
Inhaltsverzeichnis

1	Grundlagen sicherheitsgerichteter Echtzeitsysteme	1
1.1	Entwicklungsstand sicherheitsgerichteter Echtzeitsysteme	2
1.2	Eigenschaften festverdrahteter und rechnergestützter Steuerungstechnik	4
1.3	Sicherheit und Zuverlässigkeit in der Prozessautomatisierung	6
1.4	Problemstellung sicherer rechnergestützter Prozessautomatisierung	7
1.5	Zuverlässigkeit, Sicherheit und Wirtschaftlichkeit	9
1.6	Unstetigkeit als inhärentes Risiko informationstechnischer Systeme	12
1.7	Spezielle Anforderungen des Echtzeitbetriebs	14
1.8	Einfachheit als Entwurfsprinzip	15
1.9	Sicherheitsnormen und -vorschriften	16
1.10	Ursachen und Auswirkungen von Fehlern und Ausfällen	17
1.10.1	Fehlerursachen	18
1.10.2	Hardware-Fehler	18
1.10.3	Software-Fehler	18
1.10.4	Fehlerauswirkungen	20
1.10.5	Fehlerklassifizierung	20
1.11	Allgemeines Prinzip der Fehlererkennung	22
1.11.1	Fehlererkennung durch Plausibilitätsprüfung	22
1.11.2	Fehlererkennung durch Vergleich	23
1.12	Diversität	23
1.12.1	Diversitätsarten	24
1.12.2	Fehlererkennbarkeit durch Diversität	26
	Literatur	26
2	Konzepte zur sicherheitsgerichteten Prozessautomatisierung	29
2.1	Maßnahmen zur Erzielung von Sicherheit in der Prozessautomatisierung	29
2.1.1	Ausschluss von Fehlern und Ausfällen	30
2.1.2	Verminderung der Wahrscheinlichkeit von Fehlern und Ausfällen	30
2.1.3	Beeinflussung der Auswirkung von Fehlern und Ausfällen	30
2.1.4	Implementierungsmöglichkeiten	31

2.2	Festverdrahtete Elektronik in der sicherheitsgerichteten Prozessautomatisierung	32
2.2.1	Regeln der Technik	32
2.2.2	Besonderheiten der Elektronik im Vergleich zu älteren Steuerungstechniken	34
2.2.3	Sicherheitskonzepte	36
2.2.4	Sicherheitsgerichtete verdrahtungsprogrammierbare Steuerungen	42
2.3	Einkanalige sicherheitsgerichtete Prozessdatenverarbeitung	43
2.4	Zweikanalige sicherheitsgerichtete Prozessdatenverarbeitung	43
2.4.1	Fühler und Stellglieder	44
2.4.2	Lichtsignale	45
2.4.3	Ventile	46
2.5	Mehrkanalige sicherheitsgerichtete Prozessdatenverarbeitung	46
2.6	Systemstrukturen	47
2.6.1	Zweikanalige Systemstrukturen	48
2.6.2	Verteilte Systemstruktur	51
2.6.3	Struktur von Mensch-Maschine-Systemen	51
	Literatur	53
3	Hardware-Systeme zur sicheren Prozessdatenverarbeitung	55
3.1	Einkanalige sicherheitsgerichtete SPSen	56
3.1.1	Gegenüberstellung sicherheitsgerichteter VPS und SPS	56
3.1.2	Baumusterprüfung und Anlagenabnahme	57
3.1.3	Forderung der Normen an die SPS-Systemstruktur	58
3.1.4	Aufbau sicherheitsgerichteter SPSen	58
3.1.5	Tests in sicherheitsgerichteten SPSen	60
3.1.6	Programmierung sicherheitsgerichteter SPSen	62
3.2	Zweikanalige Hardware-Systeme	63
3.2.1	Realisierung mit Elektronik	63
3.2.2	Realisierung mit Mikroelektronik	70
	Literatur	73
4	Zweikanalige sicherheitsgerichtete Rechnersysteme	75
4.1	Das System SIMIS	75
4.2	Das System LOGISAFE	76
4.3	Das System LOGISIRE	78
4.3.1	Maßnahmen zur Gewährleistung von Sicherheit	79
4.3.2	Hard- und Software-Struktur des LOGISIRE	80
4.3.3	Detaillierte Beschreibung der Funktionen des LOGISIRE-Betriebssystems	84
4.3.4	Fazit	86
4.4	Das System DIMI	87

