

Informatik-Fachberichte 160

Herausgegeben von W. Brauer
im Auftrag der Gesellschaft für Informatik (GI)

Hubert Mäncher

Fehlertolerante dezentrale Prozeßautomatisierung



Springer-Verlag
Berlin Heidelberg New York
London Paris Tokyo

Autor

Hubert Mäncher
MAGNUM GmbH
Kiesstraße 63, D-6100 Darmstadt

CR Subject Classifications (1987): C.2.4, C.3, D.4.5, D.4.7, J.7

ISBN-13:978-3-540-18754-7

e-ISBN-13:978-3-642-73324-6

DOI: 10.1007/978-3-642-73324-6

CIP-Titelaufnahme der Deutschen Bibliothek.

Mäncher, Hubert:

Fehlertolerante dezentrale Prozessautomatisierung / Hubert Mäncher. – Berlin; Heidelberg; New York; Tokyo: Springer, 1987

(Informatik-Fachberichte; 160)

Zugl.: Darmstadt, Techn. Hochsch., Diss. u. d. T.: Mäncher, Hubert: Ein fehlertolerantes Mikrorechnersystem für die dezentrale Prozessautomatisierung

ISBN-13:978-3-540-18754-7

NE: GT

Dieses Werk ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere die der Übersetzung, des Nachdrucks, des Vortrags, der Entnahme von Abbildungen und Tabellen, der Funksendung, der Mikroverfilmung oder der Vervielfältigung auf anderen Wegen und der Speicherung in Datenverarbeitungsanlagen, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Eine Vervielfältigung dieses Werkes oder von Teilen dieses Werkes ist auch im Einzelfall nur in den Grenzen der gesetzlichen Bestimmungen des Urheberrechtsgesetzes der Bundesrepublik Deutschland vom 9. September 1965 in der Fassung vom 24. Juni 1985 zulässig. Sie ist grundsätzlich vergütungspflichtig. Zuwiderhandlungen unterliegen den Strafbestimmungen des Urheberrechtsgesetzes.

© by Springer-Verlag Berlin Heidelberg 1987

VORWORT

Das vorliegende Buch gibt die Darmstädter Dissertation (D17) "Ein fehlertolerantes Mikrorechnersystem für die dezentrale Prozeßautomatisierung" mit einer erweiterten Einführung und einem verkürzten Anhang wieder. Sie entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter bei Herrn Prof. Dr.-Ing. R. Isermann am Institut für Regelungstechnik (Fachgebiet Regelsystemtechnik und Prozeßlenkung) an der Technischen Hochschule Darmstadt.

Mein besonderer Dank gilt daher Herrn Prof. Dr.-Ing. R. Isermann für die Anregung zu dieser Arbeit, für die Gelegenheit zu ihrer Durchführung und für die stete Förderung.

Herrn Prof. Dr.rer.nat. M. Glesner danke ich für das der Arbeit entgegengebrachte Interesse und die Übernahme des Korreferats.

Mein weiterer Dank gilt allen Kollegen für die anregenden Diskussionen und kritischen Anmerkungen zu dieser Arbeit, der Institutswerkstatt und zahlreichen Studenten für ihre Hilfe beim Aufbau des Experimentiersystems.

Schließlich danke ich auch meiner Familie, die mich oft entbehren mußte, für ihre Geduld und Rücksichtnahme.

Die finanziellen Mittel für die Durchführung dieser Arbeit stellte die Deutsche Forschungsgemeinschaft im Rahmen des Schwerpunktprogrammes "Technische Grundsatzfragen beim Einsatz von Mikroprozessoren" (Forschungsvorhaben Is 14/28 und Is 14/30) zur Verfügung. Auch für diese Unterstützung sei gedankt.

