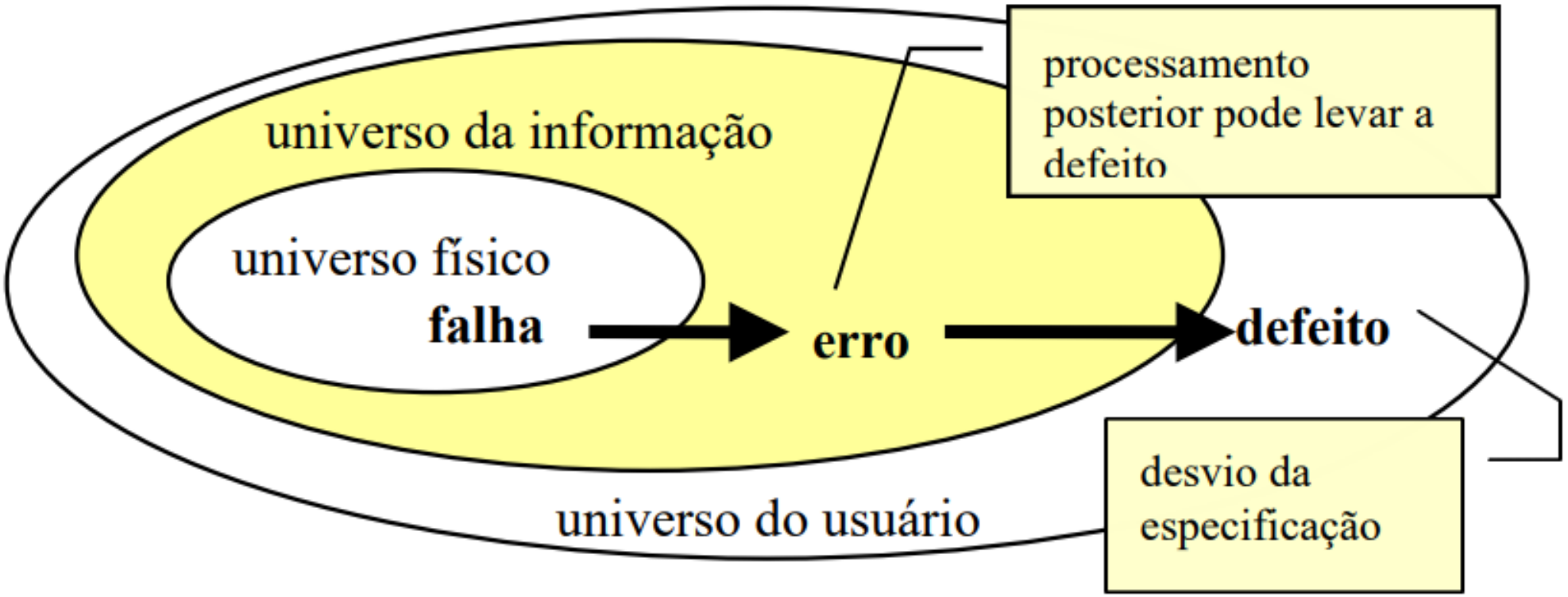
Tolerância à Falhas

Prof. Maurício Acconcia Dias

Microcontroladores II

Como entender o fenômeno



Tolerância à Falhas

- É um atributo que habilita o sistema para ser tolerante a falhas.

- É o conjunto de técnicas utilizadas para detectar, mascarar e tolerar falhas no sistema.

Sistema tolerante à falhas

- São aqueles que possuem a capacidade de continuar provendo corretamente os seus serviços mesmo na presença de falhas de hardware ou de software.
- São aqueles em que os defeitos não são visíveis para o usuário, pois o sistema detecta e mascara, ou recupera, defeitos antes que eles alcancem os limites do sistema (ponto de fuga da especificação).

Surgimento

- Aplicações críticas necessitam de sistemas confiáveis
- Porém, o estudo de sistemas tolerantes a falha também tem seu foco direcionado para casos menos rígidos.

Histórico (+/-)

- 1834: primeira afirmação sobre erros em cálculos computacionais (Dr. Lardner em Babbages's calculating engine)
- Final 40 até meio 50: técnicas baseadas em redundância para melhorar confiabilidade
- 1965: teorias de mascaramento por redundância foram relacionadas ao conceito de Failure Tolerance (Pierce).
- 1967: técnicas práticas de detecção de erros, diagnóstico de falhas e recuperação agrupadas no conceito de sistemas tolerantes a falhas (Avizienis)

Histórico (+/-)

- 1969: conceito de cobertura de falhas, no campo da modelagem de confiabilidade (Bouricius, Carter e Schneider)
- 1970: criado o IEEE-CS TC on Fault-Tolerant Computing.
- 1975: conceitos sobre software tolerante a falhas (Randell).
- 1977: Programação n-versão (Avizienis e Chen).
- 1980: criado o IFIP WG 10.4 Dependable Computing and Fault Tolerance.

