Zeitschrift: Bulletin des Schweizerischen Elektrotechnischen Vereins, des Verbandes Schweizerischer Elektrizitätsunternehmen = Bulletin de l'Association suisse des électriciens, de l'Association des entreprises électriques suisses

Herausgeber: Schweizerischer Elektrotechnischer Verein ; Verband Schweizerischer Elektrizitätsunternehmen

Band: 80 (1989)

Heft: 11

Artikel: Fehlertolerante Rechner im Einsatz

Autor: Kirrmann, H.

DOI: https://doi.org/10.5169/seals-903686

Nutzungsbedingungen

Die ETH-Bibliothek ist die Anbieterin der digitalisierten Zeitschriften auf E-Periodica. Sie besitzt keine Urheberrechte an den Zeitschriften und ist nicht verantwortlich für deren Inhalte. Die Rechte liegen in der Regel bei den Herausgebern beziehungsweise den externen Rechteinhabern. Das Veröffentlichen von Bildern in Print- und Online-Publikationen sowie auf Social Media-Kanälen oder Webseiten ist nur mit vorheriger Genehmigung der Rechteinhaber erlaubt. Mehr erfahren

Conditions d'utilisation

L'ETH Library est le fournisseur des revues numérisées. Elle ne détient aucun droit d'auteur sur les revues et n'est pas responsable de leur contenu. En règle générale, les droits sont détenus par les éditeurs ou les détenteurs de droits externes. La reproduction d'images dans des publications imprimées ou en ligne ainsi que sur des canaux de médias sociaux ou des sites web n'est autorisée qu'avec l'accord préalable des détenteurs des droits. En savoir plus

Terms of use

The ETH Library is the provider of the digitised journals. It does not own any copyrights to the journals and is not responsible for their content. The rights usually lie with the publishers or the external rights holders. Publishing images in print and online publications, as well as on social media channels or websites, is only permitted with the prior consent of the rights holders. Find out more

Download PDF: 09.12.2025

ETH-Bibliothek Zürich, E-Periodica, https://www.e-periodica.ch

Fehlertolerante Rechner im Einsatz

H. Kirrmann

Rechner übernehmen zunehmend Schutz- und Leitfunktionen in Industrie, Verkehr und Handel. Fallen sie aus, entstehen Verluste durch Produktionsausfälle oder Unfälle. Zur Erhöhung der Zuverlässigkeit werden fehlertolerante Rechner eingesetzt. Deren Aufwand kann in Grenzen gehalten werden, wenn sie die Toleranz der geleiteten Strecke ausnützen. Der Beitrag beschreibt die entsprechenden Architekturen sowie die Anforderungen der verschiedenen Einsatzgebiete.

Les calculateurs prennent en charge la conduite et la protection de processus de plus en plus complexes. Leur défaillance cause des pertes par manque de production ou par dommages. Quand leur fiabilité devient insuffisante, des calculateurs tolérants aux fautes prennent leur place. Ceux-ci coûtent plus et doivent se rembourser par les pannes et les dommages évités. La tolérance aux fautes devient plus abordable quand on exploite la tolérance du processus aux pannes du calculateur. Cette étude résume les propriétés de processus typiques et les architectures de sûreté de fonctionnement des calculateurs qui en découlent.

Adresse des Autors
Dr. *Hubert Kirrmann*, dipl. El.-Ing. ETH,
Asea Brown Boveri Forschungszentrum,
5405 Dättwil.

Beim Einsatz von Rechnern in Industrie, Verkehr und Handel wird verlangt, dass Verfügbarkeit (kein Ausfall) und Sicherheit (kein Unfall) gegenüber herkömmlichen Lösungen nicht zurückstehen. Verfügbarkeit und Sicherheit können zwar durch eine hohe Qualität der Elemente gesteigert werden; dem sind jedoch wirtschaftliche und technische Grenzen gesetzt. Dort, wo die Zuverlässigkeit nicht ausreicht, wird Redundanz eingesetzt, d.h. ein Mehr an Betriebsmitteln, das ohne Ausfälle nicht notwendig wäre. Sogenannte fehlertolerante Rechner besitzen Redundanz, damit ihre eigenen Ausfälle überbrückt werden können oder damit sie sich bei Ausfällen zumindest definiert verhalten. Diese Rechner sind heute noch wenig verbreitet. Der Grund liegt nicht bei technischen Schwierigkeiten, sondern beim Verhältnis zwischen den Mehrkosten und dem erbrachten Gewinn an Verfügbarkeit oder Sicherheit.

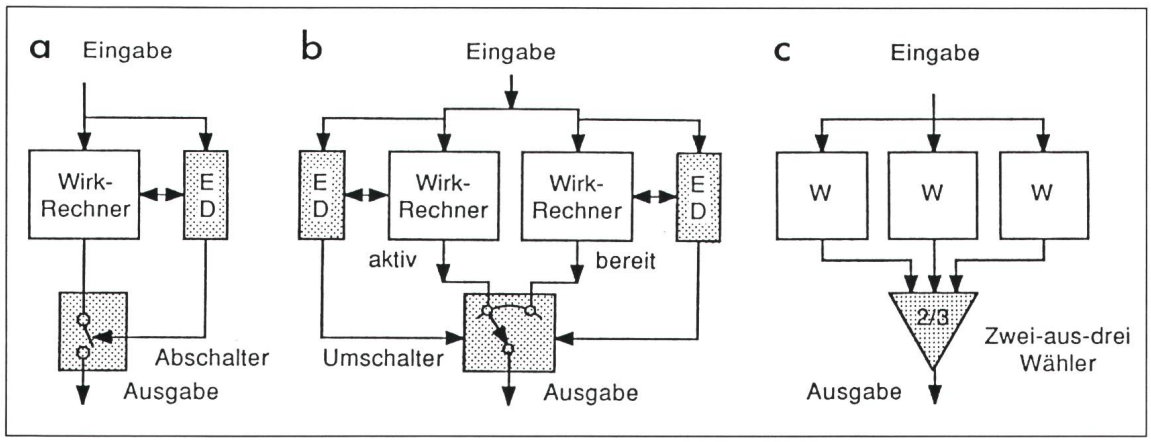
In Anwendungen, die eine hohe Verfügbarkeit verlangen, müssen sich fehlertolerante Rechner auszahlen durch entsprechende Einsparungen an den Produktionsausfallkosten. Ihr Einsatz macht also nur Sinn, wenn die Produktionsausfallzeit gemessen an der Standzeit der Anlage gross ist. In Anwendungen wie Druckereien, wo Ausfälle der Strecke selbst häufig sind, lohnt sich ihr Einsatz kaum. In Anwendungen, die eine hohe Sicherheit verlangen, werden fehlertolerante Rechner meist durch die Betriebsvorschriften aufgezwungen. Dort müssen sie sich bezahlt machen durch die Versicherungsprämien, die sie ersparen. Oft ist die Verwendung eines fehlertoleranten Rechners geradezu Bedingung für die Bewilligung eines Automatisierungsschrittes. Sie erleichtert den Zulassungsbehörden, die für sie etwas undurchschaubare Digitaltechnik anstelle bewährter elektromechanischer oder elektronischer Geräte anzunehmen.

Bei der Entwicklung einer Anlage muss man früh entscheiden, ob die Redundanz der Steigerung der Sicherheit oder der Verfügbarkeit dienen soll. Dazu müssen die Eigenschaften der geleiteten Strecke mitbetrachtet und ausgenützt werden. Darum gibt es keine Struktur, die alle Anforderungen zu decken vermag. In diesem Aufsatz werden die verschiedenen Anforderungen aufgestellt und die Rechnerstrukturen, die sich daraus ableiten, beschrieben.

Funktion des Rechners in einer Anlage

In einer automatischen Anlage soll ein Rechner die *Strecke* leiten. Unter «Strecke» versteht der Automatiker geregelte physikalische Vorgänge wie Antriebe oder Flugzeuge. Dieser Begriff soll hier erweitert werden auf alle physischen Prozesse, die durch einen Rechner beeinflusst werden. Darunter fallen Übermittlungseinrichtungen genauso wie Fernsehstudios oder Banken. Die Gesamtheit der Einrichtungen, die zur Leitung einer Strecke benötigt wird, wird als *Leitanlage* bezeichnet. Darunter fallen die Rechner (von denen es mehrere 100 in einem Kraftwerk geben kann), ihre Ein- und Ausgabeeinrichtungen sowie die Übermittlungswege. Die Trennung zwischen Leitanlage und Strecke ist willkürlich, sie ergibt sich aus den verschiedenen Verantwortungsbereichen der beteiligten Firmen oder Abteilungen. Wir definieren als Leitanlage den Teil des Systems, dessen Verlässlichkeit uns angeht. Zur eigentlichen Strecke gehören im allgemeinen die Leistungsteile (Motoren, Aktoren) sowie der Prozess selbst.

