

Rechnerarchitektur Tests Wintersemester 2021/22

Test 1:

Was sind korrekte Darstellungen für die Dezimalzahl 76?

- ☐ a. $(76)_2$
- ☒ b. $0x4c$
- ☒ c. $(00001001100)_2$
- ☐ d. $(4B)_{16}$

Welche Zahlen sind equivalent zu $(75)_8$?

- ☒ a. $0x3D$
- ☐ b. $(100)_{16}$
- ☒ c. $(111101)_2$
- ☐ d. $(3c)_{16}$

Welche Zahlen können im Stellenwertsystem zur Basis 2 mit 8 Bit dargestellt werden?

- ☐ a. 2^8
- ☒ b. 255
- ☒ c. 0
- ☒ d. 102
- ☐ e. 656

Welche der folgenden Aussagen geben Moore's Hauptaussage in dem Artikel "Cramming more components onto integrated circuits" korrekt wieder?

- ☐ a. Die Taktrate von Prozessoren wächst exponentiell.
- ☒ b. Die kostenoptimale Anzahl an Transistoren pro Die wächst exponentiell.
- ☐ c. Es ist möglich, immer schnellere GPUs zu bauen.
- ☒ d. Die Anzahl der Transistoren in CPUs wächst exponentiell.

Wie viele n -stellige Boolesche Funktionen gibt es?

- ☐ a. $2^{(n^n)}$
- ☐ b. $2^{(2n)}$
- ☐ c. unendlich viele
- ☒ d. $2^{(2^n)}$

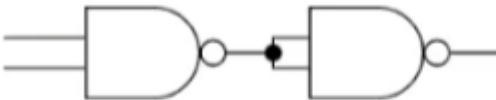

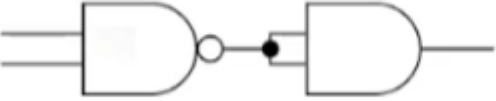
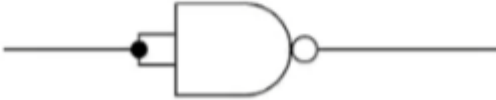
Test 2:

Welche Boolesche Funktion ist durch die folgende Wahrheitstabelle dargestellt?

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

- ☐ a. $(x_1 \wedge x_2) \vee x_1$
- ☐ b. $x_1 \wedge \overline{x_2}$
- ☐ c. $(\overline{x_1} \vee x_2) \wedge (x_1 \vee \overline{x_2})$
- ☒ d. $(\overline{x_1} \wedge x_2) \vee (x_1 \wedge \overline{x_2})$

Welche Schaltung implementiert den AND-Operator mit NAND-Gattern?

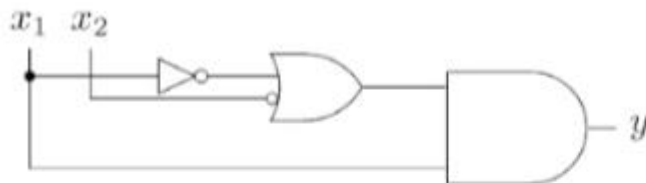
- ☒ a. 
- ☐ b. 
- ☐ c. 
- ☐ d. 

Welche Beschreibung passt zur dargestellten Funktion $f(x_1, x_2, x_3)$?

x_1	x_2	x_3	y
0	0	0	0
1	0	0	1
0	1	0	1
1	1	0	0
0	0	1	1
1	0	1	0
0	1	1	0
1	1	1	0

- ☐ a. Die Funktion gibt 1 aus, wenn der Mond aus grünem Käse ist.
- ☐ b. Die Funktion gibt 1 aus, wenn x_1 den gleichen Wert hat wie x_3 .
- ☒ c. Die Funktion gibt 1 aus, wenn genau einer der drei Eingänge x_1, x_2, x_3 den Wert 1 annimmt.
- ☐ d. Die Funktion gibt 1 aus, wenn die Zahl $x_3 \cdot 2^2 + x_2 \cdot 2^1 + x_1 \cdot 2^0$ ungerade ist.

Betrachten Sie diese Schaltung.



Welche der folgenden Booleschen Funktionen passt zu der Schaltung?

- ☐ a. $(\overline{x_1} \vee x_2) \wedge \overline{x_1}$
- ☐ b. $\overline{x_1} \wedge x_2 \vee x_1 \wedge \overline{x_2}$
- ☐ c. $\overline{(\overline{x_1} \vee x_2) \wedge x_1}$
- ☒ d. $(\overline{x_1} \vee \overline{x_2}) \wedge x_1$

Welche der folgenden Zahlen sind äquivalent zu $(184)_{10}$?

Tipp: Eine der Binärzahlen ist richtig.

☐ a. $(271)_8$

☒ b. $(b8)_{16}$

☐ c. $(228)_{16}$

☐ d. $(10110111)_2$

☒ e. $(10111000)_2$

Test 3:

Gatter und Wahrheitstabelle



Welche der folgenden Wahrheitstabellen gehört zu obigem Gatter?

