ESTE EXEMPLAR CORRESPONDE À
REDAÇÃO FINAL DA TESE DE DOUTORADO
DEFENDIDA POR CZESLAU L. BARCZAK,
APROVADA EM 15 DE DEZEMBRO DE 1987

PROF. MSURICIO PROFFT
ORIENTADOR

ARQUITETURAS DE COMPUTADORES PARA CONTROLE DE PROCESSOS INDUSTRIAIS, COM CARACTERÍSTICAS DE TOLERÂNCIA A FALTAS.

Autor: Czeslau Lubomiro Barczak

UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA DE CAMPINAS

DEPARTAMENTO DE ENGENHARIA MECANICA

Tese de Doutorado.

TITULO DA TESE: ARQUITETURAS DE COMPUTADORES PARA CONTROLE DE PROCESSOS INDUSTRIAIS, COM CARACTERISTICAS DE TOLERANCIA A FALHAS.

Autor: Caesiau Lubconiro Barczak.

Orientador: Dr. Maurício Prates de Campos Filho.

Aprovado por:

Prof. Dr. Mauricio Prates de Campos Filho, Presidente.

Prof. Dr. Eduardo Assis Alvarenga

Prof. Dr. Odmar Simões Pires

Prof. Dr. Amauri Garcia

Prof. Dr. Rezende Gomes dos Santos

Campinas, 15 de dezembro de 1987.

AGRADECIMENTOS.

Quero expressar meus agradecimentos aos Professores Dr. Maurício Prates de Campos Filho e Dr. Amauri Garcia, por seu encorajamento, pelo auxílio e pelas úteis sugestões com que contribuiram durante a confecção deste trabalho. Desejo, também, agradecer aos membros da Banca Examinadora, Professores Dr. Eduardo Alvarenga, Dr. Odmar Simões Pires e Dr. Rezende Gomes dos Santos, por seu interesse em meu trabalho e pelos diversos comentários que fizeram.

Agradeço ao meu colega e amigo Dr. Irany A. Azevedo pela leitura e pelas críticas de algumas partes deste trabalho.

Agradeço aos Professores Fredymarck G. Leão e Ulderico Mandolesi pela oportunidade de realização do curso e do trabalho, que teve auxílio e suporte financeiro concedido pelo Programa CAPES-PICD.

DEDICATORIA.

Dedico esta tese à minha família

Heloisa

Ione

André

pela paciência com que suportaram

este transe.

SUMARIO

O problema considerado neste trabalho está relacionado com a implementação de computadores e controladores no controle de processos industriais, em cuja arquitetura prevê-se a inclusão de meios de diagnose e de tratamento de faltas que podem ocorrer durante a fabricação e durante a operação do sistema, atribuindo-lhe características de tolerância a faltas. O objetivo é a utilização plena de recursos de projeto e de construção de computadores tolerantes a faltas para controle industrial, dentro de certos critérios de limitação desses recursos.

No capítulo I, discute-se a obtenção de dados e critérios bem como o estabelecimento de modelos do problema. São estabelecidas definições claras dos diversos elementos necessários para decidir se, onde e como implementar a tolerância a faltas. Uma discussão sobre a terminologia e algumas sugestões são incluidas no capítulo II, para melhor definição de termos em português. Com certa ênfase são discutidos os conceitos de falta, falha e erro, seu diagnóstico e tratamento, bem como as formas de tolerá-los.

Algumas soluções já adotadas em sistemas existentes são apresentadas e discutidas no capítulo III. Embora aplicadas, em geral, a situações muito críticas, foram estudadas para se obter uma noção mais clara de suas limitações e incorporar um certo "insight" do problema.

No capitulo III, discute-se alguns aspectos de modelagem e do uso de redundâncias em um sistema. Modelos de Markov são explicados, modelos para sistemas tolerantes a faltas são introduzidos, a análise combinatória de sistemas com redundâncias e modelos de Markov para sistemas com reparo são discutidos.

Para a análise do comportamento de um sistema com caracteristicas de tolerância a faltas e no qual se considere as faltas que podem ocorrer durante a sua fabricação bem como durante sua operação, alguns modelos são desenvolvidos e investigados. No capitulo V, desenvolve-se modelos matemáticos para a análise do comportamento de sistemas como acima exposto, considerando que o processo a ser analisado é estocástico em sua natureza, e muito complexo. A escolha de modelos de Markov pode ser considerada natural. Um grafo de Markov é utilizado e um modelo de equações diferenciais é estabelecido. Com esses modelos pode-se avaliar o comportamento do sistema em termos de confiabilidade, disponibilidade, ou outros critérios similares. Um modelo geral de Markov é proposto.

Na modelagem, considera-se que faltas de fabricação podem incluir falhas de componentes que ocorrem durante o processo de difusão ou outras falhas que ocorrem na fabricação de circuitos integrados. Faltas também podem ocorrer durante a montage dos circuitos, durante a montagem de subsistemas, e podem incluir, ainda, falhas humanas e erros de projeto (de componentes, de circuitos, de subsistemas). Todas essas falhas são consideradas como faltas de fabricação. Por outro lado, faltas de operação podem incluir falhas de componentes ou subsistemas (faltas

de hardware), erros de programação (faltas de software), erros de operação, erros de manutenção, erros de manuseio.

Alguns exemplos de aplicação e estudos de casos são apresentados no capítulo VI, acompanhados de uma discussão dos resultados. Conclusões são apresentadas no capítulo VI. Diversas sugestões para futuros trabalhos são também apresentadas no Apêndice A.

ABSTRACT.

The problem investigated in this work is related to the use of computers and controlers in industrial process control, considering that faults may occur in the system during the manufacture and during the operation of the system. The means used for diagnosis and processing of this kind of faults, and the caracteristics of fault-tolerance of the system are also discussed. The target of the work is to enhance the perspectives for design and construction of such fault-tolerant machines, for use as controlers in industrial processes, considering some criteria for the limited resourses.

In chapter I, data and criteria for the design of a fault-tolerant computer are presented, and the possibilities of modeling the problem. A review of definitions and of the elements necessary for decisions like if, where and how to implement fault tolerance, is made. A short discussion about the terminology and some sugestions are included in chapter II for a better definition in portuguese. Some enphasis is made in relation to the definitions of the concepts: fault, failure and error, the diagnosis and the means to tolerate them.

Some actual solutions are presented in chapter III. These cases are usually applied to very critical situations but, in order to clear the limitations of this solutions and to acquire some insight and skill for the proposition

of new solutions of the problem, it was decided to include them.

In chapter IV, some modeling aspects and the use of redundance in a system are presented. Markov models are explained, mathematical models for fault-tolerant systems are introduced and combinatorial analysis of redundant elements and Markov models for systems with repair are discussed.

In order to analyse a fault-tolerant system, where faults may initiate during the manufacture of the system and during the operation of the system, some models are investigated. In chapter V, mathematical models are developed for the analysis of the behavior of a system like above explained, considering that the processes to be analysed are stochastic in its nature, and very complex. A Markov graph model and the corresponding diferential equations model are established. With this kind of models one can evaluate the system behavior in terms of reliability, availability, or other similar criteria. A general Markov model is proposed.

In the modeling, is considered that manufacture faults may include component failure during the difusion process or other failures that occur in the integrated circuit manufacture. Failures may initiate during the circuit assembly, or subsystems assembly, and may include human failures and design errors (of components, circuits, subsystems). All this failures are considered as manufacture failures. Operating faults on the other hand, can include component and subsystems failure (hardware failures), programming errors (software failures), operating errors, repairing errors, handling errors.

Some applications and case studies are made in chapter VI, and the results are discussed. Conclusions are presented in chapter VII. Many future work suggestions are presented and discussed in Apendix A.

INDICE

CAPITULO I

1 - INTRODUÇAO

1.1 - O PROBLEMA DO CONTROLE INDUSTRIAL

Antes da introdução dos circuitos LSI (Large Scale Integration) no mercado, especialmente do microprocessador, a instalação de um sistema de processamento em certas aplicações requeriam demasiado investimento de capital. Em virtude do alto custo, o uso de redundâncias em computadores ou de computadores redundantes constituia uma técnica disponivel apenas para aplicações especiais, muito criticas. Nos últimos anos, o projeto de computadores tolerantes a faltas tornou-se mais viavel e extensivo, sendo aplicado agora em casos onde antes não se aplicaria nem mesmo um computador sem essa caracteristica.

Os sistemas de controle industrial por meio de computador devem obedecer determinadas especificações de confiabilidade e de disponibilidade, isto é, devem realizar as tarefas assinaladas pelo maior tempo possivel e sem apresentar falhas que possam interferir no processo controlado. As falhas devem ser identificadas com a maior brevidade de modo que a reparação seja realizada rapidamente e, na maioria dos casos, sem desligar o processo.

Para aumentar a confiabilidade e a disponibilidade do computador de controle, é muito importante escolher adequada-

mente as técnicas de tolerância a faltas que devem ser utilizadas e as capacidades do sistema tanto de hardware quanto de software. O tempo necessário para a deteção e o reparo de falhas, os tipos e modos de faltas que se deseja tolerar, os mecanismos adotados para a tolerância a faltas e outras considerações tem grande influência sobre o projeto [107].

1.1.2 - O CONTROLE ANALOGICO

Antes da introdução dos computadores digitais como elementos de controle, os sistemas de controle industrial foram afetados essencialmente por tres fatores:

1.- Os sistemas de controle analógico utilizaram controladores analógicos de uma malha formando grupos; técnicas de melhoria da confiabilidade se restringiam aos controladores.

2.- O controle em uma só malha sendo mais simples permitia a utilização de controle manual no caso de falha de um controlaor.

3.- Não havia a pressão de métodos alternativos que incentivassem maior grau de confiabilidade ou melhor performance dentro desta classe de instrumentação.

Com a introdução do computador digital como elemento de controle, e com a incorporação de instrumentação digital, mais precisa que a analógica, sistemas de controle com mais de uma malha fechada puderam ser implementados. Entretanto, em certos casos, o sistema de controle além de incorporar um computador digital manteve o controle analógico de modo

ficou com três niveis: o operador, o sistema analógico e o computador digital.

1.1.3 - CONTROLE DIGITAL.

Em empresas de manufatura onde são implantados processos de fabricação por meio de células contendo máquinas operatrizes de controle numérico e outros meios automatizados de produção, são usados atualmente sistemas hierárquicos de controle bastante complexos [79, 99, 124]. Embora exista certa variedade de organização, estas estruturas hierárquicas tem sido subdivididas em niveis, em número variavel de dois até cerca de seis, dependendo da complexidade do sistema controlado.

O nivel 1, o mais próximo do processo, poderá ser distribuido e associado a diferentes tarefas, o mesmo podendo ocorrer em niveis acima porém em menor escala. A estrutura tenderia, assim, a ser descentralizada nos niveis mais próximos ao processo e mais centralizada nos níveis superiores de supervisão e gerência da fábrica.

Os seguintes fatores são considerados como pontos de partida para enfatizar que a confiabilidade será elemento vital no campo da automação industrial [160, 58]:

1.- Em relação aos custos de outros equipamentos de uma fábrica (planta), os custos do hardware de processamento são cada vez menores.

2.- Uma vasta capacidade de cálculo e de memória

podem ficar disponiveis para controle e automação de qualquer tarefa da planta, supondo a existência de algorítmo adequado.

3.- A capacidade de processamento disponivel pode também executar a monitoração e o teste de outros recursos do sistema.

4.- O projeto e a realização de computadores e controladores com utilização de técnicas de tolerância a faltas para aplicações industriais, deve ser um tanto mais simples que os aplicados a casos críticos envolvendo equipamentos aero-espaciais, considerando que a manutenção está disponivel e pode-se substituir elementos ou módulos que falham. Entretanto, um sistema industrial pode ser muito grande e uma hierarquia de computadores de diferentes dimensões e capacidades pode ser necessária para realizar as diversas operações.

Estes fatores orientam a organização do controle hierárquico com pequenos computadores realizando as tarefas de controle do nivel mais próximo da máquina ou equipamento, até computadores de maior porte que realizam as tarefas mais gerais.

Por outro lado, deve-se ainda considerar que há quinze ou vinte anos, um computador de porte medio tinha uma capacidade (memória, velocidade, periféricos, etc.) comparavel à de certos microcomputadores de hoje. Isto quer dizer que as tarefas de mais alto nivel de uma instalação industrial de alguns anos atrás podem hoje ser realizadas por um microcomputador relativamente simples e barato. A mesma estrutura computacional, embora de portes diferentes, pode realizar as tarefas correspondentes a mais de um dos niveis hierárquicos e, portanto, alguns des-

tes níveis deixam de ter razão de existir.

Com base nestas idéias pode-se chegar à proposição de que não apenas as grandes instalações podem ser controladas por computadores digitais. Um controle por computador pode ser aplicado, e com muitas vantagens, a processos simples e a pequenas instalações ou empresas, a exemplo do que já ocorre na parte administrativa dessas empresas. Um ou vários microcomputadores podem ser usados para realizar as tarefas daqueles níveis e, neste último caso, a técnica de redundâncias como mecanismo para a tolerância a faltas pode ser adotada.

A fábrica do futuro será automatizada de uma forma flexível e integrada por computadores e a produção integrada representa o núcleo da inovação na produção. Um exemplo da complexidade dessa integração pode ser visto na instalação IMPACT, realizada em uma mostra [6]. No futuro imediato, os seguintes pontos básicos podem ser identificados, com relação aos meios de produção:

- precisam tornar-se cada vez mais flexíveis,
- precisam tornar-se integráveis através de sistemas distribuídos de computadores.
- precisam apresentar sempre maior confiabilidade e disponibilidade.

Essas necessidades demonstram que o projeto e a construção de computadores e controladores para processos industriais, em particular os de manufatura, exigirão arquiteturas tolerantes a faltas.

1.2 - DEFINIÇAO DO PROBLEMA ABORDADO.

Sistemas tolerantes a faltas vem se tornando uma opção viável em aplicações a controle de processos. Utilizando lógica redundante e software para detetar e contornar faltas do processador, das interfaces de entrada e de saida e demais módulos que compõe um computador ou um controlador, esses sistemas podem ainda produzir informações de diagnose para orientar e facilitar sua manutenção. O resultado é um equipamento de elevada disponibilidade, caracterizado por um tempo médio entre falhas (MTBF) muito grande e por um tempo médio de reparo (MTTR) muito pequeno [98].

Os primeiros sistemas tolerantes a faltas foram extremamente dispendiosos, devido ao alto custo envolvido em seu projeto (muitas vezes único), no aumento do número e complexidade dos subsistemas associados à verificação de faltas e recuperação do sistema e, ainda, pela redundância de componentes e subsistemas. Sua aplicação apenas podia ser justificada em casos muito críticos.

Atualmente, um amplo espectro de possíveis aplicações é disponível, em virtude de dois fatores: o decréscimo dos custos e o aumento das capacidades dos microprocessadores e de circuitos associados, resultado da evolução tecnológica VLSI. Não apenas os preços dos dispositivos e equipamentos necessários às funções relacionadas à tolerância a faltas tem diminuido, como a

confiabilidade desses mesmos dispositivos e equipamentos tem aumentado e, com isso, reduz-se a necessidade de equipamentos separados de segurança, paineis de controle e unidades duplicadas, cujas funções podem ser incorporadas aos processadores.

Importantes inovações tem ocorrido em duas escalas diferentes de integração VLSI: na escala de circuito integrado propriamente dito ("chip") envolvendo arquiteturas multielementos, e na escala de placa ("wafer") incluindo arquiteturas integradas. Esta última apresenta o potencial de realizar-se a integração de sistemas completos de multiprocessamento em uma única placa, eliminando as etapas de separação e encapsulamento dos "chips". A interconexão, além disso, passa a ser interna em vez de ser através de circuitos impressos, o que diminue os retardos dos circuitos e melhora a confiabilidade [43, 82]. Estruturas tolerantes a faltas integradas em VLSI tem sido propostas e projetadas. Em memórias, por exemplo, integra-se códigos de deteção e correção de erros, havendo já projetos com códigos DEC-TED (correção de dois erros - deteção de tres erros) [130, 148]. Considerações sobre o projeto de microprocessadores tolerantes a faltas e computadores em um único circuito integrado, tem tido sua viabilidade técnica e comercial demonstrada [117, 132, 73, 101, 44, 55].

A tolerância a faltas em VLSI implica na integração de circuitos redundantes e de teste, por meio dos quais as desejáveis propriedades de tolerância a faltas possam ser utilizadas, e sejam alcançados dois objetivos: verificação da operação adequada do sistema e habilidade para permanecer operacional em

presença de faltas, possivelmente com alguma degradação de sua perfomance [161,109, 84]. Estudos recentes focalizam as inovações em arquiteturas VLSI, especialmente as realizadas por interconexão de um número elevado de elementos, incluindo redundâncias de meios em diversos níveis e englobando processadores, memórias, circuitos de comutação, circuitos de teste autotestáveis, comunicações, etc. [139, 128, 91, 152, 82]. Na proposição dessas arquiteturas um fator deve ser considerado: o índice de quebra (porcentagem esperada de circuitos com algum defeito, originado durante os processos de fabricação) que deve diminuir. A minimização das faltas de fabricação e de teste constitue uma solução. No entanto, ela é difícil de implementar, se não impossível, e muito onerosa na medida em que cresce o número de elementos integrados e suas dimensões diminuem. A incorporação de redundâncias revela-se uma solução viável, já demonstrada na prática. A operação confiavel do sistema, torna-se cada vez mais dificil de obter com o crescimento do número de elementos interconectados onde, de modo similar, cresce o número de faltas. A redundância de elementos, também aqui, pode aumentar a confiabilidade (ou outras formas de avaliaç~ao da performance como a disponibilidade.

Procedimentos de tolerância a faltas baseiam-se no diagnóstico e no tratamento de faltas pressupondo-se a existência de meios apropriados de redundância [149, 78]. Nesta introdução descreve-se as medidas necessárias e as alternativas fundamentais de procedimentos relativas à tolerância a faltas em sistemas. A discussão das vantagens e desvantagens de

diferentes classes de procedimentos indica que a sua adequação depende, fortemente, do tipo de falhas a serem toleradas e dos tempos admissíveis de resposta [107, 19].

1.2.1 - NECESSIDADE DA TOLERANCIA A FALTAS.

A tolerância a faltas tem sua origem no esforço de se atingir alta confiabilidade em um sistema, usando subsistemas de baixa confiabilidade [110, 112]. A necessidade de tal melhoramento da confiabilidade existe sempre em instalações cujo mau funcionamento é temido e não pode ser aceito pois estão em jogo a vida humana, prejuizos materiais substanciais ou atividades cuja interrupção não é conveniente [11]. O propósito da tolerância a faltas é oferecer uma solução alternativa para o problema da faltas de um sistema, no qual, a deteção de faltas e a reposição do sistema em operação normal, são realizadas como funções internas do próprio sistema. A tolerância a faltas é o único atributo de um sistema digital que torna possivel a continuação da operação do sistema, dentro do comportamento especificado, após a ocorrência de faltas. Em sistemas de computação para aplicações comuns uma disponibilidade suficiente é atingida com computadores modernos e sistemas de programação comprovados, sem a utilização de medidas especiais adicionais. No entanto, exigências de alta confiablidade, podem forçar o emprego de procedimentos de tolerância a faltas, especialmente em tarefas de automação que precisam ser cumpridas sem a ação direta do homem, desde que não ocorram prejuizos sensiveis em relação aos

resultados esperados. Podem-se tolerar certas faltas por longo tempo, muitas ou todas as faltas por algum tempo, mas nunca todas as faltas por todo o tempo. Um sistema tolerante a faltas deverá ser especificado tendo em vista essas diferenças [66].

Insuficiente confiabilidade pode prejudicar, por um lado, a segurança, isto é, a conservação do sistema livre do perigo de falhas que provoquem algum dano, para o próprio sistema e para o ambiente em que o sistema opera e, por outro lado, a disponibilidade, ou seja, a media temporal de utilidade do sistema [135]. Por exemplo, exigências de segurança estarão presentes em viagens espaciais [126], assim como na automação industrial [157, 161, 86] Exigências de elevada disponibilidade estão presentes, por exemplo, nos meios de comunicação, onde exige-se que em 40 anos de funcionamento haja, no máximo, 2 horas de parada , contando-se com a manutenção e troca de sub-sistemas, o que resulta na necessidade de utilização de computadores para a comutação e que possuam características de tolerância a faltas [11]. A medida em que produtos complexos como, por exemplo, circuitos VLSI, tem seu custo de fabricação e de operação reduzido, as exigências quanto à sua confiabilidade tem aumentado. O controle de processos e o controle da manufatura em fábricas automáticas são outros exemplos de necessidade da utilização de redundâncias e da tolerância a faltas.

A utilização cada vez maior de computadores comerciais, exige menos a disponibilidade continua do sistema e mais a garantia da integridade dos dados, ou seja, a preservação de conjuntos de dados que concordem com a realidade. Mesmo na

ocorrência de quebra, por curtíssimo tempo, dos meios de armazenamento ou, ainda, apesar de entradas errôneas do operador, os dados armazenados não podem ser adulterados, devido a sua significância econômica para a empresa. Por isso, entre os sistemas oferecidos no mercado de sistemas com características de tolerância a faltas, dominam os sistemas comerciais de processamento. Além disso, considerações de rentabilidade exigem o emprego de tolerância a faltas para atingir uma duração maior do funcionamento sem manutenção. O fato de que um sistema de computação pode continuar funcionando depois de aparecer uma falta, significa que existe maior número de soluções para a execução da manutenção corretiva [13].

Aplicações que preveem a instalação de computadores em locais de difícil acesso ou mesmo sem acesso, forçam um funcionamento sem manutenção, por toda a duração da missão como, por exemplo, em espaçonaves. Computadores industriais serão, evidentemente, mais simples, porque a manutenção é possível e módulos que falham podem ser substituídos. Entretanto, um sistema industrial pode ser muito complexo, a manufatura integrada é uma opção realizável e um conjunto de computadores de diferentes capacidades e hierarquias pode ser necessário [160]. Dessa forma, justifica-se a implementação de sistemas controladores em cuja arquitetura sejam incorporadas características de tolerância a faltas. Os benefícios que podem advir da aplicação dessas técnicas a equipamentos utilizados na indústria terão de ser balanceados com os aumentos de custo e de complexidade.

1.2.2 COMPUTADORES TOLERANTES A FALTAS

O custo ou os prejuizos que uma falha pode causar envolve, em primeiro lugar, as perdas associadas à não execução da tarefa (ou execução incorreta). Por outro lado, podemos colocar em questão os custos e prejuizos causados pela manutenção defeituosa ou erronea. O emprego de uma arquitetura tolerante a faltas pode, assim, ser determinada por meio de critérios que envolvem não as causas em si mas sim pelos prejuizos advindos de sua não utilização. Alguns tipos de aplicação estarão vinculados a este critério:

1. Alta confiabilidade.

O sistema, como um todo, não pode falhar, embora algumas de suas partes possam ficar inoperantes temporaria ou permanentemente. Sistemas de energia, sistemas telefonicos e outros grandes sistemas com grande disponibilidade de recursos, são altamente confiaveis [85].

2. Processamento de sinais.

Supercomputadores constituem um exemplo tipico de arquitetura de computadores com elevada capacidade e velocidade de processamento. Sua elevada complexidade exige alta performance e, portanto, necessita mecanismos de tolerância a faltas [42].

3. Computação crítica.

Os sistemas de controle em tempo real, nos quais uma falha devida a erros de computação pode causar graves danos ao equipamento ou mesmo à vida humana, provavelmente são os mais críticos do ponto de vista de tolerância a faltas. Não só os cálculos executados devem ser exatos como demoras excessivas podem ser também prejudiciais. Aplicações na indústria aero-espacial, sistemas de transporte, hospitais e fábricas constituem exemplos deste crescente mercado [10, 95, 105].

4. Longa vida.

Sistemas que nunca requerem qualquer manutenção, tal como naves espaciais automáticas, exigem computadores em que as redundâncias são de tal ordem que, por toda a vida do equipamento não ocorrerá redução de sua performance. Sistemas deste tipo podem ou não realizar computações críticas e podem ou não ser reconfigurados à distância [52, 126].

5. Manutenção não crítica.

Em certos casos, os custos de manutenção de um equipamento podem tornar-se excessivos ou até maiores que o custo total do equipamento. Equipamentos de campo, em especial equipamentos militares ou os instalados em locais distantes e de difícil acesso, fábricas automáticas constituem exemplos de aplicações onde, após a ocorrência de uma falta, o computador deve continuar a operar normalmente até que uma nova sub-unidade possa estar disponível para a substituição [5, 13].

Enquanto os quatro primeiros tipos de aplicações tem recebido grande atenção do ponto de vista da tolerância a faltas, muito menos tem sido realizado quanto ao quinto tipo relacionado [125].

Além dos inúmeros outros aspéctos a serem considerados, a modularização será um dos itens importantes para a implementação de tolerância a faltas em equipamentos onde a manutenção poderá ser adiada.

No projeto de computadores com tolerância a faltas, quatro implementações distintas podem ser encontradas:

1.- Deteção de faltas: compreende o hardware e software usados para a verificação da existência das faltas;

2.- Diagnose de faltas: engloba hardware e software capazes localizar e identificar as faltas;

3.- Confinamento de faltas: compreende as técnicas usadas para evitar a propagação de informações contendo erros causados por uma falta, antes que essa falta seja detetada;

4.- Recuperação de faltas: corresponde aos mecanismos utilizados para a correção da falta, seja por meio de circuitos de voto, seja pelo uso de sobressalentes.

Para a realização dessas implementações, nos últimos anos houve um crescente desenvolvimento de diversas técnicas, entre as quais devemos citar como muito importantes as seguintes:

1.- Circuitos auto verificadores;

2.- Memórias contendo circuitos tolerantes a múltiplas faltas;

3.- Circuitos de geração de "clock" de alta confiabilidade;

4.- Circuitos de comunicação de alta confiabilidade;

5.- Técnicas de codificação para deteção concurrente de faltas especificas.

Essas técnicas (e outras) podem ser usadas de diferentes modos, de conformidade com os requisitos de tolerância a faltas, a tecnologia usada ou disponivel, os requisitos de espaço, de custos, e inúmeros outros aspéctos. Sua incorporação no sistema pode-se dar na forma hardware, isto é, pela inclusão de subsistemas redundantes de hardware, e na forma de software, ou seja, pela inclusão de programas redundantes ou contendo redundâncias. Uns e outros representam elevação de custos de fabricação. Quanto mais cedo forem incorporados e quanto mais baixo o nivel de sua incorporação, tanto menores serão esses custos. Os resultados econômicos de uma produção mais confiável deverão compensá-los.

1.2.3. - BREVE HISTORICO DA EVOLUÇAO DE COMPUTADORES TOLERANTES A FALTAS.

Antes dos anos 40, a confiabilidade havia sido considerada apenas como um fator qualitativo e subjetivo ou intuitivo. Os conceitos quantitativos apareceram durante a segunda guerra mundial [40]. No entanto, desde a primeira geração de computadores a confiabilidade, assim como métodos de deteção de faltas, manutenção preventiva, paradas provocadas por faltas transitórias tem sido tópicos inerentes ao seu projeto, aparecendo em várias publicações a partir de 1947. O primeiro computador tolerante a faltas de que se tem notícia, denominado SAPO, foi construido na Checoslováquia entre os anos de 1950 e 1954, tendo sido utilizadas técnicas de triplicação e voto em todos os circuitos da máquina, em que se empregou tecnologia de relés [11]. O sussessor deste foi o computador EPOS, construido com válvulas a vácuo e núcleos magnéticos, e que também continha caracteristicas bastante completas de tolerância a faltas.

Nos anos seguintes, com o desenvolvimento da tecnologia eletrônica, a confiabilidade dos componentes em geral e, em especial dos componentes digitais, aumentou de tal forma que os padrões de confiabilidade dos equipamentos passaram a ser atingidos sem a necessidade de que a tolerância a faltas estivesse incluida no sistema. A deteção de erros permaneceu apenas relacionada a equipamentos periféricos de menor confiabilidade, mais especialmente aqueles que envolviam dispositivos eletro-

-mecânicos. Apenas bits de paridade permaneceram em uso e, mesmo estes, apenas em algumas máquinas de segunda geração. O conceito de tolerância a faltas incluído no computador desapareceu do vocabulário dos projetistas e os efeitos desta mudança de atitude ainda podem ser sentidos em computadores de gerações mais avançadas como, por exemplo ILLIAC IV e CRAY-1. Também nos microprocessadores desenvolvidos na década de 70 esta característica não está incluida. No entanto, nessa mesma década, diversos computadores com essa capacidade foram desenvolvidos, muitos dos quais apresentados por empresas comerciais [11, 72, 73, 136].

1.2.4 - OBJETIVOS DA TESE.

Tendo em vista as considerações feitas acima, pode-se estabelecer os seguintes objetivos principais deste trabalho:

1. Estabelecer modelos para a análise e avaliação da implementação de técnicas de tolerância a faltas em arquiteturas de computadores, nos quais se considere a ocorrência de faltas de fabricação e faltas de operação nos subsistemas;

2. Estabelecer modelos que permitam a inclusão de características especificas dos diferentes tipos de subsistemas que deverão constituir o sistema;

3. Estabelecer as equações matemáticas dos modelos, de modo a permitir o estabelecimento de comparações entre diferentes configurações que possam ser adotadas em projetos de

arquiteturas;

4. Levando em conta que os modelos elaborados permitem fazer hipóteses de ocorrência de faltas durante a fabricação, mostrar a aplicação das soluções obtidas a casos onde se inclua essa possibilidade bem como a possibilidade de se efetuar reparos de subsistemas ainda antes da entrada em serviço do sistema;

5. Mostrar que as soluções obtidas podem ser aplicadas a casos onde se deseja comparar subsistemas com redundâncias em paralelo e com redundâncias com sobressalentes, tendo em vista que o aumento da confiabilidade cresce com o número de redundâncias mas tende a uma saturação ou passa por um máximo;

6. Mostrar que, apesar da relativa complexidade resultante dos modelos, sua aplicação a problemas específicos de projeto e implementação de computadores para controle de processos industriais, é uma técnica vantajosa, tendo em vista a utilização mais adequada de recursos.

CAPITULO II

SOBRE A TERMINOLOGIA.

Propostas de terminologia no campo da tolerância a faltas, baseiam-se na tecnologia de confiabilidade. Esta tecnologia está definida, em muitos paises, em normas técnicas, que se referem a componentes e, por isso, tratam o problema de tolerância a faltas de modo bastante incompleto. Nos Estados Unidos, o tratamento de computadores tem certo destaque, mas também não existe um padrão geralmente aceito. No Brasil a normalização é reduzida e a terminologia é estabelecida com base em normas ou autores estrangeiros, quando são aplicáveis. Do ponto de vista da informática, propostas de terminologia feitas até agora, se referem muito pouco ao fato de que os sistemas de computação somente podem funcionar em conjunto com seu software e, por isso, quando se fala de tolerância a faltas, precisa-se fazer referência a hardware e a software separadamente. Uma proposta da IFIP (Grupo de Trabalho 10.4 - "Reliable Computing and Fault Tolerance") procura respeitar todos os aspectos relativos aos sistemas computacionais, de uma forma ponderada [9].

A definição obrigatória dos termos deixa entrever um uso não muito homogêneo nas publicações de língua inglesa, de modo que o conceito da IFIP será sucintamente explicado. Está sendo proposto, como termo chave, o termo "dependability" (fide-

dignidade) deixando para o termo inglês "reliability" o aspécto quantitativo da probabilidade de sobrevivência. Ao mesmo tempo, propõe-se estruturar todo o campo de termos técnicos relativos a tolerância a faltas, a partir deste novo ponto de vista. O termo "dependability" engloba, segundo a figura 1, a descrição de insuficiências de um sistema, dos meios de tolerância a faltas e das caracteristicas de confiabilidade como probabilidade de sobrevivência e disponibilidade. O termo confiabilidade tem sido utilizado com seu significado no aspécto quantitativo, mais ligado a padrões percentuais e relacionado a tempos especificos. Fidedignidade deve ser usado como termo chave, mais abrangente.

A fidedignidade e, evidentemente, a confiabilidade podem ser prejudicadas quando existem insuficiências na forma de:

. faltas (faults) , que indicam um estado (lógico) não desejado de um subsistema ou que não cumpre as especificações, e que pode provocar erros nas operações conseqüentes de sistema sem falhas;

. falhas (failures) , que descrevem o evento (fisico) de que a função de um subsistema é necessária mas não é realizada corretamente; é uma alteração dos valores dos parâmetros do sistema e provoca uma falta;

. erro (error) , que descreve o evento de que uma função é realizada incorretamente, devido a uma entrada incorreta, ou provoca uma seqüência incorreta de eventos [9, 11].

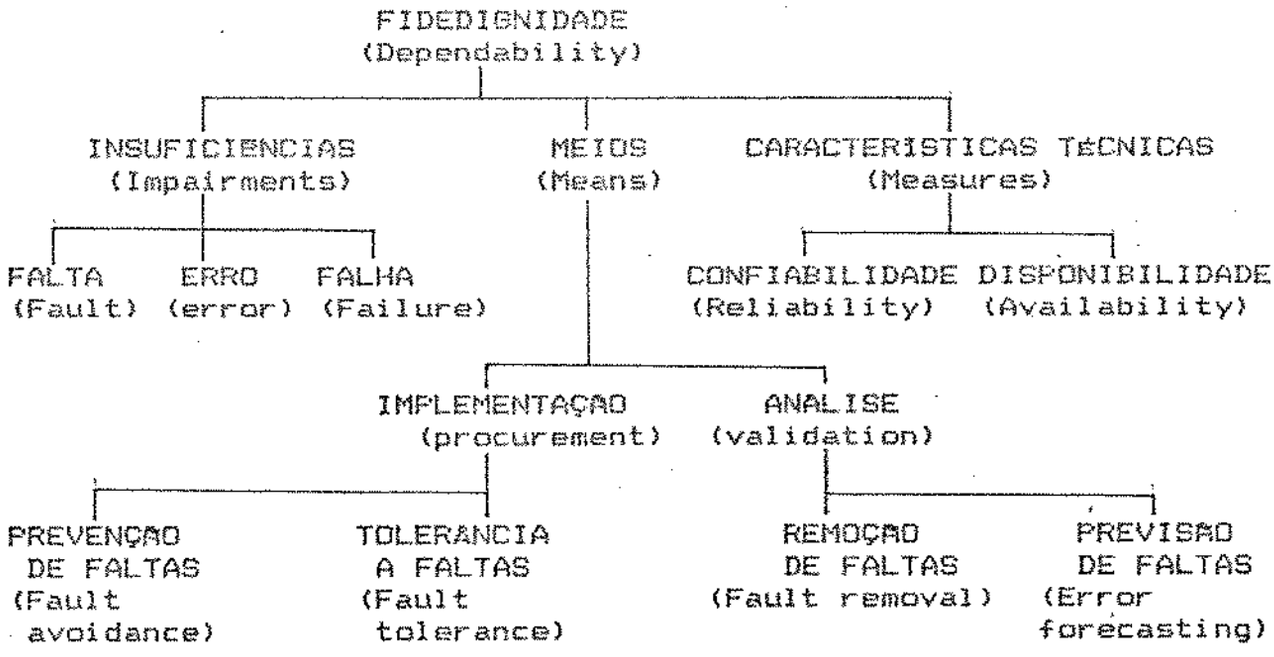


Figura 1. Sistemática na terminologia no ambiente da tolerância a faltas, proposta pela IFIP.

Falta é o termo que descreve uma causa de falha. Falha é o termo que descreve um mau funcionamento, ou seja, a realização de uma função fora das especificações. O ramo central da figura 1 mostra as duas formas de realização dos meios, isto é, de um lado a prevenção de faltas (englobando hardware e software) e a tolerância a faltas (emprego de redundâncias para a continuaçãp do funcionamento mesmo na ocorrência de faltas). Ambas as medidas se justificam somente quando a análise assegura seu uso no sentido de aumento da fidedignidade. A remoção de faltas mostra que a correção de faltas no projeto resultou de procedimentos corretos de tolerância a faltas. A análise da previsão de faltas deve descrever a possível ocorrência e as

possíveis conseqüências de todas as possíveis faltas, por exemplo, usando a avaliação quantitativa com base em um modelo estocástico.

A figura 2 mostra, em complemento da figura 1, um detalhamento mais profundo dos procedimentos de tolerância a falta. A estrutura mostrada é um detalhamento da proposta da IFIP [9, 136] na qual há uma distinção entre o tratamento de falhas efetivas e latentes.

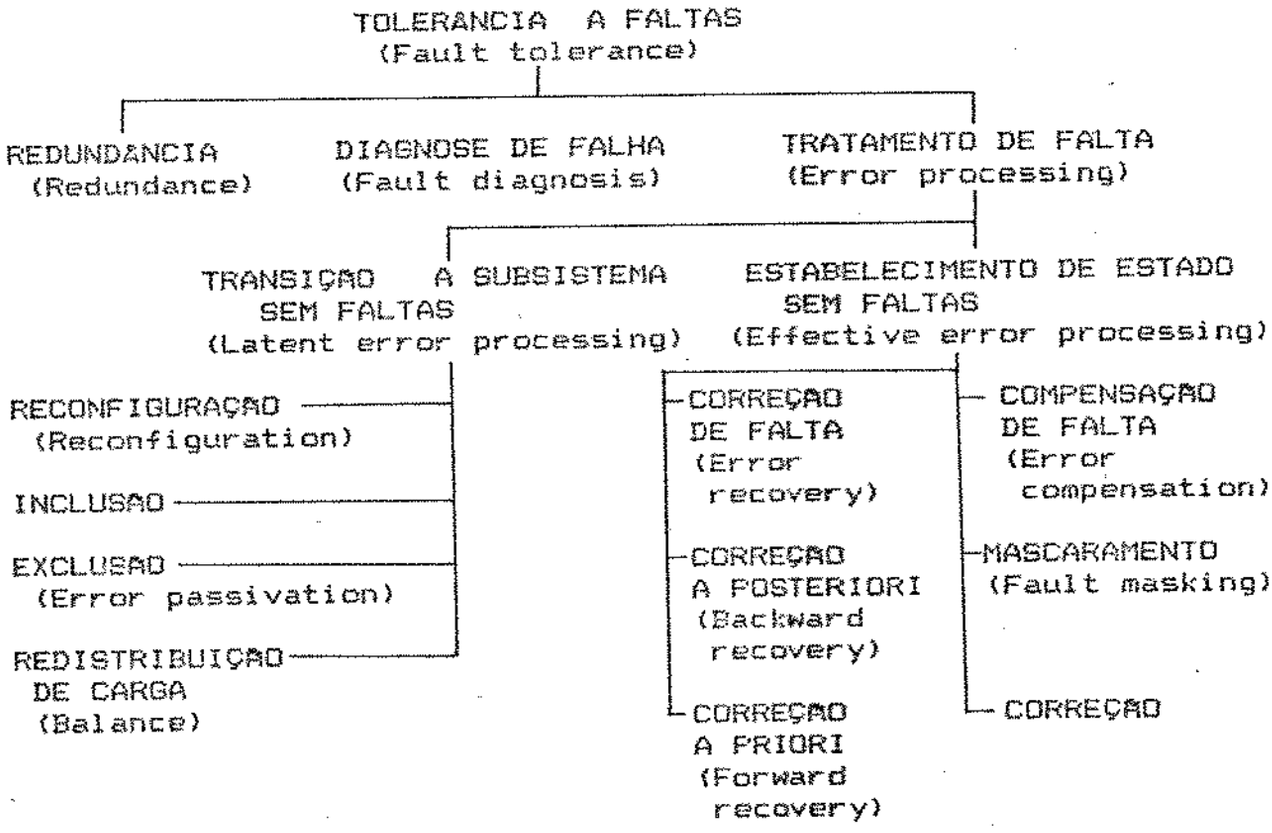


Figura 2. Estrutura dos procedimentos de tolerância a faltas.

