KIT-Fakultät für Informatik
Prof. Dr. Uwe Hanebeck, Prof. Dr.-Ing. Jörg Henkel

Musterlösungen zur Klausur

Digitaltechnik und Entwurfsverfahren (TI-1)

und

Rechnerorganisation (TI-2)

am 13. August 2024, 08:00 – 10:00 Uhr

Name:	Vorname:	Matrikelnummer:
Bond	James	007

Digitaltechnik und Entwurfsverfahren (TI-1)	
Aufgabe 1	12 von 12 Punkten
Aufgabe 2	8 von 8 Punkten
Aufgabe 3	6 von 6 Punkten
Aufgabe 4	10 von 10 Punkten
Aufgabe 5	9 von 9 Punkten

Rechnerorganisation (TI-2)	
Aufgabe 6	7 von 7 Punkten
Aufgabe 7	9 von 9 Punkten
Aufgabe 8	12 von 12 Punkten
Aufgabe 9	9 von 9 Punkten
Aufgabe 10	8 von 8 Punkten

Gesamtpunktzahl:	90 von 90 Punkten

	Note: **1,0**

Aufgabe 1 *Rechnerarithmetik und Codes* (12 Punkte)

1. Einschrittige Kodierung: 1 P.
   Aufeinanderfolgende Zahlen sind so durch Binärzeichen dargestellt, dass sich stets nur ein einziges Binärzeichen ändert.

2. 1011_G in Dezimalzahl: 2 P.
   Es gilt:

```
res  = mask = 1011      // res = binaeres Ergebnis

while (mask)            // Solange die Bitmaske ungleich 0 ist
{
mask = mask >> 1        // Rechtsverschiebung um 1 Bit
res = res XOR mask      // Exklusives Oder mit der Bitmaske
}
```

   Ergebnis: $1011_G = 1101_2 = 13_{10}$

3. 7_{10} in Gray-Kodierung: 2 P.
   Durch Hochzählen: $0000 \rightarrow 0001 \rightarrow 0011 \rightarrow 0010 \rightarrow 0110 \rightarrow 0111 \rightarrow 0101 \rightarrow \underline{\underline{0100}}$

4. Vorteil der Gray-Kodierung: 1 P.
   Bei mechanischen Abtastern: Vorteile bei der A/D-Wandlung, da nur ein Bit geändert wird.

5. Nachteil der Gray-Kodierung: 1 P.
   Keine feste Stellenwertigkeit: Die Ausführung von arithmetischen Operationen ist aufwändiger.

6. Länge der Mantisse: 1 P.

   - 32-Bit: 23
   - 64-Bit: 52

7. 0100 0000 0100 0000 0000 0000 0000 0000_{IEEE} als Dezimalzahl: 2 P.

   - Vorzeichen: 0
   - Exponent: $10000000_2 \rightarrow 128 - 127 = 1$
   - Mantisse: $1,1...0_2$
   - Ergebnis: $(-1)^0 \cdot 2^1 \cdot 1.1...0_2 = 2^1 \cdot 1.5 = 3$

8. Hamming-Code für 1011_2: 2 P.

   Datenwort: 1011_2

   Aufspreizen des Datenwortes:

m_4	m_3	m_2	k_3	m_1	k_2	k_1
1	0	1	-	1	-	-

   Bestimmung der Prüfbits:

$$\begin{aligned} k_1 &= m_1 \oplus m_2 \oplus m_4 = 1 \oplus 1 \oplus 1 = 1 \\ k_2 &= m_1 \oplus m_3 \oplus m_4 = 1 \oplus 0 \oplus 1 = 0 \\ k_3 &= m_2 \oplus m_3 \oplus m_4 = 1 \oplus 0 \oplus 1 = 0 \end{aligned}$$

   Codewort: 1 0 1 0 1 0 1

Aufgabe 2 *Schaltfunktion* (8 Punkte)