Ob die Rechner die Strecke nur steuern, d.h. Befehle eines Operateurs



Figur 1
Musterarchitekturen zum Erreichen der Stetigkeit, der Integrität und der Maskierung

Prüfredundanz (ED, Error Detection)

2/3 Zwei-aus-drei-Wählschaltung

a Integer: keine falschen Daten

b Stetig: Daten bleiben nicht aus

c Zuverlässig: Fehler werden maskiert

weiterleiten oder regeln, d.h. selbständig aufgrund der gemessenen Prozessvariablen eingreifen, ist für die Zuverlässigkeit wesentlich. Die Funktion der Rechner lässt sich in eine *Leitfunktion,* d.h. die Steuerung des Prozesses zur Aufrechterhaltung der Produktion, und in eine *Schutzfunktion,* d.h. das selbsttätige Einwirken auf den Prozess zur Vermeidung von Schadenfällen, unterteilen. Die Leit- und Schutzfunktion sollen von unterschiedlichen Teilen der Leitanlage wahrgenommen werden, die vorzugsweise von verschiedenen Fabrikanten stammen.

Ausfallarten

Fehler sind unvermeidlich. Man kann lediglich ihre Auswirkungen durch einen geschickten Entwurf mildern. Man unterscheidet dabei *physische* und *systematische* Fehler. Physische oder Hardwarefehler treten als Folge von Alterung, Verschleiss oder äusseren Einwirkungen auf. Darunter fallen z.B. ausgefallene Transistoren. Systematische oder Softwarefehler treten als Folge eines mangelhaften Entwurfs auf. Darunter fallen sowohl Programmierfehler wie mangelhafte Auslegung der Hardware. Während ein behobener Entwurfsfehler nicht mehr auftreten sollte, kann sich ein behobener physischer Fehler jederzeit wieder ereignen. Die Grenze wird fliessend, wenn physische Fehler als Folge von Entwurfsfehlern auftreten, z.B. wenn der Kühlkörper eines Transistors falsch dimensioniert ist und dieser infolge Überhitzung versagt. Wir betrachten hier in erster Linie physische Fehler, denn systematische Fehler lassen sich durch Rechnerarchitekturen kaum beheben.

Fehler können *transient* oder *permanent* sein. Im Gegensatz zu permanenten Fehlern beschädigen transiente Fehler die Hardware nicht, können aber den Betrieb verunmöglichen, wenn sie gespeichert werden. Wir betrachten hier in erster Linie permanente Fehler, da diese den schlimmeren Fall darstellen.

Verhaltensweisen bei Ausfall

Das Verhalten eines Digitalrechners im Fehlerfall ist wegen seiner Komplexität nicht voraussehbar. Dies erschwert im Vergleich zu den klassischen elektromechanischen oder analogen Geräten die Abschätzung der Konsequenzen eines Ausfalles. Trotzdem lassen sich die Ausfälle einer Leitanlage grob auf zwei Arten reduzieren:

1. Der Rechner liefert falsche Daten: Darunter fallen nicht beabsichtigte Daten oder zu Unzeiten gelieferte richtige Daten. Ein solches Verhalten wird als *Integritätsbruch* bezeichnet. Eine Anlage, die - trotz Ausfall eines ihrer Teile - keine falschen Daten liefert, wird als *integer* bezeichnet. Zur Sicherstellung der Integrität kann es notwendig sein, die Anlage - zumindest zeitweise - abzuschalten. Solche Anlagen werden als Fehlstopp-(Fail-Stop-)Anlagen bezeichnet. Der gelegentlich verwendete Ausdruck Fail-Safe ist hier inkorrekt, da eine Abschaltung nicht unbedingt sicher ist.

2. Der Rechner liefert keine Daten: Das Ausbleiben der Leitfunktion wird als *Stetigkeitsbruch* bezeichnet. Es kann sich dabei um ein kurzzeitiges *Aussetzen* handeln, verursacht z.B. durch transiente Fehler. Es kann sich aber auch um eine längerfristige *Panne* handeln, die erst durch menschlichen Eingriff wieder behoben wird. Eine Leitanlage, die trotz Ausfall eines ihrer Teile imstande ist, ihre Funktion ohne menschlichen Eingriff weiterzuführen, ohne dass dies zu einer Panne führt, heisst *stetig*. Ein kurzes Aussetzen ist jedoch oft unvermeidlich, währenddem die Anlage zeitweise falsche Daten übermitteln kann. Solche Systeme werden als *Fail-Operate* bezeichnet oder fehlertolerant (im engeren Sinn).

Beide Ausfallarten können auch kombiniert vorkommen. Demgegenüber gibt es *maskierende* Leitanlagen, die den Ausfall eines ihrer Teile vollkommen vor der Strecke verstecken können. Maskierende Systeme sind sowohl stetig wie integer, sie werden auch als *hochzuverlässig* bezeichnet; nach aussen treten keine Fehler in Erscheinung.

Ausfallbeständige Rechner

Wenn die Auswirkungen eines Leitanlagenausfalls auf die Strecke unerträglich sind, dann wird *Redundanz* zur Erhöhung der Verlässlichkeit eingesetzt. Fehlertoleranz ist die Fähigkeit einer Anlage, sich bei Ausfall eines ihrer Teile definiert zu verhalten, d.h. keine falschen Daten zu produzieren (Integrität) und richtige Daten weiter zu produzieren (Stetigkeit).

Die Musterarchitekturen zum Erreichen von Integrität, Stetigkeit und Maskierung durch Redundanz zeigt die Figur 1. Sie werden weiter hinten näher beschrieben.

Man kann sich fragen, wieso nicht grundsätzlich maskierende Rechner verwendet werden. Dies liegt daran, dass z.B. maskierende 2/3-Rechner mehr als das Sechsfache eines Simplex-Rechners kosten. Integre Rechner kosten das Vierfache und stetige Rechner immerhin das Dreifache eines Simplex-Rechners. Darum gewinnen

technische Lösungen an Interesse, die nicht ideal sind, sondern die Eigenschaften der geleiteten Strecke ausnützen.

Wird Maskierung nicht vorausgesetzt, dann ist zuerst die Frage zu beantworten, wie lange die Strecke einen Integritäts- oder einen Stetigkeitsbruch toleriert. Dies setzt voraus, dass das Verhalten der Strecke bei Ausfall des Rechners bekannt ist. Diese Kenntnis ist ohnehin nötig, denn das Auftreten eines Fehlers im Leitrechner ist keineswegs die einzige Störung, die die Strecke erfährt. Man darf nicht vergessen, dass Fehlertoleranz die Wahrscheinlichkeit des Auftretens eines Fehlers nur verringert, nie aber aufhebt. Selbst maskierende Rechner können nur eine begrenzte Anzahl Fehler überwinden. Fehlertoleranz nützt nur, wenn feststeht, was für Ziele für die Sicherheit und Verfügbarkeit anvisiert werden und welche Eigenschaften der Strecke berücksichtigt werden können.

Einfluss des Ausfalles auf die Strecke

Die Vielfalt an Strecken ist sehr gross. Jede Strecke reagiert anders auf den Ausfall ihrer Leitrechner. Strecken werden in der Regelungstechnik in zwei Hauptklassen eingeteilt, in *kontinuierliche* und *sequentielle*. Kontinuierliche Strecken werden meist geregelt, sequentielle meist gesteuert; die letzteren besitzen also keine Rückführung. Diese Unterscheidung ist für die *Fehlertoleranz* ebenfalls wichtig.

Kontinuierliche Strecken

Kontinuierliche Strecken, wie z.B. geregelte elektrische Antriebe, werden gewöhnlich durch Differentialgleichungen meist im Laplace- oder z-Bereich modelliert. Der Zustandsraum ist kontinuierlich, die Ausgangswerte lassen sich stufenlos verstellen. Solche Strecken sind meist *monoton* , was die Voraussetzung für eine automatische Regelung darstellt. Monoton heisst, dass durch Verstellen der Eingangsgrösse die Strecke in einen beliebigen Zustand gebracht werden kann. So lässt sich z.B. durch Verstellen der Klemmenspannung das Drehmoment eines Elektromotors sowohl erhöhen als auch erniedrigen. Diese Reversibilität gilt jedoch nur innerhalb eines bestimmten Bereiches (Regelbereiches) des Zustandsraumes.

Kontinuierliche Strecken reagieren auf Integritätsbruch (falsche Daten) in gleicher Weise wie auf Störungen. Solange sich diese in einem verträglichen Rahmen halten, können sie ausgeregelt werden. Voraussetzung ist, dass die Regelung funktioniert, dass also der Rechner rechtzeitig wieder zum normalen Betrieb übergeht. Kontinuierliche Strecken tolerieren hingegen kein langes Ausbleiben der Steuerung; sie haben meist keine natürliche Stabilität (sonst würden sie ja keine Regelung brauchen). Das Ausbleiben der Steuerung wird von der Strecke nur während einer begrenzten *Toleranzfrist* geduldet. Ein längeres Aussetzen führt entweder zu einem Schaden oder zu einem Nothalt, der durch die Sicherheitsvorrichtungen ausgelöst wird. Ein Nothalt ist meist verbunden mit einer längeren Panne der Anlage, so dass auch diese Situation zu beträchtlichen Ausfällen führen kann.

