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

Aufgabenblätter zur Prüfung

Digitaltechnik und Entwurfsverfahren (TI-1)

und

Rechnerorganisation (TI-2)

am 07. März 2025, 08:00 – 10:00 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 *Rechnerarithmetik* (9 Punkte)

1. Verwenden Sie das aus der Vorlesung bekannte *Horner-Schema* zum Umrechnen der Zahl 712_{10} der Basis 10 in die Basis 7. Geben Sie die Umrechnungsschritte an. 2 P.

2. Gegeben sei die Zahl $712,25_{10}$. Berechnen Sie die Darstellung dieser Zahl im IEEE 754-Standard für einfache Genauigkeit. Geben Sie die Umrechnungsschritte an. Geben Sie ebenfalls die hexadezimale Repräsentation der als IEEE 754-Standarddarstellung gefundenen Zahl an. 3 P.

3. Bei einem Multiplizierer werden die Teilprodukte p_0, p_1, p_2, p_3, p_4 und p_5 erzeugt. Für die Addition der Teilprodukte soll eine Schaltung entworfen werden, die sechs binäre Stellen addiert, um eine 3-Bit Summe $s_2\, s_1\, s_0$ zu berechnen (siehe Beispiel). 4 P.

   Beispiel:

p_0	0	1	1
p_1	0	1	0
p_2	0	1	1
p_3	0	1	0
p_4	0	1	0
$+\ p_5$	0	1	1
$s_2\, s_1\, s_0$	000	110	011

   Diese Aufgabe soll mit Hilfe der in Abbildung 1 dargestellten Schaltung gelöst werden. Vervollständigen Sie hierzu die Schaltung auf dem Lösungsblatt. Dabei dürfen Sie *keine* zusätzlichen Gatter verwenden.

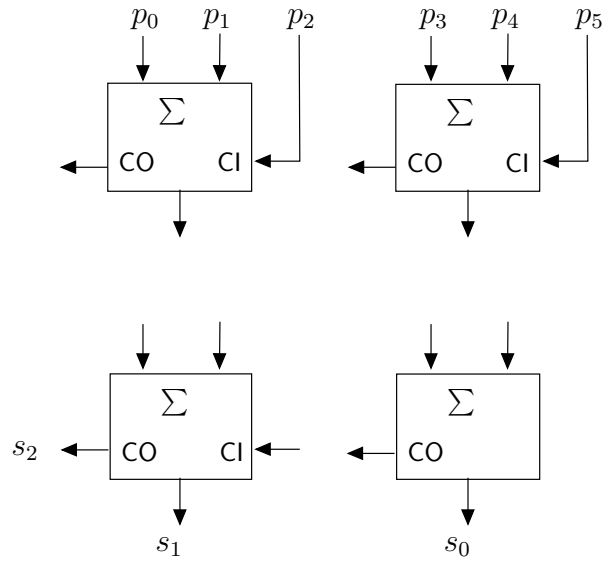


Abbildung 1: Schaltung zur Berechnung der Summe $s_2\, s_1\, s_0$

Aufgabe 2 *Schaltfunktion* (7 Punkte)

Gegeben sei die durch die folgende Wahrheitstabelle definierte Schaltfunktion f:

d	c	b	a	f
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

1. Tragen Sie die Schaltfunktion f in das bereitgestellte KV-Diagramm ein und bestimmen Sie alle Primimplikanten. 5 P.

2. Geben Sie alle disjunktiven Minimalformen von f an. 2 P.

Aufgabe 3 *Schaltnetze* (12 Punkte)

Eine Schaltfunkion z sei durch das Schaltnetz in Abbildung 2 gegeben.

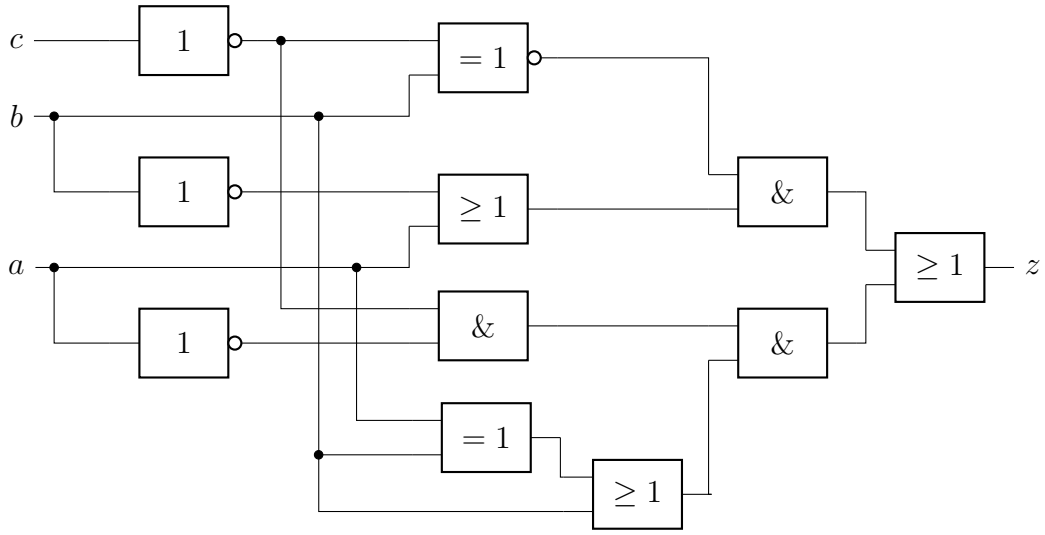


Abbildung 2: Schaltnetz der Funktion $z = f(c, b, a)$

1. Geben Sie zu dem in Abbildung 2 dargestellten Schaltnetz die Schaltfunktion z an. 1 P.
2. Formen Sie z schaltalgebraisch in eine minimierte zweistufige disjunktive Form um. 3 P.

