

# KLAUSUR

## Digitaltechnik 1

Sommersemester 2023

Prüfungsfach: Digitaltechnik 1

Studiengang: Technische Informatik

Semestergruppe: TIB2, IEP2

Fachnummer: 1052032

Erlaubte Hilfsmittel: keine

Zeit: 90 min.

**Tragen Sie hier bitte Ihren Daten ein:**

Name:	Vorname:	Matrikelnummer:


**Wichtiger Hinweis für die Bearbeitung der Aufgaben:**

Schreiben Sie bitte Ihre Lösungen möglichst auf die Aufgabenblätter.

Sollte der vorgesehene Platz nicht reichen, verwenden Sie bitte jeweils die Rückseite.

Viel Erfolg wünscht Ihnen

Jonas Fuhrmann und Michael Koidis

Prüfungsfach: <b>Digitaltechnik 1</b>	Sommersemester 2023	
Name, Vorname:	Mat.-Nr.:	

# 1 Boolesche Algebra

## 1.1 Boolesche Gleichung

(7 Punkte)

### Schaltungsanalyse

Gegeben ist die boolesche Gleichung:  $Y = A \wedge C \vee B \wedge \bar{C} \vee A \wedge \bar{B} \vee B \wedge \bar{A}$

Zeichnen Sie dazugehörige Schaltung

Wie ist die Funktionslänge  $l$  und die Schachteltiefe  $k$  der Schaltung?

l =

k =

### Funktionstabelle

Bestimmen Sie die Funktionstabelle der booleschen Gleichung  $Y = A \wedge C \vee B \wedge \bar{C} \vee A \wedge \bar{B} \vee B \wedge \bar{A}$

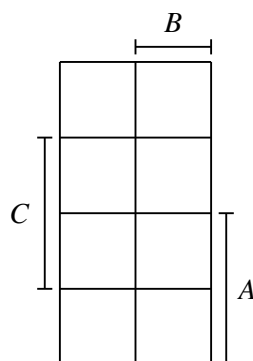
	C	B	A					
0	0	0	0					
1	0	0	1					
2	0	1	0					
3	0	1	1					
4	1	0	0					
5	1	0	1					
6	1	1	0					
7	1	1	1					

## 1.2 Minimierung

(6 Punkte)

Bestimmen Sie die disjunktive Minimalform  $Y_{DMF}$  der booleschen Gleichung  $Y = A \wedge C \vee B \wedge \bar{C} \vee A \wedge \bar{B} \vee B \wedge \bar{A}$ . Übertragen Sie zuerst Ihre Lösung aus Aufgabe 1.1 in die Tabelle 1, füllen dann das KV-Diagramm aus und bestimmen danach die disjunktive Minimalform  $Y_{DMF}$ .

	C	B	A	Y
0	0	0	0	
1	0	0	1	
2	0	1	0	
3	0	1	1	
4	1	0	0	
5	1	0	1	
6	1	1	0	
7	1	1	1	



$Y_{DMF} =$

Tabelle 1: Ergebnis aus Aufgabe 1.1

Zeichnen Sie die Schaltung der oben bestimmten disjunktiven Minimalform  $Y_{DMF}$ ?

Wie ist die Funktionslänge  $l$  und die Schachteltiefe  $k$  der zur disjunktiven Minimalform  $Y_{DMF}$  gehörenden Schaltung?

$l =$

$k =$



## 2.2 Zahlendarstellung nach IEEE 754

(5 Punkte)

Wandeln Sie die Dezimalzahl  $(25,625)_{10}$  in eine Gleitkommazahl in einfacher Genauigkeit nach IEEE 754 in hexadezimaler Schreibweise um.

Hinweis: Eine Gleitkommazahl in einfacher Genauigkeit (32 Bit) ist nach IEEE 754 wie folgt codiert:


Bits	1	8	23
	VZ von $M$	$E + 127$	$ M $ ohne $m_0$

- Das Bit 31 (MSB) kennzeichnet das Vorzeichen.
- Die nächsten 8 Bit 30...23 geben den Exponenten an (Offsetdarstellung um 127).
- Die nächsten 23 Bit 22...0 geben die normalisierte Mantisse ohne die Vorkomma-Eins an.

