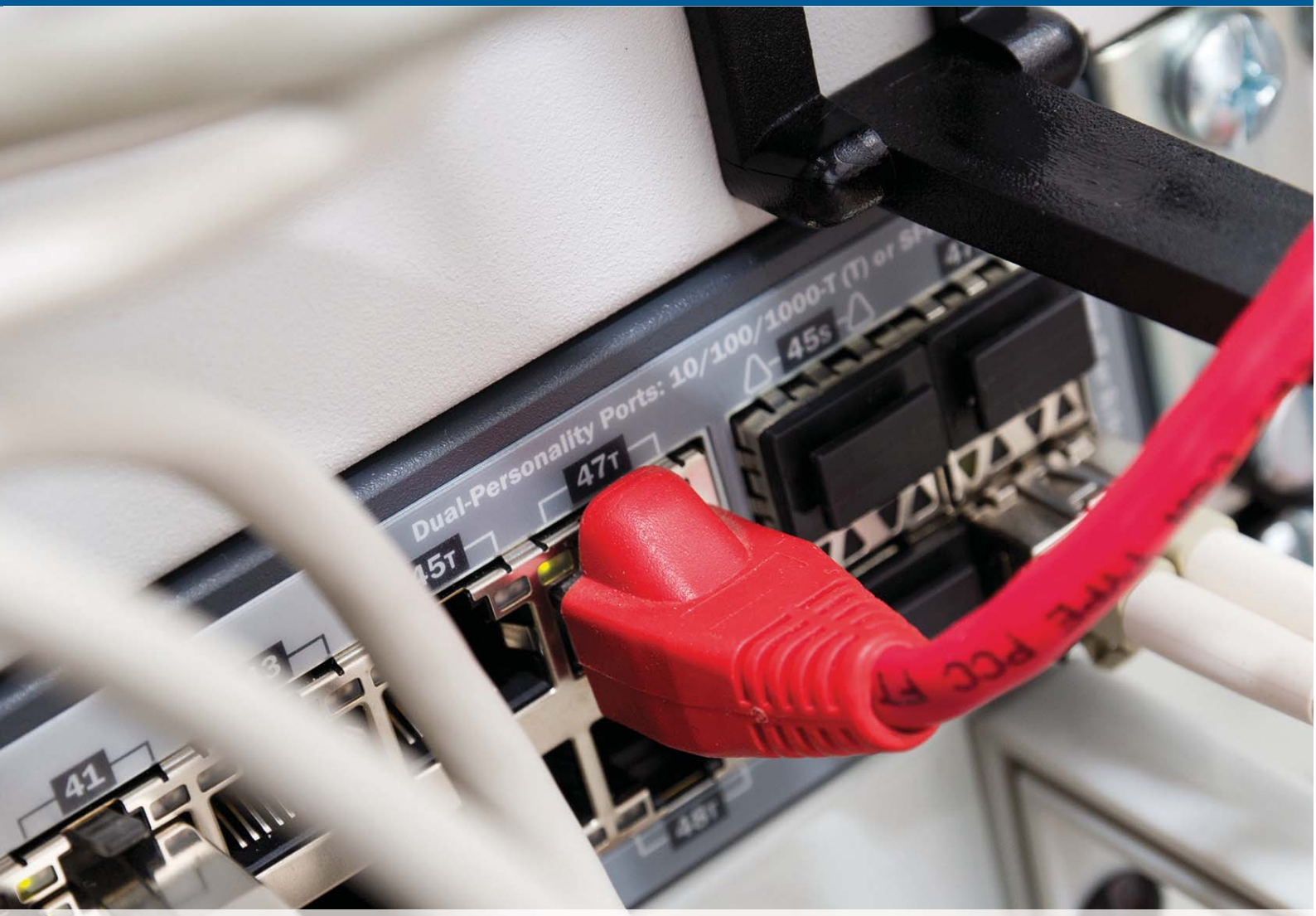
STUTTGARTER BEITRÄGE ZUR PRODUKTIONSFORSCHUNG

HANS-PETER BOCK

Fehlertolerante numerische Steuerung



Universität Stuttgart

Fraunhofer IPA

STUTTGARTER BEITRÄGE ZUR PRODUKTIONSFORSCHUNG BAND 45

Herausgeber:
Univ.-Prof. Dr.-Ing. Thomas Bauernhansl
Univ.-Prof. Dr.-Ing. Dr. h.c. mult. Alexander Verl
Univ.-Prof. a. D. Dr.-Ing. Prof. E.h. Dr.-Ing. E.h. Dr. h.c. mult. Engelbert Westkämper

Hans-Peter Bock

Fehlertolerante numerische Steuerung

FRAUNHOFER VERLAG

Kontaktadresse:
Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA, Stuttgart
Nobelstraße 12, 70569 Stuttgart
Telefon 0711 970-00, Telefax 0711 970-1399
info@ipa.fraunhofer.de, www.ipa.fraunhofer.de

STUTTGARTER BEITRÄGE ZUR PRODUKTIONSFORSCHUNG

Herausgeber:
Univ.-Prof. Dr.-Ing. Thomas Bauernhansl
Univ.-Prof. Dr.-Ing. Dr. h.c. mult. Alexander Verl
Univ.-Prof. a. D. Dr.-Ing. Prof. E.h. Dr.-Ing. E.h. Dr. h.c. mult. Engelbert Westkämper

Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA, Stuttgart
Institut für Industrielle Fertigung und Fabrikbetrieb (IFF) der Universität Stuttgart
Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW)
der Universität Stuttgart

Titelbild: © tournee – fotolia.com

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über www.dnb.de abrufbar.
ISSN: 2195-2892
ISBN (Print): 978-3-8396-0890-6

D 93

Zugl.: Stuttgart, Univ., Diss., 2014

Druck: Mediendienstleistungen des Fraunhofer-Informationszentrum Raum und Bau IRB, Stuttgart
Für den Druck des Buches wurde chlor- und säurefreies Papier verwendet.

Fraunhofer-Informationszentrum Raum und Bau IRB
Postfach 80 04 69, 70504 Stuttgart
Nobelstraße 12, 70569 Stuttgart
Telefon 0711 970-2500
Telefax 0711 970-2508
E-Mail verlag@fraunhofer.de
URL http://verlag.fraunhofer.de

GELEITWORT DER HERAUSGEBER

Produktionswissenschaftliche Forschungsfragen entstehen in der Regel im Anwendungszusammenhang, die Produktionsforschung ist also weitgehend erfahrungsbasiert. Der wissenschaftliche Anspruch der „Stuttgarter Beiträge zur Produktionsforschung" liegt unter anderem darin, Dissertation für Dissertation ein übergreifendes ganzheitliches Theoriegebäude der Produktion zu erstellen.

Die Herausgeber dieser Dissertations-Reihe leiten gemeinsam das Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA und jeweils ein Institut der Fakultät für Konstruktions-, Produktions- und Fahrzeugtechnik an der Universität Stuttgart.

Die von ihnen betreuten Dissertationen sind der marktorientierten Nachhaltigkeit verpflichtet, ihr Ansatz ist systemisch und interdisziplinär. Die Autoren bearbeiten anspruchsvolle Forschungsfragen im Spannungsfeld zwischen theoretischen Grundlagen und industrieller Anwendung.

Die „Stuttgarter Beiträge zur Produktionsforschung" ersetzt die Reihen „IPA-IAO Forschung und Praxis" (Hrsg. H.J. Warnecke / H.-J. Bullinger / E. Westkämper / D. Spath) bzw. ISW Forschung und Praxis (Hrsg. G. Stute / G. Pritschow / A. Verl). In den vergangenen Jahrzehnten sind darin über 800 Dissertationen erschienen.

Der Strukturwandel in den Industrien unseres Landes muss auch in der Forschung in einen globalen Zusammenhang gestellt werden. Der reine Fokus auf Erkenntnisgewinn ist zu eindimensional. Die „Stuttgarter Beiträge zur Produktionsforschung" zielen also darauf ab, mittelfristig Lösungen für den Markt anzubieten. Daher konzentrieren sich die Stuttgarter produktionstechnischen Institute auf das Thema ganzheitliche Produktion in den Kernindustrien Deutschlands. Die leitende Forschungsfrage der Arbeiten ist: Wie können wir nachhaltig mit einem hohen Wertschöpfungsanteil in Deutschland für einen globalen Markt produzieren?

Wir wünschen den Autoren, dass ihre „Stuttgarter Beiträge zur Produktionsforschung" in der breiten Fachwelt als substanziell wahrgenommen werden und so die Produktionsforschung weltweit voranbringen.

Alexander Verl Thomas Bauernhansl Engelbert Westkämper

Fehlertolerante numerische Steuerung

Von der Fakultät Konstruktions-, Produktions- und Fahrzeugtechnik

der Universität Stuttgart

zur Erlangung der Würde eines Doktors der

Ingenieurwissenschaften (Dr.-Ing.) genehmigte Abhandlung.

Vorgelegt von

Hans-Peter Bock

aus Bad Mergentheim.

Hauptberichter:	Univ.-Prof. Dr.-Ing. Dr. h.c. mult. Alexander Verl
Mitberichter:	Univ.-Prof. Dr.-Ing. Michael Weyrich
Tag der mündlichen Prüfung:	24. Oktober 2014

Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW)

der Universität Stuttgart

2015

Vorwort des Autors

Die vorliegende Arbeit entstand während und nach meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW) der Universität Stuttgart.

Mein besonderer Dank gilt meinem Doktorvater, Herrn Professor Dr.-Ing. Dr. h.c.mult. Alexander Verl, für die Betreuung meiner Arbeit und die Übernahme des Hauptberichts sowie für die vertrauensvolle Zusammenarbeit und abwechslungsreiche Zeit am Institut.

Herrn Professor Dr.-Ing. Michael Weyrich danke ich herzlich für die kurzfristige Bereitschaft zur Übernahme des Mitberichts. Herrn Professor Dr.-Ing. Hans-Christian Reuss danke ich für die freundliche Übernahme des Prüfungsvorsitzes.

Allen Mitarbeitern des ISW danke ich für die stets konstruktive und freundschaftliche Zusammenarbeit. Die vielen Diskussionen und Vorträge in Abteilungs- und Instituts-Kolloquien sowie auf Kongressen haben mich fachlich sowie persönlich positiv geprägt.

Für die gründliche Durchsicht meiner Arbeit und die wertvollen Anregungen gilt den Herren Dr.-Ing. Ulrich Laible, Dr.-Ing. Stefan Staudt sowie Dr.-Ing. Armin Lechler mein besonderer Dank.

Besonderers möchte ich mich bei der Industriellen Steuerungstechnik GmbH (ISG) für die gute Zusammenarbeit und wertvolle Unterstützung bedanken, insbesondere bei den Herren Dr.-Ing. Dieter Scheifele, Ulrich Eger, Roland Beh, Klaus Maier und Edmund Buchal.

Der Firma TRUMPF Werkzeugmaschinen GmbH + Co. KG, insbesondere Dr.-Ing. Gerhard Hammann und Klaus Bauer, danke ich für die Unterstützung und die flexiblen Möglichkeiten, Zeit für das Erstellen dieser Arbeit zu finden.

Meiner Oma Gertrud Waldmann, meinen Eltern, Isolde und Klaus Bock, sowie meiner gesamten Familie danke ich sehr herzlich für die kontinuierliche Unterstützung von der Schulausbildung über das Studium hinweg bis zur Promotion. Meiner Eva und allen Freunden danke ich für das Verständnis und die moralische Unterstützung während des Erstellens dieser Arbeit.

Tamm, im Januar 2015 Hans-Peter Bock

Kurzinhalt

Die andauernde Leistungssteigerung von Rechnern nach dem noch geltenden Mooreschen Gesetz ermöglicht computerbasierten numerischen Steuerungen den Einsatz in Anwendungsbereichen, die bisher allein dem Menschen vorbehalten waren. Damit werden numerische Steuerungen vermehrt auch in sicherheitskritischen Anwendungen eingesetzt wie beispielsweise der minimal invasiven Chirurgie. Dieser Anwendungsbereich stellt sehr hohe Anforderungen an die Sicherheit des Assistenzsystems. Aus diesem Grund sind diese Systeme fehlersicher ausgelegt und gewährleisten, dass ein interner Fehler des Systems einen Patienten nicht unmittelbar verletzt. Um dieser Anforderung gerecht zu werden nehmen bestehende Systeme zur Erhöhung der Sicherheit eine Reduktion der Zuverlässigkeit in Kauf. Zukünftig ist dies nicht weiter ausreichend, da bestimmte Operationstechniken bei Ausfall des Assistenzsystems nur durch einen deutlich größeren manuellen Eingriff weitergeführt werden können oder im schlechtesten Fall überhaupt nicht möglich sind. Eine Aufgabenstellung liegt hierbei darin, eine fehlertolerante numerische Steuerung zu entwerfen, welche den bestehenden Sicherheitsanforderungen gerecht wird und gleichzeitig die Zuverlässigkeit gegenüber bestehenden Systemen deutlich erhöht. Herausfordernd ist dabei, dass möglicherweise in der Steuerung auftretende Fehler zu keiner Konturverletzung der Sollbahn führen dürfen.

Diese Arbeit untersucht hierzu bestehende Ansätze zur Erhöhung der Zuverlässigkeit von Systemen sowie ein Operationsassistenzsystem, das die Sicherheit dessen numerischer Steuerung erhöht. Dabei wird der fehlersichere Ansatz des Operationsassistenzsystems um Fehlertoleranz erweitert. Das Ziel ist der Entwurf eines numerischen Steuerungssystems das durch Redundanz auf Basis

einer Mehrheitsentscheidung interne Fehler maskieren sowie fehlerhafte redundante Einheiten durch Rekonfiguration ausgliedern kann, so dass keine fehlerhaften Sollwerte ausgegeben werden. Das System weist damit nach außen hin eine fehlerfreie Funktion auf, so dass ein begonnener Bearbeitungsprozess ohne Unterbrechung zu Ende geführt werden kann.

Short summary

The continued increase in performance of computers since the inception of Moore's Law enables computer based numerical control (CNC) systems for use in applications that were previously only reserved for humans. These CNC systems are increasingly being used in safety-critical applications such as minimally invasive surgery. This application places very high demands on the safety of the assistance system. For this reason, these systems are designed to be fail-safe and to ensure that an internal error of the system will never cause a patient to be injured directly. To meet this requirement existing systems condone a decrease of reliability to increase safety. In the future this is no longer sufficient, since certain surgical techniques can be performed only by a much larger manual intervention in case of failure of the assistance system, or even not be possible in the worst case. The task is to design a fault-tolerant numerical control, which fulfills the state of the art safety requirements while increasing reliability over existing systems significantly. A subsequent challenge is that potential errors inside the numerical control must not lead to a contour violation of the set path.

This thesis examines the existing approaches to increase the reliability of systems as well as an operational assistance system which increases the safety of its numerical control. Thereby the fail-safe approach of operational assistance system is extended to a fault tolerance approach. The goal is to design a CNC system that can mask internal errors through redundancy based on a majority decision, as well as outsource faulty redundant units through reconfiguration, so that no defective setpoints are output. Thus the system outwardly has a correct function, so that a started process can be completed without interruption.

Inhaltsverzeichnis

Abkürzungsverzeichnis

λ	Ausfallrate
$F(t)$	Ausfallwahrscheinlichkeit
$R(t)$	Zuverlässigkeitsfunktion
AV	Antriebsverstärker
AWL	Anweisungsliste [DIN03]
CCF	Common Cause Failure [DIN11b]
DFG	Deutsche Forschungsgemeinschaft
DPR	Dual Ported RAM
EKS	Echtzeitkommunikationssystem
Ent.	Entscheider
FMEA	Fehlzustandsart- und -auswirkungsanalyse [DIN06]
Fpr.	Funktionsprüfung
FSK	Frequency Shift Keying
FTA	Fault Tree Analysis [DIN81]
HIL	Hardware in the Loop
HMI	Human Machine Interface

HW	Hardware
IPC	Industrie PC
LAN	Local Area Network
M	Motor
MISRA	Motor Industry Software Reliability Association
MTBF	Mean Time Between Failure
NC	Numerical Control
PC	Personal Computer
PCI	Peripheral Component Interconnect
PL	Performance Level [DIN08]
RAM	Random Access Memory
sercos	Serial Real-Time Communication System
SIL	Safety Integrity Level [DIN13, DIN11a]
SW	Software
TÜV	Technischer Überwachungsverein
VM	Virtual Machine, Maschinensimulation
Zust.	Zustand
$\mathscr{O}$	Landau Symbol zur Angabe der Komplexität

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Nach dem immer noch geltenden Mooreschen Gesetz verdoppelt sich die Komplexität integrierter Schaltkreise mit minimalen Komponentenkosten alle ein bis zwei Jahre [Moo65, Sti05]. Computerhardware wird somit bei gleichbleibenden Kosten immer leistungsfähiger. Diese andauernde Leistungssteigerung ermöglicht computerbasierten numerischen Steuerungen den Einsatz in immer komplexer werdenden Anwendungsbereichen, die bisher allein dem Menschen vorbehalten waren. Damit werden numerische Steuerungen vermehrt auch in sicherheitskritischen Anwendungen eingesetzt. [LBP01, Lai05, KLV09]

Einer der Anwendungsbereiche, der sehr hohe Anforderungen an Sicherheit und Zuverlässigkeit stellt, ist die minimal invasive Chirurgie. Assistenzsysteme hierfür sind bisher fehlersicher ausgelegt und gewährleisten, dass das System bei einem internen Fehler den Patienten nicht unmittelbar verletzt [Lai05]. Zukünftig ist dies nicht weiter ausreichend, da eine Operation durch die Umstellung auf einen manuellen Eingriff verlängert wird. Bei bestimmten Operationstechniken ist solch eine Umstellung sogar nur durch einen deutlich größeren Eingriff durchführbar oder im schlechtesten Fall überhaupt nicht möglich. [Hei00, PWHB04]

Auch beim Fräsen sehr teurer Werkstücke mit hohem Zerspanungsanteil und dadurch langen Bearbeitungszeiten, wie zum Beispiel Turbinenschaufeln oder Flugzeugteilen [Lan05], ist eine hohe Zuverlässigkeit gefordert. Verursacht ein Fehler im Steuerungssystem eine Bahnabweichung, so zerstört dies meist das Werkstück. Dies ist bei Werkstücken mit einer Herstellungsdauer von mehreren Tagen sehr kostspielig. Bei Schiffsschrauben muss nach Zerstörung des Werkstücks beispielsweise erst zeitaufwendig ein neuer Rohling gegossen werden.

Weicht das Werkstück zu sehr von den Entwürfen ab drohen hohe Konventionalstrafen, insbesondere wenn die Schiffsschraube das Schiff dadurch nicht auf die vorgeschriebene Geschwindigkeit bringt oder gar störende Vibrationen auftreten. [hit09]

In [Lai05] wurde für ein Operationsassistenzsystem bereits die Sicherheit dessen numerischer Steuerung erhöht. Dies erfolgte durch einen redundanten Ansatz, bei dem sich zwei redundante Steuerungskanäle auf diversitärer Hardware gegenseitig überwachen. Treten bei den redundanten Berechnungen beider Kanäle zu große Abweichungen auf, so wird die Maschine automatisch über einen Not-Halt stillgesetzt. Die Sicherheit ist dadurch gewährleistet, jedoch kann die Operation mit dem System nicht fortgeführt werden. Das System ist folglich sehr sicher jedoch weniger zuverlässig.

Das Ziel dieser Arbeit ist die Erhöhung der Zuverlässigkeit numerischer Steuerungssysteme unter Beibehaltung der durch bestehende Ansätze hinzugewonnenen Sicherheit. Die Erhöhung der Zuverlässigkeit bedeutet im Umkehrschluss eine Verringerung der Ausfallwahrscheinlichkeit der Steuerungsfunktionalität zwischen Bearbeitungsbeginn und -ende eines anfangs fehlerfreien Steuerungssystems.

In dieser Arbeit wird dazu der fehlersichere Ansatz um Fehlertoleranz erweitert. Das fehlertolerante numerische Steuerungssystem soll Fehler in einem redundanten Steuerungskanal korrigieren. Nach außen hin soll das System eine fehlerfreie Funktion aufweisen, so dass ein begonnener Bearbeitungsprozess ohne Unterbrechung zu Ende geführt werden kann.

2. Motivation und Zielsetzung

Numerisch gesteuerte Maschinen und Roboter werden heutzutage nicht mehr nur in den klassischen Einsatzgebieten wie Fräs- oder Drehbearbeitung eingesetzt. In neuen Einsatzgebieten wie der Medizintechnik stehen dabei zunehmend Menschen in unmittelbarem Kontakt zur Maschine und sind bei Fehlfunktionen besonders gefährdet.

Beispielsweise spielt die Sicherheit des Patienten bei chirurgischen Eingriffen die wichtigste Rolle [KEB+03a, KEB+03b]. Diese können durch robotergestützte Operationstechniken bereits mit sehr hoher Präzision durchgeführt werden. Wird bei einer robotergestützten Operationstechnik die Vorgehensweise beim Eingriff selbst beibehalten, so ist für das Assistenzsystem eine fehlersichere Auslegung noch ausreichend. Bei neuartigen Operationstechniken, die erst durch robotergestützte Assistenzsysteme ermöglicht werden, kann ein Wechsel zu einer konventionellen Operationstechnik, wenn überhaupt möglich, nur mit großen Risiken für den Patienten erfolgen. [Hei00]

Die Zuverlässigkeit gewinnt damit für solche Systeme maßgeblich an Bedeutung. Hierbei reicht die notwendige robuste und möglicherweise redundante mechanische und elektrische Auslegung dieser Assistenzsysteme allein nicht mehr aus. Das Steuerungssystem, welches die koordinierte Bewegung des Assistenzsystems oder allgemein einer Maschine veranlasst, muss ebenfalls genau betrachtet werden.

2.1. Aufbau numerischer Steuerungssysteme

Die Aufgabe einer numerisch gesteuerten Maschine liegt darin, ein Instrument zu positionieren oder mit einem Werkzeug eine Bearbeitung durchzuführen. Dazu muss die Maschine das Instrument oder Werkzeug relativ zum Patient oder Werkstück auf einer vorbestimmten Bahn verfahren. Die Maschine besitzt dazu mehrere bewegliche Achsen, die durch Motoren angetrieben werden. Das Werkzeug ist dabei über eine Kinematik mit den verschiedenen Achsen der Maschine verbunden. Mit der Position jeder einzelnen Achse kann die Position des Werkzeugs im Raum eindeutig berechnet werden. Ebenso existiert für eine bestimmte Werkzeugposition eine meist eindeutige Kombination von Achspositionen.

Jede Achse ist mit einem Antriebsverstärker oder Frequenzumrichter verbunden, welcher den Motor der Achse mit Energie beaufschlagt und somit die Vorschubkraft der Achse steuert. Die Vorgabe für die Vorschubkraft bekommt der Antriebsverstärker von einer numerischen Steuerung in Form einer Strom-, Geschwindigkeits- oder Positionsvorgabe. Die numerische Steuerung bekommt wiederum von der Positionssensorik jeder Achse deren aktuelle Position zurückgemeldet.

Die numerische Steuerung befähigt eine Maschine dazu, ihr Werkzeug mit kontrollierter Geschwindigkeit einer vorbestimmten Bahn durch den Raum folgen zu lassen. Diese Bahn durch den Raum wird dabei während der Bewegung von der numerischen Steuerung aus einem NC-Programm [DIN83] eingelesen und verarbeitet. Die numerische Steuerung rechnet dabei die eingelesenen kartesischen Bahndaten in Sollwerte für die verschiedenen angetriebenen Achsen der Maschine um. Dies kann je nach Kinematik der Maschine von sehr einfach, wie beispielsweise bei einer seriellen Kinematik, bis zu sehr komplex werden, wie bei Parallelkinematiken. [PW97]

Damit die Bewegung auf der Bahn des Werkzeugs im Raum sehr präzise erfolgt, müssen die Antriebe der Achsen synchron mit Sollwerten beaufschlagt werden. Dabei zerlegt die numerische Steuerung die Bahn durch den Raum in sehr viele kleine Schritte, die sie den Antrieben der Achsen nacheinander in kur-

zen Zeitabständen vorgibt. Diese Zeitabstände werden Zykluszeit genannt und liegen bei numerischen Steuerungen hauptsächlich im Bereich von $0,5ms$ bis $5ms$ [WB05]. Solch niedrige Zykluszeiten sind notwendig, damit kleine Konturen auch mit hohen Vorschubgeschwindigkeiten bearbeitet werden können.

Das NC-Programm wird entweder vor dem Beginn des Bearbeitungsprozesses vom Maschinenbediener aus einer Datei geladen oder von einem übergeordneten System vorgegeben. Gestartet wird der Programmablauf vom Maschinenbediener. Dieser gibt ebenfalls die Bearbeitungsgeschwindigkeit als Prozentwert der programmierten Vorschubgeschwindigkeit vor.

2.2. Zuverlässigkeit des numerischen Steuerungssystems

Im Folgenden wird ein Bearbeitungsprozess mit einer numerisch gesteuerten Maschine angenommen, welcher keine Unterbrechung erlaubt. Der nicht zu unterbrechende Bearbeitungsprozess fordert damit eine sehr hohe *Zuverlässigkeit*. Diese definiert DIN 40041 [DIN90] wie folgt:

> Beschaffenheit einer Einheit bezüglich Ihrer Eignung, während oder nach vorgegebenen Zeitspannen bei vorgegebenen Anwendungsbedingungen die Zuverlässigkeitsforderung zu erfüllen.

Der Begriff Zuverlässigkeit wird im Folgenden für die Wahrscheinlichkeit verwendet, dass ein zu Beginn einer Bearbeitung störungsfreies System den Bearbeitungsprozess ohne Abweichung vollständig durchführt.

Eine *Störung* definiert [DIN90] zu:

> Fehlende, fehlerhafte oder unvollständige Erfüllung einer geforderten Funktion durch eine Einheit.

Nach [Wag96, Sch92, Vos88] bedeutet störungsfrei, dass die Funktionsfähigkeit des Systems keiner Beeinträchtigung unterliegt. Eine innere Störung bedeutet nicht automatisch Auswirkungen auf das äußere Verhalten das Systems. So

kann ein System, das eine Störung enthält, jedoch von außen betrachtet durchaus noch seinem Sollverhalten entsprechen und seine Funktion ordnungsgemäß erfüllen.

Die *Verfügbarkeit* sagt aus, wie wahrscheinlich ein System zu einem beliebigen Zeitpunkt funktionsfähig ist. Die Norm [DIN90] unterscheidet hierbei zwischen momentaner Verfügbarkeit und stationärer Verfügbarkeit. Momentane Verfügbarkeit definiert sie folgendermaßen:

> Wahrscheinlichkeit, eine Einheit zu einem vorgegebenen Zeitpunkt der geforderten Anwendungsdauer in einem funktionsfähigen Zustand anzutreffen.

Die stationäre Verfügbarkeit definiert sie zu:

> Mittlere Betriebsdauer zwischen zwei Ausfällen dividiert durch die Summe aus mittlerer Betriebsdauer und mittlerer Störungsdauer.

Im Folgenden wird der Begriff Verfügbarkeit im Sinne der stationären Verfügbarkeit verwendet. Dies entspricht der Wahrscheinlichkeit, das System zu einem beliebigen Zeitpunkt störungsfrei anzutreffen. Die Verfügbarkeit darf hierbei nicht mit dem Begriff Zuverlässigkeit verwechselt werden.

Zur Veranschaulichung des Unterschieds zwischen Zuverlässigkeit und Verfügbarkeit zeigt Abbildung 2.1 im oberen Diagramm den Zustand des Systems, im unteren den Betriebsmodus. Die Verfügbarkeit des gezeigten Systems entspricht dem Quotient aus der mittleren Betriebsdauer und der Summe aus mittlerer Betriebsdauer und mittlerer Störungsdauer. Die Zuverlässigkeit hingegen ist die Wahrscheinlichkeit des Auftreten keines Fehlers während der Bearbeitung bei zu Prozessbeginn störungsfreiem System. Sie hängt maßgeblich davon ab, wie sich eine auftretende Störung auf den Betriebsmodus auswirkt.

Ein *Fehler* liegt nach [Wag96, Sch92, Vos88] vor, wenn das System von seinem Sollverhalten abweicht. Ebenso definiert [DIN90] den Begriff Fehler als „Nichterfüllung einer Forderung“. Im Folgenden wird der Begriff Fehler im

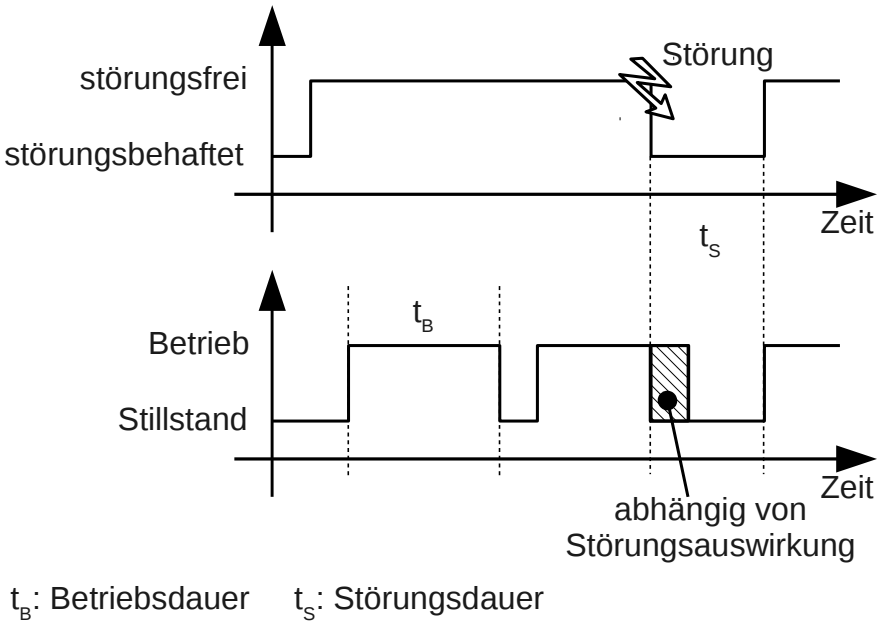


Abbildung 2.1.: Veranschaulichung von Betriebs- und Störungsdauer

Sinne von Abweichung vom Sollverhalten verwendet. Ein Fehler bedingt dabei eine vorhergehende und andauernde Störung.

Eine Spezialisierung des Falles Fehler ist der *Ausfall*, den [DIN90] als „Beendigung der Funktionsfähigkeit einer materiellen Einheit im Rahmen der zugelassenen Beanspruchung“ definiert. Der Begriff Ausfall wird im Folgenden als ein Nichterfüllen der geforderten Funktion verwendet.

Bleibt ein Fehler unbehandelt, so kann als Folge ein *Schaden* entstehen. Ein Schaden bedeutet nach [Wag96, Sch92, Vos88] das Beschädigen oder Zerstören von Komponenten wie Werkstück, Mensch oder Maschine. So kann ein Werkstück oder die Maschine durch Abweichen des Werkzeugs von der Sollbahn zerstört werden. Ebenso kann das Ausbleiben des Sollverhaltens im Falle eines Ausfalls bei zeitkritischen Prozessen indirekt Schaden am Werkstück verursachen.

2.2.1. Zuverlässigkeitskenngrößen

Um Aussagen über die Zuverlässigkeit eines Systems treffen zu können gibt es Zuverlässigkeitskenngrößen, welche diese quantifizieren. So definiert For-

mel 2.1 nach [Pri03] die Zuverlässigkeitsfunktion $R(t)$, welche die Überlebenswahrscheinlichkeit eines Systems abhängig von der Betriebsdauer t angibt. Definiert ist die Zuverlässigkeitsfunktion dabei über das Verhältnis zwischen Anzahl der zum Zeitpunkt t noch funktionierenden Systemen $n(t)$ zu der ursprünglich vorhandenen Anzahl aller Systeme $n0$. Diese Zahl muss sehr groß sein, damit ein statistisch aussagekräftiger Wert ermittelt werden kann. Komplementär zur Zuverlässigkeitsfunktion gibt die Funktion $F(t)$ in Formel 2.2 die Ausfallwahrscheinlichkeit eines Systems abhängig von der Betriebsdauer an.

$$R(t) = \frac{n(t)}{n_0}; \quad n(t)\text{: Anzahl überlebender Systeme} \tag{2.1}$$

$$F(t) = \frac{N(t)}{n_0}; \quad N(t)\text{: Anzahl ausgefallener Systeme} \tag{2.2}$$

$$1 = R(t) + F(t); \quad \text{wegen } n_0 = N(t) + n(t) \tag{2.3}$$

Die Ausfälle sind über den gesamten Lebenszeitraum eines Systems nicht gleich verteilt. So zeigt die Badewannenkurve nach Weibull [Wei51, LG99], dass es bei technischen Systemen nach einer anfangs hohen, schnell abnehmenden Frühausfallrate, zu einem langen Zeitraum mit konstanter Ausfallrate λ kommt, in dem sporadische Ausfälle dominieren. Nach langer Betriebsdauer steigt die Ausfallrate durch Alterserscheinungen wieder an. Technische Systeme werden daher teilweise direkt nach der Herstellung eingefahren, so dass Systeme mit frühen Ausfällen gar nicht erst ausgeliefert werden. Ausgelieferte Systeme haben somit eine konstante Ausfallrate während ihrer spezifizierten Betriebsdauer. Die Überlebenswahrscheinlichkeit errechnet sich damit nach [LG99, Pri03] zu Formel 2.6.

$$\lambda(t) = -\frac{\mathrm{d}}{\mathrm{d}t}\{\ln R(t)\} = -\frac{\frac{\mathrm{d}R(t)}{\mathrm{d}t}}{R(t)} \tag{2.4}$$

$$R(t) = e^{-\int_0^t \lambda(\tau)\,\mathrm{d}\tau} \tag{2.5}$$

$$R(t) = e^{-\lambda t}; \quad \text{mit konstanter Ausfallrate } \lambda \tag{2.6}$$

Bei konstanter Ausfallrate berechnet sich diese aus dem Kehrwert der mittleren Zeitdauer zwischen zwei aufeinanderfolgenden Ausfällen, dem MTBF, eines Systems zu Formel 2.7. Mathematisch betrachtet bedeutet der MTBF den Zeitpunkt, zu dem bei konstanter Ausfallrate noch $e^{-1} = 36,79\%$ aller anfangs vorhandenen Systeme funktionsfähig sind.

$$\lambda = \frac{1}{MTBF} \tag{2.7}$$

Die Überlebenswahrscheinlichkeit eines redundanten Systems berechnet sich aus den Zuverlässigkeitsfunktionen der Einzelsysteme. Für die Überlebenswahrscheinlichkeit eines m von n Systems gilt nach [Pri03, Ech90] die Formel 2.8.

$$R_{m\ von\ n}(t) = \sum_{i=m}^{n} \binom{n}{i} R^i(t)(1-R(t))^{n-i} \tag{2.8}$$

2.2.2. Zuverlässigkeit des numerischen Steuerungssystems

Mit der gegebenen Anforderung, dass der begonnene Bearbeitungsprozess ohne Unterbrechung durchgeführt werden muss, wird eine Betrachtung des gesamten Zusammenwirkens innerhalb des Steuerungssystems notwendig. Die Abbildung 2.2 stellt den Datenfluss des Gesamtsystems dar. Das Steuerungssystem wird im Folgenden genauer betrachtet.

Aus Abbildung 2.2 lässt sich die Zuverlässigkeitskette des bewegungserzeugenden Steuerungssystems wie in Abbildung 2.3 dargestellt ableiten. Hinzugekommen ist dabei der Block Datenübertragung, ohne welche das Steuerungssystem seiner Aufgabe nicht gerecht werden kann. Sind die Zuverlässigkeitswerte jeder Komponente in der Zuverlässigkeitskette bekannt, so lässt sich die Zuverlässigkeit des Steuerungssystems durch einfache Multiplikation aller Werte berechnen.

Eine Störung innerhalb des Steuerungssystems führt dabei nicht zwingend zu einer Beschädigung des Werkstücks. Einen Schaden nimmt das außerhalb des Steuerungssystems liegende Werkstück erst, wenn die Störung nach außen

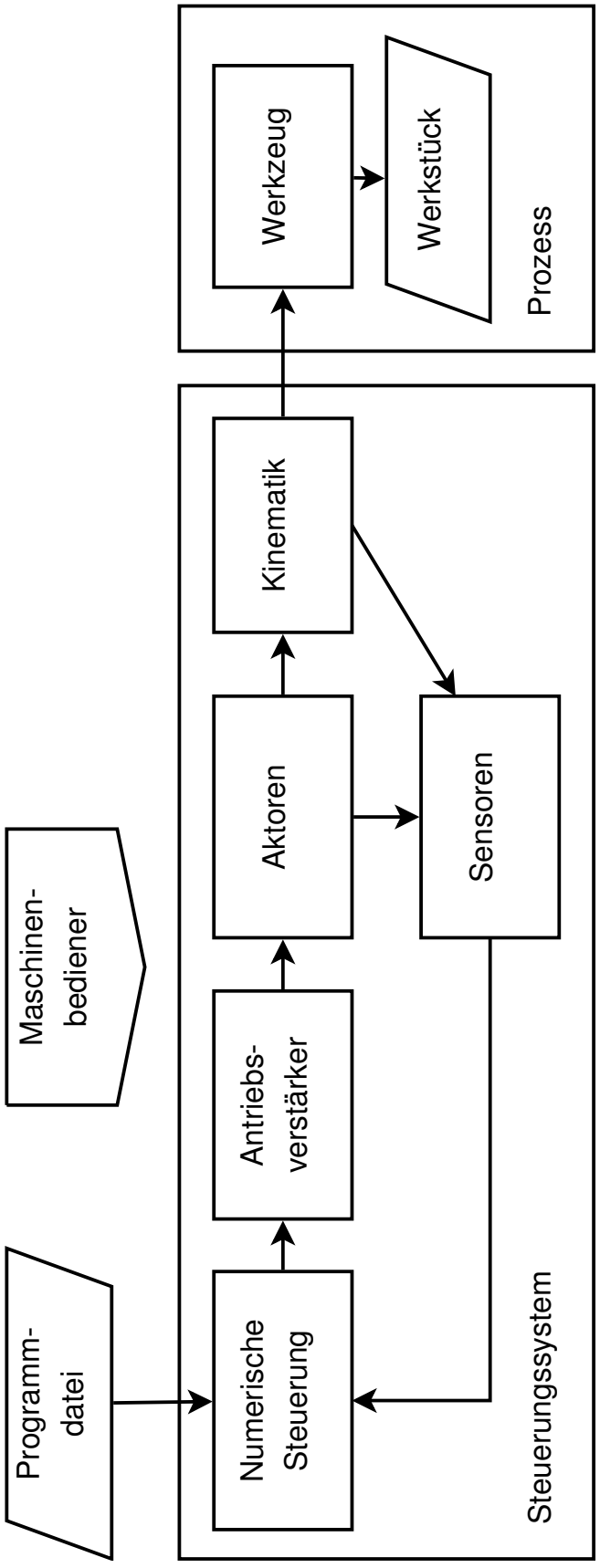


Abbildung 2.2.: Datenfluss des Bearbeitungsprozesses

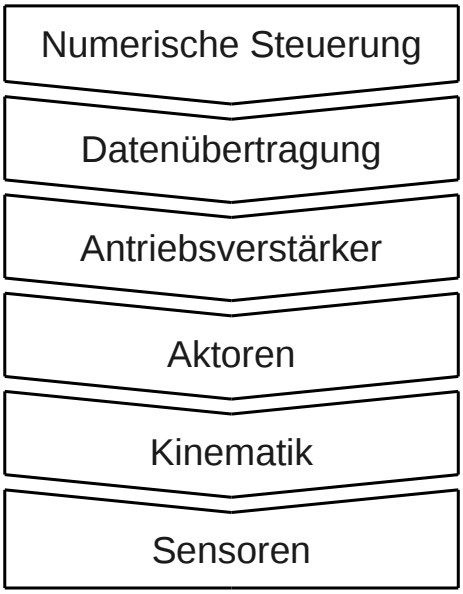


Abbildung 2.3.: Zuverlässigkeitskette des Steuerungssystems

tritt und als Fehler mit Abweichung vom Sollverhalten oder als Ausfall mit Nichterfüllen der Funktion sichtbar wird.

Besonders kritisch ist ein Fehler der numerischen Steuerung, da sich deren Fehlerfreiheit nicht einfach überwachen lässt. Erzeugt die numerische Steuerung Sollwerte die von der Sollbahn abweichen, jedoch innerhalb der von Überwachungsmechanismen erlaubten Grenzen liegen, wird das Werkstück beschädigt. Die Wahrscheinlichkeit einer direkten Beschädigung des Werkstücks infolge eines solchen unerkannten Fehlers der numerischen Steuerung ist daher sehr hoch.

2.3. Problemstellung

Je nach Art der Anwendung sind unterschiedlich hohe Anforderungen an die Zuverlässigkeit des Steuerungssystems gestellt. Werden billige Werkstücke bearbeitet, so liegt der Fokus auf einer hohen Verfügbarkeit sowie der Sicherheit, dass die Maschine weder seiner Umgebung noch sich selbst schadet. Der Verlust von einzelnen Werkstücken ist in diesem Fall vertretbar.

Sind jedoch teure Werkstücke oder gar ein Mensch vom Prozess betroffen, so stellen sich sehr viel höhere Anforderungen an die Zuverlässigkeit des Gesamtsystems. Im Folgenden wird daher das noch hinnehmbare Risiko für den Eintritt eines schwerwiegenden Schadens quantifiziert.

2.3.1. Analyse des hinnehmbaren Schadenrisikos

Der Ansatz aus der früheren Norm DIN EN 954-1 [DIN97], die sicherheitsbezogene Teile von Steuerungen betrachtet, durch geeignete Maßnahmen die Wahrscheinlichkeit eines potentiellen Schadens zu verringern, wie Abbildung 2.4 zeigt, ist hier nicht mehr ausreichend. Funktionale Sicherheit verringert die Wahrscheinlichkeit eines zu Schaden führenden unkontrollierten Fehlverhaltens zu Gunsten einer sicheren Abschaltung, die nur zu einem Ausfall ohne oder mit geringerem Schaden führt.