Eine weitere Schaltfunktion sei durch die folgende Gleichung gegeben:

$$g(d, c, b, a) = \overline{a} \vee \overline{c}\,\overline{b} \vee c\,d.$$

Alle Variablen stehen **nur bejaht** zur Verfügung.

3. Realisieren Sie die Funktion g durch Verwendung von ausschließlich NAND-Gattern mit 2 Eingängen. Zeichnen Sie das Schaltnetz. 4 P.
4. Entwickeln Sie eine Multiplexer-Realisierung der Funktion g unter der Randbedingung, dass Sie nur über drei 2:1-Multiplexer und keine weiteren Schaltgatter verfügen. Zeichnen Sie das Schaltnetz. 4 P.

Aufgabe 4 *Laufzeiteffekte* (7 Punkte)

Gegeben sei der Funktionsausdruck einer Funktion i:

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

d \ b \ c \ a				
	1	1	0	0
	1	1	1	1
	1	1	1	0
	0	0	1	0

1. Untersuchen Sie die folgenden Übergänge des Funktionsausdrucks. Geben Sie jeweils an, ob der Übergang hazardbehaftet ist und falls ja um was für einen Fehler es sich dabei handeln würde. 6 P.

   (a) $B_0 \rightarrow B_8$

   (b) $B_0 \rightarrow B_{14}$

   (c) $B_3 \rightarrow B_9$

2. Wie lassen sich Funktionshazards beheben? Nennen Sie eine Möglichkeit. 1 P.

Aufgabe 5 *Schaltwerke* (10 Punkte)

Gegeben sei das in Abbildung 3 dargestellte Schaltwerk.

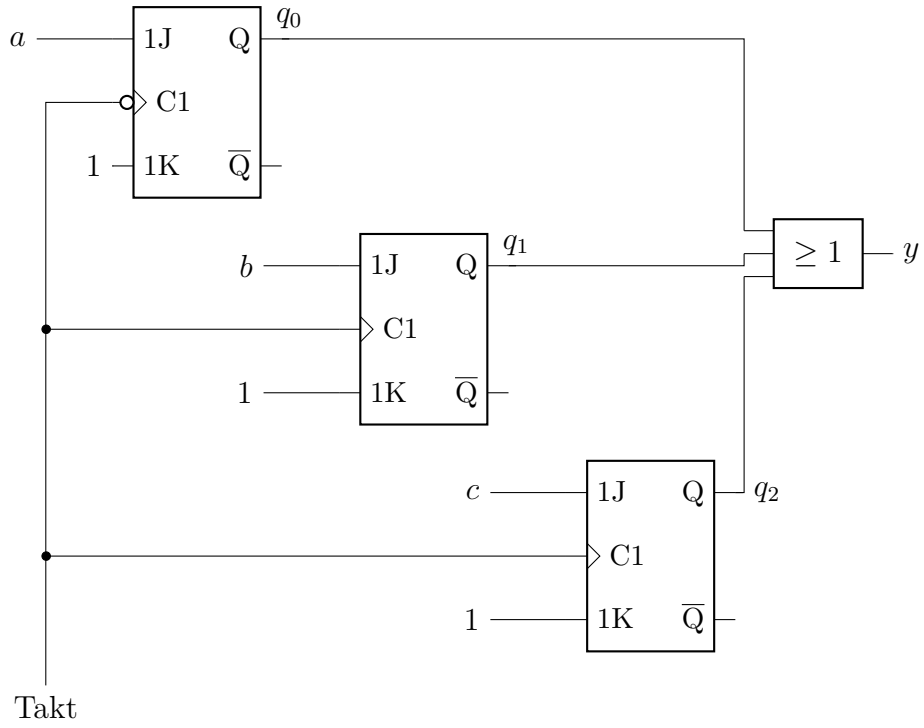


Abbildung 3: Schaubild eines Schaltwerks

1. Geben Sie die Verläufe der Signale q_0, q_1, q_2 und y an, indem Sie das auf dem Lösungsblatt angegebene Zeitdiagramm vervollständigen. (**Hinweis:** Achten Sie auf die Ansteuerung der Flipflops.) [7 P.]

2. Wie kann ein JK-Flipflop als Frequenzteiler für das Taktsignals f_0 realisiert werden? Zeichnen Sie das Schaubild des Schaltwerks. [1 P.]

