OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG

INF FAKULTÄT FÜR INFORMATIK

# Schriftliche Prüfung

| | |
|---|---|
| im Fach: | **Rechnersysteme** |
| Studiengang: | B (PF CSE / IF; WPF CV / WIF) |
| am: | 29. Juli 2009 |
| Bearbeitungszeit: | 120 Minuten |
| zugelassene Hilfsmittel: | keine |

Vorname: _____

Nachname: _____

Matrikelnummer: _____

Studiengang: _____

Anzahl verwendete Zusatzblätter: _____

Bitte beschriften Sie jedes weitere Blatt in der rechten oberen Ecke wenigstens mit Ihrer Matrikelnummer!

Verwenden Sie den freien Platz auf den Aufgabenseiten für Ihre Lösung.

Viel Erfolg!

Punktetabelle (bitte leer lassen)

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | |

# Aufgabe 1

Sie wollen eine 32-Bit-Zahl über ein Netzwerk verschicken. Sie lautet:

1000 0001 1010 0000 0111 0111 0001 1101

Die Zahl liegt dabei im Little-Endian-Format vor. Ihr Netzwerk verwendet aber die Big-Endian-Formatierung. Ändern Sie das gegebene Wort in die netzwerkkonforme Darstellung!

**Lösung:** 0001 1101 0111 0111 1010 0000 1000 0001

(Anmerkung: Die Bytereihenfolge wird umgekehrt: Das vormals letzte Byte wird das neue erste Byte und so weiter.)

# Aufgabe 2

1. Analysieren Sie den nachstehenden Programm-Ausschnitt! Welche Funktion wird dadurch erfüllt? Kommentieren Sie das Programm zeilenweise!

```
FELDANF  equ      10000          ;
FELDEND  equ      FELDANF+100    ;
         move.l   #FELDANF,A1    ;
         move.l   #1,D1          ;
LOOP     move.b   D1,(A1)+       ;
         cmp.l    FELDEND,A1     ;
         ble      LOOP           ;
```

| Funktion:**Lösung:** Im Programmausschnitt werden die Elemente eines Feldes byteweise mit dem Wert 1 vorbelegt. |
|---|

2. Ändern Sie das Programm so, dass die Zahlen 1 bis 100 geschrieben werden! Markieren Sie dazu im vorgegebenen Quelltext die Stelle, an der Ihre Änderung eingefügt werden soll, mit einem Pfeil!

| Eigener Quelltext:**Lösung:** |
|---|

**Lösung:**

```
FELDANF  equ      10000          ; Startadresse des Feldes
FELDEND  equ      FELDANF+100    ; Endadresse des Feldes
         move.l   #FELDANF,A1    ; Startadresse ins Register A1
         move.l   #1,D1          ; D1 mit 1 initialisieren
LOOP     move.b   D1,(A1)+       ; 1 in das akt. Feldelement
         cmp.l    FELDEND,A1     ; Vergleich auf Feldende
         ble      LOOP           ; Verzw. zu LOOP, falls Feldende nicht erreicht
```

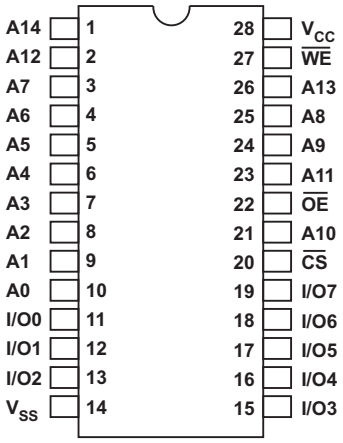
Für die Zahlen 1 bis 100 in den Zellen wäre folgende Änderung notwendig:

```
FELDANF  equ      10000          ; Startadresse des Feldes
FELDEND  equ      FELDANF+100    ; Endadresse des Feldes
         move.l   #FELDANF,A1    ; Startadresse ins Register A1
         move.l   #1,D1          ; D1 mit 1 initialisieren
LOOP     move.b   D1,(A1)+       ; 1 in das akt. Feldelement
         add.b    #1,D1          ; Zu schreibender Wert um 1 erhöhen
         cmp.l    FELDEND,A1     ; Vergleich auf Feldende
         ble      LOOP           ; Verzw. zu LOOP, falls Feldende nicht erreicht
```

*Gewichtung: 1) 60% und 2) 40%*

# Aufgabe 3

Es sollen 256 Kilobyte Speicher an eine CPU angebunden werden. Dazu stehen die in der Abbildung dargestellten byteorganisierten Speicherbausteine zur Verfügung.




1. Wie viele dieser Speicherbausteine müssen an den Prozessor angebunden werden, um die geforderten 256 Kilobyte zu erreichen?

   **Lösung:** Der dargestellte Chip hat 15 Adressleitungspins -> $2^{15}$Adressen -> 32KB Kapazität -> es werden 8 solcher Bausteine benötigt, um auf 256 KB zu kommen.

2. Nennen und erläutern Sie Vor- bzw. Nachteile von bitorganisierten gegenüber byteorganisierten Speicherbausteinen!

   **Lösung:**
   Vor- und Nachteile

   - Nachteil
     - Bei einem byteorganisierten Baustein werden die drei oberen Adressbits genutzt, um jeweils einen davon auszuwählen. -> es ist immer nur einer der 8 Chips aktiv und verbraucht Strom. Bei der bitorganisierten Variante sind immer alle 8 Chips aktiv. Letztere verbrauchen somit mehr Strom
   - Vorteile
     - Für den bitorganisierten Speicher ist kein zusätzlicher Adressdekoder notwendig, um jeweils nur einen der Bausteine zu aktivieren, da alle 8 Chips direkt mit dem Adressbus verbunden werden können. Es tritt keine zusätzliche Schaltverzögerung dieses Decoders auf, die den Speicherzugriff verlangsamt
     - In der bitorganisierten Variante sind die verwendeten Bausteine kleiner (weniger Pins je IC), wodurch der Platzbedarf sinkt

*Gewichtung: 1) 40%, 2) je 20 % pro Vor- und Nachteil*

# Aufgabe 4

Nennen und erläutern Sie mindestens drei Vorteile, die sich aus der Reduktion des Befehlssatzes einer CPU auf häufig verwendete Befehle ergeben.

**Lösung:** Bei kleinerem Befehlssatz müssen auch weniger Befehle in der CPU implementiert werden -> es werden weniger Transistoren und Chipfläche benötigt -> Es bleibt Platz für zusätzlichen Speicher.

**Lösung:** Bei weniger Befehlen muss weniger Platz für den OpCode reserviert werden -> kleiner OpCode -> weniger komplexe Befehlsdekodierung -> Einsparung von Transistoren und Chipfläche ... siehe letzter Vorteil
und: im Befehlswort bleibt mehr Platz für Daten oder Adressen -> Es können größere bzw. mehr Konstanten oder Adressen darin untergebracht werden -> Es muss seltener auf den Speicher zugegriffen werden

**Lösung:** Durch das Wegfallen selten genutzter Befehle verringert sich die notwendige Entwicklungszeit für den Prozessor da dieser weitaus weniger komplex ist

**Lösung:** Funktion selten genutzter komplexer Befehle kann mit mehreren einfacheren umgesetzt werden falls dies nötig sein sollte -> keine Einschränkung im Funktionsumfang

*Gewichtung: 33% pro Argument, max. 100%*

# Aufgabe 5

Erläutern Sie den Zusammenhang zwischen dem Einsatz einer mikroprogrammierten Steuereinheit und dem speichertechnologischen Entwicklungsfortschritt.

**Lösung:**

Die immer schneller werdenden Zugriffszeiten für Hauptspeicher bzw. Caches (darin liegen die Befehle) haben dazu geführt, dass der zusätzliche Speicherzugriff auf einen Mikroprogrammspeicher nicht mehr durch viel schnellere Zugriffszeiten des letzteren (ns gegen µs Zugriffszeiten betreffend) aufgewogen wird.

# Aufgabe 6

Erläutern Sie zwei wichtige in der Vorlesung behandelte Konzepte im Rahmen einer RISC-Architektur und benennen Sie das jeweilige Ziel.

**Lösung:** Konzept der überlappenden Registerfenster und wie es aussieht. Ziel: Verringerung von Kopier-Operationen.

**Lösung:** Pipelining und wie es im Prinzip funktioniert. Ziel: Reduzierung der Zykluszeit (Zeit, in der ein Befehl bearbeitet wird) auf die Ausführungszeit für ein Pipeline-Stadium.

**Lösung:**

Dritte Möglichkeit:

Einheitliches Befehlsformat. D.h. minimale Dekodierlogik und damit schnellere Dekodierung.

*Gewichtung: Nennung des Konzepts (zwei max.): je 10%. zusätzlich: bei Registerfenstern: Erwähnung Parameterübergabe 5%, Vermeidung von Speicherzugriffen (durch mehr Register auf CPU) 5%, Nennung der Registerklassen 10%, Darstellung der Überlappung (zur Parameterübergabe) 10%. Zielnennung 10. bei Pipelining: Beschreibung zur Parallelisierten Abarbeitung von Befehlen 15%, Ziel (auch Beschleunigung akzeptiert) 15%, bei Befehlsformat: minimale Dekodierlogik 20%, schnellere Dekodierung 20%*

# Aufgabe 7

Ein sequentieller Prozessor mit einer Zykluszeit von 25 ns wird in einen Fünf-Pipeline-Stadien-Prozessor umgewandelt, bei dem die einzelnen Stadien Latenzzeiten von 5, 7, 3, 6 bzw. 4 ns haben. Welche Zykluszeit hat dieser Prozessor unter der Annahme, dass die Latenz der Pipeline-Schalter 1 ns beträgt?

**Lösung:** $7 + 1 = 8$ [ns]

# Aufgabe 8

1. Was versteht man unter absoluter Cache-Performanz?

   **Lösung:** $1/\#$ CPU-Aussetzerzyklen $=$ absolute Cache-Performanz

2. Erläutern Sie kurz, wie bei gleichbleibender absoluter Cache-Performanz trotzdem die relative Cache-Performanz sinken kann!

   **Lösung:** CPU-Leistung erhöhen durch CPI verringern bzw. Taktrate erhöhen -> relative Cache-Performanz (= relativer Anteil der # CPU-Aussetzerzyklen von Programmausführungszeit insgesamt) steigt

*Gewichtung: je 50% auf 1) und 2)*

# Aufgabe 9

1. Welches sind die zu optimierenden Hauptparameter zur Erzielung einer hohen Cache-Performanz?

2. Nennen Sie jeweils eine Vorgehensweise dafür!

**Lösung:**

1. a) Minimierung von Fehlerraten bzw. b) Fehlerzeiten

2. a) Ausnutzen der Assoziativität bzw. b) durch Multilevel-Cache.
   Weiteres, aber weniger wichtig:
   a) Größere Blöcke/Seiten, b) Early restart/Critical word first bzw. ver*** Speicherorganisation

*Gewichtung: 1.) 50%, 2.): 25% Assoziativitätsausnutzung bzw. Block-/Seiten-Vergrößerung, 25% auf Multilevel-Cache bzw. Early-restart/CWF*

# Aufgabe 10

Nennen Sie die Strategien für das Platzieren bzw. Wiederfinden von Blöcken bzw. Seiten, die Sie in der Vorlesung kennengelernt haben! Erläutern Sie stichpunktartig deren Vor- und Nachteile in Form von in der Vorlesung verwendeten Maßen!

**Lösung:**

- Direkt abgebildet (direct mapped): Gute Trefferzeit, schlechte Trefferrate
- Mengenassoziativ (set associative): Mittelding
- Vollassoziativ (fully associative): schlechte Trefferzeit, gute Trefferrate

*Gewichtung: Nennung je 10%, Vor-/Nachteile 75% (25% direct mapped, 25% vollasso., 20% mengenasso.)*

# Aufgabe 11

Erläutern Sie kurz, aus welchem Grund das Konzept des TLBs eingeführt wurde!

**Lösung:**

Seitentabellen, die bei der Nutzung eines virtuellen Speichers angelegt werden, liegen im Hauptspeicher. Bei jedem Speicherzugriff wird erst auf die Seitentabelle zugegriffen, um die eigentliche Adresse eines Datums zu ermitteln. Erst dann wird in einem zweiten Speicherzugriff auf das gewünschte Datum zugegriffen.

Diese Speicherzugriffsverdopplung kann durch einen TLB in weiten Teilen auf nur einen Speicherzugriff reduziert werden: Der TLB ist ein Cache, der bereits angefragte Informationen aus der Seitentabelle zwischenspeichert. Also: Vermeidung eines zweiten Speicherzugriffs pro Prozessorreferenz durch Speichern einer einmal ermittelten physischen Seitenadresse in einem Cache.

*Gewichtung: 25% Nennung Grundkonzept (Verwaltung virtueller Speicher), 25% auf Umsetzungsbeschreibung (erst Zugriff Tabelle, dann Datum), 25% auf Beschreibung, was TLB ist (Cache für zugegr. Seitenadr.), 25% auf Vermeidung 2. Speicherzugriff*

## Aufgabe 12

Nennen Sie eine Hauptmaßnahme im Entwurf von Speicherhierarchien, die auf der alleinigen Annahme von räumlicher Lokalität beruht!

**Lösung:**

Transfereinheiten (zwischen den Speicherebenen) $\gg$ Prozessorreferenzeinheiten

# Aufgabe 13

Ein Einzelplatzrechner berechnet eine Simulationsaufgabe in 40 Stunden. Um welchen Faktor lässt sich diese Berechnung beschleunigen, wenn sechzehn Rechner parallel eingesetzt werden und 40 Prozent des Programmcodes parallel ausführbar sind? Wie lange dauert die parallel ausgeführte Berechnung?

Beschleunigungsfaktor (speedup)

**Lösung:** 1,6

Dauer der parallelen Berechnung

**Lösung:** 25 Stunden

**Lösung:**

Rechnung:

Vernachlässigt man die Koordination/Kommunikation zwischen den Prozessoren und berücksichtigt nur die Aufteilung eines Programms in parallelisierbare und nicht-parallelisierbare Teile, so ergibt sich die Grenze der Beschleunigung folgendermaßen: Beschleunigung$= \frac{n}{1+(n-1)\cdot f}$

Wobei $n$ die Anzahl der parallel eingesetzten Rechner und $f$ der Anteil des sequentiellen Codes an der Gesamtausführungszeit.

Die sequentiell ermittelte Gesamtausführungszeit geteilt durch die Beschleunigung ergibt die optimal mögliche (parallele) Ausführungsdauer.

$n = 16$; $f = 1 - 40\% = 0,60$

Die Beschleunigung beträgt folglich maximal $\frac{16}{1+15\cdot 0,6} = \frac{16}{1+9} = 1,6$. Die neue Ausführungsdauer beträgt $\frac{40h}{1,6} = 25h$

Diese Aufgabe kann auch „zu Fuß“ berechnet werden: 0,40 des Codes sind parallel berechenbar, d.h. 0,60 der Zeit sind nicht verkürzbar (24 Stunden). Die übrigen 16 Stunden verteilt man auf die 16 Rechner und erhält so die parallele Zeit ($\frac{16}{16} = 1$), 1 Stunde. Beide Zeiten zusammen ergeben die durch Parallelberechnung beschleunigte Zeit von $24 + 1 = 25$ Stunden.

Alte Zeit durch Neue Zeit ergibt die Beschleunigung ($\frac{40}{25} = 1,6$).

*Gewichtung: Beschleunigung: 70%, Stunden 30 %*

# Aufgabe 14

Betrachten Sie die Konzepte der Parallelisierung und des Pipelining.

1. Welchem gemeinsamen Ziel dienen sie?

   **Lösung:** Performanz-Steigerung des Rechners

2. Was ist der wesentliche Unterschied in der Vorgehensweise?

   **Lösung:** Pipelining: feinstgranular parallele Befehlsabarbeitung bei im Grunde gleichbleibendem Ressourceneinsatz.
   Parallelisierung: Aufteilung eines Programms auf möglichst viele Rechner (-einheiten) (grobgranulare P.) bei stark erhöhtem Ressourceneinsatz.

*Gewichtung: 1) 20%, 2) 80%*