Department Informatik
Lehrstuhl für Informatik 3 (Rechnerarchitektur)
Friedrich-Alexander-Universität Erlangen-Nürnberg
Martensstraße 3, 91058 Erlangen

Klausur zu Grundlagen der Rechnerarchitektur und -organisation

27.09.2011

....
Matrikelnummer Geburtsdatum Vorname Name

- Es sind keine Hilfsmittel erlaubt
- Legen Sie den Personalausweis griffbereit auf den Platz!
- Dieses Aufgabenheft umfasst Seiten. Überprüfen Sie bitte die Vollständigkeit.
- Gesondert beigelegte Blätter werden nicht bewertet.
- Schreiben Sie bitte deutlich, Unleserliches wird nicht bewertet!
- Es darf nicht mit der Farbe rot geschrieben werden!
- Offensichtlich falsche oder überflüssige Antworten können zu Punktabzug führen.

Durch meine Unterschrift bestätige ich:

- den Empfang der vollständigen Klausurunterlagen
- die Kenntnisnahme der obigen Informationen

Erlangen, den 27.09.2011 .....
(Unterschrift)

Ich bin damit einverstanden, dass mein Prüfungsergebnis der Klausur unter Angabe der Matrikelnummer veröffentlicht wird.

Erlangen, den 27.09.2011 .....
(Unterschrift)

Aufgabe	1	2	3	4	5	6	7	8	Bonus	Summe
max. Punktzahl	24	18	19	18	7	3	7	4		
erreichte Punktzahl										

Aufgabe 1 – Thema Mikroprogrammierung

Die nachstehende Abbildung zeigt ein aus der Vorlesung hinreichend bekanntes Rechenwerk inkl. Seinen angeschlossenen Befehlsspeicher. Die Register *A*, *B*, *C*, *D*, *E* und *F* können jeweils pro Takt beschrieben werden und pro Takt kann das Leitwerk auch immer ein neues Mikroinstruktionswort ausgeben. Gehen Sie ferner davon aus, dass ein Register in einem Takt beschrieben werden kann und gleichzeitig seine an den Ausgängen anliegenden Daten weitergeleitetet werden können. Die Information an den Eingängen eines gerade beschriebenen Registers kann aber erst im nächsten Takt an den Ausgängen wieder abgegriffen werden. Das Register F enthält die Adresse des nächsten aus dem Speicher auszulesenden 17-Bit breiten Befehls.

