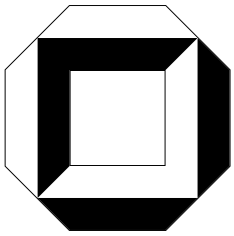


Universität Karlsruhe
Institut für Rechnerentwurf und Fehlertoleranz
Prof. Dr. Wolfgang Karl

Klausur Rechnerstrukturen
Wintersemester 2004/05
Musterlösung

Aushang der Ergebnisse: ab 18. April 2005

Musterlösung 1 (Leistungsbewertung) *10 Punkte*

a) $Zyklen = Codesize * CPI,\ Takt = \frac{Zyklen}{Zeit} = \frac{Codesize*CPI}{Zeit}$ **2P**

A: $t = \frac{3*10^6*\frac{4}{3}}{\frac{8}{5}*10^{-3}} = \frac{5}{2} * 10^9 = 2.5GHz$

B: $t = \frac{\frac{8}{5}*10^6*\frac{5}{4}}{2*10^{-3}} = 1 * 10^9 = 1GHz \rightarrow$ Prozessor A hat den höheren Wert.

b) MIPS = takt/CPI **1P**
A: $\frac{\frac{5}{2}*10^9}{\frac{4}{3}} = \frac{15}{8} * 10^3 MIPS$
B: $\frac{10^9}{\frac{5}{4}} = \frac{4}{5} * 10^3 MIPS \rightarrow$ Prozessor A hat den besseren Wert.

c) CPU2000 besteht aus SPECint und SPECfp zur Messung von Integer- und FPU-Leistung ($\frac{1}{2}$P). Die Messung kann bezüglich Geschwindigkeit (SPECint/fp) und Durchsatz (SPECint/fp_rate) erfolgen ($\frac{1}{2}$P). Bei der Messung kann entweder konservativ optimiert werden (Standardoptimierung, identische Compileroptionen für alle Programme) oder aggressiv und individuell (1P). **2P**

d) **5P**

(d.1) Anzahl Instruktionen: $(500 + 25 + 200 + 75) * 10^3 = 800 * 10^3$ *4P*

Taktzyklen: $(500 * 2 + 25 * 10 + 200 * 5 + 75 * 2) * 10^3 = 2400 * 10^3$

Zykluszeit: $\frac{1}{2.5GHz} = \frac{1}{\frac{5}{2}GHz} = \frac{2}{5} * 10^{-9} = 0.4 * 10^{-9}s = 0.4ns$

Ausführungszeit: $Zyklen * Zykluszeit = (2400 * 10^3) * (\frac{2}{5} * 10^{-9})s = 960 * 10^{-6}s = 960\mu s$
CPI: $\frac{Zyklen}{Instruktionen} = \frac{2400*10^3}{800*10^3} = \frac{24}{8} = 3$

MIPS: $\frac{Instruktionen}{Ausführungszeit*10^6} = \frac{800*10^3}{960} = \frac{5}{6} * 10^3 = 833\frac{1}{3}\ MIPS$

MFLOPS: wie MIPS, wobei Anzahl der Befehle und Ausführungszeit nur für Fließkommaberechnung
$\frac{25*10^3}{(25*10^3*10)*(\frac{2}{5}*10^{-9})*10^6} = \frac{1}{4*10^{-3}} = 2500\ MFLOPS$

(d.2) Es finden überproportional viele Speicherzugriffe statt, offenbar erzeugt der Benchmark viele Register-Verdrängungen. Diesem Umstand könnte mit einer Vergrößerung der Registeranzahl begegnet werden. *1P*

