



```
;al;  
e;  
d(GetFirmwareEnvironmentVariable) then  
f(pBuffer);  
EnvironmentVariable(  
cureBoot'),  
be4df61-93ca-11d2-aa0d-00e098032b8c}'),  
0 then  
ffer =
```

Devid Espenschied

# Systemprogrammierung mit Delphi

Hardwarezugriff über Windows-API und  
Windows-Kernelmodus-Treiber

---

# Systemprogrammierung mit Delphi

---

Devid Espenschied

# Systemprogrammierung mit Delphi

Hardwarezugriff über Windows-API und  
Windows-Kernelmodus-Treiber

Devid Espenschied  
Berlin, Deutschland

ISBN 978-3-658-43454-0      ISBN 978-3-658-43455-7 (eBook)  
<https://doi.org/10.1007/978-3-658-43455-7>

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <https://portal.dnb.de> abrufbar.

© Der/die Herausgeber bzw. der/die Autor(en), exklusiv lizenziert an Springer Fachmedien Wiesbaden GmbH, ein Teil von Springer Nature 2024

Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Jede Verwertung, die nicht ausdrücklich vom Urheberrechtsgesetz zugelassen ist, bedarf der vorherigen Zustimmung des Verlags. Das gilt insbesondere für Vervielfältigungen, Bearbeitungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen.

Die Wiedergabe von allgemein beschreibenden Bezeichnungen, Marken, Unternehmensnamen etc. in diesem Werk bedeutet nicht, dass diese frei durch jedermann benutzt werden dürfen. Die Berechtigung zur Benutzung unterliegt, auch ohne gesonderten Hinweis hierzu, den Regeln des Markenrechts. Die Rechte des jeweiligen Zeicheninhabers sind zu beachten.

Der Verlag, die Autoren und die Herausgeber gehen davon aus, dass die Angaben und Informationen in diesem Werk zum Zeitpunkt der Veröffentlichung vollständig und korrekt sind. Weder der Verlag noch die Autoren oder die Herausgeber übernehmen, ausdrücklich oder implizit, Gewähr für den Inhalt des Werkes, etwaige Fehler oder Äußerungen. Der Verlag bleibt im Hinblick auf geografische Zuordnungen und Gebietsbezeichnungen in veröffentlichten Karten und Institutionsadressen neutral.

Planung/Lektorat: David Imgrund

Springer Vieweg ist ein Imprint der eingetragenen Gesellschaft Springer Fachmedien Wiesbaden GmbH und ist ein Teil von Springer Nature.

Die Anschrift der Gesellschaft ist: Abraham-Lincoln-Str. 46, 65189 Wiesbaden, Germany

Das Papier dieses Produkts ist recycelbar.

---

## Vorwort

Liebe Leserinnen und Leser,

der Begriff Systemprogrammierung tauchte bereits vor mehreren Jahrzehnten unter dem Gesamtbegriff der Softwareentwicklung auf und beschreibt eine software- und hardwarenahe Entwicklung, die als systemnah bezeichnet werden kann. Auch wenn diese Thematik viel Wissen über die Hardware und deren Schnittstellen erfordert, wird man früher oder später damit konfrontiert – spätestens dann, wenn eine Windows API-Funktion benötigt wird (von denen wirklich sehr viele existieren). Embarcadero's Delphi bietet hierfür eine gute bis sehr gute API-Integration und liefert entsprechende Object Pascal-Implementierungen mit, die man leicht nutzen kann. Die dazugehörige Dokumentation stammt gleichzeitig von Microsoft's sehr umfangreichen Internet-Dokumentationsseiten.

Meine Motivation für dieses Buch ist tatsächlich die über 30jährige Erfahrung in diesem Bereich (damals noch unter DOS) und die Faszination, Hardware direkt mit Software anzusprechen – auf einer der untersten möglichen Zugriffsebenen. Gleichzeitig gab es den Wunsch, diese Art und Weise der Entwicklung dem Leser über verständliche Didaktik zu vermitteln. Ich habe mich bemüht, einige der häufig als trocken und extrem techniklastig bezeichneten Themenblöcke transparent mit flüssigem Schreibstil darzustellen und hoffe, dass mit diesem Wissen und der gleichzeitig entwickelten Beispielapplikation *PC Analyser* eine praxisnahe Umsetzung als Bereicherung für eigene Aufgaben und Projekte ermöglicht wird.

Trotzdem ist mir natürlich bewusst, dass der IT-Markt rasant fortschreitet und die mit diesem Buch aktuelle Speichertechnologie DDR5 bspw. bald durch eine neue Generation ersetzt wird. Deswegen könnte ich einer möglichen Neuauflage in ein paar Jahren sehr wahrscheinlich nicht abgeneigt sein.

Einen großen Dank möchte ich an Matthias Eißing von Embarcadero Deutschland richten, der mir viele Fragen beantwortet und mit seinem Wissen dieses Buch abgerundet hat, sowie eine unterstützende Hilfe während des Schreibprozesses war. Leider hat er die Veröffentlichung nicht mehr erleben dürfen, da er am 14. Februar 2024 plötzlich und unerwartet im Alter von 53 Jahren von uns gegangen ist. Herr Eißing war seit 1997 für Borland/Inprise/CodeGear/Embarcadero tätig und hat Delphi seit Version 1 begleitet.

Dass sein Todestag auf den Valentinstag und damit auf den 29. Geburtstag von Delphi fiel, ist bezeichnend, denn sein Wissen über Delphi hat ihm in der Delphi-Community den Status einer Legende eingebracht.

Für dieses Buch sind Ihre Meinung und Feedback sehr wichtig, weil es mir hilft, neue Auflagen entsprechend noch kundenorientierter zu verfassen. Für eine Kommunikation schreiben Sie mir daher gerne an meine E-Mail-Adresse:

[d.espe@gmx.de](mailto:d.espe@gmx.de)

und besuchen Sie auch gerne meine Homepage, auf der ich weitere Details zum Buch zusammengestellt habe:

[www.pcanalyser.de](http://www.pcanalyser.de)

Den Quellcode für die begleitende Beispielapplikation ist in meinem GitHub-Repository verfügbar:

<https://github.com/DevidEspe/PCAnalyser>

Und abschließend folgen einige wenige und wahrscheinlich die einzigen emotionalen Worte in diesem Buch. Dieses Buch widme ich meiner Tochter Catharina als Dank für alles, was ich mit ihr bisher erleben durfte und mich noch möglichst viele Jahre meines Lebens begleiten wird. Du bedeutest mir alles.

Berlin  
im Juni 2024

Devid Espenschied

---

# Inhaltsverzeichnis

## Teil I Grundsätzliches

<b>1</b>	<b>Einführung in die Systemprogrammierung</b>	<b>3</b>
1.1	Was ist Systemprogrammierung?	3
1.2	Historie	3
1.3	Nutzen & Notwendigkeit	4
1.4	Herangehensweise & Windows-Architektur	4
1.5	Warum Delphi?	5
1.6	Voraussetzungen	6
1.7	Datenblätter, Spezifikationen und weitere Quellen	6
1.8	Zusammenfassung	7
<b>2</b>	<b>Erstellung einer begleitenden Beispielapplikation</b>	<b>9</b>
2.1	Überblick, Struktur und Konzeption	9
2.2	Syntaktische Elemente	9
2.3	Erstellung der Klasse TSystemAccess für den Hardwarezugriff	10
2.3.1	Erstellung von zusätzlichen Klassen für die Hauptklasse	11
2.4	Oberflächengestaltung	15
2.4.1	Programmsteuerungskomponente (obere GroupBox)	16
2.4.2	Systemzugriffskomponente (mittlere GroupBox)	21
2.4.3	Programmprotokollkomponente (untere GroupBox)	23
2.5	Benutzerrechte oder Administratorrechte	24
2.5.1	Rechte-Steuerung über Anwendungsmanifest	26
2.6	Programmnavigation	29
2.7	Zusammenfassung	35
	Literatur	36

