Lösung

Aufgabe 1. Pumping-Lemma, kontextfreie Grammatiken $(5 + 5 = 10$ Punkte)

Für ein Alphabet E, $w \in E^\star$, $a \in E$ bezeichne $|w|_a$ die Anzahl der a's in w.

Ein wohlgeformter Klammerausdruck ist ein Wort der Sprache

$$L = \{w \in \{(,)\}^\star \mid |w|_(= |w|_) \text{ und } w = uv \Rightarrow |u|_(\geq |u|_)\}.$$

Ein Wort $w \in L$ darf also jede Kombination aus öffnenden und schließenden Klammern sein, bei der die Gesamtanzahl von öffnenden Klammern der Gesamtanzahl von schließenden Klammern entspricht und bei der in jedem Präfix von w höchstens so viele schließende wie öffnende Klammern vorkommen.

Beispiele für wohlgeformte Klammerausdrücke sind: $\lambda, (()), ()()(), ()((())()) \in L$.

(a) Zeigen Sie mithilfe des Pumping-Lemmas für reguläre Sprachen, dass L nicht regulär ist.

Lösung: Wir betrachten für ein $n \in \mathbb{N}$ das Wort $w = (^n)^n \in L$ mit $|w| >= n$.

Laut Pumping-Lemma muss es, falls L regulär ist, eine Zerlegung $w = xyz$ geben mit

(1) $|xy| \leq n$,

(2) $|y| \neq 0$ und

(3) $\forall i \in \mathbb{N} : xy^iz \in L$.

Nach (1) und (2) gilt: $y = (^{n-k}$ für ein $k \in \{0, \ldots, n-1\}$.

Wenn wir also $i = 0$ setzen, erhalten wir $xy^iz = (^{n-(n-k)})^n = (^k)^n \notin L$, da $n \neq k$.

Nach (3) folgern wir, dass L nicht regulär ist.

(b) Zeigen Sie durch Angabe einer kontextfreien Grammatik G mit $L(G) = L$, dass L kontextfrei ist. Definieren Sie die Grammatik vollständig.

Lösung:

$G = (N, T, P, S)$ mit

- $N = \{S\}$,
- $T = \{(,)\}$,
- $P = \{S \to (S) \mid SS \mid \lambda\}$.

Aufgabe 2. Reguläre Sprachen (1 + 7 = 8 Punkte)

Wie in Aufgabe 1 sei L die Sprache der wohlgeformten Klammerausdrücke:

$$L = \{w \in \{(,)\}^\star \mid |w|_(= |w|_) \text{ und } w = uv \Rightarrow |u|_(\geq |u|_)\}.$$

Für $n \in \mathbb{N}$ bezeichne L_n die wohlgeformten Klammerausdrücke mit einer Länge von höchstens n:

$$L_n = \{w \in L \mid |w| \leq n\}.$$

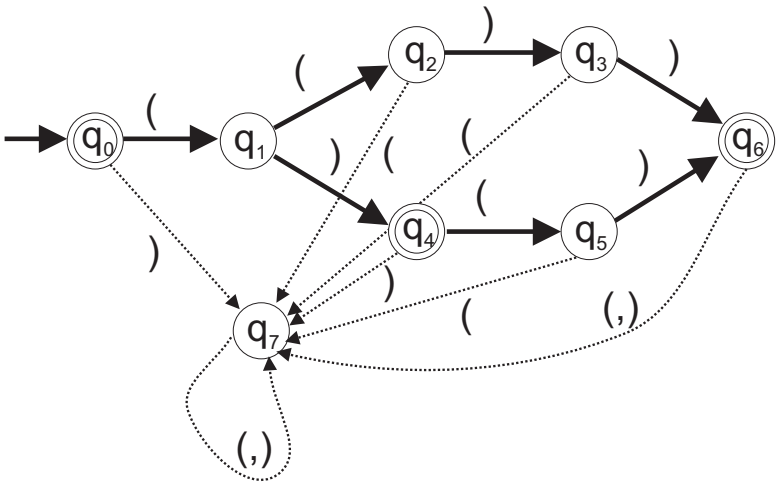
Es gilt bspw. $L_2 = \{(), \lambda\}$.

