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

Aufgabenblätter zur Prüfung

Digitaltechnik und Entwurfsverfahren (TI-1)

und

Rechnerorganisation (TI-2)

am 17. August 2020, 9:00 – 11:00 Uhr

- Beschriften Sie bitte gleich zu Beginn jedes Lösungsblatt deutlich lesbar mit Ihrem Namen und Ihrer Matrikelnummer.
- Diese Aufgabenblätter werden nicht abgegeben. Tragen Sie Ihre Lösung deshalb ausschließlich in die für jede Aufgabe vorgesehenen Bereiche der Lösungsblätter ein. Lösungen auf separat abgegebenen Blättern werden nicht gewertet.
- Außer Schreibmaterial sind während der Klausur keine Hilfsmittel zugelassen. Täuschungsversuche durch Verwendung unzulässiger Hilfsmittel führen unmittelbar zum Ausschluss von der Klausur und zur Note „nicht bestanden“.
- Soweit in der Aufgabenstellung nichts anderes angegeben ist, tragen Sie in die Lösungsblätter bitte nur Endergebnisse und Rechenweg ein. Die Rückseiten der Aufgabenblätter können Sie als Konzeptpapier verwenden. Weiteres Konzeptpapier können Sie auf Anfrage während der Klausur erhalten.
- Halten Sie Begründungen oder Erklärungen so kurz und präzise wie möglich. Der auf den Lösungsblättern für eine Aufgabe vorgesehene Platz lässt nicht auf den Umfang einer korrekten Lösung schließen.
- Die Gesamtpunktzahl beträgt 90 Punkte. Zum Bestehen der Klausur sind mindestens 40 Punkte zu erreichen.

Viel Erfolg und viel Glück!

Aufgabe 1 *Schaltfunktionen* (7 Punkte)

Gegeben sei die Schaltfunktion f:

$$f(c, b, a) = \text{MAXt}(0,2,3,4)$$

1. Geben Sie die disjunktive Normalform (DNF) der Schaltfunktion f an. 1 P.

2. Tragen Sie die Schaltfunktion f in das im Lösungsblatt vorbereitete KV-Diagramm ein. Geben Sie alle Primimplikante von f an und zeichnen Sie die zugehörigen Blöcke im KV-Diagramm ein. 3 P.
   Geben Sie für jedes Primimplikant an, ob es sich um ein Kernprimimplikant, ein Wahlprimimplikant oder ein entbehrliches Primimplikant handelt.
   Geben Sie eine disjunktive Minimalform (DMF) der Schaltfunktion f an.

3. Die Schaltfunktion f soll mit Hilfe eines 1:8-Demultiplexers und *möglichst wenigen* weiteren Gattern realisiert werden. Geben Sie das zugehörige Schaltnetz an. 1 P.

4. Gesucht ist eine Schaltfunktion $g(d, c, b, a)$. Sie soll folgende Eigenschaften haben:

   - g ist nicht die Nullfunktion
   - g besitzt keine Kernprimimplikante 2 P.

   Kann eine solche Schaltfunktion g existieren? Falls ja, zeichnen Sie eine Variante von g in das im Lösungsblatt vorbereitete KV-Diagramm ein. Falls nein, begründen Sie Ihre Antwort.

Aufgabe 2 *Schaltfunktionen, CMOS* (10 Punkte)

1. Gegeben sei die Schaltfunktion $g(c, b, a)$: 5 P.

$$g(c, b, a) = \Big((\,\overline{c} \vee \overline{b}\,) \wedge (\,\overline{b} \vee \overline{a}\,) \Big) \;\vee\; (\,c \wedge \overline{a}\,)$$

   Realisieren Sie die Schaltfunktion $g(c, b, a)$ durch ausschließliche Verwendung von NAND-Gattern mit zwei Eingängen. Die Eingangsvariablen c, b und a liegen *nur* nicht-negiert vor. Wandeln Sie die Schaltfunktion zuerst entsprechend um. Zeichnen Sie das resultierende Schaltbild.

2. Gegeben sei die folgende Schaltfunktion $h(c, b, a)$: 3 P.

$$h(c, b, a) = \mathrm{NAND}_3\Big(\mathrm{NAND}_2(a, b)\,,\; \mathrm{NAND}_2(a, c)\,,\; \mathrm{NAND}_2(b, c) \Big)$$

   Zeichen Sie die CMOS-Transistorschaltung zu $h(c, b, a)$.