Com o objetivo de aumentar a confiabilidade, duas formas de aproximação podem ser utilizadas:

. a evitação de faltas ou intolerância a faltas;

. a tolerância a faltas.

A intolerância a faltas resulta de práticas conservadoras de projeto, usando componentes e subsistemas de elevada confiabilidade, testes exaustivos (burn-in) e cuidadosa verificação, tudo com o objetivo de reduzir a probabilidade de ocorrência de uma falta. Nos projetos tolerantes a faltas, de outro lado, a redundância de meios é usada para compensar a ocorrência de faltas. A redundância pode ser incorporada de tres formas:

. redundância em tempo, que envolve a execução repetida de operações de cálculo, às vezes por métodos diferentes;

. redundância de componentes ou de hardware, em que componentes extras são incorporados para substituir aquele que falha, podendo estar conectado na forma k dentre n, também chamada reserva quente, e na forma de sobressalente (standby), também chamada reserva fria.

. redundância de programas ou de software, em que programas adicionais são incorporados, por exemplo, para comparar resultados efetuados por diferentes programas.

Um sistema com redundâncias, em resposta a uma falta, poderá passar por vários estágios de processamento ou de tratamento de faltas. Todo projeto tolerante a faltas deverá envolver a seleção de meios de processamento de faltas com a combinação de alguns ou de todos esses estágios [136].

Uma descrição sumária desses estágios segue:

. Confinamento - quando uma falta ocorre, o confinamento limita a repercução dos efeitos da falta a apenas uma parte do sistema, evitando a contaminação das outras partes. O confinamento pode ser alcançado por meio de uso de circuitos detetores de faltas, verificações de consistência e múltiplos testes tipo pedido/confirmação, antes de realizar-se uma determinada função. As técnicas podem ser aplicadas em software e em hardware.

. Deteção - muitas técnicas são disponíveis para a deteção de faltas, tais como, paridade, verificação de consistência, violação de protocolos. Não sendo perfeitas, porém, envolvem certo período de tempo antes que a falta seja detetada, chamado período de latência. As técnicas de deteção podem ser "off-line", isto é, o subsistema não pode operar enquanto estiver em teste e "on line", que permite uma detecção em tempo real, ou seja, o sistema continua em operação enquanto está em teste.

. Mascaramento - as técnicas de mascaramento escondem os efeitos de uma falta. A informação redundante se sobrepõe à informação incorreta. Em sua forma pura, o mascaramento não deteta faltas mas pode ser associado a outras técnicas. Circuitos de voto constituem um exemplo típico. O mascaramento é também chamado redundância estática.

. Religamento - uma segunda tentativa (ou mais) pode provocar a operação do sistema, por exemplo no caso de faltas transitórias, desde que não tenha ocorrido dano físico.

. Diagnose - se uma técnica de deteção não fornece suficiente informação quanto à localização da falta ou sobre suas propriedades, a diagnose de faltas poderá ser um próximo passo.

. Reconfiguração - se uma falha permanente é localizada, o sistema pode reconfigurar seus componentes de modo a isolar o componente que falhou e substitui-lo por outro, sobressalente. Por outro lado, o sistema pode ter sua capacidade diminuida ou degradada pela desconexão do componente que falhou. Este processo é chamado degradação suave.

. Recuperação - após a deteção de uma falta e após a reconfiguração do sistema, os efeitos dos erros devem ser eliminados. Muitas vezes o sistema é colocado em um ponto do processo anterior à falha e reiniciado ("rollback"). A latência da falta é importante pois o sistema deve reiniciar o processo de modo a evitar os efeitos de erros não detetados que ocorreram antes da deteção daquele erro em particular.

. Reinicio - a recuperação pode não ser possivel se houver muito dano nas informações ou se o sistema não dispõe dessa modalidade de operação. Um reinicio "a frio" corresponde à recarga completa do sistema, sem nenhum aproveitamento do processamento em curso. Em um reinicio "a quente", o sistema reassume as operações a partir de um ponto de deteção de falta (se não ocorreu nenhum outro dano). O reinício, pode ainda, ser efetuado com perda de parte do processamento.

. Reparo - após a deteção de uma falta, um componente ou um

subsistema é reparado e recolocado no sistema. O reparo pode ser "on-line" ou "off-line". Neste último caso, o componente não é imprescindível à operação do sistema podendo ser removido, ou então, o sistema deve cessar sua operação (corresponde a uma falha do sistema). No reparo "on-line", o componente é substituido por um sobressalente por meio da reconfiguração do sistema, ou a operação pode ser realizada sem o componente e sem a interrupção do processamento (como na redundância estática ou na degradação suave).

. Reintegração - após a substituição de um componente, o módulo reparado deve ser reintegrado ao sistema. No reparo "on-line" a reintegração é realizada sem interromper o processamento.

Na figura 3 mostra-se um cenário que ilustra alguns dos conceitos anteriores. Podemos também observar conceitos que envolvem medidas relativas à confiabilidade: tempo médio

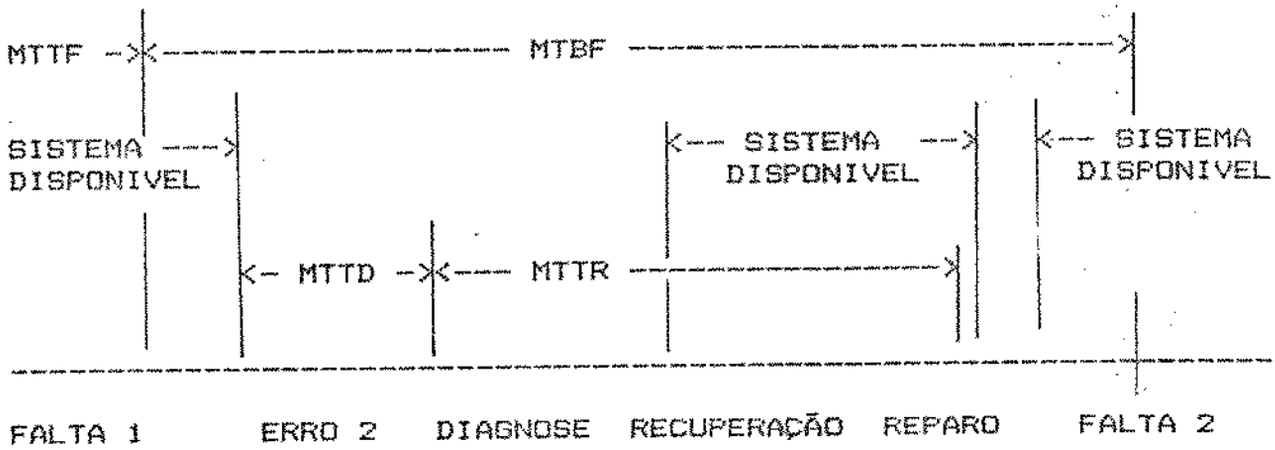


Figura 3. Deteção "on-line" e reparo "off-line".

entre faltas (MTBF), tempo medio até a primeira falta (MTTF), tempo medio até a deteção ou latência de erro (MTTD), tempo medio de reparo (MTTR), e disponibilidade.

Uma proposição de classificação de técnicas de tolerância a faltas, levando em consideração os estágios de resposta do sistema às falhas, como acima definido, bem como as estratégias de resposta, estão mostradas na figura 4. Devido a importância de uma clara definição dos conceitos e uma clara demarcação de limites e de linhas de discussão, o assunto voltará a ser discutido no item em que tratamos especificamente de redundâncias [11, 136].

A deteção de faltas não produz, por si só, a tole-

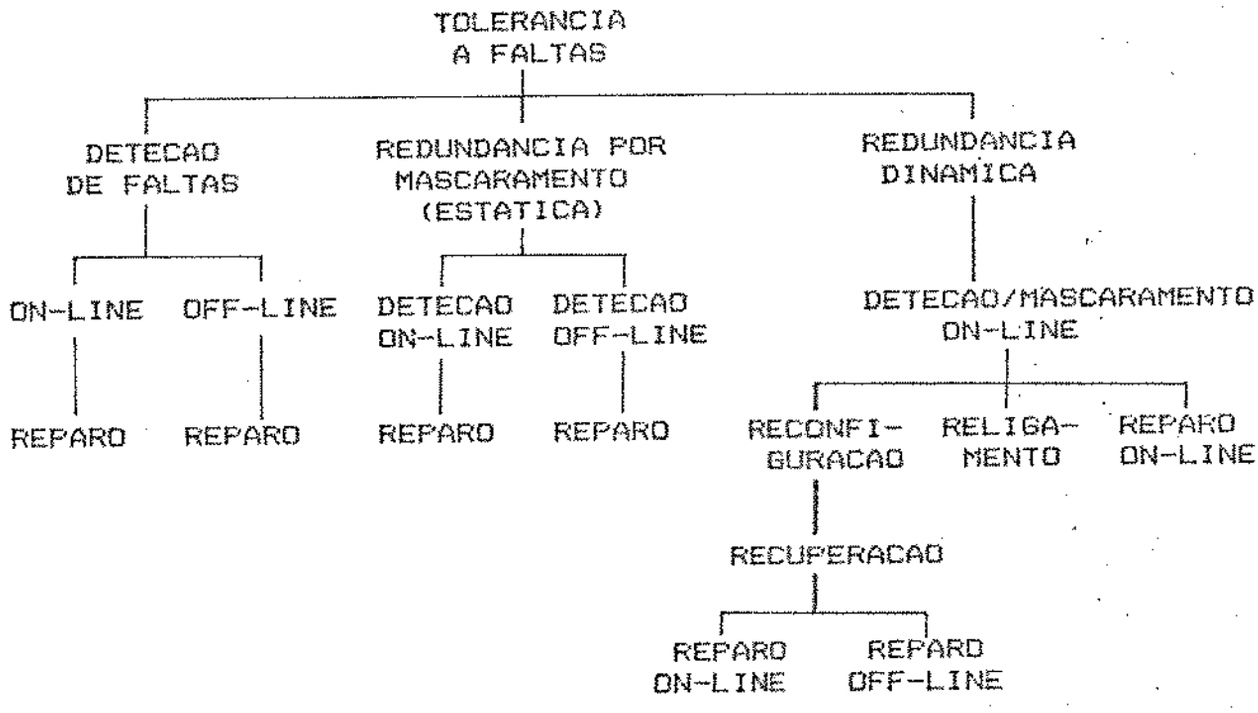


Figura 4. Estratégias de tolerância a faltas.

rância a faltas, apenas produz um alerta quando as faltas ocorrem. A redundância estática ou mascaramento tolera faltas, mas não produz nenhum alerta quando as faltas ocorrem. A redundância dinâmica compreende sistemas cuja configuração pode ser dinamicamente alterada como uma resposta a uma falta. Compreende, também, sistemas em que a redundância estática, suplementada por deteção "on-line", permite reparo "on-line".

2.1 - CAUSAS E IMPORTANCIA DAS FALHAS EM COMPUTADORES.

As aplicações de computadores tem sido cada vez mais diversificadas, abrangendo áreas onde sua operacionalidade é crítica, como por exemplo, controle em tempo real, controle de sistema de transporte e de telecomunicações, controle de naves espaciais, e inúmeras outras. Em particular, a operação de sistemas automatizados ou autônomos de fabricação depende, em grande parte, da disponibilidade dos sistemas de processamento e de controle [13, 117].

O projeto, a construção e a utilização de computadores com elevado grau de confiabilidade, depende de mecanismos de deteção, de diagnóstico, de confinamento e de recuperação de faltas. Porconseguinte, a identificação das fontes e das causas das falhas é essencial. Dentro do contexto de um sistema de processamento de dados, a fidedignidade é um sinônimo de execução correta de um programa, cuja função pode ser o processamento em si, considerado de um modo geral, ou o controle de um processo ou ainda de um sistema de automação, em particular.

Faltas devidas ao mau funcionamento ou à falha de um componente, isto é, de hardware, constituem apenas uma das possiveis causas do mau funcionamento ou da parada completa de atividade de um computador. Assim, previsões baseadas apenas neste tipo de falhas não serão muito reais nem corretas,

geralmente conduzindo a resultados demasiadamente otimistas [46, 69, 88, 93].

Algumas das causas de falhas mais importantes podem ser localizadas em determinadas partes da estrutura ou então provocadas por determinadas fontes de falhas. Um sistema digital como um computador pode ser subdividido em alguns elementos ou subsistemas, mais ou menos essenciais à sua operação:

1.- Unidade Central de Processamento:

Esta unidade contém os circuitos de controle de processamento (registros, contadores, etc.), de cálculo (unidade lógica e aritmética, registros, etc.), de entrada e saida, e de memória. Quando são usados microprocessadores, mono ou multichip, pode-se englobar esses elementos em processador e memória, essencialmente.

Os processadores raramente apresentam falhas. O problema é que quando uma falta ocorre no processador ela provoca uma pane grave. O erro pode ser um só bit trocado na informação em um registro, um endereco por exemplo, e que causará total disfunção do sistema.

Os computadores atualmente possuem circuitos de paridade para deteção de faltas nas memórias (os microcomputadores e microprocessadores de modo geral não os possuem, exceto alguns poucos mais recentes). Falhas em memórias podem causar também graves problemas que os programas dos sistemas operacionais não conseguem contornar.

Tanto os circuitos de processadores como os de memórias são

bastante sensíveis a transitórios de tensão de alimentação que também podem provocar falhas [55, 63, 73, 119].

2.- Equipamentos Periféricos:

Falhas em equipamentos periféricos são relativamente mais comuns mas, de um modo geral, não causam a parada total do sistema. Em geral ocorre uma certa degradação do sistema mas o sistema como um todo ainda pode operar, mesmo que precariamente. Isso se refere ao computador em si. Se, entretanto, o quipamento periférico é a máquina controlada em um processo ou que faz parte do sistema de manufatura, as faltas nos periféricos e em suas interfaces passam a ser tão importantes quanto no próprio computador que as controla [98, 111, 130].

Nestes casos, falhas de origem eletrônica serão tão importantes quanto falhas de origem mecânica, e muitas vezes as interfaces são as partes que maiores problemas apresentam.

3.- Intercomunicação entre Módulos:

Assim como diversos outros tipos de falhas, é aceita geralmente a idéia de que falhas no sistema de comunicação continuarão a ocorrer. Mesmo utilizando códigos para deteção e correção de erros, existem limites práticos, geralmente econômicos, que impedem sua total eliminação.

4.- Software:

Erros de software são de origem humana e podem provocar graves faltas no sistema, especialmente se o computador deve possuir funções críticas e vitais. Do ponto de

vista da confiabilidade, os erros de software devem ser encarados de modo um pouco diferente em relação aos causados pelo hardware. Considera-se que um componente falhou quando suas características ficarem fora das especificações ou fora das margens de operacionalidade. Isso acarreta alterações no hardware e seu colapso em certos casos. Um trecho de software não falha propriamente: um programa não realiza as funções que deveria por que contém algum erro que se manifesta ao ser encontrado por ocasião da execução do programa. Isso não acarreta obrigatoriamente a falha do sistema. O software, além disso, continua o mesmo após o erro ter sido encontrado. As falhas em hardware alteram ou degradam sua estrutura, muitas vezes de modo permanente. Enquanto a redundância em hardware é uma boa medida para a melhoria da confiabilidade, a duplicação de software não produz nenhuma alteração neste sentido pois o mesmo erro estará presente em ambas as cópias e técnicas diferentes deverão ser usadas [88, 93, 103, 106, 118, 123].

5.- Falhas de Causa Comum:

Entre as várias causas de faltas ou de má operação de um sistema, as falhas de causa comum podem levar à graves consequências se não houver alguma forma de proteção. Um operador não habilitado pode provocar falha do equipamento por desconhecer os comandos corretos; outro, habilitado, pode cometer erros de comando; um sistema automático pode falhar. A origem da causa de falta pode ser humana, de hardware ou de software, ou uma combinação delas e que fazem

todo o sistema falhar em vez de apenas parte dele [40, 41].

As origens de falhas de causa comum podem ser reunidas em classes que possuem, por sua vez, natureza comum:

A.- Influências externas provocadas pelo meio ambiente:

1.- Ambiente normal - pó, humidade, temperatura, sujeira, vibração.

2.- Ambiente anormal - fogo, terremoto, enchente, ventos, transitórios.

B.- Problemas causados por erros e descuidos na construção:

1.- Erros de projeto - falhas não consideradas, interdependência de subsistemas.

2.- Erros de fabricação - materiais, processos, manuseio, montagem, especificação.

3.- Fontes de energia comuns - faltas, distribuição ineficiente.

4.- Erros de instalação - ambiente anormal, montagem, local, ajustes.

C.- Problemas causados por erros humanos (após instalação):

1.- Manutenção - ajustes, reparação, inspeção, manuseio, ferramental.

2.- Operação - descuido, ignorância, falta de informações, ambiente prejudicial, motivação.

2.1.1 - TAXONOMIA DA TOLERANCIA A FALTAS.

Existem dois modos de incrementar a confiabilidade em um sistema: a intolerância a faltas e a tolerância a faltas. O primeiro resulta da utilização de componentes de elevada confiabilidade, da eliminação de componentes que apresentem mortalidade infantil durante testes iniciais e outros cuidados que se resumem em evitar a ocorrência de uma falta durante o período normal de utilização do equipamento. Em um projeto de sistema tolerante a faltas, por outro lado, a redundância de meios, é usada para produzir o cancelamento dos efeitos das faltas. A redundância pode ser manifestada no tempo, isto é, tarefas são realiza-

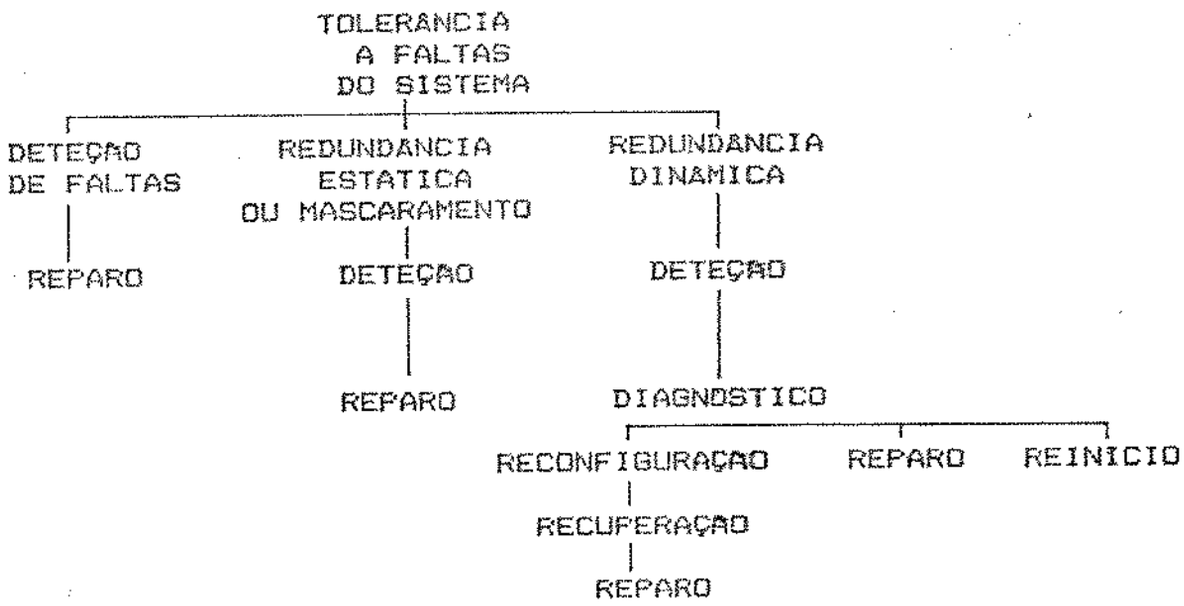


Figura 5 - Estratégias de Tolerância a Faltas.

das mais de uma vez e, talvez, por métodos diferentes. A redundância pode ser também obtida pela multiplicação de componentes ou módulos [7, 8, 136].

Um sistema com redundâncias pode passar por um ou mais estágios, como resposta à ocorrência de uma falha: confinamento de falhas, deteção de falhas, mascaramento, reinício, diagnóstico, reconfiguração, recuperação, reparo, reintegração. Tomando em conta esses estágios, respostas a faltas ocorridas no sistema, estratégias de tolerância a faltas podem ser então classificadas conforme ilustrado na figura 5 e as técnicas principais empregadas para aumento da confiabilidade em conexão com cada modo de intolerância ou de tolerância a faltas estão listados na tabela I [11, 21, 27, 40, 136].

TABELA I - Classificação das Técnicas de Confiabilidade.

ESTRATÉGIA	TÉCNICA
Intolerância	Modificação do ambiente
	Alteração da qualidade
	Nivel de integração de componentes
Deteção de Faltas	Duplicação
	Códigos de deteção de erros (paridade, soma, códigos aritméticos e cíclicos)
	Auto-teste e lógica a prova de falhas
	Temporizadores cão de guarda e limitadores

	Verificação de continuidade
Redundância estática	Voto
	Códigos de correção de erros (Hamming, SEC-DED e outros)
	Mascaramento
Redundância dinâmica	Duplicação reconfiguravel
	Voto reconfiguravel
	Sobressalentes
	Degradação
	Reconfiguração
	Recuperação

2.1.2 - INTEGRAÇÃO COMPUTADOR-MANUFATURA

A configuração da hierarquia de um sistema de computação para uma grande manufatura, formando um sistema integrado "manufatura-computador", é dividida em niveis.

No nivel mais próximo ao processo estão os controladores programaveis ou microcomputadores ou ainda, equipamentos especiais que podem incluir um microprocessador nos circuitos. Esta parte do equipamento tem sofrido inúmeras modificações e ampliações nos últimos anos, incluindo-se uma certa tolerância a faltas tendo sido utilizada redundância em certos casos [145]. A confiabilidade das partes eletromecânicas da máquina operatriz tem se elevado e há uma preocupação maior com as interfaces eletrônicas e os próprios controladores cujos componentes tem melhorado neste aspécto.

Um sistema de controle tolerante a faltas deve realizar pelo menos três coisas [86, 87].

1) O computador ou o seu equivalente deve possuir meios para identificar uma falta. A possibilidade de ocorrerem faltas simultâneas deve ser considerada;

2) o sistema de controle deverá continuar em operação apesar da falta. Claro que, além do sistema de controle, os meios utilizados para o processo de manufatura também devem continuar operacionais;

3) o sistema de controle deve possuir meios para

corrigir o problema. Esta correção, em geral é suposta automática, o que pode não corresponder à solução mais econômica, ou mesmo à melhor solução de uma forma geral. Uma nave espacial não tripulada exige um sistema automático de reconfiguração, altamente confiavel e seguro e extremamente complexo e caro, pois de uma falha pode resultar o fracasso total da missão. Entretanto, em um sistema de manufatura, mesmo totalmente automático, não é esta, necessariamente, a solução mais conveniente.

Alguns fatores inerentes a uma indústria de manufatura permitem estabelecer alguns critérios para a seleção de mecanismos para a tolerância a faltas. Do ponto de vista do processo em si, podemos encontrar diferentes constantes de tempo que implicam em diferentes técnicas de controle, conforme mos-

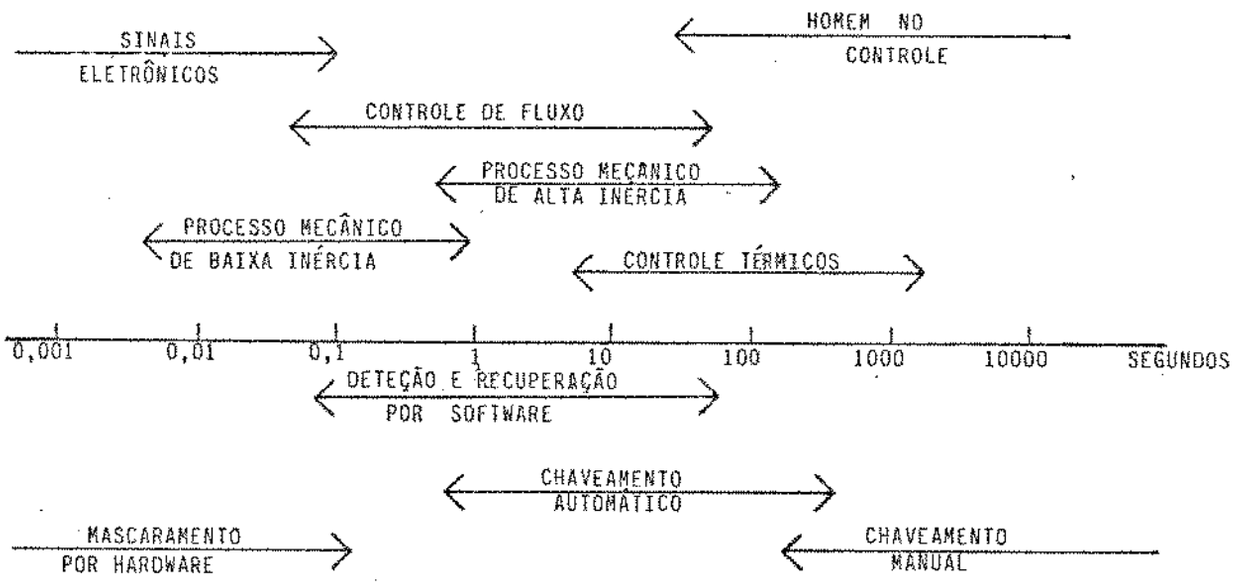


Figura 6 - Constante de Tempo de Processos e Formas de Deteção e Recuperação de Faltas.

tra-se na figura 6 [107, 157, 160].

Os processos mecânicos podem ser subdivididos em dois grupos de tal modo que os chamados processos mecânicos de pequena inércia corresponderão aos processos contínuos enquanto que os de inércia mais elevada correspoderão aos de batelada. Em processos de manufatura e em alguns tipos de processos contínuos ou semi-contínuos como o processamento de alimentos, as constantes de tempo estarão compreendidas entre o décimo ou centésimo do segundo até alguns segundos ou mais. É necessário considerar também que, em diversos casos, o homem estará fazendo parte do processo e, portanto, fará parte da malha de controle.

Com base nestes fatos, os mecanismos de tolerância a faltas poderão envolver hardware e software assim como alguma forma de chaveamento automático para a recuperação.

Devido a importância da velocidade ou rapidez com o processo se realiza, em certas aplicações críticas de controle por computador, as faltas provenientes do sistema de controle não deverão atingir e interferir no processo em si. O sistema de controle deve eliminar ou mascarar quaisquer erros de modo a impedir que possam chegar ao processo.

Alguns métodos ou mecanismos podem ser economicamente mais viáveis, de modo a obter-se as características descritas. Uma arquitetura distribuida de computação e de controle, permite reduzir os riscos de perda total de controle no caso da ocorrência de uma falha de algum componente. Por outro lado, o controle distribuido implica em software mais complexo pois envolve controle de comunicação e manuseio de funções especiais de

reconfiguração, de recuperação [131], e de organização do software bem como do manuseio de funções de tolerância a faltas [158]. Por outro lado, a seleção entre um sistema de controle centralizado ou distribuido é determinada por considerações em que os requisitos para tolerância a faltas constitue apenas uma parte [125].

Um sistema de diagnóstico de falhas bem como os meios para sua correção constituem partes importantes de um projeto [9, 136]. O Tempo Medio Entre Faltas - MTBF e o Tempo Medio Para Reparo - MTTR), também constituem dados importantes em um projeto com inclusão de tolerância a faltas. Laduzinsky [86] comenta opiniões de que, em instalações industriais, controladores e barramentos redundantes são as partes que falham menos e que a maioria das falhas ocorre na instrumentação e nos atuadores.

Recentemente, computadores e controladores programaveis típicos tem sido usados em redundância dupla, já que esta solução revelou-se economicamente viável. Novos computadores especialemnte projetados para esta finalidade esbarram com o problema da viabilidade econômica. Neste último caso, de maneira a sobrepujar qualquer outra consideração, os custos de desenvolvimento de software revelam-se muito altos enquanto que o software existente pode ser adaptado a um sistema de dois processadores

Quando considera-se as constantes de tempo do processo, torna-se claro que a rapidez com que faltas são detetadas e corrigidas pode ser fundamental, como nos processos contínuos, por exemplo (mas não exclusivamente neles). Um sistema

modular triplo ou múltiplo de computadores redundantes permite que cada um tenha acesso imediato aos resultados dos outros dois, podendo realizar comparações a nivel de hardware. Uma vantagem importante destes sistemas é a de que os tempos de intercomunicação entre computadores são pequenos fazendo com que o controle seja disponivel por mais tempo. Um sistema triplo de computação opera corretamente na presença de uma falta, não havendo pausa quando ocorre um erro. A falta é mascarada pelo hardware e não afeta a programação.

Na medida em que o controle de determinado processo deva ser ininterrupto e menos dependente de falhas no sistema todo, outros itens que estão no sistema além dos próprios computadores devem ser considerados. A redundância da informação proveniente de um sensor, assim como a certeza de operação de um atuador, são essenciais. Estender os mecanismos de tolerância a faltas a todos os elementos do processo e integrá-los a um sistema de computação tolerante a faltas será portanto, fundamental nos sistemas de controle de processos e de manufatura do futuro, na qual a tolerância a faltas será uma das mais importantes exigências do projeto.

2.2 - MODELOS DE FALTAS.

O comportamento de um sistema computacional, na ocorrência de faltas, resulta do choque entre os resultados das faltas e das medidas defensivas realizadas, ou seja, medidas de tolerância a faltas. A descrição da capacidade de tolerância a faltas baseia-se, primeiramente, em uma separação de faltas e sistema computacional. A definição dos termos não é baseada na mera tradução dos termos de outras línguas, mas para um melhor entendimento de termos de significado mais ou menos difuso, e que estão abertos à discussão [11, 19, 136].

O modelo de faltas tem como objetivo descrever, em qualquer instante, as insufuciências de um sistema computacional. Por isso, precisa designar os subsistemas atingidos pela falta (observação estrutural de faltas) e deve indicar de que modo suas funções são prejudicadas (observação funcional de faltas). Ambas as formas de observar faltas podem apresentar graus diferenciados de detalhamento. Pode-se considerar como subsistema com falta, componentes discretos, ou módulos mais complexos, circuitos integrados ou mesmo computadores em um sistema multicomputador. A função prejudicada pode ser descrita explicitamente (um resultado atrazado, por exemplo) ou implicitamente (um desvio em uma função específica). Cada perda de função deve-se a uma falta no subsistema que deveria executar essa função, mesmo que esta falta não possa ser localizada, ou cujo conhecimento não é importante para um determinado procedimento de tolerância a faltas.

2.2.1 - TIPOS DE FALTAS.

O significado bem definido e consistente dos termos falta, falha e erro, deve ser dado, em discussões que envolvem tolerância a faltas. Tendo por base que um sistema fonte (F) produz um resultado ou um serviço esperado (S), e que o fornece a um sistema usuário (U), as seguintes definições são aceitas, de acordo com a proposta da IFIP [11, 19]:

. Uma FALHA ocorre quando o usuário (U) percebe que a fonte (F) cessou a remessa do serviço esperado (S). Por exemplo, um temporizador "cão de guarda" (U) percebe que o programa em execução (F) não produziu a zeragem (S) do temporizador, antes do término do prazo.

Diversos tipos de falhas podem ser identificadas, simples, severas, catastróficas, e mais de um nivel de serviço pode ser especificado.

. Um ERRO ocorre, quando alguma parte da fonte (F) assume um estado indesejado, contrário à especificação ou à espectativa do usuário (U). Por exemplo, um erro de comparação, quando dois somadores recebem os mesmos operandos e, simultaneamente, fornecem resultados ao comparador, não havendo identidade em todos os bit.

Erros podem ser considerados (pelo usuário) como uma falha ou como um decréscimo de nivel do serviço. Erros de paridade permanentes serão considerados falha. Um segmento permanentemente apagado de um display pode ser aceito como um serviço ruim.

Um erro permanece latente enquanto não for detetado e não produzir uma falha.

. Uma FALTA é detetada se ocorre uma falha da fonte (F) ou um erro é observado no ambiente da fonte. A causa da falha ou do erro é uma falta. Em geral podem ser identificadas mas, em certos casos, permanecem como simples hipótese a ser verificada. Por exemplo, uma falta física é estar a saida de um bloco lógico grampeada em 1.

Uma falta é latente enquanto não causa erros e permanece como uma causa potencial. Um bloco lógico grampeado em 1 em um circuito integrado sobressalente desligado, é um exemplo típico.

Para complementar, pode-se assumir que uma falha é uma perda de serviço que é percebida pelo usuário (U) nas vizinhanças da fonte (F), isto é, nos pontos em que a fonte é monitorada pelo usuário. Um erro é um estado indesejado da fonte (F) e que existe nas vizinhanças ou dentro da fonte, e que pode ser percebido como uma falha quando é propagado para suas vizinhanças e se manifesta nessas vizinhanças. Um falta é uma causa identificada ou suposta (mediante hipóteses) de uma falha ou de um erro.

As diferenças entre falhas, erros e faltas são determinadas pela localização das vizinhanças da fonte e dos serviços por ela fornecidos. A perda do serviço é percebida pelo usuário como falha; um estado indesejado dentro de uma fonte, provocado por uma falta, é considerado um erro.

Como os recursos (fontes) são subsistemas aninhados, a falta pode ser percebida como falha quando o âmbito ou

vizinhança dos serviços prestados é deslocado para localizações mais internas.

Para finalizar, uma fonte tolerante a faltas pode detetar faltas por meios que reconhecem erros (algoritmos de detecção). A recuperação (algoritmo de recuperação) produz a correção de erros e a eliminação de faltas, se forem permanentes. A presença de faltas dentro da fonte não conduz a falhas nas suas vizinhanças, isto é, o serviço esperado é fornecido correto ao usuário sem a necessidade de preservar a perfeita integridade da fonte.

Para poder-se deduzir as causas das insuficiências de um sistema, que podem ter existência em instantes muito distintos, é necessário observar o ciclo de vida desse sistema. Por meio de modelos de fases pode-se separar os passos de projeto, produção e operação do sistema [7, 8]. Uma implementação de sistema torna-se operacional, no caso de hardware, pela utilização de componentes e subsistemas e, no caso de software, pelo uso de compiladores, carregadores e outros programas [88, 94]. Na fase de operação, que se segue a algumas iterações das fases de projeto, implementação e teste, modificações do sistema podem ser introduzidas para atender exigências de operação ou manutenção [18, 23, 51, 52, 60].

Em cada parte do processo descrito, podem surgir faltas que conduzem, mais cedo ou mais tarde, a uma falha ou a um erro. Distingue-se os seguintes tipos de faltas [9, 11, 19, 136]:

1. Erros de projeto, que ocorrem antes do início da opera-

ção, e que podem ser caracterizados por:

. erros de especificação (a especificação contradiz as exigências do usuário),

. erros de programação (a programação contradiz as especificações),

. erros de interpretação (as especificações são mal interpretadas),

. erros de documentação (a documentação contradiz a programação),

2. Erros de produção, caracterizados por:

. erros de implementação (a implementação contradiz as especificações),

. materiais defeituosos (controle de qualidade inadequado),

. meios de produção inadequados (compilador inadequado para a elaboração do programa),

3. Erros de operação (faltas na fase de utilização do sistema), caracterizados por:

. defeitos físicos aleatórios que independem do processo de envelhecimento do sistema,

. falhas provocadas por desgaste, causadas por influências externas,

. erros de operação, causados pelo homem,

. erros intensionais (sabotagem).

A estruturação se refere apenas às causas de falhas mas não às consequências destas, o tipo de faltas define, em parte, a escolha de um procedimento de tolerância a faltas.

2.2.2 - UM MODELO DE CAMADAS.

A observação estrutural e funcional de faltas, pode ser incorporada em um modelo que subdivide um sistema complexo em diversos subsistemas, especificando as interfaces entre esses subsistemas. Um modelo deste tipo estrutura cada função e pode determinar qual subsistema está com falta e quais as funções que falharam em cada interface. O nivel de detalhamento pode ser em função das necessidades de localização das faltas. De modo semelhante às técnicas usuais de estruturação de sistemas complexos [89, 160], divide-se o sistema em camadas de tal forma que as camadas inferiores fornecem serviços para as camadas superiores.

O modelo de um computador pode englobar subsistemas de hardware e de software, e descreve as faltas de ambos. A figura 6 mostra um sistema multicomputador, seguindo essa técnica, estruturado em camadas, com associação de funções nas fron-

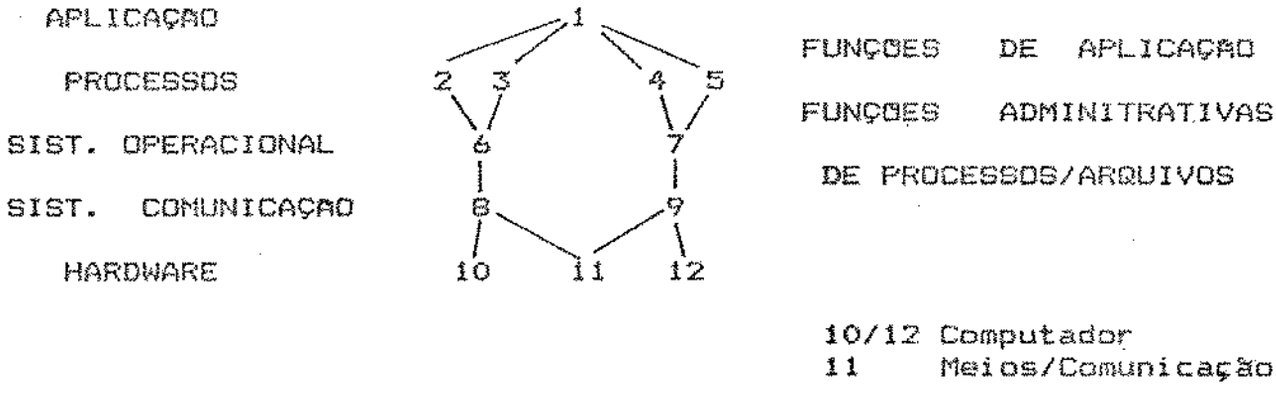


Figura 6 - Exemplo de estruturação em camadas.

teiras entre as camadas. O modelo de camadas apresenta as insuficiências do sistema associando a cada subsistema e a cada instante, o predicado "com falta" ou "sem falta", dependendo se cumpre ou não as especificações do subsistema. Adicionalmente, o modelo precisa descrever o possivel comportamento do subsistema com falta, especificando a função com falta. O modelo não define um método de especificação para descrever estados e funções com falta, mas permite que modelos concretos de faltas sejam emoldurados por ele [19].