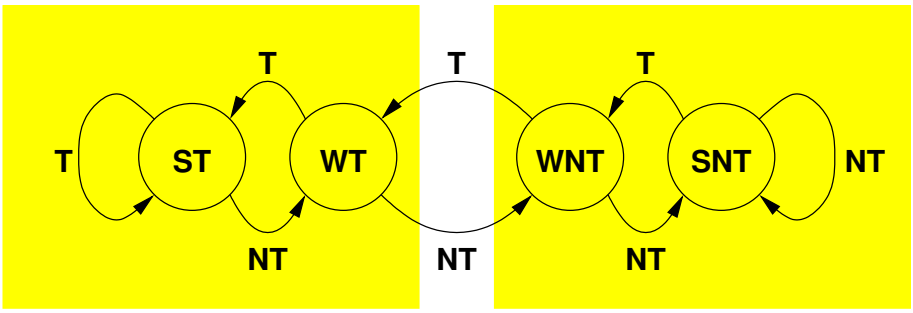
Musterlösung 2 (Prozessorarchitektur) *15 Punkte*

a) Always taken (immer springen) und always not taken (nie springen) **1P**

b) Always taken (immer springen) ($\frac{1}{2}$P), Schleifen werden bis zur Abbruchbedingung korrekt vorhergesagt. ($\frac{1}{2}$P) **1P**

c) Bei Fehlvorhersage Änderung der grundsätzlichen Vorhersage, d.h. Kippen von *Taken* nach *Not Taken*. ($\frac{1}{2}$P) Hierdurch wird bei aufeinanderfolgenden Schleifendurchläufen eine Fehlvorhersage beim Schleifenanlauf erzeugt. ($\frac{1}{2}$P) **1P**

d) Zustandsautomat mit 4 Zuständen bestehend aus zwei strongly/weakly-Zustandspaaren. Ausgang eines Sprungs bedingt Zustandswechsel. Durch Einführung der *weakly*-Zustände erfolgt Vorhersagewechsel erst nach zwei Fehlvorhersagen. **2P**



e) Ein (m,n)-Korrelationsprädiktor nutzt das Verhalten der letzten m Sprünge für die Auswahl aus 2^m Prädiktoren. Jeder Prädiktor stellt einen n-Bit Prädiktor für den jeweils einzelnen Sprung dar. Die globale Vergangenheit der letzten m Sprünge kann in einem m-Bit Schieberegister, dem Sprungverlaufsregister/Branch History Register (BHR), gespeichert werden. Der Inhalt des BHR wird als Adresse benutzt, um eine Sprungsverlaufstabelle (Pattern History Table, PHT) zu selektieren. **2P**

f) Der (m,n)-Korrelationsprädiktor kann durch die Auswertung des Sprungverlaufes mittels BHR und PHT Sprungmuster erkennen. **1P**

g) Für $m = 0$, d.h. das Weglassen der Historie, ist der Korrelationsprädiktor identisch zu einem einfachen n-Bit Prädiktor. **1P**

h) Bedingte Ausführung bzw. Predication ($\frac{1}{2}$P): Zur Laufzeit werden die voneinander unabhängigen Befehle zur Ausführung angestoßen ($\frac{1}{2}$P), der Prozessor führt die Befehle auf den möglichen Progra mmverzweigungen aus, aber speichert die hierbei Ergebnisse nicht endgültig. ($\frac{1}{2}$P), Validierung durch Überprüfung der Predicate Register (1: Instruktion abgeschlossen, 0: Ergebnis verwerfen) ($\frac{1}{2}$P). **2P**

i) Instruction Fetch (IF), Instruction Decode (ID), Operand Fetch (OF), Execution (EX), Result Writeback (WB). **1P**

j) Die Durchlaufgeschwindigkeit der Elektronen ist endlich. Somit muß die Komplexität einer Pipelinestufe verringert werden, wodurch bei gleicher Gesamtfunktionalität mehr Pipelinestufen notwendig werden. **1P**

k) RISC-Befehle haben einheitliche feste Länge und (weitgehend) einheitliche Kodierung, als Operanden können nur Register verwendet werden. CISC-Befehle hingegen können variable Befehlslängen aufweisen, Operanden können auch im Hauptspeicher liegen. **2P**

Hierdurch muß der Zyklus IF/ID ggf. mehrfach durchlaufen werden (vgl. Prä- und Post-Bytes), bis der Befehl vollständig geladen und dekodiert ist. Weiterhin muß der Zugriff auf Operanden im Hauptspeicher serialisiert werden (konsekutive Hauptspeicherzugriffe).

