Prüfungsklausur Computersysteme (25211)
Teil 1608
SS 2015

Prof. Dr. J. Keller

19.09.2015

Bewertungsschema

Aufgabe	a	b	c	d	e	total
I-1	3	2	2	2	3	12
I-2						4
I-3	2	3	4			9
I-4	4	4				8
I-5						4
I-6	5	3				8
I-7	2	3				5

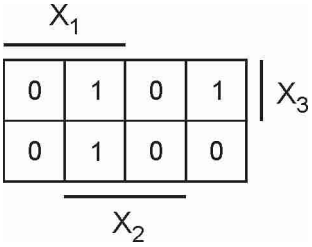
Aufgabe I-1 Schaltfunktionen

a) Bestimmen Sie die Kosten der boole'schen Formel $X \wedge \bar{Y} \vee \bar{Y} \wedge Z$. (1 P.)

Welche der folgenden Aussagen über die Kosten $C(f)$ der von einer boole'schen Formel berechneten Schaltfunktion $f(X, Y, Z)$ treffen zu? (2 P.)

trifft zu:	trifft nicht zu:	
☐	☐	$C(f)$ kann kleiner als die Kosten der Formel sein.
☐	☐	$C(f)$ muss gleich den Kosten der Formel sein.
☐	☐	$C(f)$ kann gleich den Kosten der Formel sein.
☐	☐	$C(f)$ kann größer als die Kosten der Formel sein.

b) Gegeben ist folgendes KV-Diagramm einer Schaltfunktion in 3 Variablen X_1, X_2, X_3.



Geben Sie die Primimplikanten der Schaltfunktion an. Geben Sie weiterhin an, welche der Primimplikanten Kernimplikanten sind. (2 P.)

c) Gegeben ist die folgende reduzierte Primtermtabelle, die nicht mehr weiter vereinfacht werden kann. Hierbei bezeichnet die Spalte K die Kosten der Primimplikanten P_i. Geben Sie an, welche Primimplikanten zu einer Überdeckung mit minimalen Kosten gehören. (2 P.)

	M1	M2	M3	M4	M5	M6	K
P1	X	X					3
P2	X			X	X		5
P3			X		X		2
P4				X		X	3
P5		X	X			X	5

d) Welche der folgenden Aussagen treffen zu? (2 P.)

trifft zu:	trifft nicht zu:	
☐	☐	$((X_1 \vee X_2) \wedge X_3)$ ist ein vollständig geklammerter boole'scher Ausdruck gemäß Definition 1.8 des Kurstextes.
☐	☐	$(X_1 \vee ((X_2) \wedge X_3))$ ist ein vollständig geklammerter boole'scher Ausdruck gemäß Definition 1.8 des Kurstextes.
☐	☐	Es gibt Schaltfunktionen, die mehrere Minimalpolynome besitzen.
☐	☐	Es gibt Schaltfunktionen, die mehrere Minimalpolynome besitzen, und bei denen die Minimalpolynome nur aus Kernimplikanten bestehen.

e) Verwandeln Sie die folgenden boole'schen Formeln mithilfe des Distributivgesetzes und der Morgan-Regeln in eine möglichst kurze disjunktive Normalform (DNF, Polynom). Hierbei sollen Monome, die bei jeder Einsetzung den Wert 0 erhalten, sowie Teilmonome anderer Monome der DNF eliminiert werden. (3 P.)

$(X_1 \vee X_2) \wedge (X_3 \vee X_1 \wedge \bar{X}_2)$

$\overline{(X_1 \vee X_2)} \wedge \overline{(X_3 \wedge (X_1 \wedge \bar{X}_2))}$

Lösungsvorschläge

Zu a) Die Kosten betragen 5. Die erste und die dritte Aussage treffen zu.

Zu b) Primimplikanten X_1X_2, $\bar{X}_1\bar{X}_2X_3$. Beide Primimplikanten sind auch Kernimplikanten.

Zu c) Die Primimplikanten $P1$, $P3$ und $P4$ bilden die Überdeckung mit minimalen Kosten.

Zu d) Die erste und die dritte Aussage treffen zu.

