Prof. Dr.-Ing. Jürgen Teich
Lehrstuhl für Informatik 12
(Hardware-Software-Co-Design)
Universität Erlangen-Nürnberg

Klausur
Grundlagen der Technischen Informatik

05. April 2013

Name	
Matrikelnummer	
Studienrichtung	

Aufgabe	1	2	3	4	5	Σ
max. Punkte	15	15	20	20	10	80
erreichte Punkte						
Note						

Organisatorische Hinweise

Bitte sorgfältig lesen und die Kenntnisnahme durch Unterschrift bestätigen

---

1. Bitte legen Sie Ihren Studentenausweis bereit.
2. Als Hilfsmittel sind nur Schreibmaterialien und ein beidseitig handbeschriebenes DIN A4-Blatt zugelassen.
3. Schmierpapier wird nicht abgegeben und auch nicht korrigiert.
4. Sie können bei der Aufsicht zusätzliche Bearbeitungsblätter anfordern.
5. Unleserliches wird nicht bewertet.

Erklärung

1. Im Falle einer während der Prüfung auftretenden Prüfungsunfähigkeit zeige ich dies sofort der Aufsicht an und befolge deren Anweisungen. Ich weiß, dass ich die volle Beweislast trage. Ich lasse mir das Formular des Prüfungsamts, das für diese Fälle vorgesehen ist, aushändigen und verfahre nach den dort niedergelegten Richtlinien.
2. Ich weiß, dass im Falle des Täuschungsversuchs oder der Benutzung unerlaubter Hilfsmittel („Unterschleif") der Prüfungsausschuss die Entscheidung treffen kann, die betroffene Prüfungsleistung als mit „nicht ausreichend" bewertet gelten zu lassen.
3. Ich habe die obigen Hinweise zur Kenntnis genommen.

Erlangen, den 05. April 2013 .....

Unterschrift

Einwilligung

Ich bin damit einverstanden, dass mein vorläufiges Ergebnis anonymisiert, jedoch unter Angabe der Matrikelnummer, am Mitteilungsbrett und auf der Webseite des Lehrstuhls für Informatik 12 veröffentlicht wird.

Die Bekanntgabe des vorläufigen Ergebnisses begründet keinen Rechtsanspruch.

Die Bekanntgabe des endgültigen Ergebnisses erfolgt ausschließlich durch das Prüfungsamt.

Erlangen, den 05. April 2013 .....

Unterschrift

Aufgabe 1 (Zahlensysteme) (15 Punkte)

a) Wie lautet der Wertebereich einer n Bit Binärzahl in 2er-Komplement-Darstellung? (1 Punkt)

b) Wie lautet der Wertebereich einer mit n Bit darstellbaren vorzeichenlosen Binärzahl? (1 Punkt)

c) Konvertieren Sie die Dezimalzahl 179_{10} in eine vorzeichenlose Binärzahl. (1 Punkt)

d) Konvertieren Sie die Dezimalzahl -77_{10} in eine 8 Bit breite Binärzahl in 2er-Komplement-Darstellung. (2 Punkte)

e) Konvertieren Sie die Oktalzahl 33653337357_8 in das Hexadezimalsystem. (2 Punkte)

f) Beantworten Sie folgende Auswahlfragen. Jede richtige Antwort ergibt einen Punkt, jede falsche Antwort führt zu einem Punktabzug, nicht beantwortete Fragen werden nicht gewertet, weniger als null Punkte sind nicht möglich. (4 Punkte)

1. Die Zahl $(1,1)_{10}$ lässt sich ohne Rundungsfehler in die Gleitkommadarstellung nach IEEE 754 umwandeln. ☐ wahr ☐ falsch
2. Die Zahl $(0,125)_{10}$ lässt sich ohne Rundungsfehler in die Gleitkommadarstellung nach IEEE 754 umwandeln. ☐ wahr ☐ falsch
3. Die Addition zweier Zahlen im Zweierkomplement mit 32 Bit ist aufwändiger als die Addition zweier vorzeichenloser Binärzahlen mit 32 Bits. ☐ wahr ☐ falsch
4. Eine Multiplikationseinheit für zwei vorzeichenlose 32 Bit breite Binärzahlen a und b mit einem 64 Bit breitem Ergebnis $s = a \cdot b$ kann auch für die Multiplikation zweier 32-Bit-Zahlen im Zweierkomplement verwendet werden. ☐ wahr ☐ falsch