Sequentielle Strecken

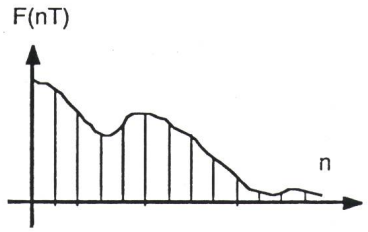
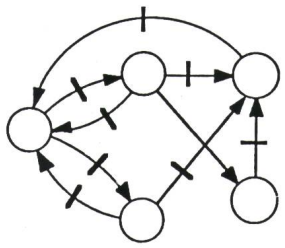
Sequentielle Strecken, wie z.B. Montageroboter oder Aufzüge, werden gewöhnlich durch endliche Automaten, Petri-Netze oder Übergangsmatrizen beschrieben. Die Zustände der Strecke sind endlich und wohl definiert (z.B.: Lift ist im 4. Stock). Der Übergang von einem Zustand zum anderen ist abrupt und meist nichtmonoton, d.h. er kann nicht rückgängig gemacht werden durch Wegnahme des Signales, das den Übergang ausgelöst hat. Zum Beispiel kann ein gerufener Aufzug durch Loslassen des Rufknopfes nicht wieder in das Stockwerk zurückversetzt werden, wo er herkam.

Sequentielle Strecken reagieren deshalb äusserst empfindlich auf falsche Leitbefehle. Eine automatische Kompensation ist ein intelligenter Vorgang, der meist mit erheblichem Aufwand verbunden ist. Es ist z.B. mühsam, eine Fabrikationsstrasse in einen früheren Zustand zurückzuversetzen, wenn einmal falsche Befehle ausgegeben worden sind. Darum ist die Toleranzfrist einer sequentiellen Strecke gegenüber Fehlleitung grundsätzlich Null. Hingegen sind die Schäden bei Ausbleiben von Steuerbefehlen meist auf die Produktionsausfallkosten hin begrenzt. Die tolerierte Aussetzfrist ist relativ lang; danach steigen die Ausfallkosten aber beträchtlich an.

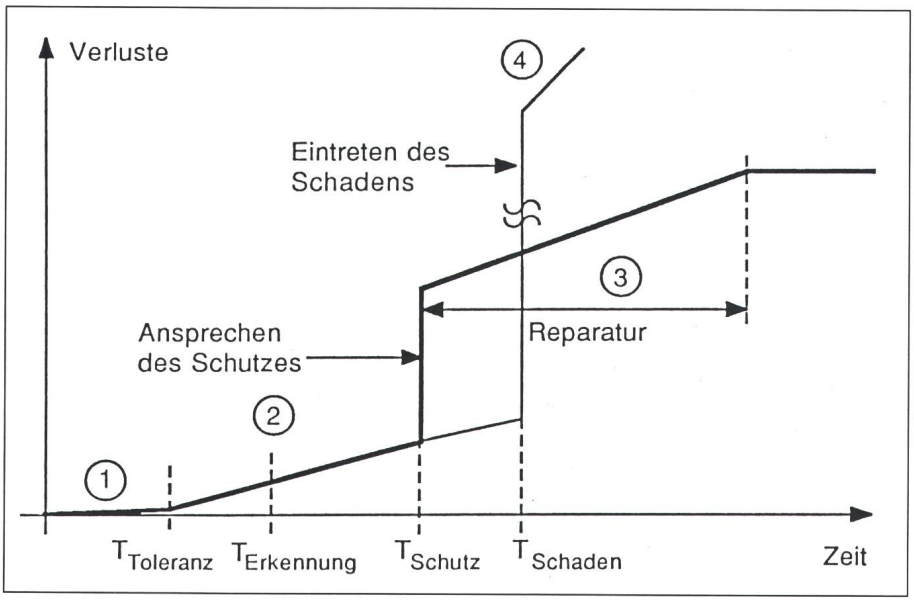
Die Eigenarten der kontinuierlichen und sequentiellen Strecken sind in Figur 2 zusammengefasst.

Sicherheit und Verfügbarkeit

Sicherheit ist die Wahrscheinlichkeit, dass eine Strecke durch Ausfall

Kontinuierliche Strecken	Sequentielle Strecken
Sie werden modelliert durch Differentialgleichungen, Laplace- oder z-Transformation. 	Sie werden modelliert durch endliche Automaten, Petri-Netze. 
Sie dulden kurzzeitige Fehlleitung (kontinuierliche Strecken sind im allgemeinen reversibel). Sie dulden nur kurzzeitig ein Aussetzen der Leitung (die Strecke könnte den reversiblen Zustandsraum verlassen). Sie gewähren kurze Toleranzfrist bei Fehlleitung oder Aussetzen der Leitung.	Sie dulden keine Fehlleitung (Zustandsübergänge sind nicht rückgängig zu machen). Sie dulden relativ langes Aussetzen der Leitung (das Beharren in einem Zustand ist meist ungefährlich). Sie gewähren keine Toleranzfrist bei Fehlleitung, dafür aber lange Aussetzfrist.
Sie verlangen Stetigkeit.	Sie verlangen Integrität.

Figur 2 Kontinuierliche und sequentielle Strecken



Figur 3 Ausfallkosten in Funktion der Ausfallzeit

Während der Toleranzfrist der Strecke (1) wird der Ausfall zunächst nicht bemerkt. In dieser Zeitspanne sollte die Redundanz einspringen. Danach (2) erhöht sich der Ausschuss oder die Produktion setzt aus, bis (bei Strecken mit sicherer Seite) der Schutz anspricht. Wenn die Redundanz den Fehler nicht überbrücken kann, bevor der Schutz anspricht, steigen die Ausfallkosten beträchtlich, denn der Entscheid zum Wiedereinschalten muss durch Menschen nach Klärung der Lage getroffen werden. Falls der Ausfall permanent ist, muss mit einer längeren Reparatur gerechnet werden (3). Schliesslich kann es vorkommen, dass ein Schaden eintritt (4), welcher die Kosten ins Unermessliche treiben kann.

ihrer Leitanlage keine Schäden erleidet, die ein bestimmtes Mass überschreiten. Je nach Strecke kann Sicherheit durch Integrität, durch Stetigkeit oder nur durch Zuverlässigkeit der Leitanlage erbracht werden. Die Vielfalt dieser Massnahmen spiegelt sich in den Vorschriften wider [1]. Ein wichtiges Merkmal einer Strecke ist das Vorhandensein einer *sicheren Seite*, d.h. eines sicheren Zustandes, der sich ohne Wirken der Leitanlage erreichen lässt, z.B. Notstopp bei einem Reaktor oder Öffnen des Schalters bei einer Hochspannungsleitung. Strekken, die keine sichere Seite kennen, werden durch das Ausbleiben der Leitfunktion gefährdet, sie treten nach kurzer Zeit aus dem Regelbereich. Bei solchen Strecken hilft keine Schutzeinrichtung. Darunter fallen Fahrzeuge, die nicht spurgeführt sind, wie Flugzeuge, Raketen oder unstabile chemische Prozesse.

Wie der Ausfall des Rechners die Sicherheit oder die Verfügbarkeit beeinflusst, hängt von seiner Funktion ab, nämlich, ob er eine Leitfunktion oder Schutzfunktion ausübt.

● Wenn ein Rechner eine *Leitfunktion* ausübt, führt sein Ausfall zu einem Produktionsunterbruch, d.h. er vermindert die Verfügbarkeit. Durch falsche Befehle, d.h. durch mangelnde Integrität, ist die Strecke gefährdet. Die vorhandenen Schutzmechanismen sollten diese Situation erkennen und die Strecke zu einem sicheren Zustand bringen. Dies ist nur möglich, solange die Strecke eine *sichere Seite* besitzt.

● Wenn der Rechner eine *Schutzfunktion* ausübt, gefährdet er die Strecke durch Ausbleiben dieser Funktion. Er verliert die Fähigkeit, die Strecke zur sicheren Seite zu führen. In diesem Fall ist es wichtig, den Ausfall des Rechners zu erkennen und zu beheben, bevor der nächste Auslösefall auftritt. Mangelnde Integrität vermindert die Verfügbarkeit (Überfunktion), mangelnde Stetigkeit vermindert die Sicherheit (Unterfunktion).

● Wenn die Strecke keine sichere Seite kennt, dann gefährden sowohl falsche Daten wie das Ausbleiben der Daten die Strecke. Der Rechner hat sowohl eine Leit- wie eine Schutzfunktion. In diesem Fall kommen maskierende Rechner zum Einsatz.

Ein grobes Bild der Ausfallkosten in Funktion der Ausfallszeit gibt die Figur 3.