Quando ocorrer uma falta, o subsistema em questão transita espontaneamente para o estado com falta. Os subsistemas que usam os serviços deste, também podem entrar em estado com falta, e o erro pode se espalhar a muitos subsistemas. A falha em um subsistema de camada mais baixa, que ainda não influe em subsistemas de camadas mais altas, constitue uma falta latente. O instante de utilização da função indica o instante em que uma falha concecutiva ocorrerá.

2.2.3 - FALTAS DE HARDWARE E FALTAS DE SOFTWARE.

Se ocorrer um erro causal em um subsistema de hardware, isto é, nas partes físicas do sistema, o erro e chamado de hardware. Similarmente, se o erro ocorre em subsistemas de software, isto é, na programação, diz-se que o erro é de software. Sistemas que toleram essas faltas dispõe de capacidade de redundância em hardware e em software. Eles descrevem a forma

de implementação dos meios de tolerância a faltas e fazem uma distinção entre elas.

Faltas em subsistemas podem ser permanentes e o estado com falta é continuo e ininterrupto até que sejam executadas, com sussesso, medidas de tolerância a faltas, incluindo-se as de manutenção. Nas faltas transitórias, o estado com falta desaparece espontaneamente e, eventualmente, pode aparecer de novo mais tarde. O subsistema permanecerá sem faltas sem que tenham sido tomadas quaisquer providências ou ações. Este comportamento, entretanto, é sempre baseado em uma falta permanente de um subsistema de camada mais baixa. Falhas de software podem estar presentes permanentemente em um programa sem causar falhas e podem aparecer como uma consequência transitória no processo de computação. As falhas consequentes de falhas transitórias podem, de novo, ser permanentes.

Do ponto de vista funcional, um sistema tem capacidade de tolerância a faltas quando as funções aplicativas permanecem sem falhas, ou seja, a camada superior do modelo fica sempre sem falhas (durante sua vida útil), mesmo quando ocorrem faltas em alguns subsistemas, e que pertençam ao conjunto de faltas a serem toleradas. A tolerância a faltas, segundo este ponto de vista, pode ser encarada como a ausência virtual de faltas. O conjunto de faltas a serem toleradas precisa ser definido. Considera-se a hipótese de uma ou de n faltas, e se atribue a cada subsistema com falta uma função com falta. As medidas de tolerância exigem tempo para a execução do procedimento e, às vezes, não podem reconstituir odesempenho do sistema original. A especi-

ficação de cada função deve prever uma redundância de tempo e uma possivel diminuição de desempenho. As exigências de tolerância a faltas em um sistema serão, assim, caracterizadas pelas funções aplicativas que não podem ser afetadas pelo conjunto de faltas que serão toleradas e pelos meios redundantes disponíveis.

2.3 - REDUNDANCIAS .

Um procedimento de tolerância a faltas, composto de diagnose e tratamento de faltas, necessita de meios adicionais àqueles exigidos pela função aplicativa. Por um lado, os subsistemas existentes devem ser ampliados por certas funções e, por outro lado, novos subsistemas devem ser adicionados para fornecer a função aplictivà que falhou e, além disso, prover a execução de funções de diagnose e tratamento das faltas. Todas essas características de um subsistema são englobadas sob o termo redundância. [74, 76, 95, 121, 137]. A redundância prevê a existência de meios, prontos para funcionar, além daqueles mínimos necessários à execução da função prevista [11, 149]. Pode-se dizer que a redundância é um meio desprezível na ausência de falta, por exemplo, um bit de paridade , um flip-flop para seu registro, um programa que administre a função e tempo para realizá-la. As redundâncias, por si mesmas não produzem efeitos de tolerância a faltas. Devem existir procedimentos específicos de tolerância associados a elas [65, 112].

2.3.1 - CARACTERISTICAS DA REDUNDANCIA .

Sistemas tolerantes a faltas dispõe de redundâncias cuja combinação estabelece sua utilidade. As classes seguintes explanam alguns aspéctos de uma técnica de redundância

[9, 19, 136].

. Redundância estrutural consiste da ampliação de um sistema por meio de subsistemas adicionais, do mesmo tipo ou de diferentes tipos. Em relação ao modelo de camadas, isto aumenta o número de subsistemas em uma camada, mas não o número de camadas. Por outro lado, hardware estrutural provoca sempre custo adicional que, normalmente, deve ser minimizado.

. Redundância funcional corresponde à ampliação do sistema por meio de funções adicionais para fins de redundância. Constituem funções redundantes todas aquelas que ultrapassam as funções aplicativas como, por exemplo, bit de paridade, ampliação do sistema operacional para administrar processos substitutivos, ou a ampliação dos meios de comunicação para que o sistema possa contornar subsistemas com falta [15, 26, 49, 54, 75, 106, 153, 162, 163].

. Redundância por diversidade significa a implementação redundante de uma função aplicativa por diferentes subsistemas, não idênticos entre si. Em hardware isso pode ser realizado por meio de lógica dual, em software por programas diversos com base em algoritmos diferentes e realizados por equipes diferentes, em projeto por diferentes equipes e diferentes formas de funções [9]. O melhor momento de utilização da diversidade é em um estado ainda rudimentar do projeto do sistema.

. Redundância de informação quer dizer que existe informação adicional além daquela necessária como informação útil. Um código redundante pode permitir o armazenamento ou transissão de dados, o reconhecimento e a correção de erros [32, 120, 155]. Um

código ECC (Error Correcting Code) pode restaurar a informação quando o erro não excede um número determinado de bit. Outras formas de redundância de informação incluem informações duplicadas ou contêm uma variante de uma informação, por exemplo, o número de elementos de uma lista, sendo a lista uma informação [10, 48, 118, 150].

. Redundância de tempo corresponde a tempo adicional e disponivel para a execução de funções adicionais. Quase todos os procedimentos de tolerância a faltas necessitam de certa redundância de tempo, podendo variar desde uma porcentagem infima até a múltiplos do tempo necessário ao próprio procedimento de tolerância a faltas. Certos processamentos precisam ser realizados repetidamente e isso exige tempo.

2.3.2 - MODOS DE ATIVAÇÃO DE REDUNDANCIAS .

Os meios redundantes podem ser ativados durante o funcionamento normal do sistema, isto é, antes da ocorrência de uma falta, ou somente após a ocorrência desta.

Na redundância estática, o meio redundante produz uma função durante todo o tempo da missão. Freqüentemente, a redundância estática é estrutural, na forma de número aumentado de elementos ou subsistemas. Um exemplo tipico é o sistema k dentre n nos quais, pelo menos k devem estar operando corretamente. Os subsistemas produzem resultados idênticos e um sistema de voto escolhe os resultados sem erros (voto de maioria), reali-

zando um teste relativo.

Na redundância dinâmica ou de reservas, os meios redundantes somente são ativados após a ocorrência de uma falta. A redundância dinâmica funcional corresponde à realização redundante de uma mesma função. Na estrutural, um subsistema de reserva ou sobressalente é ativado quando ocorrer uma falta no subsistema primário. Uma falta no sistema primário só pode ser descoberta pela utilização de testes absolutos. Não existem meios para comparar diferentes resultados e decidir qual é correto.

Conforme seja a utilização dos subsistemas sobressalente, a redundância poderá ser "fria", quando os subsistemas sobressalentes não executam tarefas e permanecem passivos até sua ativação (às vezes, desligados) [126], ou será uma redundância "quente", se os subsistemas sobressalentes realizam uma tarefa desde o começo, e que deixa de ser realizada por ocasião de uma falta [10, 67, 68, 74, 142, 146].

A redundância será coniderada mútua se os subsistemas sobressalentes estão disponíveis mutuamente como reservas. Na ativação, por ocasião de uma falta, um subsistema assume as tarefas do subsistema que falhou, adicionalmente às suas próprias [86, 87]. Desse modo, nenhuma tarefa deixa de ser executada mas, devido ao aumento da carga de tarefas, um desempenho pior deverá ser esperado, mesmo que apenas com relação ao tempo. Havendo vários sobressalentese sendo a carga distribuida igualmente pelos subsistemas remanescentes, o grau de capacidade de tolerância a faltas diminue com o aumento dasfaltas havendo, no entanto, uma degradação suave do desempenho.

2.4 - DIAGNOSE DE FALTAS.

A diagnose de faltas em um sistema é realizada, freqüentemente, usando um subsistema para verificar outro subsistema. Um módulo com falta, entretanto, pode diagnosticar falta em outro módulo que está em bom estado. Em um sistema com módulos interconectados e que podem diagnosticar-se mutuamente, é conveniente determinar que classes de módulos com falta podem ser corretamente evidenciados.

Um sistema com t faltas é diagnosticavel se qualquer conjunto de até t unidades com falta pode ser corretamente diagnosticado. Um sistema é sequencialmente diagnosticavel se, para qualquer conjunto de até t unidades com falta, pelo menos uma pode ser identificada [27].

Supondo que há n<=2t subsistemas, um sistema com t faltas é diagnosticavel se:

a) há, pelo menos, 2t + 1 subsistemas,

b) cada subsistema seja diagnosticavel por, pelo menos, t outros subsistemas.

O sistema com t faltas será sequencialmente diagnosticavel quando um conjunto de faltas não satisfaz as condições de sistema diagnosticavel e as condições de falta contém um elemento comum. Se o sistema contém menos que 2t + 1 subsistemas, não é sequencialmente diagnosticavel para t faltas.

Modelos mais realistas consideram sistemas em que mais de um módulo pode ser necessário para testar outro módulo,

dando origem a sistemas com t faltas dentre s módulos, onde t<=s, e onde s módulos devem ser substituidos para que um conjunto de t módulos com falta seja reparado [11, 77, 85].

Erros ou incosistências se manifestam através dos resultados de uma operação. Poucas são as faltas que se manifestam de imediato. Um subsistema deve, por algum tempo, passar do estado normal para um estado em teste, durante o qual são observadas as reações do subsistema a certas entradas de teste. Cada tipo de diagnose procura verificar o maior número possivel de faltas, ou seja, deve fornecer uma resposta adequada para todas as faltas a serem toleradas [56, 62, 63, 143]. O tempo de execução das funções aplicativas limita o tempo de teste. Conforme seja o detalhamento da associação das respostas a faltas no subsistema, pode-se distinguir:

. o reconhecimento de faltas, em que a resposta indica se todos os subsistemas estão sem falta. Em caso negativo, continua-se sem saber qual ou quais subsistemas estão com falta.

. a localização de faltas, em que os subsistemas com falta são identificados.

Muitas vezes, uma falta só será reconhecida por meio de uma falta conseqüente [45]. A localização de faltas, por outro lado, só terá sentido na medida em que a unidade com falta possa ser substituida [29, 32]. Além disso, podemos ainda distinguir um teste estrutural, no qual se examina uma estrutura previamente definida que consiste de subsistemas menores sem falta, dos testes funcionais, nos quais se verifica a função previamente definida e a ser executada.

Na diagnose de faltas, é necessário distinguir o subsistema a ser testado do subsistema testador. Nos grafos de diagnose, os subsistemas são identificados e as relações de teste são indicadas. Se o subsistema testador testa a ele próprio, um autoteste é realizado e o subsistema é autotestavel [1, 14, 62, 64, 155].

Se um processador executa certa seqüência de instruções que formam um teste, cuja correta execução necessita dos conteúdos de certos registros do próprio processador, trata-se de um autoteste. Os resultados podem ser verificados pelo próprio processador, constituindo um subsistema autotestável, ou por outro processador que irá usar esses resultados. De modo geral, a distinção entre teste e autoteste não permite concluir quanto à localização [12, 22, 32, 33, 36, 75, 146]. Um autoteste será conveniente quando o subsistema tem acesso a seus próprios elementos ou aos resultados de seus próprios processamentos.

Em um teste absoluto, verifica-se se são cumpridos predicados pré-estabelecidos, e que podem ser independentes das entradas de teste. Os testes absolutos são, tipicamente, usados na redundância dinâmica, já que os resultados só são disponíveis no sistema primário mas não nos sobressalentes.

Já no teste relativo, são comparados resultados de diferentes subsistemas ou por meio de diferentes processamentos. O teste relativo predomina na redundância estática (sistemas k dentre n). Se, na maioria dos resultados, não há concordância, o processo permite o reconhecimento da falta e o desligamento do sistema. Quanto ao desligamento, essa solução é se-

gura.

Um teste absoluto só reconhece faltas que demonstram algum desvio do resultado em relação a um predicado. Um predicado ou código capaz de descobrir todas as faltas é muito difícil de ser determinado [4, 16, 97, 104]. Além disso, sua utilização demandará grande tempo de processamento.

Programas de N versões usam o teste relativo [9]. Todas as distintas versões processam com os mesmos dados iniciais e os resultados são comparados. Em contraste, o teste absoluto é usado em blocos de recuperação de um sistema [123]. Um bloco de programa é executado e os resultados são comparados a predicados. Se não forem cumpridos, os mesmos dados são processados por um bloco de programa de versão distinta.

2.5 - TRATAMENTO DE FALTAS.

Uma vez que as faltas são reconhecidas e localizadas, um sistema de processamento tolerante a faltas precisa executar um procedimento de modo que todos os subsistemas sem faltas possam ser utilizados, podendo-se usar duas formas:

. um subsistema com faltas será substituido por um outro subsistema sem faltas,

. um subsistema com faltas será transformado em um sistema sem faltas.

Em muitos casos uma combinação dos dois procedimentos deve ser usada. A duração de um reparo é muito variavel

e pode ser grande quando é manual. Isso prejudica a disponibilidade, de tal forma que, em sistemas de computação é usada normalmente a reconfiguração automática, ocorrendo a substituição de subsistemas de hardware com faltas por outros sem faltas. No nivel de funções aplicativas, a distinção de estado com falta e sem falta depende do progresso ou estágio do processo. Um estado sem falta de um processo não pode ser atingido pela simples transição a um processo substitutivo, sendo necessário atualizar os valores de variáveis, de modo que o processo substitutivo alcance um estágio que cumpre os prerequisitos.

CAPITULO III

IMPLEMENTAÇAO DE COMPUTADORES TOLERANTES A FALTAS.

Muitos são os exemplos que podem-se utilizar para descrever o desenvolvimento e o atual estado da arte do projeto e da implementação de computadores que incorporam alguma forma de tolerância a faltas. Embora hoje as aplicações de máquinas deste tipo estejam se diversificando, é inegavel que, originariamente, os computadores tolerantes a faltas estão ligados a atividades espaciais, aeronáuticas e militares. A descrição desses estudos, partícularmente a descrição algo detalhada do programa "STAR" e do projeto "SIFT", é justificada pelo fato de que constituem pontos de partida para estudos e pesquisas nessa área.

3.1 - O PROGRAMA "STAR: Self Testing and Repair Computer".

De 1961 a 1972 foi realizado no Jet Propulsion Laboratory do California Institute of Technology, um programa de pesquisas que culminou com a construção do computador denominado "STAR" (Self Testing and Repairing), um computador uniprocessador. Os estudos iniciais apontaram para a conclusão de que a redundância dinâmica prometia as melhores soluções no projeto de um sistema de computação tolerante a faltas [10]. A redundância dinâmica ou "stand-by" requer um procedimento de dois passos para a eliminação de uma falta:

1.- A falta é detetada, ou seja, a presença da falta é determinada,

2.- A falta é corrigida, isto é, a unidade que falhou é substituida (ou então o programa é repetido ou o sistema reconfigura, etc.).

Em contrapartida, na redundância estática emprega-se o mascaramento, utilizado em sistemas digitais diversos com especial referência dos sistemas denominados TMR (Triple Modular Redundancy) ou redundância tripla e NMR (N-Modular Redundancy) ou redundância N-tupla [27].

Estudos analíticos de modelos com elementos em sistemas série-paralelo, indicam que a utilização de sobressalentes em redundância dinâmica resulta em uma vida média cerca de uma ordem de grandeza maior que sistemas não redundantes [24, 135].

Algumas vantagens qualitativas adicionais podem ser obtidas com a redundância dinâmica:

1.- Maior isolamento de faltas catastróficas não independentes, especialmente importantes em circuitos eletrônicos densamente empacotados. Observe-se que, na década de 60, quando o sistema "STAR" foi projetado, o estado da arte da tecnologia de circuitos integrados incluia a integração em pequena escala (Small Scale Integration SSI) e em média escala (Medium Scale Integration MSI), mas ainda não existiam os microprocessadores e outros circuitos integrados de tecnologia mais avançada de larga escala de integração (Large Scale Integration LSI) ou de muito larga escala de integração (Very Large Scale Integration

VLSI) que, hoje, tornam-se cada vez mais comuns,

2.- Sobrevivência do sistema até que todos os elementos sobressalentes tenham falhado,

3.- Uso de repetição de programas para a eliminação de erros causados por falhas transitórias,

4.- Adaptabilidade do sistema quanto ao número e tipo de sobressalentes,

5.- Utilização da potencialmente menor taxa de falhas de elementos não energizados,

6.- Menor consumo de energia e utilização plena das capacidades de conexão dos circuitos integrados (leque de entrada e de saida) em relação a sistemas com redundância estática,

7.- Facilidades de verificação das unidades sobressalentes por meio de programas de diagnóstico.

Técnicas e modelos aplicáveis à avaliação da confiabilidade de sistemas com sobressalentes foram desenvolvidos e mostrarama elevada sensibilidade de pequenas variações de um parâmetro adicional denominado suficiência (coverage), definida como sendo a probabilidade de que o sistema se recupere após a ocorrência de uma falha [24, 25]. Uma alteração deste parâmetro de 1,0 para 0,98, por exemplo, pode resultar em uma redução, em uma ou mais ordens de grandeza, da duração da missão com uma dada confiabilidade (ver também 5.1.2).

A maioria das técnicas usadas para aumentar a confiabilidade de um sistema como, por exemplo, a adição de maior número de sobressalentes, serão inúteis em face de uma suficiência inadequada. A adição de verificação, diagnose e outras téc-

nicas que possam melhorar o fator de cobertura de falhas revela-se mais promissora.

Em uma outra fase do programa "STAR" foram realizados trabalhos no sentido de identificar e solucionar problemas de projeto de um computador tipo uso geral incorporando redundância dinâmica. Tres áreas de pesquisa foram focalizadas:

1.- Métodos para deteção de falhas,

2.- Estudos sobre subdivisão do sistema em subsistemas com necessidades mínimas de interconexão,

3.- Estudos sobre o núcleo ou processador de teste e reparo, denominado "hardcore", envolvendo as alternativas de tecnologia e de lógica para a implementação das funções de deteção de falhas e de comutação dos elementos sobressalentes.

No computador "STAR" emprega-se uma mistura balanceada de redundância com sobressalentes, codificação, monitoração, replicação com voto, redundância de componentes e repetição de programas. O objetivo principal é a obtenção de um sistema autoreparavel em hardware e proteção contra faltas transitórias em software, capaz de apresentar tolerância a faltas transitórias, permanentes, aleatórias e catastróficas.

Uma configuração padrão foi implementada com sub-sistemas, cada um suplementado por um ou mais sobressalentes não energizados. Os métodos principais de deteção de erros e de recuperação são os seguintes:

1.- Todos os dados e instruções são codificados com códigos de deteção de erro e a deteção de falta ocorre durante a execução dos programas,

2.- O computador é dividido em um conjunto de unidades funcionais contendo seus próprios decodificadores de instruções e geradores de seqüências,

3.- A deteção de faltas, recuperação e substituição são controladas por uma unidade especial. No caso de falha na memória, software é usado para ampliar as capacidades do hardware nessas tarefas,

4.- Faltas transitórias são identificadas e seus efeitos corrigidos pela repetição de um segmento do programa atual,

5.- Faltas permanentes são eliminadas por substituição das unidades que falham,

6.- A substituição é implementada por comutação das linhas de alimentação dos circuitos. Unidades não energizadas fornecem apenas "0" lógico nas suas saidas e não podem produzir um "1" lógico,

7.- Circuitos especiais monitoram a sincronização e a operação interna das unidades funcionais,

8.- O processador de teste e reparo é protegido por triplicação e substituição dos elementos que falham na triplicata por sobressalentes.

Na figura 1 está representado o diagrama de blocos do computador "STAR", cujas sub-unidades são:

COP - Processador de controle; contém o contador de programa e registros de índice.

LOP - Processador lógico; executa operações lógicas em operandos (duas unidades energizadas).

MAP - Processador aritmético; executa operações aritméticas em operandos.

ROM - Memória com programas permanentes; 16 Kpalavras (duas unidades energizadas).

IOP - Processador de entrada e saida.

IRP - Processador de interrupção.

TARP - Processador de teste e reparo; monitora as operações e implementa a recuperacão (tres unidades energizadas).

M-I BUS - Barramento de entrada para a memória (duplo de quatro fios).

M-O BUS - Barramento de saida da memória (duplo de quatro fios).

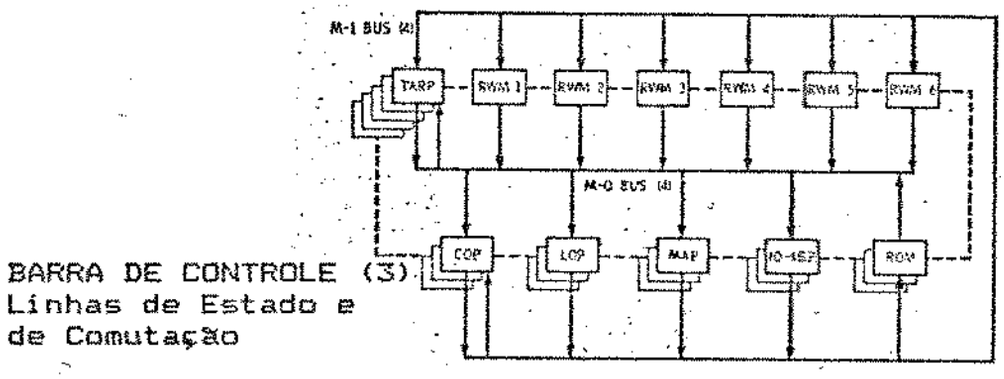


Figura 1 - Organização do Computador "STAR".

As palavras são de 32 bit e são transmitidas como oito conjuntos de 4 bit cada. As instruções tem um código de operação de 12 bit sendo codificadas como mostramos na figura 2.

Um dado de 32 bit contém uma palavra de verifica-

ção c(b) de 4 bit, cujo valor é dado por:

$$c(b) = 15 - |b|_{15}$$

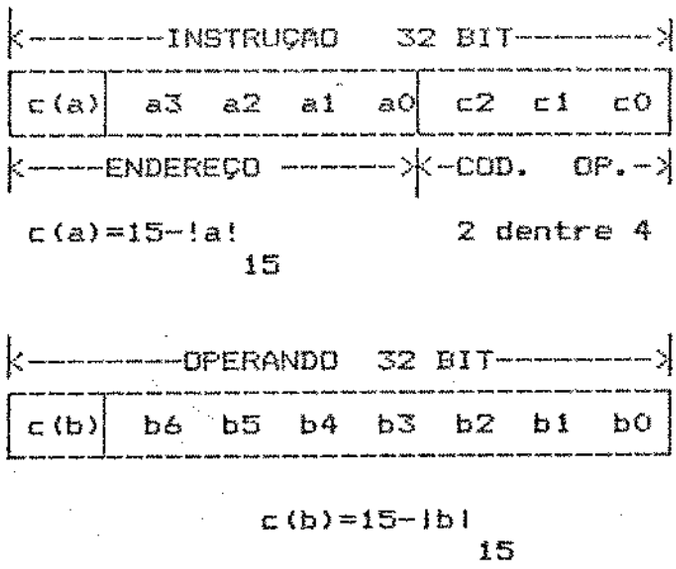


Figura 2 - Formato de Instruções e Dados do "STAR"

Na expressão acima, $|b|_{15}$ significa resíduo módulo 15 de b, sendo b o valor binário dos restantes 28 bit da palavra. O valor de c(b) faz com que todos os dados de 32 bit sejam múltiplos de 15. O algoritmo de teste calcula o resíduo da palavra de 32 bit e, sendo zero (representado por 1111) indica que a palavra está correta enquanto que outros valores indicam erro [10, 12]. Essa verificação é realizada concurrentemente com a transmissão da palavra para os barramentos.

As instruções tem um código de operação de 12 bit e um endereço codificado de 20 bit, tendo 4 bit como palavra de verificação. Os 16 bit restantes são codificados do mesmo modo como os dados. O código de operação é dividido em tres partes, cada uma das quais codificada no código 2 dentre 4. Um testador

de códigos é conectado ao barramento dos bits de tes- te, como mostra-se n figura 3.

O modelo utilizado para a análise de confiabilidade do "STAR" consiste de oito unidades funcionais e um procesador de teste e reparo, em uma cadeia série (figura 4). As unidades

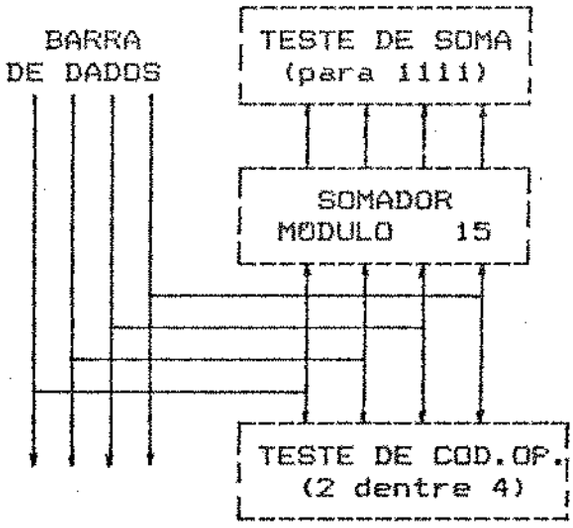


Figura 3 - Crcuito Testador e Barra de Teste.

são supostas iguais e análises para um número de sobressalentes igual a 2 e igual a 3 foram realizadas. O processador de teste e reparo foi modelado como um sistema de redundância híbrida H(3,S) de redundância modular tripla (TMR) e sobressalentes em número S. As demais unidades foram modeladas com redundâncias com sobressalentes não energizados, para aumentar a confiabilidade [10, 24, 25, 102].

O fator de suficiência ("coverage") é considerado por meio de dois aspéctos da arquitetura:

1. pela inclusão de um detetor de faltas e inicia-

dor de recuperação, na forma de um processador independente.

2. pela aplicação de um fator de auto-teste (FAT) variavel com a complexidade da unidade. Um fator FAT = 8/21 foi adotado, tendo sido estabelecido por meio de uma comparação detalhada de complexidade entre computadores anteriores e a complexidade do computador STAR [10].

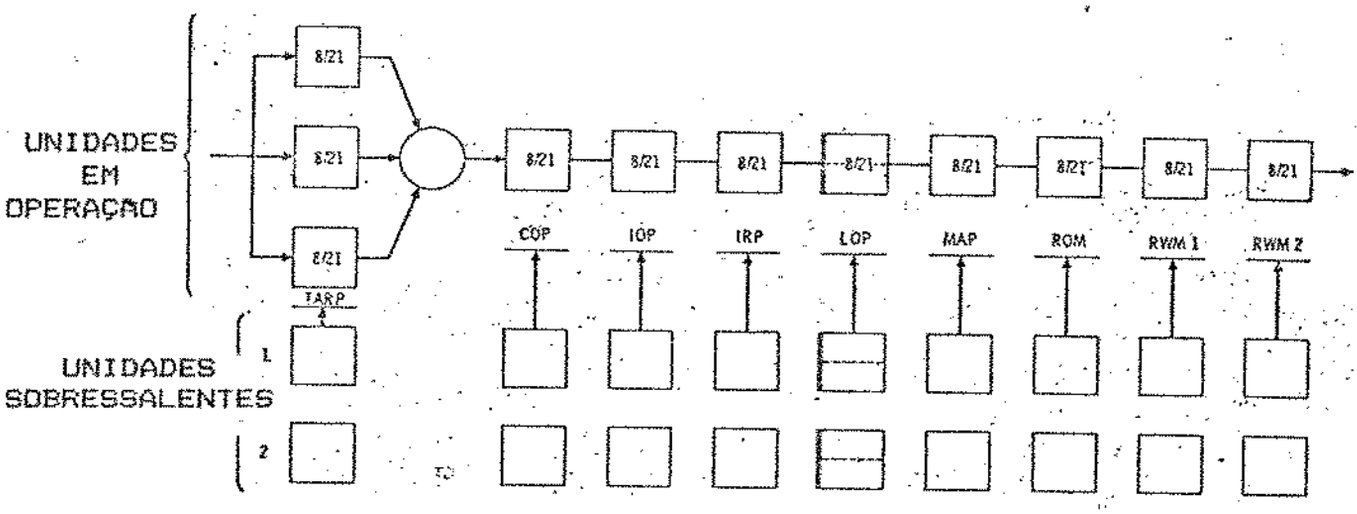


Figura 4 - Modelo de Confiabilidade do Computador "STAR".

Os resultados são mostrados resumidamente na tabela I, supondo dois limites das taxas de falhas: k = 1 se a taxa de falhas das unidades energizadas e das não energizadas são iguais; k = oo se a taxa de falhas das unidades não energizadas é zero.

TABELA I - CONFIABILIDADE DO SISTEMA STAR

Tempo da Missão	k = oo		k = 1	
	S = 3	S = 2	S = 3	S = 2
6 meses	0,9999998	0,99997	0,999995	0,99982
5 anos	0,997	0,97	0,966	0,87
10 anos	0.96	0,79	0,71	0,45

Uma observação adicional importante que deve-se fazer é a de que ao usar sobressalentes não energizados, além de permitir o aproveitamento da potencialmente menor taxa de falhas da unidade, a demanda de energia é mantida estável, contribuindo para uma provavel diminuição das taxas de falhas das fontes de alimentação e sistemas correlatos.

Em resumo,observa-se que no projeto STAR, foram usadas técnicas de redundância dinâmica, com subsistemas sobressalentes desligados. Essencialmente, o sistema utiliza tolerância a faltas de hardware, implementadas por hardware. O fator de suficiência (coverage) revela-se muito importante nos resultados. O software implementado tem mais a ver com as funções aplicativas do que com as funções de tolerância a faltas do sistema, embora diversas formas de codificação e repetição de programas tenham sido usadas, numa tentativa de uso balanceado de muitas técnicas. O baixo nivel de integração de componentes eletrônicos teve uma enorme influência no projeto básico e, hoje, o mesmo projeto, provavelmente, não seria realizado dessa mesma forma.

3.2 - O PROJETO "SIFT: Software Implemented Fault Tolerant Computer.

O computador "SIFT - Software Implemented Fault Tolerance" foi desenvolvido para aplicações críticas em aeronaves e que deve alcançar elevada confiabilidade por meio da distribuição de réplicas de programas entre diversos procesadores. A deteção e análise de erros e a reconfiguração do sistema são realizadas por software. As unidades principais são minicomputadores e as interfaces de entrada e saida são controladas por microcomputadores. O isolamento das faltas é realizado com o auxílio de um barramento redundante especial que interconecta as unidades. Os resultados de processamentos redundantes são votados após cada iteração [74, 125, 126, 158]

A filosofia em que se apóia o projeto SIFT baseia-se no fato de que o estudo de tolerância a faltas relativo a componentes e blocos lógicos simples não é relevante para a época. Faltas em linhas singelas de informação do tipo "grampeado em zero" e "grampeado em um" não são mais suficientes para consideração dos modos de falha de componentes eletrônicos de alta densidade de integração, de tecnologia LSI e VLSI, que são muito complexos e que podem afetar a performance de equipamentos de modo sutil. No projeto SIFT a unidade de deteção de faltas e de reconfiguração é um módulo processador/memória ou então um barramento com seu controlador. O projeto não considera hipóteses sobre os modos de falha, distinguindo apenas unidades com falha

e unidades sem falha. Códigos de deteção e correção de erros em memórias não são usados.

A tolerância a faltas no sistema SIFT é implementada, tanto quanto possivel, em software. Isso inclui deteção e correção de erros, diagnóstico, reconfiguração e a prevenção de que a unidade com falha possa provocar efeitos adversos no sistema. A estrutura do hardware é mostrada na figura 5, pela qual podemos identificar que se trata de uma arquitetura distribuida e não uniprocessador. Há, portanto, redundância em hardware por multiplicação dos módulos.

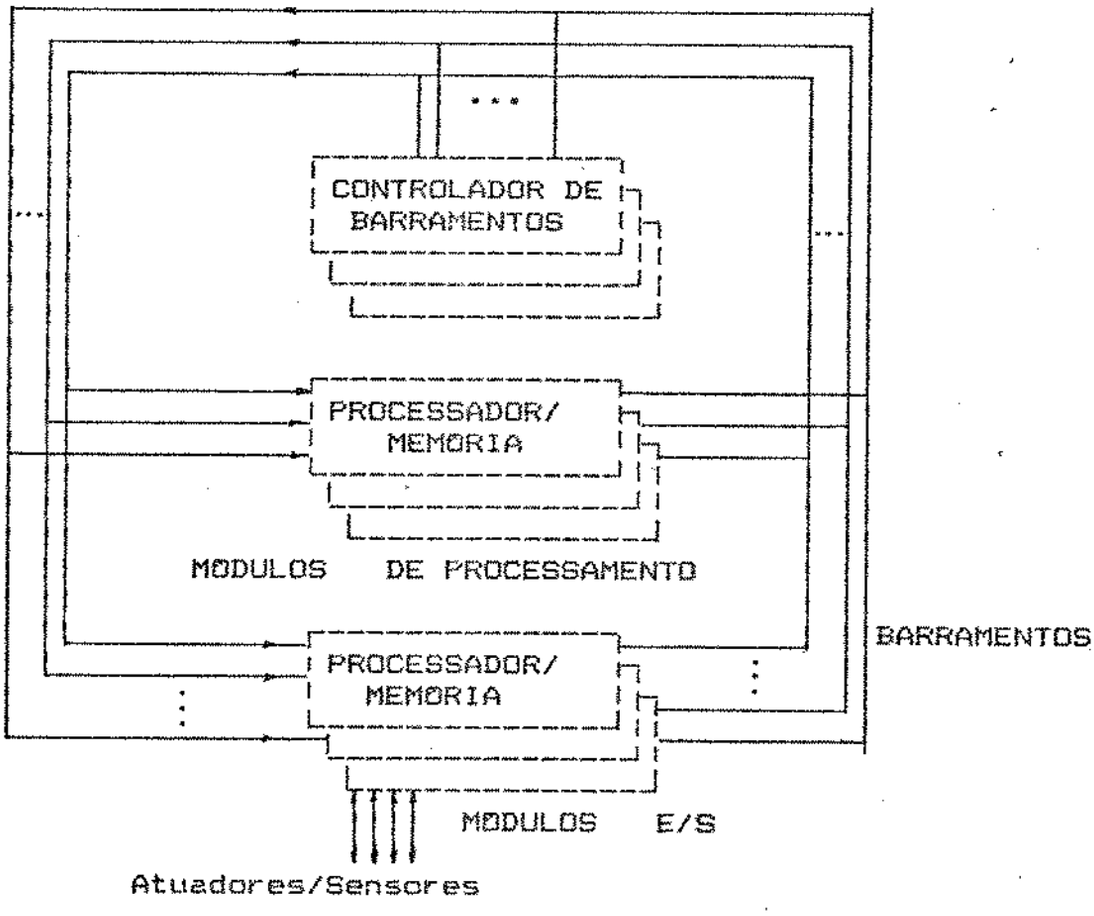


Figura 5 - Estrutura do Sistema SIFT.

O sistema SIFT executa um conjunto de tarefas, cada uma das quais consiste de uma seqüência de iterações. Os dados de entrada para cada iteração de uma tarefa são os dados de saida produzidos pela iteração anterior de algumas tarefas, que podem incluir esta. A confiabilidade é aperfeiçoada fazendo com que cada iteração de uma tarefa seja executada independentemente por certo número de módulos, cujo resultado é armazenado na memória associada ao processador que recebe os dados. O processador que usa a saida dessa iteração determina seu valor examinando as saidas geradas pelos processadores que realizaram a iteração e, tipicamente por meio de voto de dois dentre tres, escolhe o valor do resultado. Se as cópias das saidas não são identicas, um erro ocorreu e por meio dos valores armazenados na memória desse módulo o sistema executivo do SIFT determina qual unidade falhou.

Não há sincronização na execução das iterações de modo que falhas transitórias simultâneas em vários processadores não deverão produzir erros idênticos ou correlacionados nas várias réplicas. A proteção contra a corrupção de dados é realizada pelo modo como as unidades comunicam entre si. Qualquer processador pode ler de qualquer memória mas só pode escrever na memória a ele associada. Para evitar que um processador leia dados já corrompidos em outra unidade, o processador sempre recebe múltiplas cópias dos dados de modo que não só os dados errados são eliminados por voto bem como a unidade com falta é localizada.

O modelo de confiabilidade utilizado para análise do sistema SIFT está mostrado na figura 6. é um modelo do tipo Markoviano em que se supõe que as taxas de falhas do hardware

são constantes e que todas as falhas são permanentes. Cada estado do sistema é representado por um conjunto de números inteiros na forma (h,d,f) com h<=d<=f<=p. O estado representa uma situação em que f falhas em processadores individuais ocorreram, das quais d falhas foram detetadas e h das falhas detetadas foram "manipu-

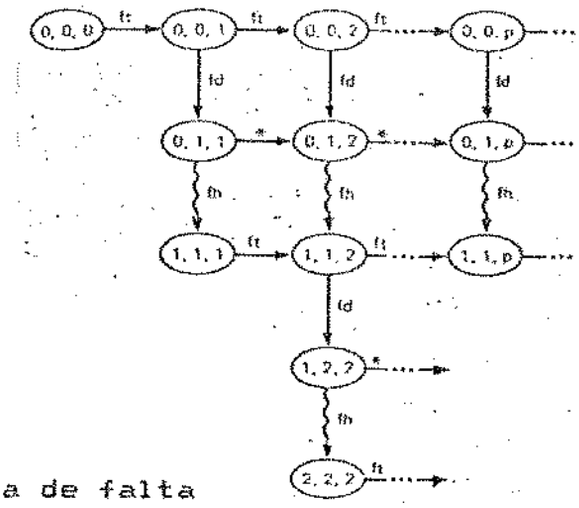


Figura 6 - Modelo de Confiabilidade do SIFT.

ladas" por reconfiguração de p módulos. Há tres tipos de transições possiveis entre estados:

1.- (h,d,f) -> (h,d,f+1) indica que ocorreu uma falha em um processador.