Diese Maßnahmen verringern jedoch die Zuverlässigkeit des Gesamtsystems, da sie weitere Komponenten hinzufügen, die wiederum Ausfallen können aber zum Betrieb notwendig sind. Erkennt beispielsweise eine sicherheitsrelevante Komponente einen internen Fehler führt dies zu einer sicheren Abschaltung gefolgt vom Ausfall des Gesamtsystems, obwohl das Steuerungssystem selbst fehlerfrei funktionieren würde.

Die Nachfolgenorm zu [DIN97], die EN ISO 13849-1 [DIN08], definiert zusätzlich zu den bestehenden qualifizierten nun auch quantifizierte Anforderungen. So wird darin anhand einer Risikoanalyse der für das betrachtete System zur Risikominderung notwendige Performance Level (PL) bestimmt. Jeder Performance Level definiert dabei den zulässigen Wahrscheinlichkeitsbereich für das Auftreten eines gefährlichen Ausfalls innerhalb eines Betrachtungszeitraums von einer Stunde. Die Abbildung 2.5 aus [HSA+08] zeigt diese Bereiche für alle unterschiedlichen Performance Level und stellt ihnen gleichzeitig die Safety Integrity Level (SIL) aus den Normen DIN EN 62061 [DIN13] sowie DIN EN 61508 [DIN11a] gegenüber.

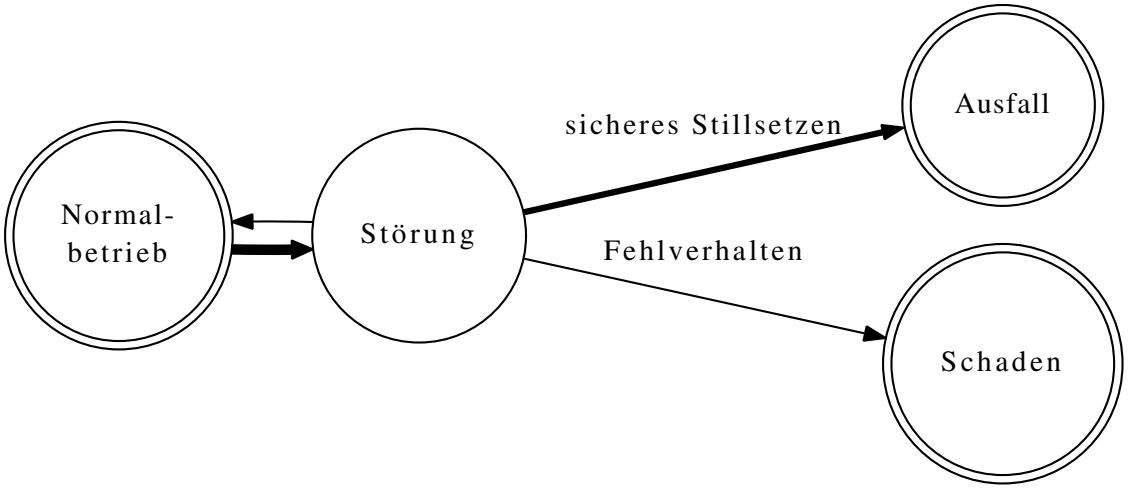


Abbildung 2.4.: Sicheres Stillsetzen verringert lediglich die Wahrscheinlichkeit eines Schadens.

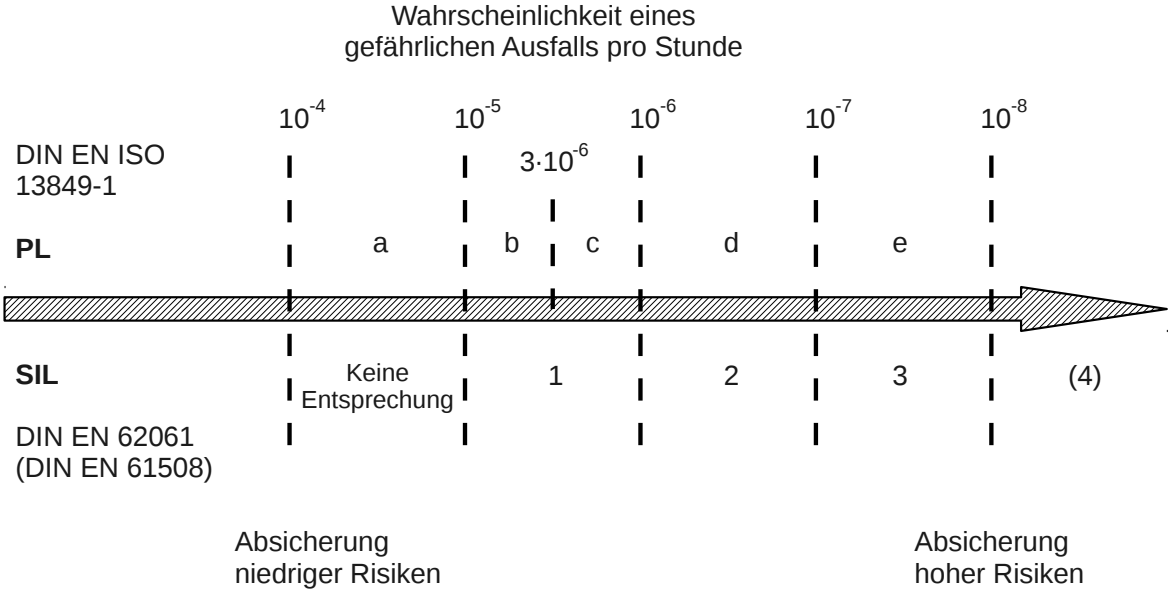


Abbildung 2.5.: Zulässige Wahrscheinlichkeit eines gefährlichen Ausfalls pro Stunde [HSA+08]

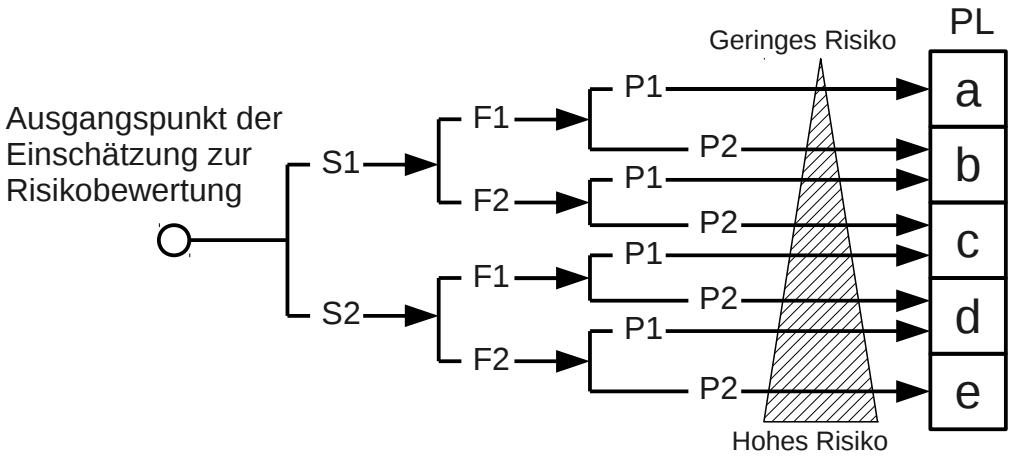


Abbildung 2.6.: Risikograph nach [DIN08] zur Bestimmung des Performance Levels

Der von einer sicherheitsrelevanten Funktion geforderte Performance Level wird in einer Risikoanalyse nach Abbildung 2.6 der [DIN08] ermittelt. Dabei werden der Schweregrad *S* einer potentiellen Verletzung, die Dauer und Häufigkeit *F* der Gefährdungsexposition sowie die Möglichkeiten der Vermeidung der Gefährdung *P* nach folgenden Kriterien abgeschätzt:

S1: leichte, üblicherweise reversible Verletzungen

S2: schwere, üblicherweise irreversible Verletzungen inklusive Tod

F1: seltene bis weniger häufig oder kurze Dauer der Exposition

F2: häufige oder lange Exposition

P1: Gefährdungsvermeidung ist möglich

P2: Gefährdungsvermeidung nicht möglich

Ausgehend vom ermittelten Performance Level für eine sicherheitsrelevante Funktion gibt [DIN08] einen erlaubten Wahrscheinlichkeitsbereich für einen gefährlichen Ausfall pro Stunde vor.

Für einen Anwendungsfall wie beispielsweise einem Operationsassistenzsystem, durch das bei einem fehlerhaften Eingriff in den menschlichen Körper

schwere Verletzungen (S2) verursacht werden, bei dem die Expositionsdauer gewöhnlich über einen längeren Zeitraum (F2) notwendig ist und systembedingt auch eine Gefährdungsvermeidung kaum möglich ist (P2), ergibt sich ein Performance Level von „e“. Dies entspricht nach Abbildung 2.5 dem SIL Level 3.

Die Wahrscheinlichkeit eines gefährlichen Ausfalls des Gesamtsystems pro Stunde darf damit nicht höher als 10^{-7} (0,00001%) liegen. Dies entspricht nach Formel 2.7 einem MTBF von etwa 1141 Jahren. Zum Vergleich liegt der MTBF von Personal Computern nach [SRC06] zwischen 1.000 und 5.000 Stunden, für Industrie PCs meist bei 70.000 Stunden, das sind knapp 8 Jahre und entspricht nach Formel 2.7 einer Ausfallwahrscheinlichkeit im Bereich von 10^{-5} (0,0014%) pro Stunde. Dies entspricht nach Abbildung 2.5 nur dem Performance Level „a“. Dies ist kritisch wenn ein IPC als Steuerungsrechner eingesetzt wird und sicherheitsrelevante Aufgaben übernehmen muss, da ein Fehler des Steuerungsrechners sich unmittelbar als gefährlicher Ausfall auswirkt.

2.3.2. Analyse des Steuerungssystems bezüglich sicherheitsrelevanter Funktionen

Besitzt ein Bearbeitungsprozess keinen unmittelbar sicheren Zustand, so ist ein Not-Halt als einzige Maßnahme zur Verhinderung von Schaden ungeeignet. Reine Sicherheitsfunktionen zum Stillsetzen reichen nach Erkennen von Störungen damit nicht weiter aus. Vielmehr muss in solch einem Fall die Zuverlässigkeit des Steuerungssystem der notwendigen Wahrscheinlichkeit entsprechen, dass der Bearbeitungsprozess bis zu einem sicheren Zustand gebracht werden kann. Dadurch dehnt sich die zu berücksichtigende Sicherheitskette auf das gesamte Steuerungssystem aus, da dieses zum Aufrechterhalten des Prozesses notwendig ist.

Mit der Fehlerbaumanalyse (FTA) gibt die Norm DIN 25424 [DIN81] eine Methode vor, durch die bestimmt wird, welche Ursachen zu einem unerwünschten Ereignis führen. Dabei wird ein Fehlerbaum aufgebaut, der vom Allgemei-

nen ins Spezielle geht. An der Baumwurzel steht der unerwünschte Fehlerfall, den es zu verhindern gilt. Ausgehend davon werden Zweige gebildet, die zu dem unerwünschten Ereignis führen können. Die Blätter des Fehlerbaums bilden Eingangs- oder Basisereignisse. Eingangsereignisse können wiederum in einen weiteren Fehlerbaum zerlegt werden, während Basisereignisse für sich alleine stehen.

Als unerwünschtes Ereignis gilt hier die fehlerhafte Bewegungserzeugung durch das elektrische Steuerungs- und Antriebssystem. Die Abbildung 2.7 zeigt die Fehlerbaumanalyse aus [Lai05] zu diesem unerwünschten Ereignis.

Jeder einzelne der in der Fehlerbaumanalyse aufgezeigten Zweige kann zu einer fehlerhaften Bewegungserzeugung führen. Daher sind für Antriebsverstärker sowie Überwachungsgeräte bereits hinreichend zuverlässige Komponenten verfügbar. Ebenso existieren Lösungsansätze für speicherprogrammierbare Steuerungen [Sch94a], Kommunikationssysteme [PSR01, Sta06] sowie für robuste Mechanik [Gar07]. Die Verhinderung der Ausgabe eines fehlerhaften Sollwertes durch einen Fehler in der numerischen Steuerung wird im Folgenden betrachtet.

Eine von der Steuerung fehlerhaft vorgegebene Abweichung von der Sollbahn ist äußerst kritisch. Solange die ausgegebenen Werte innerhalb des für sie erlaubten Arbeitsbereichs liegen, kann die Abweichung von keinem Überwachungsgerät weder erkannt noch kompensiert werden. Aus diesem Grund wird der Fokus im weiteren auf das numerische Steuerungssystem gelegt. Dieses muss in einer weiteren Fehlerbaumanalyse auf Fehlerursachen hin untersucht werden. Die Abbildung 2.8 aus [Lai05] zeigt diese Analyse für das unerwünschte Ereignis, dass die numerische Steuerung falsche Positionssollwerte ausgibt.

Ein Fehlerbaum wird für Hardwarekomponenten gewöhnlich nicht bis zum kleinstmöglichen Bauteil ausgeführt, da diese Bäume zu komplex werden würden. Stattdessen wird für die einzelnen Bauteile einer Hardwarekomponente während der Entwurfsphase eine FMEA (Fehlzustandsart- und -auswirkungsanalyse) nach [DIN06] durchgeführt. Dadurch wird sowohl der Einfluss eines

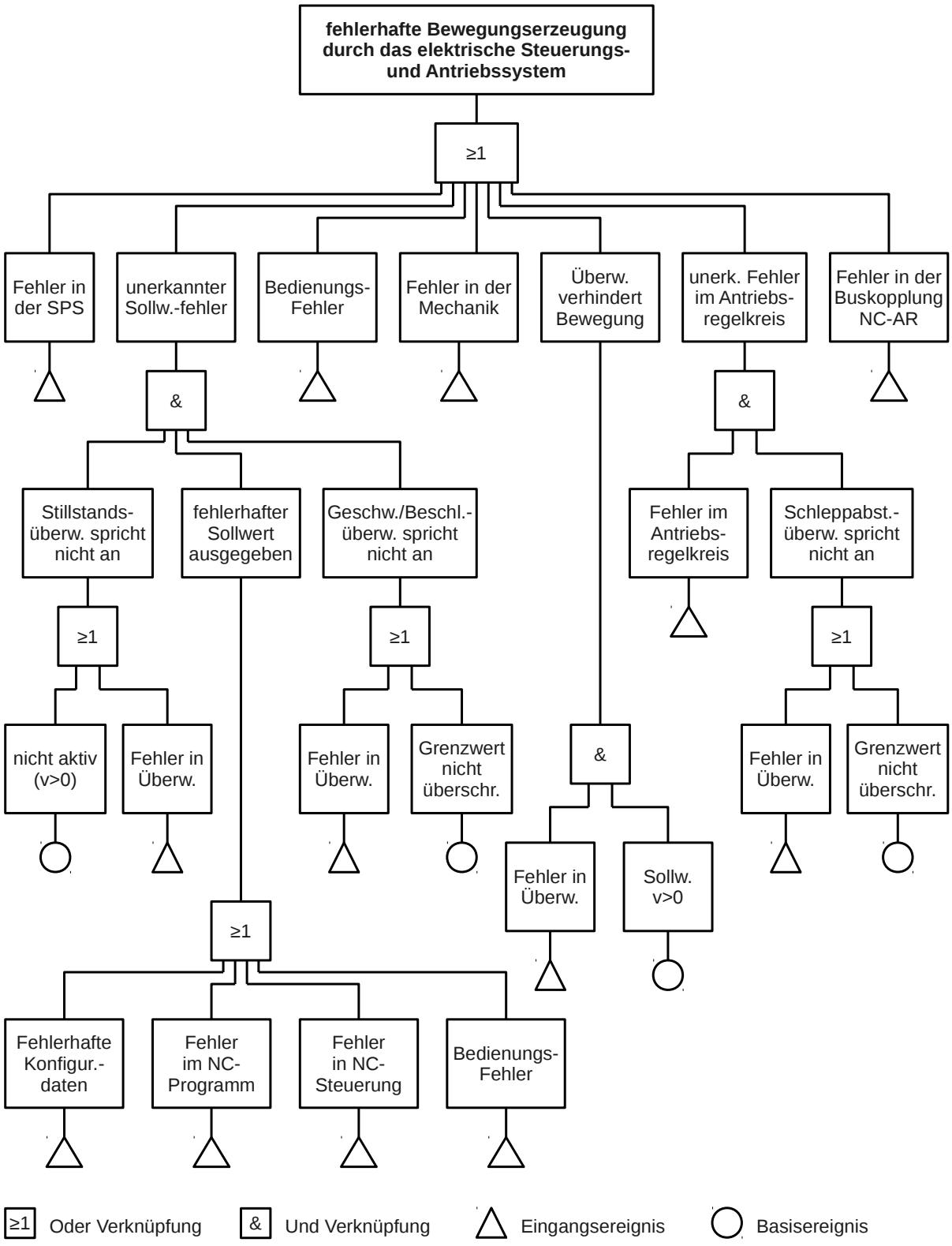


Abbildung 2.7.: Fehlerbaumanalyse des Steuerungssystems [Lai05]

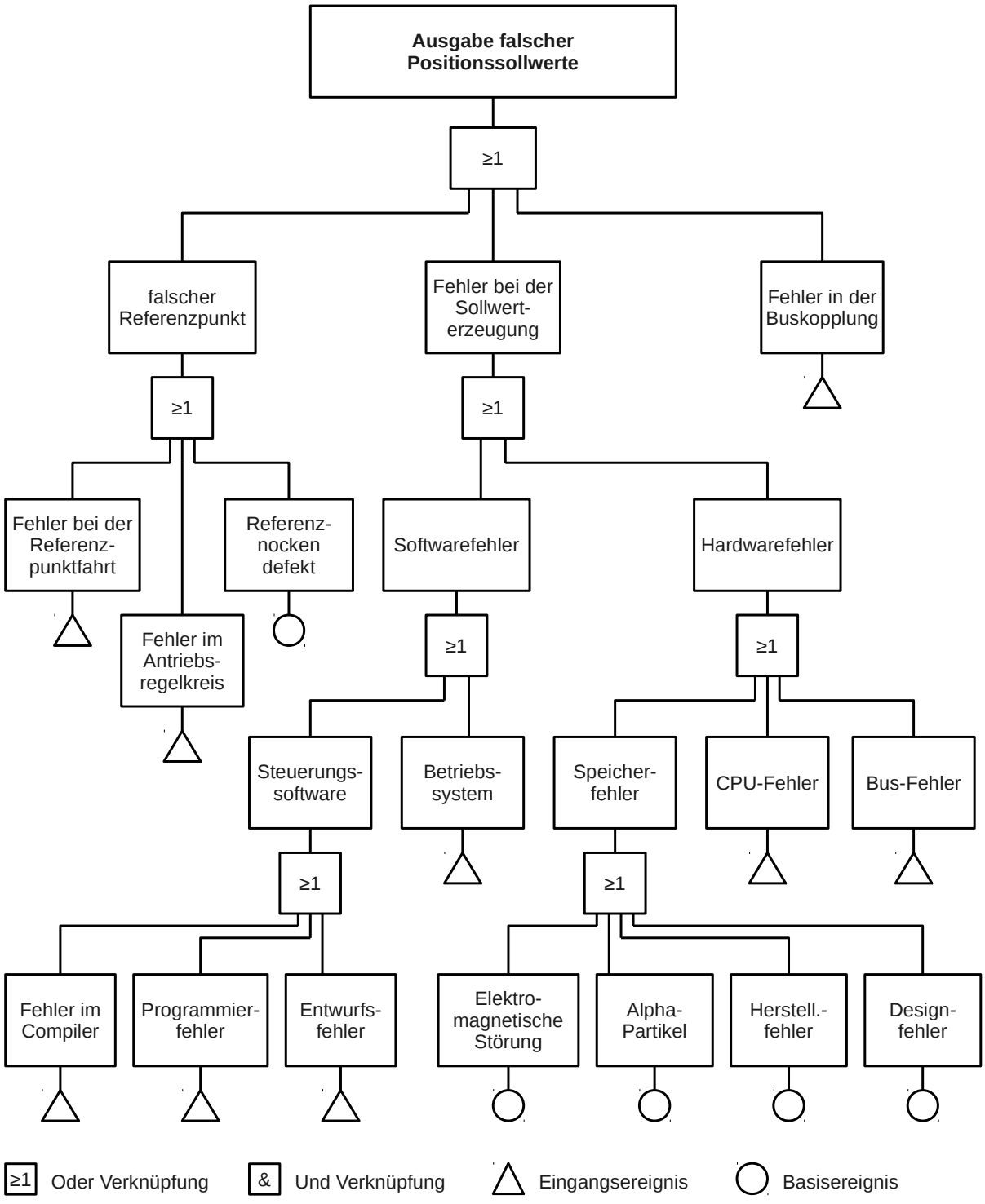


Abbildung 2.8.: Fehlerbaumanalyse der numerischen Steuerung [Lai05]

Bauteils auf das Gesamtsystem bestimmt, als auch die Ausfallwahrscheinlichkeit des Gesamtsystems quantifiziert. Da diese Analyse sehr aufwendig und umfangreich ist, sollten dafür sinnvollerweise entsprechende Softwarewerkzeuge eingesetzt werden. [KJG+11]

Eine FMEA wird für die fehlertolerante numerische Steuerung in dieser Arbeit nicht durchgeführt, da sie einer konkreten Hardwarekomponente bedarf und nicht auf abstrakte, bauteillose Geräte angewendet werden kann. Stattdessen werden dafür empirische Werte für die Ausfallraten der Hardwarekomponenten aus [SRC06] verwendet.

2.3.3. Bewertung der Analyse

Aus der vorangegangenen Analyse ergibt sich, dass für besondere Anwendungsfälle, die nicht zu jedem beliebigen Zeitpunkt einen sicheren Zustand besitzen, die Ausgabe fehlerfreier Positionssollwerte zu einer sicherheitsrelevanten Funktion wird. Hierbei verdient die fehlerfreie Erzeugung derselben eine sehr hohe Aufmerksamkeit, weil das numerische Steuerungssystem ein sehr komplexer Verbund aus Software und Hardware ist. Der Fokus dieser Arbeit liegt daher auf der Verhinderung der Ausgabe fehlerhaft berechneter Sollwerte. Fehler in der Busankopplung oder fehlerhafte Referenzpunkte werden nicht behandelt.

2.4. Zielsetzung

Das Ziel dieser Arbeit ist die Erhöhung der Wahrscheinlichkeit, dass es durch Fehler bei der Sollwerterzeugung nach Beginn des Bearbeitungsprozesses weder zu einem Ausfall noch zu einem Schaden kommt.

Zur Erreichung des Ziels soll die fehlertolerante numerische Steuerung im Gegensatz zu einer gewöhnlichen numerischen Steuerung aus Kapitel 2.3.1 nach dem Auftreten einer Störung den Prozess nicht abbrechen. Vielmehr soll die fehlertolerante numerische Steuerung in einen Zustand wechseln, in dem sie die Störung kompensiert sowie den Prozess nach Abbildung 2.9 mindestens

fehlersicher aufrecht erhält. In diesem neuen Zustand soll die numerische Steuerung weiterhin ihre volle spezifizierte äußere Funktion erfüllen. Der Wechsel vom Normalbetrieb in den neuen Zustand soll von außen betrachtet anhand der ausgegebenen Sollwerte nicht sichtbar sein.

Die Startbedingung zur Aufnahme einer Bearbeitung durch das fehlertolerante numerische Steuerungssystem ist, dass dieses keinerlei Störungen enthält. Das numerische Steuerungssystem soll sich so zu Bearbeitungsbeginn immer in dem fehlertoleranten Betriebszustand ohne vorhandene Störung befinden. Das Steuerungssystem soll die Fehlerfreiheit vor dem Bearbeitungsbeginn durch eine Funktionsprüfung und geeignete Selbsttests feststellen. Im folgenden fehlertoleranten Betriebszustand sollen Fehler oder Ausfälle von Teilkomponenten sowohl erkannt als auch behandelt werden. Weiterhin soll das Steuerungssystem nach Auftreten einer Störung in einen fehlersicheren Betriebszustand wechseln. In diesem Betriebszustand muss das Steuerungssystem einen weiteren Fehler oder Ausfall einer Teilkomponente sicher erkennen und unmittelbar darauf in einem sicheren Betriebszustand anhalten, so dass kein direkter Schaden verursacht wird.

Ausgehend von dieser Zielsetzung stellt sich die Aufgabe, dass Maßnahmen erarbeitet sowie validiert werden, welche die Zuverlässigkeit des numerischen Steuerungssystems um das notwendige Maß erhöhen. Im Folgenden sollen dazu Normen sowie bereits bestehende Ansätze im Themenfeld von Fehlertoleranz und numerischen Steuerungen untersucht und bewertet werden. Ausgehend davon sollen die notwendigen Maßnahmen abgeleitet und die daraus resultierenden Arbeitspunkte abgeleitet und vertieft werden. Zur Validierung der Eignung der gewählten Maßnahmen soll an einem Testaufbau die Erhöhung der Zuverlässigkeit durch Simulation von Fehlern nachgewiesen werden.

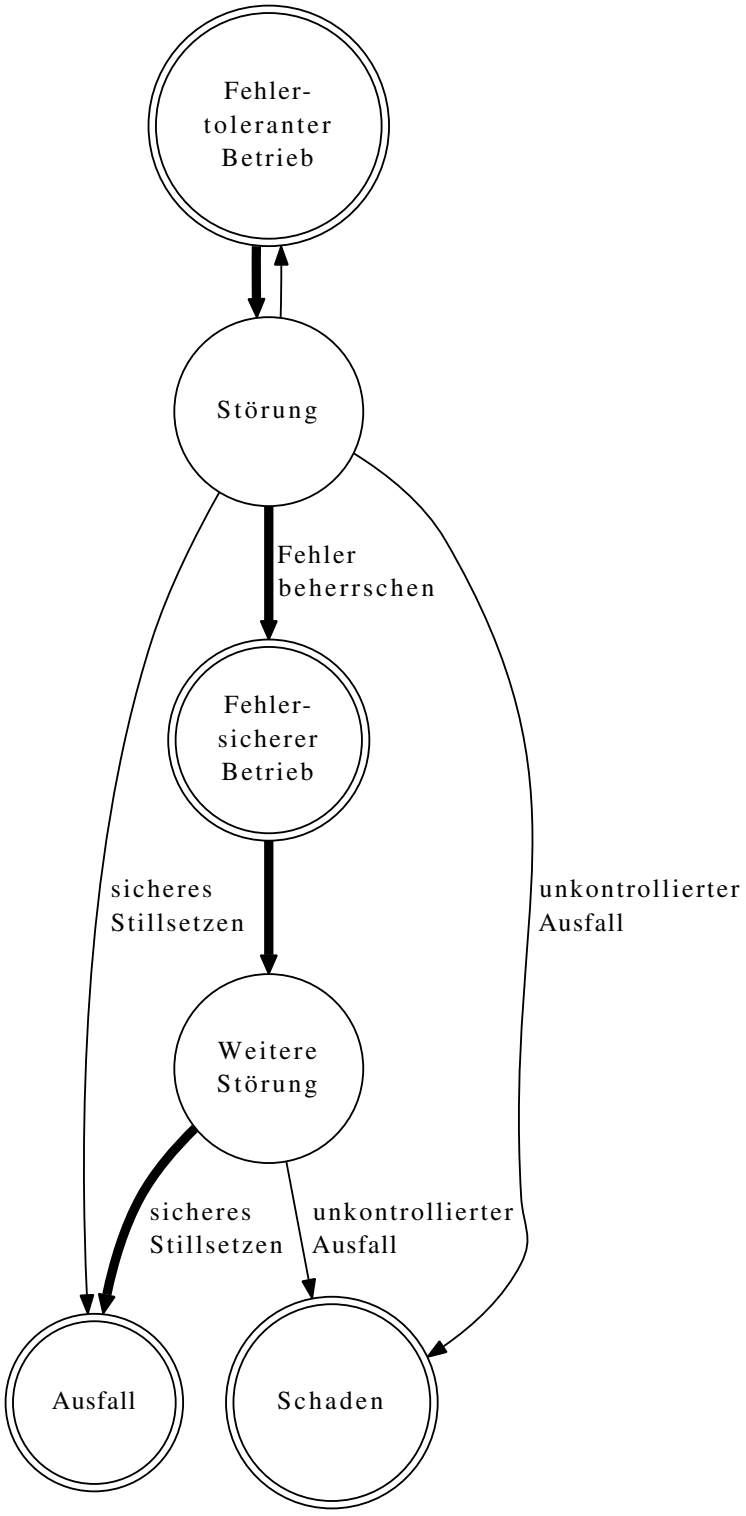


Abbildung 2.9.: Ablauf bei Fehlerfällen

3. Stand der Technik

Während dem Bearbeitungsprozess können sehr viele unterschiedliche Fehler auftreten. Um im Folgenden wirksame Maßnahmen gegen die Auswirkungen eines Fehlers bestimmen zu können, werden Fehler zunächst abstrahiert und einer Klassifikation unterzogen. Anhand dieser Klassifikation werden verschiedene Fehlerklassen voneinander abgegrenzt und die zu behandelnden Fehlerklassen identifiziert.

3.1. Klassifikation von Fehlern

Gemäß der in Abbildung 3.1 dargestellten Klassifikation von Fehlern [Lai05, VDE90], können Fehler an drei verschiedenen Orten auftreten:

- außerhalb des betrachteten Systems
- in der Systemhardware
- in der Systemsoftware

Fehler außerhalb des Systems sind Fehler in den Randbedingungen nach denen ein System ausgelegt ist. Beispiele hierfür sind Strom- und Betriebsstoffversorgung. Fehler außerhalb des Systems können und müssen unabhängig von der Steuerung erkannt und behandelt werden. Für den inneren Aufbau des Systems soll jedoch berücksichtigt werden, dass es strukturell nicht zwingend von den selben äußeren Ressourcen abhängt. Dies ist notwendig, damit ein äußerer Ausfall mit der selben Ursache (CCF) nicht alle Teilsysteme gleichsam betrifft. [DIN11b]

Fehler innerhalb des Systems können entweder in der Software vorhanden sein oder in der Hardware entstehen. Softwarefehler sind dabei generell syste-

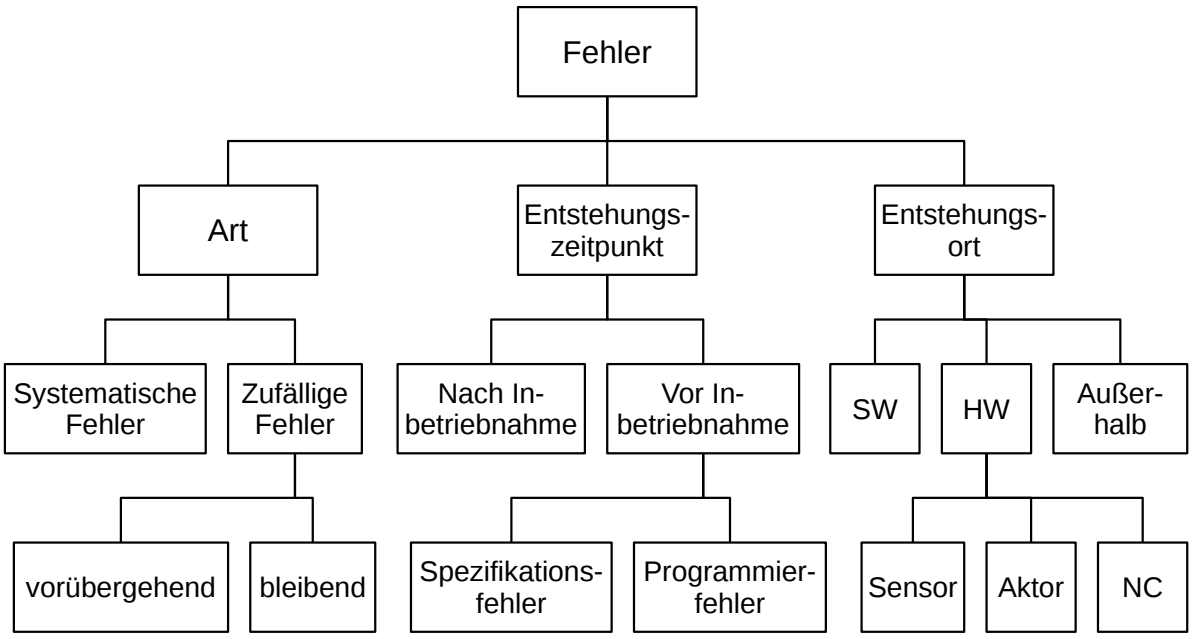


Abbildung 3.1.: Klassifikation von Fehlern [Lai05, VDE90]

matischer Natur. Sie bestehen aus Spezifikations- oder Programmierfehlern und werden vor der Inbetriebnahme in das System eingebracht. Da Software keinem Alterungsprozess unterliegt, wird eine fehlerfreie Software niemals falsche Werte ausgeben [LL13]. Eine fehlerhafte Software wird hingegen immer fehlerbehaftete Werte liefern.

Fehler in Software können dadurch lediglich vor der Inbetriebnahme vermieden oder im Betrieb erkannt, jedoch nicht behoben werden. Für die Vermeidung von Softwarefehlern bestehen Fehler vermeidende Maßnahmen wie Vorgehensmodelle zur Softwareentwicklung, Programmanalyse und Reviews, Fehlersimulationstests oder die Nutzung von Quelltextanalyse-Tools [Hof13, LL13, SL10, Lig09, Bür05]. Maßnahmen zur Fehlererkennung im Betrieb sind zusätzliche Überwachungsmechanismen wie zum Beispiel Plausibilitätstests oder Tendenzüberwachungen. Eine weitere Möglichkeit Softwarefehler zu erkennen ist, die Berechnungsergebnisse diversitärer Software auf Übereinstimmung zu prüfen. Diversitäre Software bedeutet, dass das für die selbe Funktion mehrere Programme von unterschiedlichen Entwicklern unabhängig voneinander entwickelt werden. [Lai05]

In [Bür05] untersucht Bürger analytische Qualitätssicherungsmaßnahmen für numerische Steuerungssoftware. Diese Maßnahmen kamen bereits für die erfolgreiche Abnahme eines Operationsassistenzsystem durch den TÜV zum Einsatz. Fehler durch Software werden daher im Folgenden nicht weiter betrachtet.

Im Gegensatz zu Softwarefehlern können Hardwarefehler nicht nur systematischer, sondern auch zufälliger Natur sein. Systematische Hardwarefehler entstehen vor der Inbetriebnahme einer Maschine. Deren Ursachen liegen in Spezifikations-, Design- oder Herstellungsfehlern. Systematische Hardwarefehler lassen sich ebenfalls wie Softwarefehler nur vorab vermeiden oder im Betrieb erkennen. Zur Fehlervermeidung werden Maßnahmen wie Reviews, Baumusterprüfungen und Untersuchungen auf elektromagnetische Verträglichkeit eingesetzt. Eine Erkennung von systematischen Hardwarefehlern im Betrieb lässt sich nur durch diversitäre sowie redundante Hardware sicher erreichen. Diversitäre Hardware ist hierbei sehr wichtig, damit ein möglicher systematischer Hardwarefehler auf eine der redundanten Komponenten begrenzt bleibt. Ohne Diversität, also bei baugleichen redundanten Komponenten, könnte ein systematischer Hardwarefehler zu gleichem fehlerhaftem Verhalten jeder einzelnen redundanten Komponente führen.

Zufällige Hardwarefehler hingegen treten nach der Inbetriebnahme auf. Sie können vor Bearbeitungsbeginn durch Maßnahmen wie Selbsttests, Prozessortests oder Speichertests erkannt werden. Durch redundante Hardware lassen sich zufällig auftretende Fehler in der Hardware erkennen, indem die redundant berechneten Ergebnisse miteinander verglichen werden.

Damit zufällige wie auch systematische Hardwarefehler sicher erkannt werden können, muss die Steuerungshardware redundant sowie diversitär ausgelegt sein. Redundant, damit ein zufälliger Fehler durch das System erkannt wird. Diversitär, damit systematische Spezifikations- oder Produktionsfehler nicht in den redundanten Systemen gleichzeitig an der selben Stelle auftreten und zu dem selben fehlerhaften Ergebnis führen.

Maßnahmen zur Fehlervermeidung vor der Inbetriebnahme sind nicht Inhalt dieser Arbeit. Ebenfalls nicht betrachtet werden Softwarefehler sowie Fehler

außerhalb des Systems. Im Folgenden werden nur Maßnahmen zur Fehlerbehandlung diskutiert und die zu behandelnden Fehlerklassen auf folgende Hardwarefehler eingeschränkt:

- Zufällig auftretende Fehler in der numerischen Steuerungshardware.
- Systematische Fehler in der numerischen Steuerungshardware.

3.2. Allgemeine Maßnahmen zur Fehlerbehandlung

Echtle behandelt in [Ech90] Fehlertoleranzverfahren theoretisch und umfassend. Darin werden die folgenden Maßnahmen zur Fehlerbehandlung eingeführt:

- Rekonfiguration
- Rückwärtsfehlerbehebung
- Vorwärtsfehlerbehebung
- Fehlerkorrektur
- Fehlermaskierung

Die verschiedenen Verfahren zur Fehlerbehandlung strukturiert Echtle wie in Abbildung 3.2 dargestellt. Die verschiedenen Verfahren teilt er dabei zusätzlich in die Unterpunkte Fehlerausgrenzung, Fehlerbehebung und Fehlerkompensierung auf.

3.2.1. Fehlerausgrenzung

Das Ziel der *Fehlerausgrenzung* ist das Herstellen eines Systems ohne Störung. Dies wird durch Rekonfiguration des Systems erreicht.

Die *Rekonfiguration* steht als Überbegriff für die drei Maßnahmen Verlagerung, Ausgliederung sowie Eingliederung. Die *Verlagerung* verschiebt die Funktion eines fehlerhaften Systems auf ein noch funktionierendes System mit entsprechend freien Ressourcen. Durch Verlagerung ist es einer rekonfigurierbaren Werkzeugmaschine nach [Kir11] beispielsweise möglich, die Kinematik-

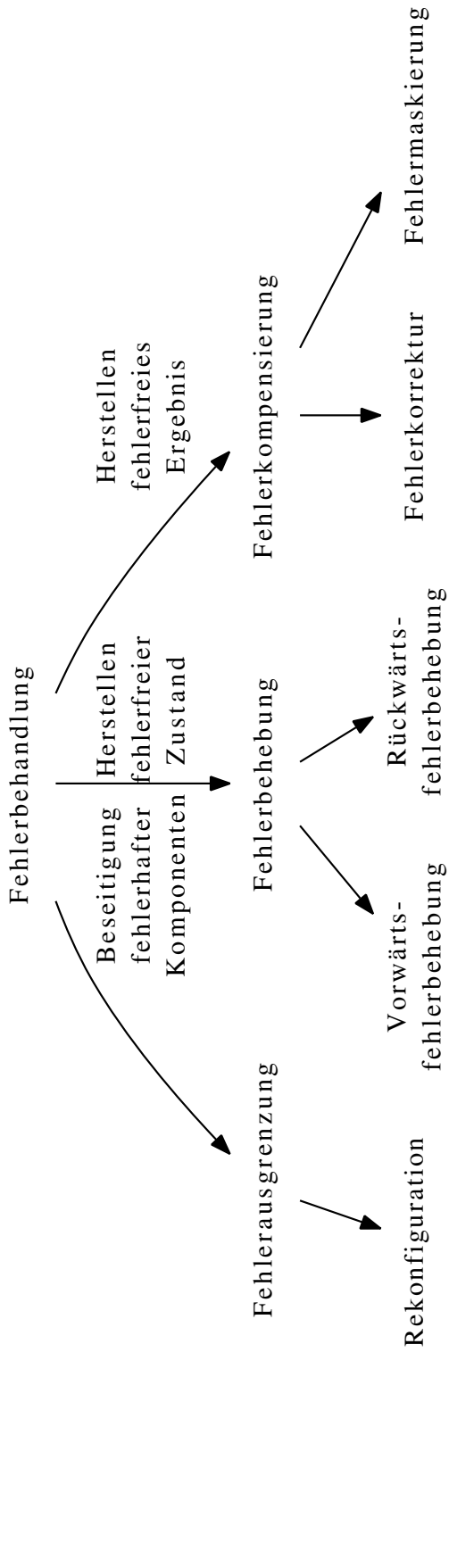


Abbildung 3.2.: Übersicht der Maßnahmen zur Fehlerbehandlung [Ech90]

konfiguration einer numerischen Steuerung bei Ausfall einer Maschinenachse auf eine redundant vorhandene Achse zu adaptieren.

Mit einer *Ausgliederung* wird ein fehlerhaftes Teilsystem aus dem Gesamtsystem entfernt. Eine *Eingliederung* fügt dem System ein fehlerfreies Teilsystem hinzu.

3.2.2. Fehlerbehebung

Das Ziel der *Fehlerbehebung* ist das Wiederherstellen eines Zustands ohne Fehler. Diesem Unterpunkt sind die Vorwärts- und die Rückwärtsfehlerbehebung zugeordnet.

Rückwärtsfehlerbehebung bedeutet, nach Erkennen eines Fehlers die aktuellen Ergebnisse zu verwerfen und das System an einen definierten und bekannten fehlerfreien Zustand zurückzusetzen. Ab diesem Punkt werden die folgenden Berechnungen nochmals durchgeführt. Die Rückwärtsfehlerbehebung bedingt, dass die Software um Rückfallpunkte erweitert wird, an welche bei Erkennen eines Fehlers zurückgesprungen und wieder aufgesetzt werden kann.

Kompliziert wird die Rückwärtsfehlerbehebung sobald mehrere Programme am Prozess beteiligt sind und Daten miteinander austauschen. Hierbei muss beachtet werden, dass alle Programme, die mit Daten des als fehlerhaft erkannten Prozesses gearbeitet haben, ebenfalls auf einen geeigneten Rückfallpunkt zurückgesetzt werden. Findet viel Datenaustausch zwischen den Prozessen statt, so wird die Bestimmung der richtigen Rückfallpunkte zunehmend komplex.

