Tolerância a Falhas

Roteiro

- Introdução
- Histórico
- Áreas de aplicação
- Conceitos
- Detecção e Recuperação de erros
- Arquiteturas
- Técnicas de Tolerância a Falhas
- Referências

Introdução

- Meta dos sistemas tolerantes a falha
  - Nenhum defeito de qualquer componente deverá acarretar num defeito do sistema. Para que seja possível atingir esta meta, lança-se mão de estratégias redundantes por meio de hardware ou de software.

Histórico(1)

- 1953
  - "Diagnostic Programs and Marginal Checking for Large Scale Digital Computers"
    - Nova York
    - 5 artigos publicados
- Final dos anos 50
  - Bell Systems
    - ESS (Electronic Switching System)
      - Inoperância de 2h a cada 40 anos

Histórico(2)

- Década de 60
  - Programa de Exploração Espacial (NASA)
    - OAO (Orbiting Astronomical Observatory)
      - Confiabilidade de 95% para 1 ano de missão
    - Computador Guia da familia Saturno
      - Confiabilidade de 99% para 250h de missão

Histórico(3)

- Década de 70
  - SIFT - Software Implemented Fault Tolerance
  - FTMP - Fault Tolerant Multiprocessor
    - Utilizados em sistemas de transporte aereo de passageiros
    - Probabilidade de falha abaixo de 0,0000001 para 10h de vôo

Áreas de Aplicação(1)

- Aplicações críticas de STR
  - Medicina, controle de processos e controle áereo
- Aplicações seguras de STR
  - Transportes Urbanos
- Aplicações de STR de longa vida
  - Viagens espaciais, satélites e sondas

Áreas de Aplicação(2)

- Aplicações Técnicas
  - Telefonia e Telecomunicações
- Aplicações comerciais de alta disponibilidade
  - Sistemas de transação e Servidores de rede
- Internet
  - Equipamentos com alta disponibilidade

Conceitos

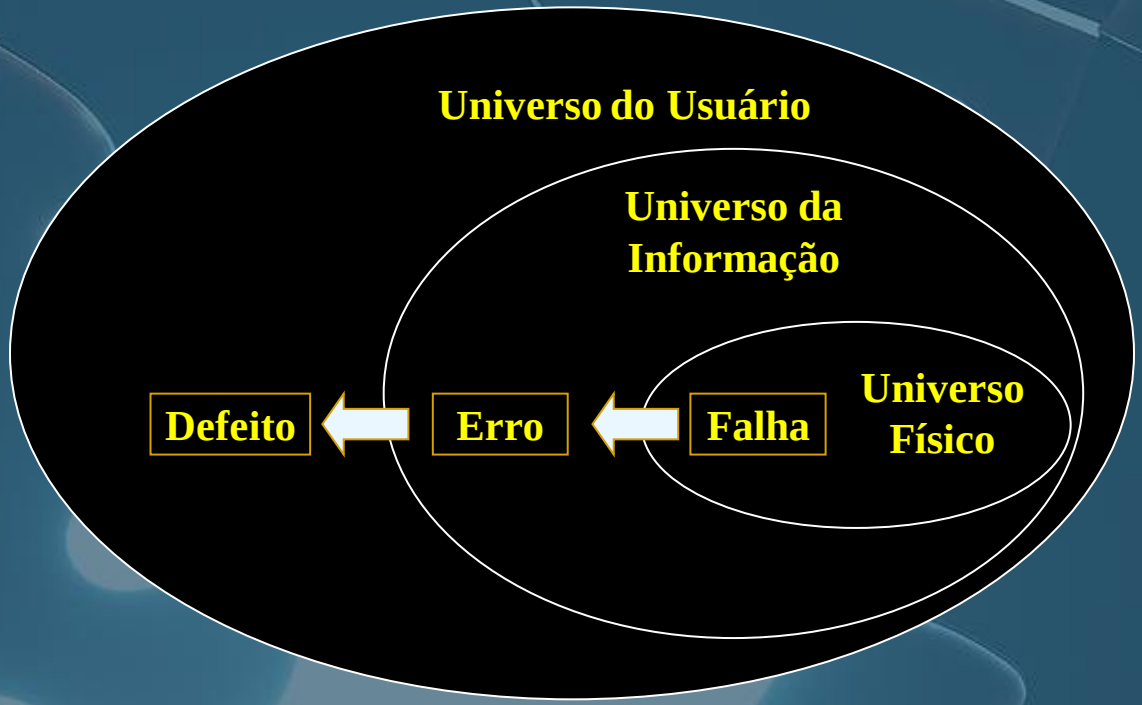
- Defeito
  - É definido como uma violação da especificação e não do projeto
- Erro
  - Parte do estado do sistema capaz de conduzir a um defeito
- Falha
  - É o fenômeno que provoca o surgimento do erro, que pode ser uma causa física ou algorítmica do erro

