

KLAUSUR Informationstechnik

Sommersemester 2015

MUSTERLÖSUNG

Prüfungsfach:	Informationstechnik
Studiengang:	Wirtschaftsinformatik, Softwaretechnik
Semestergruppe:	WKB 1, SWB 1
Fachnummer:	1051002
Erlaubte Hilfsmittel:	keine
Zeit:	90 min.

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.

Reiner Marchthaler und Hans-Gerhard Groß

1 Kombinatorische Schaltung

1.1 KV-Diagramm

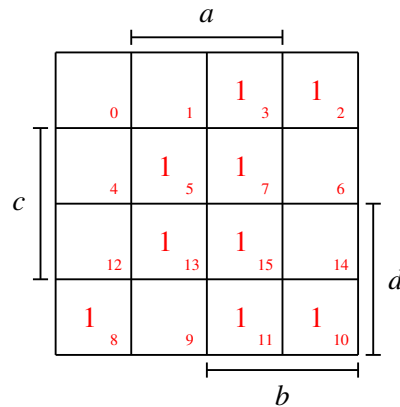
(15 Punkte)

Gegeben ist eine kombinatorische Schaltung. Diese wird durch eine Funktion Y_1 (siehe Tabelle 1) beschrieben.

	d	c	b	a	Y_1	Y_2
0	0	0	0	0	0	0
1	0	0	0	1	0	0
2	0	0	1	0	1	1
3	0	0	1	1	1	1
4	0	1	0	0	0	0
5	0	1	0	1	1	X
6	0	1	1	0	0	0
7	0	1	1	1	1	1
8	1	0	0	0	1	X
9	1	0	0	1	0	0
10	1	0	1	0	1	1
11	1	0	1	1	1	1
12	1	1	0	0	0	0
13	1	1	0	1	1	X
14	1	1	1	0	0	0
15	1	1	1	1	1	1

Tabelle 1: Funktionstabelle

1. Bestimmen Sie die DMF des Signals Y_1 mit Hilfe des KV-Diagramms und die Funktionslänge I_{DMF1} .



$$Y_{DMF1} = b\bar{c} \vee ac \vee \bar{a}\bar{c}d$$

$$I_{DMF1} = 10$$

2. Es wird festgestellt, dass die Eingangsbedingungen:

$$(d,c,b,a) = (0,1,0,1),$$

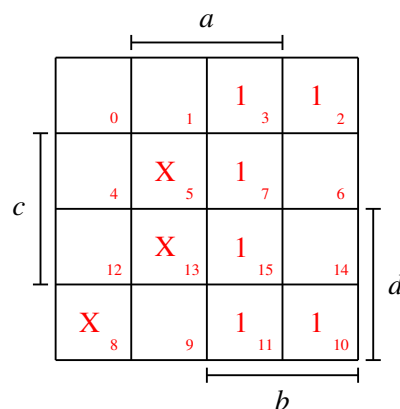
$$(d,c,b,a) = (1,0,0,0) \text{ und}$$

$$(d,c,b,a) = (1,1,0,1)$$

nicht auftreten können und festgelegt, dass dieser Umstand zur Minimierung der Schaltung herangezogen werden soll.

Tragen Sie diesen Umstand in die Tabelle für das Signal Y_2 ein.

Ermitteln Sie die DMF für die minimierte Funktion Y_2 mit Hilfe d. KV-Diagramms und bestimmen Sie d. Funktionslänge I_{DMF2} .



$$Y_{DMF2} = b\bar{c} \vee ab \quad \text{bzw.} \quad b\bar{c} \vee ac$$

$$I_{DMF2} = 6$$

2 Zahlendarstellung und Codierung

2.1 Festkommadarstellung

(14 Punkte)

1. Wandeln Sie die Hexadezimalzahl $(A1, C)_{16}$ in eine Dezimalzahl (Zahlenbasis 10) um.

(2 Punkte)

$$\begin{aligned}
 &(1010\ 0001, 1100)_2 \\
 &= 128 + 32 + 1 + 0,5 + 0,25 = (161,75)_{10}
 \end{aligned}$$

