Projeto de Circuitos Eletrônicos com Estatística

17 de junho de 1998

Autor
Mário Vaz da Silva Filho

Orientador
Prof. Dr. Alberto Martins Jorge

Universidade Estadual de Campinas - UNICAMP
Faculdade de Engenharia Elétrica e de Computação - FEEC
Departamento de Eletrônica e Microeletrônica -DEMIC

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas - FEEC / UNICAMP, como parte dos requisitos exigidos para a obtenção do título de DOUTOR EM ENGENHARIA ELÉTRICA.

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

Si38p

Silva Filho, Mário Vaz
Projeto de circuitos eletrônicos com estatística / Mário Vaz da Silva Filho.--Campinas, SP: [s.n.], 1998.

Orientadores: Alberto Martins Jorge.
Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Circuitos eletrônicos. 2. Estatística. 3. Controle de qualidade. 4. Confiabilidade (Engenharia) 5. Circuitos eletrônicos – Metodologia. 6. Simulação (Computadores) I. Jorge, Alberto Martins. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

ÍNDICE

Capítulo III : CPSPICE - SPICE Paralelo sob CPS

Capítulo IV - Problemas, Resultados e Propostas

Agradecimentos

A meus pais, Mário e Maria de Lourdes, meu amor e gratidão.

À Léa, Aluísio, Eduardo e Anamaria, pelo amor que cultivamos, que orienta minha vida, e pelo apoio dado em todos momentos desta tese, nas ausências, nos sacrifícios de férias e fins de semana, especialmente no último mês deste trabalho.

Aos amigos de construção do Laboratório de Eletrônica e Dispositivos – LED / UNICAMP, em especial a Alberto Martins Jorge, Wilmar Bueno de Moraes, Oseas Valente de Avilez, pelo incentivo e companheirismo na luta pelo desenvolvimento da tecnologia nacional, para o progresso do País e educação do seu povo.

Ao colega de construção do Laboratório de Projeto de Circuitos Integrados da UFRJ, Antônio Carneiro de Mesquita Filho, e aos alunos como Vladimir Castro Alves, que trabalharam em conjunto neste laboratório, na mesma luta do LED, pela Microeletrônica no Brasil.

A Alberto Santoro, pelo apoio e amizade, e pelo espaço de trabalho criado no Laboratório de Cosmologia e Física Experimental de Altas Energias do Centro Brasileiro de Pesquisas Físicas, que permitiu esta tese, bem como a todos os colegas que trabalharam por aquele laboratório, especialmente Jorge Amaral e Mariano Sumrell Miranda, pelas contribuições a esta tese.

Ao Rodrigo Natalizi, pelo bom trabalho feito no CPSPICE e também a Ricardo Fukui, Roberto Schendel, Jorge Sabino e Sidimar Quesada, pelos estudos sobre otimização e modelos, todos mais do que alunos, bons companheiros.

.

Aos amigos da biblioteca do Fermilab pela ajuda na pesquisa bibliográfica.

Aos colegas e alunos do DEL / UFRJ, que sempre me incentivaram e me mostraram a importância e o divertimento de ser um professor.

Minha admiração e gratidão a todos aqueles que me nortearam ao contribuir com seu trabalho e suas vidas para o Progresso do Brasil e seu Povo, em especial pela Educação deste. A eles dedico esta tese. E em contraponto declaro todo o meu ódio sincero àqueles que por motivos pessoais, por ignorância ou filosofia, trabalham contra o desenvolvimento do Brasil e a Educação do seu Povo.

A todos estas pessoas dedico esta tese.

RESUMO

Apresenta-se uma proposta de metodologia de projeto de circuitos eletrônicos, baseada no uso de programas de computador para simulação de circuitos e tratamento estatístico de dados, que busca garantir qualidade de produto no projeto. Se baseia em recursos computacionais da área de Física de Altas Energias, que possibilitam custo baixo, modularidade e adaptabilidade altas, e acesso livre ao software de projeto, possibilitando uma ampla gama de aplicações em ensino e desenvolvimento de projeto de circuitos. Isto é demonstrado no programa CPSPICE, criado para simulação de circuitos eletrônicos com estatística pelo método de Monte Carlo, para ser executado em computadores paralelos sob sistema operacional UNIX em comunicação segundo protocolo TCP/ IP. Utiliza as bibliotecas CPS - Software para Processos Cooperativos, para tornar paralelos e cooperativos processos independentes, e CERNLIB, para a análise estatística de dados. E os programas SPICE, para a simulação de circuitos eletrônicos e PAW - Physics Analysis Workstation, para análise e visualização gráfica dos resultados da simulação por Monte Carlo.

Os resultados das simulações pelo SPICE são escritos em disco ou fitas magnéticas como arquivos de n-tuplas, para serem processados pelo programa HBOOK em modo "batch" ou interativamente pelo programa PAW -- Physics Analysis Workstation. Estes programas também permitem a documentação do projeto com alta qualidade gráfica.

ABSTRACT

This thesis presents a circuit design methodology aimed to guarantee at design level the quality of production in a well-controlled environment. It is based on computer programs for circuit simulation and statistical data processing. The use of computational resources from High Energy Physics - HEP allows low cost, high modularity and adaptability, and free access to the software developed, opening a broad range of applications in teaching and circuit design. This is demonstrated with CPSPICE, our electronic circuit statistical simulation program developed to run in a parallel environment under UNIX operating system and TCP/IP communications protocol using CPS - Cooperative Processes Software. Also used are SPICE, an electronic circuit simulation program and CERNLIB software package, including PAW - Physics Analysis Workstation.

CPSPICE simulates electronic circuits by Monte Carlo method, through several different processes running simultaneously SPICE in UNIX parallel computers or workstation farms. Data transfer between CPS processes for a modified version of SPICE2G6 is done by RAM memory, but can also be done through hard disk or tape files if no source files are available for the simulator, and for bigger simulation output files.

Simulation results are written in a HBOOK file as n-tuples, to be examined by programs like HBOOK in batch mode or interactively by PAW -- Physics Analysis Workstation. The results can be visualized through histograms and graphics, and analyzed by statistical procedures available. HBOOK file can be stored on hard disks for small amount of data, or into magnetic tapes for large amount of data. These programs provide also resources to document the design with a high graphic quality.

Introdução

Nesta tese se apresenta uma proposta de metodologia de projeto de circuitos eletrônicos com garantia de qualidade no sentido dado em [Taguchi], que é o de controlar por projeto variações das características dos circuitos estritamente dentro de especificações dadas. Esta proposta não está limitada à eletrônica tal como é aqui aplicada no projeto de circuitos analógicos, podendo servir de base para o desenvolvimento de programas de computador para auxílio a projetos em outras áreas [Vaz95].

A garantia de qualidade do projeto depende do controle que é possível ter, nesta etapa anterior à produção, das variações das características dos circuitos, resultantes da variabilidade inerente aos componentes eletrônicos [Divekard, Dutton, El, Herkowitz, IEE82, IEEE86, INTEL, Khalily, Kurker, Nassif, Pfiester, Rankin, TEXAS]. Este problema pode ser tratado se for possível estabelecer modelos estatísticos para as variações dos componentes [Beck, Getreu, Klassen, Meijer, Oehm, Spence, Styblinski, Strojwas, Yang]. Com base nestes modelos é possível obter modelos estatísticos do comportamento do circuito, para tomada de decisões, quanto à especificação de componentes adequados aos objetivos alvo do projeto. Estes objetivos estão relacionados com fatores tais como desempenho do circuito, custo, rendimento ou perda de produção [Balaban, Becker, Bell, Spence, Taguchi]. Este problema é tratado nesta tese, em termos mais gerais, com a adoção de métodos de Monte Carlo [Hammersley, Rubinstein, Spence].

A escolha das ferramentas computacionais para auxílio a projeto de circuitos com estatística, se baseou na adequação destas para fins de ensino e pesquisa em universidades e centros de desenvolvimento tecnológico [Mead, Natalizi, Schetchman, Vaz95]. O software utilizado foi na sua totalidade não comercial, oriundo de universidades como a da Califórnia em Berkeley, ou centros de pesquisa com o Centre Européene de Recherches Nucleares – CERN, e o Fermi National Accelerator Laboratory – Fermilab. O objetivo foi o de permitir uso e desenvolvimento de programas de auxílio a projeto com garantia de qualidade, dentro de universidades e centros de pesquisa, de qualquer parte do mundo que não disponham de recursos para adquirir software comercial, como por exemplo o caso de professores e pesquisadores brasileiros no momento atual. Por este motivo, cabe ressaltar que todo o software aqui utilizado e desenvolvido é público e gratuito, devido ao fato de ter sido desenvolvido em instituições públicas. Estes programas são tanto de conhecimento quanto de uso público, e acessíveis via Internet.

Em resumo, a metodologia de projeto de circuitos com estatística aqui proposta consiste na aplicação de métodos de Monte Carlo, através do uso de ferramentas computacionais desenvolvidas para a Física de Altas Energias [Brun, CERN_, CPS_, D0_, Eadie, Nash], de domínio público e de fácil aquisição e manutenção, acessível a universidades e centros de pesquisa em geral. Destina-se a alunos, professores e pesquisadores que queiram desenvolver projetos de circuitos com garantia de qualidade, bem como aprender e desenvolver métodos estatísticos de projeto. Aqui se apresentam estas ferramentas e seu uso no desenvolvimento do programa CPSPICE para simulação de circuitos eletrônicos analógicos pelo método de Monte Carlo, utilizando o simulador SPICE [Cohen, Nagel, Vinci].

O método de Monte Carlo consiste na integração numérica de resultados de simulação de eventos estatísticos, criados através da geração de números aleatórios com distribuições estatísticas bem estabelecidas. O método é robusto no sentido de ter convergência garantida

estatisticamente para um resultado coerente com os modelos adotados. A precisão do método depende da precisão dos modelos e do número de eventos simulados. O método usualmente exige grande número de simulações, que no caso de circuitos eletrônicos requerem muito tempo de execução, mesmo em computadores de alto desempenho. [Hammersley, Rubinstein]

Para reduzir o tempo de execução das simulações é proposto o uso de computação paralela de granulação grossa, onde as amostras do circuito analisado são simuladas por processos paralelos independentes, em computadores conectados via protocolo TCP/IP, sob sistema operacional UNIX ou por "farms" [Nash] de computadores. O paralelismo é conseguido com o auxílio da biblioteca de programas CPS - Software de Processos Cooperativos [CPS_, Fausey, Nash, Wolbers], desenvolvido em colaboração LAFEX / CBPF com o FERMILAB, similar ao PVM [PVM]. Atualmente recursos como o "thread " do UNIX [Computer95, Unix, White] ou programas como o Client Server [D0_RJS], desenvolvido no LAFEX, permitem esta paralelização com vantagens tais como maior alcance, eficiência e segurança.

O volume de dados gerados neste processo é relativamente grande, crescendo com o tamanho dos circuitos, o número de variáveis envolvidas e a complexidade das especificações a serem verificadas. Para processar e armazenar tais dados, calcular e visualizar as estatísticas, existem programas como o PAW - Physics Analysis Wokstation [Brun, CERN_PAW] e HBOOK [CERN_HBO], bem como equipamentos tais como discos rígidos e fitas de vídeo de 8mm, ambos com capacidades de até 10 G bytes [Nash, Stith].

Nesta tese estas idéias foram aplicadas no desenvolvimento de um programa de simulação por Monte Carlo de circuitos eletrônicos denominado CPSPICE, direcionado para otimização de projeto de circuitos eletrônicos necessários aos novos detetores de partículas que estão sendo construídos para o estudo da Física de Altas Energias no Fermilab [D)_Upg, D0_NIM]. Desta forma se cria uma parceria entre as áreas de Física e Engenharia Eletrônica, que se espera que vá contribuir para a ampliação deste trabalho.

Este trabalho está organizado da seguinte forma: no capítulo 1 serão apresentados os conceitos básicos e as diretrizes deste trabalho. O capítulo 2 trata, dentro de um enfoque estatístico, dos temas relacionados com o projeto por computador de circuitos eletrônicos: análise, síntese, modelagem e otimização. São utilizados para isto exemplos simples advindos de aplicações diferentes, usando métodos diferentes. O capítulo 3 apresenta o CPSPICE, programa desenvolvido para demonstrar as aplicações da metodologia proposta para o projeto de circuitos, detalhando seus componentes e sua implementação, como possível referência para outros desenvolvimentos de simuladores de Monte Carlo, segundo a metodologia de projeto aqui proposta. O capítulo 4 contém os resultados e possíveis desdobramentos deste trabalho, e as conclusões.

Capítulo I - Conceitos Básicos

1.1 Abrangência deste trabalho

O poder e o alcance das ações humanas vem sendo continuamente ampliados através do uso de **máquinas**, sistemas cuja complexidade e capacidade crescem com o aperfeiçoamento de seus **circuitos eletrônicos**, componentes básicos para o processamento (aquisição, memorização, transformação e transmissão) da **informação**. Os atributos destes circuitos estão relacionados à capacidade de processarem a informação. Esta capacidade vem crescendo exponencialmente no tempo, os circuitos vem se tornando sistemas cada vez mais complexos, de projeto cada vez mais difícil e dispendioso.

Circuitos eletrônicos são sistemas que relacionam informação a potenciais e correntes elétricos, como resultado do confinamento de ondas eletromagnéticas, delimitadas no tempo e no espaço dentro dos componentes eletrônicos, para poderem representar a informação sob forma de **sinais elétricos** individualizados [capítulos 1 e 2 de Brenner, capítulo 1 de Desoer, capítulos 1 e 5 de Ramo]. A **análise** e a **síntese** de circuitos eletrônicos, normalmente assumidos como não irradiantes, fisicamente realizáveis, causais, se baseiam em leis como as de Ohm, Kirchhoff e Telegen [capítulos 1 e 2 de Brenner, capítulos 1 à 5 de Chua&Lin, capítulo 1 de Desoer] e métodos numéricos [Albrecht, Beck, Biles, Blum, Press].

A associação da informação a elétrons, como cargas elétricas armazenadas ou fluindo, dá o caráter **discreto** à eletrônica, porém aumenta em muito a complexidade e o custo da memorização e processamento de informações de caráter distribuído, tal como imagens em movimento. Para este tipo de aplicações, principalmente para aumentar a velocidade de transmissão e processamento, e a densidade de armazenamento, está se integrando a eletrônica à **fotônica**, o que eventualmente poderá permitir tratar a informação na sua forma distribuída, associada diretamente à ondas eletromagnéticas. Entretanto a eletrônica a **parâmetros concentrados**, feita com base em componentes discretos ou integrados, apresenta um espectro de aplicações tão amplo que não se deve esperar uma perda sensível de sua importância em um futuro próximo.

Nesta tese se propõe uma **metodologia de projeto com garantia de qualidade** no sentido dado por **Taguchi**, que é de **controlar especificamente a variação das características dos produtos, através do controle das variações dos componentes** [Taguchi]. A proposta, aqui aplicada no projeto de circuitos analógicos, e demonstrada no desenvolvimento do **CPSPICE**, ferramenta desenvolvida para auxiliar o projeto de circuitos eletrônicos com estatística, não está limitada à eletrônica como tal e pode servir de base para desenvolvimento de outros **programas de auxílio por computador a projeto** em outras áreas da engenharia.

A garantia de **qualidade** do projeto depende do **controle** que é possível ter, nesta etapa anterior à produção, das variações das características dos circuitos relacionadas com a variabilidade inerente aos componentes eletrônicos [Divekard, Dutton, El, Herkowitz, IEE82, IEEE86, INTEL, Khalily, Kurker, Nassif, Pfiester, Rankin, TEXAS]. Este problema pode ser tratado, se for possível **controlar a qualidade dos componentes e estabelecer modelos estatísticos para as variações dos destes** [Beck, Getreu, Klassen, Meijer, Oehm, Spence,

Styblinski, Strojwas, Yang]. Então, com base nestes modelos, **analisar os circuitos** para obter **modelos estatísticos comportamentais**, para **tomada de decisões** quanto à especificação de componentes adequados aos **objetivos-alvo do projeto**. Estes objetivos se relacionam a fatores como **desempenho do circuito, custo, rendimento ou perda de produção** [Balaban, Becker, Bell, Spence, Taguchi]. Este problema é tratado nesta tese em termos os mais amplos e gerais, com a adoção de **simulações de Monte Carlo** [Hammersley, Rubinstein, Spence] e **ferramentas de auxílio a projeto com estatística**, adequadas às **condições de trabalho** em universidades e outras instituições de ensino e pesquisa com recursos computacionais limitados.

1.2 Objetivos

Os **componentes** dos circuitos eletrônicos tem seus **atributos** relacionados à capacidade de **armazenar e fazer fluir cargas elétricas, fluxos magnéticos, potenciais** e **correntes elétricas** [Brenner, Desoer, Ramo]. Eles tem se tornado cada vez mais complexos ao integrar múltiplas funções [El, Mead, TEXAS], verdadeiros **micro sistemas** criados com base em padrões de funcionalidade e limites tecnológicos dos processos de fabricação em cristais semicondutores [Colclaser, Glaser, Till]. O **projeto** destes sistemas apresenta **dificuldades** crescentes de execução resultantes de fatores como :

1. grande número de componentes,
2. alta complexidade dos modelos destes componentes,
3. utilização de recursos no limite da tecnologia, de controle impreciso,
4. imprecisão dos modelos e incertezas nos resultados de análises e sínteses,
5. especificações elétricas múltiplas e conflitantes, que exigem soluções de compromisso,
6. redução de prazos para a conclusão de projeto e amadurecimento do produto,
7. redução de custos de projeto e de produção,
8. atendimento à padrões de funcionalidade e qualidade do mercado internacional.

O avanço tecnológico permite a realização de circuitos e sistemas cada vez mais complexos, mas exige o aperfeiçoamento contínuo das **metodologias de projeto**, para acompanhar o aumento da complexidade e a dinâmica dos **processos de fabricação**. Este aperfeiçoamento metodológico inclui a criação das **ferramentas computacionais para auxílio a projeto**, para permitir ao projetista aumentar sua **produtividade** em termos de uma maior **rapidez e qualidade** de projeto, reduzir **custos de produção** e ao mesmo tempo resolver os problemas de projeto acima listados.Estes recursos são vitais para projetistas em países como o Brasil, face ao mercado de trabalho hoje internacionalizado em grande escala.

Daí surge o **objetivo principal desta tese**: apresentar uma **metodologia de projeto de circuitos com estatística**, baseada na aplicação de **métodos de Monte Carlo** e em **ferramentas computacionais** desenvolvidas com base em recursos tecnológicos da **Física de Altas Energias**, de forma a serem acessíveis a universidades e centros de pesquisa em geral, e portanto de domínio público e de fácil aquisição e manutenção por alunos, professores e pesquisadores, para que estes possam desenvolver projetos de circuitos com garantia de qualidade, aprender e desenvolver métodos estatísticos de projeto, como uma forma de projeto mais completa e efetiva. Aqui se apresentam estas ferramentas, seu uso e utilidade no desenvolvimento de um programa de simulação pelo método de Monte Carlo de circuitos eletrônicos analógicos baseado no programa SPICE.

1.3 O problema da tolerância em circuitos eletrônicos

O **projeto de circuitos eletrônicos** tem sido ensinado e realizado quase que exclusivamente de **forma determinística**, na qual se assume que os valores das variáveis e dos parâmetros dos circuitos são únicos e não variam no tempo ou de um circuito para outro [Chua, Desoer, Orsini, Ruheli]. Mas esta não é a realidade dos circuitos fabricados em série, ou mesmo protótipos construídos com componentes discretos produzidos em massa e não selecionados [Balaban, Becker, Bell, Spence]. O mesmo acontece, mas de forma diferente, com circuitos integrados em larga escala [Bhattacharyya, Inohira, Maly, Michael, Shah, Shyu].

Em qualquer destes casos, cada unidade produzida pode apresentar **características diferentes do especificado**, caso alguns de seus componentes venham a ter valores sensivelmente diferentes dos **valores nominais** estabelecidos no projeto, devido às **variações nos processos de fabricação** não incluídas nos **modelos**. Em conseqüência disto pode haver **uma degradação de desempenho** em uma parcela dos circuitos construídos, e mesmo uma fração não desprezível de unidades pode **deixar de atender às especificações**. O resultado pode ser tanto uma **redução da qualidade** como um todo, quanto da **quantidade de unidades funcionais produzidas**. Em conseqüência disto, de um modo geral, se tem um aumento de **perda ou de custo do produto**, também colocado na literatura como uma redução do **lucro ou do rendimento de produção** [Becker, Opalski, Spence, Strojwas, Styblinski].

1.4 O problema da confiabilidade

Associado a este problema de rendimento de produção, existe também o problema da **confiabilidade** dos circuitos que saem de fábrica funcionando dentro das especificações, mas **com o tempo** deixam de fazê-lo devido à alterações, também aleatórias, por **envelhecimento** ou uso que seus componentes possam ter, incluindo falhas súbitas [Becker, Bowles, Kim]. Tudo isso implica na necessidade de **manutenção corretiva ou preventiva** e portanto também em aumento de custo do produto. Uma solução comum adotada para resolver este problema é **testar** as unidades fabricadas, e **consertar** as que falharem no atendimento às especificações. Isto pode ser razoável para pequenos volumes de produção ou produtos especiais. Outra solução igualmente comum, que atende a casos de maior volume de produção, é projetar os circuitos com componentes de **ajuste**, estabelecendo técnicas de **aferição** que permitam que todas as unidades atendam às especificações.

Entretanto tem crescido o número de aplicações que exigem **maior garantia** de funcionamento correto dos circuitos, como por exemplo, próteses em medicina, equipamentos de aviação, sondas espaciais. Nestes casos é necessário ter circuitos com **muito baixa probabilidade de falhar**, que só pode ser obtida com a seleção de componentes e técnicas de fabricação de **alta confiabilidade**, e em projetos baseados em métodos e modelos estatísticos bem estabelecidos.

Em qualquer caso, **confiabilidade** é implícita à **qualidade** dos circuitos, quando esta é associada a **repetibilidade** de características, com pequenas variações de circuito para circuito no **tempo de vida útil**. Deve-se observar que não importa que este tempo de vida útil seja o maior possível mas sim que cada unidade fabricada supere o valor mínimo

especificado. A razão disto é que a qualidade do circuito segundo Taguchi, é dada pelo custo de uso do circuito, o qual é relacionado ao atendimento às suas especificações ao longo de sua vida útil. A qualidade não é dada pelo potencial de desempenho do circuito, pois este está associado ao custo de sua fabricação, função de sua complexidade e de sua possível vida útil. Resumindo um circuito de **baixo custo** pode apresentar uma **qualidade superior** ao de um circuito de **alto custo**, caso o primeiro atenda à todas as suas especificações ao longo de sua vida útil, também especificada, enquanto que o segundo deixe de atender a um único ítem de suas especificações. Por outro lado, um circuito tem uma qualidade maior do que o outro, feito para a mesma aplicação, quando **leva mais tempo para falhar** ou se suas **falhas ocorrem em um número menor de unidades** comparativamente ao outro.

1.5 Uma proposta de metodologia de projeto de circuitos eletrônicos

Mesmo quando possíveis, as soluções de ajuste e aferição apresentadas na seção anterior geralmente encarecem a produção e quase sempre não se aplicam ao caso de **circuitos integrados em larga escala**. A solução efetiva dos problemas de qualidade e confiabilidade, de forma geral e econômica, é identificar e corrigir as causas das falhas nas etapas anteriores à produção. Isto nos conduz ao **projeto otimizado de circuitos,** e à criação de modelos mais completos para os componentes. Dentro deste enfoque a tolerância dos componentes e as variações permissíveis da resposta dos circuitos passam a ser representadas por variáveis adicionais àquelas representativas dos valores nominais, e métodos especiais de análise e síntese são usados ao longo do projeto para que os circuitos apresentem **desempenho e rendimento de produção ótimos** [Bell, Becker, Spence]. Estes métodos podem ser diferenciados segundo sua natureza em **métodos determinísticos ou estatísticos**.

Nos métodos determinísticos os **valores nominais** dos parâmetros são substituídos por **intervalos de valores** [Antognetti84, Ruheli]. Estes métodos geralmente partem do princípio de que o **rendimento de produção é 100%** e que as variações dos valores dos componentes são tão pequenas que podem ser relacionadas **linearmente** com as **perturbações** nas respostas dos circuitos. Isto facilita o cálculo da **sensibilidade** da resposta do circuito à variação de seus componentes, por exemplo usando **o método do sistema adjunto de equações do circuito** [Bordewijk, Brayton, Chua, Leung, Rohrer].

No caso de **grandes variações**, a análise de sensibilidade exige a solução numérica de sistemas de equações não lineares, feita por métodos iterativos que demandam um maior esforço computacional do que os métodos de solução direta. **Métodos iterativos** como os de Newton-Raphson e Gauss-Newton **não tem garantia de convergência** mesmo que a solução exista e seja única. Na existência de **soluções múltiplas**, o problema de encontrar a **solução ótima** ou mais adequada fica ainda mais difícil de se obter com estes métodos. [Albrecht, Beck, Biles, Blum, Press]

Somente **métodos estatísticos** podem resolver estes problemas [IEE82, IEEE86, Spence, Rubinstein]. Além disso estes métodos possuem uma característica importante, a de admitir que o **rendimento de produção,** para fins de otimização do projeto do circuito, possa ser inferior a 100%. O cálculo do rendimento de produção consiste na integração de um sistema de equações não lineares onde os limites de integração não são definidos explicitamente, e o número de variáveis pode ser grande. Este problema pode ser resolvido

de forma eficiente por **métodos de Monte Carlo**, métodos estatísticos cujo custo computacional em memória e tempo de CPU independe do número de variáveis [Hammerslein, Rubinstein]. Isto porque o método se baseia na **amostragem** do universo de possibilidades, e portanto sua **precisão** depende exclusivamente do **número de amostras** simuladas com sucesso. Daí a característica intrínseca do método de Monte Carlo, de **convergência não muito rápida** (proporcional à raiz quadrada do número de amostras) **mas robusta**, garantida por amostragem e modelagem estatisticamente adequadas.

Com o **cálculo do rendimento de produção** o projetista ganha elementos para poder relaxar a tolerância e ao mesmo tempo controlar o número de circuitos falhos resultantes, reduzindo duplamente o custo total da produção. Além disso, ao admitir rendimentos de produção inferiores a 100% e levar em conta a freqüência de ocorrência de circuitos falhos, os métodos estatísticos possibilitam o cálculo dos **valores** e das **tolerâncias**, **ótimos** para cada **componente crítico**, visando **maximizar desempenho e minimizar o custo total de produção** [Becker, Bell, IEEE86, IEE82, Spence].

Além disso, a teoria estatística de circuitos também permite o projeto de circuitos de processamento de **sinais aleatórios**, tendo como base simulações estatísticas [Lawson, Ziel]. São exemplos desses circuitos, para os quais a informação desejada é a distribuição estatística dos sinais que processam, os circuitos de filtragem adaptativa de sinais denominados **redes neuronais** (neural nets) [Furth, Gowda], os circuitos para tomadas de decisão ótima chamados **circuitos difusos** (fuzzy circuits) [Huertas, Ketner, Zadeh] e circuitos com junções supercondutoras tipo Josephson [Jaeckel]. Deve-se observar que estas aplicações tem hoje maior peso econômico que no passado, na medida em que os circuitos citados estão sendo aplicados intensivamente em diversas áreas da instrumentação eletrônica, em especial na área de **Física de Altas Energias**.

Mas talvez a aplicação mais importante do projeto estatístico se dá na **integração em larga escala** de circuitos, cujo mercado é cada vez mais amplo e mais competitivo, com maior **volume de produção**, menor **custo por unidade**, e também menor **tempo de obsolescência** de produtos [Colclaser, Glasser, INTEL, Mead, TEXAS, Til]. O projeto destes circuitos deve ser otimizado visando maximizar ao mesmo tempo o rendimento de produção, o desempenho do produto, e minimizar os prazos de conclusão de projeto e o tempo de estabilização da produção em série do circuito integrado [Antognetti84, IEE82].

Como os circuitos integrados **no estado da arte** são extremamente **sensíveis** à variações de processos, por terem componentes com **dimensões muito reduzidas**, essa otimização necessita de técnicas estatísticas que modelem e predigam os efeitos destas variações de modo realista e preciso. Além de atender esta finalidade, as técnicas de análise estatística são freqüentemente mais robustas e mais eficientes do que as de análise determinística, quando não são a única alternativa possível [Becker, Biles, IEE82, IEEE86, Spence].

1.6 Condições de implementação da proposta

As vantagens citadas da aplicação de métodos estatísticos no projeto de circuitos são conhecidas de longa data [Bell, IEE82, IEEE86]. Mas foram adotados em pequena escala, dadas as múltiplas dificuldades de aplicação, primeiro pelo **alto custo da modelagem**, a **falta**

de modelos e dados sobre os componentes, sejam discretos ou integrados, que são **informações sensíveis** do ponto de vista industrial. Segundo, pela **falta de ferramentas computacionais** disponíveis, sejam hardware ou software, para o **cálculo das estatísticas** dos circuitos.

A **oferta** no mercado internacional destes recursos tecnológicos de software e hardware de apoio ao projeto estatístico de circuitos eletrónicos tem recentemente crescido muito, disponibilizando grande **variedade de produtos a custos razoáveis**, tanto no que se refere a **processamento numérico**, **simulação estatística**, **tratamento e visualização** de grande volume de dados, quanto no que diz respeito à **automação** de medidas elétricas, eletrônicas e ópticas, devendo ser citados em particular :

- sistemas automáticos de medidas, modulares, controlados por computador,
- instrumentos de medidas equipados com recursos de processamento digital de sinais, inclusive osciloscópios, medem os parâmetros estatísticos de sinais, em tempo real ;
- computadores paralelos de alto desempenho a baixo custo, que aceleram em muito as simulações por Monte Carlo;
- pacotes de subrotinas para processamento estatístico de dados, tratamento e visualização de sinais estatísticos, baseados em padrões e protocolos comuns, que facilitam a montagem e a modificação de programas aplicativos.

Esta capacidade tecnológica é especialmente forte na área de **Física de Altas Energias**, cujos laboratórios registram e processam estatisticamente enormes volumes de dados. Nesta área também existe uma grande demanda por sistemas eletrônicos para **aquisição e processamento de sinais dos detetores**, sinais estes de natureza aleatória, dos quais se deseja conhecer a distribuição estatística [Nash]. Nos grandes detetores de partículas, em aceleradores como o Tevatron do FERMILAB e o LEP do CERN, estes sistemas eletrônicos tem que processar um grande volume de dados muito rapidamente [Wolbers]. Alguns circuitos além de velozes devem também ter dissipação de potência, massa e volume reduzidos, para não interferirem nos processos físicos em estudo. Estes requisitos são conflitantes do ponto de vista da otimização dos circuitos, e são geralmente de difícil solução. Há portanto nesta área um grande interesse no projeto e simulação em bases estatísticas de circuitos Existe ao mesmo tempo uma disponibilidade de hardware e software para processamento estatístico de dados [Brun, CERN_, Nash], que podem ser usados na criação de ferramentas de auxílio ao projeto e à simulação de circuitos eletrônicos com base em métodos estatísticos [Vaz95].

Por serem **instituições públicas** que recebem investimentos de diversas fontes e possuem grande volume de recursos, estes Centros de Pesquisa em Física de Altas Energias são parceiros ideais para as universidades e centros de pesquisa públicos hoje sob intensa pressão financeira mas ainda interessados no desenvolvimento de circuitos de alta complexidade e confiabilidade, e baixo custo, assim como no **desenvolvimento de metodologias e ferramentas de projeto** para tal fim, pelo fato de poderem fornecer tanto o mercado, com a demanda daquele tipo de circuitos, como oferecer soluções computacionais adequadas para auxílio ao projeto destes circuitos, de livre acesso e baixo custo.

1.7 Condições de mercado

Há hoje um interesse crescente no aperfeiçoamento do modo com que os circuitos eletrônicos vem sendo projetados, o que inclui a criação de ferramentas de auxilio a projeto que permitam aumentar a produtividade e reduzir os prazos de concepção e produção, enfim aumentar a qualidade de projeto. Este interesse surge da **globalização** forçada da economia mundial, preconizando que a **produção** seja feita em qualquer parte do mundo, desde que resulte em **menor custo** e eventualmente **maior qualidade**, e se torne possível com base em uma **abertura de mercados** irrestrita. A qualidade de produção é alcançada através de uma **automação** eficiente dos processos de **produção** e principalmente dos processos de **concepção**, que devem garantir perfeita aderência aos **padrões** do mercado internacional e serem realizados visando desempenho e custo ótimos.

Os países plenamente industrializados dominam esta sistemática de longa data, como mostra a **literatura** técnica sobre o projeto em bases estatísticas de sistemas eletrônicos de alta confiabilidade e de alto rendimento de produção [Bell, IEE82, IEEE86, Spence]. Por **razões** que por serem **políticas** e **culturais** devem ser tratadas em um outro trabalho, esta sistemática não foi adotada em países como o Brasil. As **razões técnicas** podem ser parcialmente identificadas com a falta de pesquisa e desenvolvimento tecnológico nesta área, a inexistência de um **parque industrial** amplamente produtivo, bem como na falta de computadores digitais de alto desempenho, tanto do hardware quanto do software necessários à estas aplicações nas universidades, centros de pesquisa e indústrias de menor porte.

Outra motivação para esta tese vem da possibilidade de países como o Brasil utilizarem tecnologia eletrônica, advinda de países industrializados, em diversas aplicações vitais para o seu desenvolvimento.. A grande disponibilidade de **módulos** integrados permite a realização de sistemas eletrônicos de alta complexidade e larga faixa de operação, que tornam possível a produção destes sistemas, em qualquer escala, em qualquer parte do mundo. Cabe aos engenheiros eletrônicos dos países tecnicamente subdesenvolvidos buscarem espaço no **mercado** interno e mesmo no mercado externo em termos de produtos que possam ser produzidos com base nestes módulos, dentro de um regime de concorrência na qual a aparente desvantagem em termos de recursos econômicos e técnicos possa ser superada pela criatividade na **concepção** de produtos destinados tanto ao mercado interno quanto ao externo, assim como pelo desenvolvimento e uso de diversas ferramentas que permitam o projeto destes sistemas com uma **garantia de qualidade**, isto é onde o projeto também seja **padronizado e automatizado** como as linhas de produção montadas para o mesmo fim.

1.8 Ferramentas e recursos disponíveis

O **desenvolvimento de ferramentas de projeto** de circuitos voltado para a finalidade apresentada na seção anterior é uma das atividades de cooperação entre o Laboratório de Projeto de Circuitos da UFRJ - LPC e o Laboratório de Cosmologia e Física Experimental de Altas Energias do Centro Brasileiro de Pesquisas Físicas - LAFEX. O principal recurso disponível consiste no conhecimento contido na **literatura técnica** especializada dos países industrializados, que remonta à década de 70, quando se iniciou o desenvolvimento acelerado

da Microeletrônica, mantida até os dias de hoje. Uma pequena amostra deste acervo pode ser vista nas referências bibliográficas deste trabalho.

