

HTBLA Hollabrunn - Höhere Abteilung für Elektronik - Technische Informatik

Datum: 20.01.2016	2. Test	HWE	Aufgabe:	Wertung:	Bewertung:
Klasse: 5BHEL	Lehrer:	DUM	1	25	25
Note:	Wertungsskala:		2	15	5+4+2
	0 - 50 Nicht genügend		3	28	26
	51 - 62 Genügend		4	32	28
	63 - 75 Befriedigend				
	76 - 89 Gut				
	90 - 100 Sehr gut				
Zeit: max. 15" + Vorber.	Schüler: Grounham Gruppe A		Summe:	100	92

1) Ein RISC-Prozessor soll eine 5-stufige Phasenpipeline erhalten.

- (5P) Welche Stufen wird diese sinnvollerweise haben? (Bezeichnung entsprechend dem Hauptgeschehen in der jeweiligen Stufe bitte ausschreiben)
- (5P) Angenommen es gäbe keine Pipeline-Hazards. Wieviele Taktzyklen werden im Idealfall benötigt, um ein lineares Programm bestehend aus 10 Befehlen in der 5-stufigen Pipeline fertigzustellen? (nehmen Sie an, dass die Pipeline beim Programmstart leer war).
- (15P) Benennen Sie die drei wichtigsten Pipeline-Hazards und erläutern Sie, wie jeweils mit einer bestimmten Maßnahme Abhilfe geschaffen werden kann.

2) a) (5P) Was versteht man unter einer „superskalaren Prozessorarchitektur“

b) (5P) Vergleichen Sie „in order execution“ mit einer „out of order execution engine“ und argumentieren Sie, warum nahezu alle moderne Superskalarp Prozessoren „out of order“ arbeiten.

c) (5P) Welches besonderes Problem ergibt sich bei „out of order execution engines“ und wie wird dieses gelöst?

7.1) Instruction Fetch, Instruction Decode, Execute, Memory, Write Back

b) 14 Takte

c) Data Hazards: ~~werden~~ Es wird zum Bsp. im Befehl das Ergebnis der vorherigen Befehls gebraucht. RAW -- Read after write; WAR Write after read
einfaches Behalten: Bubbles einführen (Takte gehen verloren); Better bei Superskalaren Out of order mit Register Renaming (z.B. Tomasulo Algorithmus); oder Score-Boarding

Control-Hazards: Branch Hazards → einfache Abhilfe: Bubbles; besser: Branch Prediction (mit Branch Target Buffer letzte Ergebnisse berücksichtigen) wenn falsch vorhergesagt → Flush

Structural-Hazards: Zu wenig Funktionsblöcke, Verbindungen, ... Hardware Lösung: mehr bauen

2) a) mehrere Pipelines parallel

b) In-Order: Befehle werden in der Reihenfolge ausgeführt, wie sie gekettet wurden → Data Hazards → Geschw. Verlust; Out of order kann dies vermeiden, da die Reihenfolge der Befehle sinnvoll durch die HW „gewählt“ wird → schneller

c) ~~Data Hazards~~ Die Reihenfolge der Ergebnisse stimmt oft nicht, deshalb werden sie gepuffert

Complete +
Reg. Renaming +

3) Zeigen Sie, wie der angegebene Befehl in der CPU entsprechend der dargestellten Blockdarstellung in einem Taktzyklus ausgeführt werden kann. **GrA: LW \$09, 0x12 (\$01)** **GrB: SW \$07, 0x01 (\$03)**
Der Befehl steht auf der Programmspeicherstelle 0x A. 0

a) Beschriften sie alle relevanten Leitungsbündel mit den in ihnen transportierten Werten (in Hex), alle nichtbeschrifteten Leitungen seien don't care.

c) Geben Sie die Werte (0, 1) aller relevanten Steuersignale an (alle nichtbeschrifteten Leitungen seien don't care).

Achtung: alle Zahlen bei dieser Aufgabe sind positive Hexadezimalwerte; führende Nullen sollen beim Anschreiben weggelassen werden.

