Universität des Saarlandes
FR 6.2 - Informatik
Prof. Dr. W.J. Paul
Dipl. Inform. M. Klein
Dipl. Inform. T. In der Rieden

2. Teilklausur Informatik II

(Datum: 09.08.2003)

---

1. **Aufgabe** **(4 + 6 Punkte)**

   Beweise:

   (a) $\sum_{i=0}^{n} i = \frac{n \cdot (n+1)}{2}$

   (b) $\sum_{i=0}^{n} i^2 = \frac{n \cdot (n+1) \cdot (2n+1)}{6}$

2. **Aufgabe** **(10 Punkte)**

   Seien $a, b \in \{0,1\}^n$. Wir definieren:

   $a <_{lex} b \Leftrightarrow \exists\, i \in \{0, ..., n-1\} :\ a[n-1 : i+1] = b[n-1 : i+1]$ und $a_i < b_i$.

   Beweise: $\langle a \rangle < \langle b \rangle \Leftrightarrow a <_{lex} b$.

3. **Aufgabe** **(3 + 3 Punkte)**

   Seien $a \in \{0,1\}^n$. Beweise:

   (a) $\langle \overline{a} \rangle = -\langle a \rangle - 1 \pmod{2^n}$.

   (b) $\langle a \rangle = [a] + a_{n-1} \cdot 2^n$.

4. **Aufgabe** **(5 Punkte)**

   Seien $x_1, ..., x_n \in \{0,1\},\ n \geq 2$.

   Beweise: $\overline{\bigvee_{i=1}^{n} x_i} \equiv \bigwedge_{i=1}^{n} \overline{x_i}$

5. **Aufgabe** **(5 Punkte)**

   Für $a, x \in \{0,1\}^n$ definieren wir $c(a) := \bigvee_{i=1}^{n} x_i^{\overline{a_i}}$

   $T(f)$ sei der Träger von f.

   Beweise den Darstellungssatz für konkunktive Normalformen:

   Für jede n-stellige Schaltfunktion f gilt

   $$f(x_1, ..., x_n) \equiv \bigwedge_{a \notin T(f)} c(a)$$

6. **Aufgabe** **(3 Punkte)**

   Gib einen booleschen Ausdruck an, der aus den Inhalten von IR, PC und MAR das Signal *misaligned* berechnet.

Universität des Saarlandes
FR 6.2 - Informatik
Prof. Dr. W.J. Paul
Dipl. Inform. M. Klein
Dipl. Inform. T. In der Rieden

2. Teilklausur Informatik II

(Datum: 09.08.2003)

---

7. **Aufgabe** **(5 + 5 Punkte)**

Def.: Ein kombinierter n-Incrementer/Decrementer ist ein Schaltkreis mit Eingängen $a \in \{0,1\}^n,\ dec \in \{0,1\}$ und Ausgängen $b \in \{0,1\}^n$, so dass gilt:

$$\langle b \rangle = \begin{cases} \langle a \rangle + 1 \pmod{2^n}\ ;\ dec = 0 \\ \langle a \rangle - 1 \pmod{2^n}\ ;\ dec = 1 \end{cases}$$

(a) Baue einen kombinierten n-Incrementer/Decrementer aus einem n-Incrementer und maximal $2n$ *xor*-Gattern.

(b) Beweise die Korrektheit deiner Konstruktion. Dabei darf die Korrektheit des Incrementers vorausgesetzt werden.

8. **Aufgabe** **(6 + 12 Punkte)**

Seien $a, b \in \{0,1\}^n;\ s, t \in \{0,1\}^{n+1};\ n = 2^k$. Zur Erinnerung: Ein n-Bit Modified Conditional Sum Addierer berechnet die Schaltfunktion $MCSA_n : (\langle s \rangle, \langle t \rangle) := (\langle a \rangle + \langle b \rangle, \langle a \rangle + \langle b \rangle + 1)$.

Gegeben sei ein n-Bit Carry Lookahead Addierer, dessen Carry-In Signal fest auf 0 gezogen ist.

(a) Erweitere diesen Schaltkreis unter Verwendung der Propagate-Ausgänge zu einem n-Bit MCSA. Die zusätzlichen Kosten dürfen maximal $n + 1$ betragen.

(b) Beweise die Korrektheit deiner Konstruktion. Dabei darf die Korrektheit des Carry Lookahead Addierers vorausgesetzt werden.