(a) Geben Sie L_4 an.

Lösung: $L_4 = \{()(), (()), (), \lambda\}$.

(b) Geben Sie einen deterministischen endlichen Automaten A an mit $L(A) = L_4$. Definieren Sie den Automaten vollständig.

Lösung:



Aufgabe 3. Turingmaschinen (10 Punkte)

Gegeben sei für das Alphabet $E = \{0, 1, \$\}$ die Sprache

$$L = \{\$^m u \$^n \in E^\star \mid m, n \in \mathbb{N}_0,\ u \in \{0,1\}^+\}.$$

Die Wörter in L bestehen also aus einem beliebig langen Strom von \$-Zeichen, der an genau einer Stelle durch eine beliebige nichtleere Sequenz von Nullen und Einsen unterbrochen wird.

Bspw. gilt:

- $0, \$1\$, 110\$\$\$, \$\$\$1101\$\$ \in L$;
- $\lambda, \$\$\$, 0\$1, \$\$11\$10, \$\$11\$10\$\$ \notin L$.

Geben Sie eine Turingmaschine M an, die für Eingabewörter $w \in L$ alle Zeichen der Eingabe mit 1, für Eingabewörter $w' \notin L$ mit 0 überschreibt. Anschließend soll die Turingmaschine auf einem dieser Felder in einem Endzustand halten. Definieren Sie die Turingmaschine vollständig.

$M = (S, E, B, \delta, s_0, F)$,
$S = \{s_0, s_1, s_2, s_N, s_{NN}, s_{EE}\}$,
$E = \{0, 1, \$\}$,
$B = \{0, 1, \$, \star\}$,
$F = \{s_{EE}\}$.

δ	0	1	\$	$\star$	
s_0	$(s_1, 1, N/R)$	$(s_1, 1, N/R)$	$(s_0, 1, R)$	$(s_N, \star, N)$	
s_1	$(s_1, 1, R)$	$(s_1, 1, R)$	$(s_2, 1, R)$	$(s_{EE}, \star, L)$	
s_2	$(s_N, 0, N/R)$	$(s_N, 1, N/R)$	$(s_2, 1, R)$	$(s_{EE}, \star, L)$	
s_N	$(s_N, 0, R)$	$(s_N, 1, R)$	$(s_N, \$, R)$	$(s_{NN}, \star, L)$	
s_{NN}	$(s_{NN}, 0, L)$	$(s_{NN}, 0, L)$	$(s_{NN}, 0, L)$	$(s_{EE}, \star, R)$	
s_{EE}					

Aufgabe 4. Komplexitätstheorie (7 + 3 = 10 Punkte)

(a) Geben Sie für die folgenden Komplexitätsklassen jeweils ein aus der Vorlesung bekanntes Problem an, das sicher in der jeweiligen Klasse liegt, oder begründen Sie, warum Sie keines angeben können.

(1) P

(2) NP

(3) $P \cap NP$-vollständig

(4) $\{R \mid \forall Q \in NP\colon Q \leq_{pol} R\}$

(5) $\mathrm{DTime}(n^3) \setminus P$

Lösung:

1. Wortproblem für Typ3-,
   ∼Typ2-Sprachen,
   $\{(G, w) \mid G$ ist kontextfrei und $w \in L(G)\}$,
   Sortieren,
   Automatenminimierung, und andere
2. alle aus 1), alle aus 4)
3. keine Angabe möglich, Klasse leer, falls $P \neq NP$ o. ä.
4. Die Klasse ist NP-schwer. SAT, 3-SAT, CLIQUE, Halteproblem und andere
5. Es gibt keines, denn $\mathrm{DTime}(n^3)$ ist Teilmenge von P.

(b) Für zwei Probleme Q, R gelte: $Q \leq_{pol} R$.

(1) Welche Aussage über die Schwierigkeit der Probleme Q, R wird hier getroffen?