KURZFASSUNG

Es werden Verfahren zur Implementierung von Fehlertoleranz in dezentralen Automatisierungssystemen untersucht. Ausgehend von einer Analyse der angestrebten Ziele Zuverlässigkeit, Sicherheit und Wirtschaftlichkeit wird eine modifizierte Fehlertoleranz-Klassifizierung angegeben. Die exemplarische Realisierung eines fehlertoleranten Mikrorechnersystems basiert auf der globalen Grundstruktur dezentraler Automatisierungssysteme, in der durch die Zusammenfassung lokaler Subsysteme eine Rekonfiguration zur Tolerierung bestimmter Hardware-Fehler möglich wird. Dezentral realisierte Vergleichs- und Votiereinrichtungen erlauben auch die Tolerierung beliebiger Einfachfehler. Das zugehörige Echtzeitbetriebssystem ist mit der für Anwendungsprogramme transparenten Kommunikation in der Lage, alle Aufträge unabhängig vom Ort des Systemaufrufs und des Zielobjektes auszuführen. Zur Unterstützung des Betriebs von Echtzeitprogrammen werden Funktionszustände zur flexiblen Anpassung an äußere Abläufe behandelt. Anwenderprogramme in Form konfigurierbarer Software-Bausteine zeigen beispielartig die Realisierung fehlertoleranter Funktionen verschiedener Klassen, die gleichzeitig betrieben werden können. Ihre Erprobung an analog simulierten Testprozessen demonstriert die Anwendbarkeit der vorgestellten Konzepte.

SCHLAGWÖRTER

Fehlertoleranz; Fehlertoleranzklassen; Zuverlässigkeit; Sicherheit; Mikrorechner; Fehlertolerantes Mikrorechnersystem; Dezentrales Automatisierungssystem; Fehlertolerantes Automatisierungssystem; Bussystem; Kommunikation; Rekonfiguration; Dezentraler Voter; Echtzeit-Betriebssystem; Mehrrechner-Betriebssystem; Fehlertolerante digitale Regelung.

ABSTRACT

This work investigates methods to implement fault-tolerance in distributed automation systems. Starting with an analysis of the intended goals reliability, security, and economy a modified fault-tolerance classification is shown. The realization of a fault-tolerant microcomputer system as an example uses the basic global structure of distributed automation systems, in which collecting local subsystems enables a system reconfiguration to tolerate particular hardware-faults. Tolerance against unspecified single faults is also achieved by distributed voting and comparing mechanisms. Using its transparent communication functions the according real-time operating system is able to perform all system services independent of the location of the calling task and the destination object. Function-states of real-time programs, supporting their flexible adaption to the outside technical process, are treated. Some user programs in form of configurable software show the realization of fault-tolerant functions in different classes, which may be used simultaneously. Their closed-loop test with analog computers demonstrates the applicability of the presented conceptions.

KEYWORDS

Fault-Tolerance; Fault-Tolerance Classification; Reliability; Security; Microcomputer; Fault-Tolerant Microcomputer System; Distributed Automation System; Fault-Tolerant Automation System; Bussystem; Communication; Reconfiguration; Distributed Voter; Real-Time Operating System; Multicomputer Operating System; Fault-Tolerant Digital Control.

INHALTSVERZEICHNIS

1. EINFÜHRUNG	1
1.1 Motivation	4
1.2 Übersicht	6
2. AUSWAHL DER FEHLERTOLERANZVERFAHREN	8
2.1 Ziele der Fehlertoleranz	10
2.1.1 Zuverlässigkeit	10
2.1.2 Sicherheit	11
2.1.3 Wirtschaftlichkeit	12
2.2 Bewertung von Fehlertoleranz	13
2.2.1 Gegenüberstellung der Begriffe und Kenngrößen	13
2.2.2 Geschlossene Darstellung der Fehlertoleranzkenngrößen	20
2.3 Klassenauswahl als Projektierungsverfahren für die Anwendung	21
2.3.1 Zuverlässigkeitsklassen für Tasks	22
2.3.2 Fehlertoleranzklassen für Funktionen	24
2.3.3 Auswahl einer Fehlertoleranzklasse	29
2.4 Realisierungsprinzipien	30
2.4.1 Industrielle Randbedingungen	30
2.4.2 Gliederung der Verfahren	32
2.4.3 Ausgewählte Verfahrenskombinationen	33
2.4.3.1 Klasse Z - Zuverlässigkeit	33
2.4.3.2 Klasse S - Sicherheit	35
2.4.3.3 Klasse W - Wirtschaftlichkeit	37
2.5 Gegenüberstellung ausgewählter Verfahren	39
3. DIE SYSTEM-HARDWARE	41
3.1 Die FIPS-Hardware-Struktur	41