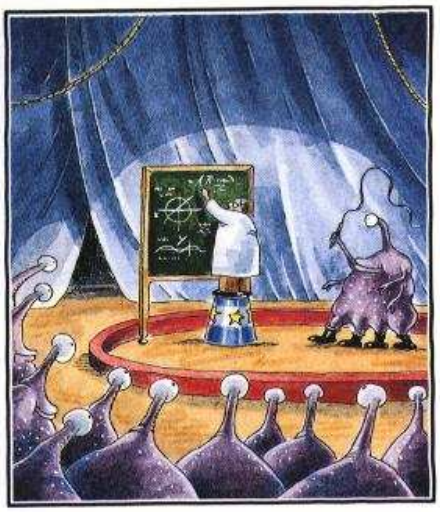
9. **Aufgabe** **(2 + 2 + 2 Punkte)**

Die Interrupts der DLX können in verschiedene Kategorien eingeteilt werden. Erkläre mit wenigen Worten die folgenden in diesem Zusammenhang auftretenden Begriffe und gib jeweils ein Beispiel für einen Interrupt dieser Kategorie:

(a) maskierbar

(b) repeat

(c) continue



Abducted by an alien circus company, Professor Doyle is forced to write calculus equations in center ring.

Universität des Saarlandes
FR 6.2 - Informatik
Prof. Dr. W.J. Paul
Dipl. Inform. M. Klein
Dipl. Inform. T. In der Rieden

2. Teilklausur Informatik II

(Datum: 09.08.2003)

10. **Aufgabe** **(5 + 5 Punkte)**

Du bist auf einer einsamen Insel gestrandet. Um nicht selber die ganze Zeit *SOS* funken zu müssen konstruierst du aus zufällig vorhandenen Einzelteilen einen Automaten. Dieser soll folgende Eigenschaften besitzen:

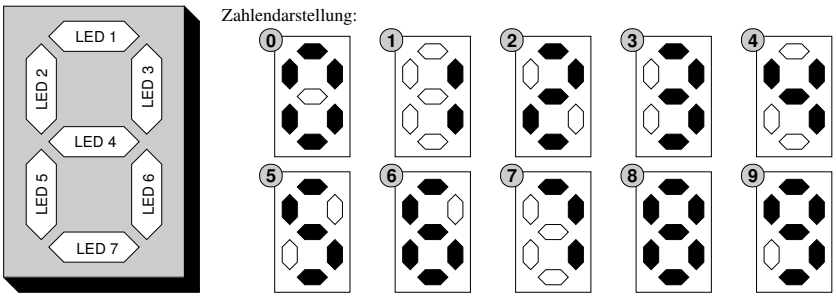
- Der momentane Zustand des Automaten ist binär codiert.
- Der Automat hat die Eingangssignale $start, stop \in \{0,1\}$ und die Ausgangssignale $S, O \in \{0,1\}$.
- Sobald *start* aktiviert wurde soll der Automat immer wieder die Folge $S\ O\ S$ ausgeben.
- Sobald *stop* aktiviert wurde sendet der Automat, sofern er gerade am senden ist, die aktuelle Folge noch fertig und springt dann in den Startzustand.

(a) Spezifiziere den Automaten durch einen Graphen.

(b) Gib einen booleschen Ausdruck an, der die Next State Funktion berechnet.

11. **Aufgabe** **(6 + 4 Punkte)**

Zu entwerfen ist ein kaskadierbarer Zählerautomat für 7-Segment-Anzeigen.



Der Automat hat die Eingangssignal $reset, c_{in} \in \{0,1\}$ und die Ausgangssignale $LED[1{:}7] \in \{0,1\}^7$ und $c_{out} \in \{0,1\}$.

Sei $led: \{0,...,9\} \to \{0,1\}^7$ die durch obige Abbildung definierte Funktion, die Ziffern ihre 7-Segment-Darstellung zuordnet mit weiss = 0 und schwarz = 1.

Für den Automaten soll gelten:

- Der momentane Zustand z des Automaten ist binär codiert.
- Für $z' = \delta(z, reset, c_{in})$ soll gelten:

$$\langle z' \rangle = \begin{cases} 0 & ;\ reset = 1 \\ \langle z \rangle + 1 \pmod{10} & ;\ reset = 0 \wedge c_{in} = 1 \\ \langle z \rangle & ;\ sonst \end{cases}$$

- $c_{out} = 1 \Leftrightarrow \langle z \rangle = 9 \wedge reset = 0 \wedge c_{in} = 1$
- $LED[1{:}7] = led(\langle z \rangle)$.