Conceitos

- A relação entre erro,falha e defeito em um modo simples é:
  - A manifestação sobre o sistema de uma falha gera um erro
  - A manifestação de um erro sobre o serviço gera um defeito

Tolerância a falhas
Terminologia

- Falha, erro e defeito
- Modelo dos Três Universos



Conceitos

- Tipos de falhas
  - Falhas físicas
  - Falhas cometidas pelo Homem
  - Falhas de Projeto (especificação)
  - Falhas de interação
- Tipos de Erros
  - Erro Latente
    - Criado após a ocorrência da falha
  - Erro Efetivo
    - Ocorre quando da ativação do erro latente

Etapas de Detecção e Recuperação de Erros

- Detecção de erro;
- Confinamento e avaliação do dano causado pela falha;
- Recuperação do erro;
- Tratamento da falha

Considerações iniciais sobre os testes de detecção

- Teste baseado na especificação;
- Completude e precisão do modelo;
- Especificações de testes e do sistema sem partes em comum;
- Recomenda-se testes por módulos

Estágios de atuação dos testes

- Verificação de último momento
  - Empregados após os resultados gerados
- Verificação de primeiro momento
  - Ativados durante a atividade do sistema

Técnicas de Detecção de Erros

- Replicação
- Temporização
- Coerência
- Reversão
- Diagnóstico

Redundância(1)

- Redundância de Hardware
  - Introdução de componentes extras
- Redundância de Software
  - Introdução de elementos
- Redundância de Informação
  - Duplicação de dados
- Redundância Temporal
  - Re-execução de tarefas

HARDWARE	SOFTWARE	INFORMAÇÃO
Componente	Micro-Instrução	Bit
Circuito	Instrução	Byte
Subsistema	Bloco ou módulo de Programa	Palavra
Sistema	Programa	Tabela
	Sistema Lógico	

Redundância(2)

- As redundâncias do tipo hardware ou software podem ser categorizadas como:
  - Estática (por mascaramento)
  - Dinâmica (por substituição)
  - Híbrido

Recuperação de erros

- Técnicas
  - Correção do erro
  - Troca de componente
  - Reconfiguração
- Abordagens
  - Por avanço
  - Por retorno

Arquiteturas de sistemas tolerantes a falhas

- Microprocessadores
  - Mestre-verificador
- Sistemas de grande porte
- Computadores de bordo
  - FTMP
  - SIFT