Erhöhung der Verlässlichkeit durch Redundanz

Zum Erreichen der *Fehlertoleranz* ist Redundanz notwendig. Redundanz umfasst alle Betriebsmittel (Hardware, Software, Zeit), die für die eigentliche Funktion im fehlerfreien Fall nicht benötigt werden. Dabei unterscheidet sich die *Prüfredundanz*, die der Fehlererkennung dient, von der *Wirkredundanz*, die die Funktion eines ausgefallenen Wirkteiles übernehmen kann. In erster Linie dient die Prüfredundanz der Integrität und die Wirkredundanz der Stetigkeit. Ohne Prüfredundanz lässt sich aber Stetigkeit nicht erreichen. Umgekehrt ist es möglich, Integrität durch Wirkredundanz zu erreichen, wie wir sehen werden.

Prüfredundanz

Der Gütefaktor der Prüfredundanz ist der Deckungsgrad *c*, der die Wahrscheinlichkeit ausdrückt, dass ein Fehler innerhalb nützlicher Frist entdeckt wird. Die Prüfredundanz kann aus seriellen (Off-Line-)Prüfprogrammen bestehen, die zwischen Verarbeitungsschritte eingeflochten werden. Sie überprüfen die Richtigkeit der Ergebnisse oder die Funktionsfähigkeit der Hardware (Diagnose). Solche Off-Line-Tests sind aber nicht in der Lage, transiente Fehler zu erkennen; dafür verursachen sie nur geringe Hardwarekosten.

Einen besseren Deckungsgrad erreichen parallele On-Line-Tests, die neben der eigentlichen Datenerzeugung verlaufen. Dazu ist aber zusätzliche Hardware notwendig. Bei regulären Strukturen werden dabei Kodierungsmassnahmen ergriffen wie Paritätsprüfung auf Bussen, Checksumme bei Speicherelementen, zyklische Prüfsummen (CRC) bei seriellen Übertragungen. Eine Anlage, bei der jeder Einzelfehler entdeckt wird, gilt als *selbstprüfend*. Wenn selbst Fehler der Fehlerdetektion nicht unentdeckt bleiben, spricht man von *vollkommener Selbstprüfung*.

Wirkredundanz

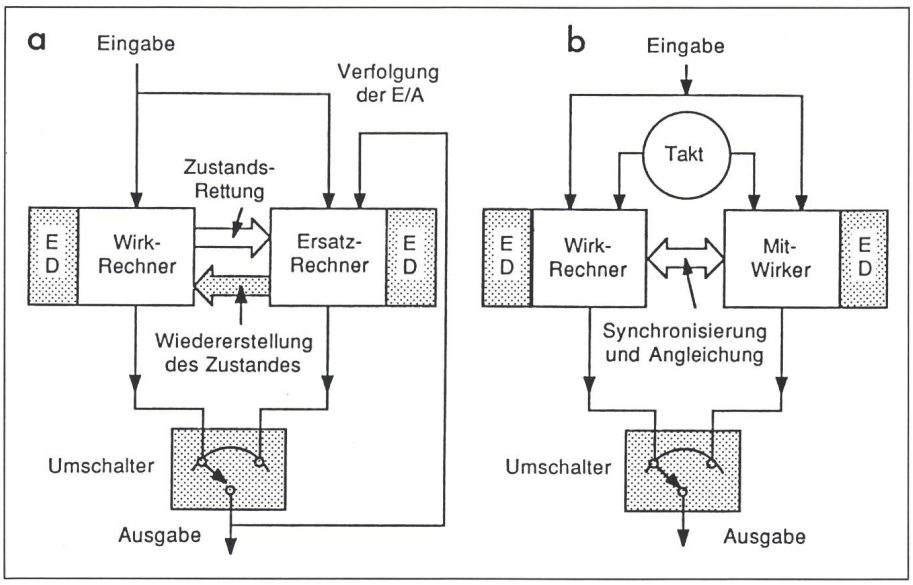
Die Wirkredundanz ist imstande, die Funktion eines ausgefallenen Teiles zu übernehmen. Die Wirkredundanz kann *identisch* sein zum Wirkteil, den es ersetzt, oder *divers*, d.h. von unterschiedlichem Entwurf. Die zweite Massnahme fällt unter Softwareredundanz und wird hier nur am Rande betrachtet. Der Grad der Bereitschaft der Wirkredundanz ist unterschiedlich. Es genügt nicht, eine Ersatz-Hardware bereitzustellen, diese muss auch mit einem möglichst aktuellen Zustand geladen werden, damit die Umschaltung

stossfrei verläuft. Zur Aktualisierung der Wirkredundanz bei digitalen Rechnern werden zwei Verfahren verwendet, Nachführung und Synchronlauf (Fig. 4).

Nachführung: Der Rechner, der an der Strecke angeschlossen ist, der *Wirkrechner,* führt den *Ersatzrechner* in regelmässigen Abständen nach. Zu diesem Zweck werden *Rücksetzpunkte* in die Software eingeführt, die regelmässig dem Ersatzrechner den Stand des Wirkrechners mitteilen. Im Fehlerfall nimmt der Ersatzrechner die Arbeit beim letzten Rücksetzpunkt wieder auf, d.h. die Umschaltung ist nicht ganz nahtlos. Wirk- und Ersatzrechner werden durch die Zustandsrettung belastet, jedoch ist der Ersatzrechner sonst frei für andere Arbeiten. Die Anwendung muss also die Zustandsrettung berücksichtigen. Der einfachste Fall eines nachgeführten Rechners besteht aus einem einfachen Rettungsspeicher. Er wird eingesetzt, wenn nur transiente Fehler erwartet werden. Die Wiederholung nach erkannten Störungen ist ein Sonderfall davon.

Synchronlauf: Bei dieser Lösung führen der Wirkrechner und der Ersatzrechner die gleichen Programme zur gleichen Zeit aus. Da Digitalrechner deterministisch sind, sollten ihre Zustände stets identisch sein. Ihre Ausgaben können zum Zweck der Fehlerentdeckung verglichen werden. Dies wird zur Fehlerentdeckung durch «Verdoppelung und Vergleich» verwendet. Synchronlaufende Rechner können also im Gegensatz zu nachgeführten Rechnern zur Prüfredundanz eingesetzt werden. Der synchronlaufende Ersatzrechner kann nicht für andere Arbeiten verwendet werden, hingegen ist die Umschaltung vom Wirk- zum Ersatzrechner nahezu nahtlos, da die Zustände beider Rechner bis zum Ausfall nahezu identisch sind.

Die Schwierigkeit bei synchronlaufenden Rechnern ist, dass Digitalrechner entgegen der Theorie nicht deterministisch sind: So können z.B. die Speicher unterschiedliche Verzüge haben; die Uhren gehen leicht anders. Marginale Vorgänge, die im Simplexrechner kaum eine Rolle spielen, müssen berücksichtigt werden. Insbesondere müssen die Unterbrechungsanforderungen angeglichen werden. Schliesslich bekommen die redundanten Rechner auch ihre Daten aus redundanten Quellen, die nicht deterministisch sind und ebenfalls angeglichen werden müssen. Die Angleichung ist anwendungsabhängig, sie hängt von der Bedeutung der Eingabesignale ab.



Figur 4 Nachführung und Synchronlauf
ED Error Detection
a Nachführung
b Synchronlauf

Mit Prüfredundanz und Wirkredundanz lassen sich fehlertolerante Strukturen aufbauen. Entsprechend den Ausfallarten werden zwei fehlertolerante Verhalten der Leitanlage angestrebt: Integrität und Stetigkeit.

