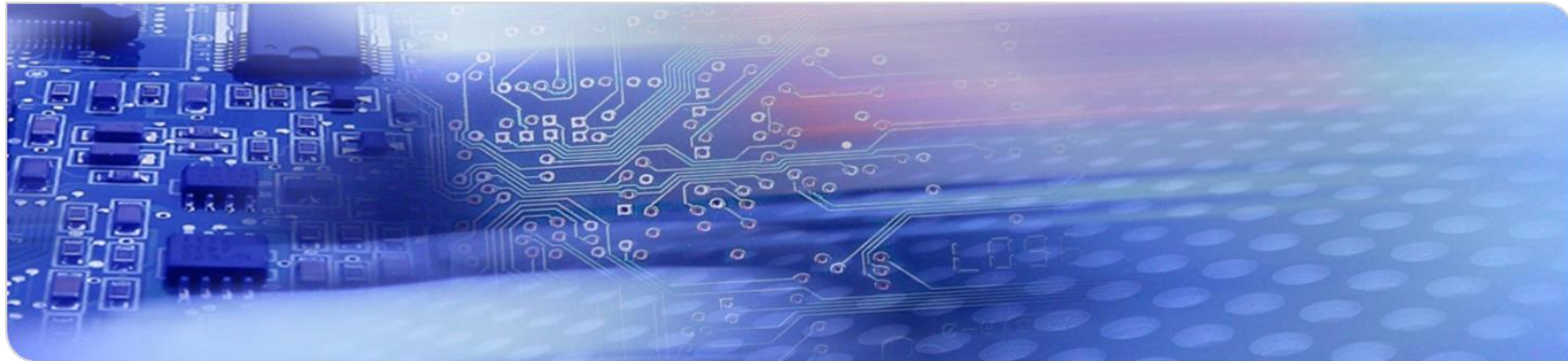


# Hardware/Software Co-Design

Übung 5 - Wiederholung  
M.Sc. Fabian Lesniak

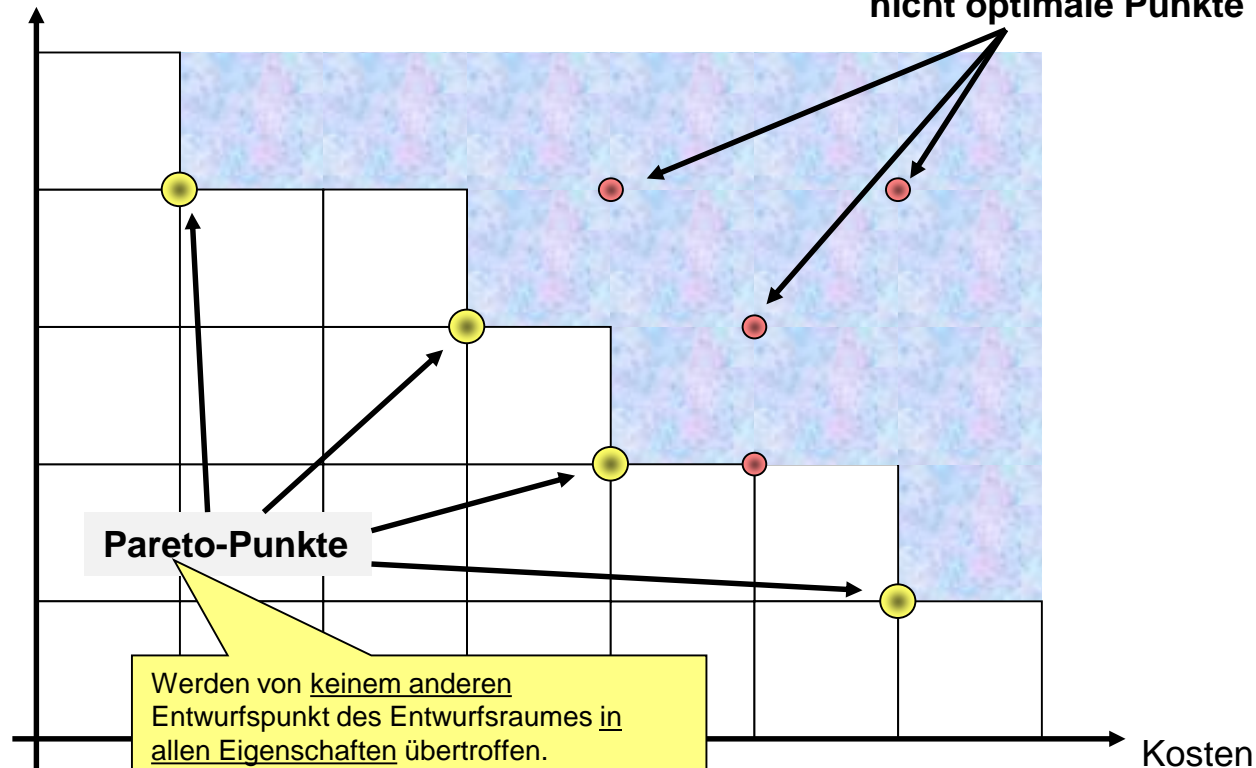


# Agenda

- Wiederholung ausgewählter Themen
  - Schätzung der Entwurfsqualität
    - Exaktheit & Treue
  - Hardware-Performanz
    - Taktschlupfminimierung

# 3.3 Optimale Entwurfspunkte

Ausführungszeit



## 3.4 Metriken und Exaktheit der Schätzung

### ■ Qualitätsmaße/Metriken:

- Performance, Kosten (Fläche), Leistungsaufnahme, Energiebedarf, Zuverlässigkeit, Testbarkeit, Time-to-market, ...