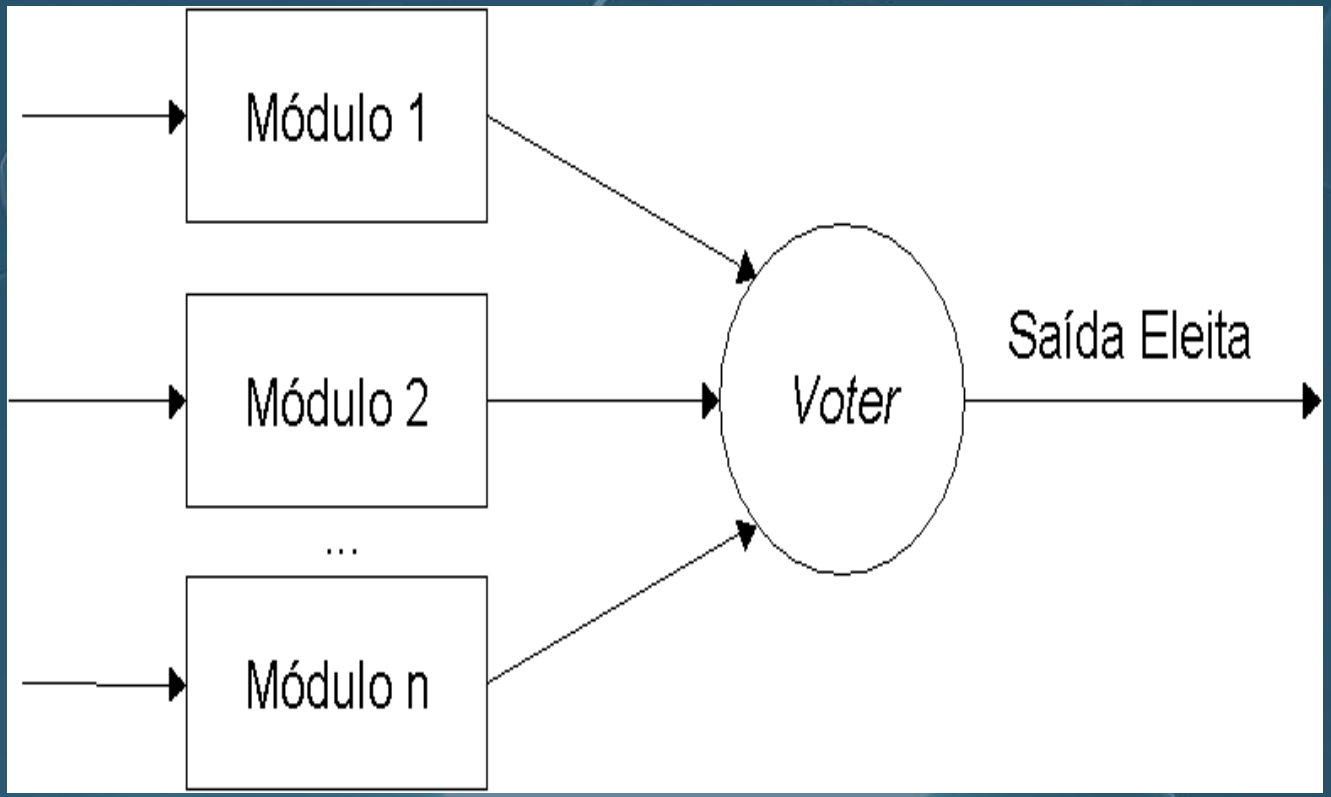
Técnicas de Tolerância a Falhas

- NMR
  - 2MR
  - TMR
  - Mid-value select
- Flux-Summing
- Standby Sparing
- NMRcom Spares
- Códigos de Detecção e Correção de erros
- Verificação de Consistência
- Verificação de Capacidade
- N-Versões
- Blocos de Recuperação
- Utilização de Dispositivos Específicos

NMR (n-modular redundancy)

- N módulos, com mesma funcionalidade
- Mecanismo de voto majoritário
- Alta confiabilidade
- Tipo de Redundância Utilizada
  - Hardware - replicação de componentes físicos
  - Tempo - um mesmo componente, porém realiza o processamento n vezes em instantes distintos

NMR



NMR

- Meio de Implementação
  - Hardware
  - Híbrida
- Objetivo
  - Detecção e mascaramento de falhas do sistema
- Problema do ponto único de falha (*single-point-of-failure*)
  - Confiabilidade limitada ao componente votante

2MR (Duplicação com Comparação)

- Variação da NMR
  - Utilização de apenas dois módulos replicados
  - Mecanismo de voto majoritário se resume a um comparador
- Objetivo
  - Permite apenas detecção de erros
  - Redundância de tempo - detecção de falhas temporárias
  - *Alternating Logic* – detecção de falhas permanentes
    - Codificação da entrada e decodificação da saída

TMR (Triple Modular Redundancy)