2.- (h,d,f) -> (h,d+1,f), com d<f, indica que houve deteção de uma falta.

3.- (h,d,f) -> (h+1, d,f), com h<d, indica que houve a correção de uma falta e reconfiguração.

O objetivo do projeto SIFT foi atingir a marca de 10^{-9} falhas por hora para um período de dez horas, considerando que as taxas de falhas dos módulos principais seria de 10^{-4} falhas por hora e a dos módulos de entrada e saida seria de 10^{-5} falhas por hora. A tabela I é um resumo de valores esperados de probabilidade de falha para um sistema constituido de cinco processadores. Supõe-se que os recursos estarão exauridos se restarem apenas dois módulos intactos. Uma falta dupla corresponde a ocorrência de uma segunda falta enquanto o sistema está reconfigurando como consequência de uma primeira falta o que, por hipótese, causa a falha do sistema [105, 158].

TABELA I - PROBABILIDADE DE FALHA DO SISTEMA SIFT

(Cinco processadores - 10 horas)

CAUSA DA FALHA	PROBABILIDADE DA FALHA
Recursos exauridos	5×10^{-12}
Falta dupla (100 ms)	7×10^{-11}
Falta dupla (1 s)	7×10^{-10}

3.3 - REQUISITOS E EXPERIENCIAS .

Neste capitulo procura-se ilustrar caracteristicas especificas de tolerância a faltas por meio de exemplos concretos encontrados em sistemas de computação comercialmente disponiveis e, também, esclarecer sua importância para seu emprêgo em sistemas de controle de processos industriais. São apresentados sistemas de computação tolerantes a faltas para diversas aplicações de modo a demonstrar a grande variedade e as diferentes características da tolerância a faltas. Com isso, deseja-se inspirar discussões sobre as possibilidades de realização de sistemas e a fazer uso das soluções descritas no desenvolvimento de projetos próprios para que possa ser atingida a melhoria da confiabilidade e disponibilidade, a necessária integridade de dados e a maior segurança no emprego de sistemas computacionais.

Quando se considera o campo de emprego de tolerância a faltas em computadores, fica claro que o emprego atual não é definido somente pelos critérios dscritos e pelas caracteristicas de desempenho da tolerância a faltas. Pode-se, porém, descrever realizações que demonstram caracteristicas bem especificas de tolerância a faltas em sistemas. Uma escolha é, primordialmente, subjetiva e deve servir como estimulo para dar maior importância aos aspéctos de tolerância a faltas em algumas classes de aplicações e a fazer uso das possibilidades existentes de realização, para obter maior disponbilidade, melhorar a probabili-

dade de sobrevivência e a necessária integridade de dados e maior segurança de operação.

Em artigos diversos, foi mostrado que os diferentes campos de emprego possuem bem diferentes exigências e prioridades com respeito aos procedimentos de tolerância a faltas, métodos de diagnose de falhas, métodos de redundância com relação à arquitetura na implementação do sistema. Os exemplos descritos a seguir, deveriam demonstrá-lo pelas medidas concretas tomadas. Neste contexto, deve-se concentrar nas características relevantes à tolerância a faltas. Quaisquer outras características podem ser esclarecidas mediante consulta aos manuais de sistemas correspondentes [5, 13, 23, 29, 35, 36, 42, 58, 69, 87, 107, 122, 151, 161].

Como campos essenciais para emprego de sistemas de computação tolerantes a faltas, pode-se distinguir:

. Emprego em aplicações críticas com sistemas de tempo real em medicina, aeronáutica, manufatura, e outros [68, 95, 99, 157, 158, 160],

. Emprego em aplicações de sistemas de tempo real de longa duração, sem possibilidade de manutenção, na astronáutica, satélites não tripulados, etc.[10, 52, 74, 105, 126],

. Emprego em outras aplicações técnicas e comerciais com sistemas de comutação, computadores de nó de redes de computadores, sistemas comerciais e redes locais [19].

Para os projetistas de sistemas, não é discutivel em todos os campos de emprego de sistemas computacionais, que as características que dizem respeito à tolerância a faltas se tor-

nam sempre mais importantes. Mesmo que o problema da tolerância a faltas tenha acompanhado o processamento de dados desde seu início, o desempenho do processamento com uma relação preço/desempenho, a dimensão dos sistemas e o desenvolvimento de software, permitem o emprego de sistemas computacionais em novos e cada vez mais amplos campos de aplicação e criam, com isso, uma nova dimensão para este tema. De início, necessariamente aumenta, com isso, a complexidade dos sistemas como um todo e se definem campos de emprego numa razão crescente de sua sensibilidade com relação a falhas do sistema, e onde a quebra de componentes introduz conseqüências não aceitáveis. Mesmo assim, as tarefas correspondentes não podem ser resolvidas sem o emprego de computadores e nas classes de aplicações em que medidas adicionais precisam ser tomadas. Propriedades de tolerância a faltas precisam ser, por isso, parte integrante da implementação do sistema.

Apesar disso, a situação no mercado mostra que, no total, são usados poucos sistemas computacionais tolerantes a faltas, mesmo quando foram, como mostram os exemplos, implementados implicitamente e explicitamente a propriedades singulares de tolerância a faltas. As publicações citadas, mostram que existem muitas tentativas de solução mas, também, que o mercado comercial fica aquém das exigências com respeito à tolerância a faltas. A razão disso está relacionada ao fato de que as propriedades de tolerância a faltas se baseiam na disponibilidade de componentes adicionais e redundância de componentes, que é a razão para este desempenho desfavorável. É certo que, com respeito a esse ponto, a tecnologia VLSI melhora sensivelmente a im-

plementação da tolerância a faltas e abre novos caminhos para a sua solução. Os vistos até agora e em uso, encontram-se em diferentes estágios de desenvolvimento. Existe um grande número de sistemas experimentais e vários sistemas especiais orientados para uma aplicação definida. Produtos que são de uso geral e comercialmente disponíveis são muito poucos, existindo apenas em certos campos de aplicação.

Nos sistemas comercialmente disponíveis dominam fabricantes especializados e, em geral, muito conhecidos. Os fabricantes tradicionais oferecem somente soluções parciais Um estudo sobre o volume total do mercado de sistemas tolerantes na Europa ocidental [19] , indica a cifra de US$ 162 milhões em 1984. Para 1985, esperase que o valor suba para US$249 milhões. A previsão para 1990 é de US$ 1,9 bilhões. Avalia-se, ao mesmo tempo, que em 1990 as porcentagens de sistemas instalados nas aplicações mais importantes serão cerca de 20% nas bolsas de valores, 10% em bancos, 10% em manufatura, 13% no comércio e 8% em comunicações. O maior mercado atual é a Inglaterra com US$ 46 milhões (valor dos sistemas instalados), seguido pela República Federal da Alemanha (29 milhões), Escandinávia (19 milhões) e Benelux (14 milhões).

3.3.1 - COMPUTADORES PARA TELECOMUNICAÇÕES .

Sistemas digitais de comutação telefônica são caracterizados pela exigência de disponibilidade do sistema e de

realizar sua função, sendo a duração entre a quebra e a recuperação de, no máximo, 2 horas em 30 anos [11, 85] As medidas e providências são, por isso, orientadas no sentido de garantir o funcionamento sem perturbações, e de atingir um neutralização global de falhas, além de serem usados componentes de alta qualidade e longo tempo de vida. As exigências para esse campo são caracterizads por:

. reconhecimento de falhas através de supervisão, sincronização e exame em certos ciclos,

. tratamento automático de alarmas e localização automática das falhas de hardware bem como de software,

. tratamento de falhas automático por reconfiguração ou reinício,

. equipamentos de teste e diagnose adicionais para uso por operadores de modo a possibilitara procura de falhas e o reinicio da operação no mais curto período de tempo possivel,

. segurança de arquivos de dados para medidas efetivas de correção de falhas.

A alta disponibilidade do sistema inteiro exige, além disso, a capacidade de acoplamento e desacoplamento de componentes, sem a interrupção do sistema.

3.3.2 - COMPUTADORES PARA PROCESAMENTO COMERCIAL.

Um grande número de aplicações comerciais é hoje realizado na forma de processamento de transações. Muitos usuá-

rios manuseiam, iterativamente, uma arquivo de dados grande e de acesso comum. O conjunto de mudanças ou perguntas que representam uma ocorrência comercial é executado como uma transação que transforma o arquivo de dados de um estado para outro. As exigências características para os sistemas de processamento de transações, são [133]:

. alta disponibilidade para a execução continua dos acontecimentos comerciais, nos termos normais de trabalho,

. alta integridade dos dados para uma representação precisa dos fatos comerciais no arquivo de dados,

. tratamento automático de falhas em defeitos de hardware e defeitos de transações,

. reconfiguração de subsistemas (de hardware e de software) sem perturbar o funcionamento normal.

3.3.3 - SISTEMAS DE REDE LOCAL.

Endende-se por sistema de rede local, um sistema computacional que, em rede local, oferece aos usuários de uma estação de trabalho, serviços como [96]:

. aquisição e armazenamento de dados,

. administração e emissão de documentos,

. acesso a outras redes, públicas ou não, ou serviços semelhantes.

Freqüentemente, os usuários de uma rede local formam um grupo de trabalho que, primordialmente no campo de escri-

tórios , em parte autonomamente e, em parte cooperativamente, resolvem problemas comerciais de aplicação. As exigências características são aqui semelhantes aos sistemas de transações:

. alta disponibilidade do "server" para apresentação contínua dos serviços,

. administração automática e redundante dos arquivos de dados para as estações de trabalho,

. tratamento automático de falhas de defeitos de hardware e de erros nos serviços,

. configuração aumentada ou reconfiguração da rede, sem interromper o serviço corrente.

3.3.4 - SISTEMAS DE TEMPO REAL.

Computadores de tempo real, usados no controle, supervisão e automação de sistemas industriais e para a construção de sistemas de comunicações, encontram-se em um ambiente de sistemas que [86, 98, 99]:

. proporciona somente intervalos muito pequenos de tempo para o reconhecimento de falhas de modo a garantir a continuação do processamento em tempo real,

. não permite, em geral, o restabelecimento (backward recovery), por causa da falta de redundância no tempo, pois eventos antecedentes não são reproduzíveis normalmente para o funcionamento do sistema, causado por condição de tempo real,

. mesmo com o emprego de sistemas de múltiplos processadores, por exemplo, no processamento paralelo, criam-se poucos novos intervalos de tempo para utilizar propriedades de tolerância a faltas,

. exige , por razões de comportamento em tempo real, redundâncias estáticas que continuam a controlar o processo industrial de maneira bem definida,

. exige do sistema, além da necessidade de reconhecimento e mascaramento de falhas, a capacidade de permanecer em um estado seguro, ou de transitar imediatamente para um tal estado, isto é, realizar um comportamento a prova de falhas (fail-safe), e assegurar alta integridade de dados,

. por causa de procedimentos típicos de supervisão precisam

utilizar métodos de diversificação ou verificação.

Tornaram-se conhecidos neste campo de aplicação os sistemas August-System: Series 300 e o sistema TELEPERM M da Siemens. As propriedades de tolerância a faltas para um sistema de tempo real, são demonstradas pelo sistema RDC (Really Distributed Control Computer System) de configuração distribuida de um sistema multi-microcomputadores [22, 71, 127, 160], com propriedades de tolerância a faltas, que foi realizado como um sistema piloto para controle de 28 alto-fornos e substituiu, nesta aplicação, todos os sistemas convencionais de controle e comando. A análise desse projeto, mostra que a tolerância a faltas não é definida somente pela construção do hardware e pela parte do sistema operacional que diz respeito à supervisão, mas também da:

. escolha dos meios de comunicação (fibra ótica),

. tecnologia de construção,

. técnicas de programação e compilação para sistemas de multiprocessadores,

. interface homem-máquina amistosa ao usuário.

que definem os critérios para tolerância a faltas do sistema inteiro.

3.3.5 - ALGUMAS OBSERVAÇÕES .

Na escolha de exemplos como acima descritos, a situação atual do mercado foi anotada usando-se, sempre que possivel,

sistemas existentes para exemplificar a utilização da tolerância a faltas em computadores. Isso permite descrever as interdependencias de funções do sistema e as propriedades da tolerância a faltas, e que são distintas em cada área especifica. Ao contrário do que ocorre em modelos, representam limitações com relação à tolerância a faltas. Deve-se observar que, em quase todos os sistemas disponiveis, geralmente do tipo "turn-key", o usuário precisa ainda tomar medidas complementares e integradoras para providenciar a tolerância a faltas em seu sistema e para o seu uso especifico. Além disso, os sistemas toleram apenas falhas especificas de hardware. Falhas múltiplas e falhas de software, necessitam de análise e tratamento especifico adicional.

Analisando as diversas publicações com relação à tolerância a faltas, observa-se que em várias aplicações, são descritas soluções para sistemas computacionais tolerantes a faltas que ultrapassam as áreas aqui tratadas, reforçando nossa opinião de que, em quase todas as áreas de uso de computadores, a tolerância a faltas terá significado fundamental para a operação confiavel de um sistema de controle de processos industriais. Em sua realização, certamente serão usadas novas tecnologias oriundas do atual desenvolvimento de VLSI.

CAPITULO IV

MODELOS MATEMATICOS

4.1 - MODELOS DE MARKOV

Existem quatro tipos de modelos de probabilidade de Markov, cada um relacionado, de modo particular, com o conceito de confiabilidade. Modelos de Markov são funções de duas variáveis aleatórias que são: o estado x de um sistema e o instante t da observação desse sistema. Os tipos de modelos provêm do fato de que cada uma dessas variáveis aleatórias pode ser continua ou discreta. Quando os estados são discretos e os instantes são também discretos, periódicos por exemplo, o modelo se constitue em uma cadeia de Markov. Se os estados são discretos mas a observação é em qualquer instante de tempo, o modelo será um processo de Markov.

Um processo de Markov é descrito por um conjunto de probabilidades p_{ij} que estabelecem a probabilidade de transição de um estado i para um estado j quaisquer do sistema x. Em qualquer modelo de Markov, as probabilidades p_{ij} dependem exclusivamente dos estados i e j, e são independentes de todos os estados anteriores, com excessão do último estado i.

A distribuição de Poisson, segundo a qual são estabelecidas as relações de probabilidade de um processo de Poisson, que é um tipo particular de processo de Markov, depen-

de do número de ocorrências no tempo, cuja probabilidade é constante em um intervalo pequeno de tempo Δt. Caracteristicamente, o número de faltas no tempo, para um grupo de componentes que opera em modo de substituição instantânea, isto é, paralela, ou no modo de sobressalente, constitue um processo de Poisson [135]. As ocorrências (faltas ou falhas) são discretas e o tempo é contínuo, o que caracteriza um modelo de eventos discretos e tempo contínuo.

Essencialmente, um processo de Poisson é baseado nas seguintes premissas:

1. a probabilidade de que uma transição ocorra do estado n para o estado n+1, durante o intervalo de tempo Δt é λt, onde λ é o número de ocorrências na unidade de tempo.

Nas aplicações à confiabilidade, λ corresponde ao número de falhas na unidade de tempo, ou seja, é a taxa de falhas. As ocorrências, neste modelo básico, são irreversíveis, o que significa que o número de ocorrências não decresce no tempo.

Modelos de Markov onde transições em dois sentidos são permitidas, constituem a base para a modelagem de sistemas reparáveis, nos quais um componente que falhou segundo uma taxa de falhas λ, é reparado ou recuperado e, em seguida, restituido ao sistema [7, 8, 21, 135]. O componente reparado pode ser considerado como novo, isto é, sua taxa de falhas é a mesma que a que tinha no intervalo de tempo em que operou antes da primeira falha.

2. cada ocorrência é independente das outras ocorrências, isto é, a lei que as governa é a união das pro-

babilidades,

3. a probabilidade de transição de duas ou mais ocorrências, no mesmo intervalo de tempo Δt é desprezível. Esta é apenas uma outra forma de estabelecer a independência entre as ocorrências. Pode-se dizer que a probabilidade de duas ocorrências no mesmo intervalo de tempo Δt é dada pelo produto das probabilidades de cada ocorrência, isto é, $(\lambda\Delta t).(\lambda\Delta t)$, e que se constitue um infinitésimo de segunda ordem, podendo ser desprezado.

As probabilidades de estados e as probabilidades de transição podem ser representadaspor um sistema de equações de diferença. A probabilidade da ocorrência n no instante t, do sistema x, pode ser posta na forma:

$$P(x=n,t) = P_n(t) \qquad [1]$$

Se não ocorrerem transições até o instante $t+\Delta t$, a seguinte equação de diferença pode ser escrita:

$$P_0(t+\Delta t) = (1 - \lambda\Delta t)\, P_0(t) \qquad [2]$$

da qual pode-se obter:

$$\lim_{\Delta t \to 0} \frac{P_0(t+\Delta t) - P_0(t)}{\Delta t} = \lim_{\Delta t \to 0} (-\lambda).P_0(t) \qquad [3]$$

e da qual, utilizando a definição de derivada, pode-se obter:

$$\frac{d\,P_0(t)}{dt} = (-\lambda).P_0(t) \qquad [4]$$

Esta equação indica que a probabilidade de não existirem ocorrências no instante $t+\Delta t$ é $P_0(t+\Delta t)$, e é dada pela probabilidade $P_0(t)$ de não existirem ocorrências no instante t, multiplicada pela probabilidade $(1-\lambda\Delta t)$ de não existirem ocorrências no intervalo Δt.

No caso de uma ocorrências no instante $t+\Delta t$, tem-se que:

$$P_1(t+\Delta t) = (\lambda\Delta t).P_0(t) + (1-\lambda\Delta t).P_1(t) \qquad [5]$$

Por esta relação, a probabilidade de uma ocorrência no instante $t+\Delta t$ poderá existir:

1. se não havia ocorrência no instante t, ou seja, $P_0(t)$, e uma ocorrência existiu no intervalo Δt, sendo sua probabilidade $\lambda\Delta t$, ou então,

2. já havia uma ocorrência no instante t, $P_1(t)$, e nenhuma outra existirá no intervalo Δt, o que pode ser expresso por $(1-\lambda\Delta t)$.

A expressão [3] pode ser generalizada para uma ocorrência n no instante $t+\Delta t$, na forma:

$$P_n(t+\Delta t) = (\lambda\Delta t)P_{n-1}(t) + (1-\lambda\Delta t)P_n(t) \qquad [6]$$

sendo n=1, 2, 3, ...

Da expressão acima, pode-se obter, pela definição de derivada, tal como pela expressão [3]:

$$\lim_{\Delta t \to 0} \frac{P_n(t+\Delta t) - P_n(t)}{\Delta t} = \lim_{\Delta t \to 0} \lambda P_{n-1}(t) + \lim_{\Delta t \to 0} (-\lambda) P_n(t) \qquad [7]$$

e, ainda:

$$\frac{d P_n(t)}{dt} = \lambda P_{n-1}(t) - \lambda P_n(t) \qquad [8]$$

As equações [2] e [6] realmente descrevem um sistema discreto no tempo, já que o tempo é dividido em intervalos Δt. Mas fazendo $\Delta t \to 0$, obtém-se equações diferencaiais que descrevem um processo de Poisson, em tempo contínuo, tal como nas expressões [4] e [8].

Para completar a descrição de um sistema, um conjunto de condições iniciais será necessário. Considerando que para t=0, aconteceram m ocorrências, pode-se escrever:

$$P_0(0) = P_1(0) = \ldots = P_n(0) = 0 \qquad \text{com } n \neq m$$

e que

$$P_m(0) = 1 \qquad [9]$$

A solução de [4] com a condição inicial $P_0(0) = 1$, fornece:

$$P_0(t) = e^{-\lambda t} \qquad [10]$$

A solução da equação diferencial obtida de [5], com $P_0(0)=1$ e $P_1(0)=0$, fornece:

$$P_1(t) = \lambda t . e^{-\lambda t} \qquad [11]$$

e, similarmente, a solução de [8], fornece uma expressão geral:

$$P_n(t) = \frac{(\lambda t)^n}{n!} e^{-\lambda t} \qquad [12]$$

Isso mostra que o processo de Poisson é um tipo especial de processo de Markov, que pode ser estabelecido a partir das tres premissas básicas. Não é necessário mencionar a distribuição binomial, como é feito em certos textos [40, 135].

Uma importante interpretação de $P_0(t)$ segue. Se t_0 é o instante da primeira ocorrência, $P_0(t)$ é a probabilidade de não existirem ocorrências, isto é:

$$P_0(t) = P(t<t_0) = 1 - P(t_0<t) \qquad [13]$$

Desse modo, $1-P_0(t)$ é a função distribuição acumulada da variavel aleatória t_0, o instante da ocorrência. A função densidade para o instante da primeira ocorrência será obti-

do pela derivada:

$$f(t) = \frac{d}{dt}\left(1 - e^{-\lambda t}\right) = \lambda e^{-\lambda t} \qquad [14]$$

Isso significa que o instante da primeira ocorrência é exponencialmente distribuido e que, considerando que cada ocorrência é independente, o intervalo de tempo entre quaisquer duas ocorrências é exponencialmente distribuido.

Define-se, a seguir, a matriz de transição de estados. Para a formulação de um modelo de Markov, considerando especificamente o modelo de estados discretos e tempo contínuo, deve-se definir todos os estados mutuamente exclusivos do sistema. é o caso, por exemplo, de um único componente não reparavel, e que poderá estar em um de dois possiveis estados: bom estado s_0 e quebrado s_1. O conjunto de equações de estado descreve as transições probabilisticas do estado inicial, para t=0, para o estado final, de equilíbrio. As probabilidades de transição devem obedecer as seguintes regras:

1. a probabilidade de transição, no intervalo de tempo Δt, de um estado para outro é dada por

$$z(t)\Delta t \qquad [15]$$

onde z(t) é o acaso ou casualidade associada com os os estados em questão,

2. as probabilidades de mais de uma transição, no intervalo de tempo Δt, constituem infinitésimos de ordem superior

e podem ser desprezadas.

Para um sistema em que $z_i(t)$, para todos os componentes i, é uma constante no tempo, isto é:

$$z_i(t) = \lambda_i \qquad [16]$$

um modelo homogêneo será estabelecido. Para um único componente, a probabilidade de estar no estado s_0 (bom estado), no instante $t+\Delta t$, é

$$P_{s0}(t+\Delta t) = [1 - z(t).\Delta t]P_{s0}(t) + 0.P_{s1}(t) \qquad [17]$$

que é formada por duas parcelas: a primeira, é a probabilidade do sistema estar no estado s_0 (bom estado), no instante t, $P_{s0}(t)$ multiplicada pela probabilidade de que não ocorreu falha no intervalo Δt, isto é, $[1 - z(t).\Delta t]$; a segunda parcela é a probabilidade do sistema estar no estado quebrado s_1 no instante t, $P_{s1}(t)$, multiplicada pela probabilidade de reparo, no intervalo Δt, que é zero (não há reparo, por hipótese).

De modo análogo, a probabilidade do sistema estar no estado s_1 no instante $t+\Delta t$ é:

$$P_{s1}(t+\Delta t) = 1.P_{s1}(t) + [z(t).\Delta t]P_{s0}(t) \qquad [18]$$

formada por duas parcelas: a probabilidade de permanecer no estado s_1 (quebrado), que é 1; e a probabilidade de transição do estado s_0 para o estado s_1, que é a probabilidade de falha

$z(t).\Delta t$, multiplicada pela probabilidade de estar no estado s_0.

Uma matriz de transição de estados pode ser definida usando-se as equações [17] e [18]:

ESTADO INICIAL	ESTADO FINAL s_0	ESTADO FINAL s_1
s_0	$1 - z(t).\Delta t$	$z(t).\Delta t$
s_1	0	1

A matriz de transição de estados tem a propriedade de que a soma dos elementos de uma fila é igual a 1.

As expressões [17] e [18] podem ser rearranjadas de modo a se obter:

$$\frac{P_{s0}(t+\Delta t) - P_{s0}(t)}{\Delta t} = -z(t).P_{s0}(t) + \frac{0}{\Delta t}.P_{s1}(t)$$

$$\frac{P_{s1}(t+\Delta t) - P_{s1}(t)}{\Delta t} = z(t).P_{s0}(t) \qquad [19]$$

O termo envolvendo $P_{s1}(t)$ no segundo membro da primeira equação [19] pode ser cancelado, se não há reparo. Passando aos limites das equações para $\Delta t \to 0$, e usando a notação $\dot{P}$ para designar derivadas, obtém-se:

$$\dot{P}_{s0}(t) = -z(t).P_{s0}(t)$$

$$\dot{P}_{s1}(t) = z(t).P_{s0}(t) \qquad [20]$$

Em geral, interessa examinar um sistema supondo que, inicialmente, o sistema está em bom estado. Dessa forma:

$$P_{s0}(0) = 1$$

$$P_{s1}(0) = 0 \qquad [21]$$

A solução das equações [20] com as condições iniciais [21], fornece:

$$P_{s0}(t) = \exp-\int_0^t z(\delta).d\delta]$$

$$P_{s1}(t) = 1 - \exp-\int_0^t z(\delta).d\delta] \qquad [22]$$

sendo, evidentemente:

$$P_{s0}(t) + P_{s1}(t) = 1$$

o que significa que é suficiente calcular $P_{s0}(t)$, que indica a probabilidade do componente permanecer em bom estado. Por definição, isso expressa a confiabilidade do componente. No valor da

condiç~ao inicial $P_{s0}(0)$ pode-se ainda incluir uma falha inicial, que ocorre antes do sistema ser energizado ou ligado. Um tratamento mais geral será dado a este problema no desenvolvimento de um modelo geral (veja capítulo V).

4.1.1 - GRAFOS DE MARKOV.

Um modelo de Markov pode ser caracterizado por um grafo, composto de nós que representam os estados do sistema, e de ramos que representam as probabilidades de transição. Para o problema descrito pelas equações [17] e [18], ou pela correspondente matriz de transição de estados, o grafo será como o da figura 1.

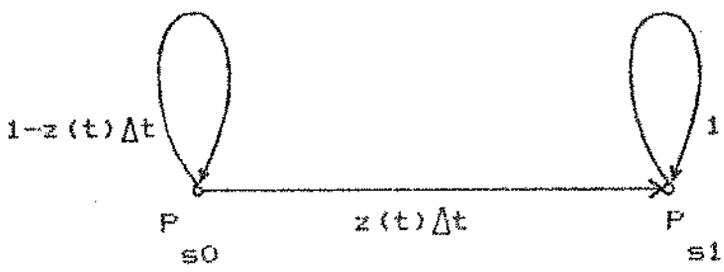


Figura 1 - Grafo de Markov para Um Elemento não Reparavel.

Tratando os nós como fontes de sinais e as probabilidades de transição como coeficientes de transmissão, pode-se escrever equações que indiquem que a probabilidade de estar em um nó no instante $t+\Delta t$ é a soma dos sinais que chegam àquele nó. Todos os outros nós são considerados fontes no instante t e todas

as probabilidades de transição devem ser consideradas como ganhos de transmissão.

Pode-se escrever diretamente as equações diferenciais como na equação [20], seguindo as seguintes regras. A variação, no tempo, da probabilidade em um nó, isto é, sua derivada, será igual à soma dos produtos dos coeficientes de ganho dos sinais que chegam ao nó pelas probabilidades dos nós de origem desses sinais. Os fatores Δt devem ser eliminados das expressões. Nos laços fechados (de um nó para o próprio nó) os fatores de ganho unitário devem ser igualados a zero. Sinais algébricos negativos serão usados para indicar um sinal que sai de um nó. Para a figura 1, por exemplo, as equações serão:

$$\dot{P}_{s0} = -z(t).P_{s0}$$

$$\dot{P}_{s1} = z(t).P_{s0} \qquad [23]$$

A operação em paralelo ou de um sobressalente, em um sistema de dois componentes, pode ser ilustrada pelo grafo da figura 2. Sendo x_1 e x_2 os componentes, haverá quatro possíveis estados do sistema: $s_0 = x_1 x_2$, $s_1 = \bar{x}_1 x_2$, $s_2 = x_1 \bar{x}_2$ e $s_3 = \bar{x}_1 \bar{x}_2$, onde a barra indica que o componente falhou.

O grafo de Markov correspondente está mostrado na figura 2.

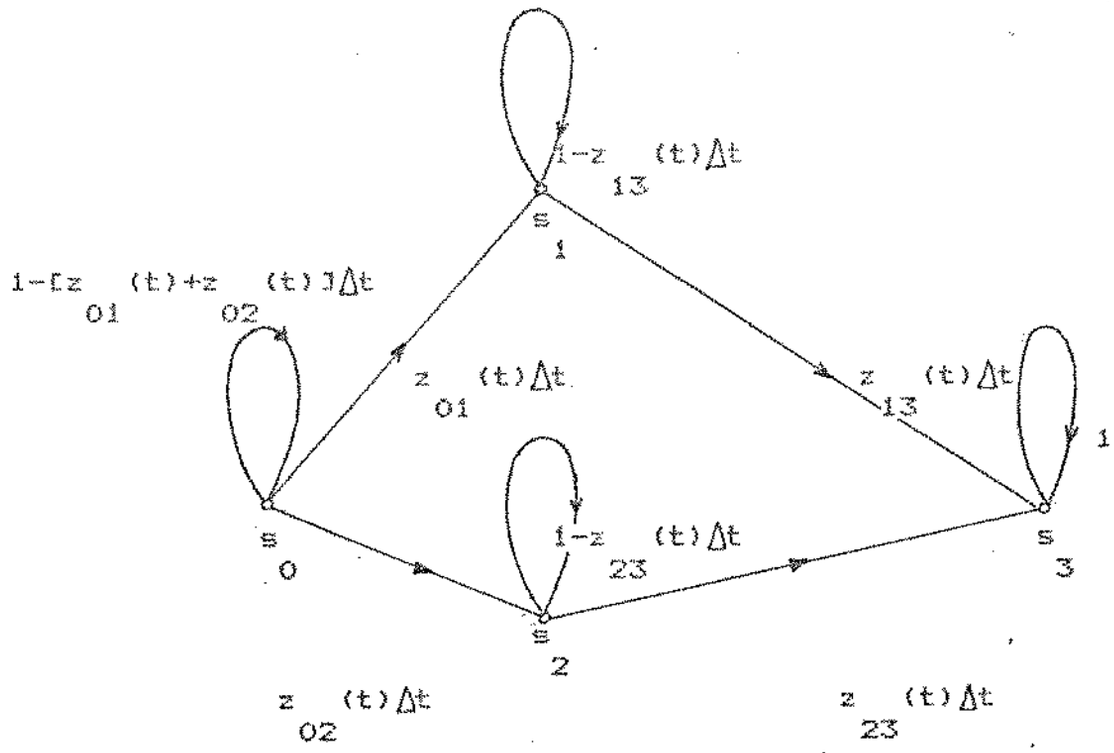


Figura 2 - Grafo de Markov para Sistema de Dois Componentes.

Do grafo, por inspeção, e usando as regras estabelecidas, pode-se escrever as equações:

$$\dot{P}_{s0}(t) = -[z_{01}(t)+z_{02}(t)]P_{s0}(t)$$

$$\dot{P}_{s1}(t) = z_{01}(t)P_{s0}(t) - z_{13}(t)P_{s1}(t)$$

$$\dot{P}_{s2}(t) = z_{02}(t)P_{s0}(t) - z_{23}(t)P_{s2}(t)$$

$$\dot{P}_{s3}(t) = z_{13}(t)P_{s1}(t) + z_{23}(t)P_{s2}(t)$$

$$P_{s0}(t) + P_{s1}(t) + P_{s2}(t) + P_{s3}(t) = 1 \qquad [24]$$

4.2 - CONSIDERAÇÕES SOBRE A FUNÇÃO ACASO.

As equações dadas por [24] são muito difíceis de resolver para uma função z(t) qualquer. Se ela for explicitada, entretanto, as soluções serão bem mais simples de obter, especialmente se z(t) é uma constante, isto é, uma função independente do tempo t.

Desse modo, se $z_{01}(t)=\lambda_{01}$, $z_{01}(t)=\lambda_{01}$, $z_{02}(t)=\lambda_{02}$, $z_{02}(t)=\lambda_{02}$, $z_{13}(t)=\lambda_{13}$, $z_{13}(t)=\lambda_{13}$ e $z_{23}(t)=\lambda_{23}$, $z_{23}(t)=\lambda_{23}$ e, considerando ainda as condições iniciais seguintes: $P_{s0}(0)=1$ (sistema inicialmente em bom estado) e $P_{s1}(0)=P_{s2}(0)=P_{s3}(0)=0$, pode-se obter as seguintes soluções:

$$P_{s0}(t) = e^{-(\lambda_{01}+\lambda_{02})t}$$

$$P_{s1}(t) = \frac{\lambda_{01}}{\lambda_{01}+\lambda_{02}-\lambda_{13}}\left[e^{-\lambda_{13}t} - e^{-(\lambda_{01}+\lambda_{02})t}\right]$$

$$P_{s2}(t) = \frac{\lambda_{02}}{\lambda_{01}+\lambda_{02}-\lambda_{23}}\left[e^{-\lambda_{23}t} - e^{-(\lambda_{01}+\lambda_{02})t}\right] \quad [25]$$

A probabilidade $P_{s3}(t)$ pode ser calculada usando a última das relações [24].

Se o sistema considerado é série, isto é, a quebra de qualquer componente leva o sistema a um estado não operacional, a confiabilidade do sistema será dada por:

$$R_s(t) = P_{s0}(t) = e^{-(\lambda_{01}+\lambda_{02})t} \qquad [26]$$

Se o sistema é paralelo, a quebra de um componente não impede que o sistema continue operacional, logo:

$$R_p(t) = P_{s0}(t) + P_{s1}(t) + P_{s2}(t) =$$

$$= \left[1 - \frac{\lambda_{01}}{\lambda_{01}+\lambda_{02}-\lambda_{13}} - \frac{\lambda_{02}}{\lambda_{01}+\lambda_{02}-\lambda_{23}}\right] e^{-(\lambda_{01}+\lambda_{02})t} +$$

$$+ \frac{\lambda_{01}}{\lambda_{01}+\lambda_{02}-\lambda_{13}} e^{-\lambda_{13}t} + \frac{\lambda_{02}}{\lambda_{01}+\lambda_{02}-\lambda_{23}} e^{-\lambda_{23}t} \qquad [27]$$

Na relação [27] os coeficientes λ não são funções do tempo mas, ainda assim, a equação é bastante geral. Muitas simplificações podem ser consideradas se os componentes do sistema são todos iguais, com as mesmas taxas de falha.

4.3 CONSIDERAÇÕES SOBRE FALHAS DEPENDENTES.

Um sistema com componentes em paralelo pode ser constituido de modo que um componente possa falhar sem que o sistema deixe de operar (normalmente com algum decréscimo de sua performance). O sistema falha apenas quando todos os componentes

em paralelo falham. Ainda não considerando o reparo de um componente que falhou, vê-se que o cálculo da confiabilidade de um sistema paralelo depende da função densidade de falha de cada componente. Se a operação de um componente afeta significativamente o ambiente em que o sistema opera que, por sua vez, tem sua taxa de falhas (ou de um outro componente) dependente de algum parâmetro ambiental, ter-se-a uma situação de falha dependente. Um caso semelhante ocorre quando uma variavel fluxo em um componente altera sua taxa de falhas como, por exemplo, em uma fonte de alimentação comum a vários subsistemas eletrônicos. Um outro exemplo é a operação de máquinas cuja taxa de falhas depende da intensidade do fluxo de peças.

Em principio, a confiabilidade de um sistema com componentes em paralelo será dada pela expressão genérica da probabilidade de sussesso [53, 135]:

$$R(t) = P(x_1 + x_2 + \ldots + x_n) =$$

$$= P(x_1) + P(x_2) + \ldots + P(x_n) -$$

$$- [P(x_i\, x_j) + P(x_i\, x_3) + \ldots + P(x_i\, x_j)] +$$

$$+ [P(x_i\, x_j\, x_k) + \ldots\ldots + P(x_i\, x_j\, x_k)] +$$

$$+ \ldots + (-1)^{n-1} \cdot P(x_1\, x_2 \ldots x_n) \qquad [28]$$

onde cada termo pode ser expresso por:

$$P(x_i x_j \ldots x_1) = P(x_1)P(x_2/x_1)P(x_3/x_2 x_1)\ldots$$
$$l=2,3\ldots n$$
$$\ldots P(x_l/x_1 x_2 \ldots x_{l-1}) \qquad [29]$$

A interpretação e o cálculo dos termos das probabilidades condicionais não é muito clara nem simples para problemas físicos, e as técnicas desenvolvidas por Markov são melhores de se aplicar em certos casos, como é o caso da solução pelo grafo da figura 2.

A relação [27] pode ser aplicada diretamente aos casos em que a taxa de falhas dos componentes muda em função de algum parâmetro ambiental ou do processo, mas não do tempo, conforme mostrado na figura 3.

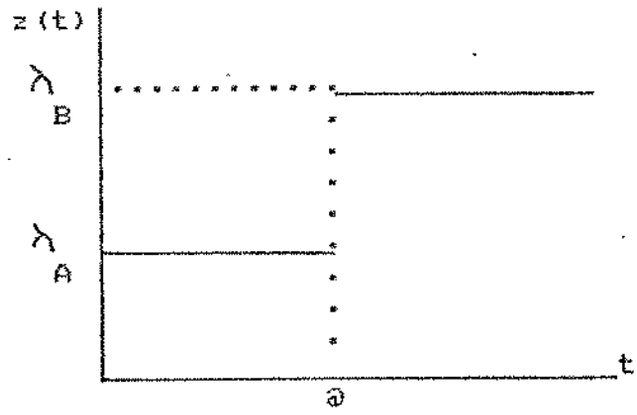


Figura 3 - Modelo de Falha Dependente, Constante no Tempo.

Nesta figura λ_A representa a taxa de falhas (cons-

tante) de um componente e λ_B representa a taxa de falhas (também constante) após a falha de outro componente, que ocorre no instante ∂.

Por outro lado, a mesma relação [27] pode também ser aplicada a sistemas com sobressalentes. Se o componente x_1 está ligado e x_2 é sobressalente, $\lambda_{01}=\lambda_A$. A taxa λ_2 será igual a zero se o sobressalente é frio e supostamente não pode falhar no estado desligado; e será igual a λ_C se pode falhar no estado desligado ou se é um sobressalnte quente, energizado mas não conectado ao sistema. Também $\lambda_{13}=\lambda_B$ e λ_{23} não precisa ser especificado se $\lambda_{02}=0$ pois, neste caso, o estado s_2 tem uma probabilidade que é zero, ou ainda igual a λ_D se $\lambda_{02}=\lambda_C$. De qualquer modo, a taxa de falhas para t>∂ independe de ∂, embora o instante da mudança súbita da taxa de falhas seja uma função de ∂.