1. KV-Diagramm: 5 P.

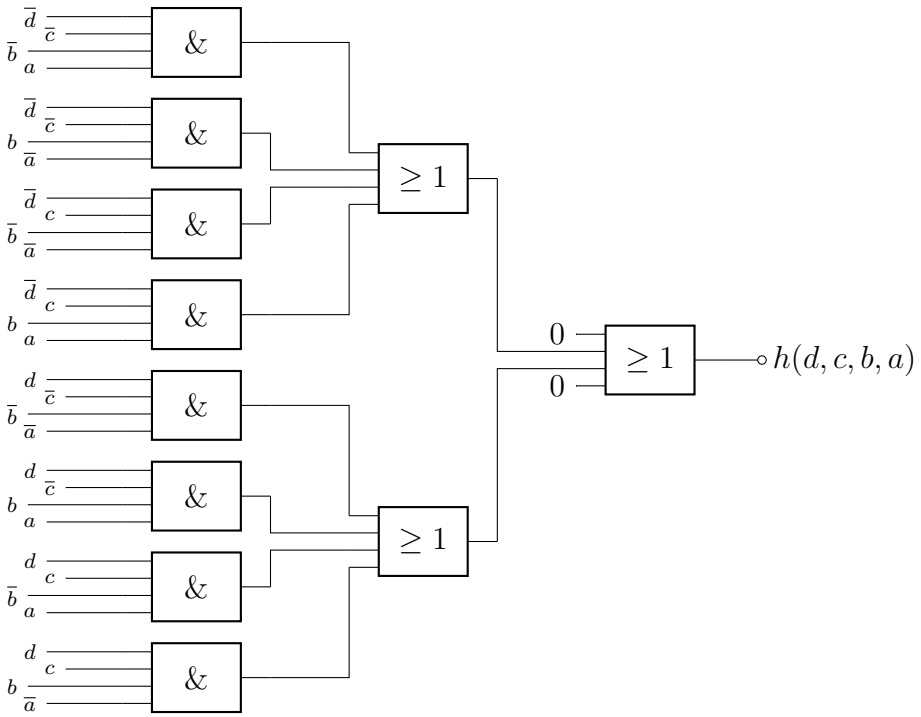
$h(d,c,b,a)$:

d \ c				
	a			
	0	1	0	1
	0	1	5	4
b	1	0	1	0
	2	3	7	6
	0	1	0	1
	10	11	15	14
	1	0	1	0
	8	9	13	12

Die Primimplikanten sind:

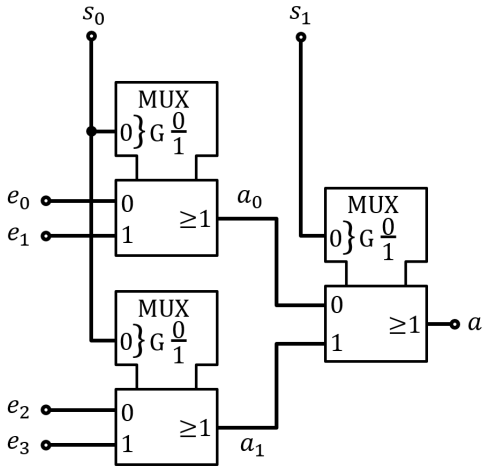
$\overline{d} \wedge \overline{c} \wedge \overline{b} \wedge a$	$\overline{d} \wedge \overline{c} \wedge b \wedge \overline{a}$	$\overline{d} \wedge c \wedge \overline{b} \wedge \overline{a}$	$\overline{d} \wedge c \wedge b \wedge a$
$d \wedge \overline{c} \wedge \overline{b} \wedge \overline{a}$	$d \wedge \overline{c} \wedge b \wedge a$	$d \wedge c \wedge \overline{b} \wedge a$	$d \wedge c \wedge b \wedge \overline{a}$

2. Schaltnetz: 3 P.



Aufgabe 3 *Bausteine und CMOS-Technologie* (6 Punkte)

1. Realisierung des 4:1-Multiplexers: 4 P.



Begründung:

$$\begin{aligned} a &= \overline{s}_1\,\overline{s}_0\,e_0 \;\vee\; \overline{s}_1\,s_0\,e_1 \;\vee\; s_1\,\overline{s}_0\,e_2 \;\vee\; s_1\,s_0\,e_3 \\ &= \overline{s}_1\,(\overline{s}_0\,e_0 \;\vee\; s_0\,e_1) \;\vee\; s_1\,(\overline{s}_0\,e_2 \;\vee\; s_0\,e_3) \\ &= \overline{s}_1\,a_0 \;\vee\; s_1\,a_1 \end{aligned}$$

2 P.

2.