g) Gegeben sei eine Recheneinheit (8-Bit-ALU) zur Berechnung von $r = x \mathrm{op} y$ mit den Operationen $\mathrm{op} \in \{+, -, >, \geq, <, \leq, =, \neq\}$ sowie den Booleschen Verknüpfungen $\mathrm{op} \in \wedge, \vee$ und $\neg\}$. Die Operanden x und y sowie das Ergebnis r sind dabei 8 Bit breite Binärzahlen in Zweierkomplement-Darstellung. Ein Überlauf tritt bei einer Operation auf, wenn das Ergebnis der Operation außerhalb des Wertebereichs einer 8 Bit breiten Binärzahl in Zweierkomplement-Darstellung liegt.

Geben Sie einen Test an, um zu entscheiden, ob bei der Addition mit der 8-Bit-ALU $s = a + b$ ein Überlauf aufgetreten ist. Dem Test stehen ausschließlich die 11 Operationen der 8-Bit-ALU, Konstanten darstellbar als 8 Bit breite Binärzahlen in Zweierkomplement-Darstellung, die drei Werte a, b und s, sowie beliebige Zwischenergebnisse zur Verfügung. (4 Punkte)
Ein solcher Test könnte beispielsweise wie folgt aussehen: $s \geq 7 \vee a > 2 \wedge (b \geq a)$

Aufgabe 2 (Minimierung von Schaltfunktionen) (15 Punkte)

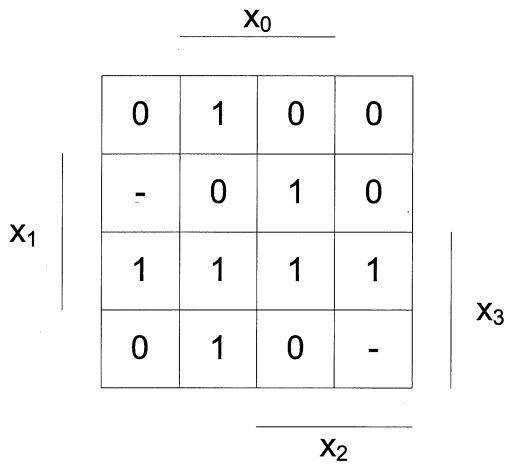
a) Vereinfachen Sie folgenden Ausdruck der Schaltfunktion $f_1(x_3,x_2,x_1,x_0) = \overline{x_3} \cdot \overline{x_1} \cdot x_0 + \overline{x_3} \cdot \overline{x_1} \cdot \overline{x_0} + x_2$ so weit wie möglich. (1 Punkt)

b) Realisieren Sie die Schaltfunktion $f_1(x_3,x_2,x_1,x_0)$ aus Teilaufgabe 2a) als CMOS-Schaltung. (3 Punkte)

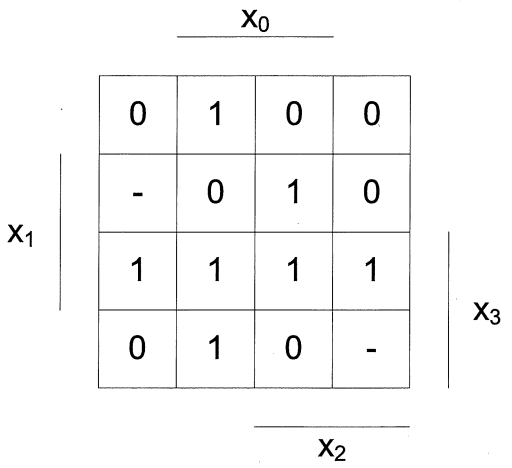
VDD

GND

c) Gegeben sei eine Schaltfunktion $f_2(x_3,x_2,x_1,x_0)$. Bestimmen sie alle Primimplikate und Primimplikanten im jeweiligen Symmetriediagramm dieser Funktion und geben Sie die entsprechenden schaltalgebraischen Ausdrücke an. (5 Punkte)



Primimplikate:



Primimplikanten:

d) Geben Sie alle KMF(s) und DMF(s) für die Schaltfunktion $f_2(x_3,x_2,x_1,x_0)$ aus Teilaufgabe 2c) an und bestimmen Sie die kostengünstigste Minimalform. Die Kostenfunktion berücksichtigt die Anzahl der notwendigen Gattereingänge. Alle Signale stehen invertiert und nicht invertiert zur Verfügung. (2 Punkte)