## Teil II Systemprogrammierung mit Delphi

<b>3</b>	<b>Maschinen-Details über die API</b>	<b>39</b>
3.1	Maschinenname	40

3.2	Benutzername .....	42
3.3	Administrator .....	43
3.4	Benutzerkontext .....	52
3.5	Letzter Startvorgang .....	54
3.6	Systembetriebszeit .....	54
3.7	Caps/Num/Scroll Lock .....	55
3.8	Zusammenfassung .....	58
	Literatur .....	58
<b>4</b>	<b>Windows-Details und installierte Software über die API .....</b>	<b>61</b>
4.1	Windows-Details .....	63
4.1.1	Windows-Name .....	65
4.1.2	Windows-Version .....	74
4.1.3	Service Pack .....	77
4.1.4	Codename .....	78
4.1.5	Kompatibilitätsmodus .....	79
4.1.6	Installationszeitpunkt .....	82
4.1.7	Windows- und Systemverzeichnis .....	84
4.2	Installierte Software .....	86
4.3	Windows-Verzeichnisse .....	101
4.4	Umgebungsvariablen .....	110
4.5	Zusammenfassung .....	116
	Literatur .....	116
<b>5</b>	<b>System-Details über System Management BIOS .....</b>	<b>119</b>
5.1	Spezifikation .....	120
5.2	Datenerhebung damals und heute .....	122
5.2.1	Interpretation der Rohdaten .....	125
5.3	Entwicklung einer Basisklasse .....	128
5.3.1	Basisklassenfunktionen .....	133
5.3.2	Kernladefunktionen .....	134
5.3.3	Allgemeine Hilfsfunktionen .....	147
5.3.4	Binär- und Texthilfsfunktionen .....	154
5.3.5	Veröffentlichte Eigenschaften .....	158
5.3.6	Hauptauswertung der Rohdaten .....	159
5.4	Entwicklung einer Hilfsklasse .....	159
5.4.1	Tabellendefinitionen .....	160
5.4.2	Tabellenrecords .....	163
5.4.3	Listenfunktionen .....	171
5.5	Ermittlung der SMBIOS-Details .....	204
5.5.1	BIOS-Details .....	206
5.5.2	System-Identifikation .....	214



5.5.3	Hauptplatine . . . . .	217
5.5.4	Gehäuse/Chassis . . . . .	220
5.5.5	Prozessor(en) . . . . .	226
5.5.6	Prozessor-Caches . . . . .	236
5.5.7	Anschlüsse . . . . .	242
5.5.8	Steckplätze . . . . .	244
5.5.9	Speicher-Überblick . . . . .	253
5.5.10	Speicher-Modul(e) . . . . .	256
5.5.11	Spannungssensor(en) . . . . .	268
5.5.12	Kühlungssensor(en) . . . . .	271
5.5.13	Temperatursensor(en) . . . . .	273
5.5.14	Stromstärkesensor(en) . . . . .	276
5.5.15	TPM-Gerät(e) . . . . .	279
5.5.16	Weitere Strukturen und Tabellen . . . . .	282
5.6	Darstellung der SMBIOS-Details . . . . .	282
5.7	Zusammenfassung . . . . .	286
	Literatur . . . . .	287
<b>6</b>	<b>Prozessor-Informationen über CPUID . . . . .</b>	<b>289</b>
6.1	Spezifikationen . . . . .	292
6.2	Entwicklung einer Basisklasse . . . . .	293
6.2.1	Basisklassenfunktionen . . . . .	298
6.2.2	Cachefunktionen . . . . .	300
6.2.3	Fähigkeitsfunktionen . . . . .	301
6.2.4	Namensfunktionen . . . . .	301
6.2.5	Zugehörigkeitsfunktionen . . . . .	302
6.2.6	CPUID-Funktionen . . . . .	302
6.2.7	Prozessornummerfunktionen . . . . .	304
6.2.8	Binäre Hilfsfunktionen . . . . .	304
6.2.9	Haupterkennungsfunktionen . . . . .	305
6.2.10	Veröffentlichte Eigenschaften . . . . .	335
6.3	Entwicklung der Hilfsunits . . . . .	336
6.3.1	Prozessor-Datenbank . . . . .	336
6.3.2	Cache und Fähigkeiten . . . . .	341
6.4	Darstellung der Prozessor-Details . . . . .	345
6.4.1	Darstellung der Prozessor-Übersicht . . . . .	345
6.4.2	Darstellung der Cache-Details . . . . .	352
6.4.3	Darstellung der Fähigkeiten-Details . . . . .	354
6.5	Zusammenfassung . . . . .	360
	Literatur . . . . .	360

**Teil III Windows Kernelmodus-Treiber**

<b>7</b>	<b>Entwicklung eines Kernelmodus-Treibers mit Visual Studio</b>	365
7.1	Überblick	365
7.2	Voraussetzungen und Installation.	366
7.2.1	Zusätzliche Installationen	369
7.3	Treiberfunktionen und I/O Control Codes	372
7.4	Erstellung eines neuen Treiber-Projektes.	374
7.5	Erstellung einer Headerdatei	379
7.6	Erstellung der Treiber-Hauptdatei	385
7.6.1	DriverEntry-Routine	387
7.6.2	Create, Close und Unload-Funktionen.	393
7.6.3	DeviceControl-Funktion	394
7.7	Treiber-Kompilierung und Bereitstellung	416
7.8	Signierung, Notwendigkeit, Zertifikate	417
7.8.1	Test-Signierung	419
7.9	Zusammenfassung	421
	Literatur.	421
<b>8</b>	<b>Anbindung des Kernelmodus-Treibers in Delphi</b>	423
8.1	Treiberstatus.	423
8.2	Installieren und Starten des Treibers in Delphi	425
8.2.1	Öffnen des Treibers	431
8.2.2	Generierung von I/O Control Codes	432
8.3	Abbildung der Treiberdefinitionen.	436
8.4	Implementierung der Treiberfunktionen basierend auf I/O Control Codes	440
8.4.1	Treiber-Transfertest	442
8.4.2	Treiber-Version	444
8.4.3	Prozessor Modell-spezifische Register lesen.	445
8.4.4	Prozessor Modell-spezifische Register schreiben	446
8.4.5	Daten vom PCI-Bus lesen	447
8.4.6	Daten auf den PCI-Bus schreiben	449
8.4.7	Speicheradressen lesen.	450
8.4.8	Speicheradressen schreiben	452
8.4.9	Daten von I/O Ports lesen	453
8.4.10	Daten auf I/O Ports schreiben	455
8.5	Stoppen und Entfernen des Treibers in Delphi	456
8.6	Zusammenfassung	460
	Literatur.	461

**Teil IV Hardwarezugriffe über den Kernelmodus-Treiber**

<b>9</b>	<b>Prozessoren</b>	465
9.1	Treiberzugriff auf Modell-spezifische Register (MSR)	465
9.2	Spezifikationen	466
9.3	Erweiterung der Prozessor-Basisklasse	467
9.3.1	Ermittlung des Microcode-Updates	468
9.3.2	Ermittlung der Intel TjMax-Referenztemperatur	469
9.4	Entwicklung der Hilfsunit für MSR-Listen	472
9.5	Darstellung der MSR-Details	474
9.5.1	Darstellung ausgewählter MSRs	474
9.5.2	Darstellung von MSR-Listen	481
9.6	Schreiben von MSRs	484
9.6.1	Intel Turbo Boost Technology	487
9.6.2	AMD Core Performance Boost	491
9.7	Zusammenfassung	496
	Literatur	496
<b>10</b>	<b>PCI-Bus</b>	497
10.1	Ermittlung über Windows-Mechanismen	497
10.2	Ermittlung über Konfigurationsmechanismen	498
10.3	PCI-Konfigurationsbereich	500
10.4	Entwicklung einer Basisklasse	501
10.4.1	Basisklassenfunktionen	504
10.4.2	Kernerkennungsfunktionen	505
10.4.3	PCI-Datenbankfunktionen	526
10.4.4	Binäre Hilfsfunktionen	534
10.4.5	Veröffentlichte Eigenschaften	534
10.5	Darstellung der PCI-Geräte	535
10.6	Zusammenfassung	559
	Literatur	559
<b>11</b>	<b>Arbeitsspeicher und System Management Bus</b>	561
11.1	Historie und Grundlagen	561
11.2	SMBus-Basisadresse als Voraussetzung	563
11.3	Grundkommunikation mit SMBus-Geräten	563
11.4	Entwicklung einer Basisklasse	565
11.4.1	Basisklassenfunktionen	568
11.4.2	SMBus-Kernfunktionen	569
11.4.3	DDR4/DDR5 Seiten-Auswahlfunktionen	578
11.4.4	SMBus Mutex-Funktionen	581