Histórico (+/-)

- 1982: sete position papers no FTCS-12 com conceitos e terminologias.
- 1985: síntese dos conceitos e terminologias por Laprie.
- 1990: artigo sobre TFSD no FTCS.
- 1992: livro Dependability: Basic Concepts and Terminology (Laprie, Spring-Verlag).
- 1994: livro Fault Tolerance in Distributed System (Jalote, Prentice Hall)

Falhas são relativas à tarefas

- Tarefas são trechos de código ou subrotinas para fornecer uma funcionalidade específica.
- Podem ser classificadas quanto sua prioridade
  - Aperiódicas: podem ser disparadas a qualquer momento.
  - Esporádicas: podem ocorrer a qualquer momento, porém o tempo mínimo entre as ativações é conhecido.
  - Periódicas: intervalo entre disparos é conhecido e fixo.

Falhas são relativas à tarefas

- Tarefas são trechos de código ou subrotinas para fornecer uma funcionalidade específica.
- Podem ser classificadas quanto sua ocorrência.
  - Não-Críticas: atrasos e falhas são toleráveis.
  - Semi-Críticas: atrasos são toleráveis, mas falhas não.
  - Críticas: falhas ou atrasos não são toleráveis.

Falhas são relativas à tarefas

- Falha: problema ou imperfeição no nível físico.
  - Curto circuito;
  - falhas algorítmicas;
- Podem ser classificadas em:
  - Transitórias: ficam ativas por um certo período de tempo.
  - Intermitentes: faltas transitórias ativas periodicamente.
  - Permanentes: após ocorrer, permanecem ativas até correção.

Ocorrência de falhas (f/m)

Automotive Embedded System Component	Failure Rate λ
Military Microprocessor	0.022
Typical Automotive Microprocessor	0.12
Electric Motor Lead/Acid battery	16.9
Oil Pump	37.3
Automotive Wiring Harness (luxury)	775

Erro e defeito

- Erro: desvio na exatidão ou precisão da computação. Ocorrem no nível computacional. Associados a valores incorretos do estado do sistema. São causados por falhas.
- Defeito:
  - sistema não corresponde ao esperado. É um desvio da especificação
  - Também é caracterizado quando serviços não são fornecidos da forma ou prazo esperados pelo usuário

Tudo isso nos leva a...

- Dependabilidade
  - Uma propriedade de um sistema computacional, tal como usabilidade, desempenho e custo.
  - Dependabilidade diz respeito a habilidade de entregar um serviço comprovadamente confiável, ou seja, habilidade do sistema para evitar defeitos inaceitáveis para seus usuários.

Dependabilidade de um sistema

- Disponibilidade: diz respeito a média de tempo disponível para acesso.
- Confiabilidade: diz respeito a continuidade da entrega de serviço correto.
- Integridade: impedimento de alterações de estado impróprias.
- Segurança (safety): diz respeito a garantias de não haver defeitos catastróficos ao usuário ou ambiente.

Dependabilidade de um sistema

- Confidencialidade: impedimento de acesso indevido.
- Mantenabilidade: habilidade para reparo e modificações eficientes.
- Segurança (security): proteção contra acessos, ou controle, não autorizados ao estado do sistema.
- Testabilidade: facilidade para testar o sistema (ponto de teste, testes automatizados)