DMF(s):

KMF(s):

e) Gegeben sei folgende Funktionstabelle für die Schaltfunktion $f_3(x_3,x_2,x_1,x_0)$. Ermitteln Sie alle Primimplikanten mit Hilfe des Quine/McCluskey-Verfahrens. (4 Punkte)

x_3	x_2	x_1	x_0	$f_3(x_3,x_2,x_1,x_0)$
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Aufgabe 3 (Automaten und Flipflops) (20 Punkte)

a) Im Folgenden soll ein Fahrerwarnsystem entworfen werden. Dieses ist wie folgt spezifiziert: Eine Fahrspurerkennung im Auto dient dazu, das ungewollte Überfahren von Spurbegrenzungslinien zu verhindern und den Fahrer optisch zu warnen. Hierzu verfügt das System über vier verschiedene Eingaben, die jedoch nicht gleichzeitig anliegen können. Sobald das Überfahren einer Linie erkannt wurde, liefert die Eingabe *Links* die Klassifikation als Spurbegrenzung auf der linken Seite, *Rechts* hingegen repräsentiert die Spurbegrenzung auf der rechten Seite. Blinkt der Fahrer, ist das System wieder normal betriebsbereit und wartet auf eine neu erkannte Linie. Sofern direkt nach Erkennen einer Linie das *Blinken* nicht gesetzt ist, soll der Fahrer unmittelbar auf der jeweiligen Seite solange mit je einer Warnlampe auf sein Fehlverhalten hingewiesen werden, bis von Außen über *Sicher* das Beenden der Warnung und damit auch die Rückkehr in den normalen Betriebsmodus vorgegeben wird. Sollte das System direkt nacheinander sowohl links als auch rechts ein Überfahren erkennen, so dürfen, unabhängig von einem möglichen Blinken, keine Warnungen erfolgen und das System soll wieder normal betriebsbereit sein.
Die Ein- und Ausgaben sind dabei wie folgt durch binäre Variablen codiert:

Eingabe	i_1	i_0
Links	0	0
Rechts	0	1
Blinken	1	0
Sicher	1	1

Eingabe des Automaten

Ausgabe	o_1	o_0
NoWarn	0	0
WarnLinks	0	1
WarnRechts	1	0

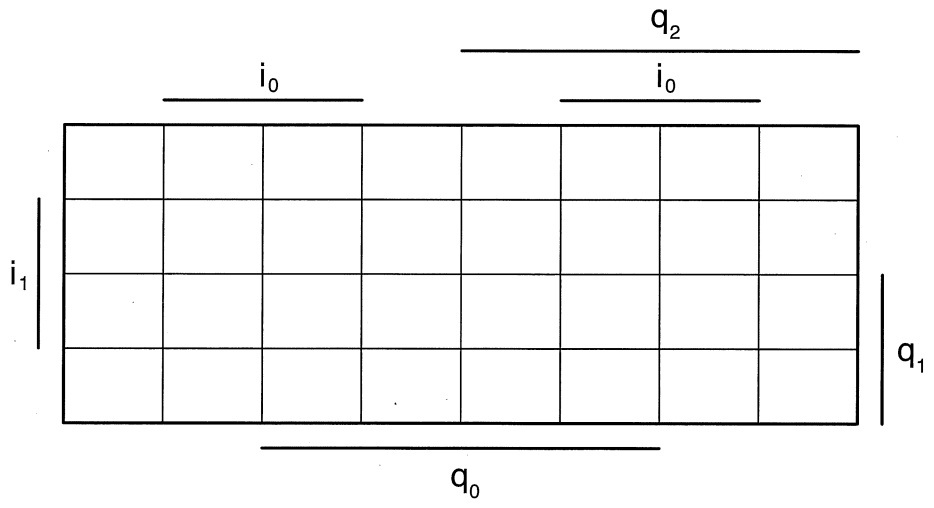
Ausgabe des Automaten

1. Entwickeln Sie das beschriebene Fahrerwarnsystem als Moore-Automat mit maximal 5 Zuständen und geben Sie den Automatengraphen an. (4 Punkte)

2. Vervollständigen Sie die nachfolgend gegebene Automatentafel des Fahrerwarnsystems unter Verwendung von taktflankengesteuerten T/D/JK-Flipflops. (6 Punkte)