☒ a.

x_1	x_2	y
0	0	1
0	1	1
1	0	1
1	1	0

☐ b.

x_1	x_2	y
0	0	1
0	1	0
1	0	0
1	1	0

☐ c.

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

☐ d.

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

Welches KV-Diagramm gehört zu folgender Wahrheitstabelle?

x_1	x_2	x_3	y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

a.

	x_2		
	1	1	0
x_3	1	0	0
	x_1		

☐

b.

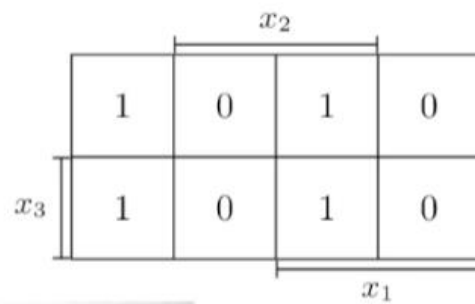
	x_2		
	0	0	1
x_3	0	1	1
	x_1		

☐

c.

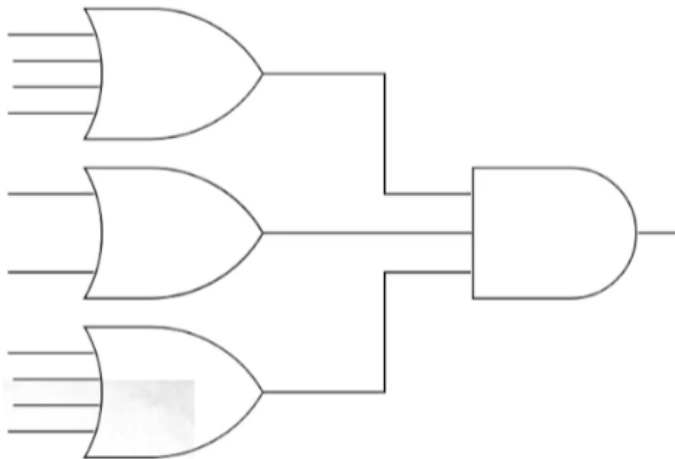
	x_2		
	1	1	0
x_3	1	0	0
	x_1		

Markieren Sie die minimale Funktion, welche aus dem folgenden KV-Diagramm ausgelesen werden kann.



- ☒ a. $y = (x_1 \wedge x_2) \vee (\overline{x_1} \wedge \overline{x_2})$
- ☐ b. $y = (x_2 \wedge \overline{x_3}) \vee (\overline{x_2} \wedge x_3)$
- ☐ c. $y = (\overline{x_1} \wedge x_2) \vee (x_1 \wedge \overline{x_2})$
- ☐ d. $y = (\overline{x_2} \wedge \overline{x_3}) \vee (x_2 \wedge x_3)$

Sie haben für die Realisierung der gegebenen Schaltung nur Standardbauteile mit jeweils zwei Eingängen pro Gatter zur Verfügung. Wie viele dieser Standardgatter benötigen Sie, um die Schaltung zu realisieren? Was ist die minimale Anzahl an Ebenen, wenn Sie die Schaltung mit Standardbauteilen realisieren?



- ☒ a. 9 Standardgatter
- ☐ b. 7 Standardgatter
- ☐ c. 8 Standardgatter
- ☐ d. 3 Ebenen
- ☒ e. 4 Ebenen

Welche Aussagen sind korrekt?

- ☒ a. Alle Booleschen Funktionen können mithilfe der Negation und der Konjunktion dargestellt werden.
- ☐ b. Jede Boolesche Funktion benötigt mindestens ein NOT.
- ☐ c. Alle Booleschen Funktionen können mithilfe der Konjunktion und der Disjunktion dargestellt werden.
- ☒ d. Alle Booleschen Funktionen können mithilfe der NOR-Verknüpfung dargestellt werden.
- ☐ e. Alle Booleschen Funktionen können mithilfe der XOR-Verknüpfung dargestellt werden.

Test 4:

Sie wollen das folgende Gatter mit *Standardgattern* mit jeweils zwei Eingängen realisieren. Wie viele Gatter sind mindestens nötig? Was ist die minimale Anzahl der Ebenen in der resultierenden Schaltung?



- ☐ a. 8 Standardgatter
- ☒ b. 4 Standardgatter
- ☐ c. 3 Standardgatter
- ☐ d. 4 Ebenen
- ☒ e. 3 Ebenen
- ☐ f. 2 Ebenen

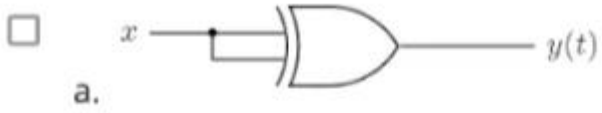
Was ist die Negation der Booleschen Funktion $y(x_1, x_2, x_3, x_4) = (\overline{x_1} \cdot x_2 \cdot x_4) + \overline{x_2} + x_4$?