Aproveitou-se também o software e hardware de **sistemas distribuídos** das "farm" de estações de trabalho tipo **RISC**, disponíveis no LAFEX, tanto pela sua adequação ao problema quanto pelo seu **baixo custo** em relação ao desempenho, adequado não só para equipar universidades e centros de pesquisa como também indústrias de pequeno ou médio porte [Nash88, Nash92, Stith, Wolbers]. Cabe ressaltar que tais ferramentas foram desenvolvidas e adquiridas dentro de uma cooperação internacional entre instituições públicas que recebem investimentos públicos de seus países. Por isso estas ferramentas são de conhecimento e uso público, ao contrário daquelas desenvolvidas dentro de empresas privadas, destinadas exclusivamente ao setor produtivo destas empresas quando apresentam características tecnológicas mais avançadas, e quando não ficam disponíveis é a um alto custo , proibitivo para universidades, centros de pesquisa e pequenas empresas.

O hardware necessário para simulação de Monte Carlo consiste nos equipamentos que permitem a execução simultânea de vários programas de simulação, para reduzir o tempo de execução dos mesmos, e o processamento e armazenamento de grande volume de dados:

- computadores paralelos ou "farm" de estações de trabalho para os quais o tempo de processamento de cada simulação realizada em paralelo com as demais é muito superior ao tempo de transferência de dados entre os processos paralelos que compõem a simulação como um todo [Computer85, Nash].
- redes de comunicação entre computadores protocolo TCP/IP, tanto local (Ethernet) quanto ampla (Internet) [Computer85].
- unidades de armazenamento de dados : conjuntos de leitura e escrita em discos rígidos e fitas de vídeo de 8 mm [Stith].
- terminais e impressoras gráficas de qualidade adequada para permitir a visualização de histogramas e gráficos de alta resolução.
- equipamentos de medidas automáticas com capacidade de processamento estatístico de sinais, incluindo osciloscópios digitais, pontes de impedâncias e medidores diversos.

O software consiste nas seguintes bibliotecas e programas para simulação de circuitos, processamento estatístico e a visualização de grande volume de dados :

- CERNLIB - esta biblioteca de programas, desenvolvida no CERN, permite a geração de números aleatório, o processamento estatístico de dados e a visualização dos mesmos e a documentação do trabalho com alta qualidade gráfica.
- PAW - este programa permite a operação interativa dos programas da CERNLIB.
- CPS - esta biblioteca de subrotinas permite a troca de dados e mensagens entre processos independentes em uma ou mais máquinas UNIX, tornar cooperativos os programas de simulação. Foi desenvolvido no Fermilab em colaboração com o LAFEX.
- SPICE - este programa de simulação de circuitos eletrônicos com ênfase em circuitos integrados é um padrão internacional para uso de universidades, centros de pesquisa e indústrias em todo o mundo, tendo sido desenvolvido na Universidade da Califórnia em Berkeley.
- Programas comerciais para controle automático de equipamentos de medidas, com capacidade de processamento estatístico de sinais.

No trabalho cooperativo das universidades com os centros de pesquisa em Física de Altas Energias todo este acervo está disponível, em uso para a construção e operação dos grandes detetores de partículas. Porém a contribuição mais importante que veio desta área para esta tese consiste no hardware e software de computação paralela, que permite a simulação de grande número de amostras de eventos independentes exigido pelos métodos de Monte Carlo, e que será examinada a seguir.

1.9 Computação paralela

O **CPS - Software para Processos Cooperativos** - [CPS_R, CPS_U, Fausey, Nash91] é um pacote de ferramentas de **programação paralela**, que facilita a partição de uma tarefa computacional em um conjunto de **processos distribuídos** em um ou mais computadores com sistema operacional **UNIX**, ligados através de **protocolo TCP-IP**. A computação paralela adotada neste trabalho é do tipo de **granulação grossa**, onde o paralelismo se dá ao nível das rotinas ou programas, e não no nível de linhas do código como em máquinas vetoriais.

Por isso, o CPS é adequado a tarefas que gastam muito tempo de **uso de CPU** e pouco tempo de **transferência de dados entre CPU**. No caso do computador **ACP II** /6-8/, isto significa uma taxa de pelo menos 2000 bytes de instruções executadas por byte transferido. A simulação de circuitos eletrônicos com o **SPICE** se encaixa perfeitamente nesta condição.

O CPS, assim como seu similar, o programa PVM, surgiu antes do crescimento da computação paralela, quando os sistemas operacionais UNIX ainda não tinham recursos como o **"thread"**, para permitir **troca de mensagens e dados** entre processos independentes. Atualmente, com os modernos sistemas operacionais UNIX suportando computadores com múltiplas CPU e linguagens como JAVA possibilitando computação paralela através de redes locais e internacionais como a Internet, tornam dispensáveis programas como o **CPS**. Ele é apresentado neste trabalho como uma **referência** simples e efetiva para apresentação dos problemas de paralelismo em computação distribuída, com soluções simples a nível de **chamada de subrotinas**.

Uma proposta mais geral de computação paralela está sendo implementada no LAFEX com o programa **Cliente-Servidor** [D0_RJS]. É um programa que é executado de forma distribuída em computadores ligados via protocolo TCP/IP, que busca dentro de uma lista de computadores disponíveis, aqueles que estão inativos para **realizar remotamente tarefas computacionais** necessárias a uma simulação de Monte Carlo. Desta forma se consegue uma **utilização mais efetiva de máquinas** e um **maior poder de cálculo** com os **recursos disponíveis** em uma **colaboração internacional**.

No que diz respeito a hardware, os computadores paralelos **fracamente acoplados** são os candidatos mais adequados à simulação de Monte Carlo, por terem o **custo mais baixo** tanto de **aquisição** quanto de **manutenção**, e permitirem um aumento de capacidade computacional linear com o aumento do número de computadores envolvidos, desde que o tempo gasto na transmissão de dados seja bastante inferior ao tempo de processamento, que é o caso mais comum na simulação de circuitos eletrônicos. Este trabalho se iniciou com a vinda do computador **ACP II** [Nash] para o LAFEX, e se ampliou no

uso do computador **IBM SP2** do Laboratório Nacional de Cálculo Científico - LNCC. Ambos os computadores são da categoria acima mencionada, e em ambos foi possível instalar o CPS, o qual foi usado na implementação do CPSPICE. Como o hardware evolui muito rapidamente no tempo, dobrando de velocidade e quadruplicando o espaço de memória a cada tres anos, as estações de trabalho RISC utilizadas neste trabalho podem hoje ser substituídas com vantagens por placas de computadores pessoais de alto desempenho que possuam interface Ethernet. Como o sistema é distribuído, cada adição de máquina acrescenta poder de cálculo, e pode ser feita de forma independente, na medida em que existam recursos para aquisição, com **livre escolha** de fabricante e tipo, desde que a máquina possa ser conectada via protocolo TCP/IP e use UNIX.

1.10 CPSPICE : auxílio ao projeto de circuitos eletrônicos com estatística

O **projeto de circuitos e sistemas eletrônicos com estatística,** para ser realizado, precisa do auxílio de diversas ferramentas computacionais. Para introduzir tal metodologia nas universidades e centros de pesquisa, é necessário equipar as instituições com as ferramentas já citadas mas também e principalmente criar um ambiente de trabalho onde possam ser desenvolvidos os programas para auxílio a projeto de circuitos de interesse. Nesta tese o trabalho foi direcionado para otimização de projeto de circuitos eletrônicos necessários aos novos detetores de partículas que estão sendo construídos para o estudo da Física de Altas Energias no Fermilab [D)_Upg, D0_NIM]. Desta forma se cria uma parceria entre as áreas de Física e Engenharia Eletrônica, que se espera que vá contribuir para a ampliação deste trabalho.

Dentro desse objetivo, foi desenvolvido o **programa de análise estatística de circuitos eletrônicos CPSPICE** para multi computadores com sistema operacional **UNIX** [UNIX, White], baseado no programa de simulação de circuitos **SPICE** [Cohen, Nagel, Vaz89] e nas bibliotecas **CPS** - Software para Processos Cooperativos [Ávila, CPS_, Fausey, Miranda] e **CERNLIB** [Brun, CERN_}.

CPSPICE é constituído por 3 programas distintos que se comunicam e trocam dados quando **executados em paralelo sob controle do cps_jm**, o programa que administra o CPS. Um programa, o **CLASSE1**, lê as informações da simulação desejada, a descrição do circuito, os modelos estatísticos, e com base nestes gera arquivos com as amostras de circuitos e os comandos de simulação do SPICE. O segundo programa, **CLASSE2**, é executado em diversas cópias em paralelo para simular as amostras de circuitos independentemente, com base nos dados fornecidos pelo CLASSE1. O resultado das simulações das amostras é recebido pelo **CLASSE3**, que estrutura os dados de todas as simulações para análise estatística e visualização através de histogramas e outros gráficos.

A **visualização** dos dados, a geração de estatísticas dos circuitos e da documentação de projeto é feita posteriormente pelo programa **PAW**, com base no arquivo escrito pelo CLASSE3. O programa PAW é **interativo**, mas permite a criação de arquivos para execução em modo batch dos programas da CERNLIB. Com o PAW é possível fazer **filtragem de dados** e **análises estatísticas** diversas usando tanto rotinas numéricas padrões como programas feitos especificamente para uma tarefa de geração, armazenamento, seleção, transformação e visualização de grande volume de dados em bases estatísticas.

O **CPSPICE** simula circuitos eletrônicos pelo método de **Monte Carlo** [Hammerslein, Rubinstein] executando múltiplas cópias do programa **SPICE** em processos distintos em **computadores paralelos ou redes de estações de trabalho** com sistema operacional **UNIX**. A transferência de dados entre os diversos processos é feita pelo **CPS** de diferentes modos dependentes da versão do SPICE: por **memória** para a versão 2G6, que foi modificada para este fim, e por **arquivos em disco** para a versão 3F3, mantida no original. Este último esquema pode ser estendido a outros simuladores cujos programas fonte não sejam disponíveis ou pode ser usado quando for muito grande o volume de dados resultantes das simulações das amostras de circuitos.

A análise do resultado das simulações fornece a estatística dos parâmetros de projeto do circuito, que pode servir de base para **calcular e otimizar desempenho, rendimento ou perdas de produção**. Fornece também para o projetista uma **visão mais ampla** da **qualidade** do circuito, da **sensibilidade** à variações dos componentes e dos **modos de operação** dos circuitos. Finalmente permite observar as **o comportamento do simulador** quando utilizado em **circuitos padronizados**. A análise estatística é uma técnica robusta e versátil, e a principal ferramenta matemática para tratar de conjuntos ou coletivos de elementos, como são os circuitos quando projetados com garantia de qualidade, como amostras de um conjunto, as quais devem atender à especificações do conjunto.

Como o projeto de circuitos tem uma grande **diversidade de objetivos e métodos**, na medida em que a atual versão do CPSPICE, destinada ao projeto de circuitos analógicos, ofereça resultados positivos no auxílio a projetistas, outras versões deverão ser desenvolvidas para outros simuladores que atendem a outros objetivos como por exemplo a simulação e otimização de sistemas digitais. Outras áreas da engenharia poderão igualmente se beneficiar desta ferramenta. Por este motivo nesta tese é apresentado no capítulo 3 um detalhamento da implementação do CPSPICE, também descrito em [Natalizi, Vaz95], que para ser entendido necessita antes da compreensão do problema da variabilidade dos circuitos e sua solução com base em métodos estatísticos de projeto, apresentado no capítulo seguinte.

Capítulo II - O Projeto de Circuitos com Estatística

2.1 Introdução

A **produção** de equipamentos eletrônicos segundo as regras atuais do **mercado internacionalizado**, exige **cuidados adicionais de projeto** tais como maximização do **desempenho** do produto, aderência aos **padrões** internacionais, minimização de possibilidade de **perdas** de produção, redução dos **prazos** de desenvolvimento e produção e **garantia da qualidade** do produto entregue para consumo no sentido de reduzir custos no uso, seja por falha de operação ou por não atendimento às especificações [Taguchi].

A **qualidade** de um circuito eletrônico depende da qualidade e do grau da automação tanto da **produção** quanto do **projeto**. A qualidade do projeto por sua vez depende da **metodologia** adotada e das **ferramentas de trabalho** associadas a ela. A metodologia de projeto a ser adotada para este fim deve ser compatível com a **natureza** do problema, que é **estatística** [Spence, Becker]. A **complexidade** do projeto com garantia de qualidade, somada aos prazos reduzidos de projeto e produção que o mercado demanda para os circuitos eletrônicos atuais, exige o uso de ferramentas de **automação de projeto**. Para reduzir o tempo de **aprendizado** e ampliar a **aceitação** destas ferramentas é necessário que elas sejam uma **extensão** das ferramentas atuais, e ofereçam recursos mais avançados no que se refere ao **tratamento do problema estatístico** que é o **controle de qualidade**.

Este capítulo aborda este problema, definindo suas bases e oferecendo uma possível solução, que é o uso de ferramentas computacionais padronizadas pelo uso na **Física de Altas Energias** [CERN_LIB, Nash, Vaz95]. Estas ferramentas permitem resolver antigos problemas de projeto de circuitos eletrônicos revividos pela citada internacionalização de mercado, como a necessidade de:

- otimizar a razão **custo/desempenho** dos produtos feitos dentro de padrões de qualidade pré definidos, o que significa reduzir por projeto custos e perdas de produção;
- reduzir **prazos de desenvolvimento e produção**, bem como de aprendizado de projeto e circuitos;
- garantir a **qualidade do produto** entregue para consumo, no sentido dado em [Taguchi], isto é como garantia de prejuízos mínimos advindos de falha ou desvio sensível no atendimento às especificações durante o tempo de uso previsto para os equipamentos.

Este trabalho propõe como solução efetiva para tratar os problemas de **rendimento de produção e de confiabilidade**, identificar e corrigir as possíveis causas das falhas antes da produção, **otimizando o projeto** com base em **modelos** adequados. A causa dos problemas citados é a variabilidade dos componentes, e a solução está na especificação dos mesmos em termos de **precisão e estabilidade** à variações ambientais e temporais, e da própria confiabilidade dos componentes, ou seja da probabilidade de **falha catastrófica** dos mesmos. Todas estes fatores podem ser predizíveis em **conjuntos** com grande número de unidades, em função de observações e medidas feitas em um tempo suficientemente longo, que demonstrem uma coerência ao longo de todo o tempo. Considera-se então que o problema é de natureza estatística e pode ser modelado e resolvido com base em **matemática estatística**. Para isto são portanto necessários **modelos estatísticos** dos componentes para permitir o cálculo por **métodos estatísticos** das variações do desempenho dos circuitos [Bell,

Becker, Spence], como extensão dos métodos de projeto determinísticos usualmente adotados em engenharia eletrônica.

Os **componentes**, os **estímulos** e as **respostas** dos circuitos passam a ser descritos por **variáveis aleatórias**, que substituem as variáveis **determinísticas** correspondentes aos valores **nominais** tratados no projeto clássico, e vão permitir o cálculo direto das variações de desempenho dos circuitos. Este cálculo pode ser realizado de diferentes modos, dependendo dos parâmetros utilizados para definir desempenho e forma de operação do circuito. Por exemplo, a análise de um amplificador de carga se concentra na resposta à transientes, nas relações entre tensões e correntes quando estas variam continuamente no tempo. Para isto o programa simulador de circuitos SPICE é a melhor opção, tendo sido adotado neste trabalho pelo grau de confiança reconhecido a nível industrial e acadêmico. Já um circuito de coincidências de um sistema de "trigger" deve ser analisado em termos de sua resposta a sinais digitais variáveis no tempo, por um simulador lógico ou funcional.

Introduzindo o problema da **variabilidade** dos componentes visando a garantia da qualidade dos circuitos em ambos os casos, exige que os respectivos simuladores passem a integrar um sistema que cria **amostras do circuito** dentro de um modelo estatístico realista, analisa as amostras usando o simulador apropriado para a análise desejada de cada amostra, e armazena os dados para posterior **análise estatística de dados**.

O objetivo final de um projeto visando qualidade é a **minimização da variabilidade do desempenho dos circuitos**, ou das **perdas de produção**, como será visto a seguir em exemplos distintos, onde tanto a **análise** quanto a **síntese** de circuitos são feitas por métodos **estatísticos** [Bell, Becker, Spence], tendo como objetivo garantir a qualidade de projeto e portanto dos circuitos fabricados.

Para melhor ilustrar a aplicação deste método de projeto, serão apresentados **exemplos** de projeto estatístico de circuitos, seguindo a **seqüência de operações** de um **projeto tradicional** :
1 - definição de **especificações**,
2 - escolha de possíveis **arquiteturas e componentes**,
3 - **modelagem** matemática de componentes e circuitos,
4 - **síntese** do circuito : definição dos componentes e suas ligações;
5 - **análise** do circuito : testes de atendimento às especificações;
6 - **otimização** : valores ótimos dos componentes, repetição das etapas 4 a 6;
7 - **avaliação** e comparação das possíveis arquiteturas, repetição das etapas 2 a 6;
8 - **documentação** para implementação da produção.

As seguintes etapas experimentais tem uma importância vital para o projeto de circuitos:
4a - desenho de **máscaras** de processos ou de circuitos impressos,
5a - construção de **protótipos** e de "cabeça de série" ou ainda simulação por computador,
7a - **testes de aceitação** dos circuitos fabricados ou simulados.

Entretanto este **trabalho se limita** aos aspectos de **concepção e simulação de circuitos**, focalizando o **projeto elétrico** dos mesmos, e por isto nele não se inclui nenhuma destas etapas experimentais, apesar de que um projeto se **inicia** no estabelecimento das especificações, e **termina** no desenho de figuras geométricas de alta precisão tais como

circuitos impressos para componentes discretos, e máscaras de processos de fabricação para circuitos integrados monolíticos ou híbridos. E é efetivamente concluído com a **avaliação** dos circuitos sendo **utilizados em diversas condições**.

2.2 - Projeto de Circuito com Tolerâncias - Considerações e Exemplos

O primeiro exemplo, dado a seguir, é o projeto de um simples **atenuador resistivo**, seguindo a seqüência de operações da metodologia tradicional de projeto de circuitos analógicos, ampliada por métodos estatísticos com base em programas computador feitos para aplicações gerais em Física [CERN_LIB, CERN_PAW] e no CPSPICE, programa desenvolvido nesta tese para simulação estatística de circuitos eletrônicos analógicos.

O problema de projetar circuitos com tolerância será abordado aqui com a apresentação passo a passo de um projeto simples, o de um atenuador resistivo, seguindo primeiramente uma seqüência de operações dentro da metodologia tradicional de projeto de **circuitos analógicos**, estendendo posteriormente o projeto com base em métodos estatísticos. Os circuitos analógicos formam a base dos circuitos de medidas, controle e telecomunicações, e mesmo circuitos digitais são analisados como circuitos analógicos quando operam nos seus limites de velocidade.

2.2.1. Especificações e diretrizes de projeto

Este exemplo é uma variante de um clássico da literatura [Becker, Bell]. Trata-se do projeto de um **atenuador de 10 vezes**, que apresenta uma **impedância de saída** igual à da carga, de 50 ohms, com **tolerâncias** respectivamente de 1% e 2% na **faixa de freqüências** de 0 a 50 MHz, dentro de **condições ambientais** restritas à temperaturas entre 0 e 50 graus centígrados. Uma situação típica, onde o desempenho dos circuitos é definido por **múltiplos parâmetros** e para o qual existem **múltiplos objetivos**. Considera-se também como objetivo de projeto que o circuito tenha o **menor número de componentes**, visando tanto a redução de **custos** quanto de **possibilidades de falhas** do circuito.

Circuitos eletrônicos **processam informação** contida em **sinais elétricos**, segundo **especificações** codificadas na linguagem dos seus usuários e projetistas. Estas podem variar de uma **aplicação** para outra, assim como varia o grau de importância dos seus **atributos**.

Para que estas especificações possam ser apresentadas aos programas que auxiliam o projeto de circuitos, devem ser transformadas em **relações matemáticas** como a equação (1), onde os possíveis valores dos componentes, estímulos e respostas do circuito são representadas pelas **variáveis** p, x, y, respectivamente contidas nos **domínios P, X, Y** e de dimensões q, n e m :

$$(1) \qquad y = f_S(x,p)\ ,\ y \in Y \subset R^m \ \ \forall\, x \in X \subset R^n\ ,\ \forall\, p \in P \subset R^q$$

Nesta equação os **estímulos** ao circuito são representados por **x**, que representa grandezas **elétricas** (tensão, corrente, potência), **ambientais** (temperatura, pressão, luz) ou **temporais** (tempo, freqüência, fase). Os **parâmetros dos componentes**, tais como valores nominais, tolerâncias, dissipação máxima de potência, estabilidade com temperatura, são

representados por **p**. As respostas do circuito, ou **parâmetros de projeto**, são as variáveis dependentes **y,** também grandezas elétricas ou temporais, que podem ser aleatórias ou não, de acordo com a natureza de **x** ou **p**.

No projeto clássico de circuitos sem tolerância, ou de **tolerância zero**, p, x, e y assumem valores únicos, cada um dos conjuntos P, X e Y contém um único elemento.

Quando existem tolerâncias para os componentes do circuito dentro de um projeto determinístico, se assumem intervalos de valores no lugar de valores nominais. Assim se espera como resposta de um circuito famílias de funções $\mathbf{f_S(x,p)}$, contidas no domínio **Y**. Os estímulos são representados por variáveis independentes restritas aos domínios **X**, e podem ser definidos por famílias de funções. Os valores dos componentes devem em conseqüência ser delimitados em **P** para atender às especificações dadas em (1).

Quando se considera no projeto que o tipo de componente é único, com tolerância pré fixada, diz-se que o projeto é de **tolerância fixa**. Caso seja possível escolher entre diversos tipos de componentes, com tolerância e estabilidade diferentes, que devem ser especificadas por projeto, diz-se que este é do tipo de **tolerância variável**.

Os métodos determinísticos apresentam uma dificuldade nos casos onde podem não ser atendidas as especificações : não se pode avaliar a **freqüência** de ocorrência dos valores que levam à falha ou ao sucesso no atendimento às especificações. Portanto não se pode avaliar o **grau de importância das falhas**, e em conseqüência não se pode calcular **rendimento ou perdas de produção**, ou ainda o **custo total** de produção. Para isso é necessário que sejam adotados métodos estatísticos. Já nos métodos estatísticos de análise se atribuem aos componentes e parâmetros de circuito, **funções densidade de probabilidade**. Assim se pode quantificar o atendimento às especificações. Falhas passam a ser admissíveis, fica possível otimizar o projeto em termos de rendimento de produção, perdas por falhas e custos na medida em que seja possível associá-los à **tolerância** e à **confiabilidade** dos componentes [IEE82, IEEE86, Becker, Spence, Taguchi].

Nesta visão o projeto de um circuito consiste em definir a **arquitetura** e seus **componentes** tais que as variáveis de saída **z** do circuito projetado coincidam dentro de uma precisão ε com as respostas esperadas **y**, tal como dada na equação (2):

(2) $$z = f_D(p,x) \text{ tal que } |y - z| < \varepsilon \ \forall \ z \in Z \subset Y \subset R^m, \ x \in X \subset R^n, \ p \in P \subset R^p$$

onde z, x e p são os valores, ou ocorrências das variáveis aleatórias Z, X e P, as quais são definidas em termos de distribuições estatísticas.

Concluindo, este exemplo tem as seguintes especificações clássicas :

1 - Variáveis de projeto : tensões de entrada Ve e saída Vs do atenuador.
atenuação A = [Vs / Ve] e impedância de saída Z = [Vs / Is]
temperatura T.

2 - Limites: tensões máximas de 2 volts
atenuação A = 10 ± 0,1 (1%) , impedância Z = 50 ± 1 ohms (2%)
temperatura T = 25 ± 25 oC

Do ponto de vista estatístico, atenuação e impedância são definidos por distribuições estatísticas. Em um exemplo dentro do padrão de qualidade 6 σ, estas distribuições seriam do tipo gaussiana com um desvio padrão 6 vezes menor do que a tolerância especificada [Taguchi]. Os modelos para A e R seriam as funções densidade de probabilidade :

(3) $p(A) = (\sqrt{(2 \bullet \pi)} \bullet \sigma_A)^{-1} \bullet \exp(-(\sqrt{(2)} \bullet \sigma_A)^{-1} \bullet (A - E(A)))^2$, $E(A)=10$; $\sigma_A = 0{,}01333...$

(4) $p(R) = (\sqrt{(2 \bullet \pi)} \bullet \sigma_R)^{-1} \bullet \exp(-(\sqrt{(2)} \bullet \sigma_R)^{-1} \bullet (R - E(R)))^2$, $E(R)=50$; $\sigma_R = 0{,}1333...$

Dessa forma, a probabilidade de que um circuito apresente atenuação ou impedância de saída fora dos limites acima é da ordem de 1 por cada milhão. Não se espera portanto que todos os circuitos fabricados atendam a todas as especificações, mas se procura fazer com que este número seja insignificante, escolhendo componentes com valores nominais e tolerâncias compatíveis com o custo permitido.

2.2.2. Escolha de possíveis arquiteturas e componentes

A escolha da arquitetura, ou do circuito, geralmente recai naquelas já **consagradas pelo uso**, que já tenham apresentado bons resultados em aplicações semelhantes. O circuito escolhido para este exemplo está indicado de forma esquemática na figura 1. A escolha dos resistores, sejam discretos em filmes de carvão ou metálicos, ou integrados em filme espesso ou fino, deve garantir simplicidade e precisão necessárias. O projeto portanto consiste na definição dos **valores nominais**, **precisão** e **estabilidade** necessários a estes dois resistores, para que as especificações dadas sejam atendidas. Diversas arquiteturas podem ser estudadas ao mesmo tempo.

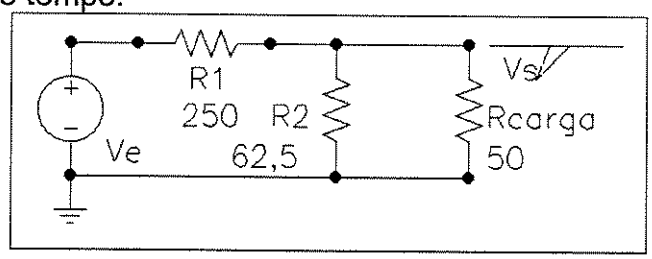

Figura 1 - Esquemático do atenuador resistivo do exemplo 1

2.2.3. Modelos dos componentes e análise do circuito

Assumindo-se um modelo resistivo simples para os resistores e carga, se simplifica a análise do circuito esquematizado na Figura 1, cujo resultado são as relações biunívocas entre as resistências R_1 e R_2, a atenuação **A** e a impedância de saída **R,** indicadas nas equações (5-6) :

(5) $$A = \frac{Ve}{Vs} = 2 \bullet \left(1 + \frac{R1}{R2}\right) = 10$$ (6) $$R = \left(\frac{R1 \bullet R2}{R1 + R2}\right) = Rc = 50 \text{ ohms}$$

Isto não é típico de um projeto de circuito, onde geralmente não é possível estabelecer relações analíticas **explícitas** entre os parâmetros de projeto e os valores nominais dos componentes, mas **relações numéricas** obtidas através de análises de medidas realizadas em **protótipos, ou simulações** realizadas por programas como o SPICE. Estas simulações

por computador são tradicionalmente numéricas. Entretanto alguns programas comerciais como Mathematica e MAPLE e acadêmicos como o REDUCE permitem **análises algébricas** que permitem ao projetista visualizar diretamente as relações entre variáveis e parâmetros de projeto, como neste exemplo as expressões algébricas da atenuação e impedância de saída do atenuador. Quando o resultado da análise algébrica resulta em **funções biunívocas**, como neste caso extremamente particular, a síntese se funde com a análise, e o processo de projeto se simplifica muito.

O projeto nesta etapa deve ser **expandido com a estatística** : A e R foram especificadas no início do projeto como variáveis aleatórias, devido ao fato de serem os valores de R_1 e R_2 **aleatórios**, por causa da variabilidade dos resistores. Portanto R_1 e R_2 devem ser especificados como funções dos parâmetros de projeto A e R, dados em (3) e (4) como distribuições estatísticas p(A) e p(R).

2.2.4. Valores nominais e tolerância dos componentes : a síntese do circuito

Em termos tradicionais, a **síntese** do circuito feita com o cálculo dos valores nominais é completada com o cálculo da **tolerância** dos componentes. Neste passo do projeto se escolhe um modelo matemático f_D para as **especificações** do circuito, que garanta uma solução para a equação (7), que pode ser vista como o problema inverso ao da análise, dado em (1) :

(7) $$p = f_D^{-1} (z = y , x)$$

Na **síntese de circuitos**, z representa a **resposta** do circuito e y são os **valores desejados** para a resposta do circuito. E na **síntese de modelos** para componentes ou circuitos, y representa os **valores medidos** da resposta. A solução deste problema pode não existir, caso as especificações sejam muito restritivas ou conflitantes. E caso existam, podem não ser únicas. A forma usual de garantir uma solução para (7) é aplicar **métodos de otimização com restrições**, onde se delimita os domínios dos parâmetros de componentes a especificar e das variáveis de projeto:

$$\mathbf{z \in Z \subset Y \subset R^m , \ p \in P \subset R^p}$$

na busca para atender aos objetivos do projeto, que podem ser expressos na forma algébrica geral, não linear :

(8) $$\min_P \| z - y \| \qquad g(z, p) \geq 0$$

ou na forma linear, mais simples e portanto mais utilizada :

(9) $$\min_P \| z - y \| \qquad A z \geq b$$

Este problema sempre que possível deve ser **particionado em outros menores** e mais apropriados para uma solução numérica, ou mesmo analítica. Quando a **norma do erro** em (8) e (9) é quadrática usam-se **métodos de regressão ou mínimos quadrados** para resolver o problema da equação (10) :

(10) $$\min_P [(z - y)^T * W * (z - y)]$$

E quando é a norma máxima usam-se **métodos minimax** ou de programação linear para resolver o problema da equação (11):

(11) $\min_P [\max | z_i - y_i |]$

Entre os **algoritmos** mais usados na solução de (10) ou (11) se destacam os de Nelder-Mead, Fletcher-Powell e Gauss-Newton [Billes, Blum, Eadie, Kuester, Press]. Nenhum deles tem **convergência** garantida para uma solução caso ela exista, ou seja, não se pode garantir que não exista uma solução caso não ocorra convergência. Busca-se possibilitar a convergência criando as funções objetivo tais como **funções convexas**, com um único mínimo, para evitar a ocorrência de **mínimos locais** que fazem a convergência depender do ponto de partida da pesquisa do mínimo, reduzindo a possibilidade de convergência para o **mínimo global** [Biles, Brayton]. Mas tal transformação nem sempre é possível. Algumas técnicas para alcançar o mínimo global em funções não convexas tem sido propostas na literatura [Haase, Moebus]. Elas são normalmente baseadas em **métodos estocásticos**, o mais conhecido dos métodos é o de simulação de recristalização por resfriamento lento, ou "**simulated annealing**" [Kirkpatrick, Szu, Press_SA], de alto custo computacional.

Existem três linhas de solução para o problema da síntese, dependendo de considerações sobre tolerância dos componentes e das respostas do circuito: as análises determinísticas de **pior caso**, e as análises estatísticas pelo **método dos momentos**, aplicável apenas aos casos de pequenas variações e sistemas lineares, e por **métodos de Monte Carlo**, de aplicação geral, baseado em simulações de eventos.

O método de pior caso pode ser visto neste projeto simples como a solução das equações (3) e (4) sujeita às restrições dadas nas especificações, começando pelo cálculo do valor nominal, que neste exemplo esquematizado na Figura 1 é direto :

(12) $R_1 = \frac{A \bullet R}{2} = 250 \text{ ohms}$ (13) $R_2 = \frac{A \bullet R}{A-2} = 62.5 \text{ ohms}$

Em projetos mais complexos geralmente os valores nominais são obtidos através de **varredura** de parâmetros, processos de **tentativa e erro** nem sempre eficientes. Ou ainda pelo uso de **programas de otimização** disponíveis nas bibliotecas de programas numéricos como IMSL e CERNLIB. Estes programas são baseados em métodos numéricos iterativos, que não garantem a convergência para uma solução ótima, e portanto não evitam o uso de métodos de tentativa e erro.

O projeto segue nesta linha com o cálculo da **tolerância máxima** para os resistores que mantém os valores de A e R dentro dos limites estabelecidos. Novamente a simplicidade do circuito permite aqui o cálculo analítico dos valores extremos para R_1 e R_2, apesar das equações (5) e (6) não serem lineares :

(14) $A_{max} = 10,1 \geq A = 2 \bullet \left(1 + \frac{R1}{R2} \right) \geq A_{min} = 9,9$

(15) $R_{max} = 51 \geq R = \left(\frac{R1 \bullet R2}{R1 + R2} \right) \geq R_{min} = 49$

As soluções possíveis para as inequações (14) e (15) estão indicadas na figura 2 pela área hachurada. Todos os pares de valores possíveis para R_1 e R_2 que atendem às especificações deste circuito, podem ser identificados nesta área, cujos vértices são os pontos extremos calculados a partir das equações (5) e (6) : (A; R; R_1; R_2) = (10,1; 51; 257,5; 63,5), (9,9; 51; 252,4; 63,9), (10,1; 49; 247,5; 61,1), (9,9; 49; 242,6; 61,4).

Vê-se na figura 2 que enquanto não for explicitada uma relação ou outra restrição sobre os valores nominais e tolerância dos resistores, existem infinitas possibilidades de atendimento às especificações dadas, que podem ser visualizadas como áreas retangulares inseridas dentro da área permitida.

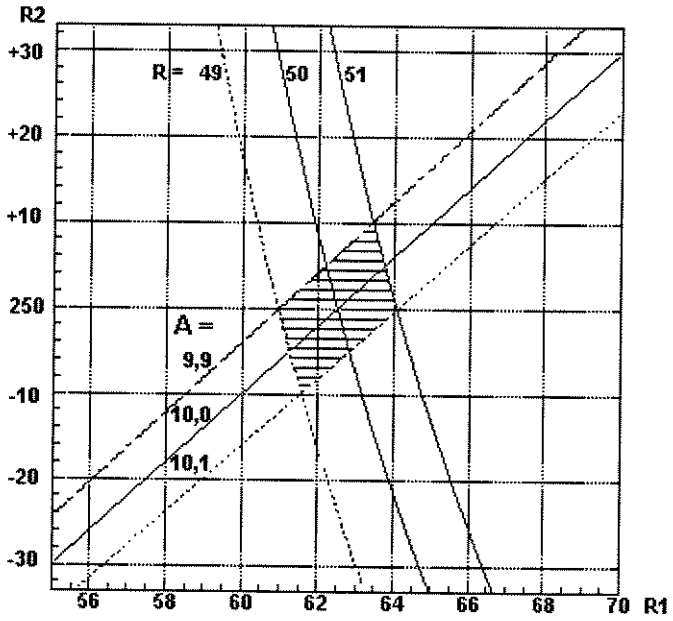

Figura 2 - Valores possíveis para os valores das resistências

Chega-se a esta mesma conclusão pela **análise variacional** das equações (5) e (6), sujeitas às restrições dadas por (14) e (15), da qual se obtém relações algébricas para as variações máximas de A e R ou R_1 e R_2 :

(17) $$\frac{dR}{R} = \frac{\frac{R2}{R1 + R2} \bullet \frac{dR1}{R1} + \frac{R1}{R1 + R2} \bullet \frac{dR2}{R2}}{1 + \frac{R1}{R1 + R2} \bullet \frac{dR1}{R1} + \frac{R2}{R1 + R2} \bullet \frac{dR2}{R2}} \leq 0{,}02$$

(16) $$\frac{dA}{A} = \frac{R1}{R1 + R2} \bullet \left(\frac{dR1}{R1} - \frac{dR2}{R2} \right) \bullet \frac{1}{1 + \frac{dR2}{R2}} \leq 0{,}01$$

As equações (15) e (16) permitem relacionar tolerância e valores nominais de R_1 e R_2 para qualquer amplitude de variação destes, pois não foram baseadas no uso de derivadas. Resolvendo o sistema de equações acima, sem acrescentar restrições para às variações de R_1 e R_2 , significa cair no caso particular em que as variações dos valores de R_1 e R_2 são independentes e mutuamente exclusivas, isto é quando um resistor varia o outro mantém seu valor.