Musterlösung 3 (Speicherhierarchie/Caches) *10 Punkte*

a) Wortadresse w: 8 Datenworte, also 3 Bit **3P**
Index i: $log_2(Sätze)$
Tag t: $48 - i - w = 45 - i$

Cache	Sätze	Blöcke/ Satz	i	t
direct-mapped	32	1	5	40
4-way s.a.	8	4	3	42
fully assoc.	1	32	0	45

b) Freiere Auswahl des Blocks, dadurch weniger Verdrängung. **1P**

c) Geringere Hardware-Komplexität bei vergleichbar geringer Verdrängung **1P**

d) $r_M = 1 - r_H = 1 - 0.8 = 0.2$ **1P**
$t_A = r_H * t_H + r_M * t_M = (2.5 * 0.8 + 0.2 * 250)ns = (2 + 50)ns = 52ns$

e) Verbesserung ist nur durch Erhöhung von r_H möglich. Dies kann erfolgen durch Wechsel der Cache-Organisation (weniger Verdrängung) oder durch Einfügen einer weiteren Cache-Hierarchieebene, d.h. eines L2-Caches, so daß insgesamt weniger Hauptspeicherzugriffe erfolgen. **2P**

f) Hauptspeicher basiert auf DRAM-Technologie; DRAM-Zellen sind vergleichsweise klein und kostengünstig zu fertigen, bedingen aber eine vergleichsweise hohe Zugriffszeit und erfordern Refresh. Cache-Speicher basiert auf SRAM-Technologie. SRAM-Zellen sind 4-6x größer als DRAM-Zellen und entsprechend teuer zu fertigen. Es hat also in erster Linie wirtschaftliche Gründe, daß man den Hauptspeicher nicht aus Cache-Speicher fertigt. **2P**

Musterlösung 4 (Cache-Kohärenz und Parallelverarbeitung) *15 Punkte*

a) MESI kommt in speichergekoppelten Systemen mit gemeinsam genutztem Hauptspeicher, sogenannten UMA-Systemen, zum Einsatz. (1P) Die zwei Basiseigenschaften sind: **2P**

- Alle Prozessoren greifen gleichermaßen auf den gemeinsamen Speicher zu. *0.5P*
- Die Zugriffszeit aller Prozessoren auf den gemeinsamen Speicher ist gleich. *0.5P*

b) Bus Snooping, d.h. das Mitschnüffeln von Buszugriffen. Deshalb müssen die Prozessoren im System Zugriffe aller anderen Prozessoren mitlesen können (gemeinsamer Bus). **1P**

c) Die zuständige Einheit ist der Cache-Controller. Um das MESI-Protokoll zu unterstützen, werden drei Signalleitungen implementiert: invalid (zur Invalidierung von Einträgen in anderen Caches), shared (zur Anzeige, ob ein extern zu ladender Block bereits als lokale Kopie vorhanden ist) und retry (Aufforderung an externen Prozessor, den Speicherzugriff zu unterbrechen und nach Rückschreiben der lokalen Kopie wieder aufzunehmen). Weiterhin müssen Cache-Zeilen um zwei Zustandsbits erweitert werden, um die vier Zustände des MESI-Protokolls kodieren zu können. **3P**

d) **3P**

- Block ist eine exklusive lokale und modifizierte Kopie. *1P*
- Bei Zugriff durch den entfernten Prozessor wird dieser mittels Retry-Signal unterbrochen und die Kopie zurückgeschrieben. Dem entfernten Prozessor wird dann (ebenfalls durch das Retry-Signal) signalisiert, seinen Zugriff zu wiederholen. *2P*
- Je nach Art des Zugriffes durch den entfernten Prozessor wird der lokale Prozessor den Zustand für diese Cache-Zeile nach *shared* (bei Lesezugriff) oder *invalid* (bei Lesezugriff mit Schreibabsicht bzw. Schreibzugriff) ändern. *1P*

e) **3P**

- $E(n) = \frac{S(n)}{n} \rightarrow E(16) = \frac{S(16)}{16} = \frac{12}{16} = \frac{3}{4} = 0.75$ *1.5P*

