



UNIVERSITÄT KARLSRUHE (TH)

Fakultät für Informatik

Institut für Prozessrechentechnik, Automation und Robotik (IPR)

Prof. Dr. U. Brinkschulte, Dr. T. Asfour

## Aufgabenblätter

zur Klausur „Technische Informatik I/II“

am 26. Januar 2006, 18.00 – 20.00 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 die Endergebnisse 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 bitte so kurz wie möglich. (Der auf den Lösungsblättern für eine Aufgabe vorgesehene Platz steht übrigens in keinem Zusammenhang mit dem Umfang einer korrekten Lösung!)
- Die Gesamtpunktzahl beträgt 90 Punkte. Zum Bestehen der Klausur sind mindestens 40 Punkte zu erreichen.

*Viel Erfolg und viel Glück !*

**Aufgabe 1** *Schaltfunktionen*

(11 Punkte)

Eine unvollständig definierte Schaltfunktion  $y = f(d, c, b, a)$  sei durch ihre Eins- und *don't care*-Stellen (Abkürzung d) gegeben:

$$y = \text{MINt}(0, 1, 7, 8, 15) \vee d(4, 9)$$

1. Tragen Sie alle Primimplikanten der Funktion ins KV-Diagramm im Lösungsblatt ein und geben Sie eine disjunktive Minimalform (DMF) der Funktion  $f$  an. 2 P.
2. Tragen Sie alle Primimplikate der Funktion ins KV-Diagramm im Lösungsblatt ein und geben Sie eine konjunktive Minimalform (KMF) der Funktion  $f$  an. 3 P.

Gegeben sei eine Schaltfunktion  $z = g(d, c, b, a)$ , von der man weiss, dass  $\bar{c} b \bar{a}$  und  $\bar{d} \bar{c} \bar{a}$  *Kernprimimplikanten* dieser Funktion sind.

3. Welche der im Lösungsblatt angegebenen Produktterme können definitiv **keine** Primimplikanten der Funktion  $z$  sein? Tragen Sie in diesem Fall ein **X** in der Tabelle im Lösungsblatt. Geben Sie jeweils eine Begründung Ihrer Antwort an. (Keine Punkte bei fehlender Begründung) 2 P.

Die Funktionen  $f_1$ ,  $f_2$ ,  $f_3$  und  $f_4$  sollen mit Hilfe eines PLA-Bausteins realisiert werden.

$$f_1(c, b, a) = \text{MINt}(3, 6, 7)$$

$$f_2(c, b, a) = \text{MINt}(0, 1, 4, 5, 6)$$

$$f_3(c, b, a) = \text{MINt}(2, 3, 4)$$

$$f_4(c, b, a) = \text{MINt}(2, 3, 4, 7)$$

4. Personalisieren Sie den im Lösungsblatt angegebenen PLA-Baustein, indem Sie geeignete Leitungskreuzungen der UND- und der ODER-Matrix markieren. 4 P.

## Aufgabe 2 *Spezielle Bausteine*

(8 Punkte)

1. Die Schaltfunktion

$$y = f(c, b, a) = \bar{c} \vee \bar{b} \bar{a}$$

4 P.

soll in der CMOS-Technologie realisiert werden. Es stehen Ihnen ein NOR-Gatter, ein NAND-Gatter, und ein Inverter-Gatter zur Verfügung. Geben Sie das Transistor-Schaltbild an.

2. Für die Fehlererkennung in einem 4-Bit-Code wird ein Schaltnetz benötigt, welches die anliegenden Eingangsvariablen auf ungerade Parität überprüft, d.h die Funktion

4 P.

$$p = \text{odd}(w, x, y, z) = w \oplus x \oplus y \oplus z$$

realisiert. Dabei bezeichnet  $\oplus$  den Quersummen-Operator.

Realisieren Sie das Schaltnetz unter ausschließlicher Verwendung eines 8:1-Multiplexers und eines Inverters. Zeichnen Sie die Schaltung.

