Inhaltsverzeichnis

1 Basis-Programmier-Modell 1

1.1 Speicherorganisation und Segmentierung 1
1.1.1 Das Modell des „flachen“ Adreßraums 2
1.1.2 Das Modell des segmentierten Adreßraums 3
1.2 Daten-Typen 4
1.3 Register 12
1.3.1 Allgemeine Register 13
1.3.2 Segment-Register 13
1.3.3 Implementierung des Stack 15
1.3.3.2 Das Stack-Segment-Register SS 15
1.3.3.2 Das „Stack-Pointer“-Register ESP 16
1.3.3.3 Das „Stack-Frame-Pointer“-Register EBP 16
1.3.4 Das Flag-Register (EFLAG) 18
1.3.5 Das „Instruction-Pointer“EIP 19
1.4. „Real Mode“-Architektur 19
1.4.1 Speicheradressierung im „Real Mode“ 21
1.5 Adressierungsarten 23
1.5.1 Register-Adressierung 23
1.5.2 Unmittelbare Adressierung 23
1.5.3 Speicheradressierung 23
1.5.3.1 Direkt 27
1.5.3.2 Register-Indirekt 27
1.5.3.3 Based 27
1.5.3.4 Indexed 28
1.5.3.5 Scaled Indexed 28
1.5.3.6 Based Indexed 28
1.5.3.7 Based Scaled Indexed 28
1.5.3.8 Based Indexed mit „Displacement“ 29
1.5.3.9 Based Scaled Indexed mit „Displacement“ 29
1.6 Segment-Register-Auswahl 29
1.7 Ausnahmen und Interrupts 31
1.7.1 Divisionsfehler 31
1.7.2 Einzelschritt 31
1.7.3 Breakpoint 32
1.7.4 Overflow 33
1.7.5 Feldgrenze 33
1.7.6 Undefinierter Operations-Code 33
1.7.7 Prozessor-Erweiterung nicht vorhanden 33
1.7.8 Fehler der Prozessor-Erweiterung 33

1.8 Physikalische Abbildung logischer Adressen 34
1.9 Segment-Längen bei 32-Bit und 16-Bit-Adressierung 35
1.9.1 USE32-Segment 36
1.9.2 USE16-Segment 37
1.10 Das „Paging“-Konzept 38
1.11 Befehlsformat 44
1.11.1 Befehls-Prefix 45
1.12.2 „Address Size Prefix“ 45
1.12.3 „Operand Size Prefix“ 45
1.11.4 „Segment Override Prefix“ 46
1.11.5 Operations-Code 47
1.11.6 ModRM-Byte (Modus Register/Memory) 49
1.11.7 Das „Displacement“ 52
1.11.8 Die hexadezimalen Codierungen der 16-Bit-Adressierungsarten mit ModRM 55
1.11.9 Die hexadezimalen Codierungen der 32-Bit-Adressierungsarten mit ModRM 56
1.11.10 Das SIB-Byte 57
1.11.11 Die hexadezimalen Codierungen der 32-Bit-Adressierungsarten mit SIB 60
1.11.12 Unmittelbarer Operand 61

2 Memory Management 63

2.1 Memory Management 63
2.2 Virtuelle Adressen 63
2.3 Segmente und Segment-Deskriptoren 66
2.3.1 Das Granularity-Bit G 68
2.3.2 Das „Default Operation Size“-Bit D 72
2.4 Deskriptor-Tabellen 73
2.5 Privileg-Ebenen 80
2.6 Übersetzung von virtuellen in physikalische Adressen 82
2.7 Segment-Register im „Protected Mode“ 83
2.8 Deskriptoren-Typen 87
2.9 Alias-Deskriptoren 87
2.9.1 Programmierbeispiel: EPROM-Tabellen in RAM kopieren 91
2.10 Initialisierung der System-Segment-Register GDTR und IDTR 93
2.10.1 Programmierbeispiel: GDTR und IDTR laden 95
2.10.2 Programmierbeispiel: RAM-Tabellen dynamisch erweitern 97
2.11 Initialisierung des System-Segment-Registers LDTR 99
2.11.1 Programmierbeispiel: Laden des LDTR-Registers im „Protected Mode“ 105
2.12 RAM-Tabellen als Daten-Segmente 106