O problema se altera completamente se z(t) é uma função do tempo e, neste caso, outras técnicas devem ser aplicadas, como por exemplo, o método da junção de função densidade ou o método dos estados intermediários [21, 38, 39,40, 135].

4.4 - NÚMERO DE ESTADOS DO SISTEMA.

A complexidade de um modelo de Markov depende diretamente do número de estados do sistema considerado, já que, a cada estado, corresponde uma equação diferencial. O número de estados m de um sistema de n componentes é dado por:

$$m = \binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \ldots + \binom{n}{n} = 2^n \qquad [30]$$

Nem sempre é necessário determinar quais componentes falham, bastando saber quantos. Neste caso, o número de estados diminui para n+1. Um grafo de Markov compactado pode ser usado para esse fim.

4.5 - SISTEMAS COM COMPONENTES REPARAVEIS .

Sempre que o tempo e os custos medios de reparo de uma peça ou componente de um equipamento constituem uma fração do custo inicial do equipamento, o reparo do componente deve ser considerado. O tempo entre falhas, o tempo de reparo, o número de falhas em um certo intervalo de tempo e o tempo relativo de operação, são valores de mérito que devem sempre ser considerados, além da confiabilidade e disponibilidade.

A disponibilidade A(t) é uma função que descreve os benefícios do reparo, em sistemas que podem tolerar períodos

sem operação ou de operação com decréscimo de performance. é definida como a probabilidade de que o sistema está operando no instante t, contrastando com o conceito de confaibilidade R(t), que é a probabilidade de que o sistema opera durante o intervalo 0 a t. A disponibilidade não contém informações sobre falhas anteriores do sistema, ou por quantos ciclos de falha-reparo o sistema já passou. Para um único componente não reparavel, A(t)=R(t). Para um componente reparável A(t)>=R(t).

Em sistemas com redundâncias, o reparo pode beneficiar tanto a confiabilidade quanto a disponibilidade. Em um sistema de dois componentes em paralelo, não havendo reparo o sistema falha quando ambos os componentes falham. Havèndo reparo, quando um componente falha e o outro continua operando, o primeiro pode ser reparado e recolocado em operação antes que o segundo falhe. A próxima falha poderá ser no primeiro ou no segundo componente. O sistema permanece operacional enquanto o tempo de reparo for menor que o intervalo entre falhas sussessivas. A longo prazo, poderá ocorrer uma falha no componente sobrevivente antes que o componente que falhou seja reparado. Neste caso, o sistema falha. Entretanto, a confiabilidade do sistema é, claramente, aumentada por meio do reparo.

Se as taxas de falha λ forem consideradas constantes, assim como as taxas de reparo μ, isto é, se as respectivas funções densidade são exponenciais, modelos de Markov podem ser utilizados. Os modelos para confiabilidade e para disponibilidade diferem, pois é necessário estabelecer estados finais para o modelo de confiabilidade, isto é, estados a partir dos quais o

sistema não transita para nenhum outro estado.

O grafo de Markov para a confiabilidade de um único elemento não reparavel é o da figura 1, onde $z(t)=\lambda$. As equações [20] fornecem:

$$R(t) = P_{s0}(t) = 1 - P_{s1}(t) = e^{-\lambda t} \qquad [31]$$

O grafo de Markov para a disponibilidade de um único componente reparavel é o da figura 4.

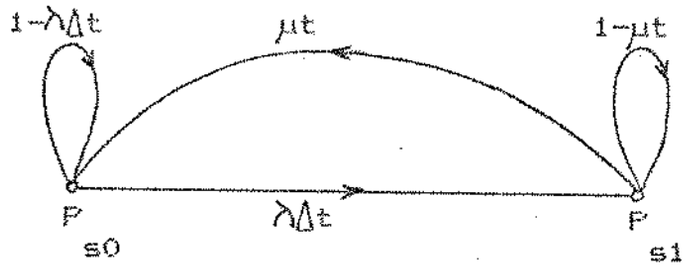


Figura 4 - Grafo de Markov para Um Componente Reparavel.

Usando-se as regras já estabelecidas, as equações diferenciais serão:

$$\dot{P}_{s0}(t) = -\lambda P_{s0}(t) + \mu . P_{s1}(t)$$

$$\dot{P}_{s1}(t) = \lambda P_{s0}(t) - \mu . P_{s1}(t)$$

$$P_{s0}(0) = 1 \quad e \quad P_{s1}(0) = 0$$

das quais pode-se obter:

$$A(t) = P_{s0}(t) = \frac{\mu}{\lambda+\mu} + \frac{\lambda}{\lambda+\mu} e^{-(\lambda+\mu)t} \qquad [32]$$

A disponibilidade de estado estacionário $A_{ss}(t)$, corresponde ao limite de $A(t)$ quando $t \to \infty$, sendo, neste caso:

$$A_{ss}(t) = \frac{\mu}{\lambda+\mu} \qquad [33]$$

enquanto que o limite da confiabilidade quando $t \to \infty$ é zero.

4.6 - OBSERVAÇÕES FINAIS.

Grafos de Markov podem ser estabelecidos para sistemas reparáveis com qualquer número de componentes. Os modelos são particularmente aplicáveis a sistemas de manufatura e a sistemas eletrônicos, onde as taxas de transição de estado operacional para estado de falha são constantes no tempo. Os efeitos do reparo de componentes e de subsistemas pode ser incluído nos modelos, desde que as taxas de reparo sejam também constantes no tempo. Para sistemas com muitos estados, o modelo na forma de grafo pode ser estabelecido com facilidade. O equacionamento e a solução analítica, sempre desejáveis, podem apresentar-se muito complexos. Para sistemas com estados regularmente distribuídos no grafo, as soluções analíticas apresentam características de simetria, permitindo induções e generalizações.

CAPITULO V

DESENVOLVIMENTO DE NOVOS MODELOS.

5.1 - MODELOS MATEMATICOS PARA SISTEMAS TOLERANTES A FALTAS.

Quando se realiza a análise de confiabilidade de um sistema complexo, é muito difícil tratá-lo como uma entidade completa. Decompor o sistema em subunidades funcionais, compostas de elementos, componentes e subsistemas, é uma forma óbvia de tratamento. Em geral supõe-se que que cada unidade pode assumir dois estados: bom estado e mau estado ou então sem falha e com falha [38, 41]. Embora analisáveis por modelos de Markov, muitas vezes é mais conveniente estabelecer uma descrição tipo diagrama de blocos, e o cálculo da confiabilidade é realizado em termos de confiabilidade das subunidades. Estruturas série-paralelo ocorrem com freqüência, assim como combinações delas. A análise combinatória, embora simples, pode ser aplicada a sistemas bastante complexos, desde que apresentem essa estrutura [39, 60, 70].

5.1.1 - ANALISE COMBINATORIA

A operação de um sistema que depende da operação correta de todos os seus componentes, configura uma estrutura

série, cuja confiabilidade é:

$$R_s(t) = P(x_1)P(x_2/x_1)\ldots P(x_n/x_1 x_2 \ldots x_{n-1}) \qquad [34]$$

onde estão expressas as probabilidades condicionais. Se não há interação entre os componentes, as falhas são independentes e, neste caso:

$$R_s(t) = \prod_{i=1}^{n} P(x_i) \qquad [35]$$

Em alguns sistemas, há mais de um meio de realizar uma mesma função. No caso de falha de um meio (ou mais), o sistema ainda pode operar corretamente. O sistema pode, então, ser representado por uma estrutura paralela onde n meios estão conectados entre a entrada e a saida, e todos devem falhar para que o sistema falhe. Essa configuração é, às vezes, chamada de sistema redundante. Entretanto, adotando uma nomenclatura mais coerente, o termo redundante será usado para designar a introdução proposital de elementos, com a finalidade de aumentar a confiabilidade do sistema ou para dar-lhe características de tolerância a faltas.

A probabilidade de sussesso de uma configuração paralela pode ser expressa como:

$$R_p(t) = 1 - P(\bar{x}_1)P(\bar{x}_2/\bar{x}_1)\ldots P(\bar{x}_n/\bar{x}_1 \bar{x}_2 \ldots \bar{x}_{n-1}) \qquad [36]$$

onde a barra indica que o componente falhou.

Supondo que as falhas são independentes:

$$R_p(t) = 1 - \prod_{i=i}^{n} P(\bar{x}_i) \qquad [37]$$

Nos sistemas cujo sussesso de operação depende de que k unidades dentre as n que o compõe, ou sistemas k dentre n, tem n-k unidades de reserva (k<=n). Se todas as unidades são idênticas, a probabilidade de sussesso é dada por:

$$R_{k/n}(t) = \sum_{i=k}^{n} \binom{n}{i} . R^{i} . (1-R)^{n-i} \qquad [38]$$

onde R é a probabilidade de sussesso de qualquer unidade. Se as unidades não são idênticas, será necessário enumerar explicitamente todas as combinações que resultam em sussesso, e que serão em número de $\binom{n}{k}$. Para k=n a fórmula corresponde a um sistema série, e para k=1 corresponde a um sistema paralelo.

Em um sistema paralelo, todos os canais estão ativos desde o inicio (t=0) e operam até que uma falha ocorre, permanecendo conectados no sistema mesmo após a falha.

Nos sistemas com sobressalentes, nem todos os canais paralelos estão conectados e funcionando ao mesmo tempo. Para um sistema constituido de N elementos idênticos, dos quais N-1 são sobressalentes, supondo que os elementos não ativos tem taxa de falhas iguais a zero e que o intervalo de tempo até a falha tem distribuição exponencial, isto é, que a taxa de falhas é constante, a confiabilidade pode ser determinada por:

$$R_{ss}(t) = e^{-\lambda t} \sum_{i=0}^{N-1} \frac{(\lambda t)^i}{i!} \qquad [39]$$

Estruturas envolvendo N componentes idênticos, dos quais pelo menos k (k<=N) são necessários para que o sistema opere normalmente, e N-k são sobressalentes, terão maior confiabilidade que uma estrutura paralela k dentre n, sendo expressa pela equação:

$$R_{k/N}(t) = e^{-k\lambda t} \sum_{j=0}^{N-k} \frac{(k\lambda t)^j}{j!} \qquad [40]$$

5.1.2 - CONSIDERAÇÕES SOBRE REDUNDANCIAS .

Em termos simples, usa-se elementos criando novos canais paralelos na estrutura de um sistema, com a finalidade de melhorar sua confiabilidade. A redundância é um requisito fundamental para a sobrevivência do sistema na ocorrência de falhas. Muitas formas de redundância podem ser estabelecidas (veja capitulo II). Algumas técnicas especiais foram desenvolvidas especificamente para tratar sistemas digitais, onde a redundância é usada com freqüência e por várias razões, entre as quais pode-se citar:

1. a natureza binária dos sinais digitais,

2. a possibilidade de especificar um sistema por meio de um conjunto de funções lógicas ou tabelas de estado ló-

gico, e que podem ser implementadas por diferentes circuitos,

3. apenas alguns poucos circuitos digitais básicos ou blocos lógicos são suficientes para a implementação de um vasto conjunto de funções, possibilitando o uso de redundância a nível de componente eletrônico ou a nível de função.

4. a possibilidade de usar a diversidade de circuitos redundantes para as mesmas funções [9].

Uma configuração clássica deve aqui ser avaliada, para fins de comparação. A conhecida redundância modular tripla TMR ou voto de maioria, constituida por tres módulos idênticos cujas saídas são comparadas por um circuito de voto, constituem uma forma de redundância estática que mascara as faltas mas não as elimina. Supondo que o circuito de voto é perfeito (não quebra), a confiabilidade do sistema TMR é dada por:

$$R_{TMR} = 3R^2 - 2R^3 \qquad [41]$$

Trata-se de um caso particular da equação [38]. Esse modelo é considerado demasiado pessimista [24, 25] já que, em muitos casos, um módulo falha e a operação continua correta. O sistema TMR/simplex deteta uma discordância de resultados, desliga o módulo que falhou e prossegue com um dos dois módulos que não falharam. A confiabilidade do sistema TMR/simplex é dada por:

$$R_{TMR/S} = \frac{3}{2}R - \frac{1}{2}R^3 \qquad [42]$$

A confiabilidade do sistema TMR pode ser aumentada usando linhas triplicadas, operação TMR e n-3 sobressalentes. Quando uma unidade falha, um sobressalente é ciclicamente conectado. Para um sobressalente apenas, a expressão da confiabilidade é dada por:

$$R_{TMR/1} = 6R^2 - 8R^3 + 3R^4 \qquad [43]$$

Por cálculo direto vê-se que $R_{TMR/1} > R_{TMR/s}$ e que $R_{TMR/1} > R_{TMR}$ para valores de R maiores que 0,5. Ainda outras análises podem mostrar que um sistema TMR apresenta muito maior confiabilidade do que as equações clássicas levam a concluir[25]. O tempo médio entre falhas é uma forma muito pobre para ser usada como avaliação comparativa e que o sistema TMR deverá ser aplicado em sistemas cujo tempo de missão é bastante menor que $1/\lambda$.

Em geral, o projeto de módulos com sobressalentes é mais difícil devido à inclusão dos circuitos adicionais de verificação e de reconfiguração. Por meio de um projeto adequado, as falhas dos circuitos de comutação podem ser indistinguíveis das faltas dos módulos receptores e, desse modo, suas taxas de falha podem ser englobadas nestes [24]. Estes estudos e análises mostram a aplicabilidade da redundância com sobressalentes em computadores de controle industrial.

Em arquiteturas de computadores, um parâmetro a mais deve ser considerado. A suficiência (coverage) é definida como um parâmetro condicional da efetiva recuperação do sistema, por ocasião de uma falta e tentativa de reconfiguração, e que

prossegue processando as informações, sem perda de informações essenciais, ou seja:

$$c = Pr(\text{sistema recupera} \mid \text{sistema falha}) \qquad [44]$$

Este parâmetro constitue a probabilidade de que o sistema sobrevive sem danos irreparáveis, após uma falha. Exatamente em que se constitue e que valor deve assumir é um problema a ser definido pelo projetista e para cada problema particular. A confiabilidade é muito sensível a este parâmetro, de modo que supor c=1 é, no mínimo, temerário [24]. Incorporando o parâmetro, um sistema de N componentes sendo N-k sobressalentes, terá por confiabilidade:

$$R_{k/N}(t,c) = e^{-k\lambda t} \sum_{j=0}^{N-k} \frac{(ck\lambda t)^i}{i!} \qquad [45]$$

5.1.3 - EFEITO DO NUMERO DE MODULOS DE RESERVA E SOBRESSALENTES.

Importantes conclusões podem ser obtidas a partir de uma análise das expressões [38] para sistemas paralelos k dentre n e [40] para sistemas com N-k sobressalentes. Os gráficos da figura 1 mostram a dependência da confiabilidade com o número de elementos redundantes.

Na figura 2, mostra-se a dependência do crescimen-

to da confiabilidade com o aumento do número de elementos redundantes, para c=1 e para c>1.

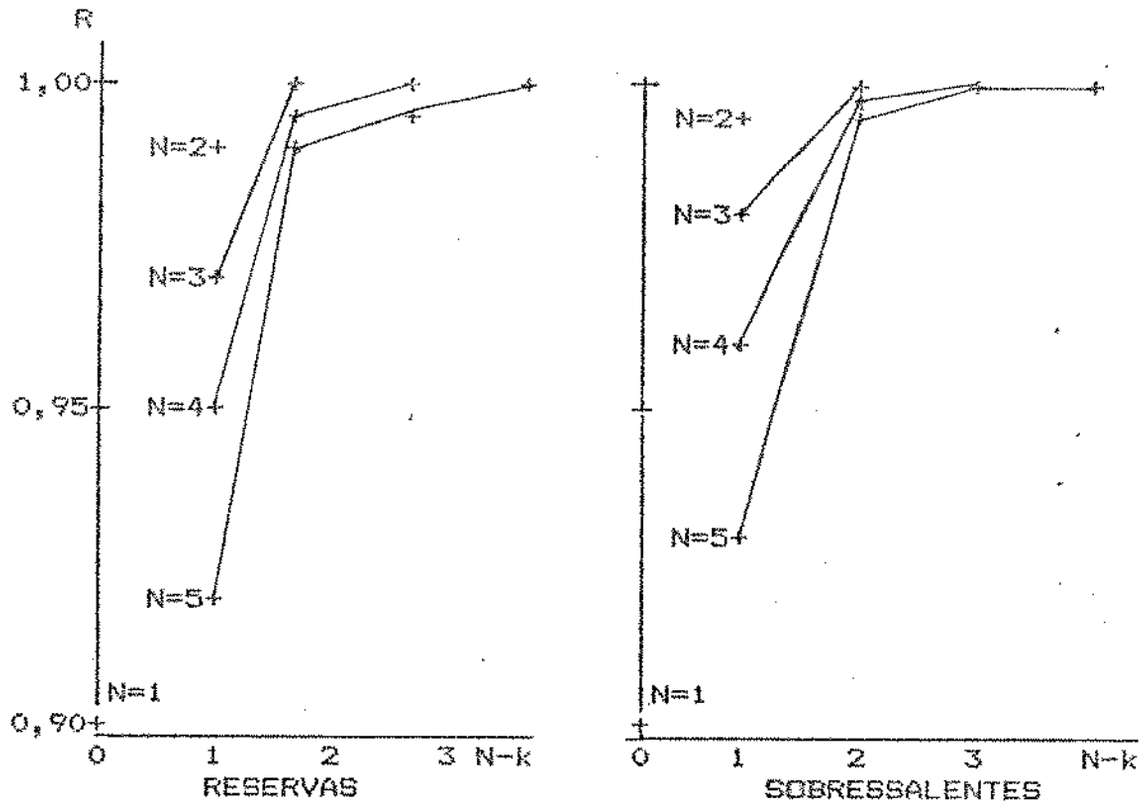


Figura 1 - Confiabilidade em Função do Número de de Elementos Redundantes.

Observa-se que, tanto para redundâncias de reservas paralelas ou quentes quanto para redundâncias com sobressalentes ou frias, a confiabilidade é uma função saturável com o aumento do número de elementos redundantes.

Caso se considere o parâmetro suficiência c, a confiabilidade é uma função que tem um máximo, localizado nas vizinhanças do valor 2 para número de sobressalentes.

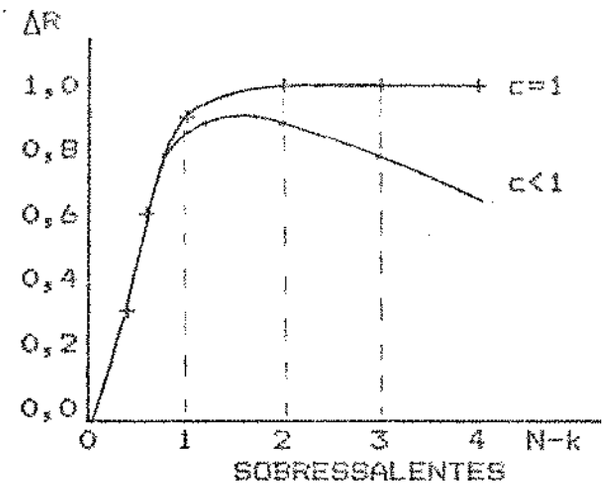


Figura 2 - Crescimento da Confiabilidade com o Número de Sobressalentes, para Suficiência c=1 e c<1.

5.2 - CONSIDERAÇÕES SOBRE FALTAS DE FABRICAÇÃO E FALTAS DE OPERAÇÃO .

Na arquitetura de um sistema tolerante a faltas, devemos encontrar diferentes tipos de subsistemas empregados com redundâncias para efeito de tolerância a faltas. O modelo matemático a ser empregado em estudos de comportamento de sistemas desta natureza, deve incluir as diferenças entre esses subsistemas, tanto estruturais como as relativas a parâmetros como confiabilidade, taxas de falhas, etc. e, ainda, as funcionais, quando for este o caso.

Para o tipo de análise que se seguirá, por hipótese, o sistema será constituido por i tipos (i = 1, 2, ...n) de subsistemas. Cada tipo de subsistemas poderá admitir certo número de defeitos, admitindo-se até s_i elementos com defeito de fabricação. Os subsistemas de tipo i poderão ser reconfigurados de tal modo que, se o número de elementos com defeito é menor ou igual a s_i, o sistema continua operacional. Se o número de defeitos é menor que s_i, o sistema possuirá redundância residual, ao ser colocado em operação. Se o número de defeitos é igual a s_i, o sistema não terá mais redundâncias mas, ainda assim, poderá beneficiar-se da capacidade de tolerância a faltas do sistema como um todo, desde que uma degradação suave possa ser admitida.

Para um sistema com tal grau de complexidade no que se refere à quantidade de possiveis estados, a modelagem por modelos de Markov se constitue uma escolha natural. Modelos podem

ser desenvolvidos para se obter expressões de confiabilidade e de performance de arquiteturas com tolerância a faltas de fabricação e com tolerância a faltas operacionais.

Diversas formas de redundância, incorporadas no "chip" e no "wafer", já desde seu projeto, tem sido propostas recentemente [50, 84, 154]. As propostas incluem a interconexão de alguns poucos tipos de elementos na forma de arquiteturas ditas regulares, ou seja, usando módulos padrões de forma a constituir estruturas modulares, em um único "chip" ou em um único "wafer". Por meio de redundâncias, a tolerância a faltas pode ser alcançada. Além disso, melhora-se a performance do sistema por meio da maior disponibilidade [30].

Modelos analíticos de arquiteturas com redundância a nivel VLSI ou WSI permitem avaliar os resultados de uma dada estratégia de tolerância e a quantidade de redundância a ser incorporada, quando tipos diferentes de elementos são considerados, abrindo novas perspectivas para aplicações em controladores industriais [3, 69, 82, 83, 115, 117, 152]. Importantes inovações devem ocorrer em projetos de arquiteturas multi-elementos em uma única placa ("chip"), apresentando o potencial de construção de sistemas multi-processadores, eliminando as conexões externas, o que ainda não foi devidamente aproveitado nem mesmo nos supercomputadores, mas deverá passar a ser muito breve [42, 44, 57, 138].

Do ponto de vista da tolerância a faltas, há dois aspectos a considerar: a redução da quantidade de defeitos que podem aparecer durante os processos de fabricação e o aumento da performance do sistema quando em operação. Na medida em que a

densidade de componentes em um circuito aumenta e, de outro lado suas dimensões diminuem, a expectativa de defeitos e a consequente porcentagem de rejeitos torna-se um problema de grande significado. A melhoria dos processos de fabricação e dos testes é a solução óbvia. No entanto esta solução é, em primeiro lugar, muito cara e, em muitos casos, difícil (se não impossível) de implementar. Dessa forma, a incorporação de redundâncias para fins de tolerância a faltas de fabricação torna-se uma solução opcional e efetivamente prática, tendo sido demonstrada em certos circuitos de alta densidade [30]. Em lugar de rejeitar um circuito ou subsistema, pode-se aceitá-lo como em bom estado, apesar de apresentar algum defeito de fabricação, utilizando-se a reestruturação dos circuitos.

De modo similar, o aumento de componentes ou subsistemas e de suas interconexões torna mais difícil atingir a operação com alta confiabilidade e, também neste caso, redundâncias podem ser incorporadas ao sistema de modo a incrementar a confiabilidade e melhorar a performance.

Modelos analíticos permitem a análise e avaliação dos efeitos de técnicas de tolerância a faltas, examinar diferentes topologias e determinar a quantidade de redundância que pode ou deve ser acrescentada ao sistema, dentro das limitações de certos parâmetros como custo. Diferentes esquemas podem resultar mais convenientes ou apresentar menores custos em diferentes situações e para diferentes objetivos. Na avaliação de uma estratégia de tolerância a faltas, os seguintes aspéctos devem ser considerados:

1. Tipo de faltas a serem toleradas:

a) Faltas na fabricação (relativamente comuns e variando com os processos de fabricação),

b) Faltas de operação (mais raras e dependentes da confiabilidade atribuida ao sistema),

2. Custos associados às faltas:

a) Faltas de fabricação (custos de fabricação apenas),

b) Faltas de operação (custos de fabricação + custos de operação + prejuizos das funções aplicativas).

As vantagens em termos de custo de um determinado método para tolerância a faltas de fabricação pode não ser vantajoso para faltas de operação, e pode também variar muito para diferentes processos defabricação.

Em circuitos integrados VLSI, faltas de ambos os tipos, isto é, de fabricação e de operação, não admitem reparo da parte que falhar. A reestruturação interna deve ser usada e, em função disso pode-se adotar as seguintes capacidades de reestruturação:

1. Reestruturação estática na fabricação (por exemplo, fusiveis, conexões, componentes simples, etc.),

2. Reestruturação dinâmica na operação (uso de reservas ou sobressalentes de subsistemas),

o que, por outro lado, se traduz em investimento adicional de hardware na forma de:

1. Componentes e chaves lógicas redundantes (redundâncias e chaves para sua conexão),

2. Processadores e conexões redundantes (barramentos, etc.)

Os processadores são, tradicionalmente, considerados como o recurso mais importante de um sistema de processamento [91]. Elementos para comutação são adicionados para a interconexão de processadores e melhorar sua utilização [128, 139]. Linhas de interconexão adicionais podem ser usadas para conectar os processadores em bom estado. O espaço, ou seja, a área de circuito necessária para esses fins deve ser muito menor que os próprios processadores, para que se tenha alguma vantagem substancial. Deve-se considerar, nestas soluções, que os elementos adicionais também podem falhar e não apenas os processadores. Quanto mais componentes forem usados maior será sua suscetibilidade para apresentarem defeitos ou faltas.

Ainda outro aspécto que se deve considerar são os seguintes modos alternativos para introduzir redundâncias:

1. Na arquitetura do elemento básico (ou módulo),

2. Na arquitetura do sistema,

podendo-se incorporar ambos modos no mesmo sistema.

5.3 - MODELOS DE "CHIPS" COM TOLERANCIA A FALTAS.

Expressões para cálculo de probabilidade de se ter, em n_i elementos de tipo i, x_i defeitos de fabricação, são baseadas na distribuição binomial negativa que se enquadra razoavelmente em dados de produção de circuitos integrados [84, 140, 141], podem ser obtidas na forma:

$$P(X_i = x_i) = \frac{\Gamma(x_i + \alpha_i)}{x_i ! \, \Gamma(\alpha_i)} \cdot \frac{\left(\frac{A_i d_i}{\alpha_i}\right)^{x_i}}{\left(1 + \frac{A_i d_i}{\alpha_i}\right)^{\alpha_i + x_i}} \qquad [46]$$

onde: d = densidade média de defeitos

α = parâmetro representativo de aglomeração de defeitos

A = área de circuito que contém os elementos tipo i.

Em "chips" sem redundancias, a probabilidade de se ter zero defeitos d fabricação em todos os elementos é dada por:

$$Y = \prod_{i=1}^{n} P(X_i = 0) = \prod_{i=1}^{n} \left(1 + \frac{A_i d_i}{\alpha_i}\right)^{\alpha_i} \qquad [47]$$

onde: n = número total de diferentes tipos de elementos.

Em circuitos com redundâncias, pode-se assumir as seguintes hipóteses:

1. há N_i elementos de tipo i
2. até s_i elementos com defeito podem ser tolerados.

A probabilidade de que o circuito possa ser aceito é igual à probabilidade de que qualquer número de defeitos é tolerável desde que sejam restritos a s_i elementos do tipo i.
Isto pode ser expresso como

$$Y = \prod_{i=1}^{N_i} P(\text{há defeitos em apenas } s_i \text{ elementos})$$

$$= \prod_{i=1}^{N_i} \sum_{j=0}^{s_i} a_j^{(i)} \qquad [48]$$

onde: $a_j^{(i)}$ = P(os defeitos de elementos tipo i estão distribuidos em j dentre N_i elementos)

Se Y_i é a probabilidade de zero defeitos em um único elemento de tipo i, isto é,

Y_i = P(zero defeitos de um único elemento tipo i)

então $a_j^{(i)}$ poderá ser calculado pela expressão

$$a_j^{(i)} = \binom{N_i}{j} \binom{N_i - j}{i} Y_i^{\,j} (1 - Y_i)^{\,j} \qquad [49]$$

5.4 - ESTRATÉGIAS PARA A RECUPERAÇÃO DO SISTEMA.

Um modelo de Markov aplicável a um sistema que tem i tipos de elementos contendo j defeitos dentre N_i elementos pode ser estabelecido. O grafo de Markov para esta caso está mostrado na figura 1.

Assumem-se as seguintes hipóteses:

1.há um único estado de falha do sistema F; nos outros estados o sistema é operacional mesmo que existam elementos com faltas,

2. há f_i elementos com falta,

3. m_i elementos de tipo i podem falhar se s_i elementos deste tipo já são defeituosos no início da operação (t=0),

4. u_i é o número de elementos ativos (em uso para determinada função aplicativa),

5. N_i é o número total de elementos de tipo i.

Tem-se que, o número de elementos ativos de tipo i deve ser menor que o número de elementos residuais em bom estado, deste tipo, isto é:

$$u_i <= N_i - f_i \qquad [50]$$

e, por outro lado, f_i inclui elementos com defeitos de fabricação que podem atingir o valor s_i somados aos elementos tipo i que falham em operação que podem ser até m_i, ou seja:

$$f_i <= s_i + m_i < N_i \qquad [51]$$

Se, em t = 0, menos que s_i elementos de tipo i tem defeito, mais que m_i faltas em elementos tipo i podem ser toleradas para t > 0.

De um modo geral, até t = 0, isto é, até o início da operação do sistema, o sistema apresentará um único estado sem faltas (nenhum elemento de tipo i apresentou defeito de fabricação) e certo número de estados com algum defeito de fabricação. As transições entre esses estados será sempre no sentido de uma falta pois assume-se que não há reparo nestas fases. Após sua entrada em serviço, o sistema poderá apresentar, adicionalmente a esses estados, outros estados com faltas de operação. Na fase operacional do sistema pode-se supor que subsistemas que falham podem ser recuperados ou não. A transição para o estado de falha do sistema F ocorrerá a partir de qualque um dos estados descritos, quando um elemento tiver uma falta e o sistema não pode se recuperar ou reconfigurar para um estado operacional. Essa transição pode ocorrer, portanto, antes mesmo que o sistema entre em operação, por causa de uma falta de fabricação que produz efeitos no sistema de modo que ele atinge o estado F.

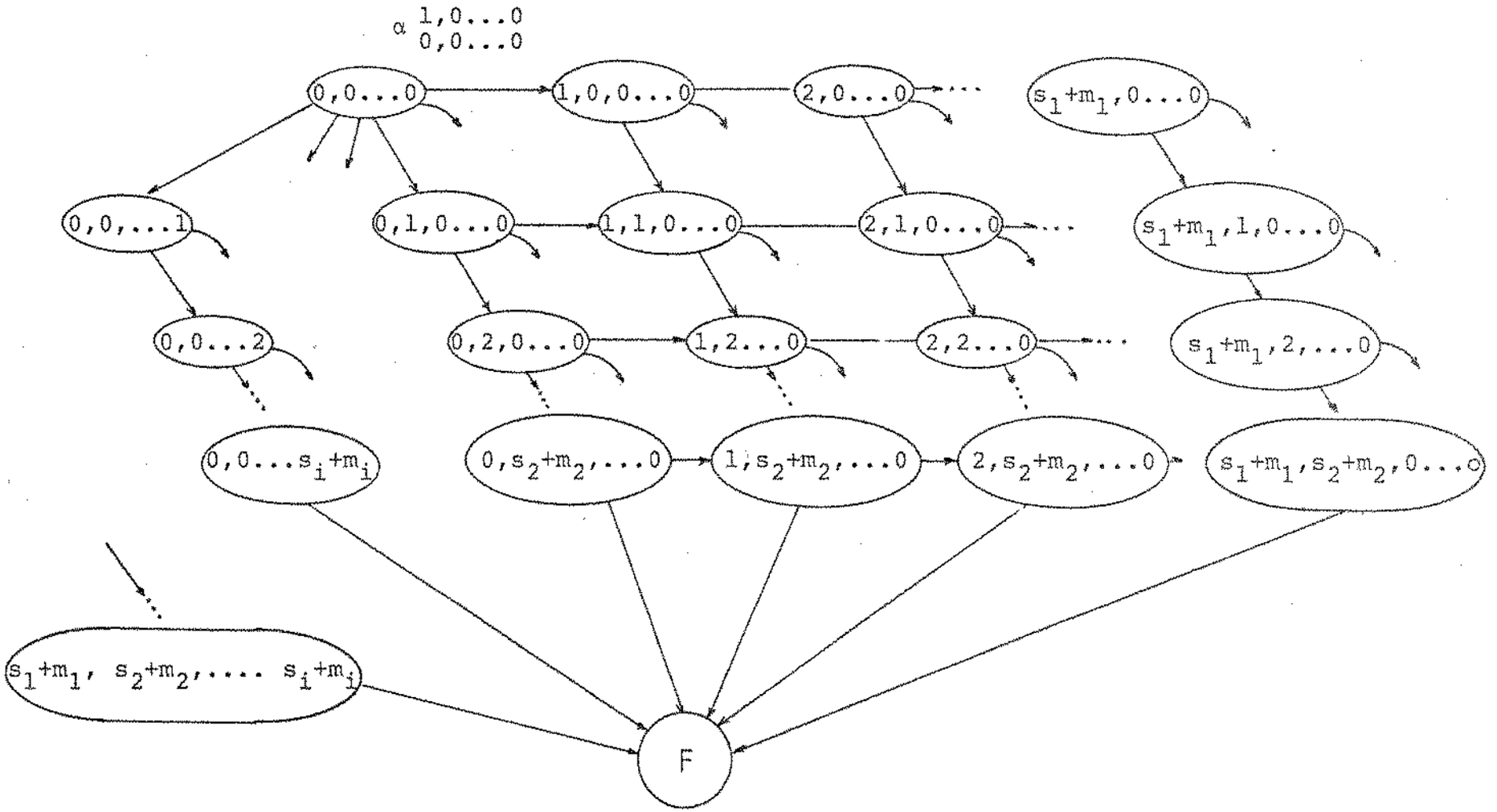


FIGURA 1 - DIAGRAMA DE MARKOV DE ESTADOS PARA i TIPOS DE SUBSISTEMAS

5.5 - EQUAÇÕES DIFERENCIAIS DO MODELO GERAL.

Utilizando o grafo de Markov da figura 1, pode-se adotar as hipóteses seguintes para estabecimento das equações de um modelo geral de um sistema:

1. O sistema é constituido por elementos ou subsistemas de i tipos diferentes,

2. Cada tipo de elemento admite até s_i faltas de fabricação e m_i faltas de operação,

3. Se o número de faltas, para $t >= 0$, é menor que s_i o sistema é operacional com redundância residual,

4. Se o número de faltas para $t >= 0$ é igual a s_i o sistema é operacional; ocorrendo mais uma falta no subsistema i, o sistema ainda poderá beneficiar-se da capacidade de tolerância a faltas com degradação suave,

5. Sendo N_i o número de elementos ou subsistemas do tipo i então

$$f_i <= s_i + m_i < N_i \qquad [52]$$

6. O número de elementos ativos, into é, elementos em uso para a execução das funções, deve ser

$$u_i <= N_i - f_i \qquad [53]$$

7. O estado genérico do sistema é definido pelo conjunto de

elementos ou subsistemas com falta f_i, ou seja

$$(f_1, f_2, \ldots, f_i) \qquad [54]$$

onde cada f_i é dado pela relação da hipótese 5.

8. Se, estando o sistema em um estado qualquer

$$(f_1, f_2, \ldots, f_i) \qquad [55]$$

ocorrer uma falta em excesso para algum tipo particular de subsistema e, em conseqüência, , o sistema não puder se recuperar, ou ainda se, como efeito de uma falta qualquer o sistema não puder se recuperar, mesmo havendo redundância residual, o sistema transita para o estado com falta F. Isto significa que o estado F pode ser alcançado a partir de qualquer outro estado, ou seja, que as probabilidades de transição de qualquer estado para o estado F não são nulas.

Define-se como taxa de transição de um estado genérico

$$(f_1, f_2, \ldots, f_i) \qquad [56]$$

para outro estado, também genérico

$$(f_1+1, f_2, \ldots, f_i) \qquad [57]$$

como conseqüência de uma falta em um subsistema de tipo 1, ou então outros estados equivalentes

$$(f_1, f_2+1, \ldots, f_i) \qquad [58]$$

por conseqüência a falta em elemento de tipo 2, etc., a seguinte expressão:

$$\alpha^{f_1,f_2,\ldots,f_i+1}_{f_1,f_2,\ldots,f_i} \qquad [58]$$

ou então

$$\alpha^{f_1,f_2+1,\ldots,f_i}_{f_1,f_2,\ldots,f_i} \qquad [59]$$

e assim por diante. Essas taxas de transição dependem das taxas de falha dos elementos do sistema e da probabilidade de que o sistema recupera com sussesso no caso de falta.

As taxas de transição de qualquer estado genérico para o estado com falta F, são definidas do mesmo modo, isto é,

$$\alpha^{F}_{f_1,f_2,\ldots,f_i} \qquad [60]$$

Também aqui, dois tipos de estratégias de recuperação do sistema podem ser adotadas, no caso de ocorrência de uma falta:

1. Após uma falta, todos os elementos remanescentes, de todos os tipos, são ativos. Não há, neste caso, sobressalentes não ativos. Para esta hipótese, a taxa de transição do sistema, de um estado genérico para outro, será dada pela expressão:

$$\alpha^{f_1,f_2,\ldots,f_i+1}_{f_1,f_2,\ldots,f_i} = (N_1 - f_1)\,\lambda_i P_{i\,1} \qquad [61]$$

onde: $N_1 - f_1$ corresponde ao número de elementos ativos, sem falta, de tipo 1,

λ_1 é a taxa de falhas de elementos de tipo 1, suposta constante.

P_1 = Pr(sistema recupera ! ocorre falta em elemento tipo 1)

De modo semelhante, define-se a taxa de transição do sistema, ao ocorrer falta de elemento de tipo 2, como sendo

$$\alpha_{f_1, f_2+1, \ldots, f_i}^{f_1, f_2, \ldots, f_i} = (N_2 - f_2)\, P_2 \qquad [62]$$

onde $N_2 - f_2$, e P_2 são definidos de modo análogo aos da equação [61].

As demais taxas de transição serão definidas da mesma maneira. Como o sistema depende da operação de todos os $(N_i - f_i)$ elementos remanescentes de cada tipo, a cada falta que ocorrer em um subsistema, o sistema sofrerá uma degradação de sua performance. Por outro lado, o sistema estará sempre utilizando ao máximo todas as suas capacidades.

