Testfragen und Aufgaben zu Bitbefehlen

1. Bestimmen Sie den Inhalt der Register AX, BX, CX, DX nach der Ausführung der folgenden Befehle:

```
mov ax,0FEDCh
and ax, 1234h
mov bx, 1234h
or  bx, 4321h
mov cx, 6789h
xor cx, 9876h
mov dx, 12EFh
not dx
```

   Lösung: AX=1214h, BX=5335h, CX=FFFFh, DX=ED10h

2. Bestimmen Sie den Inhalt der Register AL, BL, CL, DL nach der Ausführung der folgenden Befehle:

```
mov al,95h
shl al,2
mov bl,95h
sar bl,1
mov cl,95h
rol cl,2
mov dl,95h
sal dl,1
rcl dl,1
```

   Lösung: al=54h, bl=CAh, cl=56h, dl=55h

3. Schreiben Sie eine Programmsequenz, die folgende Bitoperationen an Register AX vornimmt:

   - Bit 0 und Bit 5 löschen
   - Bit 1 und Bit 3 setzen
   - Bit 2 und Bit 7 invertieren
   - alle anderen Bits unverändert lassen

   Formulieren Sie die Konstanten binär und hexadezimal!

   Lösung:

```
and ax,1111111111011110b    ; oder and ax,0FFDEh
or  ax,0000000000001010b    ; oder or ax,0Ah
xor ax,0000000010000100b    ; oder xor ax,84h
```

4. Führen Sie nur mit Bitbefehlen sowie `mov` und `add` folgende Berechnung aus:

   AX = 17*BX + 9*CX (Registerüberlauf wird nicht betrachtet.)

   Lösung:

```
mov si,bx   ; Kopie von bx anlegen
shl bx,4    ; bx=bx*16
add bx,si   ; bx=bx*17
mov di,cx   ; Kopie von cx anlegen
shl cx,3    ; cx=cx*8
add cx,di   ; cx=cx*9
add bx,cx   ; bx=17*bx + 9*cx
mov ax,bx   ; Ergebnis nach ax
```

5. Schreiben Sie ein Programmstück, das nur mit Bitbefehlen sowie `add` arbeitet und folgendes ausführt: Der Inhalt von ax soll um eins erhöht werden, wenn in Register BX Bit 7=0 ist!

```
; Loesung1:
    not bx       ; invertieren
    shr bx,7     ; shift right bx,7: Bit 7 ist jetzt LSB
                 ; es geht auch: rol bl,1
    and bx,1     ; LSB stehen lassen, restliche Bits 0 setzen
    add ax,bx    ; addieren

; Loesung2:
    shl bx,8     ; shift left bx,8: Bit 7 ist jetzt MSB
    shr bx,15    ; Bit 7 ist jetzt LSB, andere Bits sind 0
    xor bx,1
    add ax,bx    ; addieren

; Loesung3:
    shl bx,9     ; shift left bx,9: Bit 7 jetzt im Carryflag
    mov bx,0     ; Bit 7 ist jetzt LSB, andere Bits sind 0
    rcl bx,1     ; rotate through carry left,CF in LSB von BX
    add ax,bx    ; addieren
```