4.4.1	Hardware-Strukturen	88
4.5	Erprobung des Systems DIMI	90
4.5.1	Hardware	92
4.5.2	Ausfallsicherheitsgerichtetes Verhalten	94
4.5.3	Der Vergleich	95
4.5.4	Der technische Prozess	97
4.5.5	Die Steuerung	97
4.5.6	Ergebnisse der praktischen Erprobung	100
	Literatur	102
5	Entwicklung sicherheitsgerichteter Software	103
5.1	Systementwurf sicherheitsgerichteter Software	104
5.1.1	Fehlervermeidung	104
5.1.2	Fehlertoleranz	107
5.2	Qualitätssicherung von Software	110
5.2.1	Maßnahmen zur Software-Qualitätssicherung	110
5.2.2	Planung der Software-Qualitätssicherung	111
5.2.3	Struktur von Entwicklungsprojekten	114
5.2.4	Software-Anforderungsspezifikation	114
5.3	Ein Werkzeug zur Anforderungsspezifikation	115
5.3.1	Anforderungserfassung	116
5.3.2	Systementwurf	117
5.3.3	Projekt- und Konfigurationsverwaltung, Qualitätssicherung	118
5.3.4	Bewertung	119
5.4	Prinzipien des Programmentwurfs und der Programmcodierung	120
5.5	Software-Diversität	121
5.5.1	Vollständige Diversität	123
5.5.2	Gezielte Diversität	124
5.5.3	Übersetzerdiversität	125
5.5.4	Diversitäre Implementation	127
5.5.5	Diversitäre Spezifikation	128
5.5.6	Funktionelle Diversität	128
5.5.7	Zur Anwendung der Diversitätsarten	130
5.5.8	Mehrkanalige Software-Realisierung	130
	Literatur	131
6	Software-Verifikation	133
6.1	Prinzipien der Software-Verifikation	133
6.1.1	Verifikationsplan	134
6.1.2	Verifikationstechniken	134
6.1.3	Anforderungsverifikation	137
6.1.4	Entwurfsverifikation	137

6.1.5	Modul- und Codeverifikation	138
6.1.6	Integrationsverifikation von Hard- und Software	139
6.1.7	Rechensystemvalidierung	139
6.2	Ausgewählte Software-Verifikationstechniken	140
6.2.1	Begutachtungen und Revisionen	140
6.2.2	Strukturiertes Nachvollziehen und Inspektionen	144
6.2.3	Software-Tests	146
6.2.4	Diversitäre Rückwärtsanalyse	150
6.3	Validierung von Echtzeitsystemen	153
6.3.1	Ereignissimulation	154
6.3.2	Simulation externer Umgebungen und Ausgabeverifikation	155
	Literatur	160
7	Quantitative Bewertung sicherheitsgerichteter Echtzeitsysteme	161
7.1	Bewertungsgrundlagen	161
7.1.1	Fehler und Ausfälle in zweikanaligen Systemen	162
7.2	Bewertung identischer Kanäle hinsichtlich gefährlicher Ausfallarten ...	163
7.2.1	Mittlere Zeit bis zum sicherheitsbezogenen (gefährlichen) Doppelausfall	164
7.2.2	Beispiel: 2-aus-3-Wertungsschaltung	166
7.3	Bewertung identischer Kanäle hinsichtlich gefährlicher Fehlerarten ...	171
7.4	Bewertung diversitärer Kanäle hinsichtlich gefährlicher Ausfallarten ...	172
7.4.1	Beispiel: Diversitäre Implementierung der 2-aus-3-Wertungsschaltung	172
7.4.2	Mittlere Zeit bis zur Ausgabe sicherheitsbezogener (gefährlicher, fehlerhafter) Werte	176
7.4.3	Berechnung der MTDS für obiges Beispiel	177
7.4.4	Verbesserung der Ausfallerkennbarkeit durch Vergleich von Zwischenergebnissen	177
7.4.5	Zusammenstellung der Ergebnisse aus den Beispielen	178
7.4.6	Bewertung von Software-Diversität	180
7.5	Bewertung diversitärer Kanäle hinsichtlich gefährlicher Fehlerarten ...	181
7.5.1	Ein-Bit-Vergleich	181
7.5.2	Beispiel: Software-Implementierung der 2-aus-3-Wertungs- schaltung	183
7.5.3	Mehr-Bit-Vergleich	184
7.5.4	Beispiel: Zwei-Bit-Vergleich	187
7.5.5	Beispiel: Vier-Byte-Vergleich	189
7.5.6	Vergleich von Analogwerten	191
7.6	Bedeutung der Eingabewerte	191
	Literatur	192

