Berner Fachhochschule
Haute école spécialisée bernoise
Bern University of Applied Sciences

1 Basis Fragen (Punkte: 4)

Geben Sie eine jeweils technische Erklärung und erläutern Sie die Bedeutung des Vorfalls.

1.1 Was bedeutet Carry (Punkte: 1)

Ihre Antwort:

Carry bedeutet behalte.

Wird bei arithmetischen Operationen gebraucht

1.2 Was bedeutet Overflow (Punkte: 1)

Ihre Antwort:

Wenn etwas zu gross wird. Overflow = überläuft

⇒ Bei arithmetischen Operationen auf einen „signed Integer" die Grösse nicht reicht wird das Overflow Flag gesetet.

1/2

1.3 Was versteht man unter Parity (Punkte: 1)

Ihre Antwort:

Bedeutet Parität. ~~Zeigt zu~~ ?

Das Flag zeigt an, ob im unteren Teil vom Byte die Anzahl der 1 gerade oder ungerade ist.

Mehr - 1/4

1.4 ROL (Punkte: 1)

Gegeben ist folgender Assembler Code:

```
mov rax, 0FFH
mov rbx, 0FF000000000000FFH
and rbx, rax
rol rbx,60
```

Was befindet sich danach in rbx?

Ihre Antwort:

Es wird 60 mal nach links rotiert. 60:4=15

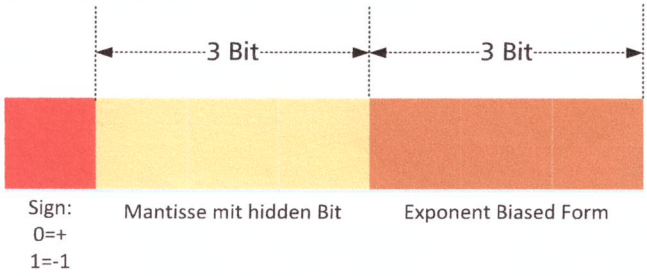
Da hex

Am Schluss ist in ebx:

~~0F~~ FFF000000000000FH

2 Floating Point Zahlen (Punkte: 3)

Gegeben ist folgendes Format:



Fragen:

- Erläutern Sie das Format (Bedeutung).
- Welches ist die grösste und welches die kleinste mit diesem Format darstellbare Zahl? Geben Sie zur Beantwortung dieses Punktes präzise technische Erklärungen.
- Wie wird die Zahl -1.1_{10} in diesem Format dargestellt?

Rechnungshilfe:
$0.1_2=0.5_{10}$ $0.01_2=0.25_{10}$ $0.001_2=0.125_{10}$

Ihre Antwort:

- $-1,1 =$ ~~Negativ~~ neg → 1 Sign

  Vorkomma $1:2 = 0\ 1$

  Nachkomma $0,1 \cdot 2 = 0$
  $0,2 \cdot 2 = 0$
  $0,4 \cdot 2 = 0$
  $0,8 \cdot 2 = 1$
  $0,4 \cdot 2 = 0$

  $\Rightarrow 1'011'000_B$

  Exponent = ~~127~~ 3 (1 0)

- Der Exponent in der ~~Bias~~ Biased Form ist 4-1=3

  Das Sign ist das Vorzeichen

Sieht K

1

3 Umrechnen etc. (Punkte: 6)

3.1 Assembler Addition (Punkte: 1.5)

Gegeben ist folgender Code:

```
; Code 1
mov al, 127
add al, -128

; Code 2
mov al, 127
sub al, 128
```

Fragen:

- Vergleichen Sie Code1 und Code2.
- Was passiert technisch in beiden Fällen?
- Werden die Codeteile fehlerfrei assembliert?
- Ihr Kommentar und Ihre Erläuterung?