Abbildung 1: Darstellung von Gleitkommazahl in einfacher Genauigkeit (32 Bit) nach IEEE 754

<b>normalisierte Zahl</b>	$\pm$	$0 < \text{Exponent} < \text{max}$	Mantisse beliebig
<b>denormalisierte Zahl</b>	$\pm$	0000 0000	Mantisse nicht alle Bits 0
<b>Null</b>	$\pm$	0000 0000	0...0
<b>Unendlich</b>	$\pm$	1111 1111	0...0
<b>NaN</b>	$\pm$	1111 1111	Mantisse nicht alle Bits 0

Tabelle 3: Sonderfälle Gleitkommazahl in einfacher Genauigkeit (32 Bit) nach IEEE 754

Prüfungsfach: <b>Digitaltechnik 1</b>	Sommersemester 2023	
Name, Vorname:	Mat.-Nr.:	

### 2.3 Blockcodes

(11 Punkte)


Gegeben ist die Generatormatrix und die Codeworte  $\mathbf{X}_0$  bis  $\mathbf{X}_3$

Erzeugen sie die Parity-Check-Matrix  $H^T$  aus der obigen Generatormatrix  $G$ .

Wie lautet die mit Hilfe der Generatormatrix  $G$  erzeugten Codeworte  $\mathbf{Y}_0$  bis  $\mathbf{Y}_3$ ?

Wie groß ist die Hammingdistanz des mit der Generatormatrix  $G$  erzeugten Codes?

Wie viele Bitfehler können sicher **erkannt** werden und wie viele Fehler können **korrigiert** werden?

Prüfungsfach: <b>Digitaltechnik 1</b>	Sommersemester 2023	
Name, Vorname:	Mat.-Nr.:	

## 2.4 VHDL

(9 Punkte)

Gegeben ist die in VHDL beschriebene Komponente DECREMENTER. Vervollständigen Sie die Komponente so, dass die Funktion  $Y = (X) - 1$  ausgeführt und die Condition Flags, Carry (CF), Overflow (OF), Zero (ZF) und Negative (NF) korrekt berechnet werden!

*Tipp: Das Overflow Flag ist dann gesetzt, wenn das MSB des Operanden 1 und das MSB des Ergebnisses 0 entspricht.*

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;