2.12.1 Programmierbeispiel: Zugriff auf Alias-Deskriptoren 108
2.12.2 Programmierbeispiel: Alias-Segment 112
2.13 Starten eines 80486-Programms im „Real Adress Mode“ 114
2.14 Der „Paging“-Mechanismus 115
2.14.1 Das „Page Directory“-Basis-Register 117
2.14.2 Programmierbeispiel: Laden der Page-Directory-Basis-Adresse und Einschalten des Paging-Mechanismus 118
2.14.3 Adressierung der „Page-Directory“-Eintragungen 119
2.14.4 Adressierung der „Page-Tables“ 120
2.14.5 Adressierung der „Page-Table“-Eintragungen 122
2.14.6 Adressierung der Anwender-„Pages“ 123
2.14.7 Adressierung der Anwender-Page-Eintragung 125
2.14.8 Kombination der Segment- und „Paging“-Übersetzungsmechanismen 126
2.14.9 Ein Segment-Deskriptor je „Page-Table“ 127
2.14.10 Segment-Deskriptoren und flacher Adreßraum 129
2.14.11 Ein Segment umspannt mehrere Pages 129
2.14.12 Eine Page umspannt mehrere Segmente 132
2.15 Das Cache-Speichersystem 133
2.15.1 Einleitung 133
2.15.2 Cache-Organisation 136
2.15.2.1 Direct-Mapped-Cache 136
2.15.2.2 Assoziativer Zweiweg-Cache 141
2.15.2.3 Assoziativer Vierweg-Cache 143
2.15.3 Speicher-Schreiboperationen 147
2.15.3.1 Write-Through-Methode 148
2.15.3.2 Write-Back-Methode 149
2.15.3.3 Buffered Write-Through-Methode 150
2.15.4 Das Cache in Multiprozessorsystemen 154
2.15.4.1 Grundlagen 154
2.15.4.2 Snooping 160
2.15.4.3 Semaphore 163
2.15.5 Speicherbereiche ausklammern 172
2.15.6 OnChip-Cache-Management 177
2.15.7 Testen des Cache 180
2.15.7.1 Testregister 181
2.15.7.2 Das Cache-Daten-Testregister TR3 181
2.15.7.3 Cache-Status-Testregister TR4 182
2.15.7.4 Cache-Control-Register TR5 182
2.15.7.5 Cache-Testoperationen 183
2.15.8 Nicht-burstbare, nicht-cachebare „2-2“-Buszyklen 190
2.15.9 Cachebare Burstzyklen 191
2.15.10 Der „Translation Lookaside Buffer“ 194
2.15.10.1 Die Verantwortlichkeit des Betriebssystems 196
2.15.11 Testen des „Translation Lookaside Buffers“ 197

2.15.11.1 Test-Register 197
2.15.11.2 Das TLB-Kommando-Testregister TR6 198
2.15.11.3 Das TLB-Daten-Testregister TR7 199
2.15.11.4 TLB-Testoperationen 201

3 Schutztypen 205

3.1 Schutztypen 205
3.2 Schutz-Implementierung 205
3.3 Memory Management und Schutz 207
3.4 Privilegebenen und Schutz 212
3.5 Struktur des Software-Systems 216
3.6 Daten-Segment/Executable-Segment-Deskriptoren 219
3.6.1 Gemeinsamkeiten der „Access-Bytes“ 220
3.7 Das „Access Byte“ des Executable-Segment-Deskriptors 221
3.7.1 Das Readable-Bit (R) 221
3.7.2 Das Conforming-Bit (C) 221
3.8 Das „Access Byte“ des Daten-Segment-Deskriptors 223
3.8.1 Das Writable-Bit (W) 223
3.8.2 Das Expansion-Direction-Bit (ED) 224
3.9 Das Flag-Feld im Daten- und Executable-Segment-Deskriptor 225
3.9.1 Das Big-Bit (B) 226
3.10 Limit-Überprüfung 229
3.11 Ablaufsteuerung eines Programms ohne Privileg-Wechsel 231
3.11.1 Privileg-Ebene einer Task 231
3.11.2 Programmierbeispiel: Intra-Level-Steuerung 235
3.11.3 Requestor's Privilege Level (RPL) 236
3.12 Ablaufsteuerung mit Privileg-Wechsel 238
3.12.1 Der „Call Gate“-Deskriptor 239
3.12.2 Programmierbeispiel: Privileg-Wechsel über einen „Call Gate“-Deskriptor 241
3.12.3 „Call Gate“-Deskriptor Privilege-Level (DPL) 242
3.12.4 Regeln für „Interlevel“-Programmabläufe 243
3.13 Interlevel-Rückkehr 245
3.14 „Interlevel“-Prozeduraufruf mit Parameterübergabe 247
3.14.1 Das „DWORD COUNT“-Feld 247
3.14.2 Programmierbeispiel: Parameterübergabe 247
3.15 JMP-Befehl und „Call Gate“-Deskriptor 251
3.16 Daten-Zugriffe 252
3.16.1 Zugriff auf Daten im Code-Segment 254
3.17 Parameter-Gültigkeit 255
3.17.1 Programmierbeispiel: Privileg-Verletzung 256
3.17.2 Anwendung des ARPL-Befehls 259
3.17.3 Programmierbeispiel: Privileg-Anpassung 262