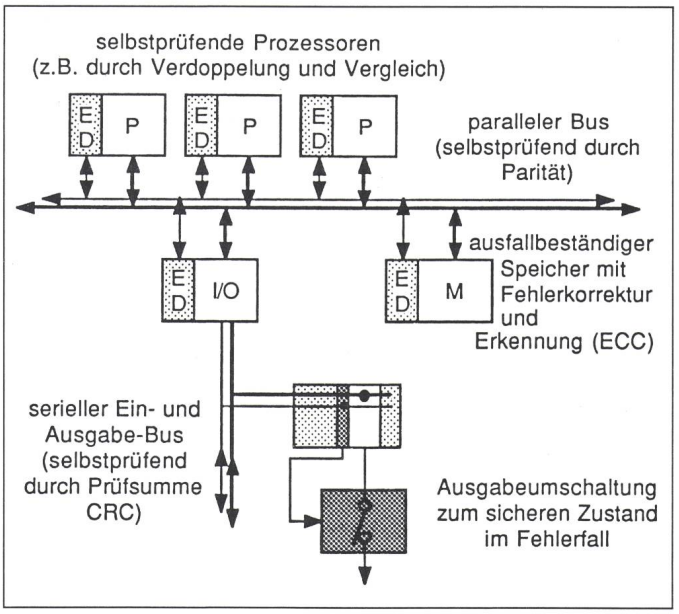
Integres Verhalten

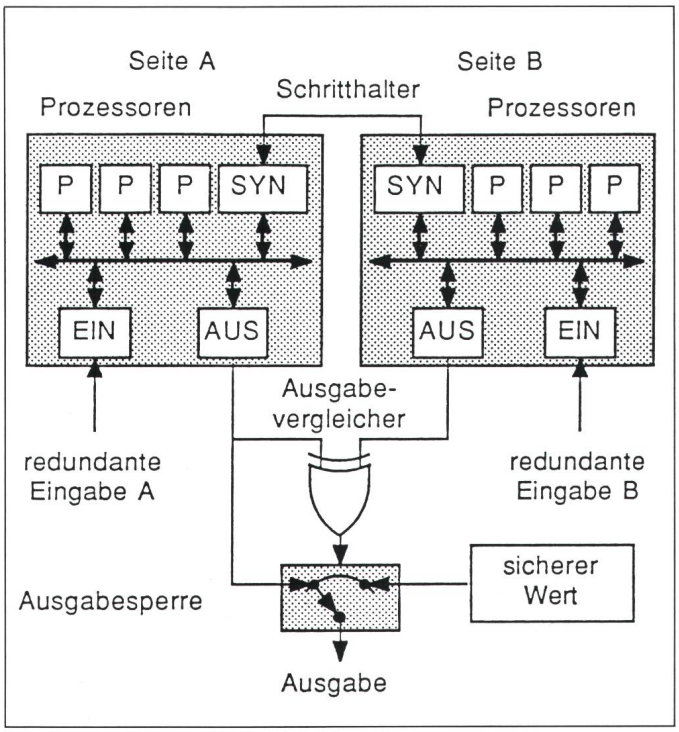
Leitanlagen, die keine falschen Daten liefern, werden als *integer* bezeichnet. Zum Erreichen der Integrität dient eine Prüfredundanz, die beim Auftreten eines Fehlers die Ausgabe abkoppelt. Dies gewährleistet ein Fail-Stop-Verhalten. Der Fehlerdetektor wird möglichst nahe bei der Strecke angesiedelt, ja am besten in die Strecke integriert. Bei Unstimmigkeiten wird die Anlage abgeschaltet. Dies ist allerdings nur sinnvoll, wenn die Strecke eine sichere Seite besitzt.

Selbstprüfung

Es ist im Prinzip möglich, Leitanlagen so aufzubauen, dass jeder einzelne Teil selbstprüfend ist. Die Figur 5 zeigt

Figur 5 Selbstprüfender Rechner
Prüfredundanz
ED Error Detection





Figur 6
Verdoppelung und Vergleich(1/2)
Eins-aus-zwei-Struktur (1/2)
SYN Synchronisierungseinheit

ein Beispiel eines selbstprüfenden Rechners. Auf diesem Bild wurde die Prüfredundanz mit einem Raster gekennzeichnet. Speicher wurden durch ECC abgesichert, Busse durch Parität und Prozessoren durch Verdoppelung und Vergleich. Es werden aber keine integren Rechner auf dem Prinzip der Selbstprüfung aufgebaut. Der Grund liegt darin, dass bei so komplexen Elementen wie Prozessoren und Rechnerkarten gemeinsame Fehlerursachen nicht genügend ausgeschlossen werden können. Lediglich bei seriellen Datenübertragungen kann der Nachweis für einen ausreichenden Dekkungsgrad der Kodierung erbracht werden. Dies ist z.B. in den Normen für Fernwirkprotokolle der IEC festgehalten [2]. Selbstprüfung, auch wenn sie nicht vollkommen ist, nützt immer, sei es auch nur, um die Reparaturzeit zu reduzieren. Darum werden Leitanlagen mit umfangreichen Selbsttests versehen, die bis zu 30% der Software ausmachen können.

Verdoppelung und Vergleich

Wenn hohe Anforderungen an die Integrität der Leitanlage gestellt werden, kommt Verdoppelung und Vergleich zum Zuge (Fig. 6). Es wird dabei eine Wirkredundanz in Synchronlauf zur Fehlererkennung herangezogen. Die Syn-Verbindung übernimmt die Synchronisierung, die Angleichung der Eingaben und vergleicht einige interne Zustände. Der Vergleicher setzt die Ausgabe auf einen vorgegebenen, sicheren Wert bei Nichtübereinstimmung. Diese Anordnung gewährt Integrität solange nicht der gleiche Fehler auf beiden Seiten auftritt.

Das Prinzip kann durch gemeinsame Fehlerursachen überlistet werden. Zu diesen zählt in erster Linie die Software, falls sie auf beiden Seiten identisch ist. Zwar können Prüfprogramme bestimmte Softwarefehler entdecken. Damit lässt sich aber keine hundertprozentige Integrität erreichen, denn der Deckungsgrad der Eigenprüfung der Software ist beschränkt. Man versucht, mit Hilfe diverser Software, d.h. unabhängig voneinander entwickelter Software, wenigstens Programmierfehler auszuschalten. Die Resultate sind bescheiden: Abgesehen davon, dass die Entwicklungskosten sich mehr als verdoppeln, neigen Programmierer zu den gleichen Fehlern, um so mehr, als sie ihre Aufgabe aus den gleichen Spezifikationen ableiten und oft die gleiche Schulbank gedrückt haben. Ausserdem kann nicht sichergestellt werden, dass die Ausgabe identisch ist: sie kann auch ähnlich sein, ohne dass ein Fehler vorliegt. Dadurch erschwert sich der Vergleich erheblich.

Integre Geräte werden meist in sicherheitsrelevanten Anwendungen eingesetzt und unterliegen den TÜV- und VDE-0116-Vorschriften [1]. Beispiele von 2/2 Rechnern sind der H50-SK3 der Firma Paul Hildenbrandt und der Siemens SEL Simis. Letzterer wird in der Eisenbahnsteuerung eingesetzt.

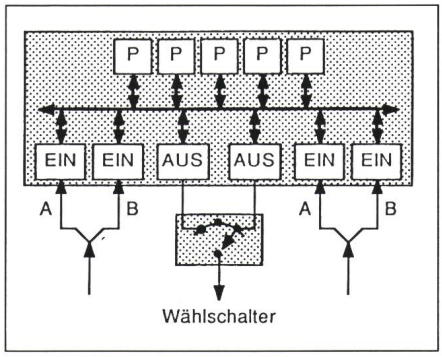
Solche Leitanlagen, die eher stoppen als falsche Daten liefern, zeigen ein Fail-Stop- oder Fehlstopp-Verhalten. Damit eine Anlage selbständig weiterfahren kann, braucht sie überdies Stetigkeit, d.h. Wirkredundanz.

Stetige Anlagen

Anlagen, die ihre Funktion trotz Ausfall eines ihrer internen Elemente fortsetzen können, werden als *stetig, fehlertolerant* oder als *fail-operate* bezeichnet. Zu diesem Zweck ist Wirkredundanz nötig. Dabei kommt es vor, dass falsche Daten während einer bestimmten Zeit herausgegeben werden. Auch ist die Wiederherstellung der Funktion nicht augenblicklich: es bedarf einer mehr oder weniger langen *Aussetzzeit*. Die Aussetzzeit muss kleiner als die Fehlertoleranzzeit der Strecke sein.

Replizierte Ein- und Ausgabe

Obwohl sich das Augenmerk auf ihn richtet, ist der Rechner keineswegs das schwächste Glied in der Regelungskette, die sich von den Fühlern zu den Aktoren erstreckt. Bevor der Rechner verdoppelt wird, sollten empfindlichere Teile redundant ausgelegt werden. Als Faustregel gilt, dass die Zuverlässigkeit von der Peripherie zum Rechner hin zunimmt. Die ausfallträchtigsten Teile stehen im Feld und sind ungünstigen Auswirkungen ausgesetzt wie Temperaturschwankungen, elektromagnetischen Störungen, Lärm und Vibrationen. Die Rechner hingegen befinden sich in der Warte in einer geschützten Umgebung. Darum ist es sinnvoll, Leitanlagen zu bauen, bei denen nur die Ein- und Ausgabeteile verdoppelt oder verdreifacht werden. Andere empfindliche Teile können ebenfalls redundant ausgelegt werden, so z.B. die Stromversorgung oder die Ventilation. In vielen Fällen werden mit diesen Massnahmen die Zuverlässigkeitsziele bereits erreicht und es erübrigt sich eine redundante Auslegung des Rechners. Die Figur 7 zeigt das Prinzip einer teilreplizierten Anlage. Bei dieser kann keine vollkommene Integrität erreicht werden, da Lükken in der Fehlerkennung bestehen und der Rechner selbst eine offensichtliche gemeinsame Fehlerursache darstellt. Eine teilweise Integrität kann durch programmierte Prüfverfahren



Figur 7 Teilreplizierte Anlage

P Prozessoren (Verarbeiter und Kontroller)

EIN, AUS redundante Ein- und Ausgabe

und Verriegelung der Ausgabe erreicht werden.

