

Digitaltechnik

Wintersemester 2021/2022

8. Vorlesung












TECHNISCHE
UNIVERSITÄT
DARMSTADT





Umfrage zur letzten Woche

1. Speicherelemente (Fortsetzung)
2. Synchrone sequentielle Logik
3. Zeitverhalten synchroner sequentieller Logik
4. Parallelität
5. Zusammenfassung

Anwendungs- software		Programme
Betriebs- systeme		Gerätetreiber
Architektur		Befehle Register
Mikro- architektur		Datenpfade Steuerung
Logik		Addierer Speicher
Digital- schaltungen		UND Gatter Inverter
Analog- schaltungen		Verstärker Filter
Bauteile		Transistoren Dioden
Physik		Elektronen

Überblick der heutigen Vorlesung



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ Sequentielle Logik
 - ▶ Speicherelemente (Fortsetzung)
 - ▶ Synchrone sequentielle Logik
- ▶ Zeitverhalten synchroner sequentieller Logik
- ▶ Parallelität



Harris 2013/2016
Kap. 3.2 - 3.3, 3.5 - 3.7

Agenda



TECHNISCHE
UNIVERSITÄT
DARMSTADT

1. Speicherelemente (Fortsetzung)

2. Synchrone sequentielle Logik

3. Zeitverhalten synchroner sequentieller Logik

4. Parallelität

5. Zusammenfassung

Anwendungs- software		Programme
Betriebs- systeme		Gerätetreiber
Architektur		Befehle Register
Mikro- architektur		Datenpfade Steuerung
Logik		Addierer Speicher
Digital- schaltungen		UND Gatter Inverter
Analog- schaltungen		Verstärker Filter
Bauteile		Transistoren Dioden
Physik		Elektronen

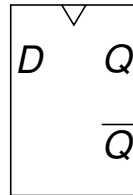
Wiederholung: D-Flip-Flop LQ8-2 RQ8-2



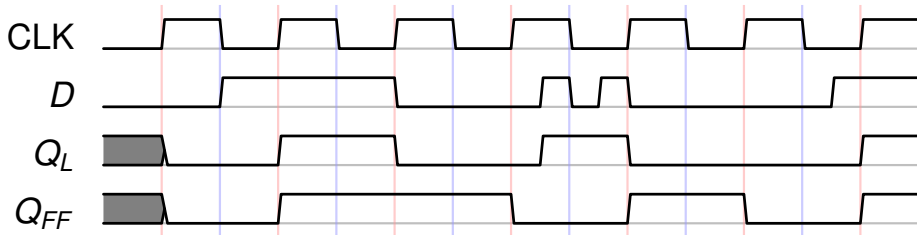
TECHNISCHE
UNIVERSITÄT
DARMSTADT

► Taktflanken-gesteuert

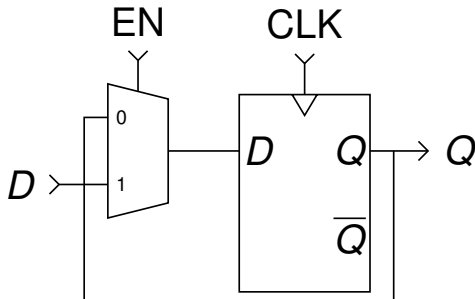
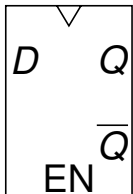
- genau bei steigender CLK Flanke wird $Q = D$
- es wird der Wert von D übernommen, der **unmittelbar vor** der Taktflanke anliegt



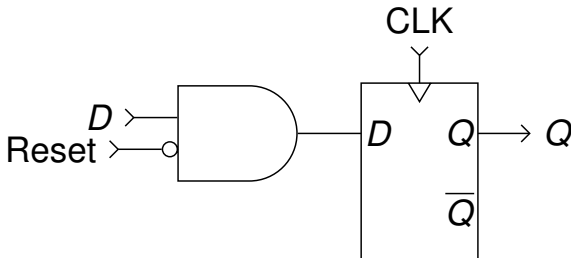
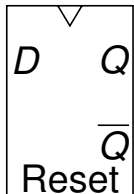
► Vergleich D-Latch und D-Flip-Flop:



- ▶ Freigabeeingang (enable EN) steuert, wann Daten gespeichert werden
 - ▶ $EN = 1 \rightarrow D$ wird bei steigender CLK-Flanke gespeichert
 - ▶ $EN = 0 \rightarrow Q$ bleibt auch bei steigender CLK-Flanke unverändert
- ▶ Anwendungsbeispiele
 - ▶ Zähler
 - ▶ Speicher mit Adressdecoder



- ▶ Reset setzt internen Zustand unabhängig von D auf 0
 - ▶ synchron: nur zur steigenden Taktflanke wirksam
 - ▶ asynchron: jederzeit (unabhängig von CLK)
- ▶ Anwendungsbeispiele
 - ▶ sequentielle Schaltung in definierten Ausgangszustand versetzen
- ▶ setzbare Flip-Flops analog



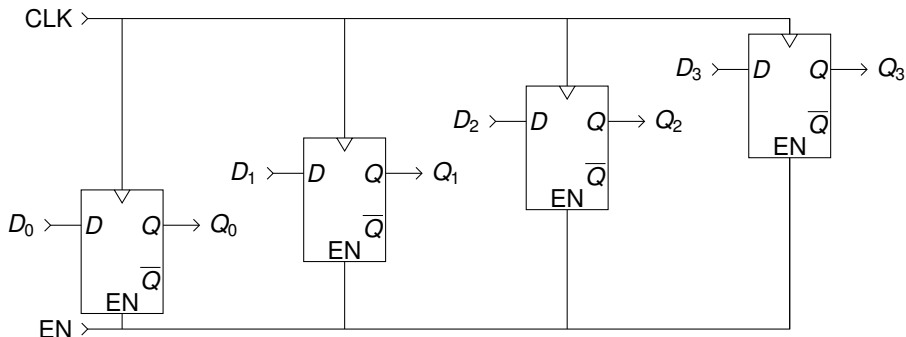
Anwendungsbeispiel: (Shift-)Register

LQ9-1 RQ9-1



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ Register bestehend aus parallelen D-Flip-Flops
- ▶ Bei Shift-Register ist $D_i = Q_{i-1}$



Agenda



TECHNISCHE
UNIVERSITÄT
DARMSTADT

1. Speicherelemente (Fortsetzung)

2. Synchrone sequentielle Logik

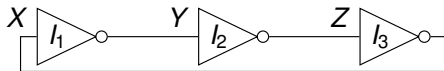
3. Zeitverhalten synchroner sequentieller Logik

4. Parallelität

5. Zusammenfassung

Anwendungs- software		Programme
Betriebs- systeme		Gerätetreiber
Architektur		Befehle Register
Mikro- architektur		Datenpfade Steuerung
Logik		Addierer Speicher
Digital- schaltungen		UND Gatter Inverter
Analog- schaltungen		Verstärker Filter
Bauteile		Transistoren Dioden
Physik		Elektronen

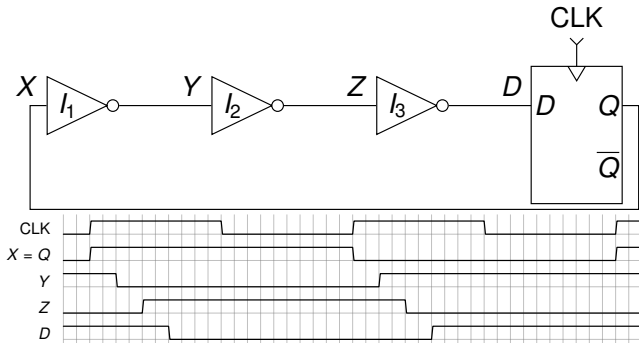
- ▶ alle nicht-kombinatorischen Schaltungen
- ▶ erlaubt Rückkopplungen, bspw:



⇒ instabile (oszillierende) Schaltung

- ▶ Verhalten abhängig von Herstellungsprozess, Spannung, Temperatur
- ▶ nicht vorhersagbar

- ▶ Rückkopplungen durch Register aufbrechen
 - ▶ halten den Zustand der Schaltung
 - ▶ ändern Zustand nur zur Taktflanke
- ⇒ gesamte Schaltung *synchronisiert* mit Taktflanke





- ▶ Regeln für Aufbau
 - ▶ jedes Schaltungselement ist entweder Register oder kombinatorische Schaltung
 - ▶ mindestens ein Schaltungselement ist ein Register
 - ▶ alle Register werden durch gleiches Taktsignal gesteuert
 - ▶ jeder zyklische Pfad enthält mindestens ein Register
- ▶ Anwendungsbeispiele
 - ▶ Pipelines (diese Vorlesung)
 - ▶ Endliche Zustandsautomaten (nächste Vorlesung)
- ▶ Wie schnell kann so eine Schaltung betrieben werden?
 - ⇒ Was ist die kürzeste Taktperiode?
- ▶ Zu untersuchen sind das Zeitverhalten von:
 - ▶ kombinatorischen Schaltungen
 - ▶ Registern

Agenda



TECHNISCHE
UNIVERSITÄT
DARMSTADT










1. Speicherelemente (Fortsetzung)

2. Synchrone sequentielle Logik

3. Zeitverhalten synchroner sequentieller Logik

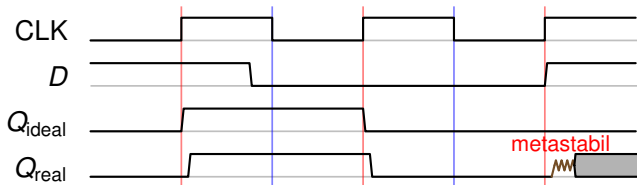
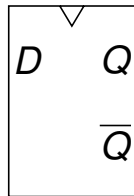
4. Parallelität

5. Zusammenfassung

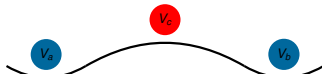
Anwendungs- software		Programme
Betriebs- systeme		Gerätetreiber
Architektur		Befehle Register
Mikro- architektur		Datenpfade Steuerung
Logik		Addierer Speicher
Digital- schaltungen		UND Gatter Inverter
Analog- schaltungen		Verstärker Filter
Bauteile		Transistoren Dioden
Physik		Elektronen

Zeitverhalten eines Registers (Flip-Flop)

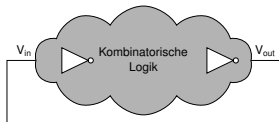
- ▶ Flip-Flop übernimmt D zur steigenden Taktflanke
- ▶ Was passiert bei zeitgleicher Änderung von D und CLK?
- ▶ bisher vereinfachte Annahme:
 - ▶ Wert unmittelbar vor der Taktflanke wird übernommen
- ▶ Aber:
 - ▶ Was heißt “unmittelbar”?
 - ▶ Wie schnell wird neuer Zustand am Ausgang sichtbar?
 - ▶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



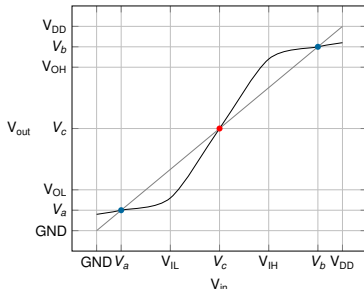
- ▶ In Digitaltechnik:
 - ▶ zeitlich begrenzter und undefinierter Zustand
 - ▶ geht nach zufälliger Verzögerung in einen stabilen Zustand über



- ▶ Beispiel:



- ▶ Rückkopplung stabil für $V_{out} = V_{in}$
 - V_a repräsentiert 0
 - V_b repräsentiert 1
 - V_c im "verbotenen" Spannungsbereich
 - ▶ kleine Änderung an V_{in}
→ große Änderung an V_{out}



