Name:	Vorname:

Prof. Dr. B. Seeger
Martin Schneider

Klausur zur Informatik IIIa

WS 99/00

Beginn: 8:15 Uhr
Ende: 10:45 Uhr

Bitte in Druckschrift ausfüllen:

Nachname:	
Vorname:	
Fachbereich:	
Matrikelnummer:	
Geheimwort:	

Tragen Sie auf jedes Blatt Ihren vollständigen Namen ein.

Falls der Platz für die Antworten nicht ausreicht: Geben Sie eine deutliche Referenz auf ein Zusatzblatt und versehen Sie das Zusatzblatt mit Ihrem vollständigen Namen.

Es sind keine Hilfsmittel erlaubt.

Falls Sie eine Frage haben, so wenden Sie sich bitte *leise* an einen der Tutoren.

Beachten Sie: An etlichen Stellen in der Klausur sollen Sie vorgegebene Tabellen ausfüllen. Nicht in allen Fällen ist es notwendig, *sämtliche* Zeilen und Spalten dieser Tabellen auszufüllen!

Bitte nicht ausfüllen:

Aufgabe:	1	2	3	4	5	6	7	8	Σ
Maximum:	13	24	23	16	16	16	24	28	160
Punkte:									

Name:	Vorname:

Aufgabe 1: Rechneraufbau

Aufgabe 1a: Speichersysteme

Finden Sie vier Kriterien, nach denen Sie die Speichersysteme Festplatte, RAID-Array, Register, RAM, Cache, Diskette, CD-ROM, CD-Writer und DVD-ROM sinnvoll klassifizieren können!

Ordnen Sie die oben angegebenen Speichersysteme bezüglich jedes Ihrer Kriterien (ab und zu ist ein „=“ oder ein „≤“ sinnvoller als ein „<“)! Beschreiben Sie in Zweifelsfällen, was Sie genau unter „<“ verstehen!

Kriterium

_____: _____ < _____ < _____ < _____ < _____

< _____ < _____ < _____ < _____

_____: _____ < _____ < _____ < _____ < _____

< _____ < _____ < _____ < _____

_____: _____ < _____ < _____ < _____ < _____

< _____ < _____ < _____ < _____

_____: _____ < _____ < _____ < _____ < _____

< _____ < _____ < _____ < _____

Aufgabe 1b: Binärsystem

Welche Vor- und Nachteile hat es, dass moderne Computer intern im Binärsystem und nicht im Dezimalsystem rechnen?

Name:	Vorname:

Aufgabe 2: Zahldarstellung

Aufgabe 2a: verschiedene Zahlsysteme

Vervollständigen Sie die folgende Tabelle. Pro Zeile ist ein und dieselbe *natürliche* Zahl in verschiedenen Zahlsystemen dargestellt. Die Addition wird pro Spalte durchgeführt.

	binär	oktal	hexadezimal
	110101011011		
+			AFFE
=			

Aufgabe 2b: Zweierkomplementzahlen

Schreiben Sie die Zahl -87 als 1-Byte-Zweierkomplementzahl und als 2-Byte-Zweierkomplementzahl.

-87 als 1-Byte-Zahl:

-87 als 2-Byte-Zahl:

Aufgabe 2c: Zweierkomplement

Welche Vorteile hat die Zweierkomplementdarstellung im Vergleich zur Darstellung mit Vorzeichen?

Erweitern Sie die Zweierkomplementdarstellung auf beliebige Basen! Achten Sie darauf, dass ähnlich viele positive wie negative Zahlen darstellbar sein sollen. Geben Sie eine Definition und ein Beispiel an!

Name:	Vorname:

Aufgabe 2d: Gleitkommazahlen

Interpretieren Sie die Bitkette 0 10000011 01001010000000000000000 als 32-Bit IEEE Gleitkommazahl. Wie lautet die Dezimaldarstellung?

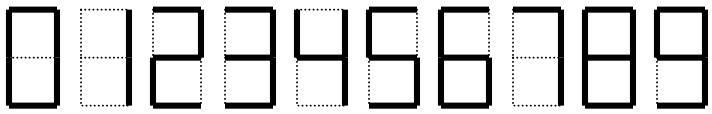
Erklären Sie den Sinn der Normierung bei Gleitkommazahlen!