Eine gewisse Stetigkeit des Rechners selbst kann ebenfalls erreicht werden, allerdings nur zur Behebung transienter Fehler. Zu diesem Zweck wird der wesentliche Teil des Zustandes regelmässig in einen nichtflüchtigen Speicher gerettet (z.B. in einen Kernspeicher oder einen batteriegepufferten Speicher). Eine Wiederanlaufprozedur startet den Rechner erneut mit dem letztgültigen Datensatz. Eine solche Übung lohnt sich aber nur, wenn die Wahrscheinlichkeit eines transienten Fehlers hoch ist.

DCD-Strukturen sind nicht in der Lage, einen zweiten Fehler sicher zu entdecken und zu überwinden. Der Betrieb darf also nach dem ersten Ausfall nicht beliebig lange fortgesetzt werden, da Integrität und Stetigkeit durch die Wahrscheinlichkeit eines zweiten Fehlers vor der Reparatur eingeschränkt werden. In der Praxis werden solche Systeme nur dann verwendet, wenn der Betrieb für kurze Zeit weitergeführt werden darf, z.B. bei einer Seilbahn zum Erreichen der Talstation unter menschlicher Aufsicht.

Diese Struktur ist also weder stetig noch integer gegen einen zweiten Fehler. Ihre Verlässlichkeit wird noch weiter eingeschränkt durch die Wahrscheinlichkeit einer falschen Umschaltung (Fehldiagnose). Insbesondere bei transienten Fehlern besteht die Möglichkeit, dass überhaupt kein Fehler entdeckt wird. In diesem Fall wird willkürlich einer der Rechner als fehlerhaft erklärt.

Auf der Prozessebene werden DCD-Strukturen zur Steuerung und Regelung eingesetzt, so z.B. der Siemens S5-115F, der H 50-H der Firma Paul Hildenbrandt und der Safir des Asea-Brown-Boveri-Forschungszentrums. Auch hier gelten die Sicherheitsrichtlinien nach VDE 0116. Auf der Leitebene findet z.B. das SDR 1300 von Krupp-Atlas Anwendung.

1/2-teilsynchrone Strukturen

Der Vergleich der Resultate erfordert eine umfassende Synchronisierung, die bei komplexen Systemen erhebliche Zeitverluste mit sich bringt. Wenn die Selbstprüfung der Hardware genügend ist, kann ein fehlerhafter Rechner auch ohne Vergleich und Diagnose festgestellt werden. Dadurch entfällt die Notwendigkeit, beide Rechner streng zu synchronisieren, und damit auch der Vergleicher. Da die Leitebene zwar komplex ist, dafür aber eine grosse Toleranzzeit aufweist, wird oft die Synchronisierung gelokkert. Es werden teilsynchrone Rechner verwendet, so z.B. das System Becontrol 40 von ABB [4] oder das Teleperm AS 220 H von Siemens. Die Umschaltung erfolgt aufgrund der Selbstprüfung und spezieller Datenvergleiche. Da der Vergleicher entfällt, lassen sich leichter diverse Softwarepakete einsetzen. Die Flugsteuerung des Airbus benützt eine Redundanz dieser Art: zwei unterschiedliche und von verschiedenen Programmierern in verschiedenen Sprachen programmierte Rechner üben abwechselnd die Kontrolle aus [3].

1/2-Strukturen mit Nachführung

In bestimmten zeitkritischen Anwendungen ist selbst der Aufwand für

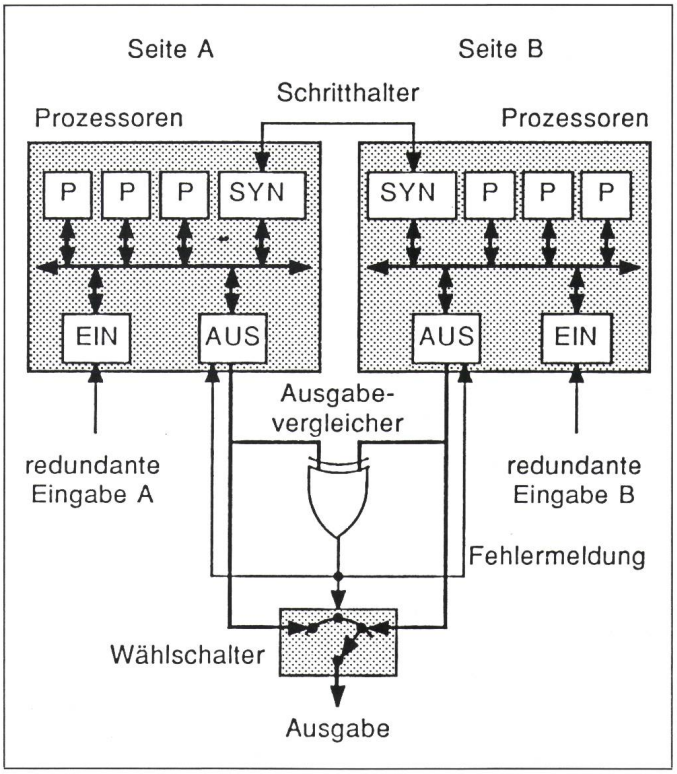
Verdoppelung, Vergleich und Diagnose (DCD)

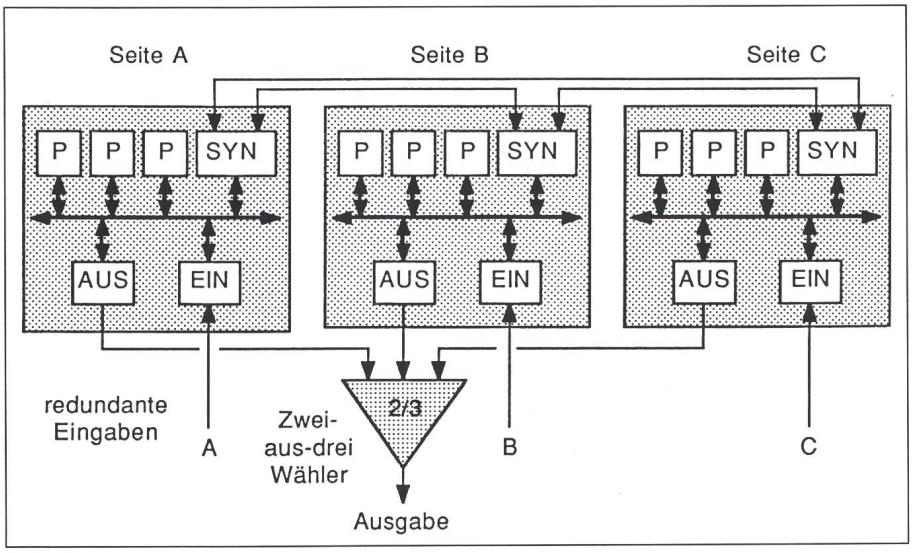
Die DCD (Duplex, Compare and Diagnose)-Struktur ist die billigste Art, zu einem integren, stetigen Rechner zu kommen. Äusserlich ist sie der 2/2-Anordnung (Verdoppelung und Vergleich, Fig. 6) ähnlich. Die DCD-Struktur (Fig. 8) bietet zunächst Integrität gegen einen ersten Fehler. Um Stetigkeit zu erreichen, muss festgestellt werden, welcher Rechner ausgefallen ist.

Der Unterschied zur Verdoppelung-und-Vergleich(2/2)-Struktur in Figur 6 besteht in der Auswertung der Fehlermeldung. Die Fehlerentdeckung startet eine Diagnose. Während der Diagnose ist die Strecke nicht geführt, also muss die Strecke eine ausreichende Toleranzzeit aufweisen. Die Dauer des Aussetzens hängt vom gewünschten Fehlerdeckungsgrad ab. Nach der Diagnose wird der Rechner, der wahrscheinlich ausgefallen ist, abgekoppelt, und der Vergleicher wird abgeschaltet. Der Betrieb kann weitergeführt werden.

Figur 8 Verdoppelung, Vergleich und Diagnose

SYN Synchronisierungseinheit





Figur 9 Zwei-aus-drei-Struktur

eine Teilsynchronisierung zu hoch. Es kommen dann nachgeführte Rechner (Fig. 4) zum Zuge, wobei der Ersatzrechner in regelmässigen Abständen vom Wirkrechner nahegeführt wird. Die resultierende Struktur ist zwar stetig, ihre begrenzte Integrität hängt jedoch voll von der Selbstprüfung ab. Beispiele dafür sind der Siemens S5-150H oder der PHSC von ABB. Es ist zu bemerken, dass kommerzielle fehlertolerante Rechner wie der Tandem nach dem Prinzip der Nachführung arbeiten, insbesondere um die Rechnerleistung des Ersatzrechners auszunützen.

Hochzuverlässige Anlagen

Leitanlagen, die bei Ausfall eines ihrer internen Elemente weder falsche Daten ausgeben, noch ihre Funktion unterbrechen, werden als *fehlermaskierend* oder hochzuverlässig bezeichnet. Die Aussetzzeit einer maskierenden Leitanlage ist grundsätzlich Null.