Zeitanforderungen an DFF Eingangssignal



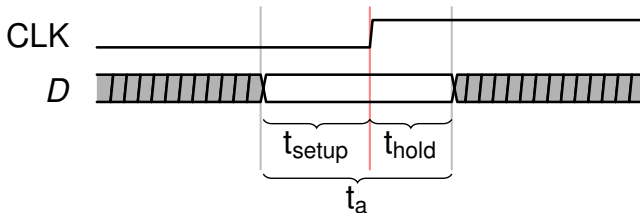
- ▶ Dateneingang D muss im Abtast-Zeitfenster um Taktflanke stabil sein, um Metastabilität zu vermeiden

t_{setup} Zeitintervall vor Taktflanke, in dem D stabil sein muss ("setup time")

t_{hold} Zeitintervall nach Taktflanke, in dem D stabil sein muss ("hold time")

t_a Abtastzeitfenster: $t_a = t_{\text{setup}} + t_{\text{hold}}$ ("aperture time")

- ▶ Größenordnung: 10 ps

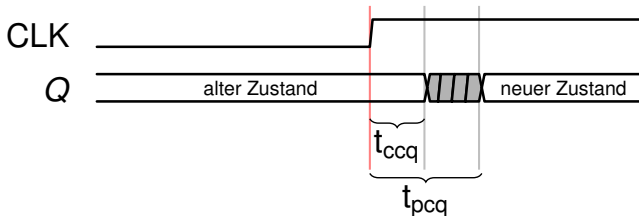


Zeitcharakteristik des DFF Ausgangssignals



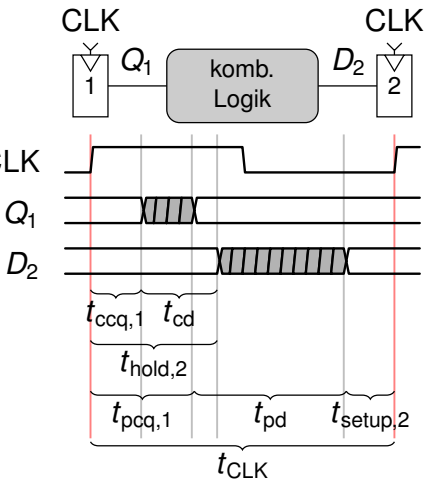
TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ Verzögerung des Registerausgangs relativ zur steigenden Taktflanke
 - ▶ Kontaminationsverzögerung (t_{ccq}): kürzeste Zeit bis Q umschaltet (“contamination delay clock-to-Q”)
 - ▶ Laufzeitverzögerung (t_{pcq}): längste Zeit bis Q sich stabilisiert (“propagation delay clock-to-Q”)
- ▶ Größenordnung: 10 ps





- ▶ kombinatorische Logik zwischen zwei Registern hat min. Verzögerung t_{cd} und max. Verzögerung t_{pd}
 - ▶ D_2 abhängig von Verzögerungen der Gatter und des *ersten* Registers CLK
- ⇒ Timing-Bedingungen des *zweiten* Registers müssen erfüllt werden
- ▶ $t_{ccq,1} + t_{cd} \geq t_{hold,2}$
 - ▶ $t_{pcq,1} + t_{pd} + t_{setup,2} \leq t_{CLK}$
- ⇒ maximale Taktrate wird durch *kritischen Pfad* bestimmt
- ▶ $f_{CLK} = \frac{1}{t_{CLK}} \leq \frac{1}{t_{pcq,1} + t_{pd} + t_{setup,2}}$