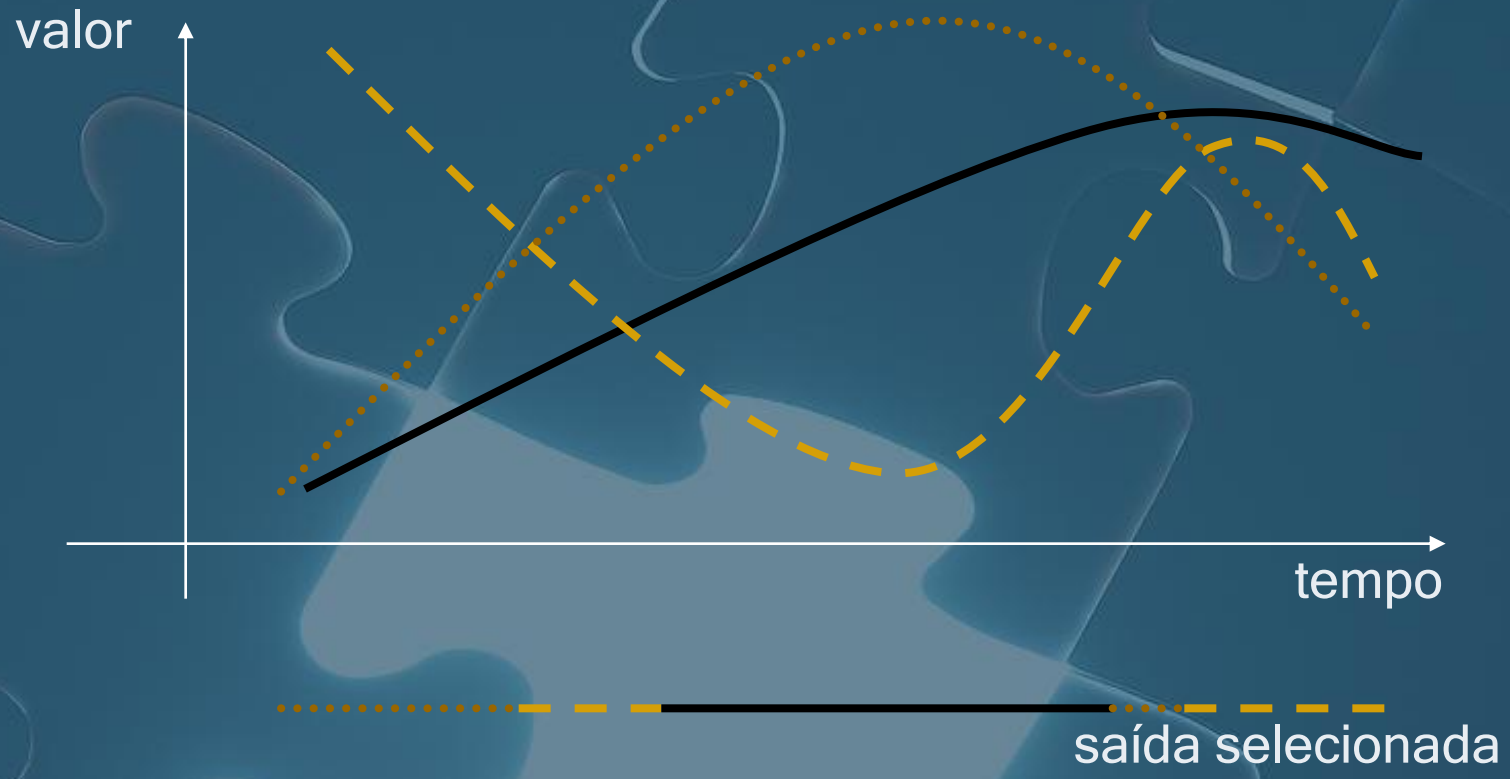
- Variação da NMR
  - Utilização de três módulos replicados
  - Forma mais utilizada das técnicas NMR
- Objetivos
  - Além de detectar, também mascara falhas

Mid-Value Select

- Variação da NMR
  - Não existe sistema de votação majoritária
  - Saída que apresenta o valor médio dentre as demais saídas
- Pequenas variações na saída não são erros
  - Conversão analógico-digital
- Numéro impar de módulos