  $S(n) = \frac{T(1)}{T(n)} \rightarrow T(16) = \frac{T(1)}{S(16)} = \frac{128}{12} = \frac{32}{3} = 10\frac{2}{3}$

  $I(n) = \frac{P(n)}{T(n)} \rightarrow I(16) = \frac{P(16)}{T(16)} = \frac{48}{\frac{32}{3}} = \frac{48*3}{32} = \frac{9}{2} = 4.5$

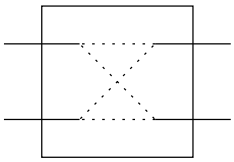
- Amdahls Gesetz: $T(n) = T(1) * (\frac{(1-a)}{n} + a)$ *1.5P*
  $\rightarrow \frac{32}{3} = 128 * (\frac{(1-a)}{16} + a)$

  $\rightarrow \frac{32}{3*128} = \frac{(1-a)}{16} + a \rightarrow \frac{32*16}{3*128} = (1-a) + 16a = 1 + 15a$

  $\rightarrow \frac{4}{3} - 1 = 15a \rightarrow \frac{1}{3} = 15a$
  $\rightarrow a = \frac{1}{45}$

f) Diskonnektivität: $d = \frac{N}{e} \rightarrow \frac{24}{6} = 4$ **1P**
Kosteneffektivität: Verbindungsgrad*max(Diameter,d)=$4 * max(3, 4) = 16$

g) Es handelt sich um Zweierschalter mit dem folgenden Schaltsymbol: **1P**



Musterlösung 5 (Fehlertoleranz) *10 Punkte*

a) Sogenannte *paceful degradation*, d.h. das System ist bis zum Totalausfall einsatzfähig, allerdings bei Teilausfall mit nur eingeschränkter Leistungsfähigkeit. **1P**

b) **8P**

(b.1) *1P*

```
            +--TX1
S1--S2--M--+
            +--TX2
```

(b.2) S=S1 and S2 and M and (TX1 or TX2) *1P*

(b.3) Parallele Sendeeinheiten: $\Phi(txp) = 1 - (1 - \Phi(TX))^2$ *1P*
Gesamtsystem also: $\Phi(S) = \Phi(SP)^2 * \Phi(M) * (1 - (1 - \Phi(TX))^2)$

(b.4) *2P*

Dreifach-Redundanz (TMR) mit Entscheider, d.h. Verdreifachung des Controllers und Vergleich der Ausgaben durch einen Entscheider; das resultierende System ist ein zwei-aus-drei-System.

```
M1--+
M2--+--V--
M3--+
```

$$\Phi(TMR + V) = \Phi(V) * \sum_{k=2}^{3} \binom{3}{k} * \Phi(M)^k * (1 - \Phi(M))^{3-k}$$

(b.5) $\Phi(alt) = 0.9999 * \Phi(SP)^2 * (1 - (1 - \Phi(TX))^2)$ *3P*
Neues System:

$$\Phi(M_{123}) = \sum_{k=2}^{3} \binom{3}{k} * 0.9^k * (1 - 0.9)^{3-k} = \sum_{k=2}^{3} \binom{3}{k} * 0.9^k * 0.1^{3-k}$$

$$= 3 * 0.9^2 * 0.1^1 + 1 * 0.9^3 * 0.1^0 = 0.9^2 * (0.3 + 0.9) = 0.972$$

$\Phi(M_{neu}) = \Phi(M_{123}) * \Phi(V) = 0.972 * 0.9 \ll 0.9999$
Das TMR-System weist aufgrund der schlechten Ausfallraten der Einzelkomponenten eine schlechtere Funktionswahrscheinlichkeit auf, weswegen in dieser Konfiguration dem Ur-System der Vorzug zu geben ist.

c) Die Garantie umfaßt lediglich Initialausfälle (Produktionsfehler); von der Langlebigkeit seiner Produkte ist er nicht überzeugt, d.h. er erwartet hohe Betriebs- und Altersausfälle. **1P**