Bitte hier unbedingt Matrikelnummer und Adresse eintragen, sonst keine Bearbeitung möglich.

Postanschrift: FernUniversität, D-58084 Hagen

Name, Vorname

Straße, Nr.

PLZ, Wohnort

FERNUNIVERSITÄT
EINGANG

INF

FERNUNIVERSITÄT
D-58084 Hagen

Fachbereich Informatik

Kurs: **1708 „Technische Informatik II"**

Nachklausur am 05.03.2005

Zutreffendes unbedingt ankreuzen !

Hörerstatus:

- ☐ Vollzeitstudent
- ☐ Teilzeitstudent
- ☐ Zweithörer
- ☐ Gasthörer
- ☐ Bachelor
- ☐ Lehramt
- ☐ . . . . . .

Klausurort:

- ☐ Berlin
- ☐ Bochum
- ☐ Frankfurt
- ☐ Hamburg
- ☐ Karlsruhe
- ☐ Köln
- ☐ München
- ☐ Bregenz
- ☐ Wien
- ☐ . . . . . .

Aufgabe	1	2	3	4	5	6	Summe
erreichbare Punktzahl	10	20	20	25	25		100
bearbeitet							
erreichte Punktzahl							

Note: _____

Hagen, den _____ Betreuer _____

Aufgabe 1: (10 Punkte)

Geben Sie für die folgenden Aussagen an, ob sie richtig oder falsch sind.	richtig	falsch
a) Die **Register-Selektion** in Peripheriebausteinen geschieht in allen Mikrorechner-Systemen immer über spezielle Steuersignale.	☐	☐
b) Bei allen Mikroprozessoren liefert die **Leistungsangabe** in MIPS *(Million Instructions per Second)* und MOPS *(Million Operations per Second)* stets denselben Wert.	☐	☐
c) **Statische Speicherbausteine** sind dadurch gekennzeichnet, daß sie nach dem Ausschalten der Betriebsspannung ihre gespeicherte Information nicht verlieren.	☐	☐

Ergänzen Sie die folgenden Sätze:

d) Der ..... ist ein serielles Bussystem, das hauptsächlich zur Verbindung von ..... verwendet wird und zunächst für den Einsatz im Automobilbau entwickelt wurde.

e) Ein Peripheriebaustein, der mit Hilfe eines Dualzählers verschiedene digitale Zeitfunktionen erzeugen kann, wird als .....[2] bezeichnet. In seiner Funktion als ..... kann er zur Überwachung der zeitgerechten Abarbeitung eines Programms eingesetzt werden.

f) Durch welche speziellen Hardwareeigenschaften ist ein **Digitaler Signalprozessor** (DSP) in der Lage, pro Taktperiode wenigstens einen Zweiadreßbefehl auszuführen? Nennen Sie wenigstens 3.

1. .....

2. .....

3. .....

---

[1] Abkü. ung und volle Bezeichnung verlangt.
[2] Deuts. e und englische Bezeichnung verlangt.

Aufgabe 2: Datentypen und Datenformate (20 Punkte)

In dieser Aufgabe geht es um die Umwandlung von Zahlen im IEEE-754-Standard vom 64-bit-Format ins 32-bit-Format und den dabei verwendbaren verschiedenen Rundungsverfahren. Dazu sei zunächst die folgende 64-bit-Zahl Z_{64} in binärer Schreibweise gegeben:

Z_{64}

1	1	0	0	0	1	1	0	1	0	1	1	1	1	0	1	0	1	1	0	1	1	1	1	0	1	1	0	1	0	1	1
63			59				55				51				47				43				39				35				32

Bitfelder: .....

0	1	1	1	1	0	1	0	1	0	1	1	1	1	0	1	0	1	1	0	1	1	1	1	0	1	1	0	1	1	1	0
31			27				23				19				15				11				7				3				0

Bitfelder: .....

Z_{32}

																															T_0
31			27				23				19				15				11				7				3				0

Bitfelder: .....

a) Wie lautet die Zahl in Hexadezimalform? Z_{64} = $.....

Welcher Dezimalzahl Z_{10} in Exponentschreibweise zur Basis 2 entspricht dies?

(Es reicht, vom Mantissenteil die ersten 4 Stellen zu berechnen.)