## Aufgabe 3 *Schaltwerke*

(13 Punkte)

Gegeben sei ein synchrones Schaltwerk mit 2 flankengesteuerten JK-Flipflops, einer Eingangsvariable  $x$  und einer Ausgangsvariable  $y$  (siehe Bild 1).

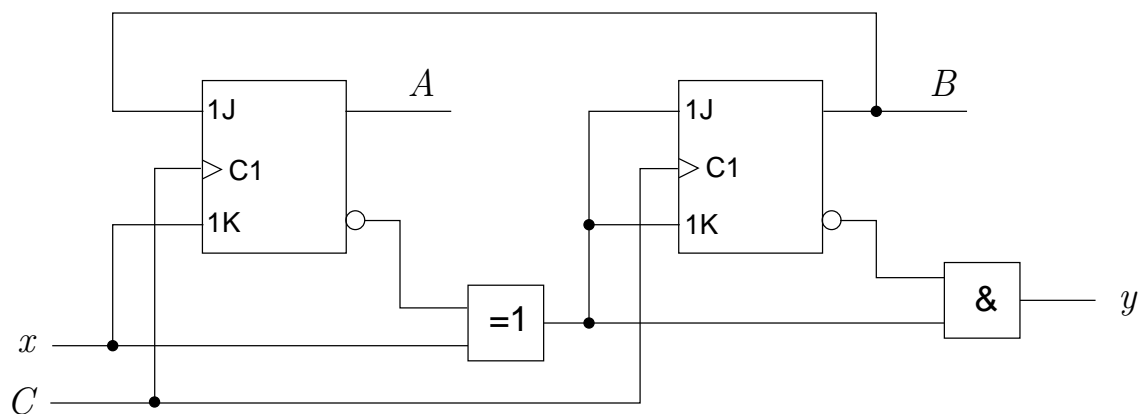


Bild 1: Schaltwerk I

1. Wieviele Zustände kann das Schaltwerk maximal haben?

1 P.

2. Stellen Sie die kodierte Ablaufabelle des Schaltwerks auf. Stellen Sie Ihre Lösung schrittweise dar.

5 P.

**Hinweis:** Bestimmen Sie die Ansteuerfunktionen der Flipflops, die Zustandsübergangsgleichungen und die Ausgabefunktion des Schaltwerks.

In Bild 2 ist der Automatengraph eines zweiten synchronen Schaltwerks mit der Eingangsvariablen  $e$ , der Ausgangsvariablen  $a$  und den Zuständen  $Z_0, Z_1, Z_2$  und  $Z_3$  dargestellt.

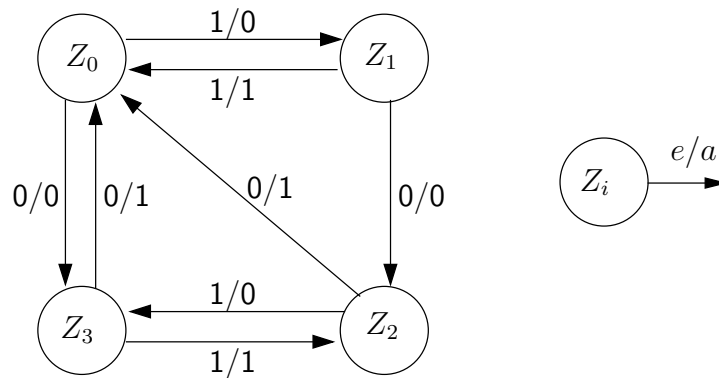


Bild 2: Automatengraph vom Schaltwerk II

3. Nehmen Sie an, dass sich das Schaltwerk am Anfang im Zustand  $Z_0$  befindet. Vervollständigen Sie die im Lösungsblatt angegebene Tabelle der Zustands-, Eingabe- und Ausgabefolgen des Schaltwerks. 3 P.

In Bild 3 ist ein weiteres Schaltwerk dargestellt.