3. Wie wird ein T-Flipflop aus einem D-Flipflop realisiert? Zeichnen Sie das Schaubild des Schaltwerks. [2 P.]

Aufgabe 6 *MIMA-Architektur* (10 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 $\mathsf{R} = 1$ bzw. $\mathsf{W} = 1$ sein muss. Eine ALU-Operation sei nach einem Takt abgeschlossen.

1. 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:
   ⋮

2. Welchem Ausführungsmodell unterliegt die MIMA-Architektur? Was zeichnet dieses Ausführungsmodell aus? — 2 P.

Gegeben sei der folgende MIMA-Assemblercode:

```
1                LDC 0x0ABCD
2                ADD 0x00001
3                XOR 0x00002
4                EQL 0x00003
```

Der Speicher sei wie folgt belegt:

Adresse	**Datum**
0x00001	`0x000011`
0x00002	`0x000AB0`
0x00003	`0x00A16E`

3. Geben Sie die gesamten hexadezimalen Inhalte des Akkumulators nach jedem Befehl der Codesequenz an? Welcher Wert hat das Meldesignal N nach Ablauf des Codes? — 3 P.

Aufgabe 7 *RISC-V Assembler* (8 Punkte)

Betrachten Sie den folgenden RISC-V Assemblercode:

```
    li    t0, 0           # Zähler i = 0
    li    t1, 10          # Obergrenze (n = 10)
    li    t2, 1           # Startwert
    li    t3, 0x10010000 # Speicheradresse für das Ergebnis
    li    t4, 0           # Summe = 0

loop:
    sll   t5, t2, t0      # t5 = 1 << i
    add   t4, t4, t5      # Ergebnis aufsummieren
    addi  t0, t0, 1       # i = i + 1
    blt   t0, t1, loop    # Wiederhole, solange i < n

    sw    t4, 0(t3)       # Speichere das Ergebnis in den Speicher

```

1. Beschreiben Sie in Worten, was obiger Code berechnet. 2 P.
2. Was ist der Inhalt des Speichers an der Adresse `0x10010000` in hexadezimaler Schreibweise nach Ablauf des obigen Assemblercodes. 2 P.
3. Ändern Sie den Code so, dass die Schleife nur gerade Werte von i berücksichtigt ($i = 0, 2, 4, 6, \dots$). 2 P.
4. Ersetzen Sie den Speicherzugriff (`sw t4, 0 (t3)`) durch eine Implementierung, die das Ergebnis in einem Array speichert, wobei jedes Zwischenergebnis in ein separates Speicherwort geschrieben wird. 2 P.

Aufgabe 8 *Pipelining* (9 Punkte)

1. Erläutern Sie die Aufgaben der einzelnen Pipelinestufen der Ihnen aus der Vorlesung bekannten RISC-V Pipeline bei der Abarbeitung eines Lade-Befehls. 2 P.

2. Warum können Ausgabeabhängigkeiten (*output dependence*) und Gegenabhängigkeiten (*anti-dependence*) in der Ihnen aus der Vorlesung bekannten RISC-V Pipeline nicht zu Konflikten führen? 2 P.

3. Warum stellen bedingte Sprünge generell ein Problem für die Pipelineimplementierung dar? Geben Sie zwei Möglichkeiten zur Behandlung dieses Problems an. 3 P.

4. Erklären Sie die Begriffe *Result-Forwarding* und *Load-Forwarding*? 2 P.

Aufgabe 9 *Cache-Speicher* (10 Punkte)

Gegeben sei ein Mikroprozessorsystem mit einem Cache und einem Hauptspeicher. Um das System schlank zu halten, wird **keine** virtuelle Speicherverwaltung verwendet und alle nötigen Daten können im Hauptspeicher vorgehalten werden. Der Hauptspeicher verfügt über eine Kapazität von 1024 KiByte. Der Cache ist als 4-fach-satzassoziativer Cache mit einer Größe von 64 KiByte und einer Blockgröße von 64 Byte organisiert. Der Cache ist mit einer Write-Through-Policy mit Write-Allocate ausgestattet. Nach langer Laufzeit des Systems wird festgestellt, dass die Trefferquote des Caches bei 90% liegt. Ein Zugriff auf den Hauptspeicher dauert 200 ns, ein Zugriff auf den Cache 10 ns.

1. Berechnen Sie die durchschnittliche Zugriffsdauer t_m auf ein Datum. 1 P.
2. Berechnen Sie die Anzahl der Sätze des Caches. 1 P.
3. Berechnen Sie die Länge des Tags. 1 P.

