Aufgabe 1: Multiple Choice

(15 Punkte)

Sind folgende Aussagen zum Thema **Von-Neumann-Architektur** richtig (r) oder falsch (f)?		r	f
a	Ein Grundprinzip des Von-Neumann-Rechners ist ein gemeinsamer Speicher für Daten und Programme.	X	
b	Der klassische Von-Neumann-Rechner besitzt eine MISD (Multiple Instruction – Single Data) Architektur.		X
c	Der Von-Neumann-Flaschenhals bezeichnet die Schnittstelle zwischen Datenprozessor und Befehlsprozessor.		X
d	Der Befehlszyklus einer CPU entsprechend der von-Neumann-Architektur unterscheidet zwischen der Fetch- und der Execution-Phase.	X	
e	Ein von-Neumann-konformer Speicher besteht aus unterschiedlich großen Zellen.		X

Sind folgende Aussagen zum Thema **Fehlererkennung und -korrektur** richtig (r) oder falsch (f)?		r	f
a	Hamming-Codes nutzen Paritätsbits zur Fehlererkennung.	X	
b	Der Hamming-Algorithmus kann nur auf Datenwörter mit einer Länge von $l=2^x$ mit $x \in \mathbb{Z}$ angewendet werden.		X
c	Zur Korrektur von Einzelbitfehlern in einem Codewort mit 16 Datenbits benötigt man 4 Prüfbits.		X
d	Unter dem Hamming-Abstand versteht man den durchschnittlichen Abstand zwischen fehlerhaften Bits in einem Codewort.		X
e	Für die Korrektur von d Einzelbitfehlern benötigt man Codewörter mit mindestens einem Abstand von 2d + 1.	X	

Sind folgende Aussagen zum Thema **Cache** richtig (r) oder falsch (f)?		r	f
a	Der Cache-Speicher ist in der Regel kleiner als der Hauptspeicher.	X	
b	In der Speicherhierarchie dient der Cache als Bindeglied zwischen CPU und Arbeitsspeicher.	X	
c	Der Vorteil des Cache-Speichers ergibt sich aus dem Lokalitätsprinzip.	X	
d	Der Zugriff auf Register ist immer langsamer als ein Zugriff auf den Cache-Speicher.		X
e	Bei der Verwendung eines *Split-Cache* halbiert sich die Datenmenge, die pro Zeiteinheit geliefert wird.		X

Aufgabe 2: Multiplexer

(12 Pkt.)

Bearbeiten Sie folgende Aufgaben:

(2) a. Erläutern Sie in einem Satz die Funktionsweise eines Multiplexers.

Mithilfe von Selektoreingaben kann im Mux aus mehreren Eingaben eine ausgewählt werden, die dann durchgereicht wird.

(1) b. Wie viele Steuerleitungen werden für einen n-Eingaben Multiplexer benötigt?

$\lceil \log_2 n \rceil$

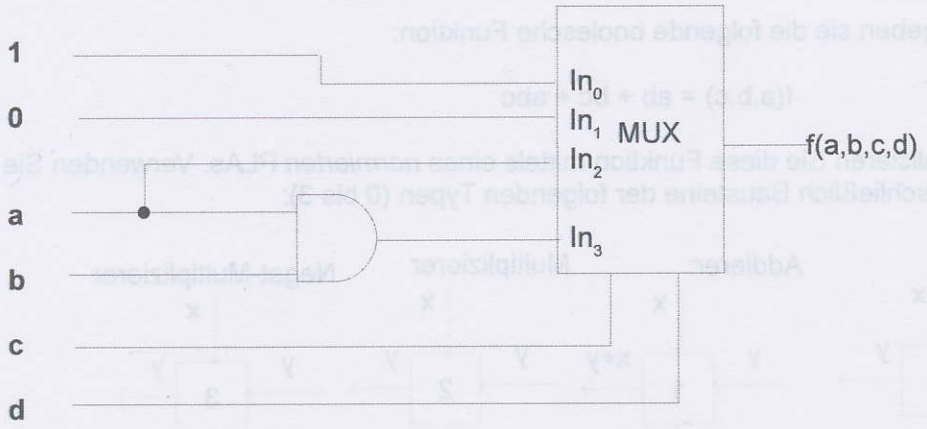
(1) c. Komplexere Multiplexer lassen sich durch das Verschalten mehrerer einfacher Multiplexer realisieren. Wie viele 2-Eingaben-Multiplexer sind notwendig, um einen 4-Eingaben-Multiplexer zu realisieren?