Técnicas

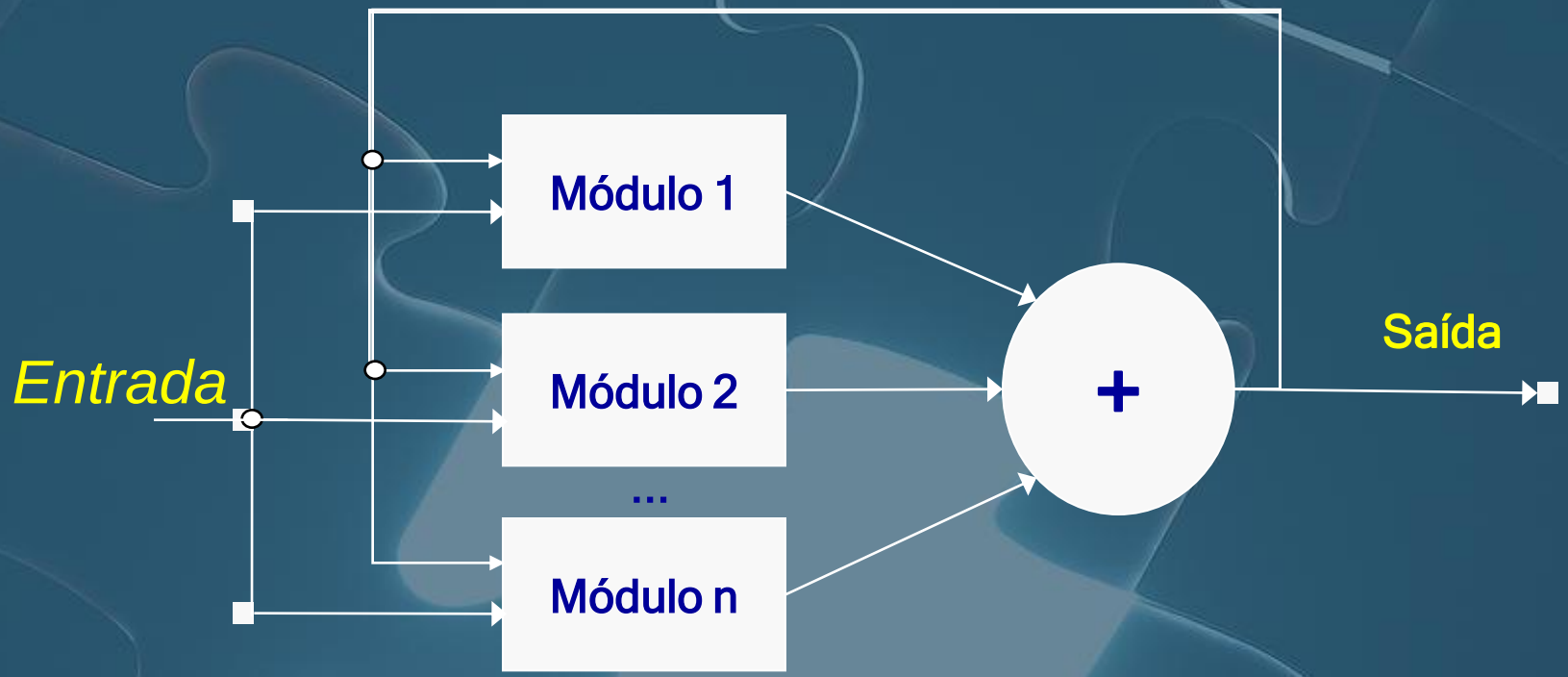
- Mid-Value Select



Flux-summing

- Utiliza a propriedade inerente dos sistemas de controle com realimentação
- Consiste em utilizar módulos redundantes e um transformador

Flux-summing



Flux-summing

- Objetivo
  - Mascarar as falhas do sistema
- Aspectos Gerais
  - Tolera falhas totais, falhas parciais e falhas transitórias

Standby Sparing

- Utiliza módulos redundantes
- Deve ser utilizada com algum método para detecção de falhas
- Duas abordagens
  - Hot Standby Sparing
  - Cold Standby Sparing