$Z_{10} \approx (-1)^{\cdots} \cdot 2^{\cdots} \cdot \ldots\ldots_{10}$

b) Kennzeichnen Sie im oben stehenden Bild die verschiedenen Bitfelder durch senkrechte Striche und benennen Sie sie! (Vorzeichen V, Charakteristik C, Mantissenteil M)
Kennzeichnen Sie diese Bitfelder ebenso im 32-Bit-Format im Bild unten!
Markieren Sie für Z_{64} durch einen senkrechten doppelten Strich die Bitposition, ab der bei der Rundung vom 64-bit- ins 32-bit-Format alle Bits weggelassen werden! (Beachten Sie die Längen der Charakteristiken.)

Lage des Doppelstriches zwischen Bit _____ und _____

c) Welche Bedingung muß für die Charakteristik C_{64} von Z_{64} gelten, damit eine Rundung ins 32-bit-Format ohne Fehler in der Charakteristik C_{32} möglich ist? Gehen Sie bei Ihrer Lösung vom (zulässigen) Exponenten E der zu rundenden Zahl aus. (Berechnung erforderlich!)

dezimal:

binär:

d) Tragen Sie im oben stehenden Bild die Zahl Z_{32} in binärer Form ein, die man durch Konvertierung der 64-bit-Zahl Z_{64} ins 32-bit-Format erhält, wenn als Rundungsverfahren „Abschneiden" *(Truncate)* gewählt wird. Geben Sie an, wie man aus der 64-bit-Charakteristik C_{64} und dem Exponenten E die 32-bit-Charakteristik C_{32} gewinnt.

dezimal:

binär:

C_{64} = ...$_2$ ⇒ C_{32} = ...$_2$

e) Die niederstwertige Tetrade T_n von Z_{64}, die bei der Rundung ins 32-bit-Format <u>nicht</u> abgeschnitten wird, habe den Wert $T_n = 1011_2$, die höchstwertige Tetrade T_{n-1} des abgeschnittenen Restes den Wert $T_{n-1} = 1101_2$.

Skizze: Z_{64} = (Bit 63).....1011 | | 1101.....(Bit 0) (| | Grenze der Rundung)

Ergänzen Sie die folgende Tabelle durch die Werte, die man für die niederstwertige Tetrade T_0 von Z_{32} erhält, wenn man die vier im Kurs beschriebenen Rundungsverfahren ins 32-bit-Format auf Z_{64} anwendet:

Rundung zur

i) nächstgelegenen 32-bit-Zahl,
ii) nächstgrößeren 32-bit-Zahl,
iii) nächstkleineren 32-bit-Zahl,
iv) betragsmäßig nächstkleineren 32-bit-Zahl („Abschneiden").

Vorzeichen der Mantisse	**Rundung zur 32-bit-Zahl**			
	nächstgelegenen	nächstgrößeren	nächstkleineren	Abschneiden
positiv (+/0)				
negativ (-/1)				

Aufgabe 3: MMX-Rechenwerk (20 Punkte)

Wiederholung aus Kurs 1708:

Ein MMX-Rechenwerk *(Multimedia Extension)* unterstützt gepackte Datenformate, die in einem 64-bit-Register wahlweise 8 Bytes, 4 (16-bit-)Wörter, 2 (32-bit-)Doppelwörter oder ein 64-bit-Wort (Quadword) unterbringen. Alle Werte können vorzeichenlos oder vorzeichenbehaftet sein. Negative Werte werden dabei im 2er-Komplement dargestellt. Spezielle MMX-Befehle wirken parallel auf diese Datenformate, d.h. es können z.B. durch einen einzigen Addier-Befehl zweimal 8 Bytes addiert werden. Ein Übertrag zwischen den einzelnen Werten der gepackten Daten findet dabei nicht statt. Die Datenbreite wird im Assemblerbefehl spezifiziert. Der erwähnte Addier-Befehl hat z.B. die Form: PADDX, wobei X = B, W, D für 8 Bytes, 4 Wörter, 2 Doppelwörter steht.

Der MMX-Befehlssatz enthält u.a. die in der folgenden Tabelle angegebenen Befehle.

