

## Prüfung IT-Systeme I (Probeklausur)

---

Datum :  
Bearbeitungszeit : 90 Minuten  
Prüfer : Prof. Dr. Anlauff und Prof. Dr. Ruckert  
Hilfsmittel : Alle eigenen  
Erreichbare Punkte : 86

Name	:
Vorname	:
Studiengruppe	:
Matrikelnummer	:
Unterschrift	:

Bitte kontrollieren Sie, ob Sie ein vollständiges Geheft mit 11 Seiten erhalten haben.

**Aufgabe 1:****(8 Punkte)**

Rechnen Sie folgende Zahlen zwischen den angegebenen Zahlssystemen um. Auch auf den Rechenweg kann es Punkte geben!

System zur Basis $B = 4$	System zur Basis $B = 10$	System zur Basis $B = 12$
1230		
		AB3

**Aufgabe 2:****(8 Punkte)**

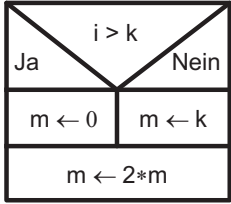
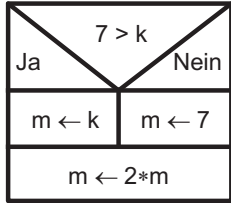
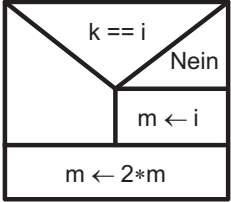
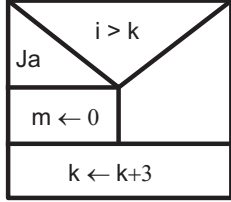
- 1.) Welche Bitmuster haben die in der ersten Spalte angegebenen Zahlen im 2-Komplement, wenn für ihre Darstellung 64 Bit zur Verfügung stehen? Tragen Sie das Ergebnis in die Spalte "Bitmuster" ein.
- 2.) Geben Sie in der Spalte "MMIX-Befehle" jeweils eine effiziente Möglichkeit an, das Bitmuster in ein Register des MMIX zu bringen, ohne die Werte aus dem Speicher zu laden.

	<b>Bitmuster</b>	<b>MMIX-Befehle</b>
+1		
-1		
kleinste darstellbare negative Zahl		
größte darstellbare positive Zahl		

**Aufgabe 3:**

**(20 Punkte)**

Schreiben Sie eine effiziente Folge von MMIX-Befehlen, die die folgenden Struktogramme realisieren. Dabei sollen Sie die Definitionen  $i$  IS \$0,  $k$  IS \$1 und  $m$  IS \$2 voraussetzen. Alle anderen Register sind frei verwendbar.

<p>a)</p> 	
<p>b)</p> 	
<p>c)</p> 	
<p>d)</p> 	

**Aufgabe 4:**

**(16 Punkte)**

Gegeben sind die folgenden Felder im Datensegment.

```

      LOC   Data_Segment
      GREG  @
feldA  OCTA 1,2,3,4,5
feldB  TETRA 0
      LOC   feldB+10*4
      GREG  @
feldC  BYTE 0
      LOC   feldC+1000
    
```

Welche der folgenden Feldzugriffe sind möglicherweise fehlerhaft? Begründen Sie Ihre Antwort. Achten Sie auf Alignment (Ausrichtung von Bytes, Wydes etc.), Syntax und Feldgrenzen.

	Begründung
LDA \$1,feldA	
• LDO \$0,\$1,4	
LDA \$1,feldA	
• LDO \$0,\$1,40	
LDA \$2,feldB	
• LDT \$0,\$2,4*8	
LDA \$2,feldB	
SET \$255,3	
• LDT \$0,\$2,4*\$255	
LDA \$2,feldB	
• LDT \$0,\$2,4*32	
LDA \$3,feldC	
• LDO \$0,\$3,5	
LDA \$3,feldC	
• LDB \$0,\$3,50	
LDA \$3,feldC	
• LDB \$0,\$3,500	

**Aufgabe 5:****(10 Punkte)**

Gegeben sei folgendes Listing eines MMIX-Programms:

```

                LOC Data_Segment
($254=#20000000  GREG @
                00000000)
2000000000000000: oflo BYTE Integer Overflow,#d,#a,0
...000: 496e7465
...004: 67657220
...008: 4f766572
...00c: 666c6f77
...010: 0d0a00

                % Interrupt Handler for Integer Overflow
                LOC 32
0000000000000020: LDA $255,oflo
...020: 23fffe00
...024: 0000701  TRAP 0,Fputs,StdOut
...028: 00000000  TRAP 0,0,0
                LOC #100
...100: fe000016  Main GET $0,rA enable oflow check
...104: e3014000  SET $1,1<<14
...108: c0000001  OR $0,$0,$1
...10c: f6160000  PUT rA,$0

...110: e0018000  SETH $1,#8000 groesste darstellbare
...114: c4010101  NOR $1,$1,$1      positive Zahl
...118: 21010101  ADD $1,$1,1

...11c: 00000000  TRAP 0,0,0
Symbol table:
Main = #0000000000000100 (1)
oflo = #2000000000000000 (2)