3.1.1 Globale Grundstruktur dezentraler Automatisierungssysteme	42
3.1.2 Ergänzungen für die Fehlertoleranz	44
3.1.3 Systematische Gliederung und mögliche Ausbaustufen	46
3.1.3.1 Die Modulebene	46
3.1.3.2 Die Lokalsystemeebene	46
3.1.3.3 Die Regionalsystemeebene	47
3.1.3.4 Die Globalsystemeebene	49
3.2 Das Lokalsystem als Basissystem	51
3.2.1 Anforderungen der Fehlertoleranz an das Basissystem	51
3.2.2 Der Aufbau eines Lokalsystems	53
3.2.3 Die Baugruppen des Basissystems	56
3.2.3.1 Der Mikrorechner (MIC)	56
3.2.3.2 Die Speicherbaugruppe (M)	58
3.2.3.3 Das Konsolen-Interface (CI)	60
3.2.3.4 Die Analog-Ein/Ausgabe (AIO)	63
3.2.3.5 Die Binär-Ein/Ausgabe (BIO)	66
3.2.3.6 Der Globalbuskoppler (GBC)	68
3.2.3.7 Die Stromversorgung (PS)	68
3.3 Der Regionalbus als zusätzlicher Kommunikationsweg	71
3.3.1 Anforderungen an einen Kommunikationsweg	71
3.3.2 Das Prinzip des Regionalbus	72
3.3.3 Der Aufbau eines Regionalbuskopplers	73
3.3.4 Die wichtigsten Funktionen der Regionalbuskoppler	76
3.3.5 Die Nutzung für die Fehlertoleranz	79
3.4 Die Realisierung dezentraler Voter	83
3.4.1 Anforderungen an dezentrale Voter	84
3.4.2 Die Anordnung für hohe Zuverlässigkeit	85
3.4.3 Die Anordnung für sicherheitsgerichtetes Verhalten	87
3.4.4 Die Bedeutung des Schaltelements	89
3.4.5 Ein Implementierungsbeispiel	92
3.5 Die realisierte System-Hardware	94
3.5.1 Die Aufbautechnik	94
3.5.2 Der Umfang des aufgebauten Systems	94

4. DIE SYSTEM-SOFTWARE	99
4.1 Die Anforderungen an das Betriebssystem	102
4.1.1 Randbedingungen in der Automatisierungstechnik	102
4.1.2 Spezielle Anforderungen durch die Fehlertoleranz	103
4.2 Das Betriebssystem REX im Überblick	106
4.2.1 Die Systemphilosophie	106
4.2.2 Behandelte Objekte	108
4.2.3 Schichten und Oberflächen	109
4.3 Die dezentrale Datenhaltung	115
4.3.1 Die Segmentierung von Anwenderprogrammen	116
4.3.2 Regional zugängliche System-Informationen	119
4.3.3 Die Lokalsystembeschreibung	122
4.4 Die Koppel-Software	124
4.4.1 Die Funktionen der Koppel-Software	124
4.4.2 Ein Beispiel für die regionale Ausführung eines Systemaufrufs	127
4.5 Die Exekutive	129
4.5.1 Die Hierarchie in der Exekutive	129
4.5.2 Die Bearbeitung von Jobs und Tasks	131
4.5.2.1 Die unbewußten Exekutivzustände	132
4.5.2.2 Die bewußten Funktionszustände	138
4.5.3 Kommunikationsmittel für Jobs und Tasks	142
4.5.3.1 Direkte Kommunikation	143
4.5.3.2 Botschaftenorientierte Kommunikation	143
4.5.4 Die Prozeß-Ein/Ausgabe	146
4.5.4.1 Die analoge Eingabe als Beispiel	147
4.5.5 Eine Übersicht über Systemaufrufe	148
4.6 Der Konfigurator	159
4.6.1 Der Systemstart	160
4.7 Das Dialogsystem	163
4.7.1 Die Schnittstelle zum Anwenderprogramm	164
4.7.2 Die Bedienfunktion	165
4.8 Die Bibliothek	169
4.9 Die Realisierung des Betriebssystems	171