11.4.5	Kernerkennungsfunktionen . . . . .	589
11.4.6	SPD-Größenfunktionen . . . . .	599
11.4.7	SPD-Interpretationsfunktionen . . . . .	621
11.4.8	Binär- und Text-Hilfsfunktionen . . . . .	670
11.4.9	Veröffentlichte Eigenschaften . . . . .	670
11.5	Entwicklung einer Hilfsklasse . . . . .	671
11.6	Darstellung der Speichermodule . . . . .	674
11.7	Zusammenfassung . . . . .	681
	Literatur. . . . .	682
<b>12</b>	<b>Externe Drittanbieter-Komponente für den Systemzugriff . . . . .</b>	<b>685</b>
12.1	MiTeC System Information Component Suite . . . . .	686
12.1.1	Einleitung. . . . .	686
12.1.2	Klassenübersicht . . . . .	686
12.1.3	Enthaltene Demos . . . . .	689
12.1.4	Applikationen. . . . .	692
12.1.5	Lizenzierung . . . . .	694
12.2	HWiNFO SDK. . . . .	696
12.2.1	Einleitung. . . . .	698
12.2.2	Erkennungsbandbreite . . . . .	699
12.2.3	Individualisierung . . . . .	700
12.2.4	Bestandteile . . . . .	701
12.2.5	SDK im Einsatz . . . . .	701
12.2.6	Lizenzierung . . . . .	705
12.3	Zusammenfassung . . . . .	705
	Literatur. . . . .	706
<b>13</b>	<b>Ausblick . . . . .</b>	<b>707</b>
 <b>Teil V Anhang</b>		
<b>14</b>	<b>Beispielprogramm und Quellcodes . . . . .</b>	<b>711</b>
	<b>Stichwortverzeichnis. . . . .</b>	<b>715</b>

---

## Abbildungsverzeichnis

Abb. 2.1	Klassenübersicht der im Verlauf dieses Buches entwickelten Haupt- und Unterklassen, die allesamt auf TObject basieren und per Instanzen zugreifbar gemacht werden (ohne Vererbung im klassischen Sinne) . . . . .	13
Abb. 2.2	Beispielhafte Darstellung der Programminfos . . . . .	17
Abb. 2.3	Beispielhafte Darstellung des Programmkontextes . . . . .	19
Abb. 2.4	Details und Steuerung des Kernelmodus-Treibers . . . . .	20
Abb. 2.5	Exemplarische Darstellung der linken TreeView-Komponente mit den Kategorien und Untergruppen (die Kategorie PCI-Bus enthält eine Auflistung aller PCI-Geräte, die hier aus Gründen der Übersicht eingeklappt wurden) . . . . .	22
Abb. 2.6	Windows-Benutzerkontensteuerung mit unsignierter Applikation. . . . .	27
Abb. 2.7	Windows-Benutzerkontensteuerung mit signierter Applikation. . . . .	27
Abb. 2.8	Projektoptionen für automatisches Anwendungsmanifest . . . . .	28
Abb. 2.9	Beispielhafte Darstellung der gesamten Programmoberfläche mit allen VCL-Komponenten und geladenem Kernelmodus-Treiber . . . . .	34
Abb. 3.1	Beispielhafte Darstellung der Maschinen-Details . . . . .	40
Abb. 4.1	Aktivierung und Optionen des Windows-Kompatibilitätsmodus. . . . .	62
Abb. 4.2	Beispielhafte Darstellung der Windows-Details. . . . .	63
Abb. 4.3	Windows PE Registrierungs-Editor mit dem WinPE-Schlüssel und Versionsdetails . . . . .	67
Abb. 4.4	Versionsangaben bei Windows 11 mit einer internen Version von 10.0 . . . . .	69
Abb. 4.5	Einstellungen für den Kompatibilitätsmodus unter Windows 11 . . . . .	80
Abb. 4.6	Uninstall-Schlüssel aus der Windows-Registrierung für RAD Studio . . . . .	88
Abb. 4.7	Beispielhafte Darstellung der installierten Software . . . . .	99
Abb. 4.8	Exemplarische Darstellung der ermittelten Windows-Verzeichnisse (hier Windows 10 ×64 Pro) . . . . .	109

Abb. 4.9	Die Bearbeitung der Windows-Umgebungsvariablen mit sauberer Trennung zwischen Benutzer- und Systemvariablen. . . .	110
Abb. 4.10	Exemplarische Darstellung der ermittelten Windows-Umgebungsvariablen (hier Windows 10 ×64 Pro). . . . .	115
Abb. 5.1	MiTeC WMI Explorer mit einer WMI-Abfrage der SMBIOS-Rohdaten. . . . .	124
Abb. 5.2	Exemplarische SMBIOS-Auswertung mit MiTeC SMBIOS Explorer . . . . .	126
Abb. 5.3	File Scanner-Funktionalität zur komfortablen Suche nach SMBIOS-Dumps . . . . .	127
Abb. 5.4	Byte-Feld SMBiosData in der WMI-Tabelle MSSmBios_RawSMBiosTables . . . . .	138
Abb. 5.5	Binärwert SMBiosData in der Windows-Registrierung . . . . .	141
Abb. 5.6	Word-Feld „Memory Device – Type Detail“ der SMBIOS-Spezifikation für Tabellentyp 17. . . . .	195
Abb. 5.7	Byte-Feld „Voltage Probe – Location and Status“ der SMBIOS-Spezifikation für Tabellentyp 26 . . . . .	198
Abb. 5.8	Byte-Feld „Cooling Device – Device Type and Status“ der SMBIOS-Spezifikation für Tabellentyp 27 . . . . .	200
Abb. 5.9	Byte-Feld „Temperature Probe – Location and Status“ der SMBIOS-Spezifikation für Tabellentyp 28 . . . . .	201
Abb. 5.10	Byte-Feld „Current Probe – Location and Status“ der SMBIOS-Spezifikation für Tabellentyp 29 . . . . .	203
Abb. 5.11	Mögliche BIOS-Details anhand des Tabellentyps 0. . . . .	207
Abb. 5.12	Mögliche System-Information für die Identifikation anhand des Tabellentyps 1. . . . .	215
Abb. 5.13	Mögliche Hauptplatinen-Details anhand des Tabellentyps 2. . . . .	217
Abb. 5.14	Mögliche Gehäuse- oder Chassis-Details anhand des Tabellentyps 3. . . . .	221
Abb. 5.15	Mögliche Prozessor-Details anhand des Tabellentyps 4. . . . .	227
Abb. 5.16	Mögliche Prozessor-Cache-Details anhand des Tabellentyps 7. . . . .	237
Abb. 5.17	Mögliche Details zum internen/externen Anschluss anhand des Tabellentyps 8. . . . .	243
Abb. 5.18	Mögliche Details zum Steckplatz anhand des Tabellentyps 9 . . . . .	245
Abb. 5.19	Mögliche Details zum gemeinsamen Speicheradressraum anhand des Tabellentyps 16 . . . . .	253
Abb. 5.20	Mögliche Speichermodul-Details anhand des Tabellentyps 17 . . . . .	257
Abb. 5.21	Mögliche Spannungsdaten anhand des Tabellentyps 26 . . . . .	268
Abb. 5.22	Mögliche Kühlungsdaten anhand des Tabellentyps 27 . . . . .	271
Abb. 5.23	Mögliche Temperaturdaten anhand des Tabellentyps 28 . . . . .	273
Abb. 5.24	Mögliche Stromstärken anhand des Tabellentyps 29 . . . . .	276