- Schaltfunktion z:
  Entweder Funktionstabelle aufstellen und Schaltfunktion ermitteln oder scharf hinschauen und die Schaltfunktion ablesen.

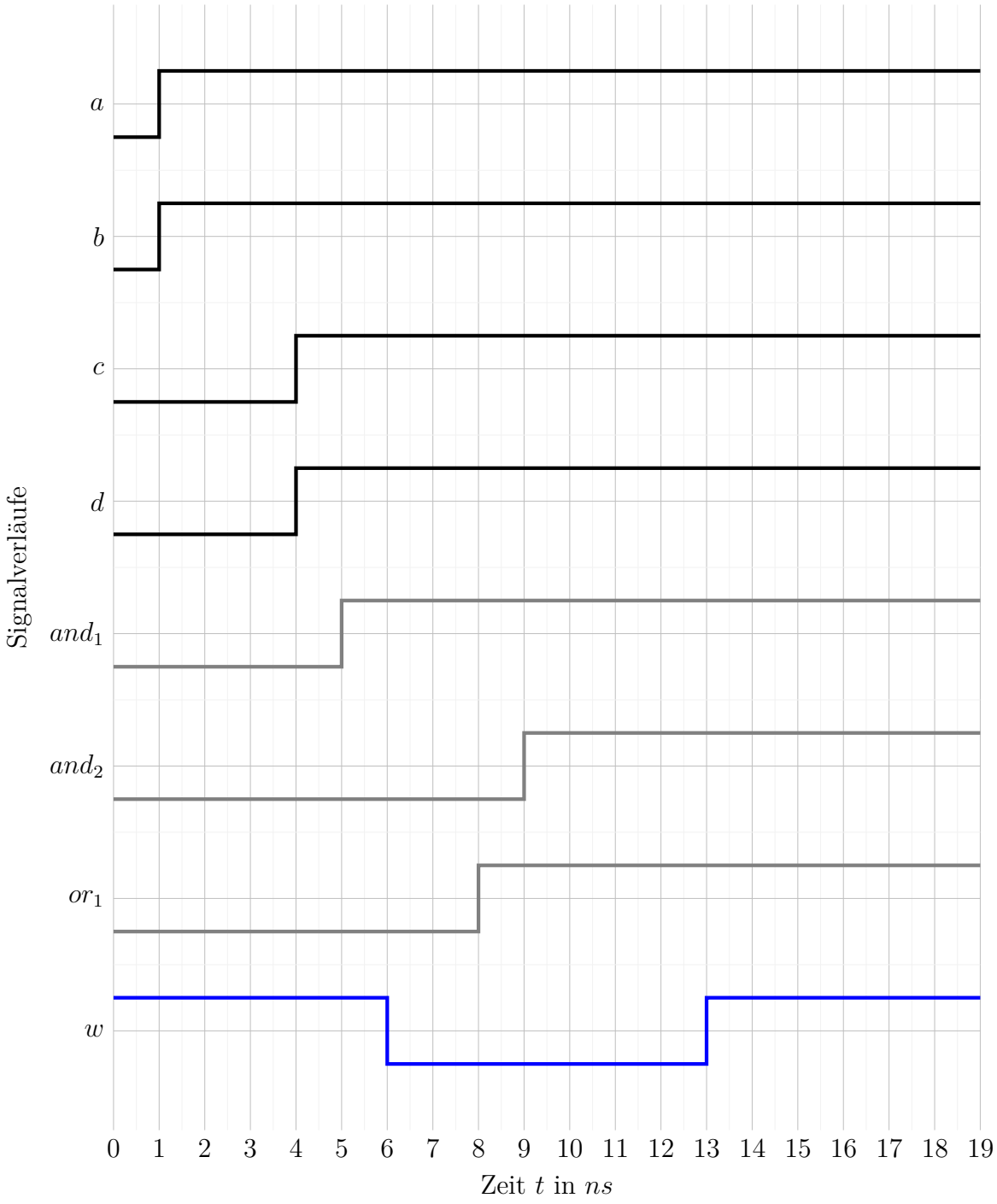
  Hier: scharfes Hinschauen. Die Schaltung besteht aus einem Inverter und zwei Transsmission-Gates, die von c gesteuert werden. Ist $c = 0$ ($c = 1$), dann liegt der Wert von a (b) am Ausgang z.

  $$\text{Schaltfunktion:} \quad z(c,b,a) \;=\; \overline{c}\,a \;\vee c\,b$$

- Bauteil:
  2:1-Multiplexer mit c als Steuervariable, a und b als Variablen an den Eingängen 0 und 1 des Multiplexers.