3 2-Eingaben-Multiplexer

(2) d. Stellen Sie die Kurzform der Funktionstabelle eines 4-Eingaben-Multiplexers mit den Eingangsleitungen In_0, In_1, In_2, In_3, den Steuerleitungen S_0, S_1 und der Ausgangsleitung Out auf. Tragen Sie Ihre Lösung in die folgende Tabelle ein:

S_0	S_1	Out
0	0	In_0
0	1	In_1
1	0	In_2
1	1	In_3

(6) e. Gegeben sie folgendes Schaltnetz:



Leiten Sie direkt aus diesem Schaltnetz die Definition der realisierenden Funktion f(a,b,c,d) ab und geben Sie diese in **disjunktiver Form** an.
Der 4-Eingaben-Multiplexer soll dabei gemäß Ihrer aufgestellten Funktionstabelle aus der vorherigen Teilaufgabe d arbeiten!
Hinweis: Überlegen Sie sich zunächst, wie die Wahrheitstabelle für f(a,b,c,d) aussieht.

$f(a,b,c,d) = \bar{c}\bar{d} + c\bar{d}a + cdab$

(c̄d0 fällt weg: da x·0 = 0)

[Relevant ist hier natürlich nur alles, wo 1 am Ende herauskommt]

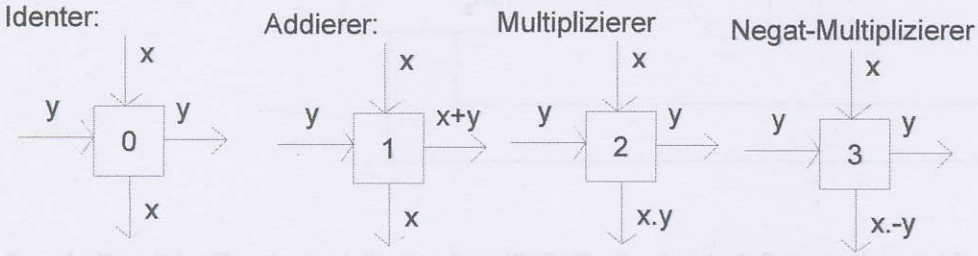
Aufgabe 3: Programmierbare Logische Arrays (PLAs)

(10Punkte)

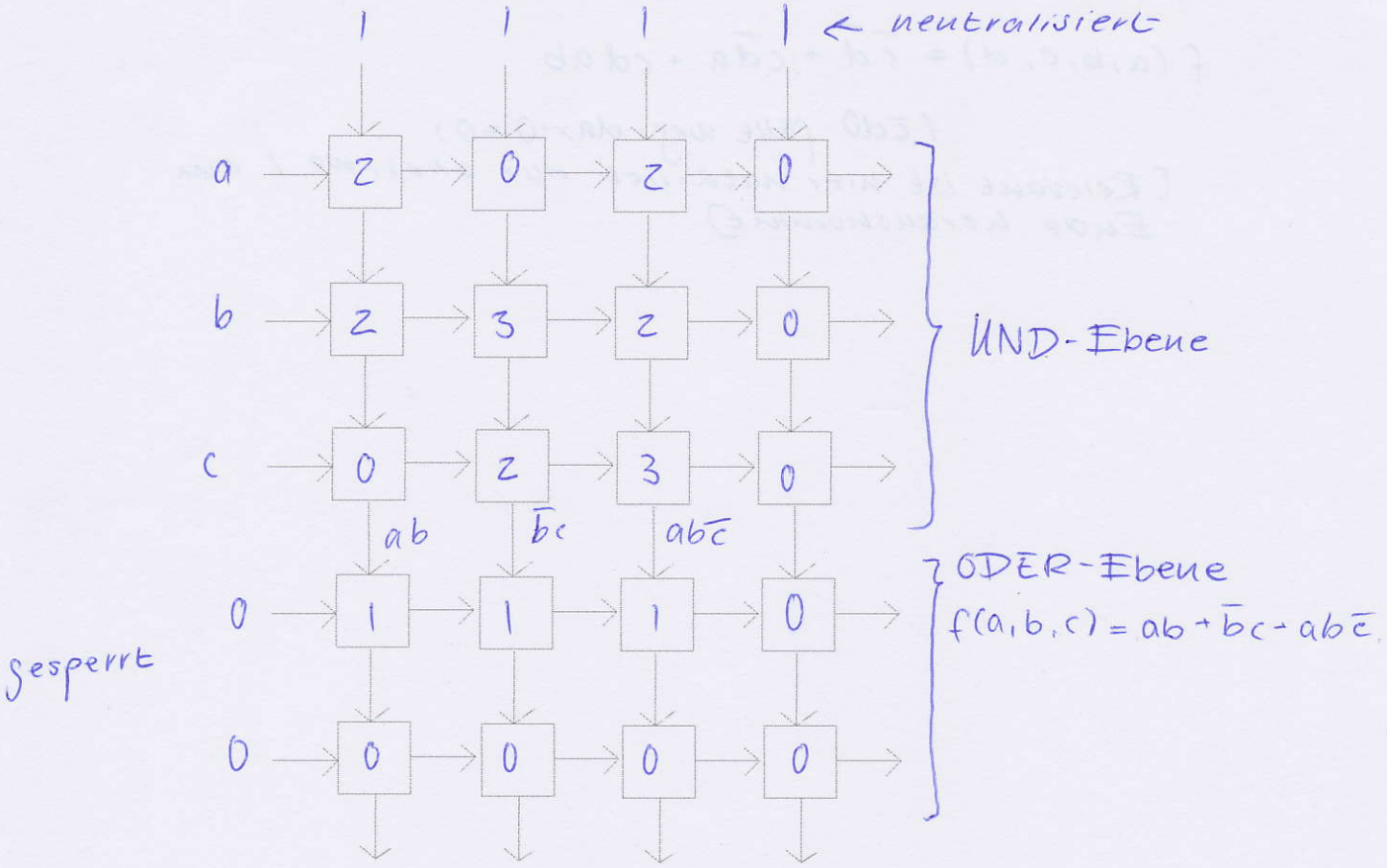
a. Gegeben sie die folgende boolesche Funktion:

$$f(a,b,c) = ab + \bar{b}c + ab\bar{c}$$

Realisieren Sie diese Funktion mittels eines *normierten* PLAs. Verwenden Sie ausschließlich Bausteine der folgenden Typen (0 bis 3):



Tragen Sie dazu jeweils die Typ-Nummer des verwendeten Bausteins in die folgende Vorlage ein. Kennzeichnen Sie zudem die Und- und die Oder-Ebene. Markieren Sie gesperrte und neutralisierte Eingänge. Beschriften Sie eingehende Pfeile mit der jeweils anliegenden logischen Funktion. Beschriften Sie ebenfalls ausgehende Pfeile, an denen das gewünschte Ergebnis anliegt mit der entsprechenden logischen Funktion.



Aufgabe 4: Optimierung von Schaltnetzen

(21 Pkt.)

a. Gegeben sei folgende Wahrheitstabelle einer Funktion $f(x_1,x_2,x_3,x_4)$

	x_1	x_1	x_1	x_1	$f(x_1,x_2,x_3,x_4)$
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	0
14	1	1	1	0	1
15	1	1	1	1	1

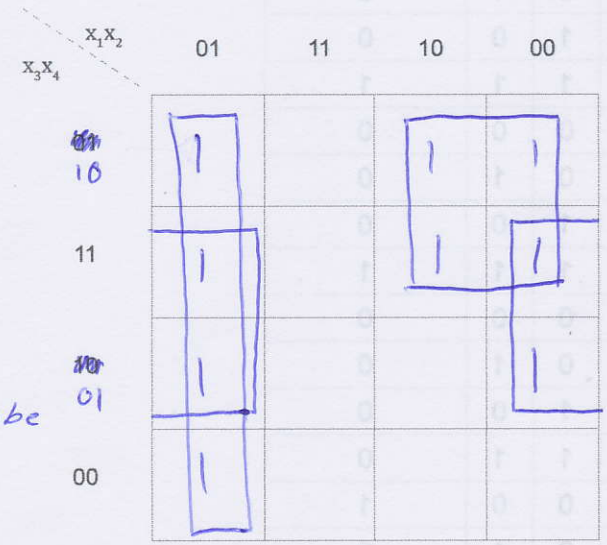
(6) Leiten Sie aus dieser Wahrheitstabelle die Schaltfunktion ihrer vollständigen disjunktiven Normalfunktion (DNF) her.

$$f(x_1,x_2,x_3,x_4) = \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4 + \bar{x}_1\bar{x}_2x_3x_4 + \bar{x}_1x_2x_3x_4 + x_1\bar{x}_2\bar{x}_3\bar{x}_4 + x_1x_2\bar{x}_3\bar{x}_4 + x_1x_2x_3x_4$$

(4) b. Gegeben sei folgende Termdarstellung der Funktion $g(x_1,x_2,x_3,x_4)$:

$$g(x_1,x_2,x_3,x_4) = x_1x_2x_3x_4 + x_1\bar{x}_2x_3x_4 + x_1x_2x_3\bar{x}_4 + \bar{x}_2\bar{x}_3x_4 + \bar{x}_2x_3 + \bar{x}_1\bar{x}_2$$