Abb. 5.25	Mögliche TPM-Details anhand des Tabellentyps 43 . . . . .	279
Abb. 6.1	Prozessor-Details über die Windows-Registrierung. . . . .	291
Abb. 6.2	Prozessor-Details über die Windows Management Instrumentation. . . . .	292
Abb. 6.3	Generische Details im EAX-Register nach dem CPUID-Aufruf mit dem Funktionswert 1 . . . . .	310
Abb. 6.4	Cache-Deskriptoren nach dem CPUID-Aufruf mit dem Funktionswert 2 . . . . .	317
Abb. 6.5	Intel Basis-Fähigkeiten nach dem CPUID-Aufruf mit dem Funktionswert 1 im Register ECX . . . . .	320
Abb. 6.6	Exemplarische Prozessor-Details für einen Intel Core i7-12700H (12. Generation) . . . . .	352
Abb. 6.7	Exemplarische Prozessor-Details für einen AMD Ryzen 7 PRO 5850U mit integrierter Grafik. . . . .	352
Abb. 6.8	Exemplarische Cache-Details für einen Intel Core i7-12700H (12. Generation) . . . . .	355
Abb. 6.9	Exemplarische Cache-Details für einen AMD Ryzen 7 PRO 5850U mit integrierter Grafik. . . . .	355
Abb. 6.10	Exemplarische Fähigkeiten-Details für einen Intel Core i7-12700H (12. Generation) . . . . .	358
Abb. 6.11	Exemplarische Fähigkeiten-Details für einen AMD Ryzen 7 PRO 5850U mit integrierter Grafik . . . . .	359
Abb. 7.1	Visual Studio 2019 Installation (hier für die Community Edition) . . .	367
Abb. 7.2	Windows Software Development Kit Installationsauswahl . . . . .	368
Abb. 7.3	Windows Software Development Kit Featureauswahl . . . . .	368
Abb. 7.4	Windows Driver Kit – Integration in Visual Studio als Installationsabschluss . . . . .	369
Abb. 7.5	VSIX Installer mit Auswahl der installierten Visual Studio Versionen . . . . .	370
Abb. 7.6	Visual Studio Optionen für die Aktivierung des klassischen Projektdialogs. . . . .	371
Abb. 7.7	Assistent für neues Projekt in der neuen Ansicht (ab VS 2019). . . . .	371
Abb. 7.8	Assistent für neues Projekt in der altbekannten Ansicht (bis vor VS 2019). . . . .	372
Abb. 7.9	Aufteilung von I/O Control Codes . . . . .	373
Abb. 7.10	Neues Treiber-Projekt erstellen und konfigurieren. . . . .	377
Abb. 7.11	Mehrere Plattformen mit Win32 und x64 als Auswahl . . . . .	378
Abb. 7.12	Treibersignierungsoptionen in den Projekt-Eigenschaften von Visual Studio . . . . .	419
Abb. 7.13	Testmodus-Benachrichtigung auf dem rechten unteren Desktop (hier Win 10 Pro) . . . . .	420

Abb. 9.1	Exemplarische MSR-Details für einen Intel Core i7–12700 H (12. Generation) . . . . .	485
Abb. 9.2	Exemplarische MSR-Details für einen AMD Ryzen 7 PRO 5850U mit integrierter Grafik . . . . .	486
Abb. 10.1	PCI-Details aus der Windows-Registrierung innerhalb der Struktur HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\PCI . . . . .	498
Abb. 10.2	Intel C620 PCH Datenblatt mit der Zuordnung der Device ID A223h und dem SMBus-Kontroller . . . . .	520
Abb. 10.3	Intel C620 PCH Datenblatt mit den bereitgestellten Registern und Offset 20h-23h für die SMBus-Basisadresse . . . . .	520
Abb. 10.4	Intel C620 PCH Datenblatt mit dem SMB Base Address (SBA) Registerinhalt ab Offset 20h . . . . .	521
Abb. 10.5	Mögliche Ressourcenoptionen für die Integration der 3 PCI-Datenbanken . . . . .	530
Abb. 10.6	Mögliche Darstellung der PCI-Geräte im übergeordneten PCI-Bus-Knoten . . . . .	537
Abb. 10.7	Exemplarische Darstellung des PCI-Bus-Überblicks inkl. Statistik . . . . .	543
Abb. 10.8	Exemplarische Geräte-Identifikation anhand eines Intel-Gerätes innerhalb eines Dell-Notebooks . . . . .	547
Abb. 10.9	Exemplarische Zusatzdetails eines Intel PCI-Gerätes . . . . .	548
Abb. 10.10	Exemplarische Darstellung der Geräte-Typdaten eines Intel PCI-Gerätes . . . . .	549
Abb. 10.11	Exemplarische Darstellung der Geräte-Kontrolle eines Intel-Gerätes . . . . .	551
Abb. 10.12	Exemplarische Darstellung des Geräte-Status eines Intel-Gerätes . . . . .	554
Abb. 10.13	Exemplarische Darstellung des Geräte-Dumps eines Intel-Gerätes . . . . .	558
Abb. 11.1	Gängiges DDR4-Speichermodule mit einem mittig ausgerichteten SPD-EEPROM . . . . .	562
Abb. 11.2	Exemplarische Darstellung der ersten 256 Byte eines SPD-EEPROMs für DDR5-Speichermodule . . . . .	594
Abb. 11.3	BIOS-Option SPD Write Disable auf einer MSI PRO Z690-A WIFI Hauptplatine . . . . .	596
Abb. 11.4	DDR3-Kapazitätsberechnung: Spezifikationsbereich <i>Calculating Module Capacity</i> . . . . .	604
Abb. 11.5	DDR4-Kapazitätsberechnung: Spezifikationsbereich <i>Calculating Module Capacity</i> . . . . .	608
Abb. 11.6	DDR5-Kapazitätsberechnung: Spezifikationsbereich <i>Calculating Module DRAM Capacity</i> . . . . .	613



Abb. 11.7	Exemplarische Darstellung des SPD-EEPROMs von DDR3-Speichermodulen .....	640
Abb. 11.8	Exemplarische Darstellung des SPD-EEPROMs von DDR4-Speichermodulen .....	651
Abb. 11.9	Exemplarische Darstellung des SPD-EEPROMs von DDR5-Speichermodulen .....	661
Abb. 11.10	Exemplarische Darstellung eines Gerätedumps für ein DDR5-Speichermodul .....	662
Abb. 11.11	Exemplarische Darstellung des SMBus-Überblicks mit DDR4-Speichermodulen .....	678
Abb. 11.12	Exemplarische Darstellung des SMBus-Überblicks mit DDR5-Speichermodulen .....	678
Abb. 11.13	Exemplarische Darstellung der Speichermoduldetails inkl. Dump des SPD-EEPROMs (hier für ein DDR3-Speichermodul) ....	681
Abb. 12.1	Exemplarische Darstellung der Demo 1 für die Klasse TMiTeC_SystemInfo .....	692
Abb. 12.2	Master-Projekt MSI mit allen in der Suite befindlichen Erkennungsroutinen .....	694
Abb. 12.3	Leistungszähler-Explorer für die Überwachung der Windows-Leistungsindikatoren .....	695
Abb. 12.4	WMI Explorer mit der Darstellung von WMI-Inhalten und Ausführung von WQL-basierten Befehlen .....	695
Abb. 12.5	HWiNFO64 System-Zusammenfassung als leistungsstarke Engine für das SDK .....	697
Abb. 12.6	HWiNFO64 Sensor-Status als leistungsstarke Engine für das SDK .....	698
Abb. 12.7	Systeminformationen der Software OCCT basierend auf dem HWiNFO SDK .....	702
Abb. 12.8	Sensorüberwachung der Software OCCT basierend auf dem HWiNFO SDK .....	702
Abb. 12.9	BenchMate als Benchmark-Überwachungstool (hier für die PI-Berechnung „Super PI“) .....	704

---

# Tabellenverzeichnis

Tab. 2.1	Programminfos in der linken oberen Programmsteuerungskomponente . . . . .	17
Tab. 2.2	Mögliche Werte der Eigenschaft <i>Architecture</i> von <i>TOSVersion</i> . . . . .	17
Tab. 2.3	Programmkontext in der mittleren oberen Programmsteuerungskomponente . . . . .	18
Tab. 2.4	Systemkomponenten-Überblick . . . . .	23
Tab. 2.5	Windows Ring-Modell der x86-Architektur . . . . .	25
Tab. 2.6	Verfügbare Systemzugriffe für den aktiven Programmkontext . . . . .	25
Tab. 2.7	Rechte-Übersicht der Ausführungsebene im Anwendungsmanifest . . . . .	29
Tab. 3.1	Eingabe- und Ausgabeparameter der API-Funktion <i>GetComputerName</i> . . . . .	41
Tab. 3.2	Eingabe- und Ausgabeparameter der API-Funktion <i>GetUserName</i> . . . . .	42
Tab. 3.3	Eingabe- und Ausgabeparameter der API-Funktion <i>OpenThreadToken</i> . . . . .	44
Tab. 3.4	Zugriffsrechte für Access-Token-Objekte . . . . .	45
Tab. 3.5	Eingabe- und Ausgabeparameter der API-Funktion <i>GetTokenInformation</i> . . . . .	47
Tab. 3.6	Informationswerte, die einem Zugriffstoken zugewiesen oder von ihm abgefragt werden können. . . . .	48
Tab. 3.7	Eingabe- und Ausgabeparameter der API-Funktion <i>AllocateAndInitializeSid</i> . . . . .	50
Tab. 3.8	Auszug der häufigsten virtuellen Tasten-Kennungen . . . . .	57
Tab. 4.1	Strukturinhalt von OSVERSIONINFOEX für die Verwendung von <i>GetVersionEx</i> . . . . .	64
Tab. 4.2	Eingabe- und Ausgabeparameter der API-Funktion <i>GetProductType</i> , Details in [6] . . . . .	67
Tab. 4.3	Übersicht der Windows-Versionsnummern mit dazugehörigen Windows-Varianten . . . . .	71
Tab. 4.4	Windows-Linienunterscheidung anhand <i>wProductType</i> . . . . .	71

