Luiz Gustavo Turatti

SIMULAÇÃO ACELERADA DE BAIXO CUSTO PARA APLICAÇÕES EM NANOENGENHARIA DE MATERIAIS

CAMPINAS

2013

UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

Luiz Gustavo Turatti

SIMULAÇÃO ACELERADA DE BAIXO CUSTO PARA APLICAÇÕES EM NANOENGENHARIA DE MATERIAIS

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Doutor em Engenharia Elétrica, na área de Eletrônica, Microeletrônica e Optoeletrônica.

Orientador: Prof. Dr. Jacobus Willibrordus Swart
Co-orientador: Prof. Dr. Stanislav Moshkalev

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL
DA TESE DEFENDIDA PELO ALUNO LUIZ GUSTAVO TURATTI
E ORIENTADA PELO PROF. DR. JACOBUS WILLIBRORDUS SWART

_____

CAMPINAS

2013

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva – CRB 8/5974

Turatti, Luiz Gustavo, 1977-
T84s Simulação acelerada de baixo custo para aplicações em nanoengenharia de materiais / Luiz Gustavo Turatti. – Campinas, SP : [s.n.], 2013.

Orientador: Jacobus Willibrordus Swart.
Coorientador: Stanislav Moshkalev.
Tese (doutorado) – Universidade estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Computação de alto desempenho. 2. Programação paralela (computação). 3. Simulação por computador. 4. Método de Monte Carlo. 5. Feixes de íons focalizados. I. Swart, Jacobus Willibrordus, 1950-. II. Moshkalev, Stanislav, 1952-. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Low cost accelerated simulation for application in nanoengineering materials
Palavras-chave em inglês:
High performance computing
Parallel programming
Computer simulation
Monte Carlo Method
Focused íon beam
Área de concentração: Eletrônica, Microeletrônica e Optoeletrônica
Titulação: Doutor em Engenharia Elétrica
Banca examinadora:
Stanislav Moshkalev [Coorientador]
João Carlos de Moraes Morselli Junior
Marcelo Antonio Pavanello
José Alexandre Diniz
Luiz Otávio Saraiva Ferreira
Data da defesa: 29-11-2013
Programa de Pós-Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE DOUTORADO

Candidato: Luiz Gustavo Turatti

Data da Defesa: 29 de novembro de 2013

Título da Tese: "Simulação Acelerada de Baixo Custo para Aplicações em Nanoengenharia de Materiais"

Prof. Dr. Stanislav Moshkalev (Presidente): _____

Prof. Dr. João Carlos de Moraes Morselli Junior: _____

Prof. Dr. Marcelo Antônio Pavanello: _____

Prof. Dr. José Alexandre Diniz: _____

Prof. Dr. Luiz Otávio Saraiva Ferreira: _____

RESUMO

Este é um trabalho multidisciplinar que aborda questões de química, física, engenharia elétrica (nanoengenharia) e principalmente avanços obtidos com simulações por computador. Os programas comumente utilizados para simulações de fótons/íons focalizados em outro material consomem recursos computacionais por diversas horas ou até dias, para concluir os cálculos de determinado experimento, como a simulação de um processo efetuado com o equipamento FIB/SEM (*Focused Ion Beam/Scanning Electron Miscroscopy*), por exemplo.

Através do uso de ambientes computacionais virtualizados, associados a programação paralela em CPU (*Central Processing Unit*) e GPGPU (*General Purpose Graphics Processing Unit*) é possível reduzir significativamente o tempo da simulação de horas para minutos, em situações de interação de partículas, que envolvem aproximação de colisões binárias (BCA, *Binary Collision Approximation*) e o Método de Monte Carlo (MMC), principalmente.

O uso de placas gráficas (comumente utilizadas para jogos) potencializou o poder de processamento numérico para uso acadêmico a baixo custo, reduzindo o tempo para obtenção de resultados que foram comprovados experimentalmente.

A utilização de programas análogos que empregam BCA e MMC, tais como TRIM/SRIM (*Transport of Ions in Matter*, atualizado para *Stopping and Range of Ions in Matter*), MCML (*Monte Carlo for Multi Layered media*) e CUDAMCML (*Compute Unified Device Architecture*, MCML) auxiliam a comparação de ganho de desempenho entre CPU e GPGPU evidenciando o melhor desempenho desta última arquitetura, com CUDA.

Em simulações equivalentes com matrizes esparsas executadas em CPU e GPGPU, a redução do tempo de processamento variou entre três e quinze mil vezes, respectivamente. Com o Método de Monte Carlo, a redução foi de até cento e quarenta e uma vezes para melhores resultados. As simulações de alto desempenho e baixo custo computacional permitem antever algumas situações experimentais, diminuindo a necessidade de explorar todas as possibilidades práticas e, dessa forma, reduzindo o custo com laboratório.

Palavras-chave: aceleração, feixe de íons focalizados, FIB/SEM, GPU, GPGPU, MCML, Método de Monte Carlo, otimização, computação de alto desempenho, programação paralela (computação), simulação por computador, TRIM/SRIM.

ABSTRACT

This is a multidisciplinary work that addresses issues of chemistry, physics, electrical engineering (Nanoengineering) and especially advances obtained with computer simulations. Programs commonly used for simulations of photons/ions focused onto other materials consume computational resources for several hours or even days, to complete the simulations of a process performed with the equipment FIB/SEM (Focused Ion Beam/Scanning Electron Miscroscopy), for example.

Through virtualized computing environments associated with parallel programming on CPU (Central Processing Unit) and GPGPU (General Purpose Graphics Processing Unit) is possible to significantly reduce the simulation total time from hours to minutes in the interactions of particles, involving binary collision approximation (BCA) and Monte Carlo method (MMC), mostly.

The use of graphics cards (generaly used for games) enhanced the numerical processing power to be used in academia with low cost and reduced the time to obtain results experimentally verified.

The use of similar software using BCA and MMC, such as TRIM/SRIM (Transport of Ions in Matter, upgraded to Stopping and Range of Ions in Matter), MCML (Monte Carlo for Multi Layered media) and CUDAMCML (Compute Unified Device Architecture, MCML) helped us to make a comparison of performance between CPU and GPGPU showing the best performance of the latter architecture, with CUDA.

In equivalent simulations using sparse matrices in CPU and GPGPU, the time reduction of processing varied between three and fifteen thousand times, respectively. With the Monte Carlo method, reduction was up to one hundred forty one times for best results. Simulations of high performance and low computational cost allow us to predict some experimental situations, reducing the need to explore all practical possibilities and thus, reducing the lab costs.

Keywords*: high performance computing, focused íon beam, FIB/SEM, GPU, GPGPU, MCML, Monte Carlo Method, optimization, , parallel programming (computer), computer simulation, TRIM/ SRIM.*

SUMÁRIO

AGRADECIMENTOS

À DEUS, sempre;

Agradeço ao Prof. Dr. Jacobus Willibrordus Swart, pela orientação, confiança, apoio, paciência e amizade;

Ao Prof. Dr. Luiz Otávio Saraiva Ferreira, pela introdução a GPGPU, acesso aos primeiros equipamentos com suporte a CUDA, apoio, paciência e amizade;

Ao Prof. Dr. Stanislav Moshkalev, pela coorientação, apoio, paciência e valiosas sugestões;

Aos meus pais, Sandra e Wilson, minha irmã Ana Beatriz e minha noiva Maria Fernanda, pelo amor, incentivo, companheirismo e compreensão em todos os momentos;

Aos meus colegas de trabalho Alfredo Rodrigues Vaz, Frederico Hummel Cioldin e Marco Aurélio Keiler, e aos colegas do Departamento de Mecânica Computacional da FEM, Giovani Bernardes Vitor, Hugo Sakai Idagawa, Josué Labaki e Liliana de Ysasa Pozzo; e aos colegas de trabalho pelo apoio, sugestões e discussões produtivas.

À secretária do DSIF, Jaqueline Bisson Ercolini Lopes, pelas instruções, atenção e ajuda nos momentos que precisei;

À equipe do Centro de Componentes Semicondutores (CCS) que sempre auxiliou no esclarecimento de dúvidas sobre processos de micro&nanofabricação;

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), pela apoio financeiro recebido nesta etapa de aperfeiçoamento.

“Nenhuma linguagem de programação é perfeita, tampouco há uma que seja melhor;
Existem aquelas que simplesmente atendem fins específicos.”
(Herbert Mayer)

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CNPq	Conselho Nacional de Desenvolvimento Científico e Tecnológico
CPG	Comissão de Pós-Graduação
CPU	Central Processing Unit, Unidade Central de Processamento
DSIF	Departamento de Semicondutores, Instrumentos e Fotônica
FEEC	Faculdade de Engenharia Elétrica e de Computação
FEM	Faculdade de Engenharia Mecânica
FLOPS	Float point Operations Per Second, Operações com números reais por segundo
GPU	Graphical Processing Unit
GPGPU	General Purpose Graphical Processing Unit
IPS	Instructions Per Second, Instruções por segundo
SRIM	Stopping and Range of Ions in Matter
TRIM	Transport of Ions in Matter
UNICAMP	Universidade Estadual de Campinas

Capítulo 1

INTRODUÇÃO

A pesquisa sobre novos materiais e melhoria de processos em micro e nanofabricação envolve ensaios numéricos e práticos, revisões de teorias e de trabalhos em andamento que definirão o foco para novas realizações, com sucesso em resultados inovadores.

A nanoengenharia traz aos pesquisadores além da oportunidade de redução da escala dos processos estudados na academia e indústria, a oportunidade de aprendizado e renovação dessas metodologias, seja de maneira prática ou através de simulações computacionais.

As simulações computacionais projetam os possíveis resultados numéricos de determinados processos (físico-quimicos, por exemplo), conforme os parâmetros de entrada desejados e métodos escolhidos pelo usuário, que também podem ser representados por figuras ou gráficos. O sucesso do modelo elaborado pode ser avaliado através da comparação com resultados experimentais. Dessa forma é possível reduzir custos com equipamentos e manutenções através da verificação de procedimentos antes de sua execução prática.

1.1 Motivação

Nanoengenharia é a extensão da aplicação da nanotecnologia, explorando a manipulação de materiais em escalas menores, geralmente entre 0,2 e 100 nanometros (um nanometro equivale a 10^{-9} metro). Isto permite explorar, pesquisar, criar e desenvolver novos materiais e novos dispositivos. Entretanto o custo do uso de equipamentos para estas pesquisas é elevado, a exemplo do uso do equipamento FIB/SEM (*Focused Ion Beam/Scanning Electron Miscroscopy*; apresentado no capítulo 2, tabela 2.4), principalmente se os estudos forem somente empíricos (aprendizado feito sob circunstância de acertos e erros).

Para evitar desperdícios, a redução de custos em laboratório é primordial para a manutenção dos equipamentos em funcionamento e o apoio da computação torna-se muito importante para melhorar processo e antever situações experimentais com precisão, reduzindo o número de testes para definição de modelos no processo de nanofabricação.

A principal motivação deste trabalho é a redução de tempo necessário para simulações em computador dos processos de nanofabricação desenvolvidos e pesquisados no CCS/UNICAMP (Centro de Componentes Semicondutores da Universidade Estadual de Campinas), com a melhor utilização possível dos recursos financeiros disponíveis para tal. Com a interação contínua do trabalho de pesquisa e simulação é possível aprimorar os modelos de processos de nanofabricação reduzindo consequentemente os custos com laboratório.

Este trabalho pretende demonstrar principalmente os ganhos obtidos com simulações em GPGPUs (*General Purpose Graphics Processing Units*) e as possíveis melhorias em programas que abordam a interação de íons com a matéria.

1.2 Proposta do trabalho

Avaliação da infraestrutura computacional e dos programas disponíveis para uso gratuito ou com baixo custo de aquisição, que possuam preferencialmente o código-fonte aberto para auxiliar na tomada de decisão de projetos de nanoengenharia, tais como métodos de distribuição de processamento, processamento paralelo; e os programas TRIM/SRIM (*Transport of Ions in Matter*, posteriormente chamado de *Stopping and Range of Ions in Matter*) [1], MCML (*Monte Carlo for Multi Layered media*) [2,3] e CUDAMCML (*Compute Unified Device Architecture MCML,* ou seja, o mesmo programa ajustado para funcionar em GPGPU) [4], além de outros programas livres disponíveis no repositório do laboratório Sandia [5] (devido a complexidade de codificação, tais programas não foram amplamente explorados para este trabalho). Isto foi feito com a revisão teórica sobre a física envolvida nos processos de nanofabricação e da resolução numérica dos modelos envolvidos nos processos de simulação computacional compreendidos pela fabricação e caracterização de nanoestruturas baseadas em carbono em contato com eletrodos metálicos.

Considerando a complexidade das iterações numéricas envolvidas para os cálculos e a recursividade envolvida nesses cálculos, onde resultados de uma iteração são importantes para os cálculos seguintes, a dinâmica molecular e comportamento das colisões binárias e suas consequências nos átomos adjacentes envolve muitos conceitos, tornando difícil a compreensão de todo o processo para a escrita de um novo programa para computador a partir do zero. Para

isto foram pesquisadas as possibilidades gratuitas para obter códigos fonte como exemplo para testar sua eficiência em diferentes cenários apresentados neste trabalho.

Devido à necessidade de vários dias para obter resultados de problemas com grande volume de dados numéricos (por exemplo, o uso do TRIM/SRIM para número total de íons definido como 10^7; sendo este o limite do programa) em um processador convencional, observou-se que a arquitetura das GPGPUs é uma alternativa excelente para simulações em tempo real, porque o processador gráfico é otimizado para cálculos nestes dispositivos, permitindo modularizar de maneira eficiente o tratamento do grande volume de dados numéricos, reduzindo o tempo para algumas horas.

São apresentadas as possibilidades para simulação computacional com baixo custo, comparado ao investimento em equipamentos para laboratório, na ordem de dez mil reais para início de pesquisa com equipamento e programas, assim como as soluções que pretendemos desenvolver com parcerias e equipes multidisciplinares para aperfeiçoamento dos programas estudados. Destacaram-se as simulações efetuadas com TRIM/SRIM e a possibilidade de redução do tempo necessário para simulação em função da aceleração obtida com GPGPUs em relação a CPUs (*Central Processing Unit*) convencionais, seja otimizando a utilização de um programa, seja em relação a reescrita de parte do código-fonte de programa para computador em si, discutido no capítulo 4. Além da otimização do programa é necessária a distribuição do trabalho utilizando licenciamento diferenciado (por exemplo, atualmente o TRIM/SRIM é distribuído como *freeware*, ou seja, uso gratuito com código-fonte fechado; para receber melhorias e atualizações, requer adequação da licença de distribuição), acompanhado de documentação eletrônica detalhada, que permita ajustes e aprimoramentos do trabalho desenvolvido.

Dentre as alternativas pesquisadas, há possibilidade de continuação deste trabalho através da integração de melhorias propostas em uma única interface, com as soluções discutidas no capítulo 4 e apresentadas na conclusão do trabalho.

1.3 Organização da tese

Este trabalho encontra-se dividido da seguinte forma:

No capítulo 2 são mostrados os princípios e teorias do freamento de íons na matéria e equipamentos para nanofabricação.

No capítulo 3 há um panorama da evolução tecnológica, sendo abordado o progresso dos processadores, da supercomputação, da liguagem de programação, processamento paralelo e virtualização e dos processadores gráficos. Há, ainda, explanações sobre o processo de simulações, métodos numéricos e simulações com TRIM/SRIM e com MCML.

O capítulo 4 contém discussões e resultados das pesquisas realizadas e experiências contemplando virtualização e comparação de resultados de simulações em computador entre CPUs e GPGPUs na resolução do Método de Monte Carlo em programas como o TRIM/SRIM e MCML.

O capítulo 5 apresenta as conclusões, perspectivas e trabalhos futuros.

Capítulo 2

REVISÕES E TEORIAS

O estudo da química, desde a Grécia antiga até os dias atuais permitiu o desenvolvimento do modelo atômico vigente, através das teorias de Antoine Laurent de Lavoisier, John Dalton, Joseph John Thompson, Ernest Rutherford e Niels Henrick David Bohr. No início do século XX, Niels Bohr postulou sobre o comportamento de elétrons em torno de um núcleo positivo e sobre a perda de energia quando uma partícula transpassa outro material, buscando o equilíbrio eletrônico [6,7]; Hans Albrecht Bethe e Felix Bloch abordaram as teorias de Thomson e Bohr sob a perspectiva da mecânica quântica [8,9]. No mesmo século, foram realizados estudos abordando a análise de fragmentos da fissão e freamento de partículas num gás de elétrons livres. A partir dos estudos da química surgiram as teorias de Bohr e Bethe sobre a análise individual da colisão do íon com os elétrons do átomo. Esse foi o ponto de partida para a evolução dos diversos modelos existentes: correção de camadas de Fano, teoria de J.F.Lindhard, M.Scharff e H.E.Schiott (LSS) [10,11], os modelos de Northcliffe e Schilling [12] além do modelo de James F. Ziegler, Jochen P. Biersack e U. Littmark (ZBL) [1], teoria da aproximação de convolução unitária (por T. Schiwietz e P.L.Grande) [13] e teoria binária (por A.Schinner e P.Sigmund) [14]. Esses tópicos serão abordados neste capítulo.

2.1 Princípios do freamento de íons na matéria

Desde os primórdios da humanidade, sempre houve curiosidade do homem sobre sua origem e sobre como as coisas são feitas, ou seja, sobre o que constitui a matéria. Há registros de que os gregos, há mais de 2400 anos, iniciaram tais estudos empiricamente. Da observação dessas vivências e de elementos biogênicos (que geram vida) surgiu a alquimia, cuja continuidade é explorada até os dias de hoje, através da evolução da química com a comprovação científica necessária para registrar novos conhecimentos.

Robert Boyle em 1661, através da obra "O químico cético" (*The Sceptical Chymist*) publicou os primeiros princípios básicos da química, cuja continuidade dada por Antoine Laurent

Lavoisier, aproximadamente um século mais tarde, através da lei de conservação de massa, moldou a química como a conhecemos hoje. Lavoisier é considerado o pai da química experimental [15].

Outros filósofos como Leucipo de Mileto, Demócrito de Abdera, Epicuro de Samos e Lucrécio (Titus Lucretius Carus) acreditavam na existência do átomo, elemento indivisível, cujo conjunto de partículas do mesmo tipo compunha a matéria. O atomismo buscava naquela época desvencilhar o pensamento sobre a origem das coisas vinculado à teologia. Assim, em 1803, John Dalton elaborou a primeira tabela de pesos atômicos, e com ela a idéia de que átomos compunham a matéria. Em continuidade a este trabalho, é importante destacar que em 1897, Joseph John Thompson propôs seu modelo considerando que o átomo era, divisível, em partículas carregadas positiva e negativamente. Thompson acreditava que o átomo era composto por vários elétrons incrustados e embebidos em uma grande partícula positiva, como passas em um pudim (figura 2.1). Tal modelo permaneceu como válido até 1911, com a descoberta do núcleo atômico por Ernest Rutherford (figura 2.2). Em continuidade a este trabalho, em 1913, Niels Henrick David Bohr conseguiu interpretar algumas das propriedades das séries espectrais do hidrogênio e a estrutura do sistema periódico dos elementos. Bohr explicava o modelo de Rutherford considerando a teoria quântica.

Dessa forma, pode-se esclarecer que os conceitos seguintes consideram que um elemento químico é um conjunto formado por átomos que possuem o mesmo número atômico (Z). Cada elemento é reconhecido por um símbolo. A figura 2.3 ilustra genericamente como cada elemento químico é representado na tabela periódica.

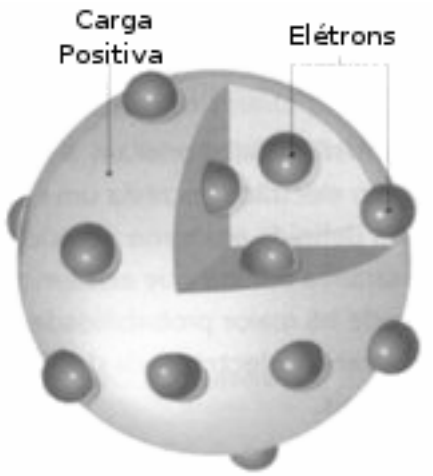


Figura 2.1: Modelo atômico de J.J.Thompson

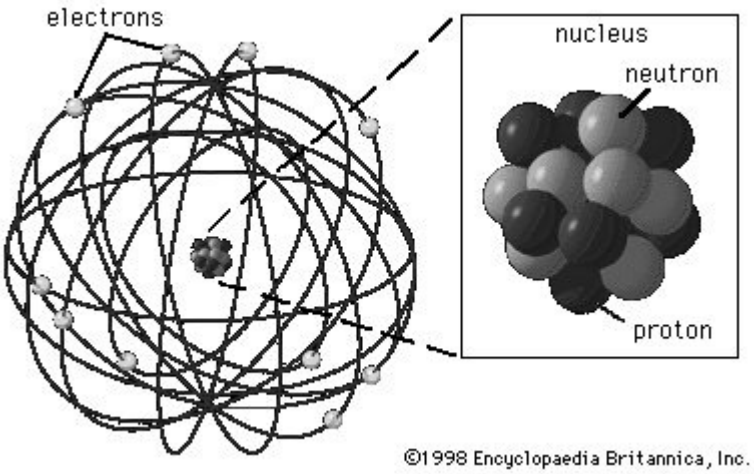


Figura 2.2: Modelo atômico de Rutherford

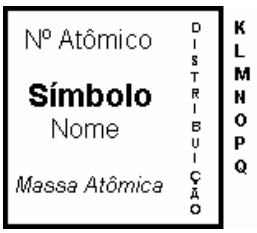


Figura 2.3: Identificação de elemento atômico. Todo elemento na tabela periódica é identificado por seu número atômico e respectivas características. (anexo A). O primeiro elemento descrito foi o Hidrogênio, representado pelo símbolo H.