Zu e) $(X_1 \vee X_2) \wedge (X_3 \vee X_1 \wedge \bar{X}_2) = X_1X_3 \vee X_2X_3 \vee X_1\bar{X}_2$

$$\begin{aligned} & \overline{(X_1 \vee X_2)} \wedge \overline{(X_3 \wedge (X_1 \wedge \bar{X}_2))} \\ = \; & \bar{X}_1 \wedge \bar{X}_2 \wedge (\bar{X}_3 \vee \bar{X}_1 \vee X_2) \\ = \; & \bar{X}_1 \wedge \bar{X}_2 \end{aligned}$$

Aufgabe I-2 Binärzahlen und Arithmetik

Rechnen Sie die Zahlendarstellungen um. (je 1 P.)

$\langle 11100101 \rangle =$ ☐

$[10111100] =$ ☐

$\text{bin}_8(135) =$ ☐

$\text{twoc}(-43) =$ mit 7 Stellen einschl. Vorzeichen ☐

Lösungsvorschläge

$\langle 11100101 \rangle = 2^7 + 2^6 + 2^5 + 2^2 + 2^0 = 128 + 64 + 32 + 4 + 1 = 229$

$[10111100] = -2^7 + 2^5 + 2^4 + 2^3 + 2^2 = -128 + 32 + 16 + 8 + 4 = -68$

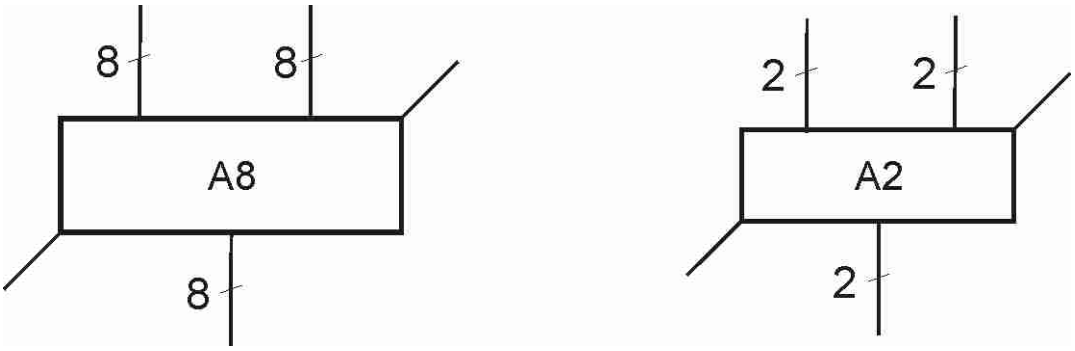
$\text{bin}_8(135) = \text{bin}_7(2^7 + 2^2 + 2^1 + 2^0) = 10000111$

$\text{twoc}(-43) = \text{twoc}(-2^6 + 2^4 + 2^2 + 2^0) = 1010101$

Aufgabe I-3 Schaltnetze

a) Geben Sie ein Schaltnetz für einen $(1,1)$-Multiplizierer an, d.h. ein Schaltnetz, das eine 1-Bit-Binärzahl a mit einer 1-Bit-Binärzahl b multiplizieren kann. Das Ergebnis soll mit m beschriftet werden. (2 P.)

b) Gegeben seien ein 8-Bit Carry-Chain-Addierer und ein 2-Bit Carry-Chain-Addierer (s.u.). Ergänzen Sie das Schaltbild zu einem 10-Bit Carry-Chain-Addierer. Die Eingänge sollen mit $a_9 \ldots a_0$, $b_9 \ldots b_0$ und c_0, die Ausgänge mit $s_{10} \ldots s_0$ beschriftet werden. (3 P.)

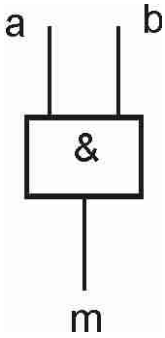


c) Geben Sie die Wertetabelle für einen 1-Bit-Subtrahierer an, d.h. für ein Schaltnetz mit Eingängen a, b, c_{in} und Ausgängen c_{out}, s, bei dem gilt $-2c_{out}+s = a-b-c_{in}$. (4 P.)