3. Warum ist die in Abbildung 1 dargestellte CMOS-Transistorschaltung für die Realisierung eines Gatters nicht geeignet? Die Negation der Eingangsvariablen a und b können als gegeben angenommen werden. 2 P.

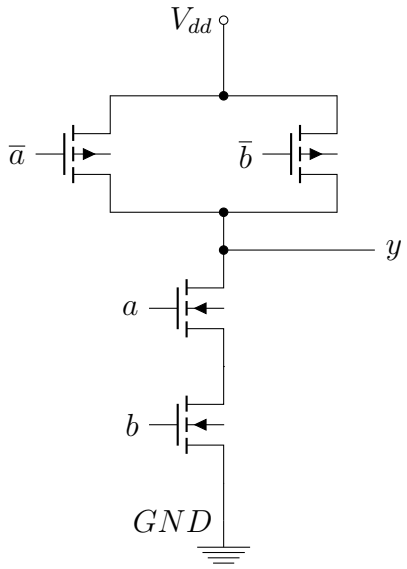


Abbildung 1: CMOS-Transistorschaltung

Aufgabe 3 *Laufzeiteffekte* (6 Punkte)

Gegeben ist das in Abbildung 2 dargestellte Schaltnetz. Die beiden Gatter haben jeweils eine Laufzeit von 1 ns. Der Verlauf des Eingangssignals E ist in Abbildung 3 dargestellt.

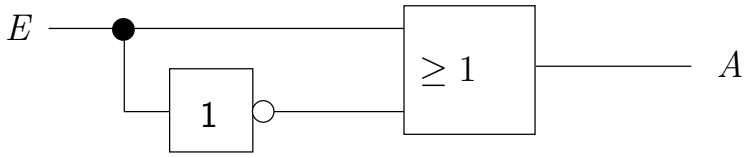


Abbildung 2: Schaltnetz

1. Vervollständigen Sie **im Lösungsblatt** den Verlauf des Ausgangssignals A im Zeitintervall zwischen 3 ns und 12 ns. 2 P.

2. Der Verlauf des Ausgangssignals A weist einen Hasardfehler auf. Welche Art von Hasardfehler ist dies und wodurch ist dieser Fehler bedingt? Begründen Sie Ihre Antwort. 2 P.

3. Zeichnen Sie ein Schaltnetz, das am Ausgang das Signal B erzeugt, wenn das Signal E am Eingang anliegt. Als Gatter stehen Ihnen zwei Inverter und ein NAND–Gatter mit zwei Eingängen zur Verfügung. Die Gatter haben jeweils eine Laufzeit von 1 ns. 2 P.

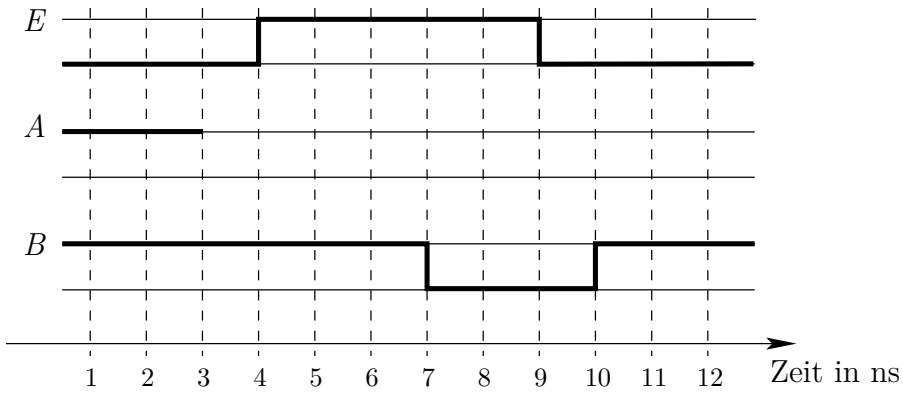


Abbildung 3: Verlauf der Signale E, A und B

Aufgabe 4 *Schaltwerke* (11 Punkte)

1. Was ist der Unterschied zwischen einem Mealy- und einem Moore-Automaten? 1 P.

Es wurde ein Viren-Scanner als synchrones Schaltwerk entworfen, welcher einen binären Eingabestrom (Variable x) auf das Bitmuster *(Virus Signature)* 10X1 überprüft. Dabei steht X für eine 0 oder eine 1.
Beim Erkennen eines derartigen Musters wird eine 1 (Variable y) im nächsten Taktzyklus ausgegeben. Deshalb wurde das Schaltwerk als Moore-Automat realisiert. Ein Beispiel einer Eingabe-Ausgabe-Folge sieht folgendermaßen aus:

t:	1	2	3	4	5	6	7	8	9	10	11	...
x(t):	0	0	1	0	1	1	0	0	1	1	0	...
y(t):	0	0	0	0	0	0	1	0	0	1	0	...