Zwei-aus-Drei

Die Musterarchitektur für maskierende Rechner ist die 2/3-Anordnung (Fig. 9). Drei Wirkrechner führen die gleiche Aufgabe zur gleichen Zeit aus. Dabei wählt eine Majoritätsschaltung die wahrscheinlichste Ausgabe. Ein fehlerhafter Rechner wird überstimmt. Da die Majoritätsschaltung immer eingeschaltet ist, treten Fehler nach aussen nicht in Erscheinung. Diese Anordnung ist sowohl stetig (mit Aussetzzeit Null) wie integer. Fehlermaskierung ist nur bis zu einer bestimmten Anzahl unabhängiger Fehler möglich. Irgendeinmal ist die Redundanz erschöpft und Fehler können nicht mehr vor der Strecke versteckt werden. Dabei ist noch zu einem gewissen Grad möglich festzulegen, ob sich der Rechner integer oder stetig zu verhalten hat: In der obigen 2/3-Anordnung kann der Umschalter bei einem zweiten Fehler (auf einem anderen Rechner) die Ausgabe noch stoppen, vorausgesetzt, dass der Verursacher des ersten Fehlers ausgeschlossen wurde (also keine Verschwörung zweier fehlerhafter Rechner vorliegt). Die 2/3-Anordnung bleibt bei Zweifachfehlern noch integer (aber nicht mehr funktionsfähig). Die 2/3-Anordnung kann auch bei einem zweiten Fehler stetig bleiben, falls man den fehlerhaften Rechner, z.B. durch eine Diagnose, eruieren kann. In diesem Fall wird der Betrieb nach relativ langem Aussetzen weitergeführt. Dies wird selten gemacht, denn 2/3-Anordnungen werden bei kritischen Strecken eingesetzt, und diese verlangen in erster Linie Integrität. 2/3-Rechner werden für die Kontrolle sicherheitsrelevanter Prozesse verwendet. Beispiele sind der Siemens 220 EHF [4], der für die Brennersteuerung eingesetzt wird.

Zwei-von-Vier

In Kernkraftwerken werden sehr hohe Anforderungen an die Verlässlichkeit der Rechner gestellt. Dort werden 2/4-Anordnungen gebraucht (Fig. 10). Dabei werden vier Rechner synchronisiert; sie führen die gleichen Aufgaben aus. Eine Majoritätsschaltung leitet die wahrscheinlichste Ausgabe weiter. Diese Majoritätsschaltung rekonfiguriert sich bei einem ersten Fehler zu einer 2/3-Anordnung. Eine Verschwörung, d.h. dass zwei oder mehrere Einheiten auf gleiche Art ausfallen, wird zwar oft befürchtet, ist aber unwahrscheinlich.

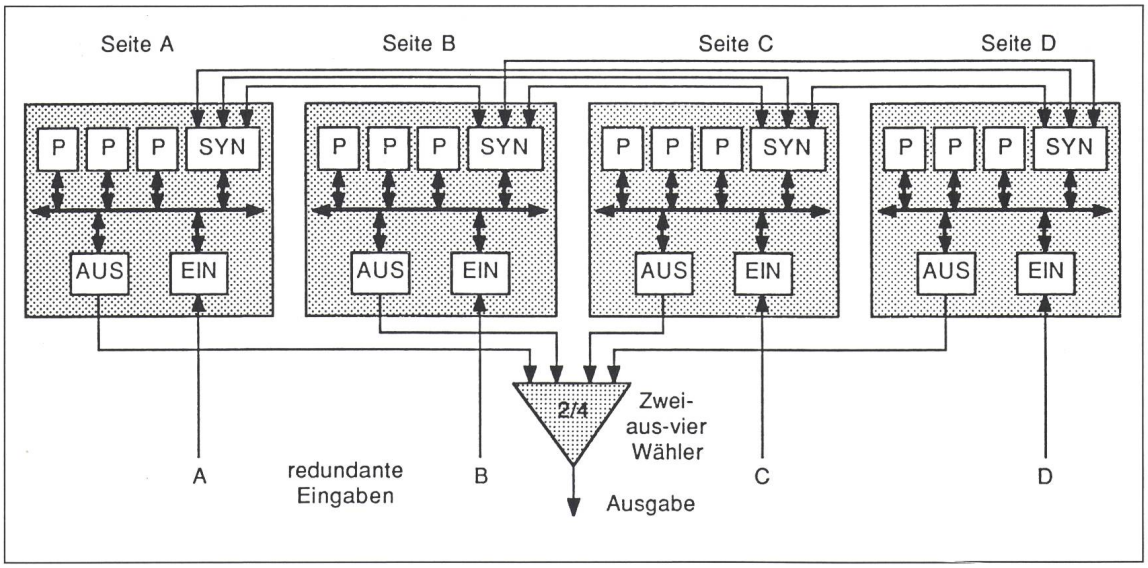
Diese 2/4-Struktur bleibt integer gegen drei Fehler und stetig gegen zwei Fehler. Sie kann auch gegen einen dritten Fehler stetig bleiben, aber dann ist sie nicht mehr integer. In dieser letzten Konfiguration wird sie z.B. im Space Shuttle [5] verwendet.

Höhere Ordnung und Software

Maskierende Strukturen höherer Ordnung können zwar gebaut werden, sind aber weder ökonomisch noch von der Verlässlichkeit her interessant. Eine Vervielfachung der Hardware ist technisch unproblematisch, dagegen ist der Aufbau eines sicheren Vergleichers oder Wählers mit erheblichem Aufwand verbunden. Diese Einheiten aber dürfen nicht wesentlich zur Unzuverlässigkeit des Ganzen beitragen. Man kann zwar diese Einheiten selbst redundant auslegen, aber irgendwo mündet die Redundanz in einer Einkanaligkeit. Am besten wird die Redundanz bis in die Strecke selbst hineingezogen, dies ist aber nicht immer möglich.

Je sicherer die Hardware ist, desto mehr tritt die Unzuverlässigkeit der Software in den Vordergrund, ein Problem, das bis heute keine befriedigende Lösung gefunden hat. Es ist offensichtlich so, dass die Integrität der Software nicht durch Wirkredundanz gesteigert werden kann. Es wird gelegentlich versucht, für kritische Anwendungen wie Flugzeugsteuerungen diverse (*n*-Version-)Programme zu verwenden. Die Resultate sind ernüchternd, da die Auswahl selbst im fehlerfreien Fall ohne Intelligenz des Vergleichers nicht durchführbar ist. Das erwähnte Beispiel des Airbus geht dem Problem des Vergleichers aus dem Weg. Ausserdem ist keineswegs gesichert, dass alle Versionen unabhängig voneinander ausfallen, stammen sie doch aus den gleichen Spezifikationen und von Programmierern, die ähnliche Fehler machen.

Figur 10
Zwei-aus-vier-Struktur



Anhang: Beispiele von Strecken

Banken, Versicherungen, Flug- und Bahnreservation

Rechner werden in der Zentrale eingesetzt. Dezentrale Rechner sind meist nur Terminals. Bankomaten und andere Geldverteiler sind nicht redunant ausgelegt; der Benützer kann auf andere Maschinen zurückgreifen.

Art der Strecke: Sequentiell.

Verfügbarkeit: Gelegentliches Aussetzen des Zentralrechners für eine Dauer kleiner als 3 min ist zulässig.

Integrität: Sehr hoch (insbesondere bei Banken).

Lösung: Duplex-Betrieb zweier Rechner (synchron oder nachgeführt) zur Erhöhung der Verfügbarkeit. Verfälschung der Daten durch Hardwarefehler wird nur auf Übertragungsstrekken überwacht. Terminals sind meist nicht redunant. Bankenterminals und Bankomaten werden durch Plausibilitätstest abgesichert (z.B. durch Rücklesen der ausgegebenen Banknoten, Kreuzvergleiche). Besondere Probleme: absichtliche Fehlereinschleusung.

Beispiel: Tandem, Stratus, DEC-Cluster, IBM.

Elektronische Vermittlung

Verfügbarkeit: Sehr hoch (etwa 3 min Aussetzzeit pro Jahr und 50 000 Linien für Bell).

Integrität: Keine besonderen Anforderungen. Der Kunde ist vom Telefon her gewöhnt, bei Falschverbindung neu zu wählen. Mit der Einführung von E-Mail (Datex, X.400) wird Integrität durch Verschlüsselung erbracht, die Verantwortung dafür wird in die Endgeräte verlegt.

Lösungen: Duplex-Betrieb zweier synchronen Rechner.

