Fehlertolerante Steuerungs- und Regelungssysteme

Fault-tolerant Control Systems

Hubert Kirrmann und Karl-Erwin Großpietsch

Diese Arbeit gibt einen Überblick über Methoden der Fehlertoleranz im Bereich industrieller Leit-, Schutz-, Steuerungs- und Regelungssysteme. Rechner übernehmen zunehmend Schutz- und Leitfunktionen in Industrie, Verkehr und Handel. Rechnerausfälle verursachen wirtschaftliche Verluste, sei es durch Ausfälle oder durch Unfälle. Wenn die Zuverlässigkeit herkömmlicher Rechner nicht ausreicht, werden fehlertolerante Rechner eingesetzt, die dafür teurer sind. Sie müssen sich bezahlt machen durch die Produktionsausfallkosten bzw. Versicherungsprämien, die sie ersparen. Fehlertolerante Rechner werden billiger und einfacher, wenn sie die Toleranz der gesteuerten Strecke ausnützen. Je nach Art der Strecke und Auswirkung des Ausfalles kommen drei Klassen von fehlertoleranten Rechnern zum Einsatz: integre Rechner („fail-stop"), stetige („fail-operate"), und fehlermaskierende (Ausfall von außen nicht sichtbar). Diese Architekturen unterscheiden sich insbesondere durch die Art, wie Fehler erkannt werden und wie die redundanten Komponenten auf dem aktuellen Stand gehalten werden, sei es durch Nachführung oder durch Synchronlauf.

In this contribution, we present a survey on fault tolerance methods in the area of logic controllers. More and more, computers take over protective and control functions in industry, traffic and commerce. Computer failures cause financial losses, either due to resulting computer downtimes or due to resulting accidents. If the reliability of conventional computers is not sufficient enough, fault-tolerant computers are used which on the other side are more expensive. They pay off by the production downtimes or assurance fees, which they make avoidable. Fault-tolerant computers can be realised at lower costs if they exploit the fault tolerance of the controlled plant. According to the individual system line and the effects of a failure, three different classes of fault-tolerant computers are used: integer (fail-stop), persistent, or fault-masking (the fault is not noticeable outside the system) computers. These architectures especially differ in how failures are detected and how the states of the redundant components are held consistent, either by explicit update actions or by synchronous operation.

1 Einführung

Dieser Überblick über Methoden der Fehlertoleranz im Bereich industrieller Leit-, Schutz-, Steuerungs- und Regelungssysteme (hier meist abkürzend als ,,Steuerungen" bezeichnet) geht davon aus, dass beim Einsatz von Rechnern in Industrie, Verkehr oder Handel sowohl Verfügbarkeit (kein Ausfall der Strecke) als auch Sicherheit (kein Unfall) gegenüber herkömmlichen Lösungen nicht zurückstehen dürfen. Verfügbarkeit und Sicherheit können zwar durch eine hohe Zuverlässigkeit der Elemente gesteigert werden; dem sind jedoch wirtschaftliche und technische Grenzen gesetzt. Dort, wo die Zuverlässigkeit nicht ausreicht, wird Redundanz eingesetzt, d. h. Betriebsmittel, die ohne Berücksichtigung von Ausfällen nicht notwendig wären.

Fehlertolerante Rechner besitzen Redundanz, um ihre eigenen Ausfälle überbrücken zu können oder um sich bei Ausfällen zumindest definiert zu verhalten. Diese Rechner sind heute noch wenig verbreitet. Dies liegt weniger an der technischen Machbarkeit als am Verhältnis der Kosten dieser Rechner zum erbrachten Gewinn an Verfügbarkeit oder Sicherheit.

Redundanz verursacht nicht nur höhere Anschaffungskosten, sondern auch höhere Unterhaltskosten – eine verdoppelte Steuerung fällt zunächst zwei mal häufiger aus als

eine einfache Steuerung. Die Redundanz lässt auch die Verlässlichkeit nicht ins Unendliche steigen, sie ist auch kein Ersatz für Qualität.

In Anwendungen, die eine hohe Verfügbarkeit verlangen, müssen sich fehlertolerante Rechner auszahlen durch die Produktionsausfallkosten, die sie ersparen. In Anwendungen, die eine hohe Sicherheit erfordern, werden fehlertolerante Rechner meist durch die Betriebsvorschriften aufgezwungen. Dort müssen sie sich bezahlt machen durch die Versicherungsprämien, die eingespart werden.

Oft ist die Verwendung eines fehlertoleranten Rechners geradezu Bedingung für die Bewilligung eines Automatisierungsschrittes. Dies erleichtert es den Zulassungsbehörden, die undurchschaubare Digitaltechnik anstelle altbewährter elektromechanischer oder elektronischer Geräte anzunehmen.

Aus diesem Grund wächst der Markt für sicherheitsorientierte Rechner etwas schneller als die übrige Branche. Sie fungieren oft als Türöffner für neue Gebiete und erlauben es, Betriebserfahrung zu sammeln, die später weniger aufwändigen Lösungen zugute kommt. In der Luftfahrt z. B. wurden zweimotorige Passagierflugzeuge für die Atlantiküberquerung erst erlaubt, als genügend Erfahrungsdaten mit drei- oder viermotorigen vorlagen. Eine Typenprüfung soll nicht nur Zahlen liefern, sondern Vertrauen in diese schaffen.

Man muss früh entscheiden, ob die Redundanz zur Steigerung der Sicherheit oder der Verfügbarkeit dienen soll. Dazu müssen die Eigenschaften der gesteuerten Strecke ausgenützt werden. Deshalb gibt es keine Struktur, die alle Anforderungen abzudecken vermag. In diesem Aufsatz, der eine Aktualisierung einer früheren Arbeit [1] darstellt, werden die verschiedenen Anforderungen diskutiert und die wichtigsten Rechnerstrukturen, die sich daraus ableiten,

1.1 Funktion des Steuerungs- und Regelungssystems in einer Anlage

Unter „Strecke“ versteht der Automationstechniker geregelte physikalische Vorgänge wie Antriebe oder Flugzeuge. Dieser Begriff soll hier erweitert werden auf alle physikalischen Prozesse, die durch einen Rechner beeinflusst werden. Die Gesamtheit der Einrichtungen, die zur Leitung einer Strecke benötigt werden, bezeichnet DIN als „Leitanlage“. Darunter fallen die Steuerungs- und Regelungssysteme (nachfolgend in dieser Arbeit stets abkürzend als „Steuerung“ bezeichnet), ihre Ein- und Ausgabeeinrichtungen sowie die Übermittlungswege. Die Trennung zwischen Leitanlage und Strecke ist willkürlich, sie ergibt sich vielmehr durch die verschiedenen Verantwortungsbereiche der beteiligten Firmen oder Abteilungen. Wir betrachten 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 steuern, d. h. Befehle eines Operateurs weiterleiten oder regeln, d. h. selbständig aufgrund der gemessenen Prozessvariablen eingreifen, wird für die Zuverlässigkeit wesentlich. Auch ist die Funktion der Rechner wichtig: es sind entweder:

- Leitfunktionen (d. h. Beobachtung und Beeinflussung des Produktionsprozesses) oder
- Schutzfunktionen (d. h. selbsttätiges Einwirken auf den Prozess zur Vermeidung von Schadensfällen).

1.2 Ausfälle der Leitanlage

1.2.1 Ausfallarten

Fehler sind unvermeidlich. Man kann lediglich ihre Auswirkungen durch geschickten Entwurf mildern. Man unterscheidet physische Fehler und systematische Fehler. Physische oder Hardwarefehler treten als Folge von Alterung, Verschleiß oder äußeren Einwirkungen auf. Darunter fallen z. B. ausgefallene Transistoren. Systematische Fehler treten als Folge eines mangelhaften Entwurfs auf. Hierzu gehören sowohl Programmierfehler wie mangelhafte Auslegung der Hardware. Als Unterscheidungsmerkmal gilt, dass ein behobener Entwurfsfehler nicht mehr auftritt, während ein behobener physischer Fehler sich jederzeit wieder ereignen kann. Die Grenze wird fließend, wenn physische Fehler als Folge von Entwurfsfehlern auftreten, z. B. wenn der Kühlkörper eines Transistors falsch dimensioniert ist und die Transistoren infolge Überhitzung versagen. Wir betrachten hier in erster Linie physische Fehler, denn systematische Fehler lassen sich durch Fehlertoleranz kaum beheben.