Tab. 4.5	Kennungen und Windows-Versionen der Umgebungsvariable __COMPAT_LAYER. ....	81
Tab. 4.6	Durchsuchte Registrierungsschlüssel, wenn WIN32 aktiv ist und WOW64 inaktiv (also ein 32 Bit-Windows zum Einsatz kommt). ....	86
Tab. 4.7	Durchsuchte Registrierungsschlüssel, wenn WIN64 aktiv ist oder WIN32 mit aktivem WOW64 (also ein 64 Bit-Windows zum Einsatz kommt) . ....	87
Tab. 4.8	Record-Felder und deren Inhalt für jede gefundene installierte Applikation . ....	89
Tab. 4.9	Zugriffsrechte für das Öffnen der TRegistry-Klasse . ....	89
Tab. 4.10	Mögliche Typen von Datenwerten, die mit TRegistry. GetDataType ermittelbar sind . ....	90
Tab. 4.11	Parameter für die Windows API-Funktion <i>SHGetSpecialFolderLocation</i> . ....	90
Tab. 4.12	Auswahl der wichtigsten CSIDL-Werte für häufig benutzte Verzeichnisermittlungen . ....	102
Tab. 4.13	Parameter für die Windows API-Funktion <i>SHGetPathFromIDList</i> . ....	103
Tab. 4.14	Implementierte API-Funktion zur Behandlung von Windows-Umgebungsvariablen. ....	103
Tab. 5.1	Übersicht der SMBIOS-Tabellen in der Spezifikationsversion 3.7.0 . ....	121
Tab. 5.2	Basisvariablen eines SMBIOS-Tabellen-Headers (Kopfbereich) . ....	122
Tab. 5.3	Eingabe- und Ausgabeparameter der API-Funktion <i>GetSystemFirmwareTable</i> . ....	137
Tab. 5.4	Veröffentlichte Eigenschaften der TSM BIOS-Klasse . ....	158
Tab. 5.5	Spezifikationsfelder versus Delphi-Variablentypen. ....	163
Tab. 6.1	Überblick der ersten CPUID-Funktionen für Intel und AMD. ....	290
Tab. 6.2	Strukturinhalt von SYSTEM_INFO für die Verwendung von <i>GetSystemInfo</i> und <i>GetNativeSystemInfo</i> . ....	308
Tab. 6.3	Eingabe- und Ausgabeparameter der API-Funktion <i>GetLogicalProcessorInformationEx</i> . ....	331
Tab. 6.4	Veröffentlichte Eigenschaften der TProcessor-Klasse. ....	336
Tab. 7.1	Erzeugte I/O Control Codes für unseren Kernelmodus-Treiber . ....	375
Tab. 7.2	Die wichtigsten IRP-Funktionscodes . ....	389
Tab. 7.3	Ausgabestruktur für IOCTL_PCANALYS_TransferTest (insgesamt 4 Bytes) . ....	395
Tab. 7.4	Ausgabestruktur für IOCTL_PCANALYS_Version (insgesamt 8 Bytes) . ....	396
Tab. 7.5	Eingabestruktur für IOCTL_PCANALYS_ReadMSR (insgesamt 4 Bytes) . ....	397
Tab. 7.6	Ausgabestruktur für IOCTL_PCANALYS_ReadMSR (insgesamt 8 Bytes) . ....	398

Tab. 7.7	Eingabestruktur für IOCTL_PCANALYS_WriteMSR (insgesamt 12 Bytes) . . . . .	400
Tab. 7.8	Eingabestruktur für IOCTL_PCANALYS_ReadPCI (insgesamt 4 Bytes) . . . . .	402
Tab. 7.9	Ausgabestruktur für IOCTL_PCANALYS_ReadPCI (insgesamt 4 Bytes) . . . . .	402
Tab. 7.10	Eingabestruktur für IOCTL_PCANALYS_WritePCI (insgesamt 8 Bytes) . . . . .	404
Tab. 7.11	Ausgabestruktur für IOCTL_PCANALYS_WritePCI (insgesamt 4 Bytes) . . . . .	404
Tab. 7.12	Eingabestruktur für IOCTL_PCANALYS_ReadMem8Bit, IOCTL_PCANALYS_ReadMem16Bit und IOCTL_ PCANALYS_ReadMem32Bit (jeweils 4 Bytes) . . . . .	406
Tab. 7.13	Ausgabestruktur für IOCTL_PCANALYS_ReadMem8Bit (insgesamt 1 Byte) . . . . .	406
Tab. 7.14	Ausgabestruktur für IOCTL_PCANALYS_ReadMem16Bit (insgesamt 2 Bytes) . . . . .	406
Tab. 7.15	Ausgabestruktur für IOCTL_PCANALYS_ReadMem32Bit (insgesamt 4 Bytes) . . . . .	406
Tab. 7.16	Cache-Verhalten während des Abbildungsvorgangs mit MmMapIoSpace . . . . .	408
Tab. 7.17	Eingabestruktur für IOCTL_PCANALYS_WriteMem8Bit (insgesamt 5 Bytes) . . . . .	410
Tab. 7.18	Eingabestruktur für IOCTL_PCANALYS_WriteMem16Bit (insgesamt 6 Bytes) . . . . .	410
Tab. 7.19	Eingabestruktur für IOCTL_PCANALYS_WriteMem32Bit (insgesamt 8 Bytes) . . . . .	410
Tab. 7.20	Eingabestruktur für IOCTL_PCANALYS_ReadPort8Bit, IOCTL_PCANALYS_ReadPort16Bit und IOCTL_ PCANALYS_ReadPort32Bit (jeweils 4 Bytes) . . . . .	412
Tab. 7.21	Ausgabestruktur für IOCTL_PCANALYS_ReadPort8Bit (insgesamt 1 Byte) . . . . .	412
Tab. 7.22	Ausgabestruktur für IOCTL_PCANALYS_ReadPort16Bit (insgesamt 2 Bytes) . . . . .	413
Tab. 7.23	Ausgabestruktur für IOCTL_PCANALYS_ReadPort32Bit (insgesamt 4 Bytes) . . . . .	413
Tab. 7.24	Eingabestruktur für IOCTL_PCANALYS_WritePort8Bit (insgesamt 5 Bytes) . . . . .	415
Tab. 7.25	Eingabestruktur für IOCTL_PCANALYS_WritePort16Bit (insgesamt 6 Bytes) . . . . .	415
Tab. 7.26	Eingabestruktur für IOCTL_PCANALYS_WritePort32Bit (insgesamt 8 Bytes) . . . . .	415