Beispiel: Analyse der Timing-Bedingungen



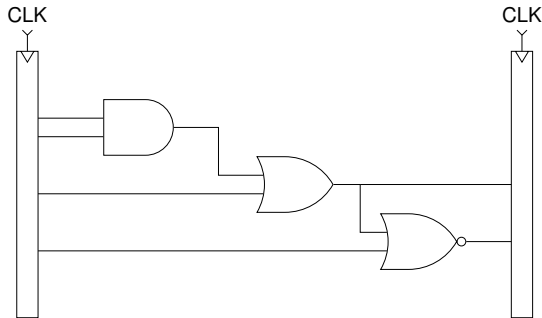
TECHNISCHE
UNIVERSITÄT
DARMSTADT

► 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



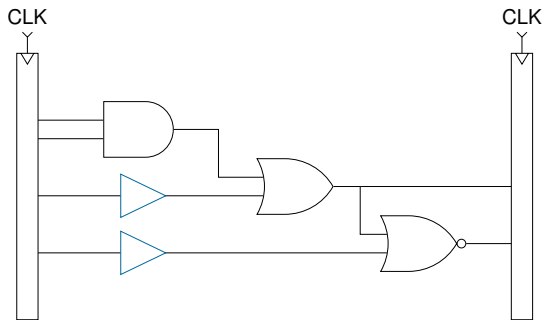
TECHNISCHE
UNIVERSITÄT
DARMSTADT

► 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:

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



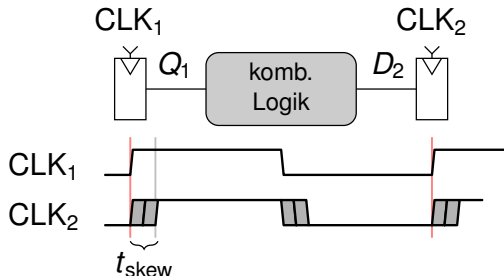
► Timing-Bedingungen:

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

Taktverschiebung (clock skew)



- ▶ 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



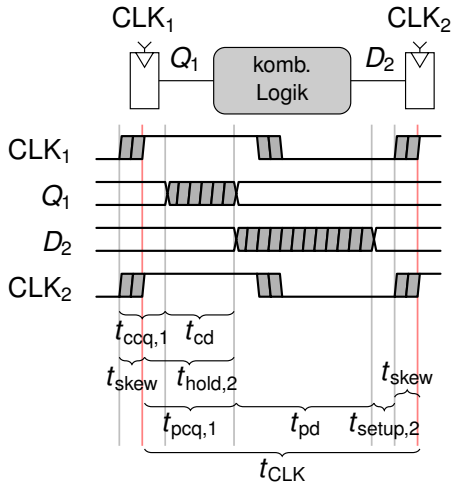
TECHNISCHE
UNIVERSITÄT
DARMSTADT

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

- $t_{ccq,1} + t_{cd} \geq t_{hold,2} + t_{skew}$
- $t_{pcq,1} + t_{pd} + t_{setup,2} + t_{skew} \leq t_{CLK}$

- ▶ i.d.R. wird Timing durch t_{skew} enger

- ▶ Taktfrequenz kann durch t_{skew} auch steigen, wenn CLK₂ sicher nach CLK₁ schaltet



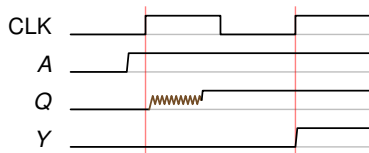
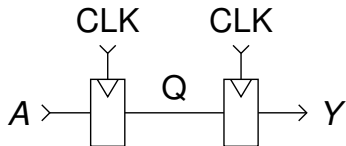
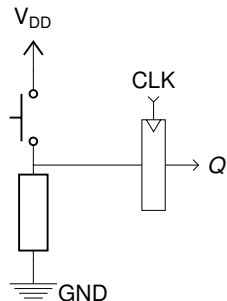
► asynchrone Eingänge:

- Benutzereingaben
- Kommunikationssignale von externen ICs

⇒ Timing-Bedingungen können nicht garantiert werden

► Schieberegister für Synchronisation

- erstes Flip-Flop kann metastabil werden
 - kippt i.d.R. vor nächster Taktflanke in stabilen Zustand
- ⇒ zweites Flip-Flop wird nicht metastabil





