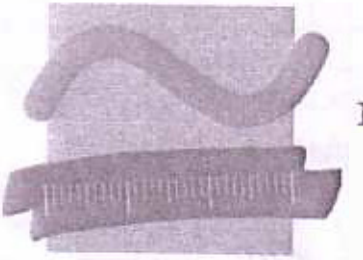
Klausur Technische Grundlagen der Informatik
Januar 2007

Dr. Wolff



Name:

Zulässiges Material: Tabelle der booleschen Funktionen, Zahlentabelle 0–16 (binär/oktal/dezimal/hexadezimal), Intel-Befehlstabelle.
Dauer: 90 Minuten. Aufgabenblatt bitte mit Name beschriften und als Deckblatt mit abgeben. Reklamationen nur bei Rückgabe. Bei Nichtabholung 1 Punkt Abzug.

Aufgabe 1

Wandeln Sie die Hexadezimalzahl 3E2A5B4 in das Oktalsystem.

Punkte: 0 ½ 0

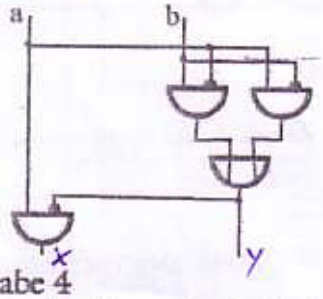
Aufgabe 2

Auf einem Intel-Prozessor werden nebenstehende Instruktionen als Rechnungen ganzer Zahlen (in Zweierkomplementcodierung) ausgeführt. Welches Resultat befindet sich an den mit „?" markierten Stellen jeweils in Register AL?

```
mov AL,5Ah
add AL,-43
; ?
mov AL,9Ah
sub AL,34h
; ?
```

Punkte: 1 ½ 0

Aufgabe 3

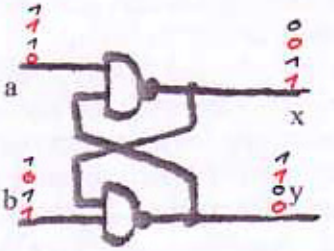


Analysieren Sie die abgebildete Schaltung. Stellen Sie die Wertetafel für die Ausgänge auf. Wie heißt diese Ihnen bekannte Schaltfunktion? Welche Schaltzeit hat die Schaltung bei einer Gatterdurchlaufzeit von t_G = 3,2 ns?

Punkte: 1 ½ 0

Aufgabe 4

Charakterisieren Sie das Verhalten der abgebildeten Schaltung: Nehmen Sie an, die Eingangswerte a und b werden in der Reihenfolge der Tabelle durchlaufen; bestimmen Sie jeweils die Ausgangswerte und tragen sie in die Tabelle ein. Wie heißt eine solche Schaltung?



a	b	x	y
0	1		
1	1		
1	0		
1	1		
0	0		

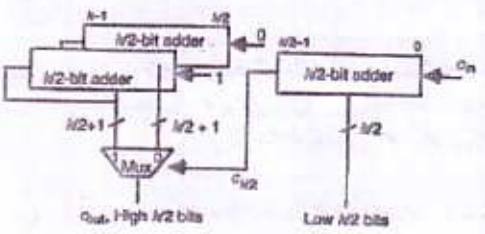
Punkte: 1½ 1 ½ 0

Aufgabe 5

Konstruieren Sie einen synchronen Zähler (eine Schaltung mit zwei Flipflops), der zyklisch die zweistelligen Zustände 00, 01, 11, 00, ... durchläuft. Der Zustand 10 kommt dabei nicht vor; sein Nachfolgezustand kann für die Konstruktion beliebig gewählt werden.