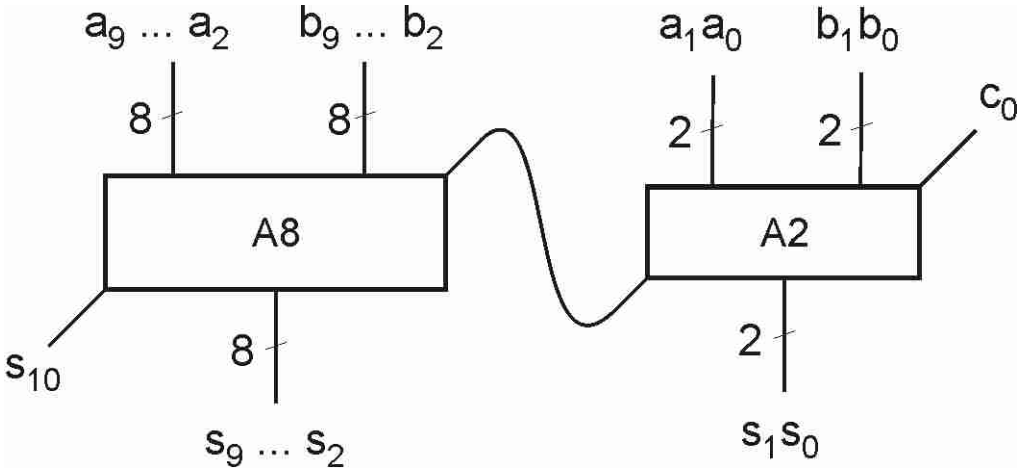
$a\ b\ c_{in}$	c_{out}	s
0 0 0		
0 0 1		
0 1 0		
0 1 1		
1 0 0		
1 0 1		
1 1 0		
1 1 1		

Lösungsvorschläge

Zu a: siehe Skizze



Zu b: siehe Skizze

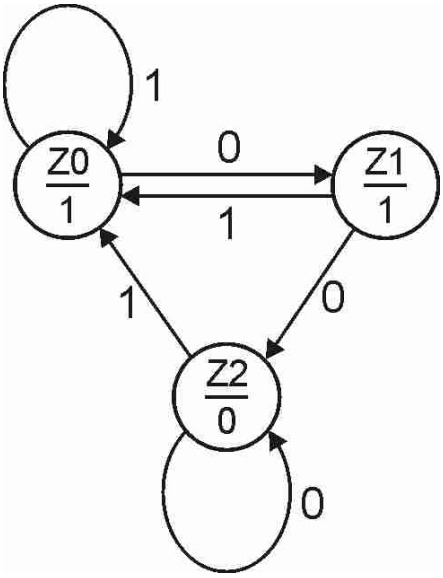


Zu c:

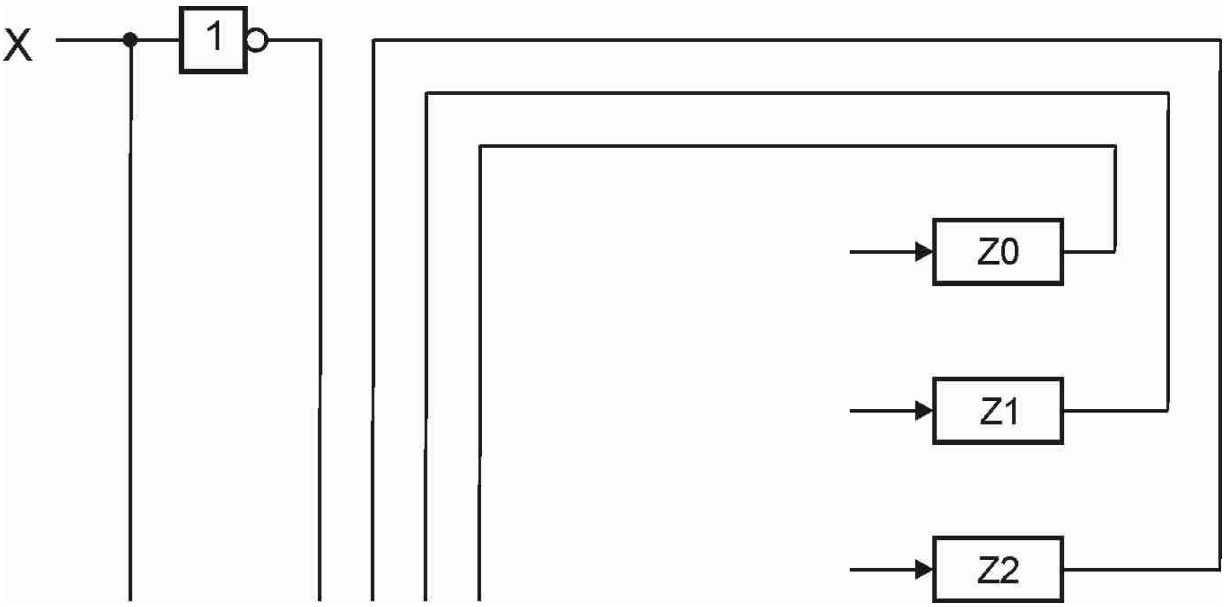
$a\ b\ c_{in}$	c_{out}	s
0 0 0	0	0
0 0 1	1	1
0 1 0	1	1
0 1 1	1	0
1 0 0	0	1
1 0 1	0	0
1 1 0	0	0
1 1 1	1	1

Aufgabe I-4 Schaltwerkssynthese und –analyse

a) Gegeben ist ein Automat mit drei Zuständen $Z0$, $Z1$, $Z2$, einem Eingangssignal X, einem Ausgangssignal Y, und dem folgenden Zustandsgraph.



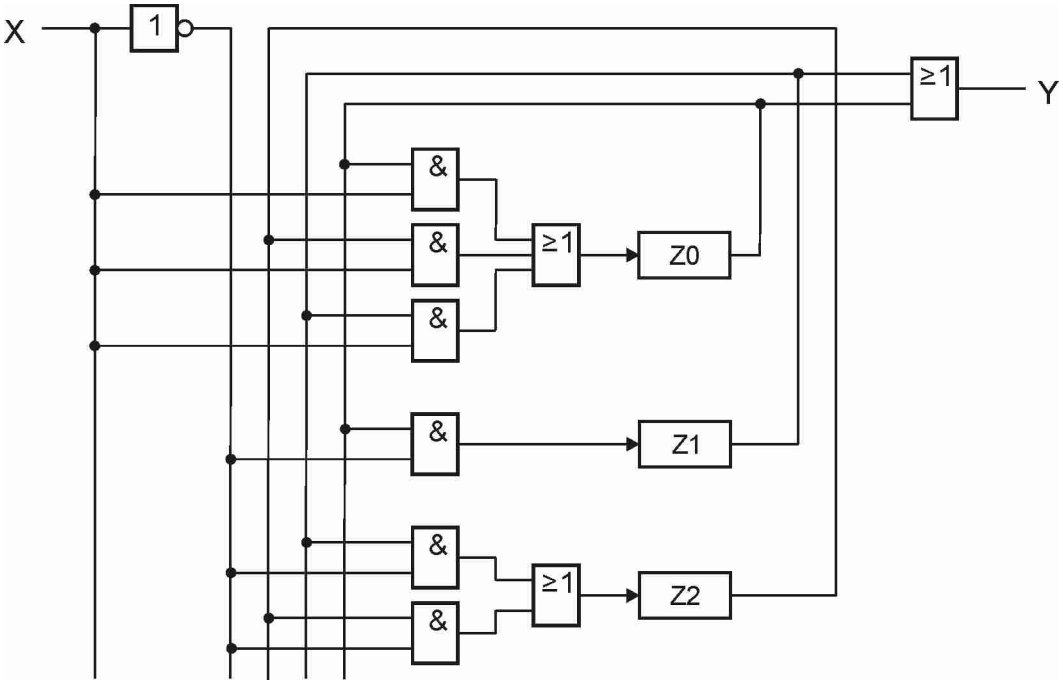
Vervollständigen Sie das Schaltwerk, bei dem eine Hot-One-Codierung gewählt wurde. Die Register sind vereinfacht als Rechtecke gezeichnet. Reset– oder Einschaltlogik braucht nicht berücksichtigt zu werden. UND– bzw. ODER-Gatter mit mehr als zwei Eingängen sind erlaubt. (4 P.)