Zustandsname	Aktueller Zustand			Eingabe		Nachfolgezustand			Ansteuerfunktion				Ausgabe	
	q_2	q_1	q_0	i_1	i_0	q_2'	q_1'	q_0'	T_2	D_1	J_0	K_0	o_1	o_0
normal	0	0	0	0	0									
normal	0	0	0	0	1									
normal	0	0	0	1	0									
normal	0	0	0	1	1									
	0	0	1	0	0									
	0	0	1	0	1									
	0	0	1	1	0									
	0	0	1	1	1									
	0	1	0	0	0									
	0	1	0	0	1									
	0	1	0	1	0									
	0	1	0	1	1									
	0	1	1	0	0									
	0	1	1	0	1									
	0	1	1	1	0									
	0	1	1	1	1									
	1	0	0	0	0									
	1	0	0	0	1									
	1	0	0	1	0									
	1	0	0	1	1									

3. Entwickeln Sie eine disjunktive Minimalform (DMF) der Ansteuerfunktion des JK-FlipFlops K_0 unter Verwendung des gegebenen Symmetriediagramms. Geben Sie den resultierenden Schaltalgebraischen Ausdruck an. (3 Punkte)
Achten Sie auf Don't-Cares sowie die vorgegebene Variablenordnung!



4. Zeichnen Sie das vollständige Schaltwerk des in Teilaufgabe 2. spezifizierten Moore-Automaten unter Zuhilfenahme der Ergebnisse aus Teilaufgabe 3. Zur Vereinfachung der Aufgabe sind die Schaltnetze für die Signale T_2, D_1 und J_0 nicht zu zeichnen, sondern stehen als Signale bereit. (4 Punkte)

b) Geben Sie ein Blockdiagramm eines Mealy-Automaten an. (2 Punkte)

c) Worin unterscheiden sich die drei Automatentypen Mealy, Moore und Medwedev? (1 Punkt)

Aufgabe 4 (Arithmetik) (20 Punkte)

a) In dieser Aufgabe soll die Verzögerungszeit eines 2-Bit Carry-Look-Ahead-Addierers zur Addition zweier 2-Bit Operanden $\mathbf{a} = (a_1, a_0)$ und $\mathbf{b} = (b_1, b_0)$ ermittelt werden. (14 Punkte)

1.) Geben Sie zunächst schaltalgebraische Ausdrücke für die Berechnung der *Summe* s_i an den Stellen $i = 0, 1$ und des Übertrags c_i an den Stellen $i = 1, 2$ unter Verwendung der Ausdrücke *generate* g_i und *propagate* p_i an. (2 Punkte)

2.) Zeichnen Sie das Schaltbild des 2-Bit Carry-Look-Ahead-Addierers unter ausschließlicher Verwendung von AND-, OR- und XOR-Gattern mit 2 Eingängen. Berücksichtigen Sie auch den Ausgang c_2 zur Anzeige eines Überlaufs. (4 Punkte)

3.) Nehmen Sie an, dass AND- sowie OR-Gatter eine Verzögerungszeit von $\tau\ ns$ haben. Hingegen sind XOR-Gatter mit einer Verzögerungszeit von $2\,\tau\ ns$ zu bewerten. Geben Sie die Laufzeiten zur Berechnung der Ausgänge s_0, s_1 und c_2 des Addierers an. (3 Punkte)

4.) Im weiteren Verlauf stehen nur noch $NAND$-Gatter mit 2 Eingängen zur Verfügung. Geben Sie zunächst an, wie die bisher verwendeten AND-, OR- und XOR-Gatter mit Hilfe von $NAND$-Gattern mit 2 Eingängen realisiert werden können und berechnen Sie erneut die Verzögerungen der Ausgänge s_0, s_1 und c_2 des Carry-Look-Ahead-Addieres wenn nur noch $NAND$-Gatter mit 2 Eingängen und einer Verzögerungszeit von $\tau\, ns$ zur Verfügung stehen. (6 Punkte)

b) Gegeben sind die beiden im Zweierkomplement kodierten Binärzahlen $a = 0\ 1001\ 0110$ und $b = 01\ 0101$. Demonstrieren Sie den Ablauf des Non-Performing Divisionsverfahrens indem Sie die Zahl a durch die Zahl b teilen und die einzelnen Rechenschritte explizit angeben.(5 Punkte)

Aufgabe 5 (VHDL) (10 Punkte)