Punkte: 2 1½ 1 ½ 0

Aufgabe 6



Punkte: 1 ½ 0

Der Conditional-Sum-Addierer berechnet in der ersten Ebene für jede Stelle die Summe zweimal, nämlich in jeweils einer Variante für den Fall, dass der Übertrag aus der vorherigen Stelle 0 bzw. 1 ist. Hierfür wird jeweils ein Halbaddierer oder Volladdierer benötigt. Es entstehen Zwischenergebnisse der Länge 1 Bit.
In jeder weiteren Ebene werden jeweils 2 Zwischenergebnisse der Länge k/2 Bit zu einem neuen Zwischenergebnis der Länge k Bit zusammengefasst, wobei aus den zuvor ermittelten Ergebnisvarianten diejenigen ausgewählt werden, die zu den nunmehr ermittelten Überträgen passen (s. Bild, Mux). Die Schaltung jeder Ebene (Addierer, MUX) sei 2-stufig realisiert.
In jedem Schritt entstehen also Zwischenergebnisse der doppelten Länge, bis zum Schluss das Gesamtergebnis in der vollen Länge berechnet wurde. Wie schnell berechnet der Conditional-Sum-Addierer mit t_G = 3,2 ns Zahlen der Längen 64 Bit bzw. 128 Bit?

Aufgabe 7

Beachten Sie die Rückseite!

Punkte: 0

Aufgabe 8

Punkte: 2 1½ 1 ½ 0

Welche Komponente eines Rechners führt jeweils die beschriebene Aufgabe aus? Wählen Sie aus folgenden Komponentennamen die jeweils speziellste: ALU, CPU, CSU, MMU, FPU, Cache

- MMU Fordere nächsten Befehl aus dem Speicher an
- Cache Prüfe, ob angefordertes Speicherwort schneller zur Verfügung gestellt werden kann
- CPU Erhöhe Programmzähler nach der Befehlsabarbeitung
- CPU Rette bei Interruptbearbeitung Rücksprungadresse auf dem Stack
- MMU Ermittle aus der Adresse, an der ein Programm Daten lesen will, die Hauptspeicheradresse, an der sich diese befinden
- MMU Prüfe, ob ein angeforderter Speicher-Lese- oder Schreibzugriff stattfinden darf
- ALU 3+5
- FPU 3.7+2.8

Aufgabe 9

Punkte: 1 ½ 0

Auf welche Weise trägt die dargestellte Interruptroutine dazu bei, dass andere Programme durch ihre Aktivierung nicht im Ablauf gestört werden? Was fehlt?

```
intr_rout_7:
        push AX
        mov AX,counter1
        mov BX,counter2
        add AX,BX
        mov counter1,AX
        pop AX
        iret ; spezieller Return-Befehl für Interrupts
```

Aufgabe 10

Punkte: 2 1½ 1 ½ 0

Wer trägt dazu bei, dass die Berechnungen der Interruptroutine das unterbrochene Programm nicht stören? Markieren Sie jeweils oder nicht (je ¼ Punkt).

- ☒ Programmierer der Interruptroutine: muss zum Zeitpunkt der Rückkehr (iret) den vorherigen Registerzustand wiederhergestellt haben
- ○ Interruptcontroller bzw. der den Interrupt verursachende I/O-Controller: darf während des Interrupts keinen weiteren Interrupt anfordern
- ☒ CPU: muss die Bearbeitung des Interrupts mit der Befehlsabarbeitung so synchronisieren, dass dieser nach Rückkehr aus der Interrupt unbeeinträchtigt fortgesetzt werden kann
- ☒ CPU: muss Programmzähler retten, bevor sie die Interruptroutine aktiviert
- ○ CPU: muss sämtliche Register retten, bevor sie die Interruptroutine aktiviert
- ○ CPU: muss den Cache leeren, bevor sie die Interruptroutine aktiviert
- ○ Programmierer des Hauptprogramms: muss gelegentlich anhand eines Flags feststellen, ob es einen Interrupt gab, und seinen Zustand entsprechend anpassen
- ☒ Programmierer des Hauptprogramms: darf den Stack nur wie vorgesehen verwenden

Aufgabe 11

Punkte: 2 1½ 1 ½ 0

Die virtuelle Speicherverwaltung MMU führt eine „Seitentabelle" zur Abbildung der Programmadressen auf physische Adressen des Hauptspeichers. Jeder Eintrag darin kann verschiedene Attribute haben. Welche der folgenden Attribute gibt es und was bewirken sie?