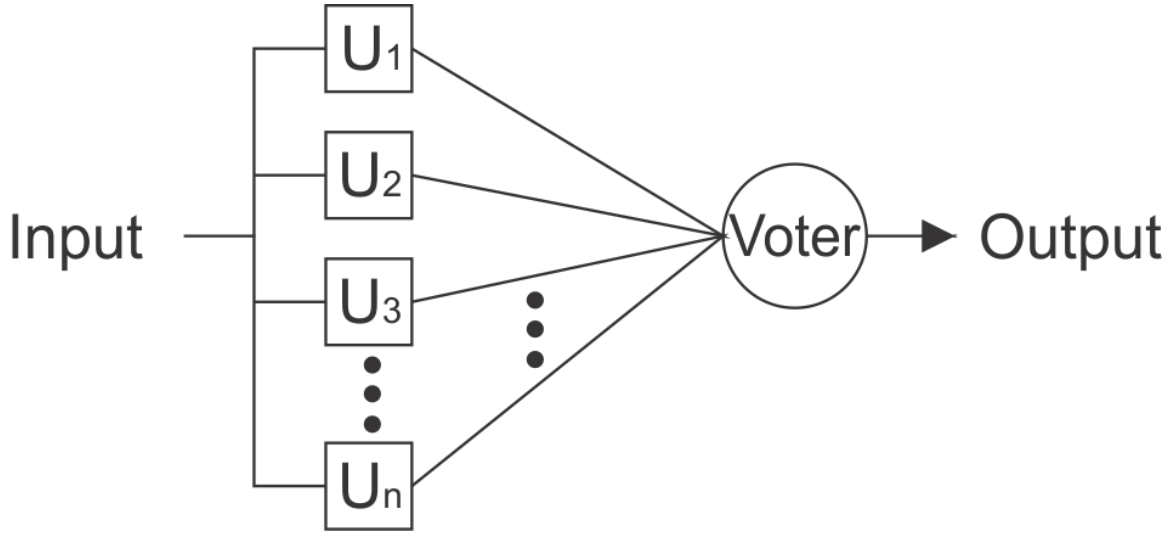
Dependabilidade de um sistema

- Sua obtenção tem base
  - Prevenção de Falhas: visa prevenir a ocorrência ou introdução de falhas.
  - Remoção de Falhas: visa reduzir o número ou a severidade das falhas.
  - Previsão de Falhas: visa estimar o número presente, a incidência futura e as consequências das falhas.
  - Tolerância a Falhas: visa entregar o serviço correto mesmo na presença de falhas.

Técnicas de Redundância

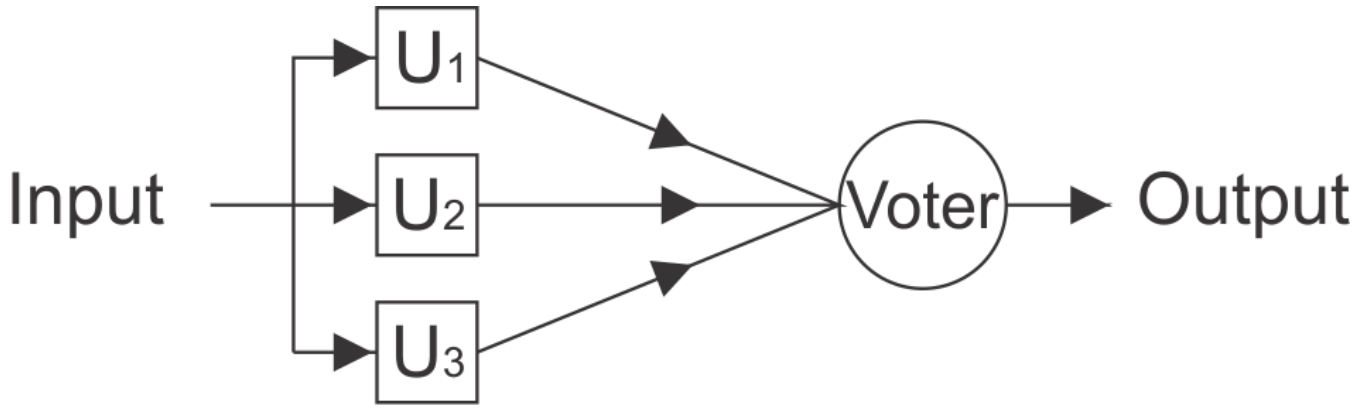
NMR (N-Modular Redundancy)

- Essa técnica consiste em replicar o hardware utilizado para processamento das informações em N módulos paralelos ligados a um sistema de votação