Fehler können transient oder permanent sein. Im Gegensatz zu permanenten Fehlern beschädigen transiente Fehler die Hardware nicht, können aber den Betrieb unmöglich machen, wenn sie zu fehlerhaften gespeicherten Daten führen.

1.2.2 Auswirkungen des Ausfalles

Trotz der Vielfalt der Ausfallserscheinungen lassen sich die Ausfälle einer Leitanlage auf zwei Arten reduzieren:

1. Die Leitanlage liefert **falsche Daten**: Darunter fallen nicht beabsichtigte Daten oder zu Unzeiten gelieferte, anscheinend richtige Daten. Ein solches Verhalten wird auch **Integritätsbruch** [1] genannt. Eine Anlage, die trotz Ausfall eines ihrer Teile keine falschen Daten liefert, bezeichnen wir als **integer** [1]. Dabei kann es notwendig sein, die Anlage abzuschalten, zumindest zeitweise. Solche Anlagen werden auch als Fehlstopp-Systeme („fail-stop“, „fail-silent“) bezeichnet. Der gelegentlich verwendete Ausdruck „fail-safe“ ist hier inkorrekt, da eine Abschaltung nicht unbedingt sicher ist.
   Führt die Steuerung hingegen eine Schutzfunktion aus, lösen falsche Daten womöglich eine unnötige Fehlfunktion (Nothalt) aus, die Stillstandkosten verursacht. Hier ist die Anlage sicher, aber still.
2. Die Leitanlage liefert **keine Daten** oder diese zu spät: Das Ausbleiben der Leitfunktion wird als **Stetigkeits-**

bruch [1] bezeichnet. Es kann sich dabei um ein kurzzeitiges Aussetzen handeln, verursacht z. B. durch transiente Fehler. Es kann aber auch eine längerfristige Panne sein, die erst durch menschlichen Eingriff wieder behoben wird. Eine Leitanlage, die trotz Ausfall eines ihrer Teile imstande ist, ihre Funktion automatisch weiterzuführen, ohne dass es zu einer Panne führt, bezeichnen wir als **stetig** (englisch: persistent). Ein kurzes Aussetzen ist oft unvermeidlich, während dem zeitweise falsche Daten entweichen können. Solche Systeme werden auch als „fail-operate“ bezeichnet.

Führt die Steuerung eine Schutzfunktion aus, kann sie nicht mehr auf Störsituationen reagieren. Hier ist die Strecke gefährdet, aber noch funktionsfähig.

Außerdem können beide Ausfallarten kombiniert vorkommen. Dafür gibt es maskierende Leitanlagen, die den Ausfall eines ihrer Teile vollkommen vor der Strecke verbergen können, d. h. nach außen treten keine Fehler in Erscheinung. Der damit verbundene hohe Aufwand macht sich nur bei besonders heiklen Strecken bezahlt.

Bei Strecken, die ununterbrochen produzieren (z. B. Kraftwerke), ist der heikle Moment bei einer fehlertoleranten Steuerung übrigens nicht der Ausfall selbst, sondern die Herausnahme und Wiedereingliederung der reparierten Teile in eine produzierende Anlage („live insertion“, „hot-swap“). Wenn durch diese Operation die Strecke – auch planmäßig – angehalten werden muss, geht ein Teil des Wertes der Redundanz verloren.

1.3 Ausfallbeständige Steuerungen

Wenn die Auswirkung eines Ausfalles der Leitanlage auf die Strecke nicht hingenommen werden kann, wird Fehlertoleranz benötigt. Fehlertoleranz ist die Fähigkeit einer Anlage, sich bei Ausfall eines ihrer Teile definiert zu verhalten, d. h. keine falschen Daten zu produzieren, richtige Daten weiter zu produzieren oder beides zu bewerkstelligen. Bild 1 zeigt die Musterarchitekturen zum Erreichen von Integrität, Stetigkeit und Maskierung durch Redundanz, die IEC 61508 unterscheidet:

- Integres Verhalten wird entweder durch lokale Fehlererkennung (Bild 1a) oder durch den Vergleich zwischen zwei Rechnern (Bild 1b/c) ermöglicht (siehe hierzu im Detail Abschnitt 4.1 / 4.2).
- Stetige Rechner (Bild 1d) beruhen auf der Möglichkeit, von dem auf einem Rechner ablaufenden Prozess auf einen synchron dazu laufenden, identischen Schattenprozess, der auf einem zweiten Rechner, dem „Mitrechner“ ausgeführt wird, umzuschalten (Abschnitt 5).
- Maskierende Redundanz (Bild 1e) beruht auf der Bereitstellung von mehr als zwei Rechnern für denselben Prozess, so dass zwischen diesen Rechnern eine Mehrheitsentscheidung ermöglicht wird (Abschnitt 6).

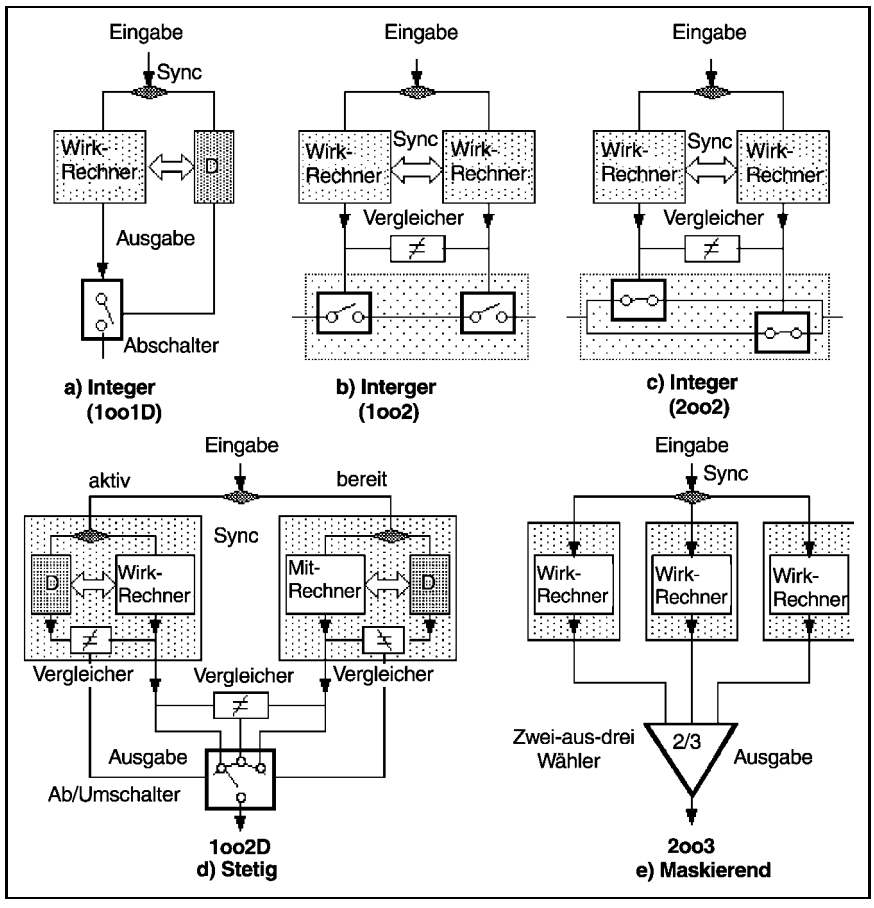


Bild 1: Musterarchitekturen zur Realisierung der Integrität, der Stetigkeit und der Maskierung. IEC 61508 bezeichnet als xooy („x out of y“), wie viele der vorhandenen y redundanten Einheiten zur Aufrechterhaltung der Produktion notwendig sind; D bedeutet die Existenz einer Selbstprüfung.

Man kann sich fragen, warum man nicht grundsätzlich maskierende Rechner verwendet. Dies liegt daran, dass (grob geschätzt) maskierende 2oo3-Rechner wegen der Verdreifachung der Basis-Hardware und der Benötigung relativ teurer Spezial-Hardware (für Vergleicher und Synchronisierer) mehr als das Sechsfache eines Simplex-Rechners kosten. Eine einfache stetige Lösung ohne Vergleicher kostet immerhin das Dreifache eines Simplex-Rechners. Darum werden technische Lösungen interessant, die nicht ideal sind, sondern die Eigenschaften der gesteuerten Strecke ausnützen.