Ebenfalls nachteilig ist der erhöhte Zeitbedarf nach Erkennen eines Fehlers. Hierbei müssen alle Berechnungen seit dem letzten Rückfallpunkt nochmals durchgeführt werden. Dies mag für Systeme durchführbar sein, die nur für sich selbst arbeiten und zeitunabhängig lediglich an einem fehlerfreien Ergebnis interessiert sind. Jedoch ergibt sich ein Problem, wenn für die Berechnung nur begrenzte Zeit zur Verfügung steht, wie es bei numerischen Steuerungen mit Zykluszeiten im Millisekundenbereich der Fall ist. Ebenfalls nicht anwendbar ist die Rückwärtsfehlerbehebung in Fällen, bei denen ein realer, irreversibler

Prozess gesteuert wird. So lässt sich einmal zu viel abgetragenes Material nicht wieder anbringen.

Die *Vorwärtsfehlerbehebung* hat zum Ziel, ein System nach Erkennen eines Fehlers wieder in einen konsistenten Zustand zu versetzen. Dabei wird nicht auf in der Vergangenheit gespeicherte Zustände zurückgegriffen. Vielmehr trifft die Vorwärtsfehlerbehebung auf Basis des erkannten Fehlers Annahmen wie mit dem Fehler umgegangen werden muss.

Die Schwierigkeit der Vorwärtsfehlerbehebung liegt darin, dass jeder denkbare Fehler in der Entwurfs- und Spezifikationsphase berücksichtigt werden muss. Für jeden dieser Fehler muss ebenso eine Behebungsstrategie definiert werden. Ein typisches Anwendungsbeispiel für eine Vorwärtsfehlerbehebung ist die Strategie des „sicheren Stillsetzens“ einer Maschine nach Erkennen eines Fehlers.

3.2.3. Fehlerkompensierung

Das Ziel der *Fehlerkompensierung* ist das Herstellen einer fehlerfreien Funktion. Dies geschieht durch Anwenden von Fehlerkorrektur oder Fehlermaskierung.

Die *Fehlerkorrektur* funktioniert nur bei Datenübertragung. Dazu werden Daten beim Speichern oder Übertragen mit redundanter Information angereichert. Der Werteraum des eigentlichen Datums wird dazu in einen erweiterten Werteraum abgebildet, so dass alle möglichen Werte im erweiterten Werteraum einen gleichen Abstand haben. Der Abstand zwischen benachbarten Werten im neuen Raum ist die sogenannte Hammingdistanz. Das ursprüngliche Datum kann bei Speicher- oder Übertragungsfehlern unterhalb der halben Hammingdistanz wieder berechnet werden.

Fehlermaskierung bedeutet fehlerhafte Ergebnisse auszugrenzen und zu ignorieren. Dies ist nur dann möglich, wenn das selbe Ergebnis mehrfach, das heißt redundant, vorliegt. In diesem Fall wählt ein Entscheider aus der Menge der redundant berechneten Ergebnissen eines zur Weitergabe aus. Echtle gibt in

[Ech90] die Methoden Medianentscheidung, Intervallentscheidung und Kugelentscheidung als Vorgehensweisen für die Maskierungsentscheidung an.

Bei der Medianentscheidung wird eine eindimensionale Menge von Ergebnissen aufsteigend sortiert. Das Element in der Mitte der sortierten Menge von Ergebnissen entspricht dem Median.

Die Intervallentscheidung funktioniert mit mehrdimensionalen Ergebnissen. Sie definiert für jede Dimension ein zulässiges Intervall. Daraufhin sucht sie eine Menge von möglichst vielen Ergebnissen, deren Werte die zulässigen Intervalle einhalten. Ist eine entsprechende Menge von Ergebnissen gefunden, so wird ein beliebiges Ergebnis daraus ausgewählt. Dieses Ergebnis hat maximal die Abweichung des vorab als zulässig definierten Intervalls. Die Kugelentscheidung ist eine Spezialisierung der Intervallentscheidung, bei der für alle Dimensionen das selbe zulässige Intervall verwendet wird.

3.2.4. Bewertung der allgemeinen Maßnahmen

Die Methoden der Fehlerausgrenzung sind hinsichtlich der Zielsetzung aus Kapitel 2.4 nicht ausreichend. Dies liegt daran, dass eine Rekonfigurationsmaßnahme die Auswirkungen einer Störung nicht beseitigt, sondern lediglich ein störungsfreies System wiederherstellt. So fehlt bei einer Verlagerung oder einem eingegliederten, störungsfreien Teilsystem der letzte fehlerfreie Systemzustand. Diesen kann es nur durch eine Rückwärtsfehlerbehebung wiederherstellen. Alternativ kann es durch eine Vorwärtsfehlerbehebung einen konsistenten Systemzustand erreichen. Die unterbrechungsfreie Funktion kann die Fehlerausgrenzung alleine jedoch nicht gewährleisten.

Die Methoden der Fehlerbehebung sind als Maßnahmen ebenfalls ungeeignet. Die Vorwärtsfehlerbehebung ist aufgrund der hohen Komplexität einer numerischen Steuerung nicht mit der Zielsetzung aus Kapitel 2.4 vereinbar. So müsste für jeden möglichen Fehler eine geeignete Vorwärtsfehlerbehebung zur fehlerfreien Aufrechterhaltung des Bearbeitungsprozesses spezifiziert und implementiert werden. Die Rückwärtsfehlerbehebung scheitert an den sehr hohen

Echtzeitanforderungen eines numerischen Steuerungssystems sowie der Spezifikation der Rückfallpunkte, die in bestehenden numerischen Steuerungen berücksichtigt werden müssen.

Von den Methoden zur Fehlerkompensation ist nur die Fehlermaskierung einsetzbar. Die Fehlerkorrektur ist ungeeignet, da sie nur bei der Speicherung oder Übertragung eines Datums wirkt. Im Fall der numerischen Steuerung kann jedoch schon das Datum an sich fehlerhaft erzeugt worden sein. Richtige Daten können jedoch nur durch Wiederholen der Rechenschritte neu berechnet werden, was einer Rückwärtsfehlerbehebung entspricht.

Für den Entscheider der Fehlermaskierung muss jeder Achse ein für die Anwendung noch als korrekt geltender Toleranzbereich zugewiesen werden. Dieser Toleranzbereich ist notwendig, da numerische Steuerungen für Berechnungen Fließkommavariablen benutzen, die nach Goldberg [Gol91] bei jeder Operation Rundungsfehlern unterliegen. Nach [Mir04] dürfen Fließkommavariablen niemals auf Gleichheit überprüft werden. Stattdessen soll geprüft werden, ob der Betrag ihrer Differenz für den Anwendungsfall hinreichend klein ist.

3.3. Strukturelle Maßnahmen zur Fehlermaskierung

Damit eine Fehlermaskierung überhaupt erst möglich wird, müssen redundant berechnete Ergebnisse vorliegen anhand derer ein plausibles Ergebnis bestimmt wird. Das Ergebnis sind in diesem Fall zyklisch auszugebende Sollwerte des numerischen Steuerungssystems. Zur Kompensierung von sporadisch auftretenden sowie permanenten Hardwarefehlern muss auch die Steuerungshardware redundant ausgelegt sein. Damit auch systematische Hardwarefehler kompensiert werden können, muss die Hardware zusätzlich diversitär sein und darf damit nicht aus herstellungsgleichen Komponenten bestehen. [LG99]

Frentzen entwirft in [Fre87, WF87] bereits 1987 eine auf Mikrocontrollern basierende fehlersichere Exzenterpressensteuerung. Um mit Mikrocontrollern

die selbe Sicherheit wie zuvor mit fester Relaisverdrahtung zu erreichen, wählt er eine redundante 3 von 3 Struktur mit Hardwarediversität. Zum Datenaustausch ist jeder Mikrocontroller mit jedem anderen fest verdrahtet, ein Bussystem wird nicht eingesetzt. Zur Programmierung des Steuerungsprogramms implementiert Frentzen einen Compiler für Anweisungslisten (AWL) nach DIN EN 61131 [DIN03], der AWL in die Maschinensprache der eingesetzten Mikrocontroller übersetzt. Im Ausblick beschreibt er die Möglichkeit eine numerische Steuerung durch gleiche strukturelle Maßnahmen fehlersicher zu machen.

Zudem stellt Frentzen Entwurfskriterien für mehrkanalige Strukturen von speicherprogrammierbaren Steuerungen auf. So fordert er eine unabhängige mehrkanalige Bearbeitung von Steuerungsaufgaben, deren Einzelkanäle bezüglich der spezifizierten Funktion gleichwertig sein müssen. Fehlersicherheit und Fehlertoleranz mit Tolerierung eines Fehlers werden gefordert, ebenso dass Störungen sich nicht gleichartig auf die Einzelkanäle auswirken dürfen. Die Fehlererkennung soll schnellstmöglich erfolgen. Weiterhin wird gefordert, dass ein Fehler keine Auswirkung auf die Kommunikation anderer Komponenten haben darf. Die Tabelle 3.1 aus [Fre87] fasst die darin gestellten Entwurfskriterien mit Anforderungen und Lösungsvorschlägen zusammen.

Neben Frentzen behandelt auch die DIN VDE 0801 [VDE90] ausgiebig verschiedene Strukturen zur Behandlung von Fehlern auf Systemebene. Jede Steuerungsstruktur wird darin hinsichtlich unterschiedlicher Kriterien bewertet. Die Tabelle 3.2 zeigt eine Übersicht über alle darin behandelten Strukturen sowie deren Bewertung.

Die strukturellen Maßnahmen aus Tabelle 3.2 sind statischer Natur und entweder auf Fehlerbeherrschung durch Abschaltung oder auf Fehlerkompensation durch Umschaltung oder Tolerierung ausgerichtet. Für Maßnahmen der Fehlerkompensation kann nach Auftreten und Erkennen eines Fehlers zusätzlich eine Rekonfiguration erfolgen. Diesen Fall bezeichnet Echtle [Ech90] als *hybride Redundanz*. Hierbei werden fehlerhafte Systeme ausgegliedert und gegebenenfalls durch Eingliederung störungsfreier Systeme der Ausgangszustand wiederhergestellt.

Komponente	**Anforderung**	**Lösung**
Steuerungs-struktur	Mehrkanalige Bearbeitung unabhängig von der Steuerungsaufgabe.	Einzelkanäle müssen bezüglich ihrer Funktion gleichwertig sein.
Redundanz-prinzip	*Fehlersicher:* Erkennung offensichtlicher und verdeckter Fehler. *Verfügbarkeit:* Tolerierung eines Fehlers.	Dreifache Redundanz.
Aufbau der Einzelkanäle	Störungen dürfen sich nicht gleichartig auf alle Einzelkanäle auswirken.	Diversität in Hardware und Software.
Vergleicher-prinzip	Schnellstmögliche Fehlererkennung.	Programmtechnisch realisierter Datenvergleich innerhalb der drei Rechner.
Rechner-kopplung	Ein Fehler darf die Kommunikation zweier Rechner nicht stören.	Kopplung jeweils zweier Rechner mit unabhängigen Übertragungskanälen.

Tabelle 3.1.: Entwurfskriterien mehrkanaliger Steuerungsstrukturen [Fre87]

● Kriterium erfüllt
◕ ...
◑ Kriterium teilweise erfüllt
◔ ...
○ Kriterium nicht erfüllt

n/a nicht vorhanden
R Redundanz
Z Zyklische Tests
& Zyklische Tests und Redundanz

	Fehlerbezogenheit (Wirksamkeit)	Verfügbarkeit	zyklische Tests / Redundanz	Zeitbedarf	Zeitverhalten bzgl. Fehlererkennung	Entwicklungsaufwand	Herstellungsaufwand	Nachweisaufwand	Art der Fehlerreaktion
Einkanalig mit Funktionsprüfung	○	n/a	Z	●	○	◕	◕	◕	n/a
Einkanalig mit Selbsttest	○	n/a	Z	◔	◑	◔	◔	◔	Abschaltung
Eink. mit Selbsttest u. Überwachungseinr.	◔	n/a	Z	◔	◑	◔	◔	◔	Abschaltung
Zweikanalig mit Fpr. und Umschalter	○	◑	&	●	○	◕	◔	◕	Umschaltung
Zweikanalig mit Selbsttest und Umschalter	◔	◑	&	◔	◑	◔	◔	◔	Umschaltung
Zweikanalig mit Vergleich	◕	n/a	&	◕	◕	◔	○	◕	Sichere Abschaltung
1 von 3 mit Funktionsprüfung	○	●	Z	●	○	○	○	◕	Umschaltung
1 von 3 mit Selbsttest	○	●	Z	◔	◑	○	○	◔	Umschaltung
1 von 3 mit Selbsttest und Überwachung	◔	●	Z	◔	◑	○	○	◔	Umschaltung
2 von 3 mit Mehrheitsentscheidung	●	◑	R	◑	●	○	○	◑	Tolerierung eines Fehlers
3 von 3 mit Vergleich	●	○	R	◑	●	○	○	◑	Abschaltung
1 von n mit Funktionsprüfung	○	●	Z	●	○	○	○	◕	Umschaltung
1 von n mit Selbsttest	○	●	Z	◔	◑	○	○	◔	Umschaltung
1 von n mit Selbsttest und Überwachung	◔	●	Z	◔	◑	○	○	◔	Umschaltung
m von n mit Mehrheitsentscheidung	●	◑	R	◑	●	○	○	◑	Tolerierung von (n-m) Fehlern
n von n mit Vergleich	●	○	R	◑	●	○	○	◑	Abschaltung

Tabelle 3.2.: Übersicht der strukturellen Maßnahmen nach [VDE90]

Ein Spezialfall dieser hybriden Redundanz ist nach Lala [Lal85] die *selbstreinigende Redundanz*. Im Gegensatz zur hybriden Redundanz nimmt selbstreinigende Redundanz ausschließlich Ausgliederungen vor. Sie liegt in Systemen vor, die im Betrieb weder repariert werden können, noch eine Eingliederung störungsfreier Systeme erlauben. Mit selbstreinigender Redundanz wird aus einem m von n System nach Ausgliederung eines störungsbehafteten Teilsystems ein m' von n' System für das $n' < n$ und $m' \leq m$ gilt.

Die selbstreinigende Redundanz hat nach [Ech90, Lal85] eine höhere Überlebenswahrscheinlichkeit als alle anderen statischen Redundanzformen. So toleriert ein 3 von 5 System zwei Ausfälle. Ein selbstreinigendes Redundanzsystem gliedert fehlerhafte Teilsysteme aus und durchläuft so die Kette „3 von 5“ über „3 von 4“ bis „2 von 3“ und toleriert somit drei Ausfälle.

3.4. Weitere Forschungsarbeiten

Schneider behandelt in seinem Beitrag zur Fehlertoleranz in [Sch94a] Fehlerreaktionen für speicherprogrammierbare Steuerungen. Er gibt für 20% aller Systemausfälle einen Ausfall der Steuerung als Ursache an und behandelt diesen Fall theoretisch sowie allgemein. Sein Lösungsansatz beschreibt Fehlerreaktionsverfahren innerhalb von SPS Programmen. Einen Ausfall der speicherprogrammierbaren Steuerung selbst behandelt er nicht.

Wagner erarbeitet in [Wag96] eine steuerungsintegrierte Fehlerbehandlung für maschinennahe Abläufe. Das Verfahren fasst alle Maschinenzustände zu einem Zustandstupel zusammen. Im Fehlerfall, beispielsweise einer bevorstehenden Kollision oder inkonsistentem Zustand von Sensorik/Aktorik, werden ausgehend von diesem Tupel in einer zur Maschine gehörenden Wissensdatenbank eine Abfolge von Einzelaktionen gesucht. Diese vordefinierten Einzelaktionen überführen die Maschine in einer Vorwärtsfehlerbehebung vom Ausgangszustand in einen fehlerfreien Zustand. Mit diesem Verfahren werden Fehler in der Maschine behandelt, jedoch kein Ausfall des Steuerungssystems.

Erdner entwirft in [Erd03] ein realzeitfähiges und fehlertolerantes Feldbussystem. Für die Realisierung der Realzeitfähigkeit diskutiert er ringförmig aufgebaute Kommunikationssysteme für deren Kommunikationsteilnehmer er die Signallaufzeiten ermittelt und zur Bestimmung eines globalen Synchronisationszeitpunkts heranzieht. Die Fehlertoleranz bezieht er rein auf die Datenübertragung für die er zwei verschiedene Verfahren anwendet. Zum einen nutzt er ein modifiziertes FSK Verfahren [Rie95] zur Erhöhung der Störsicherheit auf der Übertragungsstrecke. Zum anderen setzt er eine Hamming Kodierung ein, mit der Bitfehler bei der Übertragung im Empfänger nicht nur erkannt, sondern direkt korrigiert werden können. Durch dieses Verfahren umgeht er den Nachteil einer zeitaufwendigen Wiederholung der Datenübertragung nach dem Feststellen eines Übertragungsfehlers [Sch94b]. Numerische Steuerungen werden in der Arbeit als Nutzer solch eines Feldbussystems erwähnt jedoch nicht weiter vertieft.

Staudt beschreibt zur Validierung von Geräten in [Sta06] Verfahren, durch welche die Kommunikationsschnittstelle von Geräten wie Steuerungen oder Servoantrieben gegenüber einer Spezifikation geprüft werden können. Diese Verfahren ermöglichen das Aufdecken von systematischen Implementierungsfehlern vor der Inbetriebnahme.

In [Lai05] geht Laible intensiv auf die Problemstellung ein, wie ein durchgängiges Sicherheitskonzept für numerische Steuerungssysteme ausgelegt sein muss, um ohne Funktionseinschränkung mögliche Gefährdungen auszuschließen. Er beschreibt darin Maßnahmen zur Fehlererkennung, den Aufbau zweikanalig redundanter Steuerungen sowie Maßnahmen zur Fehlerreaktion. Fehlertoleranz wird kurz durch die grundsätzliche Möglichkeit der Nutzung einer 2 von 3 Struktur aufgezeigt, wegen der Fokussierung auf Fehlersicherheit jedoch nicht tiefer behandelt.

In Tabelle 3.3 sind diese Forschungsarbeiten hinsichtlich ihrer Schwerpunkte bewertet.

● hoch ◕ ... ◑ mittel ◔ ... ○ niedrig	Schneider, J. [Sch94a]	Wagner, M. [Wag96]	Erdner, T. [Erd03]	Staudt, S. [Sta06]	Echtle, K. [Ech90]	Bürger, T. [Bür05]	Laible, U. [Lai05]	Hein, A. [Hei00]	Frentzen, B. [Fre87]
Numerische Steuerung	○	○	○	○	○	●	●	○	○
SPS	●	●	○	○	○	○	○	●	●
Datenübertragung	○	○	●	●	○	○	◑	○	○
Fehlersicherheit	○	○	○	○	○	●	●	●	●
Fehlertoleranz	◑	◑	◕	○	●	○	◔	○	○
Echtzeit	◔	◑	●	●	○	○	●	○	◑

Tabelle 3.3.: Übersicht der Forschungsarbeiten

3.5. Die fehlersichere numerische Steuerung

Abbildung 3.3 aus [VB08] zeigt das Konzept der fehlersicheren numerischen Steuerung nach [Lai05]. Sie besteht aus zwei numerischen Steuerungen, die auf diversitären Hardwareplattformen ablaufen. Beide Steuerungen synchronisieren sich dabei durch Nutzung eines gemeinsamen Speichers, über den sie zudem Berechnungsergebnisse austauschen. Sollwerte erhält die Aktorik hier ausschließlich von Steuerung 2.

Jede der beiden Steuerungen beinhaltet dabei einen Reaktionsmanager. Diese vergleichen die eigenen berechneten Werte mit den Werten der jeweils anderen Steuerung. Stellt einer der beiden Reaktionsmanager zu große Differenzen fest, so kann er der Aktorik selbständig die Freigabe entziehen.

Tritt folglich in diesem fehlersicheren Steuerungssystem eine Störung in einer der beiden Hardwareplattformen auf, die zu fehlerhaft berechneten Sollwerten führt, so löst ein Reaktionsmanager aufgrund der Sollwertdifferenz einen

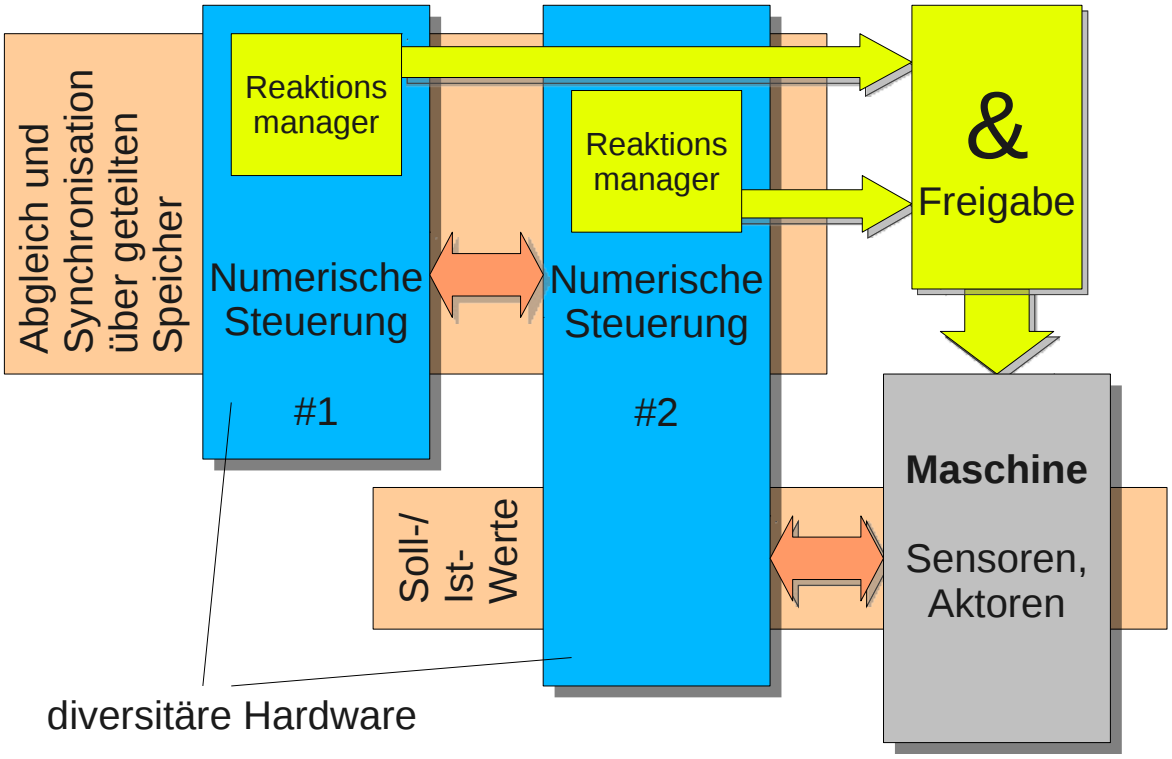


Abbildung 3.3.: Struktur der fehlersicheren numerischen Steuerung

Not-Halt aus. Dies funktioniert selbst wenn der Fehler so weitreichend ist, dass die Hardwareplattform ihn nicht mehr selbst erkennen kann. In diesem Fall wird die redundante Plattform dies erkennen und die Maschine durch Stillsetzen in einen sicheren Zustand überführen.

Die erste redundante numerische Steuerung im fehlersicheren Steuerungsaufbau läuft auf einem Steuerungs-PC mit PCI-Bus. Die zweite redundante numerische Steuerung läuft als Steckkarten PC auf einer PCI Einsteckkarte, die in den PCI-Bus des Steuerungs-PC eingebaut ist. Der Steckkarten PC enthält zusätzlich einen Speicherbereich, den sogenannten Dual Ported RAM, den er über den PCI-Bus mit dem Steuerungs-PC teilt. Sowohl der Steckkarten PC als auch der Steuerungs-PC kann direkt auf diesen Speicher zugreifen.

Abbildung 3.4 stellt den Aufbau der fehlersicheren numerischen Steuerung bezüglich Abgleich und Synchronisation dar. Jede numerische Steuerung besteht dabei aus einem nieder- und einem hochprioren Prozess. Der hochpriore Prozess behandelt alle zeitkritischen Aufgaben wie die Verarbeitung von Echt-

zeitsignalen sowie die Bahninterpolation. Er wird hierzu durch einen Timer oder ein externes Taktsignal zyklisch aufgerufen. Der niederpriore Prozess führt freilaufend alle übrigen Aufgaben aus, wie zum Beispiel die Bahnplanung oder die Kommunikation mit einer Bedienoberfläche. Der niederpriore Prozess kann jederzeit durch den höherprioren Prozess unterbrochen werden, wodurch Laufzeitschwankungen auftreten.

Sowohl im hochprioren als auch im niederprioren Prozess sind mehrere Synchronisationspunkte vorgesehen. Diese sind insbesondere am Eingang der Steuersignale der Maschinensteuertafel sowie zum Verarbeiten von Prozesssignalen der Technologiedatenverwaltung nötig, da externe Signale synchronisiert übernommen werden müssen. Der hochpriore Prozess benötigt nach dem Eingang keine weiteren Synchronisationspunkte, da er strikt deterministisch abläuft und auf beiden Steuerungen zeitgleich gestartet wird. Im niederprioren Prozess sind weitere Synchronisationspunkte enthalten, um die asynchrone Abarbeitung der Aufgaben effizienter zu gestalten. Notwendig bezüglich der Synchronisation der Gesamtsteuerung sind diese jedoch nicht.

3.6. Verbesserungspotential bestehender Lösungen

Der in Kapitel 3.5 beschriebene Ansatz eines fehlersicheren numerischen Steuerungssystems beinhaltet bereits hinreichend viele Maßnahmen zur Vermeidung von Fehlern. Zum Erkennen von Fehlern während des Betriebs ist das Steuerungssystem zweikanalig mit diversitärer Hardware ausgelegt. Auftretende Fehler werden durch dieses System behandelt, indem nach der Fehlererkennung das System mittels Notabschaltung in einen sicheren Zustand wechselt. Dies entspricht einer Vorwärtsfehlerbehebung zum Herstellen eines fehlerfreien Zustands.

Der Ansatz der fehlersicheren numerischen Steuerung entspricht der Maßnahme „Zweikanalig mit Vergleich“ aus Tabelle 3.2. Diese Struktur zeichnet sich im Gegensatz zu einem Einfachsystem durch gute Fehlerbezogenheit, ge-

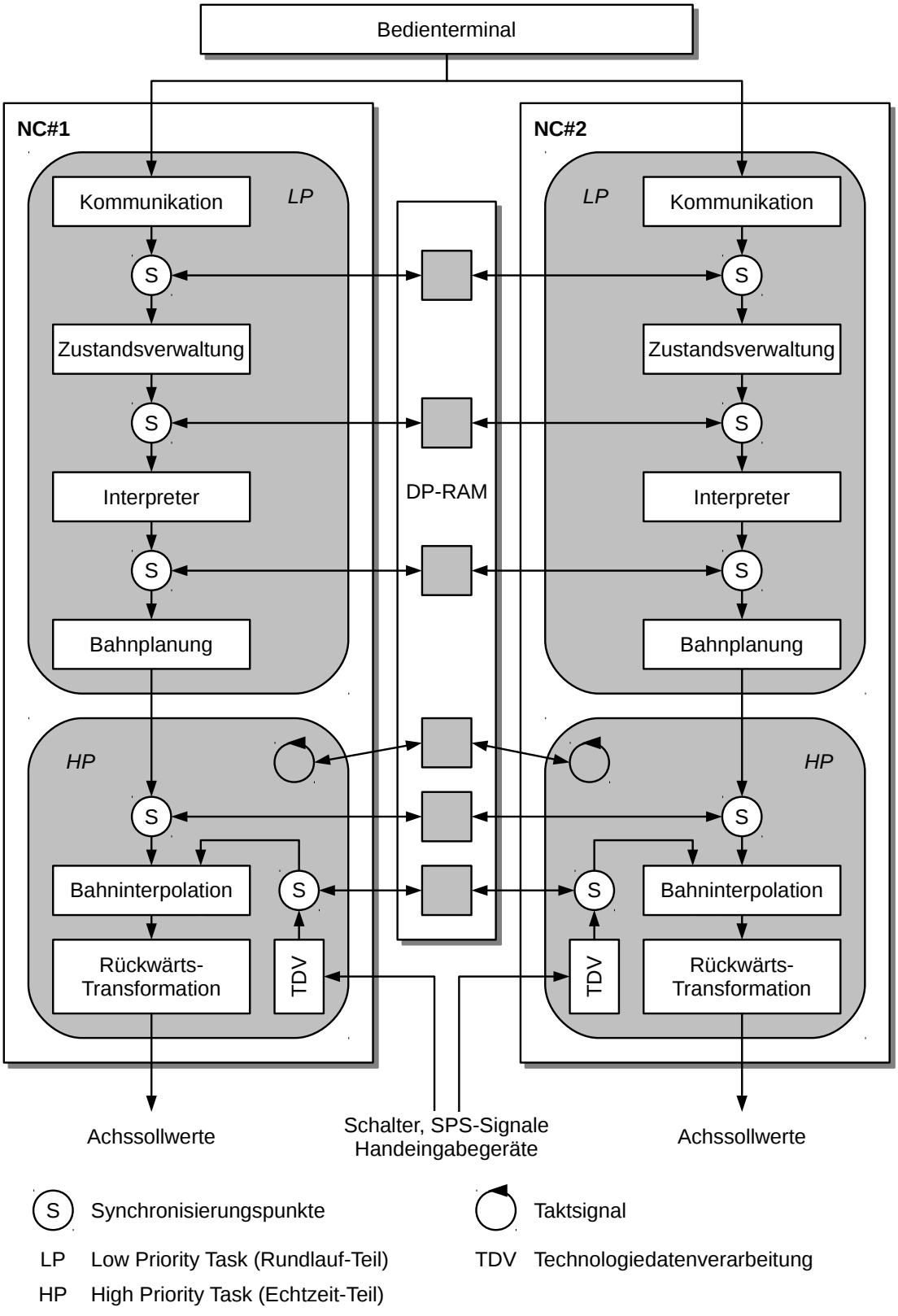


Abbildung 3.4.: Anordnung der Synchronisationspunkte der fehlersicheren Steuerung [Lai05]

ringen Zeitbedarf, gutem Zeitverhalten bezüglich der Fehlererkennung sowie vertretbarem Aufwand bei Nachweis der sicheren Funktion aus.

Der Vorteil der hinzugewonnenen Schadenfreiheit wird bei dieser Struktur jedoch mit dem Nachteil einer verminderten Überlebenswahrscheinlichkeit erkauft. Berechnet sich nach Formel 2.6 die Überlebenswahrscheinlichkeit eines einfachen Systems zu Formel 3.1, so hat ein fehlersicheres System aus zwei redundanten Steuerungen, die zum Betrieb beide fehlerfrei arbeiten müssen, nach Formel 3.2 eine geringere Überlebenswahrscheinlichkeit.

$$\begin{aligned} R_{1\ von\ 1}(t) &= R(t) \\ &= e^{-\lambda t} \end{aligned} \tag{3.1}$$

$$\begin{aligned} R_{2\ von\ 2}(t) &= R(t) * R(t) \\ &= e^{-2\lambda t} \end{aligned} \tag{3.2}$$

In Abbildung 3.5 sind beide Überlebenswahrscheinlichkeiten graphisch über die Zeit dargestellt. Darin ist offen ersichtlich, dass die fehlersichere numerische Steuerung eine geringere Zuverlässigkeit besitzt als ein Einfachsystem, das im Fehlerfall keine Schadenfreiheit bietet.

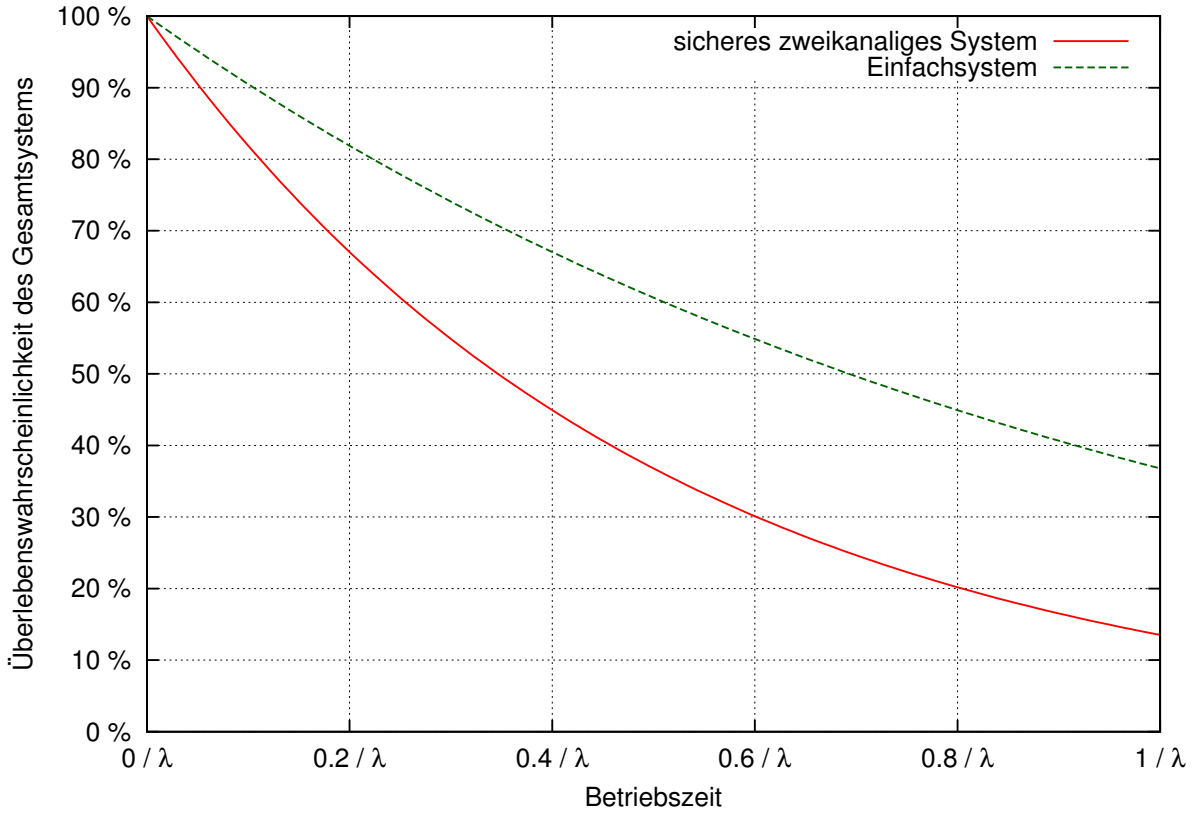


Abbildung 3.5.: Überlebenswahrscheinlichkeiten von Einfachsystem und fehlersicherem zweikanaligen System

4. Lösungsansatz

Der zweikanalige Ansatz zum sicheren Erkennen von Fehlern kann die Wahrscheinlichkeit eines Schadens signifikant senken. Die Wahrscheinlichkeit eines Ausfalls des Systems nimmt, wie in Kapitel 3.6 gezeigt, durch die höhere Anzahl von zum Betrieb notwendigen Komponenten zu. Damit sinkt die Zuverlässigkeit dieses Systems.

Das Ziel dieser Arbeit ist, unter Beibehaltung der hinzugewonnenen Sicherheit, die Erhöhung der Zuverlässigkeit. Im Umkehrschluss bedeutet dies die Verringerung der Wahrscheinlichkeit, dass es nach Beginn des Bearbeitungsprozesses durch Fehler bei der Sollwerterzeugung zu einem Ausfall oder Schaden kommt.

Um die Zuverlässigkeit eines fehlersicheren numerischen Steuerungssystems zu erhöhen sind allgemeine Maßnahmen zur Fehlerbehandlung aus Kapitel 3.2 notwendig. Die Fehlerbehebung wird dabei außen vor gelassen, da diese einen massiven Eingriff in die internen Strukturen der numerischen Steuerung darstellen würde. Der Schwerpunkt wird stattdessen auf Maßnahmen zur Fehlerkompensierung und Fehlerausgrenzung gelegt. Die Fehlerkompensierung soll fehlerhafte Ergebnisse maskieren und die Fehlerausgrenzung soll als fehlerhaft erkannte Systeme ausgliedern. Eine Eingliederung oder Verlagerung ist an dieser Stelle nicht vorgesehen, da diese genau wie die Fehlerbehebung umfangreiche Eingriffe in die internen Strukturen der numerischen Steuerung nach sich zieht. Die Fehlerkorrektur wird ebenfalls nicht benutzt, da diese nur bei Speicherung oder Übertragung von Daten wirkt, nicht jedoch bei der Berechnung derselben.

Damit eine Fehlerkompensierung sowie eine Ausgliederung fehlerhafter Systeme durchgeführt werden kann bedarf es struktureller Maßnahmen nach Kapitel 3.3. Da ein realer Prozess in der Regel nicht darauf wartet, dass eine Steuerung einen gültigen Sollwert berechnet und ausgibt, sind der Fehlerkompensierung deterministische Zeitbedingungen auferlegt. Für die spezifizierte Funktion einer numerischen Steuerung bedeutet dies für die Fehlerkompensierung, dass sie im Interpolationstakt zyklisch und rechtzeitig gültige Sollwerte an die Aktorik ausgeben muss.

Hinsichtlich der fehlertoleranten Struktur ist damit ein sehr gutes Zeitverhalten der Maßnahme essentiell. Damit der Bearbeitungsprozess durch die Fehlerreaktion keine Unterbrechung erfährt kommen nur Maßnahmen in Frage, die eine sofortige Umschaltung vornehmen oder eine Maskierung durchführen können. Maßnahmen, die eine Abschaltung durchführen unterbrechen den Prozess und sind damit ungeeignet. Ebenso wichtig ist die Fehlerbezogenheit der Maßnahme, damit ein Fehler erkannt und auch in geeigneter Art und Weise behandelt wird.

Beim Auswerten der Tabelle 3.2 hinsichtlich dieser Kriterien fällt auf, dass das Kriterium Zeitverhalten bereits alle Maßnahmen ausschließt, die eine Umschaltung vornehmen. Übrig bleiben damit die beiden strukturellen Maßnahmen:

- 2 von 3 mit Mehrheitsentscheidung
- m von n mit Mehrheitsentscheidung

4.1. Wirksamkeit der Maßnahmen

Die Zuverlässigkeitsfunktion dieser fehlerkompensierenden Systeme berechnet sich allgemein nach Formel 2.8. Abgeleitet daraus berechnet sich die Zuverlässigkeitsfunktion für ein 2 von 3 System zu Formel 4.1. Abbildung 4.1 zeigt am Beispiel eines 2 von 3 Systems, dass diese strukturellen Maßnahmen die Zuverlässigkeit gegenüber dem fehlersicheren Ansatz deutlich erhöhen.

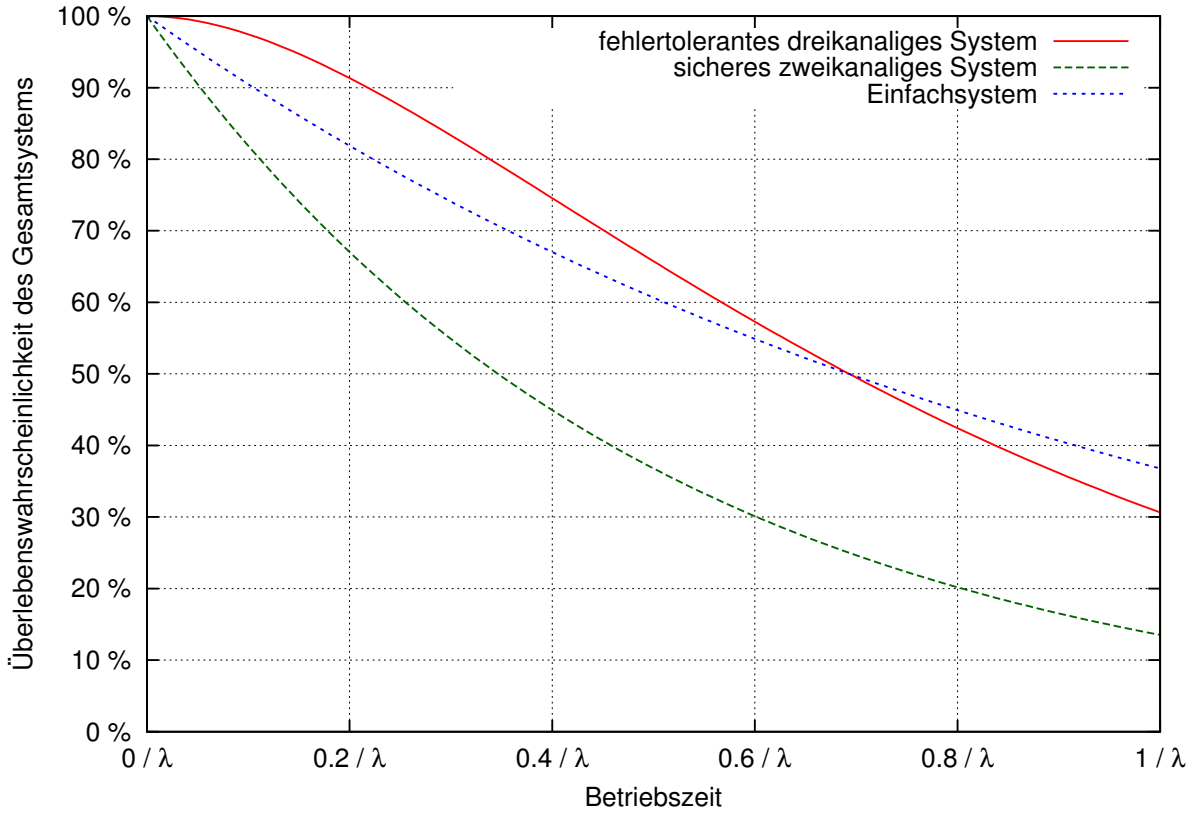


Abbildung 4.1.: Überlebenswahrscheinlichkeiten über Betriebszeit

$$\begin{aligned} R_{2\ von\ 3}(t) &= \sum_{i=2}^{3} \binom{3}{i} R^i(t)(1-R(t))^{3-i} \\ &= 3e^{-2\lambda t} - 2e^{-3\lambda t} \end{aligned} \tag{4.1}$$

Die höhere Zuverlässigkeit des 2 von 3 Systems gilt insbesondere für den Beginn der Betriebsdauer eines zunächst fehlerfreien Systems wie Abbildung 4.2 für Systeme mit einem MTBF von 70.000 Stunden über eine Betriebsdauer für 25 Stunden beispielhaft aufzeigt.

