

Inhaltsverzeichnis

1.	Die 8086/80186/80286-Basis-Architektur	1
1.1	Register-Struktur-Übersicht	3
1.2	Die allgemeinen Register	4
1.2.1	Daten-Register	4
1.2.2	Pointer- und Index-Register	5
1.3	Segment-Register	6
1.4	Fundamentale Datentypen	8
1.4.1	Bytes und Words im Speicher	9
1.5	Zusammenhang zwischen Datentypen und Register	10
1.6	Segmentierung	11
1.6.1	Die Teile eines Programms: Code, Daten und Stack	11
1.6.2	Adressierung ab der Basis-Position	13
1.6.3	Adressierung durch 8086/80186/80286	15
1.6.4	Basis-Positionen und physikalische Adressen	19
2.	Segmentierung aus der Sicht des Assembler-Programmierers	25
2.1	Die SEGMENT- und ENDS-Statements	27
2.1.1	Syntax der SEGMENT- und ENDS-Statements	27
2.2	Laden der Segment-Register	29
2.2.1	Stack-Operationen	30
2.2.1.1	Die LABEL-Direktive	33
2.2.1.2	Beispiel: LABEL-Direktive und Datenzuweisung	36
2.2.1.3	Der OFFSET-Operator	37
2.2.2	Starten eines Programms	38
2.3	Das ASSUME-Statement	43
2.3.1	Programmbeispiel 1: Anwendung der SEGMENT/ENDS- und ASSUME-Statements	45
2.3.2	Programmbeispiel 2: Anwendung der SEGMENTS/ENDS- und ASSUME-Statements	50
2.3.3	Programmbeispiel 3: Anwendung der SEGMENTS/ENDS- und ASSUME-Statements	51
2.3.4	Das Schlüsselwort NOTHING	52

3.	Definition und Initialisierung von Variablen	53
3.1	Datentypen	55
3.1.1	INTEGER (vorzeichenbehaftete Binärzahl)	55
3.1.2	ORDINAL (vorzeichenlose Binärzahl)	56
3.1.3	POINTER	56
3.1.4	STRING	56
3.1.5	ASCII (alphanumerische und Steuerzeichen)	57
3.1.6	BCD (Binär codierte Dezimalzahlen)	57
3.1.7	PACKED BCD (gepackte binär codierte Dezimalzahlen)	57
3.1.8	18-DIGIT PACKED BCD	58
3.1.9	FLOATING POINT (Gleitpunktzahlen)	58
3.2	Datenzuweisung	59
3.2.1	Hinweise zur Darstellung von Initialisierungswerten	61
3.2.2	Initialisierungsbeispiele mit Übersetzungsprotokoll	63
3.2.3	Initialisierung mit Wiederholungswerten	64
3.2.4	Zusammenfassung	66
4.	Adressierung von Variablen	69
4.1	Anwendung der Adressierungsarten	76
4.1.1	Direkt Offset	76
4.1.2	Register-Indirekt	77
4.1.2	Register Indirekt	78
4.1.3	Indexed	80
4.1.4	Based	82
4.1.4.1	Strukturen	83
4.1.4.2	Strukturzuweisung und Initialisierung	85
4.1.4.3	Adressierung von Strukturen	90
4.1.4.4	Direkt Offset	95
4.1.5	Based-Indexed mit Displacement	96
4.1.6	Based Indexed	99
4.2	Zusammenfassung der Adressierungsarten	103
4.3	Register-Operanden	104
4.4	Immediate Operanden	107
4.4.1	Übersicht über Assembler-Operatoren	110
4.4.2	Equates	111
4.4.2.1	Vorwärts-Referenzen	114
4.5	Operandentyp und Generierung von Opcodes	116
4.6	Attribut-Overrides	119
4.6.1	Segment Overrides	120