2. Geben Sie den Moore-Automatengraphen des Schaltwerks mit minimaler Anzahl an Zuständen an. Bezeichnen Sie den Anfangszustand mit S und die restlichen Zustände mit $A, B, C, \ldots$ usw. Vergessen Sie nicht, die Kanten und Knoten Ihres Graphen zu beschriften. 6 P.

In Tabelle 1 ist die kodierte Ablauftabelle eines synchronen 3-Bit-Rückwärtszählers angegeben. Die Zustandsvariablen sind mit a, b und c bezeichnet.

Zustand			Folgezustand		
a^t	b^t	c^t	a^{t+1}	b^{t+1}	c^{t+1}
0	0	0	1	1	1
0	0	1	0	0	0
0	1	0	0	0	1
0	1	1	0	1	0
1	0	0	0	1	1
1	0	1	1	0	0
1	1	0	1	0	1
1	1	1	1	1	0

Tabelle 1: Kodierte Ablauftabelle des 3-Bit-Rückwärtszählers.

Die Zustandsvariable a soll in einem D-Flipflop, b in einem JK-Flipflop und c in einem T-Flipflop gespeichert werden. Alle Flipflops sind flankengesteuert und schalten mit positiver Taktflanke.

3. Geben Sie die Ansteuerfunktionen der Flipflops in disjunktiver Minimalform an. 4 P.

Aufgabe 5 *Rechnerarithmetik & Codes* (11 Punkte)

Eine pfiffige TI-Studierende hat eine platzsparende Darstellung von Fließkommazahlen in einem einzigen Byte entwickelt. Das höchstwertige Bit stellt das Vorzeichen V dar, die vier niedrigstwertigen Bits die Mantisse M und die drei Bits in der Mitte den Exponenten E (siehe Abbildung 4)

Abbildung 4: 8-Bit-Fließkommazahl

Für alle möglichen binären Belegungen ergibt sich der Dezimalwert Z aus der nachstehenden Formel (vgl. IEEE-Fließkommazahl).

$$Z \;=\; (-1)^V \cdot 2^{E-3} \cdot 1, M$$

1. Berechnen Sie den Dezimalwert der Belegung 1001 1000. [1 P.]
2. Geben Sie die größte Dezimalzahl an, die mit diesem 8-Bit-Fließkommaformat dargestellt werden kann. [1 P.]
3. Geben Sie die kleinste positive Dezimalzahl an, die mit diesem 8-Bit-Fließkommaformat dargestellt werden kann. [1 P.]
4. Welche elementare Zahl im Intervall [-maxreal, maxreal] kann mit der oben vereinbarten Interpretation der acht Bits nicht dargestellt werden? [1 P.]
5. Wandeln Sie die in einem Zahlensystem zur Basis 4 gegebene Fließkommazahl N_4 sowohl in eine Dezimal- als auch in eine Hexadezimalzahl um. Machen Sie den Rechenweg deutlich. [2 P.]
$$N_4 \;=\; (123,02)_4$$
6. Welche zwei charakteristischen Eigenschaften besitzt die Gray-Kodierung? Warum ist die Ausführung arithmetischer Operationen im Gray-Code schwierig? [1 P.]
7. Welche Vor- und Nachteile hat die BCD-Arithmetik gegenüber der Dual-Arithmetik? [1 P.]
8. Die Elemente der folgenden Sequenz repräsentieren die gleiche ganzzahlige Zahl in Zahlensystemen verschiedener Basen. [3 P.]

   $$(10000)_r,\ (121)_{r+1},\ (100)_{r+2},\ (x)_{r+3},\ (24)_{r+4},\ (22)_{r+5},\ (20)_{r+6}, \ldots$$

   Geben Sie x und r an. Welchen dezimalen Wert hat die dargestellt Zahl? Geben Sie den Lösungsweg an.

Aufgabe 6 *MIMA-Architektur* (5 Punkte)

Die MIMA ist die Ihnen aus der Vorlesung bekannte mikroprogrammierte Minimalmaschine (**siehe nächste Seite: Architektur der MIMA**), die nach dem von-Neumann-Prinzip aufgebaut ist, d. h. Maschinenbefehle werden sequentiell abgearbeitet. In der Lese-Phase wird ein über IAR adressierter Befehl aus dem Speicher gelesen und im IR abgelegt. Die Lese-Phase dauert 5 Taktzyklen. Im 6. Taktzyklus wird der Befehl dekodiert (Dekodier-Phase). Die Ausführungsphase beginnt im 7. Taktzyklus. Nach der Ausführung des Befehls folgt ein Zugriff auf den nächsten Befehl.