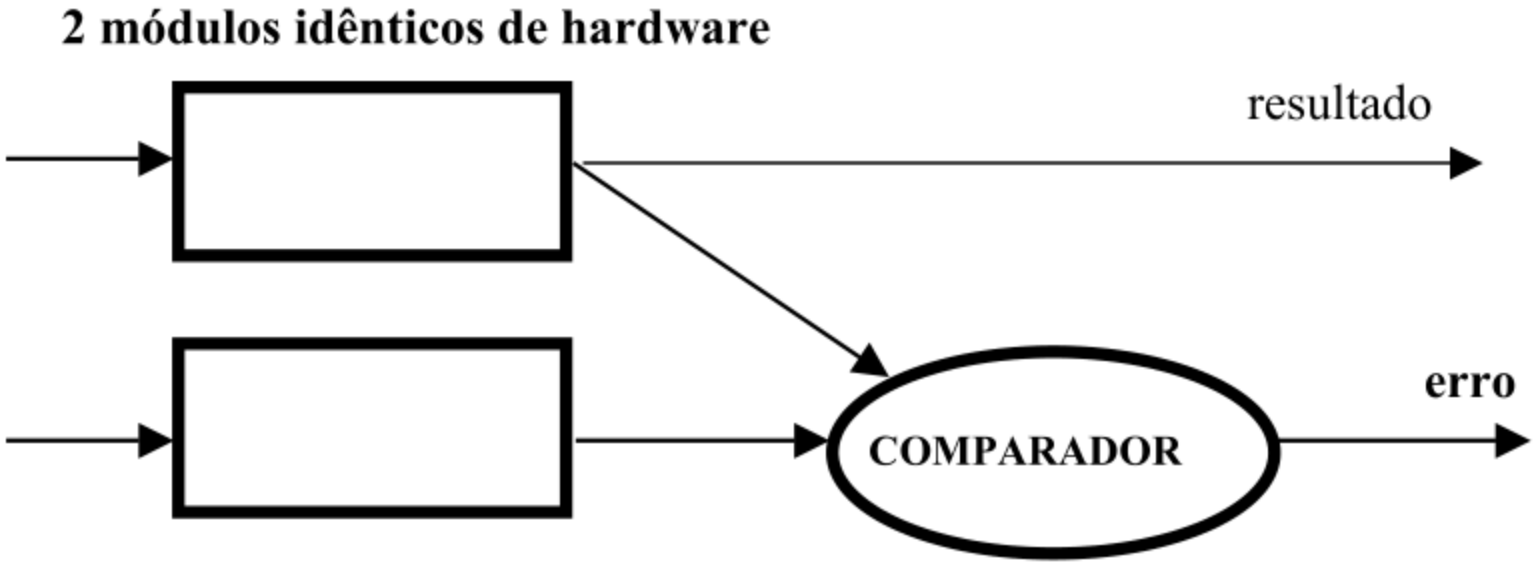
TMR (Triple Modular Redundancy)

- é uma versão do NMR com 3 módulos de hardware redundantes com o uso de sistema de votação[1] ou não



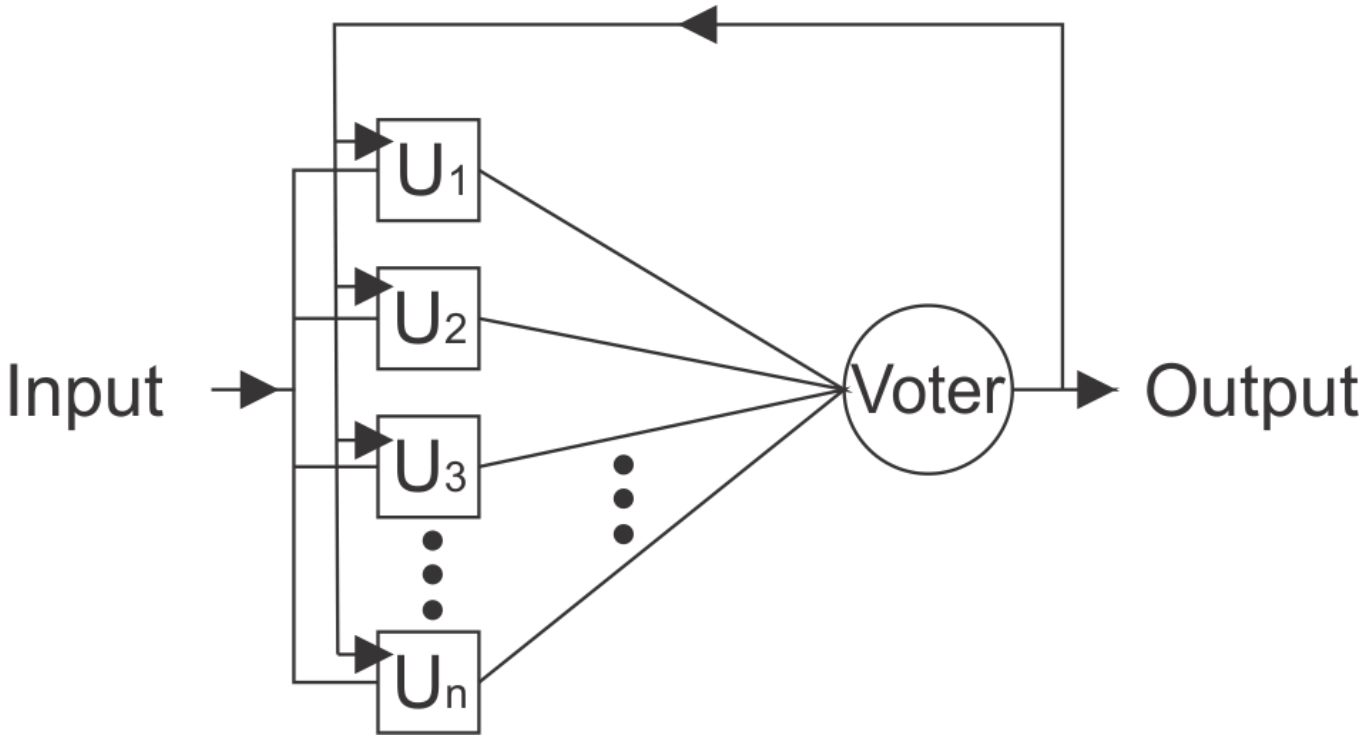
2MR (2-Modular Redundancy)