Ist Maskierung nicht vorausgesetzt, wird die Frage interessant, wie lange die Strecke einen Integritäts- bzw. einen Stetigkeitsbruch toleriert. Dies setzt voraus, dass das Verhalten der Strecke bei Ausfall des Rechners bestimmt ist. Diese Kenntnis ist ohnehin nötig, denn das Auftreten eines Fehlers im Leitrechner ist keineswegs die einzige Störung, welche die Strecke erfährt. Man darf aber 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 auf Grund der Eigenschaften der Strecke geschlossen werden kann, was für Ziele bezüglich Sicherheit und Verfügbarkeit anvisiert werden.

2 Einfluss des Ausfalles auf die Strecke

Die Vielfalt an Strecken ist sehr groß. Jede Strecke reagiert anders auf den Ausfall ihres Leitrechners. Strecken werden in der Regelungstechnik in zwei Hauptklassen eingeteilt, kontinuierliche und sequenzielle. Während kontinuierliche Strecken meist geregelt werden, werden sequenzielle Strecken meist gesteuert. Diese Unterscheidung ist ebenfalls für die Fehlertoleranz wichtig. Strecken werden meist untersucht im Hinblick auf Störungen. Hier interessiert eine besondere Störung, nämlich der Ausfall des leitenden Rechners oder der Leitanlage.

2.1 Kontinuierliche Strecken

Kontinuierliche Strecken, wie z. B. geregelte elektrische Antriebe, werden gewöhnlich durch Differenzialgleichungen 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. Zum Beispiel lässt sich durch Verstellen der Klemmenspannung das Drehmoment eines Elektromotors sowohl erhöhen als auch erniedrigen. Diese Reversibilität gilt jedoch nur innerhalb eines Teilbereiches des Zustandsraums, des Regelbereichs. Beispielsweise bleibt ein Fahrzeug im reversiblen Zustand und kann gesteuert werden, solange es auf seiner Spur bleibt.

Kontinuierliche Strecken reagieren auf Integritätsbruch (falsche Daten) 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. Hingegen tolerieren kontinuierliche Strecken kein langes Ausbleiben der Steuergröße: sie haben meist keine natürliche Stabilität im zulässigen Bereich, da sie sonst keine Regelung brauchen würden. Das Ausbleiben der Steuergröße wird nur während einer begrenzten Toleranzfrist von der Strecke geduldet. Diese hängt davon ab, ob die Strecke sich im eingeschwungenen Zustand befindet oder nicht; aber man geht vom schlimmsten Fall einer hochdynamischen Situation aus. Ein längeres Aussetzen führt entweder zu Schaden (auch ohne Unfall kann die Strecke frühzeitig altern) oder zu einem Nothalt, der durch die Sicherheitsvorrichtungen ausgelöst wird. Nothalt ist meist verbunden mit einem längeren Ausfall der Anlage und mit Stillstandkosten.

2.2 Sequenzielle Strecken

Sequenzielle Strecken, wie z. B. Montageroboter oder Aufzüge, werden gewöhnlich durch endliche Automaten, Petri-Netze oder Übergangsmatrizen beschrieben. Die Zustände der Strecke sind endlich und wohldefiniert (z. B. Lift ist im vierten Stock). Der Übergang von einem Zustand zum anderen ist abrupt und meist nicht-monoton, d. h. er kann nicht rückgängig gemacht werden durch Wegnahme des Signals, 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. Reversibilität kann nur erreicht werden, wenn es eine Umkehroperation gibt. Diese kann nicht vorausgesetzt werden (es ist schwierig, ein irrtümlich rot gespritztes Auto wieder in den Zustand vor dem Spritzvorgang zu bringen).

Sequenzielle Strecken sind deshalb äußerst empfindlich gegen falsche Steuergrößen, denn eine automatische Kompensation ist ein intelligenter Vorgang, der meist mit erheblichem Aufwand verbunden ist. Zum Beispiel ist es mühsam, eine Fabrikationsstraße in einen früheren Zustand zurückzuversetzen, wann einmal falsche Befehle ausgegeben worden sind. Darum ist die Toleranzfrist einer sequenziellen Strecke gegenüber Fehlleitung grundsätzlich Null, von der Steuerung wird Integrität verlangt.

Hingegen sind die Schäden bei Ausbleiben von Steuerbefehlen meist begrenzt auf die Produktionsausfallkosten. Die tolerierte Aussetzfrist ist relativ lang, danach steigen die Ausfallkosten beträchtlich.

2.3 Sicherheit und Verfügbarkeit

Sicherheit ist die Wahrscheinlichkeit, dass eine Strecke durch Ausfall 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 [2; 3].

Ein wichtiges Merkmal einer Strecke ist das Vorhandensein einer sicheren Seite („safe side“), 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. Strecken, die keine sichere Seite kennen, werden durch das Ausbleiben der Leitfunktion gefährdet, sie treten aus dem Regelbereich. Bei solchen Strecken hilft keine Schutzeinrichtung. Darunter fallen z. B. Fahrzeuge, die nicht spurgeführt sind, wie z. B. Flugzeuge oder Raketen, aber auch instabile 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 einer Produktionsunterbrechung, 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 in einen 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 des Rechners vermindert die Verfügbarkeit der zu schützenden Strecke, mangelnde Stetigkeit vermindert ihre Sicherheit.
- Wenn die Strecke keine sichere Seite kennt, dann gefährden sowohl falsche Daten als auch das Ausbleiben der Daten die Strecke. Der Rechner hat sowohl eine Leit- wie eine Schutzfunktion. In diesem Fall kommen maskierende Rechner zum Einsatz.

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, welche die Funktion eines ausgefallenen Wirkteils 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.

3.1 Prüfredundanz

Der Gütefaktor der Prüfredundanz ist der Überdeckungsgrad c, der die Wahrscheinlichkeit ausdrückt, dass ein Fehler innerhalb vorgegebener Frist entdeckt wird. Die Prüfredundanz kann durch Software realisiert werden, z. B. durch Prüfprogramme, die zwischen Verarbeitungsschritte eingeflochten werden, oder durch Wiederholen (manchmal in anderer Weise) von Operationen und Vergleich. Solche Tests können aber nicht alle Fehler erkennen; dafür verursachen sie wenig Hardwarekosten.

Ein besserer Überdeckungsgrad wird durch parallele Tests erreicht, die neben der eigentlichen Datenerzeugung ablaufen. Dazu ist zusätzliche Hardware notwendig. Bei regulären Strukturen wird dabei Codierung verwendet, wie Paritätsprüfung auf Bussen, Checksumme bei Speicherelementen, zyklische Prüfsummen (CRC) bei seriellen Übertragungen. Bei Prozessoren ist die Komplexität derart hoch, dass Fehler am besten durch Duplizierung und Vergleich erkannt werden. Eine Anlage, bei der jeder Einzelfehler entdeckt wird, gilt als selbstprüfend. Wenn selbst Fehler der Fehlerdetektion nicht unentdeckt bleiben, spricht man von „vollkommener Selbstprüfung“.

3.2 Wirkredundanz

Die Wirkredundanz ist imstande, die Funktion eines ausgefallenen Teiles zu übernehmen. Die Wirkredundanz kann identisch sein zum Wirkteil, den sie ersetzt, oder diversitär, d. h. aus unterschiedlichen Entwürfen stammend. Das zweite fällt hauptsächlich 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 (Bild 2).

- Nachführung

Der Rechner, der an der Strecke angeschlossen ist, der Wirkrechner, führt den Rechner in Bereitschaft, den Ersatzrechner, in regelmäßigen Abständen nach. Zu diesem Zweck werden Rücksetzpunkte in die Software eingeführt, die regelmäßig dem Ersatzrechner den Stand des Wirkrechners mitteilen [4]. Im Fehlerfall nimmt der Ersatzrechner die Arbeit beim letzten Rücksetzpunkt wieder auf, d. h. die Umschaltung ist nicht immer nahtlos. Wirk- und Ersatzrechner werden durch die Zustandsrettung belastet, jedoch ist der Ersatzrechner sonst frei für andere Arbeiten oder darf, z. B. zum Stromsparen oder zur Verminderung der Ausfallrate (Raumsonden!) abgeschaltet werden. Um den zu rettenden Kontext möglichst klein zu halten, muss die Anwendung die Zustandsrettung durchführen.