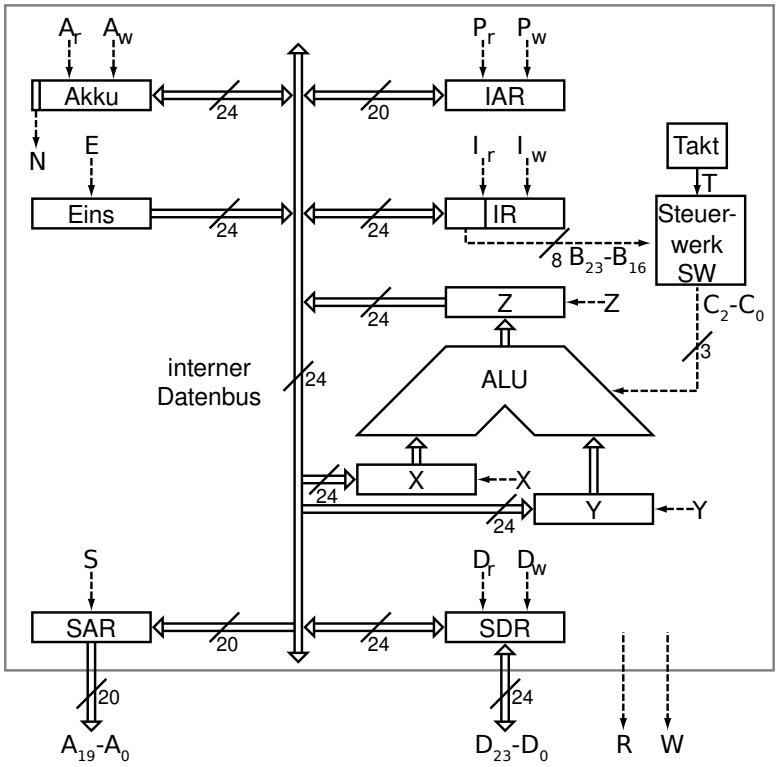
Nehmen Sie an, dass ein Hauptspeicherzugriff (Lesen und Schreiben) drei Takte dauert und währenddessen $\mathsf{R} = 1$ bzw. $\mathsf{W} = 1$ sein muss. Eine ALU-Operation sei nach einem Takt abgeschlossen.

Geben Sie das Mikroprogramm für die Lese-Phase (Fetch-Phase) in Register-Transfer-Schreibweise an, d. h. in der Form 5 P.

1. Takt: $\mathsf{IR} \rightarrow \mathsf{SAR}; \quad \mathsf{R} = 1$
2. Takt:

⋮

Architektur der MIMA



$C_2C_1C_0$	*ALU Operation*
0 0 0	tue nichts (d.h. Z → Z)
0 0 1	X + Y → Z
0 1 0	rotiere X nach rechts → Z
0 1 1	X AND Y → Z
1 0 0	XOR Y → Z
1 0 1	XXOR Y → Z
1 1 0	Eins-Komplement von X → Z
1 1 1	falls X = Y, -1 → Z, sonst 0 → Z

OpCode	*Mnemonik*		*Beschreibung*
0	LDC	c	C → Akku
1	LDV	a	<a> → Akku
2	STV	a	Akku → <a>
3	ADD	a	Akku + <a> → Akku
4	AND	a	Akku AND <a> → Akku
5	OR	a	Akku OR <a> → Akku
6	XOR	a	Akku XOR <a> → Akku
7	EQL	a	falls Akku = <a>:-1 → Akku sonst : 0 → Akku
8	JMP	a	a → IAR
9	JMN	a	falls Akku < 0 : a → IAR
F0	HALT		stoppt die MIMA
F1	NOT		bilde Eins-Komplement von Akku → Akku
F2	RAR		rotiere Akku eins nach rechts → Akku

Register

Akku: Akkumulator
X: 1. ALU Operand
Y: 2. ALU Operand
Z: ALU Ergebnis
Eins: Konstante 1
IAR: Instruktionsadressregister
IR: Instruktionsregister
SAR: Speicheradressregister
SDR: Speicherdatenregister

