

WS 11/12

Klausur: Rechnerarchitektur

Datum: 08.02.12

Prüfer: Prof. Dr. J. Neuschwander

Zeitdauer: 90 Min.

Name:

Matr.-Nr.:

- MUSTERLÖSUNG -Hinweise:

Beschriften Sie jedes weitere abgegebene Lösungsblatt deutlich lesbar mit Ihrem Namen und Ihrer Matrikelnummer.

Es sind für diese Klausur als Hilfsmittel nur Taschenrechner, ein DIN-A4 Blatt als Formelsammlung sowie der Befehlssatzes des Prozessors M68000 zugelassen.

**Bei einer angegebenen Lösung muss der Lösungsweg nachvollziehbar sein.**

Maximal erreichbare Punktzahl der Klausur: 37 Punkte

Mindestpunktzahl zum Bestehen der Klausur: 17 Punkte

Aufgabe 1: 8 Punkte  
Aufgabe 2: 4 Punkte  
Aufgabe 3: 6 Punkte  
Aufgabe 4: 7 Punkte  
Aufgabe 5: 6 Punkte  
Aufgabe 6: 6 Punkte

## 1. Aufgabe: ( 8 P )

Gegeben sei folgende Programmsequenz für einen RISC-Prozessor:

```

S1:  ADD    R1, R2, 2      ; R1:= R2 + 2
S2:  SUB    R4, R3, R1     ; R4:= R3 - R1
S3:  LOAD   R5, (A1)       ; R5:= <(A1)>
S4:  MUL    R3, R5, 5      ; R3:= R5 * 5
S5:  ADD    R3, R3, 3      ; R3:= R3 + 3
S6:  ADD    R2, R1, R5     ; R2:= R1 + R5
  
```

Die einzelnen Befehle werden in einer DLX-Pipeline verarbeitet: Die Latenzzeit jeder Pipelinestufe beträgt 6 ns.

Beim Start des Programmstücks sind die Register folgendermaßen belegt:

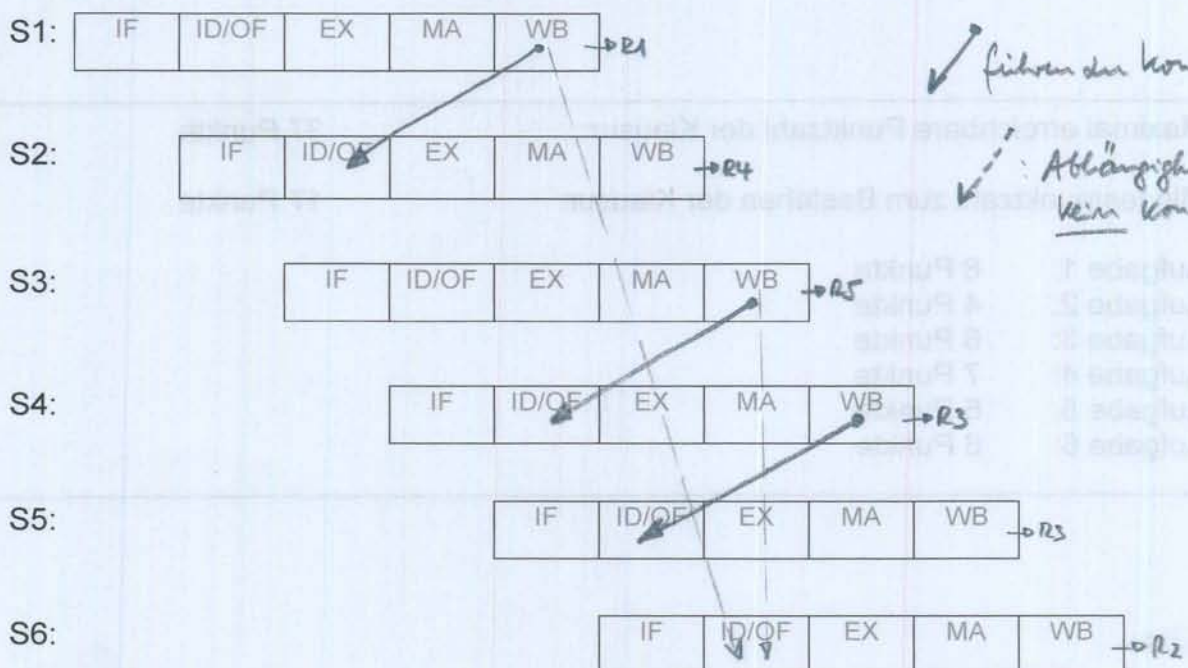
R1	R2	R3	R4	R5	<(A1)>
3	5	7	9	2	1

- a) Geben Sie die Registerbelegung nach Ablauf des Programmstücks auf der DLX-Pipeline an. Tragen Sie die jeweiligen Werte in die folgende Tabelle ein.

R1	R2	R3	R4	R5
7	8	10	4	1

$R1 := 5 + 2 = 7$   
 $R4 := 7 - 3 = 4$   
 $R5 := 1$   
 $R3 := 2 \cdot 5 = 10$   
 $R3 := 7 + 3 = 10$   
 $R2 := 7 + 1 = 8$

- b) Markieren Sie mit Pfeilen die Datenabhängigkeiten, die im gegebenen Programm zu Pipeline-Konflikten führen.





- c) Die auftretenden Pipeline-Konflikte sollen vom Compiler behandelt werden. Ergänzen Sie obiges Programmstück mit möglichst wenigen NOP-Befehlen so, dass das Ergebnis dem der normalen, sequentiellen Bearbeitung entspricht.

S1  
NOP  
NOP  
S2  
S3  
NOP  
NOP  
S4  
NOP  
NOP  
S5  
S6

1P

- d) Wie viele Takte werden benötigt, um dieses Programm aus c) abzuarbeiten?

$$T = n + k - 1 = 12 + 5 - 1 = 16$$

0,5P

- e) Die DLX-Pipeline wird mit der maximalen Taktfrequenz von 125 MHz betrieben. Wie groß ist dann die Latenzzeit eines Pipelineregisters?

$$\tau = \underbrace{\tau_{\max}}_{6ns} + \tau_{\text{reg}} \quad f_{\max} = \frac{1}{\tau} \Rightarrow f_{\max} = \frac{1}{6ns + \tau_{\text{reg}}}$$

$$\Rightarrow \underline{\tau_{\text{reg}}} = \frac{1}{f_{\max}} - 6ns = \frac{1}{125 \cdot 10^6} - 6ns = 0,008 \cdot 10^{-6} - 6ns$$

$$= 8ns - 6ns = \underline{\underline{2ns}}$$

1P

- f) Wie hoch ist der Durchsatz obiger DLX-Pipeline?

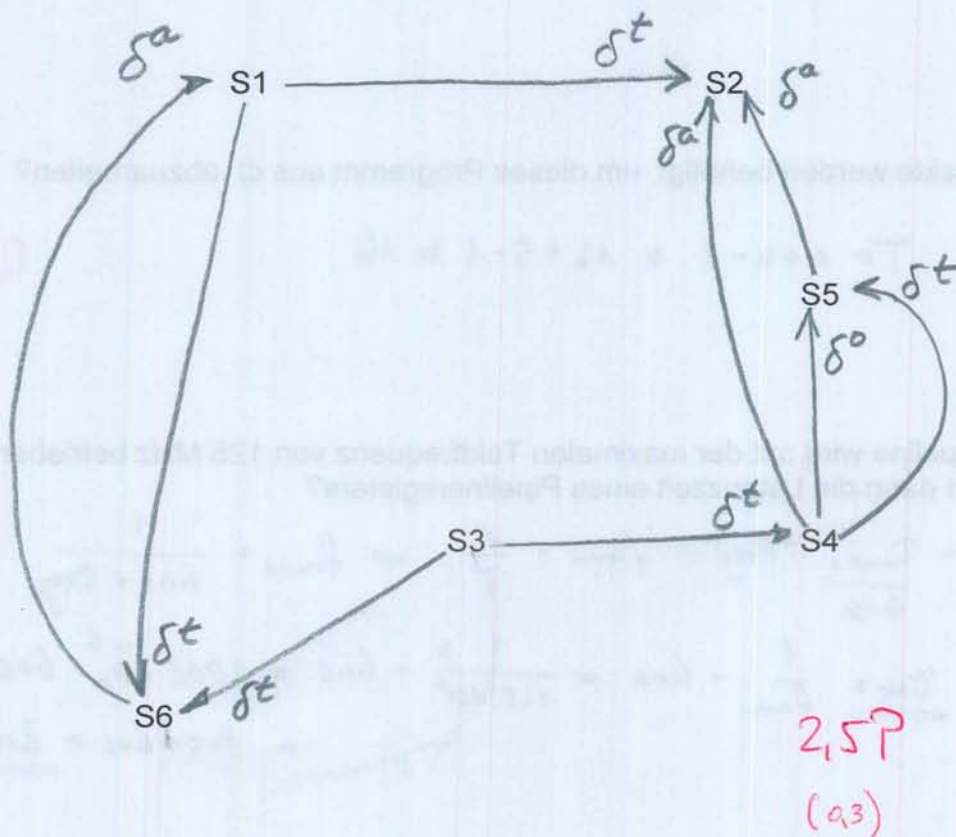
$$D = 1$$

0,5P

g) Geben Sie alle Datenabhängigkeiten im Programmstück an.

S1:	ADD	R1, R2, 2	; R1:= R2 + 2
S2:	SUB	R4, R3, R1	; R4:= R3 - R1
S3:	LOAD	R5, (A1)	; R5:= <(A1)>
S4:	MUL	R3, R5, 5	; R3:= R5 * 5
S5:	ADD	R3, R3, 3	; R3:= R3 + 3
S6:	ADD	R2, R1, R5	; R2:= R1 + R5

Stellen Sie die jeweiligen Abhängigkeiten durch einen Pfeil dar und benennen sie diese Abhängigkeit.





## 2. Aufgabe: (4 P)

- a) In einem Rechner läuft die CPU mit einer Taktfrequenz von 1GHz. Die durchschnittliche Anzahl von Taktzyklen pro Befehl beträgt 4. Die CPU soll beim Re-Design so verbessert werden, dass die CPI-Rate auf 2 verringert und die Programmausführungszeit auf ein Viertel reduziert wird. Mit welcher Taktfrequenz muss die neue CPU dann betrieben werden?

a: alt  
n: neu

$$CPU_{zeit} = \frac{n \cdot CPI}{f} \Rightarrow CPU_{zeit a} = \frac{n \cdot CPI_a}{f_a} ; CPU_{zeit n} = \frac{n \cdot CPI_n}{f_n} = \frac{CPU_{zeit a}}{4}$$

$$\Rightarrow \frac{n \cdot CPI_a}{f_a} = \frac{1}{4} \cdot \frac{n \cdot CPI_n}{f_n} \Rightarrow f_n = \frac{CPI_n}{CPI_a} \cdot 4 \cdot f_a$$

$$f_n = \frac{2}{4} \cdot 4 \cdot 1 \text{GHz} = 2 \text{GHz}$$

2P

- b) Gegeben sei ein Prozessor mit getrennten Befehls- und Datencachespeicher. Die Fehlzugriffsrate des Befehls cachespeichers beträgt 20%, die des Datencachespeichers beträgt 50%. Der Zugriff auf den Befehls cachespeicher erfolgt in 10 ns, der Zugriff auf den Datencachespeicher erfolgt in 20 ns. Ein Hauptspeicherzugriff dauert 80 ns. Auf dem Prozessor läuft ein Programm *test*, dessen Speicherzugriff zu 80% aus Zugriffen auf Befehle und zu 20% aus Zugriffen auf Daten bestehen. Gehen Sie davon aus, dass die Cachespeicher und der Hauptspeicher gleichzeitig adressiert werden und dass das Datum beim Laden aus dem Hauptspeicher direkt dem Prozessor zugeführt wird. Berechnen Sie die mittlere Speicher-Zugriffszeit für das Programm *test*.

$$\tilde{t}_m = t_{code} \cdot H + t_{as} \cdot (1-H)$$

Befehlscode

$$H = 80\%$$

$$t_{code} = 10 \text{ns}$$

$$t_{as} = 80 \text{ns}$$

Datencode

$$H = 50\%$$

$$t_{code} = 20 \text{ns}$$

$$t_{as} = 80 \text{ns}$$

test

80% Befehlscode

20% Datencode

$$\tilde{t}_{m \text{ Befehl}} = 10 \text{ns} \cdot 0,8 + 80 \text{ns} \cdot 0,2 = 24 \text{ns}$$

$$\tilde{t}_{m \text{ Daten}} = 20 \text{ns} \cdot 0,5 + 80 \text{ns} \cdot 0,5 = 50 \text{ns}$$

$$\tilde{t}_{m \text{ test}} = \tilde{t}_{m \text{ Befehl}} \cdot 0,8 + \tilde{t}_{m \text{ Daten}} \cdot 0,2$$

$$= 24 \text{ns} \cdot 0,8 + 50 \text{ns} \cdot 0,2 = 29,2 \text{ns}$$

2P



## 3. Aufgabe: ( 6 P )

je 0,5P

(Zur Ermittlung der Punktzahl in dieser Teilaufgabe werden von den richtig angekreuzten Aussagen die falsch angekreuzten Aussagen abgezogen; Nicht angekreuzte Aussagen zählen nicht und gehen somit nicht in die Bewertung ein.)

Fragen	richtig	falsch
Der TAS-Befehl (M68000) kann zur Implementierung binärer Semaphore benutzt werden.	X	
Die symbolischen Stackoperationen <i>push</i> und <i>pull</i> sind zur Berechnung rekursiver Funktionen erforderlich.	X	
Bei der „write-back“-Strategie eines Caches können Inkonsistenzen zwischen Cache und Arbeitsspeicher vermieden werden.		X
Interleaving zwischen verschiedenen Speicherbänken dient zur Kompensation der Zugriffszeit von DRAMs.		X
Der Austausch von Zeilen in Sätzen bei mehrwege-satzassoziativen Caches kann nach dem LRU-Prinzip erfolgen.	X	
Beim „Bus Snooping“ wertet der Cache-Controller die Adressen beim Zugriff anderer Busmaster auf den Hauptspeicher aus.	X	
Zur Auflösung globaler Adressbezüge durch den Linker benötigt der Assembler den <i>Define Constant (DC)</i> -Befehl.		X
Unter Branch Recovery versteht man die Rücknahme getätigter Berechnungen bei falschen Vorhersagen im Fall spekulativer Befehlsverarbeitung.	X	
Die Datenübertragung zwischen L1- und L2-Caches erfolgt durch Blockbuszyklen.	X	
Eine statische Speicherallokierung während der Assemblierung erreicht man durch geeignete Assemblerdirektiven.	X	
Bei VLIW-Prozessoren übernimmt der Compiler die statische Befehlszuteilung auf die Ausführungseinheiten des Rechenwerks.	X	
Bei der Memory-Mapped-Adressierung existieren zwei Adressräume, einer für Speicher und einer für Ein-/Ausgabe-Einheiten.		X

## 4. Aufgabe: (7 P)

Gegeben sei ein 8-fach-satzassoziativer Cache-Speicher (A8) mit einer Cachelinegröße von 64 Byte. Die Hauptspeicheradresse umfasst 32 Bit. Der Satzindex ist 13 Bit breit. Zur Verwaltung eines Cache-Blocks werden das Valid-Bit (V) und das Dirty-Bit (D) verwendet.

a) Wie viele Bit umfasst der Tag?

Cacheline-Navigation: 6 Bit ( $\rightarrow 2^6$  byte)  
Satzindex: 13 Bit

$$\underline{\underline{\text{Tag} = 32 \text{ Bit} - 19 \text{ Bit} = 13 \text{ Bit}}}$$

$$\Rightarrow \Sigma = 13 + 6 = 19 \text{ Bit}$$

0,5 P

b) Wie viele Befehle mit einer Länge von 32 Bit können sich gleichzeitig im Cache-Speicher befinden?

i) Anzahl der Zeilen des Cache:  $= 2^{13} \times 8 = 2^{13} \times 2^3 = 2^{16}$

ii) Anzahl Befehle pro Zeile:  $= \frac{64 \text{ Byte}}{4 \text{ Byte}} = \frac{2^6}{2^2} = 2^4 = 16$

iii) # Befehle im Cache:  $\underbrace{16}_{ii)} \times \underbrace{2^{16}}_{i)} = 2^4 \times 2^{16} = 2^{20}$   
(1 048 576)

1,5 P

c) Bestimmen Sie den insgesamt erforderlichen Speicherbedarf für die Realisierung des Cache-Speichers (Adress- und Datenspeicher).

• Adresspeicher: # Zeilen  $\times$  (Tag-Größe + Satzindex)

$$\begin{aligned} \text{Kap}_A &= 2^{16} \cdot (13 \text{ Bit} + 2 \text{ Bit}) = 2^{16} \cdot 15 \text{ Bit} = 983040 \text{ Bit} \\ &= 122880 \text{ Byte} = \underline{\underline{120 \text{ KByte}}} \end{aligned}$$

• Datenspeicher: # Zeilen  $\times$  64 Byte  $= 2^{16} \times 2^6 \text{ Byte} = 2^{22} \text{ Byte} = \underline{\underline{4 \text{ MByte}}}$   
(4194304)

1,5 P

$$\begin{aligned} \Rightarrow \text{Gesamt-Speicherbedarf} &= (122880 + 4194304) \text{ Byte} = \underline{\underline{4317184 \text{ Byte}}} \\ &= \underline{\underline{4216 \text{ KByte}}} \end{aligned}$$



Gegeben sei ein direkt abbildender Cache-Speicher mit einer Speicherkapazität von 128 Byte und einer Blockgröße von 16 Byte. Als Aktualisierungsstrategie wird das Rückschreibverfahren (*write back*) verwendet. Nehmen Sie an, dass der Cache-Speicher zu Beginn leer ist. Betrachten Sie die folgenden Lese- und Schreibzugriffe auf die in hexadezimaler Schreibweise angegebenen Adressen.

- d) Ergänzen Sie in dieser Tabelle die fehlenden Einträge. Verwenden Sie dabei Miss für Cache-Miss und Hit für Cache-Hit. Geben Sie in der letzten Zeile der Tabelle an, ob der entsprechende Cache-Block in den Hauptspeicher zurückkopiert werden muss (ja) oder nicht (nein).

$$\# \text{ Zeilen} = \frac{128}{16} = \frac{2^7}{2^4} = 2^3 = 8$$

↳ Index 3 Bit (Zeilenanzahl)

Adresse	64	32	E4	18	E0	7A	A2	F0	E3
read / write	w	r	w	r	r	r	r	w	r
Index	6	3	6	1	6	7	2	7	6
Tag	0	0	1	0	1	0	1	1	1
Hit / Miss	M	M	M	M	H	M	M	M	H
write back ?	N	N	J	N	N	N	N	N	N

Index

64: 0110 0100

32: 0011 0010

E4: 1110 0100

18: 0001 1000

E0: 1110 0000

7A: 0111 1010

A2: 1010 0010

F0: 1111 0000

E3: 1110 0011

3,5P



979

b) Welche Länge muss man wählen, um die Hauptzeile?

$$4 \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4} \cdot \frac{1}{2} = \frac{1}{8} \cdot \frac{1}{2} = \frac{1}{16} \cdot \frac{1}{2} = \frac{1}{32}$$

$$4 \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4} \cdot \frac{1}{2} = \frac{1}{8} \cdot \frac{1}{2} = \frac{1}{16} \cdot \frac{1}{2} = \frac{1}{32}$$

c) Wie groß ist der Vektor, wenn die Programms 1280 Bytes zu Hauptzeile zeigt?

$$4 \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4} \cdot \frac{1}{2} = \frac{1}{8} \cdot \frac{1}{2} = \frac{1}{16} \cdot \frac{1}{2} = \frac{1}{32}$$

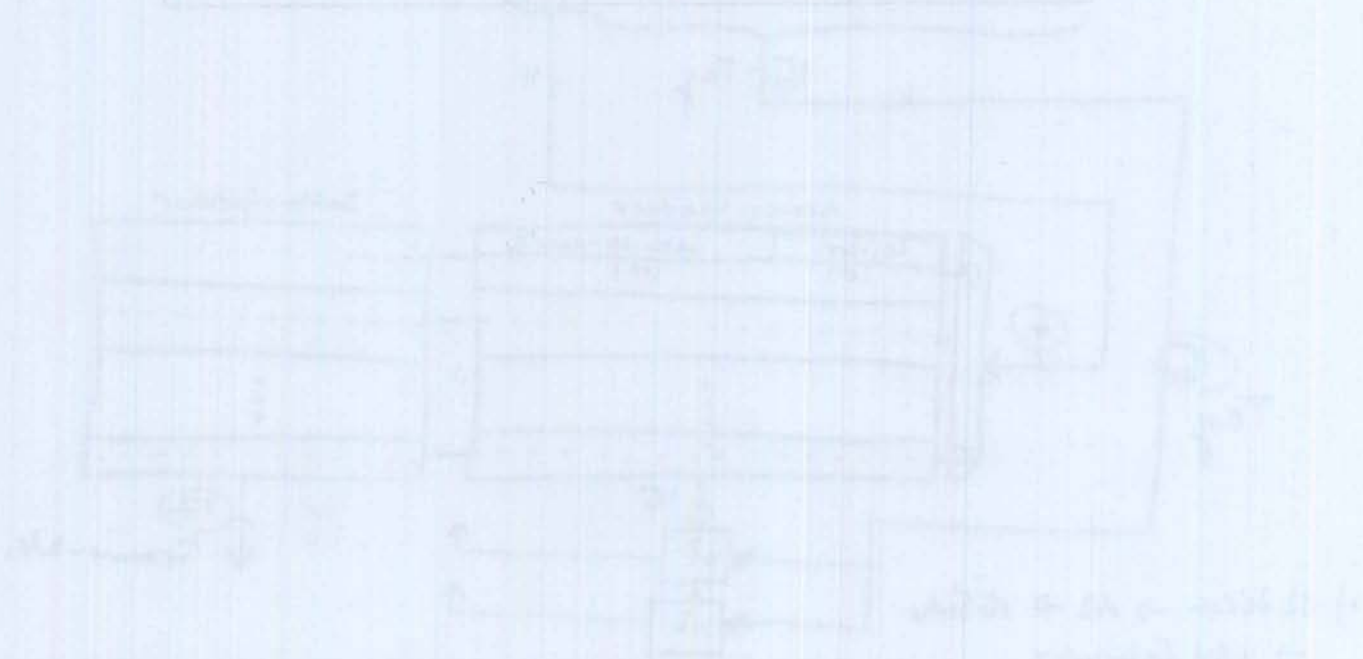
$$4 \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4} \cdot \frac{1}{2} = \frac{1}{8} \cdot \frac{1}{2} = \frac{1}{16} \cdot \frac{1}{2} = \frac{1}{32}$$

$$4 \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4} \cdot \frac{1}{2} = \frac{1}{8} \cdot \frac{1}{2} = \frac{1}{16} \cdot \frac{1}{2} = \frac{1}{32}$$

d) Skizzieren Sie die oben angegebenen Adressierung einer 2-fach-assoziativen TLB mit 32 Zeilen, die zur Beschreibung der Adressierung eingesetzt werden soll.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

95







Gegeben sei nun eine Speicherverwaltungseinheit (MMU). Der virtuelle Speicher ist in 8 Seiten mit je 1 kByte unterteilt. Der physische Speicher hat eine Kapazität von 4 kByte. Der aktuelle Ausschnitt der Seitentabelle ist in folgender Tabelle angegeben.

	Virtuelle Seitennummer	Physische Seitennummer
	0	-
	1	-
→	2	1
→	3	2
	4	-
	5	0
	6	3
	7	-

$$1 \text{ Seite} : 1 \text{ kByte} = 2^{10} \text{ B}$$

7 10 Bit für Navigation  
innerhalb Seite

- e) Ermitteln Sie die physischen Adressen (Dezimaldarstellung) zu den folgenden virtuellen Adressen (Dezimaldarstellung) :

2100, 4095

Seiten-Nr.

$$\text{i) } (2100)_{10} = \left( \underbrace{10}_{2} 0000 110100 \right)_2 \Rightarrow \text{In Frame 1 transformiert} \\ \Rightarrow (010000 110100)_2 = \underline{(1076)_{10} \text{ physisch}} \quad 1P$$

$$\text{ii) } (4095)_{10} = \left( \underbrace{1111}_{3} 1111 1111 \right)_2 \Rightarrow \text{In Frame \# 2 transf.} \\ \Rightarrow (10 1111 1111 1111)_2 = \underline{(3071)_{10} \text{ physisch}} \quad 1P$$

## 6. Aufgabe: (6 P) 6,5P

In der folgenden Aufgabe soll für alle Teilaufgaben jeweils die in Tabelle 8.1 dargestellte Ausgangssituation herrschen. Das Zeichen \$ kennzeichnet die hexadezimale Darstellung von Zahlen. Beachten Sie jeweils das Datenformat.

Prozessor-Register					Arbeitsspeicher (16 Bit)	
Datenregister	Inhalt				Adresse	Inhalt
D0	\$ 00	00	11	11	\$ 10000	\$ 00 01
D1	\$ 22	22	33	33	\$ 10002	\$ 02 03
D2	\$ 44	44	55	55	\$ 10004	\$ 04 05
D3	\$ 66	66	77	77	\$ 10006	\$ 06 07
D4	\$ 88	88	99	99	\$ 10008	\$ 08 09
D5	\$ AA	AA	BB	BB	\$ 1000A	\$ 0A 0B
D6	\$ CC	CC	DD	DD	\$ 1000C	\$ 0C 0D
D7	\$ EE	EE	FF	FF	\$ 1000E	\$ 0E 0F

Adressregister	Inhalt			
A0	\$ 00	01	00	02
A1	\$ 00	01	00	0A
A2	\$ 00	01	00	08
A3	\$ 00	01	00	00
A4	\$ 00	01	00	00
A5	\$ 00	01	00	00
A6	\$ 00	01	00	00
A7	\$ 00	01	00	00

\$ FFFFA	\$ 01 02
\$ FFFFC	\$ 03 04
\$ FFFFE	\$ 05 06

Tabelle 8.1 : Ausgangssituation Register und Hauptspeicher

- a) Tragen Sie den Inhalt der Datenregister D0 und D1 (je 32 Bit) in Tabelle 8.2 ein, nachdem mit der Ausgangssituation entsprechend Tabelle 8.1 alle folgenden Befehle nacheinander ausgeführt wurden :

Z EQU \$AA  
K DC.W \$44  
ERG DS.L \$2  
MOVE.W K, D1  
ADD.W #Z, D2  
ADD.L D1, D2  
MOVE.L D2, ERG

MOVE.W \$0044, D1 → \$ 22 22 00 44  
ADD.W \$00AA, D2 → \$ 44 44 55 55  
ADD.L D1, D2 → \$ 44 44 55 55  
→ \$ 44 44 55 55  
+ \$ 22 22 00 44  
66 66 56 43

Prozessor-Register	
Datenregister	Inhalt
D1	\$ 22 22 00 44
D2	\$ 66 66 56 43

05  
1P

Tabelle 8.2 : Lösung zu Teilaufgabe a)



- b) Tragen Sie den Inhalt der Datenregister D5, D6 und D7 sowie der Adressregister A0, A1 und A2 (je 32 Bit) in Tabelle 8.3 ein, nachdem mit der Ausgangssituation entsprechend Tabelle 8.1 die folgenden Befehle nacheinander ausgeführt wurden:

MOVE.L (A1)+, D5      → (A1) := 1000E  
 MOVE.W (A1)+, D6      → (A1) := 10010  
 MOVE.B -(A2), D7      → (A2) := 10007  
 MOVE.W -(A1), -(A0)    → (A1) := 1000E  
                               → (A0) := 10000

D5	\$ 0A 0B 0C 0D	0,5
D6	\$ CC CC 0E 0F	0,5
D7	\$ EE EE FF 07	0,5

Adressregister	Inhalt	
A0	\$ 00 01 00 00	0,5
A1	\$ 00 01 00 0E	0,5
A2	\$ 00 01 00 07	0,5

Tabelle 8.3 : Lösung zu Teilaufgabe b)

- c) Tragen Sie den Inhalt der Datenregister D3 und D4 (je 32 Bit) in Tabelle 8.4 ein, nachdem mit der Ausgangssituation entsprechend Tabelle 8.1 die folgenden Befehle nacheinander ausgeführt wurden :

SUB.L (A0)+, D3  
 LSR.L #2, D4

D3: \$ 66 66 77 77  
 - \$ 02 03 04 05  
 -----  
 64 63 73 72

Prozessor-Register		
Datenregister	Inhalt	
D3	\$ 64 63 73 72	1P
D4	\$ 22 22 26 66	1P

Tabelle 8.4 : Lösung zu Teilaufgabe c)

D4: \$ 88 88 99 99

Binär: 00  
 1000 1000 1000 1000 1001 1001 1001 1001  
 2 2 2 2 2 6 6 6  
 → fällt raus