Built in Self-tests, Scan-Path Schieberegister

### ■ Definition: Exaktheit

- Sei  $E(D)$  eine **abgeschätzte** und  $M(D)$  die **exakte** (gemessene) **Metrik** einer **Implementierung D**.
- Die **Exaktheit A** der Abschätzung ist gegeben durch :  $A = 1 - \frac{|E(D) - M(D)|}{|M(D)|}$

## 3.4 Treue der Schätzung

### ■ Definition :



- Sei  $D = \{ D_1, D_2, \dots, D_n \}$  eine Menge von Implementierungen
  - Bsp.  $D_1 = \text{ASIC}$ ,  $D_2 = \text{FPGA}$
- Die Treue  $F$  ist ein Maß für die Zuverlässigkeit einer Schätzmethode, welche die Implementierungen  $D_i$  gegeneinander vergleicht:

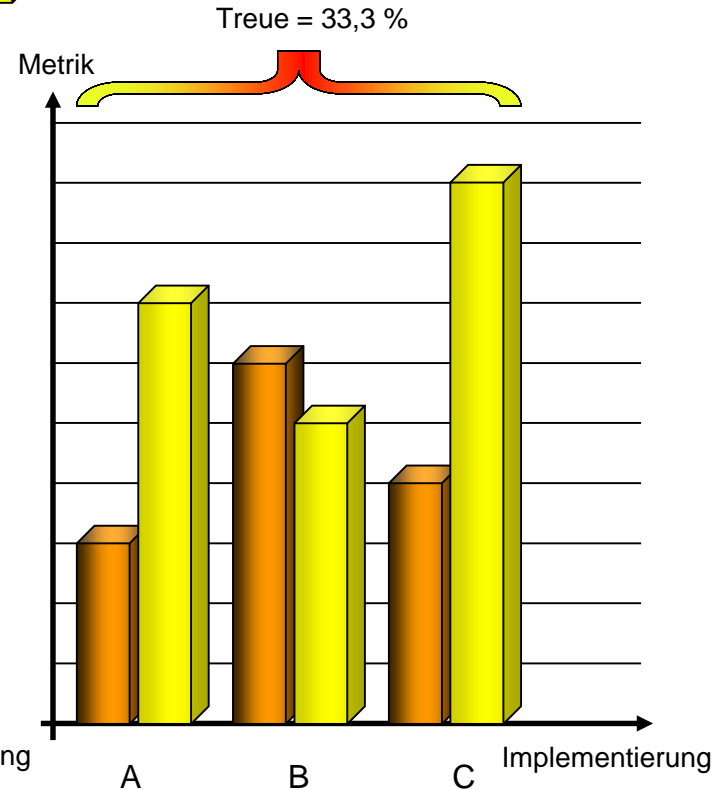
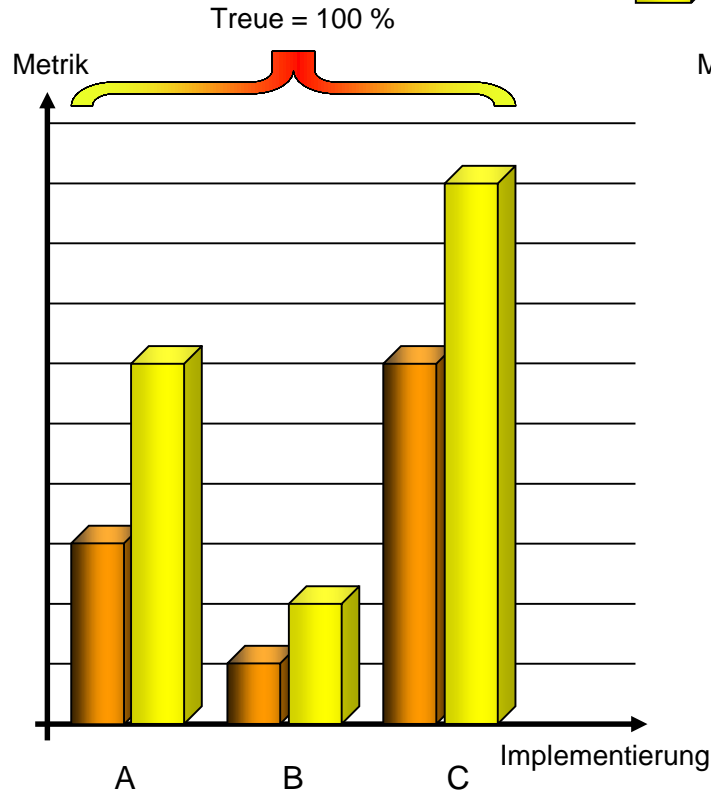
|       | $D_1$ | $D_2$ | $D_3$ |
|-------|-------|-------|-------|
| $D_1$ |       |       |       |
| $D_2$ |       |       |       |
| $D_3$ |       |       |       |

$$F = 100\% \cdot \frac{2}{n(n-1)} \cdot \sum_{i=1}^n \sum_{j=i+1}^n \mu_{i,j}$$

$$\mu_{i,j} = \begin{cases} 1 & \text{if } (E(D_i) > E(D_j) \wedge M(D_i) > M(D_j)) \vee \\ & (E(D_i) < E(D_j) \wedge M(D_i) < M(D_j)) \vee \\ & (E(D_i) = E(D_j) \wedge M(D_i) = M(D_j)) \\ 0 & \text{else} \end{cases}$$

## 3.4 Beispiel: Treue

$E(D)$   geschätzt  
 $M(D)$   gemessen



# Inhalt

- 3.1 Abstraktionsebenen
- 3.2 Systemsynthese
- 3.3 Graphenmodelle für Kontroll- und Datenfluss
- 3.4 Parameter von Schätzverfahren
- **3.5 Schätzung von Hardwaremetriken**
- 3.6 Schätzung von Softwaremetriken