Analisando-se como em [Becker] outros casos particulares de solução das equações (16) e (17), é possível examinar as conseqüências da existência de **relações de dependência** de interesse para este projeto entre os valores nominais das resistências. Para o caso da resistência R_1 ser linearmente dependente da resistência R_2, observa-se que o valor de **A** independe do valor das resistências, mas o valor de **R** passa a ter sensibilidade máxima, de forma que para atender às especificações é necessário ter :

$$(18) \qquad \left\| \frac{dR1}{R1} \right\|_{max} = \left\| \frac{dR2}{R2} \right\|_{max} \leq 0{,}025$$

Se esta dependência for negativa, o valor de **R** será bem menos sensível mas ocorre o pior caso de variação para **A** em função da variação das resistências. A solução das equações (16) e (17) mostra uma tolerância ainda mais restrita para as resistências :

$$(19) \qquad \left\| +\frac{dR1}{R1} \right\|_{max} = \left\| -\frac{dR2}{R2} \right\|_{max} \leq 0{,}006$$

Quando a resistência $\mathbf{R_1}$ for independente da resistência $\mathbf{R_2}$ temos como pior caso a definição das tolerâncias, dada por (18). Por outro lado, adotar o critério de erros relativos iguais para $\mathbf{R_1}$ e $\mathbf{R_2}$, eqüivale ao caso dado em (19):

$$(20) \qquad \left\| \frac{dR1}{R1} \right\|_{max} = \left\| \frac{dR2}{R2} \right\|_{max} \leq 0{,}006$$

Estabelecer um critério de dependência entre $\mathbf{R_1}$ e $\mathbf{R_2}$ eqüivale em termos geométricos a definir a forma do retângulo na figura 2 correspondente às tolerância das resistências, o qual deve estar inserido dentro da área hachurada para se atender às especificações.

Em projetos de maior complexidade é praticamente impossível estabelecer analiticamente, como neste exemplo, as possíveis soluções de compromisso para as especificações e as tolerância dos componentes. O problema é geralmente colocado como de sensibilidade da resposta do circuito à variação de seus parâmetros, existindo uma ampla literatura onde as soluções são baseadas em métodos de otimização de sistemas de equações. Quando estes sistemas podem ser considerados **lineares** por se assumir pequenas variações em torno do valor nominal, é possível obter uma solução por métodos diretos, cujo esforço computacional é proporcional ao tamanho do circuito e ao número de componentes variáveis. Mas normalmente os sistemas **não são lineares**, pela natureza de seus componentes, e pelo fato de serem as variações de parâmetros geralmente grandes demais para permitir o uso de métodos baseados em derivadas. Neste caso então não se tem nenhuma garantia de existência de solução ótima, nem de convergência para as que existirem, por parte dos métodos de otimização existentes. Quando o número de parâmetros

de projeto aumenta, o esforço necessário a esta tarefa pode aumentar exponencialmente. É necessário então a adoção de técnicas especiais de síntese, como a de Monte Carlo.

2.2.5. Estabilidade dos componentes às variações do ambiente

A complexidade do projeto aumenta pouco quando se considera a influência do ambiente no circuito, pois os valores dos componentes passam a variar não só devido à tolerância dos processos de fabricação dos mesmos como também pela influência da temperatura, umidade e tempo, entre outros fatores ambientais. Mas estas variações podem ser modeladas com simplicidade, e a influência nas respostas do circuito pode ser calculada facilmente. Neste exemplo a tolerância a ser especificada aos resistores poderá ser tanto maior quanto maior for a estabilidade destes às variações ambientais, particularmente a temperatura T e o tempo t. Em termos de projeto, **tolerância e estabilidade são especificações complementares**. São geralmente adotados para ambos tipos de estabilidade modelos quadráticos ou lineares, dependendo da precisão necessária :

(21) $$R(T) = R_o \bullet [1 + E_{T1} \bullet \Delta T + E_{T2} \bullet \Delta T^2) \quad \approx R_o \bullet (1 + E_{T1} \bullet \Delta T)$$

(22) $$R(t) = R_o \bullet [1 + E_{t1} \bullet \Delta t + E_{t2} \bullet \Delta t^2] \quad \approx R_o \bullet (1 + E_{t1} \bullet \Delta t\ )$$

A especificação destes parâmetros é feita em número limitado de opções de escolha de tipos de resistores. O objetivo de reduzir ao mínimo o custo de produção leva à adoção da maior tolerância e menor estabilidade possíveis para os resistores. Isto é feito por análise de casos. Por exemplo, considerando para este projeto variações de temperatura de ± 30 °C e um tempo máximo de 10 anos de operação do circuito, se observa que a escolha de resistores com tolerância de 0,2% e estabilidade com temperatura e tempo respectivamente de + 100 ppm/°C e + 100 ppm/ano, o que dá uma variação máxima total de ±(0,2% + 100 ppm/°C •30°C + 100 ppm/ano •10 anos) = ±0,6%, atendem as especificações dadas tanto quanto resistores com (0,1%, 150 ppm/°C, 50 ppm/ano), ou ainda (0,5%, 20 ppm/°C, 40 ppm/ano). A escolha de qual dos tipos adotar fica por conta de uma análise de custo.

2.2.6 - Otimização do circuito, cálculo dos valores ótimos dos componentes

A análise acima corresponde ao caso clássico de rendimento de produção de 100%, ou seja todos os circuitos fabricados naquelas condições atenderão às especificações do projeto. Pode-se buscar objetivos de projeto mais **convenientes** à produção, como por exemplo realizar um **lucro máximo** de produção ou reduzir ao mínimo seu custo global. Este custo global tem uma parte **fixa**, relacionada com montagem e testes do circuito, sendo função tanto da arquitetura quanto do custo individual dos componentes determinados anteriormente à conclusão do projeto. A parte **variável** está relacionada com o custo dos componentes a serem determinados pelo projeto. E está relacionada também com o número de circuitos que forem rejeitados por não atender às especificações. Os custos dos componentes são funções inversas das tolerância $\mathbf{t_i}$ e das variabilidades com o meio ambiente $\mathbf{v_i}$. Um exemplo de modelo de custo global dentro deste critério é dado na equação (23) :

(23) $$C_T = C_o + \sum_i C_{vi} / v_i + \sum_i C_{ti} / t_i$$

São igualmente necessárias avaliações da **freqüência de ocorrência** dos valores dos componentes, isto é, da importância destas para prever perdas ou rendimento da produção,

que implicam na necessidade de **soma ou integração** de resultados de simulações. Os métodos de **Monte Carlo** são os de mais ampla aplicação e de maior eficiência para realizar esta modelagem estatística, no caso de múltiplas variáveis e condições de contorno de integração irregulares,. Estes métodos se apresentam na forma da equação (24), onde f é uma função das múltiplas variáveis aleatórias x, valores dos componentes do circuito :

$$(24) \quad I = \int_{X1} \int_{X2} \ldots \int_{Xn} f(x_1, x_2, \ldots x_n)\, dx_1\, dx_2 \ldots dx_n \approx \sum_i f_i(x_1, x_2, \ldots x_n), \; x_i \in X_i \subset R$$

A integração é realizada acumulando-se os resultados das simulações das amostras dos circuitos, obtidas com a atribuição de valores aos componentes de acordo com os valores de um vetor aleatório multivariável $x = [x_1, x_2, \ldots x_n]$ gerado numericamente segundo modelos estatísticos das variações dos componentes [Spence, Rubinstein, Hammersley].

Os modelos estatísticos adotados nesta tese são **distribuições contínuas** como a uniforme ou a gaussiana, ou **descontínuas** como histogramas, ou combinações destas. **Programas geradores de números aleatórios** são usados na simulação de Monte Carlo dos circuitos para determinação dos valores dos componentes das amostras, que são analisadas para determinar os parâmetros de projeto, A e R. O número de amostras necessárias é determinado pela precisão desejada ou pelo tempo disponível para a simulação. O resultado são estatísticas de A e R em função das de R_1 e R_2 , como é mostrado na figuras 3a e b, histogramas feitos com PAW para pequeno e grande número de amostras. Estes histogramas, como **representações gráficas** da integração numérica realizada, fornecem informações sobre perdas e rendimentos de produção do circuito projetado, diretamente da ocorrência de valores de A e R, dentro ou fora das especificações dadas. Através do uso do programa PAW é possível **filtrar os eventos**, obter os **máximos e mínimos** de ocorrências, bem como o total destas.

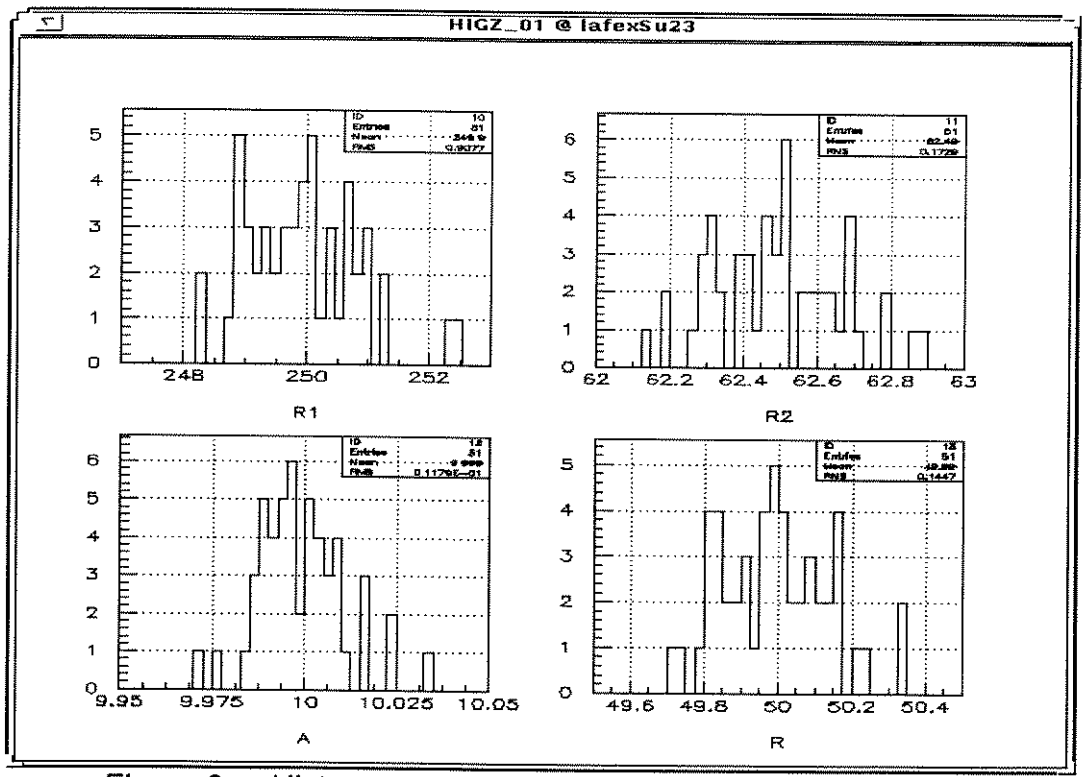

Figura 3a - Histogramas para 50 amostras geradas por PAW

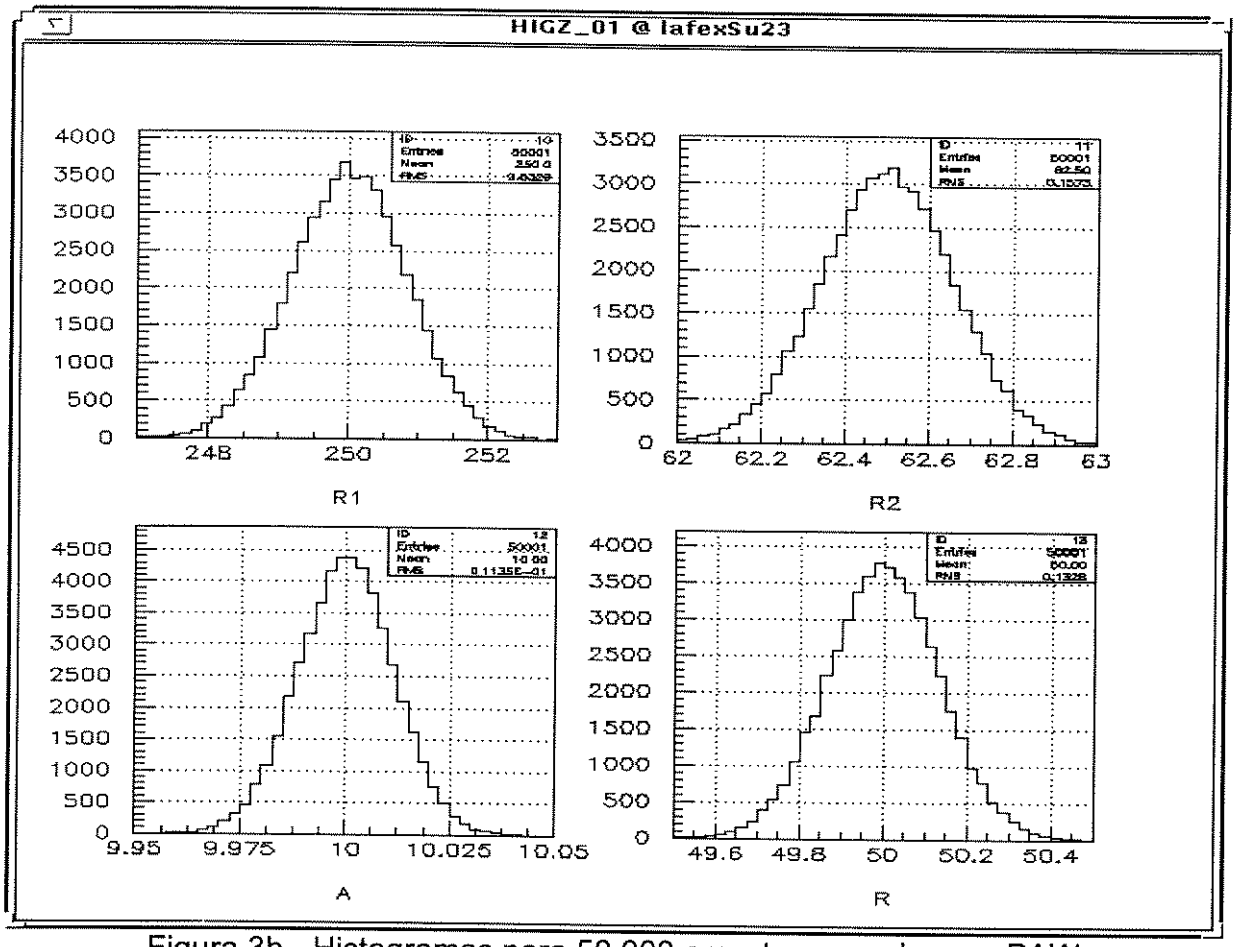

Figura 3b - Histogramas para 50.000 amostras geradas por PAW

Observa-se nas figuras 3a e 3b o problema básico da análise de Monte Carlo: o aumento de precisão da análise exige um aumento ao **quadrado do número de amostras** analisadas [Rubinstein].

Quando se considera o caso de resistores discretos, adota-se **correlação** nula entre seus valores. Neste caso, a escolha do tipo de resistores a ser usado é feita com base em **custo**, função da **tolerância e estabilidade com a temperatura**. Supondo que a escolha neste exemplo esteja limitada a 4 tipos de resistores, cada um definido por uma das seguintes tríades custo / tolerância / estabilidade :

a) 10 / 0,5% / 10 ppm/oC;
b) 5 / 0,5% / 100 ppm/oC;
c) 2 / 1% / 100 ppm/oC;
d) 1 / 2% / 100 ppm/oC.

Neste exemplo foram simulados 1000 circuitos, com base nos dados acima, tendo como modelo de custo a equação (23), e se adotou distribuições estatísticas gaussianas **com desvio padrão de 1/3 da tolerância**. A análise dos resultados apresentados na figura 4a leva à escolha de resistores do tipo (c) para realizar um circuito discreto de custo mínimo.

A integração de resistores em filme fino para este circuito possibilita introduzir um parâmetro de projeto adicional : a **correlação** entre resistências, função da **geometria** dos

resistores [Till, Glaser, Colclaser]. A figura 4a apresenta o resultado de simulações de Monte Carlo que indica uma possibilidade de **aumentar a tolerância** dos resistores, reduzindo os custos, **sem reduzir o rendimento de produção**, que é estabelecer uma correlação entre valores de R_1 e R_2 em torno de 0,7. Se isto for possível o número de unidades com falhas por lote produzido pode ser reduzido por um fator de 3 ou mais em relação ao resultado obtido com um projeto sem considerar correlação.

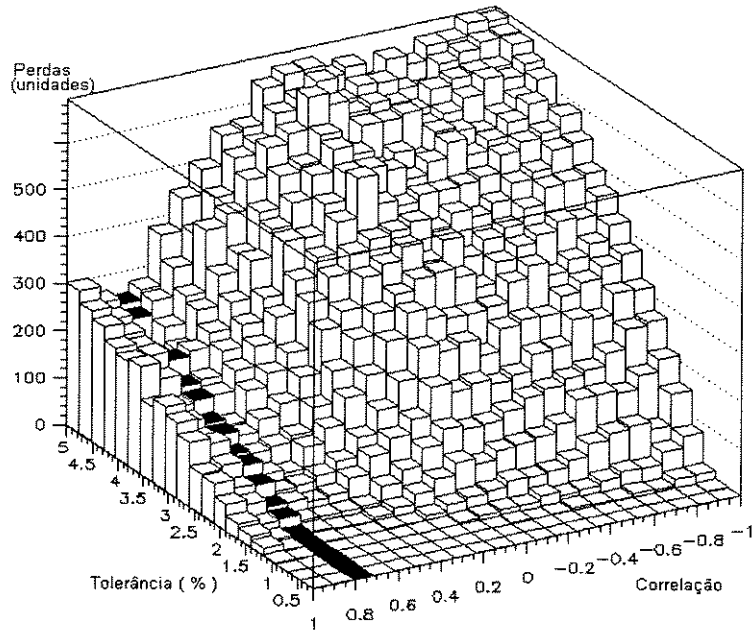

Figura 4a - Número de unidades perdidas por 1000 unidades fabricadas em função da tolerância (0,2 a 5%) e correlação (-1 a +1) dos resistores.

Geralmente para cada projeto existem inúmeras possibilidades de otimização no sentido de melhorar a qualidade, tanto em confiabilidade quanto em desempenho do circuito, alterando os valores de componentes e a própria arquitetura do circuito. A **tomada de decisões** para atender aos compromissos entre **objetivos múltiplos**, inclusive aqueles em **conflito**, deve atender aos objetivos do projeto segundo a hierarquia destes. O projetista deve evidenciar estes objetivos na simulação de Monte Carlo. Neste exemplo se o **objetivo** for obter o **máximo lucro de produção**, definido como o número de unidades que atendem à todas as especificações dividido pelo custo total de cada unidade, é necessário construir um histograma desta **função de lucro** para nele observar o máximo da função. Como pode ser visto na Figura 4a, este máximo ocorre para uma correlação em torno de 0,7, tolerância em torno de 1% e estabilidade em torno de 100 ppm. Isto pode ser considerado o projeto ótimo para o circuito deste exemplo . Quando o número de variáveis é grande, este processo deve ser aplicado aos pares de variáveis críticas, ou **automatizado** com base em métodos de otimização estatística para múltiplas variáveis e múltiplas funções objetivo. Isto é discutido na seção 2.7 e no apêndice I.

O método de projeto aqui proposto pode ser resumido na seqüência de passos :

Dados	Ferramentas	Resultados
Parâmetros de projeto	→ CERNLIB	→ modelos estatísticos de componentes.
Propostas de circuitos	→ CPSPICE	→ simulação de Monte Carlo.
Especificação do projeto	→ PAW	→ filtragem das amostras funcionais.
Otimização de custos	→ PAW	→ escolha do circuito com a respectiva

especificação dos componentes.

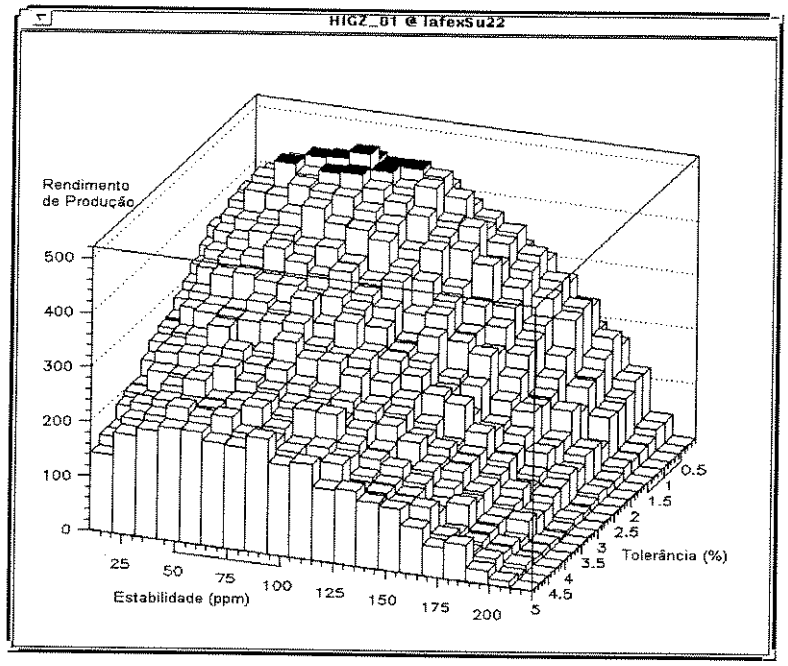

Figura 4b - Lucro de produção em função da tolerância (0,2 a 5%) e estabilidade (10 a 300 ppm/oC) dos resistores do atenuador, para o caso de correlação nula entre resistências.

2.2.7 - Avaliação de resultados e possível repetição das etapas anteriores

É necessário ressaltar que, freqüentemente ao longo de um projeto, se precisa definir certos parâmetros de projeto através da execução de experimentos ou concepções geométricas ou mecânicas, como por exemplo o desenho de máscaras de processos ou a construção de protótipos, inclusive de circuitos impressos. Nos resistores integrados a geometria influi tanto na correlação entre valores nominais quanto na tolerância. Isto pode ser usado para fins de otimização do circuito, como se observa na figura 4b, que indica um valor ótimo de correlação entre resistores, para fins de redução de perdas de fabricação, entre 0,7 e 0,8. Resistores discretos selecionados ao acaso no estoque disponível geralmente apresentam valores com distribuições estatísticas independentes.

Em muitos casos não é possível prever o comportamento do circuito por falta de modelos adequados, se tornando necessário primeiro criar estes modelos seja por simulação por computador ou por prototipagem do circuito ou mesmo a construção de uma cabeça de série de produção. Este é geralmente o caso de circuitos discretos de alta freqüência, que são muito afetados pela geometria do circuito impresso, devido a componentes parasitas resultantes, como capacitâncias entre componentes próximos, indutâncias de condutores mais longos, etc ... Estes detalhes não podem ser modelados com precisão sem serem implementados ou simulados de algum modo. Na verdade o projeto só é concluído com a avaliação positiva dos circuitos efetivamente construídos e utilizados em diversas condições. Na seção seguinte é apresentado um exemplo onde a modelagem com base em medidas "in situ" é decisiva no projeto de um circuito.

2.3. Um exemplo de aplicação do Método dos Momentos

Este segundo exemplo complementa o anterior, na medida em que se trata de um sistema **não linear**, um **circuito gerador de veto** que recebe uma palavra digital de n = 32 bits, onde cada bit representa uma resposta positiva de um dos detetores de partículas. Quando o número de respostas positivas igualar ou exceder um dado valor n_V, é necessário vetar a aquisição de dados porque fica muito provável que os dados não possam ser interpretados corretamente pelo software de análise de dados posterior à aquisição. Na escolha do circuito, mostrado na figura 5, se buscou reduzir o número de componentes utilizados ao mínimo, pelos motivos já mencionados no exemplo anterior. Pretende-se gerar um sinal digital de veto, com um único circuito, um comparador de nível que recebe o valor médio das tensões correspondentes aos bits, em um tempo inferior a 10 ns. É necessário definir parâmetros do comparador a ser usado; valores nominais, tolerância e estabilidade do conjunto de resistores que realizam a soma; e a tensão de referência do comparador para estabelecer um limiar de veto entre 8 a 12 bits.

O primeiro passo deste projeto é o estabelecimento de modelos estatísticos para os níveis de tensão dos sinais digitais. As distribuições estatísticas das variáveis aleatórias que representam os sinais digitais foram obtidas diretamente durante a operação normal das placas de circuitos a serem substituídos, com o auxílio de um osciloscópio digital. Foram adquiridas grande número de amostras, não só das tensões dos sinais de entrada, como também as somas destas tensões na associação de resistores indicada na figura 5.

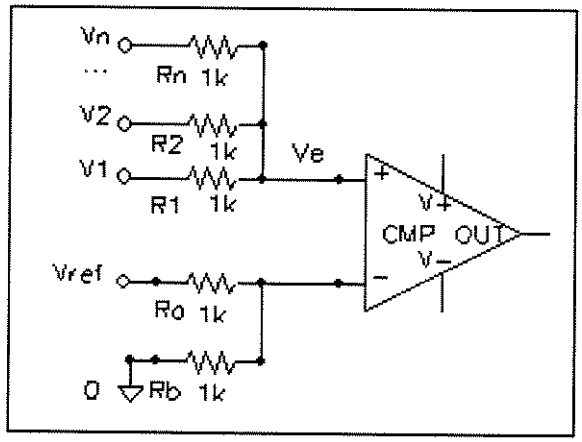

Figura 5 - Circuito analógico de veto para aquisição de dados

O osciloscópio usado tem recursos de processamento estatístico de dados e apresenta na tela as estatísticas dos sinais observados, em números e histogramas, como pode ser visto na figura 6. Estes histogramas foram armazenados em arquivos em disco de um computador acoplado ao osciloscópio.

Com estas medidas é possível determinar a distribuição estatística para um dado intervalo de tempo de variáveis de interesse para este projeto, indicadas na figura 5. Se considerando **nula** a corrente de polarização do comparador I_e, se tem o valor médio de V_e dado pela equação (17) :

(17) $$V_e = \Sigma_U(V_i \bullet G_i - I_e) / \Sigma_U(G_i) \approx \Sigma_U(V_i \bullet G_i) / \Sigma_U(G_i)$$

G_i, $i \in U$, são as condutâncias dos resistores, variáveis aleatórias independentes com distribuição normal de média zero :

(18) $G_i = G_o + g_i$, $g_i = N(0,t_G)$, $i \in U = f + v$

Os níveis de tensão V_i dos sinais digitais apresentam uma componente periódica devida à influência da comutação dos circuitos digitais rápidos, somada a uma componente aleatória devida aos outros ruído nas linhas de alimentação e terra do sistema. Na janela de tempo observada na figura 6 o ruído tem valor eficaz em torno de 27 mV, e em 98% do tempo a amplitude do sinal fica no intervalo de 3 sigma em torno da média, que é de 3,43 V (nível 1 de um sinal TTL). Medidas análogas para o nível 0 deram como resultado um valor médio de 0,53 V para a tensão e uma tensão de ruído equivalente.

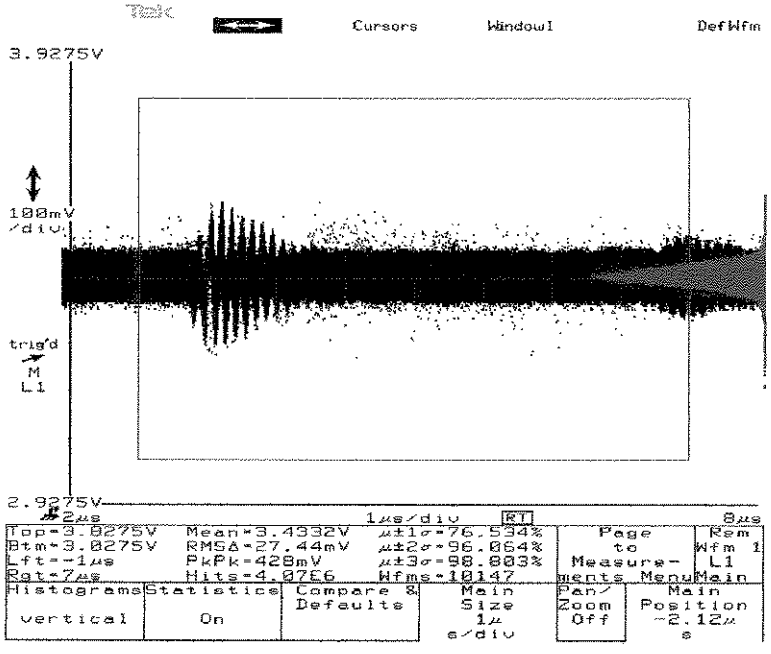

Figura 6 - Medidas estatísticas feitas com osciloscópio digital

Filtrando adequadamente estas linhas e inibindo o circuito de veto durante o tempo de ocorrência da interferência dos pulsos de comutação do sistema, o nível efetivo de ruído fica consideravelmente reduzido. Uma forma mais prática de medir a dependência do ruído de cada sinal digital é analisar diretamente a soma das tensões dos sinais digitais, aqui representada por $\mathbf{V_e}$. Isto permite evitar erros de modelagem da dependência estatística entre estes sinais e escolher o número de bits que garante uma baixa ocorrência de sinais de veto falsos. Para serem representativas as medidas devem ser feitas em diferentes períodos de tempo e com durações diferentes.

Os sinais digitais $\mathbf{V_i}$ podem ser definidos segundo a equação (19), onde v e f são as perturbações dos níveis de tensão para os sinais com estado verdadeiro ou 1, e estado falso ou 0. As distribuições gaussianas dos sinais aleatórios tem média zero e desvio padrão $\mathbf{t_F}$ e $\mathbf{t_V}$:

(19) $V_i = V + f_i = 0{,}53 + f_i$, $f_i = N(0,t_F)$, $i \in f$, para sinais de nível 0, $t_F = 0{,}023 / 0{,}53$

$= V + \Delta V + v_i = 3{,}43 + v_i$, $v_i = N(0,t_V)$, $i \in v$, para sinais de nível 1, $t_V = 0{,}027 / 3{,}43$

A tensão de referência deve se situar entre o nível de tensão para n sinais de bit 1 e o correspondente à existência de n - 1 sinais de bit um. Assim o comparador apresentará saída de nível 1 quando o número de sinais com bit 1 for maior ou igual a n. A tensão na entrada do comparador pode ser estimada pela equação (20), obtida a partir de (17) :

(20) $$V_e = [\Sigma_U(1+g_i)]^{-1} \bullet \{ V \bullet \Sigma_F[(1+f_i)\bullet(1+g_i)] + (V+\Delta V)\bullet\Sigma_V[(1+v_i)\bullet(1+g_i)] \}$$

Nesta expressão se identificam as seguintes funções de somas de variáveis aleatórias :

(21) $$g = \Sigma_U(g_i) = N(0,t_G/\sqrt{n}) \qquad f = \Sigma_F(f_i) = N(0,t_F/\sqrt{n_F}) \qquad v = \Sigma_V(v_i) = N(0,t_V/\sqrt{n_V})$$

E somas de produto com densidades de probabilidade nulas comparativamente:

(22) $$\mathbf{z_V = \Sigma_V(g_{ii}\bullet v_i) \approx 0} \qquad \mathbf{z_F = \Sigma_F(g_{ii}\bullet f_i) \approx 0}$$

O que resulta no valor esperado e variância de V_e dados por :

(23) $$E(V_e) = V + \Delta V \bullet (n_V / n)$$

(24) $$Var(V_e) = V^2\bullet Var[(n+g)^{-1}\bullet(f+v)] + \Delta V^2\bullet Var[(n+g)^{-1}\bullet(v+g_V - (n_V/n)\bullet g)]$$
$$= (V^2/n)\bullet\{ t_F^2/n_F + t_V^2/n_V + (\Delta V/V)^2\bullet[t_V^2/n_V + t_G^2/n_V - (n_V/n)^2\bullet t_G^2/n \}$$

A contribuição da tolerância dos resistores sendo menor do que a do ruído dos sinais digitais, da equação (24) simplifica a equação (24), que toma a seguinte forma :

(25) $$\mathbf{\Delta V_R < 3 \bullet V \bullet \{ [Var(f) + (1 + (\Delta V/V)^2) \bullet Var(v)] / n \}^{1/2}}$$

Esta equação, aplicada às estatísticas medidas, permite definir o número de bits que podem ser discriminados por este circuito, que resulta ser bem acima dos 32 bits da palavra usada para o veto.

2.4 Projeto de circuitos integrados com tolerância

Na seção 2.2 os circuitos eletrônicos foram classificados para fins de projeto com tolerância como discretos ou integrados. Circuitos a componentes discretos são projetados com base em componentes existentes dentro de um estoque disponível, cujos tipos podem ser muito variados, mas com valores nominais e tolerância normalmente limitados a valores discretos. Já os componentes integrados são menos variados em tipos, que são limitados pelo processo de fabricação, mas seus valores podem variar em um domínio quase contínuo, uma vez que são definidos pelo projetista pela sua geometria, limitada a dimensões mínimas possíveis ao processo de fabricação.

Em termos de dependência estatística, os parâmetros dos componentes integrados são normalmente correlacionados entre si, ao contrário dos componentes discretos. A análise estatística de circuitos discretos é mais complexa do que a de circuitos integrados, pois na modelagem estatística lote a lote, cada componente deve ser representado por uma ou mais

variáveis aleatórias, dependendo da diversidade dos tipos e a origem de seus componentes. Já os circuitos integrados apresentam poucos tipos de componentes advindos do mesmo processo de fabricação, e a modelagem estatística reflete o efeito das variações deste processo nos parâmetros elétricos, função da geometria e separação entre componentes.