Lösung: R ist bis auf polynomielle Faktoren mindestens so schwer wie Q.

(2) Was bedeutet es für Q, wenn $R \in P$ gilt?

Lösung: Dann ist Q auch in P.

(3) Was bedeutet es für R, wenn $Q \in NP$ gilt?

Lösung: Das hat für R keinerlei Bedeutung, weil Q auch ein sehr leichtes Problem in NP sein könnte (bspw. $E^\star$). Dann wäre R mindestens so schwer wie ein sehr leichtes Problem, was eine triviale Aussage ist.

Aufgabe 5. Binary-Decision-Diagram (2 + 5 = 7 Punkte)

Gegeben sei folgende Boolesche Funktion $f : \mathbb{B}^3 \to \mathbb{B}$ mit

$$f(a, b, c) = (b + a) \cdot (c' + a')$$

(a) Stellen Sie f in der nachstehenden Tabelle dar. Geben Sie dafür zunächst die Komponenten $(b + a)$ und $(c' + a')$ an.

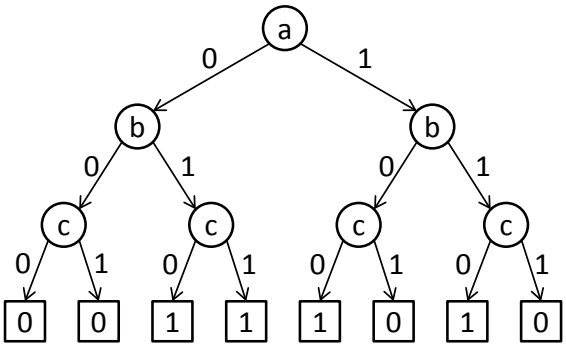
Lösung:

a	b	c	$(b + a)$	$(c' + a')$	$f(a, b, c)$
1	1	1	1	0	0
1	1	0	1	1	1
1	0	1	1	0	0
1	0	0	1	1	1
0	1	1	1	1	1
0	1	0	1	1	1
0	0	1	0	1	0
0	0	0	0	1	0

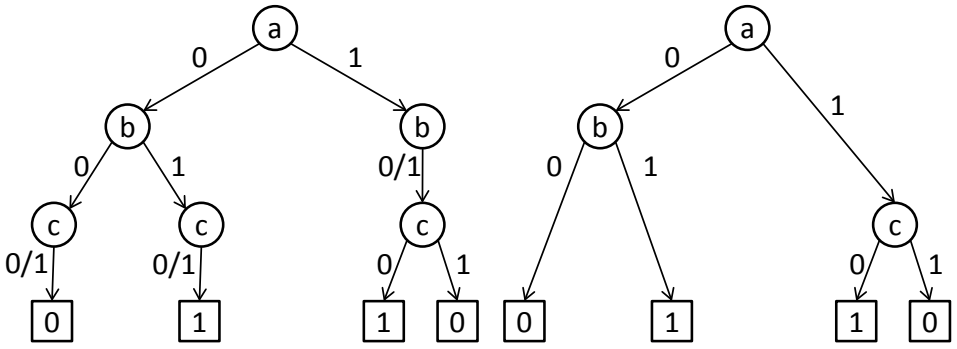
(b) Erzeugen Sie das Binary-Decision-Diagram (BDD) zu f mit der Variablenreihenfolge a - b - c. Benutzen Sie hierzu den Platz auf der kommenden Seite.

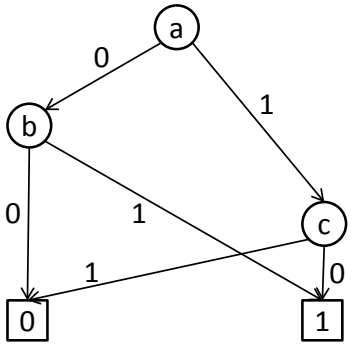
BDD:

Lösung:

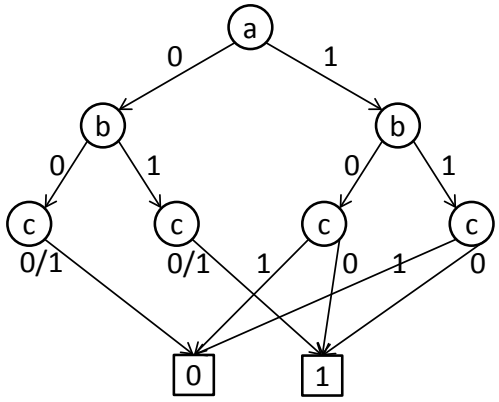


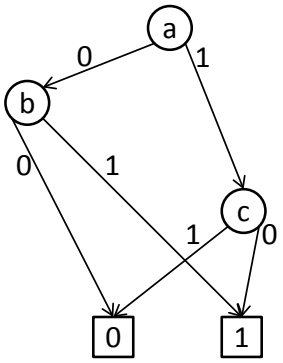
Möglichkeit 1:





Möglichkeit 2:



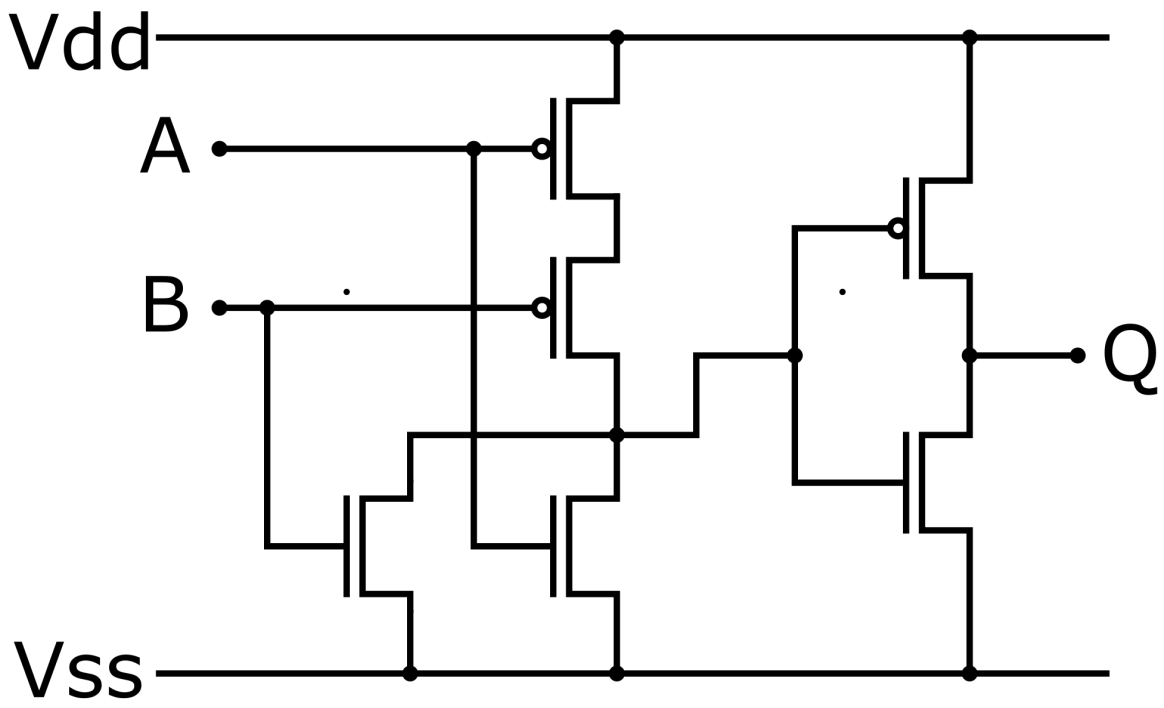

a
b
c
0
1
0
1
1
0
0
1

Aufgabe 6. CMOS (5 Punkte)

Die Boolesche Funktion $or\colon \mathbb{B}^2 \to \mathbb{B}$ sei für $b_1, b_2 \in \mathbb{B}$ definiert als

$$or(b_1, b_2) = b_1 \vee b_2.$$

(a) Zeichnen Sie in das vorgegebene Feld eine CMOS-Schaltung, die *or* implementiert.