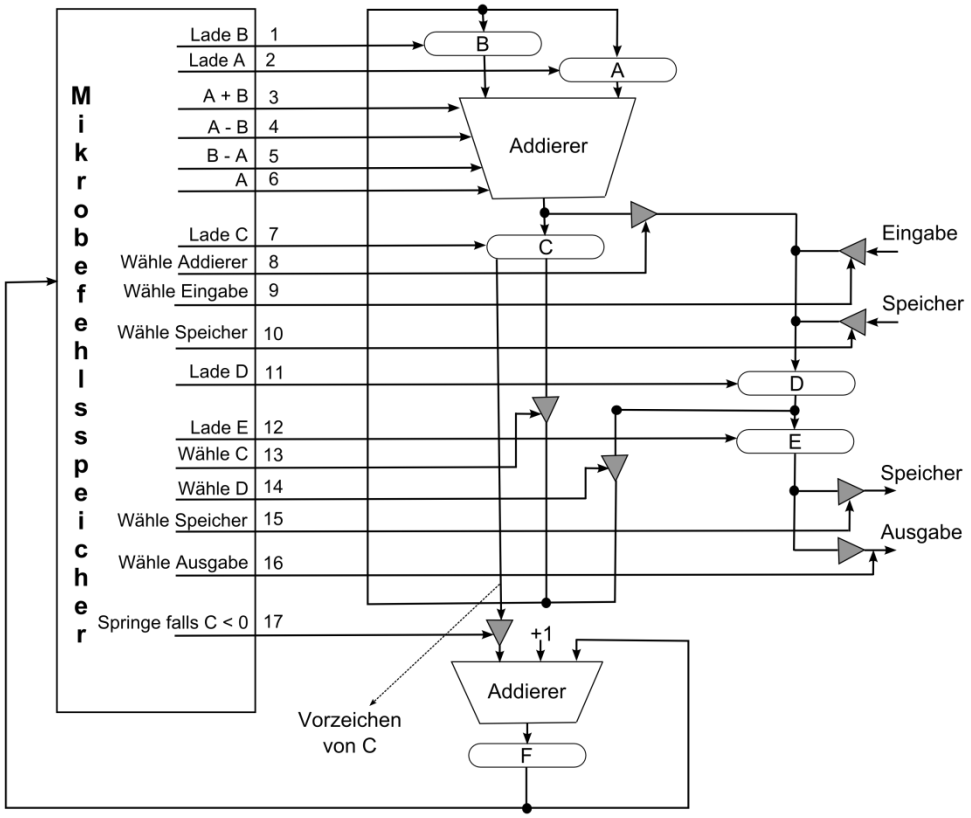


Abbildung 1: Mikroprogrammiertes Leitwerk

a) Vervollständigen Sie die nachstehende Tabelle, die dem Inhalt des Befehlsspeichers entspricht derart, dass eine Sequenz der 17 Steuersignale semantisch folgendes Programm ausführt. (Nebenbemerkung: die Anzahl der vorbereiteten Zeilen ist nicht notwendigerweise identisch mit der Anzahl an erforderlich Sequenzen) (12 Punkte)

```
if (Eingabe > Speicher) then (Eingabe – Speicher) -> Speicher
```

Adresse Befehl	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1																	
2																	
3																	
4																	
5																	
6																	
7																	
8																	
9																	
10																	
11																	
12																	
…																	

b) Beschreiben Sie in eigenen Worten wie die Mikroprogrammierung funktioniert und nennen Sie zwei Vorteile, die man daraus zieht! (Es kann auch mehr Vorteile geben, die Nennung von zwei möglichst naheliegenden genügt) (2 Punkte)

Antwort:

c) In der Vorlesung wurden zwei Varianten mikroprogrammierter Leitwerke behandelt. Um welche Variante handelt es sich jeweils bei der in Abbildung 1 gezeigten Architektur? (1 Punkt)

Antwort:

d) Wie bezeichnet man die zweite Variante? Nennen Sie jeweils einen Vorteil, den die eine Architektur gegenüber der anderen hat. (3 Punkte)

Antwort:

Die folgende Tabelle zeigt einen Mikroprorammspeicher, der ein Mikroprogramm ab einer bestimmten Adresse enthält. (Die letzte Spalte können Sie – Sie müssen nicht – dafür nutzen, den in der Bitsequenz gezeigten Befehl aufzuschreiben. Eventuell hilfreich für die Analyse)

Adresse Befehl	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Befehl in Mnemonic-Schreib-weise, z.B. B-A->D
1																		
2																		
3																		
4																		
5																		
6																		
7																		
8	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	
9	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
10	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	
11	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	end

e) Analysieren Sie die Sequenzen und schreiben Sie dieses als Makrobefehl in folgender Notation: (4 Punkte)

```
In1 op In2 -> Out; In1,In2 ∈ {A,B,Eingabe,Speicher}, Out ∈ {Ausgabe,Speicher}
```

Antwort:

f) Wie müssen diese Makrobefehle von einem Dekoder dekodiert werden, so dass sie zu dem in Tabelle 2 gezeigten Inhalten eines Mikroprogrammspeichers passen? (2 Punkte)

Antwort:

Aufgabe 2 – Mikroprozessor-Architekturen

a) In der Vorlesung wurden RISC- und CISC-Architekturen behandelt. Welche dieser beiden Prozessorarchitekturen besitzt ein mikroprogrammiertes und welche ein festverdrahtetes Leitwerk? Welche Variante kommt auf welche Weise in heutigen INTEL-Prozessoren zum Einsatz? (3 Punkte)

Antwort:

b) Wie funktioniert Pipelining auf der Befehlsebene und welcher Vorteil ergibt sich daraus? (3 Punkte)

Antwort:

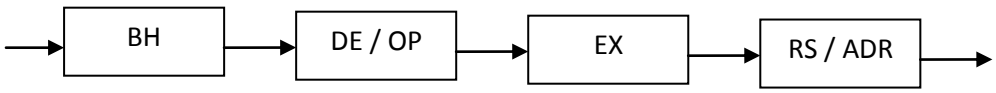
c) Berechnen Sie allgemein in Abhängigkeit einer bestimmten Anzahl *k* an Pipelinestufen und einer gegen unendlichen gehenden Anzahl *n* an Instruktionen den durch Pipelineverarbeitung erzielbaren Speed-up gegenüber einer Architektur, die rein seriell, also ohne Pipelineverarbeitung und nur mit einer Stufe arbeitet. (2 Punkte)

Antwort:

Ein hypothetisches Programm verarbeitet gemäß folgendem Ausschnitt aus einem C-Programm in einer for-Schleife iterativ aufeinanderfolgende Werte von in vier Feldern *A*, *B*, *C* und *D* gespeicherten Daten.

```
...
for (lv = 1; lv < 10; lv++) {
   A[lv] = A[lv-1] + A[lv] ;
   B[lv] = B[lv-1] - B[lv] ;
   C[lv] = C[lv-1] / C[lv] ;
   D[lv] = D[lv-1] * D[lv] ;
} ;
```

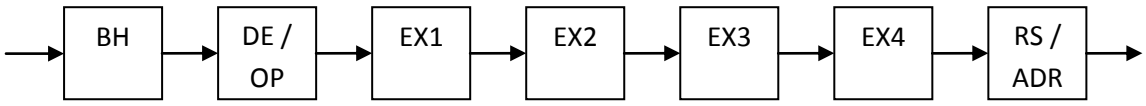
Das Rechenwerk besteht aus folgender streng linear aufgebauten Pipeline mit vier Stufen, die jeweils in einem Taktschritt durchlaufen werden. Die Ausführungsstufe soll ferner in der Lage sein, die Operationen `+`,`*`,`-` und `/` in einem Taktschritt zu bearbeiten. Auch die Zugriffe auf die Felder gehen in einem Takt vonstatten.



d) Wie viele Schritte werden benötigt, um den Schleifenkörper abzuarbeiten, wenn der Compiler die Schleife aufrollt, d.h. es werden keine Instruktionen für das Inkrementieren des Laufindex *lv* und für die Abfrage, ob die Schleife zu Ende ist, benötigt? (4 Punkte)

Antwort:

Nun soll die Pipeline des Rechenwerkes, wie nachstehend gezeigt, bei der Ausführungsphase *EX* erweitert werden.



e) Wie nennt man dieses Problem in der Fachsprache, das nun bei der Abarbeitung des Schleifenkörpers in dieser erweiterten Pipeline auftritt? (1 Punkt)

f) Wie viele Taktschritte werden in diesem Falle für die komplette Abarbeitung des Schleifenkörpers benötigt? (Evtl. ist es hilfreich den Datenfluss durch die Pipeline für die Iterationen in der untenstehenden Tabelle zu skizzieren. Für die Bepunktung der Aufgabe ist dies nicht notwendig.)

(5 Punkte)

t	BH	DE/OP	EX1	EX2	EX3	EX4	RS / ADR
0							
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							
…							

Antwort:

Aufgabe 3 – Cache-Architekturen

a) Beschreiben Sie wie die in der Vorlesung behandelten Cache-Architekturen arbeiten und welche Vor-/Nachteile diese aufweisen! (3 Punkte)

Antwort:

b) Ein Rechner besitze einen Hauptspeicher der Größe 1 GByte, der eingesetzte Cache weise eine Kapazität von 16 KByte auf, verwendet eine Blöckgröße von 16 Bytes und sei 2-fach Mengen-assoziativ organisiert. Zeigen Sie wie die Bits der Hauptspeicheradresse in diesem Falle aufgeteilt sind! (6 Punkte)

Antwort:

Im Folgenden sei ein Byte-adressierbarer Hauptspeicher mit 128 Byte Kapazität gegeben. Ein Integer-Wort besteht aus 32 Bit. Der Cache funktioniere direkt-abbildend und sei 32 Byte groß. Haupt- und Cachespeicher verwenden 8 Byte große Blöcke.