Betrachtet werde das folgende 16-Bit Gleitkommaformat: 1 Bit Vorzeichen, 5 Bit Exponent , 10 Bit Mantisse. Bestimmen Sie die kleinste und die größte Zahl >0, die damit darstellbar ist! Geben Sie unter der vereinfachenden Annahme $2^{10} \approx 1000$ die ungefähre Größenordnung der gefundenen Zahlen an ($z \approx 10^x$)!

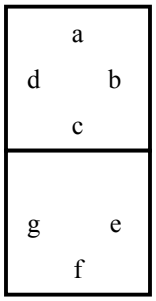
Name:	Vorname:

Aufgabe 3: Boolesche Schaltungen

Eine boolesche Schaltung habe zwei Eingänge i_0 und i_1 und sieben Ausgänge a bis g. Jeder Ausgang schalte ein Segment einer 7-Segment-Dezimalanzeige wie es in der Abbildung rechts dargestellt ist. Die darzustellenden Ziffern haben dabei die folgende Gestalt:



Wenn an der Schaltung binär 0, ..., 3 anliegt (i_0 sei das niederwertigste Bit der Binärdarstellung), so soll auf der Anzeige eine 0, ..., 3 erscheinen.



Aufgabe 3a: Schalttabelle

Stellen Sie die Schalttabelle auf!

Aufgabe 3b: Terme

Gewinnen Sie daraus möglichst einfache Terme für a, ..., g!

Aufgabe 3c:

Stellen Sie die Schaltung mit den logischen Gattern AND, OR und NOT dar!

Name:	Vorname:

Aufgabe 3d:

Läßt sich diese Schaltung allein mit AND- und OR-Gattern aufbauen? Begründen Sie Ihre Antwort!

Aufgabe 3e:

Die Schaltung soll nun auf 4 Eingänge erweitert werden (i_0 bis i_3), wobei nur die Binärzahlen 0-9 als gültige Eingaben betrachtet werden müssen. Wie sieht der Term für g (in disjunktiver Normalform, allerdings ohne die Terme für die Binärzahlen 10-15) aus?

Aufgabe 3f:

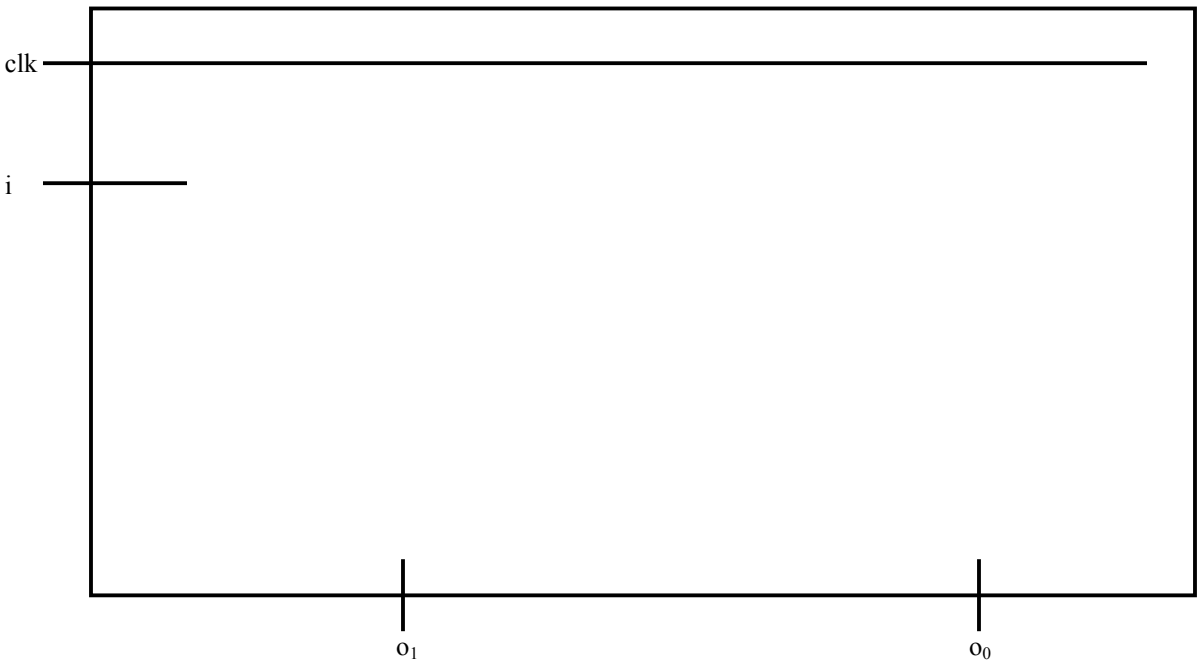
Vereinfachen Sie den Term von g unter Verwendung der Implikations- und Verschmelzungsmethode! Geben Sie die Menge aller Primimplikanten an! Ist diese Menge minimal?