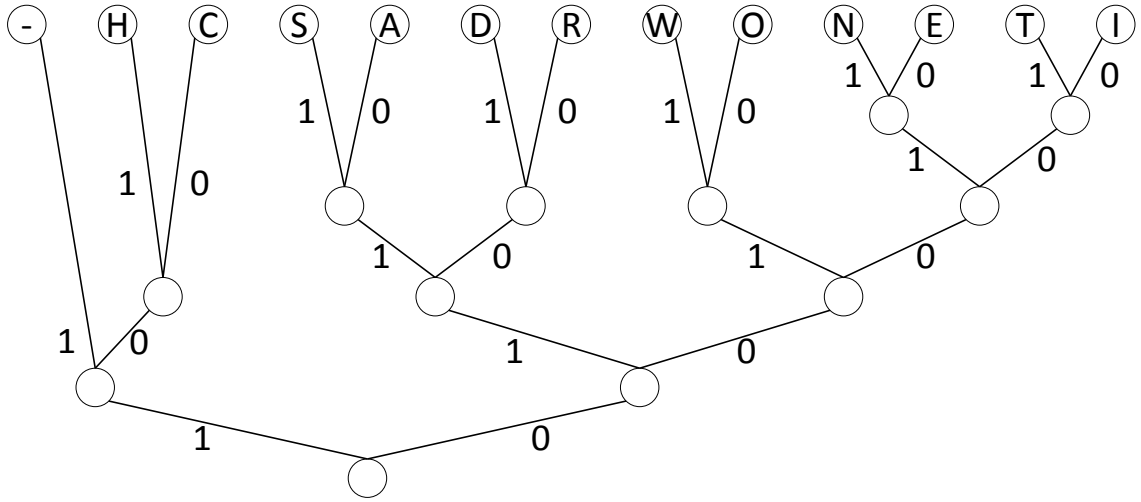


Hinweis: Überlegen Sie zunächst, welche CMOS-Schaltungen Sie bereits kennen und wie man *or* durch diese (eventuell durch Verknüpfung mehrerer) darstellen kann.

Lösung: Bspw. durch Hintereinanderschaltung von NOR und NOT.

Aufgabe 7. Huffman-Kodierung (3 + 3 = 6 Punkte)

Gegeben sei folgender Huffman-Baum.



(a) Dekodieren Sie anhand des obigen Huffman-Baums den folgenden String.

000110000010010100001110111100101001100010100

Lösung:
Die Huffman-Decodierung dieses Strings

$$00011, 00000, 100, 101, 00001, 11, 0111, 100, 101, 0011, 00010, 0100$$

ergibt „NICHT-SCHWER“

(b) Die folgende Tabelle gibt die relative Häufigkeit einzelner Zeichen in einem zu kodierenden Text an. Diese Verteilung wurde auch dem obigen Huffman-Baum zugrunde gelegt.

-	H	C	S	A	D	R	W	O	N	E	T	I
0,16	0,12	0,12	0,08	0,08	0,08	0,08	0,08	0,04	0,04	0,04	0,04	0,04

Bisher seien die einzelnen Zeichen jeweils durch 4 Bits kodiert worden. Wie groß ist die erwartete relative Ersparnis an Speicherplatz bei Verwendung des angegebenen Huffman-Codes?

Lösung:
Codelänge nach der alten Speicherung:

$\lceil ld(13) \rceil = 4$
durchschnittliche Codelänge nach der Huffman-Codierung:
$0,16 * 2 + 0,12 * 3 + 0,12 * 3 + 0,08 * 4 + 0,08 * 4 + 0,08 * 4 + 0,08 * 4 + 0,08 * 4 + 0,04 * 4 + 0,04 * 5 + 0,04 * 5 + 0,04 * 5 + 0,04 * 5 = 0,16 * 2 + 0,12 * 3 * 2 + 0,08 * 4 * 5 + 0,04 * 4 * 1 + 0,04 * 5 * 4 = 3,6$
erwartete, relative Ersparnis:
$1 - 3,6/4 = 1 - 0,9 = 0,10 \Rightarrow$ Ersparnis von 10%