ENTITY ADDER IS      -- 3 Bit Wortaddierer
    PORT(IN1, IN2 : IN  STD_LOGIC_VECTOR(2 downto 0);
          C_IN   : IN  STD_LOGIC;
          SUM     : OUT STD_LOGIC_VECTOR(2 downto 0);
          C_OUT   : OUT STD_LOGIC;
END ADDER;
ARCHITECTURE BEHAV OF ADDER IS
    SIGNAL result: STD_LOGIC_VECTOR(3 downto 0);
BEGIN
    result <= STD_LOGIC_VECTOR(UNSIGNED('0' & IN1) + UNSIGNED(IN2) + C_IN);
    SUM     <= result(2 downto 0);
    C_OUT   <= result(3);
END BEHAV;
-----
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY DECREMENTER IS
    PORT(X : IN  STD_LOGIC_VECTOR(2 downto 0);
          Y : OUT STD_LOGIC_VECTOR(2 downto 0);
          CF, OF, ZF, NF : OUT STD_LOGIC
    );
END DECREMENTER;
ARCHITECTURE FUNC OF DECREMENTER IS
    SIGNAL SUM : STD_LOGIC_VECTOR(2 downto 0);
    SIGNAL
        ----- : STD_LOGIC;
BEGIN
    ADD: ENTITY work.ADDER
        PORT MAP(
            IN1    => X,
            IN2    =>
                -----,
            C_IN   =>
                -----,
            SUM    => SUM,
            C_OUT  =>
                -----
        );
    Y <= SUM
    CF <=
        -----;
    OF <=
        -----;
    ZF <=
        -----;
    NF <=
        -----;
END FUNC;

```

### 3 Hardware

Die in Abbildung 2 dargestellte 8 Bit-ALU enthält neben einem 8 Bit Addierer eine 8 Bit-Logik-Einheit, ein 8-faches AND-Gatter sowie einen Block „Status“ zur Bildung des Carry-Flags (CF), Overflow-Flags (OF), Zero-Flags (Z) und Negativ-Flags (N).

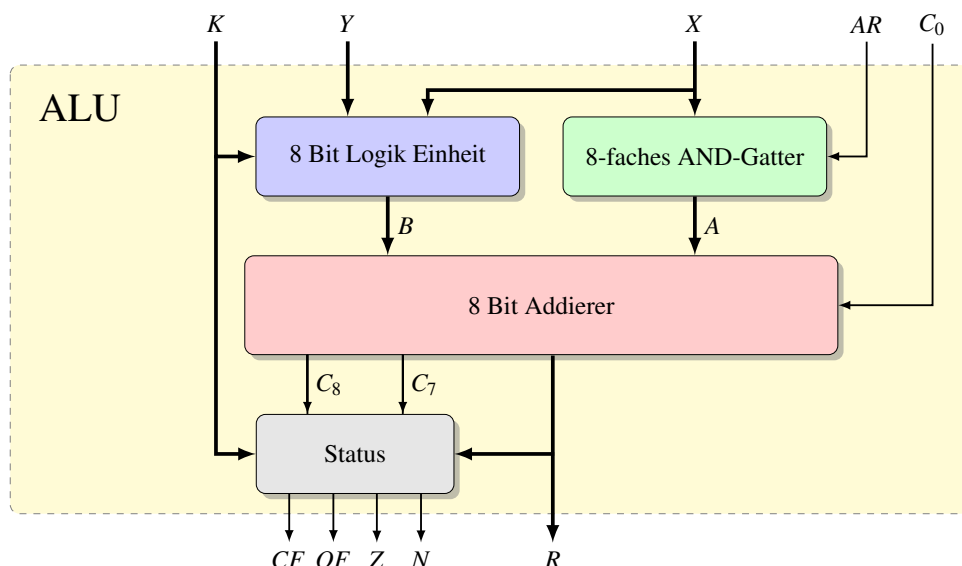


Abbildung 2: Aufbau 8-Bit ALU

Die Signale haben folgende Bitbreite:

Signalname	A	B	X	Y	R	K	AR	C <sub>0</sub>	C <sub>7</sub>	C <sub>8</sub>	CF	OF	Z	N
Breite in Bit	8	8	8	8	8	4	1	1	1	1	1	1	1	1

Tabelle 4: Bitbreite der Signale

$AR=0$  sperrt das 8-fach AND-Gatter, d.h.  $A = 0$ .  $AR=1$  schaltet  $X$  nach  $A$  durch, d.h.  $A = X$ .

Die gültigen Steuerworte  $K$  sind der Tabelle 5 zu entnehmen.

Steuerwort (K)	Ergebnis für Stelle $B_i$	Logik-Funktion
$(0000) = 0_H$	$B_i = 0$	Kontradiktion
$(0001) = 1_H$	$B_i = 1$	Tautologie
$(0010) = 2_H$	$B_i = X_i$	Identität X
$(0011) = 3_H$	$B_i = Y_i$	Identität Y
$(0100) = 4_H$	$B_i = \overline{X}_i$	Bitweise Invertierung X
$(0101) = 5_H$	$B_i = \overline{Y}_i$	Bitweise Invertierung Y
$(1000) = 8_H$	$B_i = X_i \vee Y_i$	OR
$(1001) = 9_H$	$B_i = X_i \wedge Y_i$	AND

Tabelle 5: Wirkung des Steuersignals (K) auf  $B_i$  in Abhängigkeit von  $X_i$  und  $Y_i$  ( $i = 0, \dots, 7$ ).





### 3.2 Speicherelemente

(7 Punkte)

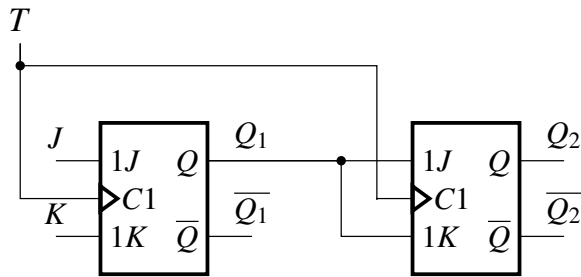


Abbildung 3: Schaltung mit zwei JK-Flipflops

$E_K^k$	$E_J^k$	$Q^{k+1}$
0	0	$Q^k$
0	1	1
1	0	0
1	1	$\overline{Q^k}$

Tabelle 8: Verkürzte Funktionstabelle eines Jump-Kill-FlipFlops

Wie nennt man die in der Schaltung in Abbildung 3 verwendete Art der Taktsteuerung?

Ergänzen Sie im folgenden Impulsdiagramm das Zeitverhalten der Schaltung aus Abbildung 3.

Die Gatterlaufzeiten können vernachlässigt werden. *Tipp: Eine Verzögerung kann dabei helfen Fehler zu vermeiden.*