Standby Sparing

- Meio de Implementação
  - Hardware ou Software
- Objetivo
  - Reconfiguração do sistema

NMR com Spares

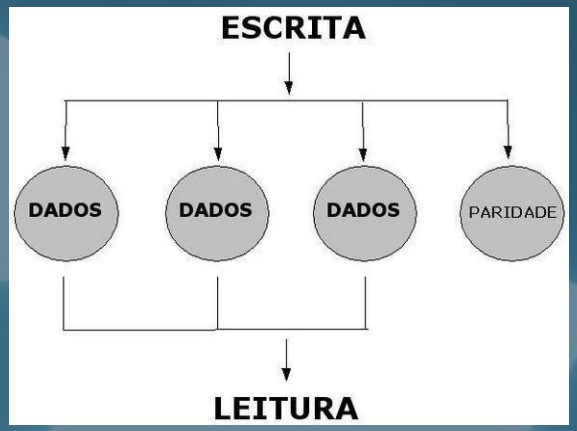
- Combina as técnicas NMR e Standby Sparing
- Consumo adicional de energia
- Pair-and-a-Spare
  - 2MR com um módulo redundante
- Objetivo
  - Detecção de falhas e reconfiguração do sistema

Códigos de Detecção e Correção de Erros

- Redundância de Informação
- Implementada por Hw ou Sw
- Tipos:
  - Duplicação
  - Paridade
  - Checksum
  - Cyclic Redundancy Check (CRC)
  - Código de Hamming

Códigos de Detecção e Correção de Erros

- Duplicação
- Paridade
  - 01001001 1 11001010 0
- RAID



Checksum

- Exemplo simples:
  - 0x25, 0x62, 0x3F, 0x52
  - 0x118 soma
  - 0xE8 complemento a 2
- Problemas:
  - Reordenação de bytes
  - Inserção ou deleção de zeros
  - Múltiplos erros cuja soma seja zero

CRC

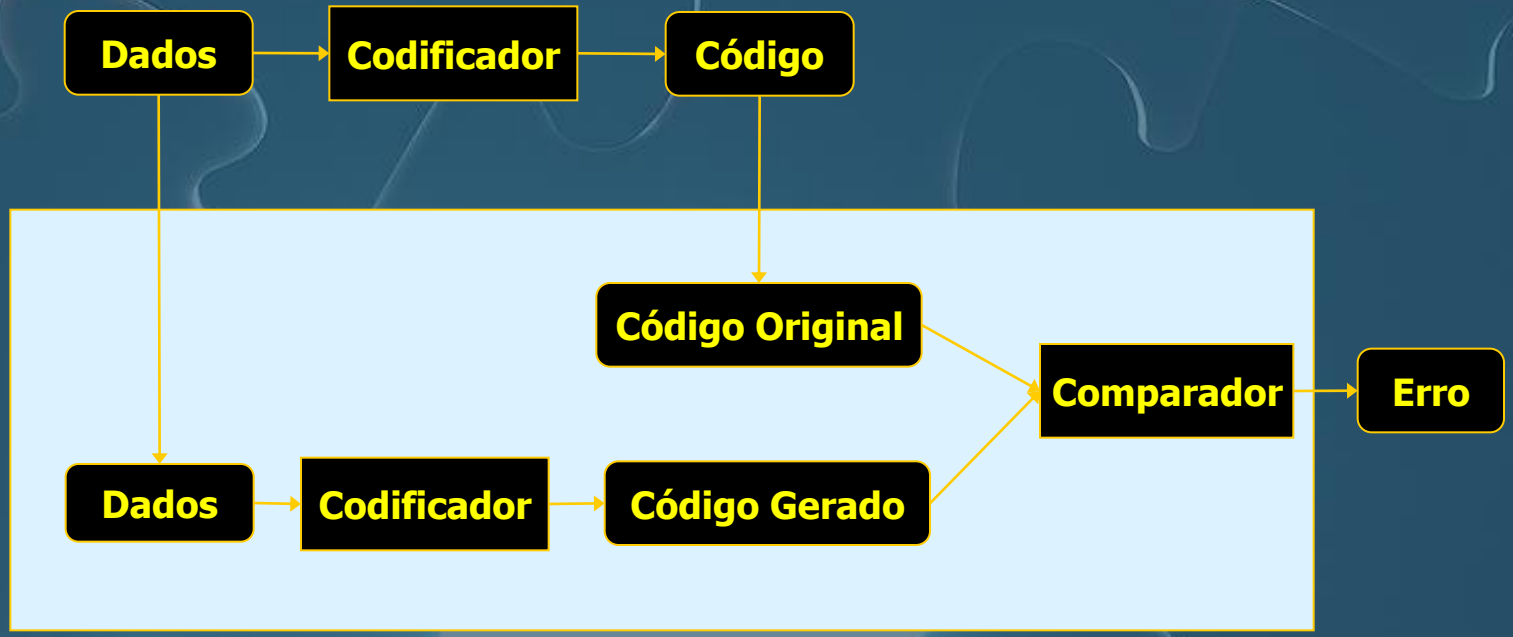
- Polinômio Gerador
  - $P(x) = x^4 + x^3 + x^0 = 11001$
- Divisão em aritmética módulo 2
- Alguns polinômios padronizados:
  - $CRC_1(x) = 1$ (paridade)
  - $CRC_5(x) = x^5 + x^2 + x^0$ (USB Token)
  - $CRC_{16}(x) = x^{16} + x^{15} + x^2 + x^0$ (USB Data)