Die folgende linke Tabelle entspricht dem Hauptspeicher und enthalte unter den entsprechenden Adressen die darin gezeigten Daten. Die rechte Tabelle entspricht dem Cachespeicher, der zunächst leer sein soll.

c) Vervollständigen Sie den Cacheinhalt wenn in einem Programm nacheinander folgende Adressen angesprochen werden. (9 Punkte)

```
0x18 zeigt auf einen Integerwert;
0x74 zeigt auf einen Integerwert;
0x66 zeigt auf einen Wert vom Typ char;
```

Hauptspeicherinhalt

Platz für Aufteilung Adressbits

Adresse	Byte0	Byte1	Byte2	Byte3
0x00 - 0x03	0x12	0x23	0x34	0x45
0x04 - 0x07	0x54	0x43	0x32	0x21
0x08 - 0x0B	0x13	0x24	0x35	0x46
0x0C - 0x0F	0x64	0x53	0x42	0x31
0x10 - 0x13	0x14	0x25	0x36	0x47
0x14 - 0x17	0x74	0x63	0x52	0x41
0x18 - 0x1B	0x15	0x26	0x37	0x48
0x1C - 0x1F	0x84	0x73	0x62	0x51
0x20 - 0x23	0x16	0x27	0x38	0x49
0x24 - 0x27	0x94	0x83	0x72	0x61
0x28 - 0x2B	0x17	0x28	0x39	0x4A
0x2C - 0x2F	0xA4	0x93	0x82	0x71
0x30 - 0x33	0x18	0x29	0x3A	0x4B
0x34 - 0x37	0xB4	0xA3	0x92	0x81
0x38 - 0x3B	0x19	0x2A	0x3B	0x4C
0x3C - 0x3F	0xC4	0xB3	0xA2	0x91
0x40 - 0x43	0x82	0x93	0xA4	0x45
0x44 - 0x47	0xC4	0xB3	0xA2	0x21
0x48 - 0x4B	0x83	0xA4	0xA5	0x46
0x4C - 0x4F	0xE4	0xC3	0xB2	0x31
0x50 - 0x53	0x84	0x95	0xA6	0x47
0x54 - 0x57	0xB4	0xB3	0xA2	0x41
0x58 - 0x5B	0x85	0x96	0xA7	0x48
0x5C - 0x5F	0xF4	0xE3	0xD2	0x51
0x60 - 0x63	0x96	0x97	0xA8	0x49
0x64 - 0x67	0x04	0xF3	0xE2	0x61
0x68 - 0x6B	0x87	0x98	0xA9	0x4A
0x6C - 0x6F	0x14	0x03	0xF2	0x71
0x70 - 0x73	0x88	0x99	0xAA	0x4B
0x74 - 0x77	0x34	0x13	0x02	0x81
0x78 - 0x7B	0x89	0x9A	0xAB	0x4C
0x7C - 0x7F	0x44	0x23	0x12	0x91

Antwort: Cacheinhalte passend vervollständigen

Cacheinhalt

	Daten							
Tag	Byte	Byte	Byte	Byte	Byte	Byte	Byte	Byte

Aufgabe 4 – Paging / MMU

a) Wie wird beim eindimensionalen Paging aus einer virtuellen Adresse die zugehörige physikalische Seite gebildet? (Hinweis: Erklärung anhand von beschrifteter Skizze möglich) (4 Punkte)

Antwort:

b) Welcher Vorteil bei der Adressgenerierung ist gegenüber der Segmentierung beim Paging gegeben? (2 Punkte)

Antwort:

Beim Zugriff auf einer Seitentabelle und dem Translation-Lookaside-Buffer und dem Cache können die gesuchten Daten jeweils vorliegen (Hit) bzw. sie fehlen (Miss). Angenommen sei nun eine MMU, die wie in der Vorlesung behandelt, die im Cache nicht die virtuelle, sondern physikalische Adressen ablegt.

c) Warum kann dann die folgende Kombination von Hits bzw. Misses nicht eintreten? (2 Punkte)