A tabela periódica traz uma enorme quantidade de elementos químicos. A maioria dos elementos é encontrada na natureza e são conhecidos como Elementos Naturais. Alguns elementos cujos átomos são criados artificialmente, em laboratórios, são chamados de Elementos Sintéticos. O processo de criação desses elementos é conhecido como síntese.

Alguns autores relacionam o início da história da interação de partículas com a matéria aos estudos sobre balística de projéteis iniciado há mais de 400 anos por Leonardo DaVinci (~1580) [16], através da pesquisa sobre como aumentar o alcance de projéteis lançados numa catapulta. Essa relação foi feita em virtude da análise da penetração de um dado projétil num alvo, com a maior eficiência possível e assim comparado com a penetração de uma partícula na

matéria. Quando considerada a história mais recente, temos a descoberta de partículas radioativas (1895) [17] e certas terminologias tradicionais que serão utilizadas.

São elas:

Íon: é um átomo em movimento, eletricamente carregado. Também tratado como projétil.

Átomo: qualquer átomo em seu estado original, cujo número atômico e massa permanecem similares àqueles encontrados na tabela periódica. Também tratado como alvo ou átomos da matéria.

Freamento: abordado inicialmente como poder de freamento (*stopping power*) ou atualmente abordado como força de freamento (*stopping force*) [18]; é a taxa de perda de energia de um íon em um alvo, definida pela equação dE/dx (equação 2.1), onde x é a profundidade.

Dentre as diversas maneiras existentes para aplicar feixes de íons em materiais para o estudo de microeletrônica, estudaremos o SRI (*Stop Range Ions in matter*) que é o estudo do freamento da interação entre partículas.

Ao passar através da matéria, o íon interage com os demais átomos da rede cristalina perdendo energia. Esse freamento da partícula é definido através da medida de unidades de energia por comprimento (por exemplo, MeV/cm), definido pela equação 2.1.

O freamento depende do tipo e da energia da partícula, além das propriedades do material por onde passa a partícula. Tanto elétrons como íons positivos perdem energia quando passam através da matéria. Logo, o freamento é tratado como uma propriedade do material, enquanto a perda de energia descreve o comportamento da partícula. Para representar isto, temos a equação 2.2 [19], que define o poder de freamento entre partículas (devido ao potencial de repulsão interatômico).

$$S(E) = -\frac{dE}{dx} \tag{2.1}$$

S é um número positivo que representa o freamento da partícula;

E representa a energia envolvida no processo;

dE representa a variação de energia; e

dx representa a distância percorrida pela partícula.

O potencial de repulsão interatômica atua da seguinte forma: i) para pequenas distâncias entre o núcleo e o íon incidente, a forca repulsiva é do tipo Coulombiana; ii) para distancias maiores a nuvem de elétrons atua como uma blindagem entre os núcleos [7].

$$V(r) = \frac{1}{4\pi\varepsilon_0}\frac{Z_1 Z_2 e^2}{r}\varphi(r/a) \tag{2.2}$$

Considerando a formula acima, temos:

V é o potencial interatômico de repulsão (Coulombiano);
φ(r/a) é a função de blindagem;

Onde:
φ(r) → 1 quando r → 0;
Z1 e Z2 são as cargas dos núcleos;
r é a distancia entre núcleos;
a é o parâmetro de blindagem.

Compreendendo um pouco melhor o que acontece na interação entre partículas e matéria, convém considerar o modelo de Rutherford, que considera que todas as cargas positivas de um átomo estão no núcleo. Sendo assim, uma partícula que passe muito próximo a este núcleo poderá ser espalhada, devido à forte repulsão coulombiana (figura 2.4).

O desenvolvimento da teoria sobre a perda de velocidade de um íon energético num sólido tem sido difícil devido à complexidade para descrever a interação de ambos. Uma vez dentro do sólido, o íon pode perder elétrons e sua carga se torna uma função do alvo onde foi disparado e, nesse percurso, o íon pode perder e receber elétrons (mudando sua polarização) e modificando a estrutura do alvo conforme a velocidade do íon que incidiu no alvo.

Esses estudos levaram adiante a discussão sobre o modelo atômico e seu funcionamento.

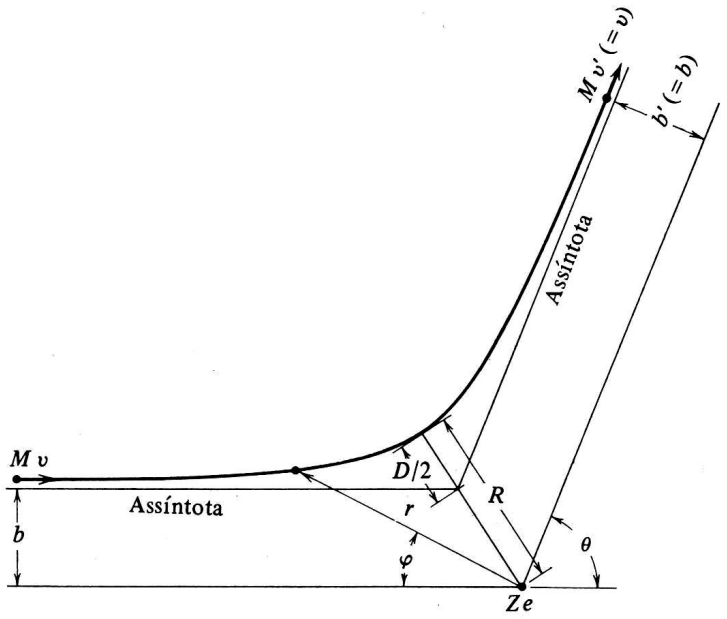


Figura 2.4: A trajetória hiperbólica de Rutherford [20], mostrando as coordenadas polares r, φ e os parâmetros b, D. Estes dois parâmetros determinam completamente a trajetória, em particular o ângulo de espalhamento Θ e a distância de maior aproximação R. A carga nuclear pontual Z_e está sobre um foco do ramo da hipérbole.

Uma das conclusões de Bohr era que a perda de energia dos íons através da matéria poderia ser dividida em duas componentes: freamento nuclear (perda de energia para o meio em função do núcleo positivo) e freamento eletrônico (perda de elétrons para o meio).

Os avanços seguintes ocorreram entre 1930 e 1935 quando H.A.Bethe e F.Bloch reiniciaram os estudos sob a perspectiva da mecânica quântica, baseado no trabalho de Bohr, e a avaliação de perda de energia para partículas leves com velocidade de 10 MeV/u.m.a. até 2 GeV/u.m.a.. Nesta abordagem teórica há limite de velocidade porque para velocidades inferiores os íons não perderiam carga e acima dessa velocidade existem correções relativísticas.

A seguir, entre 1938 e 1941 a análise de fragmentos de fissão avaliava como tratar a interação parcial dos íons (equação 2.3).

$$Z_1^{*} = Z_1^{1/3}\, V / V_0 \qquad (2.3)$$

Onde Z_1 é o número atômico do íon e o Z_1^{*} é a carga efetiva na perda de energia para o alvo; V é a velocidade do íon e V_0 é a velocidade de Bohr (~$2x10^8$ cm/seg).

A precisão sobre a previsão realizada durante a simulação matemática do freamento de íons é um elemento primordial para melhorar técnicas como implantação de íons, retroespalhamento Rutherford com íons pesados, SIMS (Secondary Ion Mass Spectrometry), entre outras [21]. Para essas aplicações o freamento a baixas energias (onde E < 25 keV/u.m.a.; por exemplo) é fundamental e a sua descrição envolve detalhes da estrutura eletrônica do íon e do átomo. A complexidade entre a velocidade de recuo de um projétil e seu estado de carga introduz características especiais que não são observadas no regime de altas energias.

Um íon, ao penetrar um meio (material), gradualmente transfere ao meio sua energia cinética em sucessivas colisões com os átomos que compõe esse meio. O poder de freamento é calculado através da taxa de energia perdida pelo íon para o meio onde foi introduzido em função da distância percorrida dentro desse meio. Em outras palavras podemos expressar o poder de freamento como a quantidade de modificações dependente dos números atômicos do íon (Z_1) e do átomo (Z_2) em função da velocidade do íon (v). De acordo com a sugestão dada por Bohr [22], o freamento de íons na matéria pode ser dividido em duas componentes:

1) Freamento Eletrônico: é a transferência de energia por meio de colisões inelásticas entre os íons e os elétrons dos átomos.
2) Freamento Nuclear : é a transferência de energia por meio de colisões elásticas entre os íons e os átomos.

A terminologia “freamento nuclear” é inoportuna, uma vez que a colisão elástica acontece entre o íon e o átomo como um todo, incluindo efeitos de blindagem eletrônica.

A direção do íon não é alterada significativamente em decorrência do freamento eletrônico, pois a massa do elétron é muito menor que a do átomo. Em contrapartida, o freamento nuclear é responsável pela produção de defeitos na estrutura de metais cristalinos através de grandes desvios na direção inicial do momento linear do íon.

O freamento eletrônico pode desencadear os seguintes processos secundários [23]:
a. ionização e/ou excitação eletrônica dos átomos.
b. ionização e/ou excitação eletrônica do íon.
c. captura eletrônica.
d. emissão de radiação eletromagnética.

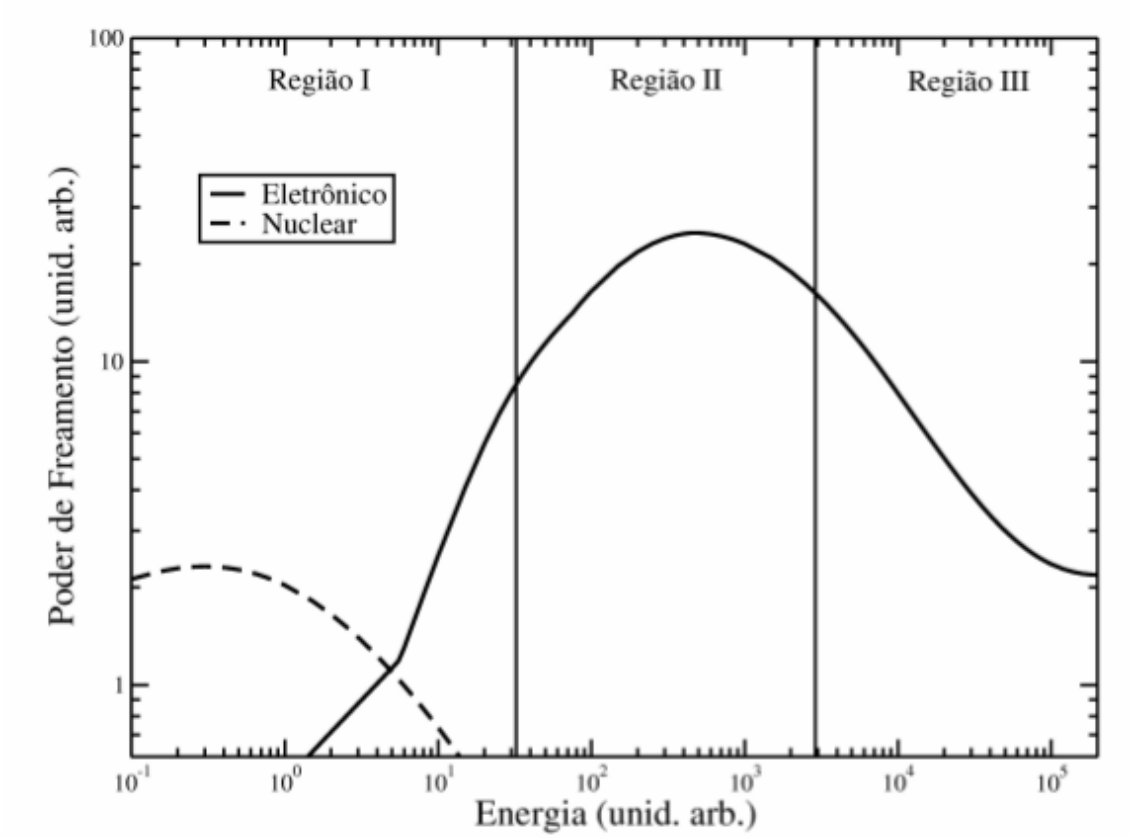


Figura 2.5: Curva característica do freamento de íons na matéria em função da energia do projétil. A súbita mudança no freamento eletrônico próximo à região de intersecção com a curva do freamento nuclear é decorrente do modelo utilizado para criar a curva.

A curva do poder de freamento em função da energia do projétil (figura 2.5) é dividida em 3 regiões diferentes. Região I de baixas energias, região II de energias intermediárias e região III de altas energias. A curva é separada em 3 regimes distintos de energias, indicadas aproximadamente. A partir do gráfico, podemos observar que o freamento eletrônico é o freamento dominante numa ampla região em energia. Isto acontece em função da perda de energia que uma partícula sofre ao adentrar outro material, até atingir o equilíbrio termodinâmico. A região de altas energias do íon (E > alguns MeV/u.m.a. para íons pesados) está representada na região III. A dinâmica de excitação/ionização das colisões com os átomos do meio nesta região é definida por uma única constante, relacionada com a energia de ionização média (equação 2.5)

dos átomos do meio e podemos normalizar as curvas do poder de freamento de diversos íons para o mesmo meio, como demonstrado na figura 2.6. A razão do poder de freamento pelo número atômico do íon ao quadrado ($(dE/dx)/Z_1^2$) é praticamente independente do íon para energias acima de 5 MeV/u.m.a.. A descrição desta relação para vários íons é a base de uma das hipóteses do modelo semi-empírico de Northcliffe e Schilling [12].

O íon progressivamente passa a capturar elétrons do meio no sentido de se neutralizar eletricamente na proporção que a velocidade do íon no material diminui. O processo de captura eletrônica ocorre principalmente ao longo das regiões I e II (conforme figura 2.5), sendo a segunda caracterizada pela presença de um máximo na curva de freamento, chamado de "pico de Bragg". O Projétil assume uma carga efetiva dependente da sua velocidade instantânea durante a captura eletrônica. A região I corresponde ao regime de baixas energias, onde $v \leq v_0 Z_1^{2/3}$, sendo v_0 a velocidade de Bohr. Segundo alguns modelos teóricos [24,25], o freamento é proporcional à velocidade do íon nesta região. A energia do pico de Bragg aumenta para Z_1 maiores, como se pode constatar na figura 2.6. O processo de freamento depende dos detalhes das estruturas eletrônicas do íon e do átomo nas energias abaixo do pico de Bragg, e as curvas do freamento de íons diferentes apresentam comportamentos diferentes. A figura 2.6 ilustra as diferenças de comportamento do freamento em baixas energias em comparação com o freamento em altas energias a despeito do modelo semi-empírico usado na figura 2.6 ser pouco preciso nessas energias. Portanto, a região de baixas velocidades é a mais difícil de descrever teoricamente.

O processo dominante é o freamento nuclear se $E < 25$ keV/u.m.a. (figura 2.5), quando passa a ser superior ao freamento eletrônico. O estado de ionização do íon é praticamente nulo nessas energias. O freamento nuclear começa a aumentar rapidamente chegando a um máximo e então decaindo a zero, enquanto a componente eletrônica diminui com a velocidade do projétil.

Dispersões na direção inicial do íon e na energia são geradas a partir do efeito sucessivo e da natureza estatística das colisões íon-átomo. Efeitos dessa natureza são chamados de "*straggling* de energia" (dispersão na energia) e "*straggling* angular" (dispersão da direção).

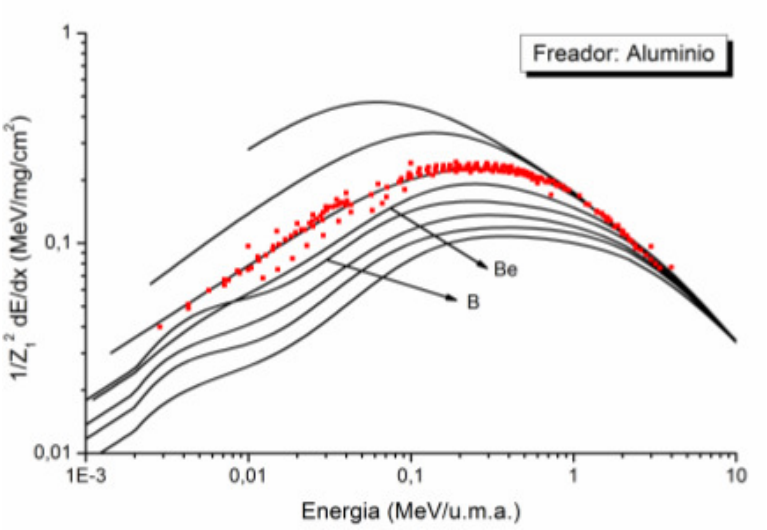


Figura 2.6: Poder de freamento de diversos íons em Alumínio. Há convergência para altas energias. Previsões extraídas do código TRIM/SRIM2008.03 [1]. Também inclusos estão os dados experimentais do freamento de Lítio em Alumínio [26], para ilustração da validade do código TRIM/SRIM2008.03 na previsão dos freamentos nessas energias.

Nos materiais cristalinos, o número de colisões pode ser reduzido de forma relevante se o projétil seguir uma trajetória entre os planos cristalinos do material. Esse tipo de efeito é denominado canalização e reflete um aumento do alcance do íon no material (figura 2.7). A perda de energia nas condições de canalização é um aspecto notável para a compreensão do freamento, pois apenas uma região de parâmetros de impacto é analisada nesse caso.

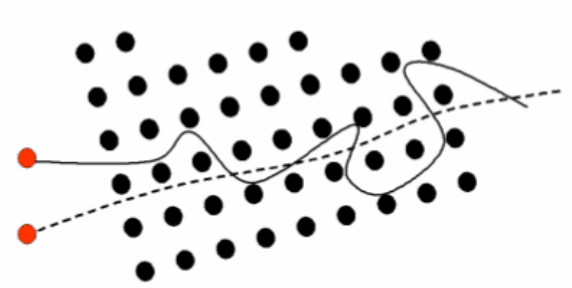


Figura 2.7: Ilustração do efeito de canalização (*channeling*; quando os íons entram no material através do plano cristalino; representado pela linha tracejada) em materiais com estrutura cristalina. Acima, a trajetória típica numa direção aleatória do material. O efeito sucessivo das colisões produz um straggling na direção do momento do íon nesse último caso.

A tabela 2.1 abaixo resume os principais momentos históricos abordados neste capítulo, que envolvem os princípios de fisica e química envolvidos no processo estudado.

Tabela 2.1 – Resumo com os momentos históricos da química e física utilizados neste trabalho

Época	**Acontecimento**
400 AC	Início do estudo de química
1661	Princípios da química em "O químico cético"
1774	Lavoisier postula lei da conservação de massas
1803	John Dalton elaborou a primeira tabela de pesos atômicos
1895	Descobertas partículas radioativas
1897	Thompson - átomos com cargas positivas e negativas
1899 - 1920	Thomson e Bohr - a velocidade da partícula é mais importante que a energia
1911	Rutherford identifica núcleo atômico
1913	Bohr - modelo de Rutherford considerando a teoria quântica
1930 – 1935	Bethe e Bloch - estudos de Thomson e Bohr sob a perspectiva da mecânica quântica
1938 – 1941	Análise de fragmentos da fissão – como tratar a interação parcial de íons
1947 – 1960	Freamento de partículas em gás livre de elétrons (Particle stopping in a free electron gas)

2.2 - Teorias de Freamento

As teorias de Bohr [8] e Bethe [9] que analisam individualmente a colisão do íon com os elétrons do átomo e formam a base teórica para os diversos modelos existentes. A colisão do ponto de vista coletivo é postulado pela teoria de Lindhard, Scharff e Schiott (LSS) [10,11] que tem como ponto de partida a interação de uma partícula com gás de elétrons livres; esta teoria também aborda o comportamento do freamento para baixas velocidades.

Nos últimos anos vários modelos teóricos vem sendo desenvolvidos dos quais se destacam: a) aproximação da convolução unitária (UCA, *unitary convolution aproximation*) [13,27], baseado na formulação da teoria de Block dependente do parâmetro de impacto. b)

Teoria binária (BT, *Binary Teory*) [14,28], semelhante a formulação do modelo de Bohr. A tabela 2.2 traz um resumo sobre as teorias de freamento.

Tabela 2.2 - Resumo sobre as teorias de freamento

Época	**Teoria**
Início Séc. XX	teoria para o freamento eletrônico
1930	Bethe - formulação quântica aplicada a freamento eletrônico
1963	Fano - "correção de camadas" devido ao movimento relativo dos elétrons atômicos
1950 - 1960	Teoria LSS
1970 - 1979	Modelo de Northcliffe e Schilling
1980 - 1989	Modelo de Ziegler, Biersack e Littmark (ZBL)
1999	Teoria da aproximação de convolução unitária, por Schiwietz e Grande
2000 - 2005	Teoria Binária, por Schinner e Sigmund

2.2.1 - Teorias de Bohr, Bethe e Block

A primeira teoria para o freamento eletrônico calculando o poder de freamento da matéria por meio da mecânica clássica não relativística foi proposta no início do século 20. Neste modelo, o íon incidente possui as características atômicas representadas na tabela periódica e carga pontual onde interagirá com os elétrons do meio. Tal relação pode ser observada na figura 2.8:

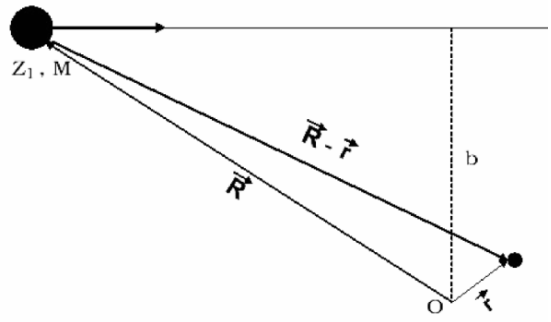


Figura 2.8: Íon com carga Z_1 e masssa M_1 >> me se desloca com velocidade v e parâmetro de impacto b.