Name:	Vorname:

Aufgabe 4: Bauteile

Aufgabe 4a: Schieberegister

Entwerfen Sie ein 2-Bit-Schieberegister mit Hilfe von RS-NOR-Flip-Flops, dessen Inhalt in jedem Taktzyklus um genau eine Bitposition nach rechts geschoben wird (Vorsicht: nicht mehr als eine Position pro Takt!). Das höchstwertige Bit erhalte in jedem Taktzyklus (wenn clk=1) den Wert, der am Eingang i anliegt. Der momentane Zustand des Registers soll über die Ausgänge o_0 und o_1 nach außen geführt werden. Achten Sie darauf, dass die Ausgänge zu jeder Zeit sinnvolle Werte besitzen.



Aufgabe 4b: Multiplexer

Mittels eines PLAs soll ein 2-Bit-Multiplexer realisiert werden. Er verfügt über sechs Eingänge (s_0, s_1 und i_0 bis i_3) und einen Ausgang o. Er legt den Wert i_j auf den Ausgang, wenn $j = 2s_1 + s_0$ ist.

Füllen Sie das untenstehende PLA aus. Verwenden Sie hierbei folgende Bezeichnungen:
Identer: leere Zelle, Addierer: 1, Multiplizierer: 2, Negat-Multiplizierer: 3.

i_0									
i_1									
i_2									
i_3									
s_0									
s_1									
									o

Name:	Vorname:

Aufgabe 5: Boolesche Algebra

Bei dieser Aufgabe geht es um boolesche Algebra. Geben Sie in jedem Schritt die **Formel** an, die Sie benutzt haben (**keine Wertetabellen**).

Aufgabe 5a: Vollständigkeit

Die Implikation „→“ ist definiert als x'+y. Stellen Sie mittels „→“ und „0“ Konjunktion, Disjunktion und Negation dar! Beweisen Sie Ihr Ergebnis mit den Regeln der booleschen Algebra.

Aufgabe 5b: Normalformen

Formen Sie den Term

$$T = (x+y)\,(zx')' + yz\,(x+y')$$

unter Benutzung der Regeln der booleschen Algebra in die konjunktive und die disjunktive Normalform um.

Ergebnis:

Konjunktive Normalform: _____

Disjunktive Normalform: _____

Name:	Vorname:

Aufgabe 6: Mikrocode

Aufgabe 6a: Mikrocode Befehl

Implementieren Sie den Mikrobefehl Jump Overflow (JO adr), der zu dem Befehl an Adresse adr (2 Bytes) verzweigt, falls das Overflow-Flag gesetzt ist. Implementieren Sie den Befehl so, dass er mit dem in der Vorlesung vorgestellten Load-Increment-Execute-Zyklus zusammenarbeitet.
Auf dem Beiblatt *Mikrocode-Simulator* finden Sie eine Liste der benötigten ALU-Function-Codes und einen Überblick zur Architektur der CPU. Es genügt, die Einsen einzutragen (ein leeres Kästchen wird als Null interpretiert). Kommentieren Sie Ihre Lösung ausführlich, insbesondere indem Sie beschreiben, was in den einzelnen Phasen abläuft.

Maschinenwort 1: Adresse: .....

Sm	MCN	ALU-FC	X-Bus

Y-Bus	Z-Bus	IO Schalter	Md	Ft

Phase 1: Phase 2: Phase 3:

Maschinenwort 2: Adresse: .....

Sm	MCN	ALU-FC	X-Bus

Y-Bus	Z-Bus	IO Schalter	Md	Ft

Phase 1: Phase 2: Phase 3:

Maschinenwort 3: Adresse: .....

Sm	MCN	ALU-FC	X-Bus

Y-Bus	Z-Bus	IO Schalter	Md	Ft

Phase 1: Phase 2: Phase 3:

Maschinenwort 4: Adresse: .....

Sm	MCN	ALU-FC	X-Bus

Y-Bus	Z-Bus	IO Schalter	Md	Ft

Phase 1: Phase 2: Phase 3:

Name:	Vorname:

Maschinenwort 5: Adresse: .....