# 3.5 Hardware - Performanz

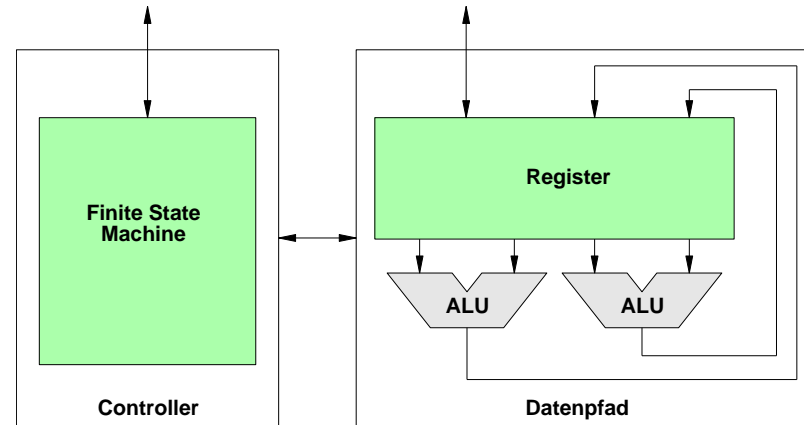
- Taktperiode T
  - Beeinflusst von Technologie, Ressourcen
- Latenz L
  - Anzahl der Taktschritte, abhängig von Datenpfadoptimierung (Logikminimierung, Scheduling, Retiming)

- Ausführungszeit:

- $T_{ex} = T \cdot L$

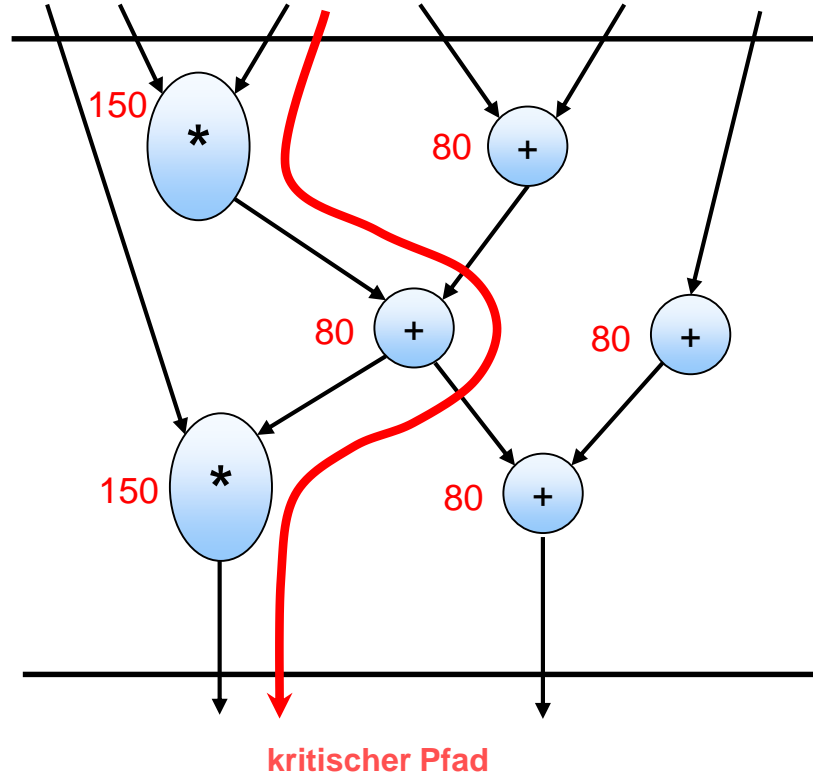
- Durchsatz:

- $R = 1 / T_{ex}$   
(entspr. zeitl. Häufigkeit des Berechnungsergebnisses)