5. SPEZIELLE ENTWICKLUNGSHILFEN	174
5.1 Hilfsdateien für die Erstellung einer Task	174
5.2 Bibliotheken und zugehörige Extern-Deklarationen	177
6. FEHLERTOLERANTE REGELPROGRAMME	180
6.1 Digitale Regler in verschiedenen Fehlertoleranz- klassen	180
6.1.1 Die Software-Realisierung der Regler	183
6.1.1.1 Rekonfigurierbare Regler der Klasse W	184
6.1.1.2 Sicherheitsgerichtete Regler der Klasse S	187
6.1.1.3 Hochzuverlässige Regler der Klasse Z	190
6.1.2 Experimente an analog simulierten Prozessen	192
6.1.2.1 Die Erzeugung von Fehlern	192
6.1.2.2 Die Versuchsanordnung	193
6.1.2.3 Ein Beispielexperiment	195
6.2 Fehlertolerierende, schnelle adaptive Mehrgrößen- regelung	200
6.2.1 Der parameteradaptive Regelkreis	201
6.2.2 Die Elemente des paramateradaptiven Regelkreises	202
6.2.3 Die Realisierung in der Fehlertoleranz- klasse W	206
7. GEGENÜBERSTELLUNG DER FEHLERTOLERANZKLASSEN	212
7.1 Die tolerierbaren Fehler	212
7.2 Die Verbesserung relevanter Kenngrößen	216
7.3 Der erforderliche Aufwand	221
8. ZUSAMMENFASSUNG	226
 LITERATURVERZEICHNIS	 229
 ANHANG	 239

VERZEICHNIS DER WICHTIGSTEN FORMELZEICHEN UND ABKÜRZUNGEN

Allgemeine Sonderzeichen

\underline{A} , \underline{a}	Matrix A , Vektor a
\underline{A}^T , \underline{a}^T	transponierte Matrix \underline{A} , transponierter Vektor \underline{a}
\hat{a}	Schätzwert für a
a^*	numerischer Wert von a
Σ	Summe
Π	Produkt

Zuverlässigkeits- und Sicherheitstheorie

$F(t)$	Ausfallwahrscheinlichkeit	
$G(t)$	Gefährdungswahrscheinlichkeit	
$P(\text{Ereignis})$	Wahrscheinlichkeit von Ereignis	
t	variable Zeit	
T	feste Zeit	
U_S	Sicherheitsunverfügbarkeit	
U_Z	Unverfügbarkeit	
V_S	Sicherheitsverfügbarkeit	
V_Z	Verfügbarkeit	
MTBF	mean time between failures	mittlere ausfallfreie Zeit
MTTD	mean time to danger	mittlerer zeitlicher Abstand zw. Gefährdungen
MTTR	mean time to repair	mittlere Reparaturzeit
MTTSR	mean time to safe repair	mittlere Zeit zur Sicherheitswiederherstellung

Weitere Symbole und Abkürzungen der Zuverlässigkeits- und Sicherheitstheorie und eine ausführlichere Erklärung finden sich in Kap. 2.2.1, Tabelle 2.1.