Sm	MCN	ALU-FC	X-Bus

Y-Bus	Z-Bus	IO Schalter	Md	Ft

Phase 1: Phase 2: Phase 3:

Maschinenwort 6: Adresse: .....

Sm	MCN	ALU-FC	X-Bus

Y-Bus	Z-Bus	IO Schalter	Md	Ft

Phase 1: Phase 2: Phase 3:

Aufgabe 6b:

Wie unterscheiden sich Mikroprogramme im ROM von Programmen im RAM?

Wie viele MB Hauptspeicher könnten adressiert werden, wenn das Adressregister MAR 24 Bit breit wäre?

Name:	Vorname:

Aufgabe 7: Assembler

In dieser Aufgabe wird die Assembler-Sprache des Intel 80386 bzw. kompatiblen CPUs zugrunde gelegt.

Aufgabe 7a: Prozeduren und Makros

Was ist der Unterschied zwischen einer Prozedur und einem Makro?

Aufgabe 7b: Flags

In den CPU-Registern BL und CL stehen die hexadezimalen Zahlen 80 und 01. Welche Flags sind nach der Ausführung der folgenden Operationen gesetzt? Tragen Sie eine „1“ ein, wenn das Flag nach der Operation gesetzt ist, eine „0“, wenn es nicht gesetzt ist.

SHL BL,CL

Overflow	Sign	Zero	Carry

CMP CL,BL

Overflow	Sign	Zero	Carry

NEG BL

Overflow	Sign	Zero	Carry

Aufgabe 7c: Assembler-Befehle

Lösen Sie die folgenden Aufgaben mit möglichst wenigen Assembler-Befehlen.

i) Löschen Sie Bit 3 und Bit 4 in Register BL.

ii) Springen Sie an die Marke `Next`, falls AX < BX (als vorzeichenbehaftete Zahlen) und CX!=DX.

iii) Springen Sie an die Marke `Gesetzt`, falls eines der Bits 0, 3 oder 6 in AL gesetzt ist (AL darf verändert werden).

Name:	Vorname:

Aufgabe 7d: Eine Assembler-Prozedur

An der symbolischen Adresse *Data1* sei eine Folge von n Bytes gespeichert. Dieser Block soll nun an die Adresse *Data2* kopiert werden, wobei keine Annahmen bezüglich der Adressen gemacht werden sollen (insbesondere könnte *Data2* innerhalb des Blocks von *Data1* liegen).

Schreiben Sie eine Assembler-Prozedur „Copy“, welche die zwei Adressen und die Blocklänge (4 Bytes) auf dem Stack erhält und den Speicherbereich von *Data1* auf *Data2* kopiert. Das Format der Adresszeiger ist dabei freigestellt (im Datensegment oder beliebig). Es darf 386er-Code geschrieben werden.

Achten Sie darauf, dass Ihre Prozedur keine Seiteneffekte hat, d.h. die Register nach einem Prozeduraufruf die selben Werte haben wie vor dem Aufruf. Kommentieren Sie Ihren Code!

Name:	Vorname:

Aufgabe 8: RISC

Aufgabe 8a: Eigenschaften von RISC- und CISC-Prozessoren

Beschreiben Sie die wesentlichen Merkmale der RISC-Technologie!

Nennen Sie Vor- und Nachteile einer langen Pipeline mit vielen Stufen im Vergleich zu einer kürzeren!

Welche Bedeutung hat die Cache-Technologie für RISC-Prozessoren?

Erläutern Sie Sinn und Funktionsweise des Delayed Branching! Gibt es Nachteile dieses Konzepts?

Aufgabe 8b: Hazards

Gegeben ist folgendes Programmstück in einer fiktiven Assemblersprache eines RISC-Prozessors:

```
ADD  R2, R1, R4    // R2 := R1+R4
LW   R2, 0(R2)     // Lade [R2] aus dem Speicher in R2
ADD  R2, R2, R3    // R2 := R2+R3
SLT  R6, R2, R4
BEQZ R6, Ende
SUB  R6, R2, R5    // R6 := R2-R5
```

Diese Befehlsfolge wird in der bekannten 5-stufigen Pipeline der DLX-Architektur abgearbeitet. Stellen Sie in der folgenden Tabelle den Ablauf der Pipeline-Verarbeitung dar. Nehmen Sie dabei an, daß der Prozessor keine Optimierungen vornimmt und daß beim Befehl `BEQZ R6, Ende` gilt, daß `R2<>R4` ist. An welchen Stellen treten Hazards auf? Beschreiben Sie jeden Hazard (Warum tritt er auf? Um was für einen Typ Hazard handelt es sich?)!