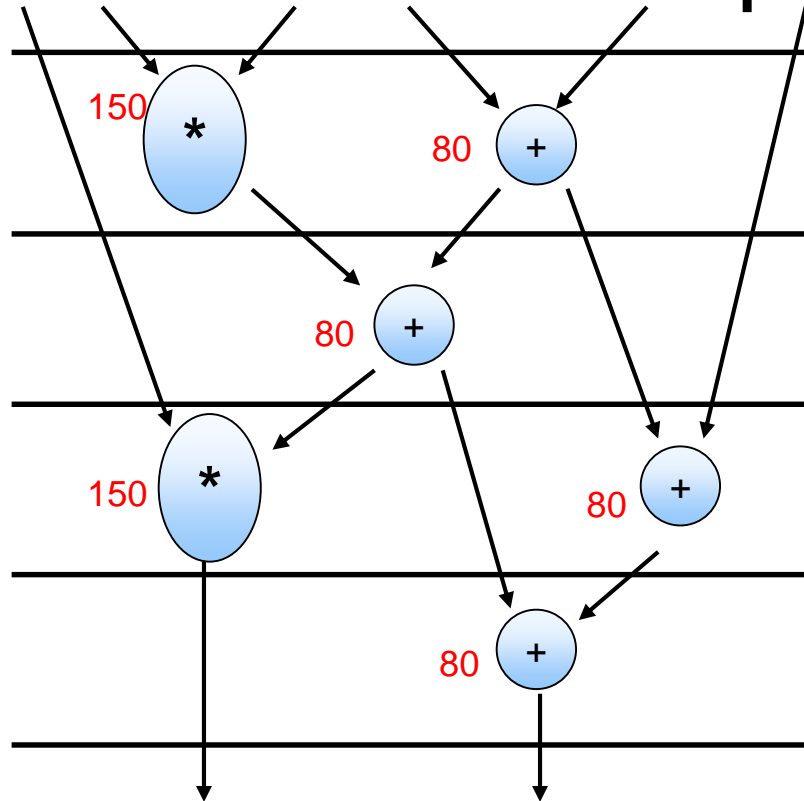


## 3.5 Hardware-Performanz: Beispiel (I)



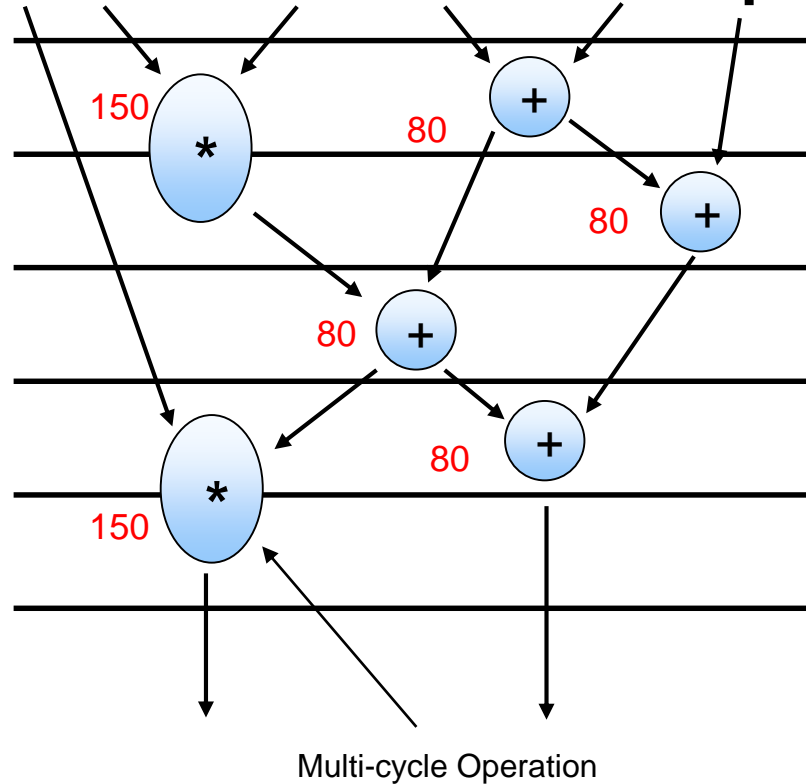
- Taktperiode  $T=380$  ns
- Latenz  $L=1$
- Ausführungszeit  $T_{ex}=380$  ns
- Ressourcen: 2 \*, 4 +

## 3.5 Hardware-Performanz: Beispiel (II)



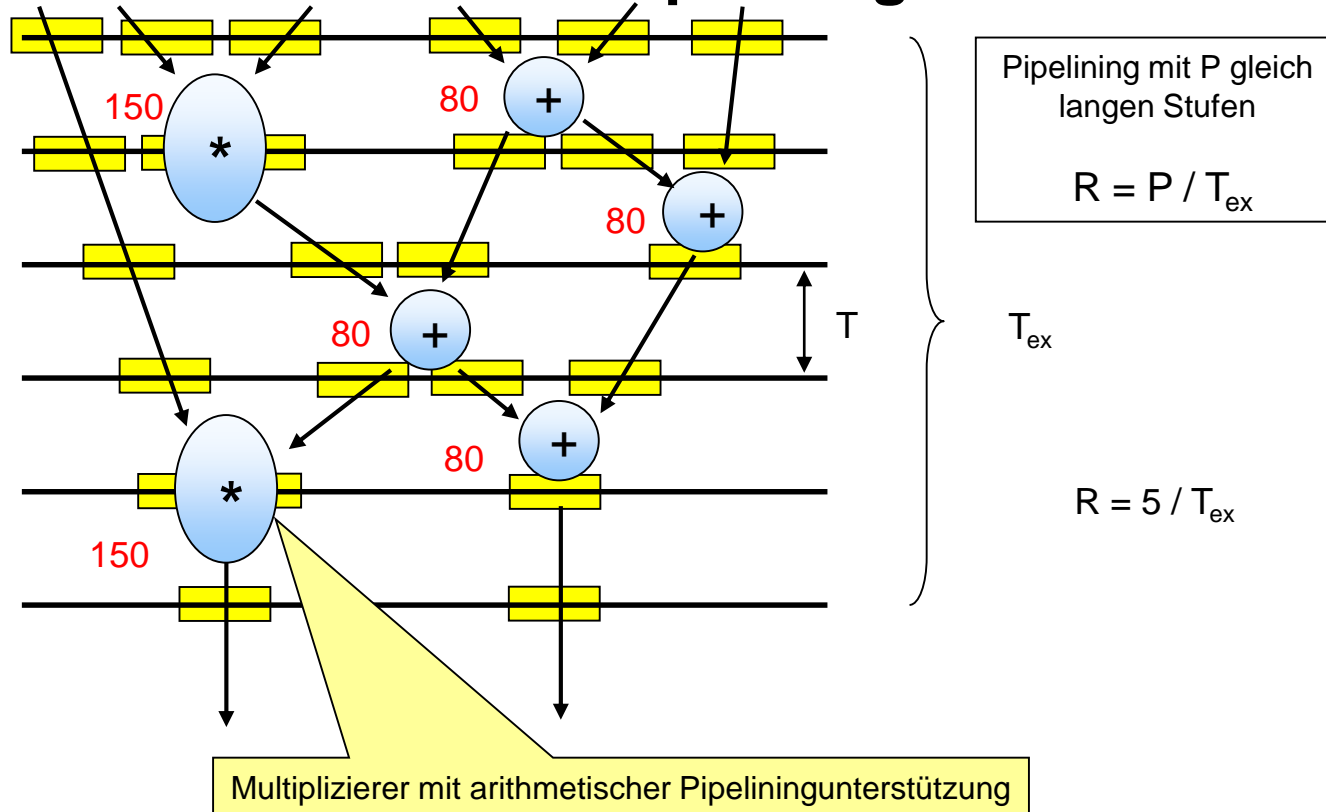
- Taktperiode  $T=150$  ns
- Latenz  $L=4$
- Ausführungszeit  $T_{ex}=600$  ns
- Ressourcen: 1 \*, 1 +
- Keine Änderung der Inputs über die 4 Takte.
- Erst danach neue Inputs anlegen.

