

Inhalt

Vorwort.....	X
1 Klassische Petri-Netz-Theorie	13
1.1 Entstehung und Anwendung	13
1.2 Was ist ein Petri-Netz?.....	14
1.2.1 Begriffe und Eigenschaften.....	17
1.2.2 Wandernde Marken oder Markenspiel.....	19
1.2.3 Darstellungsarten eines Petri-Netzes.....	30
1.2.4 Faltung	33
1.2.5 Vergröberung und Verfeinerung	34
1.3 Netzklassen	35
1.3.1 Petri-Netz-Klassen mit speziellen strukturellen Eigenschaften.....	37
1.4 Petri-Netz-Theorie am Beispiel Verkaufsautomat	40
1.4.1 Grundverhalten des Systems	41
1.4.2 Ausnahmeverhalten.....	42
1.4.3 Überarbeitetes Modell.....	43
1.4.4 Umgruppierung	44
2 Petrinetze	47
2.1 Petrinetz-Eigenschaften – Überblick und Beispiel.....	47
2.2 Sprachbetrachtung.....	48
2.3 Ablaufsprachen (SFC und Petrinetze).....	49
2.3.1 Ablauforientierte Programmierung, IEC-61131-konform.....	49
2.3.2 Ablauforientiert; Mode oder hilfreich?	50
2.3.3 Verknüpfungsorientierte Methoden [2].....	51
2.3.4 Ablauforientierte Methoden [2]	52
2.3.5 Was bietet die IEC 61131?	52
2.3.6 Petrinetze, eine Alternative zur Ablaufsprache nach IEC 61131-3 [2]	55
2.3.7 Eigenschaften Petrinetz-basierter Steuerungen	59
2.4 Petrinetze und SPS	61
2.4.1 Arbeitsweise der SPS	61
2.4.2 Moderne Entwicklungssysteme	63
2.4.3 SPS-Speicherorganisation	64
2.5 Steuerungstechnische Interpretation	65

2.5.1	Konfliktlösung	67
2.5.2	Forderung nach Sicherheit	68
2.5.3	Umgehung der Schaltregel	68
2.5.4	Eingänge (SPS bzw. POE)	69
2.5.5	Ausgänge (SPS bzw. POE)	72
2.5.6	Faltung	75
2.5.7	AWL, ST, FUP und Hochsprachen im PN	78
2.5.8	Codierungsvarianten	79
2.6	Knoten	82
2.6.1	Platz	82
2.6.2	Transition	93
2.7	Kanten	124
2.7.1	Standardkante	129
2.7.2	Sichere Kante	132
2.7.3	Testkante	135
2.7.4	Negierte Kante oder Inhibitorkante	138
2.7.5	Flanken erkennende Kante	140
2.7.6	Eingang	145
2.7.7	Negierter Eingang	148
2.7.8	Abräumkante	150
2.7.9	Datenkanten	153
2.8	Task	157
2.8.1	Allgemeine Task-Merkmale	158
2.8.2	Freilaufende Tasks	159
2.8.3	Platzabhängige Task	161
2.8.4	Triggertask	166
2.9	FUP-Elemente	169
2.9.1	FUP-Elemente	170
2.9.2	AND (Bit und Bitfolge)	172
2.9.3	OR (Bit und Bitfolge)	173
2.9.4	AND_MIX (Bitfolgen und Bit gemischt verknüpft)	174
2.9.5	AND_OR (Bitfolge)	175
2.9.6	AND_GOLB (Bitfolgen und Bit gemischt verknüpft)	177
2.9.7	AND_OR_MIX (Bit und Bitfolge)	178
2.10	Marken	179
2.11	Netzinitialisierung	179
2.11.1	Initialisierungsarten	182
2.11.2	Betriebskopf	186
2.11.3	Einfacher Bk	190
2.12	Softwarebausteine (POE) und Petrinetze	190
2.12.1	POE-Struktur für die Netz-Modell-Codierung	190
2.12.2	POE-Schnittstellen	191