2. O sistema será subdividido em elementos ativos e elementos não ativos, isto é, elementos que embora em bom estado não são necessários a determinada função do sistema. Se o número de elementos remanescentes sem falta é dado por

$$N_i - f_i \qquad [63]$$

então o número de elementos ativos deve ser sempre

$$u_i <= N_i - f_i \qquad [64]$$

Se um elemento não ativo falha, a probabilidade de que o sistema recupera dessa falha é igual a 1, isto é:

P(sistema recupera | ocorre falha em elemento não ativo) = 1

e, neste caso, a taxa de transição será composta por duas parcelas:

$$\alpha^{f_1+1,f_2,\ldots,f_i}_{f_1,f_2,\ldots,f_i} = u_i \lambda_i P_i + (N_i - f_i - u_i)\lambda_i \qquad [65]$$

Nesta expressão, a parcela $u_i \lambda_i P_i$ representa a taxa de transição do sistema, para elementos ativos de tipo i que tem taxa de falhas λ_i, e probabilidade de recuperação do sistema para falha de elemento tipo i como sendo P_i. Por outro lado, a parcela $(N_i - f_i - u_i)\lambda_i$ representa a taxa de transição do sistema para elementos não ativos do mesmo tipo i.

As taxas de transição do sistema para subsistemas de tipo 2, e demais tipos, serão análogas.

Deve-se distinguir a existência dos seguintes tipos distintos de estados do sistema:

1. Estados iniciais:

1.1. se não há defeitos de fabricação, o sistema inicia sua operação no estado que corresponde a considerar $f_1=0$, $f_2=0$,.. $f_i=0$ para t=0, e está representado no grafo de Markov como sendo $(0,0,\ldots,0)$. Existe apenas um estado deste tipo.

1.2. se há faltas de fabricação, considerando que cada subsistema admite até s_i faltas de fabricação, haverá

$$(s_1+1)(s_2+1)\ldots(s_i+1) \qquad [66]$$

possiveis estados em que o sistema pode iniciar a operação. Em todos esses estados, o número de faltas de operação de cada subsistema é zero, isto é,

$$m_i = 0 \quad \text{com } i=1,2,\ldots \qquad [67]$$

2. Estados operacionais:

Serão todos os estados

$$(f_1, f_2, \ldots, f_i) \qquad [68]$$

onde o número de faltas é dado por

$$f_i <= s_i + m_i \qquad [69]$$

isto é, são estados nos quais o número de faltas de fabricação é menor ou igual ao limite s_i, e o número de faltas de operação é menor ou igual ao limite m_i.

3. Estados terminais:

São estados em que já foi atingido o número limite de faltas de um ou mai tipos de elementos do sistema, isto é, estados como

$$(s_1+m_1, f_2, \ldots, f_i)$$

$$(f_1, s_2+m_2, \ldots, f_i)$$

...

$$(s_1+m_1, s_2+m_2, \ldots f_i)$$

...

$$(s_1+m_1, s_2+m_2, \ldots, s_i+m_i) \qquad [70]$$

A partir desses estados teminais, ocorrendo mais uma falta em qualquer um dos subsistemas de algum tipo, o sistema transita para o estado com falta F.

Os estados iniciais podem ser ilustrados para o caso de fabricação de um circuito integrado ou "chip". Durante a fabricação podem ocorrer k_i defeitos em elementos do tipo i, sendo

$$0 <= k_i <= s_i \qquad [71]$$

Supõe-se que redundâncias são adicionadas de modo que s_i defeitos podem ser tolerados por substituição de reservas j dentre N_i elementos do tipo i. Supondo, ainda, que a probabilidade de zero defeitos de fabricação em um circuito sem redundâncias seja dado pela expressão [47] então para um circuito com redundâncias, a probabilidade de que ocorram j dentre s_i defeitos, sendo N_i o número de elementos de tipo i, será dada por:

$$Y = \prod_{i=1}^{j} \sum_{j=0}^{s_i} \binom{N_i}{j} (1 - Y_i)^j \, Y_i^{\binom{N_i - j}{i}} \qquad [72]$$

onde Y_i é a probabilidade de zero defeitos para um elemento tipo i, cujos parâmetros A_i e o_i devem ser divididos por N_i, considerando que há N_i redundâncias e que a probabilidade de zero defeitos seja independente para cada elemento, e que corresponde à expressão [48].

Um estado inicial pode ser estabelecido na ocorrência de $j=k_i$ defeitos em cada tipo de elemento, isto é,

$$a^{(i)}_{k_i} = Pr(\text{há } k_i \text{ defeitos dentre } N_i \text{ elementos tipo } i) \qquad [73]$$

Considerando independentes as ocorrências em diferentes tipos de elementos, a probabilidade de que ocorram k_1 defeitos em elementos tipo 1, k_2 defeitos em elementos tipo 2, e assim sussessivamente, será dada por:

$$Pr(\text{há } k_1, k_2, \ldots, k_i \text{ defeitos de fabricação}) =$$

$$= a^{(1)}_{k_1} . a^{(2)}_{k_2} \ldots a^{(i)}_{k_i} \qquad [74]$$

Como existem $(s_1+1)(s_2+1)\ldots(s_i+1)$ diferentes estados possiveis, então a probabilidade de que o sistema opera com faltas de fabricação é dada por:

$$Pr(\text{sistema opera } ! \text{ há faltas de fabricação}) =$$

$$= \sum_{k_1=0}^{s_1} \sum_{k_2=0}^{s_2} \cdots \sum_{k_i=0}^{s_i} a^{(1)}_{k_1} a^{(2)}_{k_2} \ldots a^{(i)}_{k_i} \qquad [75]$$

Os estados terminais são dados pelas expressões [70].

A probabilidade de que o sistema está em um estado genérico $(f_1, f_2, \ldots, f_i)$ em operação, pode ser escrita com a seguinte notação:

$$P^{f_1 f_2 \ldots f_i}_{k_1 k_2 \ldots k_i} = Pr(\text{o sistema está no estado } (f_1, f_2, \ldots, f_i) \text{ no instante } t \mid \text{o sistema estava no estado inicial } (k_1, k_2, \ldots, k_i) \text{ no instante } t=0) \qquad [76]$$

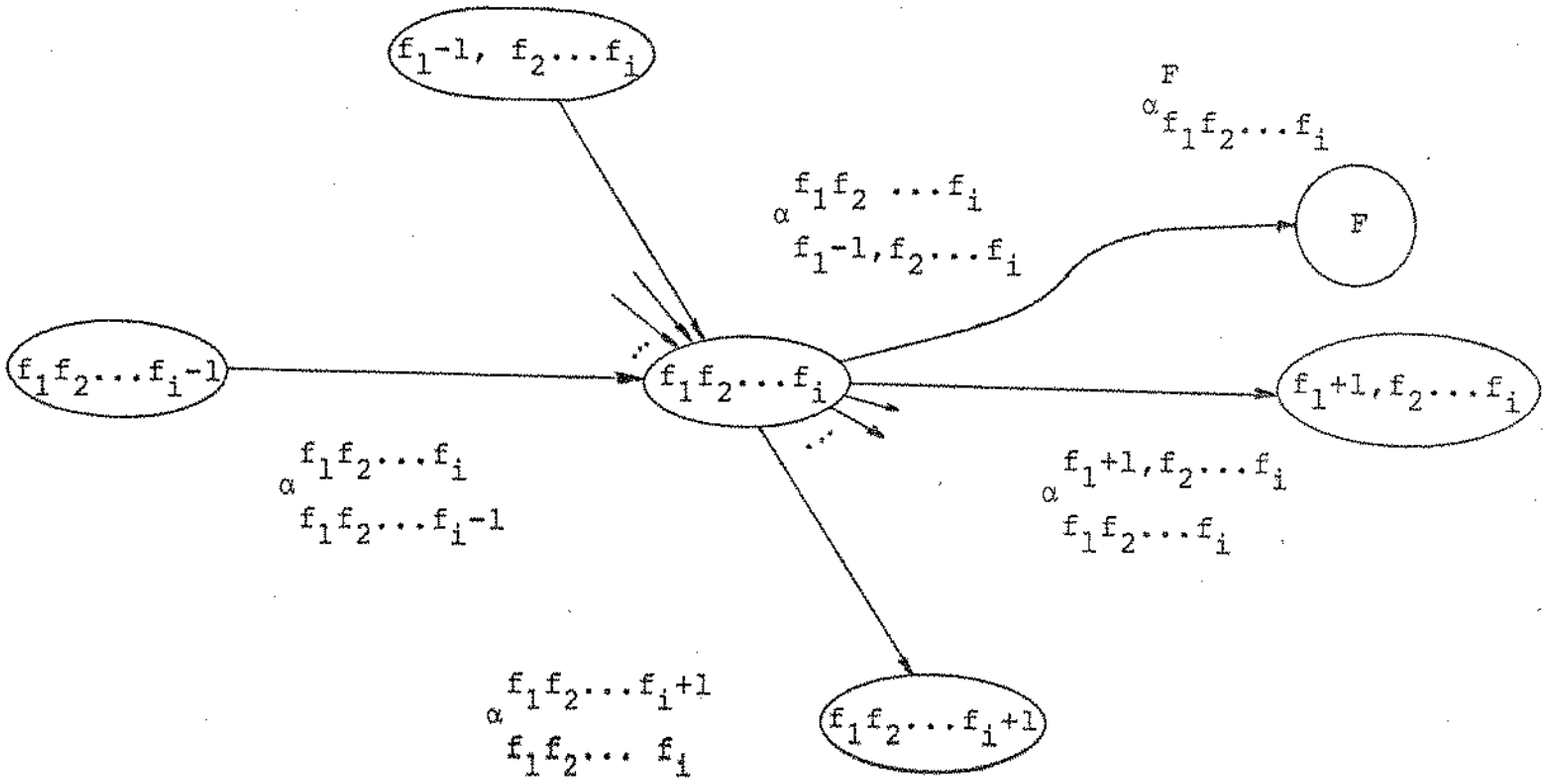

$f_1-1,\ f_2\ldots f_i$
$f_1f_2\ldots f_i-1$
$f_1f_2\ldots f_i$
$f_1f_2\ldots f_i+1$
F
$f_1+1, f_2\ldots f_i$
$\alpha^{F}_{f_1f_2\ldots f_i}$
$\alpha^{f_1f_2\ldots f_i}_{f_1-1,f_2\ldots f_i}$
$\alpha^{f_1f_2\ldots f_i}_{f_1f_2\ldots f_i-1}$
$\alpha^{f_1+1,f_2\ldots f_i}_{f_1f_2\ldots f_i}$
$\alpha^{f_1f_2\ldots f_i+1}_{f_1f_2\ldots f_i}$

FIGURA 2

No instante t=0 as condições iniciais serão dadas por:

$$P_{k_1\, k_2 \ldots k_i}(0) = 1$$

$$P_{k_1\, k_2 \ldots k_i}(0) = 0 \quad \text{para todo } f_i >= k_i \qquad [77]$$

Na figura 2, mostra-se com mais detalhes um estado genérico do sistema, conforme já definido, em que as transições e respectivas taxas de transição aparecem. Há transições possíveis de algum dos estados desde $(f_1 - 1, f_2, \ldots f_i)$ até $(f_1, f_2, \ldots, f_i - 1)$. Do estado genérico $(f_1, f_2, \ldots, f_i)$ podem ocorrer transições para qualquer um dos estados seguintes desde $(f_1 + 1, f_2, \ldots f_i)$ até $(f_1, f_2, \ldots, f_i + 1)$.

Desse modo, para de um estado inicial $(k_1, k_2, \ldots, k_i)$, a seguinte equação diferencial pode ser escrita:

$$\frac{d}{dt} P_{k_1\, k_2 \ldots k_i}(t) = -\left[\alpha^{k_1+1, k_2 \ldots k_i}_{k_i\, k_2 \ldots k_i} + \alpha^{k_1\, k_2+1 \ldots k_i}_{k_1\, k_2 \ldots k_i} + \ldots + \alpha^{k_i\, k_2 \ldots k_1+1}_{k_1\, k_2 \ldots k_i} + \alpha^{F}_{k_1\, k_2 \ldots k_i} \right] P_{k_1\, k_2 \ldots k_i}(t) \qquad [78]$$

e, para um estado genérico, o sistema transitando de um estado anterior, também genérico, obtém-se um conjunto de equações diferenciais, uma para cada estado, na forma geral seguinte:

$$\frac{d}{dt} P_{f_1\ f_2\ \cdots\ f_i}(t) = -\left[\alpha^{f_1+1,\ f_2\ \cdots\ f_i}_{f_1\ f_2\ \cdots\ f_i} + \ldots + \alpha^{f_1\ f_2\ \cdots\ f_i+1}_{f_1\ f_2\ \cdots\ f_i} + \alpha^{F}_{f_1\ f_2\ \cdots\ f_i}\right] P_{f_1\ f_2\ \cdots\ f_i}(t) +$$

$$+ \alpha^{f_1\ f_2\ \cdots\ f_i}_{f_1-1\ f_2\ \cdots\ f_i} P_{f_1-1\ f_2\ \cdots\ f_i}(t) + \ldots +$$

$$+ \alpha^{f_1\ f_2\ \cdots\ f_i}_{f_1\ f_2\ \cdots\ f_i-1} P_{f_1\ f_2\ \cdots\ f_i-1}(t) \qquad [79]$$

Para o estado inicial, desde que as taxas de transição são consieradas constantes no tempo, a solução é uma exponencial de forma geral seguinte:

$$P_{k_1\ k_2\ldots k_i}(t) = e^{-\left[\alpha^{k_1+1\ k_2\ldots k_i}_{k_1\ k_2\ldots k_i} + \ldots + \alpha^{k_1\ k_2\ldots k_i+1}_{k_1\ k_2\ldots k_i} + \alpha^{F}_{k_1\ k_2\ldots k_i}\right] t} \qquad [80]$$

Para um estado geral, uma expressão geral é mais dificil de se obter, sendo necessário uma notação em que se con-

sidera conjuntos de estados em um mesmo nivel. Designando por v o nivel b para cada estado de modo que a soma dos b elementos com falta seja b, isto é,

$$b = f_1 + f_2 + \ldots + f_i \qquad [81]$$

para um estado genérico. Desse modo, o sistem transita de um estado genérico inicial de nivel

$$b = k_1 + k_2 + \ldots + k_i \qquad [82]$$

até um estado genérico do nivel

$$b = f_1 + f_2 + \ldots + f_i \qquad [83]$$

Desde o estado inicial $(k_1, k_2, \ldots, k_i)$ até o estado $(f_1, f_2, \ldots, f_i)$ existem

$$\begin{pmatrix} (f_1 - k_1) + (f_2 - k_2) + \ldots (f_i - k_i) \\ f_1 - k_1 \end{pmatrix} \qquad [84]$$

caminhos distintos, cada um contendo

$$(f - k) + (f - k) + \ldots + (f + k) + 1 \qquad [85]$$

nós. Assim, a seqüência dada pela seguinte sussessão de niveis

$$v_{k_1+k_2+\ldots+k_i}\ ,\ v_{k_1+k_2+\ldots+k_i+1}\ ,\ \ldots\ ,\ v_{f_1+f_2+\ldots+f_i-1},$$

$$v_{f_1+f_2+\ldots+f_i} \qquad [86]$$

corresponde a um caminho desde o estado inicial até um estado

genérico, passando pelos niveis intermediários.

Com estas considerações, a solução de [79] será dada por uma expressão geral da forma seguinte:

$$P^{f_1 f_2 \ldots f_i}_{k_1 k_2 \ldots k_i}(t) = \sum_{\substack{\text{SEQUÊNCIA} \\ (k_1 k_2 \ldots k_i)\ldots(f_1 f_2 \ldots f_i)}} \alpha^{\nu_{k_1+k_2+\ldots+k_i+1}}_{k_1 K_2 \ldots k_i} \cdot \alpha^{\nu_{k_1+k_2+\ldots k_i+2}}_{\nu_{k_1+k_2+\ldots k_i+1}} \cdot \alpha^{\nu_{f_1+f_2+\ldots f_i}}_{\nu_{f_1+f_2+\ldots f_i-1}} \cdot$$

$$\cdot \sum_{b=k_1+k_2+\ldots+k_i}^{f_1+f_2+\ldots f_i} \frac{e^{-\alpha_{\nu_b} t}}{\prod_{b=k_1+k_2+\ldots+k_i}^{f_1+\ldots f_i} (\alpha_{\nu_c} - \alpha_{\nu_b})} \qquad \text{COM: } c \neq b \qquad [87]$$

Com os valores da probabilidades de ocorrência dos estados, dadas por [87], a confiabilidade do sistema, no intervalo (0,t), isto é, supondo que o sistema opera corretamente neste intervalo tendo k_1, k_2, ..., k_i defeitos em elementos de tipo 1, 2, ..., i, respectivamente, em t = 0, será dada pela soma das probabilidades de que o sistema opera corretamente, isto é:

$$R_{k_1 K_2 \ldots k_i}(t) = \sum_{f_1=k_1}^{s_1+m_1} \sum_{f_2=k_2}^{s_2+m_2} \ldots \sum_{f_i=k_i}^{s_i+m_i} P^{f_1 f_2 \ldots f_i}_{k_1 k_2 \ldots k_i}(t) \qquad [88]$$

O tempo médio entre falhas (MTBF) pode, em sua forma geral, ser obtido de:

$$MTBF = \int_0^\infty t\, f(t)\, dt \qquad [89]$$

ou então, da transformada de Laplace R(s) da função R(t):

$$MTBF = \lim_{s \to 0} R(s) \qquad [90]$$

Para cada tipo particular de sistema, pode-se estabelecer um modelo que represente os estados possíveis que podem envolver faltas de fabricação e faltas de operação. A partir de um grafo de Markov, as equações diferenciais das probabilidades de que o sistema está naquele estado, podem ser desenvolvidas e, com a sua solução, a confiabilidade e o tempo médio entre faltas podem, finalmente, ser obtidos.

A confiabilidade e o tempo médio entre falhas são, em geral, funções da quantidade de redundância adicionada ao sistema, bem como do tipo de redundância, de reserva quente ou sobressalente frio. O conhecimento dos parâmetros confiabilidade e tempo médio entre faltas permite realizar a avaliação de viabilidade, de custos, etc., para estudos de otimização dos recursos adicionados.

CAPITULO VI

APLICAÇÕES E ESTUDO DE CASOS.

Neste capítulo é apresentada, primeiramente, uma interpretação das equações do modelo geral para facilitar sua aplicação a casos específicos, tendo em vista a complexidade da notação adotada e a generalidade das soluções. Em seguida, é realizada a análise de dois casos, sendo o primeiro para um subsistema com uma redundância com unidades idênticas, isto é, apresentando os mesmos parâmetros, e a redundância é paralela. O comportamento é avaliado por meio dos modelos I e II. No modelo I supõe-se uma redundância paralela sem reparo, e no modelo II admite-se reparo da unidade que falha por falta de fabricação. No segundo caso, desenvolvido no item 6.2.3, é feita uma proposição de uma arquitetura com redundâncias e testadores incluidos para teste, diagnose e tratamento das faltas. O comportamento do sistema é, em seguida, avaliado por meio dos modelos III e IV. No modelo III admite-se redundância paralela e no modelo IV redundância com sobressalentes. Em ambos os casos é admitido o reparo das unidades que falham, em qualquer situação, o que permite a adoção posterior de hipóteses com reparo em certas situações ou mesmo sem reparo, bastando para isso tornar as taxas de reparo específicas iguais a zero.

6.1 - INTERPRETAÇAO DO MODELO GERAL.

As equações para o modelo geral, estabelecidas no capítulo V, podem ser aplicadas a problemas particulares. A notaçao geral é bastante complexa, sendo conveniente adaptá-la a casos reais para um número reduzido de elementos componentes. Supondo que o sistema é constituido por dois tipos de elementos e que as seguintes hipóteses são adotadas:

1. cada tipo de elemento tem uma redundância idêntica, admitindo uma falha;

2. se um elemento de cada tipo falha, o sistema falha;

3. se ambos os elementos de um tipo falham, o sistema falha.

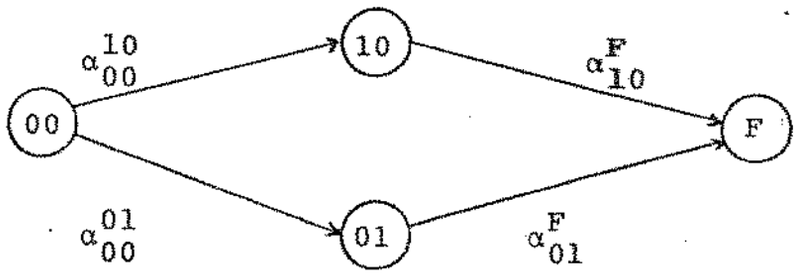


Figura 1 - Grafo de Markov para Dois Tipos de Subsistemas.

O grafo de Markov de estados será o mostrado na figura 1, na qual pode-se identificar:

ESTADOS INICIAIS POSSÍVEIS

EM t = 0 ——— 00 NÃO HÁ FALHAS NA FEBRICAÇÃO

ESTADOS TERMINAIS

10 HÁ FALHA DE FABRICAÇÃO EM ELEMENTO 1

01 HÁ FALHA DE FABRICAÇÃO EM ELEMENTO 2

ESTADOS COM FALHA DO SISTEMA

F HÁ 1 FALHA DE OPERAÇÃO EM UM ELEMENTO CUJO TIPO JÁ TEM 1 FALHA DE FABRICAÇÃO, OU
HÁ 2 FALHAS DE ELEMENTO TIPO 1 NA OPERAÇÃO; OU
HÁ 2 FALHAS DE ELEMENTO TIPO 2 NA OPERAÇÃO

A partir do grafo da figura 1 e das soluções do modelo geral, as seguintes equações diferenciais e respectivas soluções podem ser obtidas:

$$\frac{d}{dt} P_{00}(t) = - \left[\alpha_{00}^{10} + \alpha_{00}^{01}\right] P_{00}(t) \qquad [91]$$

SOLUÇÃO:

$$P_{00}(t) = e^{-\left[\alpha_{00}^{10} + \alpha_{00}^{01}\right]t} = \Pr\{\text{SISTEMA OPERA/HÁ FALHAS DE FABRICAÇÃO}\}$$

$$\left.\begin{aligned} \frac{d}{dt} P_{10}(t) &= - \left[\alpha_{10}^{F}\right] P_{10}(t) + \left[\alpha_{00}^{10}\right] P_{00}(t) \\ \frac{d}{dt} P_{01}(t) &= - \left[\alpha_{01}^{F}\right] P_{01}(t) + \left[\alpha_{00}^{01}\right] P_{00}(t) \end{aligned}\right\} \quad \begin{array}{l} \text{NÍVEL } v=1+0=0+1=1 \\ b=0 \text{ a } 1 \\ c=0 \text{ a } 1 \neq b \end{array} \qquad [92]$$

SOLUÇÃO:

$$P_{10}(t) = \alpha_{00}^{10}\left[\frac{e^{-\alpha_{10}^{F}}}{\left[\alpha_{00}^{10}+\alpha_{00}^{01}\right]-\alpha_{10}^{F}} + \frac{e^{-\left[\alpha_{00}^{10}+\alpha_{00}^{01}\right]t}}{\alpha_{10}^{F}-\left[\alpha_{00}^{10}+\alpha_{00}^{01}\right]}\right]$$

$$P_{01}(t) = \alpha_{00}^{01}\left[\frac{e^{-\alpha_{01}^{F}}}{\left[\alpha_{00}^{10}+\alpha_{00}^{01}\right]-\alpha_{01}^{F}} + \frac{e^{-\left[\alpha_{00}^{10}+\alpha_{00}^{01}\right]t}}{\alpha_{01}^{F}-\left[\alpha_{00}^{10}+\alpha_{00}^{01}\right]}\right] \quad [93]$$

Com estes resultados, pode-se obter a probabilidade de que o sistema opera com uma falha de fabricação:

$$P_{10}(t) + P_{01}(t) = Pr\{\text{SISTEMA OPERA/HÁ 1 FALHA DE FABRICAÇÃO}\} \quad [94]$$

A confiabilidade será a soma das probabilidades dos estados em que o sistema opera, ou seja:

$$R(t) = P_{00}(t) + P_{10}(t) + P_{01}(t) \quad [95]$$

e da qual pode-se obter o tempo medio entre falhas:

$$MTTF = \lim_{s\to 0} R(s) \quad [96]$$

6.2 ESTUDO DE CASOS.

Neste ítem serão desenvolvidos modelos para sistemas em que se deseja evidenciar a utilização de subsistemas redundantes, para tolerância a faltas de fabricação, com e sem reparo. O modelo I é um modelo contendo dois tipos de unidades que podem falhar por faltas de fabricação e por faltas de operação; o modelo II corresponde ao mesmo sistema porém admite-se reparo de qualquer das unidades. Em ambos os casos, entretanto, a redundância é paralela, isto é, reservas quentes.

6.2.1. MODELO I: SUBSISTEMA COM REDUNDANCIA PARALELA - UNIDADES IDENTICAS .

O sistema está representado por seu grafo de Markov, conforme figura 2. Definem-se os seguintes estados:

00 - ambas unidades sem falha,

10 ou 01 - uma unidade com falha, uma unidade sem falha,

11 - duas unidades com falha,

F - uma ou duas unidades com falha de operação, o sistema não recupera.

Todas as hipótesespreviamente estabelecidas são adotadas supondo-se, ainda, que λ é a taxa de falhas de fabri-

cação, λ_1 e λ_2 são taxas de falha por faltas de operação, não havendo reparo em nenhuma das unidades.

Evidencia-se, no grafo, o estado F do sistema, com falha por faltas de operação, não ocorrendo a recuperação d sistema.

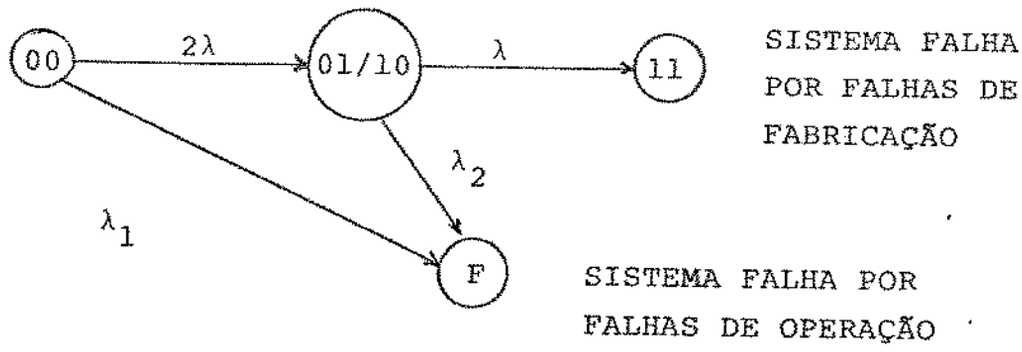


Figura 2 - Grafo de Markov para o Modelo I.

As equações diferenciais do modelo são as seguintes:

$$\frac{d}{dt} P_{00}(t) = -(2\lambda + \lambda_1)\, P_{00}(t) \qquad [97]$$

$$\frac{d}{dt} P_{01/10}(t) = -(\lambda + \lambda_2)\, P_{01/10}(t) + 2\lambda P_{00}(t) \qquad [98]$$

$$\frac{d}{dt} P_{11}(t) = \lambda\, P_{01/10}(t) \qquad [99]$$

$$\frac{d}{dt} P_F(t) = \lambda_1 P_{00}(t) + \lambda P_{01/10}(t) \qquad [100]$$

cujas soluções são:

$$P_{00}(t) = e^{-\alpha_1 t} \qquad [101]$$

$$P_{01/10}(t) = 2\lambda \left[\frac{e^{-\alpha_1 t}}{\alpha_2 - \alpha_1} + \frac{e^{-\alpha_2 t}}{\alpha_1 - \alpha_2} \right] =$$

$$= \frac{2\lambda^2}{\alpha_1 \alpha_2} + 2\lambda^2 \left[\frac{e^{-\alpha_1 t}}{(\alpha_1 - \alpha_2)\ \alpha_1} + \frac{e^{-\alpha_2 t}}{(\alpha_2 - \alpha_1)\ \alpha_2} \right] \qquad [102]$$

$$P_F(t) = \frac{2\lambda\lambda_2 + \lambda_2 \lambda_1}{\alpha_1 \alpha_2} - \frac{2\lambda\lambda_2 + \lambda_1(\alpha_2 - \alpha_1)\ e^{-\alpha_1 t}}{\alpha_1(\alpha_2 - \alpha_1)} -$$

$$- \frac{2\lambda\lambda_2\ e^{-\alpha_2 t}}{\alpha_1 \alpha_2} \qquad [103]$$

sendo, nestas equações:

$$\alpha_1 = 2\lambda + \lambda_1$$

$$\alpha_2 = \lambda + \lambda_2 \qquad [104]$$

As transformadas de Laplace dos estados 00 e 01/10 são as seguintes:

$$P_{00}(s) = \frac{1}{s + \alpha_1} \qquad [105]$$

$$P_{01/10}(t) = \frac{2\lambda}{(s + \alpha_1)(s + \alpha_2)} \qquad [106]$$

Em termos de transformada de Laplace, a confiabilidade pode ser escrita como:

$$R(s) = P_{00}(t) + P_{01/10}(t) = \frac{s + \alpha_2 + 2\lambda}{(s + \alpha_1)(s + \alpha_2)} \qquad [107]$$

da qual se obtém, como função do tempo:

$$R(t) = \left[1 + \frac{2\lambda}{\alpha_2 - \alpha_1} \right] e^{-\alpha_1 t} + \frac{2\lambda\, e^{-\alpha_2 t}}{\alpha_1 - \alpha_2} \qquad [108]$$

e ainda, a partir de [107], obtém-se o tempo medio entre falhas:

$$MTTF = \lim_{s \to 0} R(s) = \frac{3\lambda + 2\lambda}{(2\lambda + \lambda_1)(\lambda + \lambda_2)} \qquad [109]$$

Os resultados para alguns valores particulares das taxas de falha estão mostrados na figura 4, para efeito de comparação com o modelo II, desenvolvido a seguir.

6.2.2 - MODELO II: SUBSISTEMA COM REDUNDANCIA EM PARALELO PARA DUAS UNIDADES IDENTICAS , HAVENDO REPARO NO CASO DE FALHA DE FABRICAÇAO .

As mesmas hipótese adotadas para o modelo I são também aqui adotadas. Os mesmos estados são definidos, e da mesma forma. Admite-se, no entanto, que havendo falha de uma unidade por falta de fabricação, a unidade será reparada. Não se inclue o reparo por outras faltas para que não se confunda os efeitos de

outros reparos com o da hipótese.

Isto significa que o sistema pode iniciar sua operação no estado 00 (não há faltas) e pode falhar por falta de operação transitando para o estado F. Ou então, pode estar no estado 01/10 (houve uma falta de fabricação) e ser reparado, transitando para o estado 00, ou pode falhar por falta de fabricação transitando para o estado 11, ou ainda, pode iniciar sua operação neste estado 01/10 e falhar por falta de operação transitando para o estado F. O estado 11 pode ser considerado como estado com falha do sistema ou não.

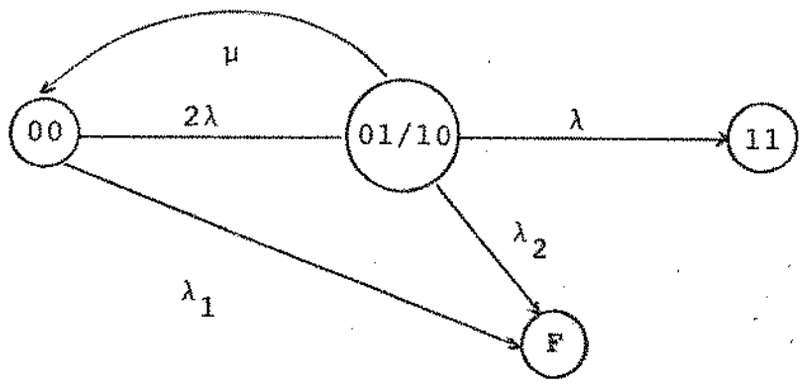


Figura 3 - Grafo de Markov para o Modelo II.

As equações do modelo II serão as seguintes:

$$\frac{d}{dt} P_{00}(t) = - (2\lambda + \lambda_1) P_{00}(t) + \mu P_{01/10}(t) \qquad [110]$$

$$\frac{d}{dt} P_{01/10}(t) = - (\lambda + \lambda_2 + \mu) P_{01/10}(t) + 2\lambda P_{00}(t) \qquad [111]$$

$$\frac{d}{dt} P_{11}(t) = \lambda P_{01/10}(t) \qquad [112]$$

$$\frac{d}{dt} P_F(t) = \lambda_1 P_{00}(t) + \lambda_2 P_{01/10}(t) \qquad [113]$$

das quais se pode obter as seguintes soluções:

$$P_{00}(t) = B_1 e^{-\alpha_3 t} + B_2 e^{-\alpha_4 t}$$

$$P_{01/10}(t) = B_3 (e^{-\alpha_3 t} + e^{-\alpha_4 t})$$

$$P_{11}(t) = B_4 + B_5 e^{-\alpha_3 t} + B_6 e^{-\alpha_4 t}$$

$$P_F(t) = B_7 + B_8 e^{-\alpha_3 t} + B_9 e^{-\alpha_4 t} \qquad [114]$$

onde os diversos coeficientes tem as seguintes expressões:

$$B_1 = \frac{\alpha_3 - b_2}{\alpha_3 - \alpha_4}$$

$$B_2 = \frac{b_2 - \alpha_4}{\alpha_3 - \alpha_4}$$

$$B_3 = \frac{2\lambda}{\alpha_4 - \alpha_3}$$

$$B_4 = \frac{2\lambda^2}{\alpha_3 - \alpha_4}$$

$$B_5 = \frac{2\lambda^2}{\alpha_3(\alpha_3 - \alpha_4)}$$

$$B_6 = \frac{2\lambda^2}{\alpha_4(\alpha_4 - \alpha_3)}$$

$$B_7 = \frac{B_{10}}{\alpha_3 \alpha_4}$$

$$B_8 = \frac{B_{10} + \alpha_3\lambda_1}{\alpha_3(\alpha_3 - \alpha_4)}$$

$$B_9 = \frac{\alpha_4\lambda_1 - B_{10}}{\alpha_4(\alpha_3 - \alpha_4)}$$

[115]

$$B_{10} = 2\lambda\lambda_2 + b_2\lambda_1$$

sendo, ainda:

$$b_1 = 2\lambda + \lambda_1$$

$$b_2 = \lambda + \lambda_2 + \mu$$

$$b_3 = 3\lambda + \lambda_1 + \lambda_2 + \mu$$

$$b_4 = 2\lambda^2 + 2\lambda\lambda_2 + \lambda\lambda_1 + \lambda_1\lambda_2 + \mu\lambda_1$$

$$\alpha_3 = \frac{1}{2}\left[b_3 + \sqrt{b_3^2 - 4b_4}\right]$$

$$\alpha_4 = \frac{1}{2}\left[b_3 + \sqrt{b_3^2 - 4b_4} \right] \qquad [116]$$

A confiabilidade do modelo II será dada por:

$$R(t) = P_{00}(t) + P_{01/10}(t) = (B_1 + B_3)e^{-\alpha_3 t} + (B_2 + B_3)e^{-\alpha_4 t} \qquad [117]$$

ou então, em termos de transformada de Laplace:

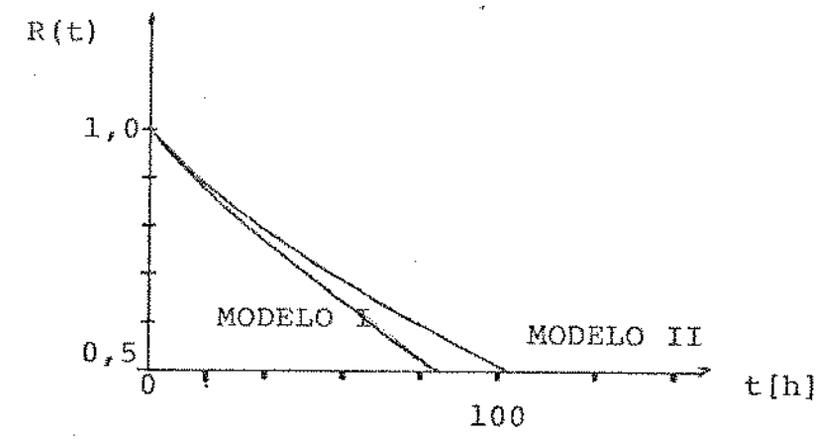
$$R(s) = \frac{s + 2\lambda + b_2}{(s + \alpha_3)(s + \alpha_4)} \qquad [118]$$

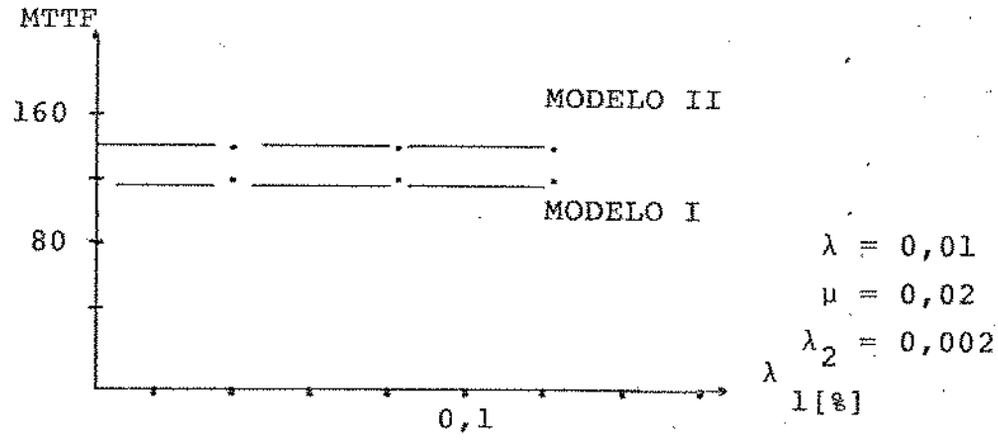
da qual se pode obter o tempo medio entre falhas:

$$MTTF = \lim R(s) = \frac{3\lambda + \lambda_2 + \mu}{2\lambda^2 + 2\lambda\lambda_2 + \lambda\lambda_1 + \lambda_1\lambda_2 + \mu\lambda\lambda_1} \qquad [119]$$

Os resultados das equações acima, para valores particulares das taxas de falhas estão mostrados na figura 4, comparando-se os modelos I e II. Esses valores são apenas para fins de análise comparativa, entretanto, são considerados típicos em circuitos eletrônicos integrados [8].

Na comparação, vê-se que a possiblidade de reparo de uma unidade que falha por faltas de fabricação produz sensivel benefício sobre o tempo medio entre falhas (e na confiabilidade do sistema). Esse benefício é mais evidente no caso de aumento da taxa de falhas de fabricação λ_2, que representa as falhas de fabricação que ocorrem após outra falta de fabricação.