## 3.5 Hardware-Performanz: Beispiel (III)



- Taktperiode  $T=80$  ns
- Latenz  $L=5$
- Ausführungszeit  $T_{ex}=400$  ns
- Ressourcen: 1 \*, 1 +
- **Vergleich mit Beispiel 1:**
  - Weniger Ressourcen nötig
  - Taktperiode kleiner
  - $T_{ex}$  etwas größer

# 3.5 Hardware-Performanz: Pipelining



## 3.5 Hardware-Performanz: Schätzung der Taktperiode

- Funktionale Einheiten  $v_k$  mit jew. Delay  $delay(v_k)$

- Methode der maximalen Operatorverzögerung

$$T = \max_k (delay(v_k))$$

- Nachteil: es muss mit einer erheblichen Unterauslastung der schnelleren Funktionseinheiten gerechnet werden.

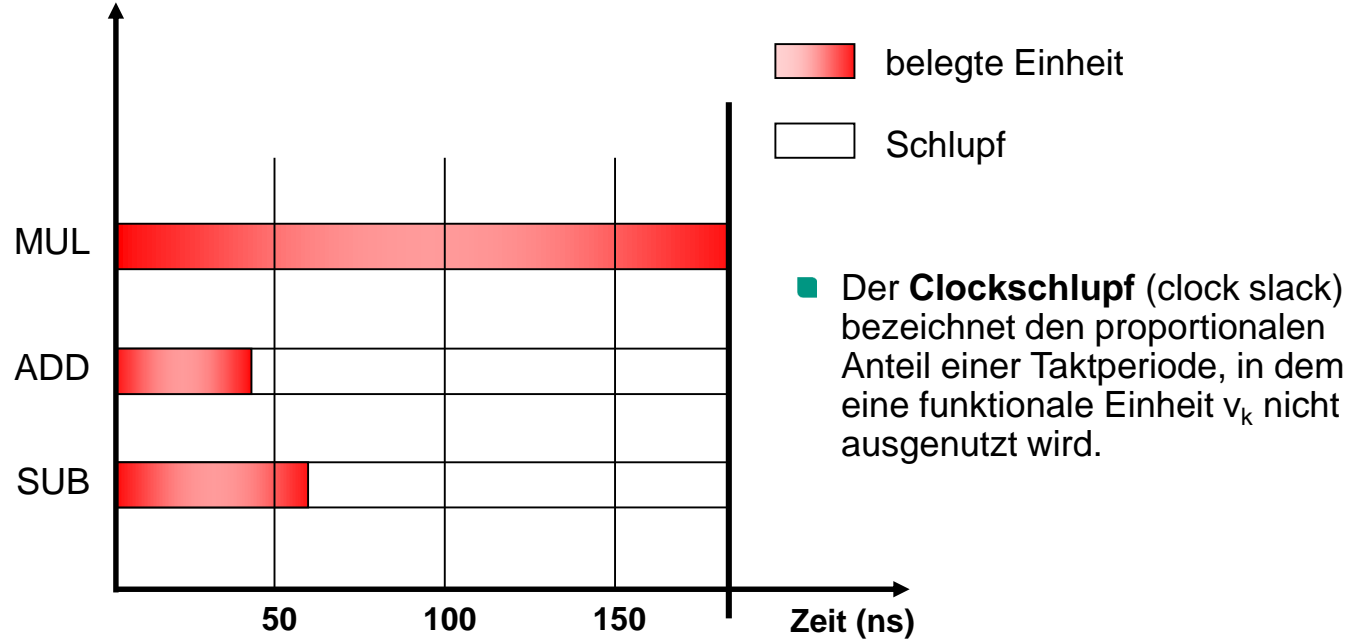
- Methode der Minimierung des Taktschlupfs (clock slack)

- Suche im Intervall  $T_{\min} \dots T_{\max}$  nach der Taktperiode  $T$  mit maximaler Taktauslastung (minimalem Taktschlupf), Scheduling oft als Nachfolgeschritt zur Bestimmung der Gesamtausführungszeit  $T_{\text{exec}}$ .

- ILP-Suche:

- Modellierung eines Latenzminierungsproblems als ILP für diskrete Werte der Taktperiode zur Minimierung von  $T_{\text{exec}}$ .

## 3.5 Hardware-Performanz: *Taktschlupf*



$$slack(T, v_k) = ( \lceil delay(v_k) / T \rceil ) \cdot T - delay(v_k)$$

## 3.5 Hardware-Performanz: Taktschlupfminimierung

- Mit  $occ(v_k)$ , der Anzahl der Operationen vom Typ  $v_k$ , und  $|V_T|$ , der Anzahl unterschiedlicher Operationstypen, ist der mittlere Schlupf für eine Taktperiode  $T$ :

$$avgslack(T) = \frac{\sum_{k=1}^{|V_T|} (occ(v_k) \cdot slack(T, v_k))}{\sum_{k=1}^{|V_T|} occ(v_k)}$$

- Ein geringerer mittlerer Taktschlupf impliziert auch eine geringere Ausführungszeit für eine feste Anzahl Ressourcen.

### ■ Taktauslastung

$$util(T) = 1 - \frac{avgslack(T)}{T}$$

- bezeichnet die prozentuale mittlere Auslastung aller Funktionseinheiten.