A quantidade de energia absorvida pelo elétron secundário ao campo eletromagnético do íon em movimento, equivale à perda de energia do íon. Interações próximas e distantes serão definidas conforme o parâmetro de impacto b_0, sendo que o elétron é tratado como livre em colisões próximas ($b < b_0$) e em colisões distantes é levado em conta a energia de ligação do elétron ($b > b_0$).

A teoria de Bohr é aplicada quando a velocidade do íon incidente for muito maior que a velocidade orbital do elétron. Neste caso a velocidade do elétron é considerada nula durante o tempo de interação. A expressão final de Bohr deriva das equações [29] estudadas e pode ser observada na equação 2.4, onde ω_0 é frequência de oscilação característica do elétron atômico.

$$\frac{dE}{dx}_{Bohr} = \frac{4\pi N Z_2 Z_1^2 e^4}{m_e v^2} ln\left(\frac{1,123 m_e v^3}{Z_1 e^2 w_0}\right) \tag{2.4}$$

Em 1930, Bethe foi pioneiro na formulação quântica aplicada a freamento eletrônico [16] considerando a velocidade do projétil alta o suficiente em relação à velocidade orbital do elétron, contudo não relativística. Na teoria de Bethe, a colisão é classificada a partir da quantidade de movimento transferido durante a colisão e relacionada mais diretamente com a energia transferida aos elétrons. Assim, a distância das colisões é inversamente proporcional à transferência de movimento, conforme publicado [29.30].

Na equação 2.5 temos a expressão do poder de freamento derivado por Bethe, onde <I> representa a energia de ionização média por elétron, sendo $<I> = hw_0$

$$\frac{dE}{dx}_{Bethe} = \frac{4\pi N Z_2 Z_1^2 e^4}{m_e v^2} ln\left(\frac{2 m_e v^2}{\langle I \rangle}\right) \tag{2.5}$$

Os argumentos dos logaritmos são discretamente diferentes nos resultados de Bohr e Bethe. Estas teorias se assemelham na competição entre as contribuições próximas e distantes ao freamento. A teoria de Block faz a conexão entre os resultados das postulações clássicas e quânticas. Block considerou em seus cálculos a perturbação na função da onda dos elétrons

atômicos decorrente da presença do íon e ignorou o efeito de ligação dos elétrons em colisões próximas. O resultado obtido para íons não relativísticos é dado pela equação 2.6, de Bethe-Block, onde ψ é a derivação logarítmica da função gama Reψ, que é a parte real de ψ.

$$\frac{dE}{dx}_{Bloch} = \frac{4\pi N Z_2 Z_1^2 e^4}{m_e v^2}\left[ln\left(\frac{2m_e v^2}{\langle I\rangle}\right) + \psi\{1\} - Re\psi\left\{1 + i\left(\frac{Z_1 c}{137v}\right)\right\}\right] \quad (2.6)$$

Este resultado se reduz ao limite clássico dado por Bohr para $Z_1c/137v >> 1$ e ao limite quântico dado por Bethe para $Z_1c/137v << 1$ [12]. Da semelhança das equações 2.4 a 2.6 é observado que o poder de freamento é o produto de dois fatores: a) o de caráter coloumbiano, que diminui suavemente em função da velocidade do íon; b) um termo logarítmico associado à estrutura do material freador (alvo), que aumenta suavemente em função da velocidade.

A partir da expressão de Bethe-Block surgiram vários estudos adicionando correções à equação. Nesse sentido, Fano considera em sua teoria o momento transferido a um elétron ligado a partir da análise em três regiões distintas da energia transferida [31]. Fano introduziu dois termos a expressão de Bethe-Block resultando numa expressão relativística.

$$\frac{dE}{dx}_{Fano} = \frac{4\pi N Z_2 Z_1^2 e^4}{m_e v^2}\left[ln\left(\frac{2m_e v^2}{\langle I\rangle}\right) - \frac{C(v)}{Z_2} + \frac{v^2}{c^2} - ln\left(1 - \frac{v^2}{c^2}\right) - \frac{\delta}{2}\right] \quad (2.7)$$

Na equação 2.7, entre colchetes, o primeiro termo é a expressão de Bethe; o segundo termo é a "correção de camadas", que simula o efeito devido ao movimento relativo dos elétrons atômicos; o terceiro termo é uma correção relacionada à densidade do meio, onde os efeitos da polarização aos átomos reduzem o poder de freamento. Os demais termos são correções relativísticas.

Posteriormente, surgiram trabalhos implementando correções a expressão de Fano. Para implementar correções adicionais, foi feita uma expansão do "*stop number*" B (dado pelo termo entre colchetes da expressão 2.7) em potenciais de Z_1. A expansão do "stop number" na expressão de Bethe-Block é:

$$B = \left[L_0(\beta) + Z_1 L_1(\beta) + Z_1^2 L_2(\beta) + ...\right] \tag{2.8}$$

A equação 2.8 possui L_0, que contém todos os fatores de correção da expressão 2.4 de Fano. O segundo termo da expressão, L_1, é chamado de correção de Barkas ou correção Z_1^3. Esta expressão contém um termo de potência ímpar, assim, ela é sensível ao sinal da carga (positivo ou negativo) e é responsável pelo reduzido poder de freamento para o antipróton em relação ao freamento do próton no mesmo meio. O termo L_2 corresponde a uma pequena correção na força de ligação atômica do elétron, é chamado de correção de Block. O termo detalhado no artigo de revisão de Ziegler [32].

2.2.2 - Teoria de LSS

A teoria LSS (Lindhard, Scharff e Schiott) foi desenvolvida nas décadas de 1950 e 1960 como uma teoria unificada para freamento nuclear e eletrônico [10,25,33], fundamentada na análise do freamento de partículas por um gás de elétrons livres, sendo que a partícula incidente é tratada como uma perturbação no estado do gás de elétrons livres [11,34]. Assim, Lindhard e seus colaboradores calcularam o freamento de um íon em um gás de elétrons livres. O poder de freamento, considerando um gás de densidade uniforme (figura 2.9) é:

$$\frac{dE}{dx} = \frac{Z_1^{*2} e^4}{m_e v^2} \rho_0 L(\rho_0, v) \tag{2.9}$$

Com $L(\rho_0, \upsilon)$, função de interação, é postulado:

$$L(\rho_0, v) = \frac{i}{\pi \omega_0^2} \int_0^\infty \frac{dk}{k} \int_{-kv}^{kv} \omega d\omega \left[\frac{1}{\epsilon^l(k,\omega)} - 1\right] \tag{2.10}$$

A equação 2.10: Sendo a frequência do plasma $\omega_0^2 = 4\pi e2\ \rho_0/m_e$; e a constante dielétrica longitudinal do gás eletrônico, $\epsilon^l(\kappa, \omega)$.

Por meio da aproximação de densidade local, o modelo do gás de elétrons pode ser aplicado ao freamento de partículas na matéria. Em suma, a aproximação da densidade local supõe um plasma independente com densidade igual a densidade eletrônica local em cada elemento do volume sólido. O freamento eletrônico é calculado a partir da soma da contribuição sobre todo o volume do átomo do meio freador (equação 2.11).

$$\frac{dE}{dx} = \frac{4\pi N Z_1^{*2} Z_2 e^4}{m_e v^2} \int_0^\infty \rho(r) L(\rho, v)\, 4\pi r^2 dr \tag{2.11}$$

Através do modelo de Thomas-Fermi para o átomo, Lindhard *et al* estimaram a densidade eletrônica do átomo [30]. A expressão anterior se reduz ao termo para freamento de altas velocidades da expressão de Bethe, incluindo as correções de camadas; posteriormente, Lindhart [25] propôs uma expressão para freamento em baixas velocidades e em sua versão final são usadas unidades reduzidas para a energia (equação 2.12) e distância (onde ϵ representa a energia reduzida de LSS; e ρ representa a distância reduzida de LSS; equação 2.13).

$$\epsilon = \frac{4\pi\epsilon_0 M_1 M_2 v^2 a}{2Z_1 Z_2 e^2 (M_1 + M_2)} \tag{2.12}$$

$$\rho = 4\pi a^2 \frac{A_1 A_2}{(A_1 + A_2)^2} Nx \tag{2.13}$$

A expressão para o freamento da teoria LSS em baixas energias torna-se extremamente simples nestas unidades (equação 2.14):

$$\frac{d\epsilon}{d\rho} = k_e \epsilon^{1/2} \tag{2.14}$$

Considerando o seguinte k_e (equação 2.15):

$$k_e = 0,0793 \frac{Z_1^{2/3} Z_2^{1/2} (A_1 + A_2)^{3/2}}{A_1^{3/2} A_2^{1/2} \left(Z_1^{2/3} + Z_2^{2/3}\right)^{3/4}} \tag{2.15}$$

A expressão da teoria de LSS pode ser reescrita nessas unidades MeV/mg/cm^3 (equação 2.16) como:

$$\frac{dE}{dx}\text{LSS} = \frac{73,9 Z_1^{7/6} Z_2 E^{1/2}}{A_1^{1/2} A_2 \left(Z_1^{2/3} + Z_2^{2/3}\right)^{3/2}} \tag{2.16}$$

A equação 2.14 deixa de ser válida para V >$Z_1^{2/3}v_0$. A dependência teórica do freamento da velocidade de recuo do íon torna-se clara nesta expressão.

2.2.3 - Teoria da aproximação de convolução unitária

Aproximação de convolução unitária (ACU) é uma teoria para descrição do freamento eletrônico em termos do parâmetro de impacto. Este modelo é útil para a situação de tunelamento do projétil. A característica ondulatória das partículas pode ser ignorada para íons pesados com energias maiores a 1 KeV e pode ser aplicada ao estudo do freamento de íons da matéria uma descrição em termos do parâmetro de impacto. Ainda, na região dos parâmetros de impacto relevantes para o freamento eletrônico, desvios na trajetória inicial destas partículas são insignificantes.

No cálculo da perda de energia provocada pela excitação e/ou ionização do átomo do meio devemos considerar as amplitudes de transição Af(b), entre o estado inicial |0>, com energia E_0 e os possíveis estados finais |f>, com energia E_f. A perda de energia ΔE é obtida pela somatória de todos os estados finais do átomo do meio.

$$\Delta E = \sum_f \left| a_f(\vec{b}) \right|^2 (E_f - E_0) \tag{2.17}$$

A realização desses cálculos, nos últimos anos, utiliza-se de métodos tradicionais da física atômica como cálculos baseados na aproximação de onda plana de Bohr (PWBA *Play Wave Bohr Aproximation*) (AOCC, *Atomic Orbital Coupled-Channel*) (CTMC, *Classical Trajectory Monte Carlo*) (CDW-EIS, *Continuous Distorted Wave*). Esses cálculos exigem grande esforço computacional porque o resultado acumula todos os cálculos efetuados anteriormente. Para aprimorar e agilizar o uso destas equações que consideram os parâmetros de impacto da colisão e obtêm resultados quantitativos satisfatórios, foi desenvolvido o modelo para ACU.

Para altos parâmetros de impacto (colisões distantes) é usada a aproximação de dipolo e a equação para o freamento é dada pela equação 2.18:

$$\frac{dE}{dx}(b) = \frac{2Z_1^2}{v^2 b^2} \sum_i f_i g\left[\frac{(E_f - E_i)\, b}{v}\right] \tag{2.18}$$

Equação 2.18: Sendo g[x] uma função envolvendo funções de Bessel modificadas e f_i as amplitudes de oscilação do dipolo.

É possível desconsiderar o movimento relativo do elétron para íons de alta velocidade e para os parâmetros de impacto mais próximos ao elétron. Nestes casos a equação de freamento é dada como apresentado a seguir (equação 2.19), onde T(b) é uma função que descreve a energia transferida na colisão e envolve funções de Bessel; r é um vetor perpendicular à direção de incidência do íon.

$$\frac{dE}{dx}(b) = \int d^2 r_\perp T\left(\overrightarrow{b} - \overrightarrow{r}_\perp\right) \int dz \rho\left(\overrightarrow{r}_\perp, \mathrm{Z}_1\right) \tag{2.19}$$

Foi proposta uma equação para descrever o freamento para todos os parâmetros de impacto utilizando uma aproximação da teoria de perturbação representada por:

$$\frac{dE}{dx}(b) = \int d^2 r_\perp \mathcal{F}\left(\overrightarrow{b} - \overrightarrow{r}_\perp\right) \int dz \rho\left(\overrightarrow{r}_\perp, z\right) \tag{2.20}$$

Sendo $\mathcal{F}\left(\overrightarrow{b}\right)$ dado pela equação 2.21

$$\mathcal{F}\left(\overrightarrow{b}\right) = \frac{2Z_1^2 e^4}{\left(4\pi\epsilon_0\right)^2 m_e v^2 b^2} \times h\left(2m_e vb/\hbar\right) \times \sum_i f_i g\left[\frac{\left(E_f - E_i\right) b}{v}\right] \tag{2.21}$$

As amplitudes de oscilações dos elétrons e a densidade eletrônica são os parâmetros principais. As previsões da teoria de Bethe-Block para íons leves e energéticos são compatíveis com os valores obtidos pelo tratamento perturbativo (cuja teoria é compreendida por métodos matemáticos utilizados para encontrar uma solução aproximada do problema que não pode ser resolvido com exatidão).

A ACU é uma extensão da aproximação perturbativa baseada no modelo de Block para colisões próximas, sendo incorporado um tratamento não perturbativo. Dentre os inconvenientes do tratamento perturbativo, o principal é a introdução de uma possibilidade de ionização em que, em alguns casos, pode exceder 100% (criação de elétrons). Para evitar esse efeito, como para íons pesados, é usado o tratamento não perturbativo da teoria de Block que normatiza as possibilidades.

O modelo ACU [13], obtém expressão semelhante à expressão 2.21 exceto pela inclusão de um termo de escala na função vetor *h*:

$$\mathcal{F}\left(\overrightarrow{b}\right) = \frac{2Z_1^2 e^4}{\left(4\pi\epsilon_0\right)^2 v^2 b^2} \times h\left(\frac{2vb}{\eta}\right) \times \sum_i f_i g\left[\frac{\left(E_f - E_i\right) b}{v}\right] \tag{2.22}$$

Sendo $\eta = \exp\left[\mathrm{Re}\psi\left(1 + i\gamma\right) - \psi\left(1\right)\right]$ e $\gamma = Z_1 e^2 / \left(4\pi\epsilon_0 \hbar v\right)$

Os dois primeiros termos da equação acima consideram a colisão com parâmetros de impacto baixo e, para n=1, corresponde a energia transferida obtida em primeira aproximação perturbativa.

O efeito de blindagem do projétil é considerado por meio de um parâmetro de impacto dependente da carga efetiva do projétil tanto para colisões distantes como próximas. Neste modelo, os elétrons do projétil não são tidos como excitações eletrônicas deste.

2.2.4 - Teoria Binária

A.Schinner e P.Sigmund postularam a teoria binária (TB) para o freamento eletrônico com fundamentos teóricos muito próximos a teoria clássica de Bohr. As teorias clássicas para a abordagem diferenciada para colisões próximas e distantes enfrentam dificuldades para incorporação do efeito de Barkas e, em menor proporção, a incorporação de correções de camadas. Essas situações mostram a necessidade um modelo clássico alternativo não perturbativo.

A teoria binária trata uma abordagem clássica não perturbativa onde a teoria de Bohr, que utiliza o potencial Coulumbiano na representação da interação íon-elétron. Há divergência em relação a forma de evitar o uso de métodos perturbativos e ausência da distinção explícita entre colisões distantes e próximas. Esta teoria ainda considera a blindagem eletrônica do íon e o movimento intrínseco dos elétrons do alvo. Na TB a interação íon-elétron é representada por um potencial do tipo.

$$V_{eff}(r) = -\frac{Z_1 e^2}{(4\pi\epsilon_0)\, r} \exp\left(-\frac{r}{a_{ad}}\right), \qquad \text{com} \quad a_{ad} = \frac{v}{w_0} \tag{2.23}$$

A energia transferida, T(b,v) pode ser determinada a partir do potencia de interação numa colisão binária através de uma equação de movimento. Devemos incluir também a energia que é transferida na forma potencial ao elétron. Isto pode ser simulado pela adição de um termo harmônico na expressão para T(b,v) [14].

A equação que calcula a seção de choque de freamento para um elétron na camada l é:

$$S_l = \int_0^\infty T_l(b,v) 2\pi b db \tag{2.24}$$

E a seção de choque de freamento é:

$$S = Z_2 \sum_l f_l S_l \tag{2.25}$$

Onde f_l são as amplitudes de oscilação dipolar para o elétron na camada l.

Assim, o poder de freamento é:

$$\frac{dE}{dx} = NS = NZ_2 \sum_l f_l S_l \tag{2.26}$$

A partir das integrais e equações (2.24 – 2.26) é possível calcular numericamente os freamentos. O resultado numérico da equação 2.23 equivale aos da teoria de Bohr conforme demonstrado [14]. Dessa forma o modelo não sobre intervenções de métodos perturbativos e efeitos de ordens superiores em Z_1 estão implícitos no modelo.

2.2.5 - Modelos Semi-Empíricos para freamento eletrônico

Modelos semi-empíricos são desenvolvidos a partir do comportamento esperado para o freamento e de dados experimentais existentes. Nos anos 70 o modelo de Northcliffe e Schilling [35] foi amplamente usado. Nos anos 80 o Ziegler, Biersack e Littmark [36] desenvolveram um método aplicável ao freamento de vários íons em meios sólidos e gasosos.

2.2.5.1 - Tabelas de Northcliffe e Schilling

Em 1970, Northcliffe e Schilling criaram um método semi-empírico para o freamento de vários íons em 24 meios diferente em sólidos e gasosos, além de registrar energias de recuo na região $0{,}0125 \leq E/A_1 \leq 12MeV/u.m.a.$.

A principal hipótese deste modelo parte do princípio que o poder de freamento relativo entre dois materiais em uma determinada velocidade independe do íon aplicado [12], ou seja, serão equivalentes. Tal modelo foi validado através de um conjunto de experimentos que produziram um rol de curvas considerando o alumínio como material freador de vários íons [37] e cuja validação foi obtida através da interpolação desses resultados [35].

2.2.5.2 - Ziegler, Biersack e Littmark (ZBL)

Em 1980, ZBL criaram um modelo semi-empírico para freamento de íons com $Z_1 \leq 92$ em meio sólido ou gasoso, numa vasta região de energia do projétil (1KeV/u.m.a. a 2GeV/u.m.a.) [36].

Para o freamento de prótons foram criadas curvas empíricas de ajustes dos dados experimentais disponíveis que foram aplicados a meios e/ou energias para os quais até então não haviam medidas experimentais. O freamento dos íons de hélio foi baseado nas curvas do freamento de prótons aplicando uma energia conveniente. A linearidade do freamento em virtude da velocidade de recuo do projétil é considerada para baixas energias, o que contempla o modelo de LSS para o freamento eletrônico.

Para freamento de baixa velocidade como LSS sugerem uma dependência linear entre o freamento e a velocidade, o que foi confirmado por diversos experimentos entre íon e meio. Exceções a esta regra são meios semicondutores como silício (Si) e germânio (Ge) onde o freamento é proporcional a $V^{0,7}$. O modelo ZBL considera a dependência linear para todas as combinações possíveis exceto aquelas que envolvem íons com Z1 <=19 e meios de carbono (C), silício (Si) e germânio (Ge), nos quais é aplicada a dependência $V^{0,75}$.

A região de energias intermediárias ($25 < E < 200$ KeV/uma) é importante para elaboração de curvas de freamento que sejam suaves ao longo de toda região de energias. Os

autores fizeram uma análise com base na teoria de Brandt e Kitagawa (BK) [38], cuja hipótese diz respeito ao estado de carga do íon. A teoria de BK supõe que os elétrons removidos do íon são aqueles cujas velocidades orbitais são menores que a velocidade do íon relativo aos elétrons do meio. A partir do modelo Thomas-Fermi a velocidade relativa do íon (v_r) é dada pela equação 2.27:

$$\begin{aligned} v_r &= v\left(1+0,2v_{TF}^2/v^2\right) && \text{para} \quad v > v_{TF} \\ v_r &= 0,75v_{TF}\left(1+2v^2/3v_{TF}^2 - v^4/15v_{TF}^4\right) && \text{para} \quad v < v_{TF} \end{aligned} \tag{2.27}$$

O grau de ionização (q) do íon incidente em função da sua velocidade relativa foi apresentado por ZBL (equação 2.28):

$$q = 1 - exp\left(0,803v_r^{*0,3} - 1,317v_r^{*0,6} - 0,382v_r^* - 0,009v_r^{*2}\right) \tag{2.28}$$

Onde $v_r^* = v_r/v_0 Z_1^{2/3}$

Após determinar o estado de carga do íon a teoria de BK estabelece a distribuição eletrônica do íon em função do grau de ionização. Essa distribuição se relaciona com o comprimento de blindagem do projétil expresso em função do número atômico do íon e seu grau de ionização.

$$\Lambda = \frac{2a_0\left(1-q\right)^{2/3}}{Z_1^{1/3}\left(1-\frac{1-q}{7}\right)} \tag{2.29}$$

Equação 2.29: Onde a_0 é o raio de Bohr

Ainda, a teoria de BK propõe uma expressão simples para a carga efetiva do íon na matéria em colisões próximas e distantes.

$$\gamma = q + C(1-q)ln\left[1 + (2\Lambda v_F/a_0 v)^2\right] \tag{2.30}$$

Equação 2.30: Onde C é uma quantidade próxima a ½

Ziegler e seus colaboradores, a partir dos dados experimentais, constataram que o melhor ajuste obtido correspondia a $C \approx (v/v_{TF})^2/2$, a partir da carga efetiva.

Desde então ZBL disponibiliza atualizações dos dados para seu programa TRIM/SRIM [1].

2.2.6 - Freamento Nuclear

O modelo de freamento nuclear é baseado na colisão binária entre íon e átomo considerando: a) a independência entre os componente eletrônico e nuclear do freamento; b) a colisão elástica regida por um potencial esfericamente simétrico.