3.18 Genaue Überprüfung von Pointer-Parametern 264
3.18.1 Anwendung der Befehle VERR (Verify a Segment for Reading) 265
3.18.2 Programmierbeispiel: Zugriffs-Überprüfung 268
3.18.3 Anwendung der Befehle LAR und LSL 270
3.18.4 Programmierbeispiel: Zugriffs- und Limitüberprüfung 276
3.19 Kombination von Segment- und Page-Schutz 277
3.19.1 Page-orientierter virtueller Speicher 281
3.19.2 Supervisor-Zugriffsrechte 283
3.19.3 User-Zugriffsrechte 286
3.19.4 Zugriff auf Stack-Daten 287
3.19.5 Redundanter Segment- und Page-Schutz 288
3.19.6 Kombination von „Page"- und „Page Table"-Schutz-Attribute 290
3.19.7 Behandlung von Page-Fehlern 292
3.19.8 Ein- und Auslagern von Pages 296
3.19.9 Programmierbeispiel: Zugriff auf Page-Table und Page-Deskriptoren und deren Manipulation bei Page-Fehlern 298
3.19.10 Auswahlkriterien für Segment- oder Page-Schutz 302

4 „Tasks" und Status-Wechsel 307

4.1 Der Task-Status 308
4.2 Task-Status-Segment-Deskriptoren 310
4.3 Initialisierung des Task-Registers 311
4.3.1 Programmierbeispiel: Task-Register laden 312
4.4 Starten einer Task 314
4.5 Task-LDT-Selektor und -Deskriptor 315
4.6 Gemeinsamer linearer Adreßraum für mehrere Tasks 316
4.7 Privileg-Wechsel und Stack-Segment-Anfangspositionen 319
4.8 Task-Wechsel 322
4.9 Verkettung von Tasks 327
4.9.1 Task-Wechsel-Effekt durch JMP, CALL 329
4.10 Task-Gates 330
4.10.1 Die Felder TSS-Selector, TYPE-Feld, das Bit (Present) 330
4.11 Starten der ersten Task („Urtask") 333
4.11.1 Programmierbeispiel: „Urtask" 335
4.12 Task-Status-Alias-Segmente und -Deskriptoren 341
4.13 Die I/O Permission Bit Map 342
4.13.1 Adressierung individueller Bits 346
4.13.2 Anwendung der ASM386/486-STRUC- und DBIT-Direktiven 348
4.13.3 Anwendung der ASM386/486-OFFSET- und BITOFFSET-Operatoren 349
4.13.4 Programmierbeispiel: Bit-Manipulation in der „I/O-Permission Bit Map" 350
4.14 Multitasking in page-orientierten Systemen 355