O problema em simular circuitos integrados reside na ausência de dados estatísticos sobre as variações dos componentes. Os fabricantes, ou "foundries", fornecem no máximo valores nominais com informação sobre a tolerância dos parâmetros dos componentes dada em termos de pior caso, normalmente sob a forma do modelo mais rápido e de maior dissipação e do modelo mais lento e menos dissipador. Logo, os modelos apresentam as variações dos parâmetros definidas nos intervalos

(26) $$[\, p_{min} \,,\, p_{max} \,] = p_o \pm \Delta p.$$

As variações da resposta do circuito obtidas por simulação de Monte Carlo permitem uma economia de tempo de execução em relação ao método de pesquisa exaustiva de todas as possíveis ocorrências de variação extrema de valores de cada componente. Seja por análise combinatória ou de Monte Carlo, as possibilidades de otimização do projeto se dão com a filtragem dos eventos correspondentes ao atendimento às especificações. Os intervalos dos parâmetros dos componentes dos circuitos funcionais correspondem à solução da equação (27) :

(27) $$[\, p_{min} \,,\, p_{max} \,] = f_C^{-1} \,(\, z_{min} \,,\, z_{max} \,,\, x \,)$$

A análise de Monte Carlo é a mais adequada ao projeto ótimo porque a partir das funções densidade de probabilidade permitidas para a resposta do circuito, pela filtragem dos resultados, fornece as funções densidade de probabilidade admissíveis para os componentes. Isto tanto permite especificar o processo de fabricação quanto às possíveis perdas ou rendimentos de produção para uma dada especificação de componentes.

Circuitos integrados monolíticos tem na literatura uma modelagem similar porém centrada na geometria dos componentes [Michael, Rankin]. No processo de fabricação destes circuitos a **pastilha** assume a mesma função do **lote** de circuitos discretos, estabelecendo-se diferentes variações entre componentes vindos de diferentes pastilhas, estas podendo vir de diferentes **lâminas** de silício que por sua vez podem vir de diferentes **linhas de produção**. Os componentes dentro de uma mesma pastilha tem seus parâmetros fortemente correlatos, de tal forma que o **descasamento** (mismatching) entre componentes de uma pastilha é destacado da variação que podem sofrer fora desta. O componente integrado tem seus parâmetros definidos individualmente pela sua **geometria** e pelo **processo de fabricação** local ao ponto da pastilha onde se situa. Por exemplo, quanto maior a área de transistores MOS e menor a distância de separação entre eles, menor o efeito das variações locais do processo de fabricação no descasamento, como é indicado na equação (37), dada em [Michael]:

(37) $$\Delta p(x,y) = t / A + | \, r \bullet d \, |$$

Nesta equação as variações relativas dos parâmetros dos componentes $\Delta\mathbf{p}$, devido à flutuações do processo de fabricação, dependem segundo um fator escalar **t** de um parâmetro geométrico do componente, tal como a área ou perímetro representados por **A,** e também da separação entre eles dada vetorialmente por **r**, segundo um fator de escala também vetorial **d**. O fator geométrico dominante depende do parâmetro e do componente, por exemplo o ganho de corrente ß de transistores bipolares laterais depende basicamente do perímetro do emissor.

O projeto de circuitos integrados consiste em definir a geometria dos componentes para realizar as características elétricas desejadas, sendo portanto a geometria o único fator de projeto de circuitos integrados quando o projetista adota uma tecnologia de fabricação de uma "foundry". O projeto estatístico de circuitos integrados exige que esta "foundry" forneça ao projetista os parâmetros t e r como variáveis aleatórias, isto é através das respectivas distribuições estatísticas necessárias à análise de Monte Carlo.

A modelagem estatística de circuitos integrados pode ser realizada em um nível mais básico, do processo de fabricação, onde as relações entre parâmetros de processo de fabricação e os parâmetros elétricos dos componentes ou variáveis de projeto são estabelecidas com base em simulações numéricas, geralmente do tipo de **elementos finitos**. [Hoensch, Hansen]. O custo de simulação deste tipo quando aplicada a um circuito inteiro é proibitivo, por ser o esforço computacional necessário muito grande. Este método é mais adequado quando aplicado à criação de modelos de componentes isolados, para serem usados na simulação elétrica dos circuitos. A grande limitação deste tipo de simulação no entanto é que as informações quanto ao processo são geralmente proprietárias, de acesso muito limitado. Daí ser mais adequado a projetistas realizar e utilizar a **modelagem a nível de componentes**, que pode ser obtida por medidas sistemáticas em amostras de circuitos e pelo uso de programas de extração de parâmetros, baseados em algoritmos de otimização ou programação não linear. Neste tipo de modelagem, adotada neste trabalho, os parâmetros elétricos dos componentes e circuitos são variáveis aleatórias cujos modelos estatísticos são funções de densidade de probabilidade.

2.5 Modelos estatísticos para projeto de circuitos com tolerância

A **probabilidade** de uma variável aleatória x assumir valores dentro do intervalo $[X_1, X_2]$ é dada pela **integral definida da densidade de probabilidade** :

$$\textbf{(28)} \qquad P(X_1 \le x \le X_2) = \int_{X1}^{X2} (f_x(x))\, dx \le 1 \qquad \textbf{(7)} \qquad P(-\infty \le x \le +\infty) = 1$$

As distribuições estatísticas são funções **normalizadas** que modelam o fenômeno estatístico, dando a freqüência de ocorrência dos possíveis valores da variável aleatória que representam. Algumas dos modelos mais usados são dados abaixo em função da média μ e do desvio padrão **s** :

(29) Distribuição uniforme : $f_x(X) = (2\sqrt{3}\,s)^{-1} \quad \forall\, x \in [\mu - \sqrt{3}\,s,\ \mu + \sqrt{3}\,s]$

$= 0 \quad \forall\, x \notin [\mu - \sqrt{3}\,s,\ \mu + \sqrt{3}\,s]$

(30) **Distribuição gaussiana :** $f_x(X) = (2\pi s^2 \exp((x-\mu)/s)^2)^{-1/2} \quad \forall x \in (-\infty, +\infty)$

Um modelo de maior precisão para componentes selecionados é a distribuição gaussiana truncada em 3 desvios padrão :

(31) $f_x(X) = (2\pi s^2 \exp((x-\mu)/s)^2)^{-1/2} \quad \forall x \in [\mu - 3s, \mu + 3s]$

$= 0 \quad \forall x \notin [\mu - 3s, \mu + 3s]$

O lote dos componentes restantes da seleção de componentes mais precisos pode apresentar uma **distribuição bimodal** como a dada abaixo em função de uma distribuição genérica f_x :

(32) $f_x(X) = (f_x(X - d/2) + f_x(X + d/2)) / 2$

A **função multinomial ou histograma** é a mais flexível forma de modelo estatístico, sendo dada por

(33) $f_x(X) = C_i \ , \forall X \notin [X_{i-1}, X_{i+1}] \ i = 1, \ldots n \ , X_1 = X_{min} \ , X_n = X_{max}$

Para representar variações de várias variáveis estatisticamente dependentes se usa funções como a conhecida **gaussiana multivariável** [Wilks, pag.163]. Porem neste trabalho são usadas apenas funções de distribuição estatística de uma variável, programadas em linguagem de alto nível [CERN_LIB, Knuth]. O modelo de dependência estatística adotado neste caso é o linear, segundo funções do tipo :

(34) $p_1 = p_{10} * (1 + r_1 * e_{11} + r_2 * e_{12} + \ldots)$

$p_2 = p_{20} * (1 + r_1 * e_{21} + r_3 * e_{23} + \ldots)$

onde e_{ij} são os coeficientes da tolerância do parâmetro i, p_{i0} os valores nominais e r_j os valores das variáveis aleatórias.

Este modelo atende satisfatoriamente a maioria dos casos de análise estatística, aonde modelos mais precisos são realizados com um maior número de termos. Entretanto podem existir casos de necessidade de modelos mais complexos, o que pode ser conseguido com outras funções, também programadas em alto nível tal como os modelos básicos das distribuições gaussiana e uniforme.

Estes modelos estatísticos podem ser ampliados para incluir outras variáveis aleatórias, tais como a taxa de **envelhecimento** no tempo x_t, e a taxa de **variação com a temperatura** x_T :

(35) $p = po * (1 + c * x_U + (1 - c) * x_L) * (1 + x_t * t) * (1 + x_T * T)$

Tratar tempo e temperatura como variáveis aleatórias em análise de Monte Carlo apresenta a vantagem de não aumentar significativamente o número de simulações necessárias para alcançar uma dada precisão, pois significa a inclusão de apenas duas em um grande número variáveis.

2.6 - A análise estatística.

Circuitos eletrônicos processam a informação contida em sinais elétricos segundo especificações, codificadas na linguagem dos seus usuários e projetistas. Pode-se observar, nos dois exemplos dados a seguir, como estas especificações podem variar de uma aplicação para outra, ou pelo menos variar o grau de importância dos atributos.

Um amplificador de carga para detetor de tiras de silício necessita ter :

- massa e volume mínimos.
- dissipação elétrica inferior a 1,5 mW.
- tempo de resposta inferior a 2 ns.
- ganho mínimo de 0,1 volt por picoCoulomb.
- linearidade de ganho dentro de 1%, até o limite de tensão de saída de 1 volt.
- impedância de saída entre 70 e 78 ohms
- nível de ruído máximo equivalente a 1000 elétrons, para um detetor de 5 pF.

Um filtro de sinal de voz precisa ser projetado tendo :

- tensões de alimentação : ±15 volts ± 10%.
- distorção inferior a 0,1 % para saída senoidal de 10 V de pico a 1 kHz, sobre 600 ohms.
- ganho de tensão na banda passante de 30 db com tolerância de 0.5 db.
- característica passa-baixa Butterworth de sexta ordem, freqüência de corte para grandes sinais entre 20 e 22 kHz.

Para fins de projeto estas especificações devem ser apresentadas aos programas que auxiliam o projeto de circuitos. Para isto elas devem ser transformadas em relações matemáticas como a equação (37), onde as tolerância dos componentes, dos estímulos e das respostas do circuito são incluídas nos domínios de variáveis **P, X, Y:**

$$\mathbf{(37)} \qquad \mathbf{y = f_S(x,p)\ ,\ y \in Y \subset R^m\ \ \forall\ x \in X \subset R^n\ ,\ \forall\ p \in P \subset R^l}$$

Os domínios das variáveis substituem os valores nominais do projeto clássico. Assim se espera como resposta de um circuito uma das funções dentro de famílias de funções $\mathbf{f_S(x,p)}$ que varrem o domínio **Y**. Os estímulos são representados por variáveis independentes restritas aos domínios **X**, que podem ser definidos por famílias de funções. Os valores dos componentes ficam por projeto contidos em **P** de modo a atender à expressão (37).

As variáveis **x**, podem ser determinísticas ou aleatórias, e representam grandezas elétricas (tensão, corrente, potência), ambientais (temperatura, pressão, luz) ou temporais (tempo, freqüência, fase). As variáveis **y** podem ser grandezas elétricas ou temporais, aleatórias ou não de acordo com a natureza de **x** ou **p**. No caso de variáveis aleatórias, a informação da freqüência de ocorrência dos valores é incluída junto com a do domínio, sob a forma de funções de distribuição estatística, enquanto para variáveis determinísticas apenas o intervalo entre os valores máximo e mínimo de cada variável é considerado.

Projetar um circuito para operar de acordo com suas especificações é escolher uma arquitetura e componentes tais que as variáveis de saída do circuito projetado, **z**, coincidam com os da resposta esperada **y**, conforme relações do tipo :

(38) $$z = f_D(p,x) \in Z \subset Y \subset R^m , \forall x \in X \subset R^n, \forall p \in P \subset R^p$$

Visto desta forma, o projeto de circuitos se baseia em um modelo matemático que relaciona domínios de parâmetros e domínios de respostas esperadas. Na análise se associa o domínio permitido à resposta do circuito, para cada estímulo previsto, a partir do domínio estabelecido para os parâmetros dos componentes. Na síntese se busca o domínio dos parâmetros a partir do domínio das respostas admissíveis. O critério adotado em cada um destes casos geralmente visa maior desempenho, menor custo de produção, ou um determinado compromisso entre os dois, estabelecido com uma função custo como :

(39) $$c = f_C(p,y,u) = [C_B + \Sigma_i (C_i / t_i)] \bullet P / Y$$

onde a função custo total de produção $f_C(p,y,u)$ depende inversamente da tolerância dos componentes t_i, e do rendimento de produção **Y** de circuitos com desempenho **P**, que representa por exemplo a rapidez, a relação sinal/ruído ou o consumo do circuito em projeto.

O rendimento de produção **P** é dado pela integral do produto de todas as funções de teste, uma para cada especificação.

Com a análise do circuito se verifica se na síntese foram atendidos os objetivos do projeto, ou seja se o modelo f_D e os valores de **p** são válidos segundo os resultados das simulações dos testes de aceitação dos circuitos. O objetivo da análise estatística de circuitos é estabelecer modelos estatísticos para as variáveis de circuito de interesse para o projetista. Consiste na simulação de um conjunto representativo de amostras de circuitos, na observação da freqüência de ocorrência dos valores das citadas variáveis, no estabelecimento da distribuição estatística correspondente, e no cálculo de parâmetros tais como média e desvio padrão. No caso de circuitos complexos, a análise é feita com base em métodos numéricos e técnicas de simulação. Existem diversos programas de simulação de circuitos, porém o SPICE [Nagel, Cohen, Vaz89, Vinci] é hoje um padrão, dadas as inúmeras versões comerciais e acadêmicas, e por isso foi escolhido para ser usado neste trabalho.

O SPICE se baseia no **método nodal modificado** [Nagel, Chua&Lin], que usa o fato de ser nula a soma das correntes incidentes em cada nó do circuito, para calcular todas as tensões dos nós, **v**, e as correntes nas fontes ideais de tensão, **i**. A partir destas variáveis são calculadas as variáveis de saída do circuito, **z**. Os estímulos **x** são representados por fontes independentes e ideais de corrente **j** ou tensão **e** do circuito. Cada componente do circuito deve ser descrito por equações que relacionem suas correntes com as tensões entre seus nós terminais. Estas equações são então incorporadas às equações nodais correspondentes. Dessa forma, qualquer circuito puramente resistivo, linear e ativo, pode ser descrito por equações do tipo :

(40) $$Y * v + A * i = j$$

$$B * v + Z * i = e$$

onde **Y** é a matriz de admitância nodal, **A** matriz de incidência das correntes **i**, devidas às fontes ideais de tensão e as fontes de corrente controladas por corrente. **B** e **Z** representam as matrizes de malha para as fontes de tensão ideais, independentes ou controladas por tensão ou corrente [L.Chua, p.131, Nagel].

Portanto, um circuito com **n** nós e **m** fontes de tensão ideais, é modelado por m+n variáveis dependentes e um igual número de equações. A solução destas equações é obtida no SPICE por meio do **algoritmo de decomposição LU com pivotagem por coluna**. Quando grande parte dos elementos de **A** são nulos, são usadas técnicas especiais mais eficientes para solução de **sistemas esparsos** [Nagel]. Métodos implícitos podem ser usados para refinar a solução obtida, para indicar ou limitar o erro numérico da solução [Blum]. Na execução do algoritmo de decomposição LU, quando o sistema é singular ocorre um pivô nulo, ficando a análise interrompida. Isto pode ocorrer não só em circuitos instáveis como um oscilador, mas também para sistemas não singulares porém mal condicionados, onde haja cancelamento de correntes ou existência de impedâncias tão altas que resulte na ocorrência de um pivô de valor inferior ao limite considerado como zero pelo algoritmo. A solução destes problemas consiste na modificação do circuito ou na reformulação da análise. Devido a isto, todo programa de análise estatística que use o SPICE deve considerar a possibilidade de resultados falsos ou incompletos, considerados na estatística como **"outliers"**.

Quando o circuito não é linear as matrizes **Y**, **B**, **A** e **Z** da equação (40) passam a ser dependentes de **v** e **i**. Este sistema de equações não lineares é resolvido no SPICE pelo método de Newton-Raphson [Chua&Lin, Nagel], a partir de uma **estimativa inicial** do ponto de operação do circuito, que deve estar suficientemente próxima da solução desejada. O sistema de equações é linearizado em torno deste ponto inicial, e se o problema admite uma solução, esta gera uma segunda estimativa, e assim sucessivamente até que seja atendido um critério de parada por limite de erro ou seja atingido um número limite de iterações. A condição necessária para a análise é que o sistema de equações admita pelo menos uma solução, e que o algoritmo adotado convirja para esta solução. Daí a importância da formulação destes sistemas por **equações lineares por partes**, no qual se baseia o método de Katzenelson [Chua&Lin,p.299], que tem garantida a solução do sistema de equações (40). Esta abordagem permitiu a L.Chua resolver o difícil problema de simular sistemas dinâmicos não lineares caóticos usando o programa NOEL [Chua_NOEL]. Outras propostas de formulação de equações lineares por partes aplicável à análise de circuitos eletrônicos podem ser encontradas em [Bokhoven, Güzelis].

Para circuitos **não lineares dinâmicos**, a solução depende de métodos **de integração numérica**, onde as matrizes do sistema são recalculadas a cada **passo** da integração, tendo em vista os valores das cargas armazenadas nos capacitores e os fluxos magnéticos armazenados nos indutores, o que na literatura é tratado como o **modelo de acompanhamento** do circuito para o método de integração numérica adotado. Com isso a solução é obtida com os mesmos algoritmos de solução de sistemas lineares. [Chua&Lin]

O caso de sistemas dinâmicos não lineares apresenta alguns problemas adicionais, que podem ser difíceis de se resolver, tal como o de encontrar todas as **possíveis múltiplas soluções** para (41), ou definir as **condições de estabilidade do sistema**. Daí a importância da escolha adequada do **método de análise**. Por exemplo a análise em regime permanente de pequenos sinais (AC) pode não convergir para uma solução na freqüência de ressonância de um oscilador, enquanto a análise de transiente pode dar o resultado correto, se forem usadas as condições iniciais e escalas de tempo corretas. A **análise de sensibilidade da resposta em cada simulação** é neste caso uma das ferramentas mais importantes no projeto de circuitos, pois permite prever o erro dos valores calculados para a resposta [Chua&Lin].

2.7 - A síntese estatística.

A síntese para projeto de circuitos eletrônicos e para modelagem de componentes e circuitos foi discutida nas seções 2.2.4 e 2.2.6. Ambas são essenciais ao projeto de circuitos eletrônicos com garantia de qualidade, e dada a natureza do problema, devem se basear em métodos estatísticos. O **tema** entretanto, quando tratado a nível da implementação da ferramenta correspondente para auxílio a projeto, se revela muito **amplo e complexo** para poder ser tratado adequadamente no âmbito desta tese. Porém este trabalho de desenvolvimento foi iniciado com a formulação de uma estratégia para ser implementada em um algoritmo com o uso de subrotinas da CERNLIB e do MINUIT, e se encontra descrito no **Apêndice I**, visando estimular o surgimento de possíveis trabalhos nesta área.

No capitulo seguinte se descreve como o SPICE foi usado para compor o simulador estatístico para circuitos analógicos **CPSPICE**, de modo a orientar a implementação de outros simuladores de Monte Carlo.

CAPÍTULO III - CPSPICE - SPICE PARALELO SOB CPS

3.1. INTRODUÇÃO - ESTRUTURA DO CPSPICE

0 **CPSPICE** é um simulador estatístico de circuitos eletrônicos desenvolvido nesta tese pela integração do simulador **SPICE** [Nagel, Vaz89] com o programa PAW - Estação de Trabalho para Física [CERN_PAW], a biblioteca de programação paralela **CPS** - SOFTWARE DE PROCESSOS COOPERATIVOS - [Fausey, Miranda] e a biblioteca de tratamento estatístico de dados CERNLIB [Brun, CERN_LIB]. As simulações são realizadas segundo métodos de **Monte Carlo** [Hammersley, Rubinstein, Spence]. CPSPICE cria e simula um determinado número de amostras a partir dos modelos estatísticos dos componentes ou fontes de sinais. Em cada amostra os valores dos parâmetros de componentes e sinais e modelos de componentes são gerados aleatoriamente obedecendo as respectivas distribuições estatísticas. As amostras são simuladas, os resultados analisados para estabelecer as distribuições estatísticas das variáveis de circuito de interesse do projetista.

Uma cópia do programa SPICE é executada para simular cada amostra. Ao fim da execução do CPSPICE os resultados das simulações são resumidos em um único arquivo de dados. Com o programa PAW pode-se então realizar a visualização e análise dos resultados, através de histogramas e outros tipos de gráficos.

0 CPSPICE é constituído por três programas diferentes, cada qual associado a uma **classe** de programas executáveis como um ou mais processos em um ou mais computadores. Assim a produção das amostras de circuitos é feita por um único programa **CLASSE1.c** que constitui a classe 1. As chamadas ao SPICE constituem a classe 2, com múltiplos processos executáveis em paralelo, cópias do programa **CLASSE2.f**. O tratamento dos resultados das simulações é feito por um único processo da classe 3, correspondente ao programa **CLASSE3.**

Estes programas trocam mensagens e dados através do software CPS, cujas rotinas sendo chamadas por estes programas possibilitam o paralelismo de todo o sistema em processadores diferentes dentro de uma mesma rede local, restrição esta imposta pela versão usada do CPS. Atualmente existem recursos de "thread" no UNIX e linguagens como JAVA e PEARL que permitem o paralelismo em bases mais amplas, envolvendo computadores via Internet em uma escala mundial, tal como vem sendo feito no LAFEX;CBPF dentro da colaboração internacional D0 [D0_RJS].

O CPSPICE foi realizado em duas versões, que diferem apenas nos métodos utilizados para a transferência de informações. A primeira versão, denominada **TDM**, faz **transferência de dados por memória** e concatena todos os resultados das simulações em um vetor, a medida em que essas vão sendo realizadas. Esse vetor é então transferido por memória para o último processo, correspondente à classe 3, onde os dados são reorganizados de forma a ser utilizados pelo programa PAW. Na segunda versão há a transferência apenas dos nomes dos arquivos que são gerados pelo SPICE ao final das simulações. Na etapa final (classe 3) os resultados das simulações devem ser extraídos destes arquivos, e por isso denominou-se essa versão de **transferência de dados por arquivo ou TDA.**

As seções seguintes descrevem detalhadamente cada etapa da simulação estatística de circuitos eletrônicos, para servir de referência para implementações de outros simuladores pelo método de Monte Carlo. Inicialmente, será apresentado o pacote CPS [Fausey, CPS_U, CPS_R], para ilustrar os conceitos da computação paralela usada na simulação pelo SPICE, e que estão inseridos em sua estrutura. Entretatno outros recursos podem ser usados para tal fim, como o PVM [PVM] ou mesmo os recursos de multiprocessamento do sistema operacional UNIX, como o "thread" [UNIX]. O processo de geração de amostras é apresentado juntamente com a biblioteca CERNLIB, especialmente o programa HBOOK [CERN_LIB, CERN_HBO, CERN_PAW]. O processo final, de criação da estrutura de dados com resultados, realizado pelo programa CLASSE3, fica em um capítulo a parte, pois para sua perfeita compreensão é necessário que haja um prévio conhecimento de todas as partes que compõem o CPSPICE : SPICE, CPS, CERNLIB e PAW.

3.2. CPS - SOFTWARE DE PROCESSOS COOPERATIVOS

O CPS é um pacote de programas desenvolvido pelo Fermilab e LAFEX [Ávila, CPS_R, CPS_U, Fausey, Miranda], que permite que uma **tarefa** computacional seja distribuída através de múltiplas UCP - unidade central de processamento. Para isso o CPS fornece ao usuário os meios necessários para fazer chamadas remotas a subrotinas, sincronismo de processos, utilização de filas de processo, gerência de mensagens e de transferência de dados. O produto final se traduz em uma utilização eficiente e em paralelo dos recursos computacionais, objetivo principal do CPS.

O ambiente para execução da versão do CPS que foi utilizada neste trabalho deve ter minimamente a seguinte configuração:

- plataformas possíveis, ligadas em rede TCP/IP : IBM RS6000 (AIX) - Silicon Graphics (IRIX) – DEC (Ultrix) - SUN (SUN O/S) - HP (UX) - MIPS (Risc O/S) – ACP R3000 (Risc OiS)
- espaço disponível para dados: 16 MB de memória RAM - de 30 MB a 2 GB de espaço livre em disco - de 35 MB a 200 MB de espaço livre para "swap" de disco

Um **tarefa** do CPS consiste de processos de três tipos, que se intercomunicam : Gerenciador de Tarefa ou **JM - Job Manager**, Gerenciador de Compartilhamento de Memória ou **SHM - Share Memory Manager**, e os Processos do Usuário ou **UP - User Process**.

O **JM** é o primeiro processo iniciado, responsável por iniciar, administrar e encerrar todos os outros processos. Uma vez iniciados todos os **UP**, o JM ativa um processo que dá partida aos **SHM**. Esse processo na verdade é uma divisão do **JM**, que funciona como dois processos, um sendo o processo pai e o outro o processo filho. O processo pai manuseia todas as saídas produzidas pelos **UP**. O processo filho inicializa cada SHM (um por computador), e então gerencia a tarefa, transmitindo mensagens recebidas dos **UP**, através dos **SHM**. Existe um limite de 256 processos por cada tarefa do CPS.

O **JM** e os **UP** não se comunicam diretamente. Os **UP** enviam mensagens para o **JM** segundo seus endereçamentos, as entregando primeiramente para o **SHM** local. O **SHM** então enviará as mensagens para **JM**. Da mesma forma, o **JM** envia mensagens para os **UP**,

endereçando-as e entregando-as para o **SHM** localizado no mesmo computador do **UP**. O **SHM** irá então entregar a mensagem. A figura 1 apresenta esse mecanismo.

Uma tarefa no CPS é dividida em tarefas lógicas relativas à aplicação do usuário. Por exemplo, a tarefa inicial do CPSPICE consiste na leitura e interpretação de dados lidos de um arquivo. A tarefa seguinte é simular amostras do circuito segundo os dados extraídos, enquanto que a tarefa final consiste em gravar os dados resultantes das simulação em uma unidade remota de disco ou fita. A tarefa executada com o CPS conterá o número apropriado de processos para realizar todas as tarefas usando ao máximo os recursos computacionais disponíveis, processos estes que se distinguem um dos outros por suas classes.

Todos os processos de uma mesma classe rodam o mesmo programa no mesmo tipo de computador. O número de classes em cada tarefa é limitado a trinta. Cada classe tem um programa escrito em linguagem FORTRAN ou C, que gera um ou mais processos. Cada processo pode usar todas os recursos do sistema operacional disponíveis na CPU onde estiver sendo executado. É permitido o acesso a periféricos, mas quantidade excessiva de escrita ou leitura de dados pode tornar mais lenta a execução da tarefa.

Cada programa deve ser escrito e guardado em um arquivo texto, compilado, e ligado com os outros arquivos objetos das outras classes e de bibliotecas associadas. A compilação e ligação devem ser feitas sob o mesmo sistema operacional em que o processo resultante irá executar. Desse modo, o usuário pode usar todas as opções dos utilitários associados ao sistema. Não é necessária uma compilação especial - a única necessidade é que o programa seja ligado com a biblioteca de rotinas do CPS e que tenha comandos de chamada às subrotinas do CPS.

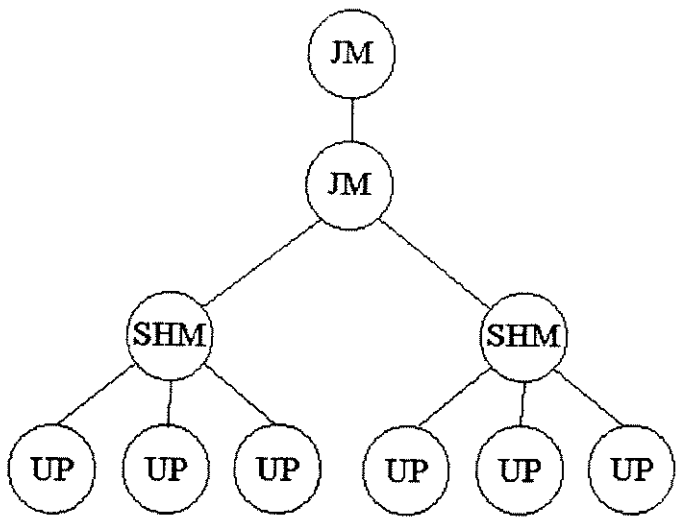

Figura 1 - Estrutura de execução de uma tarefa por CPS

Para executar uma tarefa, o usuário primeiramente prepara um Arquivo de Descrição de Tarefa (JDF - Job Description File), que informa ao sistema onde o arquivo executável de cada classe se encontra, quantos processos devem rodar por classe, qual o tipo de CPU usada para cada classe, e quantas CPU's irão ser utilizadas em cada uma das classes.

O JM, então, supervisiona qualquer fila e pontos de sincronismo utilizados, monitora as tarefas e verifica processos com problemas, e finalmente espera que algum processo do usuário chame a subrotina de final de tarefa, quando então supervisiona a parada de todos os processos.

Finalmente, o JM administra o arquivo de observação (log) de execução da tarefa, listando todos os processos que foram iniciados, a saída padrão para cada processo, escrevendo informações sobre qualquer erro que venha a ocorrer.

Todo processo em uma tarefa do CPS é um programa escrito pelo usuário. Estes processos podem ser distribuídos através de várias CPU's em uma rede. Por exemplo, considere uma tarefa com quatro processos.

Todos os processos poderiam ser executados em apenas uma CPU (figura 2a), ou poderiam também ser executados em duas CPU's. com dois processos compartilhando a mesma CPU (figura 2b) ou ainda ser distribuídos em quatro CPU's (figura 2c)

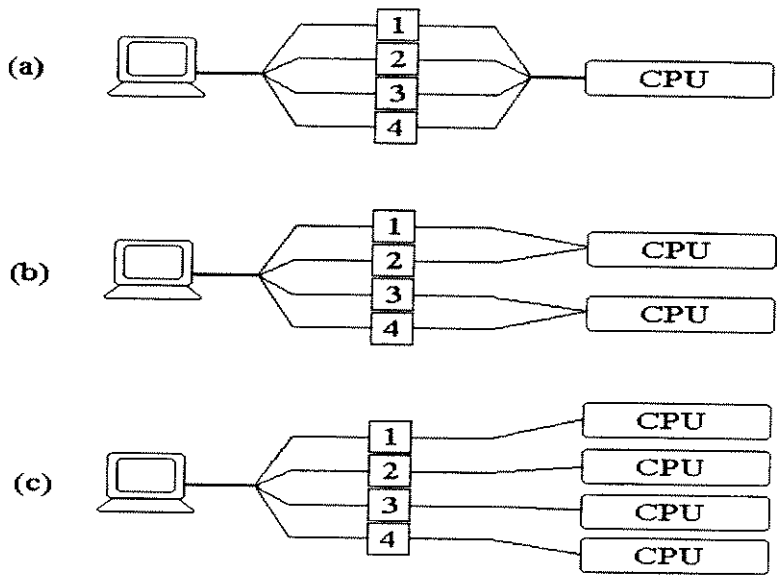

Figura 2 - Distribuições de tarefas pelo CPS

Para iniciar um **JOB** do CPS é necessário um arquivo **JDF** (Job Description File), que fornece ao JM parâmetros de controle tais como quantos processos iniciar, onde encontrar os programas escritos pelo usuário e, opcionalmente, quais os tipos de programas que devem ser executados em CPU's específicas. A utilização ótima do hardware distribuído pelo CPSPICE é conseguida através deste arquivo que controla a execução de todos os processos distribuídos. Nas referências sobre o CPS [Avila, CPS_U, CPS_R, Fausey] são apresentadas diversos aspectos da estruturação de uma tarefa a partir de um arquivo JDF, incluindo exemplos de arquivos para aplicações específicas.

O sincronismo é essencial para aplicações que utilizem processamento em paralelo, devido aos inevitáveis atrasos que podem ocorrer nos processos. O CPS tem a capacidade de criar pontos de sincronismo a partir do código escrito pelo usuário. O modo mais fácil de se obter sincronismo é definir, nos programas fontes do usuário, um ponto de sincronismo que

todos os processos devem alcançar antes de continuarem (figura 3). A função **cps sync()** permite o estabelecimento de até 128 pontos de sincronismo.

As filas são um mecanismo de passagem e controle dos processos. Uma fila consiste em uma lista de processos que estão em algum estado definido. O uso de filas de processos *(process queue)* é básico para controlar a confluência dos processos executados independentemente, de forma assíncrona e em paralelo.

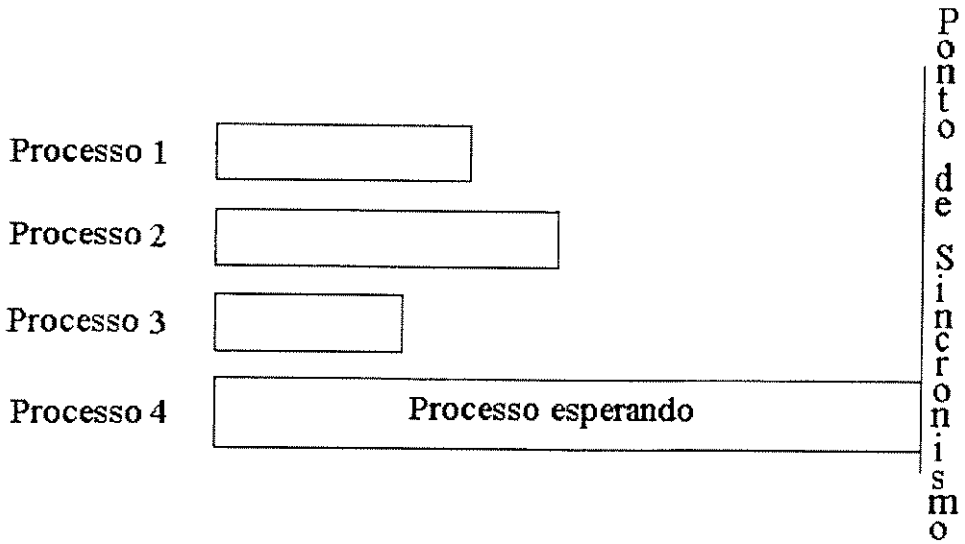

Figura 3 - Sincronismo de Processos no CPS

Um processo pode colocar-se a si mesmo em uma fila ou lá ser posto por um outro processo. Geralmente um processo vai para uma fila quando está pronto para realizar algum tipo de serviço, como por exemplo, analisar um evento, fornecer resultados para a reconstrução de um evento, escrever em uma fita, etc.

Quando um processo precisa se unir a outro processo, associado a uma fila, para realizar um serviço, ele o remove da fila (process dequeue) para poder passar os dados e ativá-lo fazendo uma chamada remota a subrotina. Quando o processo servidor estiver pronto novamente para realizar o serviço, ele é posto de volta na tila. É completo o controle de colocação e retirada de um processo em uma fila. Por exemplo, para configurar uma topologia do tipo Entrada / Análise / Saída, conforme ilustrada na figura 4, o usuário precisa declarar duas filas, digamos: LIVRE e FEITO.

Os processos utilizados para analisar os eventos devem ser colocados na fila LIVRE no início do **JOB.** O processo que lê os eventos remove estes processos da fila LIVRE para enviar através deles os os eventos para subrotinas chamadas remotamente. Ao final da execução da subrotina remota, o processo se põe a si mesmo na fila FEITO. O processo responsável pelo tratamento dos resultados (como por exemplo os escrever em uma fita. ou formatar dados) retira os processos da fila FEITO, obtém o evento, realiza sua tarefa, e coloca o processo de volta na fila LIVRE.