- Nesta, o hardware é apenas duplicado e através de comparação entre as saídas é possível evitar que saídas com falha sejam propagadas e gerem defeitos no sistema



Flux Summing

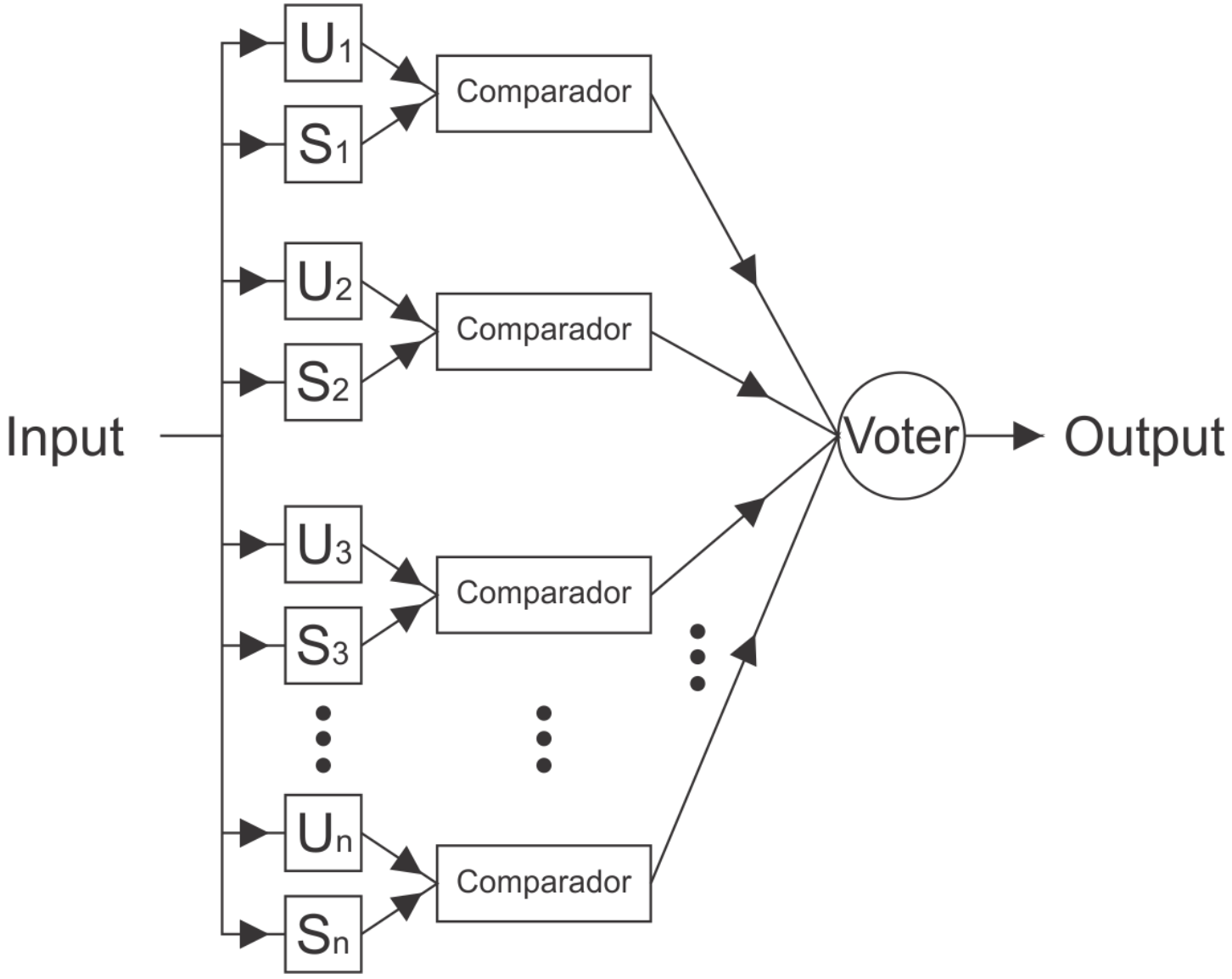
- baseia-se na correção dos sistemas de entrada à partir da saída mais votada do sistema, através de uma reentrada das saídas como entrada