2. Wandeln Sie die Dezimalzahl $(89,625)_{10}$ in eine Zahl zur Basis 8 um.

(4 Punkte)

$$\begin{array}{rcll}
 89 & : & 2 & = 44 & \text{Rest } 1 \\
 44 & : & 2 & = 22 & \text{Rest } 0 \\
 22 & : & 2 & = 11 & \text{Rest } 0 \\
 11 & : & 2 & = 5 & \text{Rest } 1 \\
 5 & : & 2 & = 2 & \text{Rest } 1 \\
 2 & : & 2 & = 1 & \text{Rest } 0 \\
 1 & : & 2 & = 0 & \text{Rest } 1
 \end{array}
 \Rightarrow Z = (101\ 1001)_2$$

$$\begin{array}{rcll}
 0,625 & \cdot & 2 & = 0,25 & + & 1 \\
 0,25 & \cdot & 2 & = 0,5 & + & 0 \\
 0,5 & \cdot & 2 & = 0,0 & + & 1
 \end{array}
 \Rightarrow Z = (0,101)_2$$

$$Z = (1011001,101)_2$$

$$Z = (131,5)_8$$

Prüfungsfach: Informationstechnik	Sommersemester 2015	Hochschule Esslingen University of Applied Sciences
Name, Vorname:	Mat.-Nr.:	

3. Geben Sie die Binärzahl $(1000\ 0011)_2$ als Dezimalzahl an falls:

(8 Punkte)

Dualcodierung (Betragzahl) zugrundeliegt: $128 + 2 + 1 = (131)_{10}$

2er-Komplement-Codierung zugrundeliegt: $131 - 256 = (-125)_{10}$

Offset-Code-Codierung zugrundeliegt: $131 - 128 = (3)_{10}$

Vorzeichen-Betrags-Codierung zugrundeliegt: $(-3)_{10}$

2.2 Zahlendarstellung nach IEEE 754

(10 Punkte)

Wandeln Sie die Dezimalzahl $(-0.25)_{10}$ in eine Gleitkommazahl in einfacher Genauigkeit nach IEEE 754 in hexadezimaler Schreibweise um.

Hinweis zu Gleitkommazahl in einfacher Genauigkeit (32 Bit) nach IEEE 754:

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

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

Tabelle 2: Sonderfälle Gleitkommazahl in einfacher Genauigkeit (32 Bit) nach IEEE 754

Platz für Berechnung:

$(-0.25)_{10} = (-0.01)_2 = (-1,0)_2 \cdot 2^{-2}$	(2 Punkte)
VZ: „-“ \rightarrow 1	(1 Punkt)
Exp: $-2 + 127 = 125 = 0111\ 1101$	(2 Punkte)
Mant. 000 0000 0000 0000 0000 0000	(1 Punkt)
$\overbrace{1\ 011\ 1110\ 1000\ 0000\ 0000\ 0000\ 0000\ 0000}^{\text{VZ}\quad\text{Exp.}\quad\text{Mant.}}$	(2 Punkte)
$\underbrace{1011}_{B}\ \underbrace{1110}_{E}\ \underbrace{1000}_8\ \underbrace{0000}_0\ \underbrace{0000}_0\ \underbrace{0000}_0\ \underbrace{0000}_0\ \underbrace{0000}_0$	
Ergebnis: <u>(BE80 0000) Hex</u>	(2 Punkte)

3 Hardware

3.1 ALU

(18 Punkte)

Die in Abbildung 2 dargestellte 4 Bit-ALU enthält neben einem 4 Bit Addierer, eine 4 Bit-Logik-Einheit, ein 4-faches AND-Gatter sowie einen Block „Status“ zur Bildung des Carry-Flags (CF), Overflow-Flags (OF), Zero-Flags (Z) und Negativ-Flags (N).