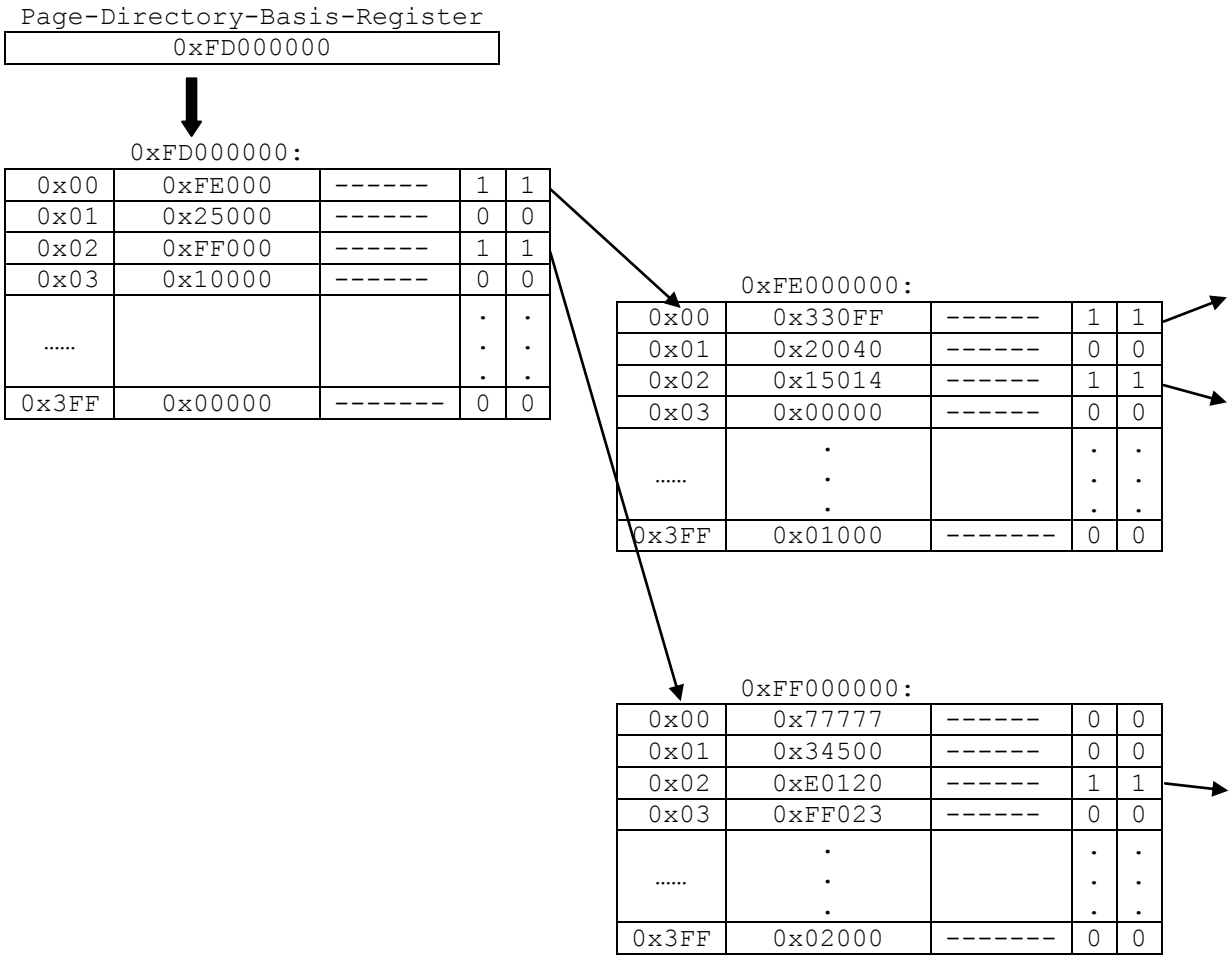
```
TLB-Hit  ->  Page-Miss  ->  Cache-Hit
```

Antwort:

Eine CPU besitzt eine Memory-Management-Unit mit folgenden Eigenschaften:

- zweistufige Adresstabellen
- je 1024 Einträge zu je 4 Byte in den Tabellen mit folgender Bitaufteilung:
  - Bit 31-12: höherwertige Bits der physikalischen Adresse
  - Bit 11-2: unbenutzt
  - Bit 1: Write-Enable-Bit
  - Bit 0: Present-Bit
- Seiten zu je 4 KByte Größe

Gegeben seien folgende Seitentabellen.



