Klausur

Einführung in die Informatik 3, WS04

Dozent: Döben-Henisch

7.Februar 2005, 10:00 – 11:30h, Raum BCN 231

NAME	
MATRIKELNUMMER	
Prüfungsart: SL oder PL	

Summe Pkte		**Note**	

Minimale Punktezahl, um Informatik3 zu bestehen: 25 Pkte

---

1. Regularien

i. Teilnehmer an der Klausur müssen sich durch Lichtbildausweis identifizieren.
ii. Es dürfen nur leere Blätter, Schreibgeräte und Taschenrechner benutzt werden. Alle in Gebrauch befindlichen Blätter müssen mit Name und Matr.Nummer gekennzeichnet sein. Arbeitsblätter ohne Kennzeichnung gelten als Täuschungsversuch. Blätter ohne Kennzeichen werden bei der Auswertung nicht berücksichtigt.
iii. Die Klausurzeit beträgt 90 Min. Man darf seinen Platz erst verlassen, wenn der Prüfungsleiter die Klausurarbeit in Empfang genommen hat.
iv. Es gilt folgende Punktetabelle:

1. >69
2. 55-69
3. 40-54
4. 25-39
5. <25

v. Das aktive Abschreiben wie auch das Zulassen von Abschreiben wird als Täuschungsversuch bewertet.
vi. Jeder, der an der Klausur teilnimmt, erklärt zu Beginn, dass er sich durch keine Umtände beeinträchtigt empfindet, die seine Klausurleistung wesentlich beeinflussen könnten.

Fragen 1-11 (Max. 26 Pkte):

1. (Pkt. 2 Was versteht man unter der *ISA* (:= Instruction Set Architecture)?
2. (Pkt. 2) Welche Funktion hat das *Flag-Register*, auch Condition-Codes genannt?
3. (Pkt. 2) Welche Funktion hat der *Programmzähler* (program counter)?
4. (Pkt. 2) Was versteht man unter einem *Bootstrap*?
5. (Pkt. 3) Welche *Ereignisse* werden in einem PC nach Betätigung des Resetschalters ausgelöst?
6. (Pkt. 3) Geht man von den Prozessen als Grundelementen des Linux-Kernels aus: welche Möglichkeiten bestehen aus Sicht des Kernels, Prozesse zu *synchronisieren*?

7. (Pkt. 2) Auf welche Weise kann die Aussenwelt den Kernelmode von Linux *beeinflussen*?
8. (Pkt. 2) Was unterscheidet einen Prozess, der *reentrant* ist, von einem Prozess, der *nicht reentrant* ist?
9. (Pkt. 2) Was ist der Unterschied zwischen einem *präemptiven* und einem *nicht-präemptiven* Kernel?
10. (Pkt. 3) Worin genau besteht die Multimedia-Erweiterung der IA32-Architektur und wodurch kann dies eine Leistungssteigerung herbeiführen?
11. (Pkt. 3) Was ist Hyperthreading und wodurch kann dies eine Leistungssteigerung herbeiführen?

Aufg.1 (Max.Pkt: 18):

Ein Computer besteht in der Regel aus mindestens einem Bus-System, über das mehrere Komponenten miteinander verbunden sind. Beschreiben Sie, *welche Komponenten* man mindestens *benötigt*, um die nachfolgenden Aufgaben lösen zu können und auf *welche Weise* die Komponenten des Computers *nacheinander aktiviert* werden.

Aufgaben:
1. Sie schieben eine Diskette in das Diskettenlaufwerk und speichern die Datei *hello.c* auf der Festplatte.
2. Sie tippen auf der Konsole *hello* um das Programm zu starten.
3. Das Programm *hello* enthält den Befehl *printf("Hallo Welt !");*

Fertigen Sie ein Schaubild einer passenden Computerarchitektur an und kommentieren Sie die Abläufe.