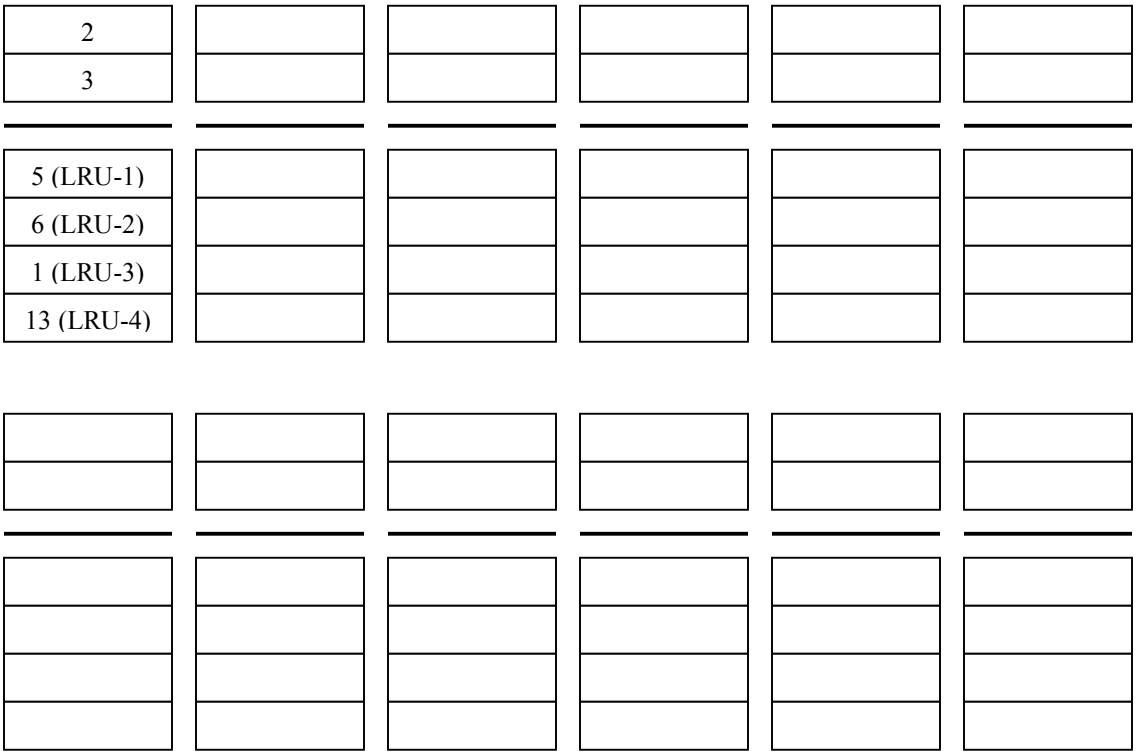
Pipelinezyklus	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`ADD R2, R1, R4`																				
`LW R2, 0(R2)`																				
`ADD R2, R2, R3`																				
`SLT R6, R2, R4`																				
`BEQZ R6, Ende`																				
`SUB R6, R2, R5`																				

Name:	Vorname:

Aufgabe 8c: Caches

Gegeben sei ein zwei Blöcke umfassender 1st-level Cache mit direkter Abbildung (f = Blocknummer mod 2) und ein vier Blöcke großer voll assoziativer 2nd-level Cache mit LRU-Strategie. Wird ein Block aus dem 1st-level Cache verdrängt, speichert ihn der 2nd-level Cache an LRU-Position 1, d.h. als den zuletzt benutzten Block. Bei einem Hit im 2nd-level Cache wird der Block dort gelöscht und in den 1st-level Cache eingefügt.

Die CPU liest in einer Anwendung Blöcke aus dem Hauptspeicher in der folgenden Reihenfolge: 3, 7, 4, 2, 13, 5, 4, 2, 5, 3. Geben Sie den Endzustand der Caches an, wenn zu Beginn der 1st-level Cache Belegung (2, 3) und der 2nd-level Cache die Belegung (5, 6, 1, 13) besaß. Wie viele Zugriffe wurden aus dem 1st-level Cache, dem 2nd-level Cache, bzw. aus dem RAM bedient?



Zugriffe:

- aus 1st-level Cache: _____
- aus 2nd-level Cache:_____
- aus dem RAM: _____