d) Welche Einträge stehen in einem voll-assoziativen TLB mit vier Eintragen, nachdem die CPU nacheinander auf die virtuellen Adressen 0x00802111, 0x003FF124, 0x00002346 und 0x003FF834 lesend zugegriffen hat? (10 Punkte)

Antwort:

e) Was passiert, wenn die CPU auf die virtuelle Adresse 0x00801234 zugreift?
(1 Punkt)

Antwort:

Aufgabe 5 – Befehlssatz-Architekturen

In der Vorlesung wurden vier Klassen von Befehlssatz-Architekturen behandelt. Gegeben sei folgendes Programm in einer Pseudokodedarstellung.

```
Push a
Push #1
Sub
Jgtz label1  (springe wenn Ergebnis aus ALU größer 0)
Push #1
Add
Jltz label2  (springe wenn Ergebnis aus ALU kleiner 0)
Push #0
Pop b
Jmp end      (unbedingter Sprung)
label1:
  Push #1
  Pop b
  Jmp end
label2:
  Push #-1
  Pop b
end:
```

a) Auf welcher Klasse von Befehlssatz-Architektur läuft dieses Programm? (1 Punkt)

Antwort:

b) Welche mathematische Funktion wird von dem Programm berechnet? (Falls die mathematische Funktion unbekannt ist, können sie auch das Ergebnis in if-then-else-Notation angeben) (3 Punkte)

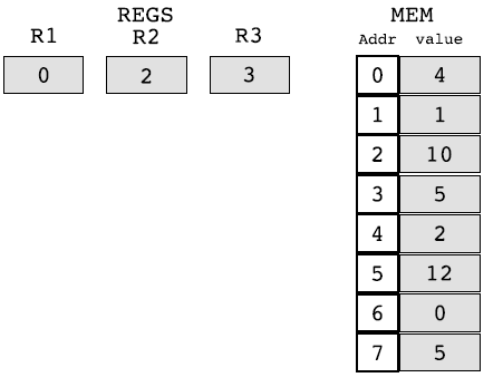
Antwort:

c) Schreiben Sie das Programm so um, dass es von einer CPU umgesetzt wird, die einer Load-Store-Architektur entspricht! Es existieren die gleichen Sprungbefehle wie oben. Ferner können Registervariablen Rx eingesetzt werden. Das Ergebnis soll am Ende ebenfalls in der Speichervariablen b stehen. (3 Punkte)

Antwort:

Aufgabe 6 – Adressierungsmodi

In folgender Abbildung sehen Sie eine mögliche Belegung dreier Register R1-R3 und eines Arbeitsspeichers, bestehend aus 8 Speicherzellen. Alle Werte sind im Dezimalsystem angegeben.



a) Welche Werte würden nach Ausführung der folgenden Additionsanweisungen im Register R1 stehen? (3 Punkte)

i) Add R1 #6 (R3)
ii) Add R1 R2 (R2)
iii) Add R1 2(R1) #5
iv) Add R1 R1 (R1+R1)
v) Add R1 (4) (5)
vi) Add R1 #3 (R3)

Hinweise: Der Additionsbefehl ist im Format Add Ziel Op1 Op2. Die angegebenen Befehle sind nicht als Befehlsfolge zu betrachten. D.h., vor Ausführung eines jeden Befehls gilt immer die Speicherbelegung aus Abb. 1. Von Wortlängen o.Ä. kann abstrahiert werden.

Antwort:

i)
ii)
iii)
iv)
v)
vi)

Aufgabe 7 – Multi-Kern-Architekturen

a) Nennen Sie drei Gründe, weshalb die Industrie seit einigen Jahren auf Multikern-Prozessortechnik setzt! (3 Punkte)

Antwort:

b) Wie lässt sich durch die Regel von Pollack ein Leistungszuwachs durch Multikern-Architekturen auch analytisch begründen? (4 Punkte)

Antwort:

Aufgabe 8 – Hardwarenahe Code-Umformungen

a) Vereinfachen Sie das folgende Code-Stück! Benutzen Sie soweit wie möglich nur **if-goto**-/**goto** Anweisungen mit einfachen Bedingungen! (4 Punkte)

```
if ((!(a < 0)) && ((a < b) || (b != 0)))
        anw1();
   else anw2();
```

Antwort:

Zusätzlicher Platz