Standby Sparing

- Também conhecida como Standby Replacement, é a técnica em que um ou mais módulos ficam em standby enquanto o módulo principal está operacional.
  - O standby sparing pode ser classificado em hot standby e cold standby.

Standby Sparing



Detecção de erros

Detecção de erros

- Como as falhas e avarias não podem ser detectadas diretamente, elas serão deduzidas a partir da detecção de erros no sistema.
- Assim, testes deverão ser executados para verificar suas ocorrências, a fim de dar-lhes o tratamento adequado.

Detecção de erros - testes

- Um teste ideal deve se basear somente na especificação do sistema, e não deve ser influenciado pelo seu design interno.
- Um teste ideal deve ser completo e correto, isto é, todos os possíveis erros projetados a serem verificados devem ser detectados, e nenhum erro deve ser declarado quando não existente.

Detecção de erros - testes

- O teste deve ser independente do sistema com respeito à suscetibilidade de falhas. Testes também podem falhar, e deseja-se que suas falhas não sejam relacionadas com falhas no sistema que está sendo verificado.

Testes de Replicação

- Testes de Replicação são muito comuns e poderosos, podendo ser bem completos e implementados sem o conhecimento do funcionamento interno do sistema.
- Tal teste implica em replicar algum componente do sistema, e comparar ou votar resultados de diferentes componentes a fim de detectar erros.
- O tipo e a quantidade de replicações dependem da aplicação.

Testes de Temporização

- Se a especificação de um componente inclui restrições no tempo de resposta, então testes de Timing podem ser aplicados.
- Basicamente, tais testes realizam uma solicitação a algum componente e verificam se o tempo de resposta excede ou não a restrição imposta na especificação.

Testes Estruturais e Semânticos

- Testes Semânticos tentam garantir se o valor é consistente com o resto do sistema.
- Testes Estruturais só consideram a informação e garantem que internamente a estrutura dos dados é como deveria ser.
  - A forma mais comum de teste estrutural é a codificação, que é usada intensamente em hardware.

Testes de Consistência

- Essa técnica consiste em realizar verificações em determinados pontos da computação, testando a consistência, ou seja, se os invariantes continuam sendo respeitados.

Testes de Diagnóstico

- A partir do conhecimento prévio de certos valores de entrada e de seus resultados de saída corretos, estes valores são aplicados ao componente e a saída é comparada com os resultados corretos.

Teste de Capacidade (Capability check)

- Essa técnica consiste em verificar a capacidade do sistema antes da execução de alguma tarefa
- Ou simplesmente utilizar o tempo livre do processador para verificar o funcionamento dos componentes do sistema.

Tratamento de Falhas

Localização da Falha

- Nesta fase, o componente defeituoso precisa ser identificado. Se não for possível sua localização, não será possível reparar o sistema para que o erro não ocorra novamente.
- Tipicamente, após detectar o erro, o componente defeituoso é identificado como sendo aquele mais próximo da origem do erro.

Reparo do Sistema

- Nesta fase, o sistema é reparado para que o componente defeituoso não seja usado novamente.
- Deve ser notado que a manutenção é feita on-line, sem intervenção manual.
- O reparo é feito por um sistema de reconfiguração dinâmica, tal que a redundância presente no sistema distribuído é usada para substituir o componente defeituoso.

Recuperação de erros

Recuperação para trás ou por retorno (Backward Recovery)

- Neste modelo, o estado do sistema é retornado a um estado anterior, na esperança de que este novo estado esteja livre de erros.
- Para isso, checkpoints periódicos são estabelecidos em um repositório estável. Quando algum erro é detectado, o sistema sofre uma volta (rollback) para o último checkpoint.
- A principal desvantagem é o overhead necessário, pois além de ser necessário criar checkpoints frequentemente, o rollback envolve certa computação pelo sistema.