Tab. 8.1	Record SERVICE_STATUS mit DWords als Treiberkonfiguration . . .	426
Tab. 8.2	Parameter für die CreateService-API-Funktion. . . . .	428
Tab. 8.3	Parameter für die CreateFile-API-Funktion . . . . .	433
Tab. 8.4	Verwendung von C++- und Delphi-Datentypen . . . . .	437
Tab. 8.5	Parameter der API-Funktion DeviceIoControl . . . . .	441
Tab. 8.6	Ausgabestruktur für IOCTL_PCANALYS_TransferTest (insgesamt 4 Bytes) . . . . .	443
Tab. 8.7	Ausgabestruktur für IOCTL_PCANALYS_Version (insgesamt 8 Bytes) . . . . .	444
Tab. 8.8	Eingabestruktur für IOCTL_PCANALYS_ReadMSR (insgesamt 4 Bytes) . . . . .	445
Tab. 8.9	Ausgabestruktur für IOCTL_PCANALYS_ReadMSR (insgesamt 8 Bytes) . . . . .	445
Tab. 8.10	Eingabestruktur für IOCTL_PCANALYS_WriteMSR (insgesamt 12 Bytes) . . . . .	447
Tab. 8.11	Eingabestruktur für IOCTL_PCANALYS_ReadPCI (insgesamt 4 Bytes) . . . . .	448
Tab. 8.12	Ausgabestruktur für IOCTL_PCANALYS_ReadPCI (insgesamt 4 Bytes) . . . . .	448
Tab. 8.13	Eingabestruktur für IOCTL_PCANALYS_WritePCI (insgesamt 8 Bytes) . . . . .	449
Tab. 8.14	Ausgabestruktur für IOCTL_PCANALYS_WritePCI (insgesamt 4 Bytes) . . . . .	449
Tab. 8.15	Eingabestruktur für IOCTL_PCANALYS_ReadMem8Bit, IOCTL_PCANALYS_ReadMem16Bit und IOCTL_ PCANALYS_ReadMem32Bit (jeweils 4 Bytes) . . . . .	450
Tab. 8.16	Ausgabestruktur für IOCTL_PCANALYS_ReadMem8Bit (insgesamt 1 Byte) . . . . .	450
Tab. 8.17	Ausgabestruktur für IOCTL_PCANALYS_ReadMem16Bit (insgesamt 2 Bytes) . . . . .	451
Tab. 8.18	Ausgabestruktur für IOCTL_PCANALYS_ReadMem32Bit (insgesamt 4 Bytes) . . . . .	451
Tab. 8.19	Eingabestruktur für IOCTL_PCANALYS_WriteMem8Bit (insgesamt 5 Bytes) . . . . .	452
Tab. 8.20	Eingabestruktur für IOCTL_PCANALYS_WriteMem16Bit (insgesamt 6 Bytes) . . . . .	452
Tab. 8.21	Eingabestruktur für IOCTL_PCANALYS_WriteMem32Bit (insgesamt 8 Bytes) . . . . .	452
Tab. 8.22	Eingabestruktur für IOCTL_PCANALYS_ReadPort8Bit, IOCTL_PCANALYS_ReadPort16Bit und IOCTL_ PCANALYS_ReadPort32Bit (jeweils 4 Bytes) . . . . .	454

Tab. 8.23	Ausgabestruktur für IOCTL_PCANALYS_ReadPort8Bit (insgesamt 1 Byte) . . . . .	454
Tab. 8.24	Ausgabestruktur für IOCTL_PCANALYS_ReadPort16Bit (insgesamt 2 Bytes) . . . . .	454
Tab. 8.25	Ausgabestruktur für IOCTL_PCANALYS_ReadPort32Bit (insgesamt 4 Bytes) . . . . .	454
Tab. 8.26	Eingabestruktur für IOCTL_PCANALYS_WritePort8Bit (insgesamt 5 Bytes) . . . . .	455
Tab. 8.27	Eingabestruktur für IOCTL_PCANALYS_WritePort16Bit (insgesamt 6 Bytes) . . . . .	455
Tab. 8.28	Eingabestruktur für IOCTL_PCANALYS_WritePort32Bit (insgesamt 8 Bytes) . . . . .	455
Tab. 8.29	Kontrollkennungen für die API-Funktion ControlService. . . . .	458
Tab. 8.30	Mögliche Fehlerkennungen der API-Funktion DeleteService. . . . .	460
Tab. 9.1	Auszugsweise Darstellung der Modell-spezifischen Intel-MSRs . . . . .	467
Tab. 9.2	MSR 64 Bit-Registeraufteilung . . . . .	489
Tab. 9.3	Bit-Felder eines 32 Bit-Registers für das Ansprechen mit binären Operanden. . . . .	494
Tab. 10.1	Belegung des Configuration Address Register (OCF8h) bei Konfigurationsmechanismus 1. . . . .	499
Tab. 10.2	Belegung des Configuration Space Enable Register (OCF8h) bei Konfigurationsmechanismus 2. . . . .	499
Tab. 10.3	Belegung des Data Register (OCFAh) bei Konfigurationsmechanismus 2. . . . .	500
Tab. 10.4	Header-Aufbau des PCI-Konfigurationsbereiches für den Standard-Header (Typ 0). . . . .	501
Tab. 10.5	Veröffentlichte Eigenschaften der TPCIBus-Klasse . . . . .	535
Tab. 10.6	Belegung des Command Registers im Offset 04h-05h . . . . .	551
Tab. 10.7	Belegung des Status Registers im Offset 06h-07h. . . . .	554
Tab. 11.1	Auszug der wichtigsten SMBus-Register auf Grundlage der SMBus-Basisadresse mit einem Offset . . . . .	564
Tab. 11.2	Belegung des SMBus Host Status Registers laut Intel-Definition in Offset 0 . . . . .	569
Tab. 11.3	Belegung des SMBus Host Control Registers laut Intel-Definition in Offset 2 . . . . .	572
Tab. 11.4	Aufbau des Host Address Register mit dem Offset 4h . . . . .	573
Tab. 11.5	Aufbau des MR11-Registers bei DDR5-Speichermodule für den Seitenwechsel . . . . .	580
Tab. 11.6	Vorbereitende und durchführende Aufgaben für die Mutex-Verwendung . . . . .	583
Tab. 11.7	Inhalt des Host Configuration Registers an Offset 40h für neuere Intel-Chipsätze. . . . .	595

Tab. 11.8	Variablenübersicht für die Kapazitätsberechnung von DDR2-Speichermodulen . . . . .	600
Tab. 11.9	DDR2-Kapazitätsberechnung: Anzahl der Ränge in Offset 5 (rechte Spalte) . . . . .	600
Tab. 11.10	DDR2-Kapazitätsberechnung: Ermittlung der Modul-Rangdichte in Offset 31 . . . . .	600
Tab. 11.11	Variablenübersicht für die Kapazitätsberechnung von DDR2 Fully Buffered DIMM-Speichermodulen . . . . .	602
Tab. 11.12	DDR2 FBDIMM-Kapazitätsberechnung: Anzahl der Zeilen- und Spaltenadressbits sowie der Bänke in Offset 4 . . . . .	602
Tab. 11.13	DDR2 FBDIMM-Kapazitätsberechnung: Anzahl der Ränge in Offset 7 . . . . .	603
Tab. 11.14	Variablenübersicht für die Kapazitätsberechnung von DDR3-Speichermodulen . . . . .	605
Tab. 11.15	DDR3-Kapazitätsberechnung: Gesamte SDRAM-Kapazität in Offset 4 . . . . .	605
Tab. 11.16	DDR3-Kapazitätsberechnung: Primäre Bus-Breite in Offset 8. . . . .	605
Tab. 11.17	DDR3-Kapazitätsberechnung: Gesamtträge und SDRAM-Gerätebreite in Offset 7 . . . . .	605
Tab. 11.18	Variablenübersicht für die Kapazitätsberechnung von DDR4-Speichermodulen . . . . .	608
Tab. 11.19	DDR4-Kapazitätsberechnung: Gesamte SDRAM-Kapazität in Offset 4 . . . . .	609
Tab. 11.20	DDR4-Kapazitätsberechnung: Primäre Bus-Breite in Offset 13. . . . .	609
Tab. 11.21	DDR4-Kapazitätsberechnung: SDRAM-Gerätebreite in Offset 12. . . . .	609
Tab. 11.22	DDR4-Kapazitätsberechnung: Die-Anzahl in Offset 6 . . . . .	610
Tab. 11.23	Variablenübersicht für die Kapazitätsberechnung von symmetrischen DDR5-Speichermodulen . . . . .	612
Tab. 11.24	Variablenübersicht für die Kapazitätsberechnung von symmetrischen DDR5-Speichermodulen (und für gerade Ranks bei asymmetrischen DDR-Speichermodulen) . . . . .	614
Tab. 11.25	DDR5-Kapazitätsberechnung: Rang-Mix und Anzahl der Paket-Ränge pro Sub-Kanal in Offset 234. . . . .	614
Tab. 11.26	DDR5-Kapazitätsberechnung: Anzahl der Sub-Kanäle pro DIMM und primäre Busbreite pro Sub-Kanal in Offset 235. . . . .	614
Tab. 11.27	DDR5-Kapazitätsberechnung: Die pro Paket und SDRAM- Dichte pro Die in den Offsets 4 (First SDRAM) und 8 (Second SDRAM) . . . . .	615
Tab. 11.28	DDR5-Kapazitätsberechnung: SDRAM I/O-Breite in den Offsets 6 (First SDRAM) und 10 (Second SDRAM) . . . . .	615
Tab. 11.29	SDRAM Geräteattribute – CAS-Latenz in Byte 18. . . . .	630