- ☒ a. $(x_1 + \overline{x_2} + \overline{x_4}) \cdot x_2 \cdot \overline{x_4}$
- ☐ b. $(x_1 + \overline{x_2} + \overline{x_4}) \cdot \overline{x_2} \cdot \overline{x_4}$
- ☐ c. $(x_1 \cdot x_2 \cdot x_4) + \overline{x_2} + x_4$
- ☐ d. $(\overline{x_1} \cdot x_2 \cdot \overline{x_4}) + x_2 + \overline{x_4}$

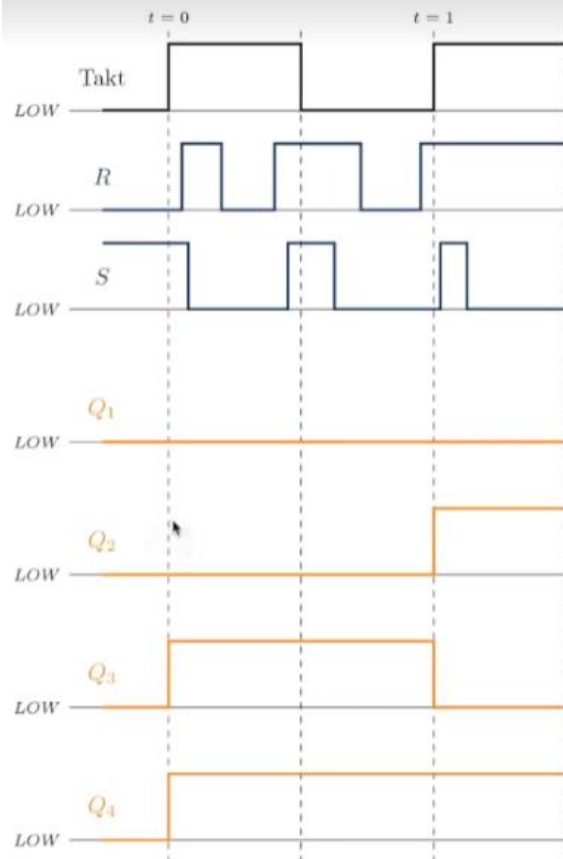
Welche Aussagen sind korrekt?

- ☒ a. Alle Booleschen Funktionen können mithilfe der NOR-Verknüpfung dargestellt werden.
- ☐ b. Alle Booleschen Funktionen können mithilfe der Konjunktion und der Disjunktion dargestellt werden.
- ☒ c. Alle Booleschen Funktionen können mithilfe der Negation und der Konjunktion dargestellt werden.
- ☐ d. Jede Boolesche Funktion benötigt mindestens ein NOT.
- ☐ e. Alle Booleschen Funktionen können mithilfe der XOR-Verknüpfung dargestellt werden.

Welche der Schaltungen können unkontrolliert oszillieren?



Gegeben sind Takt, R und S im Zeitverlauf. Welches ist das dazugehörige Q eines positiv flankengesteuerten RS-Flipflops? (Die Schaltzeit τ sei vernachlässigbar klein.)



☐ a. Q_1

☐ b. Q_2

☒ c. Q_3

☐ d. Q_4

Test 5:

Markieren Sie die minimale Funktion, welche aus dem folgenden KV-Diagramm ausgelesen werden kann.

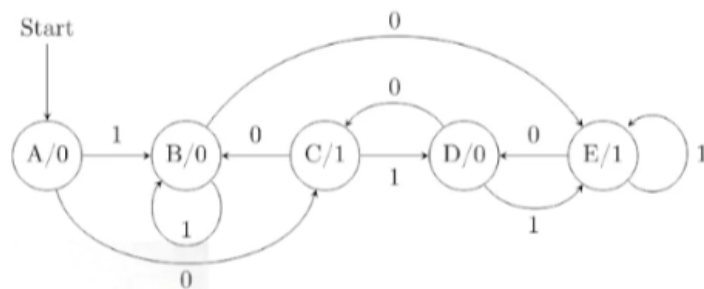
	x_2		
	1	1	0
x_3	1	0	0
		x_1	

- ☐ a. $y = \overline{x_1} \cdot \overline{x_2} + \overline{x_1} \cdot x_3 + \overline{x_2} \cdot x_3$
- ☐ b. $y = x_1 \cdot \overline{x_2} + x_1 \cdot \overline{x_3} + \overline{x_2} \cdot \overline{x_3}$
- ☒ c. $y = \overline{x_1} \cdot \overline{x_2} + \overline{x_1} \cdot \overline{x_3} + \overline{x_2} \cdot \overline{x_3}$

Welche Aussagen über synchrone und asynchrone Schaltwerke treffen zu?

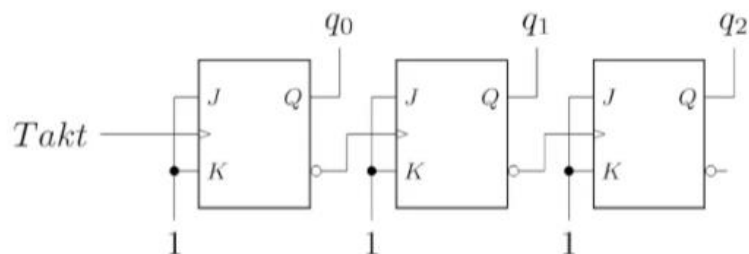
- ☒ a. Asynchrone Implementierungen derselben Funktion sind oft schneller als ein synchrones Äquivalent.
- ☐ b. Asynchrone Schaltwerke werden durch ein zentrales Taktsignal gesteuert.
- ☐ c. In synchronen Schaltwerken ist der Zeitpunkt stabiler Ausgangssignale oft nicht genau bestimmbar.
- ☒ d. Synchrone Schaltwerke können leicht systematisch entworfen werden.