A quantidade de energia transferida numa colisão íon-átomo é maior que na colisão íon-elétron para colisões próximas, o que explica o desacoplamento dos componentes nuclear e eletrônico; e implica numa separação aproximada entre os freamentos nuclear e eletrônico em função do parâmetro de impacto. A teoria clássica do espalhamento binário é baseada na conservação do momento angular durante a colisão. Por isso é postulada uma força central.

Em linhas gerais, os potenciais interatômicos são descritos como um termo Coulombiano multiplicado por uma função de blindagem.

$$V(r) = \frac{Z_1 Z_2 e^2}{4\pi\epsilon_0 r}\Psi(r) \tag{2.31}$$

Equação 2.31: Sendo r a distância entre os núcleos interagentes.

A função de blindagem considera a distribuição de carga do íon e do átomo. Modelos atômicos como os de Thomas-Fermi, Lenz-Jensen e Moliere [36] estimam a distribuição de

cargas eletrônicas. O método de Hartree-Fock [36] inclui estruturas de camadas dos átomos para estas distribuições.

2.2.6.1 - LSS

O estudo de LSS descreve as colisões íon-átomo de forma semelhante ao freamento de diversos pares de íons-átomos. Na tentativa de obter uma equação com volume reduzido de variáveis interdependentes, considerou-se o tratamento perturbativo cujo de ângulo de espalhamento do íon é função de um único parâmetro, chamado parâmetro t, que descreve qualquer colisão íon-átomo (equação 2.32). Sendo a função $f(t^{1/2})$ determinada através de métodos numéricos a partir do potencial interatômico aplicado na colisão.

$$t^{1/2} = \epsilon sen\left(\frac{\theta_{cm}}{2}\right) \tag{2.32}$$

A seção de choque diferencial do freamento nuclear é dada por (2.33):

$$d\sigma = \frac{\pi a_I^2 f(t^{1/2})}{2t^{3/2}} dt \tag{2.33}$$

Computacionalmente o potencial de Thomas-Fermi correspondente pode ser utilizado em função de uma função analítica correspondente [39]. A seção de choque diferencial relaciona-se com a seção de choque de freamento nuclear Sn por uma integral em $d\sigma$ ponderado pela energia transferida na colisão, T:

$$S_n = \int T d\sigma \tag{2.34}$$

Lindhard introduziu a seção de choque do freamento nuclear S_n (ε), onde as energias e distancias são descritas em unidades reduzidas, assim, S_n é:

$$s_n(\epsilon) = \frac{1}{\epsilon} \int \frac{dt}{2t^{\frac{1}{2}}} f(t^{\frac{1}{2}}) \tag{2.35}$$

Expressões analíticas do poder de freamento nuclear nas unidades reduzidas de LSS estão em [39] e são representadas pela equação 2.36:

$$\frac{d\epsilon}{d\rho} = N s_n(\epsilon) = \begin{cases} 0,611 e^{\left(-\epsilon^{-1/2}/1,919\right)} \left[1 - e^{\left(-\epsilon^{-1/2}/0,2406\right)}\right] & \epsilon \leq 2,4 \\ 0,5\epsilon \left[0,3 + ln\left(\frac{0,6+\epsilon^2}{\epsilon}\right)\right] & \epsilon > 2,4 \end{cases} \tag{2.36}$$

2.2.6.2 - ZBL

ZBL [36] formularam um potencial interatômico a partir de uma sistematização do cálculo numérico do potencial interatômico para várias combinações íon-atomo. O potencial de Thomas-Fermi, foi utilizado nos cálculos e a informação da distribuição de carga do íon e do átomo obtido com base no método Hartree-Fock.

O parâmetro de blindagem *a*, ajusta próximas umas das outras as diferentes funções de blindagem calculadas. ZBL desenvolveram empiricamente um parâmetro de blindagem a_u com efeito mais eficiente e dispersão em torno de 5% para os diferentes potenciais interatômicos calculados. O fator da escala a_u chamado de comprimento de blindagem universal é:

$$a_u = \frac{0,8854 a_0}{(Z_1^{0,23} + Z_2^{0,23})} \tag{2.37}$$

Através da expressão acima, os potenciais interatômicos calculados numericamente foram ajustados por uma única função de blindagem média chamada potencial universal:

$$\Phi_u(x) = 0,1818 e^{-3,20x} + 0,5099 e^{-0,94x} + 0,2802 e^{-0,40x} + 0,0282 e^{-0,20x} \tag{2.38}$$

Equação 2.38: Sendo x = r / a_u

A determinação da perda de energia é calculada a partir do ângulo de espalhamento e da energia da partícula espalhada, baseada nas distribuições atômicas calculadas a partir do método Hartree-Fock, e ajustados por um único potencial, uma vez determinada o potencial universal. Do mesmo modo que o desenvolvido na teoria LSS deriva-se a função $f(t^{1/2})$ para o potencial universal de ZBL.

A partir do potencial universal, com auxilio da unidades reduzidas de LSS, a seção de choque reduzida do freamento nuclear de ZBL é:

$$s_n(\epsilon) = \begin{cases} \frac{ln(1+1{,}1383\epsilon)}{2(\epsilon+0{,}013\epsilon^{0{,}212}+0{,}196\epsilon^{0{,}5})} & para\ \epsilon \leq 30 \\ \frac{ln(\epsilon)}{2\epsilon} & para\ \epsilon > 30 \end{cases} \quad (2.39)$$

2.2.7 - Resumo de teorias e modelos

Apesar de todo o avanço obtido até o presente momento, ainda há muito a ser detalhado nos estudos que seguirão na área de física, tendo principalmente os modelos de Bohr e Bethe como base dos estudos. Graças ao avanço da computação será possível avaliar mais rapidamente os resultados calculados e será possível a comparação com os resultados experimentais.

Dessa forma, a tabela 2.3, a seguir, traz um resumo com as informações relevantes abordadas neste capítulo, onde foram tabuladas as teorias e modelos apresentados, respectivas energias e o efeito de oscilação, denominados em Z_1 e Z_2.

Tabela 2.3: Regiões de aplicabilidade dos modelos abordados. Para as teorias de Bohr e Bethe, os limites são aproximados. Em alguns casos, os limites não são bem estabelecidos, como para as teorias LSS e Binária.

Teoria/Modelo	Energia (MeV/u.m.a.)	Z_1	Z_2
Bohr, Bethe	≥ 0,5	p, He	≤ 17
LSS	Ampla Região	Todos	Todos
ACU	0,001 – 200	≥ 1	1 – 92
Binária	Ampla Região	Todos	Todos
NS	0,0125 – 12	1 – 103	24 meios
TRIM/SRIM	1,1 eV – 2 GeV	1 – 92	Todos

2.3 – Equipamento para nanoeletrônica

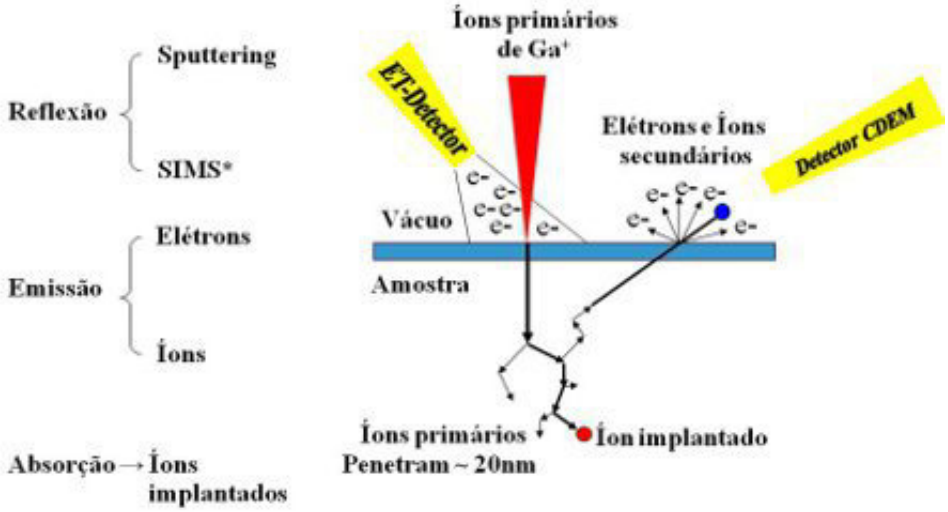
O Centro de Componentes Semicondutores (CCS) da Universidade Estadual de Campinas (UNICAMP) recebeu investimentos (FAPESP, FINEP, LQES entre outros) para viabilizar a instalação de um laboratório para estudo de nanotecnologias com suporte multiusuário. O equipamento adquirido e instalado no CCS é um Sistema de feixe duplo FIB/SEM (Focused Ion Beam/Scanning Electron Miscroscopy), modelo Nova 200 NanoLab, fabricado pela empresa FEI Co., da Holanda, que integra múltiplas funções num único equipamento.

Atualmente, uma das ferramentas mais versáteis para a fabricação e caracterização das novas gerações de nanodispositivos como MEMS e NEMS (Micro and Nano Electrical Mechanical Systems) são aquelas que compreendem as técnicas de feixes de íons focalizados (FIB). A técnica permite uma pulverização rápida e precisa de volumes restritos da ordem de alguns nm3, sem o uso de máscaras e de todas as etapas de processamento e técnicas periféricas que são necessárias na litografia convencional [40], permitindo gravar padrões na superfície diretamente, possibilitando uma prototipagem rápida de nanodispositivos, além disso, o sistema pode ser usado para a deposição de metais e dielétricos e para obter imagens de alta resolução das estruturas fabricadas através de microscopia eletrônica de varredura [41]. Em nossos experimentos foi utilizado o equipamento supra mencionado (figura 2.9).

Figura 2.9: Sistema de feixe duplo (FIB/SEM), modelo Nova 200 Nanolab.

O sistema FIB opera de modo semelhante ao SEM, fazendo a varredura da superfície por feixe de partículas. Esse modelo utiliza um feixe de íons de Ga com um diâmetro de feixe de cerca de 7 nm e a energia máxima do feixe 30 kV. O diâmetro do feixe de elétrons é aproximadamente 1 nm e a energia varia de 2 a 30 kV. A microscopia eletrônica é realizada com a emissão de um feixe de elétrons (FEG, *Field Emission Gun*) que permite a obtenção de imagens de alta resolução.

Durante a interação do feixe de íons com a superfície ocorre a produção de várias partículas secundárias (íons, átomos e elétrons; figura 2.10). Dessa forma, tanto os íons como os elétrons secundários podem ser coletados para a formação de imagens da superfície. Esse sistema pode operar em dois regimes distintos, dependendo da corrente do feixe de íons e do tempo da exposição: (i) aquisição de imagens que utiliza corrente baixa e tempos curtos e (ii) modificação da superfície que geralmente utiliza corrente bastante alta durante períodos maiores, permitindo pulverizar o substrato produzindo cortes, buracos e canais na superfície. O sistema também permite a deposição direta de metais e dielétricos como platina e óxido de silício, através de processos induzidos pelo feixe de gálio incidente na superfície em ambiente de gases especiais que são injetados diretamente sobre a superfície processada. Para deposição de metais, são utilizados gases organo-metálicos e para deposição de oxido de silício, TEOS (Tetra-Ethyl-Ortho-Silicate).



*Filtro de Massa – SIMS (Secondary Ion Mass Spectroscopy)

Figura 2.10: Interação do feixe de ions com a superfície das amostras

A pulverização da superfície ocorre através de um processo de sputtering da seguinte maneira, quando os íons incidentes altamente energéticos (5-30 kV) atingem o material alvo, produz-se uma cascata de colisão nesse material. Se um átomo da superfície recebe um impulso suficiente para ultrapassar a energia de ligação da superfície, o átomo de superfície é arrancado da mesma. Os fatores que afetam a taxa de sputtering incluem, o número atômico dos átomos do alvo, energia dos íons incidentes, o ângulo de incidência do feixe de íons, a densidade atômica do alvo, a energia de ligação da superfície do alvo e orientação cristalográfica do alvo.

Como foi citado, conforme o regime de corrente e tensão que aplicamos nos íons de gálio durante o processo de irradiação, vários fenômenos podem ocorrer simultaneamente como: pulverização, deposição e implantação, e em alguns casos esses processos podem alterar as propriedades intrínsecas dos materiais.

O equipamento vem calibrado para a corrosão de alguns materiais mais comuns para as aplicações técnológicas (Al, Au, Cu, Si, SiO2), possuindo uma biblioteca de modelos (templates) com os parâmetros de processamento. Na prática, a maior parte das amostras que utilizamos são nanoestruturadas compostas por multicamadas de materiais heterogêneos, ligas, nanoparticulados, nanocarbonos, entre outros. Dessa maneira, sempre que começamos o estudo e processamento de um novo material, faz-se necessário à calibração de novos parâmetros para lidar com esse material de forma que altere pouco ou pelo menos de forma controlável as propriedades desses materiais.

A simulação por computador, replicando dados e condições do laboratório, permite entender e estabelecer parâmetros (criar novos modelos). Dessa forma é possível racionalizar o consumo de recursos (utilização das fontes), evitando mudar as propriedades das amostras (que muitas vezes estão disponíveis em pouquíssima quantidade) e o desgaste prematuro do equipamento. Outro fator importante a considerar é que todo o material removido apenas para testes, contamina a câmara e prejudica muito a resolução nanométrica do FEG e encurtar a vida da coluna de íons do FIB.

Dessa forma, foram verificados os custos para manutenção do FIB/SEM em funcionamento, cujo custo/hora é composto pela manutenção preventiva, acrescida dos materiais utilizados como: abertura (peça que efetua o primeiro estreitamento no feixe de íons); e das fontes (a fonte de íons de gálio; e para deposição metais e óxido de silício), indicados na tabela 2.4.

Tabela 2.4: Relação de custos para funcionamento do FIB/SEM

	Custo R$	**Tempo de duração em horas**	**Custo/Hora em R$**
Fonte de elétrons	31.500,00	10.000	3,15
Fonte de íons de Gálio (Ga)	13.000,00	1.000	13,00
Abertura	11.000,00	1.000	11,00
Fonte de Platina (Pt)	12.000,00	500	24,00
Fonte de Oxido de Silício (SiO_2)	12.000,00	500	24,00
Mão de obra/Serviço técnico de manutenção FIB	52.500,00	120	437,50
Funcionário para utilização do equipamento	52.000,00	1.920	27,10

* Mão de obra para manutenção preventiva estimada em 15 dias por ano;

** Fonte de elétrons funciona 24 horas/dia ininterruptamente;

*** Fonte de abertura é trocada a cada troca de fonte de íons;

**** Uso regular é fonte de íons + elemento a ser depositado (Pt ou SiO_2)

**** Fonte de íons de gálio é Ga^+ LMIS (Liquid Metal Íon Source)

É possível observar que o custo/hora médio para uso do FIB/SEM é de aproximadamente R$ 495,00 (quatrocentos e noventa e cinco reais); e que o equipamento não pode ser desligado. Já um computador equipado para realizar simulações numéricas com CPU e GPU dedicadas, tem seu custo/hora (considerando funcionamento ininterrupto) em R$ 30,00 (considerando energia elétrica, equipamento e mão-de-obra em sala climatizada).

Dessa forma a simulação se mostra essencial e uma parcela muito importante no gerenciamento de alguns equipamentos, principalmente quando envolve complexidade e valores altos de manutenção. Além disso, os testes prévios no sistema Dual Beam para se estabelecer os parâmetros corretos de processamento empiricamente (corrente, tempo, estratégia de corte ou deposição, entre outros fatores) despendem bastante tempo.

Quando comparamos os valores de gastos na manutenção de equipamentos com os custos computacionais em termos das performances computacionais disponíveis, podemos vislumbrar quanto que é vantajoso fazer uma simulação. Assim, nesse trabalho foram realizadas várias

simulações em diferentes tipos de processadores e ambientes computacionais onde foram usados como modelo, alguns dos processos e materiais que normalmente usamos nas pesquisas do CCS.

A discussão dos resultados obtidos e publicações está no capítulo 4.

Capítulo 3

3 - EVOLUÇÃO TECNOLOGICA DOS SISTEMAS COMPUTACIONAIS

Neste capítulo será apresentado o desenvolvimento dos equipamentos computacionais até os dias atuais, abordando a evolução dos computadores, processadores e dispositivos gráficos. A lei de Moore, que prevê a duplicação da quantidade de transistores presentes nos circuitos integrados a cada dois anos está limitada à temperatura do dispositivo. Dessa forma investiu-se em alternativas para a crescente demanda de processamento através do aumento da quantidade de núcleos do processador (CPU, *Central Processing Unit*).

A disponibilidade de computadores oriundos de diferentes fabricantes com nomenclaturas e velocidades distintas exigiu uma avaliação adequada do desempenho dos processadores para comparação de quão veloz o equipamento era para solução de cálculos. Posteriormente o mesmo foi aplicado para comparação não apenas de CPUs, mas também de GPUs (GPU, *Graphic Processing Unit*).

Assim como os equipamentos para uso pessoal, os supercomputadores, máquinas desenvolvidas para executar operações específicas e cálculos de extrema complexidade, como aqueles realizados para previsão do tempo e em análises de física entre outras aplicações foram classificados conforme o poder de processamento e estão no TOP500 [42].

A evolução do modelo de programação dos equipamentos deu origem a novas linguagens e métodos mais eficientes para realização de operações numéricas que melhoraram o trabalho científico e reduziram o tempo necessário para execução dessas tarefas. Iniciou-se a padronização de equipamentos e linguagens permitindo a expansão dos computadores para uso em pesquisa, indústria e comércio. A necessidade de atender a diversas pessoas utilizando um único equipamento deu origem à novas técnicas de programação, assim como sistemas operacionais multiusuários, cujas sucessivas inovações levaram a criação do UNIX, que por sua vez fomentou o desenvolvimento da computação distribuída e processamento paralelo. A evolução seguinte foi o desenvolvimento de emuladores e máquinas virtuais que trouxeram possibilidade de expandir os estudos em computação distribuída de maneira estável e aproveitando ao máximo os recursos dos equipamentos.

A evolução dos terminais de vídeo como interface para interação com os equipamentos na década de 1970 trouxe ao mercado o processador gráfico (GPU, *Graphical Processing Unit*), cujo sucesso obtido com jogos e melhores interfaces aos usuários permitiu o passo seguinte, ou seja, processadores gráficos programáveis para uso geral (GPGPU, *General Purpose GPU*). Neste mercado a NVidia foi a pioneira a trazer alto desempenho a equipamentos de uso pessoal. Em 2006 surgiu a primeira arquitetura a suportar programação paralela (CUDA, *Compute Unified Device Architecture*) em placas de vídeo, onde além da representação dos dados gráficos, é possível o processamento de outras informações que demandam alto poder de processamento, como visão computacional, inteligência artificial, cálculo numérico e simulações.

Simulação é a representação de um contexto real através de modelos que imitam uma atividade real a fim de formular hipóteses e auxiliar na decisão final da resolução de um problema. Simulações em computador voltadas para a física e química tiveram início na década de 1960. Utilizando o método de aproximações de colisões binárias, temos programas como o TRIM/SRIM baseado no modelo ZBL que também utiliza Monte Carlo para simulação. Monte Carlo é um método estatístico utilizado em simulações estocásticas, cujo objetivo é explorar as possibilidades de um dado fenômeno em que o comportamento possa ser quantificado matematicamente. Neste capítulo, será descrito o processo de simulação utilizando animação (características, vantagens e limitações), assim como alguns aplicativos estudados.

3.1 – Evolução dos processadores

O processador (CPU, *Central Processing Unit*) é a unidade responsável por realizar as operações aritméticas e lógicas que os programas, desenvolvidos para cada arquitetura, utilizam. Quando mencionamos a expressão arquitetura neste texto, é importante lembrar que nos primórdios da computação, cada programa era específico para determinado equipamento, podendo até mesmo não ser compatível com outros equipamentos de um mesmo fabricante. A padronização dos equipamentos (hardware) e programas (software) foi o que permitiu a interoperabilidade de programas de computador e ampliação no desenvolvimento desta área.

Antes da idéia de CPU, a infraestrutura computacional precisava de adequação, ajustes e montagem do equipamento em si para realizar uma determinada tarefa. Em 1945, John Von

Neumann publicou a idéia de uma unidade central de processamento capaz de realizar diversas tarefas [43,44]. A partir daí o equipamento evoluiu com diversas finalidades, até a IBM desenvolver uma nova abordagem no início da década de 1960, com a padronização do conjunto básico de informações para computadores. O modelo de CPU atual, onde todos os circuitos estão integrados em um único chip de silício, foi possível apenas no início da década de 1970.

A partir da década de 1980, os computadores começaram a ser usados além da pesquisa e setor industrial, chegando ao comércio e residências. Chegada a primeira geração de computadores pessoais (PC, *Personal Computer*), o equipamento pôde então ter seu custo reduzido e a massificação de um padrão permitiu o desenvolvimento de novos programas.

Com relação aos programas, somente com o surgimento da linguagem C para programação de computadores, pudemos então padronizar, expandir e desenvolver programas portáveis para outras arquiteturas ou sistemas operacionais. Esta questão será explicada mais adiante.

Além dos computadores pessoais, comumente utilizados, outras arquiteturas e outras plataformas de hardware são pesquisadas até os dias de hoje. PCs eram equipamentos CISC (*Complex Instruction Set Computer*) cuja arquitetura foi inicialmente aberta para ganhar mercado, frente à Apple que utilizava arquitetura fechada e processadores RISC (*Reduced Instruction Set Computing*). A disputa entre as arquiteturas, seus produtos e soluções oferecidas se desenrolou por décadas [45,46] até chegar ao ponto de mesclar características e funcionalidades para oferecer a melhor solução possível. Daí, dentre essa evolução e a oportunidade de mercado para elaboração de dispositivos portáteis, temos o ARM (*Advanced RISC Machine*) [47].

