Plätzchen backen



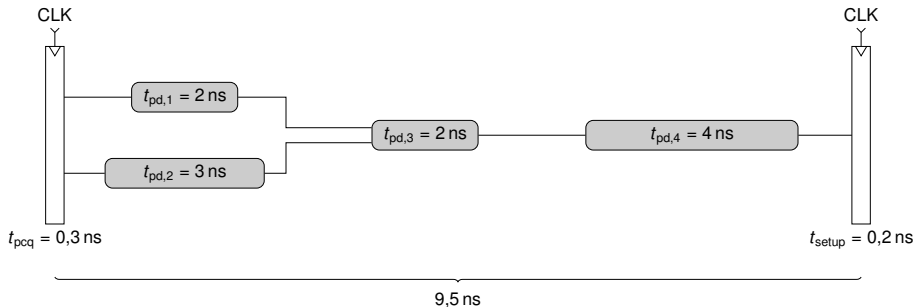
TECHNISCHE
UNIVERSITÄT
DARMSTADT



Beispiel: Pipelining in Schaltungen Ohne Pipeline-Register



TECHNISCHE
UNIVERSITÄT
DARMSTADT

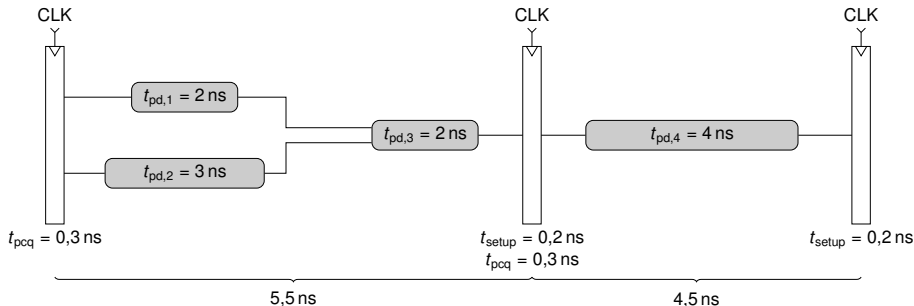


► $f_{CLK} \leq \frac{1}{9,5 \text{ ns}} = 105 \text{ MHz}$

► Latenz: 1 Takt $\equiv 9,5 \text{ ns}$

Beispiel: Pipelining in Schaltungen

Zwei Pipeline-Stufen

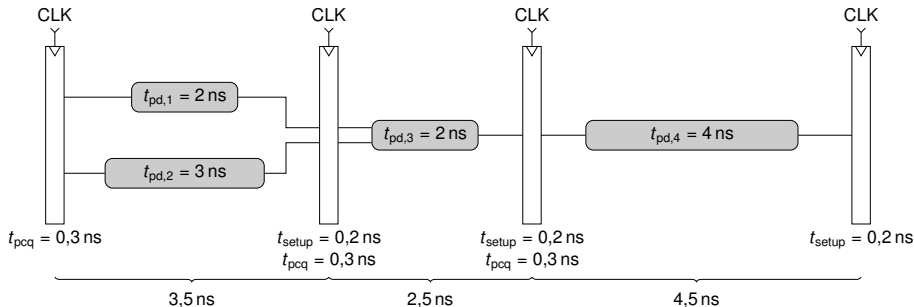


► $f_{CLK} \leq \max\left(\frac{1}{5,5 \text{ ns}}, \frac{1}{4,5 \text{ ns}}\right) = 182 \text{ MHz}$

► Latenz: 2 Takte $\equiv 11 \text{ ns}$

Beispiel: Pipelining in Schaltungen

Drei Pipeline-Stufen



► $f_{CLK} \leq \max\left(\frac{1}{3,5 \text{ ns}}, \frac{1}{2,5 \text{ ns}}, \frac{1}{4,5 \text{ ns}}\right) = 222 \text{ MHz}$

► Latenz: 3 Takte $\equiv 13,5 \text{ ns}$



- ▶ Pipelinestufen sollten möglichst gleich lang sein („ausbalanciert“)
 - ▶ längste Stufe bestimmt Taktrate
 - ▶ Latenz = Pipelinestufen / Taktrate
- ▶ mehr Pipelinestufen
 - ▶ höherer Durchsatz (mehr Ergebnisse pro Zeiteinheit)
 - ▶ aber auch höhere Latenz (länger auf Ergebnis warten)
 - ⇒ lohnt sich nur, wenn viele Datensätze bearbeitet werden müssen
- ▶ Probleme bei Abhängigkeiten zwischen Teilaufgaben
 - ▶ bspw. erst Backergebnis prüfen, bevor nächstes Blech belegt wird
- ▶ wird noch intensiv beim Thema Befehlsverarbeitung von Prozessoren behandelt

Zusammenfassung



TECHNISCHE
UNIVERSITÄT
DARMSTADT

010000000110110010000001000100111101111111
1100111100100000101110100010111000000001
0000100110111010000110010001110011010001
0010001011011011101011001110101001100110
0011100001001011001011010011011101011101
1100001011100100011110110011101000011111
10101011110000010110110010000111010000010
010111110101011111010001001100110011011
1001001011001011000000010011010000100110
010001100010111011111111010111011111011
1011110010010010011110001000011010100000111
10101111111110100011010001010000011000111
1110011010011100010010100010111101001110
0110001000001010010011000010111100001010
0000011100010100000100111100001011111101
0011101100111001101100001010001100100011



- ▶ Zeitverhalten synchroner sequentieller Schaltungen
- ▶ Parallelität

- ▶ Nächste Vorlesung behandelt
 - ▶ Hardwarebeschreibung mit SystemVerilog