Aufgabe 4 *Laufzeiteffekte* (10 Punkte)

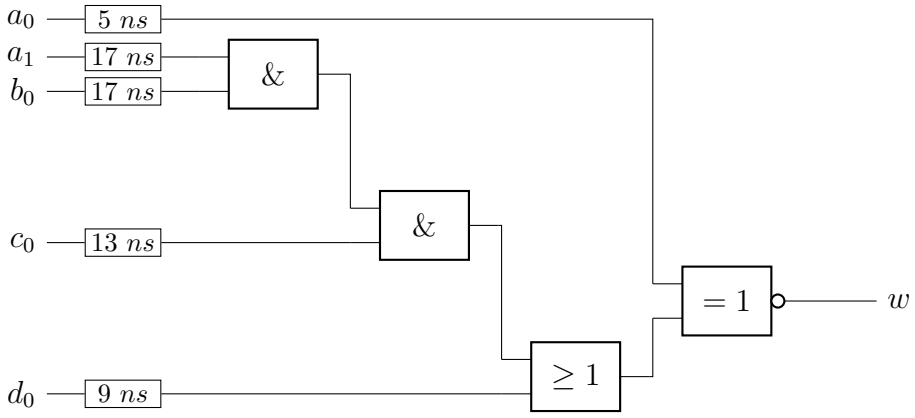
1. Laufzeitdagramm: 5 P.



2. Hazardfehler: 1 P.
   Der Signalverlauf von w weist keinen Hazardfehler auf, da es sowohl beim Wechsel von a und b oder c und d zu keinem ungewollten Wechsel des Ausgangssignal w kommt.

3. 4 P.

- Totzeitmodell:



- Strukturausdruck:

$$w = a_0 \leftrightarrow ((a_1\, b_0)\, c_0 \vee d_0)$$

Aufgabe 5 *Schaltwerke* (9 Punkte)

1. Anzahl der Zustände mit Begründung: 1 P.
   Mit drei T-Flipflops können maximal $2^3 = 8$ Zustände realisiert werden.

2. 3 P.

   - Ansteuerfunktionen der T-Flipflops:
     Ablesen aus Abbildung folgt:

$$\begin{aligned} t_0 &= e_0 \nleftrightarrow q_0 \\ t_1 &= e_1 \nleftrightarrow q_1 \\ t_2 &= e_2 \nleftrightarrow q_2 \end{aligned}$$

   - Zustandsübergangsgleichungen:
     Aus der charakteristischen Gleichung der T-Flipflops folgt:

$$\begin{aligned} q_i^{t+1} &= (q_i\ \overline{t_i}\ \vee\ \overline{q_i}\ t_i)^t \\ q_0^{t+1} &= (q_0\ (e_0 \leftrightarrow q_0)\ \vee\ \overline{q_0}\ (e_0 \nleftrightarrow q_0))^t = (q_0 \leftrightarrow (e_0 \leftrightarrow q_0))^t = e_0^t \\ q_1^{t+1} &= (q_1 \leftrightarrow (e_1 \leftrightarrow q_1))^t = e_1^t \\ q_2^{t+1} &= (q_2 \leftrightarrow (e_2 \leftrightarrow q_2))^t = e_2^t \end{aligned}$$

3. Schaltwerk mit D-Flipflops: 3 P.
   Aus der charakteristischen Gleichung der D-Flipflops folgt:

$$\begin{aligned} q_i^{t+1} &= d_i^t \\ d_0^t &= q_0^{t+1} = e_0^t \\ d_1^t &= q_1^{t+1} = e_1^t \\ d_2^t &= q_2^{t+1} = e_2^t \end{aligned}$$

Es folgt:

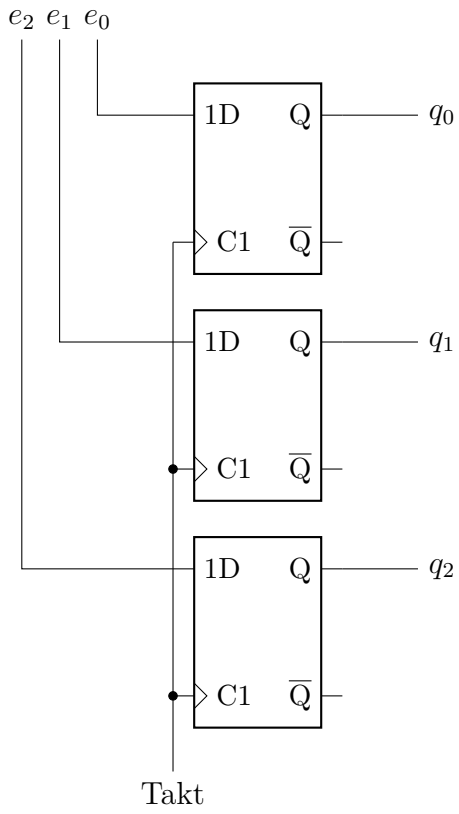


Abbildung 1: Schaltwerk mit D-Flipflops

4. T-Flipflop aus einem JK-Flipflop: 2 P.

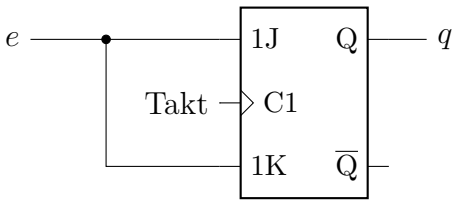
e^t	q_0^t	q_0^{t+1}
0	0	0
0	1	1
1	0	1
1	1	0

$$q^{t+1} = e^t \leftrightarrow q_0^t = (e\ \overline{q_0} \vee\ \overline{e}\ q_0)^t$$

$$q^{t+1} = \left(j_0\ \overline{q_0} \vee \overline{k_0}\ q_0\right)^t$$

$$\Rightarrow j_0 = e$$

$$\Rightarrow k_0 = e$$



Aufgabe 6 *RISC-V Assembler* (7 Punkte)

1. Pseudoinstruktionen: 3 P.

   (a) `mv t1, t2`:

```
1           add t1, t2, zero
```

   (b) `neg t1, t2`:

```
1           sub t1, zero, t2
```

   (c) `nop`:

```
1           addi zero, zero, 0
```

2. Assemblerdirektiven: 2 P.

   (a) `.data`:
   Nachfolgende Elemente werden im Datensegment beginnend mit der nächsten Adresse abgelegt.

   (b) `.asciz`:
   Speichert einen ASCII-String im Datensegment. Der String wird mit einem Null-Byte abgeschlossen.

3. Assemblersequenz: 2 P.

```
1           add a0, t0, zero
2           li a7, 1
3           ecall
```

Aufgabe 7 *Pipelining* (9 Punkte)

1. Kategorien der Datenabhängigkeiten: 1 P.
   Echte Datenabängigkeiten, Gegen-Abhängigkeiten, Ausgabe-Abhängigkeiten

2. Datenabhängigkeiten: 4 P.

   - Echte Abhängigkeiten:

   $$S_1 \to S_2\ (\texttt{t1}) \qquad S_1 \to S_3\ (\texttt{t1})$$
   $$S_2 \to S_3\ (\texttt{t2}) \qquad S_2 \to S_5\ (\texttt{t2})$$
   $$S_3 \to S_4\ (\texttt{t3}) \qquad S_3 \to S_5\ (\texttt{t3})$$

   - Gegen-Abhängigkeiten:

   $$S_2 \to S_4\ (\texttt{t1}) \qquad S_3 \to S_4\ (\texttt{t1})$$

   - Ausgabe- Abhängigkeiten:

   $$S_1 \to S_4\ (\texttt{t1})$$

3. Beseitigung der Datenkonflikte: 4 P.

```
1              S1:     anfang:  andi t2, t1, 1
2                                NOP
3                                NOP
4              S2:              beq t2, zero, weiter
5                                NOP
6                                NOP
7                                NOP
8              S3:              addi t1, t1, -1
9              S4:              j anfang
10                               NOP
11                               NOP
12                               NOP
13             S5:     weiter:  srli t1, t1, 1
14             S6:              j anfang
15                               NOP
16                               NOP
17                               NOP
18             S7:              addi t3, t0,1
19
```

Aufgabe 8 *Cache-Speicher* (12 Punkte)

1. (a) Blockgröße in Bytes: 5 Bit Byte-Offset ⇒ Blockgröße = $2^5 = 32$ Byte — 1 P.