8	Das inhärent sichere Funktionsplanparadigma	193
8.1	Architektur und Betriebsart speicherprogrammierbarer Steuerungen	193
8.2	Programmiersprachen und Programmentwicklung	198
8.2.1	Allgemeine Merkmale der IEC-Sprachen	199
8.2.2	Anweisungsliste	202
8.2.3	Kontaktplan	203
8.2.4	Strukturierter Text	203
8.2.5	Funktionsplan	203
8.2.6	Sequentieller Ablaufplan	206
8.2.7	Anwendungsbereich höherer graphischer und textueller Sprachen	210
8.3	Anwendungsspezifische Programmobjekte	211
8.3.1	Automatisierung chemischer Prozesse	212
8.3.2	Notabschaltssysteme	216
8.4	Funktionspläne mit verifizierten Bibliotheken	219
8.5	Sicherheitstechnische Abnahme von Funktionsplänen	221
	Literatur	224
9	Erstellung und Prüfung sicherheitsgerichteter Software	227
9.1	Grundlegende Methoden der Software-Qualitätssicherung	228
9.2	Qualitätssicherung der Dokumentation	229
9.3	Qualitätssicherung von Programmen	231
9.3.1	Inspektion von Programmen	232
9.3.2	Verifikation von Programmen	233
9.3.3	Symbolische Ausführung von Programmen	234
9.3.4	Test von Programmen	235
9.4	Industrielle Prüfung der Software von Prozessautomatisierungssystemen	236
9.4.1	Grundlagen der Prüfung von Software	236
9.4.2	Software-Typprüfung der Funktionen von Prozessleitsystemen	239
9.4.3	Automatische Dialogprüfung	241
9.4.4	Automatische Prüfung der Verarbeitung	243
9.4.5	Automatische Messung der Rechnerleistung	246
9.4.6	Erfahrungen	248
9.4.7	Weiterentwicklung	249
9.5	Richtlinien zur Erstellung sicherheitsgerichteter Software	250
9.5.1	Details von Software-Anforderungsspezifikationen	251
9.5.2	Entwurfsprozeduren	256
9.5.3	Software-Struktur	257
9.5.4	Selbstüberwachung	259
9.5.5	Entwurf und Codierung im Detail	260
9.5.6	Sprachabhängige Empfehlungen	262
9.5.7	Sprache und Übersetzer	263
9.5.8	Systematische Testmethoden	265