Na computação, podemos notar que a lei de Moore (figura 3.1) – que descreve em longo prazo a duplicação da quantidade de transistores inseridos nos circuitos integrados a cada dois anos, aproximadamente – está chegando a um limite onde os equipamentos não apresentam melhorias em relação à sua velocidade além da dificuldade em manter o equipamento resfriado adequadamente e com baixo consumo de energia. Para contornar essa dificuldade, desde 2005 notamos os esforços em oferecer equipamentos com mais núcleos de processamento ("*cores*") e novas tecnologias. A empresa que continua a pesquisa nesta área é a IBM, com a linha de processadores POWER [48,49], para computação corporativa de alto desempenho.

Com a evolução das tecnologias, foi necessário padronizar uma maneira de avaliar o desempenho de cada processador e classificá-lo adequadamente, pois a avaliação dos produtos através de sua nomenclatura ou capacidade em hertz (Hz, onde um hertz equivale a um ciclo por segundo, ou seja, uma instrução por segundo) não refletia a verdadeira capacidade de processamento. Para resolver esse impasse, iniciou-se um estudo referente à quantidade de instruções matemáticas com números inteiros que um equipamento seria capaz de resolver. Daí o surgimento da métrica IPS (*Instructions Per Second*) [50] e posteriormente o FLOPS (*FLoat point Operations Per Second*) [51,52], que é a métrica utilizando operações matemáticas com números reais, ou seja, números com casas decimais.

Basicamente, a precisão simples (uso de instruções com números inteiros apenas) serve comumente para fins não científicos enquanto a precisão dupla considera os resultados obtidos com números reais (R), ou seja, onde a precisão do resultado é relevante. Analogamente podemos considerar IPS como o uso do tipo inteiro, enquanto FLOPS seria o uso do tipo *double* (tipo de precisão dupla). Assim, comparando novamente os tipos mencionados com as linguagens de programação, podemos observar que a precisão dupla é duas vezes maior que a precisão simples.

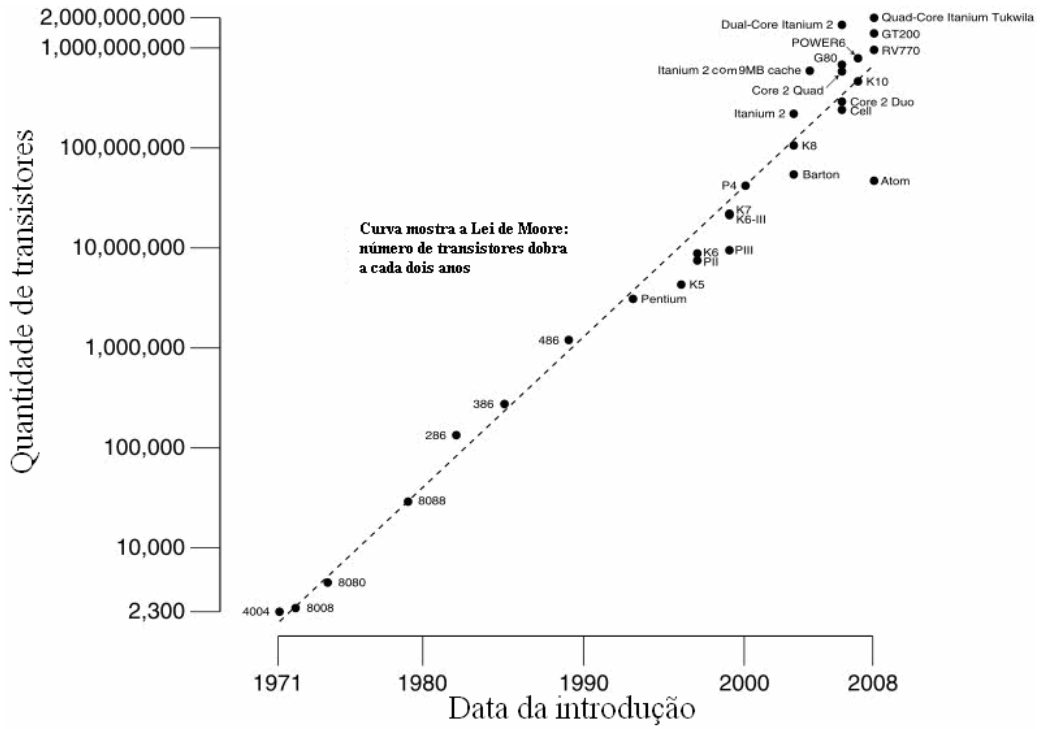


Figura 3.1: Apresenta a evolução dos processadores em relação à lei de Moore, demonstrando os modelos disponíveis no mercado desde 1971 a 2008 em relação à quantidade de transistores.

Dentre os avanços observados, algumas siglas foram introduzidas com o tempo:

HT - *Hyper-Threading*, é a tecnologia que permite a simulação de dois núcleos em um processador, tornando o sistema mais rápido quando são usados vários programas ao mesmo tempo.

MIPS - *Million Instructions Per Second*;

OS – *Operating System*, Sistema Operacional. Programa que permite ao equipamento gerenciar dispositivos e executar outros programas;

SIMD - *Single Instruction Multiple Data*, uma instrução para múltiplos dados é uma classe de computadores com suporte a paralelismo conforme a classificação de Flynn. Descreve o funcionamento do computador processando elementos de uma mesma operação simultaneamente.

SSE - *Streaming SIMD Extensions*, extensão para instruções matemáticas SIMD, com melhor tratamento de informações com ponto flutuante (recurso de precisão para casas decimais).

Devido ao pioneirismo da INTEL na produção de CPUs que equipam os computadores pessoais, todos os avanços da companhia possuem registros na Internet. Reunindo as informações publicadas foi possível tabular os dados apresentados na tabela 3.1 que referem-se aos primeiros lançamentos de cada modelo e o valor aproximado de instruções (MIPS) das CPUs INTEL [53,54,55].

Tabela 3.1: Evolução dos processadores INTEL.

Modelo	Ano	MIPS	Velocidade	Arquitetura	Destaque
4004	1971	0,0926	740 kHz	4 bit	Primeiro microprocessador
8086	1978	0.34	5 MHz	16 bit	Origem da arquitetura x86
80286	1982	0.90	6 MHz	16 bit	Aumento de desempenho
80386	1985	6.00	16 MHz	32 bit	Aumeto de endereçamento (em bits)
80486	1990	20.00	25 MHz	32 bit	Introduziu OS 32 bits
80586	1993	100.00	60 MHz	32 bit	P5/Pentium
P-III	1999	2054.00	500 MHz	32 bit	Introduziu SIMD
P4	2000	6500.00	1.5 GHz	32 bit	SSE; SIMD; HT
IA64	2001	2200.00	800 MHz	64 bit	Ainda apenas um núcleo (core)
IntelCore	2006	18000.00	1.6 GHz	64 bit	SSE3; SIMD; VMS
I64-Xeon	2006	27000.00	3.0 GHz	64 bit	Dois núcleos num mesmo chip
iSeries	2010	76000.00	3.0 GHz	64 bit	Core i3, i5, i7

3.2 – Breve histórico sobre supercomputação

"Supercomputadores são máquinas construídas sob encomenda, com capacidade de processamento grande o suficiente para resolver complexos problemas referentes às aplicações para os quais foram desenvolvidos" [56,57].

A frase exemplifica o que é a supercomputação. Equipamentos desenvolvidos com o "estado de arte da tecnologia" com a finalidade de executar operações específicas e cálculos de extrema complexidade, como aqueles realizados em análises de física mecânica, por exemplo. Outros usos para supercomputadores incluem previsão do tempo, pesquisas climáticas, criação de modelos moleculares, simulações de comportamentos estudados na física e outros semelhantes. A avaliação dos melhores equipamentos pode ser observada no TOP500 [58,59].

Top500 é uma relação com os computadores com maior poder de processamento, segundo uma lista de estatísticas organizada desde 1986, por Hans Meuer [60], organizador e fundador do primeiro evento sobre supercomputação. A relação dos supercomputadores utiliza o processo de comparação Linpack [58] para avaliar o desempenho entre dois ou mais equipamentos. O método, lista atualizada e demais informações podem ser obtidas no site [42]. Essa lista é atualizada ao menos duas vezes por ano. Tais equipamentos tem custo individual na casa de milhões de dólares norte americanos. O RoadRunner, da IBM, custou US$ 133 milhões e está operacional desde junho de 2008, trabalhando na velocidade de um petaflops.

A figura 3.2 mostra a revolução no poder de processamento oferecido pelos principais fabricantes de processadores (AMD e INTEL) entre 2000 e 2011, indicando o poderio de cálculos em FLOPS.

Podemos observar que o volume total de processamento disponível para HPC (figura 3.3 e 3.4) é crescente desde 1993, assim como o consumo de energia. Entretanto, as melhorias obtidas no processo de fabricação dos circuitos integrados permitiu a miniturização dos processadores com a mesma performance em FLOPS (figura 3.5).

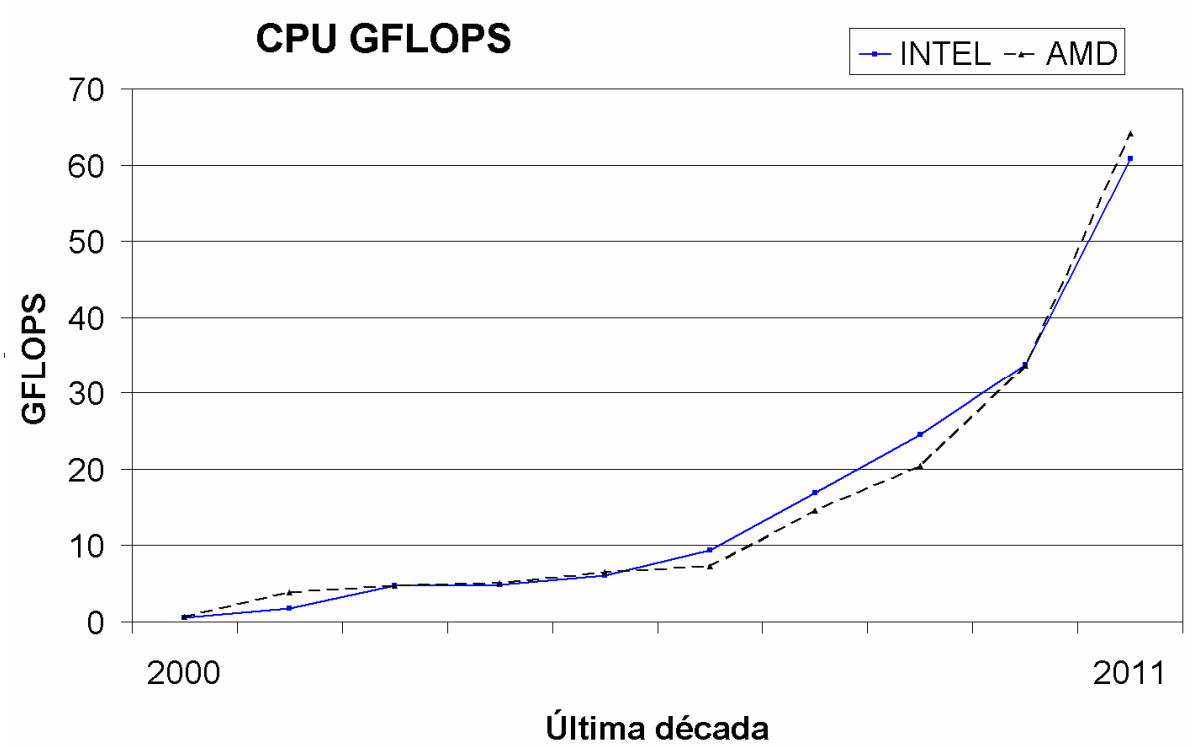


Figura 3.2: Comparação de processadores conforme a capacidade em operações realizadas com ponto flutuante por segundo. Gráfico gerado a partir das informações obtidas em fóruns de computação de alta performance (HPC) [61].

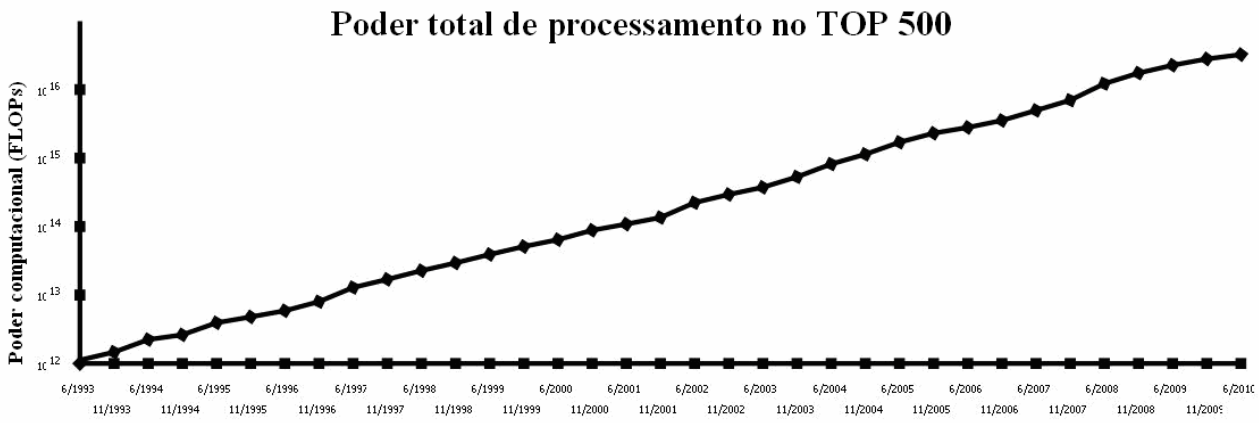


Figura 3.3: Volume total do poder computacional dos 500 melhores equipamentos do mundo, avaliados no período de 1993 a 2010. Dados em escala logarítmica.

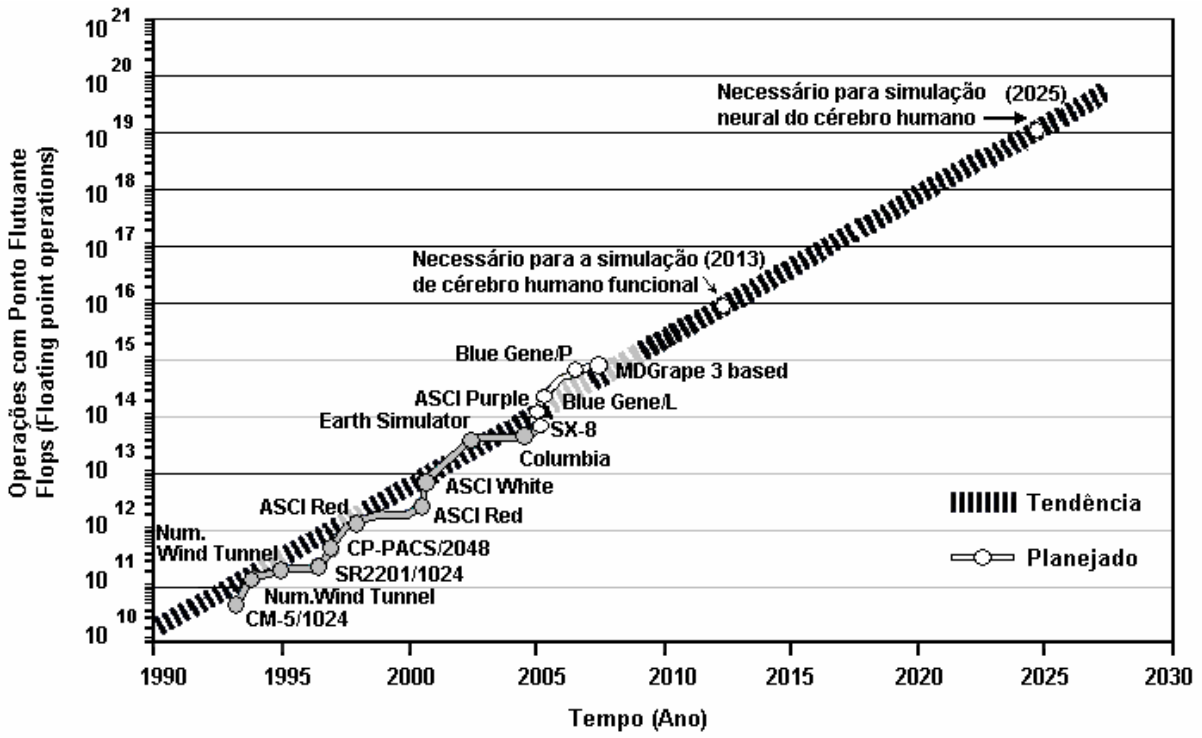
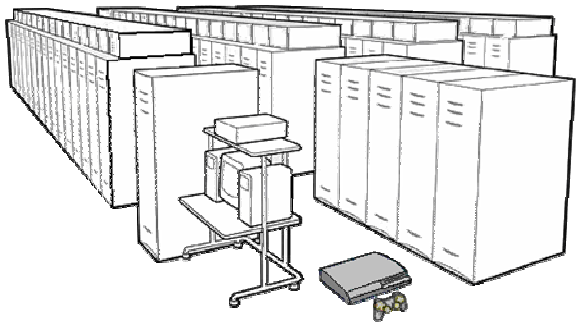


Figura 3.4: Crescimento do poder computacional dos supercomputadores [62]. Gráfico em escala logarítmica; Duplicando a capacidade a cada 1.2 anos.



	Sandia Lab's ASCI RED	**SONY Playstation 3**
Ano de Origem	1997	2006
Pico de desempenho	1.8 teraflops	1.8 teraflops*
Tamanho físico	150 m^2	0.08 m^2
Consumo elétrico	800.000 watts	<200 watts

* Para GPU; CPU acrescenta outros 0.2 teraflops

Figura 3.5: Comparação de poder computacional equivalente em relação ao tamanho e consumo de energia. Em menos de dez anos um videogame possui o mesmo poder computacional de um supercomputador [63].

O futuro da supercomputação é impressionante. Onde inicialmente os computadores foram criados para quebrar criptografia e códigos militares, evoluíram para máquinas capazes de

prever o comportamento climático, testar o comportamento de modelos de automóveis entre outras inovações [59]. O Brasil não ignora este cenário e tem seu lugar garantido entre os cinquenta melhores equipamentos do mundo [64,65]. As notícias sobre computação e processamento de alto desempenho podem ser acompanhadas no site do SINAPAD [66]. Já o poder do processamento dos supercomputadores pode estar ao alcance de todos graças ao anúncio da NVIDIA, feito em dezembro de 2008. Através da linha Tesla [67] temos a primeira promessa de supercomputação pessoal cumprida.

3.3 – Evolução das linguagens de programação, processamento paralelo e virtualização

O primeiro algoritmo, considerado o primeiro programa de computador data de 1843, quando Ada Lovelace utilizou o computador mecanico (Analytical Engine) de Charles Babbage para calcular os numeros de Bernoulli [68] (teoria numérica que calcula uma sequencia de numeros racionais). Entende-se por algoritmo a seqüência finita de passos necessários para realizar uma determinada tarefa.

A evolução da computação nas últimas décadas trouxe agilidade no meio científico, possibilitando a realização de cálculos de maneira mais rápida e eficaz num menor espaço de tempo quando comparada a execução em papel. Esta facilidade despertou o interesse de mais pesquisadores para utilizar os novos recursos.

Entretanto a execução de cálculos requer certo tempo para processamento das informações, enquanto ocorria maior procura para utilizar tais facilidades. As tarefas realizadas em computador eram organizadas para execução de processamento em lote (*batch processing*), onde não havia intervenção manual do usuário. Os dados eram inseridos no computador e aguardava-se a execução seqüencial das atividades até o término do processamento, onde então os resultados eram verificados e o usuário sabia se havia obtido sucesso ou não. Em computadores de grande porte (*mainframes*, servidores) era comum a organização de filas de trabalho para atender um maior número de usuários possível. As vantagens trazidas pelo meio científico se estenderam para o comércio e indústria.

A partir de então a história da computação registra as evolução dos equipamentos em gerações conforme as características das invenções ao passar do tempo. Inicialmente cada

equipamento tinha sua própria linguagem de máquina, específica, para realizar as atividades desejadas. Dentre os momentos importantes na evolução da ciência e tecnologia, auxiliam em diversos trabalhos até o presente as linguagens FORTRAN e C, contribuindo com os avanços publicados.

Em 1955 surge a primeira linguagem de alto nivel para uso cientifico, o FORTRAN (FORmula TRANslator) criada por John Warner Backus et al. O FORTRAN é usado até hoje com revisões periódicas, recebendo ISO/IEC 1539-1:2010 em 2010. As atualizações ao longo do tempo o mantiveram como uma linguagem apta até para tarefas em supercomputação, onde modelos numéricos são usados intensivamente em simulações, inclusive para física computacional e testes de performance utilizando números com casas decimais.

Antes da padronização dos computadores pessoais iniciada em 1981 com o IBM-PC (*International Business Machines; Personal Computer*) cada máquina possuía sua própria linguagem para trabalhar, inclusive um sistema operacional específico para cada modelo fabricado. O custo dos computadores naquela época era alto e já havia demanda para uso no meio científico. Foi então desenvolvido um sistema operacional multiusuário, capaz de efetuar múltiplos acessos ao sistema, compartilhado por diversos usuários. Como resultado surgiu o MULTICS (*Multiplexed Information and Computing Service*).

O MULTICS era um sistema operacional de tempo compartilhado (CTSS, *Compatible Time-Sharing System*), instalado em servidores projetados para atender requisições de diversos usuários ao mesmo tempo. Esta infraestrutura centralizada foi desenvolvida com ideais a frente de sua época e era capaz de oferecer alta acessibilidade através de sua estrutura modular. Dessa maneira a equipe técnica poderia realizar a análise do funcionamento do servidor e efetuar expansão de recursos do computador, se necessário.

Em 1969 Dennis Ritchie e Ken Thompson iniciaram o desenvolvimento da linguagem C nos laboratórios da AT&T Bell, que se tornou disponível em 1973 como uma linguagem de propósito geral. A portabilidade de códigos entre equipamentos tornou a linguagem C uma das mais utilizadas em todos os tempos cuja disponibilidade abrange diversas arquiteturas e sistemas operacionais. Em 1989 recebeu padronização (ANSI, American National Standards Institute) e é atualizada constantemente, sendo C11 a revisão publicada em dezembro de 2011, atribuindo novas caracteristicas à linguagem.