   (b) Anzahl der Einträge: — 1 P.

$$\text{Anzahl der Einträge} = \frac{\text{Kapazität}}{\text{Blockgröße}} = \frac{128\ KiByte}{32\ Byte} = 4\ Ki \text{ Einträge}$$

   (c) Cache-Organisation: — 2 P.
   11 Bit Index-Feld ⇒ Es lassen sich $2^{11} = 2$ Ki Sätze im Cache adressieren.

$$\text{Assoziativität} = \frac{4\ Ki}{2\ Ki} = 2$$

   Der Cache ist als 2-fach assoziativer Speicher (*2-way set associative*) organisiert.

2. Speicherbedarf: — 3 P.
   Für jede Zeile sind (Tag + 1 Statusbit + Daten pro Zeile) Bits erforderlich.

   - Daten pro Zeile 8 Byte × 8 = 64 Bit
   - Tag = 32 - 2 - 3 = 27 Bit (2 Bit Satzindex und 3 Bit Byte-Offset)

   Speicherbedarf für eine Zeile: $27 + 1 + 64$ Bit = 92 Bits

   Speicherbedarf für den gesamten Cache: 92 Bits $\cdot\ 4 \cdot 5\ = 1840$ Bits = 230 Byte

3. — 5 P.

Adresse	**0**	**16**	**48**	**48**	**56**	**8**	**8**	**56**	**32**	**0**
read/write	r	r	w	r	r	r	r	w	w	r
Index	0	2	2	2	3	1	1	3	0	0
Tag	0	0	1	1	1	0	0	1	1	0
Hit/Miss	Miss	Miss	Miss	Miss	Miss	Miss	Hit	Hit	Miss	Hit

Aufgabe 9 *Betriebssystemunterstützung* (9 Punkte)

1. Zwei verwaltende Aufgaben: 1 P.
   Betriebsmittelverwaltung oder Prozessverwaltung oder Speicherverwaltung

2. Zwei Betriebsmodi: 1 P.
   Supervisor-Modus (Kernel-Modus) und Benutzermodus

3. Zwei Kategorien: 1 P.
   Man unterscheidet zwischen *prozessorinternen* und *prozessorexternen* Programmunterbrechungen.

4. Unterteilung der virtuellen Adresse: 1 P.

5. Physikalische Adressen: 4 P.

Virtuelle		Physikalische	
Adresse	Seitennummer	Seitennummer	Adresse
1024	0	3	$3 \cdot 2048 + 1024 = \mathbf{7168}$
5666	2	–	*page fault*
12458	6	4	$4 \cdot 2048 + 170 = \mathbf{8362}$
5678	2	–	*page fault*
9004	4	6	$6 \cdot 2048 + 812 = \mathbf{13100}$
8192	4	6	$6 \cdot 2048 + 0 = \mathbf{12288}$
8400	4	6	$6 \cdot 2048 + 208 = \mathbf{12496}$
0	0	3	$3 \cdot 2048 + 0 = \mathbf{6144}$

6. Zugriff: 1 P.
   Es wird ein *page fault* ausgelöst, was eine Unterbrechungsbehandlung folgen lässt. Diese bewirkt, dass die fehlende Seite in den Hauptspeicher geladen wird.

Aufgabe 10 *Allgemeines* (8 Punkte)

1. Komponenten eines von-Neumann-Rechners: 2 P.
   Steuerwerk, Rechenwerk, Speicher, Ein-/Ausgabeeinheiten, Verbindungseinrichtungen (BUS)

2. 2 P.
   (a) Befehlsregister
   (b) Statusregister
   (c) Programmzähler
   (d) Rechenwerk (ALU)

3. 2 P.
   (a) Einheitliche Befehlslänge und einheitliches Befehlsformat
   (b) Der Zugriff aud den Speicher erfolgt nur über *Load-Store-Befehle*
   (c) festverdrahtetes Steuerwerk
   (d) Getrennte Speicher (und Busse) für Befehle und Daten

4. 2 P.
   USB:
   USB steht für *Universal Serial Bus*. Es handelt sich um eine Punkt-zu-Punkt-Verbindung.