5 Interrupts 361

5.1 Interrupt-Deskriptor-Tabelle IDT 361
5.2 Interrupt-Task 362
5.3 Interrupt-Prozedur 364
5.4 Interrupt Gate-und Trap Gate-Deskriptoren 365
5.5 Interrupt und Gate-DPL 365
5.6 Unterschied zwischen Interrupt-Deskriptoren und Trap-Gate-Deskriptoren 371
5.7 Task-Wechsel-Effekt bei Interrupts 372
5.8 „Conforming"-Interrupt-Prozeduren 372
5.9 Stack-Eintragungen nach Interrupt 373
5.10 Verlassen des „Interrupt-Handlers" 374
5.11 Task-Wechsel-Effekt bei IRET 376
5.12 Interrupt-Rückkehr innerhalb der gleichen Task 377
5.13 Externe Interrupts 379
5.13.1 Programmierbeispiel: Interrupt-Vektor-Nummern 381
5.14 Nicht-maskierbare externe Interrupts 382

6 Ausnahmen und Interrupts 383

6.1 Interaktionen zwischen Interrupt-Anforderungen und „Gate"-Deskriptoren 384
6.2 Der Fehler-Code 384
6.3 Ausnahmen uned Reservierte Vektoren 385
6.4 Ausnahme-Bedingungen und Fehlerbehandlung 338
6.4.1 Interrupt 0: Divisions-Fehler 387
6.4.2 Interrupt 1: Debug 387
6.4.3 Interrupt 3: Breakpoint 387
6.4.4 Interrupt 4: Overflow 387
6.4.5 Interrupt 5: Feldgrenze überschritten 388
6.4.6 Interrupt 6: Undefinierter Operation-Code 388
6.4.7 Interrupt 7: Prozessor-Erweiterung nicht vorhanden 389
6.4.8 Interrupt 8: Doppelfehler 389
6.4.9 Interrupt 10: Ungültiges Task-Status-Segment 390
6.4.10 Interrupt 11: Segment nicht vorhanden 391
6.4.11 Interrupt 12: Stack-Segment-Unter-/Überlauf oder „nicht vorhanden" 392
6.4.12 Interrupt 13: Allgemeine Schutzverletzung 392
6.4.13 Interrupt 14: Page-Fehler 395
6.4.13.1 Page-Fehler bei Task-Wechsel 397
6.4.13.2 Page-Fehler mit inkonsistentem Stack-Pointer 398
6.4.14 Interrupt 16: Fehler der Prozessor-Erweiterung 398
6.4.15 Interrupt 17: Alignment-Überprüfung 399

7 Eingabe/Ausgabe 401

7.1 Bus-Transfer-Mechanismus 401
7.1.1 Speicher-Selektion 403
7.1.2 I/O-Selektion 403
7.1.3 32-Bit-Datenbus-Transfer und Ausrichtung der Operanden 407
7.1.4 Alignment-Überprüfung 409
7.2 Ein-/Ausgabe und Schutz 412
7.2.1 I/O-Privileg und „I/O-Permission Bit Map“ 413
7.2.2 Änderung der I/O-Privilegebene 416
7.3 Programmierbeispiel: IOPL-Anpassung 417
7.4 Interrupt-Flag und IOPL 419
7.5 Sperren von Segmenten und Pages bei direkten I/O-Operationen 420

8 Floating-Point-Einheit 421

8.1 Die Numerik-Register 421
8.1.1 FPU-Register-Stack 421
8.1.2 FPU-Statuswort 423
8.1.2.1 Liste der Exceptions 428
8.1.3 FPU-Kontrollwort 431
8.1.4 Das FPU-Tagwort 435
8.1.5 Befehls- und Operanden-Pointer 436
8.2 Datentypen und Formate 438
8.2.1 Binär Integer 438
8.2.1.1 Binär Integer Indefinite 440
8.2.2 Packed Decimal Integer 440
8.2.2.1 Packed Decimal Integer Indefinite 442
8.2.3 Real-Zahlen 442
8.3 Spezielle Numerikzahlen 447
8.3.1 Denormal-Realzahlen 448
8.3.2 NaN (Not-a-Number) 448
8.3.2.1 Signaling NaNs 450
8.3.2.2 Quiet NaNs 450
8.3.3 Infinity (Unendlich) 451
8.3.4 Nicht-unterstützte Datenformate 452
8.4 Zahlenbereiche der unterstützten Datenformate 453
8.4.1 Zahlenbereich der SHORT-Reals 453
8.4.2 Zahlenbereich der LONG-Reals 454
8.4.3 Zahlenbereich der TEMPORARY-Reals 455
8.5 Spezielle Berechnungs-Situationen 456
8.5.1 Ungültige arithmetische Operationen 457
8.5.2 Die Exception Overflow 458