Minimieren Sie die Funktion g unter Verwendung eines Karnaugh-Diagramms grafisch. Fassen Sie dabei möglichst viele Felder zusammen. Geben Sie abschließend die minimierte Funktion in disjunktiver Form an. Verwenden Sie für Ihre Lösung die folgende Vorlage:



s. Aufgabe

Die minimierte Funktion lautet:

(3) $g(x_1,x_2,x_3,x_4) = \bar{x}_1\bar{x}_2 + x_1x_3 + \bar{x}_2x_4$

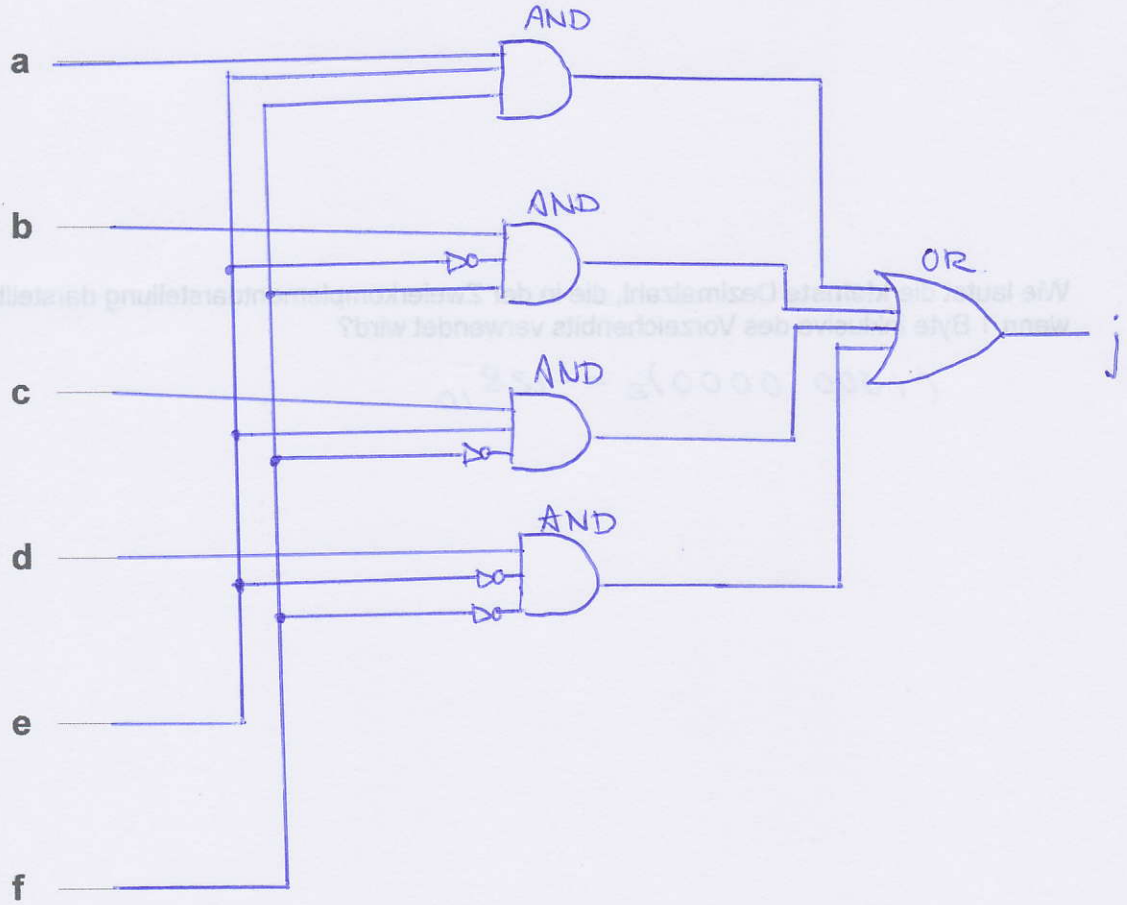
c. Gegeben sei eine minimierte Funktion j(a,b,c,d,e,f) in disjunktiver Form:

$$j(a,b,c,d,e,f) = aef + b\bar{e}f + c\bar{e}f + d\bar{e}f$$

(3)(i) Welchen Ihnen bekannten logischen Baustein realisiert diese Funktion j(a,b,c,d,e,f)? Geben Sie bei Ihrer Antwort eine **genaue Bezeichnung** des logischen Bausteins an und nennen Sie dabei die jeweilige Aufgabe der Inputparameter a – f.