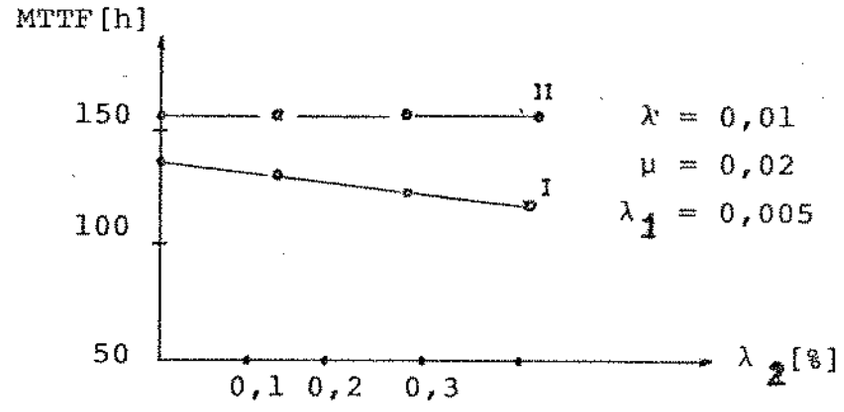


Figura 4

EFEITOS SOBRE A CONFIABILIDADE E SOBRE O TEMPO MÉDIO ENTRE FALHAS, DAS FALHAS DE FABRICAÇÃO E REPARO, PARA SUBSISTEMA COM UMA REDUNDÂNCIA.

6.2.3 - PROPOSIÇÃO DE ARQUITETURA COM MODULOS DE TESTE E RECUPERAÇÃO .

A confiabilidade, o tempo medio entre falhas e, em conseqüência, a manutenção de sistemas de controle por computador em aplicações civis e militares, em particular no controle da manufatura, constituem um problema atual e permanente. O preço pago pelas falhas do equipamento aparece como redução da performance operacional dos meios de produção e custos de manutenção. Muitas vezes, esses custos ultrapassam, e muito, os custos do próprio sistema eletrónico de controle. A diminuição desses custos pode ser alcançada aumentado-se a testabilidade ou diagnosabilidade do sistema, bem como a mantenabilidade e disponibilidade dos equipamentos, sendo a autoreparação um atributo consideravelmente desejavel. Os computadores, de um modo geral e por sua própria natureza, oferecem meios adequados (eletrónicos e de processamento) para isso [5, 13, 28, 37, 61, 80, 103, 108, 111, 122, 159].

Computadores podem ser usados, neste apecto, para testar subsistemas e realizar a diagnose de faltas, desligar subsistemas com falta e ligar subsistemas sobressalentes, modificar a configuração do hardware ou o software, para continuar operacional, mesmo com peformance prejudicada ou degradada [23, 36]. Pode ser encarado como um perito em manutenção e automanutenção, principalmente se programas peritos ou de inteligência artificial forem incluidos [93, 94, 104, 163]. Extensos programas podem hoje ser armazenados em discos compactos com leitura

a lazer e, naturalmente, tecnologia VLSI ou WSI deverá ser usada [55, 57, 69].

A proposição feita aqui, define e caracteriza alguns módulos essenciais, um tanto diferentes dos microprocessadores, mas que dispõe de características de tolerância a faltas. O módulo básico, mostrado na figura 5, pode ser multiplicado conforme mostra a figura 10, em redundância dinâmica, por exemplo.

Os atibutos principais do módulo básico são:

1. o módulo pode ser usado isolado ou multiplicado, oferecendo grande variedade de arquiteturas para um computador de controle,

2. a recuperação automática da unidade pode ser alcançada desde que autotestes são realizados por unidades de teste,

3. além dos barramentos externos, há tres blocos básicos que podem ser integrados em VLSI ou WSI ou mesmo de forma independente.

Os blocos básicos consistem das seguintes subunidades:

1. memória M com redundância MR e interface com deteção e correção de erro IM,

2. interface de barramento IB e controlador de barramento CB,

3. processador P e $\bar{P}$, comparador C e testador de barramento TB.

Em cada bloco, serão incluidos circuitos de teste e deteção de faltas de seus próprios circuitos e de sinalização

para indicação externa. A estrutura básica dos blocos está mostrada sumariamente nas figuras 6, 7, 8 e 9, onde são mostradas as subunidades do comparador e do testador de barramento.

Um sistema como o configurado na figura 10, oferece redundância adicional. Diversas hipóteses de ocorrência de faltas de fabricação e de operação podem ser feitas, bem como as possibilidades de reparo. Destaca-se dois casos particulares, analisados como modelo III e modelo IV, que seguem.

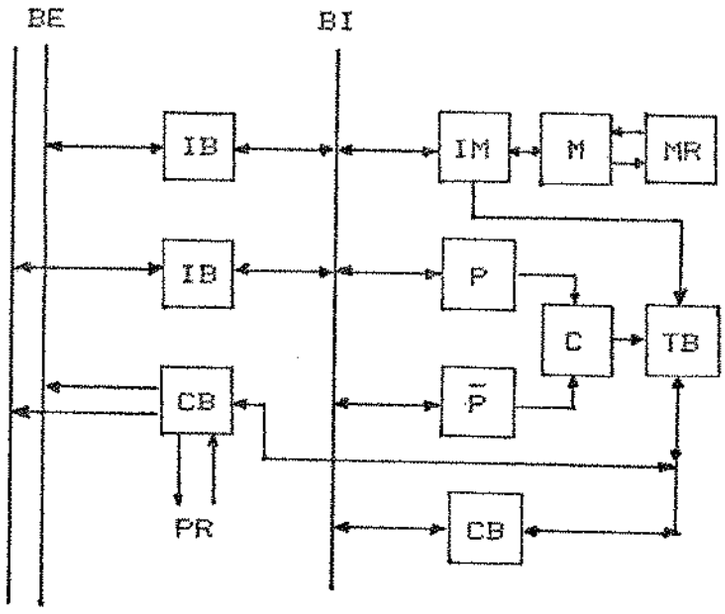


Figura 5 - Arquitetura do Módulo Básico.

BE - Barramento Externo

BI - Barramento Interno

IB - Interface de Barramento

CB - Controlador de Barramento - Autotestavel

IM - Interface de Memória

M - Memória

MR - Memória Redundante

P - Processador

P - Processador de Lógica Complementar

C - Comparador

TB - Testador de Barramento - Autotestavel

PR - Bits de Prioridade

As arquiteturas dos subsistemas de teste e recuperação pode ser vista nas figuras seguintes.

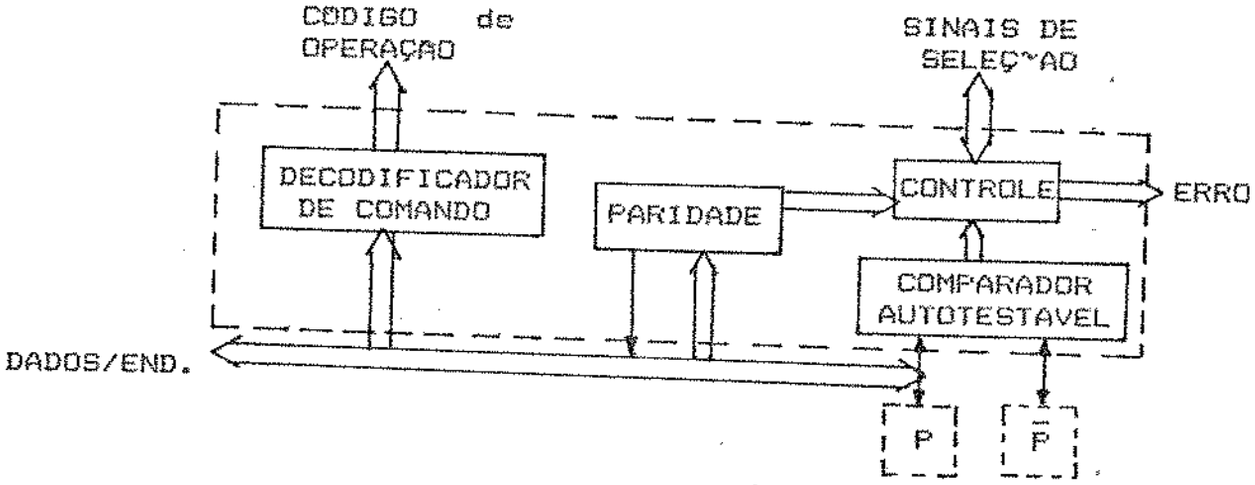


Figura 6 - Módulo do Comparador.

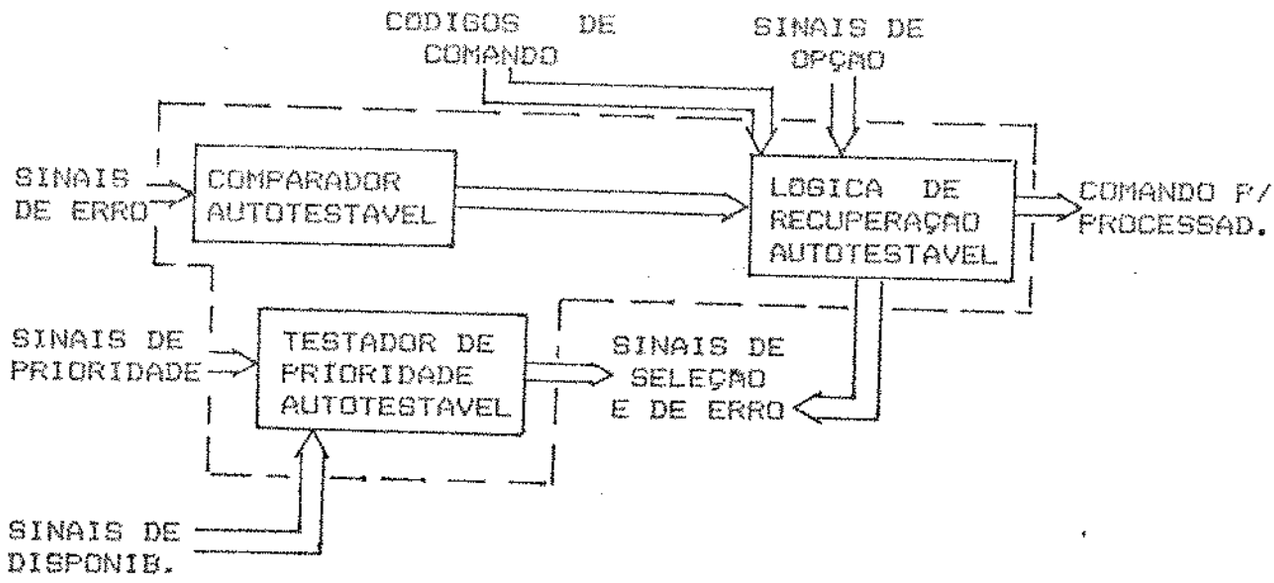


Figura 7 - Módulo do Testador de Barramento.

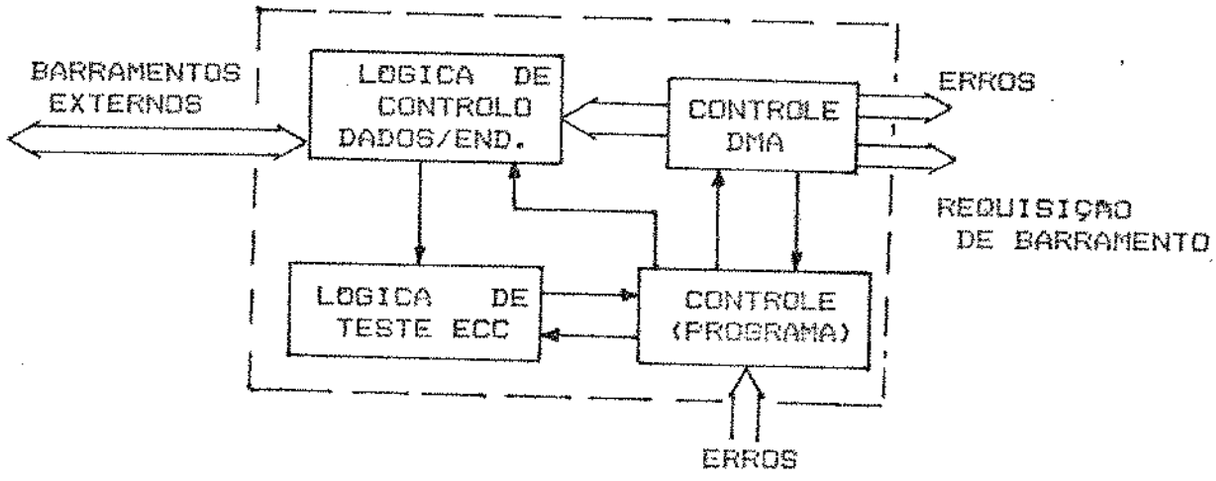


Figura 8 - Módulo da Interface de Barramento.

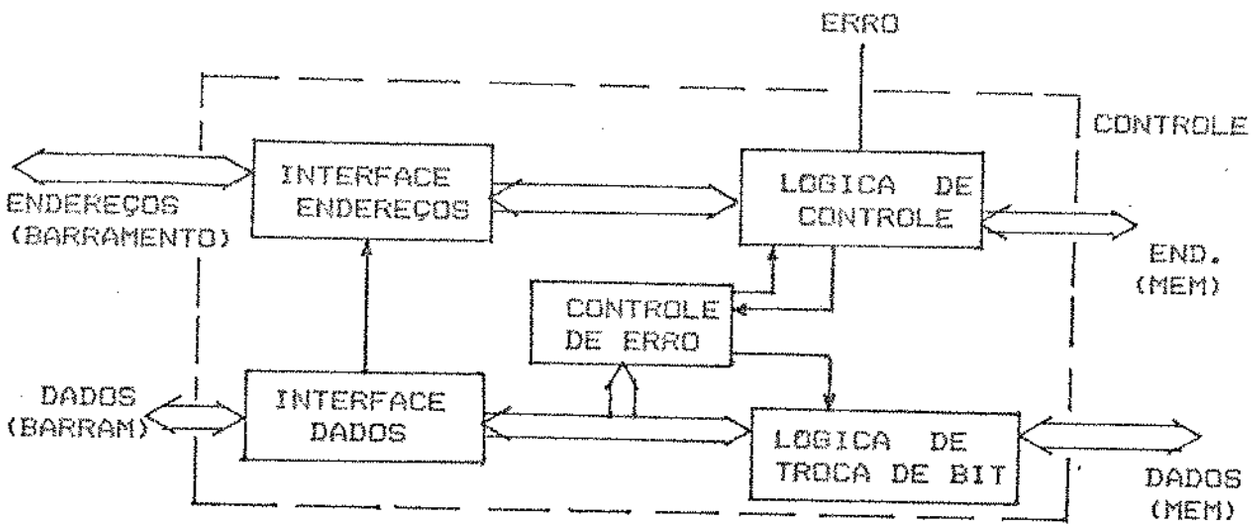


Figura 9 - Interface de Memória.

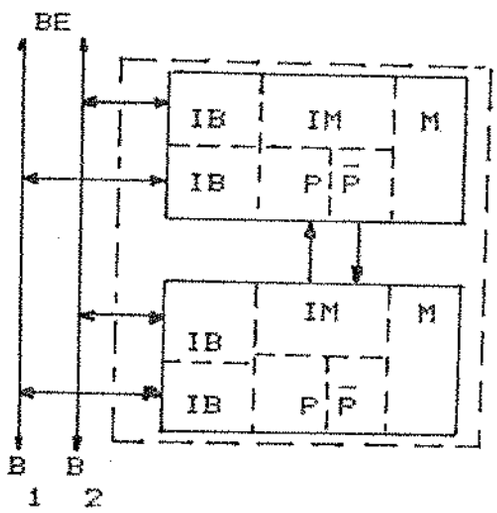


Figura 10 - Arquitetura de Computador com Duas Unidades de Módulos Básicos.

6.2.3.1 - MODELO III: REDUNDANCIA PARALELA.

O modelo III, assim como o modelo VI que se segue, são desenvolvidos para a avaliação do sistema apresentado acima, figura 10, que consta de dois módulos básicos e dois barramentos. As seguintes hipóteses serão consideradas:

A: o sistema opera se:

1. os dois módulos operam e os dois barramentos operam,

2. os dois módulos operam e um barramento opera.

B: as falhas podem ocorrer durante a fabricação ou durante a operação do sistema:

1. podem ocorrer $s_1 = 0$ falhas de fabricação nos módulos; podem ocorrer $s_2 = 1$ falhas de fabricação nos barramentos,

2. podem ocorrer $m_1 = 0$ falhas de operação nos módulos, podem ocorrer $m_2 = 1$ falhas de operação nos barramentos.

C: o reparo pode ser realizado a partir de qualquer estado ao qual o sistema chegou por falha na fabricação ou por falha na operação.

Com essas considerações, os seguintes estados podem ser definidos:

00 - sem falhas - pode ser o estado inicial de operação,

0i - uma falha de operação (barramento)

10 - uma falha de fabricação (barramento)

11 - uma falha de fabricação e uma falha de operação (barra-

mento,

02 - duas falhas de operação (dois barramentos ou um módulo e um barramento),

20 - duas falhas de fabricação (dois barramentos ou um módulo e um barramento).

Supõe-se que, nos estados 00, 01 e 10 o sistema não falha enquanto que nos estados 02, 11 e 20 o sistema falha.

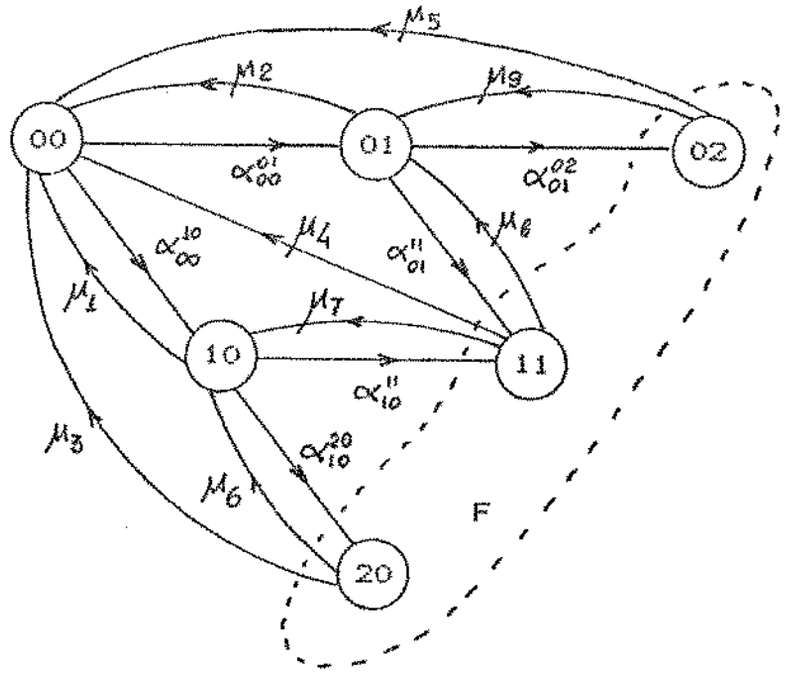


Figura 11 - Grafo de Markov para o Modelo III.

As equações diferenciais associadas à figura 11 serão as seguintes:

$$\frac{d}{dt}P_{00}(t) = -(\alpha_{00}^{01} + \alpha_{00}^{10})P_{00}(t) + \mu_2 P_{01}(t) + \mu_4 P_{11}(t) + \mu_5 P_{02}(t) + \mu_1 P_{10} + \mu_3 P_{20}(t) \quad [120]$$

$$\frac{d}{dt}P_{10}(t) = -(\alpha_{10}^{11} + \alpha_{10}^{20})P_{10}(t) + \alpha_{00}^{10} P_{00}(t) + \mu_6 P_{20}(t) + \mu_7 P_{11}(t) - \mu_1 P_{10}(t) \quad [121]$$

$$\frac{d}{dt} P_{01}(t) = -(\alpha_{01}^{11} + \alpha_{01}^{02}) P_{01}(t) + \alpha_{00}^{01} P_{00}(t) - \mu_2 P_{01}(t) + \mu_8 P_{11}(t) + \mu_9 P_{02}(t) \quad [122]$$

$$\frac{d}{dt} P_{20}(t) = \alpha_{10}^{20} P_{10}(t) - (\mu_3 + \mu_6) P_{20}(t) \quad [123]$$

$$\frac{d}{dt} P_{11}(t) = \alpha_{01}^{11} P_{01}(t) + \alpha_{10}^{11} P_{10}(t) - (\mu_4 + \mu_7 + \mu_8) P_{11}(t) \quad [124]$$

$$\frac{d}{dt} P_{02}(t) = \alpha_{01}^{02} P_{01}(t) - (\mu_5 \quad \mu_9) P_{02}(t) \quad [125]$$

As condições iniciais do sistema são:

$$P_{00}(0) = 1 \qquad P_{01}(0) = P_{10}(0) = P_{11}(0) = P_{20}(0) = P_{02}(0) = 0 \quad [126]$$

Supondo constantes as taxas de transição, correspondendo:

$$\alpha_{00}^{01} = 2\lambda f \qquad \text{e} \qquad \alpha_{10}^{20} = \lambda f \quad [127]$$

às taxas de faltas de fabricação e

$$\alpha_{00}^{01} = 2\lambda_0 \qquad \text{e} \qquad \alpha_{01}^{02} = \alpha_{10}^{11} = \alpha_{01}^{11} = \lambda_0 \quad [128]$$

às taxas de faltas de operação.

Usando-se ainda a relação:

$$\sum_{f_1 f_2} P_{f_1 f_2}(t) = 1 \quad [129]$$

pode-se obter as expressões [130] para o estado estacionário, fazendo todas as derivadas iguais a zero:

$$(\alpha_{00}^{01} + \alpha_{00}^{10})\, P_{00} = \mu_1 P_{10} + \mu_2 P_{01} + \mu_3 P_{20} + \mu_4 P_{11} + \mu_5 P_{02}$$

$$(\alpha_{10}^{11} + \alpha_{10}^{20} + \mu_1)\, P_{10} = \alpha_{00}^{10} P_{00} + \mu_6 P_{20} + \mu_7 P_{11}$$

$$(\alpha_{01}^{11} + \alpha_{01}^{02} + \mu_2)\, P_{01} = \alpha_{00}^{01} P + \mu_8 P_{11} + \mu_9 P_{02}$$

$$(\mu_3 + \mu_6)\, P_{20} = \alpha_{10}^{20} P_{10}$$

$$(\mu_4 + \mu_7 + \mu_8)\, P_{11} = \alpha_{01}^{11} P_{01} + \alpha_{10}^{11} P_{10}$$

$$(\mu_5 + \mu_9)\, P_{02} = \alpha_{01}^{02} P_{01} \qquad [130]$$

Adotando-se as seguintes expressões de coeficientes:

$$A_1 = \frac{\lambda f}{\mu_3 + \mu_6}$$

$$A_2 = \frac{1}{\mu_4 + \mu_7 + \mu_8}$$

$$A_3 = \frac{\lambda_0}{\mu_5 + \mu_9}$$

$$A_4 = 2\lambda 0 + \mu 2 - \lambda 0\ \mu 8\ A_2 - \mu 9\ A_3$$

$$A_5 = \lambda 0 + \lambda f + \mu 1 - \mu 6\ A1 - \lambda 0\ \mu 7\ A_2$$

$$A_6 = \lambda f\ \mu 7\ A_2 \ / \ A_4$$

$$A_7 = (2\lambda f + 2\lambda 0\ A_6\ ) \ / \ (A_5 - \lambda 0\ \mu 8\ A_2\ A_6\ )$$

$$A_8 = (2\lambda 0 + \lambda 0\ \mu 8\ A_2\ A_7\ ) \ / \ A_4 \qquad [131]$$

para poder deixar as expressões em forma mais compacta, pode-se obter as seguintes soluções para as equações [130]:

$$P_{00} = 1 + [A_7\ (1 + A_1 + A_2 + \lambda_0) + A\ \ (1 + A_3 + A_2\ \ \lambda f)]^{-1}$$

$$P_{10} = A_7 \cdot P_{00}$$

$$P_{01} = A_8 \cdot P_{00}$$

$$P_{20} = A_1\ \ A_7 \cdot P_{00}$$

$$P_{11} = A_2\ (\lambda_0\ A_7 + \lambda_0\ \ A_8)\ P_{00}$$

$$P_{02} = A_3\ A_8 \cdot P_{00} \qquad [132]$$

Pode-se definir, com relação às probabilidades acima, a disponibilidade de estado estacionário:

$$A_{ss} = P_{00} + P_{01} + P_{10} \qquad [133]$$

e a probabilidade de que ocorreu, pelo menos uma falta de fabricação a soma:

$$P_f = P_{10} + P_{20} \qquad [134]$$

e, de modo similar, a probabilidade de que ocorreu, pelo menos uma falta de operacão a soma:

$$P_0 = P_{01} + P_{02} + P_{11} \qquad [135]$$

O tempo medio entre falhas, supondo que há reparo, será dada por:

$$MTTF_R = \lim_{s \to \infty} R(s) \qquad [136]$$

onde R(s) é a transformada de Laplace de R(t), confiabilidade do sistema. Fazendo-se:

$$\mu_j = 0 \qquad \text{para } j = 3,4...9 \qquad [137]$$

nas equações diferenciais [120] a [125], tomando as transformadas de Laplace das equações e resolvendo somente para explicitar P_{00}, P_{01} e P_{10} pois, neste caso, a confiabilidade é dada pela soma dessas tres probabilidades (os estados 02, 11 e 20 são estados de falha do sistema), obtém-se:

$$MTTF_R = \lim_{s \to \infty} [P_{00}(s) + P_{01}(s) + P_{02}(s)] = \frac{B_1}{B_2} \qquad [138]$$

onde

$$B_1 = (\lambda_0 + \lambda f + \mu_1)\ (4\lambda_0 + \mu_2) + 2\lambda f(2\lambda_0 + \mu_2)$$

$$B_2 = (\lambda_0 + \lambda_f + \mu_1)[(2\lambda_0 + 2\lambda_f)(2\lambda_0 + \mu_2) - 2\lambda_0\mu_2] - 2\lambda_f\mu_1(2\lambda_0 + \mu_2) \qquad [139]$$

Fazendo-se $u_1 = u_2 = 0$, na expressão [138], pode-se obter o tempo medio entre falhas sem reparo:

$$MTTF = \frac{(\lambda_0 + \lambda_f)4\lambda_0 + 2\lambda_f 2\lambda_0}{(\lambda_0 + \lambda_f)[(2\lambda_0 + 2\lambda_f)2\lambda_0} = \frac{\lambda_0 + 2\lambda_f}{(\lambda_0\ \ \lambda_f)^2} \qquad [140]$$

Na figura 12 mostra-se os resultados que são obtidos para valores particulares das taxas de falha de fabricação e de operação, no gráfico (a) com as taxas de reparo especificadas e no gráfico (b) sem reparo.

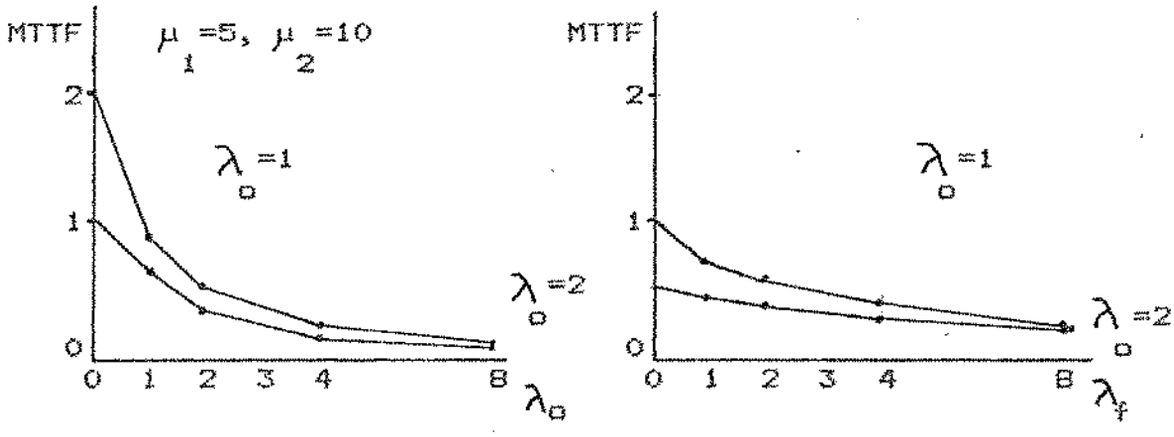


Figura 12 - MTTF para o Modelo III.

6.2.3.2 - MODELO IV: REDUNDANCIA COM SOBRESSALENTES.

O modelo matemático seguinte representa o mesmo sistema do ítem anterior, de dois módulos básicos e dois barramentos sendo, porém, as unidades redundantes constituidas por sobressalentes frios. As mesmas hipóteses gerais são admitidas para fins de comparação de resultados com o modelo III. O mesmo grafo de Markov de transições de estados pode ser usado (figura 11), porém as transições serão distintas e dadas por:

$$\alpha_{00}^{10} = \alpha_{10}^{20} = \lambda_f$$

$$\alpha_{00}^{01} = \alpha_{01}^{02} = \alpha_{10}^{11} = \alpha_{01}^{11} = \lambda_0 \qquad [141]$$

As equações diferenciais associadas ao modelo IV serão análogas às do modelo III. Também as condições iniciais serão as mesmas, de modo que conduzem às seguintes relações para o estado estacionário:

$$(\lambda_0 + \lambda_f)P_{00} = \mu_1 P_{10} + \mu_2 P_{01} + \mu_3 P_{20} + \mu_4 P_{11} + \mu_5 P_{02}$$

$$(\lambda_0 + \lambda_f + \mu_1)P_{10} = \lambda_f P_{00} + \mu_6 P_{20} + P_{11}$$

$$(\lambda_0 + \lambda_0 + \mu_2)P_{01} = \lambda_0 P_{00} + \mu_8 P_{11} + \mu_9 P_{02}$$

$$(\mu_3 + \mu_6)P_{20} = \lambda_f P_{10}$$

$$(\mu_4 + \mu_7 + \mu_8)P_{11} = \lambda_0 P_{01} + \lambda_0 P_{10}$$

$$(\mu_5 + \mu_9)P_{02} = \lambda_0 P_{01} \qquad [142]$$

As soluções dessas equações serão as seguintes:

$$P_{00} = [1 + A_7(1 + A_1 + A_2 + \lambda_0) + A_8(1 + A_3 + A_2\lambda_f)]^{-1}$$

$$P_{10} = A_7 P_{00}$$

$$P_{01} = A_8 P_{00}$$

$$P_{20} = A_1 A_7 P_{00}$$

$$P_{11} = A_2(\lambda_0 A_7 + \lambda_0 A_8)P_{00}$$

$$P_{02} = A_3 A_8 P_{00} \qquad [143]$$

onde os coeficientes A_1 até A_6 tem as mesmas expressões que as dadas em [131], e ainda tem-se:

$$A_7 = (\lambda_f + \lambda_0 A_5) / (A_5 - \lambda_0 \mu_8 A_2 A_6)$$

$$A_8 = \lambda_0 (1 + \mu_8 A_2 A_7) / A_4 \qquad [144]$$

A disponibilidade de estado estacionário será dada pela expressão:

$$A_{ss} = P_{00} + P_{01} + P_{10} \qquad [145]$$

Fazendo-se, como no modelo III, $u_j = 0$ para j=3 até 9, nas equações [142] e nas relações dadas por [144], tomando-se a transformada de Laplace e resolvendo para os estados 00, 01 e 10, pode-se obter a seguinte expressão para tempo medio entre falhas:

$$MTTF_R = \lim_{s \to \infty} R(s) = [P_{00}(s) + P_{01}(s) + P_{10}(s)] = \frac{C_1}{C_2} \qquad [146]$$

onde:

$$C_1 = (\lambda_0 + \lambda_f + \mu_1)(4\lambda_0 + \mu_2) + \lambda_f(2\lambda_0 + \mu_2)$$

$$C_2 = (\lambda_0 + \lambda_f + \mu_1)[(\lambda_0 + \lambda_f)(2\lambda_0 + \mu_2) - \lambda_0\mu_2] - \lambda_f\mu_1[2\lambda_0 + \mu_2] \qquad [147]$$

Fazendo-se, $u_1 = u_2 = 0$, na expressão [146], obtém-se o tempo medio entre falhas sem reparo:

$$MTTF = \frac{2\lambda_0 + 3\lambda_f}{(\lambda_0 + \lambda_f)^2} \qquad [148]$$

Os resultados estão representados no gráfico da figura 13, para os mesmos valores utilizados no traçado dos grá-

ficos da figura 12 referentes ao modelo III.

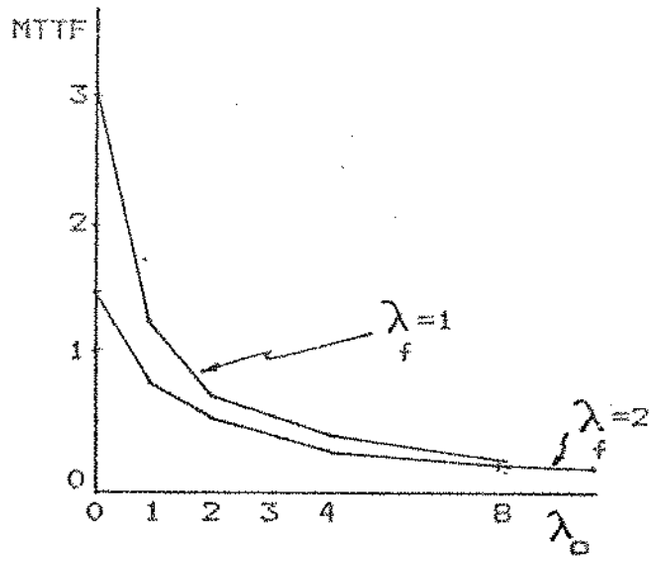


Figura 13 - MTTF para o Modelo IV.

6.2.4 - DISCUSSÃO DOS RESULTADOS.

Os resultados obtidos a partir dos modelos I e II demonstram que a utilização de reparo para falhas de fabricação conduz a um benefício sobre o tempo medio entre falhas. O benefício é maior na medida em que cresce a taxa de falhas de operação do sistema relativa à transição de estado com uma falha de fabricação prévia. O benefício é muito menos aparente se não há falhas de fabricação. Os gráficos da figura 4 mostram que o modelo com reparo produzirá um resultado global melhor, demonstrado pela confiabilidade.

Por outro lado, os resultados obtidos por meio dos modelo III e VI, mostram que, tal como já previsto no item 5.1.3 referente a redundâncias paralela e com sobressalente frio, há uma sensível diferença a favor do moedlo IV de sobressalentes frios que apresenta um tempo medio entre falhas cerca de 50% maior que o do modelo de reserva paralela. Os méritos deste modelo podem ser reduzidos em virtude do fator suficiência. A comutação do sobressalente frio deve ser realizada por meio de circuitos de elevada confiabilidade, possivelmente autotestáveis e auto reparáveis, para que as vantagens alcançadas não fiquem prejudicadas. De qualquer modo, como este parâmetro nunca será exatamente igual a 1, as curvas da figura 2 do item 5.1.3 mostram que o número de sobressalentes não precisará ser superior a um ou

dois, para se obter os maiores benefícios.

Essas conclusões parciais justificam a arquitetura proposta no item 6.2.3, na qual o número de reservas ou sobressalentes é, tipicamente, dois. Estas considerações devem fazer parte integrante de um projeto de sistema de computador para controle de processos industriais, conforme proposto inicialmente, tanto para a integração VLSI de circuitos eletrônicos como para a integração de sistemas de manufatura por meio de computadores onde essas arquiteturas tolerantes a faltas sem dúvida estarão presentes.

CAPITULO VII

CONCLUSÕES .

As conclusões seguintes podem ser formuladas como conseqüência dos estudos realizados e considerando os objetivos da tese, conforme listados no capítulo I.

1.- O modelo geral de Markov para análise e avaliação de arquiteturas de computadores, em que se considera a inclusão de meios para tolerância a faltas, foi estabelecido. Neste modelo geral, é considerada a possibilidade de ocorrência de faltas de fabricação e faltas de operação em subsistemas. Este modelo na forma de grafo, pode ser estabelecido facilmente e no qual podem ser formuladas hipóteses para faltas e para estados entre os quais o sistema transitará por efeito dessas faltas. Reparo pode também ser incluido nesse modelo. A partir do grafo de Markov, pode-se formular equações diferenciais que representam as probabilidades de que o sistema permanece em dado estado, cuja solução fornece essas probabilidades como funções do tempo.

2.- O modelo geral, desenvolvido no capítulo IV, permite incluir na análise de uma arquitetura complexa, tolerante a faltas, diferentes tipos de elementos, cada tipo com suas

redundâncias paralelas ou de sobressalentes. Por outro lado, o modelo permite, ainda, separar claramente os efeitos de faltas durante a fabricação do sistema analisado, dos efeitos das faltas que ocorrem após sua entrada em serviço. Desse modo, as diferenças estruturais, funcionais e paramétricas entre os componentes e subsistemas podem ser consideradas na avaliação de uma arquitetura.

3.- Além do estado inicial de fabricação (sem faltas), o sistema pode ser representado pelo modelo de modo que diversos estados iniciais (com alguma falta de fabricação) podem ser propostos para estados iniciais de operação e, a partir destes, estados terminais podem ser atingidos (todas as redundâncias de um ou mais tipos de elementos já falharam, exceto uma). Prevê-se que, a partir de qualquer estado, inclusive de estados existentes durante a fabricação, o sistema pode falhar devido a alguma falta que provoque falha na recuperação. Neste último caso, o sistema poderá ser construido mas não irá operar corretamente quando entrar em serviço. O modelo não prevê quando isso ocorrerá mas permite calcular a probabilidade dessa ocorrência. Neste aspécto o modelo é mais pessimista que os modelos usuais, já que considera falhas antes que o sistema inicie sua operação. Por outro lado, entretanto, permite otimizar recursos de modo que se possa prever sua operação apesar de falhas anteriores.

4.- Modelos apresentados na literatura são, geralmente, resolvidos por processos numéricos ou aproximados, tendo

em vista a complexidade do problema, causada pela grande quantidade de estados possíveis [17, 20, 41, 46, 47, 59, 83, 92, 104, 113, 129, 152, 164]. O número de estados é maior quando se considera a existência de diferentes tipos de elementos na estrutura redundante do sistema e que, ainda, podem ocorrer faltas durante a fabricação do próprio sistema, além daquelas comumente analisadas, as faltas de operação. Dessa forma, o modelo analítico apresentado, permite uma análise dos efeitos de técnicas de tolerância a faltas. Permite realizar comparações entre diferentes soluções, examinar diversas topologias, avaliar esquemas que podem resultar em menor custo, em diferentes situações e para diferentes objetivos.