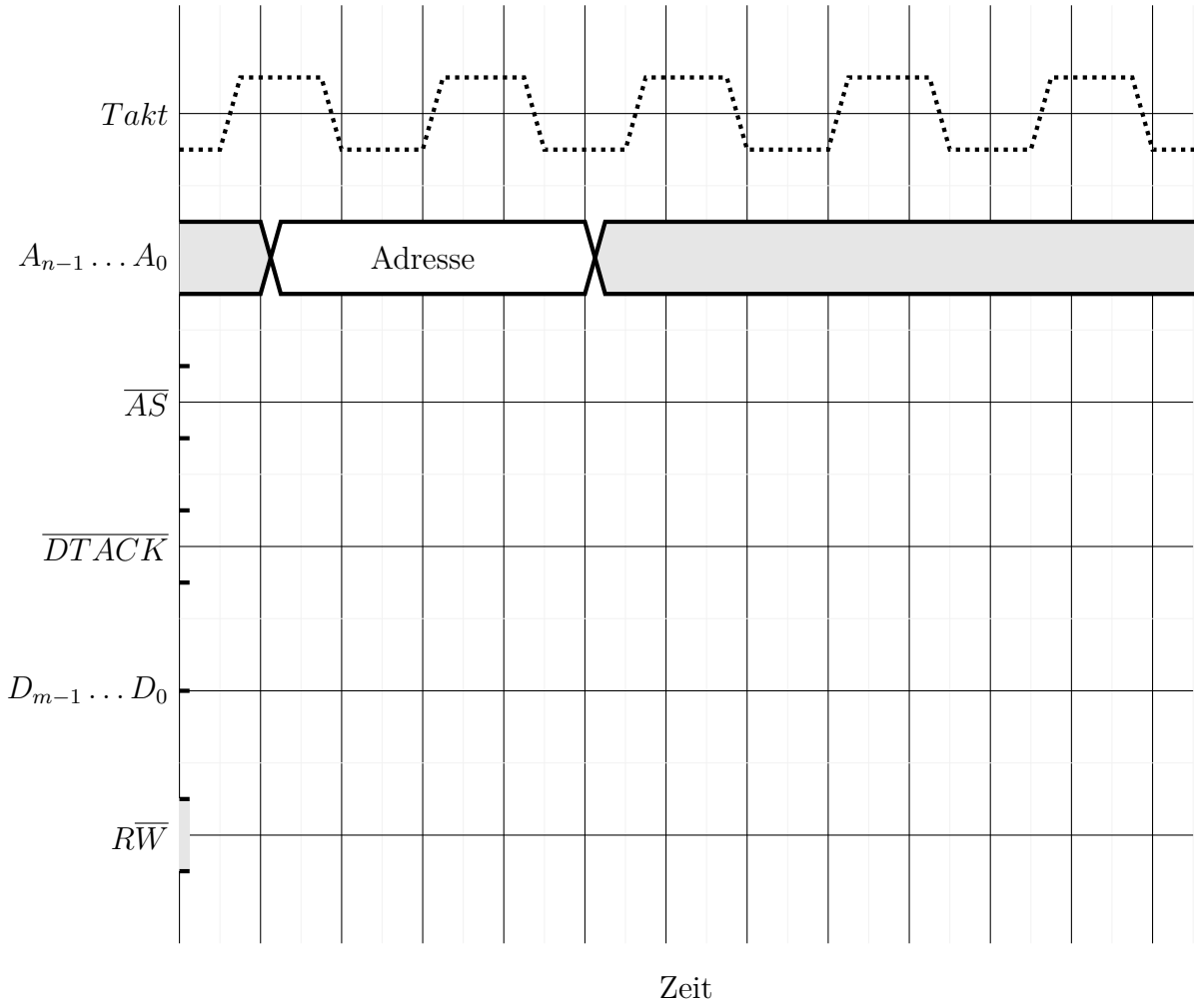
Betrachten Sie die folgenden Lese- und Schreibzugriffe auf die in hexadezimaler Schreibweise angegebenen Adressen:

hex. Adresse	**0x00 000**	**0x00 A32**	**0xFA 711**	**0x36 421**	**0xFA 717**	**0x36 729**	**0x5F 321**	**0x00 038**	**0x76 F29**	**0x76 F0A**
read/write	r	r	w	r	w	w	r	w	w	r
hex. Index										
hex. Tag										
Hit/Miss										

4. Vervollständigen Sie die obige Tabelle auf dem Lösungsblatt. Verwenden Sie dabei **Miss** für Cache-Miss und **Hit** für Cache-Hit. Gehen Sie davon aus, dass der Cache zu Beginn leer ist. 5 P.
5. Was verändert sich an der Tabelle, wenn als Aktualisierungsstrategie Write-Through mit Write-**No**-Allocate verwendet wird? Markieren Sie die entsprechenden Zellen in der Tabelle. 2 P.

Aufgabe 10 *Verbindungsstrukturen* (8 Punkte)