3	Anwendungen und Beispiele	194
3.1	Taktgenerator	194
3.2	Einfacher Taktgenerator.....	195
3.3	Zyklischer Ressourcentausch	196
3.4	FIFO-Speicher als Markenpuffer	199
3.5	FIFO-Speicher für allgemeine Daten	200
3.6	Sammelstörmeldung	202
3.7	Positionierung mittels Flankenerkennung.....	204
3.8	Grenzwertüberwachung	205
3.9	Raumüberwachung	207
3.10	Laufkatze	210
3.10.1	Lösung	211
3.10.2	Betriebskopf und Taktgeneratoren.....	215
3.10.3	Antriebssteuerung.....	215
3.10.4	Hub der Laufkatze.....	216
3.10.5	Logik-Task.....	216
3.11	Zielwahl	216
3.12	Gestaffelter Anlauf von Antrieben.....	217
3.13	Fahrstuhlsteuerung.....	219
3.14	Ampelsteuerung	221
3.14.1	Technologieschema Ampelsteuerung	222
3.14.2	Lösungsansatz.....	223
3.14.3	Datenaufbereitung.....	224
3.14.4	Kontrollnetz	225
3.15	Kübelsteuerung	229
3.15.1	Technologieschema.....	229
3.15.2	Ein- und Ausgangsbelegung	230
3.15.3	Netzstruktur	231
3.15.4	Betriebskopf.....	232
3.15.5	Synchronisation.....	233
3.15.6	Zielfahrt	234
3.15.7	Prozessabbild	237
3.15.8	Blinkanzeige	239
3.15.9	Abschließendes zum Programmentwurf	240
3.16	Späne-Absaugung	242

3.16.1	Anforderung.....	242
3.16.2	Lösung	243
3.17	Regalsteuerung.....	249
3.17.1	Technologieschema.....	249
3.17.2	Festlegung der Eingangs- und Ausgangsbelegung.....	250
3.17.3	Betriebskopf.....	251
3.17.4	Ganganwahl	251
3.17.5	Richtungsentscheidung und benötigte Antriebe.....	251
3.17.6	Antriebssteuerung	255
3.17.7	Verbesserung der Regalsteuerung.....	258
3.17.8	Benötigte Antriebe (richtungsabhängig)	259
3.17.9	Antriebssteuerung (zwei Richtungen simultan)	262
4	Methoden und Anwendungen	265
4.1	Programmstrukturen	265
4.1.1	Hierarchisch strukturierte Programmierung.....	267
4.1.2	Entwicklungszyklus	268
4.1.3	Hierarchisches Betriebsartenmodell.....	276
4.1.4	Hierarchien in Petrinetzen.....	279
4.1.5	Ablaufgraphen und Funktionseinheiten (AG und FE)	282
4.2	Synchronisationsmechanismen	285
4.2.1	Einfache Synchronisationsmechanismen	286
4.2.2	Schlüsselplätze	290
4.2.3	Schlüsselplätze als Wegweiser.....	291
4.2.4	Intelligente Schlüsselplätze (Ressourcen).....	293
4.3	Fehlerbehandlung.....	319
4.3.1	Fehlerarten	322
4.3.2	Planung der Fehlerbehandlung.....	323
4.3.3	Systemreaktion.....	324
4.3.4	Basis der Fehlererkennung und diagnostische Potentiale	325
4.3.5	Prozessfehlererkennung	330
4.3.6	Fehlererkennung im Petrinetz der Kübelsteuerung.....	331
5	Optimierung.....	338
5.1	Optimierung, was ist das?	338
5.2	Allgemeines zur optimierten Codeerzeugung	340
5.2.1	Geschwindigkeitsoptimierung	340
5.2.2	Kompakter Code, Minimierung des Datenspeichers.....	344
5.2.3	Kommunikationsverhalten verbessern	345
5.2.4	Automatische Optimierung	346
5.3	Optimierung für ablauf- und zustandsorientierte Modelle	348

5.3.1	Grundsätzliche Optimierungsprinzipien	349
5.3.2	Kennzahlen	351
5.3.3	Geeignete Strategien für Steuerungssysteme	354
5.3.4	SFC-Optimierung.....	377
5.3.5	Petrinetz-Code-Optimierung durch Faltung.....	378
5.4	Schlussbemerkung	379
6	Anhang	381
6.1	Literatur	381
6.2	In der Source benutzte Bausteine.....	383
6.3	Codierung von Bitfolgekonstrukten.....	385
6.4	Codierungsvarianten von Bit-Netzen.....	388
6.5	Übergang vom Bit- zum Bitfolgenetz	390
6.6	Übergang vom Bitfolge- zum Bit-Netz.....	391
6.7	Rotationsregister	392
6.8	Bitfolgecodierte Kette.....	393
6.9	Transparenz- und Performance-Steigerung.....	395
6.10	AWL-Befehlssatz (IEC 61131) [2]	396
6.11	Warnhinweis	398
6.12	Glossar	399
	Stichwortverzeichnis.....	417