

Jörg Franz Wollert

Erfolgreich automatisieren *unter* Microsoft Windows

Technologie – Chancen – Grenzen und Lösungen
für Automatisierungsaufgaben unter Windows

Mit 188 Abbildungen

Franzis'

Inhalt

1	Warum Windows einsetzen?	15
1.1	Marktprognosen	15
1.2	Microsoft für Industrial Applications	17
2	Die Windows-Systemfamilie	19
2.1	Historische Entwicklung	19
2.2	Betriebssystemprofile	21
2.3	Windows 3.1	23
2.3.1	Protected Mode	25
2.3.2	Enhanced Mode	26
2.4	Windows 95	28
2.4.1	Treibermodell	29
2.4.2	Aufbau und Elemente der Windows 95 Oberfläche	30
2.5	Windows 98	33
2.6	Windows NT	35
2.6.1	Profil von Windows NT 4.0 Workstation	37
2.6.2	Microkernel-Aufbau und die Windows NT-Architektur	42
2.6.3	Benutzeroberflächen	53
2.6.4	Erweiterte Management Funktionen	57
2.6.5	Systemerweiterungen	58
2.7	Windows NT 5.0	80
2.8	Windows Zukunft	62
2.9	Windows CE	63
3	Microsoft Windows Technologie	66
3.1	Softwareentwicklung für Windows	66
3.1.1	Programmiermodell	66
3.1.2	Visual Basic	71
3.1.3	Visual C++	73
3.1.4	Java	78
3.2	Komponenten Software	79
3.2.1	OLE-Steuerelemente	82
3.2.2	Nicht-Microsoft-Komponenten	83
3.2.3	OLE – Object Linking and Embedding	84

3.2.4	Das Component Object Model – COM	86
3.2.5	Einordnung von COM und DCOM	93
3.3	Prozeßkommunikation	96
3.3.1	Datenaustausch mit der Zwischenablage	97
3.3.2	Dynamischer Datenaustausch DDE	97
4	Windows als Echtzeitbetriebssystem	101
4.1	Grundlagen Echtzeitsysteme	101
4.1.1	Scheduling-Algorithmen	103
4.1.2	Anforderungen an Echtzeitbetriebssysteme	106
4.2	Windows im Echtzeitkontext	106
4.2.1	Interrupt-Handling unter Windows	106
4.2.2	Bewertung der Interruptlatenzzeit unter Windows	113
4.2.3	Windows Prozeßsystem	117
4.2.4	Besonderheiten bei Windows 95	120
4.2.5	Besonderheiten bei Windows NT:	121
4.2.6	Echtzeiterweiterungen für Windows und Windows NT	122
4.2.7	Bewertung der Echtzeitfähigkeit	127
5	Windows CE als Embedded Controller Betriebssystem	128
5.1	Einsatz und Profil von Embedded Controller	128
5.1.1	Was zeichnet Embedded Controller aus?	128
5.1.2	Entwicklungsmethodik	130
5.2	Aufbau und Funktionsweise von Windows CE	131
5.2.1	Aufbau von Windows CE	131
5.2.2	Echtzeiteigenschaften	136
5.2.3	Interrupt Handling in Device Driver	139
5.3	Entwicklungsplattformen	140
5.3.1	Prozessoren	140
5.3.2	Hitachi Super-H	141
5.3.3	AMD-Elan	143
5.4	Programmierungsumgebung und Anwendung	145
5.4.1	Programmierungsumgebung	145
5.4.2	Embedded Toolkit	146
5.4.3	Softwareentwicklung mit Java, C++ und Visual Basic	149
5.4.4	Portable Software für Windows CE	150
6	Windows als Basis für IEC-1131-Industrierechner ..	166
6.1	Speicherprogrammierbare Steuerungen	166
6.2	Systemübergreifende Steuerungssoftware	169
6.2.1	Die IEC-1131-Norm	169

6.2.2	Herstellerunabhängige kompatible Software – die PLCOpen	170
6.3	Grundlagen der Norm IEC-1131-3	171
6.3.1	Softwareengineering	171
6.3.2	Das Programmiermodell	172
6.3.3	Das Kommunikationsmodell	174
6.3.4	Allgemeine Eigenschaften und Datentypen	175
6.3.5	Programmorganisationseinheiten POE	182
6.3.6	Aufbau von Programmorganisationseinheiten	183
6.3.7	Die Programmiersprachen	188
6.3.8	Schlußbetrachtung	199
6.4	Einsatz von Industrie PC Steuerungen in der Maschinenautomation	199
6.4.1	Industrie-PC	200
6.4.2	Feldbussystem	201
6.4.3	Steuerungssoftware	202
6.4.4	SPS in neuem Gewand: Multi-SPS, Multitasking	203
6.4.5	Wirtschaftliche Aspekte einer skalierbaren PC – Steuerungstechnik	205
6.5	Automatisieren und Motion Control mit IEC1131-3	207
6.5.1	Implementierungen von Motion Control: Strukturvergleich	207
6.5.2	Motion Control und IEC1131-3	210
6.5.3	Die Motion Control-Interfacebausteine	212
6.5.4	Steuerungs-FB und Ihr Verhalten	212
7	OPC – Ein offener Standard für Automatisierungs- lösungen unter Windows	217
7.1	OLE für die Automatisierungstechnik	217
7.2	OPC- Schnittstellen	219
7.3	Funktionsweise von OPC	220
7.3.1	Client-Server Konzepte	221
7.3.2	Verteilte Prozeßautomatisierung	223
7.4	Das Klassenmodell der OPC-Schnittstelle	225
7.4.1	Das OPC Server Objekt	226
7.4.2	Das OPCGroup-Objekt	227
7.4.3	Das OPCItem Objekt	227
7.4.4	Datenzugriff über einen OPC-Server	228
7.5	OPC Ausblick	230
8	Die CALL-Schnittstelle von OpenControl	231
8.1	CALL-Engineering – Projektierung und Engineering von Automatisierungssystemen	231