8.5.2.1 Maskierter Overflow 458
8.5.2.2 Nicht-maskierter Overflow 458
8.5.3 Die Exception Underflow 459
8.5.3.1 Maskierter Underflow 459
8.5.3.2 Nicht-maskierter Underflow 459
8.5.4 Die Exception Precision 460
8.5.5 Laden von Denormals 460
8.5.6 Interpretation der Condition-Codes 461
8.5.6.1 Condition-Codes nach Examine 462
8.5.6.2 Condition-Codes nach Compare und Test 465
8.5.6.3 Condition-Codes nach FPREM und FPREM1 (Partial Remainder) 467
8.5.7 Behandlung infiniter Operanden 472
8.5.8 Behandlung von Null-Operanden 474
8.5.9 Rangordnung der Exceptions 476
8.5.10 Initialisierung der FPU 477
8.5.11 Synchronisation von Exceptions 478
8.6 Numerik-Umgebung konfigurieren 480
8.6.1 Prozessor Extension-Bit ET 480
8.6.2 Emulate Coprozessor-Bit EM 480
8.6.3 Numerik Exception-Bit EN 481
8.6.4 Monitior Coprozessor-Bit MP 481
8.7 Numerik-Task-Status 481
8.7.1 FPU-Context-Wechsel 483
8.7.1.1 Programmierbeispiel: Context-Wechsel 484

9 System Builder BLD386/486 491

9.1 Entwurf eines statisches Systems 491
9.2 Die BLD386/486-Spezifikationen 492
9.2.1 Build-Programm 494
9.2.2 SEGMENT-Definition 494
9.2.2.1 Beispiele für Segment-Definitionen 497
9.2.3 TABELLEN-Definition 498
9.2.3.1 Beispiele für Tabellen-Definitionen 501
9.2.4 Task-Definition 502
9.2.4.1 Programmierbeispiel: System-Komposition für das ASM386/486-Programm – Urtask 506
9.2.5 Gate-Definition 510
9.2.5.1 Programmierbeispiel: System-Komposition für ein ASM386/486-Programm mit Privileg- und Task-Wechsel über Gate-Deskriptoren 514

9.2.6 Alias-Definition 525
9.2.6.1 Programmierbeispiel: Alias-Task-Status-Segment 525
9.2.7 Memory-Definition 533
9.2.8 Paging-Definition 535
9.2.8.1 Programmierbeispiel: Page-orientierter Speicher 542

10 Virtueller 8086-Modus 551

10.1 Eintritt in den virtuellen 8086-Modus 551
10.2 Verlassen des virtuellen 8086-Modus 553
10.3 Task-Management 555
10.4 Der virtuelle Maschinen-Monitor 557
10.5 System-Aufrufe 558
10.6 Schutz innerhalb einer V86-Task 559
10.7 Paging für 8086-Tasks 559
10.8 Virtualisierung des Interrupt-Enable-Flags IF 563
10.9 Eingabe/Ausgabe 564

11 DEBUG-Unterstützung 567

11.1 Die Breakpoint-Register DR0..DR5 567
11.2 Das Debug-Steuerregister DR7 568
11.2.1 RW0 ... RW (Abbruch-Ereignis) 568
11.2.2 LENO ... LEN3 (Länge des Breakpoint-Feldes) 568
11.2.3 G0 ... G3 und L0 ... L3 (Breakpoint-Freigabe, global und lokal) 571
11.2.4 GE und LE (Exakter Daten-Breakpoint, global und lokal) 572
11.2.5 GD (Zugriff auf Debug-Register steuern) 572
11.3 Das Debug-Status-Register DR6 573
11.3.1 B0 ... B3 (Debug-Fault/Trap wegen Breakpoints 0 ... 3) 573
11.3.2 BD (Debug-Fault, wenn GD gesetzt ist) 573
11.3.2 BS (Debug Trap wegen Single-Step) 573
11.3.4 BT (Debug-Trap wegen Task-Switch) 574
11.4 Benützung des Resume-Flag RF 574
11.5 Erkennen von Debug-Bedingungen 575
11.6 Zugriff auf Debug-Register 575
11.7 Programmierbeispiel: Definition von Abbruch-Bedingungen 576