Código de Hamming

- Corrige erro em 1 bit e detecta erro em até 2 bits
- Bits de paridade nas posições 2^k
- Hamming(7,4):
  - p1 p2 d1 p3 d2 d3 d4
- Exemplo:
  - ? ? 1 ? 0 1 1
  - 0 1 1 0 0 1 1

Tolerância a falhas
Técnicas

- Código de Hamming
  - Detecção e correção de erros



Verificação de Consistência (Consistency Checks)

- Redundância de Hw ou Sw
- Implementação mais comum em Sw
- Invariantes do sistema
- Verificações em pontos da computação

Verificação de Capacidade (Capability Checks)

- Verifica a capacidade do sistema antes da execução de alguma tarefa
- Verifica o funcionamento dos componentes do sistema

Verificação de Capacidade (Capability Checks)

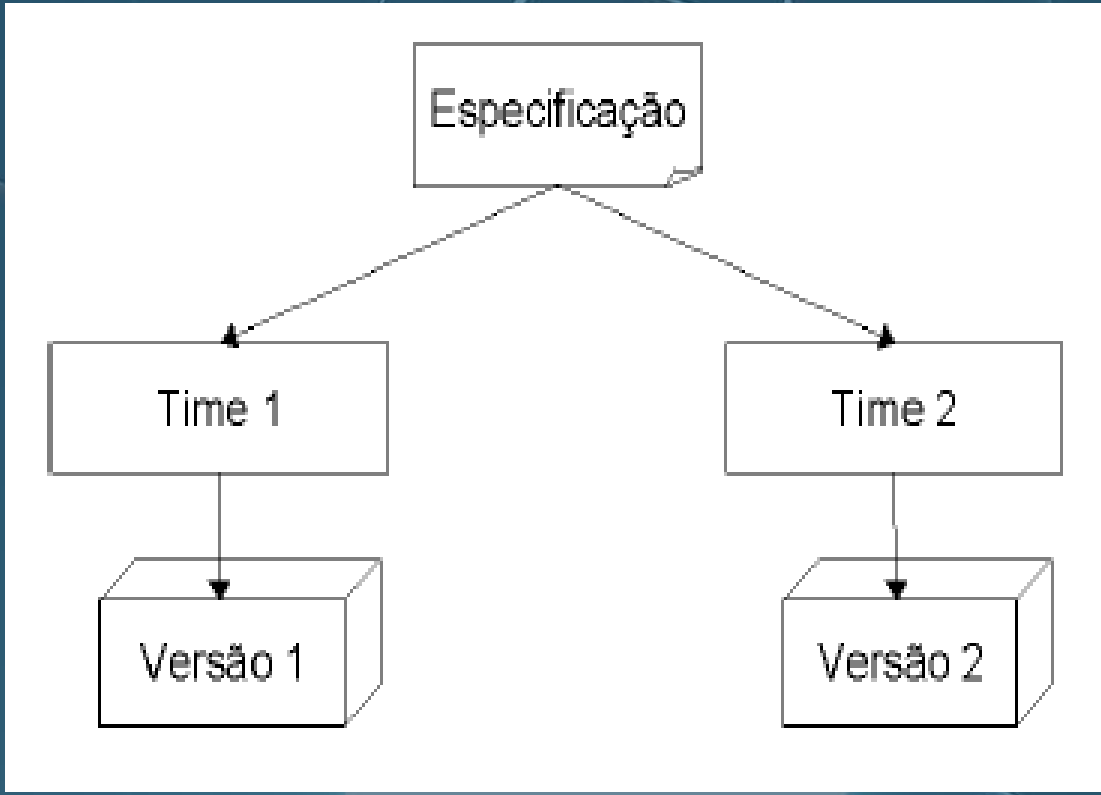
- Meio de Implementação
  - Comumente por software
- Objetivo
  - Detecção de falhas
- Aspectos Gerais