Regelungs- und Automatisierungstechnik

a_i	Nennerparameter der Prozeß-Übertragungsfunktion
$A(z)$	Nennerpolynom der Prozeß-Übertragungsfunktion
\underline{A}	Systemmatrix des Zustandsraummodells
b_i	Zählerparameter der Prozeß-Übertragungsfunktion
$B(z)$	Zählerpolynom der Prozeß-Übertragungsfunktion
$\underline{B}, \underline{b}$	Steuermatrix (-vektor) des Zustandsraummodells
$\underline{C}, \underline{c}$	Ausgangsmatrix (-vektor) des Zustandsraummodells
d	diskrete Totzeit
e	Regelabweichung oder Vorhersagefehler
\underline{I}	Einheitsmatrix
k	diskrete Zeit $k = t/T_0$
$\underline{K}, \underline{k}$	Rückführmatrix (-vektor) des Zustandsreglers
m	Prozeßmodellordnung
p_i, q_i	Parameter eines Ein/Ausgangsreglers
t	kontinuierliche Zeit
T_0	Abtastzeit
U	Stellgröße, absolut
u	Stellgröße, arbeitspunktbezogen
$\underline{U}, \underline{u}$	Stellgrößenvektoren
U_0, \underline{U}_0	Stellgrößen-Beharrungswerte
W	Sollwert, absolut
w	Sollwert, arbeitspunktbezogen
$\underline{W}, \underline{w}$	Sollwertvektoren
W_0, \underline{W}_0	Sollwert-Beharrungswerte
X	Zustandsgröße, absolut
x	Zustandsgröße, arbeitspunktbezogen
$\underline{X}, \underline{x}$	Zustandsgrößenvektoren
X_0, \underline{X}_0	Zustandsgrößen-Beharrungswerte
Y	Regelgröße, absolut
y	Regelgröße, arbeitspunktbezogen
$\underline{Y}, \underline{y}$	Regelgrößenvektoren
Y_0, \underline{Y}_0	Regelgrößen-Beharrungswerte
z	Variable der z-Transformation

DB	Regler mit Deadbeat-Verhalten
P, PI, PID	Regler mit proportionalem (P), integrierendem (I) und differenzierendem Verhalten
ZR	Zustandsregler

Datentechnik und Informatik

A/D	analog to digital	analog in digital
CPU	central processing unit	Zentraleinheit
D/A	digital to analog	digital in analog
EPROM	erasable programmable read-only memory	löschbarer programmier- barer Festwertspeicher
INT	interrupt	Interrupt, Unterbrechung
I/O	input/output	Ein/Ausgabe
NDP	numeric data processor	Numerikprozessor
NMI	non-maskable interrupt	nicht maskierbarer Int.
PIC	programmable interrupt controller	programmierbarer Int.- Steuerungsbaustein
RAM	random access memory	Schreib/Lese-Speicher
ROM	read-only memory	Festwertspeicher

Beschreibung des fehlertoleranten Mikrorechnersystems FIPS

FIPS	Fault-tolerance Implementation in distr. Processautomation Systems	Fehlertoleranz- Implementierung in dezent. Prozeßautomatisierungs- Systemen
REX	regional executive	regionale Exekutive
AIO, A-I/O	analog input/output	analoge Ein/Ausgabe
BIO, B-I/O	binary input/output	binäre Ein/Ausgabe
CI	console interface	Konsolen-Interface
G...	global...	global...
GB	global bus	Globalbus
GBC	global bus coupler	Globalbuskopplung

GS	global system	Globalsystem
L...	local...	lokal...
LB	local bus	Lokalbus
LBC	local bus coupler	Lokalbuskopplung
LOP	local operator panel	lokale Bedienkonsole
LS	local system	Lokalsystem
M	memory	Speicher
MIC	microcomputer	Mikrorechner
PS	power supply	Stromversorgung
R...	regional...	regional...
RB	regional bus	Regionalbus
RBC	regional bus coupler	Regionalbuskopplung
RS	regional system	Regionalsystem
S	switch	Schalter
SE	special extension	spezielle Erweiterung