4-Eingaben-Multiplexer mit a,b,c,d als Eingaben und e & f als Steuereingänge

(5)(ii) Entwerfen Sie nun das Schaltbild dieser Funktion j(a,b,c,d,e,f). Verwenden Sie dabei ausschließlich die elementaren Gatter *AND, OR, NOT* und tragen Sie Ihre Lösung in folgende Vorlage ein. Zur Vereinfachung Ihres Schaltbilds dürfen Sie auch gerne OR bzw. AND Gatter mit mehr als 2 Eingängen verwenden.



Aufgabe 5: Zahlendarstellung im Rechner

(20 Pkt.)

Beantworten Sie folgende Fragen im Bezug auf die Dualdarstellung von Ganzzahlen und Gleitkommazahlen:

(2) a. Erläutern Sie in einem Satz den Begriff *Zweierkomplement*.

Das Zweierkomplement ist eine arithmetische Operation (auf ganze Zahlen), durch die negative duale Zahlen aus positiven dualen Zahlen erstellt werden können (z.B. in Zweierkomplementdarstellung)

(1) b. Geben Sie eine Dezimalzahl an, die in 1er, 2er-Komlement und sign/magnitude-Darstellung durch dieselbe Bitfolge repräsentiert wird.

0 (und eigtl. auch alle Positiven, 0 reicht auch)

(2) c. Wie lautet die **kleinste** Dezimalzahl, die in der Zweierkomplementdarstellung darstellbar ist, wenn 1 Byte inklusive des Vorzeichenbits verwendet wird?

$(1000\ 0000)_2 = -128_{10}$

(2) d. Geben Sie die Zweierkomplementdarstellung der folgenden Dezimalzahl an. Verwenden Sie zur Darstellung 1 Byte (inklusive des Vorzeichenbits):
y = -94

94 01011110
10100001
+ 00000001
10100010 ← -94 = y

(2) e. Gegeben seien die beiden Binärzahlen u = 10011001 und v = 10110011 in ihrer Zweierkomplementdarstellung. Kann bei der **Subtraktion** der beiden Zahlen ein Überlauf stattfinden?
Begründen Sie Ihre Antwort **ohne** Berechnung!

Weil beide Zahlen negativ sind, rechnet man u-v als u+(-v) (positive Variante von v). Es wird also zu einer negativen Zahl eine Positive addiert. Daher kommt es nicht zum Überlauf.
(Für einen Überlauf müssten 1er an erster Stelle addiert werden, dazu kann es hier aber nicht kommen)

(3) f. **Addieren** Sie nun die beiden Zahlen u und v aus der vorherigen Teilaufgabe e! Der Rechenweg **muss** ersichtlich und nachvollziehbar sein!

100 11001
+ 10110011
1 11 11 Übertrag
→ 01001100

(2) g. Hat bei Ihrer Addition in der vorherigen Teilaufgabe f ein Überlauf (Overflow) stattgefunden? Begründen Sie kurz Ihre Antwort.

Ja, da die ursprünglichen 8 Bit zur Darstellung der Lösung nicht ausreichen
(-182 ist mit 8 Bit nicht darstellbar)

(3) h. Gegeben sei nun die Dezimalzahl x = -9.125.
Wandeln Sie diese Dezimalzahl in ihre Gleitkommadarstellung nach IEEE 754 mit einfacher Genauigkeit (32-Bit) um und tragen Sie Ihr Ergebnis in die entsprechende Vorlage ein! Das niederwertigste Bit des Exponenten befindet sich dabei auf Platz 23, das niederwertigste Bit des Signifikanden befindet sich auf Platz 0.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S	Exponent								Signifikand																						

$9{,}125 = 01001{,}001 = 1{,}00100100\ldots \cdot 2^3$

$130_{10} = 10000010_2$

(3) i. Wandeln Sie folgende Zahl, die in 32-Bit-Gleitkommadarstellung nach IEEE 754 gegeben ist, in ihre Dezimaldarstellung um! Das niederwertigste Bit des Exponenten befindet sich dabei auf Platz 23, das niederwertigste Bit des Signifikanden befindet sich auf Platz 0.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	0	1	1	1	0	1	1	1	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
S	Exponent								Signifikand																						