In welchem Zustand befindet sich der gegebene **Moore**-Automat nach der kompletten Verarbeitung der Eingabefolge 1111000? Gehen Sie davon aus, dass Sie sich am Anfang im Startknoten A befinden.



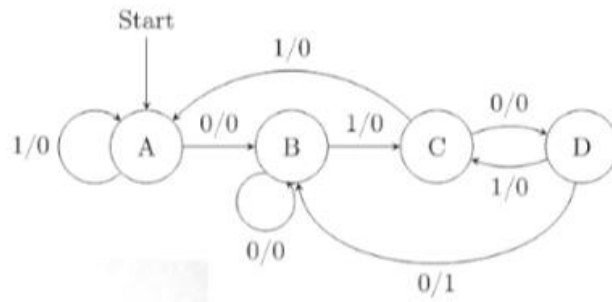
- ☐ a. A
- ☐ b. B
- ☒ c. C
- ☐ d. D
- ☐ e. E

Das folgende Schaltnetz besteht aus positiv-flankengesteuerten Flipflops. Wir interpretieren die Ausgänge $q = (q_2, q_1, q_0)$ als positive Ganzzahl in der Binärdarstellung. Welche Aussagen über die Schaltung treffen zu?



- ☐ a. Es handelt sich um ein synchrones Schaltwerk.
- ☐ b. Der Ausgang q_1 ändert seinen Wert mit jeder positiven Taktflanke.
- ☒ c. Der Ausgang q_2 ändert seinen Wert mit jeder vierten positiven Taktflanke.
- ☒ d. Die Zahl q wird abgesehen vom Überlauf mit jedem Takt um eins vergrößert.

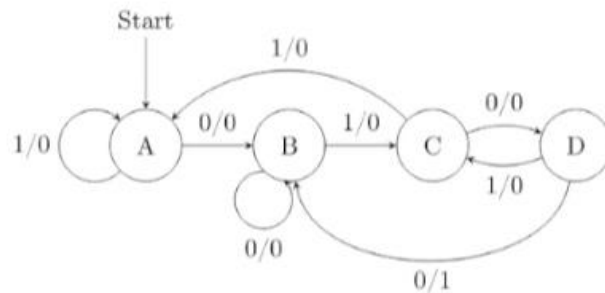
Welche Funktionalität erfüllt der folgende **Mealy**-Automat?



- ☐ a. Er gibt immer genau dann eine 1 aus, wenn die letzten eingegebenen Zeichen 0110 waren.
- ☐ b. Er gibt mit jedem Eingabebit abwechselnd 0en und 1en aus.
- ☒ c. Er gibt immer genau dann eine 1 aus, wenn die letzten eingegebenen Zeichen 0100 waren.
- ☐ d. Er gibt nur 1en aus.

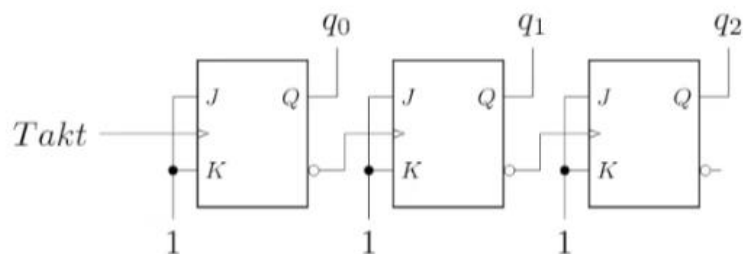
Test 6:

Welche Funktionalität erfüllt der folgende **Mealy**-Automat?



- ☐ a. Er gibt immer genau dann eine 1 aus, wenn die letzten eingegebenen Zeichen 0110 waren.
- ☐ b. Er gibt mit jedem Eingabebit abwechselnd 0en und 1en aus.
- ☒ c. Er gibt immer genau dann eine 1 aus, wenn die letzten eingegebenen Zeichen 0100 waren.
- ☐ d. Er gibt nur 0en aus.

Das folgende Schaltnetz besteht aus positiv-flankengesteuerten Flipflops. Wir interpretieren die Ausgänge $q = (q_2, q_1, q_0)$ als positive Ganzzahl in der Binärdarstellung. Welche Aussagen über die Schaltung treffen zu?

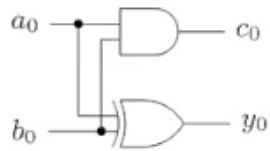


- ☒ a. Der Ausgang q_0 ändert seinen Wert mit jeder positiven Taktflanke.
- ☒ b. Die Zahl q wird abgesehen vom Überlauf mit jedem Takt um eins vergrößert.
- ☐ c. Es handelt sich um ein synchrones Schaltwerk.
- ☐ d. Der Ausgang q_2 ändert seinen Wert mit jeder zweiten positiven Taktflanke.