Aufgabe 8. Zahlendarstellung $(2 + 1 + 5 = 8$ Punkte)

Gegeben seien die folgenden beiden Binärsequenzen:

$$a = 10110101, b = 01111001$$

(a) Berechnen Sie $a+b$, wobei a, b als Zahlen in Dualdarstellung zu interpretieren sind. Geben Sie das Ergebnis sowohl in Dualdarstellung, als auch in Dezimaldarstellung an. Bei der Dezimaldarstellung reicht die Angabe der Summe der Zweierpotenzen.

Lösung:
10110101 entspricht $2^0 + 2^2 + 2^4 + 2^5 + 2^7 = 1 + 4 + 16 + 32 + 128 = 181$ (nicht gefragt)
01111001 entspricht $2^0+2^3+2^4+2^5+2^6 = 1+8+16+32+64 = 121$ (nicht gefragt)

Addition:

10110101	
01111001	
111100010	(Übertrag)
100101110	

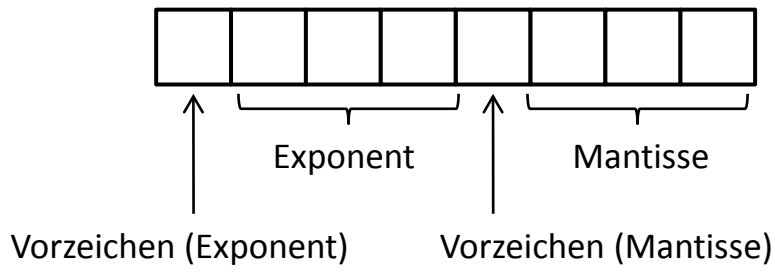
100101110 entspricht $2^1 + 2^2 + 2^3 + 2^5 + 2^8 = 2 + 4 + 8 + 32 + 256 = 302$

(b) Wieder sei die Binärsequenz a als Zahl in Dualdarstellung zu interpretieren. Bilden Sie das 2-Komplement von a.

Lösung:
Durch Kippen aller Bits und Addition von 1 ergibt sich 01001011.

(c) Betrachten Sie die folgende Darstellung einer 8-Bit-Gleitkommazahl. Der Exponent und die normierte Mantisse ohne führende Eins sind in Betrag-Vorzeichen-Darstellung gespeichert. Die Basis für den Exponenten ist 2.

Hinweis: Beachten Sie, dass diese Darstellung einer Gleitkommazahl nicht der IEEE-754-Kodierung entspricht.



Nehmen Sie für a und b diese Darstellung einer 8-Bit-Gleitkommazahl an. Geben Sie den Dezimalwert für a und b an. Auch hier reicht die Angabe der Summe der Zweierpotenzen.

Lösung:
1.011.0.101:
Vorzeichen: $-$
Exponent: $2^0 + 2^1 = 3$
Vorzeichen: $+$
Mantisse: $2^0 + 2^{-1} + 2^{-3} = 1 + 1/2 + 1/8$
$\rightarrow (1 + 1/2 + 1/8) \cdot (2^{-3}) = 13/64$

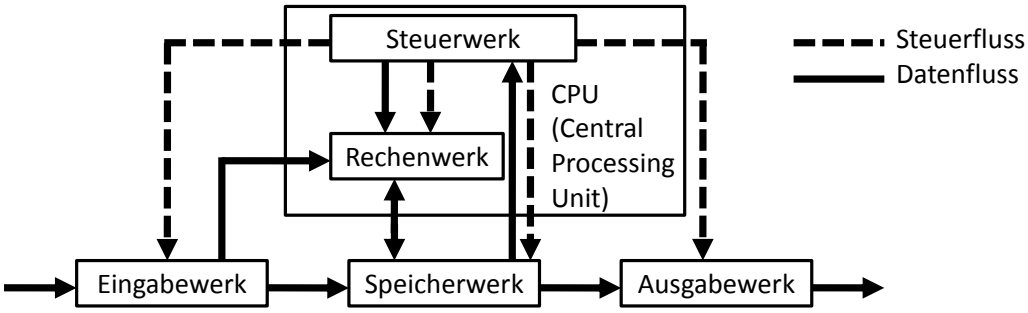
0.111.1.001:
Vorzeichen: $+$
Exponent: $2^0 + 2^1 + 2^2 = 7$
Vorzeichen: $-$
Mantisse: $2^0 + 2^{-3} = 1\frac{1}{8}$
$\rightarrow -(1 + 1/8) \cdot 2^7 = -144$