Nachgeführte Rechner werden auch dort gebraucht, wo die verarbeitete Datenmenge sehr groß ist. Dann kann ein Rechner als Ersatzrechner (Back-up) für mehrere andere gebraucht werden. Diese Lösung wird in der Steuerungs- und Regelungstechnik kaum verwendet.

Ein Spezialfall der nachgeführten Rechner wird hingegen häufig verwendet: der Wiederanlauf der Applikation auf der

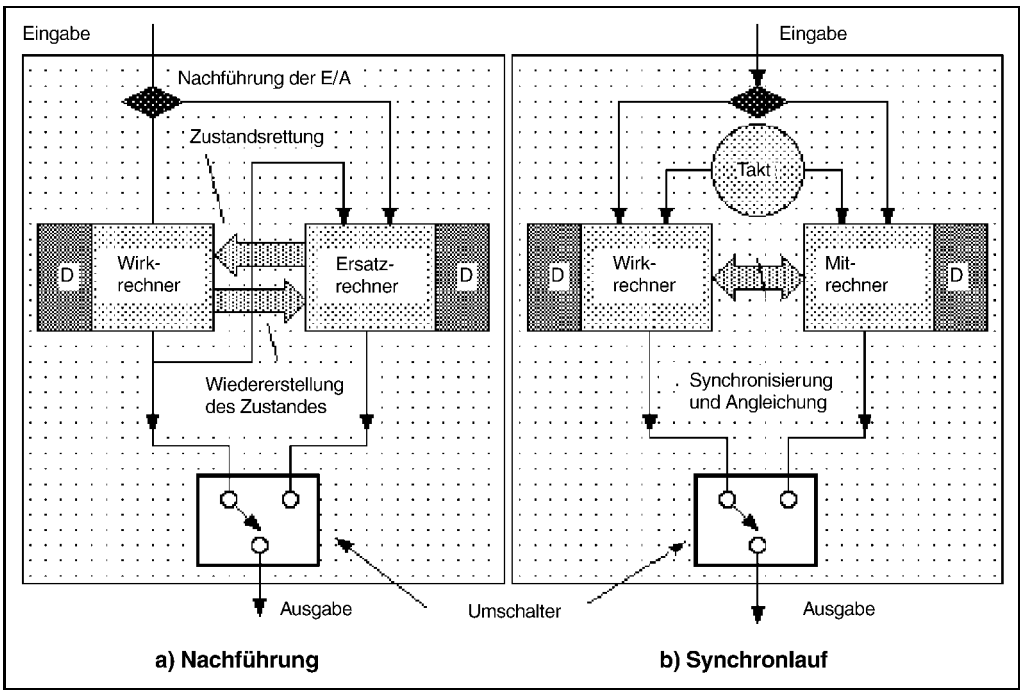


Bild 2: Nachführung und Synchronlauf.

gleichen Hardware. Dies basiert auf der Erkenntnis, dass die allermeisten Fehler transienter Natur sind (auch, und insbesondere Softwarefehler), und dass die Hardware dabei intakt bleibt. Solch ein Wiederanlauf setzt aber voraus, dass zumindest ein (konsistenter) Teil des Zustandes regelmäßig in einem sicheren Speicher abgelegt wird, welcher als Basis für das Wiederaufsetzen dient.

Die Prinzipien der nachgeführten Rechner dienen aber auch der Wiedereingliederung einer reparierten Einheit – auch bei Synchronlauf. Es genügt nicht, neue Hardware bereitzustellen, diese muss noch aktualisiert werden, um als Wirkredundanz gebraucht werden zu können.

- Synchronlauf

Der Wirkrechner und der Ersatzrechner führen die gleichen Programme zur gleichen Zeit aus. Da Digitalrechner im Prinzip deterministisch sind, sollten ihre Zustände stets identisch sein. Ihre Ausgaben können zum Zweck der Fehlererkennung verglichen werden („Verdoppelung und Vergleich“). Synchronlaufende Rechner können also im Gegensatz zu nachgeführten Rechnern als 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: z. B. können die Speicherzugriffe unterschiedlich lang sein, 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.

Eine spezielle Form der synchronen Redundanz bieten die Rechner, die Zeitredundanz ausnützen. Operationen werden auf der gleichen Hardware mehrmals, manchmal auch diversitär (d. h. mit unterschiedlichen Algorithmen), ausgeführt. Dies ist eine wirksame Art, Fehler zu erkennen, welche aber keine Wirkredundanz bietet.

Mit Prüfredundanz und Wirkredundanz lassen sich fehlertolerante Strukturen aufbauen. Entsprechend den Ausfallarten sind zwei unterschiedliche Arten fehlertoleranten Verhaltens der Leitanlage erwünscht: Integrität und Stetigkeit.

4 Integre Steuerungen

Leitanlagen, die keine falsche Daten liefern, werden als „integer“ bezeichnet. Zum Erreichen der Integrität dient Prüfredundanz, die beim Auftreten eines Fehlers die Ausgabe abkoppelt. Dies gewährleistet ein „fail-stop“-Verhalten. Der Fehlerdetektor wird möglichst nahe der Strecke angesiedelt, ja am besten in die Strecke integriert. Bei Unstimmigkeit wird die Anlage abgeschaltet. Dies ist nur sinnvoll, wenn die Strecke eine sichere Seite besitzt.

4.1 Selbstprüfung

Es ist im Prinzip möglich, Leitanlagen so aufzubauen, dass jeder einzelne Teil selbstprüfend ist. Speicher werden dabei durch fehlerkorrigierende Codes (englisch: error correcting codes (ECC)) abgesichert, Busse durch Parität und Prozessoren durch Verdoppelung und Vergleich.

Es werden aber keine integren Rechner auf dem Prinzip der durchgehenden Selbstprüfung verwendet, obwohl dies verschiedentlich versucht wurde. Der Grund liegt darin, dass

gemeinsame Fehlerursachen bei so komplexen Elementen wie z. B. Prozessoren nicht genügend ausgeschlossen werden können. Lediglich bei seriellen Datenübertragungen kann der Nachweis für einen ausreichenden Überdeckungsgrad der Codierung erbracht werden. Dies ist z. B. in den Normen für Fernwirkprotokolle [5] festgehalten.

Für die Eisenbahnsignalisierung (z. B. bei der vollautomatisierten Pariser Metro-Linie „Meteor“) hat MATRA Rechner entwickelt, welche jede Operation durch eine (mathematisch inverse) Prüfoperation im Werte- und Zeitbereich ergänzt und bei jedem Schritt eine Signatur bildet. Diese als „kodierter Monoprozessor“ bekannte Lösung läuft auf konventioneller Hardware [6]. Derartige Ideen wurden auch bei der Simatic S7-400 Steuerung aufgegriffen [7].

Selbstprüfung, auch wenn sie nicht vollkommen ist, nützt immer; und sei es nur, um die Reparaturzeit zu kürzen. Darum werden Leitanlagen mit umfangreichen Selbsttests versehen, die bis zu 30% der Software ausmachen können. Ein Selbstprüfungs-Konzept fand z. B. bei SACEM, zur fehlertoleranten Geschwindigkeitskontrolle bei Zügen Eingang [8].

Bei SEL-Alcatel ist das Elektra-Signalsystem in zwei Kanäle aufgeteilt, von denen der eine die Funktion des Systems ausführt, während der andere für das Prüfen der Sicherheitsbedingungen zuständig ist; diversitärer Entwurf dieser beiden Systemteile soll außerdem Entwurfsfehler in Software oder Hardware erkennbar machen [9].

Beim Flug-Kontrollzentrum für die Ariane-5 wird Selbstprüfung der Rechensysteme durchgeführt; dabei als ausgefallen erkannte Komponenten werden durch eine duplizierte Komponente ersetzt [10].

IEC 61508 bezeichnet eine selbstprüfende Steuerung als 1oo1D (1 aus 1 mit Diagnose).