- ☒ ausgelagert ~~swapped~~ swapped / Diese Daten wurden auf die Festplatte ausgelagert.
- ☒ gültig valid / Daten im Speicher vorhanden. Können benutzt werden.
- ☒ sticky Daten dürfen nicht ausgelagert werden.
- ○ slimy
- ☒ schreibgeschützt read only / Daten dürfen nur gelesen werden.

Bewertung	Punkte	14	13	12	11	10	9	8	7	6	5	4
	Note	1,0	1,3	1,7	2,0	2,3	2,7	3,0	3,3	3,7	4,0	5,0

Klausur TG1

Ja. '07

1.)

$3E2A5B4_{(16)}$

$= 0011\,1110\,0010\,1010\,0101\,1011\,0100_{(2)}$

$= 3705\,2664_{(8)}$ ✓

3.)

a	b	$a \cdot \overline{((\bar{a} \cdot b) + (\bar{b} \cdot a))}$	
0	0	0 0 0 0 0	0
0	1	0 0 1 1 0	0
1	0	1 1 0 1 1	x 0
1	1	1 0 0 0 0	1

Es handelt sich um die Konjunktion

$t_L = 3 \cdot t_G = 3 \cdot 3{,}2\,\mu s = 9{,}6\,\mu s$

4.)

a	b	x	y
0	1	1	0
1	1	1	0
1	0	0	1
1	1	0	1 ✓
0	0	unzulässiger Zustand	

R-S Flip-Flop ✓

ABER: Hier werden die vorherigen Eingangswerte bei a=1 und b=1 ⊗ ~~gespeichert~~. ✓

~~Dies~~ Verkömmlich werden die Werte beim R-S-FF bei a=0 und b=0 gespeichert.

⊗ in x und y ausgegeben (gespeichert).

6.)

$t_{CS1} = (\log_2 4 + 1) \cdot 2 \cdot t_0$

$t_{CS1_{64}} = (\log_2 64 + 1) \cdot 2 \cdot 3{,}2\,\mu s$

$= (6 + 1) \cdot 2 \cdot 3{,}2\,\mu s$

$= 14 \cdot 3{,}2\,\mu s$

$= 44{,}8\,\mu s$ ✓

NR: $14 \cdot 3{,}2$

42

28

$44{,}8$

$t_{CS1_{128}} = (\log_2 128 + 1) \cdot 2 \cdot 3{,}2\,\mu s$

$= 8 \cdot 6{,}4\,\mu s$

~~$= 67{,}2$~~

$= 51{,}2\,\mu s$ ✓

NR: $6{,}4 \cdot 8$

32

19

$51{,}2$

9.) Durch „push AX" wird der Registerinhalt auf den Stapel „gerettet". Durch „pop AX" wird dieser Inhalt nach der Bearbeitung der Interruptroutine wieder in das Register AX geschrieben. ✓

Es folgt „push BX", direkt nach „push AX" und „pop BX", direkt nach „pop AX", damit auch der Inhalt des Registers BX „gerettet" wird. Richtig

Klausur TG1

Jan. 07

3.)

a	b	X $a \cdot \overline{((\bar{a} \cdot b) + (a \cdot \bar{b}))}$
0	0	0
0	1	0
1	0	0
1	1	1

0	0	0	0
0	1	1	0
1	0	0	1
1	1	0	0

Es handelt sich beim linken Ausgang um die Konjunktion.

$a \cdot b$

$t_K = 3 \cdot t_G = 3 \cdot 3{,}2\,\mu s = 9{,}6\,\mu s$

a	b	Y $(\bar{a} \cdot b) + (a \cdot \bar{b})$
0	0	0
0	1	1
1	0	1
1	1	0

Es handelt sich beim rechten Ausgang um die Antivalenz.