In Abbildung 4.2 verläuft die Überlebenswahrscheinlichkeit des 2 von 3 Systems fast deckungsgleich mit der 100% Linie. Nachvollziehen lässt sich dies, indem die Ableitung der Überlebenswahrscheinlichkeit nach der Zeit in Formel 4.2 zum Zeitpunkt $t = 0$ berechnet wird. Das Ergebnis in Formel 4.5 ergibt

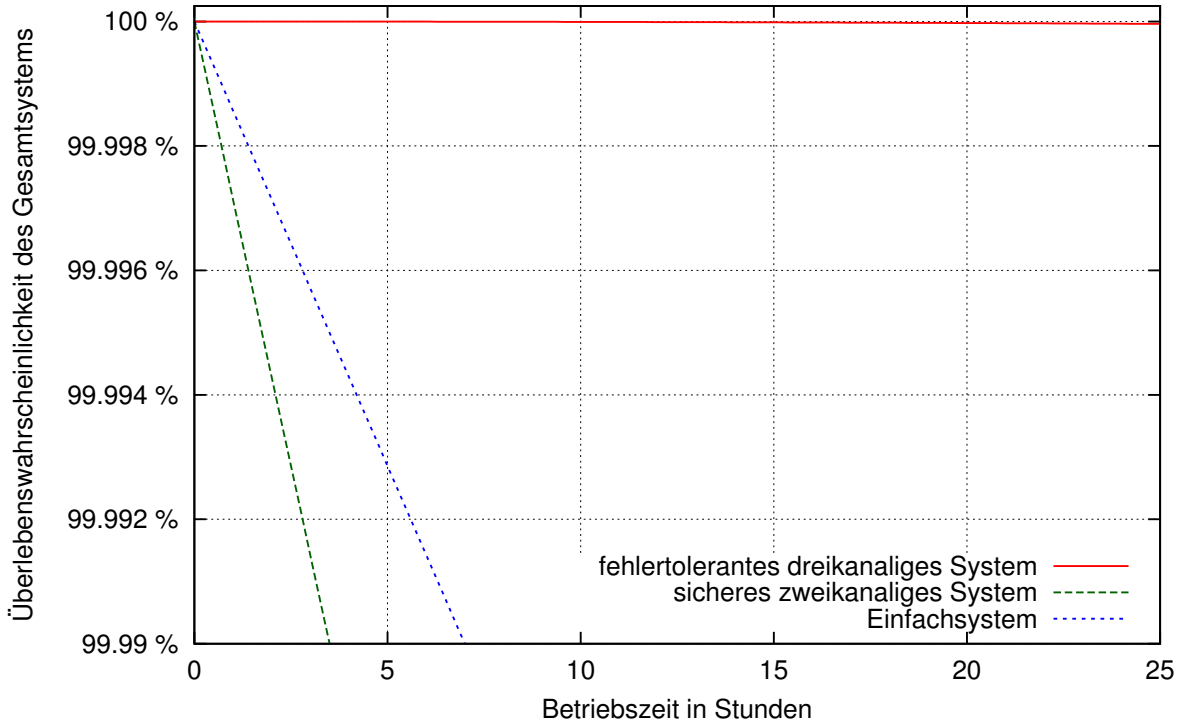


Abbildung 4.2.: Vergrößerte Überlebenswahrscheinlichkeiten zu Betriebsdauerbeginn

0 und erklärt, warum die Überlebenswahrscheinlichkeit bei 100% verweilt. Die Ableitungen des 2 von 2 sowie des Einfachsystems nach den Formeln 4.3 und 4.4 ergeben abhängig von λ nach den Formeln 4.6 und 4.7 eine negative Steigung.

$$\dot{R}_{2von3}(t) = -6\lambda e^{-2\lambda t} + 6\lambda e^{-3\lambda t} \tag{4.2}$$

$$\dot{R}_{2von2}(t) = -2\lambda e^{-2\lambda t} \tag{4.3}$$

$$\dot{R}_{1von1}(t) = -\lambda e^{-\lambda t} \tag{4.4}$$

$$\dot{R}_{2von3}(0) = 0 \tag{4.5}$$

$$\dot{R}_{2von2}(0) = -2\lambda \tag{4.6}$$

$$\dot{R}_{1von1}(0) = -\lambda \tag{4.7}$$

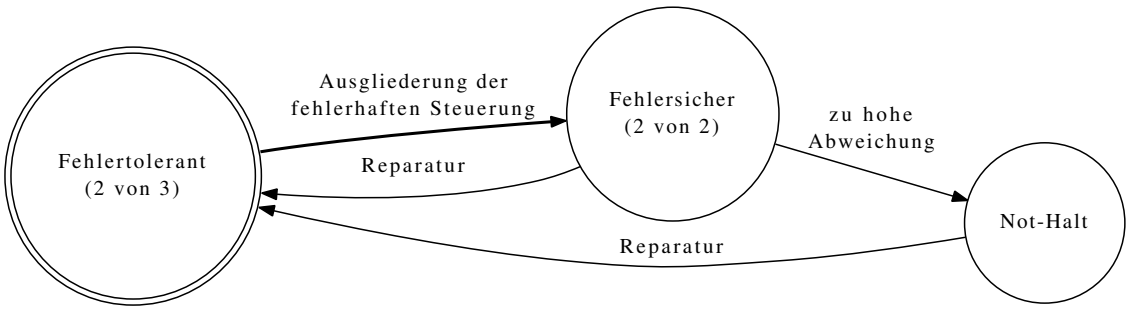


Abbildung 4.3.: Zustandsautomat für Entscheider

4.2. Rekonfiguration bei zwei verbleibenden Steuerungen

Damit eine Mehrheitsentscheidung auf Sollwertbasis getroffen werden kann, müssen mehrere redundante numerische Steuerungen miteinander synchronisiert werden. Aus dem allgemeinen Zustandsautomat aus Abbildung 2.9 auf Seite 21 wird der Zustandsautomat für den Entscheider in Abbildung 4.3 mit selbstreinigender Redundanz für das fehlertolerante Gesamtsystem abgeleitet. „Fehlertolerant“ entspricht dabei dem Normalbetriebsmodus. Die beiden Zustände „Ausfall“ und „Schaden“ werden auf den Entscheiderzustand „Not-Halt“ abgebildet.

Fehler innerhalb eines definierten Toleranzbereiches werden vom System im regulären Zustand „Fehlertolerant“ maskiert. Überschreitet eine der Steuerungen mit ihrem Sollwert den Toleranzbereich, so wird sie als fehlerhaft angenommen und aus dem Verbund ausgegliedert. Dabei können so lange Steuerungen ausgegliedert werden bis nur noch zwei fehlerfreie Steuerungen verbleiben. In diesem Fall wechselt das System mit den beiden verbleibenden Steuerungen in den Zustand „Fehlersicher“. Hier entspricht das System der fehlersicheren Steuerung aus Kapitel 3.5. Überschreitet die Differenz der Sollwerte beider verbleibender Steuerungen wiederum den Toleranzbereich, so wird dies sicher erkannt und das Gesamtsystem in den Zustand „Not-Halt“ stillgesetzt.

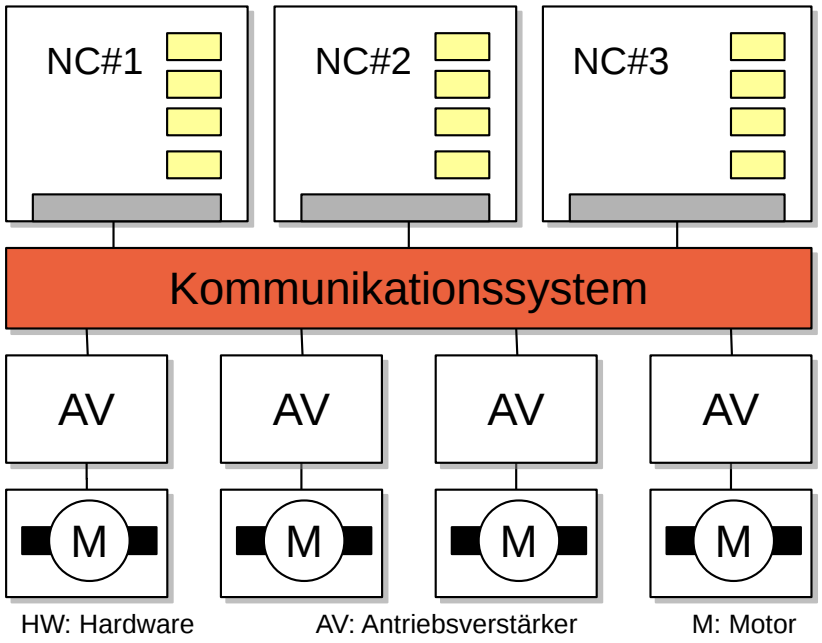


Abbildung 4.4.: Möglicher Aufbau einer fehlertoleranten numerischen Steuerung

4.3. Grundlegender Systemaufbau

In Abbildung 4.4 ist der mögliche Aufbau einer fehlertoleranten numerischen Steuerung skizziert. Die Abbildung enthält drei numerische Steuerungen. Diese müssen nach Tabelle 3.1 auf Seite 33 aus diversitärer Hardware unterschiedlicher Hersteller bestehen, damit systematische Fehler vermieden werden, die in allen Steuerungen gleichzeitig auftreten könnten.

Alle Steuerungen sind darin mit einem Kommunikationssystem verbunden, über welches sie Daten mit den Antriebsverstärkern sowie den anderen Steuerungen austauschen können. Das Anschließen mehrerer Steuerungen an ein Kommunikationssystem schließt analoge Antriebsschnittstellen wie beispielsweise die $\pm$10V Schnittstelle aus. Vorausgesetzt wird die intelligente digitale Antriebsschnittstelle mit den folgenden Anforderungen nach [PSR01].

- Hohe Datenübertragungsrate und kurze Zykluszeiten bei gleichzeitiger Störsicherheit
- Determinismus des Zugriffsverfahrens

- Synchronisierung aller Ringteilnehmer bei geringem Jitter
- Standardisierung des Protokolls
- Werkzeugunterstützung
- Kosten

Zusätzlich zu diesen Anforderungen werden aus [Erd03, VB08] die folgenden Kriterien vorausgesetzt:

Rückwirkungsfreiheit

Fällt ein Gerät aufgrund eines Fehlers aus, so soll dies keine negative Rückwirkung auf das Kommunikationssystem haben. Es wäre fatal, würde das Kommunikationssystem die Kommunikation zwischen den übrigen funktionierenden Geräten unterbrechen. Daraus resultiert, dass es jeder angeschlossenen Steuerung möglich sein soll, den Kommunikationstakt vorzugeben.

Redundante Datenleitungen

Ein fehlerhaftes Kabel soll toleriert werden. Fällt ein einzelnes Kabel aufgrund eines Fehlers aus, so soll immer noch jedes Gerät an der Kommunikation teilnehmen können.

Querkommunikation

Alle Geräte sollen direkt miteinander Daten austauschen können, ohne dass ein dediziertes Gerät für diesen Datenaustausch benötigt wird. Diese Funktionalität wird benötigt, damit sowohl die Steuerungen untereinander Daten als auch die Steuerungen mit dem oder den Mehrheitsentscheider(n) direkt austauschen können.

Synchronisierbarkeit

Es soll die Möglichkeit bestehen, redundante Kommunikationssysteme aufzubauen und diese synchron zu betreiben. Der Aufbau eines redundanten Kommunikationssystems kann notwendig sein, wenn die Eigenschaften eines vorgesehenen Kommunikationssystems hinsichtlich der Punkte Rückwirkungsfreiheit und Redundanten Datenleitungen nicht genügen.

4.4. Aufgabenstellung

Für die Erweiterung des Systems um redundante Steuerungen und einen Entscheider zur Bestimmung des richtigen Sollwerts durch Mehrheitsentscheidung ergeben sich die folgenden Fragestellungen:

1. Wo wird der Entscheider im Gesamtsystem positioniert?
2. Wie werden sowohl der Datenaustausch als auch die Synchronisation gewährleistet?
3. Mit welchem Algorithmus erzeugt der Entscheider einen gültigen Sollwert?

Jede dieser drei Fragestellungen wird in einem der folgenden drei Kapitel 5, 6 und 7 behandelt. Kapitel 8 dokumentiert den Aufbau eines Demonstrationssystems für die fehlertolerante numerische Steuerung. Ebenso zeigt dieses Kapitel anhand von Fehlersimulationstests die Wirksamkeit des redundanten Aufbaus gegenüber Fehlern in einzelnen Steuerungen.

5. Positionierung des Mehrheitsentscheiders

Für den Einsatz einer fehlerkompensierenden strukturellen Maßnahme bedarf es eines Entscheiders. Dieser Entscheider bekommt die von redundanten Systemen berechneten Ergebnisse, vergleicht diese miteinander und bestimmt einen Wert, den er als Ergebnis weitergibt. Dieser Entscheider ist ein kritischer Punkt im System. Hat der Entscheider eine Störung, so gibt er möglicherweise fehlerhafte Sollwerte aus.

Die Abbildung 4.4 zeigt einen generellen Aufbau eines Systems mit redundanten Steuerungen. Es enthält keine Information darüber, wie aus den redundanten Informationen der Steuerungen genau ein Sollwert für jeden einzelnen Antrieb bestimmt wird. Für diese Entscheidung bestehen die drei folgenden grundsätzlichen Möglichkeiten, welche im weiteren erörtert werden. Voraussetzung für alle drei Möglichkeiten ist ein Kommunikationssystem, das mehrere Steuerungen parallel zulässt

- Verteilte antriebsintegrierte Entscheider.
- Ein zentraler Entscheider.
- Verteilte steuerungsintegrierte Entscheider.

5.1. Verteilte antriebsintegrierte Entscheider

Dieser Aufbau besteht aus dezentralen Entscheidern die in den Antriebsverstärkern integriert sind. Die Abbildung 5.1 zeigt diese fehlertolerante Steuerungsarchitektur. Die Entscheider empfangen die Werte aller Steuerungen, bilden dar-

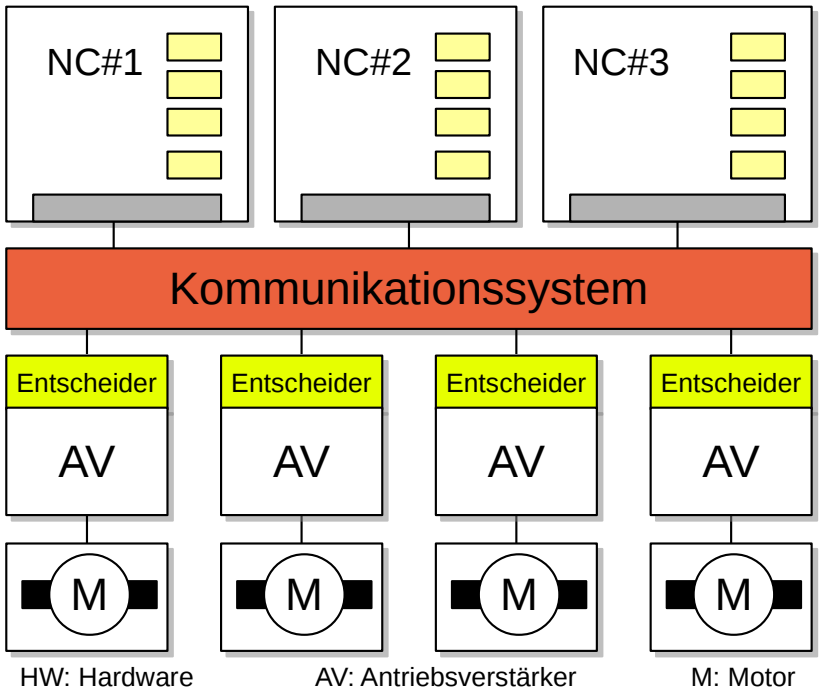


Abbildung 5.1.: Entscheider in den Antrieben

aus einen gültigen Sollwert und geben diesen an den Antriebsverstärker weiter. Der Fall, dass ein Entscheider oder der dahinter liegende Antriebsverstärker oder Motor ausfällt, kann durch redundante Hardware oder redundanter Kinematiken abgefangen werden. Dieser Fall wird jedoch nicht weiter betrachtet, da nach Kapitel 2.4 der Fokus auf der Beherrschung von Fehlern in der Sollwerterzeugung liegt.

Vorteilig an diesem Steuerungsaufbau ist, dass kein Zusatzgerät notwendig ist, da die in den Antrieben ohnehin vorhandene Rechenleistung genutzt wird. Damit kann die notwendige Funktionalität anhand von sicherer Software leicht hinzugefügt werden. Die Korrektheit dieser Zusatzsoftware muss mit den Verfahren aus Kapitel 3.4 sichergestellt werden. Die Hardware des Antriebs wird nicht geändert, so dass deren Ausfallwahrscheinlichkeit unverändert bleibt.

Nachteilig an diesem Steuerungsaufbau ist, dass unterschiedliche Antriebe zur gleichen Zeit möglicherweise die Werte unterschiedlicher Steuerungen nutzen. Dies kann zu Abweichungen von der gewünschten Bahn führen und damit das Werkstück beschädigen. Um diesen Fall zu verhindern ist entweder ein

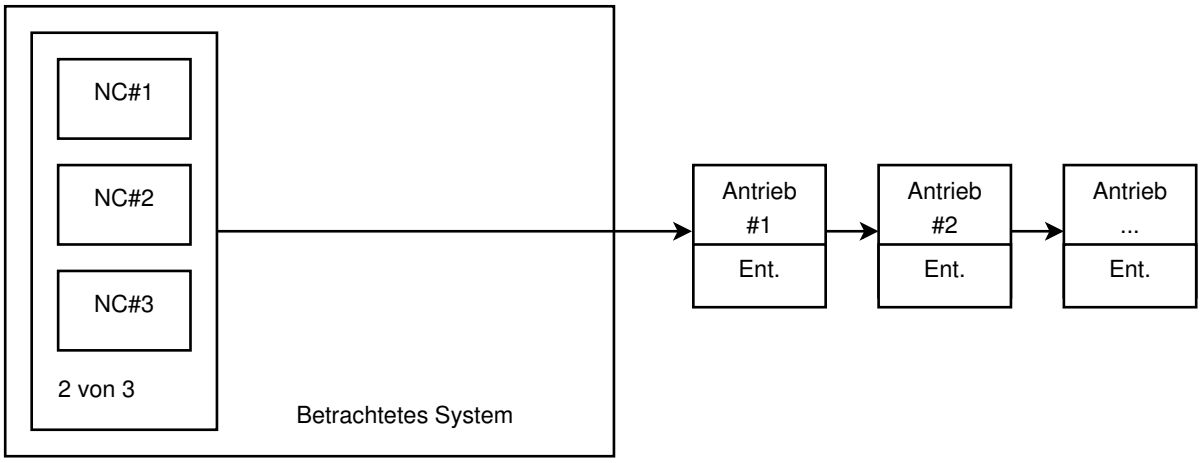


Abbildung 5.2.: Zuverlässigkeitskette des antriebsintegrierten Entscheiders

Algorithmus notwendig, der seine Entscheidung basierend auf der Gesamtheit aller Sollwerte trifft, oder aber ein Datenaustausch zwischen den Antrieben, damit diese sich auf die Werte von genau einer Steuerung festlegen können. Dieser Datenaustausch bedeutet jedoch zusätzliche Komplexität und Programmieraufwand.

Die Zuverlässigkeitskette dieses Aufbaus ist in Abbildung 5.2 dargestellt. Die Antriebe liegen außerhalb des betrachteten Bereichs, da sie nicht Teil des sollwertgebenden Steuerungssystems sind. Da sich die Zuverlässigkeit der Antriebe durch die Erweiterung um den Entscheider in Form von Software nicht ändert, ist das Herausnehmen des Entscheiders aus dem betrachteten System zulässig.

Ausgehend von Abbildung 5.2 zeigt Formel 5.1 die Überlebenswahrscheinlichkeit des betrachteten Systemaufbaus mit Entscheidern in den Antrieben.

$$\begin{aligned} R_{antriebsintegriert}(t) &= R_{2\ von\ 3}(t) \\ &= 3R_{NC}^2(t) - 2R_{NC}^3(t) \end{aligned} \tag{5.1}$$

5.2. Ein zentraler Entscheider

Dieser Steuerungsaufbau enthält genau einen zentralen Entscheider, siehe Abbildung 5.3. Hier sind die Antriebe und Steuerungen unverändert. Der notwendige Vergleich von Sollwerten wird in einer zentralen Entscheiderkomponente

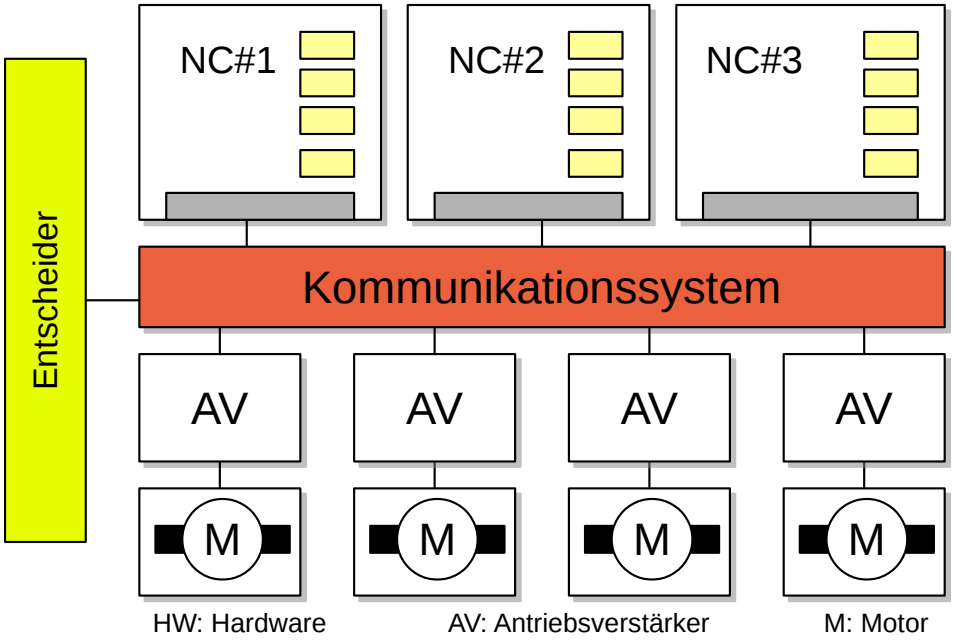


Abbildung 5.3.: Ein zentraler Entscheider

durchgeführt. Das Ergebnis des Vergleiches wird daraufhin an die Antriebe weitergegeben.

Der Vorteil dieses Aufbaus liegt darin, dass weder Antriebsverstärker noch Steuerungen modifiziert werden müssen. Ein sehr großer Nachteil ist jedoch, dass mit dem zentralen Entscheider eine gemeinsame Fehlerquelle (Single Point of Failure) in das Steuerungssystem eingebracht wird. Ein Fehler oder Ausfall des Entscheiders würde alle vorgelagerten Redundanzmaßnahmen zur Erhöhung der Zuverlässigkeit auf einen Schlag wirkungslos werden lassen. Dieser Entscheider muss daher intern unbedingt fehlertolerant ausgelegt sein.

Die Abbildung 5.4 zeigt die Zuverlässigkeitskette des Systems mit eigenständigem, zentralen Entscheider. Darin wird die serielle Zuverlässigkeitskette innerhalb des betrachteten Systems sichtbar. Die Überlebenswahrscheinlichkeit des Entscheiders wird, wie Formel 5.2 zeigt, mit der Überlebenswahrscheinlichkeit des redundanten Steuerungssystems multipliziert. Da die Überlebenswahrscheinlichkeit eines realen Systems immer kleiner als eins ist, verringert der zentrale Entscheider die Gesamtzuverlässigkeit in jedem Fall.

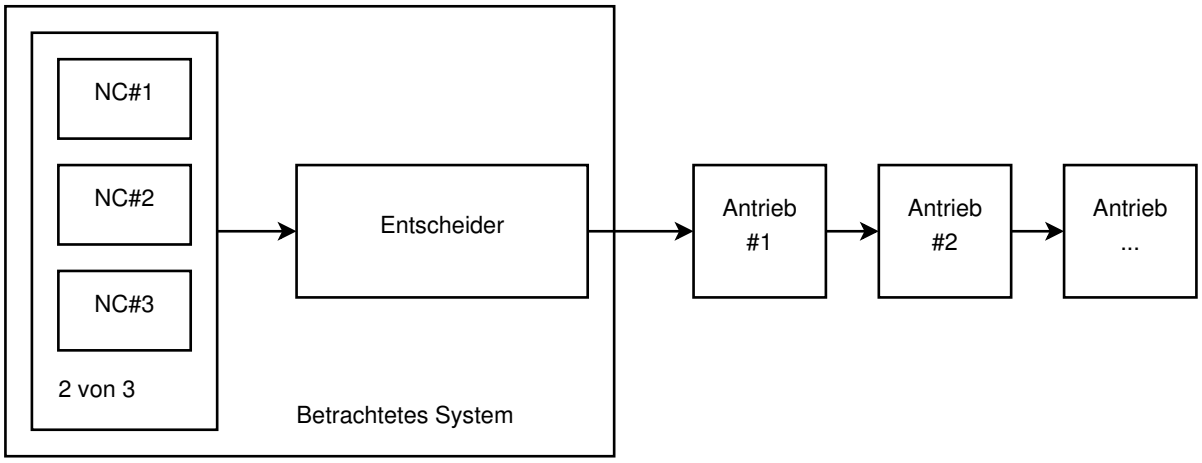


Abbildung 5.4.: Zuverlässigkeitskette des zentralen Entscheiders

$$\begin{aligned} R_{zentral}(t) &= R_{2\ von\ 3}(t) * R_{Entscheider}(t) \\ &= \left(3R_{NC}^2(t) - 2R_{NC}^3(t)\right) * R_{Entscheider}(t) \end{aligned} \tag{5.2}$$

5.3. Verteilte steuerungsintegrierte Entscheider

Den dritten Aufbau mit steuerungsintegrierten Entscheidern zeigt die Abbildung 5.5. Die Vergleiche der redundant berechneten Sollwerte werden hier in den Steuerungen durchgeführt. Die Anbindung jeder einzelnen numerischen Steuerung an das Kommunikationssystem kann durch ein Abschaltsignal von zwei anderen numerischen Steuerungen unterbunden werden. Im normalen Betrieb übergibt nur die erste Steuerung Werte für die Antriebe an das Kommunikationssystem. Die weiteren Steuerungen synchronisieren sich mit dieser ersten Steuerung und führen parallel alle Berechnungen durch, halten ihre berechneten Werte jedoch zurück. Stellt nun ein Entscheider fest, dass eine der Steuerungen fehlerhafte Berechnungen anstellt, so sendet er ein Abschaltsignal. Sind zwei Abschaltsignale von unterschiedlichen Steuerungen für eine Steuerung gegeben, so wird diese vom Kommunikationssystem abgetrennt. War die vom Bus

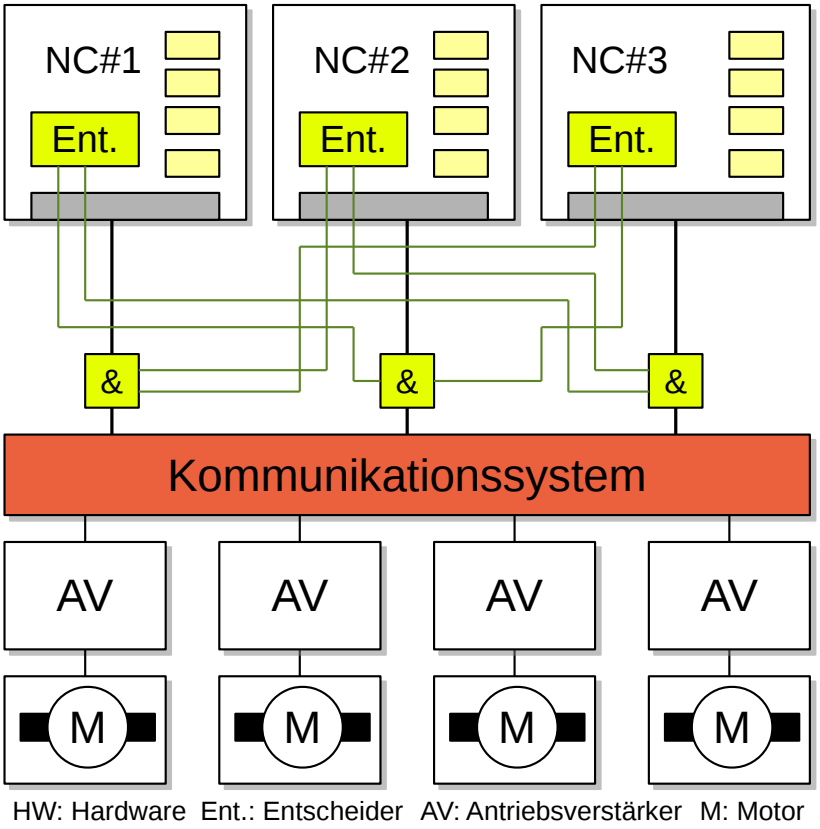


Abbildung 5.5.: Entscheider in numerischen Steuerungen

abgetrennte Steuerung die aktive Steuerung, so übernimmt eine der übrigen Steuerungen das Senden der Sollwerte an die Antriebe.

Dieser Aufbau enthält genau genommen keinen echten Mehrheitsentscheider, der aus einer Menge von berechneten Sollwerten einen auszugebenden Sollwert berechnet. Um dies zu gewährleisten, sind zusätzliche Kommunikationsverbindungen zum Sollwertaustausch zwischen den Steuerungen innerhalb eines Interpolationstaktes notwendig. Die Kommunikation sollte in diesem Fall zweimal pro Interpolationstakt erfolgen, einmal um die Sollwerte zwischen den Steuerungen auszutauschen, das zweite mal um vom Entscheider berechnete Sollwerte an die Antriebsverstärker zu verteilen.

Der Vorteil dieses Aufbaus liegt darin, dass die Antriebsfirmware nicht erweitert werden muss. Zudem werden die Sollwerte bis mindestens zum ersten Fehler immer von derselben Steuerung an die Antriebe gesendet, womit Bahn-

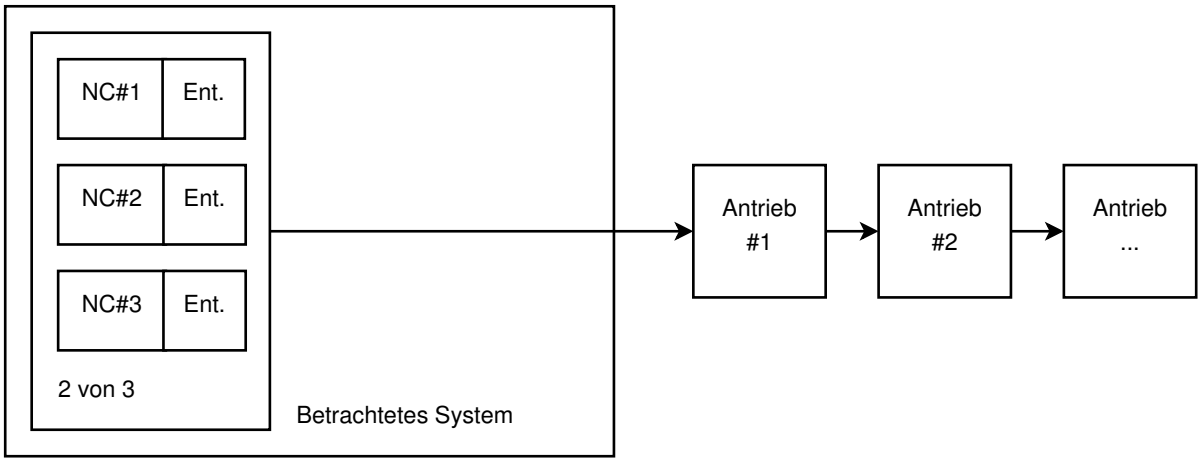


Abbildung 5.6.: Zuverlässigkeitskette für steuerungsintegrierte Entscheider

abweichungen aufgrund der Nutzung von Sollwerten unterschiedlicher Steuerungen durch die verschiedenen Achsen prinzipbedingt ausgeschlossen sind.

Nachteilig ist, dass bei einem Fehler in der aktiven Steuerung mit folgender Ausmaskierung dieser, Sprünge in den Sollwerten bei der Umschaltung auf eine der verbleibenden Steuerungen auftreten werden. Dies ist, abhängig vom Anwendungsfall, für den Prozess meist schädlich. In diesem Fall wird genau genommen keine Entscheidung über die Auswahl des auszugebenden Sollwerts getroffen, sondern nur die Entscheidung ob eine der redundanten Steuerungen weiterarbeiten darf oder ausmaskiert wird.

In Abbildung 5.6 ist die Zuverlässigkeitskette für den Systemaufbau mit steuerungsintegriertem Entscheider dargestellt. Die Formel 5.3 zeigt die zugehörige Berechnung der Überlebenswahrscheinlichkeit dieses Systems.

$$\begin{aligned} R_{steuerungsintegriert}(t) &= R^*_{2\ von\ 3}(t) \\ &= 3\left[R_{NC}(t) * R_{Entscheider}(t)\right]^2 \\ &\quad -2\left[R_{NC}(t) * R_{Entscheider}(t)\right]^3 \end{aligned} \tag{5.3}$$

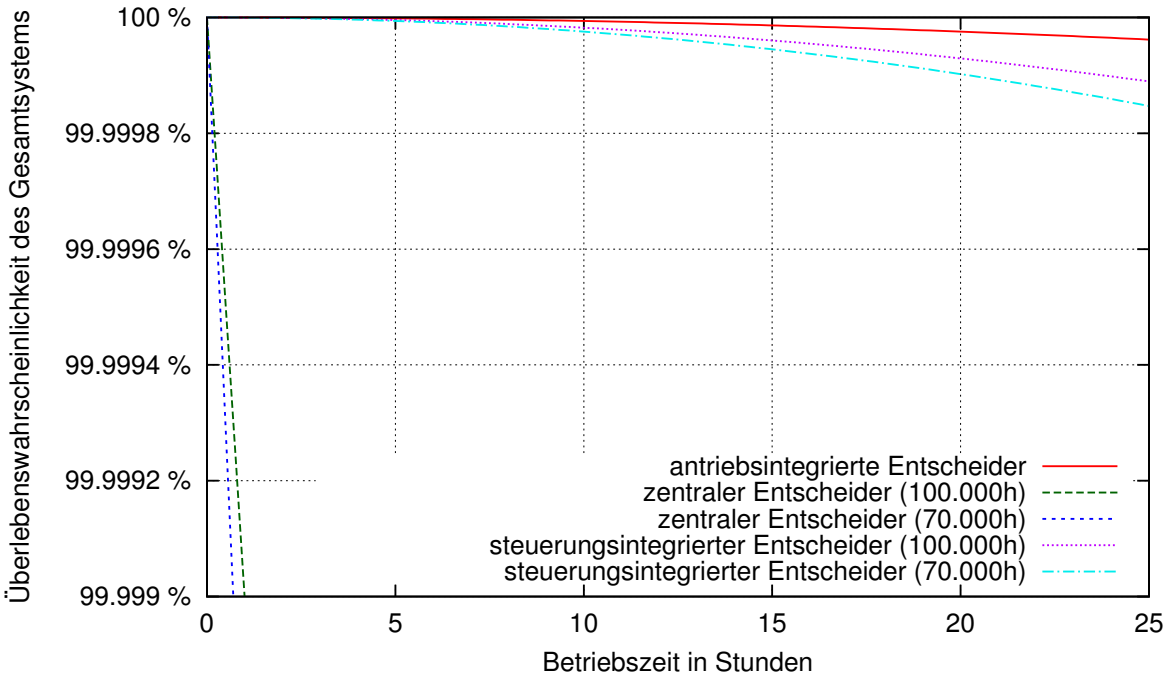


Abbildung 5.7.: Überlebenswahrscheinlichkeiten der verschiedenen Entscheiderpositionen

5.4. Vergleich der Positionierungen

Zum Vergleich der unterschiedlichen Positionierungsmöglichkeiten des Entscheiders sind in Abbildung 5.7 die Überlebenswahrscheinlichkeiten für alle drei vorangegangenen Möglichkeiten dargestellt. Zur Berechnung wurde für die Steuerungen ein MTBF von 70.000 Stunden angenommen. Für die Zuverlässigkeit der Positionierungsmöglichkeiten mit Entscheider sind für diesen jeweils zwei unterschiedliche Wert, 100.000 Stunden sowie 70.000 Stunden, als MTBF angenommen.

5.5. Zusammenfassung

In diesem Kapitel wurden drei verschiedene Ansätze zur Platzierung des Entscheiders in einem numerischen Steuerungssystem erörtert. Die Ansätze, bei denen kein Zusatzgerät notwendig ist, zeigen sich hierbei als zuverlässiger gegenüber dem Ansatz mit Zusatzgerät zur Entscheidung. Von den verteilten An-

sätzen ist die Integration der Entscheidung in die Antriebe der Integration in die Steuerungen überlegen. Der Ansatz mit den Entscheidern in den Antrieben wird daher auch für den Aufbau des in Kapitel 8 beschriebenen realisierten Systems genutzt. Zunächst wird in dem folgenden Kapitel jedoch auf die Kommunikation und Synchronisation der redundanten Steuerungen eingegangen.

6. Synchronisation redundanter numerischer Steuerungen

Damit ein Entscheider aus vorigem Kapitel eine sinnvolle Wahl treffen kann, müssen ihm zum Entscheidungszeitpunkt zusammen gehörende Sollwerte vorliegen. Dies bedeutet, dass die berechneten Sollwerte den selben Programmzustandspunkt auf allen Steuerungen wiedergeben müssen. Die einzelnen Steuerungen müssen daher zeitgleich starten und daraufhin Takt für Takt synchron weiterarbeiten.

Würden unterschiedlich schnelle Steuerungen ihre Berechnungen mit unterschiedlicher Geschwindigkeit durchführen, so bekäme der Entscheider nicht zueinander passende und höchstwahrscheinlich voneinander abweichende Sollwerte vorgelegt. Der Entscheider würde daraufhin einen Fehler erkennen, vermeintlich fehlerhafte Steuerungen ausmaskieren und zuletzt das System im sicheren Halt zum Stillstand bringen. Die exakte Synchronizität aller Steuerungen ist daher essentiell für die Zuverlässigkeit der fehlertoleranten numerischen Steuerung.

Zur Synchronisation der redundanten Steuerungen muss zunächst gewährleistet sein, dass alle Steuerungen untereinander Daten austauschen können. Diese Anforderung besteht bereits bei der fehlersicheren numerischen Steuerung aus Kapitel 3.5, bei der beide Steuerungen mit einem DPR (Dual-Ported-RAM) Baustein verbunden sind. Ein DPR-Baustein zeichnet sich dadurch aus, dass er zwei Zugriffsschnittstellen besitzt und damit zwei Prozessoren gleichzeitig auf den selben Speicher zugreifen können.

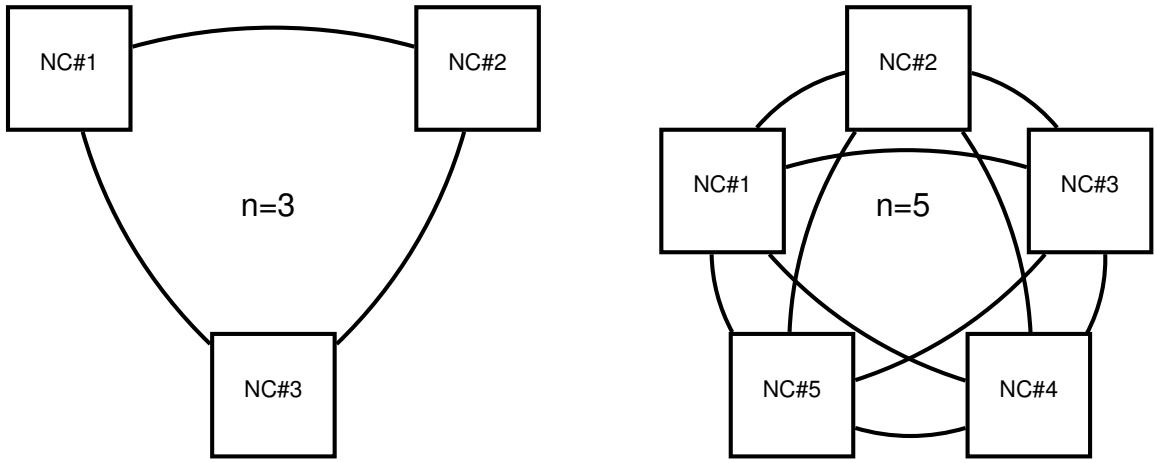


Abbildung 6.1.: Drei und fünf redundante numerische Steuerungen verbunden über DPR

Benötigen nur zwei Steuerungen zeitgleichen Zugriff auf den selben Speicher, so ist ein DPR-Baustein eine sehr geeignete Wahl. Für die fehlertolerante numerische Steuerung müssen jedoch mehr als zwei Steuerungen miteinander verbunden werden. Daher ist es notwendig alternative Möglichkeiten zum Austausch der Synchronisationsdaten für mehr als zwei Systeme zu finden. Diese Möglichkeiten werden in den folgenden Abschnitten dargelegt.