Fragen zum Debugger (Max.Pkt: 14):

Im Rahmen der Vorlesung wurde der GNU-Debugger gdb benutzt, um Programmcode und die Arbeitsweise des Programms zu untersuchen:

1. (1 Pkte) Was sind Haltepunkte (breakpoints)?
2. (1,5 Pkte) Wie können Sie Haltepunkte in einem Programm setzen?
3. (1,5 Pkte) Wie können Sie sich alle Haltepunkte anzeigen lassen?
4. (2 Pkte) Wie können Sie Haltepunkte löschen ?
5. (2 Pkte) Wie können sie sich die Inhalte von Variablen anzeigen lassen?
6. (2 Pkte) Welche Möglichkeiten gibt es, sich den Stackinhalt anzeigen zu lassen?
7. (2 Pkte) Was ist ein *Frame*?
8. (2 Pkte) Wie können sie sich die verfügbaren Register anzeigen lassen?

Aufg.3 (Max.Pkt: 22):

A3.1 (9 Pkte):

Angenommen, der folgende Befehlskode wird als nächstes von der CPU gelesen:

Opcode im Speicher: *0000-0000-0000-0000-1000-1001-1110-0101*

Wie interpretieren Sie dieses Bitmuster unter Zuhilfenahme der Tabellen im Anhang, wenn Sie wissen, dass '*1000-1001'* den Befehl *move* kodiert?

Erklärung zu Move: "The MOV (move) ... instructions transfer data between memory and registers or between registers. The MOV instruction performs basic load data and store data operations between memory and the processor's registers and data movement operations between registers. It handles data transfers along the paths listed in the Table below".

A3.2 (9 Pkte):

Angenommen, der folgende Befehlskode wird als nächstes von der CPU gelesen:

Opcode im Speicher: *0000-0000-1000-0011-1110-0100-1111-0000*

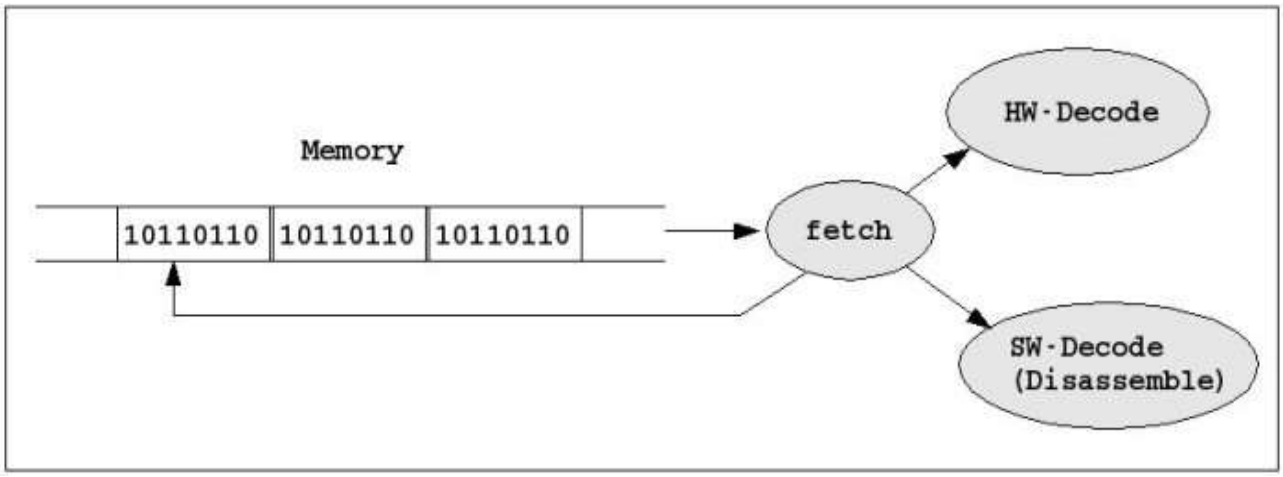
Wie interpretieren Sie dieses Bitmuster unter Zuhilfenahme der Tabellen im Anhang, wenn Sie wissen, dass '*1000-0011'* den Befehl *and* kodiert?