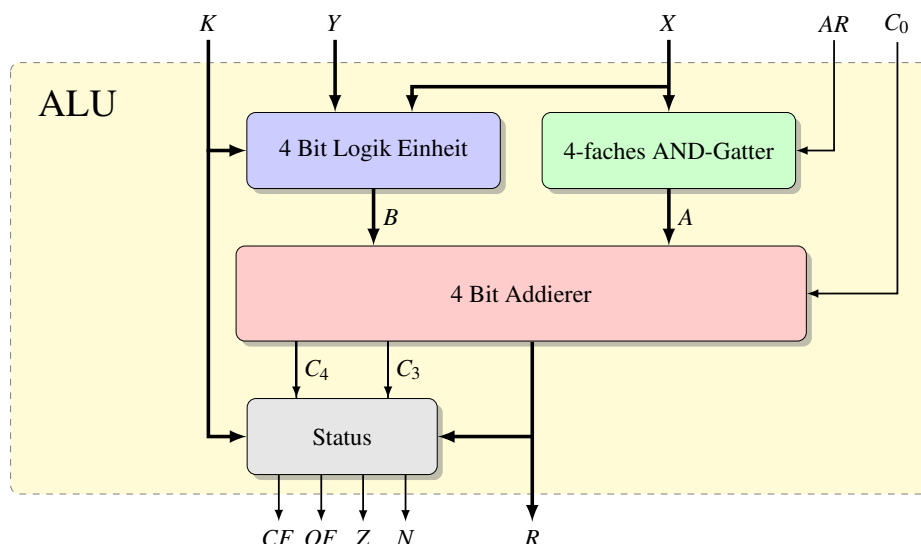


Abbildung 2: Aufbau 4-Bit ALU

Die Signale haben folgende Bitbreite:

Signalname	A	B	X	Y	R	K	AR	C ₀	C ₃	C ₄	CF	OF	Z	N
Breite in Bit	4	4	4	4	4	3	1	1	1	1	1	1	1	1

Tabelle 3: Bitbreite der Signale

Die gültigen Steuerworte des Steuersignals **K** sind der Tabelle 4 zu entnehmen.

Steuerwort (K)	Ergebnis für Stelle B_i	Logik-Funktion
(000) = 0 _H	$B_i = 0$	Kontradiktion
(001) = 1 _H	$B_i = X_i$	Identität X
(010) = 2 _H	$B_i = Y_i$	Identität Y
(011) = 3 _H	$B_i = 1$	Tautologie
(100) = 4 _H	$B_i = X_i \vee Y_i$	OR
(101) = 5 _H	$B_i = X_i \wedge Y_i$	AND
(110) = 6 _H	$B_i = \bar{X}_i$	Bitweise Invertierung X
(111) = 7 _H	$B_i = \bar{Y}_i$	Bitweise Invertierung Y

Tabelle 4: Wirkung des Steuersignals (K) auf B_i in Abhängigkeit von X_i und Y_i ($i = 0, \dots, 3$).

Hinweis: AR=0 sperrt das 4-Bit AND-Gatter und AR=1 schaltet **X** nach **A** durch!

3.2 Speicher

(13 Punkte)

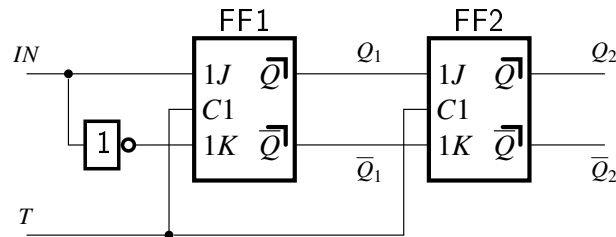
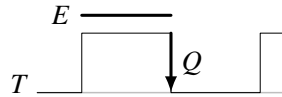


Abbildung 3: Schaltung mit zwei JK-MS-Flipflops

Hinweise zu JK-FF und MS-FF:

$1J^k$	$1K^k$	Q^{k+1}
0	0	Q^k
0	1	0
1	0	1
1	1	$\overline{Q^k}$



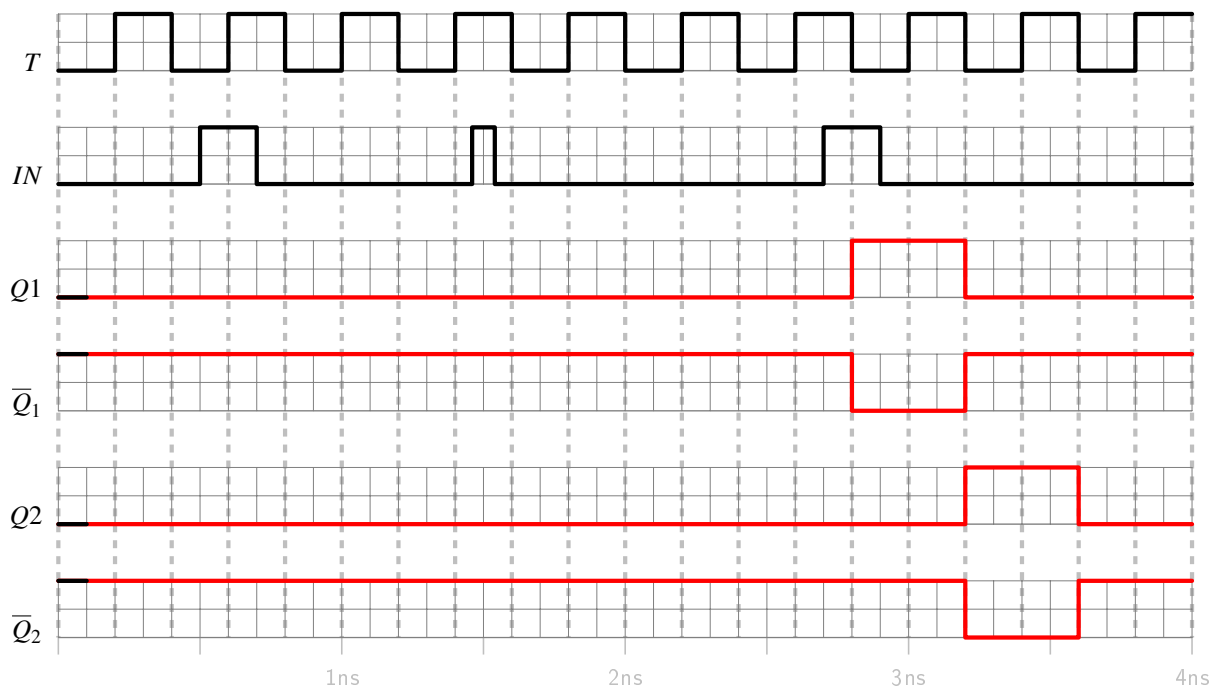
Änderung während
aktiver Taktphase
wird am Ausgang
erst **nach aktiver**
Taktphase wirksam


Tabelle 7: Vereinfachte Funktionstabelle JK-FF

Abbildung 4: Funktionsweise bei Änderung am Eingang eines MS-FF

Vervollständigen Sie im nachfolgenden Impulsdiagramm die Signale Q_1 , \overline{Q}_1 , Q_2 und \overline{Q}_2 der Schaltung aus Abbildung 3.

Die Gatterlaufzeiten sind zu vernachlässigen ($t_{P,clk \rightarrow Q,LH} = t_{P,clk \rightarrow \overline{Q},LH} = t_{P,clk \rightarrow Q,HL} = t_{P,clk \rightarrow \overline{Q},HL} = 0ns$)



Prüfungsfach: Informationstechnik	Sommersemester 2015	 Hochschule Esslingen University of Applied Sciences
Name, Vorname:	Mat.-Nr.:	

4 Offene Fragen

4.1 Boolesche Algebra

(6 Punkte)

(1) Erklären Sie kurz, was man unter der Schachtelungstiefe (k) einer Gleichung versteht und erläutern Sie welche Auswirkung die Schachtelungstiefe auf eine Schaltung hat.

(2) Erklären Sie außerdem, warum die Realisierung einer DNF/KNF immer eine Schachtelungstiefe $k \leq 2$ hat.