4.2 Verdoppelung und Vergleich

Wenn hohe Anforderungen an die Integrität der Leitanlage gestellt werden, kommt Verdoppelung und Vergleich zum Zuge (Bild 3):

Es wird dabei Wirkredundanz im Synchronlauf zur Fehlererkennung herangezogen. Die Grundannahme ist, dass beide Rechner sich identisch verhalten und gleiche Ausgangsgrößen liefern, solange keiner fehlerhaft ist. Der Vergleicher stoppt die Ausgabe bei Nicht-Übereinstimmung. Diese Anordnung gewährt Integrität, solange nicht der gleiche Fehler auf beiden Seiten auftritt.

Je nach Art der Ausgabesperre spricht IEC 61508 von 1oo2 (jede Steuerung kann die Sicherheitsfunktion auslösen) oder von 2oo2 (beide Steuerungen müssen übereinstimmen, um die Sicherheitsfunktion auszulösen).

Die Wirkredundanz setzt jedoch voraus, dass beide „Wirkrechner“ identische Eingabedaten zur gleichen Zeit bekommen, absolut identisch aufgebaut sind und sich ansonsten deterministisch verhalten. Dazu braucht man eine Synchronisierung der Rechner. Die SYNC-Verbindung übernimmt

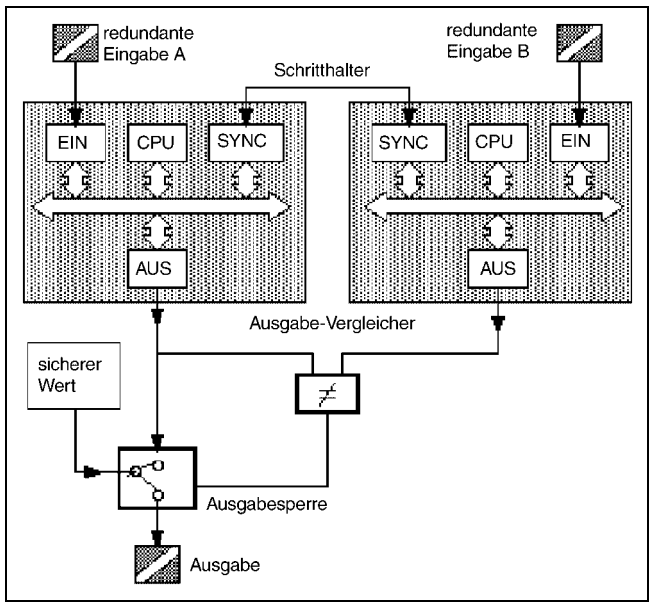


Bild 3: Verdoppelung und Vergleich (1oo2, 2oo2).

die Angleichung der Eingaben und Unterbrechungen, die Synchronisierung der Uhren, und die Synchronisierung der internen Zuständen. Beim Starten des Paares und bei Wiedereingliederung eines ersetzten Teiles gleicht sie die internen Zustände wieder an.

Das Prinzip kann jedoch durch gemeinsame Fehlerursachen überlistet werden. Zu diesen zählt in erster Linie die Software, die auf beiden Seiten identisch ist. Zwar können Prüfprogramme bestimmte Softwarefehler entdecken. Damit lässt sich aber keine 100%-ige Integrität erreichen, denn der Überdeckungsgrad der Eigenprüfung der Software ist beschränkt. Man versucht, mit Hilfe diversitärer 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, umso 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 wird der Vergleich erheblich erschwert.

Integre Geräte werden meist in sicherheitsrelevanten Anwendungen eingesetzt und unterliegen den TÜV- und VDE-Vorschriften. Beispiele von 1oo2 -Rechnern sind der H50-SK3 der Firma HIMA und die Siemens SIMIS-Eisenbahnsignalisierung. Solche Leitanlagen, die eher stoppen als falsche Daten liefern, zeigen ein „Fehlstopp“-Verhalten. Damit eine Anlage selbständig weiterfahren kann, braucht sie überdies Stetigkeit, d. h. Wirkredundanz.

5 Stetige Steuerungen

Anlagen, die ihre Funktion trotz Ausfall eines ihrer internen Elemente fortsetzen können, werden als stetig oder „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.

5.1 Redundante Ein- und Ausgabe

Obwohl sich das Augenmerk darauf richtet, ist der Rechner keineswegs das schwächste Glied in der Kette, 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 wie Temperaturschwankungen, elektromagnetischen Störungen, Lärm und Vibrationen ausgesetzt. Die Rechner hingegen befinden sich meist in der System-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 Verlässlichkeitsziele bereits erreicht, es erübrigt sich eine redundante Auslegung des Rechners.

Eine vollkommene Integrität kann nicht erreicht werden, solange Lücken in der Fehlererkennung bestehen und der Rechner selbst eine offensichtliche gemeinsame Fehlerursache darstellt. Eine teilweise Integrität kann durch programmierte Prüfverfahren und Verriegelungen der Ausgabe erreicht werden.

Eine gewisse Stetigkeit des Rechners selbst kann auch erreicht werden, allerdings nur gegen transiente Fehler. Zu diesem Zweck wird der wesentliche Teil des Zustands regelmäßig in einem nicht-flüchtigen (z. B. batteriegepufferten) Speicher gerettet. Eine Wiederanlaufprozedur startet den Rechner erneut mit dem letztgültigen Datensatz. Dies lohnt sich nur, wenn die Wahrscheinlichkeit eines transienten Fehlers hoch ist.

5.2 Verdoppelung, Vergleich und Diagnose (1oo2D)

Die 1oo2D (1 aus 2, mit Fehlerdetektion) – Struktur (vergleiche Bild 1d) ist die billigste Art, zu einem integren/stetigen Rechner zu kommen. Äußerlich ist sie der 2oo2-Anordnung (Verdoppelung und Vergleich, Bild 3) ähnlich. Die 1002D-Struktur bietet zunächst Integrität gegen einen ersten Fehler, indem die Ausgänge beider Steuerungen verglichen werden. Um Stetigkeit zu erreichen, muss entschieden werden, welcher Rechner ausgefallen ist.

Bis die defekte Steuerung identifiziert ist, wird die Strecke nicht geführt, also muss die Strecke eine ausreichende Fehlertoleranzzeit aufweisen. Die Dauer des Aussetzens hängt vom gewünschten Fehlerüberdeckungsgrad ab. Nach der Diagnose wird der Rechner, der wahrscheinlich ausgefallen ist, abgekoppelt und der Vergleicher wird abgeschaltet. Der Betrieb kann weitergeführt werden.

Allerdings sind 1oo2D-Strukturen nicht in der Lage, einen zweiten Fehler sicher zu entdecken und zu überwinden. Der Betrieb nach dem ersten Ausfall darf also 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.

Die Verlässlichkeit dieser Struktur wird noch weiter durch die Wahrscheinlichkeit einer falschen Umschaltung (Fehldiagnose) eingeschränkt. Insbesondere bei transienten Fehlern ist es möglich, dass überhaupt kein Fehler entdeckt wird oder beide Rechner sich als fehlerhaft melden.

Auf der Prozessebene wird die 1oo2D-Struktur zur Steuerung und Regelung vielfach eingesetzt, so z. B. beim Siemens S7-400F, dem HiQuad der Firma HIMA oder dem Safeguard von ABB.

6 Fehlermaskierende Steuerungen

Leitanlagen, die bei Ausfall einer ihrer internen Elemente weder falsche Daten liefern noch ihre Funktion unterbrechen, werden als fehlermaskierend bezeichnet. Die Aussetzzeit einer maskierenden Leitanlage ist grundsätzlich Null, Fehler treten nach außen nicht in Erscheinung.

6.1 Zwei-aus-Drei (2oo3)

Die Musterarchitektur für maskierende Rechner ist die 2oo3-Anordnung. 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, und die Aussetzzeit ist Null. Diese Anordnung ist sowohl stetig (mit Aussetzzeit Null) als auch integer, die Strecke merkt also nichts von einem Ausfall. Dabei gilt die Auswahlschaltung als absolut zuverlässig, sie kann selbst auch redundant ausgeführt werden.