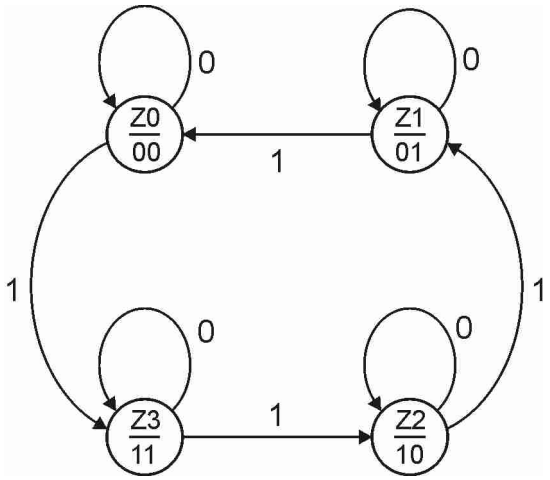
b) Gesucht ist ein Schaltwerk mit den vier Zuständen $Z0$ bis $Z3$, einem Eingangssignal X, und Ausgangssignalen Y_1Y_0. Die Ausgangssignale sollen die 2-stellige Binärcodierung des Zustands ergeben. Hat das Eingangssignal den Wert 0, dann folgt auf den Zustand Z_i wiederum der Zustand Z_i. Hat das Eingangssignal den Wert 1, dann folgt auf den Zustand Z_i der Zustand $Z_{(i+3) \bmod 4}$. Die Modulo-Operation verringert Werte, die größer oder gleich 4 sind, um 4. Zum Beispiel gilt für $i = 2$, dass $(i + 3) \bmod 4 = (2 + 3) \bmod 4 = 5 \bmod 4 = 1$ ist. Zeichnen Sie den Zustandsgraphen. (4 P.)

Lösungsvorschläge

Zu a) siehe Skizze



Zu b) siehe Skizze



Aufgabe I-5 Zustandsminimierung

Gegeben ist ein Schaltwerk mit sechs Zuständen $Z0$ bis $Z5$, einem Eingangssignal X und einem Ausgangssignal Y mit der folgenden Zustandsübergangstabelle (Z_i: aktueller Zustand, Z_i^+: Folgezustand bei Eingabe X, Y: Ausgabe). Finden Sie alle Zustandspaare (Z_i, Z_j), für die gilt: Bei jeder möglichen Eingabe sind die Ausgaben von Z_i und Z_j identisch.(4 P.)

Z_i	Z_i^+	X	Y
$Z0$	Z1	0	1
$Z0$	Z5	1	1
$Z1$	Z2	0	1
$Z1$	Z0	1	0
$Z2$	Z3	0	1
$Z2$	Z1	1	1
$Z3$	Z4	0	1
$Z3$	Z2	1	0
$Z4$	Z5	0	0
$Z4$	Z3	1	1
$Z5$	Z0	0	1
$Z5$	Z2	1	0

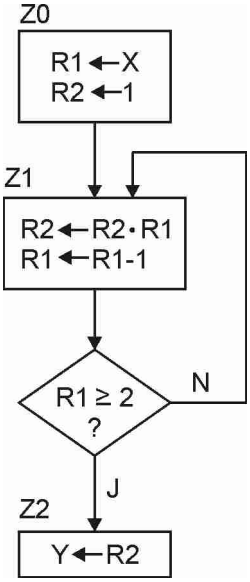
Lösungsvorschläge

Die Zustände $Z0$ und $Z2$ haben für Eingabe 0 beide Ausgabe 1, und für Eingabe 1 beide Ausgabe 1. Die Zustände $Z1$, $Z3$ und $Z5$ haben für Eingabe 0 beide Ausgabe 1, und für Eingabe 1 beide Ausgabe 0. Damit lauten die gesuchten Paare $(Z0, Z2)$, $(Z1, Z3)$, $(Z1, Z5)$ und $(Z3, Z5)$.