9.5.9 Hardware-Erwägungen	266
Literatur	266
10 Einige formale Methoden zur Programmverifikation	269
10.1 Analytische Verifikation mit Vor- und Nachbedingungen	270
10.2 Ausdrücke, Anweisungen und Beweisregeln	271
10.2.1 Syntax und Semantik	271
10.2.2 Variablen und Umgebungen	271
10.2.3 Auswertung von Ausdrücken	272
10.2.4 Ausführung einer Wertzuweisung	272
10.2.5 Die Null-Anweisung SKIP	273
10.2.6 Ausführung einer Anweisungsfolge	274
10.2.7 Ausführung einer IF-Anweisung	274
10.2.8 Ausführung einer WHILE-Schleife	274
10.3 Beweisregeln	275
10.3.1 Stärkung einer Vorbedingung, Schwächung der Nachbedingung ..	275
10.3.2 Wertzuweisungen	276
10.3.3 Verzweigungen	278
10.3.4 Anweisungsfolge	278
10.3.5 Schleifen	279
10.3.6 Beispiel: Multiplikation natürlicher Zahlen	281
10.3.7 Beispiel: Effiziente Multiplikation	283
10.4 Symbolische Ausführung von Programmen	284
10.4.1 Systematisierung	287
10.4.2 Anmerkungen zur symbolischen Ausführung	289
10.4.3 Beispiele	289
10.5 Korrektheitsbeweis eines Zeitgebers	295
10.5.1 Spezifikation	296
10.5.2 Hilfssätze	297
10.5.3 Beweis	301
10.6 Werkzeuge zur Programmverifikation	304
Literatur	306
11 Eine funktionsplanabbildende Prozessrechnerarchitektur	307
11.1 Anforderungen an die Rechnerarchitektur	308
11.2 Informationsverarbeitung	311
11.2.1 Abbildung natürlicher Systemstrukturen in der Rechnerarchitektur	311
11.2.2 Erweiterbarkeit von Rechanlagen	314
11.2.3 Parallelität zur Erhöhung der Rechenleistung	315
11.2.4 Modularität und Parallelität	321
11.2.5 Kommunikation	322

11.2.6	Aspekte des Zeitverhaltens	324
11.2.7	Betriebssystem	325
11.2.8	Regeln zum Anpassen eines Rechners an ein automatisierungs- technisches Problem	326
11.2.9	Software-Konzeption	328
11.3	Schnittstellen zu Sensoren und Aktoren	329
11.3.1	Anwendungsspezifische gegenüber Standardmodulen	330
11.3.2	Sensor-/Aktoranbindung auf der Signalebene	331
11.3.3	Erweiterbarkeit und Sensor-/Aktoranbindung auf der Busebene	338
11.3.4	Sensor-/Aktoranbindung auf der Prozessorebene	340
11.3.5	Feldbusse und sensorlokale Prozessoren	341
11.4	Ein Einsatzbeispiel aus der Robotik	341
11.4.1	Gesamtstruktur des Steuerungssystems der Karlsruher Hand	341
11.4.2	Hardware-Komponenten der Rechenanlage der Karlsruher Hand	344
	Literatur	346
12	Fallstudien sicherheitsgerichteter programmierbarer elektronischer Systeme	347
12.1	Ein leicht verifizierbares ausfallsicherheitsgerichtetes PES	348
12.1.1	Fuzzy-Logik als Entwurfsprinzip eines sicherheitsgerichteten PES	348
12.1.2	Ursache-/Wirkungstabellen	349
12.1.3	Eine auf Fuzzy-Logik beruhende programmierbare Steuerung	350
12.1.4	Sicherheitsaspekte	354
12.2	Architektur einer sicherheitstechnisch abnehmbaren SPS	355
12.2.1	Hardware-Architektur	357
12.2.2	Software-Verifikation	364
12.2.3	Einige Anmerkungen	369
12.3	Eine anwendungsorientierte asymmetrische Mehrprozessorarchitektur	370
12.3.1	Das Architekturkonzept	371
12.3.2	Die Ereigniserfassungsschicht	373
12.3.3	Die Primärreaktionsschicht	374
12.3.4	Die Sekundärreaktionsschicht	375
12.3.5	Bewertung	376
12.3.6	Anwendungen	376
12.4	Zeitgenau arbeitende Prozessperipherie	377
12.4.1	Notwendige Funktionen und ihr Aufruf in PEARL	378
12.4.2	Implementierung der Hardware-Unterstützung	379
12.4.3	Betriebssystemunterstützung	382
	Literatur	383