(1) Die Schachtelungstiefe gibt an, wieviele Gatter in einer Funktion nacheinander durchlaufen werden müssen. Sie gibt das Maß der Durchlaufverzögerung einer Schaltung an. (3 Punkte)

(2) DNF bestehen immer aus UND-Termen der Eingabeparameter, die bei mehr als einem UND-Term ODER-verknüpft werden. Dies sind maximal 2 Ebenen der Schachtelungstiefe. (3 Punkte)

4.2 Zahlen und Daten

(6 Punkte)

(1) Erklären Sie warum bei einer in IEEE754 kodierten Fließkommazahl mit doppelter Genauigkeit (64 Bit) der Exponent mit $E+1023$ im Offset kodiert ist und nicht mit $E+1024$

(2) Erklären Sie das Problem, das bei der Addition einer sehr großen und einer sehr kleinen Gleitkommazahl möglicherweise entstehen kann.

(1) Um die Sonderfälle norm. Zahl, denorm. Zahl, Null, Unendlich, NaN darstellen zu können. (3 Punkte)

(2) Durch den Exponentenausgleich, d.h. Anpassung des Exponenten der kleinen Zahl an den der großen Zahl durch Shift, kann der Wertebereich in der Mantisse so verschoben werden, dass eine Null entsteht (d.h. die signifikanten Bits werden nach rechts „herausgeschoben“). (3 Punkte)

Prüfungsfach: Informationstechnik	Sommersemester 2015	Hochschule Esslingen University of Applied Sciences
Name, Vorname:	Mat.-Nr.:	

4.3 Fehlererkennende und -korrigierende Codes

(6 Punkte)

(1) Erklären Sie kurz inwiefern (wie/warum) Redundanz bei der Erkennung und Korrektur von fehlerhaften Übertragungen hilfreich sein kann.

(2) Erklären Sie kurz inwiefern die Hammingdistanz eines redundanten Codes hierbei eine Rolle spielt.

(1) Durch gezieltes Hinzufügen von Kontrollstellen (Redundanz), die anhand des eigentlichen Nachrichteninhalts gewählt werden (z.B. Paritätsbits), kann eine Verfälschung der übertragenen Information erkannt werden. Hierbei wird Zusatzinformation über die übertragene Information (Meta-Information) mit übertragen und beim Empfänger ausgewertet. (3 Punkte)

(2) Aus der Hammingdistanz eines redundanten Codes lässt sich berechnen, wie viele Übertragungsfehler des Codes sicher erkannt und sicher korrigiert werden können. (3 Punkte).

4.4 Code Übersetzung (Kompilierung)

(6 Punkte)

Erklären Sie kurz, wo/wozu bei der Code-Übersetzung (Kompilierung) eine Binärbaum-Datenstruktur nützlich ist.

Ein Binärbaum wird (als Abstract Syntax Tree) bei der Umsetzung von Code-Elementen der höheren (abstrakteren) Sprache in den 3-Address-Code. Jeder Teilbaum, der aus jeweils einem „Wurzelknoten“ und zwei „Blattknoten“ besteht, wird jeweils in eine Assembler-Anweisung mit Befehlsadresse und Parameter-Adresse übertragen.

- Im Vergleich zum „source code“ beinhaltet der AST verschiedene Elemente wie Satzzeichen und Trennsymbole (Klammern, Strichpunkt, etc.). - Jedes Element kann mit Eigenschaften/Anmerkungen versehen werden. (Im Gegensatz zum „source code“, wo dies nicht möglich ist) - Diese Zusatzinformationen helfen z.B. bei der Fehleranalyse um den Nutzer den Fehler den Ort des Fehlers anzuzeigen

4.5 Betriebssysteme

(6 Punkte)

Erklären Sie die grundsätzlichen Gemeinsamkeiten und Unterschiede der beiden Betriebssystem-Konzepte (1) Prozess und (2) Thread.

Beide Konzepte stellen Ausführungseinheiten dar. Prozess ist die übergeordnete Ausführungseinheit, Thread die untergeordnete. Mehrere Threads eines Prozesses teilen den Speicher dieses Prozesses, d.h. sie können auf alle Speicherelemente zugreifen. Unterschiedliche Prozesse können nicht auf die jeweiligen Ressourcen zugreifen, hierfür sind spezielle Interprozesskommunikationsverfahren nötig.