Angaben:

Die Register sollen folgende Inhalte haben:

Reg	Wert	Reg	Wert	Reg	Wert	Reg	Wert
\$00	0x0	\$02	0x24	\$07	0x88	\$0c	0xc0
\$01	0x10	\$03	0x24	\$09	0x999	\$1f	0xd2

Im Datenspeicher stehen folgende Werte:

Adr	Inh	Adr	Inh	Adr	Inh	Adr	Inh
0x00	0x01	0x22	0x89	0x6a		0x74	
0x03	0x33	0x25	0x02	0x6d		0x77	

LW rt, Im16 (rs)

| Load Word

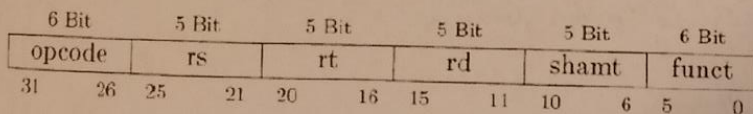
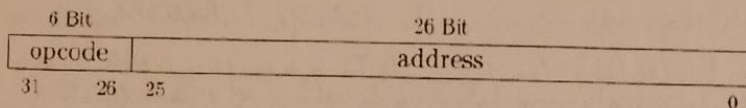
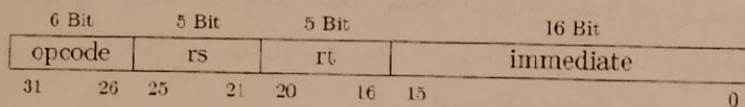
⁰⁹ ⁰⁷ ^{0x12}
 $R[rt] := Mem [R[rs] + Im16]$

SW rt, Im16 (rs)

| Store Word

$Mem [R[rs] + Im16] := R[rt]$

Befehlsformate der MIPS-Architektur:





4) (je 4 P) Ein Rechner soll einen direktabbildenden Cache besitzen. Bearbeiten Sie die folgenden Aufgaben bzw. beantworten Sie die gestellten Fragen (je 4 P).
Angaben:

- Hauptspeichergroße: 2 MiB
- Zugriffsbreite zu Speicher = Breite einer Cacheline: 16 Byte
- der Datenbereich des Cache soll 4 kiB groß sein

- 1/2 a) geben Sie die niedrigste und die höchste Hauptspeicheradresse an (hex).
b) Zerteilen Sie die Adresse in die für der Cachecontroller wichtigen Teile und benennen Sie diese; verwenden Sie als Beispiel die Hauptspeicheradresse: GRA 0x 2 4 6 8 1 Gr B: 0x 5 0 3 4 F
c) Berechnen Sie, wieviele Cachelines der Cache umfasst (Dezimal)
1/2 d) Welche Größe hat der Tagbereich des Caches in BIT (Dezimal) (+ Skizze)
e) nehmen Sie an, dass auf der angegebenen Hauptspeicheradresse der Wert: 0x 44 steht; in welcher Cacheline (Nummer in Hex) steht dieses Byte, falls der Zugriff einen Cache Hit ergibt? (+ Skizze Datenbereich)
f) Was steht dabei im Tag-Bereich im Eintrag mit dieser Nummer? (+ Skizze Tagbereich)
g) Was muss der Cache-controller bei einem Cache-Miss durchführen.
h) Beschreiben Sie den Unterschied zwischen write-back und write through Cache-Strategie?

HS $\rightarrow$ 2 MiB
CL $\rightarrow$ 16 B
Cache $\rightarrow$ 4 kiB

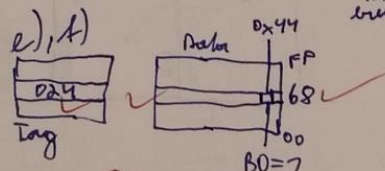
a) $2 \text{ MiB} = 2^{27}$ F
niedrigste 1 F F F F 0 hochste
0 0 0 0 0 niedrigste

b) Cache 4 kiB $= 2^{12}$ CL $= 2^4$
 $\frac{2^{12}}{2^4} = 2^8$ lines ✓

0x { 0 2 4 | 6 8 } 7
Tag Index Byte offset

c) siehe b) 2^8 lines $\rightarrow$ 256 lines ✓

d) Tag $2^9 \rightarrow$ 512 Bit
 $\times 256$
Tag 1 F F
9 bit
0 0 0



g) Der Wert vom Hauptspeicher holen (dann ist länger)
?

h) write-through: Der Cache wird direkt nach dem er verändert wird mit dem Hauptspeicher "synchronisiert"

write-back: Die Cache line wird erst dann "synchronisiert", wenn sie wieder gebraucht wird.

Während der Wert im Cache noch unterschiedlich zu dem im Hauptspeicher ist $\rightarrow$ dirty Bit