Universität Paderborn
Institut *Elektrotechnik und Informationstechnik*
Fachgebiet *Datentechnik*
Prof. Sybille Hellebrand

Klausur

Grundlagen der Rechnerarchitektur

19. September 2014

Punkteverteilung							
Aufgabe	1	2	3	4	5	6	Σ
maximale Punkte	11	9	10	20	20	20	90
erreichte Punkte							

Note:	

Aufkleber

Name:	
Matrikelnummer:	
Studienrichtung:	

Hinweise:
Für die Lösung der Klausuraufgaben sind ausschließlich die Aufgabenblätter zu verwenden. Lösungsangaben außerhalb der Aufgabenblätter („Schmierzettel“, etc.) werden bei der Bewertung nicht berücksichtigt!

Beschriften Sie jede Doppelseite mit Ihrer Matrikelnummer!

Mit Bleistift oder der Korrekturfarbe rot angefertigte Lösungen werden nicht bewertet!

Die Verwendung von „Tipp-Ex“ oder „Tintenkiller“ ist untersagt.

Es ist ein handgeschriebener DIN-A4 Zettel als Hilfsmittel zugelassen!

Es sind keine weiteren Hilfsmittel zugelassen!

Aufgabe 1: (Leistungsbewertung) 11 Punkte

a) Die Firma Limited Performance Corp. hat einen Prozessor mit einem idealen CPI-Wert von 1 entwickelt, der mit 500 MHz betrieben werden kann. Im Befehlscache befinden sich bereits alle benötigten Befehle. Um schnelle Speicherzugriffe auf Daten zu unterstützen, sollen zwei Cache-Ebenen realisiert werden (L1 und L2). Die Fehlgriffsrate für den L1-Cache sei 5 %. Bei einem Fehlgriff im L1-Cache wird auf den L2-Cache zugegriffen. Die Zeit dafür beträgt 80 ns. Von den notwendigen Zugriffen auf den L2-Cache sind 40 % Fehlgriffe und die Daten müssen dann aus dem Hauptspeicher geholt werden. Die Zeit dafür beträgt 200 ns. Wie groß ist der realistische CPI-Wert (einschließlich Speicherwartetakten), wenn 50% der Befehle auf Daten zugreifen?

Bitte kreuzen Sie die richtige Lösung an.

(3 Punkte)

- ☐ 7
- ☐ 3
- ☐ 5
- ☐ 1,98
- ☐ keine Lösung ist richtig

b) Der Chef-Ingenieur von Limited Performance schlägt eine Verbesserung der Gleitkomma-Einheit vor. Bisher benötigen einfache Befehle (alle Befehle außer Gleitkommabefehlen) im Durchschnitt 2 Taktzyklen und Gleitkommabefehle 10 Taktzyklen. Durch die Verbesserung werden für Gleitkommabefehle nur noch 5 Taktzyklen benötigt, einfache Befehle brauchen dann jedoch 3 Taktzyklen. Außerdem muss die Zykluszeit von 40 ns auf 60 ns erhöht werden. Wie groß muss der Anteil der Gleitkommabefehle mindestens sein, damit sich die Verbesserung lohnt (d.h. insgesamt ein Speedup > 1 erreicht wird)?

Bitte kreuzen Sie den richtigen Wert an.

(3 Punkte)

- ☐ größer als 3/5
- ☐ größer als 4/5
- ☐ größer als 1/2
- ☐ größer als 12/25
- ☐ keine Lösung ist richtig

c) Die Konkurrenzfirma Murx Electronics hat eine neue CPU entwickelt, die mit 500 MHz getaktet werden kann. Die Befehle lassen sich grob in drei Klassen einteilen (vgl. Tabelle).

Befehlsklasse	CPI für die Befehlsklasse
A	1
B	2
C	3

Bei einem Testlauf mit einem Beispielprogramm werden 600 MIPS erreicht. In der folgenden Tabelle sind Benchmarkprogramme mit verschiedenen Verteilungen der Befehle auf die einzelnen Befehlsklassen gezeigt. Welche davon können für den Testlauf verwendet worden sein?

Kreuzen Sie bitte alle richtigen Antworten an:

(3 Punkte)

	Programm	Anzahl der Befehle in Klasse		
		A	B	C
☐	I	$5 * 10^9$	$1 * 10^9$	$1 * 10^9$
☐	II	$10 * 10^9$	$1 * 10^9$	$1 * 10^9$
☐	III	$9 * 10^9$	$3 * 10^9$	0
☐	IV	$10 * 10^9$	$2 * 10^9$	$2 * 10^9$
☐	keines der Programme wurde verwendet			

d) Murx Electronics möchte außerdem eine Prozessor-Variante mit Pipelining auf den Markt bringen. Der Befehlsablauf wird ähnlich wie beim MIPS-Prozessor in 5 Pipelinestufen aufgeteilt. Zur Abarbeitung einer Pipelinestufe wird 1 CPU-Zyklus benötigt. Die Analyse von Benchmarkprogrammen zeigt, dass die Wahrscheinlichkeit für einen Konflikt 12,5 % beträgt. Bei einem Konflikt müssen im Durchschnitt 2 Wartezyklen eingefügt werden. Wie hoch ist der asymptotische Speedup (unendliche viele Befehle) der Pipeline-Implementierung gegenüber einer Mehrzyklen-Implementierung?

Bitte kreuzen Sie alle richtigen Antworten an.

(2 Punkte)

- ☐ 5
- ☐ 0,25
- ☐ 4
- ☐ 2,5
- ☐ keine Lösung ist richtig

Aufgabe 2: (Maschinenstrukturen) 9 Punkte

Auf einem Rechnersystem mit **Akkumulator-Architektur** soll die Berechnung

$$c = a^k + b$$

ausgeführt werden. Nach Beendigung des Programms soll sich c im Speicher befinden. Gehen Sie davon aus, dass die Variablen a und b zu Beginn im Speicher liegen und k konstant ist. Die zur Verfügung stehenden Assemblerinstruktionen sind {`mul, load, store, add`}.

a) Wie lautet der Programmcode der Berechnung und wie viele Befehle werden benötigt (in Abhängigkeit von k)?

(4 Punkte)

b) Die Berechnung kann alternativ auch auf einem Rechnersystem mit **Stack-Architektur** ausgeführt werden. Hierzu stehen die Assemlerinstruktionen {`mul, pop, push, add`} zur Verfügung. Geben Sie wieder den Programmcode der Berechnung in Abhängigkeit von k an. Welche Architektur ist für die gegebene Berechnung bezüglich Anzahl der benötigten Befehle besser geeignet? Begründen Sie Ihre Antwort.

(5 Punkte)

Aufgabe 3: (Datenpfad) 10 Punkte

a) Bei der Verarbeitung von Befehlen kann es im MIPS Datenpfad mit fünfstufiger Pipeline zu Datenhazards kommen. Um diese aufzulösen, können sowohl *Stalls* (Leerzyklen) in die Pipeline eingefügt werden als auch eine *Forwarding*-Einheit integriert werden.

Welche weitere Möglichkeit besteht meistens auch noch?

(1 Punkt)

b) Bei der Verarbeitung folgender Speicher-zu-Speicher Kopiersequenz tritt ein Datenhazard auf.

```
lw  $2, 100($5)
sw  $2, 200($6)
```

Vervollständigen Sie die zweite Zeile der Pipelinebelegung (siehe Abbildung 1) und fügen Sie die minimale Anzahl an *Stalls* (STA) in die Pipeline ein, um den Hazard aufzulösen. Begründen Sie kurz Ihre Antwort.

(3 Punkte)

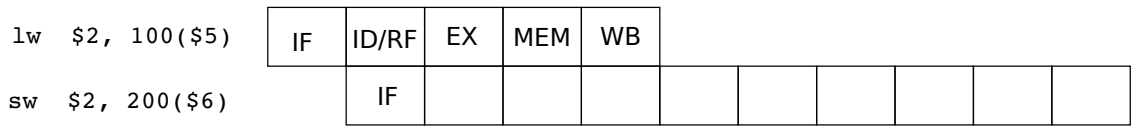


Abbildung 1: Pipelinebelegung

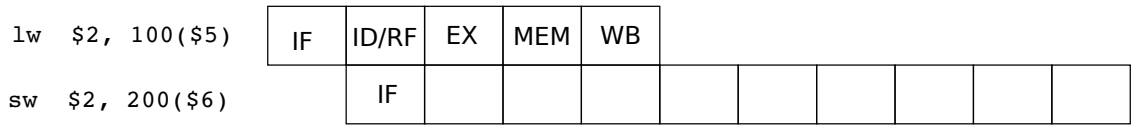


Abbildung 2: Ersatzgrafik, **ungültige Lösung streichen!**

c) Der Datenhazard, der durch die Speicher-zu-Speicher Kopiersequenz aus Aufgabenteil **b)** entstanden ist, soll nun durch eine *Forwarding*-Einheit aufgelöst werden. Modifizieren Sie den Datenpfad in Abbildung 3 entsprechend.

(Ersatzabbildung auf Seite 10, **ungültige Lösung streichen!**)

(3 Punkte)

d) Geben Sie eine mögliche *lw-sw* Instruktionssequenz an, bei der trotz der eingefügten *Forwarding*-Einheit *Stalls* notwendig sind. Begründen Sie Ihre Antwort.

(3 Punkte)

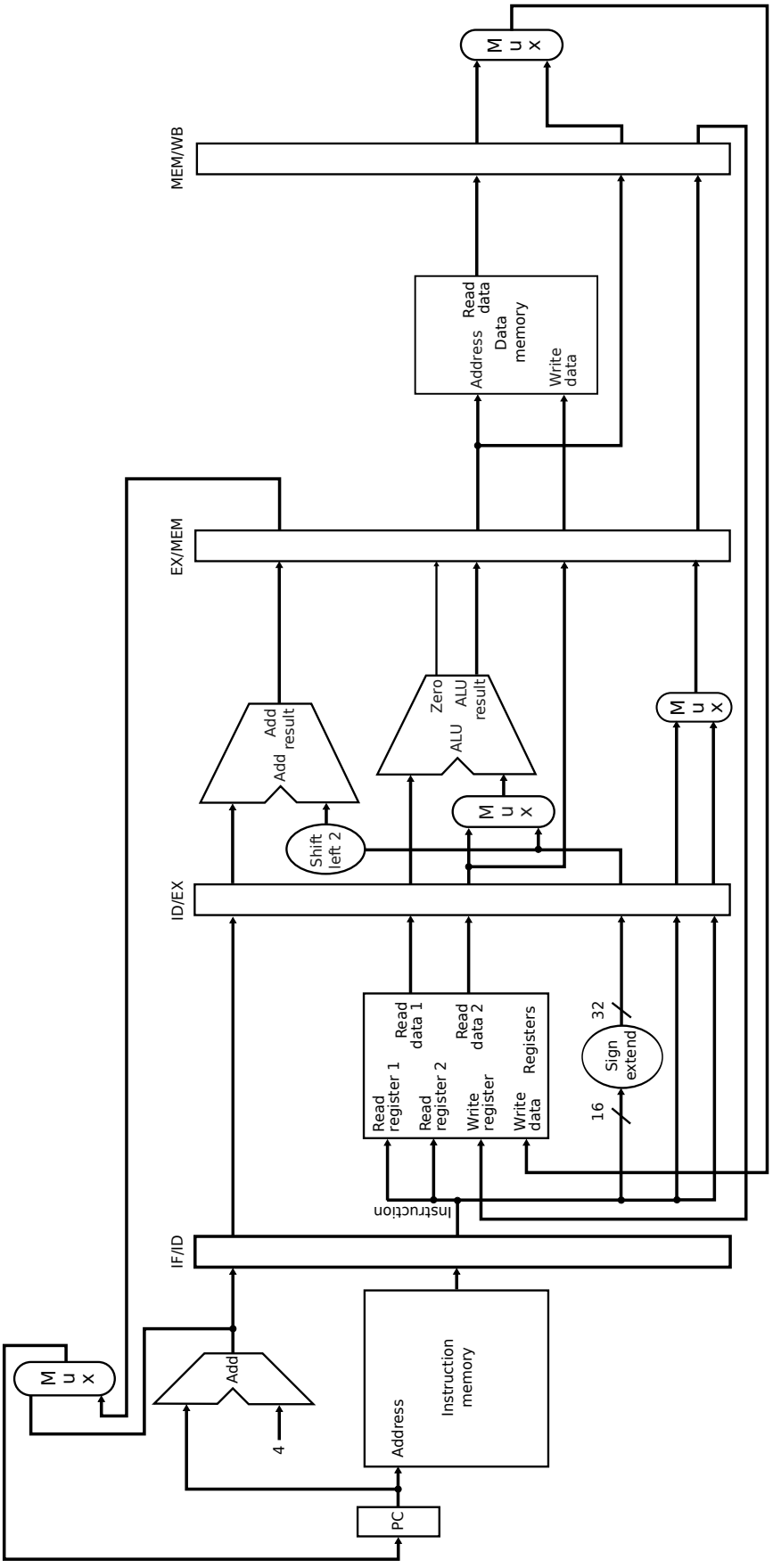


Abbildung 3: Fünfstufige Pipeline des MIPS Prozessors

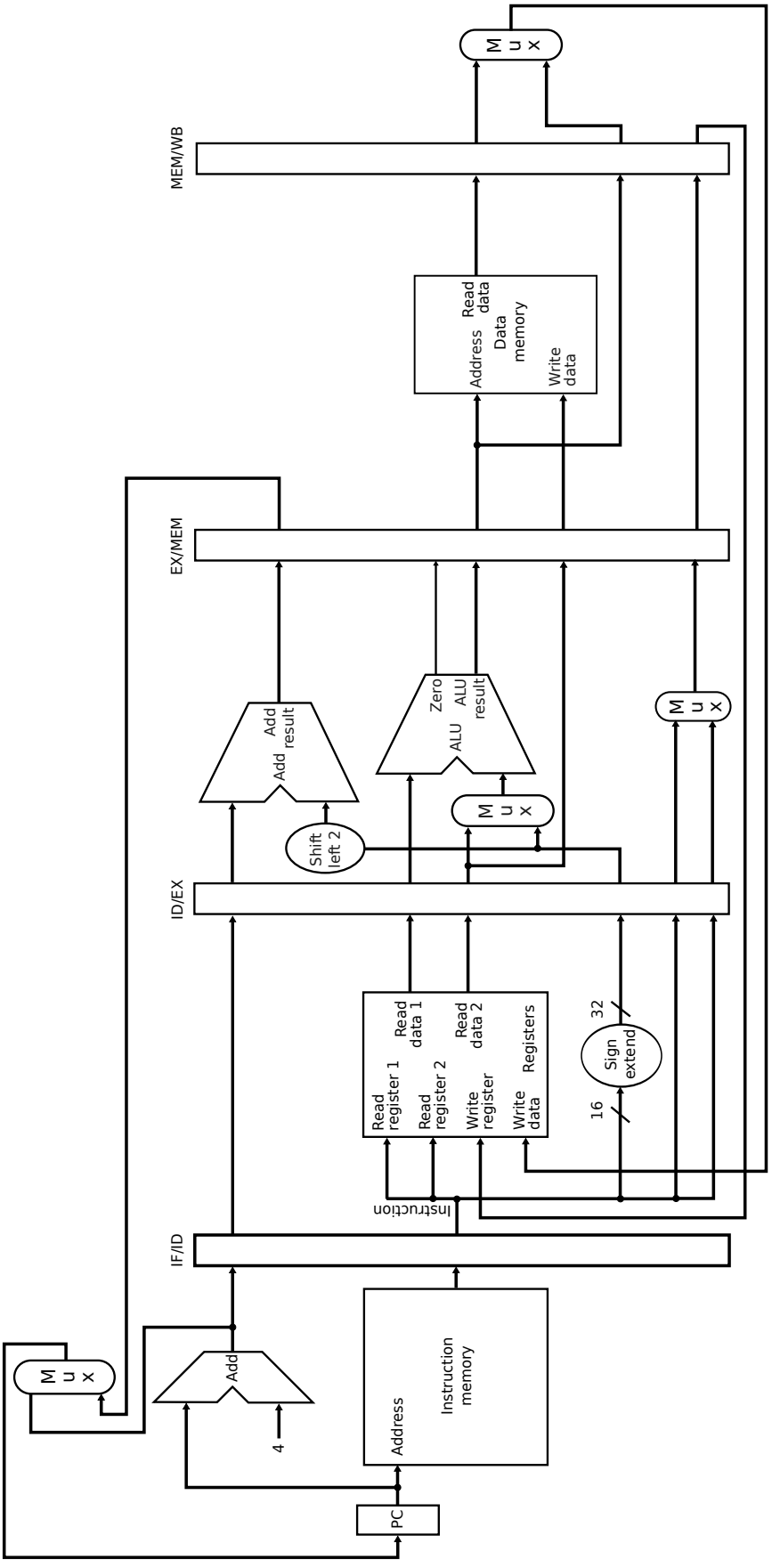


Abbildung 4: Ersatzgrafik, **ungültige Lösung streichen!**

Aufgabe 4: (Pipelining) 20 Punkte

a) Geben Sie Rechenzeit, Speed-Up und Effizienz einer Pipeline mit 10 homogen verteilten Stufen für 991 Instruktionen an! Die sequentielle Abarbeitung eines Befehls betrage 0,1 ms. (2 Punkte)

Rechenzeit sequentiell:

Rechenzeit Pipeline:

Speed-Up:

Effizienz:

b) Geben Sie Rechenzeit, Speed-Up und Effizienz einer Pipeline mit 10 inhomogen verteilten Stufen für 991 Instruktionen an! Die sequentielle Abarbeitung eines Befehls betrage 0,1 ms. Die Verzögerung der Stufen ist in nachfolgender Tabelle angegeben. (4 Punkte)

Stufe	0	1	2	3	4
Verzögerung (ms)	5/1000	1/100	2/100	1/100	1/100
Stufe	5	6	7	8	9
Verzögerung (ms)	1/100	1/100	1/100	1/100	5/1000

Rechenzeit sequentiell:

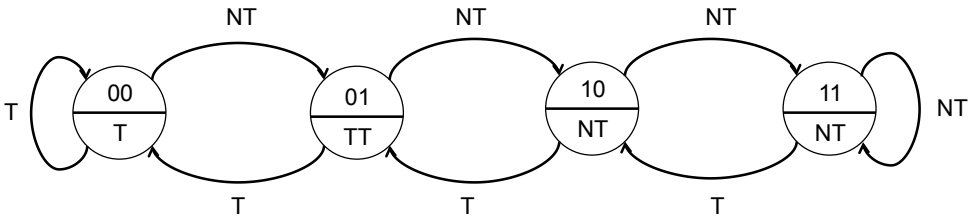
Rechenzeit Pipeline:

Speed-Up:

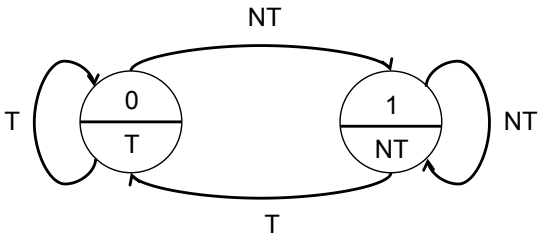
Effizienz:

c) Geben Sie für die Sprungfolge $NT\ NT\ T\ NT\ NT\ T\ T\ T$ die Ausgabe und die Anzahl korrekt vorhergesagter Ereignisse der statischen Sprungvorhersage *always taken* und der Moore-Automaten A und B mit Eingabe- und Ausgabealphabet $\{T, NT\}$ und Zuständen $\{00, 01, 10, 11\}$ bzw. $\{0, 1\}$ an. Die Zustandsüberführungs- und Ausgabefunktionen können den nachfolgenden Abbildungen entnommen werden. Weiterhin seien die Prädiktoren A und B mit 00 bzw. 0 initialisiert.

(6 Punkte)



(a) Prädiktor A



(b) Prädiktor B

Always Taken			
Zeitpunkt	Vorhersage	Sprung	Richtig oder Falsch?
0		NT	
1		NT	
2		T	
3		NT	
4		NT	
5		T	
6		T	
7		T	

Prädiktor A					
Zeitpunkt	Aktueller Zustand	Vorhersage	Sprung	Richtig oder Falsch?	Folgender Zustand
0	00		NT		
1			NT		
2			T		
3			NT		
4			NT		
5			T		
6			T		
7			T		

Prädiktor B					
Zeitpunkt	Aktueller Zustand	Vorhersage	Sprung	Richtig oder Falsch?	Folgender Zustand
0	0		NT		
1			NT		
2			T		
3			NT		
4			NT		
5			T		
6			T		
7			T		

Anzahl der Treffer

***always taken*:**

Prädiktor A:

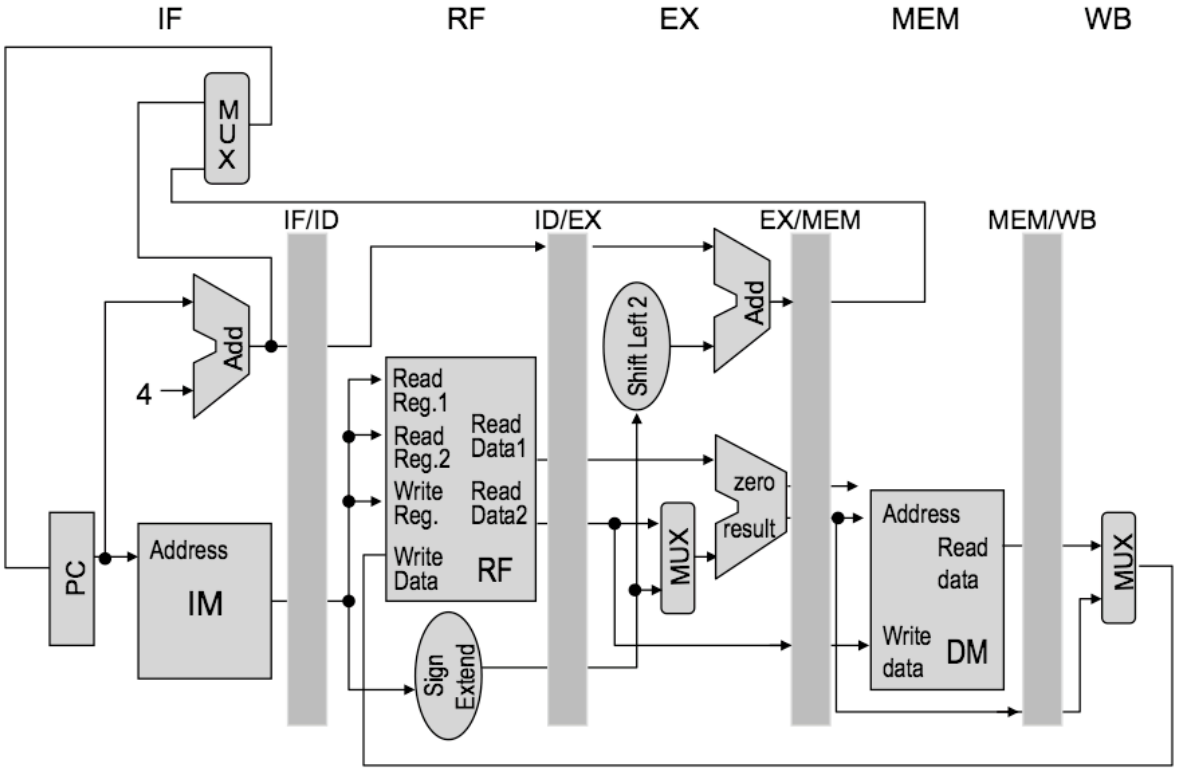
Prädiktor B:

Ersatztabelle Always Taken, **ungültige Lösung streichen!**			
Zeitpunkt	Vorhersage	Sprung	Richtig oder Falsch?
0		NT	
1		NT	
2		T	
3		NT	
4		NT	
5		T	
6		T	
7		T	

Ersatztabelle Prädiktor A, **ungültige Lösung streichen!**					
Zeitpunkt	Aktueller Zustand	Vorhersage	Sprung	Richtig oder Falsch?	Folgender Zustand
0	00		NT		
1			NT		
2			T		
3			NT		
4			NT		
5			T		
6			T		
7			T		

Ersatztabelle Prädiktor B, **ungültige Lösung streichen!**					
Zeitpunkt	Aktueller Zustand	Vorhersage	Sprung	Richtig oder Falsch?	Folgender Zustand
0	0		NT		
1			NT		
2			T		
3			NT		
4			NT		
5			T		
6			T		
7			T		

d) Untersuchen Sie die Pipeline des in der Abbildung dargestellten MIPS-Prozessors! Für alle Befehle gelte die Registerreihenfolge: **destination source**$_1$ **source**$_2$. Es ist keinerlei Forwarding implementiert. Geben Sie für die Programme P1-P3 an, ob ein Daten-, Kontroll-, Strukturhazard oder kein Hazard vorliegt! Geben Sie außerdem die Anzahl der möglichen Wartezyklen an. (6 Punkte)



P1	A:	add	\$t1	\$t2	\$t3
	B:	add	\$t2	\$t3	\$t4
	C:	add	\$t3	\$t4	\$t1
	D:	add	\$t4	\$t5	\$t6
	E:	add	\$t5	\$t6	\$t7
	F:	add	\$t6	\$t3	\$t8
	G:	add	\$t7	\$t8	\$t9

P1:

P2
```
A:  add  $t1   $t2   $t3
B:  lw   $t2   0x99($t3)
C:  add  $t3   $t4   $t5
D:  add  $t4   $t5   $t6
E:  add  $t5   $t6   $t7
```

P2:

P3
```
A:  add  $t1   $t2   $t3
B:  beq  $t2   $t3   F
C:  add  $t3   $t4   $t5
D:  add  $t4   $t5   $t6
E:  add  $t5   $t6   $t7
F:  add  $t6   $t7   $t8
G:  add  $t9   $t10  $t11
```

P3:

e) Zur Auflösung von Kontrollhazards werden verzögerte Sprünge (*delayed branches*) eingeführt, die 2 Delay- Slots vorsehen. Ändern Sie die Befehlsreihenfolge des Programms P3 aus Aufgabenteil **d)**, sodass etwaige Hazards aufgelöst werden ohne die Semantik des Programms zu verändern! Können die Hazards komplett aufgelöst werden? Um wie viel schneller wird das Programm abgearbeitet? Begründen Sie Ihre Entscheidungen!

Hinweis: Die Reihenfolge der Befehle kann über Label angegeben werden.

(2 Punkte)

P3:

Aufgabe 5: (Assembler) 20 Punkte

Nachfolgend ist ein MIPS Assemblerprogramm abgebildet. Die erste Spalte zeigt dabei die Adressen im Hexadezimalsystem, an denen die jeweiligen Instuktionen im Programmspeicher stehen.

```
0x0100  L0:  addi    $sp,     ______, _____   # Lege den Inhalt von
0x0104       sw      $ra,     ______          # $ra und $a0
0x0108       sw      $a0,     ______          # auf den Stack
0x010C       slti    $t0,     $a0,      1
0x0110       ______  $t0,     $zero,   L1     # Sprung zu L1, falls $a0 >= 1
0x0114       addi    $v0,     $zero,    1
0x0118       addi    $sp,     ______,  _____  # Stack Pointer aktualisieren
0x011C       jr      ______                   # Ruecksprung
0x0120  L1:  addi    $a0,     $a0,     -1
0x0124       jal     L0
0x0128       lw      $a0,     ______          # $a0 und $ra vom
0x012C       lw      $ra,     ______          # Stack holen
0x0130       mul     $v0,     $a0,     $v0
0x0134       addi    $sp,     ______,  _____  # Stack Pointer aktualisieren
0x0138       jr      ______                   # Ruecksprung
```

a) Vervollständigen Sie das oben abgebildete Assemblerprogramm gemäß der im Code angegebenen Kommentare.

(10 Punkte)

b) Analysieren Sie eine Ausführung des Assemblerprogramms mit der folgenden initialen Belegung der Register `$sp`, `$a0`, `$v0` und `$ra`:

(8 Punkte)

Register	Wert
`$sp`	`0x1000`
`$a0`	`0x0003`
`$v0`	`0x0000`
`$ra`	`0x0010`

Geben Sie in den beiden folgenden Tabellen die Registerbelegungen und die Belegungen der Speicherstellen zu den ersten 5 Zeitpunkten des Erreichens der Programmzeile an der Adresse 0x0100 an. Geben Sie die Inhalte jeweils vor der Ausführung der Programmzeile an, sofern sie durch die Programmausführung bekannt sind. Alle anderen Stellen lassen Sie leer. Sollte diese Zeile keine 5 mal erreicht werden, lassen Sie auch die verbleibenden Spalten leer.

	1	2	3	4	5
`$sp`	`0x1000`				
`$a0`	`0x0003`				
`$v0`	`0x0000`				
`$ra`	`0x0010`				

Tabelle 1: Registerinhalte

	1	2	3	4	5
`0x1008`					
`0x1004`					
`0x1000`					
`0x0FFC`					
`0x0FF8`					
`0x0FF4`					
`0x0FF0`					
`0x0FEC`					
`0x0FE8`					
`0x0FE4`					
`0x0FE0`					
`0x0FDC`					
`0x0FD8`					

Tabelle 2: Speicherinhalt

	1	2	3	4	5
`$sp`	`0x1000`				
`$a0`	`0x0003`				
`$v0`	`0x0000`				
`$ra`	`0x0010`				

Tabelle 3: Ersatztabelle Registerinhalte, **ungültige Lösung steichen!**

	1	2	3	4	5
`0x1008`					
`0x1004`					
`0x1000`					
`0x0FFC`					
`0x0FF8`					
`0x0FF4`					
`0x0FF0`					
`0x0FEC`					
`0x0FE8`					
`0x0FE4`					
`0x0FE0`					
`0x0FDC`					
`0x0FD8`					

Tabelle 4: Ersatztabelle Speicherinhalt, **ungültige Lösung steichen!**

c) Welchen Wert hält das Register `$v0` nach Abarbeitung des Assemblerprogramms bei einer initialen Belegung von `$a0` mit dem Wert 0x0004?

(2 Punkte)

Aufgabe 6: (Speicherverwaltung) 20 Punkte

Gehen Sie von einem Computer aus, der zur Beschleunigung der Speicherzugriffe auf den byteadressierten Arbeitsspeicher mit einem Adressraum von 2 GB einen Cache mit 8 Rahmen vorsieht, wobei der Datenbereich jedes Blocks 2 Worte zu je 32 Bit umfasst. Der Cache sei als direkt abgebildeter Cache organisiert.

a) Geben Sie die Breite des Adressbus sowie die Anzahl an Bits für Tag, Index und Offsets an.

(3 Punkte)

Adressbus: _____

Tag: _____

Index: _____

Wortoffset: _____

Byteoffset: _____

b) Wie viel Speicher wird für die Implementierung eines solchen Cache (inkl. Overhead) benötigt? Geben Sie den Rechenweg und das Ergebnis in Byte an.

(3 Punkte)

Cache-Größe (in Byte): _____

c) Der Prozessor lädt nun sequentiell die Daten der folgenden Byteadressen (führende Nullen werden nicht mit angegeben):

```
t=1:  0xc25
t=2:  0xb7a
t=3:  0xb04
t=4:  0x415
t=5:  0xbd1
t=6:  0xc21
t=7:  0x804
t=8:  0xfc7
```

Gehen Sie von folgendem Zustand des Cache zum Zeitpunkt $t = 0$ aus:

Index	V	Tag	...
0	0	...100000	...
1	0	...011000	...
2	1	...010000	...
3	1	...110010	...
4	1	...110000	...
5	1	...010100	...
6	1	...011110	...
7	0	...101101	...

Geben Sie den Zustand des Cache nach jedem Speicherzugriff in den folgenden Tabellen an und bestimmen Sie die Hitrate.
Hinweis: Es reicht aus, wenn Sie nur die Zeilen der Tabelle, die sich ändern, angeben.

(8 Punkte)

t=1: Adresse=0xc25

Index	V	Tag
0		
1		
2		
3		
4		
5		
6		
7		

t=2: Adresse=0xb7a

Index	V	Tag
0		
1		
2		
3		
4		
5		
6		
7		

t=3: Adresse=0xb04

Index	V	Tag
0		
1		
2		
3		
4		
5		
6		
7		

t=4: Adresse=0x415

Index	V	Tag
0		
1		
2		
3		
4		
5		
6		
7		

t=5: Adresse=0xbd1

Index	V	Tag
0		
1		
2		
3		
4		
5		
6		
7		

t=6: Adresse=0xc21

Index	V	Tag
0		
1		
2		
3		
4		
5		
6		
7		

t=7: Adresse=0x804

Index	V	Tag
0		
1		
2		
3		
4		
5		
6		
7		

t=8: Adresse=0xfc7

Index	V	Tag
0		
1		
2		
3		
4		
5		
6		
7		

Hitrate: _____

Adresse=

Index	V	Tag
0		
1		
2		
3		
4		
5		
6		
7		

Adresse=

Index	V	Tag
0		
1		
2		
3		
4		
5		
6		
7		

Tabelle 5: Ersatztabelle, **ungültige Lösungen streichen**!

d) Wie hoch ist die durchschnittliche Zugriffszeit, ausgehend von der Hitrate des Cache aus Aufgabenteil **c)**, wenn der Zugriff auf den Cache 20ns und der Zugriff auf den Hauptspeicher 200ns beträgt? Geben Sie auch den Rechenweg an.

(3 Punkte)

Durchschnittliche Zugriffszeit: _____

e) Nehmen Sie an, der Cache sei leer und es sei folgender Ausschnitt des Hauptspeichers gegeben. Die Adressen sind binär (führende Nullen werden nicht mit angegeben), die Daten in hexadezimaler Schreibweise angegeben:

(3 Punkte)

Adresse	Data
10 0000 1000	ff
10 0000 1001	9c
10 0000 1010	ad
10 0000 1011	c8
10 0000 1100	f0
10 0000 1101	da
10 0000 1110	aa
10 0000 1111	4c

Adresse	Data
10 0001 0000	01
10 0001 0001	d3
10 0001 0010	fa
10 0001 0011	c6
10 0001 0100	21
10 0001 0101	01
10 0001 0110	dd
10 0001 0111	df

Adresse	Data
10 0001 1000	1c
10 0001 1001	c3
10 0001 1010	d4
10 0001 1011	cd
10 0001 1100	25
10 0001 1101	e3
10 0001 1110	e2
10 0001 1111	7d

Adresse	Data
10 0010 0000	02
10 0010 0001	13
10 0010 0010	7c
10 0010 0011	51
10 0010 0100	00
10 0010 0101	a0
10 0010 0110	f4
10 0010 0111	22

Es soll nun auf das Wort an der Adresse `0x218` zugegriffen werden. Geben Sie alle Daten an, die in den Cache geladen werden (nach jedem Byte getrennt durch ein Komma):

Übertragene Daten: _____

_____

_____