Aufgabe 9. Von-Neumann-Rechner, Pipelining $(4 + 2 + 2 + 2 + 1 = 11$ Punkte)

In der Vorlesung wurde das Von-Neumann-Rechnermodell vorgestellt.

(a) Benennen Sie die einzelnen Komponenten des Von-Neumann-Rechners in der nachstehenden Skizze. Kennzeichnen Sie zudem, bei welchen der angegebenen Flüsse (Pfeile) es sich um Datenfluss (D) und bei welchen um Steuerfluss (S) handelt, indem Sie an den Pfeilen die jeweils falsche Bezeichnung streichen.

Lösung:



(b) Wo liegt eine wesentliche Schwachstelle der Von-Neumann-Architektur? Beschreiben Sie diese kurz.

Lösung:
physikalischer Von-Neumann-Engpass: Der Transport einer Vielzahl von Daten ziwschen der CPU und dem Arbeitsspeicher dauert um ein Vielfaches länger als die Ausführung der Befehle der CPU.

(c) Von-Neumann-Rechner gehören zur Klasse des SISD-Architektur. Wofür steht die Abkürzung SISD und was bedeutet diese?

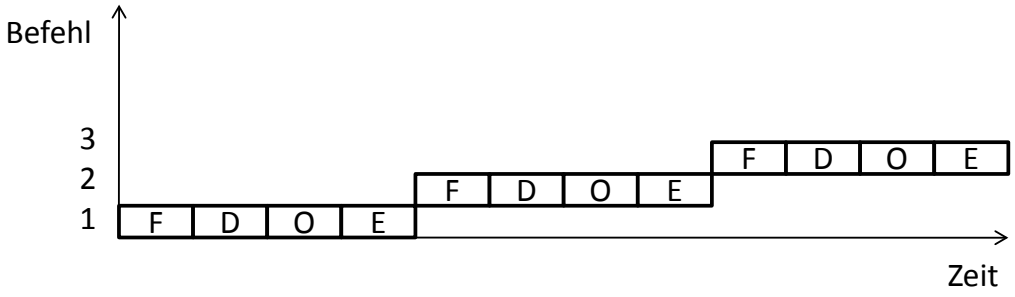
Lösung:
SISD steht für „Single-Instuction Single-Data“. Dies bedeutet, dass zu jedem Zeitpunkt nur ein einziger Befehl ausgeführt wird, d.h. die Aufgaben sequentiell ausgeführt werden.

(d) Durch die Verwendung des Pipelining können verschiedene Befehle parallel ausgeführt werden.

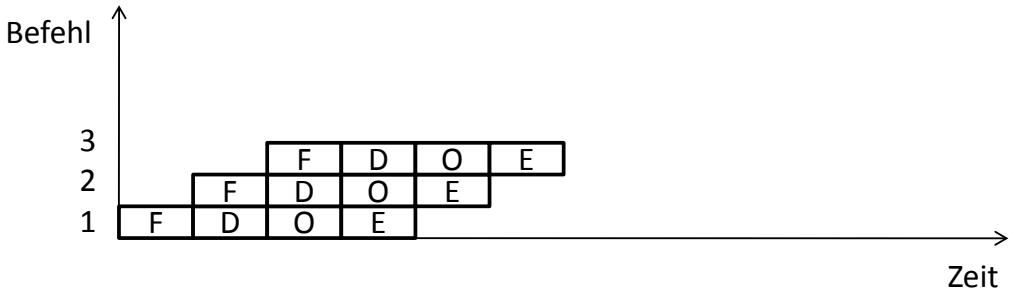
Veranschaulichen Sie in den nachstehenden Diagrammen die Abarbeitung dreier Befehle mit und ohne Pipelining graphisch. Nehmen Sie dabei an, dass jeder der Befehle $1, 2, 3$ in die vier Teilaufgaben $\boxed{F_i}$, $\boxed{D_i}$, $\boxed{O_i}$, $\boxed{E_i}$, $i \in \{1, 2, 3\}$ zerlegt werden kann.