Na década de 1970 o MULTICS foi reescrito em linguagem C, sendo posteriormente chamado de UNIX [69]. Esta evolução do sistema operacional permitiu atender a um maior número de usuários, assim como ficou mais explícita a necessidade de um bom gestor de atividades num dado meio. Qualquer programa de computador trabalha numa camada superior, uma vez que o sistema operacional (SO) é o responsável pela verificação do equipamento durante a inicialização e a camada fundamental entre o hardware e os demais programas.

Sem um SO o computador é um amontoado inútil de circuitos, pois não há como efetuar interação direta com o hardware. A figura 3.6 mostra essa hierarquia, onde o usuário utiliza o computador através do sistema operacional, que permite o uso de aplicações (editor de texto, planilha de cálculo, entre outros programas).

O MULTICS trouxe a capacidade de cada usuário ter a impressão de possuir a seu dispor um computador completo, sem interferir no trabalho de outros usuários. Esta área de trabalho virtual (VDI, *Virtual Desktop Infrastructure*) possuía características semelhantes ao serviço de terminal (*terminal service*) usado pela Microsoft em estruturas cliente-servidor.

No início da década de 1990, o UNIX foi reinventado de maneira simplificada, numa versão com suporte a interface gráfica chamado GNU/Linux, com código-fonte aberto para permitir a continuidade de inovações, além do acesso irrestrito e gratuito garantido pela GPL [70]. A história dessa evolução pode ser observada no anexo B.

Figura 3.6: Hierarquia de comunicação.

Os sistemas operacionais distribuídos deixaram como legado à programação distribuída as facilidades características no modelo cliente-servidor (figura 3.7), ou seja, os mecanismos de rede necessários para viabilizar o agrupamento de recursos e distribuição das tarefas entre os sistemas

interconectardos, realizado de maneira explícita, ou seja, o programador é quem define como a distribuição ocorrerá.

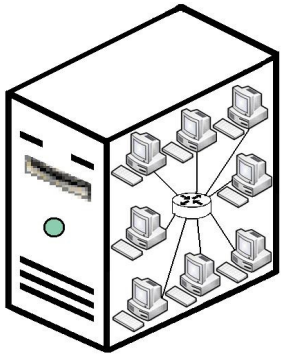


Figura 3.7: Programação distribuída (cliente-servidor)

Os aplicativos que se beneficiam da programação distribuída geralmente são escritos em linguagem C e/ou FORTRAN. Dentre as opções disponíveis, as mais comuns são:

PVM (*Parallel Virtual Machine*) [71,72] que é um pacote de software que agrupa computadores variados (WINDOWS/UNIX), conectados em rede como um único computador maior. Seu código-fonte e programa compilado estão disponíveis na Internet para as mais diversas plataformas, gratuitamente. É usada principalmente para simulações de dinâmica molecular, algoritmos matriciais e em ensino de computação concorrente.

OpenMP (*Open Multi-Processing*) [73] é uma interface para programação em ambiente com memória compartilhada em múltiplas plataformas. Permite implementar paralelização direta através de declarações explícitas. Permite dividir tarefas e reagrupar resultados. Melhores resultados são obtidos com uso em conjunto ao MPI (MPMD, *Multiple Program Multiple Data*).

MPI (*Message Passing Interface)* [74] é um padrão para comunicação de dados em computação paralela, oferecendo infraestrutura para trabalhar com vários processadores ou nodos (computador conectado a rede) de um cluster.

CUDA (*Compute Unified Device Architecture*) [75] é uma plataforma de computação paralela de propósito geral que permite utilizar ao máximo o potencial dos processadores gráficos produzidos

pela NVIDIA e que tem suporte a esta tecnologia. Maiores informações serão discutidas a seguir no item 3.5.

As opções de recursos mencionados servem para programação distribuída, ou seja, são maneiras onde se pode distribuir as tarefas (cálculos) desejadas entre uma arquitetura escolhida para otimizar a utilização de recursos computacionais e obter o resultado no menor tempo possível.

3.3.1 – Emuladores e máquinas virtuais

Por volta de 1965 pesquisadores da IBM avaliavam sobre a eficácia real de novas idéias para aplicações computacionais onde a performance da máquina era medida ativando ou desativando algumas funções do sistema. O estudo consistia em particionar a máquina em partes menores, capazes de gerenciar cada recurso individualmente, acreditando que a implementação de um ambiente virtualizado serviria ao propósito do experimento. A partir desta idéia, uma solução capaz de oferecer recursos para múltiplos usuários foi comercializada, como IBM System 370/390 (S/370 ou S/390).

A solução através de um ambiente virtual surgiu com a intenção de auxiliar em questões de portabilidade e design entre sistemas operacionais. A linguagem C para programação e o sistema operacional UNIX surgiram como pioneiros nessa solução onde mais tarde, por volta de 1985 iniciou-se o trabalho referenciado como IEEE 1003 (POSIX, *Portable Operating System Interface*) [76] para padronizar funções e bibliotecas do sistema operacional. A promessa de portabilidade entre plataformas se tornou possível somente anos mais tarde, com avanços na padronização das linguagens de programação.

Em busca do ambiente virtualizado encontramos soluções para emulação e virtualização de sistemas. Isto permite a execução de um determinado programa criado para uma plataforma A que é executado em um equipamento da plataforma B, sem a necessidade de qualquer modificação do programa.

Uma emulação replica exatamente as características da plataforma desejada, como comportamento de processador, memória e demais dispositivos, refletindo as características e

limitações de determinada plataforma. Como exemplo, há programas para computador onde replica-se as características de consoles de vídeo-games, celulares e outras plataformas de computadores (Amiga, Commodore, MSX, entre outros).

A virtualização consiste em disponibilizar ao ambiente convidado (guest) os mesmos recursos do sistema hospedeiro (host), compartilhando o acesso aos dispositivos reais de hardware. Funciona de maneira muito eficiente para hospedar diversos sistemas da mesma plataforma.

Assim, notamos que há espaço para ambas as soluções, dependendo da maneira como os programas precisarão funcionar nas plataformas hospedadas [77]. A evolução no gerenciamento de máquinas virtuais e da maneira como esses sistemas utilizam os recursos disponíveis num servidor melhorou bastante a partir de 2007, permitindo um avanço rápido nesta área desde então.

Uma máquina virtual (VM, *Virtual Machine*) é um programa de computador capaz de oferecer todo o ambiente computacional que um sistema operacional necessita para ser instalado. De modo simples podemos dizer que cada VM é "um computador dentro do computador".

O gerenciamendo dos recursos disponíveis pelo computador é feito pelo gestor, conhecido como hypervisor ou monitor (VMM, *Virtual Machine Monitor*) [78,79] que administra o processamento, uso de memória RAM e discos (reais e virtuais) entre outros recursos dos sistemas em uso. A classificação do gestor pode ser: Tipo 1, nativo (*unhosted*), onde o hypervisor lida diretamente com o equipamento em uso (implementação clássica iniciada na década de 1960); ou Tipo 2, hospedado (*hosted*), onde o gestor atua sobre um ambiente operacional, conforme ilustrado na figura 3.8.

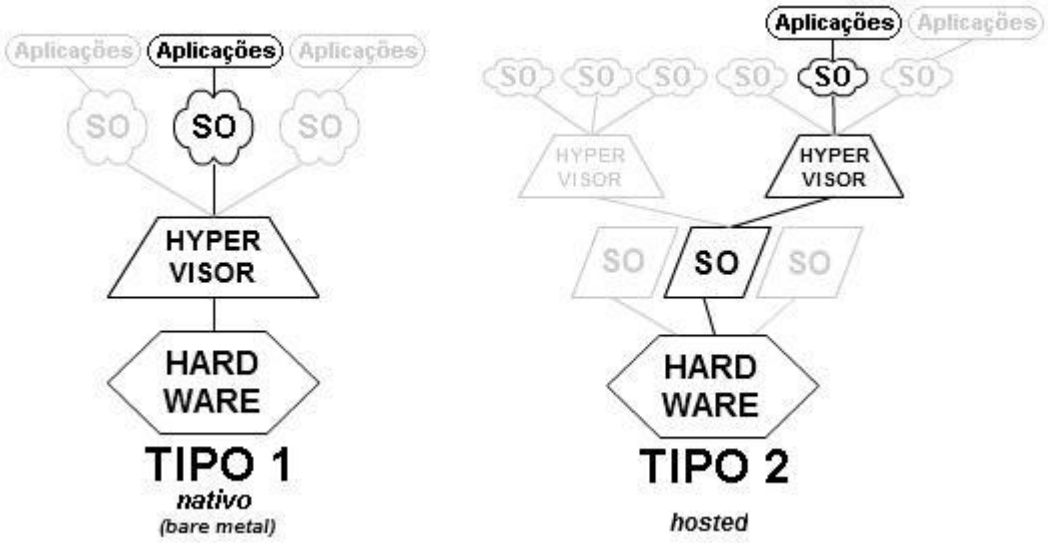


Figura 3.8: Tipos de hypervisor para virtualização

Como benefícios da virtualização, podemos destacar a possibilidade de:

- Verificar o funcionamento de programas sem comprometer a instalação de um sistema operacional, seja pelo aditivo de recursos, contaminação por malware ou vírus, assim como travamentos aleatórios e recuperação de desastres;
- Executar múltiplos sistemas operacionais ao mesmo tempo, num mesmo computador, sem a necessidade de reiniciar a máquina;
- Executar programas desenvolvidos para plataformas ou sistemas operacionais diferentes ajustando os recursos utilizados e permitindo compatibilidade, mesmo quando um dispositivo (hardware) não é suportado pelo programa "antigo";
- Disponibilizar aplicações para acesso remoto;
- Consolidar a infraestrutura, reduzindo custos com equipamentos e consumo de energia elétrica. Economia real, devido ao baixo consumo de potencial que diversos equipamentos atuais oferecem.

Dentre várias opções e projetos existentes, a seguir, estão relacionados alguns programas que foram avaliados para simulações no CCS/UNICAMP. O desempenho e escolhas são discutidas no capítulo 4.

BOCHS [80] é um software livre, com código-fonte aberto [70], emulador da plataforma x86. Permite utilizar qualquer sistema operacional disponível para plataforma x86 (iOS, Linux, MacOS, MS-DOS, Windows, entre outros). Sua utilização é robusta, porém não é trivial a configuração do programa para instalação de VMs. Requer a instalação de um modelo e seu estudo (obtido no website do projeto). Versão mais recente é 2.6.2 de 26/05/2013. Na figura 3.9 é possível observar como é a interface do BOCHS.

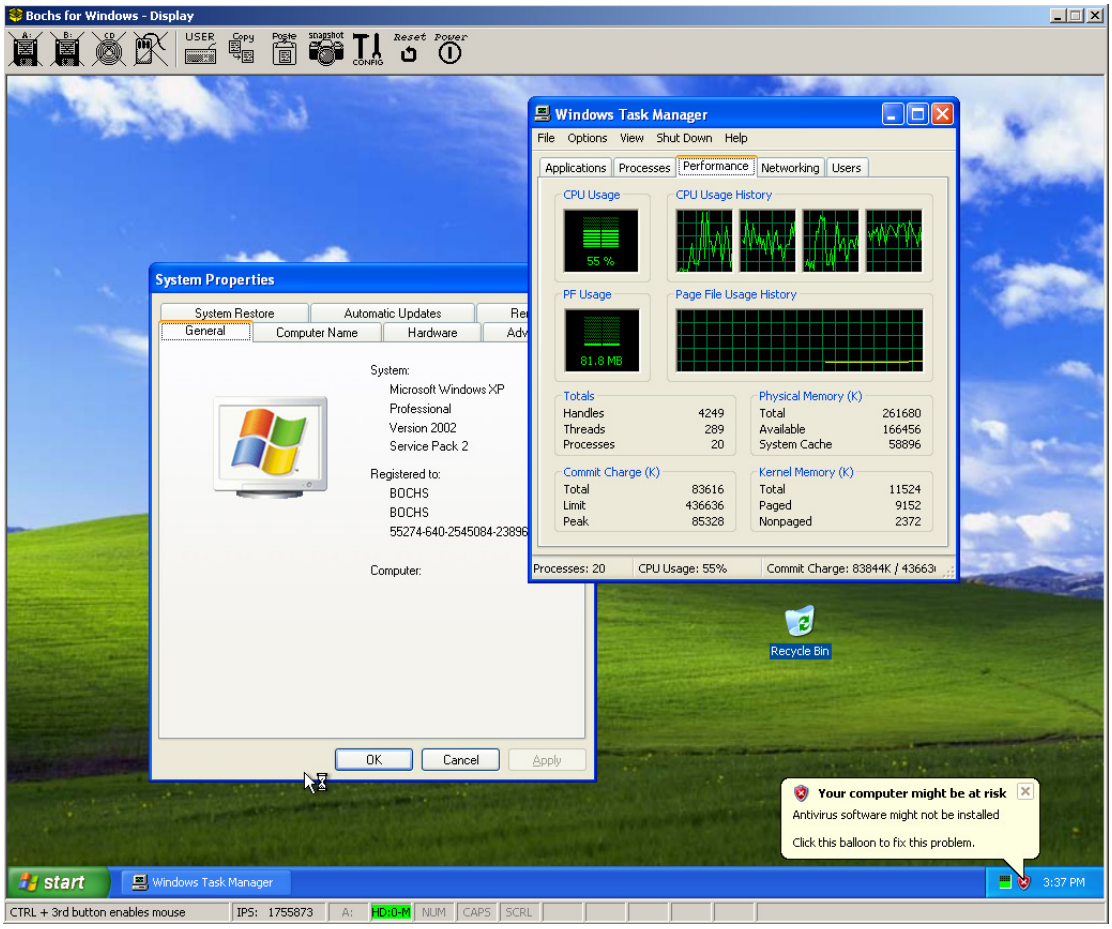


Figura 3.9: Exemplo da janela do emulador BOCHS, executando Windows XP.

Microsoft VirtualPC [81] é o programa proprietário, de uso gratuito, que emula um outro computador em ambiente Windows 7 Profissional ou superior. Permite utilizar VMs com outros sistemas operacionais além do MS-DOS (Microsoft Disk Operating System) e Windows (95 ao 8.1). Para instalar uma cópia deste programa é necessário validar o sistema operacional (verificação do host com o GenuineCheck.exe) que fornecerá então um código para continuar

com o processo de instalação. Versão mais recente é 6.1.7600.16393 de 14/02/2011. Na figura 3.10 é possível destacar a configuração de uma máquina virtual neste programa; já a figura 3.11 traz uma máquina virtual em funcionamento.

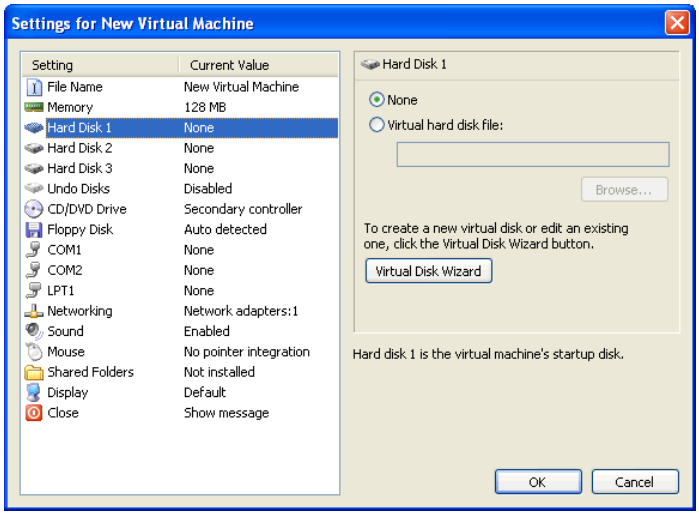


Figura 3.10: Exemplo de configuração da máquina virtual com Microsoft Virtual PC

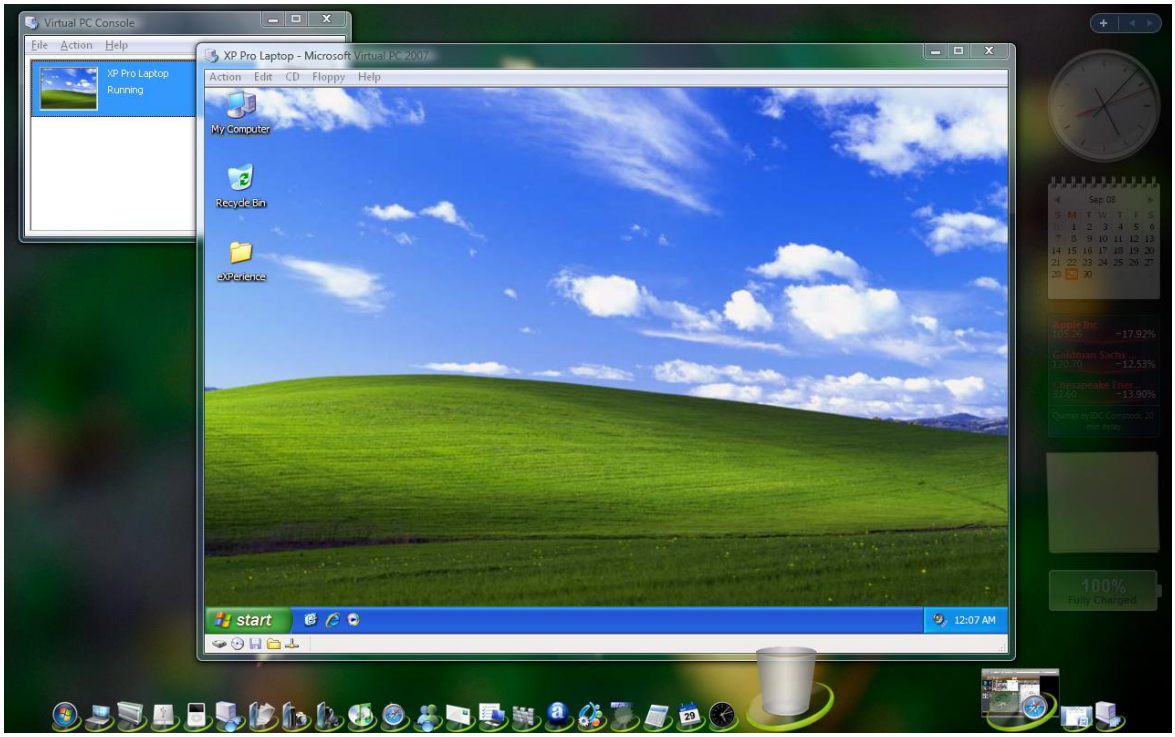


Figura 3.11: Exemplo de um Windows XP hospedado em um Windows Vista com Microsoft Virtual PC

QEMU [82] é um software livre, com código-fonte aberto [70], escrito por Fabrice Bellard, que permite executar máquinas como emulador e virtualizador. Seu hypervisor é semelhante ao de outros projetos como o Bochs e VMware, mas possui diversas características complementares, incluindo aumento de velocidade em x86, através do acesso direto ao hardware via KVM (*Kernel virtualization machine*). Quando utilizado como emulador é capaz de executar um sistema operacional próprio de uma arquitetura diferente daquela onde o programa está em execução e proporcionar uma tradução dinâmica entre as plataformas com boa performance. Quando utilizado como virtualizador, permite a execução de clientes (guest) com performance muito próxima a do sistema nativo (host). Versão mais recente 1.5.3 (estável) de 27 de agosto de 2013. A execução do QEMU abre a máquina virtual diretamente. Existem algumas interfaces para facilitar a utilização, como o QEMU Manager, apresentado na figura 3.12.

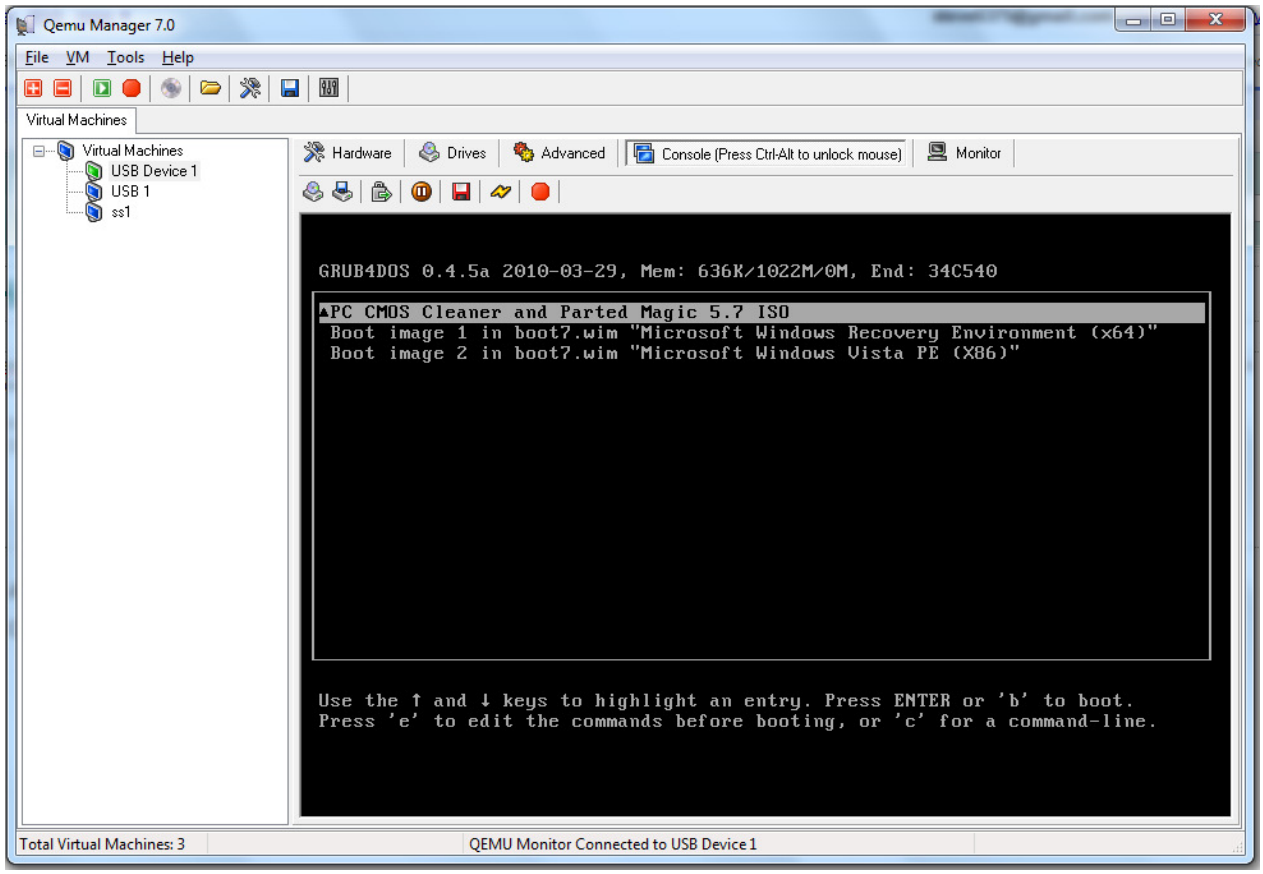


Figura 3.12: Interface QEMU Manager 7.0 [83]

VirtualBOX [84] é um software livre, com código-fonte aberto [70], de propriedade da Oracle ™, que permite a instalação de programas e sistemas para atuarem como Desktops, Servidores ou sistemas embarcados. Permite executar diversas plataformas simulataneamente, onde o limite prático é o espaço em disco e memória. Versão mais recente 4.2.16-86992, 95.2M (.exe), 106MB, 61.4MB/115MB. Sua interface, apresentada na figura 3.13, é simples e intuitiva. Permite aos usuários iniciantes fácil utilização de máquinas virtuais.

