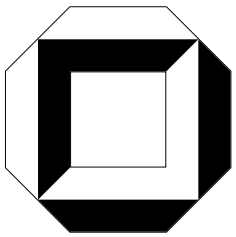


Universität Karlsruhe (TH)
Institut für Technische Informatik
Prof. Dr. Wolfgang Karl

Klausur Rechnerstrukturen
Sommersemester 2009
Musterlösung

Aushang der Ergebnisse: ab Ende September 2009

Musterlösung 1: Quantifizierung *10P*

a) **1P**

$S(f1) > S(f2)$ *0,5P*

$D(f1) < D(f2)$ *0,5P*

b) (1P) Statisch: P_{static}, $P_{leakage}$ **2,5P**

(1P) Dynamisch: $P_{switching}$, $P_{shortcircuit}$

($\frac{1}{2}$P) Aufgrund der Miniaturisierung leisten heutzutage Leckströme ($P_{leakage}$) einen wesentlichen Beitrag zur Leistungsaufnahme.

c) **2P**

wahr	falsch	
□	⊠	Der Prozessortakt ist ein hinreichendes Maß für die Leistungsfähigkeit einer Architektur.
⊠	□	CPI ist programmabhängig.
□	⊠	MIPS-Werte erlauben einen direkten Vergleich von Rechnern mit unterschiedlicher Befehlssatzarchitektur (ISA).
□	⊠	Compilertechnologie spielt keine Rolle bei der Bewertung der Rechnerleistung.

d) *(je $\frac{1}{2}$P für Gesetz und Bennenung)* **1P**

Gesetz von Little: $Q = W * D$ mit

Q: Anzahl von Aufträgen in der Warteschlange

W: Wartezeit

D: Durchsatz

e) (1P) $yield_{die} = yield_{wafer} * (1 + \frac{dpua * a_{die}}{\alpha})^{-\alpha}$ **2,5P**

($\frac{1}{2}$P) Die Wafer-Größe hat bei ansonsten gleichen Fertigungsparametern keinen Einfluss auf die Die-Ausbeute. (Daher ist die Werbeaussage hinreichend sinnlos.)

(1P) Eine Vergrößerung des Wafers führt zu einem Anstieg der Anzahl erzielbarer Dies pro Wafer ($\frac{1}{2}$P) und damit typischerweise zu geringeren Fertigungskosten / niedrigerem Preis ($\frac{1}{2}$P).

f) Chip im Gehäuse oder „nackter“ Die für Bond-out. *(*jeweils $\frac{1}{2}$P) **1P**

Musterlösung 2: Hardwareentwurf *10P*

a) Die Realisierung eines neuen Befehls ist weitgehend durch die bereits vorliegenden Befehle festgelegt ($\frac{1}{2}$P). Allgemein gilt: Bei Kenntnis eines Teils des Systems muss der Rest vorhersagbar sein ($\frac{1}{2}$P). **1P**

b) Verletzt wird hier die Angemessenheit ($\frac{1}{2}$P), da Prozessor und Speicher nicht aufeinander abgestimmt sind ($\frac{1}{2}$P). **1P**

c) Component und Package. *(je $\frac{1}{2}$P)* **1P**

d) Eingabespezifikation auf höherer Ebene, Ausschöpfung des Entwurfsraums, Flexibilität und reduzierte Fehleranfälligkeit *(je $\frac{1}{2}$P)* **2P**

e) Verhaltens-, Struktur- und Datenverfeinerung. *(je $\frac{1}{2}$P)* **1,5P**

f) Aufgrund der Formulierung des Prozesses dominiert die Verzögerung bei der Zuweisung an das Signal `flag` ($\frac{1}{2}$P), daher ist auf den Wert `X"ff"` zu prüfen ($\frac{1}{2}$P). **1P**

g) `count` ist eine Variable ($\frac{1}{2}$P), diese ist außerhalb des Prozesses nicht sichtbar ($\frac{1}{2}$P). **2,5P**
($\frac{1}{2}$P) `count` muss daher Signal sein, nicht Variable
($\frac{1}{2}$P) Zuweisung folglich mit <=-Operator, nicht `:=`.
($\frac{1}{2}$P) Es handelt sich nun um eine nebenläufige Zuweisung, daher erfolgt die Prüfung auf `count=X"00"`.

Musterlösung 3: Prozessorarchitektur *10P*

a) Formeln: **1,5P**

- Ausführungszeit in Architektur ohne Pipeline: $T_{\text{unpipelined}}(n,k) = n * k$ *0,5P*
  Ausführungszeit in Architektur mit Pipeline: $T_{\text{pipelined}}(n,k) = n + k - 1$
- Speedup: $S(n,k) = \frac{T_{\text{unpipelined}}(n,k)}{T_{\text{pipelined}}(n,k)} = \frac{n*k}{n+k-1}$ *0,5P*
- Durchsatz: $D(n,k) = \frac{\text{\# Instruktionen}}{\text{Zeit}} = \frac{n}{n+k-1}$ *0,5P*

b) 6-stufige Pipeline mit Konfliktbehandlung: **2P**

- Ausführungszeit: *1P*
  $T = 15000 + 6 - 1 + \frac{15000}{3} + 3 * \frac{15000}{5} = 15005 + 5000 + 9000 = 29005$
- Speedup: $S = \frac{6*15000}{29005} = \frac{90000}{29005} \rightarrow \frac{90000}{30000} = 3 < \frac{90000}{29005} = S < \frac{90000}{22500} = 4$ *0,5P*
  Damit: $3 < S < 4$
- Durchsatz: $D = \frac{15000}{29005} \sim \frac{15}{29} \rightarrow \frac{15}{30} = \frac{1}{2} = 0.5 < \frac{15}{29} = D < \frac{15}{25} = \frac{3}{5} = 0.6$ *0,5P*
  Also: $0.5 < D < 0.6$

c) 2-Bit-Hysteresezähler: (0,5P pro gänzlich korrektem Sprung) **1P**

Sprung	Prädiktor	Vorhersage	Ausgang	Aktualisierter Prädiktor
1	WT	T	T	ST
2	WT	T	NT	SNT
1	ST	T	NT	WT
2	SNT	NT	T	WNT

d) (1,1)-Korrelationsprädiktor: (0,5P pro korrekter Zeile) **2P**

Aktuelle Vorhersage			Ausgang	Aktualisierte Vorhersage	
BHR	Prädiktor	Vorhersage		BHR	Prädiktor
T	(T,T)	T	NT	NT	(T,NT)
NT	(T,NT)	T	NT	NT	(NT,NT)
NT	(NT,NT)	NT	T	T	(T,NT)
T	(T,NT)	NT	NT	NT	(T,NT)

e) Nach Takt 4 **3,5P**

Registerstatustabelle: (0,5P je korrekter Zeile)

Register	1	2	3	4	5	6
Wert	–	(R4)-(R3)	(R3)	–	(R5)	–
Gültig?	0	1	1	0	1	0
Reservation Station (RS#)	3	0	0	2	0	4

Reservation Stations: (0,5P je korrekter Zeile)

RS#	Leer	In FU	Opcode	Ziel	Quelle1	Gültig1	RS1	Quelle2	Gültig2	RS2
1 Addierer	1	0	sub	2	(R4)	1	0	(R3)	1	0
2 Addierer	0	0	add	4	(R4)-(R3)	1	0	(R3)	1	0
3 Multiplizierer	0	1	mul	1	(R3)	1	0	(R5)	1	0
4 Dividierer	0	0	div	6		0	3	(R4)	1	0

Musterlösung 4: Parallelverarbeitung *10P*

Quantitative Maßzahlen: *3P*

a) ($\frac{1}{2}$P) Amdahls Gesetz: **1P**

$$T(n) = \underbrace{\frac{T(1)}{n} * (1-a)}_{1} \; + \; \underbrace{T(1) * a}_{2}$$

($\frac{1}{2}$P) Es gilt: a mit $(0 \leq a \leq 1)$ ist der Anteil des Programms, der nur sequentiell ausgeführt werden kann.
Die Formel zerfällt in die Ausführungszeit des parallel ausführbaren Programmteils 1 und des rein sequentiell ausführbaren Programmteils 2.

b) ($\frac{1}{2}$P) Der Anteil an sequentiellen Operationen begrenzt die mit einem Parallelrechner erreichbare Beschleunigung. **1P**

$$S(n) = \frac{1}{a} \text{ für } n \to \infty$$

($\frac{1}{2}$P für Formel)

c) Bei fester Problemgröße und steigender Prozessorzahl wird ab einer bestimmten Prozessorzahl eine Sättigung eintreten ($\frac{1}{2}$P). Die Skalierbarkeit ist in jedem Fall begrenzt ($\frac{1}{2}$P). **1P**

Parallele Programmiermodelle: *2P*

d) **2P**

- Multiprogramming
- Gemeinsamer Speicher (Shared Address Space)
- Nachrichtenorienties Programmiermodell (Message Passing)
- Datenparallelismus

(Jeweils $\frac{1}{2}$P pro korrekter Antwort)

Verbindungsnetze und -strukturen: *2P*

e) Formel ($\frac{1}{2}$P): **1P**

$$T_{msg} = t_s + t_w * L$$

($\frac{1}{2}$P) Startzeit t_s, Transferzeit t_w, Anzahl der Datenwörter L

f) ($\frac{1}{2}$P) Reguläres Permutationsnetzwerk: p Eingänge, p Ausgänge, k Stufen mit je $p/2$ Zweierschaltern, (wobei die Zahl p normalerweise eine Zweierpotenz ist). **1P**
($\frac{1}{2}$P) Irreguläre Permutationsnetzwerke weisen gegenüber der vollen regulären Struktur Lücken auf.

Vektorverarbeitung: *3P*

g) Vektorisierung der innersten Schleife ($\frac{1}{2}$P) mittels eines vektorisierenden Compilers ($\frac{1}{2}$P). **1P**

h) Durch Verkettung von Vektorbefehlen ($\frac{1}{2}$P) oder durch einen Vektor-Verbundbefehl (bsp. Vector-Multiply-Add, $\frac{1}{2}$P). **1P**

i) Zugriff auf nichtsequentielle Speicherzellen ($\frac{1}{2}$P) und Umformen in dichte Struktur ($\frac{1}{2}$P). **1P**

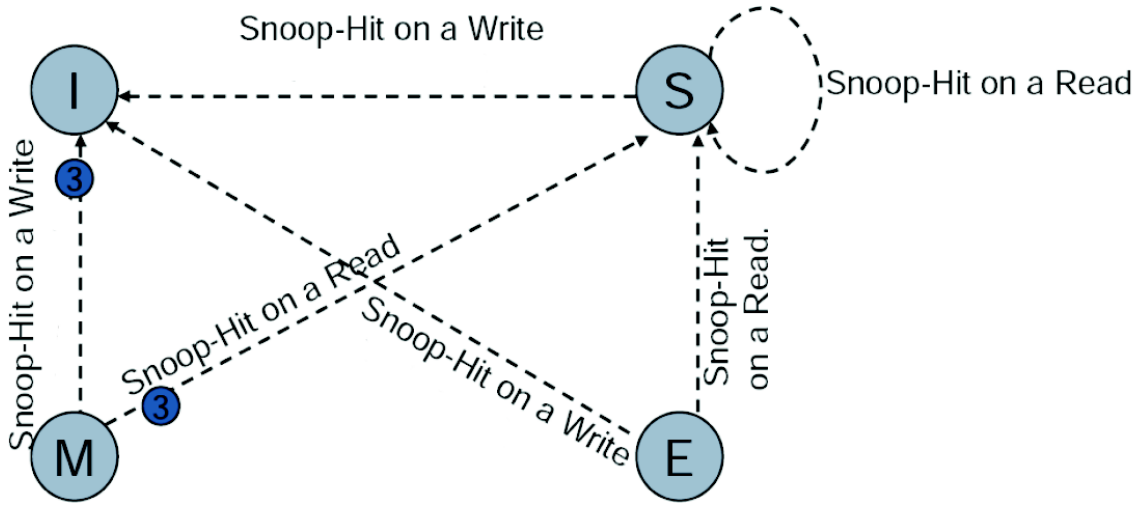
Musterlösung 5: Speicherhierarchie **10P**

a) Typen: True-Sharing-Miss und False-Sharing-Miss ($\frac{1}{2}$P) **1,5P**
Durch geschickte Umordnung der Daten lassen sich False-Sharing-Misses vermeiden. ($\frac{1}{2}$P) Die Daten müssen so im Cache angeordnet werden, so dass Daten, die von unterschiedlichen Prozessoren benötigt werden, auf unterschiedliche Cachezeilen abgebildet werden. ($\frac{1}{2}$P)

b) (wesentliche Antwortteile *markiert*) **1,5P**

- Bei konsekutiven Zugriffen *steigt die Hit-Rate* und somit die Leistung des Gesamtsystems, da *weniger Cachezeilen zur Aufnahme des Working-Sets* benötigt werden. *0,5P*
- Die durch wahlfreien Zugriff erzeugten Cache-Misses werden durch die Vergrößerung der Cachezeile *teurer*, da *mehr Daten aus dem Hauptspeicher geladen werden müssen*, dadurch sinkt in der Regel die Leistung des Gesamtsystems. *0,5P*
- *Der physikalische Aufbau wird komplexer*, da pro Verdopplung der Cachezeilengröße zur Selektion des Wortes innerhalb einer Cachezeile ein zusätzliche Auswahlleitung sowie hieraus resultierend eine (mehr als) Verdopplung der Auswahllogik erforderlich ist. Alternativ wäre ein Zeitscheibenverfahren (Zeit-Multiplexing) denkbar, auch dies resultiert jedoch in einem komplexeren Aufbau. *0,5P*

c) Für jede falsche Kante oder falsche Beschriftung $\frac{1}{2}$P Abzug. **3P**



d) *($\frac{1}{2}$P Abzug pro Fehler)* **4P**

Prozessor	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Line 1	Line 2	Line 1	Line 2	Line 1	Line 2
	init	-	-	-	-	-	-
2	rd 3			3/E			
1	rd 2	2/E					
1	rd 3		3/S	3/S			
3	rd 3		3/S	3/S		3/S	
2	wr 3		3/I	3/M		3/I	
2	rd 1				1/E		
1	wr 3		3/M	3/I			
2	rd 2	2/S		2/S			
3	rd 3		3/O			3/S	
3	wr 1				1/I		1/M
1	rd 4	4/E					
1	rd 2		2/S	2/S			

Musterlösung 6: Fehlertoleranz *10P*

a) *(Je korrekter Antwort $\frac{1}{2}$P)* **2P**

wahr	falsch	
□	⊠	Bei Unterlassungsausfällen können auch fehlerhafte Ergebnisse ausgegeben werden.
□	⊠	Haftausfälle führen dazu, dass eine fehlerhafte Komponente nie mehr ein Ergebnis ausgibt.
□	⊠	Bei einem Anhalteausfall gibt die fehlerhafte Komponente ständig den gleichen Ergebniswert aus.
□	⊠	Teilausfall schließt den Ausfall aller Funktionen einer fehlerhaften Komponente ein.

b) ($\frac{1}{2}$P) Punktverfügbarkeit V: $V = \frac{MTTF}{MTTF+MTTR}$ bzw. $V = \frac{E(L)}{E(L)+E(B)}$ **1,5P**

($\frac{1}{2}$P) Für V gilt $\lambda \sim konstant$

($\frac{1}{2}$P) Dies gilt für die Betriebsphase der Badewannenkurve.

c) 1. Ungenutzte Redundanz ($\frac{1}{2}$P), Ersatzkomponenten bleiben bis zur fehlerbedingten Aktivierung passiv. ($\frac{1}{2}$P) **3P**

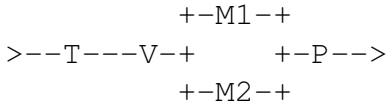
2. Fremdgenutzte Redundanz ($\frac{1}{2}$P), Ersatzkomponenten führen fremde (d.h. nicht zum betreffenden Subsystem gehörende) Funktionen, die im Fehlerfall verdrängt/ersetzt werden. ($\frac{1}{2}$P)

3. Gegenseitige Redundanz ($\frac{1}{2}$P), Ersatzkomponenten stehen sich gegenseitig als Reserve zur Verfügung. ($\frac{1}{2}$P)

d) **0,5P**

$$\varphi_m^n = \sum_{k=n}^{m} \binom{m}{k} * \varphi(K)^k * (1 - \varphi(K))^{(m-k)}$$

e) **1P**



Hinweis: E_1 bis E_3 haben keinen Einfluss auf die Zuverlässigkeit und sind daher auch nicht Bestandteil des Zuverlässigkeitsblockdiagramms.

f) Systemfunktion: $S = T \wedge V \wedge (M_1 \vee M_2) \wedge P$ **1P**

g) M redundant und daher nur bei Totalausfall funktionsuntüchtig, also gilt: **1P**
$\Phi(M_{1\vee 2}) = 1 - (1 - \Phi(M))^2$ *0,5P*

Somit ergibt sich für das Gesamtsystem:
$\Phi(S) = \Phi(T) * \Phi(V) * \Phi(M_{1\vee 2}) * \Phi(P)$ *0,5P*