Aufgabe I-6 ASM-Diagramme

a) Gesucht ist ein ASM-Diagramm, das im ersten Takt zwei positive Binärzahlen (d.h. echt größer Null) aus Eingängen X und Y in Register lädt, danach das Produkt der beiden Binärzahlen berechnet, und das Produkt auf Ausgängen Z ausgibt. Danach endet das ASM-Diagramm. Alle Ein– und Ausgabesignale, Register und Operationseinheiten sollen in hinreichender Breite zur Verfügung stehen. Als Operationen stehen Additionen und Vergleicher auf Gleichheit zur Verfügung, insbesondere **keine** Multiplikation. Geben Sie das ASM-Diagramm an, wobei keine bedingten Ausgangsboxen benutzt werden sollen. (5 P.)

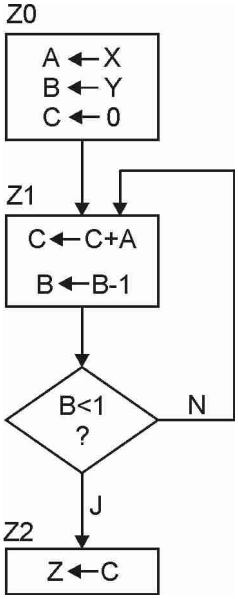
b) Gegeben ist folgendes ASM-Diagramm, wobei X die Eingabe und Y die Ausgabe darstellt. Vervollständigen Sie die Steuertabelle, wobei s_i das Steuersignal für Register R_i und m_i das Steuersignal für den Multiplexer vor Register R_i darstellt. Das Multiplexersignal m_i muss den Wert 1 in dem Zustand mit der höchsten Nummer erhalten, in dem eine Zuweisung an R_i erfolgt. In Zuständen, in denen keine Zuweisung an R_i erfolgt, muss m_i den Wert 0 erhalten, die Benutzung von don't care (X) ist nicht erlaubt. (3 P.)



Signal	$Z0$	$Z1$	$Z2$
s_1			
s_2			
m_1			
m_2			

Lösungsvorschläge

Zu a) siehe Skizze



Zu b)

Signal	$Z0$	$Z1$	$Z2$
s_1	1	1	0
s_2	1	1	0
m_1	0	1	0
m_2	0	1	0

Aufgabe I-7 Prozessoren

a) Geben Sie mögliche Gründe für die Existenz von **Nicht**-maskierbaren Interrupts zusätzlich zu maskierbaren Interrupts an. (2 P.)

b) Erklären Sie die Arbeitsweise eines Stack anhand der Wirkungsweise der Befehle PUSH und POP. (3 P.)

Lösungsvorschläge

Zu a) Einige Ereignisse können nicht warten, zum Beispiel bei drohendem Zusammenbruch der Spannungsversorgung. Der betreffende Interrupt (zum Sichern des Prozessorstatus und des Hauptspeicherinhalts auf einen nicht-flüchtigen Speicher) muss sofort ausgeführt werden, selbst wenn derzeit ein anderer, „normaler“ und damit maskierbarer Interrupt bearbeitet wird.

Zu b) Ein Stack dient zum geordneten Speichern von Daten oder Registerinhalten nach dem LIFO-Prinzip (Last-in-first-out). Hierzu gibt es ein Register namens Stackpointer SP, das die Startadresse des Stack enthält. Wir gehen im weiteren davon aus, dass der Stack im Adressraum von hinten nach vorne wächst. Der Befehl PUSH legt einen Register-Wert auf den Stack und erniedrigt den Stackpointer, so dass er auf den nächsten freien Speicherplatz zeigt:

PUSH Rx: M(SP)=Rx; SP–;

Der Befehl POP erhöht zunächst den Stackpointer, damit er wieder auf den letzten abgelegten Wert zeigt, und transferiert diesen zu einem Register:

POP Rx: SP++; Rx=M(SP);