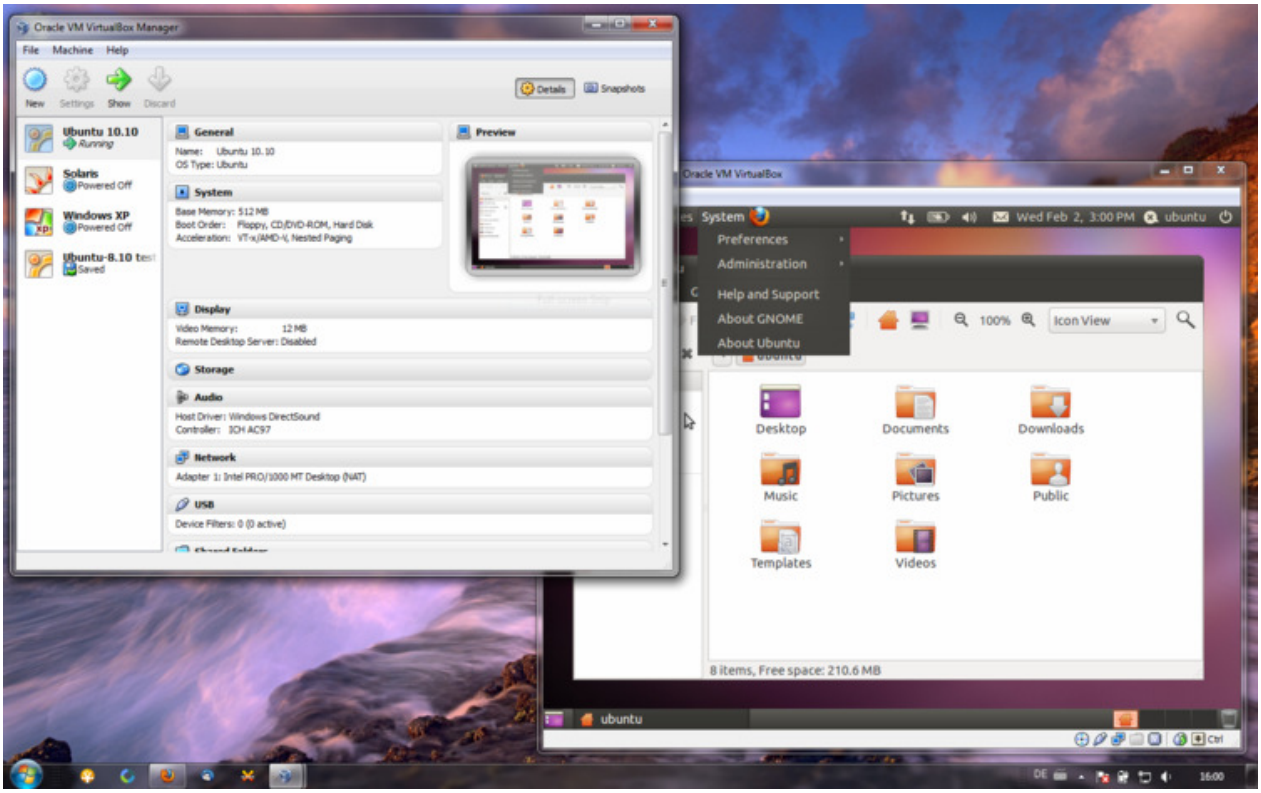


Figura 3.13: Exemplo de utilização do VirtualBox com VM Ubuntu

VMware [85] é uma empresa que comercializa produtos para virtualização de ambientes computacionais, atendendo a diversos nichos de mercado. A versão VMware Player [86] é uma distribuição gratuita para uso pessoal, que permite criar máquinas virtuais, utilizando o mesmo núcleo da versão Workstation (mais completo, com diversos recursos complementares). Suporta gestão de diversos sistemas operacionais para plataforma x86. Versão mais recente é 6.0.0 release 1295980 de 03/09/2013. A figura 3.14 ilustra a interface do Player com duas máquinas para testes. Quando selecionada uma máquina virtual, os detalhes aparecem ao lado direito. A figura 3.15 ilustra uma máquina virtual em funcionamento.

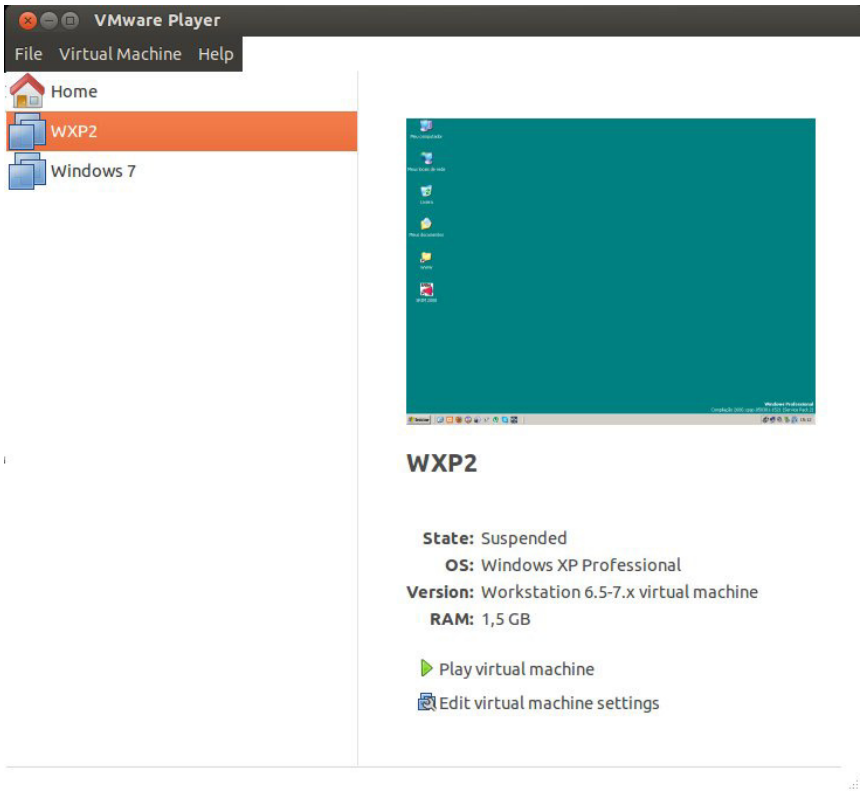


Figura 3.14: Exemplo da tela inicial do VMware Player.

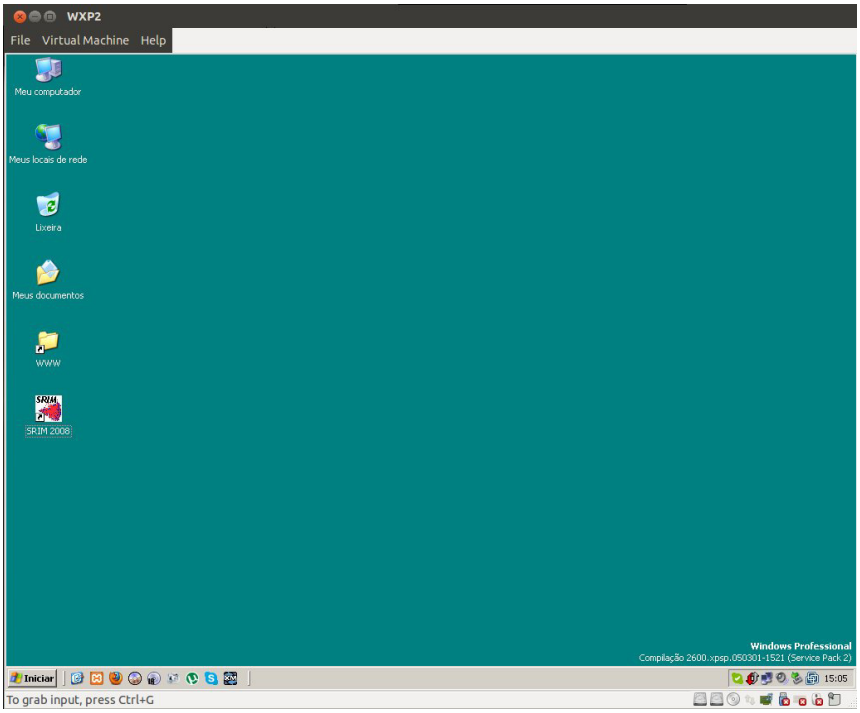


Figura 3.15: Exemplo da máquina virtual com Windows XP utilizada nas simulações com TRIM/SRIM

3.4 – Evolução dos processadores gráficos

Os primeiros terminais de vídeo apareceram por volta de 1973 como interface homem-máquina para servidores UNIX. Inicialmente os dispositivos foram projetados para oferecer retorno gráfico e facilitar a vida dos usuários de sistemas computacionais, que até então realizavam diversas tarefas às escuras, ou seja, sem visualizar as etapas ou os resultados daquilo em que trabalhavam. Até este advento era necessário aguardar todo o processamento de informações até detectar alguma falha ou erro, para então corrigir e realizar novamente toda a tarefa. Com a facilidade de visualizar o que acontecia, foi possível intervir durante a execução de um determinado processo computacional e com isto economizar muito tempo de trabalho.

Desde então esses terminais gráficos evoluíram, os computadores saíram da interação textual com o surgimento de interfaces gráficas em meados das década de 1980 e desde então evoluem constantemente até a tecnologia de interação com toque na tela, disponível nos tablets atualmente. Analisando esta evolução podemos observar que no princípio os monitores eram adaptações dos televisores domésticos e foram adequados e aprimorados com o passar do tempo [87].

A necessidade de representar melhor as informações gráficas propiciou o desenvolvimento de unidades de processamento gráficos (GPUs, *Graphical Processing Units*), cujas aplicações exigem milhões de cálculos ao mesmo tempo para oferecer uma resposta o mais breve possível. Além do mercado computacional foi o mercado voltado aos jogos que fomentou a pesquisa e desenvolvimento de unidades gráficas. Inicialmente as imagens representadas em vídeo possuíam pouca definição (lado esquerdo das figuras 3.16, 3.17 e 3.18) ou qualidade. Quando foi possível melhorar o processamento de imagens, obtivemos melhores resultados (lado direito das figuras 3.16, 3.17 e 3.18), com efeitos reais dos objetos gráficos do cenário.

Essa evolução da representação de gráficos permitiu, no final de 2006, o surgimento das GPUs NVIDIA série GeForce 8, primeira a suportar a arquitetura de programação paralela (lançada em 2007 também pela NVIDIA): CUDA (*Compute Unified Device Architecture*), através de recursos acessíveis somente com interface de programação de aplicativos (API, *Application Programming Interface*; HLSL, *High-Level Shader Language*; GLSL, *openGL Shading Language*; e Cg, acrônimo de *C for graphics* que é uma linguagem de alto nível para efeitos de sombreamento de imagens) rodando diretamente na placa gráfica, resultando em

melhora de desempenho e maior flexibilidade para os desenvolvedores. A GPGPU (*General Purpose* GPU) trouxe benefícios para computação de propósito geral (que não seja gráfico) somente quando amadureceu o suficiente para oferecer aquilo que foi denominado GPU Computing, ou seja, GPUs totalmente programáveis, consideradas as da última geração.

Com isso, o processador gráfico evoluiu para um modelo com vários núcleos e potencial computacional massivamente paralelo, com um gigantesco poder de processamento graças a grande largura do barramento de memória [88]. Dessa forma, enquanto as CPUs disponíveis no mercado possuem até 12 núcleos, é possível encontrar GPUs com centenas de núcleos disponíveis a pronta entrega, preparadas para calcular em poucos instantes.

Figura 3.16: Diferenças entre a qualidade de imagens com 8 bits em relação à outra com 24 bits.

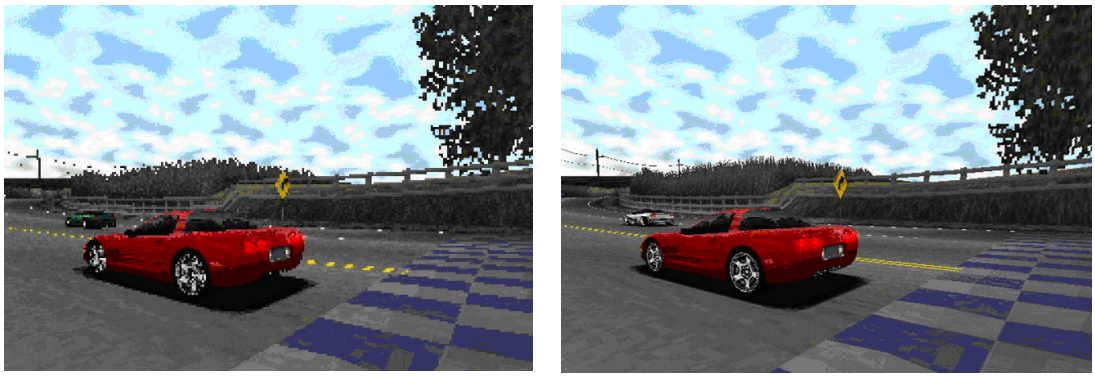


Figura 3.17: Melhoria na qualidade de apresentação de objetos gráficos

Figura 3.18: Mais realidade aos jogos, com melhores resoluções dos gráficos.

A figura 3.19 mostra as características de arquitetura, presentes em CPUs e GPUs. Essa estrutura multiprocessador é o que permite o maior tratamento de volume de dados. Observe que enquanto a GPU possui em cada "linha" uma unidade de controle com memória cache e um conjunto de unidades lógico-aritméticas (ALU, *Arithmetic Logic Unit*), a CPU possui um número reduzido de ALUs, limitando o poder de cálculos em cada processador.

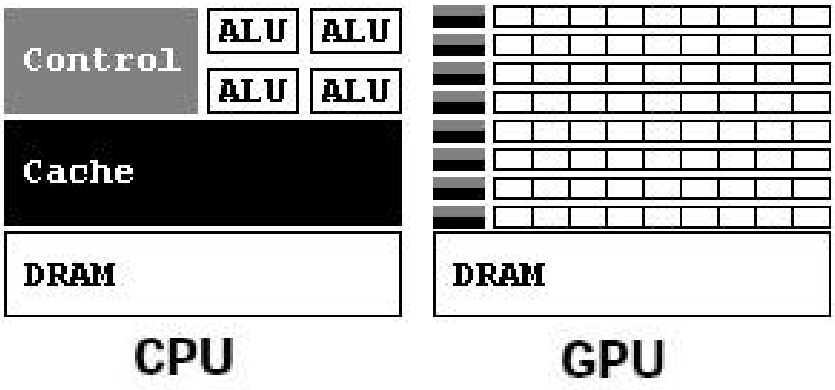


Figura 3.19: Exemplo da estrutura de um processador regular (CPU) em relação à estrutura de um processador gráfico (GPU).

O próximo passo na fabricação de sistemas especializados com maior capacidade de processamento pode vir da maior integração entre CPU e GPU através de circuitos integrados do tipo SoC (*System on Chip*, ou seja, um único chip que agrega funções). A NVIDIA já possui pesquisas nesta área, integrando processadores ARM a sua GPU no produto Tegra Series [89], a nova plataforma para supercomputação portátil, pioneira no segmento.

A figura 3.20 traz um comparativo entre o poder de processamento entre CPUs e GPUs desde 2002 até 2011, complementado pela figura 3.21 que faz a projeção entre CPUs e SoCs Tegra e a tabela 3.2 destaca detalhes contidos na figura 3.21.

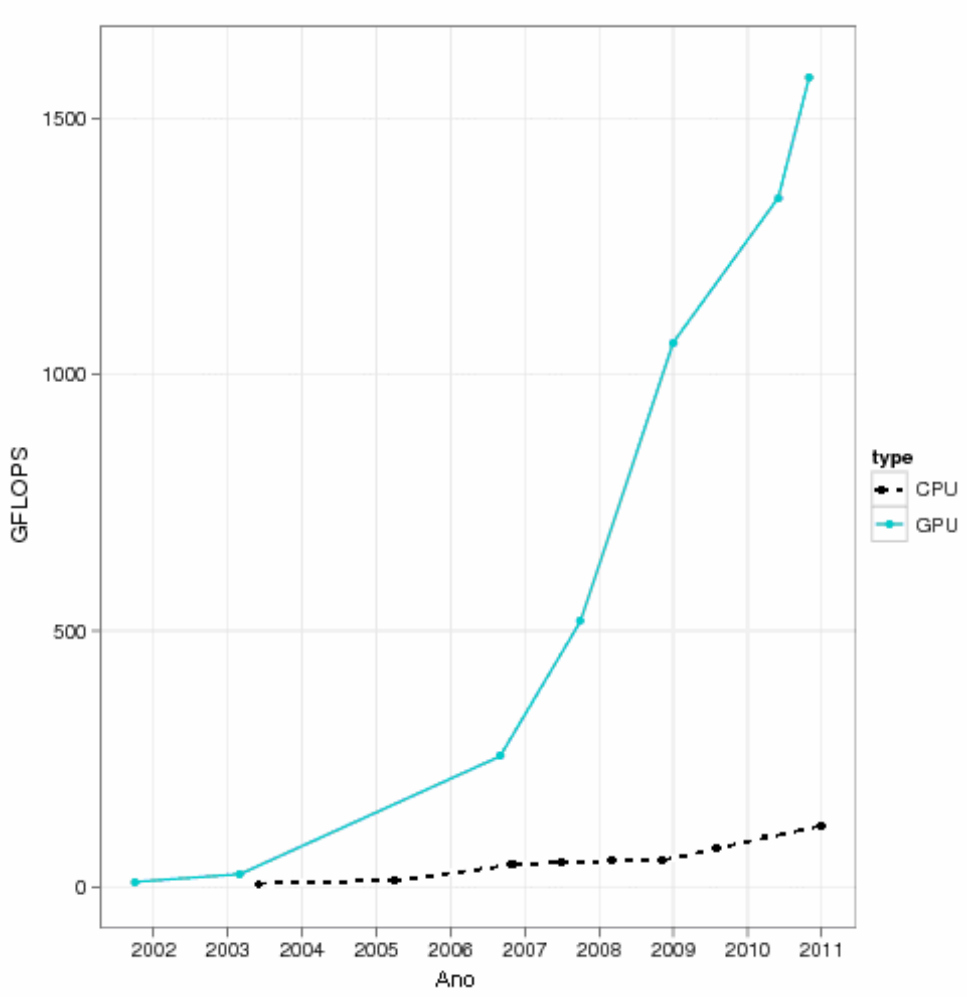


Figura 3.20: Comparativo entre o poder computacional entre CPUs e GPUs [90,91,92].

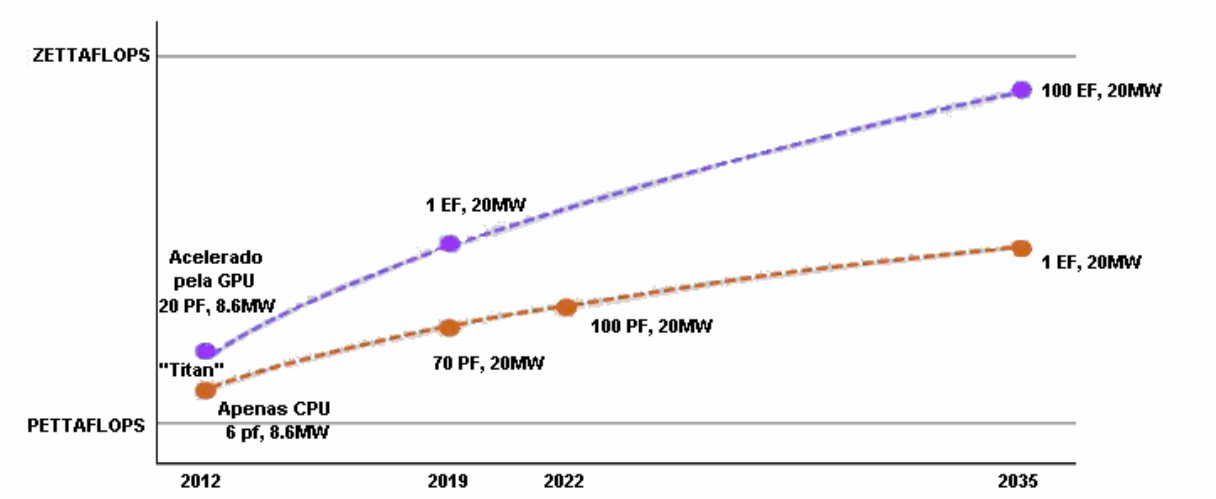


Figura 3.21: Previsão da evolução de processamento de supercomputadores [93].