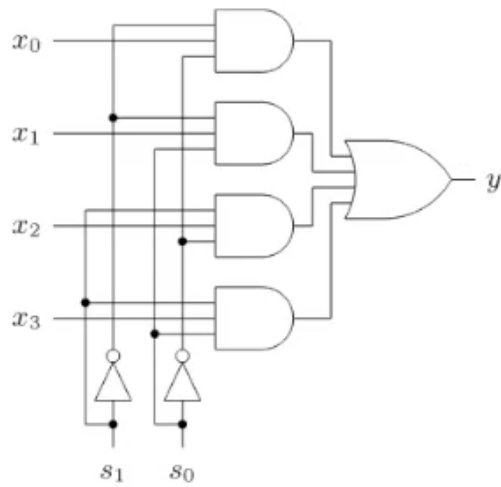
Welche dieser Schaltungen realisiert einen Halbaddierer?



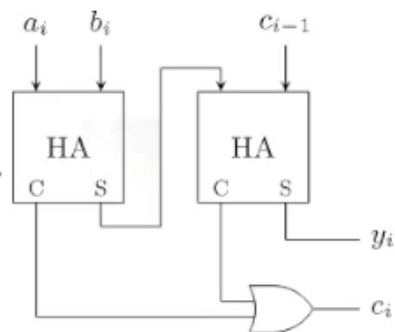
a.



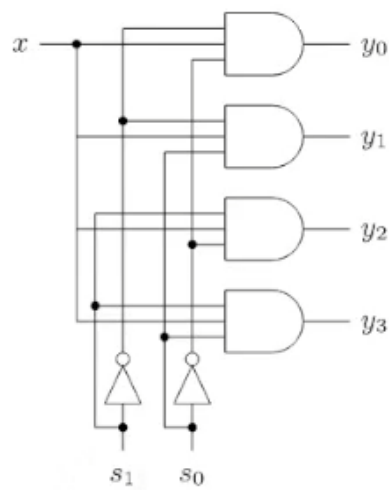
b.



c.



d.



Welches ist die Negation der **Binärzahl** $(001011)_2$ im **Zweierkomplement**?

- ☐ a. $(001101)_2$
- ☐ b. $(110100)_2$
- ☒ c. $(110101)_2$
- ☐ d. $(110111)_2$

Führen Sie die Addition der **Binärzahlen** $(1000011)_2$ und $(0100111)_2$ aus. Zählen Sie die Anzahl der Überträge.

Welche Aussage ist korrekt?

- ☒ a. Es kommt zu 3 Überträgen.
- ☐ b. Es kommt zu 7 Überträgen.
- ☐ c. Es kommt zu 1 Überträgen.
- ☐ d. Es kommt zu 5 Überträgen.

Test 7:

Welches ist die Negation der **Binärzahl** $(110010)_2$ im **Zweierkomplement**?

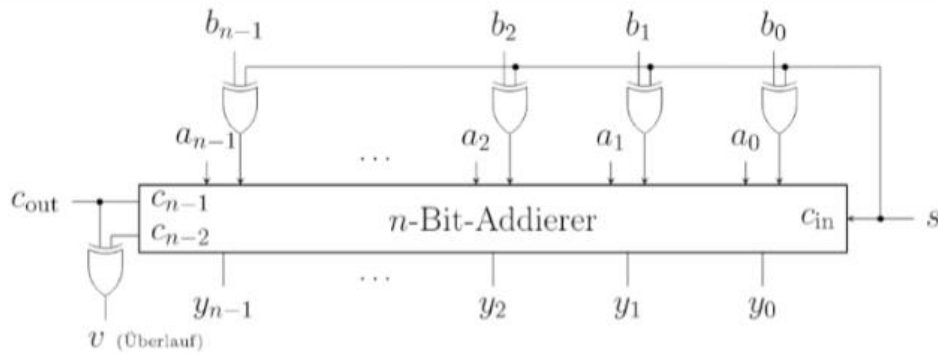
- ☐ a. $(010000)_2$
- ☐ b. $(110100)_2$
- ☐ c. $(001101)_2$
- ☒ d. $(001110)_2$

Welche gegebene Bitfolge ist die **Festkommadarstellung** der Zahl 3.625 mit $k = 3$?

- ☐ a. 0011011
- ☒ b. 0011101
- ☐ c. 0110101
- ☐ d. 1010101
- ☐ e. 1100011

Welche Multiplikationsaufgabe $y = a \cdot b$ ließe sich bei einer seriellen Multiplikation am meisten durch den Einsatz von Booth' Idee beschleunigen? Gegeben sei $a = (00110100)$.

- ☐ a. $b = 01010100$
- ☒ b. $b = 01111110$
- ☐ c. $b = 01010101$
- ☐ d. $b = 01110100$



Es seien $n = 4$, $a = (0011)_2$, und $b = (0001)_2$. Nutzen Sie das bekannte Addier- und Subtrahierwerk um $a + b$ zu berechnen. Wie ist die Leitung s zu belegen? Was liegt danach an den Ausgabeleitungen an?