+ ist hier Konkatenation

$(-1)^1_2 \cdot (1 + 01110110\,11000\ldots)_2^{135-127_{10}}$

$= -_{10} \cdot (1 + 01110110\,1100\ldots)_2^{8_{10}}$

$= -_{10} \cdot (10111011{,}0\,1100\ldots)_2$

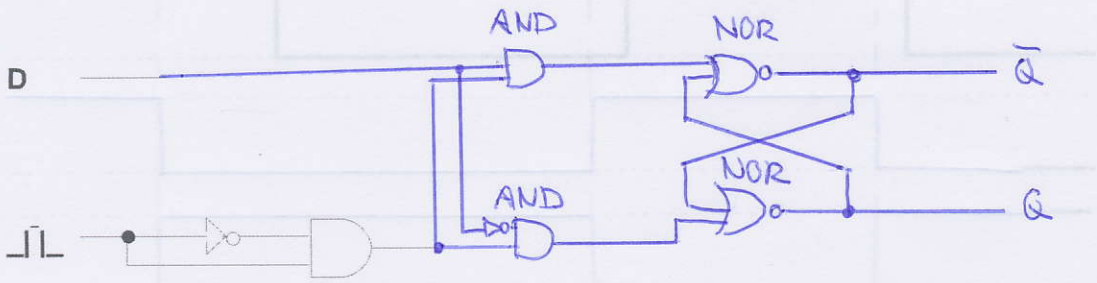
$= -374{,}75_{10}$

Aufgabe 6: Flip-Flop-Schaltung

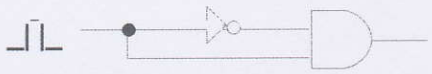
(10 Pkt.)

Bearbeiten Sie folgende Aufgaben:

(4) a. Skizzieren Sie das Schaltwerk eines D-Flip-Flops! Benutzen Sie dazu nur die elementaren Gatter vom Typ AND, NOT und NOR und tragen Sie Ihre Lösung in die folgende Vorlage ein:



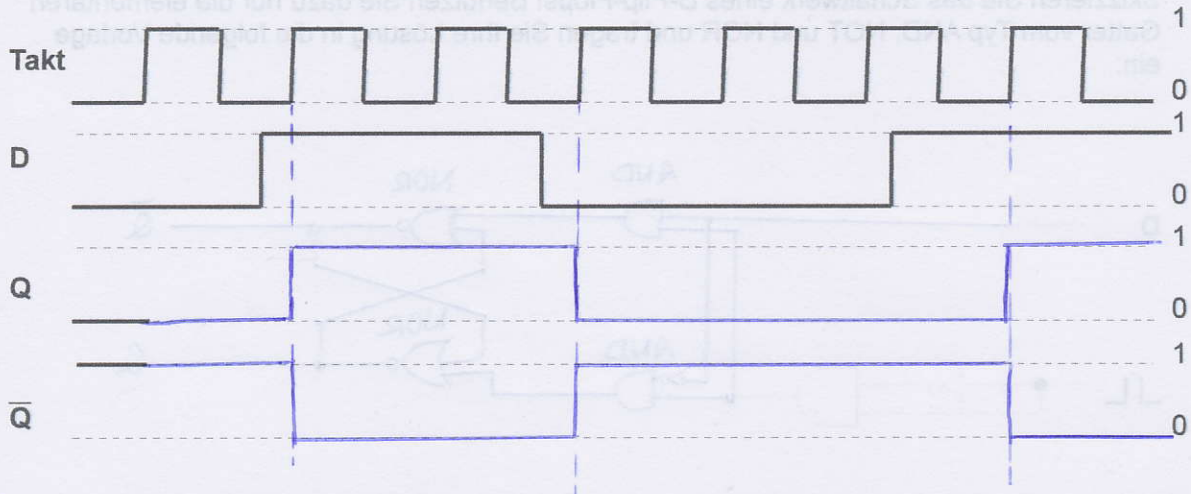
(1) b. Gehen Sie von einem D-Flip-Flop mit folgendem Taktgeber aus:



Realisiert dieser Taktgeber den Zustandsübergang bei steigender oder fallender Flanke?

Bei steigender Flanke.

(4) c. Gegeben sei das nachfolgende Impulsdiagramm eines D-Flip-Flops mit dem Taktgeber aus der vorherigen Teilaufgabe b. Vervollständigen Sie das folgende Impulsdiagramm für die Ausgänge Q und $\overline{Q}$ unter der Annahme, dass der Baustein ohne Zeitverzögerung schaltet:



(1) d. Welches Problem ergibt sich beim D-Flip-Flop im Hinblick auf das Speichern über mehrere Takte hinweg?

Es ist nicht möglich.

(Begründung falls die jemand braucht:
mit D kann man bestimmen, welchen Wert Q hat. Wir lassen einen Takt verstreichen. Wenn D den gleichen Wert wie zuvor hat, hat Q den gleichen Wert wie zuvor. Aber dazu müssen wir wissen, was D bzw. Q war.
Wir können zwar zufällig das gleiche anlegen, aber wenn wir das anders wählen, ändert sich auch Q.
Um Q nachzuschauen, müssten wir wissen, was Q war ☺)

Aufgabe 7: Fehlererkennung und -Korrektur

(8 Pkt.)