Tab. 11.36	Gesamte und verwende Bytes des SPD-EEPROM-Datenbereiches in Byte 0 . . . . .	639
Tab. 11.30	Modultyp in Byte 3 . . . . .	632
Tab. 11.31	Ermittlung von ECC-Speicher über Byte 8 . . . . .	635
Tab. 11.32	Anzahl der Bänke über Byte 4 . . . . .	636
Tab. 11.33	SDRAM-Adressierung mit Zeilen und Spalten in Byte 5 . . . . .	637
Tab. 11.34	Unterstützte CAS-Latenzen auf dem niedrigen Byte 14 . . . . .	638
Tab. 11.35	Unterstützte CAS-Latenzen auf dem höheren Byte 15 . . . . .	638
Tab. 11.37	Modultyp in Byte 3 . . . . .	642
Tab. 11.38	Erweiterter Modultyp in Byte 15 . . . . .	642
Tab. 11.39	Ermittlung von ECC-Speicher über Byte 13 . . . . .	644
Tab. 11.40	Anzahl der Bänke über Byte 4 . . . . .	646
Tab. 11.41	SDRAM-Adressierung mit Zeilen und Spalten in Byte 5 . . . . .	646
Tab. 11.42	Unterstützte CAS-Latenzen im niedrigen Bereich (Bit 7 von Byte 23 = 0) . . . . .	647
Tab. 11.43	Unterstützte CAS-Latenzen im höheren Bereich (Bit 7 von Byte 23 = 1) . . . . .	648
Tab. 11.44	Gesamte und verwende Bytes des SPD-EEPROM-Datenbereiches in Byte 0 . . . . .	649
Tab. 11.45	Modultyp in Byte 3 . . . . .	652
Tab. 11.46	Ermittlung von ECC-Speicher über Byte 235 . . . . .	656
Tab. 11.47	DDR5-Modultyp anhand der symmetrischen oder asymmetrischen Rangmischung in Byte 234 . . . . .	657
Tab. 11.48	SDRAM I/O-Breite in Byte 6 . . . . .	658
Tab. 11.49	Dies pro Paket und SDRAM-Dichte pro Die über Byte 4 . . . . .	659
Tab. 11.50	Gesamte Bytes des SPD-EEPROM-Datenbereiches in Byte 0 . . . . .	660
Tab. 11.51	Speichertypen und anzusprechende SPD-EEPROM-Bytes für JEDEC-Herstellererkennung . . . . .	667
Tab. 11.52	Speichertypen und anzusprechende SPD-EEPROM-Bytes für die Modellbezeichnung . . . . .	669
Tab. 11.53	Veröffentlichte Eigenschaften der TSMBus-Klasse . . . . .	671
Tab. 12.1	Klassenübersicht der Hauptklassen für die MiTeC System Information Component Suite . . . . .	687
Tab. 12.2	Klassenübersicht der Monitorklassen für die MiTeC System Information Component Suite . . . . .	689
Tab. 12.3	Klassenübersicht der Thread-basierten Monitorklassen für die MiTeC System Information Component Suite . . . . .	689
Tab. 12.4	Klassenübersicht der forensischen Klassen für die MiTeC System Information Component Suite . . . . .	689
Tab. 12.5	Demo-Übersicht der MiTeC System Information Component Suite (jeweils für FPC und Delphi) . . . . .	690



---

Tab. 12.6	Applikations-Übersicht der MiTeC System Information Component Suite . . . . .	693
Tab. 12.7	Themenschwerpunkte und mögliche Systemdetails des HWiNFO SDK. . . . .	700
Tab. 14.1	Kernelmodus-Treiber. . . . .	712
Tab. 14.2	Delphi-Projekt <i>PC Analyser (Quellcodes)</i> . . . . .	713
Tab. 14.3	Delphi-Projekt <i>PC Analyser (Binaries)</i> . . . . .	714

---

## Abkürzungsverzeichnis

Abkürzung	Bedeutung (Beschreibung)
ACE	Access Control Entry (Eintrag einer Tabelle, der den Zugriff auf ein Objekt kontrolliert oder überwacht und Bestandteil der ACL ist)
ACL	Access Control List (Tabelle, die Windows mitteilt, welche Zugriffsrechte jeder Anwender auf bestimmte Systemobjekte hat)
ACPI	Advanced Configuration and Power Interface (Industriestandard und Schnittstelle für EnergiEVERwaltung)
ADK	Assessment and Deployment Kit (Paketsammlung von Microsoft, um Windows automatisch installieren zu lassen)
AGP	Accelerated Graphics Port (Anschlussnorm auf Hauptplatinen zur direkten Verbindung zwischen Grafikkarte und dem Chipsatz bzw. dessen Northbridge)
ANSI	American National Standards Institute (Private, gemeinnützige und amerikanische Organisation zur Koordinierung der Entwicklung freiwilliger Normen in den Vereinigten Staaten. Der Sitz der Non-Profit-Organisation ist Washington, D.C.)
API	Application Programming Interface (Synonym für eine beliebige Programmierschnittstelle)
APIC	Advanced Programmable Interrupt Controller (Kontroller für die Verteilung von Interrupts)
BCD	Boot Configuration Data (Firmware-unabhängige Datenbank für Konfigurationsdaten, die während des Bootens benötigt werden. Sie ersetzt die Datei „boot.ini“, die ursprünglich vom NT-Loader genutzt wurde)
BIOS	Basic Input/Output System (Firmware für PCs, die während des Kaltstarts die Hardware konfiguriert, über eine GUI die Konfiguration ermöglicht und die Kontrolle ans Betriebssystem abgibt)
BIST	Built In Self Test (Gerätespezifischer Selbsttest, der nach der PCI-Spezifikation ausgelöst und ausgewertet werden kann, wenn das PCI-Gerät diesen Test unterstützt)

---

BKGD	BIOS and Kernel Developer's Guide (AMD-Dokumentation mit enthaltenen CPUID-Funktionen und Modell-spezifischen Registern (MSR))
CAS	Column Address Strobe (Speicherlatenz die benötigt wird, um eine Spalte im Hauptspeicher des Rechners zu adressieren)
CSIDL	Constant Special Item ID List (Vordefinierte Konstanten für die Ermittlung spezieller Ordner (etwa mit unterschiedlichen Namen und Speicherorten auf unterschiedlichen Systemen))
CPU	Central Processing Unit (Allgemeine Bezeichnung für den Zentralprozessor)
CPUID	Central Processing Unit Identification (Prozessorbefehl zur Ermittlung von Prozessoren mit vielen seiner Eigenschaften)
DDK	Driver Development Kit (Toolsammlung von Microsoft, die eine Entwicklung von Gerätetreibern für die Windows-Plattform ermöglicht. Die Sammlung umfasst Dokumentationen, Beispiele, Build-Umgebungen und Tools für Treiberentwickler)
DDR5	Double Data Rate 5 (Speichertechnologie in der 5. Generation, die im Vergleich zu DDR4 weniger Strom verbraucht und die Bandbreite erhöht)
DIMM	Dual Inline Memory Module (Doppelreihiges Speichermodul, das auf beiden Seiten Signale und Speicherbausteine enthält)
DMI	Desktop Management Interface (Standardverfahren zur Erkennung und Verwaltung von Rechnerkomponenten, das von der Distributed Management Task Force (DMTF) entwickelt wurde)
DMTF	Distributed Management Task Force (Normungsorganisation von Unternehmen der IT-Industrie, der eine Vielzahl von namhaften Unternehmen angehören)
DTS	Digital Thermal Sensor (Digitaler Temperatursensor auf einem Intel Prozessor-Chip, der eine Überwachung ermöglicht. Damit lässt sich erkennen, wie nah das System an der Temperaturgrenze ist, ohne zerstörerische Tests durchzuführen)
ECC	Error Correcting Code (Methode zur Erkennung von Hauptspeicherfehlern, wobei ein Hashwert über die 64 Bits einer Speicherzelle berechnet und in zusätzlichen Bits abgelegt wird)
EMS	Emergency Management Services (Dienstleistungen, die im Zusammenhang mit dem Katastrophenschutz stehen)
GPU	Graphics Processing Unit (Grafikprozessor mit Nutzung von Grafikspeicher und Bildschirmausgabe)
GUID	Globally Unique Identifier (Eindeutiger 16 Byte-Zahlenwert (128 Bit) zur Identifizierung in Computersystemen)
HKCU	HKEY Current User (Windows-Registrierungsschlüssel mit benutzerspezifischen Inhalten)