```

Ergänzen Sie die folgenden Aussagen:

- 1.) Der Befehl, mit dem die Assemblierung der Quelldatei `test.mms` inklusive Erzeugung des gegebenen Listings `test.lst` gestartet wird,

lautet \_\_\_\_\_

- 2.) Der Befehl, mit dem der MMIX-Interpreter für die in 1 erzeugte Objektdatei im interaktiven Modus gestartet wird,

lautet \_\_\_\_\_

- 3.) Angenommen, Sie befinden sich nun im interaktiven Modus. Sie können nun

- a) das Programm um einen Befehl weiterschalten  
mit \_\_\_\_\_
- b) den Inhalt von 10 Speicherzellen ab Adresse #2000000000000000  
im Hexformat anzeigen  
mit \_\_\_\_\_
- c) an der Adresse des Befehls `OR $0,$0,$1` einen Ausführungs-Breakpoint  
setzen  
mit \_\_\_\_\_
- d) die Abarbeitung des Programms bis zum nächsten Breakpoint bzw.  
bis zum Ende fortsetzen  
mit \_\_\_\_\_
- e) den Inhalt des lokalen Registers `$2` dezimal anzeigen  
mit \_\_\_\_\_
-

**Aufgabe 6:**

**(4 Punkte)**

Mit den folgenden MMIX-Befehlen sollen maximal 20 Zeichen vom Standard Input gelesen werden:

```
LDA    $255, Arg
TRAP   0, Fgets, StdIn
```

Ergänzen Sie die hierzu benötigten Datenstrukturen im Datensegment:

```
LOC    Data_Segment
GREG   @
      ???
```



**Aufgabe 7:****(8 Punkte)**

Das folgende MMIX-Programm soll eine zyklische Verschiebung von Elementen `ai` eines Arrays bewirken.

```
1 * Zyklisches Verschieben von array-Elementen
2 * feste Basis, zwei offsets
3
4      LOC   Data_Segment
5      GREG  @
6
7 A      OCTA 137,256,356,461,592,643,782,841,952
8 N      OCTA (@-A)/8      Anzahl Elemente
9
10 n     IS   $1
11 i     IS   $2
12 base  IS   $3
13 offsnew IS $4
14 offsold IS $5
15 ai    IS   $6
16 help  IS   $7
17 test  IS   $8
18
19      LOC   #100
20      GREG  @
21
22 Main  LDA   base,A      Basisadresse für array
23      SET   offsold,0
24      SET   offsnew,0
25      LDO   n,N
26      SET   i,1
27      LDO   help,A      1. Element retten
28
29 Loop  CMP   test,i,n
30      BNN  test,Weiter
31 ???
32      ADD   offsnew,offsnew,8
33      LDO   ai,base,offsnew  neues Element lesen
34      STO   ai,base,offsold  und nach links schieben
35      ADD   i,i,1
36      JMP   Loop
37
38 Weiter STO  help,base,offsnew
39      TRAP  0,Halt,0
```

Speicherbelegung vorher:

M8[#2000000000000000]=137  
M8[#2000000000000008]=256  
M8[#2000000000000010]=356  
M8[#2000000000000018]=461  
M8[#2000000000000020]=592  
M8[#2000000000000028]=643  
M8[#2000000000000030]=782  
M8[#2000000000000038]=841  
M8[#2000000000000040]=952  
M8[#2000000000000048]=9  
M8[#2000000000000050]=0

Speicherbelegung nachher:

M8[#2000000000000000]=256  
M8[#2000000000000008]=356  
M8[#2000000000000010]=461  
M8[#2000000000000018]=592  
M8[#2000000000000020]=643  
M8[#2000000000000028]=782  
M8[#2000000000000030]=841  
M8[#2000000000000038]=952  
M8[#2000000000000040]=137  
M8[#2000000000000048]=9  
M8[#2000000000000050]=0

Ergänzen Sie das Programm an der mit ??? markierten Stelle (Zeile 31) mit einem geeigneten Befehl der folgenden Auswahl.

**Hinweis:** Markieren Sie die folgenden Befehle als geeignet (Ja) oder ungeeignet (Nein).

- SET offsold,offsnew
- SET offsnow,offsold
- SUB offsold,offsold,8
- SUB offsold,offsnew,8
- ADD offsold,offsold,8
- ADD offsold,offsnew,8

**Aufgabe 8:****(12 Punkte)**

Schreiben Sie ein PUSHJ-Unterprogramm, das als Parameter eine Uhrzeit wie folgt erhält:

- Sekunden in \$0
- Minuten in \$1
- Stunden in \$2

und diese Uhrzeit um eine Sekunde weiterzählt. Die veränderten Werte für Sekunden, Minuten und Stunden sollen (in \$0 bis \$2) zurück gegeben werden.

Beispiel: aus den Werten (Stunden, Minuten, Sekunden) 5, 27, 59 soll also 5, 28, 0 werden und entsprechend aus 23, 59, 59 soll 0, 0, 0 werden etc.