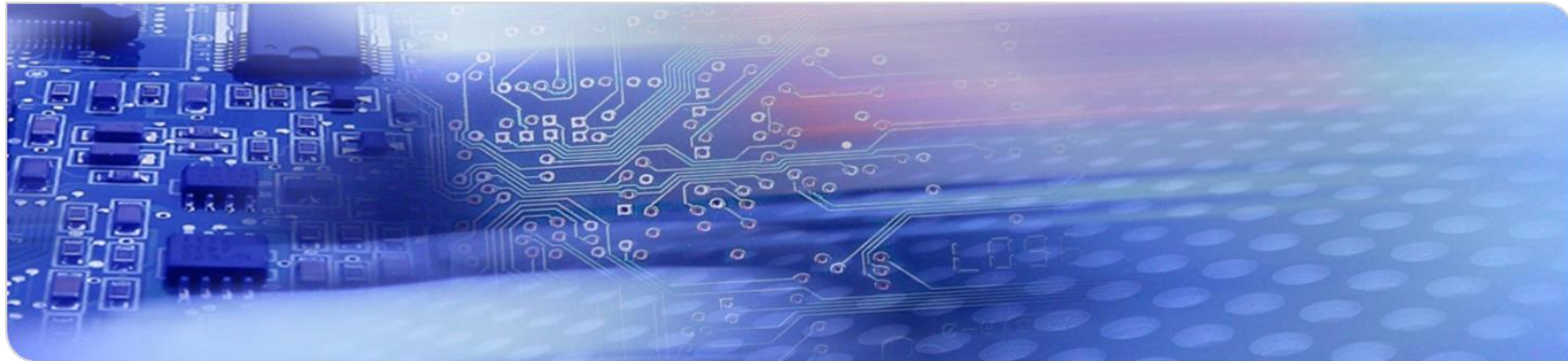
# Arbeitsphase

- Aufgabe 3.01: Design Space, Pareto Punkte
  - Realisierungsmöglichkeiten mit verfügbaren Komponenten je nach Kosten und Ausführungsgeschwindigkeit
  
- Aufgabe 3.02: Exaktheit & Treue
  - Metriken und Entwurfsqualität mit geschätzten und gemessenen Werte
  
- Aufgabe 3.03: Taktschlupf
  - Taktschlupfminimierung, Slack, mittlerer Schlupf



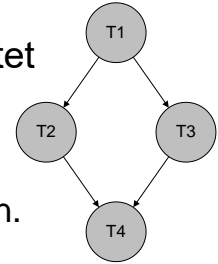
# Hardware/Software Co-Design

Übung 5 - Lösung  
M.Sc. Fabian Lesniak



# Aufgabe 3.01: Design Space, Pareto Punkte

- Task-Graph: Tasks (T1 ... T4), wobei T4 erst ausgeführt werden, wenn T2 und T3 abgearbeitet wurden. T2 und T3 hingegen können parallel ausgeführt werden.
- Verfügbaren Komponenten: MIPS, DSP, FPGA, ASIC
  - Jeweils maximale Anzahl, Kosten und Ausführungsgeschwindigkeit
  - Bspw. kostet der MIPS Prozessor 200 Einheiten und kann Task T1 in 5 und T4 in 2ms ausführen.



| Komponente | Anzahl | Kosten [€] | Ausführungszeit [ms] |    |     |    |
|------------|--------|------------|----------------------|----|-----|----|
|            |        |            | T1                   | T2 | T3  | T4 |
| MIPS       | 1      | 200        | 5                    | -  | -   | 2  |
| DSP        | 1      | 100        | -                    | 20 | 18  | 5  |
| FPGA       | 1      | 250        | -                    | 12 | 10  | -  |
| ASIC       | 1      | 400        | -                    | -  | 0,8 | -  |

- Vervollständigen Sie die Ausführungszeit und Kosten der folgenden Tabelle. Sie zeigt sämtliche Realisierungsmöglichkeiten, welcher Task auf welchem Prozessor ausgeführt werden kann.
- Tragen Sie die Lösungen aus Aufgabe a) in folgendes Kosten-Zeitdiagramm ein und markieren Sie die Pareto-Punkte.
- Was passiert, wenn die Anzahl der Komponenten nicht auf 1 beschränkt ist? Welche zusätzlichen Design-Möglichkeiten ergeben sich? Verändert sich die Menge der Pareto-Punkte?

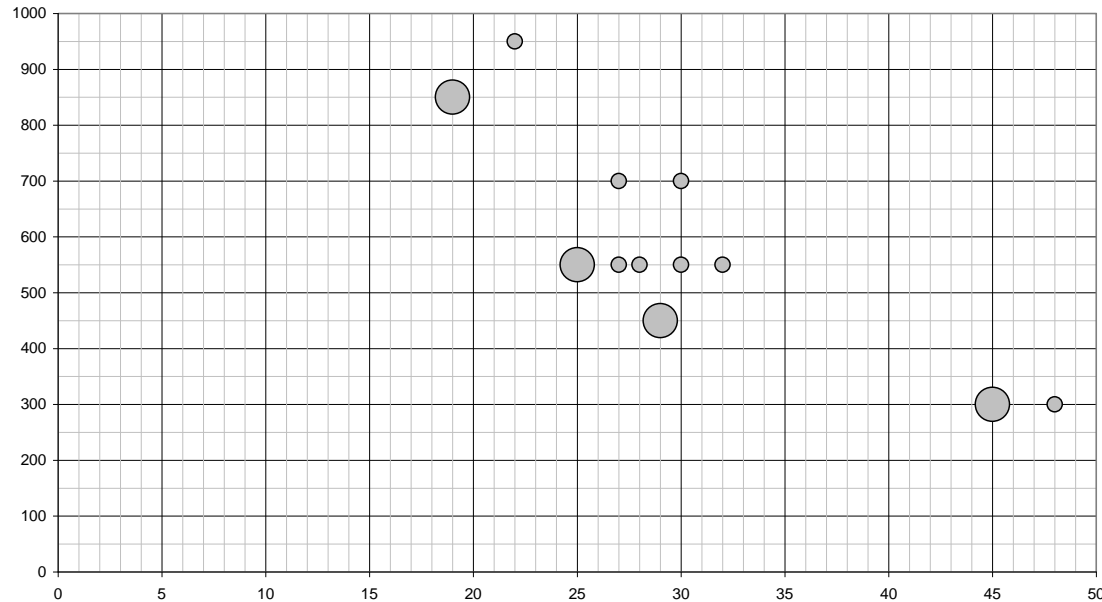
# Aufgabe 3.01: Design Space, Pareto Punkte

- Vervollständigen Sie die Ausführungszeit und Kosten der folgenden Tabelle. Sie zeigt sämtliche Realisierungsmöglichkeiten, welcher Task auf welchem Prozessor ausgeführt werden kann.