Fehlermaskierung ist nur bis zu einer bestimmten Anzahl unabhängiger Fehler möglich. Irgendwann einmal ist die Redundanz erschöpft und Fehler können nicht mehr vor der Strecke verborgen werden. Dabei ist es noch zu einem gewissen Grad möglich, festzulegen, ob sich der Rechner integer oder stetig zu verhalten hat:

- In der 2oo3-Anordnung (Bild 4) kann der Wähler bei einem zweiten Fehler (auf einem anderen Rechner) die Ausgabe noch stoppen, vorausgesetzt, der Verursacher des ersten Fehlers wurde ausgeschlossen (so dass also keine Verschwörung zweier fehlerhafter Rechner vor-

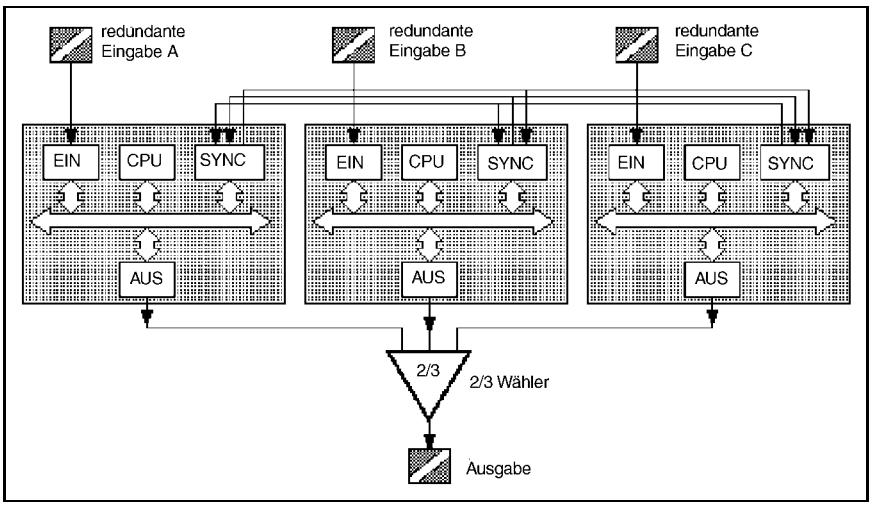


Bild 4: Zwei-aus-Drei-Struktur mit Voting (2oo3).

liegt). Die 2oo3-Anordnung bleibt bei Zweifachfehlern noch integer (aber nicht mehr funktionsfähig).

- Die 2oo3-Anordnung kann auch bei einem zweiten Fehler stetig bleiben, falls man den fehlerhaften Rechner identifizieren kann, z. B. indem eine Diagnose durchgeführt wird. In diesem Fall wird der Betrieb nach einem relativ langen Aussetzen weitergeführt. Dies wird selten gemacht, denn 2oo3-Anordnungen werden bei kritischen Strecken eingesetzt, und diese verlangen in erster Linie Integrität.

Solche Rechner werden für die Brennersteuerung nach VDE 0116 eingesetzt. Ein Beispiel ist der Siemens 220 EHF [11]. In der Eisenbahn-Signalisierung arbeitet das LZB-System von Siemens auch als 2oo3-Anordnung. Steuerungen für beliebige Anwendungen auf dieser Basis bieten ABB (Triguard) und Triconex (Trident) an.

6.2 Zwei-von-Vier (2oo4)

Dort, wo man sehr hohe Anforderungen an die Verlässlichkeit der Rechner stellt, werden 2oo4- Anordnungen gebraucht. Vier Rechner werden synchronisiert und führen die gleichen Aufgaben aus. Eine Majoritätsschaltung leitet die wahrscheinlichste Ausgabe weiter. Diese Majoritätsschaltung rekonfiguriert sich bei einem ersten Fehler zu einer 2oo3-Anordnung. Der Fall, dass zwei oder mehrere Einheiten in gleicher Weise ausfallen, wird zwar oft befürchtet, ist aber unwahrscheinlich.

Eine 2oo4-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 [12] verwendet.

Der fehlertolerante Rechner, der für das Kontrollsystem des projektierten europäischen Weltraum-Shuttles „Hermes“ entwickelt wurde, stützt sich für die Start- und Landephase auf 4-fach redundant ausgelegten Prozessorbetrieb, während für die Orbitalphase ein Selbstprüfungs-Ansatz benutzt wird, bei dem – auch zwecks Einsparen von Strom – nicht alle 4 Prozessoren arbeiten [13].

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 dürfen nicht wesentlich zur Unzuverlässigkeit des Ganzen beitragen. Am besten wird die Redundanz bis in die Strecke selbst hineingezogen: dies ist aber nicht immer möglich.

7 Software

Je sicherer die Hardware, desto eher tritt die Unzuverlässigkeit der Software in den Vordergrund. Während die Hardware-Redundanz weitgehend verstanden und zum Teil standardisiert ist, werden Spezifikation, Entwurf und Programmierung der Software noch weitgehend empirisch erstellt, obwohl sie für den größten Teil der Ausfälle verantwortlich sind. Die von der Software verursachten Unfälle haben bereits Milliardenschäden verursacht, mit dem Verlust der Ariane V – Rakete als Musterbeispiel.

Es ist offensichtlich, dass die Integrität der Software nicht durch Wirkredundanz gesteigert werden kann. Es wird gelegentlich versucht, für kritische Anwendungen wie Flugzeugsteuerungen diversitäre (N-Versionen-) Programmierung zu verwenden, wobei verschiedene Programmiermannschaften die gleiche Software entwickeln, womöglich unter Verwendung verschiedener Sprachen und Compiler [14]. Die Auswahl selbst im fehlerfreien Fall braucht einen intelligenten Vergleicher, da die Resultate ähnlich, aber nicht unbedingt identisch sind.

Erst dort, wo die Strecke selbst eine intelligente Auswahl treffen kann. z. B. indem unabhängige Rechner auf unabhängige Aktoren (z. B. Steuerungsflächen) einwirken, hat sich diese Technik etabliert. Diese Technik wird sowohl beim Airbus [15] wie auch bei der Boeing 777 [16] verwendet. Selbst dann haben jedoch die Behörden Bedenken erhoben [17] und verlangt, dass jeder Pfad einzeln validiert wird. Ein Grund ist, dass die Ausfälle der Versionen

nicht als vollständig unabhängig voneinander angesehen werden können, stammen sie doch aus den gleichen Spezifikationen und von Programmierern, die ähnliche Fehler machen [18].

Man kann sich fragen, ob der Aufwand nicht in einen besseren Erstellungsprozess und in den Test der Software statt in parallele Entwicklungen gesteckt werden soll. Auch stellt sich die Frage, ob die Erfahrungsbasis bei spezieller Software nicht zu klein ist, um wertvolle Erfahrungen zu sammeln, und ob es nicht besser wäre, handelsübliche Komponenten, sogenannte COTS (Commercial-Off-The-Shelf) – Moduln statt Spezialanfertigungen zu verwenden. Die Schwierigkeit dabei ist die Abschätzung des Verhaltens einer unbekannten Komponente (Black Box), welche nur durch ausgiebiges Testen möglich ist [19; 20]. Dem läuft auch die Versuchung des Ingenieurs entgegen, die COTS-Komponente zu verbessern, eine – selbst nur geringfügig – abgeänderte COTS-Komponente ist aber keine mehr.

8 Standardisierung und Typenprüfung

Die europäische und internationale Standardisierung fehlertoleranter Steuerungen hat in den letzten Jahren viele Fortschritte gemacht, sowohl durch allgemeine Richtlinien (z. B. IEC 61508, vergleiche Tabelle 1 und 2) wie durch gebietsspezifische Normen (z. B. EN 50128/9 für die Eisenbahntechnik). Deutschland war dabei wegweisend, mit Arbeiten des TÜVs, des Technical Committee TC7 der EWICS und der Eisenbahnsignalisierungs-Industrie, insbesondere für das Europäische Signalisierungssystem ETCS.

Im Anhang steht eine Liste von Steuerungen, die durch den TÜV geprüft worden sind. Der Verweis eines Herstellers auf eine erfolgreiche Typenprüfung durch den TÜV oder eine andere Instanz nach einer allgemeinen Richtlinie wie IEC 61508 soll nicht überbewertet werden. Die Typenprüfung untersucht lediglich den Prüfling auf strukturelle Fehler hin, die seine Verwendung in Gebieten verwehren würde, die eine bestimmte Integrität oder Stetigkeit ver-

Tabelle 1: Die wichtigsten Sicherheitsnormen in Europa.