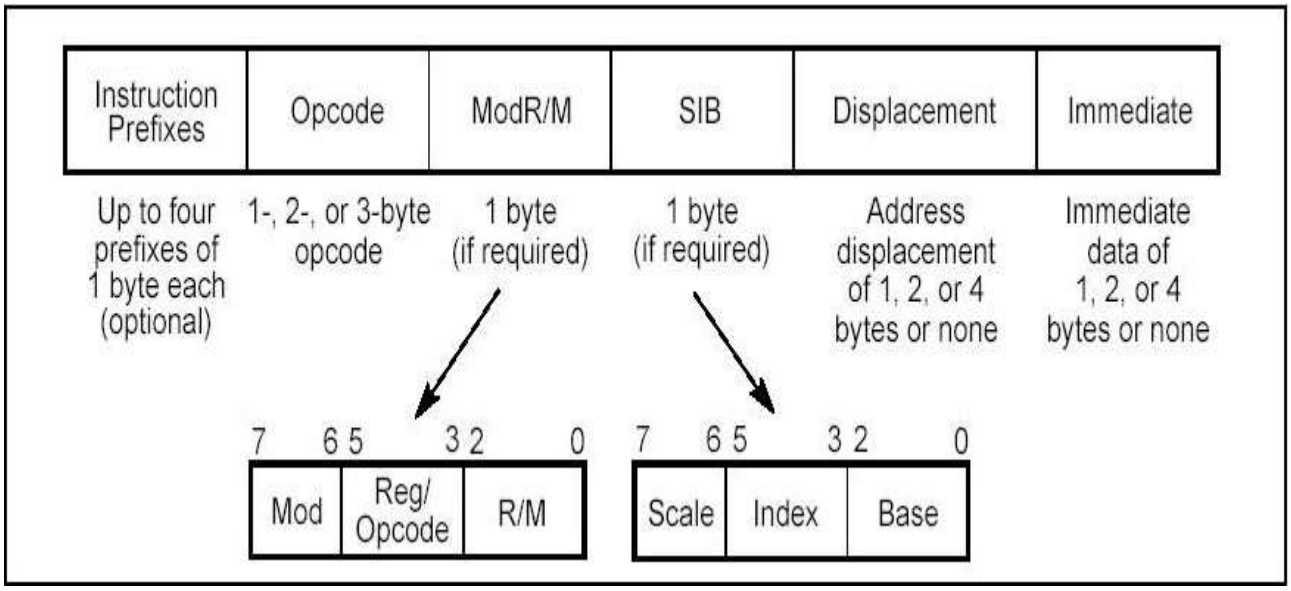
Erklärung zum AND-Befehl: "Performs a bitwise AND operation on the destination (first) and source (second) operands and stores the result in the destination operand location. The source operand can be an immediate, a register, or a memory location; the destination operand can be a register or a memory location. (However, two memory operands cannot be used in one instruction.) Each bit of the result is set to 1 if both corresponding bits of the first and second operands are 1"

A3.3 (4 Pkte):

Wie lautet ds Ergebnis der Operation, wenn der Inhalt des Registers lautet: *1000-1000*

ANHANG





Opcode						*D*	*W*

D = 0 := Daten fliessen von R/M zu REG

D = 1 := Daten fliessen von REG zu R/M

W = 0 := Datengrösse ist ein Byte

W = 1 := Datengrösse ist ein Wort oder ein Doppelwort

Sign-extension := Das Bit für das Vorzeichen (0 oder 1) wird bis zum 16-Bit Rahmen aufgefüllt, aus '00h' wird '0000h' bzw. aus '80h' wird 'ff80h'

MOD		*REG*			*R/M*		

MOD = 11 := *Register Addressing Mode*; dann spezifiziert das R/M-Feld ein Register und keinen Speicherbereich.

MOD = 00 := *Data Memory Addressing* with no displacement

MOD = 01 := *Data Memory Addressing* with an 8-Bit sign-extended displacement

MOD = 10 := *Data Memory Addressing* with an 16-Bit (32-Bit) displacement

Reg=Code	***W = 0 (Byte)***	***W = 1 (Word)***	***W = 1 (Doubleword)***
000	AL	AX	EAX
001	CL	CX	ECX
010	DL	DX	EDX
011	BL	BX	EBX
100	AH	SP	ESP
101	CH	BP	EBP
110	DH	SI	ESI
111	BH	DI	EDI