A estrutura acima foi utilizada neste trabalho. As filas LIVRE e FEITO correspondem respectivamente às filas **READY QUEUE** e **CPSPICE QUEUE.**

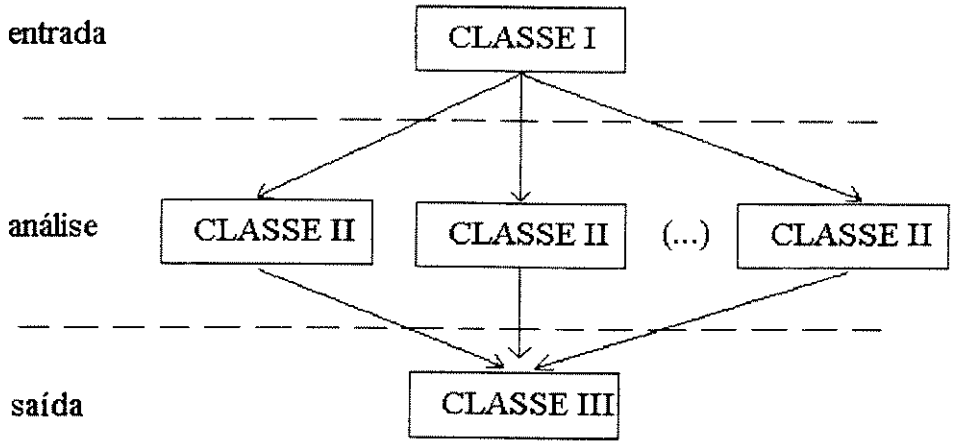

Figura 4 - Estrutura de Classes do CPSPICE

As principais rotinas do CPS utilizadas neste projeto [CPS_R, CPS_U] são as seguintes :

ROTINA	DESCRIÇÃO
cps_ini	Inicia o CPS e informa o JM que o processo está pronto. Primeira rotina a ser chamada.
cps_sync	Sincroniza todos os processos para iniciar neste ponto após todos chamarem esta rotina.
cps_declare_queue	Declara que o processo está apto a ser posto na fila especificada.
cps_queue_process	Coloca o processo na fila especificada.
cps_dequeue_process	Retira o processo da fila especificada.
cps_send	Envia dados para outros processos.
cps_get	Recebe os dados enviados por outros processos.
cps_declare_subroutine	Declara o processo servidor de rotinas remotas.
cps_call	Faz chamada a rotinas remotas.
cps_stop_process	Declara o processo encerrado.

Além do arquivo JDF, existem outros que auxiliam o CPS em sua execução. São eles:

ARQUIVO	DESCRIÇÃO
cpspice.jml	Arquivo de registro de todas as operações realizadas pelo CPS na execução da tarefa, um arquivo apenas para leitura.
cps_sdf	Listagem de todas as CPU que executarão a tarefa, com nome, tipo, número lógico assumido para a tarefa e o tipo da tarefa a executar.
.netrc	Arquivo do UNIX, lista as CPU, contas e senhas para execução remota da tarefa. Exemplo : machine view.sp1.lncc.br login mario password ********

3.3. AS CLASSES

As classes 1, 2 e 3 do CPSPICE, correspondem ao programas fonte **CLASSE1.c**, **CLASSE2.f** e **CLASSE3.f** ou **CLASSE3.c**. A descrição destes programas que se segue objetiva dar um maior detalhamento a este trabalho e servir de guia para outras implementações.

3.3.1 O PROGRAMA CLASSE1 E A GERAÇÃO DE AMOSTRAS

O programa **CLASSE1,** cujo arquivo fonte **(CLASSE1.c)** foi escrito em linguagem C ANSI, é responsável pela geração das amostras do circuito fornecido pelo usuário. A descrição do circuito em arquivo texto padronizado para o SPICE deve se chamar **original.cir.** Neste arquivo, exemplificado abaixo, devem constar para os componentes ou parâmetros de componentes variáveis, as respectivas faixas de variação e tipo de distribuição estatística :

```
SCHIMITT TRIGGER
.OP
.TEMP 35 40
.TRAN 5NS 50NS
.DC VIN -0.25 0.25 0.05
.TF V(5) VIN
.NOISE V(5) VIN 50
.AC DEC 10 1 10MEG
.FOUR 20MEG V(5)
*#MC 10 5
VIN 100 0 AC 1 SIN(0 0.1 5MEG)
VCC 101 0 DC 12
VEE 102 0 -12
Q1 4 2 6 QNL 1$10%U2$2%U3$
Q2 5 3 6 QNL 1$10%U2$2%U4$
RS1 100 2 1K$10%G1$
RS2 3 0 1K$10%G2$
RC1 4 101 10K$5%U1$
RC2 5 101 10K$5%NU1$
Q3 6 7 102 QNL 1$10%U2$2%U5$
Q4 7 7 102 QNI 1$10%U2$2%U6$
RBIAS 7 101 20K CLOAD 4 5 5PF
.MODEL QNL NPN (BF=80$20%G3$ RB=100$30%G4$ CCS=2PF
+TF=0.3NS$20$G5$ TR=6NS CJE=3PF CJC=2PF VA=50) .PRINT NOISE
+ INOISE ONOISE . PRINT TRAN V(4) V(5) .PRINT AC V(5) VP(7) VM(7) .PRINT DC V(6)
V(7)
.END
```

Neste exemplo nota-se que o modelo **QNL** (24-' linha) tem o parâmetro BF com valor nominal de 80, faixa de variação de **20%,** e distribuição estatística do tipo Gaussiana (representada pela letra G - ver tabela abaixo). Isso significa que o parâmetro BF pode assumir valores dentro de um modelo de distribuição Gaussiana, que está representado na

figura 5, com desvio padrão de 20% do valor nominal. O CPSPICE aceita os seguintes tipos de distribuição, identificados pelos respectivos símbolos: gaussiana Unimodal (G), Gaussiana Bimodal (B), Uniforme Unimodal (U), Uniforme Bimodal (W), Histograma (H).

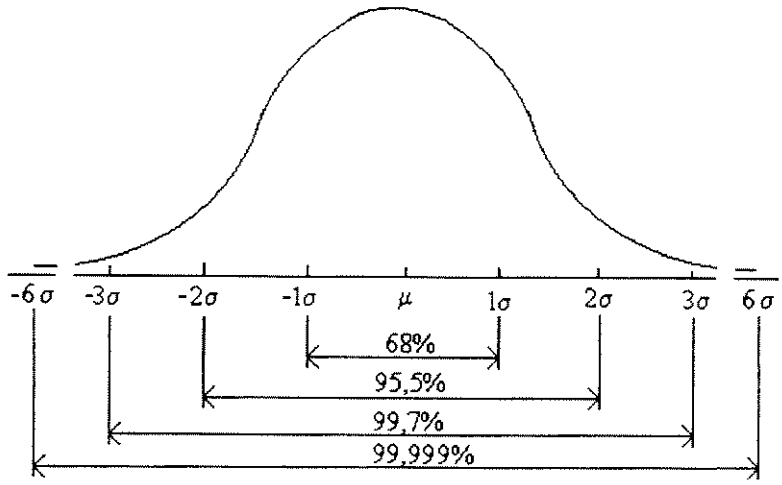

Figura 5 - Função de densidade de probabilidade de uma distribuição gaussiana

No exemplo dado se observa que uma **variável aleatória** é identificada por um símbolo seguido do nome identificador e parâmetros da distribuição. No caso dado **G1, G2. G3, U1** e **U2** são nomes de diferentes variáveis aleatórias tendo Gl, G2 e G3 distribuições Gaussiana e Ul e U2 distribuições uniforme. É importante lembrar que todas são variáveis aleatórias independentes. Caso os componentes ou parâmetros de componentes apresentem correlação estatística, utiliza-se a mesma variável estatística no modelo dos mesmos, como é o caso dos resistores RC1 e RC2, os quais apresentam correlação negativa, indicada pela letra **N** antes da Variável Aleatória U I para RC1 mas não para RC2, como se fossem resistores de derivação.

O cálculo dos valores dos componentes em cada amostra segue um modelo linear dependente de distribuições estatísticas :

(1) $$p = p0 \times (1 + tol(1) \cdot va(1) + tol(2) \cdot va(2) + ... + tol(n) \times va(n))$$

onde **p** é o novo valor do componente, **p0** é seu valor nominal, tol(i) é o valor da tolerância associada a Variável Aleatória va(i).

Assim por exemplo : Q3 6 7 102 QNL **1$10%U2$2%G5$**
indica que a área do transistor **Q3** varia **10%** de acordo com a variável U2, com distribuição uniforme, e **2%** de acordo com a variável gaussiana G5.

Para criar as amostras de um circuito o programa CLASSE1 pesquisa todas as linhas do arquivo **original.cir** e armazena aquelas que contenham Variáveis Aleatórias. O número de amostras que serão geradas dependem do comando **MC** (Monte Carlo), que define o número de amostras a serem geradas e o número de comandos **ALTER** presentes em cada amostra. O comando ALTER permite que qualquer dispositivo num circuito, podendo ser um

componente ou um parâmetro de componente. assuma um valor diferente do nominal, em uma nova simulação do circuito, no mesmo processo. Assim pode-se ter. por exemplo:

```
( )
RS1 100 2 963
.ALTER RS1 100 2 1045
.ALTER RS1 100 2 1002
.END
```

Do modo acima o SPICE realizará a simulação do circuito primeiramente considerando RS1=963 ohms. Logo em seguida simulará o mesmo circuito com R1=1045 ohms e depois novamente com R1=1002 ohms, três circuitos dentro de um mesmo processo, com apenas uma chamada ao SPICE. O equilíbrio entre a quantidade de processos diferentes a serem executados e o número de comandos ALTER por processo (ou seja, o número de comandos ALTER em um circuito) fará com que a execução total do CPSPICE seja mais eficiente quanto ao uso das CPU e do sistema de comunicação entre elas. O comando MC define o número de amostras a serem geradas e o número de .ALTER por amostra da seguinte forma:

#MC <número de amostras> <número de ALTER por circuito>

Após identificar todas as linhas relacionadas com a simulação estatística, o CLASSE1 entra em operação cíclica construindo dados para simulações incluindo comandos ALTER. Durante o processo de análise do arquivo **original.cir** o CLASSE1 armazena os nomes das Variáveis Aleatórias encontradas, e o valor calculado para ela, valor este que é usado ao longo de toda a amostra.

Para o cálculo dos valores de cada Variável Aleatória são usadas rotinas de geração de números aleatórios do HBOOK [HBOOK, PAW].

Na versão do CPSPICE que utiliza envio de dados por memória, todas as informações resultantes das simulações pelo SPICE pelo CLASSE2 são armazenadas pelo CLASSE3 em um único vetor, chamado **iocps**. O CLASSE1 escreve em *iocps* os nomes e valores das Variáveis Aleatórias,após a interpretação do **original.cir**, arquivo de dados para o CPSPICE.

Os nomes das Variáveis *Print* também são armazenados em *iocps*. Variáveis *Print* são tensões e correntes analisadas e impressas através do comando .PRINT do SPICE (com exceção das análises NOISE e DISTO - ver tabela a seguir). Essa foi a solução adotada no CPSPICE para o armazenamento desses nomes (como será visto no capítulo sobre o SPICE), que se dá de forma simples: a cada linha do arquivo **original.cir** analisada no CLASSE1, caso esta contenha um comando .PRINT, é feita uma chamada para a uma rotina que armazena os nomes das Variáveis *Print* encontradas. As Variáveis *Print* são sempre relacionadas a um tipo de análise. especificados no quadro a seguir.

ANÁLISE	DESCRIÇÃO
DC	imprime os resultados da análise DC realizada para a variável especificada. que pode ser uma tensão ou corrente:
AC	imprime tensões e correntes resultantes da análise AC, dadas pela parte real (VR ou IR), imaginária (VI ou II), magnitude (VM ou IM), fase (VP ou IP) ou dB (VDB ou IDB).

TRAN	imprime tensões e correntes resultantes da análise TRANSIENTE.
NOISE	imprime ONOISE (ruído na saída) e INOISE (ruído na entrada) tal como em AC.
DISTO	imprime HD2, HD3, SIM2, DIM2 ou DIM3 tal como em AC.

O vetor *iocps* é, em termos de programação em C, uma *union* composta de inteiros e caracteres. Isso permite que ambos os tipos sejam escritos em *iocps,* cuja estrutura é do tipo sequencial contendo os seguintes campos :

Texto do circuito - Variáveis Aleatórias - Variáveis *Print* - Resultados das simulações

O campo **resultados das simulações** é preenchido durante as simulações, sendo enviado para o SPICE apenas os três primeiros campos. Por ignorar tudo que segue o comando .END. os campos correspondentes as Variáveis *Print* e as Variáveis Aleatórias se tornam transparentes para o SPICE. que na prática recebe apenas a descrição do circuito a ser simulado.

Na versão do CPSPICE de transferência de dados por arquivo - TDA, existem muitas simplificações em relação a versão anterior. Primeiramente. não há a necessidade de *iocps.* Na medida em que as amostras de circuitos são geradas, elas são armazenadas na forma de arquivos texto, e apenas os nomes desses arquivos são transmitidos ao SPICE. Para cada ciclo de criação de amostra pelo CLASSE1 é criado um arquivo diferente. O processo de geração de Variáveis Aleatórias se mantém, assim como o de geração de amostras. Fica desnecessária a determinação das Variáveis *Print* pelo CLASSE1, tal como explicado adiante na seção onde se apresenta o programa CLASSE3.

O CLASSE1 está estruturado em termos de CPS na seguinte forma: a cada ciclo o CLASSE1 retira da fila READY o número do processo correspondente ao CLASSE2 **(cps dequeue process),** e faz uma chamada remota à subrotina **(cps call)** assim que acaba a geração de uma nova amostra do circuito. Na versão TDM. antes de fazer a chamada remota é necessário que o CLASSE1 envie o vetor *iocps,* através da rotina **cps send.** Na versão TDA toda a informação é enviada como argumento da rotina **cps call.**

Ao final da geração de amostras, quando os ciclos todos já foram encerrados. o CLASSE1 aguarda até que a fila READY fique cheia novamente, o que significa que o CLASSE3 terminou suas tarefas, e coloca um marcador de final de fila na fila CPSPICE, da qual o CLASSE3 retira informações, para que o CLASSE3 possa encerrar seu processo ao detetar que chegou ao fim desta fila. Abaixo se encontra uma tabela com as funções do CPS utilizadas no programa CLASSE1. na mesma ordem em que elas aparecem no seu arquivo fonte.

NOME DA SUBROTINA	DESCRIÇÃO
cps_init e cps_sync	iniciam e sincronizam as classes do CPS;
cps dequeue_process	retira o processo CLASSE2 da fila READY;
cps send	apenas na versão TDM, envia *iocps* para CLASSE2
cps call	faz a chamada remota a subrotina.
cps wait queue	aguarda que a fila READY fique cheia
cps queue~rocess	coloca o processo na Fila CPSPICE.
cps stop~rocess	última rotina chamada do CPS, encerra o processo

3.3.2. O PROGRAMA CLASSE2 E A SIMULAÇÃO PARALELA DAS AMOSTRAS

O programa CLASSE2 funciona como um **servidor de subrotinas remotas,** declarando subrotinas que poderão ser acessadas por outros processos. Neste projeto uma única subrotina chamada remotamente pelo CLASSE1 faz chamada ao programa SPICE, ao terminar a geração de uma amostra de circuito. Esta rotina inicia o SPICE e envia os dados da amostra de circuito.

Na versão TDM o CLASSE1 envia para a subrotina remota o vetor **iocps**, do qual para o SPICE receber os dados, ele teve algumas de suas rotinas fontes modificadas, como será visto adiante.

Na versão TDM é necessário declarar o bloco de dados que será utilizado para o transporte de *iocps.* Só assim será possível ao CLASSE1 enviar os dados para o CLASSE2, que após realizada a simulação no SPICE, onde os resultados são escritos em **iocps**, *o* envia para o CLASSE3. A rotina **cps_declare_block** é responsável pela declaração do bloco de dados. feita no processo servidor, que no caso é o CLASSE2.

Além do bloco de dados e das filas. é necessário que o CLASSE2 declare as subrotinas que pretende servir a outros processos (processos clientes, que no caso é o CLASSE1). A única subrotina em questão simplesmente inicia o SPICE, e foi denominada de **CPSPICE SUBROUTINE.**

O programa CLASSE2 precisa declarar-se apto a ser posto em uma fila por outros processos ou por ele próprio, o que é feito pela rotina **cps_declare_queue,** para ambas as filas utilizadas neste projeto (READY e CPSPICE). O programa não cria a fila, apenas associa um nome à fila, tornando o processo em questão apto a estar nesta fila. O CPS permite o uso de 128 filas diferentes, que existem antes mesmo de iniciada a tarefa.

Tendo declarado o bloco de dados, as filas e a subrotina. e estabelecido o ponto de sincronismo **(cps_sync),** a rotina **cps_service_calls** torna o CLASSE2 um servidor de subrotinas remotas. Isso faz com que o processo espere até que seja feita uma chamada a subrotina remota por outros processos. A seguir se encontra uma tabela com todas as rotinas do CPS utilizadas no CLASSE2. na ordem em que aparecem.

NOME DA FUNÇÃO	FUNÇÃO NO CLASSE3
cps init	deve ser a primeira rotina chamada, inicia o CPS.
cps_declare_subroutine	declara a subrotina CPSPICE SUBROUTINE, que o CLASSE2 servirá aos processos clientes.
cps_declare_queue	declara o CLASSE2 apto a ser colocado (e retirado) nas filas READY e CPSPICE.
cps_queue_process	CLASSE2 coloca a si próprio na fila READY, consultada pelo CLASSE1 sobre qual processo enviar dados.
cps_sync	ponto de sincronismo.
cps service calls	última rotina a ser declarada, a partir da qual o CLASSE2 fica servidor de subrotinas remotas

Na versão TDA, o CLASSE1 fornece a subrotina remota os nomes dos arquivos gerados, e esta rotina chama o SPICE para simular o circuito segundo o arquivo cujo nome foi fornecido. Com isso nenhuma rotina do SPICE precisa mais ser modificada. Isso possibilita aplicar este método a outros simuladores dos quais não se disponha do program fonte como o SPICE. Isto será discutido mais adiante, na seção sobre o CLASSE3.

Para cada amostra de circuito gerada pelo CLASSE1, se tem na versão TDA uma simulação do SPICE3f4 por um programa CLASSE2. Então todo o processo de simulação estatística envolve a execução de um programa CLASSE1 e um programa CLASSE3, com ciclos para a construção de amostras e recepção de resultados das simulações destas amostras, bem como a execução de programas CLASSE2 correspondentes à simulação de cada uma das amostras.

3.4. O PROGRAMA SPICE

Como explicado antes, alguns programas fontes do SPICE foram modificados para a versão TDM. Todas essas modificações se relacionam com o vetor **iocps**, que passa a funcionar como entrada e saída para o SPICE.

O arquivo fonte **spice.f,** escrito em linguagem FORTRAN 77, foi alterado para receber como entrada o vetor **iocps**. Abaixo segue um trecho deste arquivo, onde se pode notar que as partes comentadas são justamente referentes a forma de entrada do SPICE. O comeqo do programa também foi modificado para que o SPICE funcionasse como uma subrotina. a ser chamada pelo CLASSE2:

```
CPS' SUBSTITUIDO PROGRAM SPICE POR SUBROUTINE CPSPICE
CPS* PROGRAM SPICE
      SUBROUTINE CPSPICE
( )
CPS* DECLARAÇÃO DO COMMON UTILIZADO NO CPS, CPS IN E CPS OUT
      COMMON /CPS IO/LIN, LOU, CPSIO(10000)
CPS*
( )
C CHECK IF A RAW DATA FILE SHOULD BE WRITTEN
C OPEN FILE IF SO REQUIRED
CPS*  COMENTADO PORQUE OS DADOS VEM PELO COMMON E NAO PELO ARQUIVO
CPS*        ICOST=IARGC()
CPS*        IF(ICOST.EQ.O)GOTO 1
CPS*        CALL GETARG(1,OPTION)
CPS*        CALL GETARG(2,FILENAME)
CPS*        IPOSTP=IOPRAW(ICOST,OPTION//CHAR(0),FILENAME//CHAR(0))
( .)
```

Toda escrita em **iocps** é acompanhada de um identificador, aqui chamado de *rótulo* (etiqueta). Com isso, é possível saber, em uma análise posterior realizada pelo CLASSE3, a que se refere cada dado encontrado em *iocps.* Um exemplo pode ilustrar melhor esse mecanismo. Durante uma análise de correntes em fontes de tensão e tensões em fontes de corrente, para um circuito qualquer, o SPICE fornece o seguinte resultado:

```
VOLTAGE SOURCE CURRENTS
NAME CURRENT
VI N -7.668D-06
VCC -2.517D-03
VEE 2.532D-03
TOTAL POWER DISSIPATION 6.06D-02 WATTS
```

O arquivo fonte do SPICE responsável pelos cálculos e impressão desses resultados chama-se **dcop.f,** e está escrito em FORTRAN 77 para a versão 2G4 do SPICE. Nele o seguinte trecho foi acrescentado, para escrever em *iocps* os rótulos DCVC e TRVC para as análises TRANSIENTE (TRVC) ou DC (DCVC) :

```
( )
CPS* GUARDA ESPACO PARA NÚMERO DE PONTOS
      IF((MODE.EQ.1).AND.(MODE DC. EQ.1)) CPSIO(LOU+1) = 8HDCVC
      IF((MODE.EQ.1).AND.(MODE DC. EQ.2)) CPSIO(LOU+1) = 8HTRVC
      ICPSI= LOU+3
      LOU= ICPSI
CPS*
(..)
```

Os resultados são escritos em *iocps* conforme o seguinte trecho do programa :

```
CPS* NOME E CORRENTE DAS FONTES DE TENSÃO
      CPSIO(LOU+1) =VALU E(LOCV)
      CPSIO(LOU+2)=CREAL
      LOU= LOU+2
CPS*
( )
CPS* NOME E TENSÃO DAS FONTES DE CORRENTE
      CPSIO(LOU+1)=VALUE(LOCV)
      CPSIO(LOU+2)=CREAL
      LOU= LOU+2
CPS*
( )
CPS* NÚMERO DE FONTES E POTÊNCIA DISSIPADA TOTAL PARA CPS
      CPSIO(ICPSI-1)= POWER
      CPSIO(I CPS1) =(LOU-ICPS1)/2
CPS*
( )
```

Do mesmo modo, para outros tipos de análise, se acrescentam textos equivalentes. Desta forma, sempre que os resultados de tais análises forem simulados e impressos, eles também serão escritos em **iocps**, juntamente com os rótulos, para que o CLASSE3 possa identificá-los. Este ponto será novamente examinado no capítulo referente ao CLASSE3.

A alteração no arquivo fonte do SPICE foi feita para as seguintes análises :

- análise de tensões nodais: arquivo dctran.f;
- análise de correntes em fontes de tensão. tensões em fontes de correntes e análise de potência dissipada: arquivo dcop.f;
- análise da função de transferência do circuito, impedância de entrada e impedância de saída: arquivo sstf.f;

- análise de FOURIER e análise de distorção harmônica: arquivo fouran.f;
- análises DC, TRANSIENTE e AC: arquivo ovtpvt.f:

O comando TEMP do SPICE faz todas as análises serem repetidas para cada valor diferente de temperatura especificado, realizando novas simulações no mesmo processo, tal como no comando .ALTER que muda o valor de um parâmetro de modelo ou componente. Para cada temperatura diferente e para cada comando ALTER há portanto a necessidade de um rótulo identificador da nova simulação.

Para as variáveis Print correspondentes às análises DC, AC. TRAN, NOISE e DISTO, o modo pelo qual o SPICE armazena os nomes das variáveis analisadas (tensões nodais e correntes) faz com que o nome tenha várias partes, correspondentes à estrutura de armazenamento do nome no SPICE. O nome dessas variáveis não é uma estrutura lógica para o SPICE, apenas um conjunto de caracteres. Por isto, para fazer com que esses nomes sejam armazenados em **iocps,** se faz o armazenamento dos nomes durante o processo do CLASSE1, como citado anteriormente.

Um outro problema surgiu pelo fato dos nomes e valores serem armazenados em momentos diferentes, tendo que ser inseridos no momento da leitura de **iocps** pelo CLASSE3. Para garantir que os nomes sejam armazenados na mesma ordem dos resultados, como o SPICE imprime os resultados dos comandos .PRINT sempre na mesma ordem, independente da ordem de digitação no arquivo original contendo o circuito, o CLASSE1 constroi uma estrutura hierárquica no **iocps** para todos os comandos .PRINT encontrados.

Como se vê, os problemas que podem surgir quando se altera um programa simulador são variados, e desta forma apontam para que no desenvolvimento do simulador de Monte Carlo se use o programa original do simulador e se desenvolva apenas os programas necessários para processar os arquivos para entrada e saída de dados.

3.5. O PROGRAMA PAW

A simulação de Monte Carlo para ser precisa exige um grande número de amostras cuja simulação gera um grande volume de dados, que devem ser armazenados para posterior análise, através de visualização e cálculos para obtenção da estatística desejada. Este é precisamente o objetivo do programa PAW.

PAW é um sistema para análise interativa de dados, que integra vários pacotes e ferramentas para processamento numérico e apresentação gráfica de dados. Seu uso se dá em aplicações para Física Experimental, onde uma grande quantidade de dados são processados [Brun, CERNP], tal como nas simulações estatísticas de circuitos e sistemas eletrônicos. Por ter sido muito usado nesta área, tendo sido aperfeiçoado e verificado intensivamente, PAW é uma das ferramentas mais confiáveis para este fim e por seu amplo alcance e versatilidade se tornou um apêndice básico do CPSPICE.

Dentre os pacotes que integram o PAW estão o **HBOOK** e sua interface gráfica **HPLOT,** ambos usados neste projeto. HBOOK é um pacote de subrotinas, consistindo de algumas centenas de subrotinas escritas em FORTRAN, que permite o usuário definir, encher

e apresentar simbolicamente estimadores de densidade de uma ou duas dimensões, na forma de histogramas, gráficos de espalhamento (scatter-plots) ou tabelas, além de permitir o uso de **Ntuplas.** que serão vistas a seguir.

Normalmente o HBOOK apresenta seus resultados em uma impressora. mas com o uso do pacote HPLOT torna-se possível a apresentação em telas de alta resolução. Esses dois pacotes, quando integrados pelo PAW, constituem um ambiente de trabalho interativo e coerente, possibilitando ao usuário muitas formas de análise e apresentação de resultados.

Dentre as funções do HBOOK e HPLOT destacam-se:

- histogramas e Ntuplas de duas e três dimensões;
- projeções e cortes de histogramas e Ntuplas bidimensionais;
- controle completo do conteúdo de histogramas;
- ferramentas para minimização e parâmetrização;
- operações e comparações com histogramas;
- geração de números aleatórios;
- grande variedade de tipos de gráficos :
  1. histogramas, gráficos de barras, histogramas sombreados, barras de erros, cores;
  2. curvas e superfícies arredondadas (smoothed);
  3. curvas de níveis, gráficos de superficie e de espalhamento;
  4. janelamento automático.

A utilização HBOOK e HPLOT é simples. As operações básicas requerem o conhecimento das chamadas à subrotinas de acordo com os manuais [CERN_LIB, CERN_HBOOK], que mostram de forma detalhada a utilização dos programas.

Ntuplas são estruturas bidimensionais, tais como uma matriz ou tabela. onde cada linha corresponde a um evento, e cada coluna a uma variável. São muito utilizadas pelos físicos experimentais na análise de fenômenos que envolvem grandes quantidade de dados, tal como em um detetor de partículas, onde ocorrem muitos eventos que posteriormente sofrem diversos tipos de análises. Em outras palavras, as Ntuplas não só devem ter capacidade de armazenar grande volume de dados corno também devem ter habilidade para tratá-los.

Um determinado fenômeno fisico, tal como a colisão de partículas em um acelerador, ocorre inúmeras vezes, e durante o processo são observadas diversas variáveis, que assumem novos valores a cada novo evento. Da mesma forma, a simulação estatística de circuitos eletrônicos implica na criação de várias amostras de circuitos a partir de um circuito inicial, e cada simulação de uma nova amostra pode ser vista como um evento.

Para a análise dos resultados das simulações é necessário a criação de variáveis dentro do circuito, que assumirão valores diferentes nas simulações. Então, a estrutura da simulação estatística de circuitos é identica a ocorrência de um fenomeno ñsico, e assim o armazenamento de resultados na forma de Ntuplas é uma boa opção.

Existem dois tipos de Ntuplas: RWN (Row Wise Ntuples) e CWN (Column Wise Ntuples). De modo geral pode-se dizer que a diferença básica entre esses dois tipos de Ntuplas está na forma como o PAW armazena internamente os dados, através de arquivos

RZ. Este tipo de arquivo é parte integrante do pacote para gerenciamento de estruturas de dados ZEBRA, desenvolvido no CERN, que é totalmente transparente para o usuário.

Do ponto de vista do usuário não há diferenças entre os dois tipos de Ntuplas no momento de visualizar os resultados, já que todos os comandos de impressão e seleção de dados são exatamente os mesmos. Entretanto, Ntuplas CWN são mais flexíveis, pois além de permitirem o armazenamento de variáveis multi-dimensionais, os dados podem ser acessados com muito mais velocidade. Apesar dessas vantagens, este tipo de Ntupla só foi utilizado na versão TDA.

As rotinas do HBOOK implementadas no programa CLASSE3 [CERN_LIB, CERN_PAW] são as seguintes :

SUBROTINA	DESCRIÇÃO
HLIMIT	define o tamanho máximo do bloco de COMMON PAWC.
HROPEN	abre um arquivo HBOOK de acesso direto.
HBOOKN	aloca uma Ntupla RWN em memória
HFN	inclui um evento em uma Ntupla RWN.
HBNT	aloca uma Ntupla CWN em memória, como HBOOKN.
HBNAME	especifica as variáveis de uma Ntupla CWN.
HFNT	preenche uma Ntupla CWN, equivale a HFN para RWN
HROUT	transfere uma Ntupla para um arquivo de acesso direto
HREND	fecha uma arquivo de acesso direto.

As rotinas acima estão estruturadas dentro do programa CLASSE3, apresentado na seção seguinte, onde também se descreve o papel que cada uma das rotinas desempenha no processo de armazenamento dos resultados.

3.6. O programa CLASSE3 e a armazenamento de dados

A função do programa CLASSE3 é transformar os resultados das simulações em dados compatíveis com o programa PAW. responsável pela análise dos resultados. Como já discutido, o CLASSE3 tanto pode receber como entrada o vetor *iocps,* quanto os nomes dos arquivos gerados pelo SPICE.

No primeiro caso da versão TDM, quando um vetor *iocps,* contendo todos os resultados de uma simulação, é transferido por memória, o CLASSE3 deve fazer uma identificação dos *rótulos* (indicadores colocados pelas rotinas modificadas do SPICE) e dos respectivos resultados.

Já no segundo caso a tarefa é parecida com a do CLASSE1, pois em posse dos nomes dos arquivos gerados pelo SPICE em cada simulação realizada **(*.out),** o CLASSE3 pesquisa os nomes dos arquivos através da busca de padrões. A esse método chamou-se transferência de dados por arquivo - TDA.

Apesar da existência de duas versões, a estrutura básica do CLASSE3 se mantém em ambas. Consiste em ciclos onde a cada volta são recebidos os resultados correspondentes a

um processo (o vetor iocps para o caso da versão TDM ou o nome do arquivo com extensão **.out,** para o caso da versão TDA, construída a base de dados correspondente aquela simulação e feitas chamadas as rotinas do HBOOK. Ou seja, esquematicamente:
1. obtém as informações referentes a uma simulação;
2. filtra as informações e cria uma base de dados;
3. faz chamadas as rotinas do HBOOK;
4. retorna ao passo 1.

Em ambas as versões encontram-se indicadores para o primeiro ciclo, ou **indicadores de primeira passagem,** iniciados com valor 0 e, ao final do primeiro ciclo, recebem o valor 1. Com isso é possível a criação de estruturas apenas uma vez. ou seja, apenas no primeiro ciclo. Essas estruturas dependem de informações contidas nas simulações, e por isso precisam ser criadas de dentro da estrutura de ciclos, mas apenas uma vez devido a sua repetição a cada ciclo. Uma dessas estruturas é o denominado **vetor de legenda.**

Cada variável em uma Ntupla recebe um nome, que por uma limitação do PAW só pode ter até oito caracteres. Ao realizar as análises dos resultados. já dentro do programa PAW, o usuário deve identificar as variáveis através de seus nomes. mas, com apenas oito caracteres disponíveis, muitos precisam ser abreviados. A experiência de alguns físicos experimentais consultados ao longo deste projeto mostra que trabalhar com abreviações pode ser um empecilho a médio prazo, na medida em que a tendência é esquecer a que corresponde cada abreviação. Um circuito com muitas variáveis e com um volume de dados muito grande pode ser trabalhado durante dias e até mesmo meses. por diferentes pessoas em diferentes lugares, e daí a necessidade de se identificarem claramente as variáveis necessárias.

A solução adotada foi a criação do chamado **vetor de legendas,** que é apenas um arquivo texto contendo o nome de cada variável, mas com um limite bem mais amplo do que os oitos caracteres impostos pelo PAW. Esse arquivo é construído apenas uma vez, durante o primeiro ciclo no CLASSE3. Abaixo se encontra um exemplo do **vetor de legendas** que. além dos nomes das variáveis. contém ainda o título do circuito (extraído da descrição do circuito), data, hora e versão do SPICE utilizada.

CIRCUITO TESTE PARA O CPSPICE - ATENUADOR RESISTIVO
06/12/96 11:52:12 SPICE2G4 10/10/94
1) U1
2) U2
3) TEMPERATURA
4) DCNV NODE 1
5) DCNV NODE 2
6) POTÊNCIA TOTAL DISSIPADA
7) DCVC VIN
8) V(2)NIN
9) IMPEDANCIA DE ENTRADA
10) IMPEDANCIA DE SAÍDA EM V(2)

Cada arquivo de dados resultantes de uma simulação estatística possui um arquivo correspondente de legendas. que deve ser utilizado sempre que se trabalhar no PAW. As variáveis, que são identificadas por um nome e um número no arquivo de legendas, nas

Ntuplas possuem a seguinte forma: v<número da variável>. Isso quer dizer que a variável de número N no arquivo de legendas chama-se vN na Ntupla, e sempre que se fizer referência a essa variável para fins de plotagem de gráficos ou realização de cortes, utiliza-se o nome vN. Desta forma se preservam os nomes das variáveis. que podem ser consultados no arquivo de legendas, e se garante uma forma simples de trabalho com o programa PAW.

O CLASSE3 funciona retirando os processos da fila CPSPICE e recebendo os dados correspondentes aquele processo. Na versão TDM, além de retirar o processo da fila CPSPICE, o CLASSE3 deve receber o conteúdo de **iocps**. Já na versão TDA, os nomes dos arquivos são passados como argumento da rotina que retira os processos da fila. A cada fim de ciclo o CLASSE3, nas duas versões, coloca na fila READY o processo que inicialmente foi retirado da fila CPSPICE. Desta forma, quando a fila READY estiver cheia significa que o CLASSE3 já acabou de executar suas tarefas. As rotinas do CPS usadas no CLASSE3 são as seguintes :

SUBROTINA	DESCRIÇÃO
cps init e cps sync	para iniciar e sincronizar os processos do CPS
cps_dequeue_process	retira o processo da fila CPSPICE.
cps queue process	coloca o processo tirado de CPSPICE em READY
cps_stop_process	termina o processo.