13	Unterbrechungsfreie asynchrone Echtzeitverarbeitung	
	mit Zustandswiederherstellung zur Laufzeit	385
13.1	Rechenprozessverarbeitung ohne asynchrone Unterbrechungen	385
13.1.1	Paradigmen des Echtzeitbetriebs	386
13.1.2	Entwurfsprobleme	394
13.1.3	Lösungskonzept	396
13.1.4	Ausführungsbeispiel	412
13.2	Neuaufsetzen im laufenden Betrieb redundant arbeitender Prozessoren	425
13.2.1	Neuaufsetzen von Echtzeitsystemen	425
13.2.2	Geräte Technische Sortierung von Datenwörtern nach Altersklassen	435
13.2.3	Beispielhafter Aufbau der Schaltungskomponenten	446
	Literatur	457
14	Ein sicherheitsgerichteter Feldbus	459
14.1	Feldbusse	460
14.2	Signalcodierung	463
14.2.1	Fehlerreduktion durch Signalcodierung	463
14.2.2	Signalcodierung mit modifiziertem FSK-Verfahren	464
14.2.3	Detektierung signalcodierter Datenbits	469
14.2.4	Detektierung von Synchron- und Statussignalen	477
14.2.5	Aufbau des signalangepassten Filters	479
14.2.6	Aufbau der Ein- und Ausgangsstufen	484
14.3	Datencodierung	484
14.3.1	Datencodierung von Nibbles	485
14.3.2	Berechnung der Korrekturstellen	486
14.3.3	Hamming-Codierung von Daten-Nibbles	488
14.3.4	Decodierung gesicherter Daten-Nibbles	490
14.3.5	Fehlerwahrscheinlichkeiten	493
14.3.6	Vergleich mit marktüblichen Feldbussystemen	498
14.4	Zeitsynchronisierung auf Ringbussen	500
14.4.1	Zeitmessung auf einem Doppelringbus	500
14.4.2	Zeitmessung auf einem Einzelringbus	504
14.4.3	Bearbeitung zeitabhängiger Aufträge	508
14.4.4	Synchronisierung der Slave-Bausteine	509
14.5	Doppelringbus	512
14.5.1	Steuerung	513
14.5.2	Kommunikationskanäle	514
14.6	Summenrahmentelegramm	516
14.6.1	Signalcodierung bestimmter Telegrammabschnitte	517
14.6.2	Aufbau der Slave-Bausteine	518
14.6.3	Synchronisation der Slave-Bausteine	519

14.6.4	Prioritätssteuerung der Slave-Bausteine	520
14.6.5	Übertragungszeit des Summenrahmentelegramms	521
14.6.6	Vergleich mit marktüblichen Feldbussystemen	522
	Literatur	527
15	Die sicherheitsgerichtete Echtzeitprogrammiersprache HI-PEARL	529
15.1	PEARL als Spezifikationssprache	529
15.1.1	Zur Rolle von Spezifikationen in der Programmentwicklung	530
15.1.2	Spezifikationskonzepte in PEARL	532
15.1.3	Beispiel	534
15.1.4	Fazit	535
15.2	HI-PEARL	536
15.2.1	Überblick über PEARL	536
15.2.2	Motivation zur Definition von HI-PEARL	538
15.2.3	Mehrrechner-PEARL	539
15.2.4	Zur Entwicklung fehlertoleranter und robuster Echtzeitprogramme erforderliche Funktionalitäten	541
15.3	Sichere Sprachmittel zur Formulierung von Ablaufplänen	561
	Literatur	563
16	Ablaufplanung und Zuteilbarkeitsanalyse für den Mehrprozessbetrieb	565
16.1	Graphisches Programmieren im Großen	566
16.2	Zeitsynchrone Zuteilung	567
16.3	Terminbezogene Prozessorzuteilung	568
16.3.1	Struktureigenschaften des Antwortzeitalgorithmus	572
16.3.2	Hinreichende Bedingungen der zeitgerechten Ausführbarkeit unter Beachtung von Betriebsmittelreservierungen	574
16.3.3	Nicht präemptive Antwortzeit zuteilung	578
16.3.4	Vermeidung von Kontextumschaltungen ohne Verletzung zeitgerechter Ausführbarkeit	579
16.3.5	Überlastvermeidung durch lastadaptive dynamische Zuteilung	581
16.4	Zuteilbarkeitsanalyse von Echtzeitsystemen	583
16.4.1	Zuteilbarkeitsanalysierbare Echtzeitprogrammiersprachen	585
16.4.2	HI-PEARL: Die zuteilbarkeitsanalysierbare Version von PEARL	586
16.4.3	Ein Zuteilbarkeitsanalysator für HI-PEARL	587
16.4.4	Ausblick	593
	Literatur	594
	Sachverzeichnis	597