Steuersignale vom SW

– für den internen Datenbus

A_r: Akku liest
A_w: Akku schreibt
X: X-Register liest
Y: Y-Register liest
Z: Z-Register schreibt
E: Eins-Register schreibt
P_r: IAR liest
P_w: IAR schreibt
I_r: IR liest
I_w: IR schreibt
D_r: SDR liest
D_w: SDR schreibt
S: SAR liest

– für die ALU

C_2-C_0: Operation auswählen

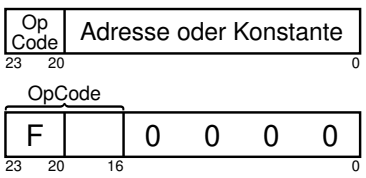
– für den Speicher

R: Leseanforderung
W: Schreibanforderung

Meldesignale zum SW

T: Takteingang
N: Vorzeichen des Akku
B_{23}-B_{16}: OpCode-Feld im IR

Befehlsformate



Aufgabe 7 *MIPS-Assembler* (9 Punkte)

1. Schreiben Sie die folgenden Kontrollstrukturen in MIPS-Assembler um. Die Variablen `i, j` und `k` stehen in den Registern `$s3, $s4` und `$s5`. [3 P.]

   (a)
   ```
   if ( i==j )
   k = i+j;
   ```

   (b)
   ```
   if ( i!=j )
   k = i+j;
   else
   k = i-j;
   ```

   (c)
   ```
   if ( i < j )
   k = 1;
   else
   k = 0;
   ```

2. Führen Sie den folgenden MIPS-Code aus und geben Sie die Änderungen in den Register- und Speicherinhalten an. Verwenden Sie die im Lösungsblatt angegebenen Tabellen. [4 P.]

```
addi $t3, $t0, 0x12
lw   $t1, 0($t3)
or   $t4, $t1, $t0
sw   $t4, 0x04($t2)
```

Registersatz

Register	Inhalt
$t0	0x16
$t1	0x32
$t2	0x24
$t3	0x00
$t4	0x1234

Hauptspeicher

Adresse	Inhalt
$0x20	0x10
$0x24	0x30
$0x28	0x40
$0x2C	0x50
$0x30	0x60

3. Der MIPS-Prozessor besitzt bekannterweise einen Fließkomma-Arithmetik-Prozessor als Coprozessor. Warum dürfen bei Arithmetik-Operationen mit doppelter Genauigkeit nur die Register des Coprozessors mit gerader Registernummer verwendet werden? [2 P.]

Aufgabe 8 *Pipelining* (10 Punkte)

1. Erläutern Sie die Aufgaben der einzelnen Pipeline-Stufen der DLX-Pipeline für bedingte Sprünge mit PC-relativer Adressierung. 2 P.

2. Das folgende MIPS-Programmstück soll auf einem Prozessor mit DLX-Pipeline ausgeführt werden.

```
S1:    lw    $t1, 1000($zero)
S2:    lw    $t2, 1004($zero)
S3:    add   $t3, $t2, $t1
S4:    addi  $t4, $t2, 8
S5:    subi  $t5, $zero, 2
S6:    and   $t5, $t3, $t2
S7:    sw    $t4, 1000($zero)
S8:    sw    $t2, 1004($zero)
S9:    sw    $t3, 1008($zero)
```

   (a) Bestimmen und klassifizieren Sie alle Datenabhängigkeiten und im Programmstück. 5 P.

   (b) Nehmen Sie an, dass keine *Forwarding*-Techniken implementiert sind und die auftretenden Pipelinekonflikte durch Einfügen von NOP (*No Operation*) Befehlen behoben werden müssen. 3 P.

   Ergänzen Sie das obige Programm, so dass es korrekte Ergebnisse liefert. Sie dürfen dabei die Reihenfolge der Befehle **nicht** ändern und so wenig NOP-Befehle wie möglich einfügen. Sie können die Bezeichner S1, S2, usw. als Abkürzung der gesamten Befehlszeile nutzen.

Aufgabe 9 *Cache-Speicher* (12 Punkte)

1. Bei einem Cache-Speicher mit einer Speicherkapazität von 256 KiByte ist die Hauptspeicheradresse in ein 16 Bit Tag-Feld, ein 12 Bit Index-Feld und ein 4 Bit Byte-Offset unterteilt. Geben Sie bei der Beantwortung der folgenden Fragen den Lösungsweg an.

   (a) Wie viele Einträge besitzt der Cache-Speicher? 1 P.

   (b) Wie ist der Cache-Speicher organisiert? 2 P.