Beispiel: Bell 3B20D, Ericcson AXE-10 [2;3]

Leitstellen in Industrie, Kraftwerke

In der Leitebene (Leitstand und Bedienplätze) gelten die gleichen Anforderungen wie bei kommerziellen Anlagen. In vielen Fällen ist die Störanfälligkeit der Strecke selbst so hoch, dass sich der Einsatz von redundanten Rechnern nicht lohnt. Der störanfälligste Teil ist ohnehin die Prozessperipherie (Fühler, Leistungselektronik). Diese Teile werden oft doppelt oder dreifach ausgeführt. Für Kraftwerke und Kernkraftwerke gelten spezielle Vorschriften.

Verfügbarkeit: Meist haben Rechner nur eine Überwachungsfunktion und erhöhen den Bedienungskomfort. In vielen Fällen kann der Betrieb manuell weitergeführt werden. Die erlaubte Aussetzzeit kann deswegen relativ hoch sein (etwa bis zu einer Minute).

Integrität: Da Befehle meist nicht über den Rechner weitergeleitet werden, ist die Anforderung an seine Integrität gering.

Lösung: Duplex-Betrieb zweier Rechner in Bereitschaft oder Synchronlauf.

Beispiel: Doppel-VAX, SDR-1300 (Krupp-Atlas), Tandem (Lagerhaltung).

Netzleittechnik

Elektrizitäts-, Wasser- und andere Netze werden vermascht und redundant ausgelegt. Das Netzleitsystem darf ihre Verfügbarkeit nicht beeinträchtigen.

Verfügbarkeit: 12 min pro Monat Ausfallfrist.

Integrität: Wird in erster Linie von den Übermittlungswegen verlangt. Innerhalb der Kommandostationen wird Integrität durch Rückmelden erreicht. Von den Übermittlungswegen wird eine sehr hohe Integrität verlangt. In den USA vertraut man mehr auf Rückmeldung (Befehl geben, Bestätigen, Entriegeln), in Europa mehr auf die Datensicherung des Übertragungsweges.

Lösung: Duplex-Betrieb mit Synchronlauf, verteiltes System.

Beispiel: Asea Brown Boveri Becontrol 40 Netzleitsystem, Siemens Teleperm.

Prozessnahe Automatisierung

Auf der Feldebene sind die Reaktionszeiten viel kürzer, dafür sind die Datenmengen, die den Zustand beschreiben, viel kleiner.

Art der Strecke: kontinuierlich und sequentiell.

Verfügbarkeit: Die Unzuverlässigkeit des Leitrechners darf zu keiner nennenswerten Herabsetzung der Verfügbarkeit führen.

Integrität: hoch bis sehr hoch bei sequentiellen Strecken, relativ unkritisch bei kontinuierlichen Strecken.

Lösung: bei unkritischen Strecken: Verdoppelung der Ein- und Ausgabe,

wo dies nötig ist. Bei kritischen Strekken: DCD, 2/3.

Schutzsysteme

Je nach Strecke ist Überfunktion (unzweckmässige Abschaltung) oder Unterfuntion (Ausbleiben des Auslösebefehles im Störfall) gefährlicher. Generatorschutz z.B. muss Unterfunktion vermeiden, Sammelschienenschutz muss Überfuntion vermeiden.

Art der Strecke: sequentiell (Arbeit/Abschalten).

Verfügbarkeit: Unterfunktion gefährdet die Strecke.

Integrität: Überfunktion erniedrigt die Verfügbarkeit.

Lösung: Verwendung mehrerer, unabhängiger Schützen, deren Auslöseausgänge ODER-verknüpft sind (gegen Unterfunktion), oder UND-verknüpft sind (gegen Überfunktion).

Eisenbahn-Signalisierung

Dieses Gebiet ist sehr stark reglementiert, es kommen praktisch nur Sonderlösungen zum Einsatz, die eine sehr hohe Qualität von Anfang an ausweisen.

Integrität: Sehr hoch.

Lösung: Verdoppelung und Vergleich, besondere Auslegung.

Beispiel: Siemens/SEL Simis [6]

Fahrzeugelektronik

Da die Leitfunktion von der Schutzfunktion getrennt ist, wird das Gewicht bei der Fahrzeugelektronik auf die Verfügbarkeit gelegt.

Verfügbarkeit: Sehr hoch. Bei der Bahn z.B. 1 Totalausfall pro 700 000 km, was einem Ausfall pro Monat für das gesamte SBB-Netz entspricht.

Lösung: Ausnützung der Redundanz des Fahrzeuges (unabhängige Drehgestelle), Duplex-Rechner, regelmässige Wartung.

Beispiel: ABB-Zugselektronik.

Führerlose Fahrzeuge

Man unterscheidet spurgeführte Fahrzeuge (Bahn, Seilbahn) und aktiv geführte Fahrzeuge (durch Induktionsschleifen oder Mikrowellen geführte Fahrzeuge). Im ersten Fall hat die Strecke eine sichere Seite (Notbremsung). In erster Linie ist Integrität notwendig, damit die vorgeschriebene Geschwindigkeit nicht überschritten wird. In zweiter Linie wird Verfügbarkeit verlangt, um die Bereitschaft trotz höherer Störanfälligkeit der integren Rechner zu erhöhen. Bei aktiv geführten Fahrzeugen wird in erster Linie Stetigkeit verlangt.

Verfügbarkeit: weniger als 1 Ausfall auf 1000 km Netz pro Tag.

Integrität: Das Fahrzeug darf nicht in einen gefährlichen Zustand wegen Fehlleitung gelangen.

Lösung: 2-aus-3-Anordung.

Beispiel: Bart (San Francisco), VAL (Lilles), Elektronisch gelenkter Bus, Brown Boveri LZB [7].

Flugzeuge

Von der Flugzeugelektronik wird in erster Linie Verfügbarkeit verlangt. Eine Ausnahme bildet die Elektronik für die automatische Landung, die während der Landephase eine sehr hohe Zuverlässigkeit aufweisen soll. Diese ist nur durch Maskierung zu erreichen. Die gleichen Vorschriften gelten für den Autopiloten tieffliegender Kampfflugzeuge. Natürlich unstabile Flugzeuge werden von der Nasa untersucht. Ihr Regelungssystem soll die gleiche Zuverlässigkeit wie ein Flügel aufweisen, denn der Verlust der Regelung führt zum Verlust des Flugzeuges.

Verfügbarkeit: Die Rechner sollen die Verfügbarkeit des Flugzeuges nicht wesentlich senken.

Integrität: Toleranzzeit etwa eine Sekunde.

Zuverlässigkeit: 10^{-10} pro Flugstunden.

Lösung: 2-aus-3-Anordnung mit zusätzlichen Ersatzrechnern und Rekonfiguration.

Beispiel: Sift [8], FTMP [9], AFTC [10].

Literatur

[1] *H. Hölscher* und *J. Rader:* Mikrocomputer in der Sicherheitstechnik. TÜV-Verlag Rheinland, 1984.

[2] IEC, Tec. Com. No. 57, Telecontrol, teleprotection and associated telecommunications for electric power systems, draft. December 1987.

[3] *J.C. Rouquet* und *P. Traverse:* Safe and reliable computing aboard the airbus and ATR aircrafts. Proc. 5th Int. Workshop on Safety of Computer Control Systems, Safecomp 86, Sarlat, France, p. 93...97.

[4] *M. Euringer, W. Reichert & T. Schlierf:* Hochverfügbare und fehlersichere Prozessautomatisierung mit redundanten Systemen. Siemens Energie & Automation 8 (1986)5, S. 334...336.

[5] *A. Spector* and *D. Gifford:* The space shuttle primary computer system. Comm. ACM, Sept. 1984, p. 874...900.

[6] *J. Wallace* and *W. Barnes:* Designing for ultra-high availability: The Unix RTR operating system. IEEE Comp. 17(1984)8, p. 31...39.

[7] *B. Ossfeldt* and *I. Jonsson:* Recovery and diagnostics in the central control of the AXE switching system, IEEE Trans. Comp. June 1980, p. 482...491.

[8] *D. Goudie, M. Davis* und *A. Spatz:* Die BBC-Netzleitsystem-Familie Becontrol. Brown Boveri Mitteilungen 71(1984) Sept./Okt., S. 406...415.

[9] *E. Schrodi:* Die vollkommene digitale Redundanz – AS 220 H. Interkama-Seminar-Referat 3.8, 1983.

[10] *H. Strelow* und *H. Üebel:* Das sichere Mikrokomputer Simis. Signal und Draht, 1978, S. 82...86.

[11] *A. Schmidt, R. Faller* und *W. Pöttig:* Elektronisch gelenkter Bus. Elektronik, 1984, S. 41...46.

[12] *J. Goldberg* et al. Development and analysis of the SIFT computer. Nasa contractor report 172146, SRI International, Feb. 1984.

[13] *A. Hopkins, T.B. Smith* and *J. Lala:* FTMP – a highly reliable fault-tolerant multiprocessor for aircraft. Proc. IEEE, special issue on fault-tolerant computing, Oct. 1978, p. 1221...1239.

[14] *H. Schmid, J. Lam, R. Naro, K. Weir:* Critical issues in the design of a reconfigurable control system. FTCS-14, Orlando, Florida, June 1984.