6.1. Nutzung von Dual-Ported-RAM

Ein naheliegender Ansatz ist, jede einzelne der redundanten Steuerungen mit den übrigen über einen DPR-Baustein zu verbinden. Bei diesem Ansatz werden für drei redundante Steuerungen bereits drei DPR-Bausteine benötigt, wie Abbildung 6.1 veranschaulicht. Mit jeder zusätzlichen Steuerung vergrößert sich die Gesamtzahl der DPR-Bausteine um die Anzahl der bereits vorhandenen Steuerungen. Für fünf redundante numerische Steuerungen sind bereits zehn DPR-Bausteine für die vollständige Vernetzung notwendig.

Allgemein berechnet sich die Anzahl m der benötigten DPR-Bausteine für n Steuerungen nach der Formel 6.1. Die Anzahl der benötigten DPR-Bausteine wächst mit der Anzahl redundanter Steuerungen folglich näherungsweise quadratisch, $\mathcal{O}(n^2)$. Dieser Ansatz skaliert somit schlecht. Der hardwaretechnische

Aufwand ist für eine steigende Anzahl redundanter numerischer Steuerungen kaum vertretbar.

$$\begin{aligned} m &= \sum_{k=2}^{n} k-1 \\ &= \frac{n^2-n}{2} \end{aligned} \tag{6.1}$$

Durch den Ausfall eines Speichers ist hier nur die Verbindung zwischen zwei Steuerungen betroffen. Damit eine von n Steuerungen ihre Synchronisationsverbindung zu allen übrigen verliert, müssen mindestens $n-1$ Speicher ausfallen. Der Ausfall einzelner Verbindungen ist damit tolerierbar und beeinträchtigt somit das Gesamtsystem nicht.

6.2. Geteilter Speicher mit Arbitrierungslogik

Eine weiterer Ansatz besteht darin, über einen Datenbus mit Arbitrierungslogik einen gewöhnlichen Speicherbaustein zu verwenden, siehe Abbildung 6.2. So bekommen mehrere Steuerungen zeitversetzt vorübergehend exklusiven Zugriff auf den Speicher. Genutzt wird dieses Verfahren beispielsweise im PCI-Bus von PC-Systemen. Dort können alle busmasterfähigen Geräte den PCI-Bus arbitrieren und auf den Haupt- und Peripheriespeicher des Hostsystems zugreifen.

Hardwaretechnisch skaliert dieser Ansatz sehr gut. Es wird lediglich ein einzelner Speicherbaustein benötigt. Die Steuerungen sind mit diesem über ein Bussystem und einer Arbitrierungslogik verbunden. Die Komplexität des Hardwareaufbaus beträgt somit $\mathcal{O}(n+1)$ und wächst damit annähernd linear.

Sehr problematisch an diesem Aufbau ist jedoch, dass mit dem Speicherbaustein eine gemeinsame Fehlerquelle (Single Point of Failure) in das System eingebracht wird. Fällt dieser Speicherbaustein aus, so fehlt auf einen Schlag allen redundanten Steuerungen die Synchronisation sowie der Datenaustausch. Bei solch einem Ausfall ist der übrige redundante Ansatz nutzlos.

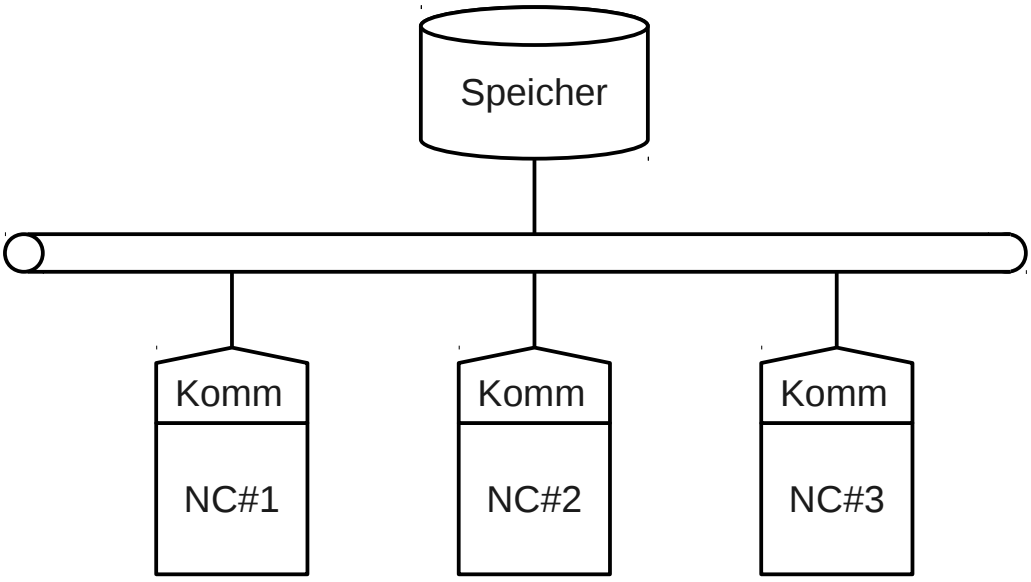


Abbildung 6.2.: Drei redundante numerische Steuerungen an einem gemeinsamem Speicher

6.3. Verteilter Speicher

Bei der dritten Möglichkeit enthält jede Steuerung einen eigenen gemeinsamen Speicherbereich, siehe Abbildung 6.3. Dieser ist über ein Kommunikationssystem freigegeben, so dass jeder Kommunikationssystemteilnehmer lesend wie auch schreibend darauf zugreifen kann. Für diesen Aufbau ist ein Kommunikationssystem Voraussetzung, das die nötigen Datenzugriffe ermöglicht.

Die Anzahl der notwendigen Speicher ist identisch zur Anzahl der redundant vorhandenen numerischen Steuerungen. Zusätzliche Speicherbausteine werden nicht benötigt. Die Komplexität des Hardwareaufbaus steigt somit linear: $\mathcal{O}(n)$.

Fällt einer der geteilten Speicher aus, so ist nur die zugehörige Steuerung davon betroffen. Die verbleibenden Steuerungen können weiterarbeiten. Wichtig ist, dass die Daten zuverlässig über das Kommunikationssystem ausgetauscht werden. Dieses kann, um Ausfällen vorzubeugen, wiederum redundant ausgelegt werden.

Die Synchronisation erfolgt hier durch das Kommunikationssystem. Dieses tauscht zyklisch die konfigurierten Speicherbereiche aus und erzeugt daraufhin

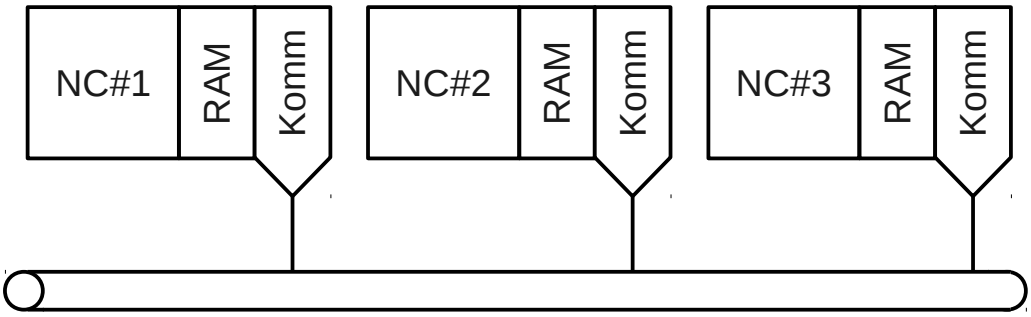


Abbildung 6.3.: Verteilter redundanter Speicher verbunden mit Kommunikationssystem

eine Rückmeldung an die Steuerung. Über diese Rückmeldung kann die Weitertaktung ohne zusätzlichen programmiertechnischen Aufwand erfolgen.

6.4. Bewertung der Synchronisationsmodelle

Zur Bewertung wird zusätzlich zu den Kriterien aus Kapitel 4.3 noch die Skalierbarkeit mit betrachtet werden. Die Skalierbarkeit soll hierbei den notwendigen Aufwand zum Hinzufügen von Redundanz zu einem System bezeichnen. Je geringer der Aufwand, desto besser sei die Skalierbarkeit des Systems. In diesem Fall bezieht sich die Skalierbarkeit auf den notwendigen Aufwand um das fehlertolerante numerische Steuerungssystem durch weitere redundante Steuerungen noch zuverlässiger zu machen. Hierbei spielen sowohl Hardwareaufwand wie auch Konfigurations- und Programmieraufwand eine Rolle. Insgesamt ergeben sich damit die folgenden Kriterien:

- Hohe Datenübertragungsrate und kurze Zykluszeiten bei gleichzeitiger Störsicherheit
- Determinismus des Zugriffsverfahrens
- Synchronisierung aller Ringteilnehmer bei geringem Jitter
- Standardisierung des Protokolls
- Werkzeugunterstützung
- Kosten
- Rückwirkungsfreiheit

- Redundante Datenleitungen
- Querkommunikation
- Synchronisierbarkeit mit redundantem Kommunikationssystem
- Skalierbarkeit

Das erste Kriterium „hohe Datenübertragungsrate und kurze Zykluszeiten bei gleichzeitiger Störsicherheit“ ist hauptsächlich abhängig von der Übertragungsphysik sowie der Datenmodulation. Die Topologien zur Synchronisation der Steuerungen sind grundsätzlich unabhängig von der Übertragungsphysik und können alle mit derselben betrieben werden. Dies bedeutet, dass sich die unterschiedlichen Topologien in diesem Kriterium nicht voneinander unterscheiden.

Bezüglich des „Determinismus des Zugriffsverfahrens“ gibt es Unterschiede zwischen den Topologien. Die erste Variante mit den DPR-Bausteinen ist absolut deterministisch, da jede NC zu jedem Zeitpunkt sofort auf alle Speicherstellen zugreifen kann. Der Determinismus der beiden übrigen Topologien ist abhängig von der Zugriffsarbitrierung auf das Speicher- beziehungsweise Kommunikationssystem, da in beiden Fällen Ressourcen geteilt werden.

Die „Synchronisierung aller Ringteilnehmer bei geringem Jitter“ ist bei allen Topologien machbar. Effizient umsetzbar ist sie nur bei der Topologie mit den verteilten Speichern, da ein Kommunikationssystem genau diese Funktion bietet. So kann das Kommunikationssystem gleichzeitig allen angeschlossenen Steuerungen ein Synchronisationssignal senden, während es bei den übrigen Topologien programmtechnisch per Polling oder Busy-Waiting in die Applikation implementiert werden muss.

Die „Standardisierung des Protokolls“ bezieht sich auf die Darstellung von Daten in höheren Übertragungsschichten. Sie spielt ebenso wie die Kriterien „Werkzeugunterstützung“ und „Kosten“ im Betrachtungshorizont dieser Arbeit keine Rolle.

Die „Rückwirkungsfreiheit“ aus Kapitel 4.3 bedeutet, dass ein Fehler oder Ausfall eines Geräts, keine negative Rückwirkung auf das Kommunikationssystem haben soll. Die Topologie mit den DPR-Bausteinen entspricht diesem Kriterium. Ein Fehler in einer Steuerung oder einem DPR beeinflusst nur die

unmittelbar daran angeschlossenen Komponenten. Für die Topologie mit dem geteilten Speicher ist die Busverbindung natürlich rückwirkungsfrei anzulegen. Dennoch fällt diese Topologie aus, wenn das zentrale Element, der geteilte Speicher ausfällt, weswegen sie nicht als vollständig rückwirkungsfrei bezeichnet werden darf. Die Topologie mit verteiltem Speicher ist hingegen, ein entsprechendes Kommunikationssystem vorausgesetzt, rückwirkungsfrei.

Das Kriterium „Redundante Datenleitungen“ aus Kapitel 4.3 bedeutet, dass eine einzelne, fehlerhafte Verbindungsleitung toleriert werden soll. Für die Topologie mit den DPR-Bausteinen ist dies der Fall, da alle Steuerungen bei Ausfall einer einzelnen Leitung immer noch hinreichend miteinander vernetzt sind. Die Struktur mit dem zentralen Speicher hingegen erfüllt dieses Kriterium überhaupt nicht, da der Ausfall der Leitung zum Speicher nicht toleriert werden kann. Beim verteilten Speicher ist die Bewertung davon abhängig, mit welcher Struktur (Bus, Stern, Ring, ...) das Kommunikationssystem aufgebaut ist.

„Querkommunikation“ aus Kapitel 4.3 bedeutet, dass alle Geräte direkt miteinander Daten austauschen können. Sie ist insbesondere wichtig, da nicht alle gängigen industriellen Kommunikationssysteme diesen Datenaustausch zulassen. Die Topologie mit DPR-Bausteinen stellt diese Funktionalität offensichtlich zur Verfügung. Ebenso die Topologie mit dem geteilten Speicher, da hier jedes angeschlossene Gerät wahlfrei zugreifen kann. Bei der Topologie mit dem verteilten Speicher ist diese Funktionalität hingegen nicht automatisch gegeben. So sind viele industriellen Kommunikationssysteme auf eine reine steuerungszentrierte Master-Slave Kommunikation ausgerichtet, bei denen die Steuerungsapplikation im Busmaster Querkommunikationsdaten empfangen und weiterverteilen muss. Hierbei ist also explizit darauf zu achten, ein Kommunikationssystem mit der Funktionalität von Querverbindungen zu wählen.

Für die „Synchronisierbarkeit mit redundantem Kommunikationssystem“ aus Kapitel 4.3 soll die Möglichkeit bestehen, parallele Kommunikationssysteme aufzubauen und diese redundant und synchron zu betreiben. Die Topologien mit DPR-Bausteinen und geteiltem Speicher bieten diese Art der Synchronisation nicht an. Diese müsste in der Applikation nachprogrammiert werden. Für

die Topologie mit verteiltem Speicher gibt es Kommunikationssysteme, die eine Synchronisation auf ein weiteres System unterstützen. Diese Unterstützung muss bei der Auswahl eines Kommunikationssystems berücksichtigt werden.

Die „Skalierbarkeit“ der jeweiligen Topologien ist in den vorausgegangenen Kapiteln bereits aufgezeigt worden. Der Aufwand der Topologie mit DPR-Bausteinen wächst quadratisch und steht damit softwaretechnisch wie hardwaretechnisch in keinem Verhältnis zum Nutzen. Die verbleibenden beiden Topologien haben einen Aufwand der annähernd linear für den geteilten Speicher und linear für den verteilten Speicher anwächst.

6.5. Zusammenfassung

In diesem Kapitel wurden drei unterschiedliche Topologien zur Synchronisation und zum Datenaustausch zwischen den redundanten Steuerungen gegenübergestellt. Die Topologie mit DPR-Bausteinen ist bei fast allen Kriterien sehr gut zu bewerten. Jedoch hat sie beim Kriterium Skalierbarkeit eindeutige Schwierigkeiten, da der Aufwand im Verhältnis zur hinzugefügten Redundanz quadratisch anwächst. Die übrigen beiden Topologien liegen hinsichtlich der Skalierbarkeit deutlich günstiger. Hinsichtlich des Ausfallpotentials des bei der Topologie mit geteiltem Speicher nur einmalig vorhandenen Speichers ist die Topologie mit verteiltem Speicher für die fehlertolerante numerische Steuerung die günstigere Wahl.

Bis hier wurde nun die Positionierung des Entscheiders im Gesamtsystem sowie die Synchronisation und der Datenaustausch zwischen den Steuerungen dargestellt. Die Frage, nach welchem Algorithmus ein Mehrheitsentscheider aus einer Menge von redundant berechneten Sollwerten einen auszugebenden Sollwert bestimmt, wird im Folgekapitel behandelt.

7. Algorithmen für die Mehrheitsentscheidung

In den vorangegangenen Kapiteln wurde die Notwendigkeit für Fehlertoleranzmaßnahmen dargelegt. Weiterhin wurden mögliche Topologien und Synchronisations- beziehungsweise Kommunikationsmechanismen erörtert. Damit können redundante Sollwerte synchron erzeugt und ausgetauscht werden. Zum Verfahren einer Achse ist nun noch ein Algorithmus notwendig, der aus der Menge der redundant berechneten Sollwerte einen gültigen Sollwert für die Aktorik bestimmt.

Der Zustandsautomat aus Abbildung 4.3 auf Seite 47 gibt die Zustände für die fehlertolerante numerische Steuerung vor. Dieser Zustandsautomat ist als Flussdiagramm in Abbildung 7.1 dargestellt. Die Variable n ist dabei die Anzahl der fehlerfreien redundanten Steuerungen. Der Entscheider bekommt im Ablauf die Sollwerte aller Steuerungen zunächst synchron übergeben. Aus diesen Sollwerten bestimmt er mit einem zu definierenden Verfahren einen neuen Sollwert um die Abweichung der Sollwerte jeder einzelnen Steuerung gegen diesen zu berechnen.

Liegt die größte Abweichung außerhalb des nach Kapitel 3.2.4 notwendigen Toleranzbereichs, so wird die zugehörige Steuerung ausgegliedert. Damit nach einer Ausgliederung weitere fehlerhafte Steuerungen erkannt werden, wird der Ablauf nochmals durchlaufen. Dies wird fortgesetzt bis alle Sollwerte innerhalb des Toleranzbereichs um das ermittelte Ergebnis liegen.

Mit diesem Ablauf werden auftretende Fehler die noch innerhalb des Toleranzbereichs liegen zunächst maskiert. Überschreitet ein fehlerhaftes Ergebnis

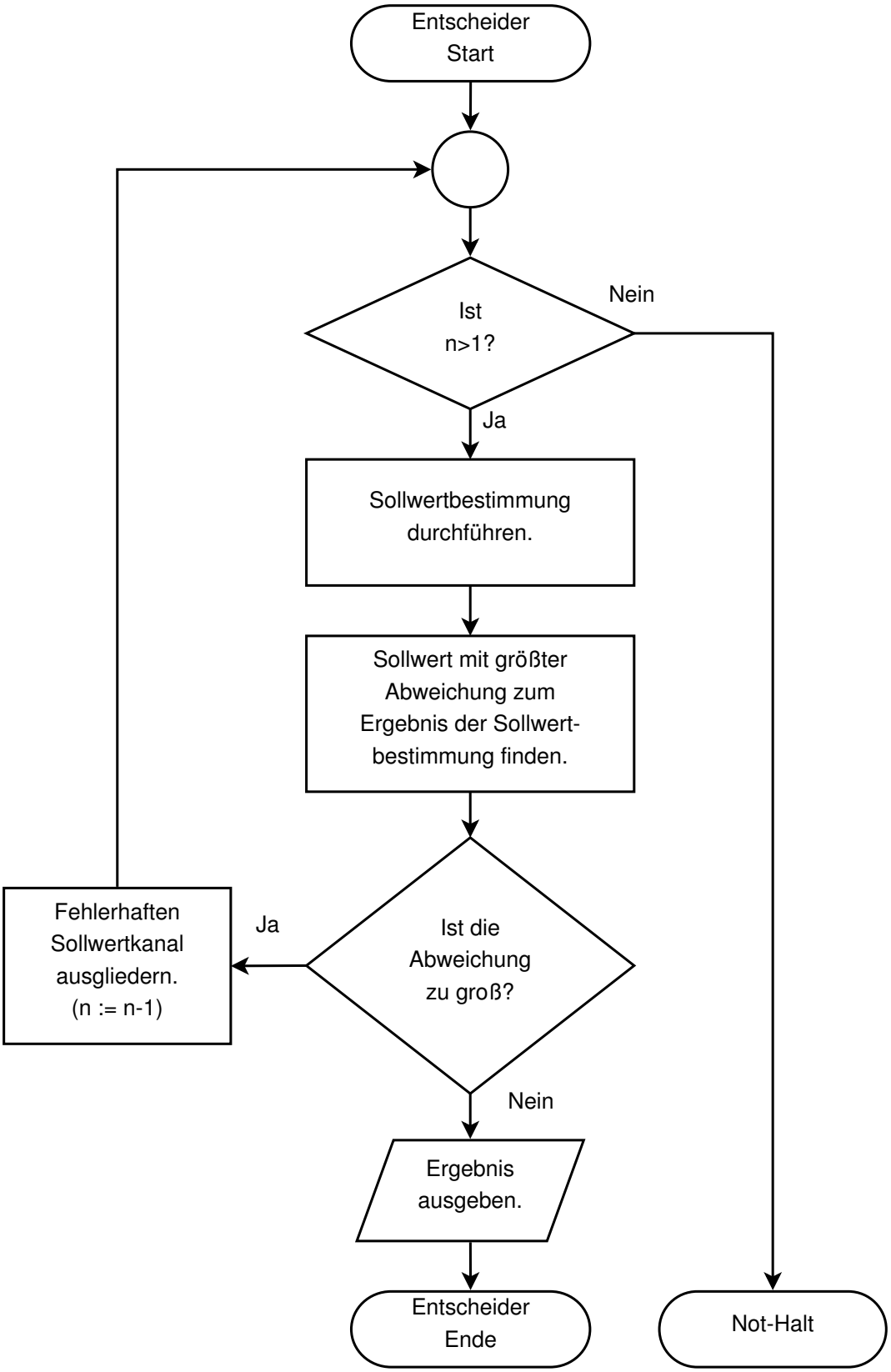


Abbildung 7.1.: Flussdiagramm des Entscheiders mit selbstreinigender Redundanz

den Toleranzbereich, so wird das Gesamtsystem rekonfiguriert und die fehlerhafte Steuerung ausgegliedert. Dieser Vorgang wird durchgeführt, bis nur noch zwei Steuerungen übrig sind. Die verbleibenden beiden Steuerungen werden als fehlersichere numerischen Steuerung nach Kapitel 3.3 weiterbetrieben.

7.1. Methoden zur Sollwertbestimmung

Das Ziel ist eine geeignete Methode zu finden, die aus einer Anzahl von Sollwerten einen einzigen Sollwert berechnet. Dazu werden hier die folgenden verschiedene Methoden näher betrachtet.

- Die Methode der fehlersicheren numerischen Steuerung aus Kapitel 3.3 die im Folgenden als First Seen bezeichnet wird. Hier wird immer der Wert der selben Steuerung ausgegeben, bis diese ausgegliedert wird und eine der verbleibenden Steuerungen für sie übernimmt.
- Das Bilden des arithmetischen Mittels aus den vorliegenden Sollwerten, was ein zunächst naheliegender Weg ist um einen Wert aus einer Menge von Werten zu bestimmen.
- Die Median Methode als Maßnahme zur Fehlerkompensierung aus Kapitel 3.2.3.
- Die Intervallmethode als Maßnahme zur Fehlerkompensierung aus Kapitel 3.2.3.

Jede einzelne Methode wird an den in Abbildung 7.2 vorgegebenen Sollwertkurven veranschaulicht. Ab Kommunikationszyklus 18 tritt darin der Ausfall der fehlerhaften Sollwertkurve auf. Ab Kommunikationszyklus 27 ist die maximal tolerierte Abweichung von 50 Einheiten überschritten. Folglich muss die Störung bis einschließlich Kommunikationszyklus 26 vom Entscheider noch maskiert werden, danach soll die fehlerhafte Sollwertkurve gemäß dem Flussdiagramm in Abbildung 7.1 ausgegliedert werden. Zu Beachten ist für jede Methode, wie die vom Entscheider ausgegebenen Werte sich im Vergleich zu den Eingangswerten und der Sollkurve verhalten.

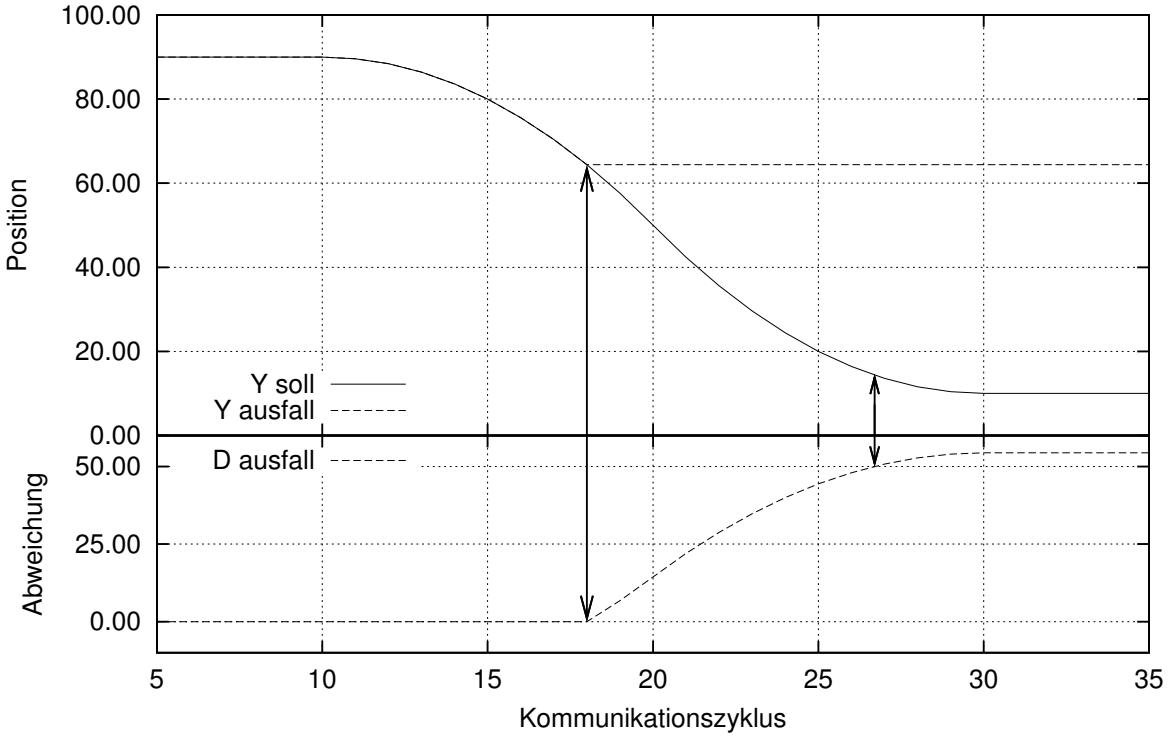


Abbildung 7.2.: Referenz-Fehlerkurve für unterschiedliche Entscheider Algorithmen

7.2. First-Seen

Die First-Seen Methode basiert auf dem selben Prinzip wie bereits die fehlersichere numerische Steuerung. Hierbei werden die Sollwerte der ersten Steuerung aus einer Menge von n redundanten Steuerungen als Ausgabewerte benutzt. Bei einer zu hohen Abweichung unter den redundanten Sollwerten wird die Steuerung mit der höchsten Abweichung aus der Menge der redundanten Steuerungen entfernt. Die Abweichung kann für jeden Wert zu einem Mittelwert aller anderen Werte gebildet werden. Ist nur noch ein Steuerung in der Menge übrig so wird das System in einen sicheren Zustand versetzt. Ansonsten werden die Abweichungen der redundanten Steuerungen neu bestimmt und wiederum überprüft.

Diese Methode funktioniert ab zwei redundanten Steuerungen. Dabei braucht zwischen dem fehlertoleranten Modus mit drei oder mehr redundanten Steue-

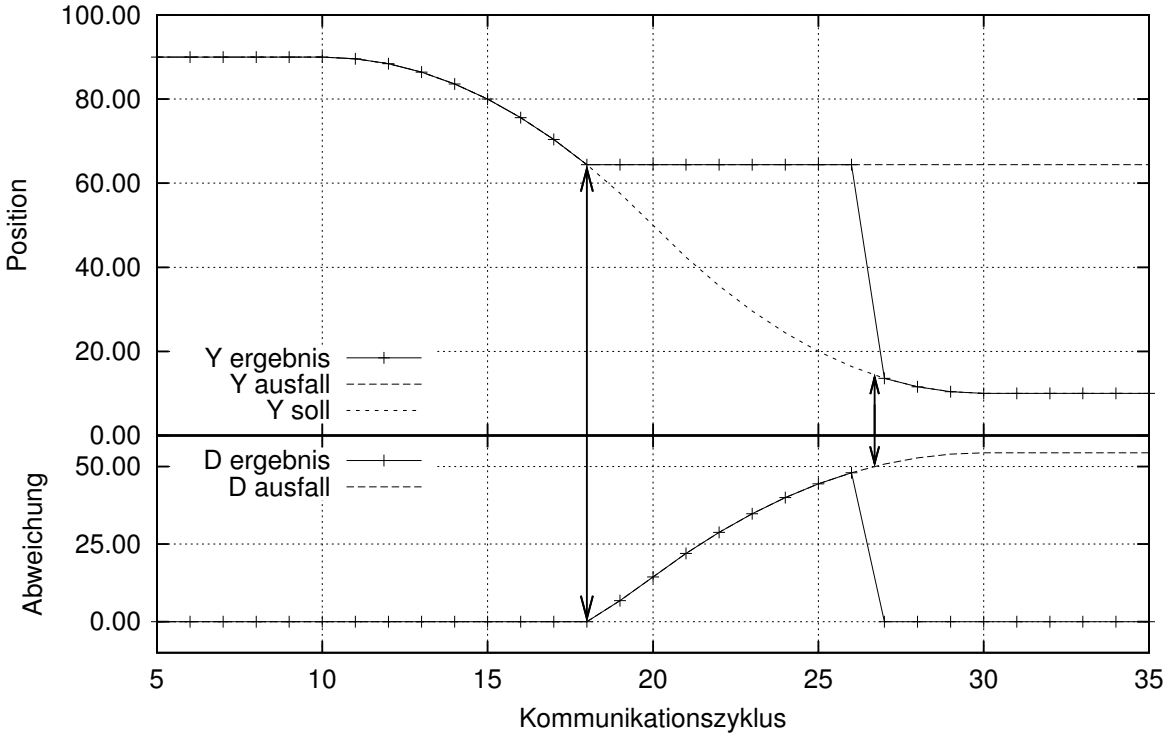


Abbildung 7.3.: Ausfall der ersten Steuerung mit der First Seen Methode

rungen und dem fehlersicheren Modus mit nur zwei Steuerungen intern nicht unterschieden werden.

Vorteilhaft ist diese Methode für einen zentralen Entscheider mit verkoppelten Achsen, da hierbei immer alle Sollwerte der selben Steuerung ausgegeben werden. Es treten keine Vermischungen unterschiedlicher Kanäle oder Kanalwechsel für verschiedene Achsen auf.

Nachteilig ist an dieser Methode, dass bei Entfernen der ersten Steuerung aus der Menge auf jeden Fall ein Sprung der Sollwerte in Höhe der erlaubten maximalen Abweichung auftritt. Abbildung 7.3 stellt diesen ungünstigen Fall dar.

7.3. Arithmetischer Mittelwert

Bei der Mittelwertmethode wird aus einer Menge von n Sollwerten das arithmetische Mittel gebildet und als Sollwert ausgegeben. Bei zu großer Abweichung

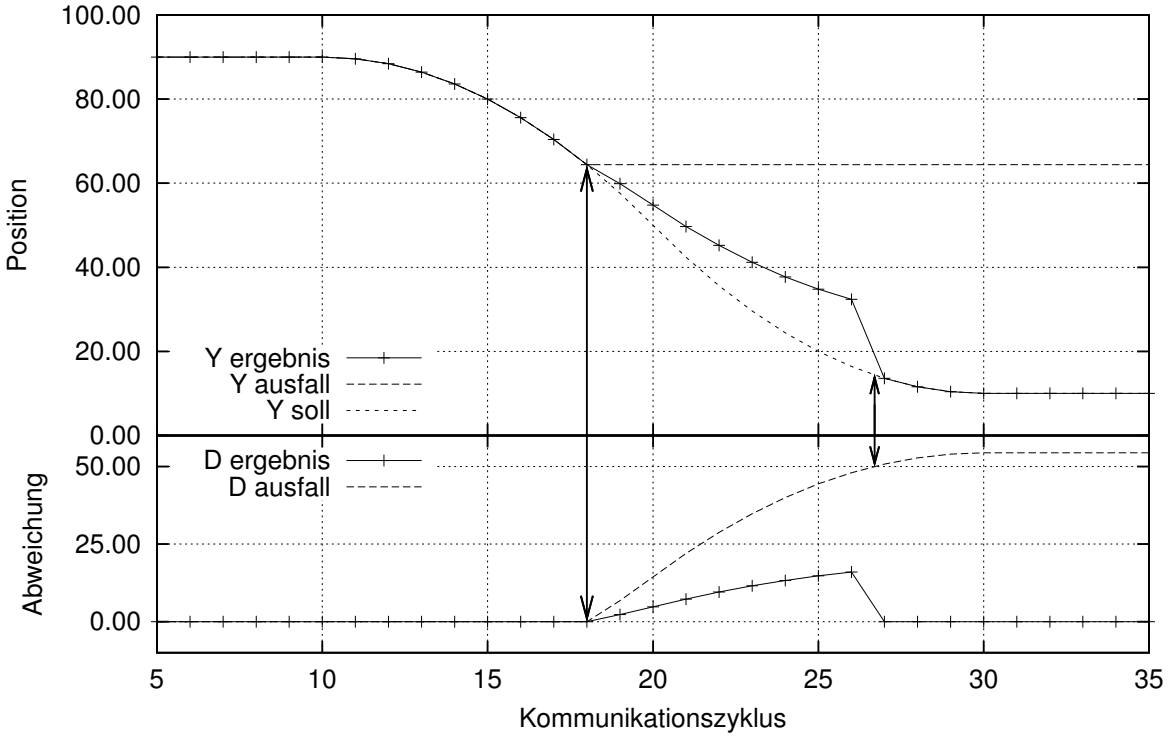


Abbildung 7.4.: Ergebnis des Entscheiders mit arithmetischer Mittelwertfunktionen

wird, wie bei der „First-Seen“ Methode, die Steuerung mit der größten Abweichung aus der Menge entfernt und die Abweichung daraufhin erneut berechnet.

An dieser Methode ist nachteilhaft, dass fehlerhafte Sollwerte einer Steuerung bereits Auswirkungen auf den Ausgang des Entscheiders haben. Zudem kommt es nach dem Entfernen der Steuerung mit der höchsten Abweichung unweigerlich zu einem Sprung in den ausgegebenen Sollwerten. Diese beiden Eigenschaften der Methode wirken sich mit $1/n$ umgekehrt proportional stark zur Anzahl der verfügbaren redundanten Steuerungen aus. Dies ist ungünstig für koordinierte Achsbewegungen entlang einer Bahn mit verkoppelten Achsen. Abbildung 7.4 zeigt beispielhaft die ausgegebenen Sollwerte sowie die Abweichung der ausgegebenen Sollwerte vor und nach dem Entfernen der fehlerhaften Steuerung.

7.4. Median

Bei der Median Methode werden die Sollwerte aller *n* redundanten Steuerungen zunächst aufsteigend sortiert. Als Ausgangssollwert wird daraufhin der Wert verwendet, der in der Mitte der sortierten Menge steht. Voraussetzung dafür ist, dass mindestens drei redundante Steuerungen vorhanden sind.

Günstig an der Median Methode ist das Ausbleiben von Sprüngen in den ausgegebenen Sollwerten wenn eine fehlerhafte Steuerung entfernt wird, siehe Abbildung 7.5. Falls die Sollwerte jedoch leichte Abweichungen voneinander haben, so wechselt der Median unter Umständen häufig zwischen den Steuerungen. Dies kann ein Nachteil sein, wenn der Toleranzbereich zu groß eingestellt ist. In diesem Fall können die Abweichungen der Steuerungen in störenden Bereichen liegen und die dabei entstehenden Knicke auf der Bahn können am Werkstück sichtbar werden. Aus diesem Grund müssen bei Anwendung des Medians die messbaren Abweichungen in der Praxis bestimmt und ausgewertet werden. Sind sie für den speziellen Anwendungsfall hinreichend gering, so ist der Nachteil durch das Wechseln der Steuerungen vernachlässigbar.

7.5. Intervallmethode

Die Intervallmethode aus Kapitel 3.2.3 verhält sich im hier betrachteten eindimensionalen Ergebnisraum wie die Methode „First Seen“. Bis zu Kommunikationszyklus 26 befinden sich alle drei Sollwerte innerhalb des zulässigen Intervalls. Da die Intervallmethode aus der Menge der Ergebnisse innerhalb des Intervalls ein beliebiges auswählen kann, wird der Verlauf im ungünstigsten aber dennoch zulässigen Fall wie in Abbildung 7.6 aussehen. Durch Anpassung der Methode in der Auswahl eines Ergebnisses aus der gültigen Ergebnismenge könnten ständige Sprünge vermieden und damit ein Verlauf wie in Abbildung 7.3 erreicht werden.

Bei Verarbeitung mehrdimensionaler Ergebnisse, wie dies gewöhnlich bei Maschinen mit mehr als einer Achse der Fall ist, bietet die Intervallmethode den

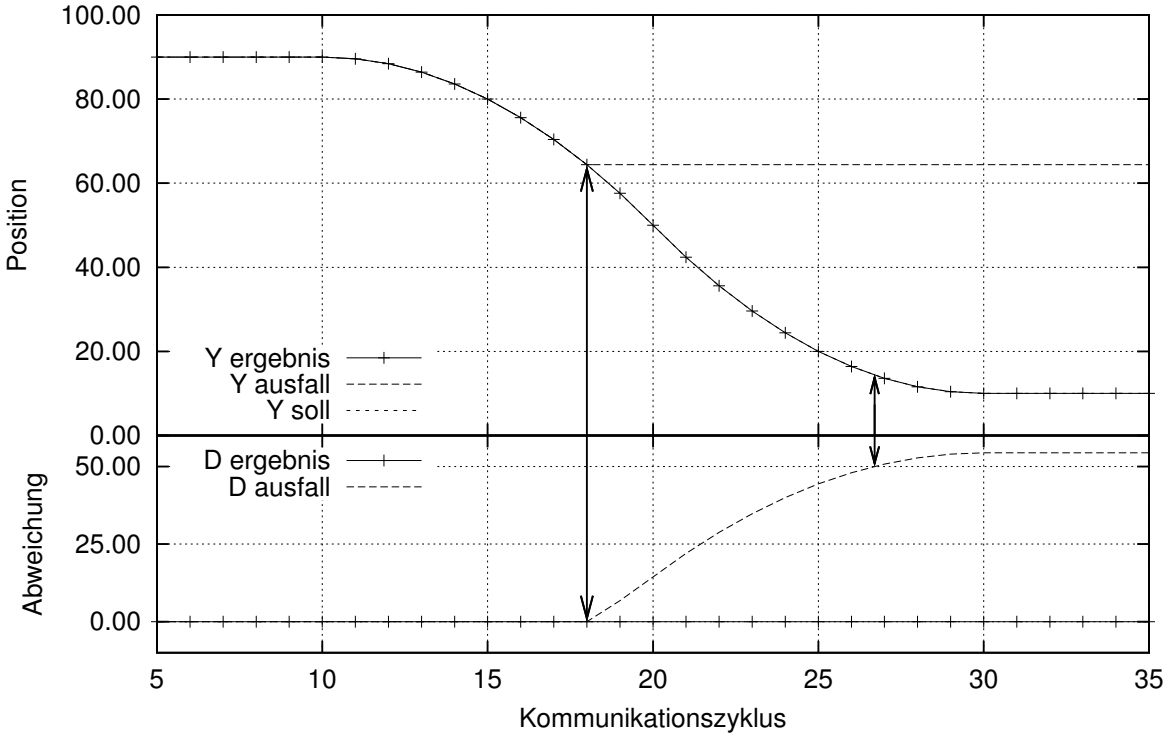


Abbildung 7.5.: Ergebnis des Entscheiders mit Median

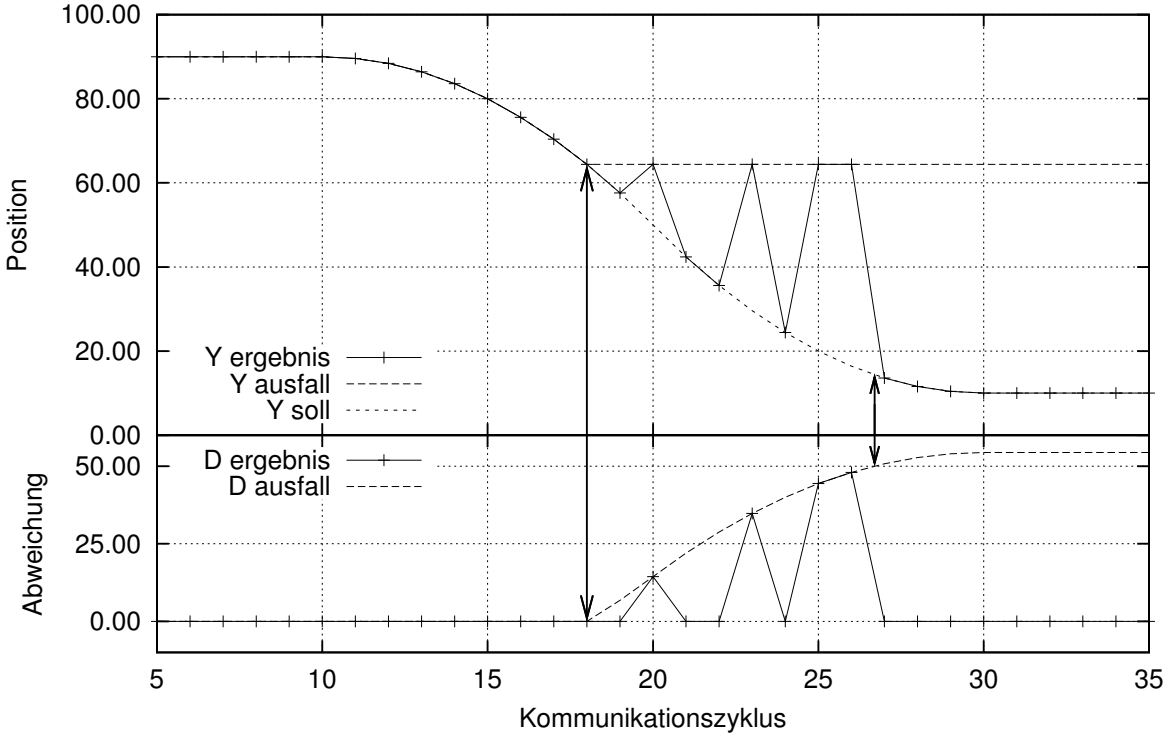


Abbildung 7.6.: Ungünstig denkbares Ergebnis mit Intervallentscheidung

Vorteil, dass sie für jede Achse die Sollwerte vom Ergebnis der selben Steuerung übernimmt. Nachteilig an der Methode ist der hohe Aufwand zur Implementierung des Entscheiders. Weiterhin benötigt die Methode eine Anpassung in der Auswahl der Sollwerte aus der Ergebnismenge, damit häufige Sprünge vermieden werden.