Mnemo		Bemerkung
Logische Operationen (wirken auf alle 64 Bits)		
PAND	*Packed And*	bitweise Und-Verknüpfung
PANDN	*Packed And-Not*	bitweise Und-Verknüpfung mit negiertem erstem Operanden
POR	*Packed Or*	bitweise Oder-Verknüpfung
PXOR	*Packed Exclusive Or*	bitweise Antivalenz-Verknüpfung
Vergleichsbefehle (für X = B, W, D)		
PCMPEQX	*P. Compare Equal*	elementeweiser, vorzeichenloser Vergleich auf gleich
PCMPGTX	*P. Compare Greater*	elementeweiser, vorzeichenloser Vergleich auf größer
Transferbefehle		
MOVQ	*Move Quadword*	64-bit-Transfer zw. MMX-Reg. und MMX-Reg./Speicher
MOVD	*Move Doubleword*	32-bit-Transfer zw. MMX-Reg. und MMX-Reg./Speicher
PSWAPD	*P. Swap Doubleword*	Vertauschen der Doppelwörter in Registern
Multiplizier/Addierbefehle (W × W → D)		
PMULADDWD	*Multiply-Add*	4fache Multiplikation mit Addition zu Doppelwörtern
PADDX	*Packed Add*	Parallele Addition mit X = B, W, D
PSUBX	*Packed Sub*	Parallele Subtraktion mit X = B, W, D

Als Ergebnis liefern die Vergleichsbefehle für jedes Element (B, W, D) den Wert $F...F (also eine Folge von ‚1'-Bits), wenn der Vergleich wahr ist, andernfalls den Wert $0...0 (also eine Folge von ‚0'-Bits).

Die Befehle werden im Zweiadreß-Format angegeben. Dabei bedeutet die Assemblernotation <Befehl> R1, R2:

Der Inhalt von Register R1 wird mit dem Inhalt von R2 durch die Operation <op> verknüpft und das Ergebnis wird im Register R1 abgelegt.

In Kurzform: R1 := R1 <op> R2.

Vereinfachend werde angenommen, daß durch den Befehl

MOVQ Ri, #<Konstante> oder MOVQ Ri, [Speicher] oder MOVQ Ri, Rj

ein MMX-Register R unmittelbar mit einer 64-bit-Konstanten, einem 64-bit-Speicherwort oder dem Inhalt eines anderen MMX-Registers geladen werden kann.

Aufgabenstellung:

Über eine Datenleitung werden die Hexadezimalziffern 0,...,9,A,...,F im ASCII-Code nach folgender Codierungstabelle übertragen:

Hex.	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
ASCII	30	31	32	33	34	35	36	37	38	39	41	42	43	44	45	46

a) Geben Sie einen Algorithmus an, der aus einem empfangenen ASCII-Zeichen nach oben stehender Tabelle das entsprechende Hexadezimalzeichen ermittelt! Das Ergebnisregister darf die errechneten Hexadezimalzeichen mit führender Null enthalten, also: $00, ..., $09, $0A, ..., $0F. (Es reicht eine „umgangssprachliche" Formulierung: ‚lade ...', ‚vergleiche ...', ‚springe ...', ‚subtrahiere ...', ‚addiere ...' usw.)

1. lade ASCII-Zeichen aus dem Speicher ins Register R0
2.
3.
4.
5.
6.
7.

.....

b) Schreiben Sie nun – ausgehend vom Algorithmus unter a) – eine Folge von MMX-Befehlen, die einen Vektor aus 8 ASCII-Zeichen nach oben stehender Tabelle aus dem Speicher liest und diese parallel in den Vektor der 8 zugeordneten Hexadezimal-Ziffern umwandelt – und zwar in der Darstellung $00, ..., $09, $0A, ..., $0F. Es stehen Ihnen die Register R0 – R7 zur Verfügung.

Nr.	Befehl	Kommentar
1	MOVQ R0, [Speicher]	; lade 8-stelligen ASCII-Vektor nach R0
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
.....		

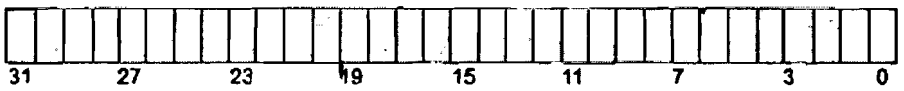
Aufgabe 4: (25 Punkte)

Ein Mikroprozessor mit 32-bit-Adreßbus und 64-bit-Datenbus besitze die in der folgenden Tabelle beschriebenen Cache-Speicher. Die Cache-Kohärenz werde jeweils durch das MESI-Protokoll gewährleistet.