- ☐ a. $s = 1, c_{out} = 0, v = 1$
- ☒ b. $s = 0, c_{out} = 1, v = 1$
- ☐ c. $s = 1, c_{out} = 1, v = 0$
- ☐ d. $s = 0, c_{out} = 0, v = 0$

Gegeben sei $p = 5$ und $m = 3$. Was ist die Dezimaldarstellung von $z = (1, 0, 1, 1, 1, 1, 0, 0, 1)$? Gehen Sie davon aus, dass z nach dem in der Vorlesung besprochenen **IEEE-754-Standard** gebildet wurde.

Erinnern Sie sich dabei an die Binärdarstellung von Gleitkommazahlen:

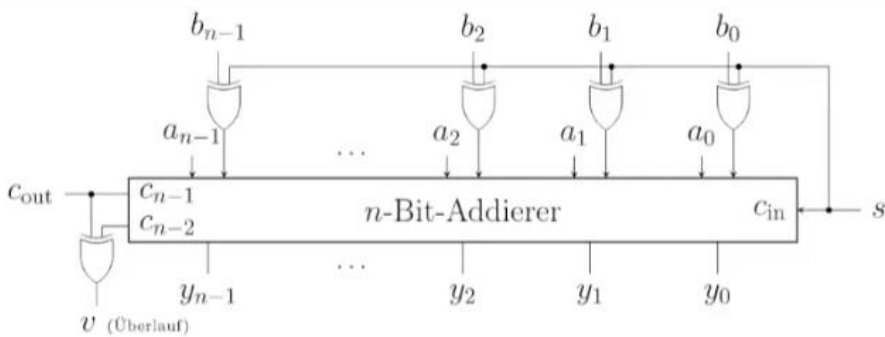
$$\underbrace{1 + p + m}_{n \text{ Bit}} (s, e_{p-1}, \dots, e_1, e_0, f_{m-1}, \dots, f_1, f_0)$$

- ☐ a. -1.890625
- ☐ b. -1.78125
- ☐ c. -1.5625
- ☒ d. -1.125

Test 8:

Welche Multiplikationsaufgabe $y = a \cdot b$ ließe sich bei einer seriellen Multiplikation am meisten durch den Einsatz von Booth' Idee beschleunigen? Gegeben sei $a = (00111000)$.

- ☐ a. $b = 00101010$
- ☒ b. $b = 00011110$
- ☐ c. $b = 10101010$
- ☐ d. $b = 01011010$



Es seien $n = 4$, $a = (0110)_2$, und $b = (0111)_2$. Nutzen Sie das bekannte Addier- und Subtrahierwerk um $a + b$ zu berechnen. Wie ist die Leitung s zu belegen? Was liegt danach an den Ausgabeleitungen an?

- ☐ a. $s = 1, c_{\text{out}} = 1, v = 1$
- ☐ b. $s = 1, c_{\text{out}} = 0, v = 0$
- ☐ c. $s = 0, c_{\text{out}} = 1, v = 0$
- ☒ d. $s = 0, c_{\text{out}} = 0, v = 1$

Angenommen, Sie möchten ein ARM-Assembler-Programm auf einem x86-Linux-Computer kompilieren und ausführen. Der Programmcode befindet sich in der Datei `prog.S`. In welcher Reihenfolge müssen Sie die folgenden Befehle ausführen?

☐ a.

```
arm-linux-gnu-ld -o prog prog.o
qemu-arm prog
arm-linux-gnu-as -o prog.o prog.S
```

☐ b.

```
qemu-arm prog
arm-linux-gnu-ld -o prog prog.o
arm-linux-gnu-as -o prog.o prog.S
```

☐ c.

```
arm-linux-gnu-as -o prog.o prog.S
qemu-arm prog
arm-linux-gnu-ld -o prog prog.o
```

☐ d.

```
qemu-arm prog
arm-linux-gnu-as -o prog.o prog.S
arm-linux-gnu-ld -o prog prog.o
```

☐ e.

```
arm-linux-gnu-ld -o prog prog.o
arm-linux-gnu-as -o prog.o prog.S
qemu-arm prog
```

☒ f.

```
arm-linux-gnu-as -o prog.o prog.S
arm-linux-gnu-ld -o prog prog.o
qemu-arm prog
```

Welcher ARM-Assemblerbefehl drückt die folgende Zuweisung $r_2 = r_1 * 8$ aus?

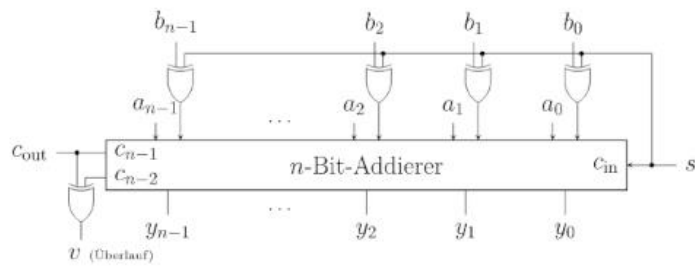
- ☐ a. `ADD r2, r1, #7`
- ☐ b. `ADD r2, r1, r1, LSL #4`
- ☒ c. `MOV r2, r1, LSL #3`
- ☐ d. `MOV r2, r1, LSL #2`

Welche der folgenden ARM-Assemblerbefehle **können** die Flags im Statusregister beeinflussen?