Implementieren Sie einen einfachen, synchron getakteten Schaltungsblock *CAESAR* zur Verschlüsselung, bzw. Entschlüsselung von 32-bit Daten mittels der Caesar-Schiffre anhand des in Abbildung 1 dargestellten endlichen Automaten. Wird im initialen Zustand `IDLE` der ein Bit wertige Eingang `b` auf `'1'` gesetzt, so soll der Mechanismus den auf dem 32-Bit breiten Eingang `I` bereitgestellten Wert in das Register `offset` speichern und in den Zustand `STORE` wechseln. Bleibt im Zustand `STORE` der Eingang `b` auf `'1'`, so sollen weiterhin die auf `I` bereitgestellten Werte in das Register übernommen werden. Wird hingegen im Zustand `STORE` der Eingang `b` auf `'0'`, der Eingang `e` auf `'1'` und der Eingang `d` auf `'0'` gesetzt, so soll in den Zustand `ENC` gewechselt werden und die auf `I` bereitgestellten Daten durch Addition des in `offset` gespeicherten Wertes verschlüsselt und auf dem Ausgang `J` ausgegeben werden. Die Operation wird so lange wiederholt, so lange sich an der Eingangskombination aus `b`, `e` und `d` nichts ändert. Wird hingegen an den Eingängen eine andere Kombination angelegt, so soll der Automat wieder in den initialen Zustand wechseln. Weiterhin soll aus dem Zustand `STORE` durch Anlegen der Kombination aus `b = '0'`, `e = '0'` und `d = '1'` in den Zustand `DEC` gewechselt und die auf `I` zur Verfügung gestellten Daten durch Subtraktion des in `offset` gespeicherten Wertes entschlüsselt werden. Im Zustand `DEC` soll, analog zum Zustand `ENC`, die Entschlüsselung so lange fortgesetzt werden, so lange sich an der Kombination aus den drei Eingängen `b`, `e` und `d` nichts ändert. Im Falle einer Änderung soll wieder in den initialen Zustand `IDLE` gewechselt werden. Durch die verbleibenden Kombinationen soll aus dem Zustand `STORE` wieder in den Zustand `IDLE` gewechselt werden. Darüber hinaus soll das Rücksetzen des Automaten aus jedem der drei Zustände `STORE`, `ENC` und `DEC` in den Initialzustand `IDLE` möglich sein.

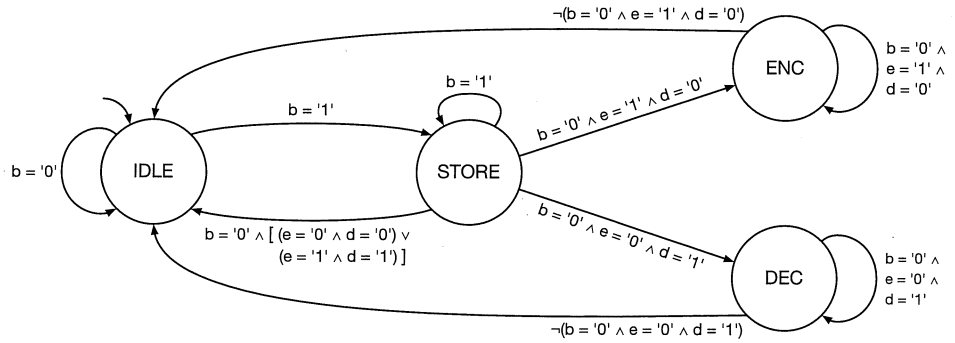


Abbildung 1: Zustandsautomat des Schaltungsblocks *CAESAR*.

a) Geben Sie eine Schnittstellenbeschreibung des oben beschriebenen Schaltungsblocks `CAESAR` in Form einer Entity in VHDL an. Verwenden Sie hierzu das vorgegebene Code-Skelett sowie die Datentypen `std_logic` und `std_logic_vector`. (*Auf die Angabe der benötigten Bibliotheken der IEEE kann hierbei verzichtet werden.*) (3 Punkte)

```
entity CAESAR is




end entity;
```

b) Geben Sie eine Implementierung des Schaltungsblockes *CAESAR* in Form einer VHDL Architecture-Beschreibung an. Verwenden Sie hierzu das vorgegebene Code-Skelett. (*Auf die Angabe der benötigten Bibliotheken der IEEE kann hierbei verzichtet werden.*) (7 Punkte)

```
architecture behave of CAESAR is

end architecture;
```