Lösung:

Abarbeitung ohne Pipelining



Abarbeitung mit Pipelining



(e) Welches Problem kann bei einer parallelen Abarbeitung von Befehlen wie dem Pipelining auftreten?

Lösung:
Deadlock-Situationen

Aufgabe 10. Assembler (4 + 2 + 3 = 9 Punkte)

Die Befehle der in der Vorlesung vorgestellten Assembler-Sprache sind folgendermaßen aufgebaut, wobei 'Q' für Quelle steht und 'Z' für Ziel:

```
OpCode Q1, (Q2,) Z
```

Für unmittelbare Adressierung wird das Präfix '#' verwendet. Ein bedingter Sprungbefehl ist JNZ (JumpNotZero), der den Befehlszähler genau dann zu Label L springen lässt, falls $Q \neq 0$. Die Notation des Sprungbefehls ist:

```
JNZ Q L
```

Gegeben sei das folgende Assemblerprogramm.

```
1.     STORE #1, R2
2. L1: MULTIPLY R2, R1, R2

3.     SUBTRACT R1, #1, R1
4.     JNZ R1, L1
5.     STORE R2, R1
```

(a) Zu Beginn stehe in Register $R1$ ein Wert $n > 0, n \in \mathbb{N}$. Wie hängt der Endwert $R1^E$ des Registers $R1$ mit seinem Anfangswert $R1^A = n$ zusammen, welche Funktion berechnet das Programm also bezogen auf $R1$?

(b) Was geschieht, wenn zu Beginn der Wert $R1^A = 0$ in $R1$ steht?

(c) Welche Funktion berechnet das Programm bezogen auf $R1$, wenn man zwischen Zeile 2 und Zeile 3 zusätzlich noch den Befehl

```
ADD R2, R2, R2
```

einfügt?

Lösung: (a) Es wird für $n > 0$ die Fakultätsfunktion $R1! = n!$ berechnet;
(b) für $n = 0$ entsteht eine Endlosschleife bzw. irgendwann ein Überlauf der Variablen R1
(c) Es wird dann die Funktion $n! \cdot 2^n$ berechnet.

Aufgabe 11. Betriebssysteme und Dateiorganisation (6 Punkte)

Kreuzen Sie an, ob die folgenden Aussagen wahr oder falsch sind. Für jede richtige Antwort erhalten Sie einen Punkt, für falsche Kreuze wird Ihnen jeweils ein Punkt abgezogen. Sie können nicht weniger als 0 Punkte in dieser Aufgabe erzielen.

Lösung:

	wahr	falsch
Das Round-Robin-Zeitscheibenverfahren berücksichtigt von Prozessen einzuhaltende Fristen.	○	⊗
Threads innerhalb desselben Prozesses nutzen einen gemeinsamen Speicherbereich und können daher effizient kommunizieren.	⊗	○
Ein Prozess kann vom Zustand „bereit“ direkt in den Zustand „blockiert“ wechseln.	○	⊗
Das Zuteilungsverfahren First-Come-First-Serve (FCFS) ist ein non-preemptive Verfahren.	⊗	○
Bei der Dateiorganisation ist die Hash-Organisation eine Art der Primärorganisation.	⊗	○
Eine Hashfunktion ist immer bijektiv, damit aus dem Primärschlüssel eine eindeutige Adresse berechnet werden kann.	○	⊗