- ☐ a. `LDRB r6, [r7]`
- ☒ b. `TST r1, r0`
- ☒ c. `SUBS r0, r0, r1`
- ☐ d. `MULEQ r0, r0, r1`

Test 9:

Betrachten Sie den Schaltplan des kombinierten Addier- und Subtrahierwerkes für n Bit.



Es sei $n = 4$. An den Eingängen liegen die Binärzahlen $a = (1110)_2$ und $b = (0101)_2$ an.

Setzen sie die Steuerleitung s so, dass das Schaltwerk $y = a + b$ berechnet. Was liegt nach der Addition an den Ausgabelleitungen c_{out} und v an?

- ☐ a. $s = 1, c_{out} = 0, v = 1$
- ☐ b. $s = 0, c_{out} = 0, v = 1$
- ☒ c. $s = 0, c_{out} = 1, v = 0$
- ☐ d. $s = 1, c_{out} = 1, v = 0$

Welche der folgenden ARM-Assemblerbefehle **können** bei der Ausführung durch die Flags im Statusregister **beeinflusst werden**?

- ☐ a. `LDR r4, =32`
- ☒ b. `ORRMI r0, r1, r2`
- ☒ c. `ADDPL r0, r1, r1`
- ☐ d. `ADD r2, r0, r1`

Betrachten Sie den Pseudocode zur Bearbeitung des Registers `r1`. Wir rechnen mit vorzeichenbehafteten Ganzzahlen.

```
r1 = r1 - 4;  
  
if(r1 == 0) {  
    r1 = r1 + 7;  
}
```

Welche Kombination von ARM-Instruktionen bildet die gewünschte Funktionalität ab?

- ☐ a.
SUB r1, r1, #4
CMP r1, #0
ADD r1, r1, #7
- ☐ b.
SUBS r1, r1, #4
ADDMI r1, r1, #7
- ☒ c.
SUBS r1, r1, #4
ADDEQ r1, r1, #7
- ☐ d.
SUBS r1, r1, #4
ADDPL r1, r1, #7

In welcher Reihenfolge müssen Sie die Code-Schnipsel A, B und C einfügen, sodass das resultierende Programm die Zeichenkette `msg` rückwärts ausgibt?

Tipp: Achten Sie auf die Struktur der Schleife. Die einzelnen Instruktionen müssen nicht genau beachtet werden.

```
.arm
.global _start
.data
msg:
.ascii "Revert me!"
len = . - msg
.align
.text
```

A:	B:	C:
<pre>_start: ldr r1, =msg ldr r2, =len sub r4, r2, #1 add r4, r4, r1 mov r3, r1 revertloop:</pre>	<pre>ldrb r8, [r3] ldrb r9, [r4] strb r9, [r3], #1 strb r8, [r4], #-1</pre>	<pre>cmp r3, r4 blt revertloop</pre>

```
print:
mov r7, #4
swi #0
exit:
mov r0, #0
mov r7, #1
swi #0
```

- ☒ a. A, B, C
- ☐ b. A, C, B
- ☐ c. B, A, C
- ☐ d. B, C, A
- ☐ e. C, A, B
- ☐ f. C, B, A

Welche der folgenden Aussagen zur Instruktion `STMFA sp!, {r0-r10, lr}` sind korrekt?

- ☒ a. Die Instruktion geht von einem Stapel aus, der nach oben wächst (in Richtung höhere Adressen).
- ☒ b. Die Instruktion speichert die Rücksprung-Adresse auf dem Stapel.
- ☐ c. Die Instruktion geht von einem Stapel aus, der nach unten wächst (in Richtung niedrigere Adressen).
- ☐ d. Die Instruktion liest 12 aufeinanderfolgende 32-Bit-Worte vom Stapel.

Test 10:


Welche der folgenden ARM-Assemblerbefehle **können** die Flags im Statusregister **beeinflussen**?

- ☐ a. `BNE loop`
- ☐ b. `ANDMI r0, r1, r2`
- ☒ c. `MULS r0, r1, r2`
- ☒ d. `TST r0, r1`

Betrachten Sie den Pseudocode zur Bearbeitung des Registers `r1`. Wir rechnen mit vorzeichenbehafteten Ganzzahlen.

```
r1 = r1 + 4;  
  
if (r1 >= 0) {  
    r1 = r1 - 3;  
}
```

Welche Kombination von ARM-Instruktionen bildet die gewünschte Funktionalität ab?