Als Schutz vor Speicherfehlern werden u.a. Codes zur Fehlererkennung und zur Fehlerkorrektur eingesetzt. Bearbeiten Sie dazu folgende Aufgaben:

(4) a. Verwenden Sie den Hamming Algorithmus und kodieren Sie das folgende 8-Bit Daten-Wort in einem 12-Bit Hamming Code:

(i) 0110 0011

Verwenden Sie dazu **gerade Parität** und kennzeichnen Sie ihr Ihrem resultierenden Codewort alle eingesetzten Paritätsbits!

1 2 4 8
0 0 0 1 1 1 0 0 0 1 1

PB1 0 0 1 0 0 1 → 2 → 0

PB2 0 0 1 0 0 1 → 2 → 0

PB4 1 1 1 0 1 → 3 → 1

PB 8 0 0 0 1 1 → 2 → 0

⇒ 0001 1100 0011

(4) b. Gegeben sei das folgende 12-Bit Codewort, welches nach dem Hamming-Alrgorithmus enkodiert ist:

1101 0110 0100

Bearbeiten Sie davon ausgehend die folgenden Schritte:

(i) Verwenden Sie wieder gerade Parität und geben Sie alle fehlerhaften Paritätsbits an.

	1	1	0	1	0	1	1	0	0	1	0	0
PB1	1		0		0		1		0		0	→ 2 ✓
PB2	1	0			1	1			1	0		→ 4 ✓
PB4			1	0	1	1					0	→ 3 ✗
PB8							0	0	1	0	0	→ 1 ✗

→ Paritätsbit 4 und 8 sind fehlerhaft und weisen auf Fehler hin

(ii) Identifizieren Sie (falls möglich) das fehlerhafte Datenbit.

4 + 8 = 12
→ Das 12. Datenbit ist falsch.

(iii) Korrigieren Sie (falls möglich) den Fehler und geben Sie ein (ggf. korrigiertes) Codewort an.

1101 0110 0101

(iv) Geben Sie das resultierende 8-Bit Datenwort an.

0011 0101

Aufgabe 8: Assemblerprogrammierung unter SPIM I

(12 Pkt.)

Beantworten Sie die folgenden Fragen zum Thema Assemblerprogrammierung des MIPS-Prozessors. **Hinweis:** Eine Übersicht zu den wichtigsten SPIM-Befehlen finden Sie am Ende des Klausurhefts.

(5) a. Gegeben sei folgendes Programm in SPIM, in dem einige Kommentare eingefügt sind:

```
1  .data
2  input:    .asciiz „Geben Sie eine Zahl ein:"
3  .text
4  main:
5       la   $a0, input
6       li   $v0, 4
7       syscall              # Kommentar 1: (v)
8
9       li   $v0, 5
10      syscall              # Zahl einlesen
11
12      bltz $v0, error      # Kommentar 2: (viii)
13
14      move $t0, $v0        # Kommentar 3: (vi)
15
16      li   $t1, 1          # Kommentar 4: (i)
17
18 while:
19      beqz $t0, output     # Kommentar 5: (ix)
20
21      mul  $t1, $t1, $t0   # Kommentar 6: (x)
22
23      sub  $t0, $t0, 1     # Kommentar 7: (vii)
24
25      j    while           # Kommentar 8: (iii)
26
27 output:
28      li   $v0, 1          # Zahl ausgeben
29      move $a0, $t1        # Kommentar 9: (ii)
30      syscall
31
32      li   $v0, 10         #Beenden
33      syscall
34
35 error:
36      li   $t1, -2         # Kommentar 10: (iv)
37      j    output          # Sprung zu Marke
```

Ordnen Sie in diesem Programm jeder Zeile, die mit „# Kommentar <Nr>:" versehen ist, den jeweils genau passenden der folgenden Kommentare zu. Tragen Sie dazu die Nummer des richtigen Kommentars aus der folgenden Liste in den obigen Coderahmen hinter der betreffenden Kommentarzeile ein!