  Mais aplicada para sistemas concorrentes

N-Versões

- Versão de software da técnica NMR
- Implementações redundantes de uma mesma especificação de software
- Geralmente feita por times diferentes
- A resposta do sistema é escolhida por votação
- As várias versões são alimentadas pela mesma entrada

Esquema



Tipo de redundância utilizada

- A técnica utiliza redundância de software
- As versões podem dividir o mesmo processador ou executar em processadores diferentes.
- Se executarem no mesmo processador, executam em regime de time-sharing

Aspectos Gerais

- A técnica de N-versões consegue detectar falhas em tempo de projeto que a técnica de NMR não conseguiria encontrar.
- Alto custo de implementação + Hardware + pessoal
- Utilizado em sistemas de alta complexidade

Aplicação em sistemas embarcados

- Além da redundância de software a técnica também exige redundância de tempo e/ou de hardware
- Em sistemas de tempo real, aplica-se redundância de Hardware

Blocos de Recuperação

- N versões de software, cada uma com uma versão de detecção de erros diferente
- Um dos módulos(primário), é posto pra executar, se este falhar em seu teste de aceitação outo é colocado pra executar
- Desta forma N-1 erros são detectados e contornados

Comentários sobre a técnica

- É utilizada a redundância de software
- Assim como a N-versões são utilizadas para projetos com alta complexidade, e propensos a falhas de projeto.
- Pouco aplicável a sistemas embarcados e pricipalmente em sistemas de tempo real, onde os deadlines têm que ser respeitados.

Dispositivos específicos

- WatchDog Timer
- Power failure interrupt
- Oscilator failure interrupt
- Power On Interrupt

WatchDog Timer

- Temporizador que deve ser resetado perioódicamente, para indicar o bom funcionamento do sistema.
- Se o watchdog não for resetado significa que uma determinda tarefa estourou seu deadline, ou algum processo ficou prendendo o processador.
- Solução Geralmente adotada: reset CPU

Considerações Finais

- Em todo Sistema de Tempo Real é essencial aceitar a possibilidade de falha, para podermos prevenir, detectar e tratar.
  - Relembrando as 4 Fases (*Anderson & Lee):*
    - Detecção
    - Confinamento e avaliação
    - Recuperação
    - Tratamento da falha

Considerações Finais

- A conseqüência de falhas geralmente são mais caras que os esforços para sua previsão e tratamento.
- Sabendo que toda falha em STR tem conseqüências, é responsabilidade de um STR bem projetado evitar que essas falhas se propaguem e gerem perdas.

Dúvidas

Referências

- *Um roteiro para exploração dos coceitos básicos de tolerância a falhas* – Taisy Silva Weber
- *Estudo e implementação do Mecanismo de Tolerância a Falhas em software por meio de Blocos de Recuperação* – Sérgio Murilo Maciel Fernandes
- *Tolerância a Falhas para Sistemas Embarcados* – Ana Carla dos Oliveira Santos