CLASSE3 usa cps_dequeue_process no lugar de cps_get da versão TDM porque a informação sobre o nome do arquivo gerado pelo SPICE é transmitida como argumento da subrotina **cps_dequeue_process.** A rotina cps_queue_process faz com que, ao terminar de executar as tarefas correspondentes a um ciclo, o CLASSE3 coloque na fila READY o mesmo processo retirado da fiüla CPSPICE. Desta forma a fila READY, ao fim da execução de todos os ciclos estará novamente cheia indicando para o CLASSE1 que o CLASSE3 já chegou ao fim. O CLASSE1, então, coloca na fila CPSPICE um indicador de hm de fila que o CLASSE3. na próxima vez que fizer chamada a subrotina **cps dequeue process.** interpreta como um sinal para encerrar seu processo

3.6.1 Versã0 com transferência de dados por memória - TDM

O vetor iocps contém, logo após ser criado pelo CLASSE1. a descrição de um circuito, e dois campos adicionais contendo informações sobre as Variáveis Aleatórias e as Variáveis Print. Durante as simulações SPICE escreve os resultados obtidos após o campo de Variáveis Print, e então o vetor iocps é transferido para o CLASSE3. Este. por sua vez, deve realizar uma análise completa de iocps. identificando os resultados das simulações e os transcrevendo, no formato apropriado, para o arquivo de dados que será utilizado pelo PAW na visualização dos resultados.

Para que o CLASSE3 possa identificar em iocps a que análise correspondem os dados encontrados é importante utilizar um rótulo antes de cada sequência de resultados, como já foi visto. Os rótulos são escritos junto com os resultados das simulações em *iocps* pelas rotinas do SPICE que foram modificadas para tal fim. Pelo rótulo o CLASSE3 interpreta os resultados que se seguem, dados pelos números em **iocps.** A seguir são apresentados todos os possíveis tipos de análises utilizadas neste projeto, com seus respectivos *rótulos.*

RÓTULO	DESCRIÇÃO
TRNV	valores de tensões nodais, resultantes da análise TRANSIENTE
DCNV	valores de tensões nodais resultantes da análise DC
TRVC	valores de tensões em fontes de correntes e correntes em fontes de tensão resultantes da análise de transientes.
DCVC	Idem da análise DC
SSTF	valores da função de transferência do circuito
FOUR	coeficientes de fourier
TRDC	resultados da análise de transientes ou DC
ACAN	resultados da análise AC
ALTER	comando ALTER, indicando que terá início uma nova rodada de simulações
TEMP	comando TEMP, indicando que terá início uma nova rodada de simulações para outra temperatura ambiente.

Todos os rótulos são seguidos por dados referentes às simulações. Por exemplo a seguir ao rótulo TRNV, que corresponde a análise de tensões nodais como parte da análise TRANSIENTE, segue-se o número de nós analisados e após isto o número e valor de tensão de cada nó. Por exemplo após o rótulo TRNV é dado o número de nós analisados, 10. Então os próximos 20 dados correspondem ao número e ao valor da tensão em cada nó. As informações contidas em um certo número de bytes no vetor iocps correspondentes a cada rótulo são da forma **informação (n posições)**

No exemplo anterior temos em 1 byte de iocps o número de nós analisados seguido do nome e valor da tensão de cada nó (número de nós * 2 bytes).

RÓTULO	INFORMAÇÕES
TRNV, DCNV	número de nós analisados (1). nome e valor da tensão do nó (2 x número de nós).
DCVC, TRVC	potência total dissipada (1); número de fontes (1); nome da fonte e sua corrente ou tensão (2 x número de fontes).
SSTF	valor de: função de transferência (1), impedância de entrada (1), impedância de saída (1), nome da função (5) ; nome do nó de saída (3).
FOUR	valor da frequência fundamental (1); valor das componentes e fases (alternando) (18).
TRDC	número de pontos na análise (n pontos); número de variáveis dependentes (n depvar) ; valor das variáveis (n pontos x (n depvar + 1)).
ACAN	número de pontos na análise (n pontos); número de variáveis dependentes (n depvar) ; valor das variáveis (n pontos x (n depvar + 1)).
ALTER	nenhuma informação além do rótulo.
TEMP	valor da temperatura na qual serão feitas as próximas análises (1)

Cabe lembrar que para cada simulação (ou seja, chamada ao CLASSE2 e ao SPICE) existe um vetor iocps, criado pelo CLASSE1, editado pelo CLASSE2 (SPICE) e enviado ao CLASSE3. Um exemplo de um trecho do vetor iocps já foi apresentado na seção referente ao CLASSE1.

Algumas análises têm a mesma forma de apresentação dos resultados. e por isso correspondem a apenas um rótulo, como é o caso das análises TRANSIENTE e DC. O rótulo ALTER não é seguido de nenhuma informação, pois ele é apenas um indicador de início de nova rodada de simulação. TEMP, além de indicar o início de nova simulação, é seguido da informação sobre o valor da temperatura em que ela se realizará.

Os resultados são apresentados em formato numérico, mas os nomes das variáveis são constituídos de caracteres, ou seja, são strings. É necessário então que o CLASSE3 identifique em iocps tanto valores numéricos quanto strings, o que reforça a idéia dos rótulos, pois seria impossível prever qual o tipo da próxima informação encontrada. E, ao tentar ler uma informação do tipo string em formato numérico, ou vice-versa, o programa acusa erro e para na maioria das vezes. Com os rótulos é possível saber qual o formato do próximo dado em iocps.

Uma outra implicação da convivência de tipos diferentes de dados em iocps é que deve haver no CLASSE3 uma estrutura similar com a **struct** utilizada no CLASSE1 para definir iocps. Em FORTRAN 77 é necessário definir uma EQUIVALENCE entre iocps. definido como um vetor de reais, e um outro vetor, chamado de **cpsca,** definido como sendo composto de caracteres. Então, sempre que uma string for lida em iocps, utiliza-se cpsca em vez de iocps.

É também função do CLASSE3 identificar os nomes e valores das Variáveis Aleatórias escritos em iocps pelo CLASSE1, que ocupam desde a posição LVA + I até LVP. Todas as Variáveis Aleatórias são também variáveis na Ntupla, de forma a poderem ser analisadas graficamente no PAW. Assim podem ser estabelecidas relações entre desvios nos circuitos e nas Variáveis Aleatórias. Uma variável de circuito (uma tensão nodal, por exemplo) muito fora da especificação pode ter como causa uma Variável Aleatória cujo valor é excessivamente alto. No caso onde esta variável estiver modelando alguma condição ambiental será necessário que sejam feitas alterações no projeto.

As Variáveis Print. do mesmo modo que as Variáveis Aleatórias, são escritas em iocps pelo programa CLASSE1, a medida em que este encontra os comando .PRINT no arquivo original. No CLASSE3, ao se identificar os rótulos correspondentes às análises que envolvem essas variáveis (análises TRANSIENTE. DC e AC), é feita uma consulta a uma subrotina **(VARPRI)** que, através de critérios de preferência estabelecidos pelo SPICE, determina qual serão as Variáveis Print correspondes àquela análise. Supondo que no arquivo contendo a descrição do circuito se tenha o seguinte:

( )
.PRINT TRAN V(1) V(2) .PRINT AC V(3) V(4) .PRINT DC V(5) V(6)
( )

O CLASSE1 armazena os nomes das variáveis na ordem em que as encontra, no caso acima, V(1), V(2), V(3), V(4), V(5) e V(6). O SPICE executa os comandos .PRINT sempre na mesma ordem: DC, AC. TRAN, NOISE e DISTO. CLASSE3 recebe, por iocps, *os* nomes das variáveis na mesma ordem em que o CLASSE1 os escreveu. No processo de identificação dos rótulos e leitura dos resultados, o primeiro rótulo encontrado será então TRDC, correspondente às análises DC e TRANSIENTE. Os resultados são portanto recebidos pelo CLASSE3 na seguinte ordem: V(5), V(6), V(3), V(4), V(1), V(2), o que implica em um descasamento entre nomes e valores das variáveis. A subrotina VARPRI foi feita para determinar a sequência correta. baseando-se na ordem pré- determinada pelo SPICE.

Na versão TDM se fez a opção para armazenamento de dados por Ntuplas RWN (Row Wise Ntuple). Entretanto este tipo de Ntupla possui algumas limitações. A principal reside no fato de que cada variável na Ntupla possui apenas uma dimensão, ou seja, a cada evento uma variável corresponde apenas a um valor. Na maior parte dos casos isso faz sentido e é aplicável. Mas para as análises DC, AC e TRANSIENTE essa limitação pode se tornar um grande problema.

Para alocar na Ntupla RWN resultados da análise de transiente, é necessário armazenar em variáveis diferentes todos os valores simulados para cada instante de tempo. Entretanto PAW tem um limite para o número máximo de variáveis de uma Ntupla. Como por exemplo em uma análise com 100 instantes de tempo e 5 variáveis de circuito resulta em 500 variáveis, que é o limite máximo de variáveis na Ntupla. Esta solução foi portanto descartada.

Outra possibilidade é modelar o sinal por pulsos. Partindo da idéia de que na maior parte das vezes a resposta transiente corresponde a um pulso, faz-se a identificação deste padrão no sinal e a extração dos parâmetros deste modelo. Para cada Variável *Print* (tensão nodal ou corrente) que participa da análise TRANSIENTE, o número de variáveis na Ntupla poderia ser reduzido para algo em torno de dez.

Na figura 6 se observa os pontos criticos do pulso pelos valores médios de tensão da rampa de subida, do patamar constante e da rampa de descida, MI, M2 e M3. **T1. T2** e **T3** *são* os instantes de tempo correspondentes para esses valores de tensão. CPSPICE identifica esses três instantes de um pulso e analisa o sinal para determinar parâmetros do pulso como **tempo de subida. tempo de descida**, etc..., para serem armazenados na Ntupla.

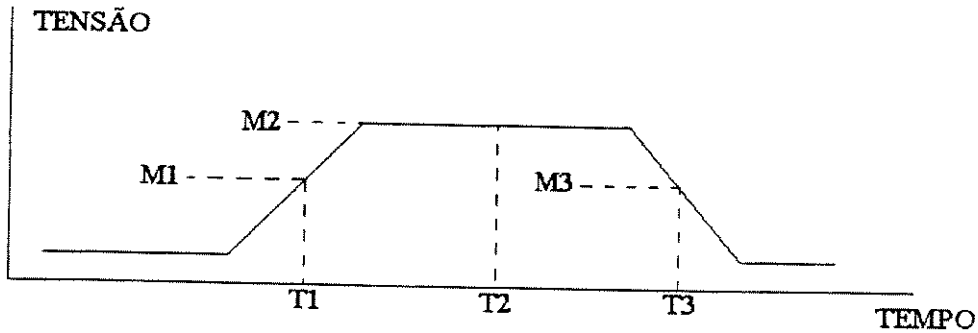

Figura 6 - Compressão de um sinal pulsado

A subrotina IDENTIFICA divide a onda quadrada em cinco estados. O primeiro estado corresponde a um valor constante. o segundo a uma rampa de subida/descida. o terceiro a um valor também constante. o quarto a uma rampa de descida/subida e finalmente o quinto estado. correspondendo a um valor constante. Esta solução também é limitada, na medida em que não atende a outros tipos de transientes, e seria necessário a construção de várias rotinas para identificar os diversos tipos de sinais, mas sempre aqueles mais comuns em circuitos eletrônicos, ficando fora da análise pelo PAW os demais sinais. A solução mais geral é o uso de Ntuplas do tipo CWN (Column Wise Ntuple), implementada na versão TDA.

3.6.2 Versão com transferência de dados por arquivos - TDA

O programa CLASSE3, em sua versão TDA teve sua estrutura muito simplificada, se comparado ao programa CLASSE3 da outra versão, tal como CLASSE1. Seu funcionamento sob certos aspectos é parecido com o do CLASSE1, pois lê de arquivos e identifica padrões de texto. O programa ficou mais simples e permitiu simplificar o CLASSE1. São utilizadas versões executáveis originais do SPICE, sem nenhuma modificação. Porém todos os arquivos criados pelo SPICE ficam disponiveis até que o CLASSE3 tenha terminado a execução de todas as tarefas. Dependendo do número de simulações requisitadas pelo usuário, isso pode significar a necessidade de um grande espaço em disco, mesmo que esse espaço seja temporário, já que após a leitura dos arquivos o CLASSE3 os apaga. Já na versãoTDM todas as informações necessárias estão contidas na memória.

O CLASSE3 funciona em ciclos, abrindo os arquivos ***.out** gerados pelo SPICE, extraindo deles os resultados das simulações e executando as rotinas HBOOK, para a criação da base de dados para o PAW. Não há mais o envio de dados através da rotina **cps get(),** como no caso da versão TDM, pois as informações necessárias ao CLASSE3 são o número do processo e o nome do arquivo de saída do SPICE, passados como argumentos das rotinas de controle de filas. A estrutura das filas foi mantida como na versão TDM.

Para identificar os valores das variáveis do circuito nas diversas simulações, CLASSE3 procura por trechos especificos no arquivo gerado pelo SPICE. Todas as análises possuem uma etiqueta, identificada por quatro asteriscos (****) iniciais. CLASSE3 faz a busca dos asteriscos, depois identifica no texto seguinte a análise em pauta. Então busca os nomes das variáveis, que são armazenados no vetor de legendas durante o primeiro ciclo, e após identifica os valores da variáveis no texto do arquivo .out.

Por exemplo no arquivo de saída do SPICE o resultado de uma análise de transferência de pequenos sinais é apresentado no seguinte trecho:

```
0 **** SMALL-SIGNAL CHARACTERISTICS
0 V(5)NIN = 8.616D+01
0 INPUT RESISTANCE AT VIN = 9.114D+03
0 OUTPUT RESISTANCE AT V(5) = 9.446D+03
```

As informações de interesse são os nomes da **função de transferência** e os valores das **impedâncias de entrada e de saída**. No se ciclo mais externo o CLASSE3 procura os asteriscos, o tîtulo correspondente à análise e depois as variáveis em questão. Para cada variável o CLASSE3 procura pelos resultados da análise, à esquerda do "=" o nome da variável e à direita o valor simulado. Os nomes das variáveis são armazenados apenas quando o primeiro arquivo da simulação de Monte Carlo é lido, o que corresponde ao primeiro ciclo, nos demais apenas o valores são identificados.

3.6.3 Ntuplas CWN

Na versão TDM a Ntupla RWN demonstrou incapacidade de armazenar grande número de variáveis. Ntuplas CWN são mais adequadas na medida em que admitem variáveis com

dimensões maiores que um, vetores. Isto resolve o problema das Variáveis Print já citado. Em uma análise de transiente a Ntupla CWN guarda variiáveis como V(1), V(2) e V(3) como três vetores de dimensão 10 cada, ou uma matriz. O número de instantes de tempo em que a análise é realizada se traduz na segunda dimensão da matriz, não afetando o número total de variáveis da Ntupla.

A construção de Ntuplas CWN implicou na criação de uma nova estrutura denominada **vetor ordem.** Esse vetor contém o valor da dimensão de cada variável da Ntupla, ou seja, é um vetor que contém a **ordem** de cada variável. Variáveis como a temperatura, por exemplo, possuem ordem 1, já uma tensão nodal. dentro de uma análise TRANSIENTE com 10 instantes de tempo possui ordem 10. O vetor ordem. tal como o vetor de legendas. é construído apenas no primeiro ciclo de CLASSE3.

A visualização de variáveis multidimensionais é idêntica à visualização das variáveis unidimensionais. Um gráfico de V(1) é mostrado em todos os instantes de tempo da simulação, ou seja, PAW exibe o vetor completo. Para visualização da variável em um instante de tempo especifico, o usuário precisa apenas fornecer o instante desejado.

A identificação e armazenamento na Ntupla das variáveis aleatórias ocorre de maneira parecida com a versão anterior. Entretanto CLASSE1 não escreve mais a cada simulação de amostra os nomes e valores das variáveis aleatórias em **iocps**, apenas no final do arquivo gerado, junto com a descrição do circuito. Como o SPICE ignora qualquer texto após o comando .END, não há nenhum inconveniente na realização desse procedimento.

As variáveis *Print* são tratadas como qualquer outra variável. O CLASSE3 interpreta as análises correspondentes no comando .PRINT e identifica as variáveis do mesmo modo usado para qualquer outro comando. Não há necessidade do CLASSE1 rastreá-las no arquivo **original.cir** com a rotina IDENTIFICA, que se torna desnecessária.

3.7 Implementação de outros simuladores

Neste capítulo foram apresentados tanto a estrutura de paralelização do CPSPICE quanto o necessário à implementação de outros simuladores. O projeto de circuitos eletrônicos necessita de diversos tipos de simulação, em especial os sistemas de controle e aquisição de dados de experimentos da Física de Altas Energias. Como por exemplo a simulação de circuitos analógicos em operação com sinais periódicos permamentes, de componentes semicondutores, de sistemas digitais e de sistemas mistos digitais e analógicos. Existindo simuladores disponíveis para estes casos, eles podem ser usados tal como o SPICE foi neste trabalho, mais ainda, diversos simuladores podem integrar um sistema maior de tal forma a contemplar no projeto seus mais diversos aspectos em termos de desempenho e custo.

No capítulo seguinte serão discutidas as limitações da implementação do CPSPICE e as possibilidades de novos trabalhos nesta área.

CAPÍTULO IV - RESULTADOS E PROPOSTAS

Neste capítulo serão discutidos os resultados obtidos neste trabalho. E também os problemas de implementação do CPSPICE, como orientação para a continuação deste trabalho tanto no uso do CPSPICE quanto no desenvolvimento de novos programas de simulação estatística, utilizando a paralelização com auxílio do CPS e o processanmento estatístico de dados da CERNLIB. São apontados os problemas de interpretação de resultados e apresentadas as conclusões deste trabalho.

4.1. PAW e CPS – Incompatibilidade

As rotinas das bibliotecas CERNLIB e CPS, que deveriam a princípio coexistir no programa da classe3, apresentaram problemas de incompatibilidade durante a compilação do programa da classe3. A solução adotada foi a separação do programa da classe3 em dois programas distintos, um responsável pela elaboração de uma base de dados. a partir dos resultados do SPICE, usando as rotinas do CPS, e o outro tendo como função criar uma nova base de dados para o PAW, através de chamadas a rotinas do HBOOK. Por não existir rotinas do CPS no segundo programa, este não pode ser iniciado pelo CPS, como ocorre com os outros processos.

A incompatibilidade entre PAW e CPS foi resolvida de maneiras semelhantes para ambas as versões, através da criação de uma quarta classe, que não participa do CPSPICE como um processo, pois o CPS não está presente com suas rotinas. Na versão TDM a classe4 é um programa executável, enquanto que na versão TDA a classe4 é criada pelo próprio classe3.

4.1.1 VERSÃO TDM - TRANSFERÊNCIA DE DADOS POR MEMÓRIA

Nesta versão foi criada a classe 4, uma "pseudo-classe" fora do processamento paralelo. O programa da classe3 gera dois arquivos texto, um contendo os resultados das simulações numéricas (class3.dat), e outro contendo os nomes das variáveis (legenda). Estes dados não podem ser analisados pelo PAW, que utiliza Ntuplas. O programa classe4, escrito em linguagem FORTRAN 77, tem a função criar e preencher a Ntupla que será usada no PAW a partir dos dados gerados pelo classe3, com auxílio das rotinas do HBOOK,.

Não sendo um processo do CPS, o programa da classe4 deve ser executado pelo usuário, após o término do programa classe3. O arquivo gerado pelo classe4 e o vetor de legendas, gerado pelo classe3, são utilizados pelo PAW para análise estatística e gráfica dos resultados das simulações. Para isto o usuário deve configurar o número total de variáveis envolvidas nas simulações. Este número pode ser obtido facilmente através de uma consulta ao arquivo de legendas, já que o número associado à ultima variável eqüivale ao número total de variáveis. Após reeditar o programa classe4, este deve ser recompilado para então poder ser executado. Esses passos são descritos em detalhe em [Natalizi, Apêndice I].

O funcionamento do classe4 consiste na leitura dos resultados das simulações contidos no arquivo texto **class3.dat** (gerado pelo classe3) e no armazenamento de cada evento na Ntupla em outro arquivo texto texto.

4.1.2 VERSÃO TDA - TRANSFERÊNCIA DE DADOS POR ARQUIVO

Na solução adotada para esta versão não é mais necessário que o usuário altere o programa classe4, pois ele é criado, compilado, executado e depois apagado dentro do programa classe3. Por isso, o classe4 torna-se transparente para o usuário, e, ao final dos processos normais do CPS, os arquivos gerados estão prontos para serem usados no PAW.

A criação do programa classe4 dentro do classe3 é feita através da função **system** do UNIX, que permite executar comandos do sistema operacional durante a execução de um programa. Deste modo, é possível compilar o programa criado, executá-lo e em seguida apagá-lo, de forma que ele se torne completamente transparente para o usuário.

Esta solução acaba com os conflitos de compilação entre os programas PAW e CPS, pois a compilação das rotinas do CPS (presentes no classe3) é separada da compilação do programa classe4 (que contém rotinas do PAW), sendo feita através do comando **system.**

4.2 PROBLEMAS DE EXECUÇÃO DO CPS EM ESTAÇÕES SUN

Durante o desenvolvimento do trabalho o sistema operacional das estações SUN utilizadas foi mudado, e o CPS apresentou problemas de execução no "cluster" das estações. Entretanto foi possível continuar a utilizá-lo no computador paralelo IBM modelo SP1, mai tarde transformado em SP2, localizado **no Laboratório Nacional de Computação Cientifica, LNCC** (ver **http://www.lncc.br**). Isso implicou em uma divisão física natural na execução do CPSPICE, na medida em que os processos de cálculo e análise foram separados. No computador paralelo se realiza o grande volume de cálculos da simulação de Monte Carlo, que inclui a geração das amostras de circuitos a partir de uma descrição fornecida pelo usuário (classe1), a simulação destas amostras através de chamadas ao SPICE (classe2), e a criação do arquivo de dados resultantes das simulações (classe3). A parte de análise de resultados, que engloba a criação da Ntupla e a visualização dos resultados no PAW, é realizada em uma estação de trabalho como uma SUN ou mesmo um computador pessoal, com alta resolução gráfica.

Esta divisão entre PAW e CPS implicou em alterações no programa classe3. Na versão que utiliza transferência de dados por memória, as mudanças foram poucas. O usuário deve transportar os arquivos gerados pelo classe3 em um computador para o outro computador onde será executado o classe4, o que pode ser feito pelo uso de programas de transferência de arquivos como o **ftp.** Neste trabalho os arquivos legenda e classe3.dat são transferidos do SP1 do LNCC para uma estação SUN no LAFEX/CBPF ou no LPC/UFRJ, onde roda o classe4. Este é executado tal como descrito anteriormente, gerando o arquivo para ser utilizado interativamente pelo PAW.

Na versão com transferência de dados por arquivo, o classe4 deixa de ser transparente para o usuário, já que não é mais possível sua compilação de dentro do classe3, pois estes estarão em processadores diferentes. O usuário deve transportar o arquivo de legendas e o programa classe4 criado pelo classe3 para o computador onde se encontra o PAW e suas bibliotecas. Após transferir estes arquivos basta que se compile e execute o programa classe4, criando o arquivo que permite a análise estatística pelo programa PAW.

4.3 SPICE E O COMPUTADOR PARALELO IBM SP2

Alguns problemas na compilação do SPICE no computador SP2 inviabilizaram o funcionamento da versão TDA original. Uma nova versão está sendo implementada. A versão TDM funciona com todas as rotinas do SPICE compiladas com o arquivo **class2.f** em um só programa executável. Esse não é o caso da versão com transferência por arquivo, onde o SPICE deve operar sozinho, como um programa independente, recebendo um arquivo texto como entrada, e gerando outro como saída de dados.

4.4 - PROPOSTAS PARA A CONTINUAÇÃO DO TRABALHO

Uma primeira proposta é a criação de um condomínio internacional de centros de projeto de circuitos com estatística, com base no uso do CPSPICE em computadores paralelos de universidades, centros de pesquisa e indústrias, voltados para a formação de pessoal capacitado para o projeto de circuitos com garantia de qualidade. Os computadores devem operar em regime de cliente-servidor, aproveitando o trabalho descrito em (D0_RJS). Grupos de pesquisadores, professores e alunos, podem assim trabalhar em conjunto, partilhando recursos de hardware e software, se comunicando via Internet através de "homepages" de cada grupo, disseminando a cultura local a nível internacional, dando um caráter mais democrático e produtivo à globalização de mercados.

A nível teórico existe uma gama de trabalhos tanto em simulação quanto em otimização de circuitos eletrônicos. Deve-se mencionar em especial o desenvolvimento de simuladores baseados em formulação linear por partes [Chua_NOEL], que possibilita garantir convergência para a solução de sistemas não lineares, e acelerar em muito tal convergência. Quanto à otimização, uma área de desenvolvimento muito interessante é o uso de técnicas de Monte Carlo visando não só atingir o mínimo global, quanto obter uma visualização mais ampla do problema de minimização em função de suas múltiplas variáveis, e da interrelação destas.

Na implementação de programas existem muitas possibilidades de aperfeiçoamento e de continuidade do trabalho aqui apresentado. Inicialmente seria interessante modificar a linguagem utilizada no programa classe3 na versão com transferência de dados por memória e nos programas classe2, em ambas as versões, pois o FORTRAN 77, apesar de ser uma linguagem poderosa para cálculos, perde para o C em termos de recursos para tratamento de dados, principalmente dados do tipo caracter, como ocorre constantemente no classe3. A mudança nos programas de classe2 seria uma forma de padronização de todas as classes para uma mesma linguagem, não constituindo, necessariamente, uma melhoria na qualidade dos programas e na qualidade final do CPSPICE.

Outra mudança, que implicaria em melhoria significativa para a versão TDM - transferência de dados por memória, é em relação ao tipo de Ntupla utilizada. As Ntuplas RWN apresentam grandes limitações se comparadas com as CWN, que só foi implementada na versão TDA, onde se mostrou mais eficiente. Deve ser feita, então, uma adaptação do programa classe3, na versão com transferência por memória, para trabalhar com Ntuplas CWN, que não apresenta dificuldades, e inclusive simplificar o programa, além de permitir a visualização dos dados das análises AC, DC e TRANSIENTE de forma completa, sem o uso de artifícios.

Em termos locais do LAFEX/CBPF, é urgente a instalação do SPICE3F4 no computador IBM modelo SP2 do LNCC, para com ele implementar a versão TDA do CPSPICE. Isto permitirá uma comparação entre as duas versões do CPSPICE em termos de velocidade, eficiência e confiabilidade das simulações, com medidas de desempenho tais como o tempo gasto na simulação dos circuitos, o espaço necessário em disco, e através dessas medidas, determinar qual das duas versões é mais adequada a uma dada configuração de *hardware,* ou mesmo para um determinado tipo de circuito. Por fim, as subrotinas geradoras de números aleatórios, no classe1, podem ser eventualmente atualizadas para métodos mais modernos e precisos.

4.5 OTIMIZAÇÃO DE CIRCUITOS

Um importante desdobramento deste trabalho de desenvolvimento do CPSPICE é a criação de um instrumento para projeto e otimização de circuitos eletrônicos, através do uso do MINUIT [CERN_MINUIT], um programa da CERNLIB para otimização de funções, que permita que valores nominais de componentes e tolerâncias de componentes sejam alterados automaticamente pelo CPSPICE visando otimizar uma dada função custo.

Isto implica na criação de medidas de desempenho para o circuito, definidas pelo projetista através de subrotinas de erro para o MINUIT. Em cada passo do processo de otimização, após realizar as alterações na descrição do circuito, o CPSPICE inicia uma nova **tarefa** do CPS, ao término da qual são efetuadas pelo MINUIT as medidas de desempenho e as alterações do circuito.

Com base em um estudo de técnicas numéricas de otimização voltadas para o projeto de circuitos eletrônicos, está sendo feita a implementação de uma **técnica de otimização estocástica** que se encontra descrita no **Apêndice 1**, como motivação para futuros desenvolvimentos, uma vez que se encontra incompleta.

4.6 RESULTADOS DE SIMULAÇÕES DO CPSPICE

O CPSPICE, em sua versão que utiliza transferência de dados por memória, tem sido utilizado em diversas análises estatísticas de circuitos para fins de ensino e teste do programa. Na figura 10, 11 e 12 são apresentados alguns resultados da primeira simulação de circuito realizada tal como descrito no arquivo schmit.cir, na seção 3.3. Os gráficos produzidos ilustram alguns dos recursos do CPSPICE e do PAW.

O número de amostras e comandos ALTER contidos no comando MC : *#MC 150 15 indicam que a simulação estatística foi feita com 150 simulações, cada uma contendo 15 alterações, feitas por comandos ALTER, dos valores dados na descrição, o que totaliza 2.401 amostras simuladas, um número significativo do ponto de vista estatístico.

Um resultado interessante da observação da figura 4.1, é a identificação de casos de potência dissipada quase nula comparada com a observada nos demais casos. Examinando-se os eventos correspondentes, se observou que correspondiam à problemas de convergência do SPICE, e casos de convergência para um ponto de operação onde todos os

transistores operam em saturação, comportamento inesperado porém possível para tal circuito.

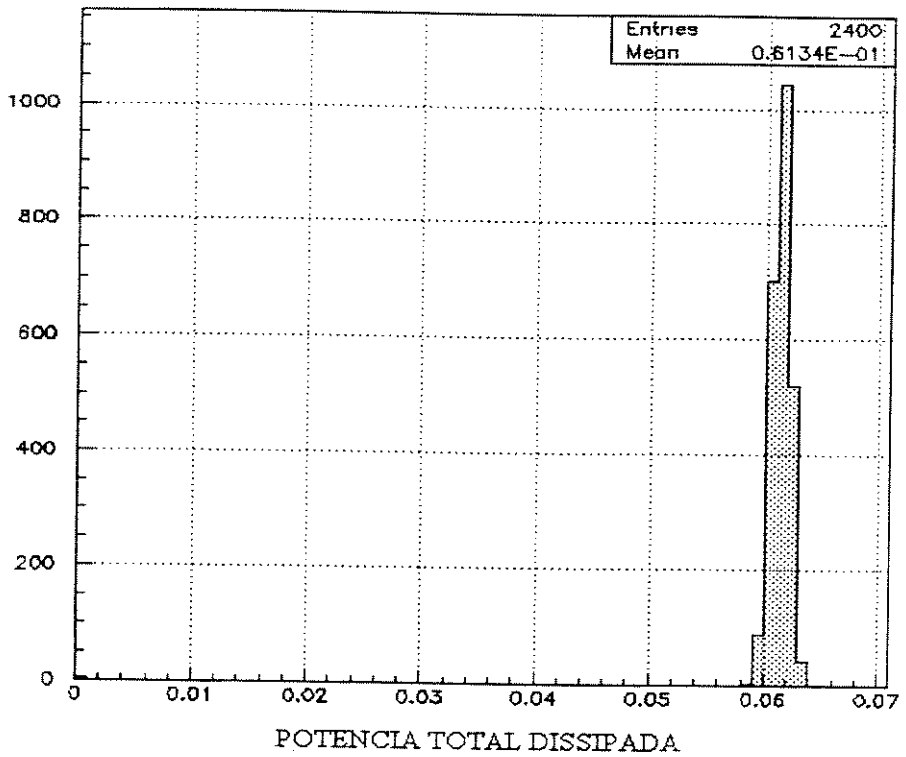

Figura 4.1 - Histograma da potência total dissipada

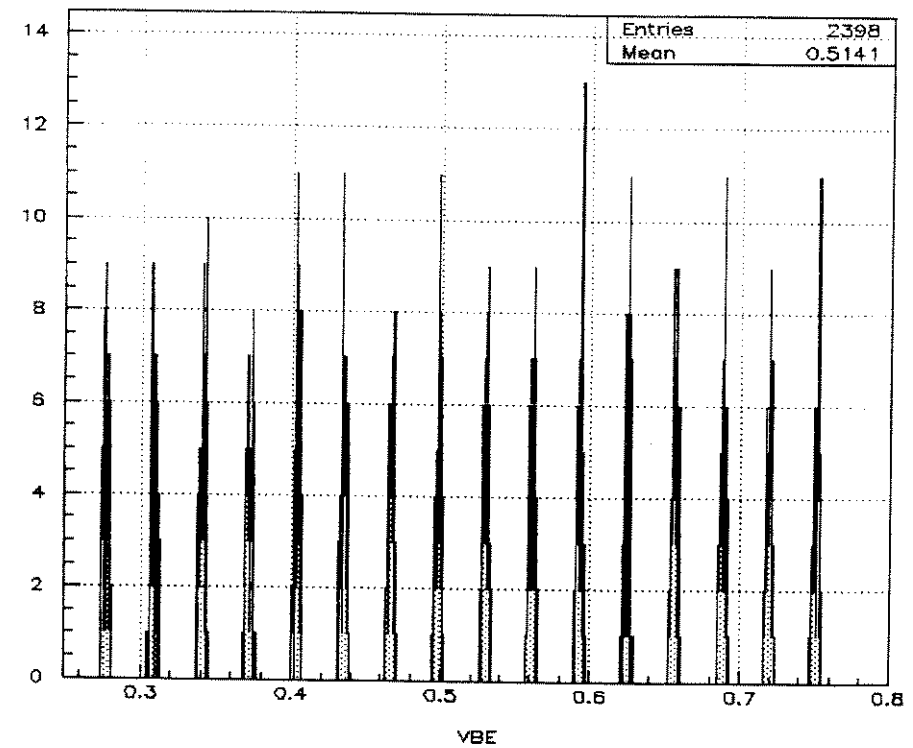

Figura 4.2 - Visualização dos valores de VBE do transistor Q1

Um resultado bem diferente se evidencia na figura 4.2, onde as tensões VBE do transistor Q1 estão com valores agrupados em intervalos. Testando-se a hipótese de que o CPSPICE não estaria gerando variáveis aleatórias de uma maneira coerente com os modelos, se observou no gráfico da figura 4.2, onde estão representadas quatro variáveis aleatórias Gaussianas, que essas variáveis obedecem às suas distribuições estatísticas.

A conclusão final foi dada pela observação da variável independente, tensão de entrada do circuito, que tal como o conjunto das soluções da análise DC está discretizada em intervalos de 150 mV. Este problema de interpretação de resultados apresentados na forma de gráficos no PAW deve sempre ser cuidadosamente estudado.