	L1-Cache	**L2-Cache**
Kapazität (Datenspeicher)	64 kbyte	256 kbyte
Verwaltung	*4-way set associative*	*direct mapped*
Länge der Einträge	8 byte (64 bit)	32 byte (256 bit)
Datenspeicher: Organisation		
Adreßspeicher: Organisation		
Adreßspeicher: Kapazität		

a) Kennzeichnen Sie im folgenden Bild – jeweils für den L1- und L2-Cache – die unterscheidbaren Bitfelder einer Speicheradresse durch senkrechte Striche und tragen Sie ihre Bezeichnungen ein.

L1: .....



L2: .....

b) Leiten Sie die Werte für die Organisation der Datenspeicher und der Organisation sowie der Kapazität der Adreßspeicher ab. Vervollständigen Sie die oben stehende Tabelle um diese Angaben. Berücksichtigen Sie dabei die erforderlichen Bits für das MESI-Protokoll.

L1:

Datenspeicher-Organisation:

Adreßspeicher-Organisation:

Adreßspeicher-Kapazität:

L2:

Datenspeicher-Organisation:

Adreßspeicher-Organisation:

Adreßspeicher-Kapazität:

c) Der Inhalt der Speicherzelle mit der Adresse $A7B0 6D5E liege in beiden Caches vor. Geben Sie für beide Caches die folgenden Werte in Hexadezimalform an:

- Wert im Adreßspeicher

  L1: $_____

  L2: $_____

- Nummer des Eintrags

  L1: $_____

  L2: $_____

- Byteauswahl

  L1: $_____

  L2: $_____

Benutzen Sie zur Ermittlung der Werte die folgende Zeichnung und markieren Sie darin die verschiedenen Bitfelder für den L1- bzw. L2-Cache:

Aufgabe 5: DSP-Programmierung (25 Punkte)

Durch die folgende Tabelle sei das Ergebnis einer Klausur zu einem Kurs der Technischen Informatik gegeben. Sie sollen in dieser Aufgabe eine Befehlsfolge für den ADSP-218x schreiben, die in möglichst wenigen Taktzyklen aus dieser Tabelle die Durchschnittsnote der Klausur ermittelt.

Note	1	2	3	4	5
%	10	14	25	18	33

a) Geben Sie an, welche Zahlendarstellung (*signed/unsigned integer, signed/unsigned fractional*) Sie besonders geeignet für die interne Verarbeitung der Noten- und Prozentwerte durch den ADSP-218x halten. (Begründung erforderlich!) Wie sind die Zahlenwerte im Speicher abzulegen und wie ist das erhaltene Ergebnis zu interpretieren?

b) Geben Sie eine Berechnungsformel für die Durchschnittsnote an und leiten Sie daraus ab, wie die Tabellenwerte so in den internen Speicherbereichen abzulegen sind, daß auf sie zur Berechnung des Durchschnittswertes möglichst effektiv zugegriffen werden kann. (Begründung erforderlich!)

c) Geben Sie eine Folge von Befehlen an, durch die alle benötigten Register der Adreßgeneratoren und Zustands-Flags geeignet initialisiert werden, d.h. das "Programm" soll sich nicht die Register-Initialisierung nach dem Rücksetzen des Prozessors zunutze machen. Die Startadressen der Tabellenwerte können Sie geeignet vorgeben. Der Programmteil soll ab Adresse 0x0100 im Programmspeicher liegen.

Adresse	Befehl	Kommentar
0x0100		
0x0101		
0x0102		
0x0103		
0x0104		
0x0105		
0x0106		
0x0107		
.....		

d) Geben Sie eine Befehlsfolge an, die die Durchschnittsnote mit möglichst wenigen Befehlen – also auch in möglichst wenigen Taktzyklen – berechnet und das (ggf. gerundete) Ergebnis im festgelegten Format in einem der Rechenwerksregister ausgibt.

Adresse	Befehl	Kommentar
0x0110		
0x0111		
0x0112		
0x0113		
0x0114		
0x0115		
0x0116		
0x0117		
0x0118		
0x0119		
.....		

e) Geben Sie das (16-bit-)Register an, in welchem das gerundete Ergebnis Ihres Programms zu finden ist.

Fractional Mode:

Integer Mode :