4.6.1.1	Zusammenfassung	125
4.6.2	Typ-Overrides	126
4.6.2.1	Zusammenfassung	129
4.7	Der TYPE-Operator	133
5.	Modulare Programmierung	137
5.1	Kombinieren logischer Segmente	143
5.1.1	Das Combine-Typ-Attribut	146
5.1.2	Das Align-Typ-Attribut	151
5.1.3	Das Class-Name Attribut	155
5.1.4	Die PUBLIC- und EXTRN-Direktiven	157
5.1.4.1	Die Platzierung von EXTRNs	159
5.1.4.2	Dummy-Segmente	161
5.1.4.3	Der SEG-Operator	163
5.1.5	GROUPS	166
6.	Prozeduren	173
6.1	Übergabe von Parametern an eine Prozedur	177
6.2	Rückgewinnung eines Wertes von der Prozedur	179
6.3	Programmbeispiel	180
6.4	HLL-Methoden (High Level Language) der Parameter-Übergabe	182
6.5	Rückgewinnung der Parameter vom Stack	190
6.6	Auswahlkriterien der Methode bei Parameterzugriffen	195
6.7	Definition von Symbolen für Parameter im Stack	195
6.8	Anwendungsbeispiel 1	197
6.8.1	Der SIZE-Operator	198
6.8.2	Der LENGTH-Operator	199
6.9	Anwendungsbeispiel 2	203
6.10	Das BP-Register als Stack-Frame-Pointer zur Adressierung lokaler Variablen	205
6.11	Anwendungsbeispiel 3	206
7.	Benutzung der JMP- und CALL-Befehle	211
7.1	Unbedingte Sprünge im Bereich von -128 bis +127 Byte	213
7.1.1	Der SHORT-Operator	215
7.2	Bedingte Sprünge außerhalb des Bereiches von -128 bis +127 Bytes	216

7.2.1	Die gerichtete-Sprung-Methode	216
7.2.2	Die Jump-around-Jump-Methode	217
7.3	JMP- und CALL-Operanden (direkt)	218
7.4	JMP- und CALL-Operanden (indirekt)	220
7.5	Labels als Operanden	226
7.5.1	Adressierbarkeit von Labels	227
7.5.2	Beispiel 1: NEAR JMP/CALL-Befehle	228
7.5.3	Beispiel 2: FAR JMP/CALL-Befehle	229
7.5.4	Typ-Overrides für Labels	230
8.	Kombinieren von Assembler- und C-Modulen	235
8.1	Das SMALL-Modell	237
8.1.1	CGROUP und DGROUP	238
8.1.2	Register-Initialisierung und Offsets im Small-Modell	240
8.1.3	Das Prozedur-Interface beim SMALL-Modell	242
8.1.4	Erweiterung des SMALL-Modells	242
8.1.5	Programmierbeispiel: Modulkombination im SMALL-Modell	243
8.2	Das LARGE-Modell	257
8.2.1	Programmierbeispiel: Modul-Kombination im LARGE-Modell	260
8.3	Das COMPACT-Modell	271
8.3.1	Stack-Layout nach der Parameter-Übergabe	273
8.4	Das MEDIUM-Modell	274
8.4.1	Stack-Layout nach der Parameter-Übergabe	277
9.	Benutzung der Numerik-Bibliotheken CEL87 und CEL287	279
9.1	Übersicht	281
9.2	Deklaration von CEL87- und CEL287-Prozeduren in Assembler-Programmen	284
9.3	Stackbenutzung der CEL87- und CEL287-Prozeduren	285
9.4	Register-Benutzung der CEL87- und CEL287-Prozeduren	285
9.5	Benutzung elementarer Numerik-Funktionen	285
9.5.1	Die Arccus-Cosinus-Funktion $y = \arccos x$	286
9.5.2	Die CEL87/287-Funktion mgerACS	292
9.5.3	Programmierbeispiel: Berechnung eines Winkels im rechtwinkligen Dreieck mit CEL87.LIB in Assembler	293
9.5.4	Benutzung von CEL87 in C	298

9.5.5	Programmierbeispiel: Berechnung eines Winkels im rechtwinkligen Dreieck mit der CEL87.LIB in C	300
10.	Absoluter Code – AT- und ORG-Direktiven	305
10.1	Die AT-Ausdruck-Direktive:	307
10.2	Die ORG-Direktive:	307
11.	RECORDs	309
11.1	Die RECORD-Direktive	312
11.2	RECORD-Zuweisung und -Initialisierung	313
11.3	Record-Operatoren	315
11.3.1	Der MASK-Operator	316
11.3.2	Der Shift-Count-Operator	317
12.	Einführung in die Intel-Makro-Prozessor-Sprache	321
12.1	Definition eines Macros ohne Parameter	324
12.2	Aufruf eines Macros ohne Parameter	325
12.3	Definition eines Macros mit Parameter	326
12.4	Aufruf eines Macros mit Parametern	327
12.5	Lokale Symbole in Macros	329
13.	Die wichtigsten Unterschiede zwischen ASM286 und ASM86	
13.1	Die GROUP-Direktive	335
13.2	Die ASSUME-Direktive	335
13.3	Segment Attribute	335
13.4	Das Stack-Segment	338
13.4.1	Die STACKSEG-Direktive	338
13.4.2	Der STACKSTART-Operator	339
13.5	Die PROC-Direktive	340
13.6	Die END-Direktive	343
14.	Der Befehlssatz	345
14.1	Abkürzungen und Symbole	347
14.2	Das Statusregister	351
14.3	Die Befehle in alphabetischer Reihenfolge	353

