HOCHSCHULE harz	FB Automatisierung und Informatik Laborpraktikum Mikrocomputertechnik & Assemblerprogrammierung		
Versuch: **MC&AP I**	Thema: **Softwareentwicklungsprozeß, Befehlssatz 8086, Debuggerfunktionen**		
Protokollführer: Mitarbeiter 1: Mitarbeiter 2:		Seminargruppe: Versuchsgruppe: Laborplatz: Datum:	
Kolloquium:	Versuchstestat:	Protokoll:	Abschlußtestat:

1. Versuchsziele

- Kennenlernen der Komponenten des SEP
- Assembler-/Linkersteuerung
- Debuggerfunktionen
- einfache Bildschirmausgabe
- Vertrautmachen mit den Funktionen des Debuggers zur effektiven Programmtestung

2. Versuchsgrundlagen

- Softwareentwicklungsprozeß für Assemblerprogramme mit TASM/TLINK/TD (Borland) für I 8086
- Befehlsliste des I 8086
- Debuggerfunktionen

3. Versuchsdurchführung

3.1 Schreiben Sie ein Programm, mit dem die Register AX, BX, CX, DX, SI und DI mit den Daten AA55H geladen werden. Beachten Sie hierzu die vorgegebene Programmstruktur (RAHMEN.ASM).

3.2 Untersuchen Sie im Debugger Möglichkeiten, Registerinhalte, Speicherinhalte und Flagbelegungen zu manipulieren.

3.3 Führen Sie mit den **unteren**, siehe die nächste Seiten, und selbst gewählten Beispielen Operationen zur Addition, Subtraktion, Multiplikation und Division von binären Festkommazahlen (vorzeichenlos und vorzeichenbehaftet) sowie von BCD-Zahlen aus. Erläutern Sie die Ergebnisse!

3.4 Schreiben Sie ein Programm, bei dem mittels „MOVSB/W“ die Zeichenkette „1. BILDSCHIRMAUSGABE“ auf dem Bildwiederholspeicher der VGA (Adr. ∅B8∅∅∅H) ausgegeben wird! Was stellen Sie fest?

3.5 Geben Sie mittels geeignetem Schleifenprogramm den gesamten Zeichensatz aufsteigend auf dem Bildschirm aus! Werten Sie die Ergebnisse aus!

Hinweis:

Zum Protokoll dieses und der folgenden Labore gehört neben der Protokollierung der Ergebnisse und der Beantwortung der Fragen auch ein möglichst gut kommentiertes Programm zu jedem Versuchsteil.

Zusatzaufgabe:

Entwickeln Sie ein Programm zur Umwandlung von allen Kleinbuchstaben in Großbuchstaben (ASCII) im direkten Bildschirmspeicherzugriff.

Rechenaufgaben für Aufgabe 3.3

Bestimmen Sie VORHER, wie das Ergebnis lautet und welche Flags gesetzt werden.

Vorzeichenlose Addition:

Beispiele:

```
CLC
Mov al, 33
Mov,22
Add al,bl   ;   Aufg1   al=??
```

```
CLC
Mov al,
Mov,
Add al,bl   ;   Aufg2   al=??
```

Aufgabe:
Jeweils zwei Register symbolisieren eine Zahl (high- und low-byte)
AH=02h AL=0F0h
BH=04h BL=1FH

Aufgabe: Programmieren Sie folgende Additionen

Reg1	Reg2	ErgReg	Cf	zf	sf	of
AL=33	BL=22					
AL=210	BL=52					
AL=210	BL=46					
AX=100	BX=1000					
AX=65530	BX=6					

Vorzeichenlose Substraktion:

Aufgabe: Programmieren Sie folgende Subtraktionen

Reg1	Reg2	ErgReg	Cf	zf	sf	of
AL=33	BL=2					
AL=33	BL=33					
AX=300	BX=2					
AL=33	BL=34					
AL=33	BL=35					

Vorzeichenbehaftete Subtraktion:

Aufgabe: Programmieren Sie folgende Subtraktionen

Substrahieren Sie mittels sbb

- Mov ax,7h
- Mov bx,5h

Vorzeichenlose Multiplikation:

Beispiele:

AH=30
BL=2
Ergebnis: AH*BL => AX = 00A4

DX=0
AX=300
BX=250
Ergebnis: AX*BL

AX=75000= 124F8h => DX=1, AX=9465=24F8h

Aufgabe: Programmieren Sie folgende Multiplikation

Reg1	Reg2	ErgReg	Cf	zf	sf	of
AX=33	BL=2					
AX=33	BL=8					
AX=300	BL=250					
AX=300	BX=300					

Vorzeichenlose Division:

Programmieren Sie folgende Divisionen:

Reg1	Reg2	ErgReg	ErgReg
AX=171	BL=10		
4101	10		

Wenn Fehler auftreten, bestimmen Sie die Ursache und dokumentieren Sie diese in Ihrem Quellcode.

Vorzeichenbehaftete Multiplikation:

Aufgabe: Programmieren Sie folgende Multiplikation

Reg1	Reg2	Operation	ErgReg	ErgReg
ax,9876h	bx,7654h	Vorzeichenlos		
ax,9876h	bx,7654h	Vorzeichenlbehaftet		

Vorzeichenbehaftete Division:
Aufgabe: Programmieren Sie folgende Multiplikation

Reg1	Reg2	Operation	ErgReg	ErgReg
AX=89ABh	BL=0A6h			
DX=1234h AX=5432h	BX=8765h			
DX=9ABCh AX=5432h	BX=8765h			

BCD-Arithmetik:

Reg1	Reg2	Operation	ErgReg	ErgReg
AX=178h	BX=178h	gepackt		
AX=1234h	BX=2345h	gepackt		
AX=0707h	0505h	ungepackt		

Farbattribute für den direkten Speicher

Bits:

- 012 Red Green Blue, Textfarbe
- 3 Intensivbit
- 456 Red Green Blue, Hintergrundfarbe
- 7 Blinkend, funktioniert nur im Vollbildschirm
-

Beispiele:

- 070h: weißer Hintergrund, schwarzer Text
- 017h blauer Hintergrund, weißer Text
- 027h grüner Hintergrund, weißer Text
- 047h roter Hintergrund, weißer Text
- 024h grüner Hintergrund, roter Text
- 0A4h Blinkend, grüner Hintergrund, roter Text, alt+Enter

Blinkend funktioniert nur im DOS-Modus: umschalten mit Alt+Enter

Anhang

IDE-Oberfläche

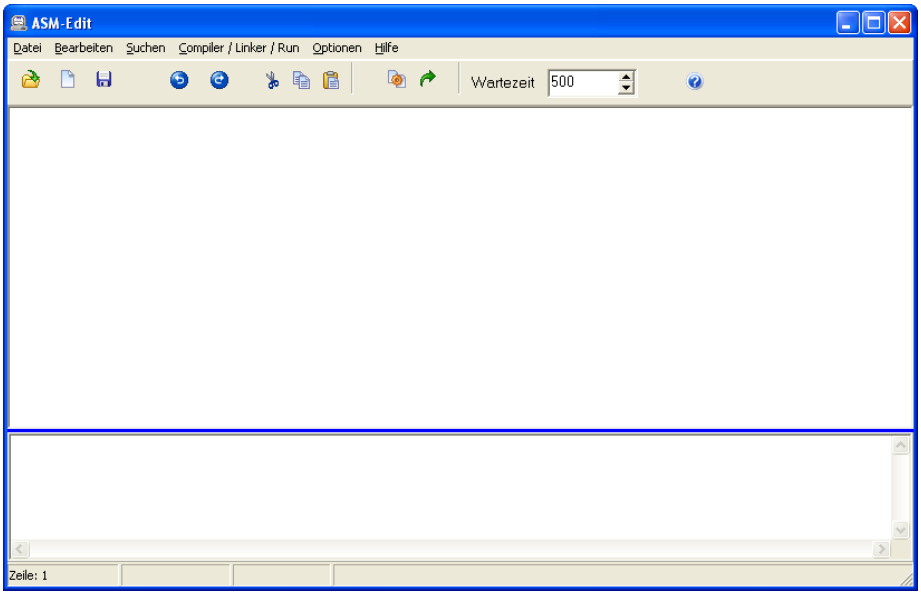

Abbildung 1 Programmoberfläche

Wichtige Hotkeys:

Strg+O Datei öffnen
Strg+N Neue Datei

F8	**Übersetzen, Compiler**
F9	**Compiler, Linker, Debugger starten**
Strg+D	**Dos-Ebene zum Testen eines ASM-Programms**

Strg+F	Suchen
F3	Weitersuchen
Strg+C	Kopieren
Strg+V	Einfügen
Strg+X	Ausschneiden
Strg+Y	Aktuelle Zeile löschen
Strg+A	Alles markieren
Strg+T	Taschenrechner
Strg+E	Explorer

Debugger

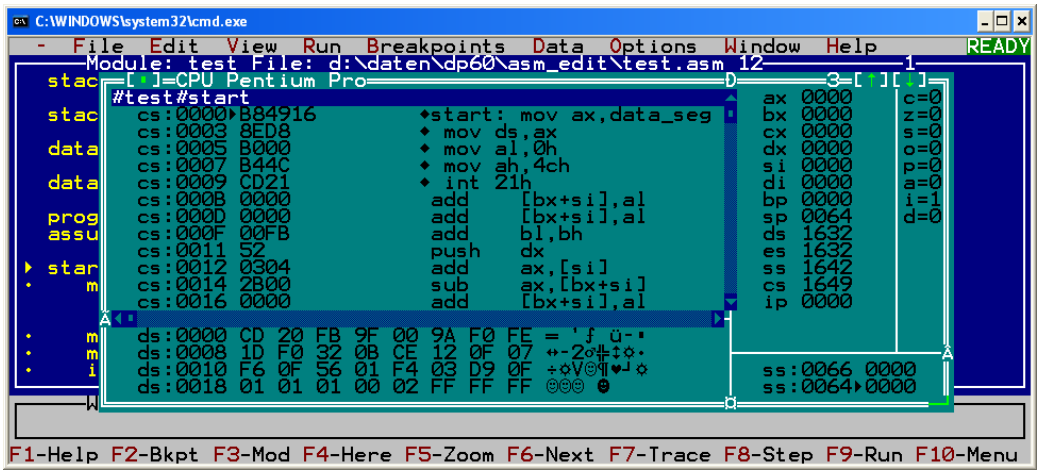

Abbildung 2 Assembler Debugger

Alt+X	Beenden
F7	Debugging in einer Prozedur
F8	Debugging über einer Prozedur
F2	Breakpoint setzen/löschen
F9	Run
F4	Run bis Cursor
Alt+F5	Dos-Bildschirm

Loop-Schleife mit F7

Mit dem Menü Options, Eintrag “Save Options” kann man die Aufteilung speichern:

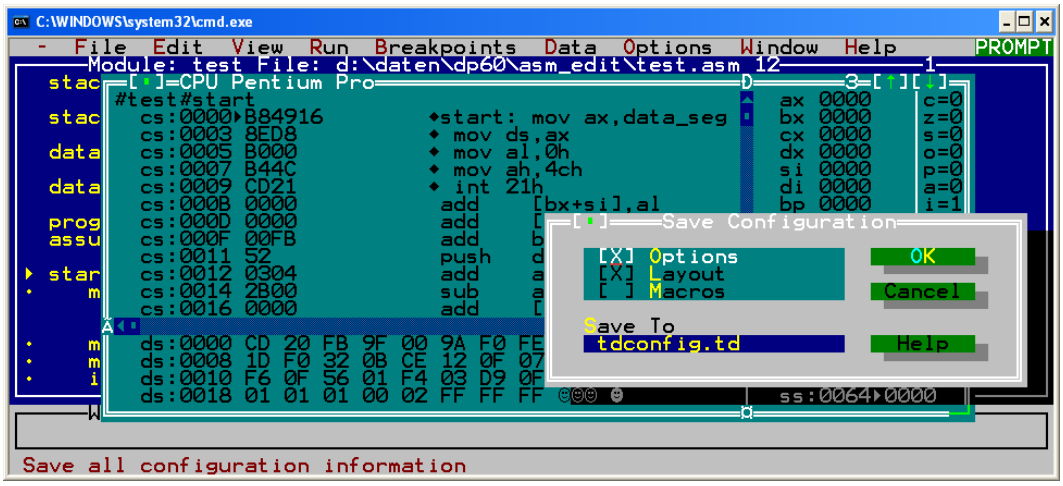

Abbildung 3 Layout der Fenster speichern