| #  | Tasks |      |      |      | Ausführungszeit [ms]         | Kosten                |
|----|-------|------|------|------|------------------------------|-----------------------|
|    | T1    | T2   | T3   | T4   |                              |                       |
| 1  | MIPS  | DSP  | DSP  | MIPS | $5 + 20 + 18 + 2 = 45$       | $200+100= 300$        |
| 2  | MIPS  | DSP  | DSP  | DSP  | $5 + 20 + 18 + 5 = 48$       | $200+100= 300$        |
| 3  | MIPS  | DSP  | FPGA | MIPS | $5 + \max(20, 18) + 2 = 27$  | $200+100+250= 550$    |
| 4  | MIPS  | DSP  | FPGA | DSP  | $5 + \max(20, 18) + 5 = 30$  | $200+100+250= 550$    |
| 5  | MIPS  | DSP  | ASIC | MIPS | $5 + \max(20, 0.8) + 2 = 27$ | $200+100+400= 700$    |
| 6  | MIPS  | DSP  | ASIC | DSP  | $5 + \max(20, 0.8) + 5 = 30$ | $200+100+400= 700$    |
| 7  | MIPS  | FPGA | DSP  | MIPS | $5 + \max(12, 18) + 2 = 25$  | $200+250+100= 550$    |
| 8  | MIPS  | FPGA | DSP  | DSP  | $5 + \max(12, 18) + 5 = 28$  | $200+250+100= 550$    |
| 9  | MIPS  | FPGA | FPGA | MIPS | $5 + 12 + 10 + 2 = 29$       | $200+250= 450$        |
| 10 | MIPS  | FPGA | FPGA | DSP  | $5 + 12 + 10 + 5 = 32$       | $200+250+100= 550$    |
| 11 | MIPS  | FPGA | ASIC | MIPS | $5 + \max(12, 0.8) + 2 = 19$ | $200+250+400= 850$    |
| 12 | MIPS  | FPGA | ASIC | DSP  | $5 + \max(12, 0.8) + 5 = 22$ | $200+250+400+100=950$ |

# Lösung Aufgabe 3.01b: Design Space, Pareto Punkte

- Tragen Sie die Lösungen aus Aufgabe a) in folgendes Kosten-Zeitdiagramm ein und markieren Sie die Pareto-Punkte.



# Lösung Aufgabe 3.01c: Design Space, Pareto Punkte

- Was passiert, wenn die Anzahl der Komponenten nicht auf 1 beschränkt ist? Welche zusätzlichen Design-Möglichkeiten ergeben sich? Verändert sich die Menge der Pareto-Punkte?

| #  | Tasks |      |      |      | Ausführungszeit [ms]        | Kosten                |
|----|-------|------|------|------|-----------------------------|-----------------------|
|    | T1    | T2   | T3   | T4   |                             |                       |
| 13 | MIPS  | DSP  | DSP  | MIPS | $5 + \max(20, 18) + 2 = 27$ | $200+100+100= 400$    |
| 14 | MIPS  | DSP  | DSP  | DSP  | $5 + \max(20, 18) + 5 = 30$ | $200+100+100= 400$    |
| 15 | MIPS  | FPGA | FPGA | MIPS | $5 + \max(12, 10) + 2 = 19$ | $200+250+250= 700$    |
| 16 | MIPS  | FPGA | FPGA | DSP  | $5 + \max(12, 10) + 5 = 22$ | $200+250+250+100=800$ |

- Dadurch ändert sich die Menge der Pareto-Punkte
  - Lösung #15 ist genauso schnell wie Lösung #11, kostet aber weniger.
  - Lösung #13 überdeckt Lösung #9 in Kosten und Ausführungszeit.

- Was ist Pareto-optimal?
- Welche Graphenmodelle gibt es?  
Welche Eigenschaften haben sie?
- In welchen Graphen kann  
Kontrollfluss abgebildet werden?
- Wie wird Programmcode in  
Graphen dargestellt?



## Aufgabe 3.02: Exaktheit und Treue

- In folgender Tabelle sind für vier Entwurfspunkte Metriken und Entwurfsqualität dargestellt, und zwar geschätzt Werte  $E(D)$  sowie die gemessenen Werte  $M(D)$ .

| Entwurfspunkt | $E(D)$ | $M(D)$ |
|---------------|--------|--------|
| W             | 112    | 100    |
| X             | 128    | 137    |
| Y             | 139    | 121    |
| Z             | 205    | 132    |

- Bestimmen Sie die Exaktheit ( $A$ ) des Entwurfspunktes W.
- Bestimmen Sie die Treue ( $F$ ) des Schätzverfahrens.

- Bestimmen Sie die Exaktheit (A) des Entwurfspunktes W.

$$A_W = 1 - \frac{|E(D_W) - M(D_W)|}{|M(D_W)|} = 1 - \frac{|112 - 100|}{|100|} = 1 - \frac{12}{100} = \frac{88}{100} = 0,88$$

- Bestimmen Sie die Treue (F) des Schätzverfahrens

$$\begin{aligned} F &= 100\% \cdot \frac{2}{n(n-1)} \cdot \sum_{i=1}^n \sum_{j=i+1}^n \mu_{i,j} \\ &= 100\% \cdot \frac{2}{4 \cdot 3} \cdot (\mu_{W,X} + \mu_{W,Y} + \mu_{W,Z} + \mu_{X,Y} + \mu_{X,Z} + \mu_{Y,Z}) \\ &= 100\% \cdot \frac{1}{6} \cdot (1 + 1 + 1 + 0 + 0 + 1) = 100\% \cdot \frac{4}{6} = 67\% \end{aligned}$$

$\mu_{W,X}$



| Entwurfspunkt | E(D) | M(D) |
|---------------|------|------|
| W             | 112  | 100  |
| X             | 128  | 137  |
| Y             | 139  | 121  |
| Z             | 205  | 132  |