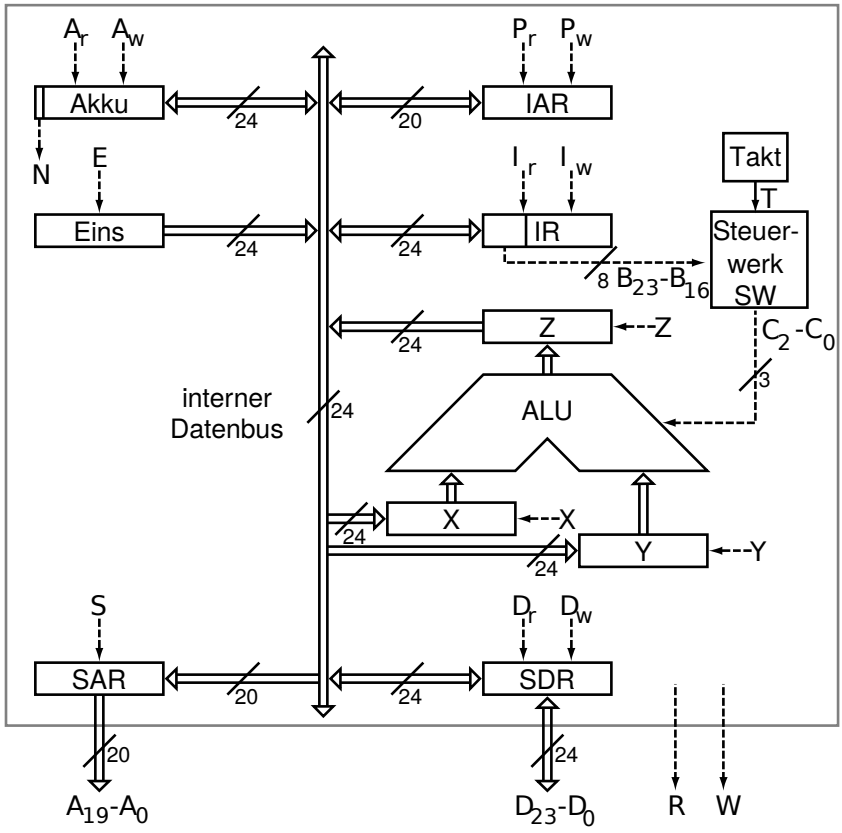
1. Gegeben sei ein asynchroner Bus als Verbindung zwischen Mikroprozessor und Speicherkomponente. Der Prozessor will lesend auf den Speicher zugreifen. Vervollständigen Sie das bereitgestellte Zeitdiagramm für das Lesen von Daten. 4 P.



2. Sie wollen mehrere Prozessorchips miteinander verbinden. Welche Verbindungsstruktur würden Sie für eine performante Implementierung wählen und warum? Nennen Sie ein Beispiel aus der Vorlesung. 3 P.

3. Was bedeutet *USB* ausgeschrieben? 1 P.

Architektur der MIMA



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

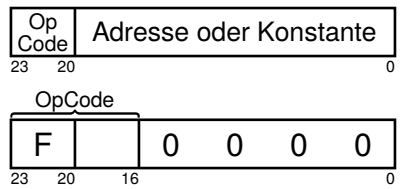
Meldesignale zum SW

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

$C_2C_1C_0$	*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	X OR Y → Z
1 0 1	X XOR 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

Befehlsformate



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

Lösungsblätter zur Klausur

Digitaltechnik und Entwurfsverfahren (TI-1)

und

Rechnerorganisation (TI-2)

am 07. März 2025, 08:00 – 10:00 Uhr

Name:	Vorname:	Matrikelnummer:

Digitaltechnik und Entwurfsverfahren (TI-1)	
Aufgabe 1	von 9 Punkten
Aufgabe 2	von 7 Punkten
Aufgabe 3	von 12 Punkten
Aufgabe 4	von 7 Punkten
Aufgabe 5	von 10 Punkten

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

Übungsscheine:	
Digitaltechnik und Entwurfsverfahren (TI-1)	von 2 Punkten
Rechnerorganisation (TI-2)	von 2 Punkten

Gesamtpunktzahl:	

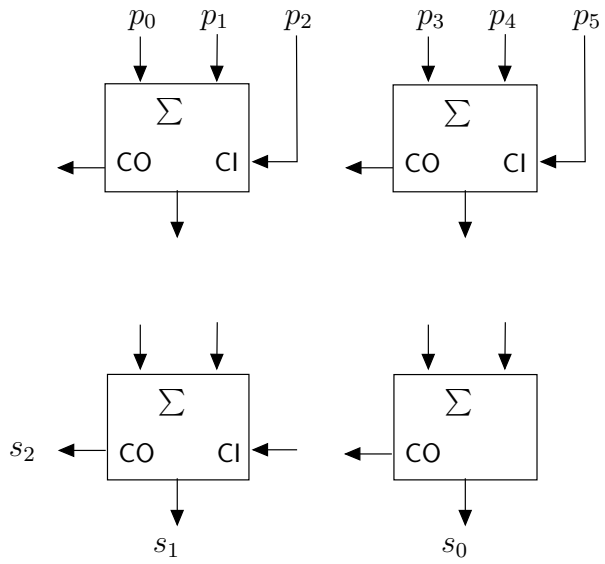
	Note:

Aufgabe 1 *Rechnerarithmetik*

1. 712_{10} in Basis 7:

2. $712,25_{10}$ in IEEE 754-Standard für einfache Genauigkeit:

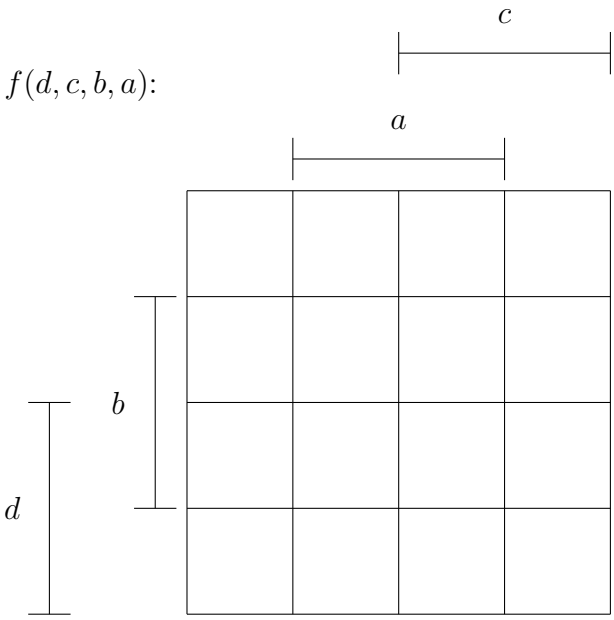
3. Schaltung zur Berechnung der Summe $s_2\, s_1\, s_0$:



Aufgabe 2 *Schaltfunktion*

1. KV-Diagramm:

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



Primimplikante:

2. Disjunktive Minimalformen:

Aufgabe 3 *Schaltnetze*

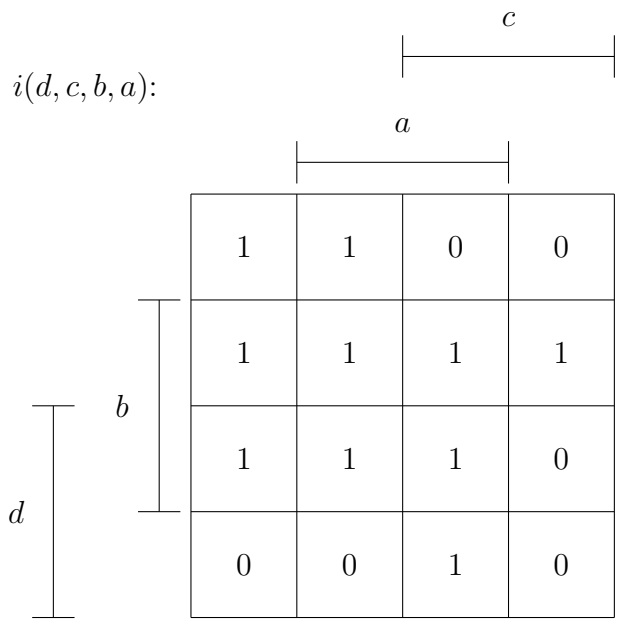
1. Schaltfunktion z:

2. Zweistufige disjunktive Form von z:

3. g durch NAND-Gatter:

4. Multiplexer-Realisierung von g:

Aufgabe 4 *Laufzeiteffekte*



1.

(a) $B_0 \rightarrow B_8$:

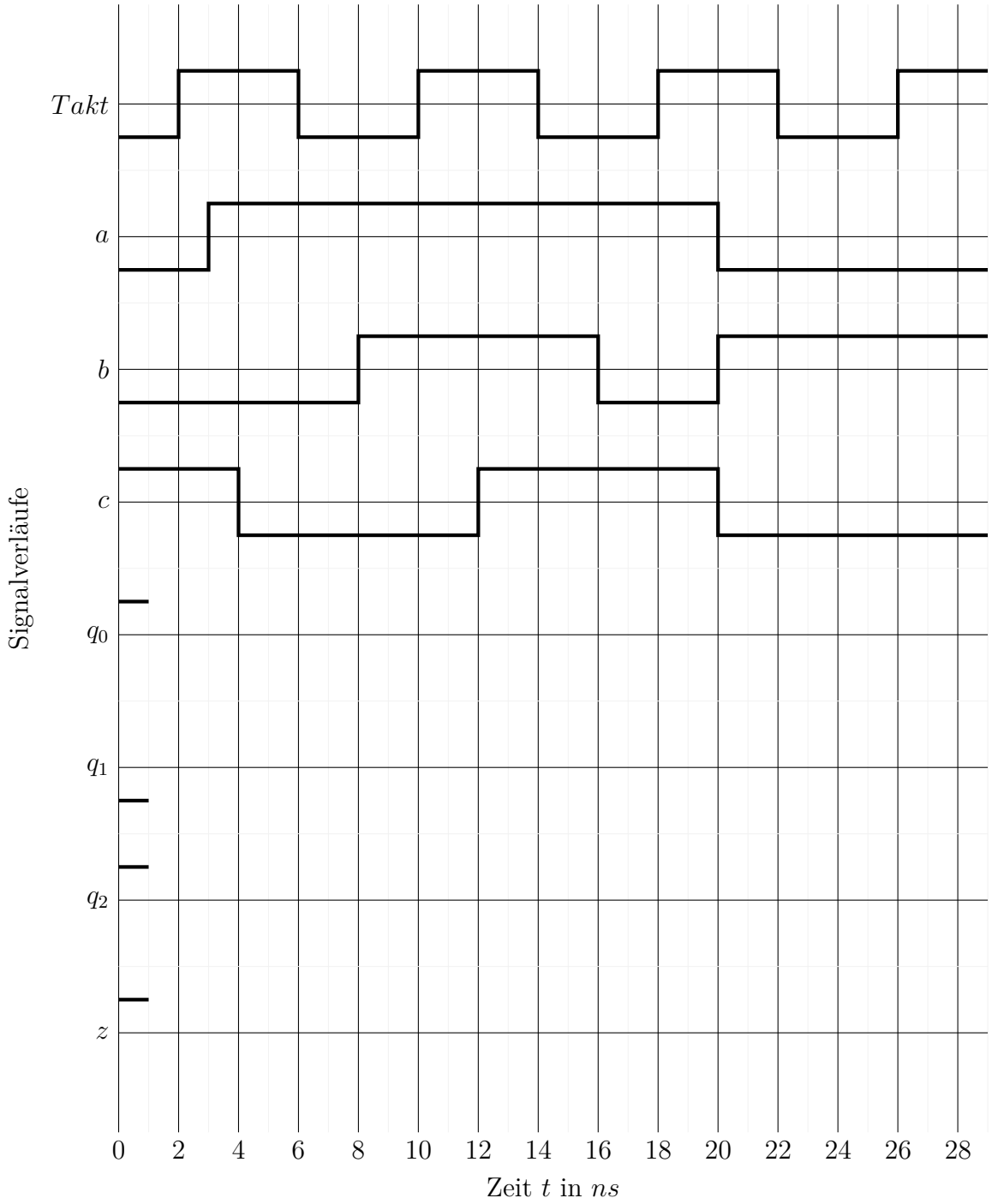
(b) $B_0 \rightarrow B_{14}$:

(c) $B_3 \rightarrow B_9$:

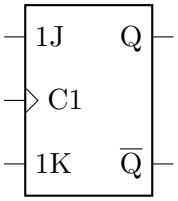
2. Behebungsmöglichkeit:

Aufgabe 5 *Schaltwerke*

1. Zeitdiagramm:



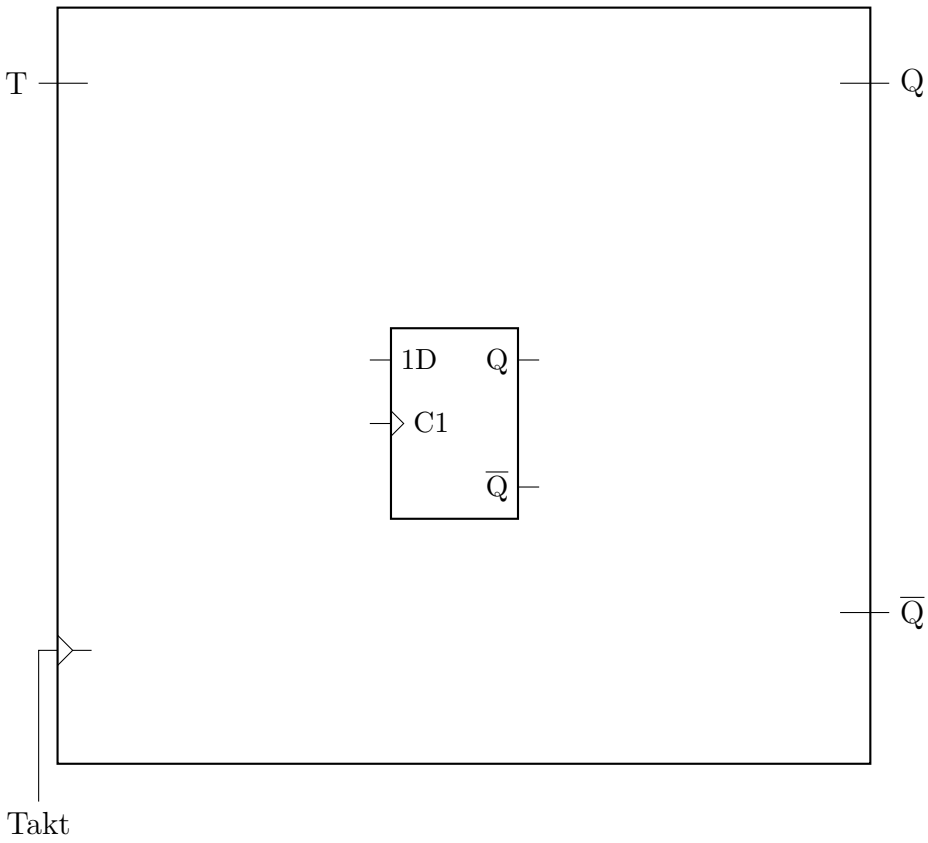
2. Frequenzteiler:



Takt mit f_0

Takt mit $\frac{f_0}{2}$

3. T-Flipflop aus D-Flipflop:



Aufgabe 6 *MIMA-Architektur*

1. Mikroprogramm für die Lese-Phase:

2. Ausführungsmodell:

3. Wert im Akkumulator und Meldesignal N:

MIMA-Assemblercode	**Inhalt des Akkumulator**
`LDC 0x0ABCD`	
`ADD 0x00001`	
`XOR 0x00002`	
`EQL 0x00003`	