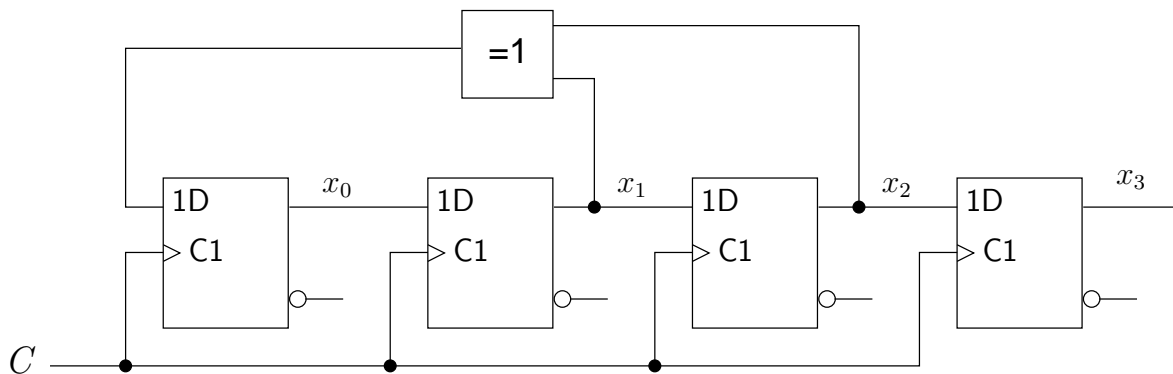


Bild 3: Schaltwerk III

4. Vervollständigen Sie die Verläufe der Signale  $x_0, x_1, x_2$  und  $x_3$  im angegebenen Zeitdiagramm im Lösungsblatt. 4 P.

## Aufgabe 4 Rechnerarithmetik (9 Punkte)

**Hinweis:** Geben Sie in dieser Aufgabe *immer* den Rechenweg an.

1. Wandeln Sie die Zahl  $86, 22_{10}$  in eine Zahl zur Basis 5 um. 1 P.
2. Wandeln Sie die Zahl  $435, 317_8$  in eine Zahl zur Basis 16 um. 1 P.
3. Wandeln Sie die Zahl  $-65_{10}$  in eine 16-Bit Zweierkomplement Zahl um. 1 P.
4. Wandeln Sie die Zweierkomplement-Zahl  $(1111111100111100)_{ZK}$  in eine dezimale Zahl um. 1 P.
5. Geben Sie die Dezimalzahl der folgenden im Maschinenformat des IEEE-Standards dargestellten Gleitkomma-Zahl an. 2 P.

$$(1100\ 0001\ 1110\ 0101\ 0000\ 0000\ 0000\ 0000)_{IEEE}$$

6. Betrachten Sie einen 4-Bit-Carry Look-ahead-Addierer mit den Eingangsvariablen  $A = (a_3a_2a_1a_0)$ ,  $B = (b_3b_2b_1b_0)$  und dem Übertrag  $\ddot{u}_0$  sowie den Ausgangsvariablen  $S = (s_3s_2s_1s_0)$ . 3 P.

In der  $i$ -ten Stelle gilt für die Signale  $g_i$  (*generate carry*, erzeuge Übertrag) und  $p_i$  (*propagate carry*, leite Übertrag weiter):

$$\begin{aligned} g_i &= a_i \cdot b_i \\ p_i &= a_i \not\leftrightarrow b_i \end{aligned}$$

Die Summe  $s_i$  und den Übertrag  $\ddot{u}_{i+1}$  sind wie folgt definiert

$$\begin{aligned} s_i &= p_i \not\leftrightarrow \ddot{u}_i \\ \ddot{u}_{i+1} &= g_i \vee p_i \cdot \ddot{u}_i \end{aligned}$$

Es sei  $A = 1110$  und  $\ddot{u}_0 = 1$ . Geben Sie  $\ddot{u}_4$  im vollständigen Operatorensystem  $(\wedge, \vee, \neg)$  als Funktion von  $b_3, b_2, b_1$  und  $b_0$  an.

## Aufgabe 5 Multiple Choice (4 Punkte)

Kreuzen Sie bitte für jede der Behauptungen im Lösungsblatt an, ob sie Ihrer Meinung nach richtig oder falsch ist. Nicht angekreuzte Behauptungen zählen nicht und gehen somit nicht in die Bewertung ein. Zur Ermittlung der Punktzahl werden von den richtig angekreuzten Behauptungen die falsch angekreuzten Behauptungen abgezogen.

## Aufgabe 6 *MIMA-Architektur* (7 Punkte)

Die MIMA ist die Ihnen aus der Vorlesung bekannte mikroprogrammierte Minimalmaschine (siehe Beiblatt: **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  $R = 1$  bzw.  $W = 1$  sein muss. Eine ALU-Operation sei nach einem Takt abgeschlossen.

Das Mikroprogramm für die Lese-Phase besteht aus fünf Mikrobefehlen:

|  |   |            |
|--|---|------------|
| 1. Takt: IAR $\rightarrow$ SAR; IAR $\rightarrow$ X; R = 1 | } | Lese-Phase |
| 2. Takt: Eins $\rightarrow$ Y; R = 1                       |   |            |
| 3. Takt: ALU auf Addieren; R = 1                           |   |            |
| 4. Takt: Z $\rightarrow$ IAR                               |   |            |
| 5. Takt: SDR $\rightarrow$ IR                              |   |            |

1. Geben Sie die Mikroprogramme für die Ausführungsphasen der folgenden Maschinenbefehle an (jeweils ab dem 7. Takt, also nach der Lese-Phase und der Dekodier-Phase):

4 P.

LDC, STV, ADD, EQL

### Beispiel:

AND:

|           |                             |
|-----------|-----------------------------|
| 7. Takt:  | IR $\rightarrow$ SAR; R = 1 |
| 8. Takt:  | Akku $\rightarrow$ X; R = 1 |
| 9. Takt:  | R = 1                       |
| 10. Takt: | SDR $\rightarrow$ Y         |
| 11. Takt: | ALU auf AND                 |
| 12. Takt: | Z $\rightarrow$ Akku        |

2. Wie viele Operationen kann die ALU der MIMA ausführen?
3. Wie viele Befehle können für die MIMA maximal implementiert werden?
4. Wie werden aus dem 24-Bit breiten Datenbus die Adressen extrahiert?

1 P.

1 P.

1 P.

**Aufgabe 7** *MIPS-Assembler*

(12 Punkte)

1. Übersetzen Sie die folgenden C-Kontrollstrukturen in MIPS-Assembler. Sie dürfen dabei **nur** die MIPS-Befehle `add`, `lw`, `beq`, `bne` und `j` verwenden.

Nehmen Sie an, dass die Variablen `i`, `j` und `k` in den Registern `$s3`, `$s4` und `$s5` stehen und dass sich die Anfangsadresse von `A` im Register `$s6` befindet. Dabei ist `A` ein Feld aus 32-Bit Integerzahlen. Verwenden Sie die Register `$t0` und `$t1` zur Speicherung temporärer Variablen.

- (a) do-while-Schleife:

|      |
|------|
| 3 P. |
|------|

```
do
    i = i + j;
while ( A[i] == k )
```

- (b) else-if-Anweisungen:

|      |
|------|
| 5 P. |
|------|

```
if (i == j)
    i = i + A[k];
else if (i == k)
    i = i + A[j];
else
    i = j + k;
```

2. Übersetzen Sie den folgenden C-Code nach MIPS-Assembler:

|      |
|------|
| 4 P. |
|------|

(a) `ptr = &i;`

(b) `i = *ptr;`

(c) `i = **ptr;`

(d) `*ptr = i;`

Dabei gilt: `int i, int *ptr;`

**Aufgabe 8** *Pipelining*

(6 Punkte)

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

```

S1:   lw    $t1, 0x100($zero)
S2:   lw    $t2, 0x104($zero)
S3:   add   $t3, $t2, $t1
S4:   addi  $t1, $t2, 8
S5:   subi  $t4, $zero, 2
S6:   and   $t5, $t3, $t2
S7:   sw    $t4, 0x100($zero)
S8:   sw    $t5, 0x104($zero)
S9:   sw    $t1, 0x108($zero)

```

1. Bestimmen Sie alle *echten* Datenabhängigkeiten im Programmstück. 4 P.
2. Nehmen Sie an, dass sowohl *load forwarding* als auch *result forwarding* implementiert ist. 2 P.

Die auftretenden Pipelinekonflikte sollen durch das Einfügen von möglichst wenigen NOP-Befehlen (*No Operation*) behoben werden.

Ergänzen Sie das obige Programm, so dass es korrekte Ergebnisse liefert. Sie dürfen dabei die Reihenfolge der Befehle **nicht** ändern.

**Aufgabe 9** *Cache-Speicher*

(12 Punkte)

1. Bei einem Cache-Speicher mit einer Speicherkapazität von 512 KByte 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) Bestimmen Sie die Blockgröße in Bytes. 1 P.
  - (b) Wieviele Einträge besitzt der Cache-Speicher? 1 P.
  - (c) Wie ist der Cache-Speicher organisiert? 2 P.
2. Es soll ein 4-fach-assoziativer (*4-way set associative cache*) Cache-Speicher mit 256 Sätzen und einer Blockgröße von 8 Byte realisiert werden. Nehmen Sie an, dass die Hauptspeicheradresse 32 Bit breit ist. Zur Verwaltung eines Cacheblocks wird nur ein Statusbit (*Valid*-Bit: V) verwendet.

Bestimmen Sie den insgesamt erforderlichen Speicherbedarf zur Realisierung dieses Cache-Speichers. 3 P.

3. Gegeben sei ein direkt-abgebildeter Cache-Speicher (*direct mapped cache*) mit einer Speicherkapazität von 32 Byte und einer Blockgröße von 8 Byte. Als Aktualisierungsstrategie wird ein Durchschreibverfahren (*write through policy*) verwendet. Bei dieser Aktualisierungsstrategie wird ein CPU-Datum bei einem *write miss* nur in den Speicher geschrieben. Bei einem *write hit* wird ein CPU-Datum sowohl in den Cache als auch in den Speicher geschrieben.

Betrachten Sie die folgenden Lese- und Schreibzugriffe auf die in dezimaler Schreibweise angegebenen Adressen:

| Adresse    | 0    | 16 | 48 | 8 | 56 | 16 | 8 | 56 | 32 | 0 |
|------------|------|----|----|---|----|----|---|----|----|---|
| read/write | r    | r  | r  | r | r  | r  | r | w  | w  | r |
| Index      | 0    | 2  |    |   |    |    |   |    |    |   |
| Tag        | 0    | 0  |    |   |    |    |   |    |    |   |
| Hit/Miss   | Miss |    |    |   |    |    |   |    |    |   |

Vervollständigen Sie die obige Tabelle im Lösungsblatt. Verwenden Sie dabei **Miss** für Cache-Miss und **Hit** für Cache-Hit.

5 P.

## Aufgabe 10 Speicher

(8 Punkte)

- Skizzieren Sie den Aufbau einer dynamischen RAM-Speicherzelle. 2 P.
- Wieviele Adressleitungen sind erforderlich bei einem Speicherbaustein mit einer Kapazität von 4096 Bits und einer 512×8-Organisation? Begründen Sie Ihre Antwort. 1 P.
- Wieviele RAM-Bausteine der Organisation 8k×1 sind notwendig, um einen Speicher mit einer Kapazität von 8k Wörter und einer Wortbreite von 8 Bit zu realisieren? Begründen Sie Ihre Antwort. 1 P.
- Wie ist ein ROM-Baustein mit der Speicherkapazität von 2048 Bits und 8 Adressleitungen organisiert? Begründen Sie Ihre Antwort. 1 P.
- Erläutern Sie die folgenden Begriffe:
  - RISC-Prozessor. 1 P.
  - Direkter Speicherzugriff (DMA). 1 P.
  - Translation-Lookaside-Buffer (TLB)*. 1 P.