2. Es soll ein 9-fach-assoziativer *(9-way set associative cache)* Cache-Speicher mit 64 Sätzen und einer Blockgröße von 16 Byte realisiert werden. Nehmen Sie an, dass die Hauptspeicheradresse 32 Bit breit ist. Zur Verwaltung eines Cacheblocks werden zwei Statusbits (*Valid*-Bit und *Dirty*-Bit) verwendet.

   Bestimmen Sie den insgesamt erforderlichen Speicherbedarf zur Realisierung dieses Cache-Speichers. Geben Sie den Lösungsweg und das ausgerechnete Endergebnis in Bytes an. 3 P.

3. Gegeben sei ein 2-fach-assoziativer Cache-Speicher (*2-way set associative cache*) mit einer Speicherkapazität von 128 Byte, einer Blockgröße von 16 Byte und einer LRU-Ersetzungsstrategie (*Least Resently Used*). Als Aktualisierungsstrategie wird das Rückschreib-Verfahren *(write back)* verwendet. Der Cache-Speicher sei initial leer. Betrachten Sie die folgenden Lese- und Schreibzugriffe auf die in dezimaler Schreibweise angegebenen Adressen:

Adresse	**64**	**32**	**64**	**0**	**112**	**64**	**128**	**48**	**240**	**0**
read/write	r	r	r	r	w	w	r	r	r	w
Index	0	2								
Tag	1	0								
Hit/Miss	Miss									
write back?	nein									

   Vervollständigen Sie diese Tabelle im Lösungsblatt. Verwenden Sie dabei **Miss** für Cache-Miss und **Hit** für Cache-Hit. Geben Sie in der letzten Zeile der Tabelle an, ob der entsprechende Cacheblock in den Hauptspeicher zurückkopiert werden muss (**ja**) oder nicht (**nein**). 6 P.

Aufgabe 10 *Speicher & Speicherverwaltung* (9 Punkte)

1. Was versteht man unter einer Speicherhierarchie? 2 P.
2. Wie viele Adressleitungen sind erforderlich bei einem Speicherbaustein mit einer Kapazität von 4096 Bits und einer 512×8-Organisation? Begründen Sie Ihre Antwort. 1 P.
3. Wie viele RAM-Bausteine der Organisation 8k×1 sind notwendig, um einen Speicher mit einer Kapazität von 8k Wörter und einer Wortbreite von 16 Bit zu realisieren? Begründen Sie Ihre Antwort. 1 P.
4. Wie ist ein ROM-Baustein mit der Speicherkapazität von 2048 Bits und 7 Adressleitungen organisiert? Begründen Sie Ihre Antwort. 1 P.

Ein Rechnersystem enthält eine Speicherverwaltungseinheit (MMU) zur Umsetzung von virtuellen in physikalische Seitenadressen (Abbildung 5). Die MMU bildet einen virtuellen Adressraum von 2^V Bytes auf einen physikalischen Adressraum der Größe 2^M Bytes ab und benutzt dabei Seiten der Größe 2^P Bytes. Nehmen Sie an, dass die MMU eine Byte-Adressierung benutzt und die gesamte Seitentabelle im Hauptspeicher ist.

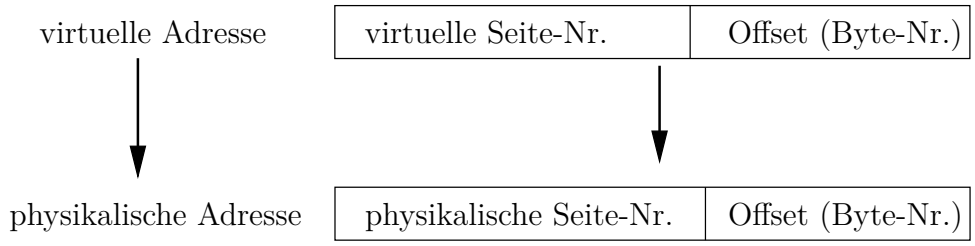


Abbildung 5: Format der virtuellen und physikalischen Adressen

5. In der folgenden Tabelle ist ein Ausschnitt aus der Seitentabelle gegeben. Welchen physikalischen Adressen entsprechen die dezimalen virtuellen Adressen 3112 und 1417, wenn $P = 9$ ist? Geben Sie den Lösungsweg an. 4 P.

Virtuelle Seitennummer	Physikalische Seitennummer
0	4
1	2
2	7
3	5
4	3
5	1
6	8
7	0
⋮	⋮