8.1.1	Inhalte von CALL-E	233
8.1.2	CALL-E Architektur	234
8.1.3	CALL-E Datenbasis	235
8.2	CALL-Runtime – Einheitliche Kommunikation von Applikationen im Feld	238
8.2.1	Inhalte von CALL-R	239
8.2.2	Implementierung der CALL-R Funktionalität unter Verwendung der OPC-Definitionen	241
8.2.3	Architektur	241
8.3	CALL-Peripherie – Echtzeitdatentransfer zwischen Feld-Komponenten	243
8.3.1	Inhalte von CALL-P	244
8.3.2	CALL-P Architektur	245
8.3.3	Datenaustausch	247
8.3.4	Management	248
8.3.5	Herstellerspezifisches Treiberinterface	249
8.3.6	Prozeßdatenabbilder	249
8.3.7	Variablen austausch zwischen Steuerungen	250
9	Industriestandards für PC-basierte Automatisierungslösungen	252
9.1	Software und Hardwarekomponenten – ein Baukasten für offene Automatisierung	252
9.1.1	Komponenten in der Automatisierungstechnik	253
9.1.2	Standard PCs als Integrationsplattform	255
9.2	Open Control und Complete Vertical Automation	256
9.2.1	Standards nutzen mit Open Control	259
9.2.2	Die CALL-Schnittstellen von Open Control – Die Basis der CVI	260
9.2.3	Auswirkungen von Open Control	261
9.3	Total-Integrated-Automation	263
9.3.1	Offenheit von TIA	265
9.3.2	Migration von Fertigungstechnik und Verfahrenstechnik ...	266
9.3.3	Ein Blick in die Zukunft	268
10	Applikationen und Anwendungen	269
10.1	PC-based Control für ein durchgängiges Informationsmodell	269
10.1.1	Neue Informationsmodelle mit Microsoft Client-Server-Technologien	271
10.1.2	PC-based Control	275
10.1.3	Internet basierende Informationssysteme und die Push Technologie	277

10.2	Werkzeuge und Bibliotheken für prozeßnahe und zeitkritische Industrieanwendungen unter Windows 95 und Windows NT	278
10.2.1	Das Konzept der Kithara Key-Board-Reihe	280
10.2.2	Zugriff auf physischen Speicher bei Windows	282
10.2.3	Direkter Zugriff auf beliebige I/O-Ports	282
10.2.4	Timer in Millisekunden-Auflösung	283
10.2.5	Interrupt-Handler im Anwendungsprogramm	284
10.2.6	Code-Ausführung auf der Anwendungsebene	286
10.2.7	Code-Ausführung auf der Kernel-Ebene	287
10.2.8	Kommunikation mit der Anwendung	290
10.2.9	High-speed Datenaustausch über „shared memory“	291
10.2.10	Genaue Zeitmessungen	292
10.2.11	Serielle Kommunikation	292
10.2.12	Praxis-Erfahrungen	294
10.3	Integrierte Software-Lösungen mit dem Interbus: PC Worx	296
10.3.1	Applikationsanalyse	297
10.3.2	Programmierung	299
10.3.3	Inbetriebnahme und Wartung	301
10.3.4	Investitionssicherheit durch intelligente Software	303
10.3.5	Ein PC basiertes Steuerungssystem zur gravimetrischen Dosierung	303
10.4	OPC als Basis für PC-integrierte Automation	307
10.4.1	OPC-Items	308
10.4.2	Item-konfiguration des OPC-Servers	308
10.4.3	Projektierung mit PROCON WIN	309
10.4.4	Funktion des OPC-Browsers von PROCON WIN	311
10.5	Motion-Control mit dem IPC	312
10.5.1	Konfigurationsebene: Ein Systemmanager	314
10.5.2	Leistungsvergleich	315
10.5.3	Praxisbeispiel: 19-Achsen-Bearbeitungszentrum	316
10.5.4	Praxisbeispiel: Mehrachssteuerung für Fa. Egger, St. Johann, Österreich	317
10.6	Beckhoff TwinCat – SPS und NC unter Windows NT	321
10.6.1	Echtzeit ohne Hardwarezusatz als Systembasis	324
10.6.2	Betriebsverhalten für die Praxis	326
10.6.3	SPS und NC als Softwaregeräte	329
10.6.4	Interface zu Windows-Applikationen	330
10.6.5	Verbindung per Message Routing	331
11	Anhang	333

Autorenbiographie	333
Literatur	337
Sachverzeichnis	341