$a \leftrightarrow b$; $a \oplus b$

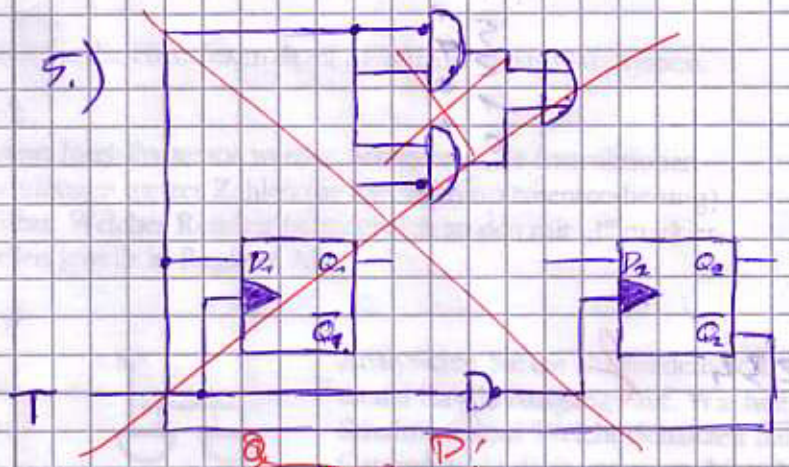
$t_A = 2 \cdot t_G = 2 \cdot 3{,}2\,\mu s = 6{,}4\,\mu s$

gefunde Fleit?

Klausur TG1

Jan. '07

5.)



Q2 → D'

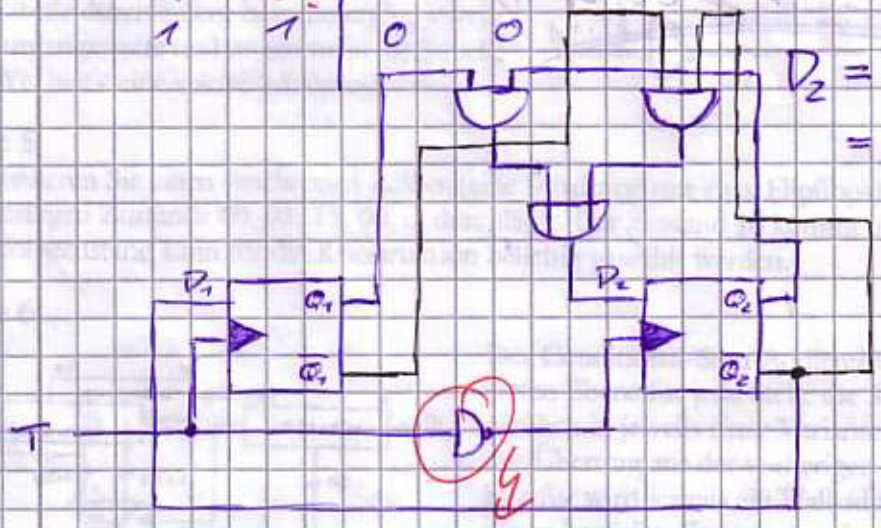
D_1	D_2	Q_1	Q_2
0	0	0	1
0	1	1	1
1	0	?	?
1	1	0	0

$D_1 = \overline{Q_2}$

$D_1 \leftrightarrow D_2 = Q_1$ ~~gleich~~

$D_2 = Q_1 \leftrightarrow Q_2$

$= (Q_1 \cdot Q_2) + (\overline{Q_1} \cdot \overline{Q_2})$



(4)

2.) NP: Hexadezimal

```
  5A
- 43
----

  54
- 2D
----
  23
```

```
43 | 1
22 | 0
11 | 1
 5 | 1
 2 | 0
 1 | 1
 0 | 1
```

$43_{(10} = 1011101_{(2)} = 2D_{(16}$

Erstes „2": 234 ✗

NP:

```
  94
- 34
----
  66
```

Zweites „2": 664 ✓