7.6. Zusammenfassung

In diesem Kapitel wurden vier verschiedene Algorithmen zur Berechnung eines Sollwerts aus einer Menge von redundant berechneten Sollwerten diskutiert. Die naheliegende Methode des arithmetischen Mittels disqualifiziert sich, da ein Fehler in jedem Fall sowohl Auswirkungen auf das berechnete Ergebnis, als auch bei der Ausmaskierung der fehlerhaften Steuerung einen Sprung im Sollwertverlauf erzeugt. Die First-seen Methode ist ein leicht umsetzbarer Algorithmus, der sich zudem gut auf einen Systemaufbau mit verteilten Entscheidern in den Steuerungen aus Kapitel 5.3 abbilden lässt. Hierbei muss hinsichtlich des Bearbeitungsprozesses jedoch besonderes Augenmerk darauf gelegt werden, dass beim Ausmaskieren der aktuell sollwertgebenden Steuerung im Sollwertverlauf in jedem Fall ein Sprung in Höhe der tolerierbaren Abweichung entsteht.

Diese Sprünge im Sollwertverlauf können bei der Intervallmethode nach Definition immer auftreten, was bei einer Verwendung dieser Methode hinsichtlich des Bearbeitungsprozesses wiederum besonders berücksichtigt werden muss. Die Median Methode erscheint von allen dargestellten Methoden in den meisten Fällen am geeignetsten zu sein, da sie bei einer fehlerhaften, noch nicht ausmaskierten Steuerung die geringsten Abweichungen aufzeigt sowie bei der Ausmaskierung keine Sprünge auftreten.

Mit den in diesen und vorigen Kapiteln besprochenen Verfahren wurde ein Demonstrator realisiert, um die Wirksamkeit zu validieren. Im folgenden Kapitel wird zunächst der Aufbau des realisierten Systems dokumentiert. Danach wird weiterhin anhand von Fehlersimulationstests die Wirksamkeit der Verfah-

ren nachgewiesen. Der Nachweis erfolgt dabei für das unterbrechungsfreie Weiterarbeiten sowie die Rekonfiguration in ein fehlersicheres System bei einem Fehler in einer Steuerung des dreifach redundanten Steuerungssystems. Weiterhin wird der Wechsel in einen sicheren Zustand bei einem Fehler in einer der Steuerungen des fehlersicheren Systems aufgezeigt.

8. Realisierung

Das im Folgenden vorgestellte System wurde im Rahmen eines Forschungsprojekts der Deutschen Forschungsgemeinschaft (DFG) aufgebaut um die Realisierbarkeit einer fehlertoleranten numerischen Steuerung zu validieren [VB05, VB06, VB08, VB09a, VB09b, VB10]. Im Folgenden wird zunächst der Aufbau des Systems erläutert. Im zweiten Teil wird das System mit Fehlersimulationstests beaufschlagt und die Reaktionen darauf vorgestellt und diskutiert.

8.1. Aufbau des Demonstrationssystems

Der Demonstrationsaufbau bestand aus drei numerischen Steuerungen und einer Hardware in the Loop (HIL) Maschinensimulation [PR04]. Dieses System wurde mit der Bearbeitung eines NC Programms beauftragt. Zur Veranschaulichung und Validierung der Fehlertoleranz wurde die Position der X-Achse in jedem Kommunikationszyklus aufgezeichnet. Das Bewegungsprofil der X-Achse aus dem NC Programm ist in Abbildung 8.1 abgebildet.

8.1.1. Systemaufbau und -topologie

Für den Aufbau des Demonstrators wurde ein 2 von 3 System mit der Topologie aus Kapitel 5.1 eingesetzt, bei der die Mehrheitsentscheidung dezentral verteilt in den Antriebsverstärkern ablief. Auf diese Weise wurde keine gemeinsame Fehlerquelle (Single Point of Failure) in das Steuerungssystem eingebracht.

Im Aufbau des Demonstrators wurde das Softwarepaket Virtuos der Firma Industrielle Steuerungstechnik GmbH verwendet. Dieses Softwarepaket ermög-

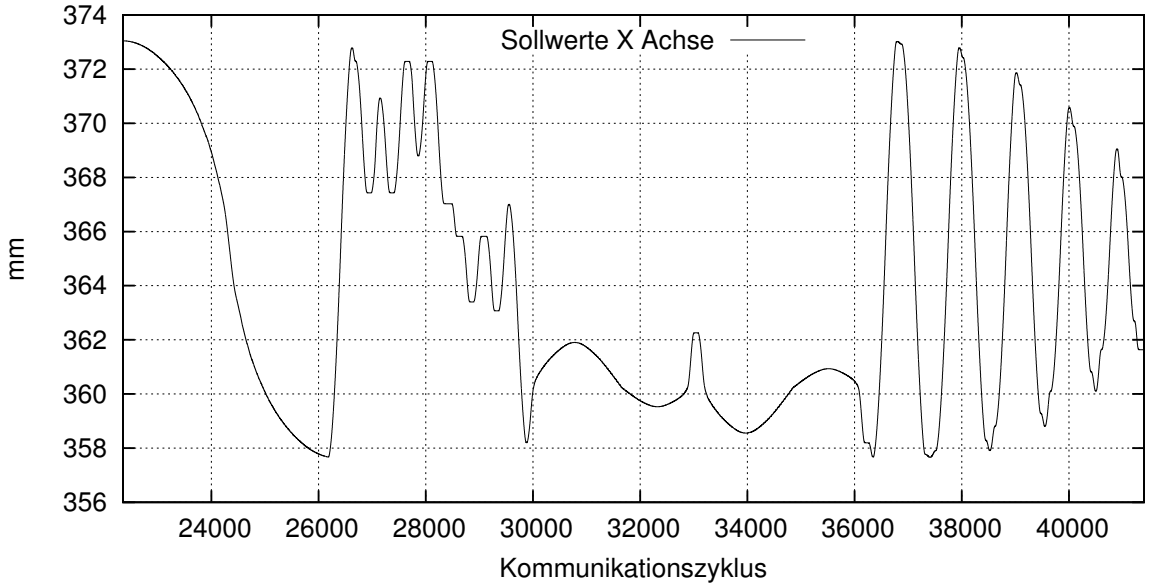


Abbildung 8.1.: Plot der X-Achse des zum Test genutzten NC Programms

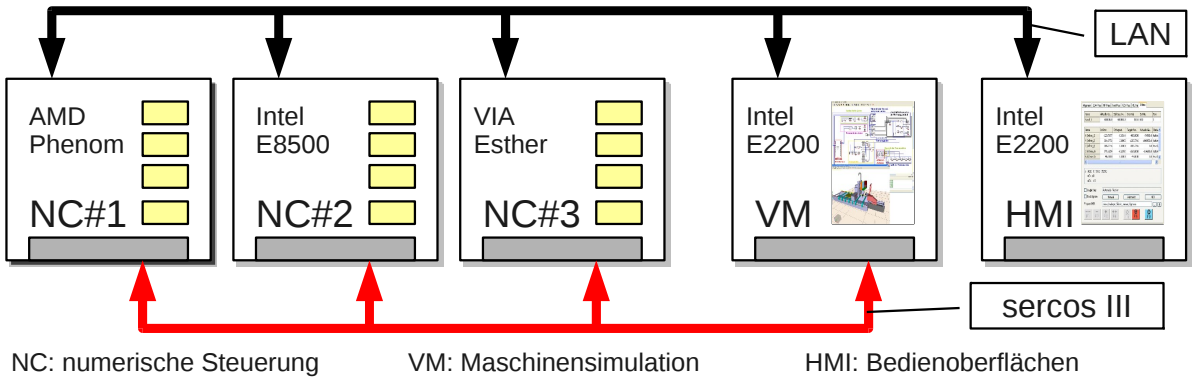


Abbildung 8.2.: Hardwareaufbau des Demonstrators

licht sowohl die echte HIL-Simulation einer kompletten Maschine, als auch die Integration des benötigten Mehrheitsentscheids. Da diese HIL-Simulation leicht um eigene Funktionalität erweitert werden kann, konnte der Mehrheitsentscheid leicht integriert werden. Die Problemstellung, dass kaum ein Antriebshersteller sowohl die Quellcodes als auch seine Entwicklungsumgebung zum Erweitern der Antriebsfirmware zur Verfügung stellt, wurde damit umgangen.

Der Aufbau des Demonstrators basierte auf fünf PCs aus diversitärer Hardware, siehe Abbildung 8.2. Zur Kommunikation waren zwei verschiedene Schnittstellenkarten eingesetzt.

Die Schnittstellenkarte für das LAN wurde zur allgemeinen, nicht echtzeitrelevanten Kontrolle und Überwachung der Rechner verwendet. Die Schnittstellenkarte für sercos III kam sowohl zur Synchronisation von numerischen Steuerungen und virtueller Maschine, als auch für die zyklische und deterministische Verteilung der Daten zum Einsatz.

Auf den ersten drei PCs, gekennzeichnet mit „NC#“, lief jeweils ein numerischer Steuerungskern. Der vierte PC, gekennzeichnet mit „VM“, simulierte mit Virtuos die zu steuernde Maschine. Auf dem fünften PC, der Bedienoberfläche „HMI“, liefen sowohl die Bedienoberfläche für Steuerungen sowie Virtuos, als auch die Visualisierung der simulierten Maschine.

8.1.2. Synchronisation der NC Kanäle

Für Abgleich und Synchronisation wurde der Ansatz aus Kapitel 3.5 in vereinfachter Form verwendet. Diese ist in Abbildung 8.3 dargestellt. Die Vereinfachung bestand darin, dass nur der deterministische, hochpriore Prozess jeder numerischen Steuerung synchronisiert wurde. Der niederpriore Prozess blieb freilaufend.

Damit die hochprioren Prozesse der redundanten numerischen Steuerungen synchron abliefen, mussten die eingehenden Daten synchronisiert werden. Das heißt, dass alle Bahninterpolationsblöcke exakt die selbe Menge an Daten vom vorherigen Bahnplanungsblock verarbeiten müssen. Um dies zu gewährleisten wurde der zwischen Bahnplanung und Bahninterpolation liegende Pufferspeicher so erweitert, dass die Datenentnahme synchronisiert erfolgen konnte. In Abbildung 8.3 sind die synchronisierten Pufferspeicher mit einem eingekreisten „S” markiert.

Damit keiner der Bahninterpolationsblöcke pro Interpolationstakt zu viel Daten aus dem Pufferspeicher entnimmt, darf jeder Bahninterpolationsblock maximal bis zum Füllstand des am geringsten befüllten Pufferspeichers lesen. Dazu teilte jede redundante Steuerung in ihren Synchronisationsdaten allen anderen Steuerungen zyklisch den Füllstand ihres eigenen Pufferspeichers mit. Anhand

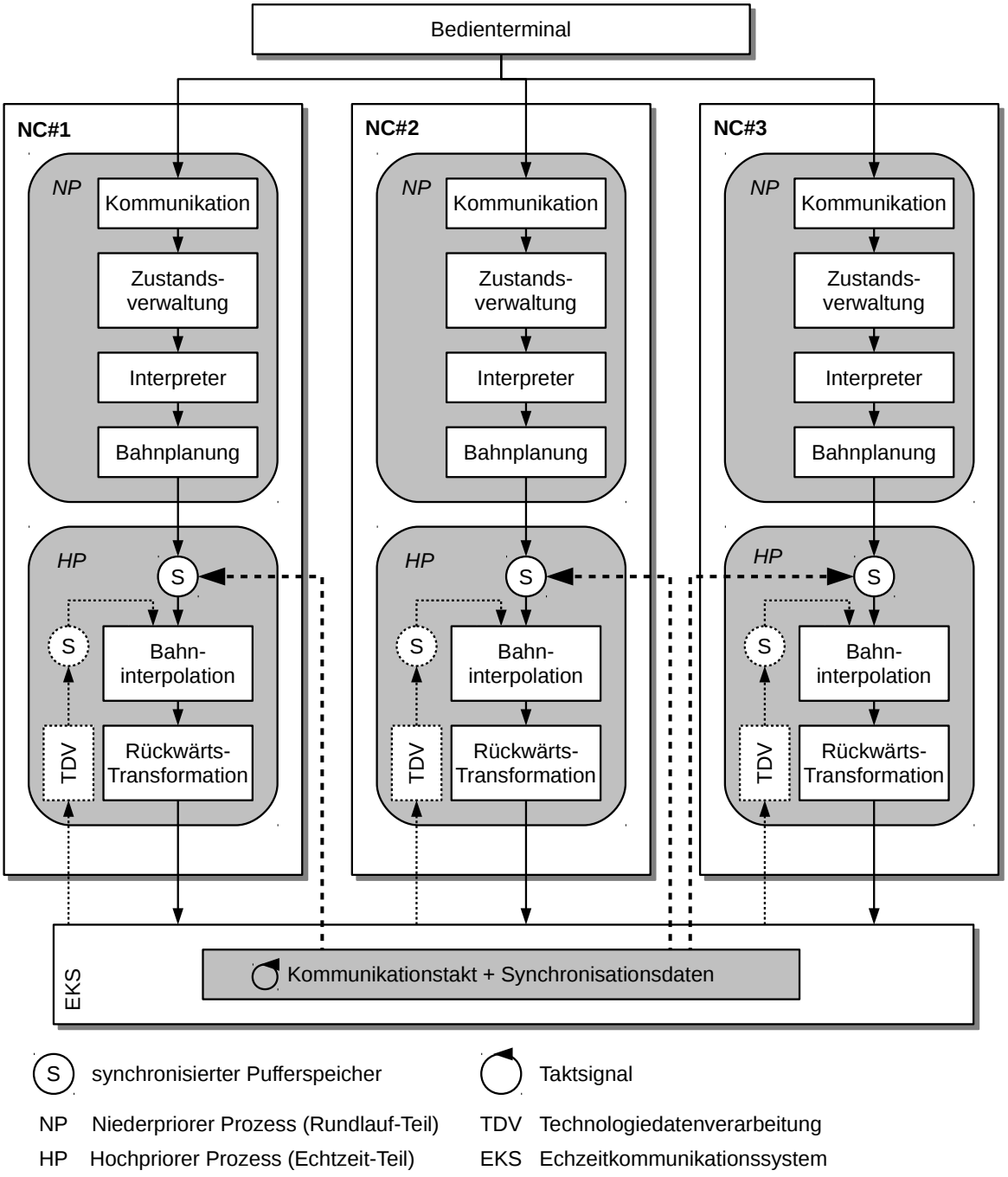


Abbildung 8.3.: Synchronisation der NC Kanäle

dieser Füllstände ermittelte die Bahninterpolation jeder redundanten Steuerung die Position, bis zu der sie Daten aus dem Pufferspeicher auslesen durfte, indem sie das Minimum aller übermittelten Pufferspeicherfüllstände berechnete.

Für den zyklischen und deterministischen Datenaustausch wurde der Ansatz mit verteiltem Speicher aus Kapitel 6.3 durch das echtzeitfähige Kommunikationssystem sercos III umgesetzt. Dieses ist in Abbildung 8.3 mit „EKS" markiert. Es genügte der Störungstoleranz aus Kapitel 6 mit seiner redundanten Doppelringstruktur bereits von Haus aus. Im Falle eines einfachen Leitungsausfalls, dem sogenannten Ringbruch, hätte sercos III von der Doppelringstruktur zu einer einfachen Linienstruktur gewechselt. Dieser Wechsel wäre von sercos III automatisch durchgeführt und hätte keiner Behandlung durch die Applikation bedurft. Der Ausfall eines Kabels im Betrieb konnte damit toleriert werden.

Jeder der an das Echtzeit-Kommunikationssystem angeschlossenen PCs erzeugte Daten, die von allen Steuerungen benötigt wurden. Die Antriebsverstärker, in diesem Fall die virtuelle Maschine, erwarteten zyklisch Sollwerte von jeder einzelnen Steuerung. Im Gegenzug benötigte jede Steuerung die Istwerte aller Antriebsverstärker. Zusätzlich fielen in jeder Steuerung Daten an den Synchronisierungspunkten an, die allen übrigen Steuerungen mitgeteilt werden mussten. Insgesamt gab es damit die drei folgenden verschiedenen Kommunikationsinhalte *Synchronisationsdaten*, *Sollwerte* und *Istwerte*.

Die Kommunikationsbeziehungen dieses Szenarios entsprachen einer typischen Multicast Anwendung. Hierbei werden Daten eines Produzenten an mehrere Konsumenten übertragen, ohne diese mehrfach Senden zu müssen. Diese Kommunikationsbeziehungen wurden auf die mit sercos III eingeführten Querverbindungen abgebildet. Eine Querverbindung in einem sercos III Netzwerk besteht dabei aus einem Produzenten und einem oder mehreren Konsumenten. Für jeden Verbindungsteilnehmer, Produzent oder Konsument, lässt sich die Art und Weise festlegen, wie die Verbindungsdaten zu interpretieren sind.

Der einfacheren Konfiguration wegen wurden für den Demonstrator die Kommunikationsinhalte *Synchronisationsdaten* und *Istwerte* in einer gemeinsamen

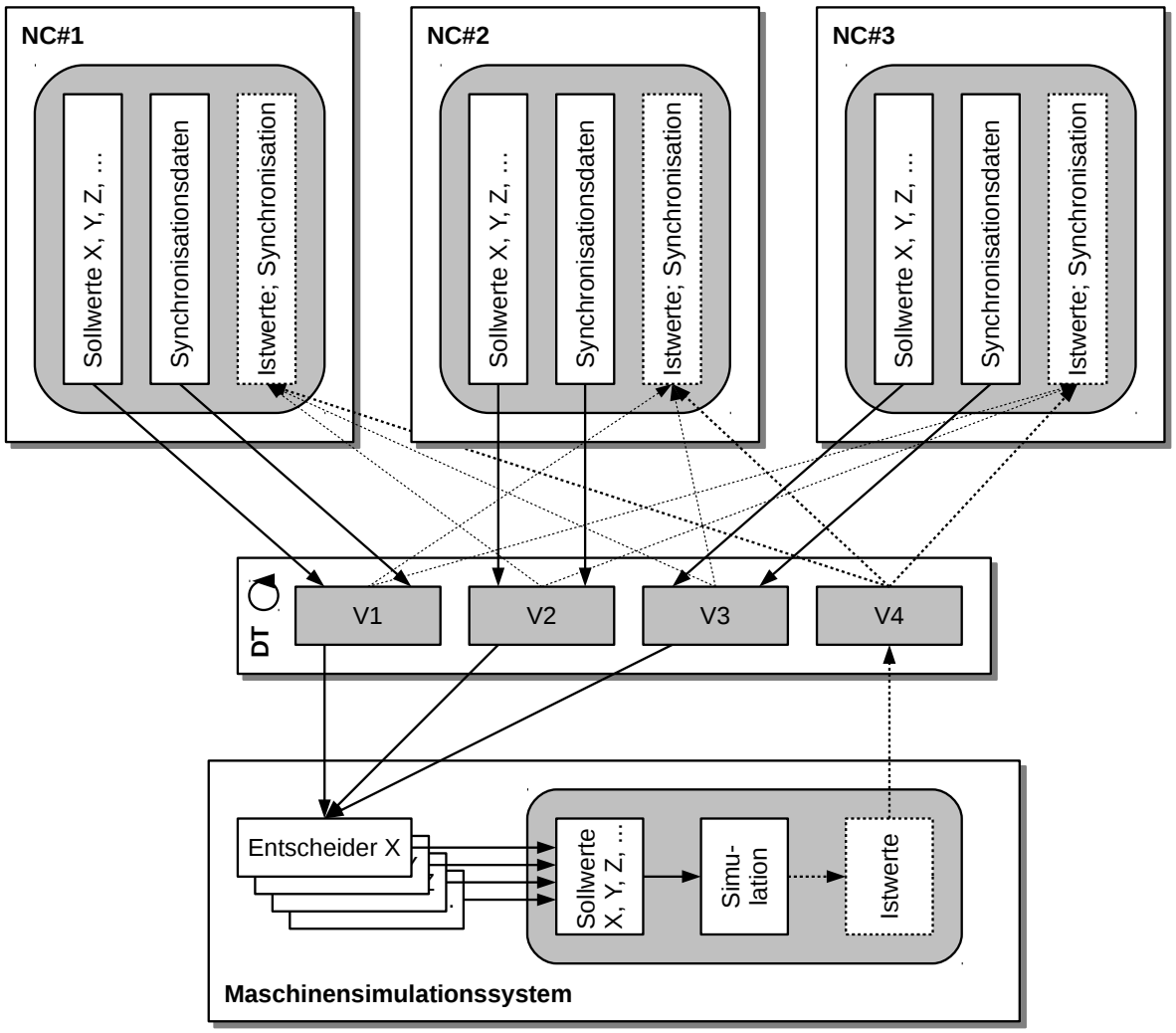
Verbindung zusammengefasst. Somit gab es im Demonstratorsystem insgesamt vier Verbindungen V1 bis V4. Jedes angeschlossene Gerät, die drei Steuerungen sowie die virtuelle Maschine, produzierten eine Verbindung und konsumierten drei Verbindungen. Abbildung 8.4 zeigt den Systemaufbau mit den konfigurierten Querverbindungen.

8.1.3. Fehlerreaktion

Für die Reaktion auf festgestellte Fehler kam das dreistufige Konzept nach Abbildung 4.3 auf Seite 47 zum Einsatz. Im Zustand „Fehlertolerant“ lief das System mit drei redundanten Steuerungen und einer 2 von 3 Mehrheitsentscheidung, bei welcher die Sollwerte nach dem Median-Algorithmus aus Kapitel 7.4 ausgewertet wurden. Der Eigenschaft des Median-Algorithmus, dass bei zwei verbleibenden Steuerungen keine Entscheidung getroffen werden kann, wurde durch den Ablauf in Abbildung 8.5 Rechnung getragen. Darin wurde der Fall, dass nur noch zwei Steuerungen bestimmungsgemäß funktionieren, durch eine Sonderbehandlung abgefangen. Der Entscheider verhielt sich in diesem Zustand wie die fehlersichere numerische Steuerung.

Bei Auftreten von zu großer Abweichung der Sollwerte einer Steuerung gegenüber dem Sollwert der Mehrheitsentscheidung sollte der Zustand des Mehrheitsentscheiders in den Zustand „Fehlersicher“ wechseln. Während des Zustandsübergangs wurde die Steuerung mit zu großer Abweichung ausgegliedert und deren Sollwerte damit nicht weiter beachtet. Fortgefahren wurde mit dem First-Seen-Algorithmus aus Kapitel 7.2, da der Median bei zwei verbleibenden Steuerungen nicht mehr funktioniert. Die beim Median zuletzt benutzte Steuerung blieb dabei als sollwertgebende Steuerung aktiv. Die übrige Steuerung blieb passiv und ihre Sollwerte wurden nur noch zum Vergleich herangezogen.

Stellte der Mehrheitsentscheider nun wiederum eine zu große Abweichung zwischen den Sollwerten der beiden Kanäle fest, so sollte er in den Zustand „Not-Halt“ wechseln und damit einen sicheren Zustand herstellen. In diesem



→ von Steuerungen produzierte Sollwerte und Synchronisationsdaten

⇢ von Steuerungen konsumierte Istwerte

⇢ von Steuerungen konsumierte Synchronisationsdaten

DT Datentelegramm des Echtzeitkommunikationssystems (vereinfacht)

V1..4 Logische Verbindung 1..4 mit je einem Produzenten und mehreren Konsumenten

Abbildung 8.4.: Verbindungen in fehlertolerantem numerischen Aufbau

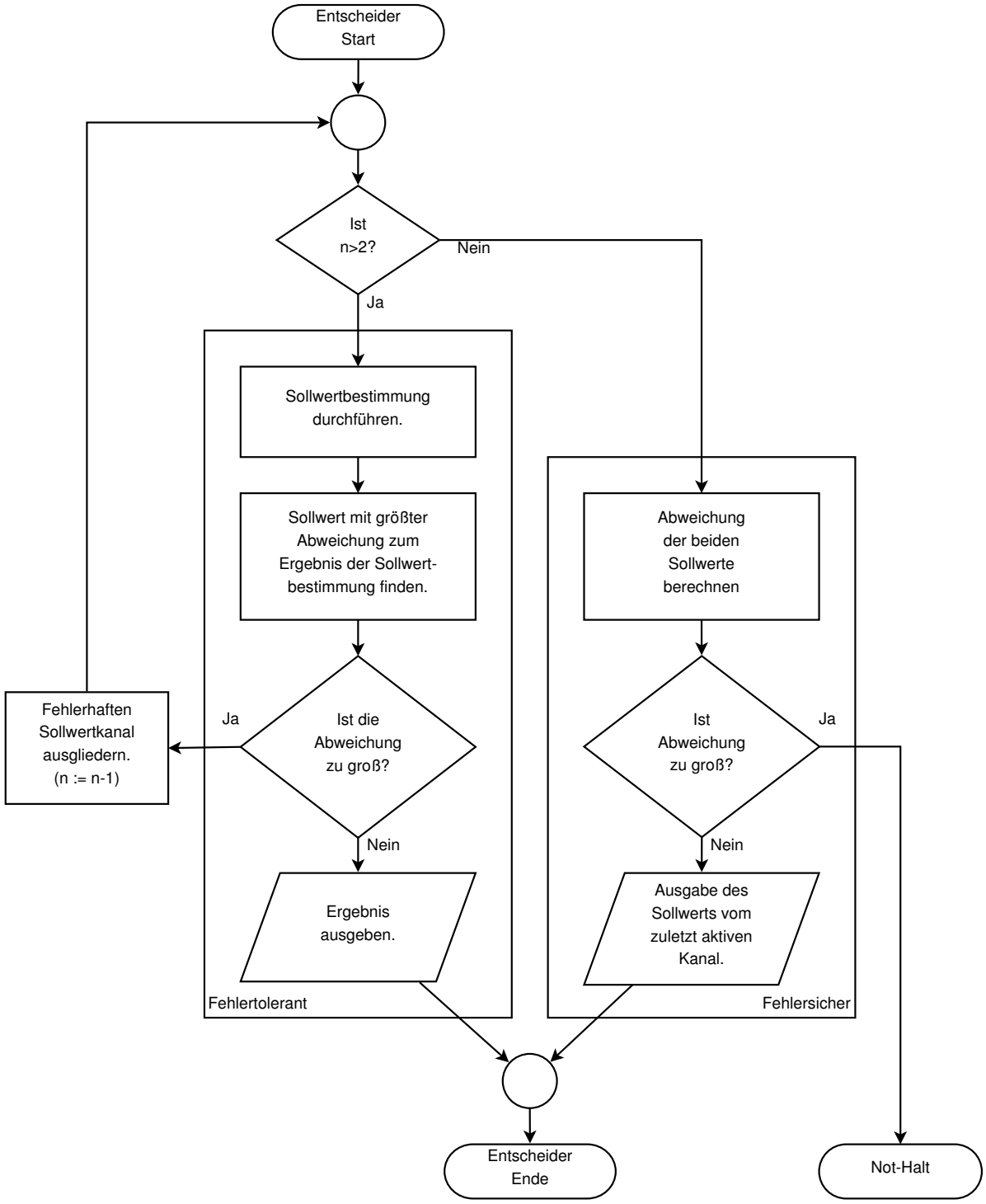


Abbildung 8.5.: Ablauf der fehlertoleranten Mehrheitsentscheidung

Zustand gab der Mehrheitsentscheider fortan ausschließlich noch den zuletzt für gültig befundenen Sollwert aus.

8.2. Fehlersimulationstest

Zum Test der fehlertoleranten Funktion des Steuerungssystems wurde während des Betriebs das Programm einer Steuerung abgebrochen um einen Absturz zu simulieren. Weiterhin wurde einer der übrigen numerischen Steuerungen vorab ein modifiziertes Programm übergeben um einen unerkannten Speicherfehler zu simulieren. Beide simulierten Fehler führten nach dem Start des Programms in den fehlerbehafteten numerischen Steuerungen zu Abweichungen der berechneten Sollpositionen gegenüber der Sollbahn.

8.2.1. Auswertung der Reaktion auf den simulierten Programmabsturz

Für die im Folgenden präsentierten Messergebnisse war eine Kommunikationszykluszeit von $2ms$ eingestellt. Der Entscheider war so eingestellt, dass er bei Überschreitung einer Toleranzgrenze von $0,2mm$ eingriff. Alle Soll- und Istwertangaben haben die Einheit $10^{-7}m$ $(0,1\mu m)$. Aufgezeichnet wurden die folgenden Daten in jedem einzelnen Kommunikationszyklus:

- die Sollwerte aller Steuerungen
- der Zustand des Entscheiders
- das Ergebnis des Entscheiders
- die Synchronisationsdaten

Tabelle 8.1 zeigt die aufgezeichneten Daten zum Zeitpunkt des ersten Ausfalls. Die Spalte „Zustand“ gibt den Zustand des Entscheiders wieder. Die Kennung „T“ steht hierbei für *fehlertolerant*. In diesem Zustand berechnete der Entscheider seinen Sollwert über die Medianfunktion aus Kapitel 7.4. Die Kennung „S“ steht für *fehlersicher*. Hier verhielt sich der Entscheider nach dem Mecha-

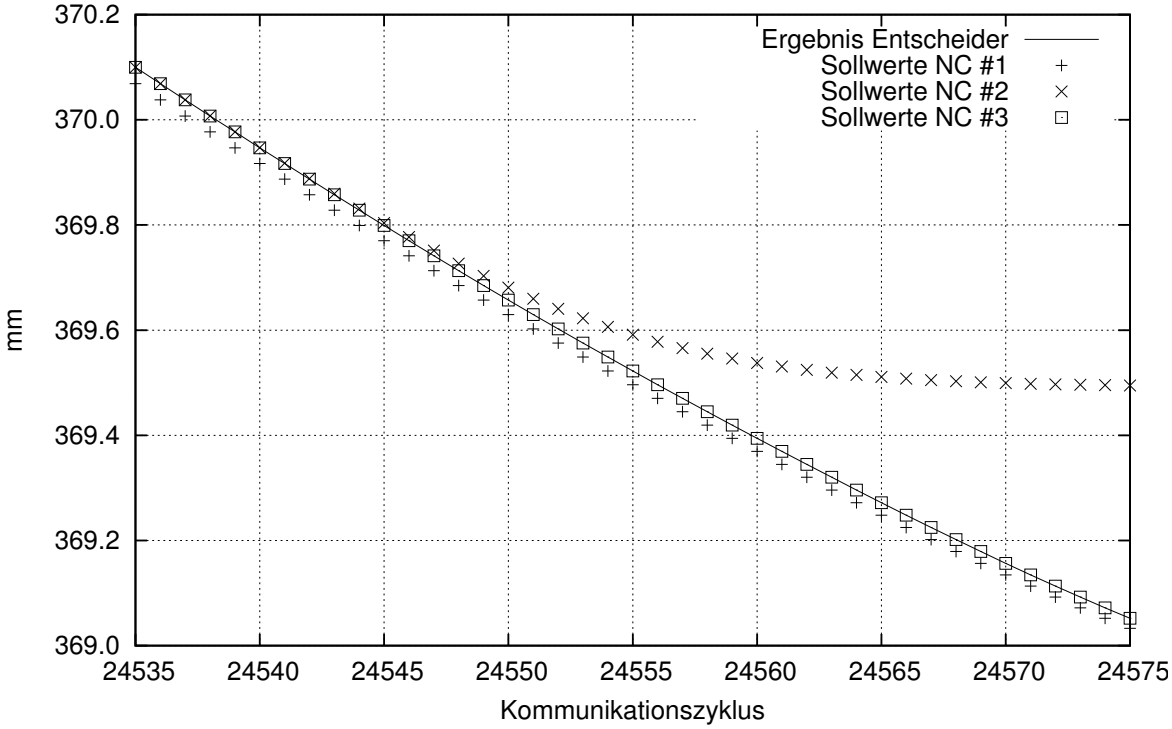


Abbildung 8.6.: Sollwerte des Steuerungssystems zum Zeitpunkt des ersten Ausfalls

nismus „First-Seen“ aus Kapitel 7.2, dem selben Mechanismus wie in der fehlersicheren numerischen Steuerung.

Deutlich zu erkennen ist, dass alle drei Steuerungen zunächst identische Sollwerte ausgeben und synchron zueinander liefen. Steuerung 1 hatte in dieser Messung ein reproduzierbare Verzögerung von exakt einem Kommunikationszyklus trotz der Synchronisationsmaßnahmen. Für die Validierung des fehlertoleranten Ansatzes war dies jedoch nicht relevant. [VB10]

Die Abbildung 8.6 zeigt neben den Sollwerten aller drei Steuerungen auch die Ausgabe des Entscheiders zum Zeitpunkt des Programmabbruchs auf Steuerung 2. Die Sollwerte der drei Steuerungen sind als Punkte, das Ergebnis des Entscheiders als Linie eingezeichnet. Der Bearbeitungsprozess wurde demnach mit korrekten Sollwerten innerhalb der Toleranzgrenze fortgesetzt, obwohl die Steuerung 2 ausgefallen war.

Zyklus	NC #1	NC #2	NC #3	Ent.	Zust.	aktiv	passiv	maskiert
24535	3700687	3700997	3700997	3700997	T	n/a	n/a	n/a
24536	3700379	3700687	3700687	3700687	T	n/a	n/a	n/a
24537	3700073	3700379	3700379	3700379	T	n/a	n/a	n/a
24538	3699769	3700074	3700073	3700073	T	n/a	n/a	n/a
24539	3699467	3699771	3699769	3699769	T	n/a	n/a	n/a
24540	3699168	3699471	3699467	3699467	T	n/a	n/a	n/a
24541	3698870	3699175	3699168	3699168	T	n/a	n/a	n/a
24542	3698574	3698883	3698870	3698870	T	n/a	n/a	n/a
24543	3698281	3698595	3698574	3698574	T	n/a	n/a	n/a
24544	3697990	3698313	3698281	3698281	T	n/a	n/a	n/a
24545	3697701	3698038	3697990	3697990	T	n/a	n/a	n/a
24546	3697414	3697771	3697701	3697701	T	n/a	n/a	n/a
24547	3697130	3697513	3697414	3697414	T	n/a	n/a	n/a
24548	3696849	3697265	3697130	3697130	T	n/a	n/a	n/a
24549	3696571	3697030	3696849	3696849	T	n/a	n/a	n/a
24550	3696296	3696807	3696571	3696571	T	n/a	n/a	n/a
24551	3696023	3696598	3696296	3696296	T	n/a	n/a	n/a
24552	3695754	3696404	3696023	3696023	T	n/a	n/a	n/a
24553	3695487	3696224	3695754	3695754	T	n/a	n/a	n/a
24554	3695223	3696061	3695487	3695487	T	n/a	n/a	n/a
24555	3694962	3695912	3695223	3695223	T	n/a	n/a	n/a
24556	3694704	3695778	3694962	3694962	T	n/a	n/a	n/a
24557	3694448	3695658	3694704	3694704	T	n/a	n/a	n/a
24558	3694194	3695552	3694448	3694448	T	n/a	n/a	n/a
24559	3693942	3695459	3694194	3694194	T	n/a	n/a	n/a
24560	3693693	3695377	3693942	3693942	T	n/a	n/a	n/a
24561	3693446	3695307	3693693	3693693	T	n/a	n/a	n/a
24562	3693201	3695246	3693446	3693446	T	n/a	n/a	n/a
24563	3692959	3695193	3693201	3693201	T	n/a	n/a	n/a
24564	3692719	3695148	3692959	3692959	S	3	1	2
24565	3692482	3695110	3692719	3692719	S	3	1	2
24566	3692247	3695078	3692482	3692482	S	3	1	2
24567	3692016	3695051	3692247	3692247	S	3	1	2
24568	3691789	3695028	3692016	3692016	S	3	1	2
24569	3691565	3695009	3691789	3691789	S	3	1	2
24570	3691346	3694993	3691565	3691565	S	3	1	2
24571	3691132	3694980	3691346	3691346	S	3	1	2
24572	3690922	3694970	3691132	3691132	S	3	1	2
24573	3690718	3694961	3690922	3690922	S	3	1	2
24574	3690521	3694953	3690718	3690718	S	3	1	2
24575	3690329	3694947	3690521	3690521	S	3	1	2

Tabelle 8.1.: Zustandsdaten des Systems zum Zeitpunkt des ersten Ausfalls

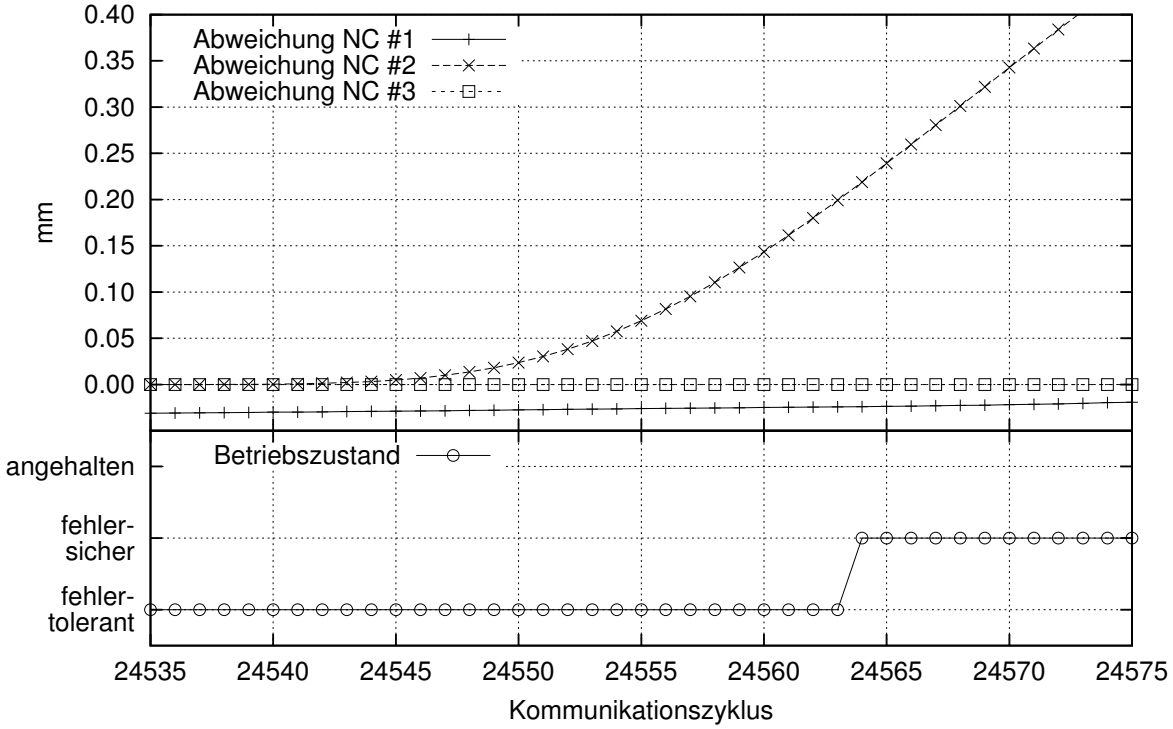


Abbildung 8.7.: Abweichung und Betriebszustand zum Zeitpunkt des ersten Ausfalls

Bemerkenswert in Hinblick auf Kapitel 3.2.4 ist, dass die von den numerischen Steuerungen berechneten Werte im Demonstrator bis auf das niederwertigste Bit identisch waren.

Weiterhin zeigt sich, dass Steuerung 2 ab Kommunikationszyklus 24.538 durch den Programmabbruch die Bewegung stoppte. Die Abweichungen zwischen Sollwerten und Ausgang des Entscheiders sind in Abbildung 8.7 dargestellt. Die Abweichung von Steuerung 1 war, abhängig von der Verfahrgeschwindigkeit, konstant. Die Abweichung der ausgefallenen Steuerung 2 wuchs schnell an. Ab Kommunikationszyklus 24.564 überschritt sie die Toleranzgrenze von $0,2mm$. Der Entscheider schaltete darauf unmittelbar in den Betriebszustand „Fehlersicher“ um. Sollwerte von Steuerung 2 maskierte der Entscheider fortan und verwendete diese nicht weiter für seine Berechnungen.