5.- A análise de sistemas e de arquiteturas, do ponto de vista d tolerância a faltas e de confiabilidade, é realizada caso a caso. Na literatura são apresetados problemas específicos e análise de casos específicos sem que, na sua maioria, haja maior preocuação com a generalização de questões de tolerância a faltas naqueles casos, excetuando-se as descrições, às vezes insuficientemente aprofundadas. Métodos heurísticos e numéricos são os mais comuns, existindo muitas análises para otimização de parâmetros. Análises mais gerais, com a aqui realizada com os modelos introduzidos, permite considerar os efeitos de certos eventos como as falhas de fabricação e, além disso, os diferentes tipos de subsistemas com redundâncias, e tratá-los sem restrições especiais, exceto quanto à simultaneidade das faltas.

6.- Um número qualquer de estados com falta de fabricação pode ser formulado por meio do modelo. Cada um desses estados pode ser o estado inicial de operação do sistema. Isto quer dizer que um sistema tolerante a faltas não precisa ser modelado considerando um único estado inicial onde, supostamente, não há faltas. Esta capacidade do modelo permite, assim, introduzir faltas mais realistas em um sistema, podendo-se investigar as conseqüências das faltas antes da entrada em serviço do sistema e, portanto, tomar providências adequadas já que elas não podem ser evitadas de modo absoluto.

7.- Uma conseqüência das conclusões acima é que, ao se projetar um sistema de computador de controle, pode-se incluir em sua arquitetura, características de tolerância a faltas de modo que exista uma probabilidade mais elevada de que, ao entrar em serviço, opere dentro das especificações, mesmo que com certa degradação dos serviços a que se destina.

8.- Uma simplificação que o modelo geral permite fazer é a generalização das condições de falha do sistema. Em um sistema de manufatura, por exemplo, ocorrendo uma falha do computador de controle, não se evidencia importante conhecer de imediato as causas da falha (exceto o que se refere à manutenção do sistema pois, nesta altura, o sistema já falhou, ou seja, já parou de operar, já que se o computador falha não há mais controle). No entanto, e não apenas para efeitos de manutenção, é importante conhecer as causas das faltas que foram toleradas,

isto é, quando o computador não falha como conseqüência dessas faltas.

9.- Durante a construção de sistemas de computação é usual a seleção de componentes realizada através de controle de qualidade. Como fase final, muitas vezes é realizado o processo "burn-in" de teste que, em princípio, deve evidenciar quaisquer componentes ou subsistemas sujeitos a falha. Como sua entrada em serviço é uma fase posterior, esta pode não se efetivar por causa de faltas que ocorrem durante ou após o teste, além de outras faltas latentes. Tolerar esse tipo de faltas constitui um ganho de performance que pode compensar os custos adicionais necessários à implementação dos meios de tolerância a essas faltas.

10.- Existindo muitos possíveis estados iniciais nos modelos desenvolvidos, evidencia-se a importância do reparo de subsistemas em qualquer fase de fabricação pois, com isso, pode-se melhorar a confiabilidade do sistema, mesmo após a fase de fabricação. O reparo durante a fabricação exige, entretanto, a existência de meios adequados de deteção e diagnose de faltas, bem como da implementação de procedimentos adequados de tolerância a essas faltas e à manutenção. Não se tomando providências para a existência desses meios, as faltas de fabricação permanecerão latentes até a falha do sistema, por ocasião do teste final ou de sua entrada em serviço, prejudicando, fortemente, a performance e, talvez, até o produto.

11.- O detalhamento com que será aplicado o modelo proposto depende da profundidade com que se deseje tratar um problema especifico. Nos casos examinados no capitulo V, procurou-se uma forma de abordagem generalisada que permitisse observar e verificar a utilização do modelo, sem que isso desse origem a cálculos demasiado longos e tediosos. Note-se que os casos tratados foram resolvidos considerando-se um equacionamento normal e completo do problema, e não uma simples abordagem numérica. O trabalho de mecanização de uma solução geral como a apresentada foge ao alcance das capacidades de computação e, de qualquer modo, seria necessária uma limitacão dos dados, tendo em conta as limitações dos recursos de computação. Soluções particulares podem ser mecanizadas a partir de algorítmo qu tome por base as equações gerais do modelo que apresentamos, caso se deseje uma abordagem numérica. Uma solução com otimização pode ser formulada, usando um dos inúmeros métodos apresentados na literatura [2, 141].

12.- A incorporação de redundâncias a nivel de componente, de circuito integrado, de "wafer", de subsistema, etc., para se obter tolerância a faltas de fabricação e de operação de um sistema, é uma solução prática, ao se tratar da consruç~ao de sistemas complexos. O investimento necessário à formulação de um modelo especifico aplicável a um caso particular, bem como o investimento necessário para a incorporação de redundâncias e dos meios para a tolerância a faltas desejada, poderão ser compensados pela maior probablidade de que, uma vez construido, o sistema irá operar corretamente, mesmo que com certa degradação

de sua performance. O exame da literatura pertinente mostra essa tendência [13, 33, 36, 44, 55, 65, 71, 80, 101, 111], mas muitas das soluções propostas ainda aguardam maior desenvolvimento da tecnologia de fabricação de circuitos integrados, especialmente os de dimensão "wafer". Os módulos propostos no capítulo VI (6.2.3) também se enquadram neste caso.

13.- Através deste estudo, pode-se notar que, com relação a um sistema com características de tolerância a faltas como, por exemplo, computadores de controle de processos industriais, a consideração da existência de estados com faltas de fabricação, em vez de considerar apenas um único estado inicial sem falhas, permite avaliar aspéctos usualmente não considerados no projeto desses sistemas e avaliar a correção da implementação de técnicas de tolerância a faltas, antes mesmo que o sistema entre em operação. Um objetivo típico de uma tal política é o projeto de sistemas que, em primeiro lugar, tenham probabilidade elevada de entrar em operação correta, isto é, ter menor probabilidade de falha ao ser dada a partida na primeira vez. Em segundo lugar, objetiva-se usar (ou não) recursos redundantes tendo em vista esta característica, ou seja, o funcionamento correto na primeira vez, e não apenas o funcionamento correto após entrar em operação supondo que entrou em operação (embora de fato essa condição possa não ter ocorrido), ou o tempo de vida ou ainda a operação sem falhas por certo tempo, ou demais características usualmente avaliadas nestes casos. Em terceiro lugar, pode-se considerar a conveniência de introduzir a detetção, a diagnose e o

tratamento de faltas de fabricação, isto é, antes da entrada em serviço do sistema, o que permite corrigir falhas e efetuar reparos em subsistemas que falham nesta condição. A comparação realizada nos modelos I e II e os resultados obtidos nos modelos III e IV, demonstram que esta característica é muito benéfica para a confiabilidade e tempo médio entre falhas (ou tempo médio até a primeira falha). A localização das redundâncias pode evidenciar técnicas específicas de deteção e diagnose de faltas com esta finalidade.

APENDICE A - SUGESTÕES PARA NOVOS TRABALHOS.

Dada a importância do problema do controle de processos industriais e considerando os mais recentes avanços nas áreas de eletrônica e computação, pode-se prever que muitos trabalhos relacionados ao assunto abordado nesta tese, serão realizados, seguindo-se algumas sugestões.

1. Uma fábrica consiste na interconexão de diversos centros de trabalho, contendo máquinas e, naturalmente, operadores humanos, sendo os centros dotados de certa flexibilidade, cada máquina controlada por um controlador-processador programável, cada centro controlado por um microcomputador e todos os centros de trabalho coordenados por meio de um plano geral de produção computadorizado. A aplicação local de procedimentos de tolerância a faltas é discutida mas os efeitos com relação à habilidade da fábrica alcançar os requisitos de produção, bem como os efeitos no rendimento econômico, permanecem como questões a serem investigadas.

2. Tradicionalmente, questões concernentes à confiabilidade, disponibilidade e mantenabilidade, são consideradas globalmente. A incorporação de diagnose e tratamento de faltas, bem como de subsistemas redundantes, a nivel de circuito integrado, podem permitir a inclusão de procedimentos de tolerância a faltas, a nivel local, durante a fabricação do componente ou subsistema [Goode-85, Moore-84, Patterson-80, Sarrazin-84].

3. Embora o modelo geral tenha sido estabelecido

tendo em vista a fabricação e operação de computadores, sua aplicação pode ser estendida a qualquer outro sistema em que se deseje analisar o problema relativo a elementos de diferentes tipos e que podem apresentar defeitos durante o processo de fabricação. Diferentes graus de detalhamento podem ser utilizados e métodos numéricos podem ser aplicados.

4. O problema da coordenação de controladores e componentes que exercem o controle da máquina, centro de trabalho e a fábrica, tendo em vista a utilização ótima dos procedimentos de tolerância a faltas, deve também ser considerado.

5. As decisões tomadas durante a operação de uma fábrica requerem informações sobre o estado do sistema. Isso implica em que uma vasta quantidade de informações, coletada por sensores, deve ser processada. O projeto do sistema deve ser desenvolvido considerando essa base de dados juntamente com o planejamento e o controle do sistema [Kimemia-82]. A tendência em sistemas flexíveis e integrados de manufatura é a instalação de diversos computadores em níveis hierárquicos que devem gerenciar as operações dos diversos centros da fábrica, sendo o próprio sistema de controle também hierárquico. Essasolução poderá apresentar melhor confiabilidade por meio de redundâncias. A base de dados, de outro lado, não precisa ser centralizada, podendo cada controlador manter as informações detalhadas e que lhe são concernentes enquanto envia conjuntos de dados para os controladores vizinhos e de niveis superiores que coordenam o planejamento e o controle da produção.

6. Modelos para controle de manufatura, conside-

rando taxas de falha como funções da carga de trabalho, podem ser desenvolvidos e aplicados. Em um centro de trabalho cujas máquinas e controladores estejam sobrecarregados, podemos esperar maiores taxas de quebra. Isto é particularmente mais provável em sistemas de manufatura onde a quebra de ferramentas e defeitos em peças produzidas são fontes de paradas das máquinas.

7. Estruturas dedicadas e tolerantes a faltas podem ser estudadas como, por exemplo, controladores de máquinas específicas. Novas estruturas poderão ser propostas a partir da análise de benefícios que possam ser obtidos pela utilização sistemática de procedimentos de tolerância a faltas.

8. A modelagem de faltas transitórias, tanto durante a fabricação quanto durante a operação de sistema como o descrito, não foi abordada. Faltas com taxas constantes podem ser usadas, considerando sua duração como função exponencial.

9. A nomenclatura, padronização, normalização e definição de termos e conceitos em português, relativos à tolerância a faltas em geral e, em particular, relativos a sistemas de manufatura, merecem atenção.

Finalmente, deseja-se destacar o fato de que o conhecimento e a tecnologia atuais permitem a construção de elementos de elevada confiabilidade e seu custo, peso dimensões e elevada integração de componentes, permitem a implementação de redundâncias a nivel de circuito integrado. Arquiteturas com caracteristcas de tolerância a faltas podem ser implementadas. O resultado potencial é o longo prazo obtido para um sistema em manter sua operação, sem um suporte técnico ou logistico para sua

continuidade. Além disso, a diagnose e a manutenção do sistema poderão ser muito simplificados e realizados sem que o sistema precise ser desligado.

CAPITULO VIII - REFERENCIAS BIBLIOGRAFICAS.

[1] Adi, W., Fast Burst Error-Correction Scheme with Fire Code, IEEE Trans. Comp. Vol. C-33 No. 7 July 1984 (613-618).

[2] Aggarwal, K.K., Redundancy Optimization in General Systems, IEEE Trans. Reliab., Vol. R-25 No. 5, December 1976 (330-332).

[3] Ajmore, M.M. & Gerla, M., Markov Models for Multiple Bus Multiprocressor Systems, IEEE Trans. Comput., Vol C-31, March 1982 (239-248).

[4] Akers Jr., S.B., Universal Test Sets for Logic Networks, IEEE Trans. Comput., Vol. C-22 No. 9, September 1973 (835-839).

[5] Albus, J.S. et alii, Hierarchical Control for Robots in an Automated Factory, Proc. 13 ISIR/Robots 7 Symposium, Chicago, Illinois, April 1983.

[6] Andersen, A & Co., IMPACT EXHIBIT Cathalog, Chicago, June 1986.

[7] Anderson, T. & Lee, P.A., Fault Tolerance Principles and Practice, Prentice Hall International, Inc., N.J., 1981.

[8] Arsenault, J.E. & Roberts, J.A. Editors, Reliability & Maintanability of Electronic Systems, Computer Science Press, Inc., Maryland, 1980.

[9] Avizienis, A.; Kelly, M., Fault-Tolerance by Design Diversity: Concepts and Experiments, Computer Vol. 17 No. 8

Aug. 1984 (67-80).

[10] Avizienis, A., et alii, The STAR (Self-Testing And Repairing) Computer: An Investigation of the Theory and Practice of Faut-Tolerant Computer Design, IEEE Trans. Comp. Vol. C-20 No. 11 Nov. 1971 (1312-1321).

[11] Avizienis, A., Fault-Tolerance: The Survival Attribute of Digital Systems, Proc. IEEE Vol. 66 No. 10 Oct. 1978 (1109-1125).

[12] Avizienis, A., Arithmetic Error Codes: Cost and Effectiveness Studies for Application in Digital System Design, IEEE Trans. Comp. Vol. C-20 No. 11 Nov. 1971 (1322-1331).

[13] Bakker, J.J.A., A Control Architecture for Automated Production Systems, Proc. 16th International Symposium on Industrila Robots & 6th International Conference on Industrial Robots Technology, 1986.

[14] Bardell, P.H. & Mc Anney, W.H., Pseudorandom Arrays for Built-In Tests, IEEE Trans. Comput., Vol. C-33 No. 7, July 1986 (653-658).

[15] Barrientos, C. & Klein, C.A., Design of a Multimicroprocesor Based Controller Using a Structured Design Approach, IEEE Trans. Ind. Electr. Vol. IE-31 No. 4 Nov. 1984 (292-298).

[16] Barsi, F. et alii, A Theory of Diagnosability of Digital Systems, IEEE Trans. Comput., Vol. C-25 No. 6, June 1976 (585-593).

[17] Bazowsky, I. & Bazowsky Jr., I, MTBF Formulas in Complex

Mission Scenarios, Proceedings 1985 Reliability and Maintainability Symposium, IEEE, Philadelphia (131-134).

[18] Berhard, R., Reliability Test Procedures, IEEE Spectrum Vol. 18 No. 10 Oct. 1981 (67-68).

[19] Belli, F. et alii, Methoden und Modelle der Fehlertoleranz, Informatik Spektrum, No. 9, 1986 (68-81).

[20] Bernhart, D., Mathematical Models for Calculating the Reliability of Multicomputer Systems, Siemens Forsch. Bd. 9 (1980) Nr. 3 (168-173).

[21] Billington, R. & Allan, R.N., Reliability Evaluation of Engineering Systems - Concepts and Techniques, Pitman Books Ltd., London, 1983.

[22] Bochmann, G. von, Distributed Systems Design, Springer-Verlag Berlin Heidelberg, 1983.

[23] Bozic, S. & Shaw, L., Is BIT a Toy, Blessing or Annoyance?, Proceedings 1985 Reliability and Maintainability Symposium, IEEE, Philadelphia (270-275).

[24] Bouricius, W.G. et alii, Reliability Modeling Techniques for Self-Repairing Computer Systems, Proc. 24th. National Conf. Association Computing Machines, 1969 (295-309).

[25] Bouricius, W.G. et alii, Reliability Modeling for Fault-Tolerant Computers, IEEE Trans. Comput., Vol. C-20 No. 11, November 1971 (1306-1311).

[26] Brahme, D.; Abraham, J.A., Functional Testing of Microprocessors, IEEE Trans. Comp. Vol. C-33 No. 6 June 1984 (475-485).

[27] Breuer, M.A.; Friedman, A.D., Diagnosis & Reliable Design

of Digital Systems, Computer Science Press, Maryland 1976.

[28] Buyukkoc, C., An Approximation Method for Feedforward Queueing Networks with Finite Buffers - A Manufacturing Perpective, IEEE Intern. Conf. Robotics & Automation, San Francisco, April 1986 (965-972).

[29] Carter, W.C.; Mc Carthy, C.E., Implementation of an Experimental Fault-Tolerant Memory System, IEEE Trans. Comp. Vol. C-25 No. 6 June 1976 (557-568).

[30] Cenker, R.P., A Fault-Tolerant 64 K Dynamic RAM, IEEE Trans., Vol. ED-26, june 1979 (853-860).

[31] Cervo, A.L. & Bervian, P.A., Metodologia Cientifica, Mc Graw Hill do Brasil Ltda., S.P., 1973.

[32] Chen, C.L., Byte Oriented Error-Correcting Codes for Semiconductor Memory Systems, IEEE Trans. Comput., Vol. C-35 No. 7, July 1986 (646-648).

[33] Cliff, R.A., Acceptable Testing of VLSI Components Which Contain Error Correctors, IEEE Trans. Comp. Vol. C-29 No. 2 Feb. 1980 (125-134).

[34] Cooke, D., et alii, Basic Statistical Computing, E. Arnold, Sussex, 1981.

[35] Couto, C.A.M., A Integração da Manufatura na Indústria Metal Mecânica, SBS: Contr. &Autom., Vol. 1 No. 3, Julho 1987 (184-193).

[36] Crouzet, Y. & Landrault, C., Design of Self-Checking MOS-LSI Circuits: Application to a Four-Bit Microprocessor, IEEE Trans. Comp. Vol. C-29 No. 6 June 1980 (532-537).

[37] Di Cesare, F., et alii, Functions of a Manufacturing

Workstation Controller, IEEE Proc. International Conference Robotics & Automation, San Francisco, California, April 7-10, 1986.

[38] Dhillon, B.S., The Analisys of the Reliability of Multistate Device Networks, Ph.D. Thesis, Ottawa, 1975.

[39] Dhillon, B.S., A Hazard Rate Model, IEEE Trans. Reliab. Vol. R-28 1979.

[40] Dhillon, B.S. & Singh, C., Engineering Reliability, J. Wiley & Sons, NY 1981.

[41] Dhillon, B.S. & Rayapati, S.N., Analysis os Redundant Systems with Human Errors, Proceedings 1985 Reliability and Maintainability Symposium, IEEE, Philadelphia (315-321).

[42] Drogin, E., Super Computers, Defense Electronics Nov. 1984 (31).

[43] Elkind, S.A. & Siewiorek, D.P., Reliability and Performance of Error-Correcting Memory and Register Arrays, IEEE Trans. Comp. Vol. C-29 No. 10 Oct. 1980 (920-927).

[44] Emmerson, R. & Mc Gowan, M.J., Fault-Tolerance Achieved in VLSI, IEEE Micro Dec. 1984 (34-43).

[45] Fautazzi, G., Multiple Fault Detection and Location in Fan-Out Free Combinational Circuits, IEEE Trans. Comput., Vol. C-23 No. 1, January 1974 (48-55).

[46] Fiorentino, E. & Soistman Jr., E.C., Combined Hardware/ Software Reliability Predictions, Proceedings 1985 Reliability and Maintainability Symposium, IEEE, Philadelphia (169-176).

[47] Fleming, R.E. et alii, Complex System RMA and T, Using

Markow Models, Proceedings 1985 Reliability and Maintainability Symposium, IEEE, Philadelphia (125-130).

[48] Friedman, A.D., Easily Testable Iterative Systems, IEEE Trans. Comp. Vol. C-22 No. 12 Dec. 1973 (1061-1064).

[49] Fujiwara, E., et alii, A Self-Testing Group Parity Checker and Its Use for Built-In Testing, IEEE Trans. Comp. Vol. C-33 No. 6 June 1984 (578-583).

[50] Fujiwara, H., Design for Testability and Bult-In Self-Test for VLSI Circuits, Microproc. & Microsyst., Vol. 10 No. 3, April 1986 (139-147).

[51] Gehringer, E.F., et alii, The Cm* Testbed, Computer Vol 15 No. 10 Oct. 1982 (40-53).

[52] Gilley, G.C., Automatic Maintenance of Spacecraft Systems for Long-life Deep-space Missions, Ph.D. dissertation, Dep. Comput. Sci., Univ. California, Los Angeles, Sept. 1970.

[53] Gnedenko, B.V. & Khinchin, A.Ya., An Elementary Introduction to the Theory of Probability, Dover Publ. Inc., N.Y., 1972.

[54] Grasso, P.A., et alii, Memory Interference in Multimicroprocessor Systems with Time-Shared Bus, IEE Proc. Vol. 131 Part E No. 2 March 1984 (61-68).

[55] Goode, A.J., Design Considerations for a Single-Chip Fault Tolerant VLSI Microprocessor, Software & Microsystems, Vol. 4 No. 3, June 1985 (53-58).

[56] Goundan, A., Hayes, J.P., Identification of Equivalent Faults in Logic Networks, IEEE Trans. Comp. Vol. C-29 No. 11 Nov. 1980 (978-985).

[57] Goyal, A.; Agerwala, T., Performance Analysis of Future Shared Storage Systems, IBM J. Res. Develop. Vol. 28 No. 1 Jan. 1984 (95-108).

[58] Gupta, A. & Toong, H.D., Microcomputers in Industrial Control Applications, IEEE trans. Ind. Electr. Vol. IE-31 No. 2 May 1984 (109-119).

[59] Hac, A., A System Reliability Model with Classes of Failures, IEEE Trans. Reliab. Vol. R-34 No. 1 April 1985 (29-33).

[60] Haire, M.J. et alii, A System Availability Top-Down Apportionment Method, Proceedings 195 Reliability and Maintainability Symposium, IEEE, Philadelphia (306-314).

[61] Han, M.H. & Mc Ginnis, L.F., Throughput Maximization in Short Cycle Automated Manufacturing, San Francisco, April 1986 (147-151).

[62] Hayes, J.P., On Modifying Logic Networks to Improve Their Diagnosability, IEEE Trans. Comput., Vol. C-23 No. 1, January 1974 (56-62).

[63] Hayes, J.P.; Mc Cluskey, E.J., Testability Considerations in Microprocessor-Based Design, IEEE Computer March 1980 (17-26).

[64] Hlawiczka, A., Compression of Three-State Data Serial Streams by Means of a Parallel Linear Feedback Shift Register Signature Analyser, IEEETrans. Comput., Vol. C-35 No. 8, August 1986 (732-741).

[65] Holberly, W.S., Maintainability Considerations in a Fault-Tolerant/Fault-Proof Systems Design, IEEE Trans. Ind.

Electr. Vol. IE-31 No. 2 May 1984 (120-129).

[66] Hopkins, A.L.Jr., Fault-Tolerant System Design: Broad Brush and Fine Print, IEEE Computer March 1980 (39-45).

[67] Hopkins, A.L.Jr., The Architectural Elements of a Symmetric Fault-Tolerant Multiprocessor, IEEE Trans. Comp. Vol. C-24 No. 5 May 1975 (498-505).

[68] Hopkins, A.L.Jr. et alii, FTMP-A Highly Reliable Fault-Tolerant Multiprocessor for Aircraft, Proc. IEEE Vol. 66 No. 10 Oct. 1978 (1221-1239).

[69] Humphry, J.A., Fault-Tolerance and Micros in the Real World, IEEE Micro Vol. 4 No. 6 Dec. 1984.

[70] Hwang, K.; Chang, T., Combinational Reliability Analysis of Multiprocessor Computers, IEEE Trans. Reliab. Vol. R-31 No. 5 Dec. 1982

[71] Ihara,. H.; Mori, K., Autonomous Decentralized Computer Control Systems, Computer Vol. 17 No. 8 August 1984 (57-66)

[72] Johnson, B.W., Fault-Tolerant Microprocessor-Based Systems, IEEE Micro Vol. 4 No. 6 Dec. 1984.

[73] Johnson, D.B., The Intel 432: A VLSI Architecture for Fault- Tolerant Computer Systems, Computer Vol. 17 No. 8 Aug. 1984 (40-48).

[74] Jet Propulsion Laboratory, Fault-Tolerant Building Block Computer Study, California Institute of Technology, Pasadena Ca., July 1978, JPL Publication 78-67.

[75] Khakbaz, J.; Mc Clusckey, E.J., Self-Testing Embedded Parity Checkers, IEEE Trans. Comp. Vol. C-33 No. 8 August 1984.

[76] Kamayama, M., Higushi, T., Design of Dependent-Failure-Tolerant Microcomputer System Using Triple-Modular Redundancy, IEEE Trans. Comp. Vol. C-29 No. 2 Feb. 1980 (202-205).

[77] Karunanithi, S., Friedman, A.D., Analysis of Digital Systems Using a New Measure of System Diagnosis, IEEE Trans. Comp. Vol. C-28 No. 2 Feb. 1979 (121-133).

[78] Kautz, W.H., Fault Testing and Diagnosis in Combinational Digital Circuits, IEEE Trans. Comput. Vol. C-17 No. 4, April 1968 (352-366).

[79] Kimemia, J.G., Hierarchical Control of Production in Flexible Manufacturing Systems, Ph.D. Thesis MIT Mass. 1982

[80] Komoda, N. et alii, An Autonomous and Decentralized Control System for Factory Automation, IEEE Computer, Dec. 1984 (73-83).

[81] Koren, I., Breuer, M.A., On Area and Yield Considerations for Fault-Tolerant VLSI Processor Arrays, IEEE Trans. Comp. Vol. C-33 No. 1 Jan. 1984 (21-27).

[82] Koren, I. & Pradham, D.K., Effect of Redundancy on Yield and Performance of VLSI Systems, IEEE Trans. Comput. Vol. C-36, No. 3, March 1987 (344-355).

[83] Koren, I.; Sadeh, E., A New Approach to the Evaluation of the Reliability of Digital Systems, IEEE Trans. Comp. Vol. C-29 No. 3 March 1980 (261-267).

[84] Koren, I., Schalev, E., Reliability Analysis of Hybrid Redundancy Systems, IEE Proc. E. Comp. Dig. Tech. Vol. 133 Part E No. 1 Jan. 1984 (31-36).

[85] Kraft, G.D. & Toy, W.N., Mini/Microcomputer Hardware Design Prentice Hall, Inc., N.J., 1979.

[86] Laduzinsky, A.J., Fault-Tolerant Control Means Never Having to Say You're Sorry, Control Eng. Feb. 1985 (71-74).

[87] Laduzinsky, A.J., Programmable Controler Delivers Fault Tolerance Using Twelve Microprocessors, Control Eng. Feb. 1985 (76-77).

[88] Lambert, J.E.; Harshal, F., Program Debugging and Performance Evaluation Aids for a Multi-Microprocessor Development System, IEE Software and Microsystems Vol. 3 No. 1 Feb. 1983 (2-10).

[89] La Pointe, J.T. & Forry, A.M., SYSREL: Reliability of Complex Redundant Systems, Proceedings 1985 Reliability and Maintainability Symposium, IEEE, Philadelphia (63-68).

[90] Lawrence, J.D.; Sillyman, S., Computer Control Boosts Burn-In Effectiveness, Industrial Electronics Equipment Design March 1985 (25-27).

[91] Leighton, F.T. & Leiserson, C.E., Wafer-Scale Integration of Systolic Arrays, IEEE Trans. Comput. Vol. C-34, May 1985 (448-461).

[92] Lerner, E.J., Modeling Techniques, IEEE Spectrum Vol. 18 No. 10 Oct. 1981 (50-55).

[93] Leveson, N.G., Software Safety in Computer-Controlled Systems, IEEE Computer Feb. 1984 (48-55).

[94] Lipow, M., A New Approach to Software Reliability, Proceedings 1985 Reliabilty and Maintainability Symposium,

IEEE, Philadelphia (262-264).

[95] Leabardi, F., Investigation and Design of a Controller of an Asyncronous System for Fault-Tolerant Aircraft Control Using Hybrid Voting Techniques, Software and Microsystems Vol. 3 No. 1 Feb. 1984 (11-18).

[96] Loques, O.G.; Kramer, J., A Flexible Fault-Tolerant Distributed Computer System (a ser publicado) PUC-RJ.

[97] Losq, J., A Highly Efficient Redundancy Scheme: Self Purging Redundancy, IEEE Trans. Comput., Vol. C-25 No. 6, June 1976 (569-578).

[98] Machulda, J., Fault-Tolerant Control: You Can Afford It, Inech ISA, September 1985 (105-108).

[99] Martucci Jr., M., Automação Industrial, Revista IF No. 147, set./out. 1984 (8-16).

[100] Matick, R.E., Comparison of Memory Chip Organizations vs Reliability in Virtual Memories, IEEE Trans. Reliab. Vol. R-32 No. 1 April 1983 (48-58).

[101] Matick, R.E.; Ling, D.T., Architecture Implications in the Design of Microprocessors, IBM Systems J. Vol. 23 No. 3 1984 (264-280).

[102] Mathur, F.P., On Reliability Modeling and Analysis of Ultra- Reliable Fault-Tolerant Digital Systems, IEEE Trans. Comp., Vol. C-20 No. 11, Nov. 1971 (1376-1382).

[103] Mellado, E.L. & Alami, R., An Execution Monitoring System for a Flexible Assembly Workcell, Proc. 16th Intern Symp. Ind. Robots and 8th Intern. Conf. Ind. Robots Technology, 1986 (955-962).

[104] Merry, M., APEX 3: An Expert System Shell for Fault Diagnosis, The GEC J. Research Vol. 1 No. 1, 1983 (39-47).

[105] Meyer, J.F., et alii, Performability Evaluation of the SIFT Computer, IEEE Trans. Comp. Vol. C-29 No. 6 June 1980 (501-509).

[106] Miller, E., Reliability Aspects of the Variable Instruction Computer, IEEE Trans. Electr. Comp., Vol. EC-16 No. 5, oct. 1967 (596).

[107] Mc Gill, W.F., Fault-Tolerance in Continuous Process Control IEEE Micro Dec. 1984 (22-33).

[108] Morris, H.M., Profitable Robotic Workcells Result from Interconecting the Islands of Automation, Control Engineering, May 1985 (81-84).

[109] Moore, W.R., A Review of Fault-Tolerant Techniques for the Enchancement of Integrated Circuit Yield, The GEC J. of Research Vol.2 No. 1 1984 (1-15).

[110] Moore, E.F. & Shannon, C.E., Reliable Circuits Using Less Reliable Relays, J. Franklin Inst. 262, 1956 (191).

[111] Nakamura, A. et alii, Controller for Industrial Robots, IEEE Proc. International Conference Robotics & Automation, San Francisco, California, April 7-10, 1986.

[112] von Neumann, J., Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components, Automata Studies S-343, Princeton Univ. Press, 1956 (43-98).

[113] Ng, Y.W.; Avizienis, A.A., A Unified Reliability Model for Fault-Tolerant Computers, IEEE Trans. Comp. Vol. C-29 No. 11, Nov. 1980 (1002-1011).

[114] Pacheco, C.A. & Tapia, J.R.B., Informática e Reserva de Mercado, Caderno para Discussão 2 NPCT-UNICAMP, Campinas, Set. 1983 (37-57).

[115] Paker, Y., Multiprocessor Systems, Academic Press Inc., London 1984.

[116] Papasratorn, B.; Prapinmongkolkarn, P., A Small-Scale Distributed Microprocessor System Using Shared Memory Technique, IEEE Trans. Ind. Electr. Vol. IE-32 No. 2 May 1985 (97-102).

[117] Patterson, D.A.; Séquin, C.H., Design Considerations for Single Chip Computers in the Future, IEEE Trans. on Comp. Vol. C-29 No. 2 Feb. 1980 (108-116).

[118] Patterson, D.A.; Séquin, C.H., A VLSI RISC, IEEE Computer, Vol. 15 No. 9, Sep. 1082 (8-21).

[119] Peattie, G., Quality Control of IC's, IEEE Spectrum Vol. 18 No. 10 Oct. 1981 (93-97).

[120] Peterson, W.W. & Weldon Jr., E.J., Error Correcting Codes, Cambridge, MA, MIT Press, 1972.

[121] Pradhan, D.K., Dynamically Restructurable Fault-Tolerant Processor Network Architectures, IEEE Trans. Comp. Vol. C-34 No. 5 May 1985 (434-447).

[122] Prates, M., Automação Industrial: Oportunidades Tecnológicas no Estado de São Paulo, Promocet SICCT, São Paulo, 1984.

[123] Randel, B., Software Fault Tolerance, Proc. EURO IFIP 1979 (721)

[124] Rembolt, A,et alii, Computer Integrated Manufacturing,

Marcel Dekker Inc., NY, 1984.

[125] Rennels, D.A., Distributed Fault-Tolerant Computer Systems, Computer Vol. 13 No. 3 March 1980 (55-65).

[126] Rennels, D.A., Architectures for Fault-Tolerant Spacecraft Computers, Proc. IEEE Vol. 66 N. 10 Oct. 1978 (1255-1268).

[127] Rohrer, H., et alii, User Definable Architecture of Special-Purpose Multiprocessor Systems Using a Set of LSI Modules, Computing Suppl. 3 1981 (149-157).

[128] Rosemberg, A.L., The Diogenes Approach to Testable Fault Tolerant Arrays of Processors, IEEE Trans. Comput. Vol. -32 october 1983 (902-910).

[129] Rutledge, R.A., Models for the Reliability of Memory with ECC, Proceedings 1985 Reliability and Maintainability Symposium, IEEE, Philadelphia (57-62).

[130] Sarrazin, D.B.; Malek, M., Fault-Tolerant Semiconductor Memories, Computer Vol. 17 No. 8 Aug. 1984 (49-56).

[131] Schmitter, E.J.; Baues, P., The Basic Fault-Tolerant System, IEEE Micro Vol. 4 No. 1 Feb. 1984 (66-74).

[132] Sedmak, R.M. & Liebergot, H.L., Fault Tolerance of a General Purpose Comuter Implemented in VLSI Integration, IEEE Trans. C-29 (6), 1980 (492-500).

[133] Serlin, O., Fault-Tolerant Systems in Commercial Application IEEE Computer 17, No. 8, 1984.

[134] Shedletsky, J. & Mc Cluskey, E.J., The Error Latency of a Fault in a Sequential Digital Circuit, IEEE Trans. Comput., Vol. C-25 No. 6, June 1976 (655-659).

[135] Shooman, M.L., Probability Reliability, an Engineering Approach, Mc Graw Hill Co., N.Y., 1968.

[136] Siewiorek, D.P., Architecture of Fault-Tolerant Computers, Computer Vol. 17 No. 8 Aug. 1984 (9-18).

[137] Shen, J.P., Hayes, J.P., Fault-Tolerance of Dynamic-Full-Access Interconnection Networks, IEEE Trans. Comp. Vol. C-33 No. 3 March 1984 (241-248).

[138] Slusarczyk, R., What Does the Future for Military IC's ?, Defense Elect. July 1985 (90-91).

[139] Snyder, L., Introduction to the Configurable Highly Parallel Computer, Computer Vol. 15, January 1982 (47-56).

[140] Stapper, C.H. et alii, Integrated Circuit Yield Statistics, Proc. IEEE, Vol. 71, April 1983 (453-470).

[141] Stapper, C.H., et alii, Yield Model for Production Optimization of VLSI Memory Chips with Redundant and Partially Good Products, IBM J. Res. Develop., Vol 24 No. 3 May 1980 (398-409).

[142] Stiffer, J.J., The Vulnerability of Computers, IEEE Spectrum Vol. 18 No. 10 Oct. 1981 (44-46).

[143] Su, S.Y.H.; Du Casse, E., A Hardware Redundancy Reconfiguration Scheme for Tolerating Multiple Module Failures, IEEE Trans. on Comp. Vol. C-29 No. 3 March 1980 (254-258).

[144] Suri, R. & Zazanis, M.A., Robustness of Perturbation Analysis Estimates for Automated Manufacturing Systems, IEEE Intern. Conf. Robotics & Automation, San Francisco, April 1986 (311).

[145] Sykora, M.R., The Design and Application of Redundant Programmable Controlers, Control Eng. Vol. 29 No. 8 July 1982 (77-79).

[146] Tamir, Y.; Séquin, C.H., Design and Application of Self-Testing Comparators Implemented with MOS PLA's, IEEE Trans. Comp. Vol. C-33 No. 6 June 1984 (493-506).

[147] Tang, C., A Job Scheduling Moedl for a Flexible Manufacturing Machine, IEEE Intern. Conf. Robotics & Automation, San Francisco, April 1986 (152-155).

[148] Tanner, R.M., Fault-Tolerant 256K Memory Designs, IEEE Trans. Comp. Vol. C-33 No. 4 April 1984 (314-322).

[149] Teoste, R., Digital Circuit Redundancy, IEEE Trans. Reliab., Vol. R-13 No. 2, June 1964 (42-61).

[150] Thatte, S.M.; Abraham, J.A., Test Generation for Microprocessors, IEEE Trans. on Comp. Vol. C-29 No. 6 June 1980 (429-441).

[151] Tonshoff, H.K. et alii, Investigations on Increasing Flexibility of Hardware and Software of Industrial Robots, J. Manufac. Systems Vol. 3 No. 2 1985.

[152] Towsley, D., Approximate Models of Multiple Bus Multiprocessor Systems, IEEE Trans. Comput. Vol. C-35, March 1986 (220-228).

[153] Toy, W.N., Morganti, M., Fault-Tolernat Computing, Computer Vol. 17 No. 8 Aug. 1984 (6-7).

[154] Treuer, R. & Aggarwall, K.K., A New BIST for PLA's with High Coverage and Low Overhead, IEEE Trans. Comput. Vol. C-36, March 1987 (369-373).

[155] Ward, R.K. & Tabandeh, M., Error Correction and Detection, a Geometric Aproach, The Computer J., Vol 27 No. 3, 1984 (246-253).

[156] Warren, C., A Mix of Standard and Proprietary Buses Marks the Latest Microcomputer Systems, Electronic Design March 1983.

[157] Wensley, J.H. & Harclerode, S., Programmable Control of a Chemical Reactor Using a Fault-Tolerant Computer, IEEE Trans. Ind. Electr. Vol. IE-29 No. 4 Nov. 1982 (258-264).

[158] Wensley, J.H., SIFT: Design and Analysis of a Fault-Tolerant Computer for Aircraft Control, Proc. IEEE Vol. 66 No. 10 Oct. 1978 (1240-1255).

[159] Wesfreid, E., Dynamic Behavior of N-Stage Transfer Lines with Unreliable Machines and Finite Buffers, IEEE Intern. Conf. Robotics & Automation, San Francisco, April 1986 (973- 976).

[160] Williams, T.J., The Development of Reliability in Industrial Control Systems, IEEE Micro Dec. 1984 (66-80).

[161] Williams, W.C., Lessons from NASA, IEEE Spectrum Vol. 18 No. 10 Oct. 1981 (79-84).

[162] Wilson, P., OCCAM Architecture Eases System Design, Comp. Design Nov./Dec. 1983 Jan. 1984.

[163] Wulf, W.A., Compilers and Computer Architecture, Computer Vol. 14 No. 7, July 1981 (41-48).

[164] Yak, Y.W. et alli, Bounded Set Approach to the Evaluation of the Reliability of Fault-Tolerant Systems, IEE Proc. Vol 132 Pt.E No. 4 July 1985 (217-223 Part 1, 224-232 Part 2).