Ihre Antwort:

- Bei beiden wird zuerst 127 ins Register al geschrieben.
- Beim 1 dann + - 128 gerechnet, was aber nicht geht, da -128 zu gross für al ist.
- Bei 2 dann mit sub subtrahiert geht weil 128 ins Register al passt.
- Al ist nur 8Bit

d. 1.5/2

3.2 Hexadezimale Rechnung (Punkte: 1.5)

Berechnen Sie folgende Aufgabe in hexadezimal (wichtig ist der Lösungsweg, also nicht nur das Resultat): $AB.AB_{16} - 7.7_{16} = XX.YY_{16}$

Ihre Antwort:

AB,AB = 1010 1011, 1010 1011 = 0,625

7,7 = 0111, 0111 0000

$AB,AB = 10 \cdot 16^1 + 11 \cdot 16^0 + 10 \cdot 16^{-1} + 11 \cdot 16^{-2} = 171,668$

$7,7 = 7 \cdot 16^0, 7 \cdot 16^{-1} = 7,438$

164 : 16 = 10 Rest 4
10 : 16 = 0 Rest 10 ; in 16 ←

0,23 · 16 = 3,68
0,68 · 16 = 10,88

164,23

4A,3B ✓ 1.5

3.3 Hexadezimale Rechnung (Punkte: 1.5)

Berechnen Sie folgende Aufgabe in hexadezimal (wichtig ist der Lösungsweg, also nicht nur das Resultat): $100_{16} : 20_{16} = XXX_{16}$

Ihre Antwort:

$256 : 32 = 8_{10}$ $\quad 8_{10}$ in hex $\Rightarrow 8_{16}$

$100 \rightarrow 1 \cdot 16^2 = 256_{10}$

$20 \rightarrow 2 \cdot 16^1 = 32_{10}$

V 1.5

3.4 Rechnung im Zweierkomplement (Punkte: 1.5)

Berechnen Sie folgende Aufgabe in binär (wichtig ist der Lösungsweg, also nicht nur das Resultat): $AB_{16} - 1010_2 = XXX_{16} = YYY_2$

Ihre Antwort:

$AB = 1010\ 1011$

0000 1010
invert 1111 0101
+ 1
1111 0110

1010 1011
+ 1111 0110
(1) 1010 0001$_B$
Carry

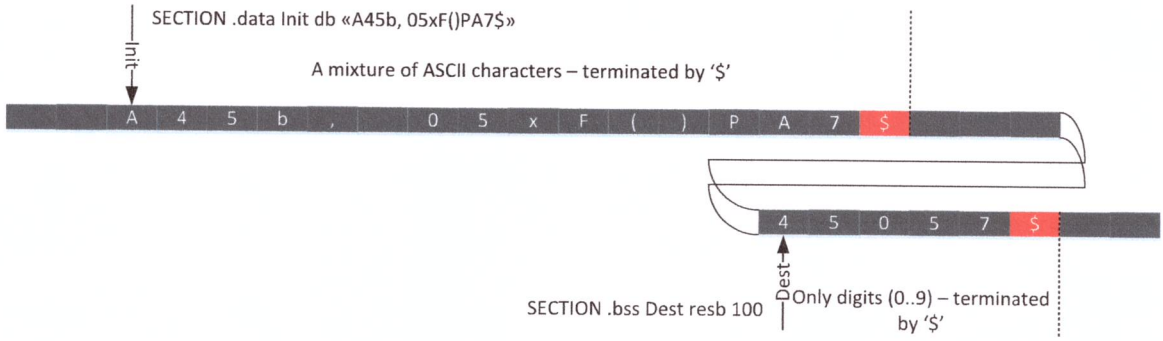
1.5

4 Assembler Prozeduren (Punkte: 10)

Schreiben Sie folgende Assembler Prozeduren:
Den push- und den pop-Teil können Sie jeweils weglassen!

4.1 Filtern von Digits (Punkte: 3)

Gegeben ist ein Speicherbereich (`Init`), in dem sich eine Folge von beliebigen ASCII Zeichen befindet.
Schreiben Sie nun eine Prozedur, die alle Digits (Zeichen `'0'` ... `'9'`) ihrer Ordnung entsprechend filtert und in einem neuen String (`Dest`) im Speicher ablegt. Der neue String soll ebenfalls mit dem `'$'` Zeichen beendet werden.