Nr.	Kommentar
(i)	Ergebnis initialisieren
(ii)	Übergabe Ergebnis für Ausgabe
(iii)	Schleife wiederholen
(iv)	Speichere -2 als Ergebnis
(v)	String ausgeben

Nr.	Kommentar
(vi)	Nutzereingabe sichern
(vii)	Eingabe := Eingabe -1
(viii)	Sprung, falls Eingabe < 0
(ix)	while Eingabe > 0
(x)	Ergebnis := Ergebnis * Eingabe

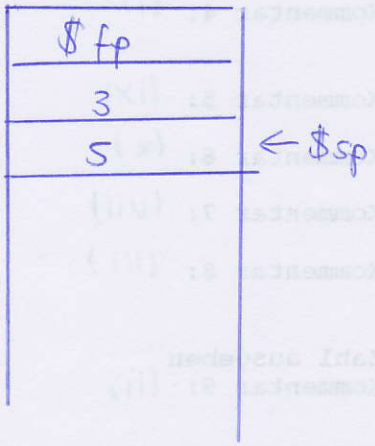
(2) b. Beschreiben Sie kurz, was das oben angegebene Programm aus Teilaufgabe a bewirkt!

Der Algorithmus berechnet die Fakultät einer Eingabe.

(4) c. Unabhängig von Teilaufgabe a) sei nun die folgende Befehlssequenz gegeben:

```
1    li   $t0, 3
2    li   $t1, 5
3    addi $sp, $Sp, -12
4    sw   $fp, 12($sp)
5    sw   $ra, 8 ($sp)
6    sw   $t0, 8 ($sp)
7    sw   $t1, 4 ($sp)
```

Zeichnen Sie den Stack, wie er nach Ausführung dieser Befehlssequenz aussieht. Tragen Sie auch den Stackpointer in Ihre Skizze ein.



(1) d. Welches Problem liegt nach der Abarbeitung der Befehlssequenz aus der vorherigen Teilaufgabe c vor?

Die Rücksprungadresse wird sofort wieder überschrieben.

100% keine Garantie

```
1   .data
2   input_a:    asciiz „Seitenlaenge a:"
3   input_b:    asciiz „Seitenlaenge b:"
4   eingabe:    .word 0 0
5   ausgabe:    .word 0
6
7   .text
8   main:
9         li    $v0, 4
10        la    $a0, input_a        # print input_a String
11        syscall
12
13        li    $v0, 5              # read input a
14        syscall
15        sw    $v0, eingabe        # Eingabe [0]:=a
16
17        li    $v0, 4
18        la    $a0, input_b        # print input_b String
19        syscall
20
21        li    $v0, 5              # read input b
22        syscall
23        sw    $v0, eingabe+4      # Eingabe [1]:=b
24
25        la    $a0, eingabe        # Pass address of eingabe to Argument_1
26        la    $a1, ausgabe        # Pass address of ausgabe to Argument_2
27        jal   triangle            # Jump to label triangle
28        jal output          # Jump to label output
29
30  triangle:
31  #Ihre Lösung:
32        lw $t0, ($a0)     # speichere input a in $t0
33        lw $t1, ($a0)+4   # speichere input b in $t1
34
35
36
37        mul   $t0, $t0, $t0       # $t0 = a^2
38        mul   $t1,  $t1, $t1      # $t1 = b^2
39
40        add   $t2, $t0, $t1       # $t2 = a^2 + b^2
41
42  #Ihre Lösung:
43        sw $t2, ausgabe)  # Speichere Ergebnis in ausgabe
44
45        jr    $ra                 # Jump back to return address
46
47  output:
48  #Ihre Lösung:
49
50        lw $a0, ausgabe   # Ergebnis in $a0 bereit fuer print
51
52        li    $v0, 1              # print int
53        syscall
54
55        li    $v0, 10             # exit.
56        syscall
57
58
```

* sw $t2, ausgabe

Aufgabe 9b: Anwendungen der Digitalisierung

(12 Pkt.)

Nennen Sie 4 Branchen, bei denen sich Ausprägungen der Digitalisierung beobachten lassen und geben Sie dazu jeweils 2 Beispiele an. Benutzen Sie für Ihre Antwort die folgende Vorlage:

Branche I: Flughafen

Beispiel 1: Informationen zum Flug jederzeit abrufbar + Infos zu Angeboten (über Smartphone etc)

Beispiel 2: Urlaub flexibler planen + kurzfristiger

Branche II: Versicherungs

Beispiel 1: Kunde kann leicht über Internet informieren, vergleichen von Versicherungen

Beispiel 2: Fahrstil von Kunden über Autos analysiert → passende Versicherung

Branche III: Handel

Beispiel 1: Werbung individuell an Kunden angepasst (durch gesammelte Infos)

Beispiel 2: Omni-Channel-Shopping → Wechsel zw. Online & Offline

Branche IV: Automobilindustrie

Beispiel 1: Autos vernetzt → bessere Reparaturen schnell erkannt

Beispiel 2: CarSharing: Rückgaben, Ausleihen einfacher und außerhalb Öffnungszeiten möglich

Viel Erfolg bei den Klausuren!

N. B.