Aufgabe 7 *RISC-V Assembler*

1. Verhalten des Codes:

2. Inhalt des Speichers an der Adresse `0x10010000`:

3. Nur gerade Werte von i:

4. Zwischenergebnisse in einem Array speichern:

Aufgabe 8 *Pipelining*

1. Aufgaben der Pipeline-Stufen:

2. Grund:

3. Bedingte Sprünge:

4. *Result-Forwarding und Load-Forwarding*:

Aufgabe 9 *Cache-Speicher*

1. Zugriffsdauer t_m:

2. Anzahl der Sätze:

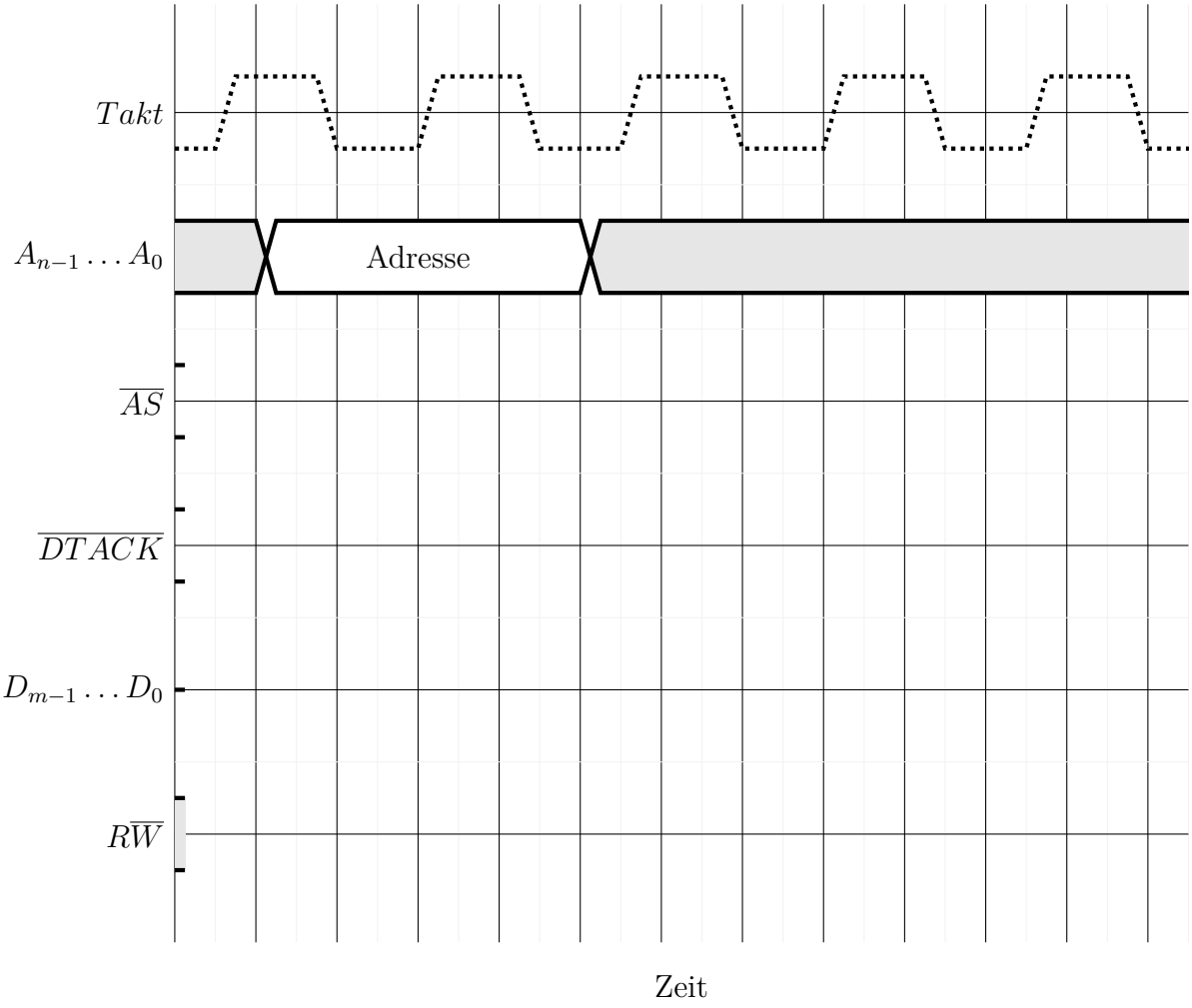
3. Länge des Tags:

4. & 5.

hex. Adresse	**0x00 000**	**0x00 A32**	**0xFA 711**	**0x36 421**	**0xFA 717**	**0x36 729**	**0x5F 321**	**0x00 038**	**0x76 F29**	**0x76 F0A**
read/write	r	r	w	r	w	w	r	w	w	r
hex. Index										
hex. Tag										
Hit/Miss										

Aufgabe 10 *Verbindungsstrukturen*

1. Zeitdiagramm für das Lesen:



2. Mehrere Prozessorchips:

3. *USB* ausgeschrieben: