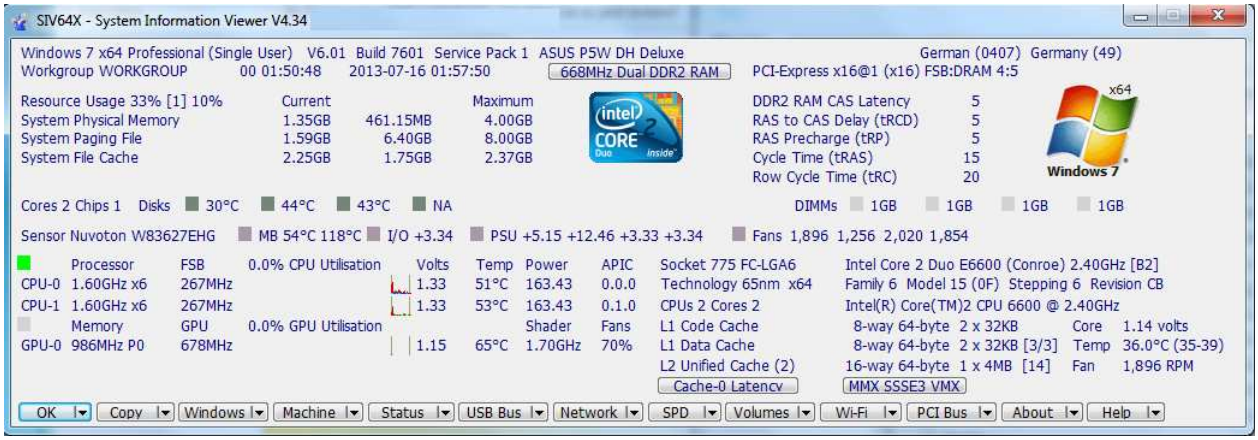
Kurs 20046 Einführung in die technische und theoretische Informatik
Lösungsvorschläge zur Übungsklausur

Aufgabe 1-1



a) Unter welchem Betriebssystem wird der PC betrieben?

> *Musterlösung:*
>
> Windows 7 x64 Professional

b) Wieviele RAM-Bausteine sind verbaut und wie groß ist der realisierte Hauptspeicher?

> *Musterlösung:*
>
> 4 x 1GB = 4GB Hauptspeicher

c) Welche charakteristischen Zeitparameter weisen die verwendeten RAM-Bausteine auf?

Musterlösung:

CAS-Latenz t_{CL}: 5 Taktzyklen
RAS to CAS Delay t_{RCD}: 5 Taktzyklen
RAS-Precharge t_{RP}: 5 Taktzyklen
Zykluszeit t_{RAS}: 15 Taktzyklen

d) Wie hoch ist die aktuelle Auslastung des Arbeitsspeichers?

Musterlösung:

Die aktuelle Auslastung liegt bei 1.35GB.

e) Welcher Prozessor ist im PC eingesetzt? Von welchem Hersteller stammt er?

Musterlösung:

Intel Core 2 Duo E6600 (Conroe) 2.40GHz [B2]

f) Wie groß sind die Caches des Processors und welche Blockgröße besitzen sie?

Musterlösung:

L1 Code-Cache: 32KB
L1 Daten-Cache: 32KB
L2 Code/Daten-Cache: 4 MB

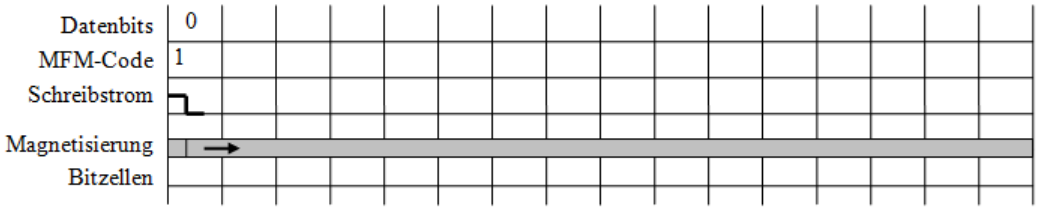
g) Wieviele Lüfter sind angeschlossen und wie schnell drehen sie sich (Angabe in rpm „rounds per minute")?

Musterlösung:

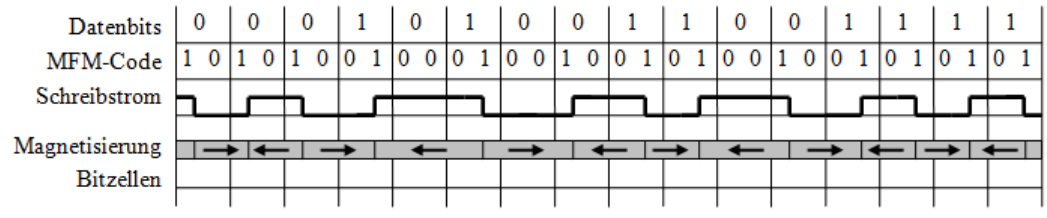
Es sind 4 Lüfter angeschlossen die sich mit 1896, 1256, 2020, 1854 rpm drehen.

Aufgabe 1-2

Die Datenbit-Folge D = 0001 0100 1100 1111 soll auf einer Festplatte mit dem MFM- Verfahren aufgezeichnet werden. Tragen Sie dazu in das folgende Bild die aufgezeichneten Datenbits, die resultierende MFM-Codierung, den Verlauf des Schreibstroms sowie die Magnetisierungsrichtung der Festplatte ein. Als Rastergröße in waagerechter Richtung ist darin die Länge einer Bitzelle vorgegeben, d.h. der kleinste Abschnitt gleichgerichteter Magnetisierung. Ordnen Sie die Daten- und Code-Bits jeweils über den Bitzellen an, in denen sie abgespeichert werden.

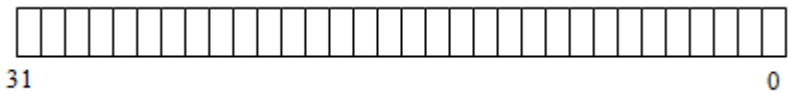


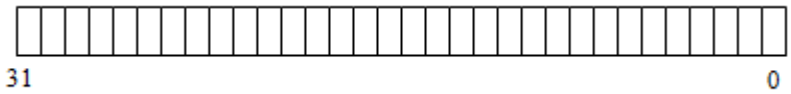
Musterlösung:



Aufgabe 2-1

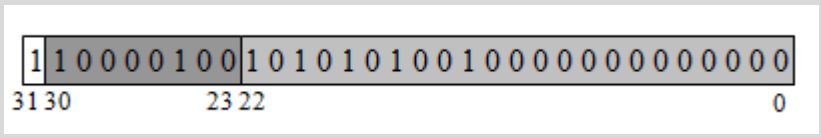
Stellen Sie die Dezimalzahlen $Z_1 = -53.28125$ und $Z_2 = 77184$ im 32-bit-Format des IEEE-754-Standards in binärer Form dar. Kennzeichnen Sie die unterscheidbaren Bitfelder (Vorzeichen, baised Exponent, Mantisse).



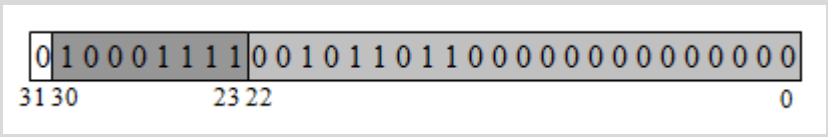


Musterlösung:

$Z_1 = (-1)^1 \cdot (2^5 + 2^4 + 2^2 + 2^0 + 2^{(-2)} + 2^{(-5)}) = -53.28125 = (-1)^1 \cdot 2^{132-127} \cdot (1.10101010010\cdots0)_2$
Daraus ergibt sich:



$Z_2 = 2^{16}+2^{13}+2^{11}+2^{10}+2^8+2^7 = 77184 = (-1)^0 \cdot 2^{143-127} \cdot (1.0010110110\cdots0)_2$
Daraus ergibt sich:



Aufgabe 2-2

Wir betrachten die folgende Adress-Befehlsfolge:

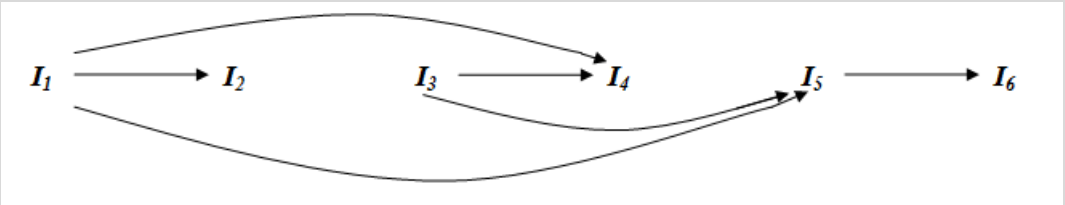
	Befehl	Bedeutung
I_1	ADD R1,R2,R3	R1=R2+R3
I_2	MUL R2,R1,#5	R2=R1*5
I_3	ADD R5,R3,R1	R5=R3+R1
I_4	ADD R3,R4,R2	R3=R4+R2
I_5	MUL R3,R5,#2	R3=R5*2
I_6	MUL R5,R3,#3	R5=R3*3

a) Welche Gegenabhängigkeiten bestehen in dieser Befehlsfolge? Stellen Sie ausschließlich die Gegenabhängigkeiten in einem Präzedenzgraphen dar. Hinweis: Verwenden Sie dabei die nachfolgende Darstellungshilfe.

I_1 I_2 I_3 I_4 I_5 I_6

Musterlösung:

Der Präzedenzgraph sieht wie folgt aus:



b) Welche zwei weiteren Abhängigkeiten können bei einer Befehlsfolge auftreten? Nennen Sie auch den Konflikt der dabei jeweils entstehen kann.

Musterlösung:

Neben den Gegenabhängigkeiten können auch echte Datenabhängigkeiten oder Ausgabeabhängigkeiten auftreten. Bei den echten Datenabhängikeiten können Lese-nach-Schreib-Konflikte (Read After Write - RAW) auftreten. Bei den Ausgabeabhängigkeiten können Schreib-nach-Schreib-Konflikte (Write After Write - WAW) auftreten.

c) Welche Konflikte außer den hier betrachteten Datenkonflikten können in einer Pipeline zusätzlich auftreten. Beschreiben Sie diese auch kurz.

Musterlösung:

Es können z.B. Struktur- oder Ressourcenkonflikte auftreten, bei denen zwei Pipeline-Stufen dieselbe Ressource benötigen, auf die aber nur einmal zugegriffen werden kann (vgl. Speicher-Lesekonflikt). Eine andere Art von Konflikten sind die sogenannten Steuerflusskonflikte. Diese treten bei Programmsteuerbefehlen auf, wenn in der Befehlsbereitstellungsphase die Zieladresse des als nächstes auszuführenden Berfehls noch nicht berechnet ist bzw. im Falle eines bedingten Sprunges noch nicht klar ist, ob überhaupt gesprungen wird.

d) Nennen und beschreiben Sie jeweils eine Software- und eine Hardware-Lösung für Datenkonflikte in einer Pipeline.

Musterlösung:

Eine Software-Lösung von Datenkonflikten ist beispielsweise das Einstreuen von sogenannten NOP-Befehlen (no operation) zwischen datenabhängigen Befehlen durch den Compiler. Die Anzahl der notwendigen NOPs hängt dabei von der Anordnung der Befehle ab. Der Compiler kann diese Anordnung ändern, um möglichst wenige NOPs verwenden zu müssen. Bei dieser Methode spricht man dann auch von der sogenannten Befehlsanordnung.
Hardwareseitig kann man beispielsweise durch die Pipeline-Sperrung, auch interlocking genannt, solche Konflikte behandeln. Dabei wird der datenabhängige Folgebefehl in der Pipeline für drei bzw. zwei Takte angehalten. Weitere Lösungen stellen beispielsweise das Forwarding oder das Forwarding mit Interlocking dar.

Aufgabe 3-1

Gegeben sei ein Cache C1 mit direkter Zuordnung. Der Cache umfasst 256 Blöcke von je 16 Bytes. Der Speicher ist byte-weise adressierbar, Adressen sind 20 Bit lang (Bit 19 bis Bit 0).

a) Geben Sie an, wieviele Bits und welche Bits der Adresse zu Tag, Index, Wortadresse gehören.

b) Gegeben sei eine Folge von Adressen (5-stellig hexadezimal), auf die nacheinander lesend zugegriffen wird:

12345
1234A
F7069
BC34D
F708A
F706F
1234A
A7069
F7082
12345

Bestimmen Sie die Anzahl der Hits von C1, wenn C1 zu Beginn leer ist.

c) Bestimmen Sie die mittlere Zugriffszeit t_{eff1} bei einer Hit-Rate von 50% in C1. Der Cache C1 habe eine Zugriffszeit von 10 ns, der Speicher eine Zugriffszeit von 50 ns, bei einem Miss soll der Zugriff sequentiell erfolgen.

d) Zwischen Prozessor und Cache C1 wird nun noch ein weiterer Cache C0 geschaltet, der eine Zugriffszeit von 1 ns hat. Bei einem Miss in C0 soll der Zugriff zwischen C0 und dem Rest des Speichersystems parallel erfolgen. Bestimmen Sie die mittlere Zugriffszeit t_{eff0} des Gesamtsystems (C0, C1, Speicher) bei einer Hit-Rate von 90% in C0, die restlichen Werte sollen so bleiben wie in Teilaufgabe c).

Musterlösung:

a) Zur Wortadresse gehören 4 Bits (Bits 3 bis 0).
Zum Index gehören 8 Bits (Bits 11 bis 4).
Zum Tag gehören 8 Bits (Bits 19 bis 12).

b) Der zweite, sechste, neunte und zehnte Zugriffs sind Hits, also 4.

c) Es gilt $t_{eff1} = 0,5 \cdot 10 + 0,5 \cdot (10 + 50) = 35$ ns.

d) Es gilt $t_{eff0} = 0,9 \cdot 1 + 0,1 \cdot t_{eff1} = 4,4$ ns.

Aufgabe 3-2

In einem physikalischen Speicher der Größe 10^6 (Adressen 0 bis 999.999) gibt es drei Segmente (ohne Seitenverwaltung), die sich an den Adressbereichen (dezimal)
20.000 bis 99.999
200.000 bis 789.999
800.000 bis 989.999
befinden, so dass sich im Adressraum vier Lücken befinden.

a) Es soll ein weiteres Segment der Größe 5.000 (dezimal) platziert werden. Geben Sie an, in welche Lücke das Segment bei Benutzung von first fit, best fit, und worst fit jeweils platziert wird.

b) Nehmen Sie an, dass nach der Platzierung des vierten Segments überlegt wird, eine Seitenverwaltung einzuführen. Zur Auswahl stehen die Seitengrößen 10.000 und 20.000 (jeweils dezimal). Wie hoch wäre jeweils die gesamte Fragmentierung bei Nutzung der Seitenverwaltung mit den beiden Seitengrößen?

Musterlösung:

a) First fit: Das Segment kommt in die erste (hinreichend große) Lücke, d.h. an die Stellen 0 bis 999.
Best fit: Das Segment kommt in die kleinste (aber hinreichend große) Lücke, also in die letzte von 990.000 bis 994.999.
Worst fit: Das Segment kommt in die größte Lücke, also von 100.000 bis 104.999.

b) Die vier Segmente haben die Größen (aufsteigend geordnet):
5.000
80.000
190.000
590.000
Bei einer Seitengröße von 10.000 entsteht nur im ersten Segment eine Fragmentierung von 5.000, also eine halbe Seite. Bei einer Seitengröße entsteht im kleinsten Segment eine Fragmentierung von 15.000, sowie in den beiden größten Segment eine Fragmentierung von jeweils 10.000, also insgesamt 35.000, also fast zwei Seiten.

Aufgabe 4-1

a) Geben Sie die drei Eliminationsregeln an, um aus einem endlichen Automaten den zugehörigen regulären Ausdruck zu erzeugen. Erläutern Sie eine der Eliminationsregeln.

b) Was ist die Länge des leeren Worts ϵ?

c) Bei einem endlichen Automaten ist der Startzustand gleichzeitig auch ein Endzustand. Geben Sie das kürzeste akzeptierte Eingabewort an.

d) Welches der Wörter ccc, abb und $aaaa$ gehört zur Sprache des regulären Ausdrucks $(a^* + b^*)c^*$ über dem Alphabet $E = \{a, b, c\}$? Gehört das leere Wort ϵ zu dieser Sprache?

Musterlösung:

a) Kantenelimination, Schleifenelimination, Knotenelimination
b) Die Länge ist 0.
c) Das kürzeste akzeptierte Wort ist das leere Wort.
d) Das erste und dritte Wort gehören zur Sprache des regulären Ausdrucks.

Aufgabe 4-2

Gegeben sei ein endlicher Automat ohne Ausgabe $EA = (S, E, \delta, s_0, F)$ mit der Zustandsmenge $S = \{s_0, \ldots, s_2\}$, dem Eingabealphabet $E = \{a, b\}$, der Übergangsfunktion $\delta : S \times E \to S$ mit der Wertetabelle

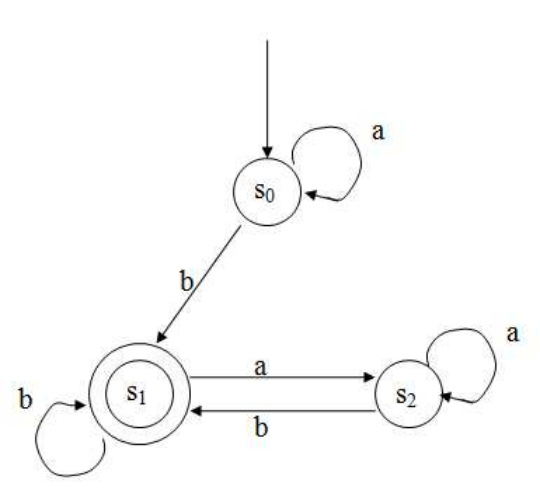
s	e	$\delta(s, e)$
s_0	a	s_0
s_0	b	s_1
s_1	a	s_2
s_1	b	s_1
s_2	a	s_2
s_2	b	s_1

a) Zeichnen Sie den Zustandsgraphen von EA und ermitteln Sie die Sprache von EA, wenn die Menge der Endzustände $F = \{s_1\}$ ist.

a) Ermitteln Sie die Sprache von EA, wenn die Menge der Endzustände $F = \{s_1, s_2\}$ ist.

Musterlösung:

a) Der Zustandsgraph ergibt sich zu



Der Automat akzeptiert ein Wort, wenn das letzte Zeichen ein b ist, da man nur mit b nach s_1 kommt. Also ist die Sprache des Automaten $E^*.\{b\}$.
b) Ist auch s_2 ein Endzustand, dann akzeptiert der Automat alle Worte, außer wenn er in s_0 endet. In s_0 zu enden ist nur möglich, wenn ein Eingabewort der Form a^* vorliegt. Also ist die Sprache des Automaten $E^* \setminus \{a\}^*$.

Aufgabe 4-3

Welche der folgenden Aussagen treffen zu:

a) Nach der Church-Turing These sind alle Probleme, die überhaupt berechenbar sind, die Probleme, die mit Modellen wie zum Beispiel universellen Turing-Maschinen oder PCs berechenbar sind.

b) Die Entscheidung, ob die Ausgabe eines beliebigen, vorgegebenen Programms mit HALLO beginnt oder nicht, ist mit einem PC berechenbar.

c) Das Problem des Handlungsreisenden mit Schranke liegt in NP, d.h. es ist ein Problem, von dem man nicht glaubt, dass es einen effizienten Algorithmus (in Polynomialzeit der Problemgröße) zu seiner Lösung gibt.

d) Algorithmen, bei denen man beweisen kann, dass sie nie mehr als ein gewisses Quantum von der optimalen Lösung abweichen, nennt man Approximationsalgorithmen.

e) Der First-Fit-Decreasing-Algorithmus für das Bin-Packing-Problem ist ein Approximationsalgorithmus mit relativer Güte 1,5.

f) Eine Approximation für das Problem des Handlungsreisenden ohne Schranke, die relativer Güte 1,5 hat, berechnet eine Rundreise, die stets mindestens 1,5 malso lang wie eine optimale Rundreise ist.

Musterlösung:

Die Aussagen a), c), d) treffen zu.

Aufgabe 4-4

Das Knotenfärbungsproblem besteht darin, bei Eingabe eines (gerichteten oder ungerichteten) Graphen mit n Knoten und $m \leq n^2$ Kanten sowie einer Zahl k anzugeben, ob es für diesen Graphen eine gültige Knotenfärbung mit k Farben gibt. Eine Knotenfärbung ist gültig, wenn zwei Knoten, die durch eine Kante verbunden sind (Richtung der Kante egal), mit verschiedenen Farben gefärbt sind.

Zeigen Sie, dass das Knotenfärbungsproblem in NP liegt, d.h. geben Sie textuell ein Verfahren an, das bei Eingabe eines Graphen (s. oben) sowie einer Knotenfärbung mit k Farben angibt, ob die Färbung gültig ist. Argumentieren Sie, dass das Verfahren eine Laufzeit hat, die polynomiell in n ist.

Musterlösung:

Für jede Kante (u, v) des Graphen bestimmen wir die Farben der beiden Endknoten u und v und testen, ob diese Farben unterschiedlich sind. Wenn die Farben gleich sind, ist die Ausgabe NEIN und das Verfahren bricht ab. Hat das Verfahren alle Kanten abgearbeitet, ist die Ausgabe JA. Da es höchstens n^2 Kanten gibt, ist die Laufzeit polynomiell in n.