- ☐ a.
`SUBS r1, r1, #4`
`ADDPL r1, r1, #3`
- ☐ b.
`ADD r1, r1, #4`
`SUBNE r1, r1, #3`
- ☒ c. 
`ADDS r1, r1, #4`
`SUBPL r1, r1, #3`
- ☐ d.
`ADD r1, r1, #4`
`CMP r1, #0`
`SUB r1, r1, #3`

Gegeben sei folgendes ARM-Schnipsel:

```
STRB r4, [r1]
ADD r1, r1, #1
```

Welche der folgenden Instruktionen implementiert die gleiche Funktionalität?

- ☐ a. `STRB r4, [r1, #1]!`
- ☐ b. `STRB r4, [r1, #1]`
- ☒ c. `STRB r4, [r1], #1`
- ☐ d. `STRB r4, [r1, r4, LSL #1]`

Sie wollen Werte **vom Stapel** in die Register `r2` bis `r5` und in das Linkregister `lr` **laden**. Sie verwenden einen **empty descending** Stack. Der Stackpointer `sp` soll entsprechend angepasst werden.

Welche der folgenden Instruktionen erfüllt diese Funktion?

- ☐ a.
`LDMFA sp!, {r2-r5, lr}`
- ☒ b.
`LDMED sp!, {r2-r5, lr}`
- ☐ c.
`STMED sp!, {r2-r5, lr}`
- ☐ d.
`STMFA sp!, {r2-r5, lr}`

Erinnern Sie sich an die ARM-Aufrufkonventionen. Welche der folgenden Aussagen sind korrekt?

- ☐ a. Der Rückgabewert kann in einem beliebigen Register gehalten werden.
- ☒ b. Das Unterprogramm muss die Register `r4` bis `r12` erhalten.
- ☒ c. Die ersten vier Argumente werden in den Registern `r0` bis `r3` übergeben.
- ☐ d. Unterprogramme dürfen `r4` bis `r12` nicht verwenden.

Test 11:

Gegeben sei folgendes ARM-Schnipsel:

```
ADD r1, r1, #2  
LDRH r4, [r1]
```

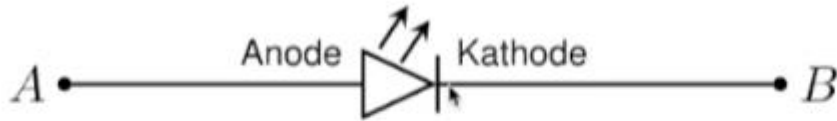
Welche der folgenden Instruktionen implementiert die gleiche Funktionalität?

- ☐ a. `LDRH r4, [r1, #2]`
- ☒ b. `LDRH r4, [r1, #2]!`
- ☐ c. `LDRH r4, [r1, r4, LSL #2]`
- ☐ d. `LDRH r4, [r1], #2`

Erinnern Sie sich an die ARM-Aufrufkonventionen. Welche der folgenden Aussagen sind korrekt?

- ☒ a. Die ersten vier Argumente werden in den Registern `r0` bis `r3` übergeben.
- ☐ b. Unterprogramme dürfen `r4` bis `r12` nicht verwenden.
- ☐ c. Der Rückgabewert kann in einem beliebigen Register bereitgestellt werden.
- ☒ d. Bei Funktionen mit mehr als vier Argumenten werden die letzten Argumente auf den Stapel gelegt.

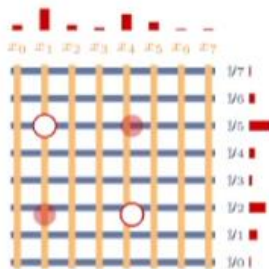
Leuchtdiode (LED)



Wie müssen A und B belegt werden, sodass die Leuchtdiode (LED) leuchtet?

- ☐ a. $A = L; B = L$
- ☐ b. $A = L; B = H$
- ☒ c. $A = H; B = L$
- ☐ d. $A = H; B = H$

Wenn ein Pro-Cap-Touchscreen mit dem Perimeter-Scan-Verfahren ausgelesen wird, entstehen bei der Verwendung von mehr als einem Finger so genannte Geisterpunkte.



Wie kann diese Mehrdeutigkeit beim Auslesen verhindert werden?

- ☐ a. Geisterpunkte können bei Pro-Cap-Touchscreens nicht verhindert werden.
- ☐ b. Durch mehrfaches Messen können Geisterpunkte erkannt und beseitigt werden.
- ☐ c. Es bedarf keiner Lösung, die Software kann alle Gesten trotzdem erkennen.
- ☒ d. Man liest die y-Messwerte pro Spalte aus. Spalten werden zyklisch ausgewählt. Nach jedem Zyklus werden die Messwerte zu einem Bild (Image) zusammengefügt.

Bei der Frage drunter sind die gelb markierten Kästchen **BEIDE** richtig!

Erinnern Sie sich an die Typen von E/A-Registern. Welche der folgenden Aussagen sind korrekt?



a. Unterbrechungsanforderungen stellen eine Alternative zum regelmäßigen Abfragen (polling) von Statusregistern dar.



b. Datenregister dienen zur Initialisierung und Funktionswahl.



c. E/A-Register sind **immer** lesbar und schreibbar.



d. Statusregister dienen zum Austausch von Zustandsinformationen.