Pause & Umfrage bis hier

Agenda



TECHNISCHE
UNIVERSITÄT
DARMSTADT

1. Speicherelemente (Fortsetzung)

2. Synchrone sequentielle Logik

3. Zeitverhalten synchroner sequentieller Logik

4. Parallelität

5. Zusammenfassung

Anwendungs- software		Programme
Betriebs- systeme		Gerätetreiber
Architektur		Befehle Register
Mikro- architektur		Datenpfade Steuerung
Logik		Addierer Speicher
Digital- schaltungen		UND Gatter Inverter
Analog- schaltungen		Verstärker Filter
Bauteile		Transistoren Dioden
Physik		Elektronen



- ▶ 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 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



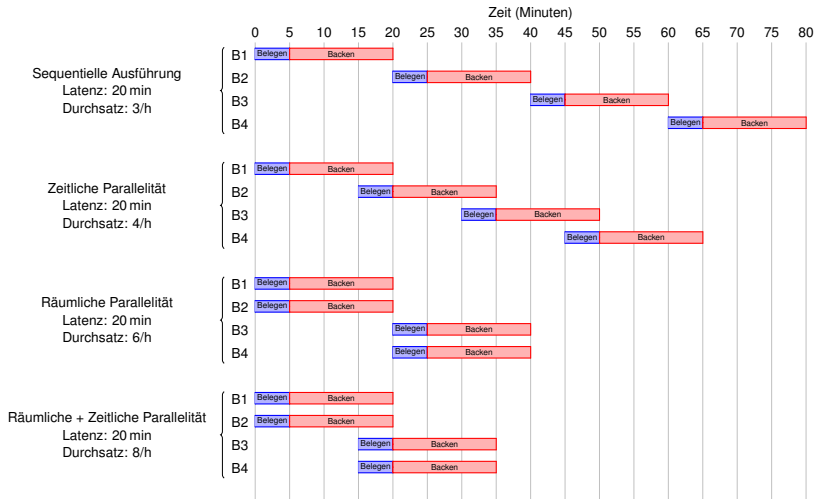
- ▶ Annahmen:
 - ▶ genug Teig ist fertig
 - ▶ 5 Minuten zum Belegen eines Bleches
 - ▶ 15 Minuten Backzeit
- ▶ *sequentiell*: ein Blech nach dem anderen belegen und backen
- ▶ *zeitlich parallel*: nächstes Blech belegen, während erstes noch im Ofen ist
- ▶ *räumlich parallel*: zwei Bäcker, jeweils mit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



Beispiel: Plätzchen backen



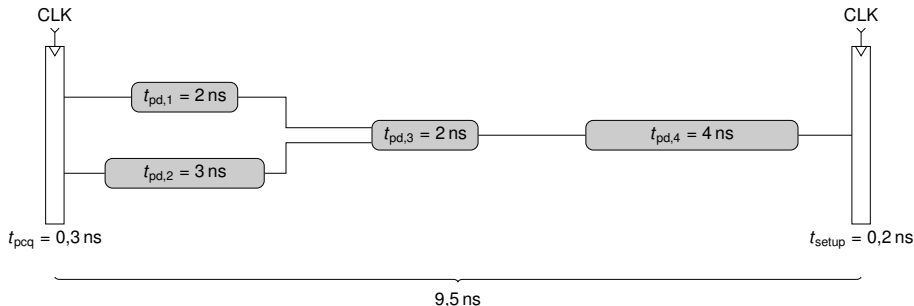
TECHNISCHE
UNIVERSITÄT
DARMSTADT



Beispiel: Pipelining in Schaltungen Ohne Pipeline-Register



TECHNISCHE
UNIVERSITÄT
DARMSTADT



► $f_{CLK} \leq \frac{1}{0,3+3+2+4+0,2 \text{ ns}} = \frac{1}{9,5 \text{ ns}} = 105 \text{ MHz}$