IEC/EN 62061	Safety of machinery – functional safety – Electrical, electronic and programmable electronic control systems
IEC/EN 61511	Functional safety of E/E/PES safety related systems – Functional safety: safety instrumented systems for the process industry sector
IEC 61508 (VDE 0801)	Functional safety of E/E/PES safety related systems – International standard (allgemeine Richtlinien)
IEC 60300	Dependability management
ISO/IEC 13849 (EN 954)	Safety of machinery – Safety-related parts of control systems
EN 50159	Requirements for safety-related communication in closed/open transmission systems
EN 50129	Railways applications – Safety-related electronic systems for signalling
EN 50128	Railways applications – Software for railway control and protection systems
EN 50126 (VDE 0115)	Railways applications – Specification and demonstration of reliability, availability, maintainability and safety (RAMS) – allgemeine Richtlinien
(VDE 0116)	Elektrische Ausrüstung von Feuerungsanlagen
DIN V 19250	Grundlegende Sicherheitsbetrachtungen für MSR-Schutzeinrichtungen (wird durch EN 61508 abgelöst)
IEC 880	Software for computers in the safety systems of nuclear power stations

Tabelle 2: Anforderungsklassen und Sicherheitsstufen.

IEC 61508	**TÜV Anforderungsklassen (DIN 19250)**	**Sporadischer Betrieb Wahrscheinlichkeit einer Funktionsweigerung**	**Kontinuierlicher Betrieb Wahrscheinlichkeit eines gefährlichen Fehlers**	**Gefahr/Risiko-Stufe**
SIL 1	AK 2 & AK3	10^{-1} bis 10^{-2} /h	10^{-6} bis 10^{-5} /h	kleinere Schäden an Anlagen und Eigentum
SIL 2	AK 4	10^{-2} bis 10^{-3} /h	10^{-7} bis 10^{-6} /h	größere Schäden an Anlagen und Eigentum. Mögliche Personenverletzung.
SIL 3	AK 5 & 6	10^{-3} bis 10^{-4} /h	10^{-8} bis 10^{-7} /h	Personenschutz
SIL 4	AK 7	10^{-4} bis 10^{-5} /h	10^{-9} bis 10^{-8} /h	Mögliche katastrophale Folgen

langen. Die Typenprüfung sagt aber nichts darüber aus, wie wahrscheinlich ein Integritätsbruch oder ein Stetigkeitsbruch ist.

Allein schon die Tatsache, dass die Anwendungssoftware nicht mitgeprüft wird, welche für mehr als die Hälfte der Ausfälle verantwortlich ist, zeigt, dass eine anwendungsspezifische Prüfung nachträglich unerlässlich ist. Es gibt zwar einige Richtlinien für die Softwaresicherheit; diese sind jedoch so allgemein formuliert, dass darüber keine Konformität geprüft werden kann.

9 Beispiele von Strecken

9.1 Vermittlungsysteme für Telekommunikation

Verfügbarkeit: sehr hoch (etwa 3 Minuten Aussetzzeit pro Jahr und 50 000 Leitungen nach Bell).
Integrität: Keine besonderen Anforderungen. Der Kunde ist vom Telefon her gewöhnt, bei Falschverbindung neu zu wählen. Die Integrität wird nicht gewährleistet, die Verantwortung für Integrität und Vertraulichkeit wird in die Endgeräte verlegt.
Lösungen: Duplex-Betrieb zweier synchronen Rechner.
Beispiel: Bell 3B20D, Ericcson AXE-10 [21; 22].

9.2 Leitstellen in Industrie und Kraftwerken

In der Leitebene (Leitstand und Bedienplätze) hat Verfügbarkeit Vorrang. 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. Da der Betrieb ununterbrochen laufen soll, wird großer Wert auf die Fähigkeit gelegt, Teile ohne Betriebsunterbrechung auswechseln zu können.

Verfügbarkeit: Meist haben Rechner nur eine Überwachungsfunktion und erhöhen den Bedienungskomfort. In vielen Fällen kann der Betrieb noch manuell weitergeführt werden. Die erlaubte Aussetzzeit kann deswegen relativ hoch sein (etwa bis zu einer Minute), was den Einsatz warmer Redundanz erlaubt.
Integrität: Die Leitanlage ist nicht auf Integrität ausgelegt. Kritische Befehle werden über separate Leitungen ausgegeben oder bedürften einer Bestätigung.
Lösung: Duplex-Betrieb zweier Rechner in Bereitschaft oder Synchronlauf. Beispiel: Compaq NonStop (früher: Tandem) [23]

9.3 Netzleittechnik

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

Verfügbarkeit: 12 Minuten/Monat Ausfallzeit.
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 USA vertraut man eher auf Rückmeldung (Befehl geben, Bestätigen, Entriegeln), in Europa vertraut man auf die Datensicherung des Übertragungsweges.
Lösung: Duplex-Betrieb mit Synchronlauf, verteiltes System. Beispiel: ABB SPIDER, Siemens SINAUT.

9.4 Prozessnahe Automatisierung

Auf der Feldebene sind die Reaktionszeiten viel kürzer, dafür sind die Datenmengen, die den Zustand beschreiben, viel kleiner. Strecke: Kontinuierlich und sequenziell.

Verfügbarkeit: Die Unzuverlässigkeit des Leitrechners darf zu keiner nennenswerten Herabsetzung der Verfügbarkeit führen.
Integrität: Hoch bis sehr hoch bei sequenziellen Strecken, relativ unkritisch bei kontinuierlichen Strecken.
Lösung: bei unkritischen Strecken: Verdoppelung der Ein- und Ausgabe, wo dies nötig ist. Bei kritischen Strecken: 1oo2D, 2oo3.

9.5 Schutzsysteme

Je nach Strecke ist Überfunktion (unzweckmäßige Abschaltung) oder Unterfunktion (Ausbleiben des Auslösebefehles im Störfall) gefährlicher. Z. B. Generatorschutz muss Unterfunktion vermeiden, Sammelschienenschutz muss Überfunktion vermeiden.

Strecke: Sequenziell (Arbeit/Abschalten). Verfügbarkeit: Unterfunktion gefährdet die Strecke.
Integrität: Überfunktion erniedrigt die Verfügbarkeit.
Lösung: Verwendung mehrerer, unabhängiger Schutzsysteme in UND-Verknüpfung gegen Überfunktion, in ODER-Verknüpfung gegen Unterfunktion. Ständige Selbstprüfung zur Entdeckung latenter Fehler.

9.6 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 für Signalisierung.
Verfügbarkeit: Sehr hoch, entsprechende Qualität der Komponenten wird verlangt. Lösung: Verdoppelung und Vergleich, 2oo3, besondere Auslegung.
Beispiel: SIEMENS LZB, Bombardier Signals AB, Alcatel Elektra

9.7 Rollmaterial

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

Integrität: Nur für die Bordsignalisierung (ATP, ATO) verlangt.
Verfügbarkeit: Sehr hoch (1 Totalausfall/700 000 km), entspricht 1 Ausfall pro Monat für das gesamte Netz
Lösung: Ausnützung der Redundanz des Fahrzeuges (unabhängige Drehgestelle), Duplex-Rechner, regelmäßige Wartung.
Beispiel: MICAS (Bombardier), SIBAS 32 (Siemens), AGATE (Alstom).

9.8 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, um die vorgeschriebene Geschwindigkeit nicht zu überschreiten. In zweiter Linie wird Verfügbarkeit verlangt, um die Bereitschaft trotz höherer Störanfälligkeit der integren Rechner zu erhöhen. Bei aktiv geführten Fahrzeuge wird in erster Linie Stetigkeit verlangt.

Verfügbarkeit: Weniger als 1 Ausfall auf 1000 km Netz / Tag.
Integrität: Das Fahrzeug darf nicht in einen gefährlichen Zustand wegen Fehlleitung gelangen
Lösung: 2oo3-Anordnung, 1oo2D mit Diversität (coded monoprocessor).
Beispiel: BART (San Francisco), Meteor (MATRA).

9.9 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 ausweisen soll. Diese ist nur durch Maskierung zu erreichen. Die gleichen Vorschriften gelten für den Autopilot tiefffliegender Kampfflugzeuge.

Natürlich instabile 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.
Ausfallrate: 10^{-10} / Flugstunde.
Lösung: 2oo3-Anordnung mit zusätzlichen Ersatzrechnern und Rekonfiguration.
Beispiel: SIFT [24], FTMP [25], Airbus [15], Boeing 777 [16].