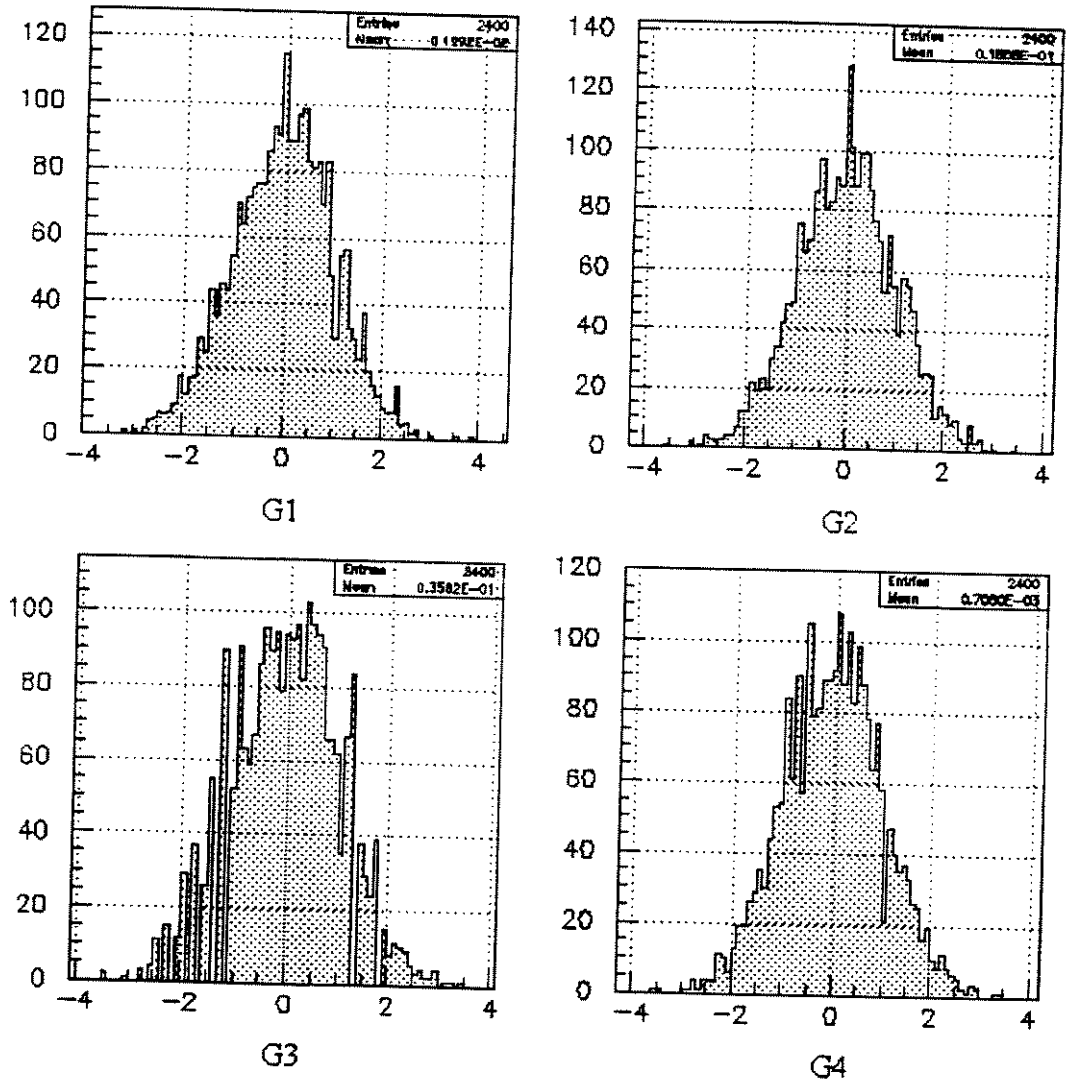

Figura 4.3 - Histogramas das variáveis aleatórias

CONCLUSÕES

Este trabalho mostrou o potencial dos recursos computacionais e instrumentais utilizados na área de Física de Altas Energias, para aplicações na área de Engenharia Eletrônica, em especial para auxílio a projeto de circuitos eletrônicos com garantia de qualidade, baseado em modelos estatísticos e simulações de Monte Carlo. Com esta tese se estabelece uma parceria que beneficia e fortalece ambas as áreas nas suas atividades-fim.

Foi desenvolvido o CPSPICE, que pode se tornar uma importante ferramenta de auxílio por computador a projeto de circuitos para uso em ensino, pesquisa e desenvolvimento. Este programa pode ser aperfeiçoado de diversos modos, e pode também servir de modelo para implementações de simulação de Monte Carlo em outras aplicações da Engenharia.

O que se pode concluir dos resultado das simulações feitas ao longo desta tese, uma das quais apresentadas no fim do último capítulo, é que as simulações pelo CPSPICE podem verificar falhas não apenas nas concepções dos circuitos mas também nos simuladores, falhas estas difíceis de se detectar com simulações determinísticas. Numa simulação desse tipo, o projetista se baseia totalmente no simulador utilizado, enquanto que, em uma simulação estatística é possível duvidar não só dos circuitos projetados como também dos métodos e processos utilizados pelo simulador. Neste sentido o SPICE2G6 se mostrou um simulador confiável tal como o SPICE3F4.

A área de projeto de circuitos e sistemas eletrônicos passou por um auge no Brasil nas décadas de 1970 a 1980, sofreu um duro impacto no início de 1990 e neste final de década se encontra em uma profunda crise, cujas raízes estão em três segmentos distintos. Em primeiro lugar o regime econômico privilegia o capital em detrimento da produção, os juros colocam o custo de capital proibitivo, inibindo o surgimento e o crescimento das empresas, e as destruindo por asfixia e imobilidade. Mais desanimadora é a constatação de que o retorno do capital produtivo é mínimo comparado com o do capital especulativo, se dando em intervalos de tempo muito mais dilatados, com muito mais incertezas. À ação dos bancos se junta o da fiscalização governamental, predatória em dois aspectos. Primeiro com a excessiva complexidade e mobilidade das regras do jogo, seguido da corrupção decorrente. Um ítem muito afetado por isto são as importações de componentes e partes de alta tecnologia, fundamentais para desenvolvimento e produção competitiva. É inconcebível que o governo do Brasil facilite tanto (oficialmente ou não) a entrada de equipamentos sofisticados de consumo, telecomunicações e computação, enquanto dificulta a entrada de componentes para desenvolvimento de tais equipamentos no País. A isso se soma a incompetência do setor gerencial e administrativo das empresas, e para finalizar, a quase inexistência de escolas técnicas de bom nível, sendo que as poucas existentes, tal como nossas escolas de engenharia, não formam adequadamente profissionais para operar dentro de um nível de produção moderno e eficiente. A abertura indiscriminada do mercado trouxe empresas que oferecem apenas serviços, que trazem seus administradores e gerentes do exterior, onde também realizam a pesquisa e o desenvolvimento de produtos, e mesmo a importação de equipamentos necessários para operar no Brasil.

Daí o desânimo que se nota neste setor Os países, tal como as pessoas, ou produzem ou se tornam amorfos, dependentes, fracos. A produção moderna e real, não ficção acadêmica ou política, envolve a eletrônica como peça fundamental de qualquer setor. É

necessário estimular a produção nacional competitiva, com base em projetos nacionais de grande porte que possibilite o desenvolvimento dos nossos profissionais para ocupar o mercado interno e externo, ao contrário do que hoje se verifica. Precisamos formar profissionais competentes em Eletrônica, e continuamente apontar à Sociedade sua importância, e os problemas e soluções que se apresentam no cenário nacional e internacional.

Com esta tese se pretende apontar para a importância da garantia de qualidade do projeto, condição indispensável para uma produção competitiva nos dias atuais, e as amplas possibilidades de desenvolvimento de ferramentas para tal fim. Fica estabelecida uma metodologia de projeto de circuitos analógicos com estatística, e a sua correspondente ferramenta de auxílio à projeto, o CPSPICE, simulador de circuitos com estatística baseado no SPICE, CPS e CERNLIB.

5 - Bibliografia

F.S.**Acton** - Numerical Methods that Work - Harper & Row, 1970, EUA.

W.E.**Biles** , J.J.**Swain** - Optimisation and Industrial Experimentation - John Wiley, 1980, EUA.

E.K.**Blum** - Numerical Analysis and Computation : Theory and Practice - Addison Wesley, 1971, EUA.

R.K.**Brayton** et alli - A Survey of Optimization Techniques for Integrated Circuit Design - Proc.IEEE Vol.69, No.10, pag. 1334-1362, Outubro 1981, EUA.

CERNLIB - Informações sobre a CERNLIB podem ser conseguidas no CERN Program Library Office, CERN-CN Division, CH-1211 Geneva 23, Switzerland, Tel. +41 22 767 4951, Fax. +41 22 767 7155, Internet : WWW.CERN.CH

L.C.S.**Dias** - Automatização do Projeto de Amplificadores CMOS de Alta Performance - Tese Mestrado, PEE / COPPE / UFRJ, 1994

J.M.**Hammersley** & D.C.**Handscomb** - Monte Carlo Methods - Metheum, 1967, Inglaterra.

F.**James** - MINUIT Reference Manual

J.G.**Lin** - Multi Objective Problems : Pareto Optimal Solutions by Method of Proper Equalily Constraints - IEEE Trans. on Automatic Control , Vol. AC-21, pag. 5, 1976, EUA.

A.C.M. **Mesquita** - Notas de Aula Programa de Engenharia Elétrica COPPE UFRJ, 1985, Brasil.

R.Y.**Rubinstein** - Simulation and the Monte Carlo method - John Wiley, 1981, EUA.

REFERÊNCIAS BIBLIOGRÁFICAS

/Agnew/ Agnew - Improved minimax optimization for circuit design, IEEE Trans. CAS 28, No.8, EUA, 1981.

/Albrecht/ P.Albrecht – Análise Numérica – Um Curso Moderno, Livros Técnicos e Científicos Editora, Brasil, 1973.

/Antognetti 84/ P.Antognetti, D.O.Pederson, H. de Man, Computer Design Aids for VLSI Circuits, Martinus Nijhoff, Holanda, 1984.

/Antognetti 88/ P.Antognetti, G.Massobrio - Semiconductor Device Modeling with SPICE, McGraw-Hill, EUA 1988.

/Ávila/ E.Ávila, M.S.Miranda - Guia do Usuário do CPS, LAFEX/CBPF, 1992, Brasil.

/Balaban/ P.Balaban, J.J.Golembeski - Statistical Analysis for Practical Circuit Design, IEEE Trans. CAS 22, No.2, pag.100-8, EUA, 1975.

/Bates/ D.M.Bates, D.G.Watts - Nonlinear Regression Analysis and its Applications, Wiley, EUA, 1988.

/Bhattacharyya/ A.B.Bhattacharyya, S.Aggarwal - Variability reduction in CMOS operational amplifiers through layour modification, IEE Proc. Pt.G V.126, No.2, pag.79-83, Inglaterra, abril, 1989.

/Beck/ J.V.Beck, K.J.Arnold - Parameter estimation in Engineering and Science, Wiley, EUA, 1977.

/Becker/ P.W.Becker, F.Jensen - Design of Systems and Circuits for Maximum Reliability or Maximum Production Yield, McGraw-Hill, EUA, 1977.

/Bell/ Bell System Technical Journal, Vol.50, No.4, abril, 1971 - Número especial sobre projeto estatístico de circuitos eletrônicos, EUA, 1971.

/Biles/ W.E.Biles, J.J.Swain - Optimization and Industrial Experiments, Wiley, EUA, 1980.

/Blum/ E.K.Blum - Numerical Analysis and Computation: Theory and Practice, Addison Wesley, EUA, 1972.

/Bokoven/ W.M.G.van Bokhoven - Piecewise Linear Analysis and Simulation - parte 2, capítulo 9 de [Ruehli].

/Bordewijk/ J.L. Bordewijk - Inter-reciprocity applied to electrical networks, Appl.Sci. Res., 6B, pag. 1-74, USA, 1956.

[Bowles] – J.B.Bowles – A Survey of Reliability-Prediction Procedures for Microelectronics Devices, IEEE Trans on Reliability, V.41, N.1, pag.2-12, EUA, março, 1992.

/Brayton/ R.K.Brayton, R.Spence - Sensitivity and Optimization, Elsevier, Holanda, 1980.

/Brenner/ E.Brenner, M.Javid – Analysis of Electric Circuit, McGraw Hill, EUA, 1959.

/Brun/ R.Brun et alli - Visualization of Scientific Data for High Energy Physics, CERN 94-06 Proc. 1993 CERN School of Computing, Itália, 1994.

/Calahan/ D.A.Calahan - Computer Aided Network Design, McGraw-Hill, EUA, 1972.

/CERN_LIB/ CERN Computer Centre - CERNLIB Program Library Short Writeups, CERN,

Suiça, 1994.

Ver sítio na WWW **http://wwwinfo.cern.ch/asdoc**, ftp anonymous@asis01.cern.ch.

/ CERN_PAW/ 15 CN/ASD Group - PAW Users Guide, Program Q121, CERN, Suiça, 1993.

/ CERN_MIN/ CN/ASD Group - MINUIT Users Guide, Program D506, CERN, Suiça, 1993.

/ CERN_HBO/ CN/ASD Group - HBOOK Users Guide, Program Y250, CERN, Suiça, 1994.

/Chua&Lin/ L.O.Chua, P.Lin - Computer Aided Analysis of Electronic Circuits, Prentice Hall, EUA, 1975.

/Chua_PWL/ Série de artigos descrevendo modelagem e análise de sistemas lineares por partes :

L.O.Chua, A.Deng - Canonical Piecewise Linear Representation, IEEE Trans.CAS 35, No.1, Pag.101-111, EUA, 1988.

L.O.Chua, A.Deng - Canonical Piecewise-linear Modeling, IEEE Trans.CAS 33, No.5, pag. 511-525, EUA, 1986.

L.O.Chua, R.L.P.Ying - Canonical Piecewise Linear Analysis, IEEE Trans.CAS 30, No.3, pag. 125-140, EUA, 1983.

L.O.Chua, A.Deng - Canonical Piecewise-Linear Analysis : Generalized Breakpoint Hopping Algorithm, International Journal of Circuit Theory and Applications, Vol.14, Pag. 35-52, EUA, 1986.

/Chua_NOEL/ L.O.Chua, A.Deng - Relatórios Técnicos do programa NOEL , Univ. California at Berkeley - EUA, 1985.

/Cirit/ M.A.Cirit - The Meyer Model Revisited : Why is charge not conserved ?, IEEE Trans. CAD 8, No.10, outubro 1989.

/Cohen/ E.Cohen - Program Reference for SPICE2, Memo ERL/UCB M-592, Univ. da California em Berkeley, EUA, setembro, 1976.

/Colclaser/ R.A.Colclaser - Microelectronics: Processing and Device Design, Wiley, EUA ,1980.

/Computer85/ Número especial da revista Computing, "Multiprocessing Technology", EUA, Junho, 1985.

/CPS_R/ M.R.Fausey et alli - CPS and CPS Batch Reference Guide, FERMILAB-GA0008, EUA, 1993.

/CPS_U/ K.Q.Sullivan - CPS Users Guide, FERMILAB - GA0009, EUA, 1993.

/Desoer/ C.A.Desoer, E.S.Kuh - Circuit Theory, McGraw-Hill, EUA, 1968.

/Director/ S.W.Director et alli – Statistical Integrated Circuit Design, IEEE JSSC, Vol.28, No.3, pag.193-202, EUA, março 1993.

/Divekar/ D.A.Divekar - Device Modelling for Statistical Circuit Simulation, Tech. Rep. ICL/SEL 5021-3, Stanford Univ., EUA, 1978.

/Dutton/ R.W.Dutton, D.A.Divekar - Bipolar Model for Statistical IC Design, Tech.Rep. ICL/SEL 5021-2, Stanford Univ., EUA, 1978.

/D0_NIM/ Dzero Collaboration - Dzero Detector, Nuclear Instruments and Methods, A338,

185 (1994). Ver também em FERMILAB-PUB-93/179-E.

/D0_RJS/ G. A. Alves et alli - WWW Oriented Remote Job Submission Monitoring and Management over Internet, Proceedings of CHEP97 - Computing in High Energy Physics, Berlim, Alemanha, 7-11, abril, 1997. Ou CBPF-NT-003/97, Brasil .

/D0_Upg/ Dzero Collaboration - Dzero Detector Upgrade, Proc.Int.Conf. on High Energy Physics, Varsóvia, Polônia, 1996.

/Eadie/ W.T.Eadie e outros - Statistical Methods in Experimental Physics, North-Holland, Holanda, 1971.

/El/ Y.El-Mansy - MOS Device and Technology Constraints in VLSI, IEEE JSSC, pag.197-203, Abril 1982.

/Fausey/ M.R.Fausey - CPS and the Fermilab Farms, FERMILAB-Conf-92-163, EUA, 1993.

/Fuller/ J.A.Fuller - Measurement Error Models, John Wiley, EUA, 1987.

/Furth/ P.M.Furth, A.G.Andreou – On Fault Probabilities and Yield Models for VLSI Neural Networks, IEEE JSSC 32, No..8, pag.1284-1287, EUA, agosto 1997.

/Getreu/ I.Getreu - Modeling the Bipolar Transistor, Tektronix Inc. part # 062-2841-00, EUA, 1976.

/Glaser/ A.B.Glaser, G.E.Subak-Sharpe - Integrated Circuit Engineering : Design, Fabrication and Applications, Addison Wesley, EUA, 1977.

/Gazelis/ C.Gazelis, I.C.Guknar - Finding Multiple Solutions of Piecewise Affine Resistive Circuits, Proc. IEEE ISCAS 88, pag.1229-1232 - EUA, 1988.

/Gnedenko/ B.V.Gnedenko - The Theory of Probability, MIR Publishers, Rússia, 1969.

/Gowda/ S.M.Gowda et alli – Design and Characterization of Analog VLSI Neural Network Modules, IEEE JSSC, Vol.28, Num.3, pag.301-313, EUA, março 1993.

/Güzelis/ C.Güzelis - Piecewise-Afine Maps: Structural Characteristics and Nonlinear Circuit Applications, PhD Thesis, Istanbul Technical University, Institute of Science and Technology, Electronics Department, Turquia, 1988,

/Haase/ J.Haase, E.Lüder - Discrete Optimization by Combining Local Search and Design Centering Techniques, Proc. ISCAS 88, pag. 2735-2738, 1988.

/Hammersley/ J.M.Hammersley, D.C.Handscomb - Monte Carlo Methods, Methuen, Inglaterra, 1967.

/Hoensch/ W.Hoensch, S.Selberherr - MINIMOS 3: A MOSFET Simulator that Includes Energy Balance, IEEE Trans. ED 34, pag.1074-1078, EUA, 1987.

/Hansen/ S.E.Hansen - SUPREM III User's Manual, Stanford Univ., EUA, 1985.

/Harjani/ R.Harjani e outros - OASYS: A Framework for Analog Circuits Synthesis, IEEE Trans. CAD 8. pag.1247-1266, EUA, dezembro, 1989.

/Herkowitz/ G.J.Herskowitz, R.B.Schilling - Semiconductor Device Modelling for Computer-Aided-Design, McGraw-Hill, EUA, 1972.

/Ho/ C.P.Ho e outros – SUPREM 3, A Program for IC Process Modeling and Simulation, Tech.Rep. SEL 84-001, Stanford Univ., EUA, 1984.

/Huang/ Q.Huang, R.Liu - A Simple Method for All Solutions of a Piecewise Linear Network, Proc. IEEE ISCAS 88, pag.1233-1236, EUA, 1988.

/Huertas/ J.L.Huertas et alli – Integrated Circuit Implementation of Fuzzy Controllers, IEEE JSSC, Vol.31, Num.7, pag.1051-1058, EUA, julho 1996.

/IEE82/ IEE - Número especial sobre análise de tolerância e projeto de circuitos, Circuits and Systems Proceedings do IEE, Vol.129, 4, Inglaterra, agosto, 1982.

/IEEE71/ IEEE - Número especial sobre projeto de circuitos eletrônicos auxiliado por computador, IEEE Trans. CT 18, No.1, EUA, janeiro, 1971.

/IEEE86/ IEEE - Número especial sobre projeto estatístico de circuitos VLSI - IEEE Trans. CAD 5, No.1, EUA, janeiro, 1986.

/Inohira/ S.Inohira et alli – A Statistical Model Including Parameter Mtaching for Analog Integrated Circuit Simulation, IEEE Trans. CAD 4, No.4, pag.621-7, EUA, 1985.

/INTEL/ INTEL Co - Components Quality/Reliability Handbook, EUA, 1987.

/Jaeckel/ H.Jaeckel, W.Baechtold - Computer Simulation Models for Digital josephson Devices and Circuits, Circuit Analysis, Simulation and Design, Ed.A.Ruehli, Elsevier, EUA, 1986.

/Kahlert/ L.O.Kahlert, L.O.Chua - A Generalized Canonical Piecewise-Linear Representation, IEEE Trans. CAS 37, No.3, pag.373-82, EUA, 1988.

/Kang/ S.M.Kang, L.O.Chua - A Global Representation of Multidimensional Piecewise-Linear Functions with Linear Partition, IEEE Trans CAS 25, No.11, pag.938-940, 1978.

/Ketner/ T. Ketner et alli – Analog CMOS Realization of Fuzzy Logic Membership Functions, IEEE JSSC 28, Num.7, Pag.857-861, EUA, julho, 1993.

/Khalily/ E.Khalily - TECAP : An Automated Characterization System, Tech. Rep. SEL-5015-1, Stanford Univ., EUA, 1979.

[Kim] – C.Kim, H.K.Lee – A Monte Carlo Simulation Algorithm for Finding MTBF, IEEE Trans on Reliability, V.41, N.2, pag.193-195, EUA, junho, 1992.

[Kirkpatrick] - S.Kirkpatrick, C.D.Gellat, M.P.Vecchi - Optimization by Simulated Annealing, Science, V.220, N.4598, pag.671-680, EUA, maio, 1983.

/Klaassen_A/ F.M.Klaassen e outros - Computer Algorithm to Determine MOS Process Parameters, Philips Res.Rep., 31, pag.84-92, Holanda, 1976.

/Klassen_M/ F.M.Klaassen - A MOS Model for CAD, Philips Res.Rep., 31, pag.71-83, Holanda, 1976.

/Knuth/ D.E.Knuth - Seminumerical Algorithms, Addison Wesley, EUA, 1981.

/Kuester/ J.L.Kuester,H.Mize - Optimization Techniques with Fortran, McGraw Hill, EUA, 1973.

/Kurker/ C.M.Kurker et alli – Hierarchical Yield Estimation of Large Analog Integrated Circuit, IEEE JSSC, Vol.28, Num.3, Pag. 203-209, EUA, março, 1993.

/Lawson/ Lawson, Uilenbeck - Threshold Signals, MIT Radiation Series, McGraw Hill, EUA, 1950.

/Leung/ K.H.Leung, R.Spence - Multiparameter Large-Change Sensitivity Analysis and

Systematic Exploration, IEEE Trans. CAS 22, No.10, pag. 796-804, EUA, 1975.

/Lewis/ D.W.Lewis - Device Model Approximation using 2 trees, IEEE Trans.CAD 9, No.1, EUA, 1990.

/Maly/ W.Maly – Modeling of Lithography Related Yield Losses for CAD of VLSI Circuits, IEEE Trans. CAD 4, No.3, pag. 166-177, EUA, july, 1985.

/Mead/ L.Conway, C.Mead.- Introduction to VLSI Systems, Addison Wesley, EUA, 1980.

/Meijer/

Meijer,P.L. - Table Models for Device Modeling, Proc.IEEE ISCAS - 90, pag. 2593-2595, EUA, 1990.

P.L.Meijer - Fast and Smooth Highly Nonlinear Multidimensional Table Models for Device Modeling, IEEE Trans.CAS 37, No.3, pag.335-46, EUA, 1990.

/Michael/ C.Michael, M.Ismail- Statistical Modeling for CAD of MOS VLSI Circuits, Kluwer, EUA, 1993.

/Miranda/ M.S.Miranda - Um Conjunto de Ferramentas para Implementação de Processos Cooperativos, Tese de Mestrado, COPPE/UFRJ, Brasil, abril, 1991.

[Moebus] D.Moebus, R.Kuntz, E.Lueder, H.U.Lauer - Some Methods to Overcome Local Optima in Optimization Procedures, Proc.ISCAS89, pag. 519-521, 1989.

/Moore/ P.Moore - Interval Arithmetik - Springer Verlag, Alemanha, 1980.

/Nagel/

L.W.Nagel, D.O.Pederson - SPICE: Simulation Program with Integrated Circuit Emphasis - Memo ERL/UCB M-382, Univ. da Califórnia em Berkeley, EUA, 1970.

L.W.Nagel - SPICE2: A Computer Program to Simulate Semiconductor Circuits, Memo ERL/UCB M-520, Univ.California / Berkeley, EUA, 1975.

[Nash]

T.Nash e alli - High Performance Parallel Computers for Science, Conference on Computational Atomic and Nuclear Physics at One Gigaflop, Oak Ridge, EUA, 1988.

T.Nash - Computing at Fermilab, CHEP91, Tsukuba, Japão, 1991.

/Nassif/ S.R.Nassif e outros - FABRICS II : A Statistically based IC Fabrication Process Simulation, IEEE Trans.CAD 3, pag.40-46, EUA, 1984.

/Natalizi/ R.A.Natalizi - Análise Estatística de Circuitos Eletrônicos, Projeto de Fim de Curso do DEL/UFRJ, Brasil, 1997.

/Nishi/ T.Nishi, L.O.Chua - Topological Conditions for a Resistive Circuit Containing Negative Nonlinear Resistors to have a Unique Solution - International Journal of Circuit Theory and Applications, Vol.15, Pag. 193-210, EUA, 1987.

/Oehm/ J.Oehm, K.Schumacher – Quality Assurance and Upgrade of Analog Characteristics by Fast Mismatch Analysis Option in Network Environment, IEEE JSSC, Vol.28, Num.7, Pag. 865-871, EUA, julho 1993.

/Ohtsuki/ T.Ohtsuki, N.Yoshida - DC Analysis of Nonlinear Networks Based on Generalized Piecewise-Linear Caracterization, IEEE Trans CT 18, 1, Pag.1461-1452, EUA, 1971.

/Oldfield/ J.V.Oldfield, J.S.Matos - Manual of VLSI CAD Tools, D.E.C.E./ Syracuse University, EUA, 1987.

/Oldham/ W.Oldham e outros - A General Simulator for VLSI Lithography and Etching Process : Application to Project Lithography, IEEE Trans. ED 26, EUA, abril 1979.

/Opalski/ L.J. Opalski - Yet Another Approach to Statistical Circuit Design, Stochastic Minimax, IEEE ISCAS 1990, EUA, 1990.

/Orear/ J.Orear - Notes on Statistics for Physicists, CLNS 82/511, Cornell Univ., EUA, 1982.

/Osowski/ S.Osowski - Canonical Modeling of Nonlinear Devices Through Circuit Optimization, Proc. IEEE ISCAS-88, pag.2711-2714, EUA, 1988.

/Orsini/ L.Q.Orsini - Projeto de Circuitos Eletrônicos por Computador, Notas de aula - EPUSP, Brasil, 1981.

/Pfiester/ J.R.Pfiester e outros - Peformance Limits of CMOS ULSI, IEEE JSSC, pag.253-63, EUA, 1985.

/Press/ W.H.Press e outros - Numerical Recipes, Cambridge Univ. Press, EUA, 1986.

/Press_SA/ W.H.Press, S.A.Teukolsky - Simulated Annealing Optimization over Continuous Spaces, Computer in Physics, pag. 425-429, EUA, julho, 1991.

/PVM/ A.Geist et alli - PVM 3 User's Guide and Reference Manual, Eng. Physics and Mathematics Division, Oak Ridge Lab., USA, 1994.

/Ramo/ S.Ramo, J.R.Whinnery, T.Van Duzer - Fields and Waves in Communication Engineering, Wiley, EUA, 1965.

/Rankin/ P.J.Rankin - Statistical Modeling for Integrated Circuits, IEE Proc.G Vol.129, No.4, pag.186-98, Inglaterra, 1982.

/Rohrer/ S.W.Director, R.A.Rohrer - A Generalized Adjoint Network and Network Sensitivities, IEEE Trans. CT 16, pag. 330-336, USA, 1969.

/Rubin/ S.M.Rubin - Computer Aids for VLSI Design - A.Wesle, EUA y, 1987.

/Rubinstein/ R.Y.Rubinstein - Simulation and the Monte Carlo Method – John Wiley, EUA, 1981.

/Ruheli/ J.Ruheli - Circuit Analysis, Simulation and Design - Advances in CAD for VLSI, Vol.3, partes 1 e 2, Elsevier, Holanda, 1986.

/Sah/ C.Sah - Evolution of the MOS Transistor, from Conception to VLSI - Proc.IEEE, pag.1280-1326, USA, 1988.

/Saleh/ R.A.Saleh e outros - SPLICE User's Guide - DEECS, Univ.California / Berkeley, USA, 1983.

/Sandberg/ I.W.Sandberg, A.N.Willson - Some theorems on Properties of DC Equations of Nonlinear Networks - B.S.T.J, Vol.48, N0 1, EUA, 1969.

/Science93/ Número especial da revista Science, "Computing in Science", Agosto, EUA, 1993.

/Shah/ A.Shah, P.Yang - MOS Technology: Trends and Challenges in the ULSI Era - Proc.MIEL95, pag.3 - 9, Nis, Sérvia , September 13-14, 1995.

/Shyu/ J.-B.Shyu et alli – Random Error Effects in Matched MOS Capacitors and Current Sources, IEEE JSSC SC19, No.6, pag. 948-955, EUA, dezembro, 1984.

/Spence/ R.Spence, R.S.Soin - Tolerance Design of Electronic Circuits - Addison-Wesley, Inglaterra, 1988.

/Stith/ - J Stith - Data Storage Technology - Fermilab Pub 91/348, EUA, 1991.

/Strojwas/ A.J.Strojwas - Selected Papers on Statistical Design of Integrated Circuits - IEEE Press, EUA, 1987.

/Styblinski/ M.A. Styblinski, M.Qu - Statistical Characterization and Modeling of Analog Functional Blocks - IEEE ISCAS 1995 Proc., pag. 121, EUA, 1995.

/Szu/ H.Szu, R.Hartley - Fast Simulated Annealing - Physics Letters A, V.122, N.3-4, pag.157 - 162, EUA, junho, 1987.

/TEXAS/ TEXAS Instrument VLSI Lab. - Technology and Design Challenges of MOS VLSI - IEEE JSSC, pag.442-448, EUA, 1982.

/Till/ C.W.Till, J.T.Luxon - Integrated Circuits: Materials, Devices and Fabrication - Prentice-Hall, EUA, 1982.

/Tadeusiewicz/ M.Tadeusiewicz, M.Ossowski - Application of the Block Jacobi Method to Piecewise-Linear Analysis of DC Diode-Transistor Networks - Proc. ISCAS-88, pag.1237-1240, EUA, 1988.

/Taguchi/

G.Taguchi - Introduction to Quality Engineering : Designing Quality into Products and Processes - UNIPUB Quality Resources, EUA, 1986.

G.Taguchi, E.A.Elsayed, T.C.Hsiang - Quality Engineering in Production Systems - McGraw-Hill, EUA, 1989.

/Tejayadi/ O.Tejayadi, I.N.Hajj - Dynamic Partitioning Method for Piecewise - Linear VLSI Circuit Simulation - International Journal of Circuit Theory and Applications, Vol.16, pag.457-472, EUA, 1988.

/UNIX/ Manuais de referência da AT&T, Univ. da Califórnia em Berkeley e da MIPS, as três versões do UNIX dos quais o LAFEX possui o código fonte e a licença para uso e desenvolvimento, desde 1993. Foi nestes códigos que engenheiros do LAFEX trabalharam no desenvolvimento e aperfeiçoamento do CPS nos EUA e no Brasil, e sob esta licença opera o ACP II no LAFEX.

/Ushida/ A.Ushida, L.O.Chua - Tracing Solution Curves of Nonlinear Equations with Sharp Turning Points - Int.J. of Circ. Theory and Applications, Vol.12, Pag. 1-21, EUA, 1984.

/Vaz81/ M.Vaz - Notas de aula de Microeletrônica I - DEL/UFRJ, Brasil, 1981.

/Vaz89/ M.Vaz e outros - Simulação de Circuitos Integrados com o SPICE - Relatório Técnico GRAM8901/R02, DEL/UFRJ, Brasil, 1989.

/Vaz95/ M.Vaz - Electronic Circuit Design with HEP Computational Tools - Proc. CHEP95, Brasil, 1995.

/Vinci/ F.Vinci - Implementação de Modelos no Programa SPICE3 - Projeto de Fim de Curso, DEL/UFRJ, Brasil, 1990.

/Yang/ P.Yang e outros - An Integrated and efficient Approach for MOS VLSI Statistical Circuit Design - IEEE Trans.CAD 5, No.1, EUA, 1986.

/White/ UNIX System V Primer- M.White, D.Martin, S.Prata, Howard W.Sams Co, USA, 1990.

/Wolbers/ S. Wolbers - Software for Parallel Processing Applications - Fermilab Conf 92/260, EUA, 1992.

/Zadeh/ A.Zadeh - Fuzzy Systems - John Wiley, EUA, 1976. ???

/Ziel/ A.van der Ziel - Noise in Measurements - John Wiley, EUA, 1976.

Apêndice 1

UM MÉTODO DE OTIMIZAÇÃO ESTOCÁSTICA DE AUXíLIO À SÍNTESE DE CIRCUITOS ELETRÔNICOS

1 - INTRODUÇÃO

Os métodos de otimização são fundamentais para síntese de circuitos e modelagem de componentes e circuitos, como pode ser visto na literatura, da qual destacamos [Bell, Brayton, Dias, IEEE, Mesquita]. Porém a sua origem, tal como grande parte de suas aplicações, está da área econômica, na qual se encontra grande parte da literatura a ela relacionada. Estes métodos, baseados em técnicas de programação matemática, permitem a determinação de máximos ou mínimos de funções representativas de objetivos a serem alcançados, geralmente medidas de erros, diferenças entre especificações e desempenho, ganhos, perdas ou custos de realização. As variáveis envolvidas neste processo podem estar sujeitas à restrições impostas por equações de desigualdade. Uma representação matemática do problema é dada pela equação 1 :

(1) $f(p_0) = \min \{ f(p) \mid g(p) \leq 0 \} = \max \{ -f(p) \mid g(p) \leq 0 \}$, $p \in R^q$, $f : R^q \rightarrow R^n$, $g : R^n \rightarrow R^m$

Nesta equação f e g são funções reais. **f** é denominada função objetivo, pois descreve os objetivos a alcançar na otimização, caso seja possível. Exemplos de objetivos de síntese de circuitos são o retardo de resposta ou custo total nulos, ganho ou banda infinitos. A função **g** representa as restrições que devem ser obedecidas em qualquer caso, tais como valores limites para os parâmetros p ou para variáveis dependentes de p, geralmente relacionadas com o desempenho de circuitos, como por exemplo ganho, impedância ou distorção harmônica. Quando o objetivo é único, ou seja n=1, a minimização (ou maximização) da função objetivo se baseia no presuposto da existência de um ponto ótimo a alcançar, um mínimo global que se destaca de outros mínimos, ditos locais. Com múltiplos objetivos, ou seja para n $\rangle$ 1, surge o problema adicional da impossibilidade de solução, devido a conflitos entre objetivos, ou o problema de determinar soluções de compromisso dentro de um universo de possibilidades, como funções de Pareto [Lin].

Geralmente, quando a equação (1) é aplicada à síntese de circuitos ou modelagem de componentes, as funções objetivo f além de serem altamente não lineares, envolvem múltiplos critérios, e a complexidade da solução do problema não permite soluções analíticas, mas sim soluções numéricas com o uso do computador digital. As operações numéricas tem precisão finita, acarretando erros que podem dificultar a deteção do mínimo,na medida em que os critérios de parada ficam sujeitos à incertezas não predizíveis a priori.