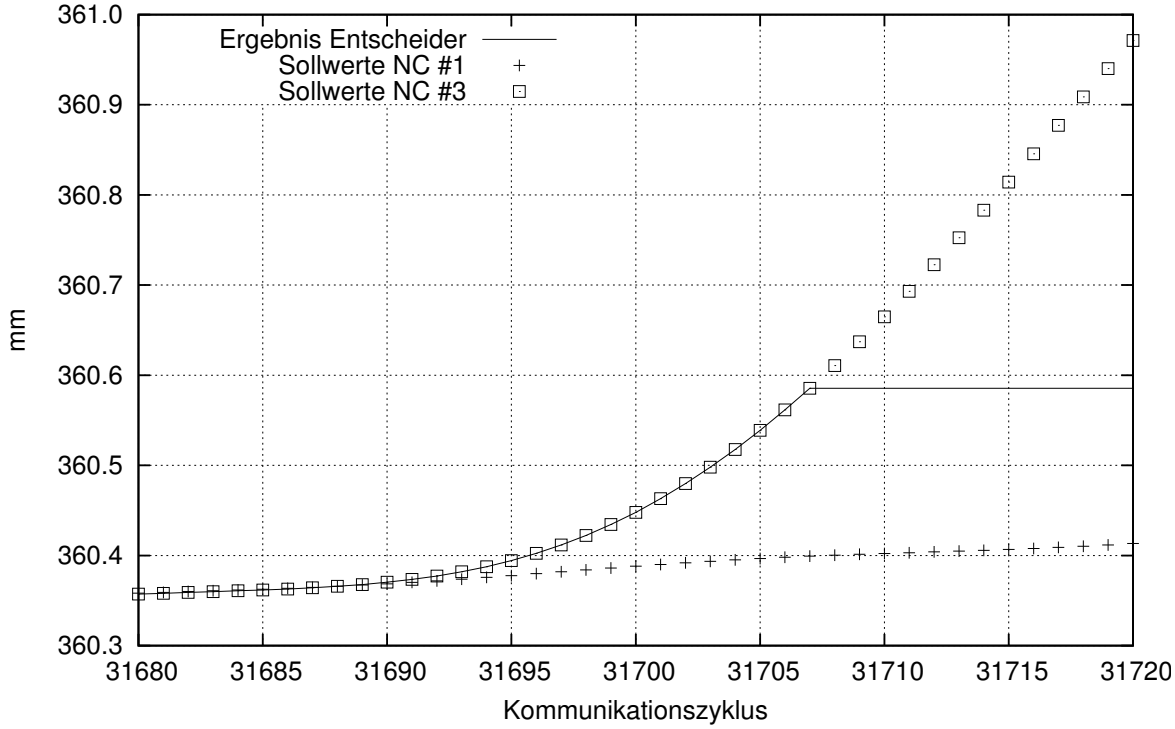


Abbildung 8.8.: Sollwerte des Steuerungssystems zum Zeitpunkt des zweiten Ausfalls

8.2.2. Auswertung der Reaktion auf den simulierten Speicherfehler

Die Sollwerte der übrigen Steuerungen zum Zeitpunkt des zweiten Ausfalls zeigt Abbildung 8.8. Hier wichen die Sollwerte durch das modifizierte NC Programm voneinander ab. Tabelle 8.2 zeigt, dass der Entscheider zunächst bis einschließlich Kommunikationszyklus 31.707 die Sollwerte von Steuerung 3 ausgab. Im folgenden Zyklus schaltete er in den Zustand „Not-Halt“ mit der Kennung „N“ und damit in den Betriebszustand „angehalten“. Fortan gab er nur noch den zuletzt gültigen Sollwert aus.

Abbildung 8.9 zeigt die steigende Abweichung der beiden Steuerungen sowie die Sicherheitsabschaltung des Entscheiders bei Überschreiten der Toleranzgrenze von $0,2mm$ in Kommunikationszyklus 31.708. Die dargestellte Abweichung bezieht sich auf die Differenz zwischen Sollwert der numerischen Steuerung und dem vom Entscheider ausgegebenen Wert. Aus diesem Grund

Zyklus	NC #1	NC #2	NC #3	Ent.	Zust.	aktiv	passiv	maskiert
31680	3603582	3694922	3603571	3603571	S	3	1	2
31681	3603591	3694922	3603581	3603581	S	3	1	2
31682	3603600	3694922	3603590	3603590	S	3	1	2
31683	3603608	3694922	3603599	3603599	S	3	1	2
31684	3603617	3694922	3603608	3603608	S	3	1	2
31685	3603626	3694922	3603617	3603617	S	3	1	2
31686	3603636	3694922	3603628	3603628	S	3	1	2
31687	3603647	3694922	3603641	3603641	S	3	1	2
31688	3603659	3694922	3603658	3603658	S	3	1	2
31689	3603672	3694922	3603678	3603678	S	3	1	2
31690	3603686	3694922	3603703	3603703	S	3	1	2
31691	3603702	3694922	3603734	3603734	S	3	1	2
31692	3603719	3694922	3603773	3603773	S	3	1	2
31693	3603737	3694922	3603820	3603820	S	3	1	2
31694	3603757	3694922	3603876	3603876	S	3	1	2
31695	3603777	3694922	3603944	3603944	S	3	1	2
31696	3603798	3694922	3604024	3604024	S	3	1	2
31697	3603819	3694922	3604116	3604116	S	3	1	2
31698	3603840	3694922	3604222	3604222	S	3	1	2
31699	3603861	3694922	3604343	3604343	S	3	1	2
31700	3603881	3694922	3604479	3604479	S	3	1	2
31701	3603900	3694922	3604630	3604630	S	3	1	2
31702	3603919	3694922	3604797	3604797	S	3	1	2
31703	3603936	3694922	3604979	3604979	S	3	1	2
31704	3603952	3694922	3605176	3605176	S	3	1	2
31705	3603967	3694922	3605388	3605388	S	3	1	2
31706	3603980	3694922	3605615	3605615	S	3	1	2
31707	3603992	3694922	3605854	3605854	S	3	1	2
31708	3604003	3694922	3606107	3605854	N	n/a	n/a	n/a
31709	3604013	3694922	3606371	3605854	N	n/a	n/a	n/a
31710	3604022	3694922	3606647	3605854	N	n/a	n/a	n/a
31711	3604031	3694922	3606931	3605854	N	n/a	n/a	n/a
31712	3604040	3694922	3607225	3605854	N	n/a	n/a	n/a
31713	3604048	3694922	3607525	3605854	N	n/a	n/a	n/a
31714	3604057	3694922	3607831	3605854	N	n/a	n/a	n/a
31715	3604067	3694922	3608142	3605854	N	n/a	n/a	n/a
31716	3604078	3694922	3608456	3605854	N	n/a	n/a	n/a
31717	3604090	3694922	3608772	3605854	N	n/a	n/a	n/a
31718	3604103	3694922	3609088	3605854	N	n/a	n/a	n/a
31719	3604117	3694922	3609402	3605854	N	n/a	n/a	n/a
31720	3604133	3694922	3609714	3605854	N	n/a	n/a	n/a

Tabelle 8.2.: Zustandsdaten des Systems zum Zeitpunkt des zweiten Ausfalls

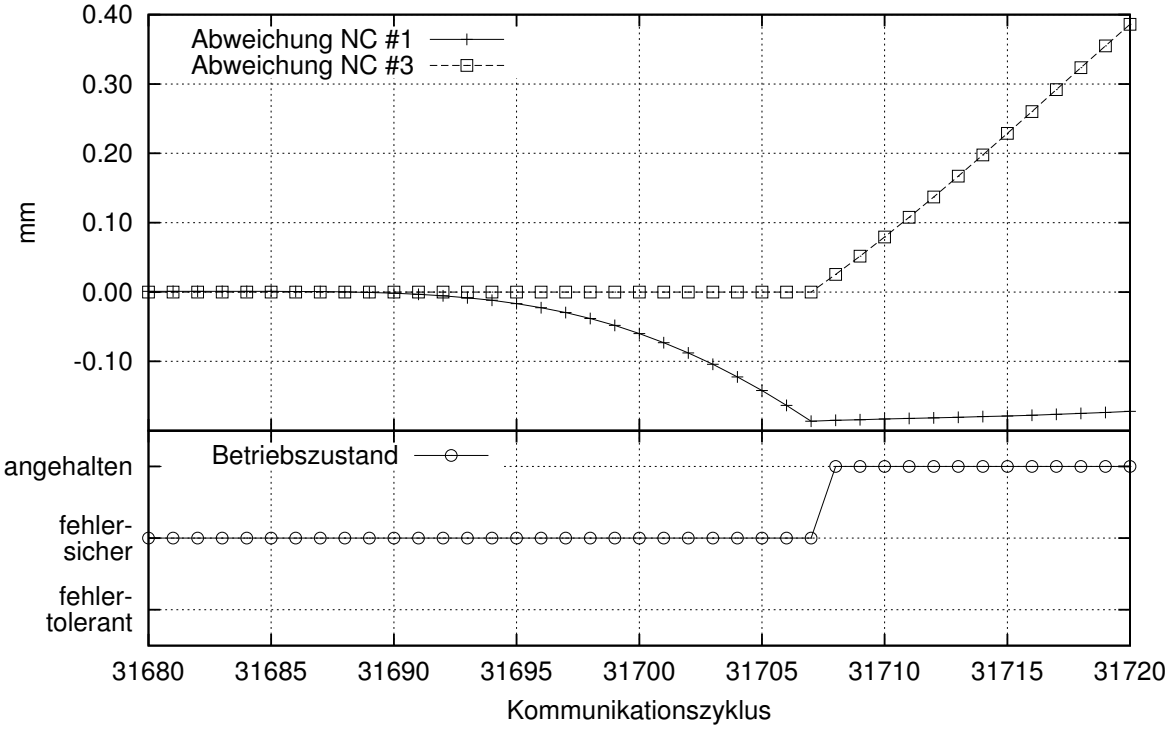


Abbildung 8.9.: Abweichung und Betriebszustand zum Zeitpunkt des zweiten Ausfalls

laufen ab dem Wechsel in den fehlersicheren Zustand die Werte beider numerischen Steuerungen auseinander.

8.3. Bewertung der Ergebnisse

Abbildung 8.10 zeigt die während des Testlaufs vom Entscheider ausgegebenen Sollwerte für die X-Achse. Die beiden Pfeile markieren die beiden Ausfälle und damit die Zeitpunkte der Reaktionen des Entscheiders. An beiden Stellen wurde eine Rekonfiguration des Systems nach Abbildung 4.3 auf Seite 47 durchgeführt.

Zum Zeitpunkt des ersten Ausfalls ist keine Abweichung von der Sollbahn aus Abbildung 8.1 festzustellen. Die vom Entscheider ausgegebenen Sollwerte wichen durch das Ausmaskieren der fehlerhaften Steuerung nicht von der Sollbahn ab. Die Bearbeitung konnte lückenlos fortgesetzt werden.

Das fehlersichere numerische Steuerungssystem hätte an diesem Punkt bereits einen Not-Halt eingeleitet. Die Bearbeitung wäre ab hier unterbrochen und würde manuelles Eingreifen erfordern. Dennoch wäre ein kontrollierter, sicherer Zustand gegeben.

Ein normales Steuerungssystem hätte an diesem Punkt ebenso die korrekte Bearbeitung unterbrochen. Allerdings wäre hier, je nach Art des Fehlers, kein Not-Halt eingeleitet worden. Im ungünstigsten Fall würden fehlerhafte Sollwerte ausgegeben und das Werkstück damit beschädigt werden.

Zum Zeitpunkt des zweiten Ausfalls ist erkennbar, dass das fehlertolerante numerische Steuerungssystem einen Not-Halt einleitete und die Bearbeitung anhielt. Somit war ein kontrollierter sicherer Zustand gegeben.

Die im Demonstratoraufbau aufgezeichneten Daten zeigen, dass es möglich ist mehrere numerische Steuerungen miteinander zu synchronisieren. Der sehr hohe Determinismus der Daten legt nahe, dass der vorhandene Zeitversatz zwischen Steuerung 1 und den übrigen Steuerungen systematischer Natur ist. Für den reinen Funktionsnachweis der Fehlertoleranz war dieser Zeitversatz nicht relevant. Die Ursache muss durch eine nochmalige Untersuchung gefunden und behoben werden.

Anhand des Demonstratoraufbaus wurde nachgewiesen, dass die Verbindung mehrerer redundanter numerischer Steuerungen zu einer fehlertoleranten numerischen Steuerung machbar ist. Die Zuverlässigkeit des Steuerungssystems erhöht sich damit durch die auf Redundanz aufbauenden Maßnahmen zur Fehlerkompensation- und Fehlerausgrenzung deutlich.

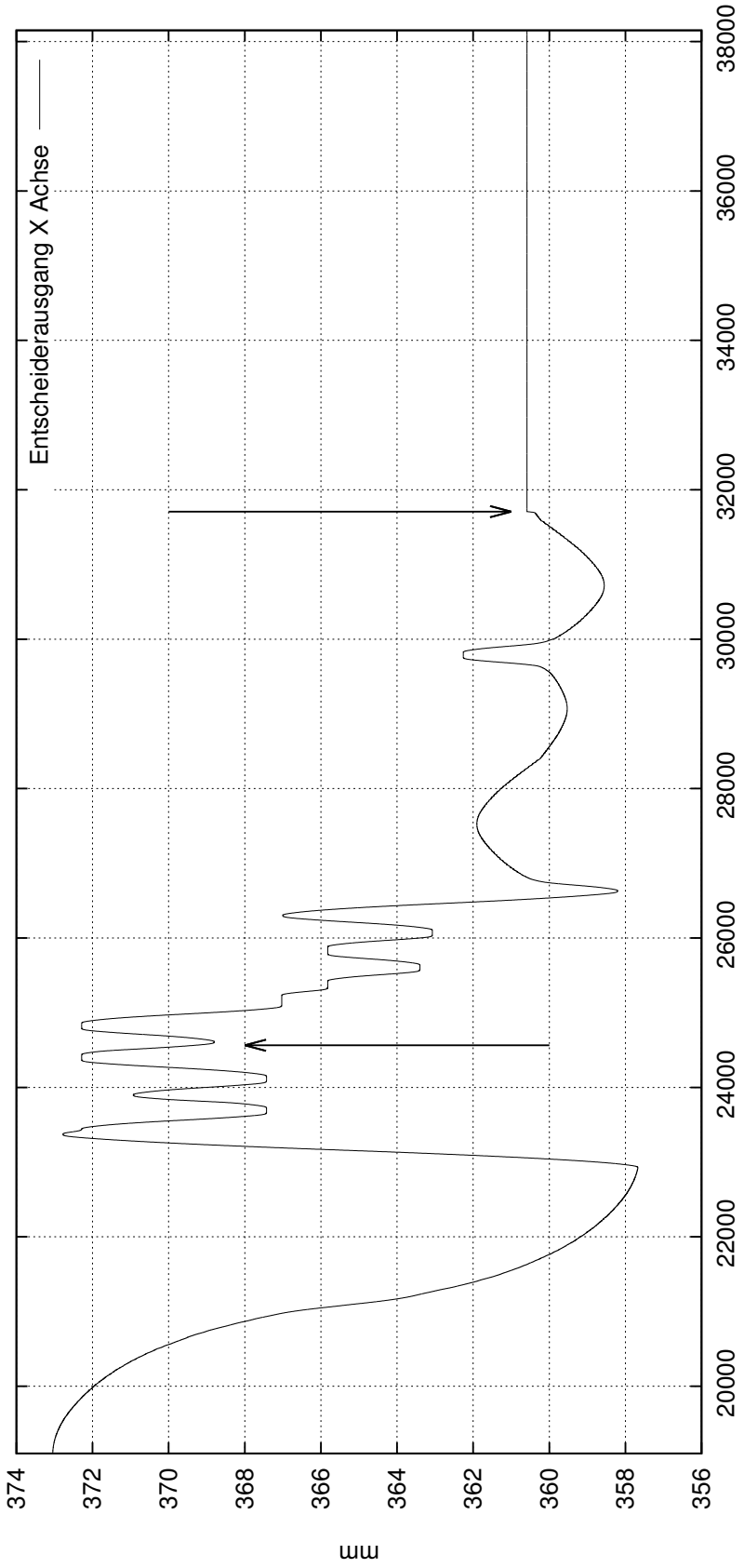


Abbildung 8.10.: Vom Entscheider ausgegebene Sollwerte für die X-Achse

9. Zusammenfassung und Ausblick

Durch immer leistungsfähiger werdende Steuerungshardware dringen numerische Steuerungen in neue Anwendungsbereiche vor, die bisher Ablaufsteuerungen oder dem Menschen vorbehalten waren. Hierbei werden numerische Steuerungen zunehmend auch in sicherheitskritischen Anwendungen eingesetzt, was den Steuerungen neue und sehr hohe Anforderungen an Sicherheit und Zuverlässigkeit abfordert.

Ein Beispiel für diese neuen, sehr hohen Anforderungen sind robotergestützte Operationsassistenzsysteme für die minimalinvasive Chirurgie. Für diese Systeme kommen bereits numerische Steuerungen zum Einsatz, die durch einen zweikanaligen Aufbau bei Auftreten eines Fehlers in einen sicheren Zustand übergehen. Dieser Ansatz erhöht die Sicherheit des Systems maßgeblich. Die Zuverlässigkeit leidet jedoch darunter.

Das Ziel der Arbeit war die Erhöhung der Wahrscheinlichkeit, dass es durch Fehler bei der Sollwerterzeugung nach Beginn des Bearbeitungsprozesses weder zu einem Ausfall noch zu einem Schaden kommt. Dieses Ziel wurde durch das entworfene fehlertolerante numerische Steuerungssystem erreicht. Dabei bot das System die gleich hohe Sicherheit wie der fehlersichere Ansatz und erhöhte die Zuverlässigkeit gleichzeitig deutlich.

Die Erhöhung der Zuverlässigkeit des sollwerterzeugenden Teils der numerischen Steuerung wurde durch strukturelle Maßnahmen mit hybrider Redundanz sowie einer Mehrheitsentscheidung durchgeführt. Die Möglichkeiten, einzelne numerische Steuerungen zu einem redundanten Gesamtsystem mit Mehrheitsentscheidung zusammenzufügen wurden aufgezeigt. So gibt es mehrere Möglichkeiten den Entscheider im Gesamtsystem zu positionieren mit unter-

schiedlichen Vor- und Nachteilen. Ebenso wurde dargelegt, dass die einzelnen numerischen Steuerungen synchron zueinander laufen müssen und wie der dazu notwendige Datenaustausch und die Synchronisation funktionieren können. Weiterhin wurden verschiedene Algorithmen besprochen, durch welche ein gültiger Sollwert aus einer Menge von redundant berechneten Sollwerten bestimmt werden kann.

Zur Validierung der untersuchten Maßnahmen wurde ein Demonstrator der fehlertoleranten numerischen Steuerung realisiert. Dieser Demonstrator wurde gezielt mit Fehlern versehen und damit die Wirksamkeit der diskutierten Maßnahmen nachgewiesen. So konnte bei Auftreten des ersten Fehlers in einer der redundanten Steuerungen, der Gesamtprozess ohne Beeinträchtigung weitergeführt werden.

Diese Arbeit hat damit praktisch die Machbarkeit einer fehlertoleranten numerischen Steuerung nachgewiesen. Durch diesen ersten Aufbau ergeben sich eine ganze Menge weiterer möglicher Forschungsfelder.

So wird in der Arbeit das betrachtete System noch auf den sollwertgebenden Teil des Steuerungssystems eingeschränkt. Für eine vollständige Betrachtung muss jedoch die gesamte bewegungserzeugende Kette mit einbezogen werden. Damit besteht die Möglichkeit, aus der Steuerung heraus auf Ausfälle in den Antrieben oder gar der Mechanik zu reagieren und Maßnahmen wie beispielsweise eine Änderung der Kinematiktransformation durchzuführen um einzelne Achsen auszumaskieren.

Weiterhin ist der im Demonstrator gewählte Ansatz mit der Integration des Mehrheitsentscheiders in die Antriebe bisher nur mit der virtuellen Werkzeugmaschine möglich. Für einen realen Einsatz werden echte Antriebe benötigt. Hier liegt die Erforschung einer sicheren Integration des Mehrheitsentscheiders beispielsweise in den offenen Antrieb [PK05] nahe.

Zuletzt bleibt noch der herausfordernde Aspekt, nicht nur fehlerhafte Steuerungen ausmaskieren und ausgliedern zu können, sondern vielmehr im laufenden Betrieb, ohne Unterbrechung, eine fehlerfreie Steuerung dem redundanten Gesamtsystem wieder hinzuzufügen.

A. Synchronizität der numerischen Steuerungen

Dieses Kapitel zeigt die Auswertung der Synchronizität der hochprioren Prozesse aller numerischen Steuerungen von der Messung aus Kapitel 8.2. Hierzu wird im Folgenden das Einfügen und Herausnehmen von Daten aus dem Pufferspeicher zwischen nieder- und hochpriorem Prozess aus Abbildung 8.3 betrachtet.

Die Füllstände der Pufferspeicher aller drei numerischen Steuerungen sind in Abbildung A.1 zusätzlich zur Ausgabe der X-Achse des Entscheiders über den gesamten Messablauf dargestellt. Dazu sind die Füllstände der Pufferspeicher jeweils mit den beiden Werten *Upper Blockcount* und *Lower Blockcount* angegeben. Der Upper Blockcount gibt an, wie viele Daten der niederpriore Prozess in den Pufferspeicher hineingeschrieben hat. Der Lower Blockcount gibt an, wie viele Daten der hochpriore Prozess dem Pufferspeicher entnommen hat. Verschiedene in Abbildung A.1 markierte Kommunikationszyklen sind dabei betrachtenswert:

5.118 Laden des NC Programms.

12.921 Start des NC Programms.

18.080 Beginn der Werkstückbearbeitung.

24.564 Erster Ausfall durch simulierten Programmabsturz.

31.708 Zweiter Ausfall durch simulierten Speicherfehler.

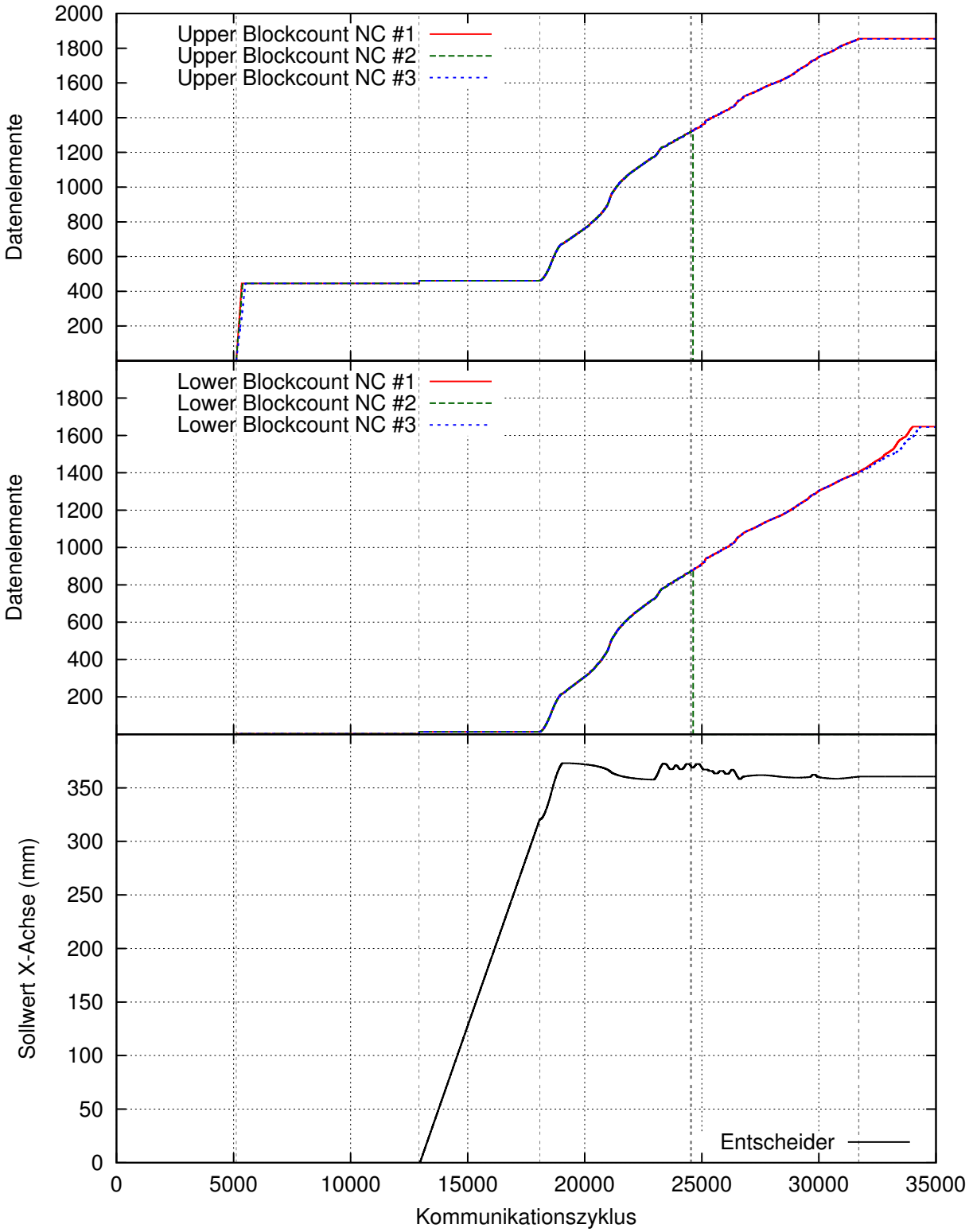


Abbildung A.1.: Pufferspeicherfüllstände und Entscheiderausgang über gesamten Messablauf

A.1. Laden des NC Programms

Bei Kommunikationszyklus 5.118 wurde allen numerischen Steuerungen der Befehl zum Laden des NC Programms gegeben. Der niederpriore Prozess jeder numerischen Steuerung begann daraufhin sofort zu arbeiten und füllte den Pufferspeicher mit Daten. Die Abbildung A.2 zeigt dabei den Füllstand des Pufferspeichers jeder numerischen Steuerung um diesen Zeitpunkt. Hierbei ist deutlich erkennbar, dass die niederprioren Prozesse der numerischen Steuerungen unterschiedlich schnell arbeiten. Ebenso ist erkennbar, was Abbildung A.3 vergrößert darstellt, dass die unterschiedlichen numerischen Steuerungen zu verschiedenen Zeitpunkten mit dem Füllen des Pufferspeichers begonnen haben.

Diese unterschiedlichen Befüllungsstartzeitpunkte und -geschwindigkeiten sind dadurch verursacht, dass verschieden schnelle Rechnersysteme für die unterschiedlichen numerischen Steuerungen eingesetzt wurden, siehe dazu auch Kapitel 8.1.1. Die erste numerische Steuerung lief auf einem Vierkernprozessorsystem, die zweite numerische Steuerung lief auf einem Zweikernprozessorsystem und die dritte numerische Steuerung auf einem Einfachprozessorsystem. Für das Gesamtsystem spielten diese unterschiedlichen Rechengeschwindigkeiten keine Rolle, da der niederpriore Prozess ohnehin nicht synchronisiert ist. Erst der hochpriore Prozess muss synchron zu den anderen Steuerungen ablaufen. Dabei gibt die langsamste Steuerung den Takt vor, da der hochpriore Prozess nicht den Upper Blockcount des eigenen Pufferspeichers verwendet, sondern stattdessen den Minimalwert des Upper Blockcount aller numerischen Steuerungen berechnet und benutzt.

Die Füllstandkurve erreicht nach kurzer Zeit einen Maximalwert und verbleibt auf diesem. Die Pufferspeicher haben hier den maximalen Füllstand erreicht. Erst nach dem Start des NC Programms entnimmt der hochpriore Prozess Daten aus dem Pufferspeicher, so dass der niederpriore Prozess neue Daten einfüllen kann.

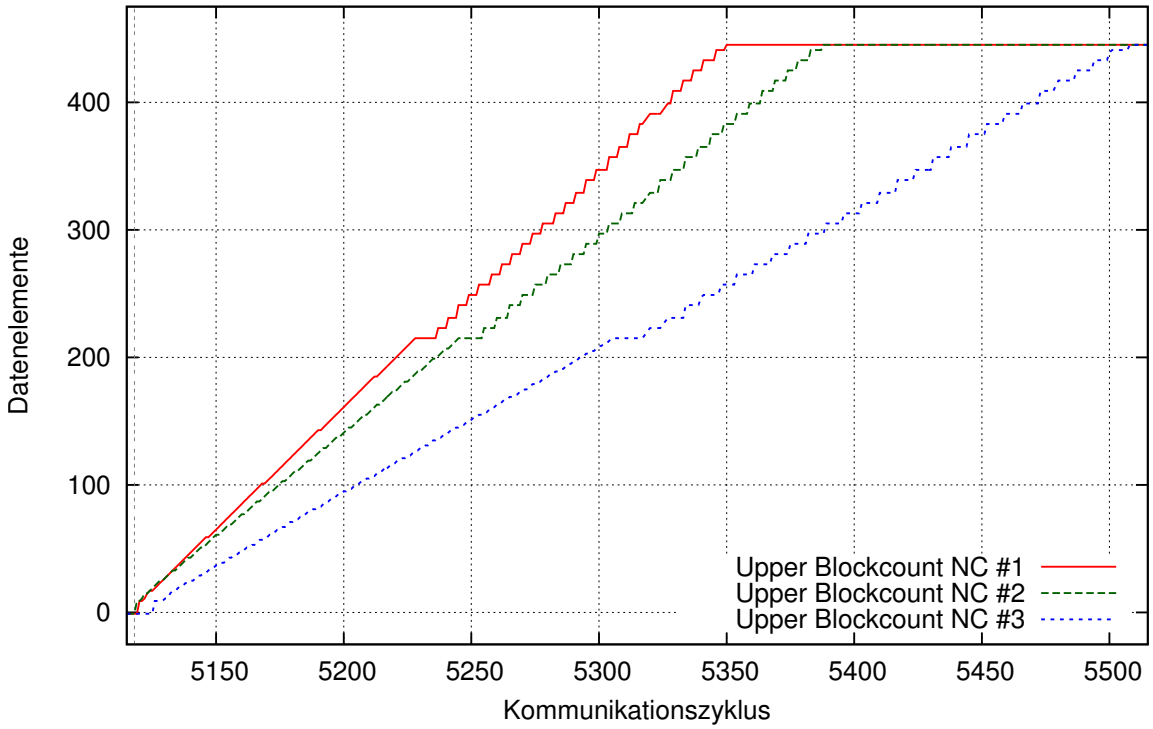


Abbildung A.2.: Anzahl der in die Pufferspeicher geschriebenen Daten um den Programmladezeitpunkt

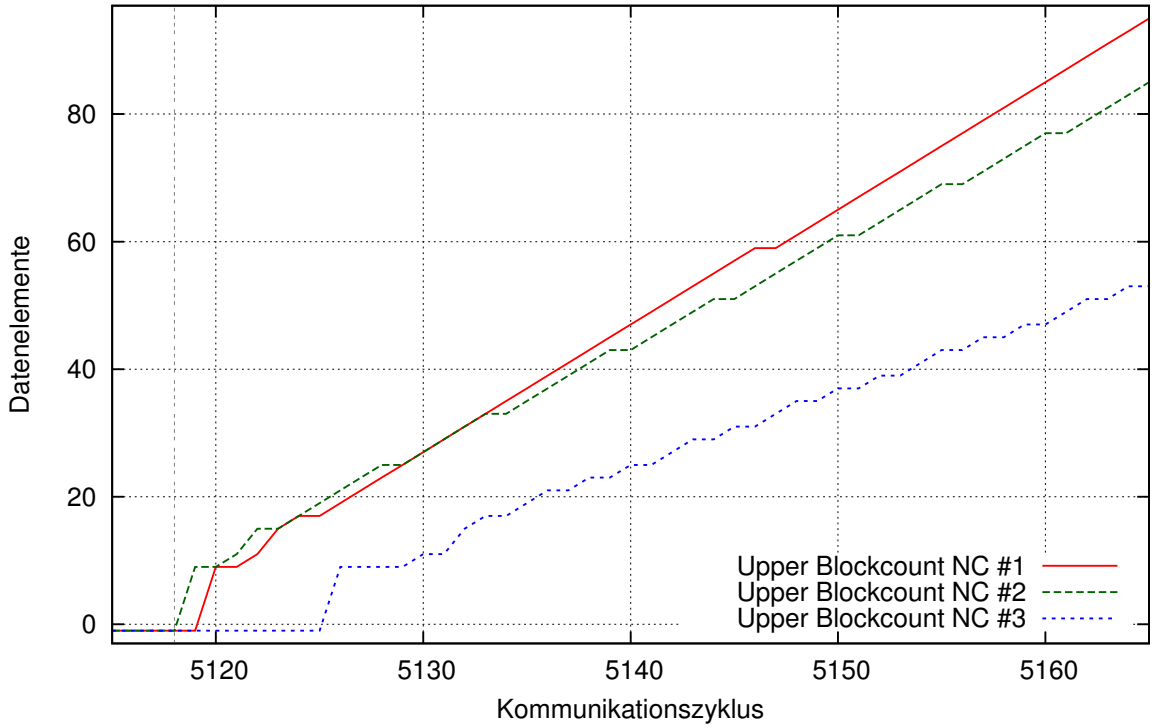


Abbildung A.3.: Unterschiedliche Befüllungsstartzeitpunkte

A.2. Start des NC Programms und Anfahrt an Werkstück

Im Kommunikationszyklus 12.921, siehe Abbildung A.1, wurde das geladene NC Programm auf allen Steuerungen gestartet. An dieser Stelle entnahm der hochpriore Prozess aller numerischen Steuerungen synchron die ältesten Daten aus dem Pufferspeicher. Der niederpriore Prozess füllte daraufhin den Pufferspeicher wieder bis zum maximalen Füllstand auf.

Bis Kommunikationszyklus 18.080 änderte sich der Füllstand des Pufferspeichers nicht, da der hochpriore Prozess den ersten Auftrag abarbeitete und die Achsen an das Werkstück heranfuhr. Dies ist in Abbildung A.1 an der gleichmäßigen Bewegung der X-Achse gut erkennbar. In Abbildung A.4 ist der Beginn der Bearbeitung erkennbar. Ab diesem Zeitpunkt wurden kontinuierlich Daten durch die hochprioren Prozesse der numerischen Steuerungen aus den Pufferspeichern entnommen und durch die niederprioren Prozesse wieder befüllt. Die Position der X-Achse zeigt, dass keine gleichförmige Bewegung mehr vorlag.

A.3. Erster Ausfall

Abweichungen der Sollwerte durch den simulierten Programmabsturz der zweiten numerischen Steuerung sind nach Tabelle 8.1 ab Kommunikationszyklus 24.538 erkennbar. Die Abbildung A.5 zeigt, dass die Pufferspeicherfüllstände bereits bei Kommunikationszyklus 24.510 voneinander abwichen und der hochpriore Prozess keine weiteren Daten mehr aus dem Pufferspeicher entnahm. Die fehlerfreien numerischen Steuerungen liefen dabei ungehindert weiter und der Entscheider gab weiterhin fehlerfreie Sollwerte für die X-Achse aus, bis er die fehlerhafte numerische Steuerung zu Kommunikationszyklus 24.564 ausmaskierte. In Kommunikationszyklus 24.623, an welchem die zweite numerische Steuerung den Pufferspeicher zurücksetzte, war sie somit bereits ausmaskiert. Die verbliebenen fehlerfreien numerischen Steuerungen liefen als fehlersiche-

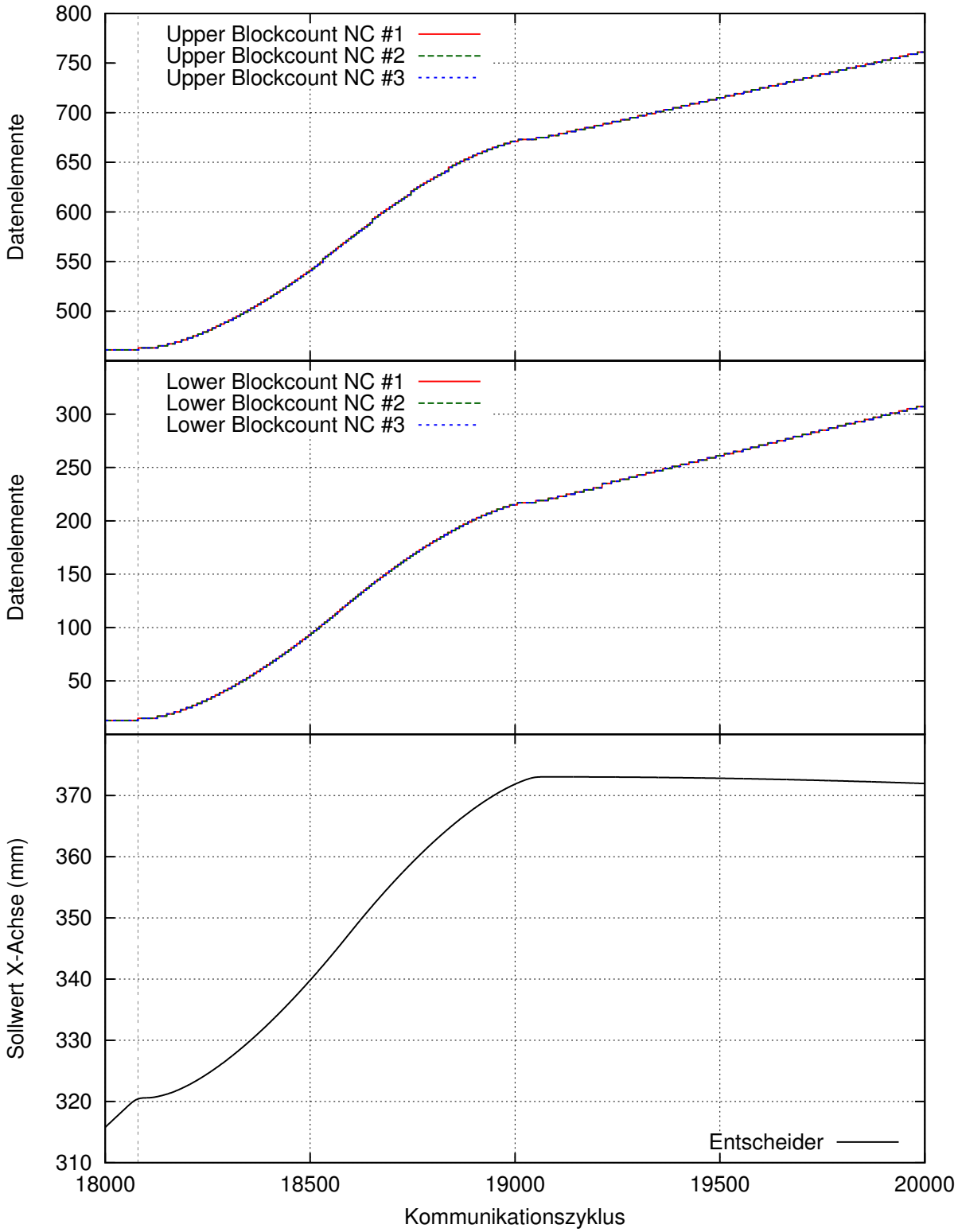


Abbildung A.4.: Pufferspeicherfüllstände und Entscheiderausgang zu Bearbeitungsbeginn

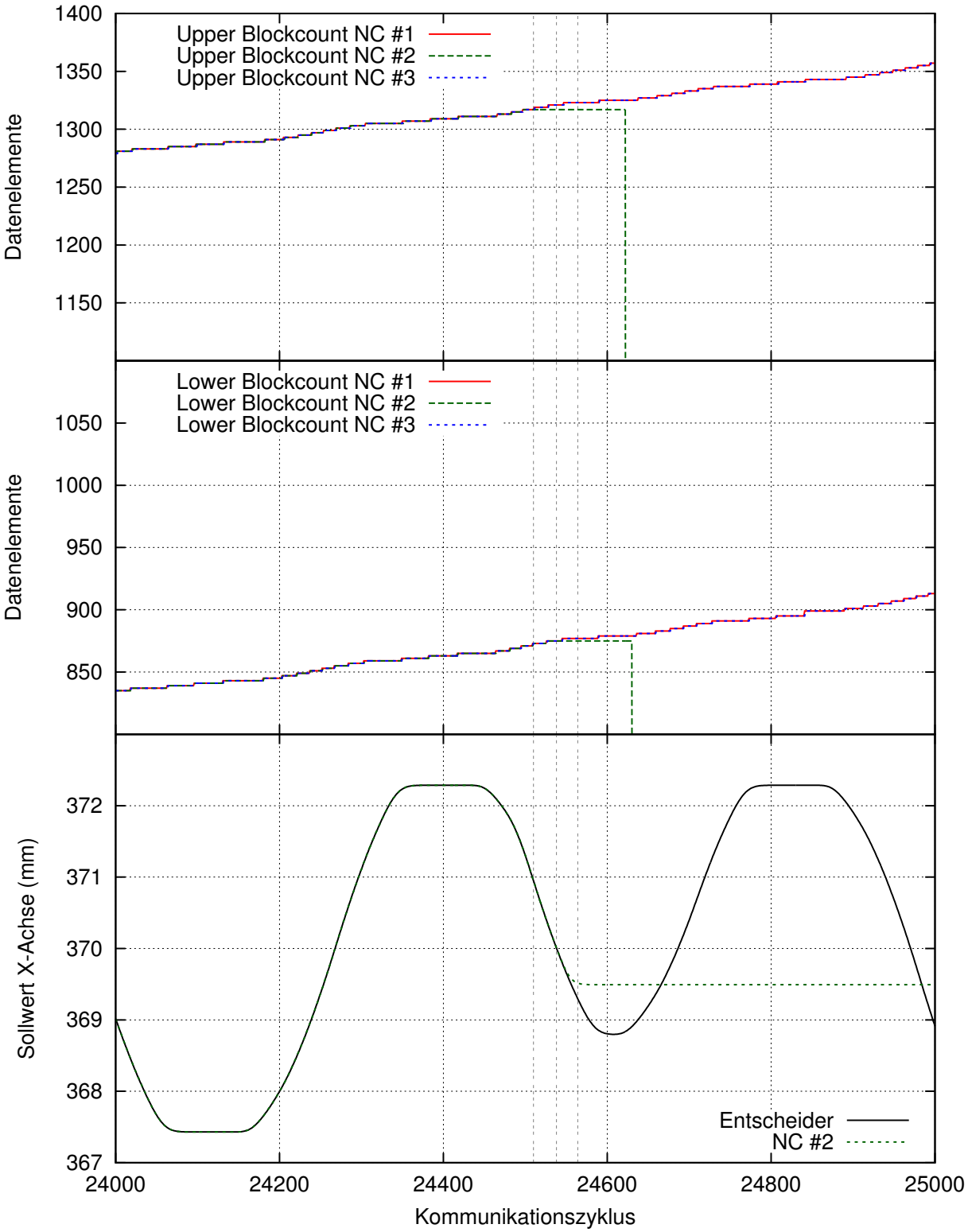


Abbildung A.5.: Pufferspeicherfüllstände und Entscheiderausgang bei erstem Ausfall

res numerisches Steuerungssystem weiter, was der Ausgang des Entscheiders der X-Achse in Abbildung A.5 belegt.