## Aufgabe 3.03: Taktschlupf

- Gegeben ist eine Menge von funktionale Einheiten  $v_k$ . Die mögliche Taktperiode der Zieltechnologie liegt zwischen 20 und 50 ns.

| Funktionale Einheit | k | delay( $v_k$ ) [ns] | occ( $v_k$ ) |
|---------------------|---|---------------------|--------------|
| MUL                 | 1 | 135                 | 9            |
| ADD                 | 2 | 45                  | 10           |
| SUB                 | 3 | 55                  | 1            |

- Was ist der Taktschlupf?
- Was bringt die Taktschlupfminimierung?
- Berechnen Sie den slack aller funktionalen Einheiten bei einer Taktperiode von 20ns.
- Berechnen Sie den mittleren Schlupf (average slack) für eine Taktperiode von 20ns.
- Raten/Überlegen Sie, welche Taktperiode den niedrigsten mittleren Schlupf hat.

# Lösung Aufgabe 3.03: Taktschlupf

## a) Was ist der Taktschlupf?

- Der Taktschlupf bezeichnet den Anteil einer Taktperiode, der von einer funktionalen Einheit nicht ausgenutzt wird.

## b) Was bringt die Taktschlupfminimierung?

- Bei der Taktschlupfminimierung wird versucht den durchschnittlichen Taktschlupf pro Operation zu minimieren. Dadurch steigt die Taktauslastung der Hardware und somit der Performanz.

# Lösung Aufgabe 3.03: Taktschlupf

c) Berechnen Sie den slack aller funktionalen Einheiten bei einer Taktperiode von 20ns.

$$\text{slack}(20\text{ns}, v_{MUL}) = \left\lceil \frac{\text{delay}(v_{MUL})}{20} \right\rceil * 20 - \text{delay}(v_{MUL}) = \left\lceil \frac{135}{20} \right\rceil * 20 - 135 = 5$$

$$\text{slack}(20\text{ns}, v_{ADD}) = \left\lceil \frac{\text{delay}(v_{ADD})}{20} \right\rceil * 20 - \text{delay}(v_{ADD}) = \left\lceil \frac{45}{20} \right\rceil * 20 - 45 = 15$$

$$\text{slack}(20\text{ns}, v_{SUB}) = \left\lceil \frac{\text{delay}(v_{SUB})}{20} \right\rceil * 20 - \text{delay}(v_{SUB}) = \left\lceil \frac{55}{20} \right\rceil * 20 - 55 = 5$$

d) Berechnen Sie den mittleren Schlupf (average slack) für eine Taktperiode von 20ns.

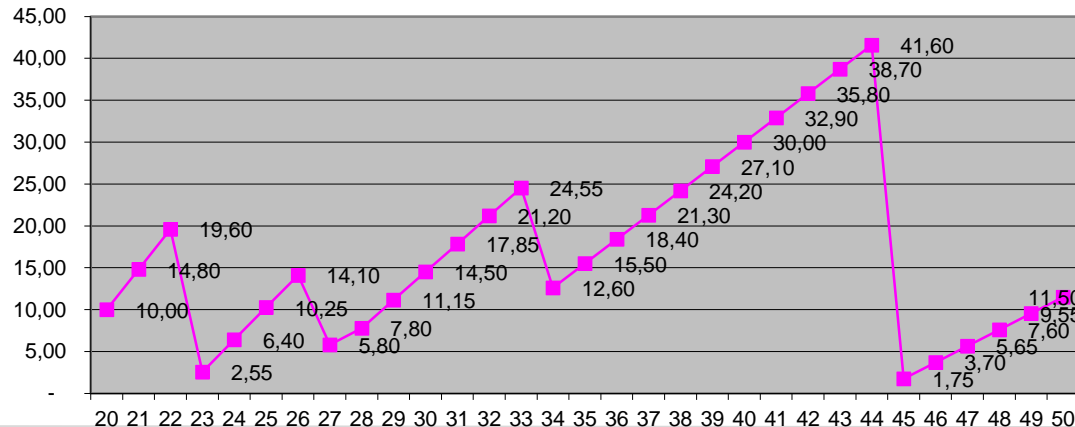
$$\text{avgslack}(T) = \frac{\sum_{k=1}^{|V_T|} (\text{occ}(v_k) * \text{slack}(T, v_k))}{\sum_{k=1}^{|V_T|} \text{occ}(v_k)}$$

$$\begin{aligned} \text{avgslack}(20\text{ns}) &= \frac{\text{occ}(v_{MUL}) * \text{slack}(20\text{ns}, v_{MUL}) + \text{occ}(v_{ADD}) * \text{slack}(20\text{ns}, v_{ADD}) + \text{occ}(v_{SUB}) * \text{slack}(20\text{ns}, v_{SUB})}{\text{occ}(v_{MUL}) + \text{occ}(v_{ADD}) + \text{occ}(v_{SUB})} \\ &= \frac{9 * 5\text{ns} + 10 * 15\text{ns} + 1 * 5\text{ns}}{9 + 10 + 1} = \frac{45\text{ns} + 150\text{ns} + 5\text{ns}}{20} = 10\text{ns} \end{aligned}$$

# Lösung Aufgabe 3.03: Taktschlupf

- Raten/Überlegen Sie, welche Taktperiode den niedrigsten mittleren Schlupf hat.

| Funktionale Einheit | k | delay( $v_k$ ) [ns] | occ( $v_k$ ) |
|---------------------|---|---------------------|--------------|
| MUL                 | 1 | 135                 | 9            |
| ADD                 | 2 | 45                  | 10           |
| SUB                 | 3 | 55                  | 1            |



- Warum erfolgt eine Schätzung der Entwurfsqualität?
- Welche Metriken können geschätzt werden?
- Was ist Hardware-Performanz? Wie setzt sich diese zusammen?
- Wieso wird die Taktperiode geschätzt? Wie kann sie optimiert werden?