(a) Spezifiziere den Automaten durch einen Graphen.

(b) Gib boolesche Ausdrücke an, die die Ausgabefunktionen für $c_{out}, LED2, LED4$ und $LED6$ berechnen.

Universität des Saarlandes
FR 6.2 - Informatik
Prof. Dr. W.J. Paul
Dipl. Inform. M. Klein
Dipl. Inform. T. In der Rieden

2. Teilklausur Informatik II

(Datum: 09.08.2003)

---

12. **Aufgabe** **(8 Punkte)**

Schreibe ein RTL-Programm, das eine vorzeichenbehaftete Integer-Division ausführt (ganzzahliger Anteil und Divisionsrest). Dabei soll gelten:

- Die Source-Operanden stehen in den Registern `R1` und `R2` mit $[\texttt{R2}] \neq 0$.
- Das Programm berechnet `R3` und `R4` mit $[\texttt{R4}] < 0 \Leftrightarrow [\texttt{R1}] < 0$ und $|[\texttt{R4}]| < |[\texttt{R2}]|$, so dass gilt: $[\texttt{R1}] = [\texttt{R3}] \cdot [\texttt{R2}] + [\texttt{R4}]$.

13. **Aufgabe** **(8 Punkte)**

Die Fibonacci-Zahl $F(n),\ n \in \mathbb{N}$ berechnet sich nach der Formel:

$F(0) = 0;\ F(1) = 1;\ F(n) = F(n-2) + F(n-1)$ für $n \geq 2$.

In Speicherzelle 1000 stehe der Wert $a \in \{0,1\}^8,\ \langle a \rangle \leq 46$.

Schreibe ein RTL-Programm, dass die Fibonacci-Zahl $F(\langle a \rangle)$ berechnet und in Register `R1` ablegt.

14. **Aufgabe** **(3 + 4 Punkte)**

(a) Schreibe ein RTL-Programm, das in jede Speicherzelle von 1024 bis 32767 die Adresse der jeweiligen Speicherzelle modulo 2^8 als neuen Inhalt hineinschreibt.

(b) Schreibe ein Programm, das das gleiche tut wie das Programm aus Teil (a). Dabei gelten jedoch folgende Voraussetzungen:

- Es sind nur absolute Adressierungen erlaubt, d.h. bei allen Speicherzugriffen berechnet sich der *mem*-Teil als (`R0 + imm`).
- Gehe davon aus, dass es einen gemeinsamen Speicher für Instruktionen und Daten gibt und dass das Programm ab Speicherzelle 0 in diesem Speicher liegt.

15. **Aufgabe** **(4 Punkte)**

Was tut folgendes Programm? Beweise die Korrektheit deiner Antwort.

```
R1 = R2 xor R1
R2 = R1 xor R2
R1 = R2 xor R1
```

Universität des Saarlandes
FR 6.2 - Informatik
Prof. Dr. W.J. Paul
Dipl. Inform. M. Klein
Dipl. Inform. T. In der Rieden

2. Teilklausur Informatik II

(Datum: 09.08.2003)