A.4. Zweiter Ausfall

Die Pufferspeicherfüllstände und den Entscheiderausgang während des zweiten Ausfalls zeigt Abbildung A.6. Hier ist erkennbar, dass die verbleibenden beiden Steuerungen ab Kommunikationszyklus 31.694 durch den simulierten Speicherfehler unterschiedliche Mengen von Daten aus dem Pufferspeicher entnahmen. In Kommunikationszyklus 31.708 überschritt die Differenz der ausgegebenen Werte der numerischen Steuerungen die Toleranzgrenze des Entscheiders. Der Entscheider schaltete darauf das System in den Not-Halt, so dass keine fehlerhafte Bewegung ausgeführt werden konnte.

A.5. Bewertung

Die vorhergehende Betrachtung zeigt, dass die hochprioren Prozesse der unterschiedlichen numerischen Steuerungen unabhängig von deren Rechenleistung die Daten aus dem Pufferspeicher synchron entnehmen. Dass die niederprioren Prozesse der numerischen Steuerungen die Daten in unterschiedlicher Geschwindigkeit in den Pufferspeicher schreiben, hat keine Auswirkung auf die Qualität der Bahntreue. Ein Vorauslaufen der schnelleren Steuerungen wird durch die Synchronisation der hochprioren Prozesse wirksam verhindert.

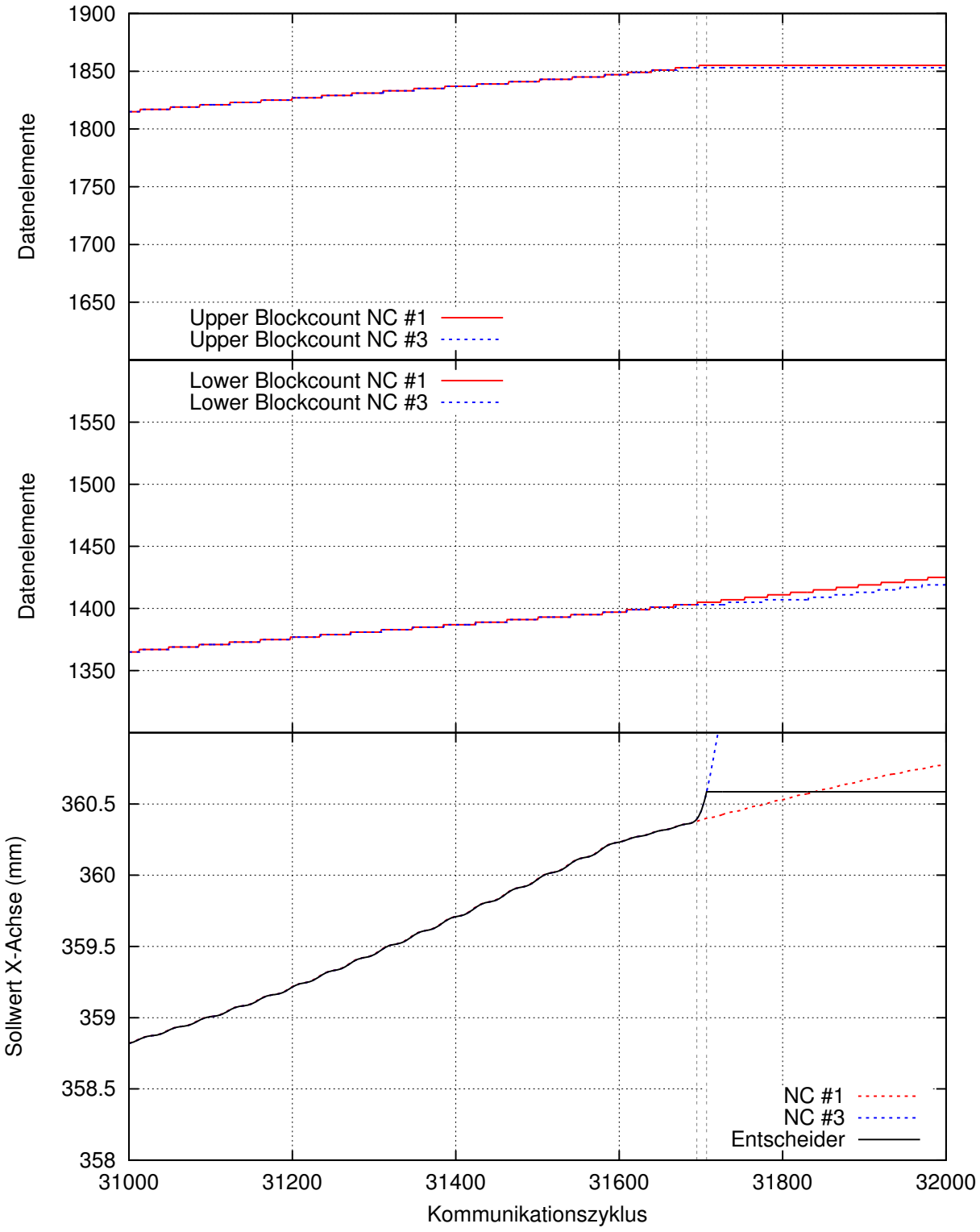


Abbildung A.6.: Pufferspeicherfüllstände und Entscheiderausgang bei zweitem Ausfall

B. Bewertung zeitversetzt laufender Steuerungen

Der zeitliche Versatz der synchron laufenden Steuerungen war für die Messung im realisierten System nach Kapitel 8.2.1 nicht relevant. Dies lag zum einen an dem geringen zeitlichen Versatz von nur einem Kommunikationszyklus, zum anderen daran, dass die verbleibenden zwei Steuerungen absolut synchron im selben Takt liefen. Durch letzteres konnte die Median Methode Ergebnisse ohne störende Wechsel der Steuerungen und Sprünge der Sollwerte ausgeben.

Für schlechtere Randbedingungen, dem zeitlichen Versatz aller Steuerungen, sollen im Folgenden dennoch die Auswirkungen auf jeden der Algorithmen zur Mehrheitsentscheidung aus Kapitel 7 untersucht werden. Vorab wird jedoch die maximal mögliche Positionsabweichung Δs abgeschätzt.

Zu einem Zeitpunkt t sei die vorgegebene Geschwindigkeit einer zu bewegenden Achse gleich $v(t)$. Sollwerte werden hierzu in jedem Kommunikationszyklus Δt_{NC} von zwei gleichartigen Steuerungen vorgegeben, die um Δn Kommunikationszyklen versetzt laufen. Bei konstanter Verfahrgeschwindigkeit $\dot{v}(t) = 0$ berechnet sich die Positionsabweichung Δs zu $\Delta n * \Delta t_{NC} * v(t)$. Die maximale Positionsabweichung lässt sich damit anhand der maximal möglichen Verfahrgeschwindigkeit nach Formel B.1 berechnen.

$$\Delta s_{max} = \Delta n * \Delta t_{NC} * v_{max} \tag{B.1}$$

Für die folgende Untersuchung der Algorithmen dient die Referenzkurve Y_{soll} aus Abbildung B.1 als Vorgabe für die Sollbahn. Die Kurve $Y_{ausfall}$ aus Abbil-

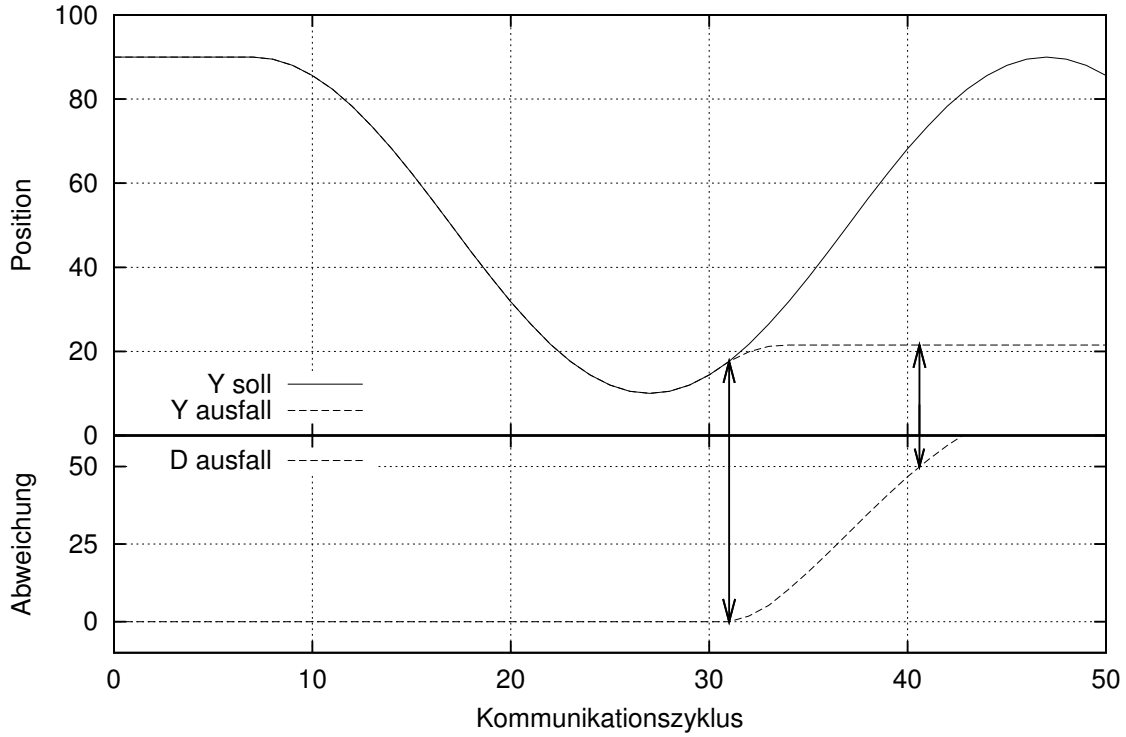


Abbildung B.1.: Sollwertkurve als Referenz für Zeitversatzauswertung

dung B.1 zeigt die Bahn der fehlerhaften Steuerung. Der Ausfall der fehlerhaften Steuerung beginnt im Kommunikationszyklus 32, die Ausgliederung erfolgt mit Kommunikationszyklus 41. Beide Zeitpunkte sind mit Pfeilen markiert.

B.1. First-Seen Methode bei Zeitversatz

Abbildung B.2 zeigt einen möglichen Verlauf mit der First-Seen Methode nach Kapitel 7.2. Die vom Mehrheitsentscheider für die Ausgabe der Sollwerte ausgewählte Steuerung entspricht hier der Steuerung die zur Sollbahn keinen Zeitversatz aufweist, jedoch ab Kommunikationszyklus 32 ausfällt.

Bis zum Ausfall zeigt die First-Seen Methode in diesem Beispiel keine Abweichung von der Sollbahn. Die vom Mehrheitsentscheider ausgegebenen Werte entsprechen der vorgegebenen Bahnkontur. Ab dem Ausfall weichen die ausgegebenen Sollwerte bis zur Toleranzgrenze von der Bahn ab. Nach dem Überschreiten der Toleranzgrenze schaltet der Mehrheitsentscheider auf Steuerung 3

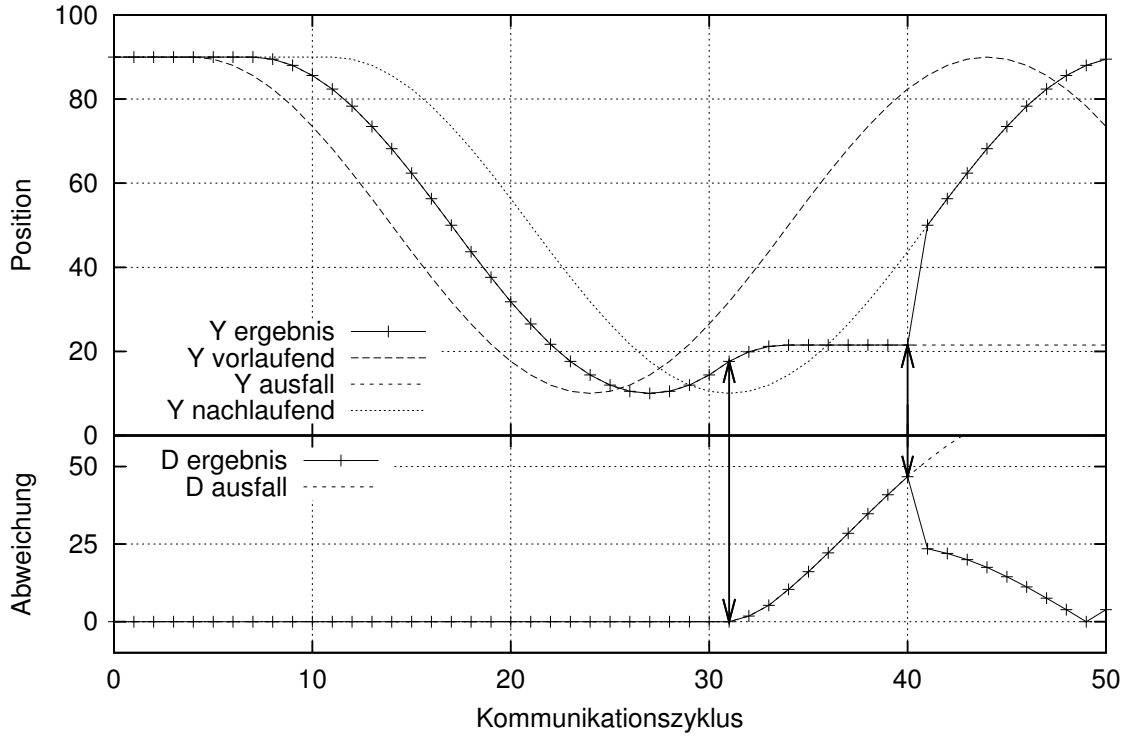


Abbildung B.2.: Ausfall einer von drei zeitversetzt laufenden Steuerungen

um und verursacht damit einen Sprung der ausgegebenen Sollwerte in Höhe der Toleranzgrenze. Danach wird die vorgegebene Bahnkontur mit zeitlicher Abweichung, jedoch in der vorgegebenen Form abgefahren.

B.2. Arithmetischer Mittelwert bei Zeitversatz

Die Verwendung des arithmetischen Mittels für den Mehrheitsentscheider aus Kapitel 7.3 veranschaulicht Abbildung B.3. Hier zeigt sich, dass die ausgegebenen Sollwerte durch die zeitversetzt laufenden Steuerungen bereits vor dem Ausfall von der Sollbahn abweichen. Ebenfalls wird deutlich, dass die Kontur der ausgegebenen Werte nicht mehr der Sollkontur entspricht, was in Kommunikationszyklus 27 sehr gut erkennbar ist. Hier erreicht das Minimum der vom Mehrheitsentscheider ausgegebenen Sollwertkurve nicht den Minimalwert der Sollkontur.

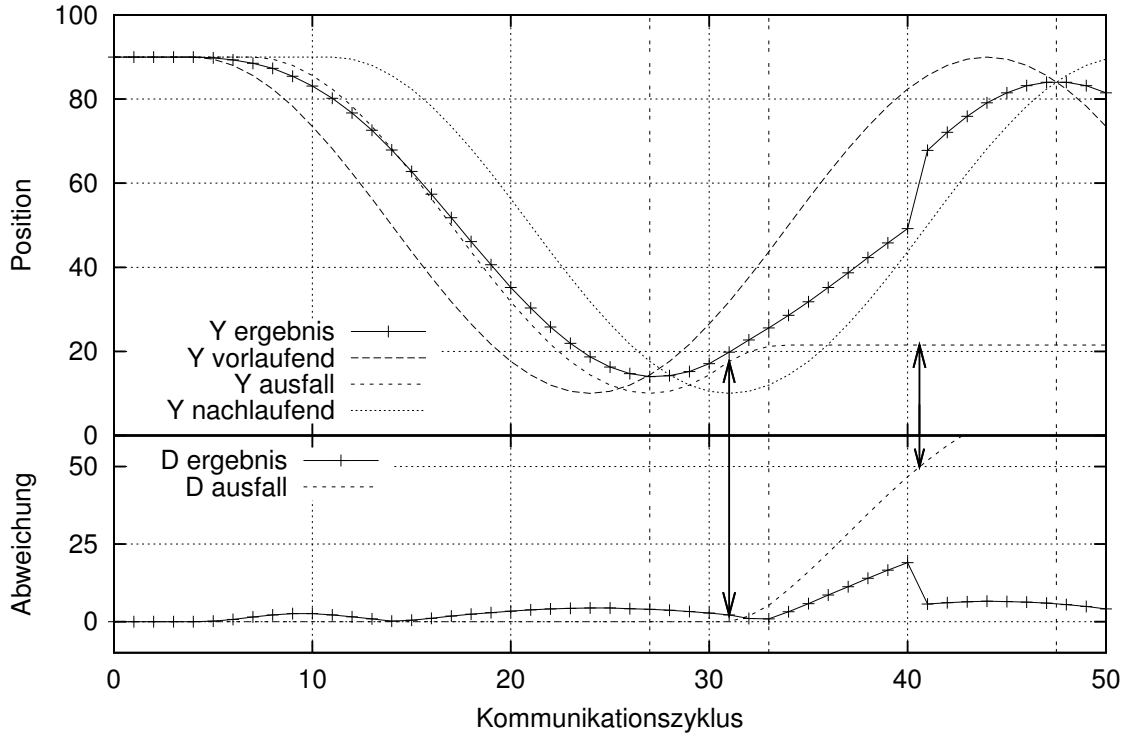


Abbildung B.3.: Ergebnis des arithmetischen Mittels mit Zeitversatz

Ab Kommunikationszyklus 33 weichen die ausgegebenen Sollwerte durch den vorausgehenden Ausfall immer stärker von der Sollbahn ab. Nach Ausgliederung der fehlerhaften Steuerung kommt es zu einem Sprung der ausgegebenen Sollwerte. Danach läuft das System mit den verbleibenden zwei Steuerungen weiter. Die vom Mehrheitsentscheider ausgegebene Kontur entspricht auch hier nicht der Sollkontur, was zu Kommunikationszyklus 47 und 48 gut erkennbar ist.

B.3. Median Methode bei Zeitversatz

Abbildung B.4 zeigt den Verlauf der Sollwerte unter Nutzung der Median Methode aus Kapitel 7.4. Die Sollkontur wird bis zum Umkehrpunkt der Bewegung richtig ausgegeben. Während der Umkehr der Bewegungsrichtung kommt es durch die Kreuzung der Sollbahnen versetzt laufender Steuerungen zu Abweichungen der ausgegebenen Kontur von der Sollkontur.

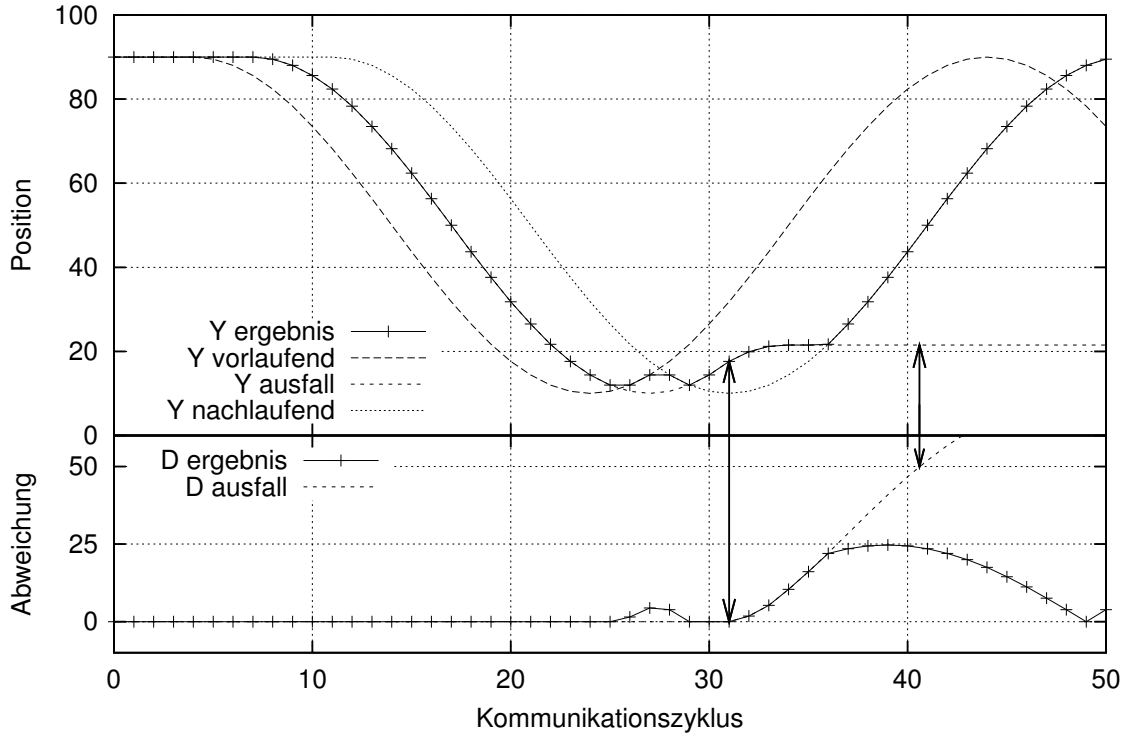


Abbildung B.4.: Ergebnis des Median mit Zeitversatz

Ab dem Ausfall steigt die Abweichung bis zur Kreuzung der Sollwerte der ausgefallenen Steuerung mit den Sollwerten der nachlaufenden Steuerung an. Nach dieser Kreuzung gibt der Mehrheitsentscheider zeitversetzt die korrekte Sollkontur aus.

B.4. Intervallmethode bei Zeitversatz

Die Intervallmethode aus Kapitel 7.5 garantiert, dass die ausgegebenen Sollwerte immer innerhalb eines vorgegebenen Toleranzbereichs liegen. In Abbildung B.5 ist beispielhaft und zur Abgrenzung von der First-Seen Methode ein sehr überzeichnetes Ergebnis dargestellt.

Die Abweichung der Intervallmethode liegt pro ausgegebenem Sollwert zufällig zwischen 0 und der von der aktuellen Geschwindigkeit abhängigen Abweichung nach Formel B.1. Zwischen Ausfall und Ausgliederung der fehlerhaften Steuerung liegt die Abweichung zwischen 0 und der Toleranzgrenze.

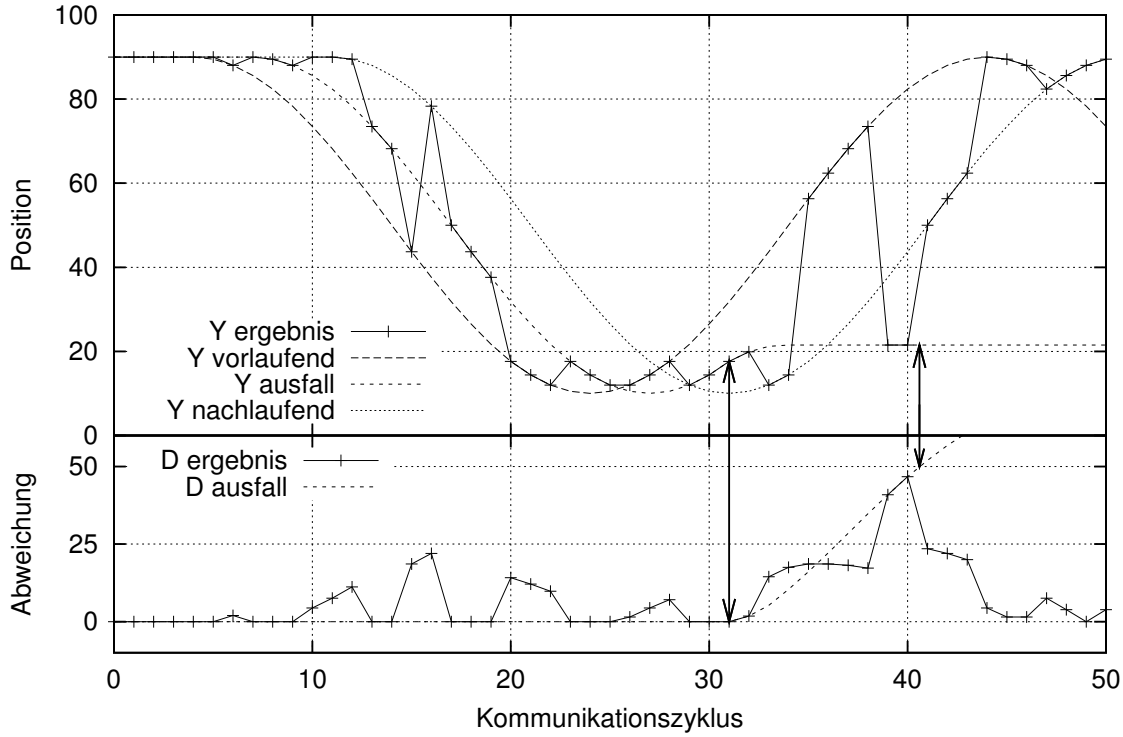


Abbildung B.5.: Mögliches Ergebnis der Intervallentscheidung

Sind Sprünge innerhalb der Toleranzgrenze für einen gegebenen Anwendungsfall nicht zulässig, so ist es sinnvoller, die Intervallmethode in dieser Form nicht einzusetzen. Allenfalls sollte sie so implementiert werden, dass der ausgegebene Sollwert nicht zufällig aus der Menge der Sollwerte innerhalb des Toleranzbereichs ausgewählt wird, um ständige Sprünge zu verhindern.

B.5. Bewertung

Bei synchronisierten jedoch zeitlich versetzt laufenden numerischen Steuerungen sollte vor der Auswahl eines Maskierungsverfahrens zunächst die Frage nach der Ursache des Zeitversatzes stehen. Bezüglich Determinismus kann diese Ursache in zwei Kategorien eingeteilt werden:

1. indeterministischer und veränderlicher Zeitversatz

2. deterministischer und konstanter Zeitversatz

Ein indeterministischer und zur Laufzeit veränderlicher Zeitversatz darf in Echtzeitsystemen, wozu numerische Steuerungen gehören, nicht vorkommen. Tritt er dennoch auf, so liegt ein unentdeckter Fehler in der Synchronisation oder den Steuerungen vor. Dieser darf nicht durch äußere Maßnahmen verdeckt, sondern dessen Ursache muss gefunden und behoben werden.

Der deterministische und konstante Zeitversatz hingegen deutet auf einen Konfigurationsfehler vor der Inbetriebnahme hin. Dieser Fehler kann innerhalb sehr enger Grenzen noch toleriert werden, wenn so wie in Kapitel 8.2.1 die bei konstantem Zeitversatz nach Formel B.1 maximal mögliche Abweichung innerhalb des Toleranzbereichs liegt. Dennoch muss auch hier das Ziel sein, die Ursache zu finden und zu beheben.

Literaturverzeichnis

[Bür05] Bürger, Thomas, 2005. *Durchgängige analytische Qualitätssicherung für numerische Steuerungssysteme.* Heimsheim: Jost-Jetter. ISBN 3-936947-56-2. Stuttgart, Univ., Diss., 2005.

[DIN81] DIN 25424-1:1981-09. *Fehlerbaumanalyse; Methode und Bildzeichen.*

[DIN83] DIN 66025-1:1983-01. *Programmaufbau für numerisch gesteuerte Arbeitsmaschinen; Allgemeines.*

[DIN90] DIN 40041:1990-12. *Zuverlässigkeit; Begriffe.*

[DIN97] DIN EN 954-1:1997-03. *Sicherheit von Maschinen - Sicherheitsbezogene Teile von Steuerungen - Teil 1: Allgemeine Gestaltungsleitsätze.*

[DIN03] DIN EN 61131-3:2003-12. *Speicherprogrammierbare Steuerungen - Teil 3: Programmiersprachen (IEC 61131-3:2003).*

[DIN06] DIN EN 60812:2006-11. *Analysetechniken für die Funktionsfähigkeit von Systemen - Verfahren für die Fehlzustandsart- und -auswirkungsanalyse (FMEA) (IEC 60812:2006).*

[DIN08] DIN EN ISO 13849-1:2008-12. *Sicherheit von Maschinen - Sicherheitsbezogene Teile von Steuerungen - Teil 1: Allgemeine Gestaltungsleitsätze (ISO 13849-1:2006).*

[DIN11a] DIN EN 61508-1:2011-02; VDE 0803-1:2011-02. *Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/ programmierbarer elektronischer Systeme - Teil 1: Allgemeine Anforderungen (IEC 61508-1:2010).*

[DIN11b] DIN EN ISO 12100, 2011-03. *Sicherheit von Maschinen. Allgemeine Gestaltungsleitsätze - Risikobeurteilung und Risikominderung (ISO 12100:2010).*

[DIN13] DIN EN 62061:2013-09; VDE 0113-50:2013-09. *Sicherheit von Maschinen - Funktionale Sicherheit sicherheitsbezogener elektrischer, elektronischer und programmierbarer elektronischer Steuerungssysteme (IEC 62061:2005 + A1:2012).*

[Ech90] Echtle, Klaus, 1990. *Fehlertoleranzverfahren.* Berlin u.a.: Springer. ISBN 978-3-540-52680-3. Karlsruhe, Univ., Habil.-Schr., 1989.

[Erd03] Erdner, Thomas, 2003. *Entwurf eines realzeitfähigen fehlertoleranten Feldbussystems.* Düsseldorf: VDI. ISBN 3-18-372210-0. Hagen, Univ., Diss., 2003.

[Fre87] Frentzen, Bodo, 1987. *Entwurf redundanter Steuerungssysteme, ein Beitrag zur Steigerung der Funktionssicherheit programmierbarer Steuerungen (PLC).* Aachen, Techn. Hochsch., Diss.

[Gar07] Garber, Thomas, 2007. *Nutzung des redundanten Freiheitsgrades von sechsachsigen Parallelkinematik-Maschinen.* Heimsheim: Jost-Jetter. ISBN 3-939890-07-3. Stuttgart, Univ., Diss., 2007.

[Gol91] Goldberg, David, 1991. What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys.* **23**(1), S. 5–48. DOI 10.1145/103162.103163

[Hei00] Hein, Andreas, 2000. *Eine interaktive Robotersteuerung für chirurgische Applikationen.* Düsseldorf: VDI. ISBN 3-18-319517-8. Berlin, Techn. Univ., Diss., 2000.

[hit09] 3sat, 2009. *Antrieb für Giganten - Schiffspropeller.* 9.11.2009. Verfügbar: http://www.3sat.de/page/?source=/hitec/139000/index.html. Zugriff: 23.09.2013

[Hof13] Hoffmann, Dirk W., 2013. *Software-Qualität.* 2., aktual. und korr. Auflage. Berlin u.a.: Springer Vieweg. ISBN 978-3-642-35699-5

[HSA+08] Hauke, Michael, et. al., 2008. *Funktionale Sicherheit von Maschinensteuerungen.* 2., geänderte Auflage. Berlin: Deutsche Gesetzliche Unfallversicherung. ISBN 978-3-88383-771-0. Verfügbar: http://www.dguv.de/medien/ifa/de/pub/rep/pdf/rep07/biar0208/2_2008.pdf. Zugriff: 10.9.2013

[KEB+03a] Korb, Werner, et. al., 2003. Development and first patient trial of a surgical robot for complex trajectory milling. *Computer Aided Surgery.* **8**(5), S. 247–256. DOI 10.3109/10929080309146060

[KEB+03b] Korb, Werner, et. al., 2003. Risk analysis for a reliable and safe surgical robot system. In: Lemke, Heinz, et. al., Hrsg.: *CARS 2003 - computer assisted radiology and surgery: proceedings of the 17th International Congress and Exhibition.* Amsterdam: Elsevier, S. 766-770. ISBN 0-444-51387-6

[Kir11] Kircher, Christian, 2011. *Selbstadaptierende NC-Steuerung für rekonfigurierbare Werkzeugmaschinen.* Heimsheim: Jost-Jetter. ISBN 978-3-939890-79-9. Stuttgart, Univ., Diss., 2011.

[KJG+11] Koller, Oliver, et. al., 2011. Ausfallraten unter Feldbedingungen berechnen. *atp edition*. **53**(10), S. 36-43. ISSN 2190-4111

[KLV09] Krüger, Jörg, Lien, Terje K. und Verl, Alexander, 2009. Cooperation of human and machines in assembly lines. *CIRP Annals - Manufacturing Technology.* **58**(2), S. 628–646. DOI 10.1016/j.cirp.2009.09.009

[Lai05] Laible, Ulrich, 2005. *Aufbau numerischer Steuerungssysteme für sicherheitskritische Anwendungen.* Heimsheim: Jost-Jetter. ISBN 3-936947-48-1. Stuttgart, Univ., Diss., 2004.

[Lal85] Lala, Parag K., 1985. *Fault tolerant and fault testable hardware design.* London: Prentice-Hall. ISBN 978-0-133082-48-7

[Lan05] Lange, Matthias, 2005. Hochleistungsfräsen von Aluminium-Bauteilen für den Flugzeugbau. In: Weinert, Klaus, Hrsg.: *Spanende Fertigung: Prozesse, Innovationen, Werkstoffe.* Essen: Vulkan-Verlag, S. 59–67. ISBN 978-3-802729-35-5

[LBP01] Laible, Ulrich, Bürger, Thomas und Pritschow, Günter, 2001. A Fail-Safe Dual Channel Robot Control for Surgery Applications. In: Voges, Udo, Hrsg.: *Computer Safety, Reliability and Security.* Berlin u.a.: Springer, S. 75–85. DOI 10.1007/3-540-45416-0_8

[LG99] Lauber, Rudolf und Göhner, Peter, 1999. *Automatisierungssysteme und -strukturen, Computer- und Bussysteme für die Anlagen- und Produktautomatisierung, Echtzeitprogrammierung und Echtzeitbetriebssysteme, Zuverlässigkeits- und Sicherheitstechnik.* 3., völlig neubearb. Auflage. Berlin u.a.: Springer. ISBN 3-540-65318-X

[Lig09] Liggesmeyer, Peter, 2009. *Software-Qualität: Testen, Analysieren und Verifizieren von Software.* 2. Auflage. Heidelberg: Spektrum Verlagsges. ISBN 978-3-8274-2056-5

[LL13] Ludewig, Jochen und Lichter, Horst, 2013. *Software Engineering: Grundlagen, Menschen, Prozesse, Techniken.* 3., korr. Auflage. Heidelberg: dpunkt. ISBN 978-3-86490-092-1

[Mir04] MISRA The Motor Industry Software Reliability Association, 2004. *Guidelines for the use of the C language in Critical Systems.* Nuneaton: MIRA. ISBN 978-0-9524156-2-6

[Moo65] Moore, Gordon Earle, 1965. Cramming more components onto integrated circuits. *Electronics International.* **38**(8), S. 114–122. Verfügbar: ftp://download.intel.com/museum/Moores_Law/Articles-press_Releases/Gordon_Moore_1965_Article.pdf. Zugriff: 13.5.2013

[PK05] Pritschow, Günter und Kramer, Christian, 2005. Open System Architecture for Drives. *CIRP Annals - Manufacturing Technology.* **54**(1), S. 375–378. DOI 10.1016/S0007-8506(07)60126-7

[PR04] Pritschow, Günter und Röck, Sascha, 2004. „Hardware in the Loop“ Simulation of Machine Tools. *CIRP Annals - Manufacturing Technology.* **53**(1), S. 295–298. DOI 10.1016/S0007-8506(07)60701-X

[Pri03] Pritschow, Günter, 2003. *Steuerungstechnik II.* Vorlesung Wintersemester 2003. Universität Stuttgart, Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen.

[PSR01] Pritschow, Günter, Staudt, Stefan und Rogers, Gavin, 2001. Intelligente standardisierte Antriebsschnittstellen. In: Neugebauer, Reimund, Hrsg.: *Offensivkonzepte wirtschaftlicher Produktionstechnik.* 1. Auflage. Zwickau: Verl. Wiss. Scripten, S. 307–326. ISBN 3-928921-71-1

[PW97] Pritschow, Günter und Wurst, Karl-Heinz, 1997. Systematic Design of Hexapods and other Parallel Link Systems. *CIRP Annals - Manufacturing Technology.* **46**(1), S. 291–295. DOI 10.1016/S0007-8506(07)60828-2

[PWHB04] Pritschow, Günter, et al., 2004. Ergonomisch gesteuertes Chirurgie-Assistenzsystem. *wt Werkstattstechnik online*. **94**(5), S. 220–225. ISSN 1436-4980

[Rie95] Riegelmayer, Wolfgang, 1995. *Verkabelungskonzepte: Grundlagen und Praxis.* 1. Auflage. Würzburg: Vogel. ISBN 3-8023-1539-1

[Sch92] Schönecker, Wolfgang, 1992. *Integrierte Diagnose in Produktionszellen.* Berlin u.a.: Springer. ISBN 3-540-55375-4. München, Techn. Univ., Diss., 1992.

[Sch94a] Schneider, Jörg, 1994. *Fehlerreaktion mit Speicherprogrammierbaren Steuerungen - ein Beitrag zur Fehlertoleranz.* Berlin u.a.: Springer. ISBN 3-540-58170-7. Stuttgart, Univ., Diss., 1994.

[Sch94b] Schnell, Gerhard, 1994. *Bussysteme in der Automatisierungstechnik.* Braunschweig u.a.: Vieweg. ISBN 3-528-06569-9

[SL10] Spillner, Andreas und Linz, Tilo, 2010. *Basiswissen Softwaretest: Aus- und Weiterbildung zum Certified Tester; Foundation Level nach ISTQB-Standard.* 4., überarb. und aktual. Auflage. Heidelberg: dpunkt. ISBN 978-3-89864-642-0

[SRC06] Alion System Reliability Center, 2006. *Typical Equipment MTBF Values.* Verfügbar: http://src.alionscience.com/pdf/TypicalEquipmentMTBFValues.pdf. Zugriff: 17.2.2012

[Sta06] Staudt, Stefan, 2006. *Spezifikation und Konformitätstest zur Interoperabilität von automatisierten Produktionsmaschinen.* Heimsheim: Jost-Jetter. ISBN 3-936947-88-0. Stuttgart, Univ., Diss., 2006.

[Sti05] Stieler, Wolfgang, 2005. *Moores Gesetz: "Noch zehn bis zwanzig Jahre".* heise online 13.4.2005. Verfügbar: http://heise.de/-152569. Zugriff: 9.9.2013

[VB05] Verl, Alexander und Bock, Hans-Peter, 2005. *Entwicklung und Untersuchung von Verfahren zur Erhöhung der Fehlertoleranz von numerischen Steuerungssystemen in sicherheitskritischen Anwendungen.* Zwischenbericht zu DFG Pr 239/58-1. Universität Stuttgart, Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen.

[VB06] Verl, Alexander und Bock, Hans-Peter, 2006. Fehlertolerante numerische Steuerung. *wt Werkstattstechnik online*. **96**(5), S. 292–296. ISSN 1436-4980

[VB08] Verl, Alexander und Bock, Hans-Peter, 2008. Anforderungen fehlertoleranter numerischer Steuerungen an Kommunikationssysteme. In: Verl, Alexander, Schumacher, Walter und Bender Klaus, Hrsg.: *SPS/IPC/DRIVES 2008.* Berlin u.a.: VDE, S. 53–60. ISBN 978-3-8007-3128-2

[VB09a] Verl, Alexander und Bock, Hans-Peter, 2009. Aufbau eines fehlertoleranten numerischen Steuerungssystems mit Sercos III. *wt Werkstattstechnik online*. **99**(5), S. 331–335. ISSN 1436-4980

[VB09b] Verl, Alexander und Bock, Hans-Peter, 2009. *Entwicklung und Untersuchung von Verfahren zur Erhöhung der Fehlertoleranz von numerischen Steuerungssystemen in sicherheitskritischen Anwendungen.* Abschlussbericht zu DFG Ve 454/1-3. Universität

Stuttgart, Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen.

[VB10] Verl, Alexander und Bock, Hans-Peter, 2010. Failure-tolerant numerical control - Architecture of a failure-tolerant numerical control system. *Production Engineering.* **4**(2-3), S. 287–293. DOI 10.1007/s11740-009-0199-4

[VDE90] DIN V VDE 0801:1990-01. *Grundsätze für Rechner in Systemen mit Sicherheitsaufgaben.*

[Vos88] Vossloh, Martin, 1988. *Modellgestützte Früherkennung und wissensgestützte Diagnose von Fehlern an Werkzeugmaschinen.* München u.a.: Hanser. ISBN 3-446-15251-2. Darmstadt, Univ., Diss., 1988.

[Wag96] Wagner, Michael, 1996. *Steuerungsintegrierte Fehlerbehandlung für maschinennahe Abläufe.* Berlin u.a.: Springer. ISBN 3-540-62656-5. München, Techn. Univ., Diss., 1996.

[WB05] Wörn, Heinz und Brinkschulte, Uwe, 2005. *Echtzeitsysteme: Grundlagen, Funktionsweisen, Anwendungen.* Berlin u.a.: Springer. ISBN 3-540-20588-8

[Wei51] Weibull, Waloddi, 1951. A statistical distribution function of wide applicability. *Journal of applied mechanics.* **18**(3), S. 293–297. Verfügbar: http://web.cecs.pdx.edu/˜cgshirl/Documents/Weibull-ASME-Paper-1951.pdf. Zugriff: 31.8.2014

[WF87] Weck, Manfred und Frentzen, Bodo, 1987. *Entwicklung eines modularen Pressensteuerungssystems: Programmierbare diversitäre 3-von-3 Steuerung.* Bremerhaven: Wirtschaftsverl. NW, Verl. für Neue Wiss. ISBN 3-88314-679-X

Diese Arbeit untersucht Ansätze zur Erhöhung der Zuverlässigkeit von Steuerungssystemen. Dabei wird der fehlersichere Ansatz eines Operationsassistenzsystems um Fehlertoleranz erweitert. Das Ziel ist der Entwurf eines numerischen Steuerungssystems, das auf Basis einer Mehrheitsentscheidung interne Fehler maskieren sowie fehlerhafte redundante Einheiten durch Rekonfiguration ausgliedern kann, ohne fehlerhafte Sollwerte auszugeben.

ISBN 978-3-8396-0890-6

9 783839 608906

FRAUNHOFER VERLAG