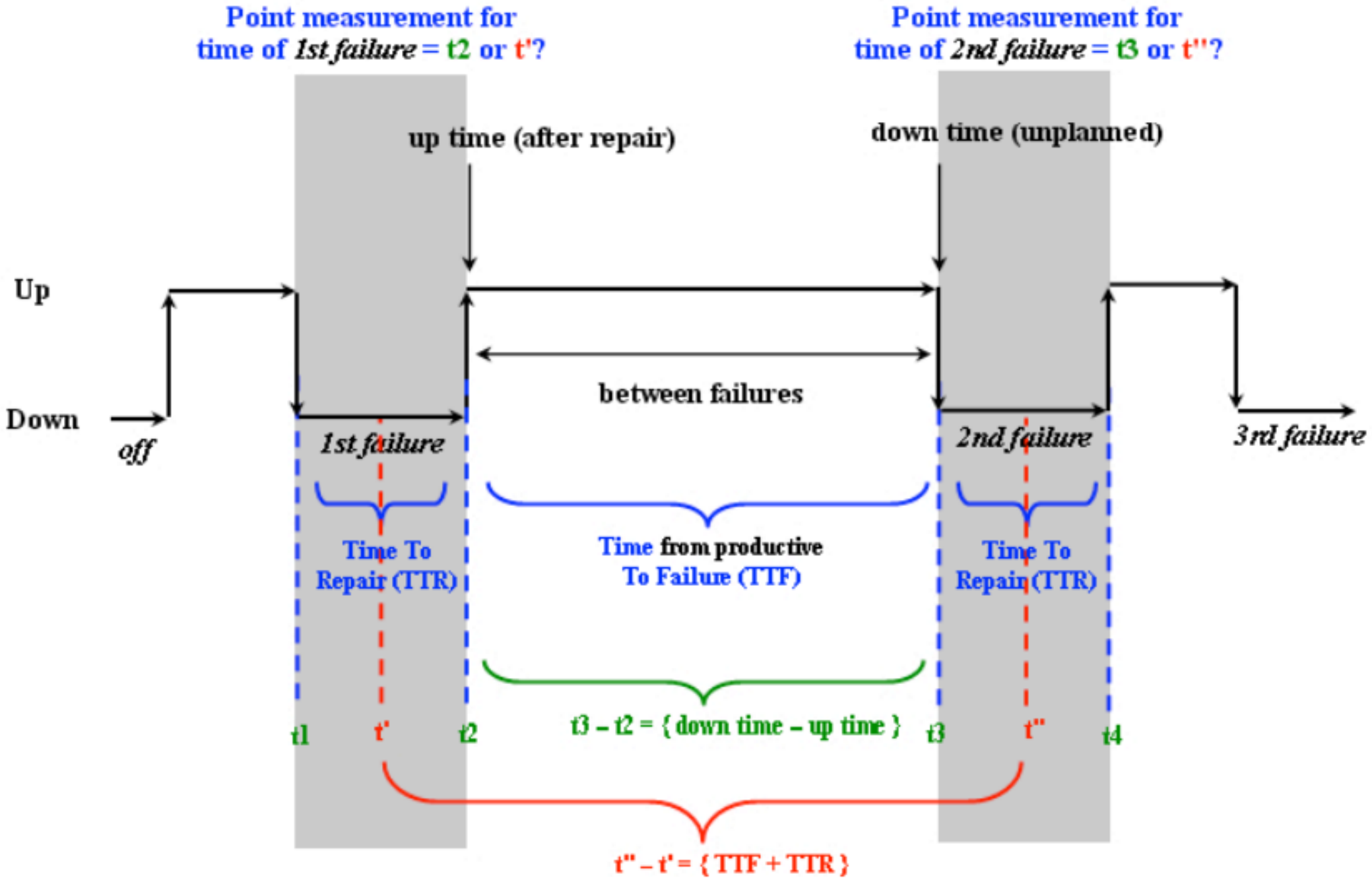
Recuperação para frente ou por avanço (Forward Recovery)

- Neste modelo, nenhum estado anterior está disponível. Ao contrário, é feita uma tentativa de se “ir para frente”, e tentar tornar o estado livre de erros aplicando-se medidas corretivas.
- Conceitualmente é interessante pois não há overhead. Entretanto, na prática se torna difícil, pois depende de uma avaliação e suposições precisas sobre o estrago.