O problema de otimização é bem colocado quando a função objetivo **f** apresenta um único mínimo ou máximo, dentro das restrições de igualdade ou desigualdade, dadas por **g**. Isto significa adotar funções objetivo bem comportados, como funções de energia ou outras

funções convexas [Blum, pag.272, Biles pag. 10], tarefa nem sempre possível, especialmente na síntese de circuitos, devido a problemas relacionados ao uso de modelos incompletos ou conflitos entre especificações, tal como ocorre na maximização do ganho e banda passante de um amplificador, no estabelecimento de limites entre regiões de operação para modelos não lineares de componentes semicondutores. Estes problemas podem não ter solução, ou ainda podem várias soluções que precisam ser analisadas como possíveis soluções de compromisso entre os vários objetivos conflitantes. Isto aponta para a necessidade de ferramentas de projeto interativas, onde o projetista guia o processo de otimização, analizando e comparando os resultados dos diversos programas disponíveis para o caso.

As técnicas numéricas determinísticas usadas em otimização são baseadas na aproximação sucessiva da função objetivo, que não garantem:

- a convergência para uma possível solução, caso ela exista [[Acton, pag.448],
- a determinação de impossibilidade de solução, ou
- a determinação das múltiplas soluções ou curvas de Pareto [[Lin].

Estas considerações levam à questão de ser ou não determinística a natureza do problema de otimização via programas de computador, na medida em que cada tentativa de obtenção do ótimo de uma função objetivo vai depender, de modo quase aleatório, das condições iniciais de pesquisa do ótimo, da natureza das funções objetivo e do próprio desenvolvimento do algoritmo numérico empregado na pesquisa do ótimo. A proposta deste trabalho é tratar o problema de otimização em projeto de circuitos como estocástico, e usar métodos estatísticos de análise e otimização das possíveis soluções, que permitam uma medida de confiança (ou da incerteza) destas soluções. O método de Monte Carlo [Rubinstein, Hammersley] parece ser o mais adequado para esta proposta, tanto pela sua generalidade de aplicação, quanto pela sua eficiência no tratamento de problemas com grande número de variáveis], na medida em que a incerteza da sua solução diminue com a raiz quadrada do número de amostras, independentemente do número de variáveis envolvidas [Rubinstein pag.114; Hammersley pag. 50]. Além disso é especialmente adequado ao uso de técnicas de programação paralela, que permite reduzir o tempo de execução do grande número de simulações de circuitos, necessário para obtenção de resultados consistentes, estatísticamente significativos.

O amplo surgimento no mercado internacional de computadores paralelos de alto desempenho e custo relativamente baixo viabilizou a adoção de técnicas estatísticas, com análise simultânea de amostras [Nash] . Bibliotecas de programas de processamento paralelo, de tratamento estatístico e visualização de grande volume de dados, foram padronizadas para diversas aplicações, especialmente para a área de física de altas energias [Brun, CERN_], facilitando a construção de programas para aplicação de técnicas estatísticas a projetos de circuitos. Desta forma diversos programas podem ser desenvolvidos com garantir a qualidade dos sistemas eletrônicos, um dos grandes temas atuais na área de engenharia.

O algoritmo Simplex Não Linear para pesquisa de mínimos de funções não lineares, é considerado na literatura como adequado a sistemas sujeitos a erros de medidas ou de modelos, como usualmente ocorre no projeto de circuitos eletrônicos [Biles & Swain, Minuit, Hbook], porém apresenta uma lenta convergência para o mínimo e pequena precisão do seu resultado final. Isto torna o algoritmo um bom candidato para a otimização estocástica, na medida em que nela se necessita de um algoritmo com convergência garantida para cada

mínimo local, cujos valor exato não importa tanto quanto a estatística dos diversos valores encontrados para cada um, um mapa das possíveis soluções. Esta informação estabelece pontos iniciais e limites de pesquisa para algoritmos de gradiente, que oferecem maior precisão na solução final e podem ser usados no refinamento das soluções.

Este método foi implementado nos computadores paralelos ACP2 e IBM SP2, utilizando o programa CPSPICE para simular os circuitos de forma paralela, e os programas MINUIT e HBOOK da CERNLIB para a programação matemática e processamento estatístico de dados. Esta modularidade reduziu grandemente o esforço de desenvolvimento e validação do programa de implementação do método.

2 - O MÉTODO DE OTIMIZAÇÃO SIMPLEX NÃO LINEAR

2.1 - A escolha do método

A otimização do projeto de circuitos eletrônicos apresenta diversas exigências quanto ao emprego de métodos de otimização [Brayton, Mesquita]. O método deve ser :

- Capaz de tratar problemas oriundos de imprecisões nos modelos e nos dados, comuns na análise e síntese de circuitos eletrônicos., e que resultam em mínimos locais e outras anomalias das funções objetivo.
- Robusto em termos da convergência para mínimos locais, uma vez que trata de sistemas de equações diferenciais contínuas altamente não lineares.
- Independente do cálculo de derivadas analíticas, que em geral não estão disponíveis para os circuitos, nem numéricas, pois ou são imprecisas ou tem alto custo computacional.
- Eficiente em termos de tempo de execução e memória utilizada.

Estas considerações eliminam os métodos de gradiente com derivadas, e destacaram o método Simplex Não Linear como um forte candidato, pois permite a pesquisa do ponto de valor mínimo ou máximo de uma função multidimensional, sem necessitar do cálculo de derivadas, sendo relativamente imune à discontinuidades tanto da derivada da função objetivo quanto da prórpia função. Sua estrutura é muito simples em comparação com a de outros métodos, como pode ser visto no artigo original de Nelder & Mead, na implementação deste algoritmo em [Kuester], no programa Amoeba e no programa MINUIT da CERNLIB, este mais elaborado e validado pelo uso entre físicos de partículas de altas energias. A seguinte análise do algoritmo é feita aqui com base na geometria e na estatística.

2.2 - O Algoritmo Simplex Não Linear

O problema consiste em determinar o mínimo de f(x), uma função escalar de n parâmetros contidos no vetor $x = (x_1, x_2, ..., x_n)^t$. O método de solução consiste no cálculo interativo $x = x + \Delta x$ tal que $f(x + \Delta x) < f(x)$, tendo como critério de parada que a variação do valor da função objetivo fique abaixo de um dado limiar : $|| f(x + \Delta x) - f(x) || < \varepsilon$.

O incremento Δx no espaço de parâmetros deve ser feito portanto na direção que permita a redução do valor da função objetivo f(x). Enquanto os métodos de derivadas tomam a direção oposta ao do gradiente da função, no Simplex Não Linear a direção é tomada a partir da comparação dos valores da função nos (n+1) vértices de um **Simplex**, que é um conjunto de (n+1) pontos no espaço a n dimensões. O Simplex é dito **regular** se seus vértices são eqüidistantes, e **geral** em caso contrário. São usadas neste processo 4 operações básicas: Reflexão, Contração, Expansão e Redução, restritas ao espaço de parâmetros de projeto. Em cada iteração distingue-se os seguintes pontos do Simplex :

1 - P_M é maior valor da função objetivo. Ocorre no vértice do Simplex com coordenadas x_M

$$f(x_M) = \max f(x_i) > f(x_i) , \quad i = (0,1,2,...,n) , \ i \neq M \qquad (2)$$

2 - P_L é o segundo maior valor da função objetivo, que ocorre no vértice do Simplex com coordenadas x_L :

$$f(x_M) > f(x_L) > f(x_i) \quad , \qquad i = (0,1,2,...,n) \ , \ i \neq \{ L, M \} \qquad (3)$$

3 - P_m é o menor valor da função, que ocorre no vértice do Simplex com coordenadas x_m :

$$f(x_m) < f(x_i) , \qquad , \qquad i = (0,1,2,...,n) \ i \neq m \qquad (4)$$

4 - P_0 é o centróide de todos os pontos X_i , exceto para i = M : $P_0 = \frac{1}{n} \bullet \sum_{j=0}^{n} Pi \ , \ j \neq M$ (5)

Sobre estes pontos são realizadas as seguintes operações básicas,:

- Reflexão : o vértice P_M é deslocado para posição : $P_r = (1 + \alpha) P_0 - \alpha P_M , \ \alpha > 0$ (6)
- Expansão : caso seja possível uma melhora suplementar no valor da função objetivo na direção da reflexão, o vértice P_r é expandido para : $P_e = \gamma P_r + (1 - \gamma) P_0 , \ \gamma > 0$ (7)
- Contração : caso o valor da função objetivo não seja reduzido após a reflexão, o vértice refletido é levado para posição $P_c = \beta P_M + (1 - \beta) P_0 , \ 0<\beta<1$ (8)
- Redução : caso não ocorra nenhuma das outras possibilidades, aproxima-se igualmente todos os vértices do ponto P_m , $P_i = P_m + \beta(P_i - P_m) \ , \ i = 1...n \ , \ i \neq m \ , \ 0<\beta<1$ (9)

Cada uma dessas operações do algoritmo de Nelder-Mead, tal como apresentada na figura 1, é empregada para uma dada situação de pesquisa, numa seqüência de operações que leva à obtenção do mínimo, apresentada na figura 2 sob forma de fluxograma.

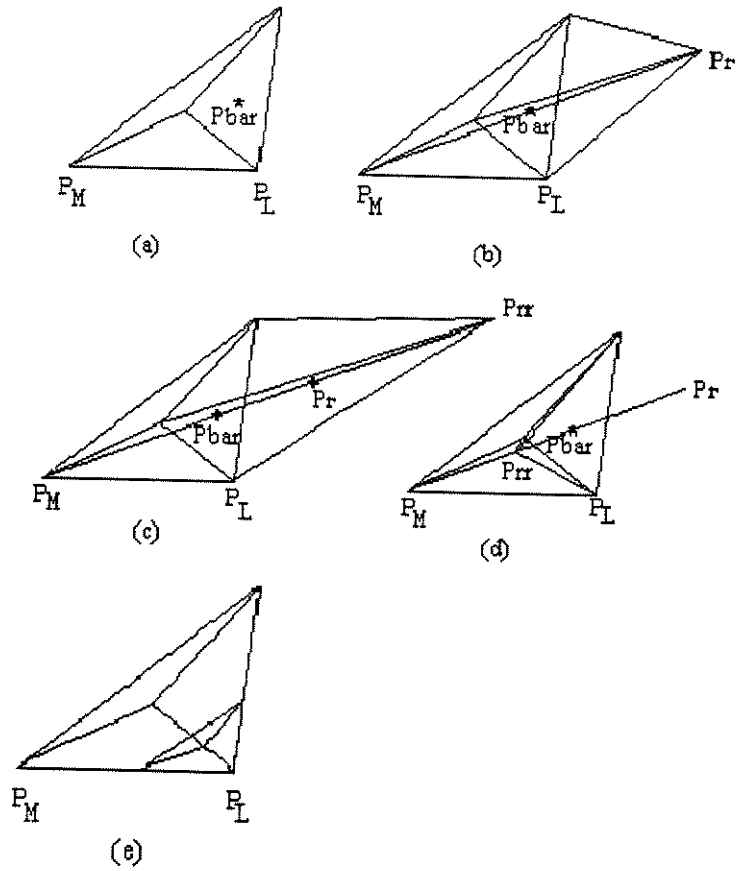

Figura 1: (a) Simplex do tipo tetraedro para funções com duas variáreis. Operações de : (b) reflexão; (c) expansão; (d) contração e (e) redução.

2.3 - Estrutura do algorítmo de minimização de Nelder - Mead : análise geométrica

Tomando por base uma analogia geométrica, o algoritmo de Nelder-Mead se inicia pela definição de um conjunto de N+1 pontos não colineares, no espaço de N parâmetros da função cujo mínimo se deseja obter. Um ponto inicial P_0 pode dar origem aos demais pontos segundo a fórmula :

$$P_i = P_0 + c^* e_i \qquad (10)$$

P_i (i = 0 a N) são os pontos de partida do algoritmo; c é o coeficiente do afastamento de P_i em relação a P_0 ; e_i é o vetor unitário de dimensão N correspondente à variável independente i. Os pontos P_i representam os vértices de um triângulo no caso de uma função com duas variáveis independentes, ou seja N = 2, um tetraedro no caso de N = 3, e um "hipertetraedro" para N > 3.. A cada interação do algorítmo classificam-se os pontos de acordo com o valor assumido pela função. Seja o de maior valor P_M, o de segundo maior P_L e o de menor valor P_m. Estes três pontos, juntamente com o ponto P_O, obtido a partir da média aritmética das coordenadas de todos os pontos com exceção do ponto P_M, criam a base de comparação para obter um novo ponto que substituirá o ponto P_M no novo conjunto de N+1 pontos, fazendo com que a pesquisa do mínimo da função progrida para uma solução.

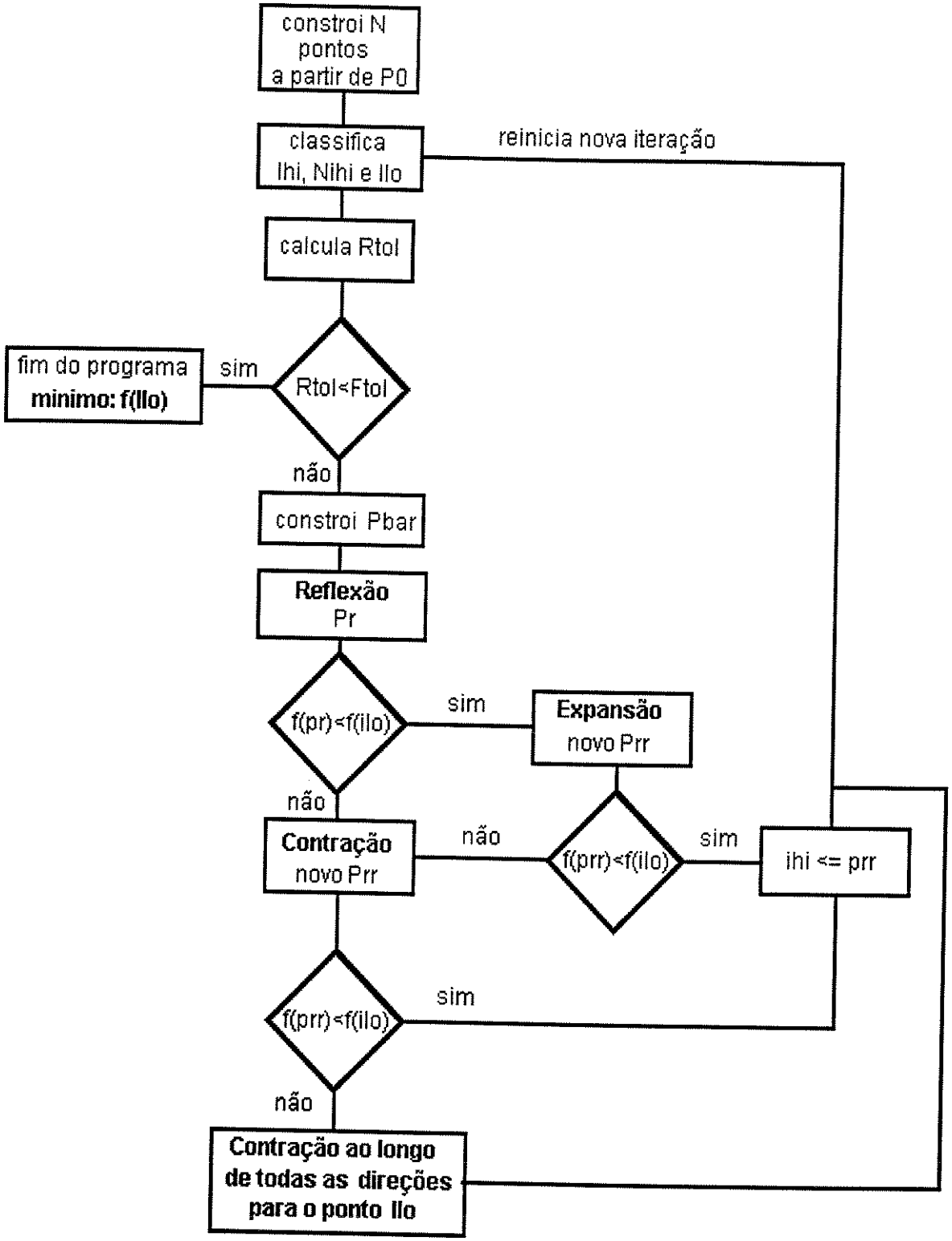

Figura 1 - Diagrama de fluxo do algoritmo de Nelder - Mead

Calculado P_0, a primeira etapa do algoritmo é a Reflexão, onde se calcula o ponto P_r, parte de um novo (hiper)tetraedo, como se fosse a reflexão do (hiper)tetraedro inicial na

direção oposta ao ponto P_M, segundo um parâmetro dado. Se o valor da função neste ponto for menor que no ponto P_m, passa-se para uma segunda etapa do algoritmo denominada Expansão, onde o parâmetro γ dá a escala de afastamento do ponto P_{rr} ao ponto P_M, passando pelo ponto P_r. P_{rr} é um novo vértice de um novo (hiper)tetraedro, substituindo P_M se o valor da função nele for menor do que em P_m ou se for menor que em P_L, ou ainda se estiver entre o valor neste e o de P_M. Se P_{rr} não satisfizer estas condições então o algoritmo realiza uma terceira etapa, a de Contração. Esta etapa utiliza o parâmetro β para calcular o ponto P_{rr} dentro do (hiper)tetraedro inicial, se este tiver valor menor do que em P_M, ele será o novo vértice no lugar de P_M.

O objetivo principal do algoritmo é encontrar um novo ponto para substituir o ponto P_M, que vai tendo valor de função mais reduzido, ou seja a cada interação deve-se obter valores cada vez menores para ele até se chegar no mínimo da função. Caso não se consiga um ponto que satisfaça algumas das restrições anteriores, faz-se a etapa de Redução, aproximando todos os vértices do (hiper)tetraedro do ponto P_m, segundo um fator η. Assim se obtém um (hiper)tetraedro em escala menor que a inicial, mantendo fixo o ponto P_m, e se reinicia a pesquisa.

O critério de convergência consiste em verificar se a diferença relativa dos pontos P_M e P_m, dada por R_{tol}, é menor do que F_{tol}, definido pelo usuário. Ou ainda se o número de interações feitas excede um limite estipulado, como pode ser visto no fluxograma apresentado na Figura 1.

2.4 - Análise estatística do algoritmo de Nelder-Mead

A seguinte análise probabilística do Simplex de Nelder-Mead [Quesada] visa dar uma outra visão do algoritmo, e mostrar alguns detalhes do método, como a influência das derivadas parciais da função objetivo no processo de convergência e da possibilidade de ocorrência de paradas indesejadas antes da convergência ao resultado ótimo

Por simplicidade, considera-se aqui a utilização do algoritmo em problemas onde a função objetivo embora convexa não é explícita e a escolha do ponto inicial é aleatória. Considerando que : a função objetivo um paraboloide como o mostrado na figura 2.b, e que o ponto inicial é obtido aleatóriamente, bem como todos os N+1 pontos da pesquisa, que estão dentro de um determinado (hiper)vale, como mostrado na figura 2.a, pode-se assumir como possível a convergência do algoritmo para o mínimo referente a este vale.

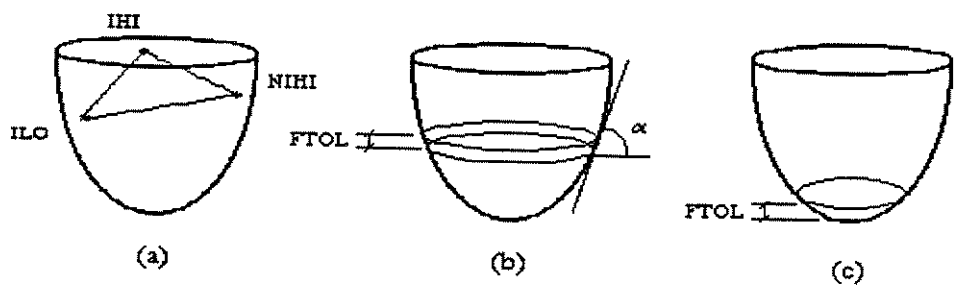

Figura 2 - Como calcular a probabilidade de ocorrer a parada da procura do mínimo

Seja uniforme a probabilidade de ocupação no espaço no parabolóide de um ponto obtido em uma interação do algoritmo. A distância deste ponto ao mínimo pode ser medida pela distância de sua projeção sobre o plano tangente ao ponto de mínimo. Nestas condições a distribuição estatística da posição da projeção do ponto neste plano é também uniforme. A parada do algoritmo se dá quando todos os pontos se situam dentro de uma região delimitada pela faixa F_{tol}, ou seja quando o valor da função objetivo for praticamente igual para todos os pontos. A probabilidade de ocorrência deste evento é portanto a da parada do algoritmo, a qual pode se dar antes de ser alcançado o mínimo, e corresponde ao produto da probabilidade de ocorrência de cada ponto dentro da região da faixa F_{tol}. A probabilidade de um ponto se situar dentro desta faixa é dada pela razão entre a diferença das áreas dos círculos C_1 e C_2, , que são projeções sobre o referido plano dos cortes que delimitam a região da faixa F_{tol}, tal como mostrada na Figura 3, e a área total da pesquisa do mínimo da função, delimitada . Os raios r_1 e r_2 são medidas das distâncias do ponto ótimo aos pontos P_m e P_{iho}, respectivamente. Se a diferença entre r_2 e r_1 for igual ou menor do que F_{tol}, considera-se atingido o ponto ótimo, cessando a pesquisa do mínimo.

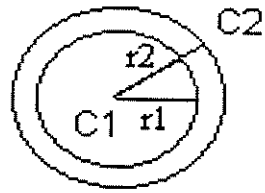

Figura 3 - Cálculo de probabilidade de parada indevida do algoritmo

A probabilidade de obter um ponto dentro da faixa F_{tol} pode ser calculada da seguinte forma. Os possíveis pontos gerados pelo algoritmo se situam dentro de um (hiper) paralelogramo de área At definido pelos limites mínimo e máximo de cada variável. Gerando pontos aleatoriamente dentro deste (hiper) paralelograma, verifica-se pelos respectivos valores da função quais estão situados dentro dos limites P_M e P_m, representados respectivamente pela áreas dos círculos C_2 e C_1 na figura 4.

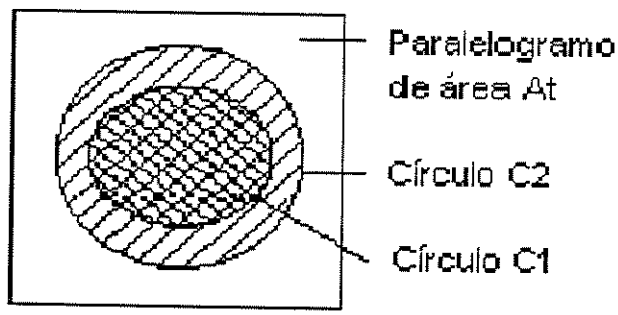

Figura 4 - Projeção dos limites do Simplex Não Linear

Seja A_1 a área contida por C_2, círculo que contém o ponto P_M e A_2 a área delimitada por C_1, círculo que contém o ponto P_m . Sejam eventos A a ocorrência de pontos P_i gerados pelo algoritmo dentro da faixa F_{tol}, isto é entre C_1 e C_2 , e eventos B a ocorrência de pontos P_i dentro de toda a região aonde se dá a pesquisa do mínimo, isto é dentro de C_2. Os pontos intermediários a P_M e P_m estão dentro da parte do parabolóide cuja projeção é delimitada

pelos círculos C_1 e C_2. A probabilidade de ocorrência de A condicionada a ter ocorrido B pode ser estimada pela relação de áreas:

$$P(A/B) = \frac{A_2 - A_1}{A_2}$$

Estas áreas podem ser estimadas pelo método de Monte Carlo pela relação do número de pontos situados nas áreas e o número total de pontos gerados na pesquisa que caem na área At:

$$A_1 = \frac{n1}{N} At \qquad A_2 = \frac{n2}{N} At = \pi \bullet r_2^2$$

$$\text{Para } A_1 = \pi \bullet r_1^2,\ A_2 = \pi \bullet r_2^2 \text{ e } N \to \infty \Rightarrow P(A/B) = 1 - (r_1 / r_2)^2 \to \frac{n2 - n1}{n2}$$

Para funções mais complexas analogamente aos círculos consideram-se curvas de níveis ao longo da superfície de estudo, delimitando assim a região provável de ocorrência do evento A tendo ocorrido o evento B. Este cálculo de probabilidade pode ser feito pelo método de Monte Carlo, que permite a integração de áreas, quaisquer que seja a geometria.

Como resultado desta análise observa-se a correlação com a probabilidade P(A/B) da derivada da função objetivo, dada na figura 2 pelo ângulo α. Quando a pesquisa está longe do mínimo a derivada da função objetivo neste caso apresenta altos valores, e α tende a 90° , ou seja a área $A_2 - A_1$ e portanto P(A/B) tendem a 0%. Perto do mínimo a derivada da função objetivo tende a 0, α tende a 0, e P(A/B) tende para 100%. Isto é se ao longo da pesquisa os pontos se encaminham para o mínimo, a derivada diminui ou seja α vai se aproximando cada vez mais do zero até que P(A/B) fica próxima de 100% que é a probabilidade do critério de parada ser atendido. Esta relação direta de P(A/B) com o gradiente significa que este método trabalha indiretamente com as derivadas parciais da função objetivo na procura do mínimo. Outro fato importante de ser abordado é que este método não garante totalmente a chegada ao mínimo, pois como foi visto existe uma probabilidade não nula do algoritmo parar antes de chegar no ponto ótimo. Deve-se observar que mesmo sendo pequena esta probabilidade sempre existe.

Um fato importante de ser considerado é que neste método não se garante a convergência para o mínimo, pois como foi visto a probabilidade de parada antes de chegar no ponto ótimo existe, apesar de pequena.

Concluindo temos as seguintes observações relativas ao algoritmo Simplex de Nelder-Mead :

1 - Devem ser utilizados um grande número de pontos iniciais, altamente dispersos, para não restringir a pesquisa do mínimo global.
2 - Os parâmetros α, β, γ, η e F_{tol} devem ser ajustados para acelerar e reduzir o valor final do erro de convergência.
3 - O indicador R_{tol} deve convergir para a mesma ordem de grandeza de F_{tol}.
4 - Devem haver mecanismos para tratar de paradas do algoritmo antes deste alcançar o ponto ótimo.

2.5 - Tratamento dos mínimos Locais

Em geral os algoritmos de otimização convergem para mínimos locais próximos da condição inicial dada. Isto significa que o algoritmo pode convergir para um mínimo que não satisfaça a tolerância especificada para a função objetivo. Nestes casos a otimização pode ser reiniciada utilizando-se um fator de escala maior para o Simplex. Este procedimento pode ser repetido até que o algoritmo convirja para um mínimo que atenda as especificações requeridas. As possibilidade de convergência para o mínimo global aumentam na medida em que a busca se dê a partir de um maior número de pontos iniciais.

2.6 - Implementação e testes do algoritmo de Nelder-Mead

O MINUIT é o programa de otimização da CERNLIB. Foi feito um teste comparativo destas rotinas em um problema de caracterização de diodos e transistores semicondutores, com base em um programa em FORTRAN que chama o MINUIT, e fornece os dados medidos e o modelo utilizado para o diodo. Os valores ótimos para os parâmetros do modelo do diodo são aqueles que minimizam o erro entre os dados experimentais e os valores simulados com o modelo.

Foram testadas rotinas que implementam os métodos Simplex e de gradiente, estes usando derivadas numéricas. Os programas utilizados e os resultados deste teste estão apresentados em anexo.

Pode-se notar que o triângulo do gráfico formado pelos valores finais de VT e Is fica mais estreito quanto mais próximo do ponto de mínimo absoluto da pesquisa. As dificuldades de obtenção do mínimo absoluto se devem aqui ao fato do modelo não ser válido para todas as regiões de polarização. Para modelar a região de baixa injeção seria necessário ter acrescentado pelo menos uma resistência em paralelo com a fonte de corrente dependente da tensão de forma exponencial e para a região de alta injeção uma resistência em série. Como os pontos coletados em laboratório abrangem todas estas regiões isto prejudicou a obtenção do mínimo absoluto .

O ponto de menor erro levou mais de 200 interações para ser encontrado e as demais pesquisas não chegaram perto deste valor, pararam bem antes com um erro que varia entre 18% e 20% como é observado nos gráficos dos valores de I_S e V_T em função do erro .

Esta análise dos resultados é importante para verificação da validade do modelo, e também nem sempre o mínimo absoluto é o único ponto de interesse , podemos estar interessados em alguns mínimos locais para produzir outras soluções de compromisso. Se o projetista conhece antecipadamente a região em que um componente que está sendo modelado vai trabalhar, ele pode verificar o mínimo local que engloba esta região e adaptar seus parâmetros baseado nesta informação .

3 - ALGORITMO SIMPLEX ESTOCÁSTICO

O algoritmo proposto envolve diversos conceitos básicos:
1. determinação em paralelo de mínimos locais.
2. regionalização por atração do Simplex Não Linear para agrupamentos de mínimos locais.
3. determinação do mínimo global ou dos pontos de Pareto.

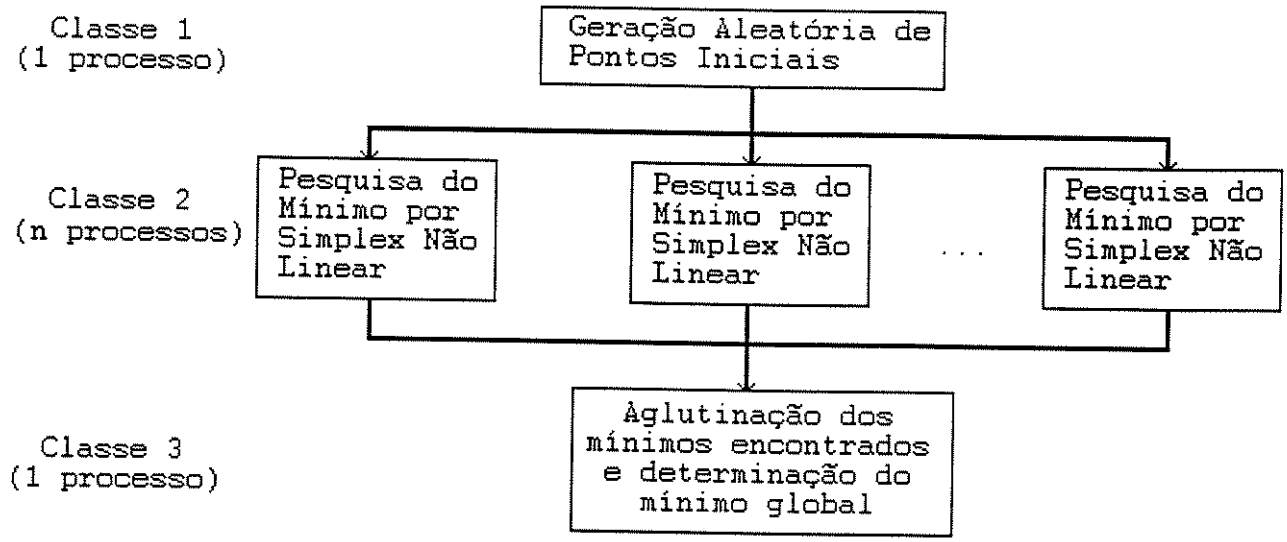

Figura 5 - Diagrama de fluxo da Programação matemática paralela do Simplex Não Linear

3.1 - Paralelização do algoritmo Simplex Não Linear

Para desenvolver uma rotina paralela precisa-se identificar as tarefas independentes entre si, que possam ser executadas simultaneamente por diversos processadores. Uma destas tarefas é a etapa de criação dos pontos do primeiro Simplex, a partir de um ponto inicial, segundo direções independentes, e o cálculo dos valores da função nestes pontos, que podem ser feitos em paralelo. Por outro lado estes pontos iniciais podem ser criados aleatoriamente sobre toda a região na qual o mínimo será pesquisado, fazendo-se em paralelo, a partir de cada um deles, pesquisas dos mínimos locais. Estas duas formas de paralelismo foram adotadas neste trabalho. A realização em paralelo de etapas do algoritmo fica descartada na medida em que implica na modificação de programas padronizados. Além disso cada etapa deste algoritmo depende da anterior, a menos que o algoritmo seja modificado para comportar a criação e pesquisa simultânea de diversos pontos novos, configurando uma frente de onda a ser analisada em paralelo para determinar qual ou quais pontos são candidatos à expansão da pesquisa, e quais devem ser abandonados, uma extensão do método de "branch and bound". Isto corresponde a um novo método, que poderá ser implementado no futuro.

3.2 - Regionalização do algoritmo Simplex Não Linear

Na etapa final do Simplex Não Linear Paralelo, cada vez que um Simplex convergir para um mínimo local, delimita-se toda a região pesquisada como sendo região de atração

para o mínimo, definida pela união dos limites externos dos Simplex criados no processo que levou à esta convergência. As regiões de atração de todos os mínimos encontrados ficam proibidas para criação de pontos iniciais para pesquisa de novos mínimos, que continua até que o espaço de busca seja reduzido a zero, ou se exceda um limite de tentativas. Nota-se que as pesquisas do mínimo, a partir de um ponto inicial aleatório, são independentes entre si, e podem ser executadas em paralelo como processos distintos. O problema aqui consiste na aglutinação dos resultados destas pesquisas, pelo agrupamento dos mínimos feito da seguinte forma : o último Simplex criado antes do final da convergência define a incerteza daquele mínimo. Quando dois mínimos apresentam entrelaçamento dos respectivos Simplex finais, eles são considerados como duas realizações do mesmo mínimo local, e as suas regiões de atração são unidas

3.3 - Seleção e Classificação de Mínimos e Regiões de Atração Associadas.

A união dos resultados de cada pesquisa independente fornecerá um mapa dos mínimos locais, dos quais será retirado o mínimo global como aquele que apresentar o mais baixo valor da função objetivo. Dará também os limites dos vales relacionados a cada mínimo e portanto uma visão global das possibilidades da síntese pretendida para fins de projeto ou de
modelagem e caracterização de componentes e circuitos.

4 - EXEMPLOS E CONCLUSÕES

Os testes realizados com as rotinas do MINUIT demonstraram que a rotina SIMPLEX é adequada a esta aplicação, mostrou-se bastante robusta e menos dispendiosa de recursos computacionais do que rotinas baseadas em métodos de gradiente, nos casos de tolerância grande aos erros de convergência.

O método de gradiente implementado pela rotina MIGRADE convergiu corretamente apenas quando os valores iniciais dos parâmetros eram próximos dos correspondentes ao mínimo erro. A precisão é muito maior do que os demais método para problemas exatos, e quando implementado com o cálculo analítico das derivadas parciais da função objetivo em relação aos parâmetros. Entretanto para problemas em que o modelo não é exato seu desempenho pode ficar inferior ao de Nelder-Mead tanto em precisão quanto em velocidade de convergência, além de deixar de convergir em um maior número de vezes. Isto o torna um algoritmo caro em termos computacionais e de utilidade relativa para o projeto de circuitos eletrônicos.

Um esquema conveniente é ter os pontos iniciais de MIGRADE definidos pelo SIMPLEX ficando a aceitação dos resultados de um ou de outro por conta do erro da função objetivo. Entretanto este esquema se revelou durante os testes útil apenas para um número bastante limitado de circuitos, ficando a aplicação do SIMPLEX como suficiente para alcançar a maior precisão possível na determinação de valores de parâmetros de circuitos em projeto ou em modelagem.