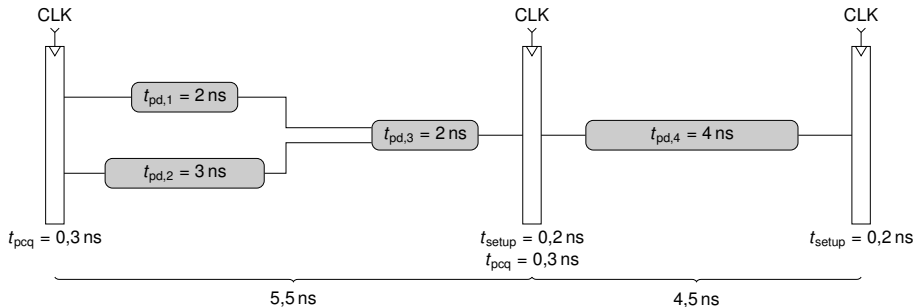
► Latenz: 1 Takt = 9,5 ns

Beispiel: Pipelining in Schaltungen

Zwei Pipeline-Stufen



TECHNISCHE
UNIVERSITÄT
DARMSTADT



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

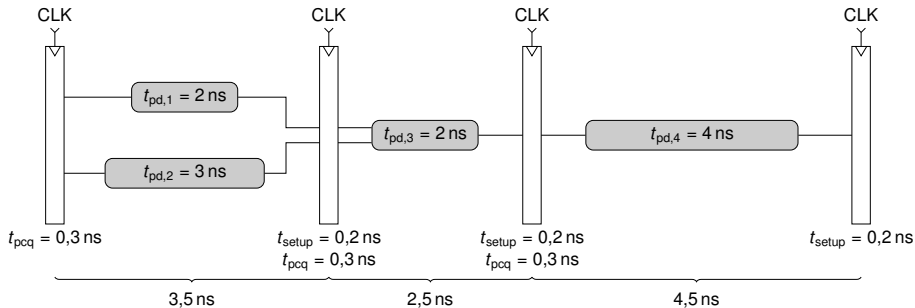
► Latenz: 2 Takte = $2 \cdot 5,5 \text{ ns} = 11 \text{ ns}$

Beispiel: Pipelining in Schaltungen

Drei Pipeline-Stufen



TECHNISCHE
UNIVERSITÄT
DARMSTADT



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

► Latenz: 3 Takte = $3 \cdot 4,5\text{ ns} = 13,5\text{ ns}$

- ▶ Pipelinestufen sollten möglichst gleich lang sein (“ausbalanciert”)
 - ▶ längste Stufe bestimmt maximale Taktfrequenz f_{CLK}
 - ▶ Latenz = # Pipelinestufen / Taktfrequenz
- ▶ mehr Pipelinestufen
 - ▶ höherer Durchsatz (mehr Ergebnisse pro Zeiteinheit), da höhere Taktfrequenz
 - ▶ 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
- ▶ Ausführliche Behandlung s. Befehlsverarbeitung von Prozessoren in LV Rechnerarchitektur

Agenda



TECHNISCHE
UNIVERSITÄT
DARMSTADT

1. Speicherelemente (Fortsetzung)

2. Synchrone sequentielle Logik

3. Zeitverhalten synchroner sequentieller Logik

4. Parallelität

5. Zusammenfassung

Anwendungs- software		Programme
Betriebs- systeme		Gerätetreiber
Architektur		Befehle Register
Mikro- architektur		Datenpfade Steuerung
Logik		Addierer Speicher
Digital- schaltungen		UND Gatter Inverter
Analog- schaltungen		Verstärker Filter
Bauteile		Transistoren Dioden
Physik		Elektronen



- ▶ Sequentielle Logik
 - ▶ Speicherelemente
 - ▶ Synchrone Schaltungen
- ▶ Zeitverhalten synchroner sequentieller Schaltungen
- ▶ Parallelität
- ▶ Nächste Vorlesung behandelt
 - ▶ Endliche Zustandsautomaten