HKLM	HKEY Local Machine (Windows-Registrierungsschlüssel mit maschinenspezifischen Inhalten)
HWCR	Hardware Configuration Register (Weit verbreitetes AMD MSR mit allgemeinen Details zur Hardware-Konfiguration)
IOCTL	Input/Output Control (Systemaufruf für gerätespezifische Ein- und Ausgabeoperationen)
IRP	I/O Request Package (Kernelmodusstrukturen, die vom Windows-Treibermodell und von Windows NT-basierten Gerätetreibern verwendet werden, um miteinander und mit dem Betriebssystem zu kommunizieren)
KMDF	Kernel-Mode Driver Framework (Microsoft Treiber-Framework, um Treiberentwicklern beim Erstellen und Verwalten von Gerätetreibern im Kernelmodus für Windows 2000 und höher zu helfen)
MDL	Memory Descriptor List (I/O-Puffer, der sich über einen Bereich zusammenhängender virtueller Speicheradressen erstreckt, und über mehrere physische Seiten verteilt sein kann, die nicht zusammenhängend sein müssen. Das Betriebssystem verwendet eine MDL, um das physische Seitenlayout für einen virtuellen Speicherpuffer zu beschreiben)
MMX	Multi Media Extension (Von Intel in 1997 entwickelte Erweiterung des Prozessor-Befehlssatzes)
MSR	Model-specific Register (Modell-spezifische Prozessorregister, die auf Kernel-Ebene des Betriebssystems ausgeführt werden und weitreichende Details liefern bzw. Einstellungen ermöglichen)
Mutex	Mutual Exclusion Object (Methode für die Synchronisierung von Datenzugriffen zwischen mehreren Prozessen)
NDA	Non-Disclosure Agreement (Vertraulichkeitsvereinbarung, die Hersteller mit Geschäftspartnern abschließen, um vertraulichen Informationsfluss zu steuern)
PCH	Platform Controller Hub (Intel-Bezeichnung für Chipsätze und die Anbindung zwischen Northbridge und Southbridge an das System)
PCI	Peripheral Component Interconnect (Bus-Standard zur Verbindung von Peripheriegeräten mit dem Chipsatz eines Prozessors – es existieren zahlreiche Varianten und Einsatzgebiete des Standards)
PEC	Packet Error Checking (Gepackter Fehlercode, der an das Ende einer Transaktion angehängt und berechnet wird)
POST	Power On-Self Test (Initialisierungsprozess des BIOS, bei dem die Hardware angesteuert und konfiguriert wird, und dem Betriebssystem bzw. dem dazugehörigen Bootloader die Kontrolle übergeben wird)
PPR	Processor Programming Reference (AMD-Dokumentation mit enthaltenen CPUID-Funktionen und Modell-spezifischen Registern (MSR))

---

PSIG	PCI Special Interest Group (Konsortium der Elektronikindustrie, das für die Spezifizierung der Peripheral Component Interconnect-, PCI-X- und PCI Express-Computerbusse verantwortlich ist. Es hat seinen Sitz in Beaverton, Oregon)
SBA	SMB Base Address (SMBus-Basisadresse, die aus dem Konfigurationsbereich des PCI-Gerätes für den SMBus-Kontroller gewonnen wird und die Basis für die SMBus-Kommunikation (inkl. der Ermittlung von Speichermodulen über SPD EEPROM) darstellt)
SDK	Software Development Kit (Toolsammlung von Microsoft, die Dokumentationen, Headerdateien, Compiler, Programmbibliotheken, Beispiele und Programmierwerkzeuge für die Entwicklung von Microsoft Windows und dem .NET-Framework enthält)
SDRAM	Synchronous Dynamic Random Access Memory (Halbleiterbasierte Speichervariante, bei der sich der Speicher mit dem Systemtakt des PCs synchronisiert)
SID	Security Identifier (Eindeutiger Sicherheits-Identifikator, den Windows automatisch vergibt, um jedes System, jeden Benutzer und jede Gruppe dauerhaft zu identifizieren)
SMART	Self-Monitoring, Analysis and Reporting Technology (Industriestandard zur Überwachung von Festplatten und SSDs mit einer Vorhersagemöglichkeit für einen potenziellen bevorstehenden Ausfall (Fitneswerte))
SMBIOS	System Management BIOS (Industriestandard zur Erkennung und Verwaltung von Rechnerkomponenten, der von der Distributed Management Task Force (DMTF) entwickelt wurde)
SMBus	System Management Bus (Zweileiterbus für die Kommunikation mit Baugruppen, etwa Sensor-Chips und Konfigurationsdaten von Speichermodulen)
SPD-EEPROM	Serial Presence Detect -Electrically Erasable Programmable Read-Only Memory (Verfahren zur automatischen Konfiguration von Speichermodulen auf einem Nur-Lese-Speicher, der elektrisch gelöscht werden kann)
SPDWD	Serial Presence Detect Write Disable (Intel-Schutzmechanismus ab DDR4-Speichermodulen, bei denen jeder Schreibzugriff auf die Speichermodule bzw. deren Basisadressen 50h–57h unterbunden wird)
SSE	Streaming SIMD Extensions (Von Intel in 1999 entwickelte Erweiterung des Prozessor-Befehlssatzes)
TPM	Trusted Platform Module (Chip auf der Hauptplatine, der einen Computer oder ähnliche Geräte um grundlegende Sicherheitsfunktionen erweitert)

TSC	Time Stamp Counter (Hochauflösende Zeitquelle in Prozessoren, die in einem 64 Bit-Register die Anzahl der Prozessorzyklen seit dem letzten Reset hochzählen)
UAC	User Account Control (Windows-Benutzerkontensteuerung)
UEFI	Unified Extensible Firmware Interface (Neuartige Rechnerfirmware als Nachfolger des BIOS mit Sicherheitsfunktionen wie SecureBoot)
USB	Universal Serial Bus (Datenübertragungstechnologie zur Verbindung von Rechnern mit externen Geräten)
WBEM	Web-Based Enterprise Management (Microsoft-Implementierung des webbasierten Unternehmensmanagements)
WDK	Windows Driver Kit (Software-Toolsammlung von Microsoft, die eine Entwicklung von Gerätetreibern für die Windows-Plattform ermöglicht. Die Sammlung umfasst Dokumentationen, Beispiele, Build-Umgebungen und Tools für Treiberentwickler)
WDM	Windows Driver Model (Framework für Gerätetreiber, das mit Windows 98 und Windows 2000 eingeführt wurde, um VxD zu ersetzen, welches in älteren Windows-Versionen wie Windows 95 und Windows 3.1 sowie dem Windows NT Driver Model verwendet wurde)
WHEA	Windows Hardware Error Architecture (Mechanismus zur Behandlung von Hardwarefehlern des Betriebssystems, der mit Windows Vista SP1 und Windows Server 2008 als Nachfolger der Machine Check Architecture in früheren Windows-Versionen eingeführt wurde)
WMI	Windows Management Instrumentation (Microsoft-Implementierung für die Infrastruktur von Verwaltungsdaten und -vorgängen unter Windows-basierten Betriebssystemen)
WQL	WMI Query Language (SQL-ähnliche Skriptsprache zur Ansteuerung von WMI)
XMP	Extreme Memory Profile (Von Intel eingeführte Zertifizierungen beginnend mit DDR3-Speichermodulen für eine bessere Ausnutzung der Speichergeschwindigkeit und damit stabileren Betrieb)