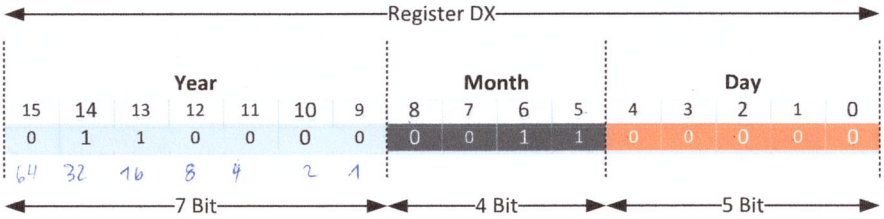
Ihre Antwort:

```
filterDigits:
; push




; pop
ret
```

Berner Fachhochschule
Haute école spécialisée bernoise
Bern University of Applied Sciences

4.2 DOS Datumsveränderung (Punkte: 4)

In DOS wird ein Datum in folgendem Format gehandelt:
Der Jahreswert wird zur Jahreszahl **1980** addiert!



Schreiben Sie eine Assembler Prozedur, die den Tageswert um den Faktor 1 erhöht und dabei natürlich auch den Monats- und den Jahreswert erhöht (wenn nötig). Wir vereinfachen die Aufgabe und ignorieren die Schaltjahre. Zudem gehen wir davon aus, dass alle Monate 31 Tage haben. **Skizzieren Sie Ihren Lösungsweg.**
Hinweis: Verwenden Sie Shift Befehle und Masken zur Lösung dieser Aufgabe.
Bestimmen Sie zudem den **maximal** mit diesem Format darstellbaren Datumswert.
Ihre Antwort:
Der maximale Datumswert ist:

```
incrementOneDay:
; push
push r8
push r9
push r10
mov r8, rdx    ;days
mov r9, rdx    ; month
mov r10, rdx   ; year

and r8, 0000000000000001FH   ; maskiese days
and r9, 00000000000000F0H    ; maskiere month
and r10,

shr r9, 5   ; alles nach rechts
shr r10, 9  ;  "

inc r8  ; day erhöhen
cmp r8, 31
jb .nochange
xor r8, r8   ; reset days
inc r9       ; month erhöhen
cmp r9, 12
jb .nochange
xor r9, r9   ; mont reset
inc r10
cmp r10

; pop
ret
```

```
.nochange
shl r8, 59   ; nach links schieben
shl r9, 60   ;  "
shl r10, 57  ;  "

xor rdx, rdx   ; Löschen
shld rdx, r10, 7  ; Jahr auffüllen
shld rdx, r9, 4   ; Month auffüllen
shld rdx, r8, 5   ; Day auffüllen

pop r10
pop r9
pop r8
ret
```

✓ 4.

Max: 31.12.65024

4.3 Fibonacci (Punkte: 3)

Gegeben ist folgende Java Anwendung, die die Fibonacci Zahlen berechnet:

Test.java

```
1  package fibo;
2
3  public final class Test {
4
5      public static void main(String[] args) {
6          for(int i=1; i<10; i++) System.out.println(fib(i));
7      }
8
9      private  static int fib(int n) {
10         int i1 = 1, i2 = 1, tmp;
11         while (n-- > 2) {
12             tmp = i1 + i2;
13             i2 = i1;
14             i1 = tmp;
15         }
16         return i1;
17     }
18 }
```

Problems Javadoc Declaration Search Console

<terminated> Test [Java Application] C:\Program Files\Java\jre1.8.0_101\bin\javaw.exe (29.10.2016, 08:10:23)

```
1
1
2
3
5
8
13
21
34
```

Schreiben Sie eine Assembler Prozedur **`Fibonacci`**, die exakt wie in Java die Berechnung einer gesuchten Fibonacci Zahl durchführt. Achten Sie darauf, dass Sie Ihre Lösung sauber kommentieren - was Ihnen die Arbeit sicher erleichtert!

Ihre Antwort:

```
Fibonacci
[redacted]
mov eax, "Zahl"   ; Zahl einlesen
cmp eax, 1        ; ist Zahl 1?
ja fibo           ; wenn nein -> Prozedur
mov ecx, 1        ; wenn ja via ecx 1 zurückgeben
jmp exit
fibo:
dec eax           ; n-1
mov edx, eax      ; n-1
push edx          ; save n-1
push eax          ; set argument n-1
call Fibonacci
pop eax           ; pop n-1
dec eax           ; n-2
push ecx
push eax
call Fibonacci
pop eax           ; = Fib(n-1)
add ecx, eax      ; = Fib(n-1)
                  Fib(n-2
exit:
ret
```

✓ 3