9.10 Straßenfahrzeuge

Zunehmend greifen Steuerungen in das Fahrverhalten der Fahrzeuge ein. Angefangen mit ABS, E-Gas und ESP wird an der direkten Lenkung (Drive-by-Wire) gearbeitet. Die Strecke erlaubt praktisch keinen Integritätsbruch und nur sehr kurze Stetigkeitsbrüche. Die Anforderungen sind in vielen Bereichen härter als in der Luftfahrt.

Verfügbarkeit: Mit derjenigen des Motors vergleichbar.
Integrität: Toleranzzeit etwa 5 ms.
Ausfallrate: 10^{-10} / Stunde.
Lösung: 1oo2D Knoten mit redundantem Netzwerk.
Beispiel: X-by-Wire [26].

Firm	**Produkt**	**Technologie**	**IEC 61508**	**DIN 19250**
ABB	ABB Safeguard 400	1oo2D	SIL 3	AK 6
ABB Industri (August)	ABB Triguard SC300E	2oo3	SIL 3	AK 6
ABB	AC31-S	1oo1D		AK 4
Bombardier	Adtranz MICAS-L	1oo2		AK 6
GE Fanuc	GMR 90-70	2oo3		AK 6
HIMA-Sella	HiQuad	1oo2D+2oo4	SIL 3	AK 6
Honeywell	FSC100R	1oo2D		AK 6
ICS	Trusted ICS (Triplex)	2oo3		AK 6
Kongsberg-SIMRAD	AIM 1000	2oo2		AK4
Pilz	PSS 3000	1oo3		AK 6
Schoppe & Faeser	Contronic 3	1oo2		SC3
Triconex (Invensys)	Trident	2oo3	SIL 3	AK 6

Siemens	S5-115F	1oo2		AK 6
Siemens	S7-400HF	1oo2	SIL 3	
Siemens	Moore Quadlog	1oo2D	SIL 3	AK 6
Yokogawa	ProSafe DSP	2oo3		AK 6
Yokogawa	ProSafe PLC	1oo2D		AK 6

(Quelle: www.tuv-fs.com)
Erläuterungen: AK Anforderungs-Klasse, SIL Safety / Integrity Level; die jeweils nachfolgende Kennziffer charakterisiert die Stärke der Anforderungen, die das Produkt nach der Norm erfüllen muss (je höher die Kennziffer, desto höher auch die Anforderungen).

Literatur

[1] H. Kirrmann, ,,Industrieller Einsatz fehlertoleranter Rechner“, Oldenbourg Verlag, it-Informationstechnik 30 (1988) Nr. 3, S. 186–195.

[2] H. Hölscher, J. Rader, ,,Mikrocomputer in der Sicherheitstechnik“, Verlag TÜV Rheinland, 1984.

[3] International Electrotechnical Commission, Functional Safety of electrical/electronic/programmable electronic safety -related systems, all parts.

[4] K.-E. Großpietsch, E. Maehle, ,,Fehlerbehandlung in komplexen nebenläufigen Systemen“, Informatik-Spektrum, Dezember 1998, S. 347–355

[5] International Electrotechnical Commission, IEC 61850-5, Telecontrol, teleprotection and associated telecommunications for electric power systems, Dezember 1987.

[6] Certification of the Coded Monoprocessor as ATP according to European standards, APM 99, Mai 1999.

[7] A. Schenk, ,,Simatic S7-400 F/FH: Safety-Related Programmable Logic Controller“, 19th Safecomp 00, Rotterdam, Oktober 2000.

[8] C. Hennebert, G. Guino, ,,SACEM: A Fault-Tolerant System for Train Speed Control“: Proc. FTCS-23, Toulouse 1993, S. 624–628.

[9] H. Kantz, C. Koza, ,,The ELEKTRA Railway Signaling-System: Field Experience with an Actively Replicated System with Diversity“, Proc. FTCS-25, Pasadena 1995, S. 453–458.

[10] J.-L. Deag, ,,The Redundancy Mechanisms of the Ariane 5 Operational Control Center“, Proc. FTCS-26, Sendai, S. 382–386.

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

[12] A. Spector, D. Gifford, ,,The Space Shuttle Primary Computer System“, Comm. ACM, Sept 1984, S. 874–900.

[13] P. David, C. Guidal, ,,Development of a Fault-Tolerant Computer System for the HERMES Space Shuttle“, Proc. FTCS-23, Toulouse 1993, S. 641–647.

[14] A. Avizienis, ,,The n-Version Approach to Fault Tolerant Systems“, IEEE Trans. Software Engineering 11(12) (1985), S. 1491–1501.

[15] J.C. Rouquet, 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, S. 93–97.

[16] Y.C. Yeh, ,,Design Consideration in Boeing 777 Fly-by-Wire Computers“, HASE’98, Betesta, Maryland, November 1998.

[17] John Rushby, ,,Partitioning in Avionics Architectures, Requirements, Mechanisms, and Assurance“, Computer Science Laboratory, SRI International, Menlo Park CA 94025 USA, Technical Report, March 1999.

[18] J.C. Knight, N.G. Leveson, ,,An Empirical Study of Failure Probabilities in Multi-Version Software“, Proc. FTCS-16, Wien 1986, S. 165–170.

[19] Koopman, Phil, ,,Comparing the robustness of POSIX Operating System“, FTCS 1999.

[20] A. Bondavalli, F. Di Giandomenico, J. Xu: Cost-Effective and Flexible Scheme for Software Fault Tolerance, Journal of Computer Systems Science and Engineering, CRL Publishing Ltd. 1993, Vol. 8, S. 234–244.

[21] B. Ossfeldt, I. Jonsson, ,,Recovery and Diagnostics in the Central Control of the AXE Switching System“, IEEE Trans. Computers, Juni 1980, S. 482–491.

[22] J. Wallace, W. Barnes, ,,Designing for Ultra-High Availability: The Unix RTR Operating System“, IEEE Computer, Vol. 17, Nr. 8, August 1984, S. 31–39.

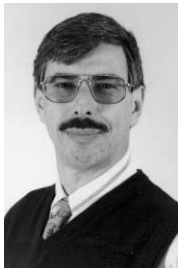
[23] http://zle.nonstop.compaq.com/view.asp?PAGE=ZLE_HomeExt

[24] J. Goldberg et al., ,,Development and Analysis of the SIFT Computer“, NASA contractor report 172146, SRI International, Februar 1984.

[25] A. Hopkins, T.B. Smith, J. Lala, ,,FTMP – A Highly Reliable Fault-tolerant Multiprocessor for Aircraft“, Proceedings of the IEEE, Special Issue on Fault-tolerant Computing, Oktober 1978, S. 1221–1239.

[26] R. Belschner et al, ,,Anforderungen an ein zukünftiges Bussystem für Fehlertolerante Anwendung aus Sicht Kfz-Hersteller“, VDI Kongress, Baden-Baden, 2000.

Manuskripteingang: 13. November 2001.



Prof. Dr. Hubert Kirrmann ist Senior Scientist am ABB Forschungszentrum in Baden (Schweiz) und unterrichtet an der Eidgenössischen Technischen Hochschule in Lausanne. Seine Arbeitsgebiete sind industrielle, fehlertolerante Leitsysteme und Kommunikationsbusse. Er ist Editor des Standards IEC 61375 (Train Communication Network) und Member of the Editorial Board of IEEE MICRO.

Adresse: ABB Forschungszentrum, CH-5405 Dättwil (Schweiz),
E-Mail: hubert.kirrmann@ch.abb.com

Dr. Karl-Erwin Großpietsch ist Wissenschaftlicher Mitarbeiter am Fraunhofer-Institut für Autonome intelligente System (FhG-AiS) in St. Augustin-Birlinghoven. Seine Arbeitsgebiete sind Fehlertoleranz, Rechnerarchitektur und autonome Systeme; die Ergebnisse dieser Arbeiten sind in über 130 Veröffentlichungen dokumentiert. Seit März 1998 ist er Sprecher des GI/ITG-Fachausschusses „Verlässlichkeit und Fehlertoleranz“; seit September 2001 ist er außerdem Chairman des Board of Directors der Euromicro.

Adresse: Fraunhofer-Gesellschaft, Institut für Autonome intelligente Systeme, D-53754 St. Augustin,
E-Mail: grosspietsch@ais.fraunhofer.de