IR[31 : 26]		Mnem.	d	Effect
Data Transfer, mem = M[RS1 + imm]				
100000	0x20	lb	1	RD=Sext(mem)
100001	0x21	lh	2	RD=Sext(mem)
100011	0x23	lw	4	RD=mem
100100	0x24	lbu	1	RD=0^{24}mem
100101	0x25	lhu	2	RD=0^{16}mem
101000	0x28	sb	1	mem=RD[7 : 0]
101001	0x29	sh	2	mem=RD[15 : 0]
101011	0x2b	sw	4	mem=RD
Arithmetic, Logical Operation				
001000	0x08	addi		RD=RS1 + imm
001001	0x09	addiu		RD=RS1 + imm (no overflow)
001010	0x0a	subi		RD=RS1 - imm
001011	0x0b	subiu		RD=RS1 - imm (no overflow)
001100	0x0c	andi		RD=RS1 $\wedge$ imm
001101	0x0d	ori		RD=RS1 $\vee$ imm
001110	0x0e	xori		RD=RS1 $\oplus$ imm
001111	0x0f	lhgi		RD=imm 0^{16}
Test Set Operation				
011000	0x18	clri		RD=(false ? 1 : 0)
011001	0x19	sgri		RD=(RS1 > imm ? 1 : 0)
011010	0x1a	seqi		RD=(RS1 = imm ? 1 : 0)
011011	0x1b	sgei		RD=(RS1 $\geq$ imm ? 1 : 0)
011100	0x1c	slsi		RD=(RS1 < imm ? 1 : 0)
011101	0x1d	snei		RD=(RS1 $\neq$ imm ? 1 : 0)
011110	0x1e	slei		RD=(RS1 $\leq$ imm ? 1 : 0)
011111	0x1f	seti		RD=(true ? 1 : 0)
Control Operation				
000100	0x04	beqz		PC=PC+(RS1 = 0 ? imm : 4)
000101	0x05	bnez		PC=PC+(RS1 $\neq$ 0 ? imm : 4)
000110	0x06	jr		PC=RS1
000111	0x07	jalr		R31=PC+4; PC = RS1

Tabelle 1: I-type instruction layout

Universität des Saarlandes
FR 6.2 - Informatik
Prof. Dr. W.J. Paul
Dipl. Inform. M. Klein
Dipl. Inform. T. In der Rieden

2. Teilklausur Informatik II

(Datum: 09.08.2003)

IR[31 : 26]		IR[5 : 0]		Mnem.	Effect
Shift Operation					
000000	0x00	000000	0x00	slli	RD=RS1$\ll$SA
000000	0x00	000001	0x01	slai	RD=RS1$\ll$SA (arith.)
000000	0x00	000010	0x02	srli	RD=RS1$\gg$SA
000000	0x00	000011	0x03	srai	RD=RS1$\gg$SA (arith.)
000000	0x00	000100	0x04	sll	RD=RS1$\ll$RS2[4:0]
000000	0x00	000101	0x05	sla	RD=RS1$\ll$RS2[4:0] (ar.)
000000	0x00	000110	0x06	srl	RD=RS1$\gg$RS2[4:0]
000000	0x00	000111	0x07	sra	RD=RS1$\gg$RS2[4:0] (ar.)
Data Transfer					
000000	0x00	010000	0x10	movs2i	RD=SA
000000	0x00	010001	0x11	movi2s	SA=RS1
Arithmetic, Logical Operation					
000000	0x00	100000	0x20	add	RD=RS1+RS2
000000	0x00	100001	0x21	addu	RD=RS1+RS2 (no overfl.)
000000	0x00	100010	0x22	sub	RD=RS1-RS2
000000	0x00	100011	0x23	subu	RD=RS1-RS2 (no overfl.)
000000	0x00	100100	0x24	and	RD=RS1 $\wedge$ RS2
000000	0x00	100101	0x25	or	RD=RS1 $\vee$ RS2
000000	0x00	100110	0x26	xor	RD=RS1 $\oplus$ RS2
000000	0x00	100111	0x27	lhg	RD=RS2[15:0] 0^{16}
Test Set Operation					
000000	0x00	101000	0x28	clr	RD=(false ? 1 : 0)
000000	0x00	101001	0x29	sgr	RD=(RS1 $>$ RS2 ? 1 : 0)
000000	0x00	101010	0x2a	seq	RD=(RS1 $=$ RS2 ? 1 : 0)
000000	0x00	101011	0x2b	sge	RD=(RS1 $\geq$ RS2 ? 1 : 0)
000000	0x00	101100	0x2c	sls	RD=(RS1 $<$ RS2 ? 1 : 0)
000000	0x00	101101	0x2d	sne	RD=(RS1 $\neq$ RS2 ? 1 : 0)
000000	0x00	101110	0x2e	sle	RD=(RS1 $\leq$ RS2 ? 1 : 0)
000000	0x00	101111	0x2f	set	RD=(true ? 1 : 0)

Tabelle 2: R-type instruction layout

IR[31 : 26]		Mnem.	Effect
Control Operation			
000010	0x02	j	PC = PC + imm
000011	0x03	jal	R31 = PC + 4; PC = PC + imm
111110	0x3e	trap	trap = 1; EDATA = imm;
111111	0x3f	rfe	SR = ESR; PC' = EPC; DPC = EDPC

Tabelle 3: J-type instruction layout