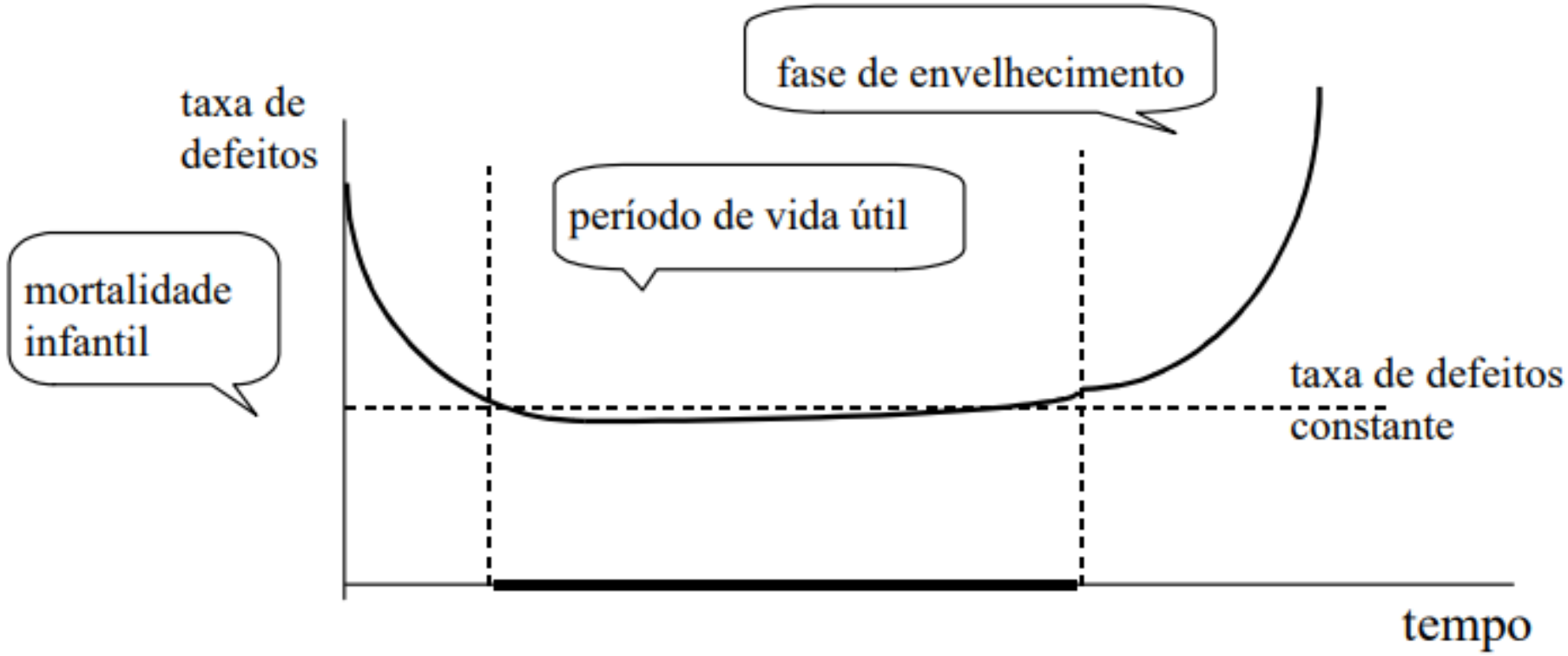
Medidas de Confiabilidade

Medidas de confiabilidade

Taxa de defeitos - failure rate, hazard function, hazard rate	número esperado de defeitos em um dado período de tempo, assumido um valor constante durante o tempo de vida útil do componente.
MTTF - mean time to failure	tempo esperado até a primeira ocorrência de defeito
MTTR - mean time to repair	tempo médio para reparo do sistema
MTBF - mean time between failure	tempo médio entre as falhas do sistema


Point measurement for
time of 1st failure = t2 or t'?
Point measurement for
time of 2nd failure = t3 or t''?
up time (after repair)
down time (unplanned)
Up
Down
off
between failures
1st failure
2nd failure
3rd failure
Time To
Repair (TTR)
Time from productive
To Failure (TTF)
Time To
Repair (TTR)
t3 – t2 = { down time – up time }
t1
t'
t2
t3
t''
t4
t'' – t' = { TTF + TTR }

Curva da banheira




Obrigado
Prof. Maurício Acconcia Dias
Microcontroladores II