

## Ein „Hallo Welt!“-Programm

© Michael Wellner 2004-02-11

### 1. Voraussetzungen

Um das Beispiel auszuprobieren brauchst du nicht mehr als ein Stand-Alone-Assembler (Im Beispiel wird mit Debug, der bei MS-DOS enthalten ist, gearbeitet.), ein paar Kenntnisse über die Computer-Hardware und einen guten Menschenverstand... ☺

Es ist ganz praktisch wenn du dir schon einmal meine PowerPoint-Präsentation „Die CPU-Register eines Pentium“ angeschaut hast.

### 2. Das Programm

Öffne die MS-DOS Eingabeaufforderung und gebe folgendes ein:

```
C:\>debug  
-a
```

Der Debug-Modus zeigt dir eine Speicheradresse an (zb.: 12AA:0100) jetzt können wir einen Befehl eingeben, Enter drücken worauf die nächste Speicheradresse angezeigt wird. Nun kannst du also das folgende Programm eingeben.

```
Mov ax, 3  
Int 10h  
Mov ax,b800  
Mov ds,ax  
Mov byte ptr [0],48  
Mov byte ptr [2],61  
Mov byte ptr [4],6C  
Mov byte ptr [6],6C  
Mov byte ptr [8],6F  
Mov byte ptr [12],57  
Mov byte ptr [14],65  
Mov byte ptr [16],6C  
Mov byte ptr [18],74  
Mov ah,4C  
Int 21
```

Nachdem du den letzten Befehl geschrieben hast drücke 2mal Enter und gib folgende Befehle ein um das Programm zu speichern:

```
-n p1  
-r cx  
CX 0000  
:E1  
-w 12AA:100
```

**Achtung!** Für 12AA:100 musst du deine erste Speicheradresse des Programms angeben.

Du musst dem Dateiname noch .exe anhängen, dass du das Programm testen kannst. Wenn du keinen Fehler gemacht hast steht dann Hallo Welt auf dem Bildschirm.

### 3. Wie funktioniert p1?

Die ersten 2 Befehle

```
Mov ax,3  
Int 10h
```

Sorgen dafür, dass der Bildschirm gelöscht wird.

Im Programm kommt eigentlich nur der mov-Befehl vor, dieser Befehl kopiert ein Datum von einem Register oder einem Speicher in ein Register oder ein Speicherplatz. Die Syntax lautet:

```
Mov Ziel,Datum
```

Das Datum kann ein Register, eine Speicheradresse oder bei den meisten Zielen auch eine Zahl sein. Der Befehl `mov ds,b800` existiert beispielsweise nicht deswegen haben wir im Beispiel `b800` erst ins `ax`-Register und von dort ins `ds`-Register geladen.

Die letzten zwei Befehle beenden das Programm.

### 1. Der cmp-Befehl

```
cmp operand1, operand2
```

Der Prozessor vergleicht 2 Operanden durch Subtraktion, das Ergebnis wird dabei nicht gespeichert. In der folgenden Tabelle wird dargestellt, wie sich der cmp-Befehl auf das C- und das Z-Flag auswirkt.

| <u>Bedingung</u> | <u>Z-Flag</u> | <u>C-Flag</u> |
|------------------|---------------|---------------|
| Op1 = Op2        | 1             | 0             |
| Op1 <> Op2       | 0             | X             |
| Op1 >= Op2       | X             | 0             |
| Op1 < Op2        | 0             | 1             |
| Op1 > Op2        | 0 und         | 0             |
| Op1 <= Op2       | 1 oder        | 1             |

Nachdem cmp-Befehl wird meist ein bedingter Sprungbefehl verwendet.

### 2. Die Tastaturabfrage

Das Programm wartet nach folgendem Interrupt auf eine Tastaturabfrage, wenn eine Taste gedrückt wurde wird der ASCII-Code in al gespeichert.

```
Mov ah, 0  
Int 16h
```

### 3. Der lodsb- und der lodsw-Befehl

Diese Befehle laden ein Byte bzw. ein Word aus der Adresse ds:si in al bzw. ax. Si wird nach der Ausführung neu gesetzt.

```
Lodsw  
lodsb
```

#### 4.Beispielprogramm

Im folgenden habe ich ein Programm geschrieben was die Tastatur abfragt und den eingegebenen Text auf den Bildschirm schreibt, wenn man die Esc-Taste drückt wird der Text in der 2.Zeile ausgegeben, danach wird das Programm beendet Programm beendet.

```
Mov ax,3
Int 10h                ;Interrupt für Programmstart
Mov ax,b800
Mov ds,ax              ;Adresse für Bildschirm in ds
Mov bx,0
Mov ah,0
Int 16h                ;BIOS-Funktion für Tastaturabfrage
Cmp al,1b
Jz 11c
Mov byte ptr [bx],al   ;Buchstabe auf Bildschirm schreiben
Add bx,2
Jmp 10d                ;Springe zurück
Mov cx,bx
Mov si,0
Lodsw
Mov bx,a0              ;Adresse für 2.Zeichen, 2.Zeile in bx
Add bx,si
Mov word ptr [bx],ax   ;zeichen schreiben
Sub cx,2
Jnz 122
Mov ah,4c
Int 21h                ;Programm beenden
```