Tabela 3.2: Informações sobre desempenho computacional apresentado na figura 3.21.

Desempenho Computacional		
Ordem de Grandeza	Quantidade (FLOPS)	Sigla
MEGAFLOPS	1.0E+06	MF
GIGAFLOPS	1.0E+09	GF
TERAFLOPS	1.0E+12	TF
PETAFLOPS	1.0E+15	PF
EXAFLOPS	1.0E+18	EF
ZETTAFLOPS	1.0E+21	ZF
YOTTAFLOPS	1.0E+24	YF

3.5 – CUDA: Processamento paralelo de baixo custo

CUDA (*Compute Unified Device Architecture*) é uma biblioteca proprietária da NVIDIA que consiste de uma extensão da linguagem C para possibilitar a programação de GPUs. As GPUs são dispositivos massivamente paralelos e com alta densidade aritmética. O intuito principal da CUDA é facilitar o desenvolvimento de aplicações GPGPU (*General Purpose Graphics Processing Unit*, ou seja, unidade de processamento gráfico de propósito geral) em placas de vídeo NVIDIA. A programação de GPGPU é aquela que usa da GPU não apenas para tarefas de reenderização gráfica, mas também para outras como processamento de imagens, visão

computacional, inteligência artificial, cálculo numérico, entre outras possibilidades. A figura 3.22 ilustra a estrutura de uma GPGPU.

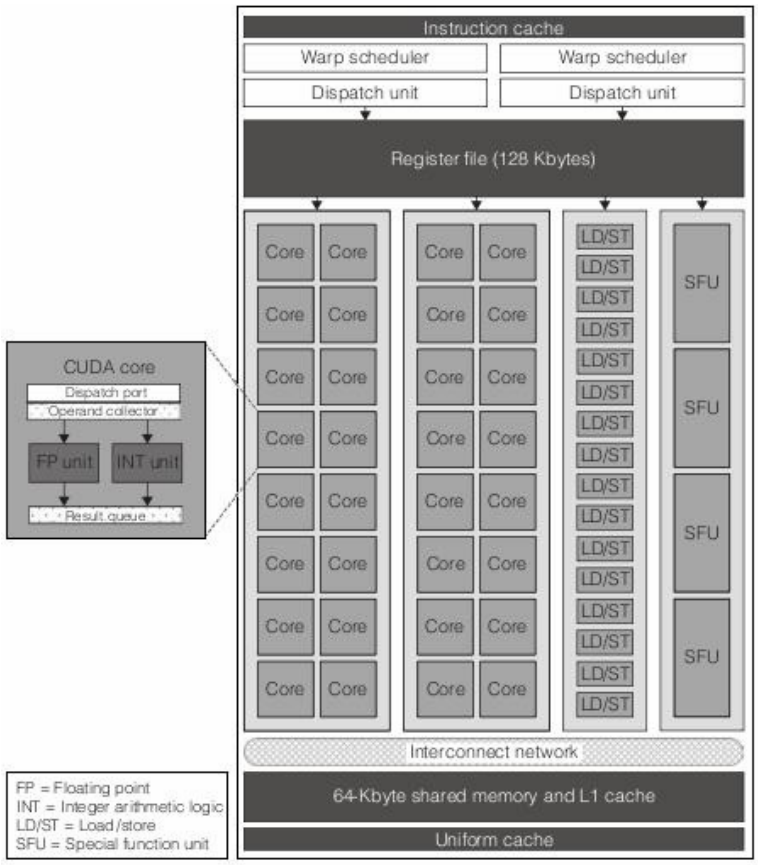


Figura 3.22: Exemplo da estrutura de uma GPGPU.

Podemos destacar as características principais para utilização de GPGPUs

Vantagens:

- GPU NVIDIA é um dispositivo massivamente paralelo;
- Tem grande desempenho para cálculos aritméticos;
- Fácil acesso a tecnologia.

Desvantagens:

- GPU NVIDIA não é componente padrão em todos os computadores;
- Não trabalha bem com código seqüencial ou recursivo;
- Há perda de tempo na transmissão de dados entre CPU e GPU.

A figura 3.23 apresenta o crescimento do poder computacional obtido em supercomputadores equipados com GPUs Tesla [94] dentre a classificação Top500.

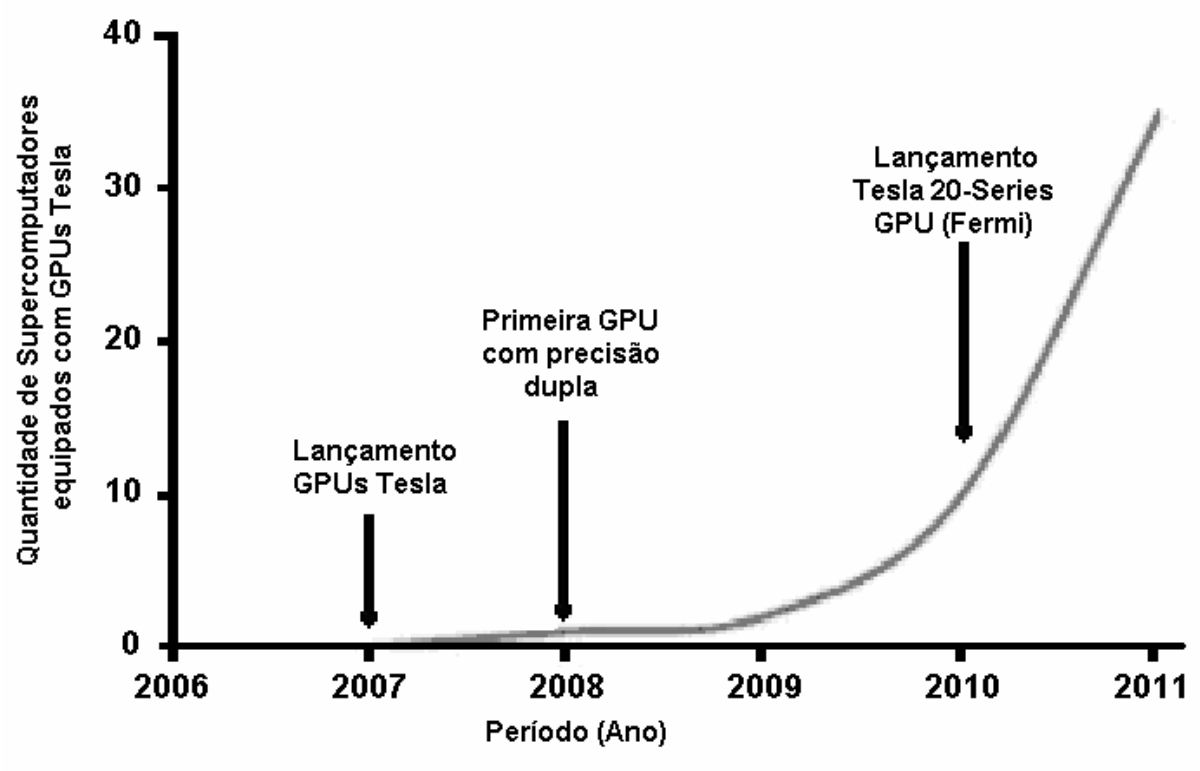


Figura 3.23: Crescimento exponencial de Supercomputadores em cinco anos

A tabela 3.3 traz as projeções para aumento de performance dos produdos NVIDIA.

Tabela 3.3: Previsões sobre performance para NVidia em 2014 [95].

	1994	**2004**	**2014**
CPU Frequency (GHz)	.1	3.2	**100**
Memory Frequency (GHz)	.03	1.2	**44**
Bus Bandwidth (GB/sec)	.1	4	**160**
Hard Disk Size (GB)	.5	200	**30 TB**
Pixel Fill Rate (MPixels/sec)	.40	3300	**270 GP/s**
Vertex Rate (MVerts/sec	.5	356	**127 GV/s**
Graphics flops (GFlops/sec)	.001	40	**10 TF/s**
Graphics Bandwidth (GB/sec)	.3	30	**3 TB/s**
Frame Buffer Size (MB)	2	256	**32 GB**

Dados reais em GHz e respectivas previsões

Além das projeções de aceleração de processamento com GPGPUs, os resultados desta pesquisa encontram-se no capítulo 4.

A seguir, para demonstrar as diferenças entre a linguagem C tradicional e o respectivo código-fonte para CUDA, a tabela 3.4 ilustra o algoritmo SAXPY (utilizado para calcular: y = ax + y) onde o bloco A (a) é o código-fonte serial (para CPU) e o bloco B (b) traz um trecho equivalente de código paralelo (para GPGPU).

Tabela 3.4: Trecho de código-fonte em liguagem C (a) e CUDA (b)

```
void saxpy(uint n, float a,
           float *x, float *y)
{  uint i;
   for (i=0; i < n; ++i)
      y[i] = a*x[i] + y[i];
}

void serial_sample()
{  // Chama a funcao serial SAXPY
   saxpy(n, 2.0, x, y);
}
```

(a)

```
__global__ void saxpy(uint n, float a,
                      Float *x, float *y)
{  uint i=blockIdx.x*blockDim.x + threadIdx.x;
   if (i < n)
      y[i] = a*x[i] + y[i];
}

void parallel_sample()
{  // Carrega o kernel paralelo SAXPY usando [n/256]
   // blocos de 256 threads cada
   saxpy<<<cell(n/256),256>>>(n, 2.0, x, y);
}
```

(b)

Como pode ser observado no trecho acima, não há mudanças muito bruscas na maneira como um programa em linguagem C (padrão C99) é escrito e o mesmo código para CUDA. Algumas mudanças que o desenvolvedor deverá notar é a maneira como o código é tratado. A porção do código que for executada pela CPU, será indicada para o "host" (hospedeiro ou anfitrião), enquanto a porção de código que for executada na GPU será indicada para o *device* (dispositivo). Assim encontramos a primeira especificidade. Há distinção entre a porção de código executado pela CPU e código excutado na GPU.

Algumas características que podem melhorar o desenvolvimento de aplicações em CUDA estão relacionados à forma como o código é escrito. Uma abordagem mais completa com exemplos pode ser obtida no website NVIDIA [96].

Podemos destacar algumas das especificidades da linguagem CUDA para exemplificar partes de seu funcionamento e como isto é empregado na orientação a objetos, pois a linguagem

CUDA especifica onde e qual a maneira em que o código estará acessível. Isto é chamado de qualificador de função e pode ser:

- `global`: A função pode ser chamada tanto pelo hospedeiro (*host*) quanto pelo dispositivo (*device*). Neste caso a função executa na GPU;
- `host`: A função é acessível apenas em CPU como um código escrito em C puro e acessível somente na CPU.
- `device`: A função é acessível somente na GPU.

Quando omitido o qualificador de função, o compilador CUDA assume que a função é executada na CPU, estando o qualificador `host` implícito na função. Ainda que a GPU atue como escravo e tem sua própria memória dedicada, é necessário alocar memória para as variáveis que vão ser passadas para as funções que executam na GPU. A função que desempenha este papel para alocação de memória é **`cudaMalloc`**. Tal função tem o seguinte protótipo:

```
cudaMalloc ( void **ptr, size_t numBytes );
```

O primeiro argumento é o endereço de memória do ponteiro (`**ptr`) que estará na GPU enquanto o segundo argumento é o número de bytes (`numBytes`) a serem alocados para esse ponteiro. Semelhante ao que utilizamos em linguagem C há um equivalente para cópia de dados em memória, a função `cudaMemcpy`. A diferença é que neste caso há um argumento adicional que informa a origem (`*src`) e o destino (`*dest`) da informação que será copiada na memória. O protótipo para a função é:

```
cudaMemcpy ( void *dest, void *src, size_t numBytes, enum cudaMemcpyKind tipo );
```

Nele temos:

- O primeiro argumento é o endereço de memória para onde os dados serão transferidos;
- O segundo argumento é de onde os dados serão copiados;
- O terceiro argumento define quantos bytes serão copiados;
- O ultimo argumento é uma enumeração que informa origem e destino.

Os valores do último argumento podem ser:

- `cudaMemcpyDeviceToHost`: Especifica que a transferência acontece do dispositivo para o anfitrião;
- `cudaMemcpyHostToDevice`: Especifica que a transferência acontece do anfitrião para o dispositivo;
- `cudaMemcpyHostToHost`: Especifica que a transferência acontece entre dois locais de memória residentes no hospedeiro;
- `cudaMemcpyDeviceToDevice`: Especifica que a transferência acontece entre dois locais de memória residentes no dispositivo.

Para iniciar os estudos com a linguagem CUDA, basta utilizar algum dos tutoriais disponíveis no apêndice A para instalação dos programas necessários em um computador desktop. Os exemplos que acompanham o pacote de desenvolvimento são muito úteis para aprender como compilar os programas (`makefile`) e observar o funcionamento dos exemplos para entender como os dados são tratados. A seguir, utilize a documentação disponível no formato eletrônico [96] para aprender a sintaxe dos comandos e os fóruns para esclarecer eventuais dúvidas. A comunidade de desenvolvedores auxilia na obtenção de materiais para estudo.

3.6 – Simulações em computador

Para a computação, uma simulação é a representação de um contexto real através da formalização de modelos que imitam uma atividade real. Através do projeto de um modelo computacional correspondente à atividade que se pretende simular, são conduzidos experimentos cuja finalidade é compreender seu comportamento e/ou avaliar estratégias para sua operação [97]. Dessa maneira é possível formular hipóteses e teorias através dos resultados observados; ou ainda prever um comportamento futuro e seus efeitos produzidos através de ajustes no sistema ou método empregado.

Os métodos para simulação em computador para calcular experimentos que envolvem a física e a interação de íons num meio são desenvolvidos desde a década de 1960. Entretanto, se

avaliarmos o surgimento dos computadores, o foco militar precede a finalidade comercial anterior à primeira guerra mundial, com simulações de balística, por exemplo.

Observando os estudos sobre a teoria de freamento eletrônico de íons, a ideia básica era acompanhar o movimento dos íons num meio simulando as colisões e o comportamento de atrito no material. Os métodos convencionais utilizados para calcular o alcance desses íons eram aproximações de colisões binárias (BCA, *binary collision approximation*) [98], onde os cálculos consideravam as sucessivas colisões individuais dos íons em relação à matéria (alvo) e os desvios dos átomos, ou seja, uma maneira para avaliar de maneira eficiente a profundidade de penetração de íons em sólidos.

Dentre os programa que utilizam o método BCA temos o TRIM/SRIM (*Transport of Ions in Matter*, posteriormente chamado de *Stopping and Range of Ions in Matter*), baseado no modelo ZBL de freamento eletrônico e potencial interatômico [1,99,100]. Este programa tem uma interface simples, que o popularizou, entretanto não considera a estrutura cristalina do material, limitando sua aplicação em várias situações.

Como a interação com cristais não é trivial, o modelo de MARLOWE [101] tenta tornar o resultado obtido o melhor possível, ampliando a confiança do BCA através da inclusão de colisões múltiplas nos cálculos efetuados. Outra maneira simples para o cálculo de múltiplas colisões atômicas é obtido através da simulação de dinâmica molecular (MD, molecular dynamics) resolvendo as equações de movimento numericamente. Tais métodos foram aperfeiçoados para otimizar os cálculos necessários [102,103].

3.6.1 – O Processo de simulação

O objetivo da simulação é auxiliar na decisão final da solução de um problema. Para atingir essa meta adequadamente é necessário associar boas técnicas de resolução de problemas com a experiência na área de engenharia de software.

Em 1990, Pegden [97] propôs os doze passos a seguir para solucionar qualquer estudo que envolva simulação:

1- Definição do problema: determinar os objetivos do estudo, ou seja, especificar qual o motivo de estudar tal problema e quais questões serão solucionadas.

2- Planejamento do projeto: garantir a disponibilidade adequada dos recursos necessários, sejam eles humanos, equipamentos (hardware) ou programas (software).
3- Definição do sistema: determinar as limitações do modelo e verificar como o sistema opera.
4- Formulação do modelo conceitual: determinar um modelo na forma de pseudocódigo ou pré-gráfico (diagrama de blocos) para definir componentes, variáveis descritivas e interações lógicas do sistema.
5- Projeto experimental: selecionar as medidas de desempenho que serão usadas, os fatores que sofrerão variações e quais os resultados obtidos com essas variações.
6- Preparação dos dados de entrada: identificar e preparar os dados de entrada.
7- Tradução do modelo: formulação do modelo numa linguagem de programação adequada.
8- Verificação e validação: confirmação de que o modelo funciona conforme o projetado pelo analista (depuração) e que os resultados são válidos e representativos conforme àqueles obtidos pelo modelo real (validação para animação).
9- Projeto experimental final: planejar um experimento que produza as informações desejadas e determinar um teste específico que será executado.
10- Experimentação: realizar a simulação para obter os dados desejados e realizar análise de sensibilidade.
11- Análise e interpretação: verificar os erros e interferências detectados nos resultados da simulação para compreender os resultados e corrigir variações.
12- Implementação e documentação: concluir o trabalho disponibilizando os resultados para uso, registrar recomendações e documentar o modelo utilizado e seu uso.

O passo que requer maior dedicação e tempo para elaboração do modelo empregado no projeto é o número 7, porque está intimamente relacionado ao tempo gasto para conhecer o problema, desenvolver o desenho do modelo e executar o experimento. Os cientistas da computação têm empenhado esforços para aumentar a velocidade das simulações através do desenvolvimento de software mais eficientes e métodos gerenciais para melhorar o rendimento dos softwares. Uma das melhoras estratégias é a “40-20-40”, na qual 40% dos esforço e do tempo devem ser aplicados aos passos de 1 a 6, 20% ao passo 7 e 40% aos passos de 8 a 12 [104,105].

3.6.2 – A animação dos modelos de simulação

A animação na computação desponta como uma ferramenta de grande importância na aplicação de modelos de simulação de sistemas do mundo real. A animação permite a visualização do modelo desejado na tela do computador. O operador visa à execução do modelo com animação e isso fornece importantes informações do desempenho do modelo que não seriam possíveis apenas como uma análise estatística de saída [97].

3.6.2.1 – A animação na avaliação do modelo

A animação tem utilidade nas várias fases do modelo de simulação, as quais serão avaliadas a seguir:

Verificação do modelo

Verificação do modelo é um processo onde é avaliado se o programa de simulação executado corresponde ao planejado. Nessa etapa não é analisado se o modelo representa o sistema real, apenas determina se há erros lógicos.

Nesta fase são detectados diversos tipos de erros cometidos no desenvolvimento do modelo, tais como a falta de criação de variáveis, falta de inicialização de valores para as variáveis, liberação de recursos alocados, entre outros erros. A descoberta destes erros, que levaria a conclusões equivocadas sobre o desempenho do sistema, acaba por evitar a perda de muito tempo com uma simulação ineficiente e custosa [106].

Nessa fase de avaliação de modelo, a animação talvez seja a forma mais eficiente de verificar erros através da observação direta dos erros lógicos.

A visualização dos erros na execução do modelo ajuda a compreender melhor o processo e evitar novos erros.

Validação do modelo

A validação do modelo é o processo no qual é analisado se os dados obtidos correspondem ao sistema real, de tal forma que possa responder questões específicas a respeito da operação do sistema.

Através da animação, o analista verifica a operação do modelo, os impactos e a interações das simplificações feitas durante a fase de desenvolvimento do modelo. Isto serve também como elo entre o analista e o usuário. O usuário, que conhece o funcionamento do sistema, auxilia o analista através do fornecimento de informações sobre a operação e como é o funcionamento do sistema real. Isto pode ser avaliado verificando como a informação foi modelada e se foi interpretada corretamente.

Interações dinâmicas

Com a interpretação estatística dos resultados, a obtenção de conclusões precipitadas é evitada. Embora a animação não substitua os métodos estatísticos de avaliação, pode melhorar substancialmente essa interpretação através da visualização do comportamento do sistema durante a análise da simulação.

A animação fornece informação sobre os processos que estão interagindo e seu desempenho, permitindo a proposição de melhorias no modelo antes de implementá-lo no sistema real. Essas melhorias frequentemente passam desapercebidas sob o ponto de vista de análises estatísticas ou gráficos de pontos das variáveis, mas tornam-se óbvios nas animações.

Apresentações dos resultados do modelo

O sucesso de um projeto de simulação é medido pelo impacto no desenho e operação do sistema real. Os gerentes do sistema real devem conhecer e usar o resultado do modelo para que a simulação alcance seus objetivos.

Assim, uma importante etapa do projeto é a apresentação do modelo e seus resultados aos gestores. Segundo Johnson e Poorte [107], o objetivo básico do processo de modelagem é prover informações confiáveis aos gestores que tomarão decisões através da aceitação ou rejeição das informações fornecidas pelos modeladores. A comunicação efetiva entre gerentes e modeladores

é de vital importância para a credibilidade do modelo e a animação pode ser um facilitador da negociação da proposta de solução.

A animação não tem como única função vender a utilidade do projeto de modelagem, ela ajuda a vender a simulação como uma aproximação da realidade [97].

3.6.2.2 – Limitações de animação na interpretação dos resultados

A animação oferece problemas quando usada como método primário de interpretação de resultados da simulação antes de ser usada como método da análise estatística. Isto pode ter dois enfoques diferentes: a velocidade de execução e a tendência sobre o tempo.

A velocidade de execução

A velocidade de execução de um modelo animado de simulação é mostrada em várias magnitudes de intensidade para que sejam compreensíveis ao observador. Por exemplo, em uma rodada de simulação para processar mil peças utilizando um determinado modelo demoraria dois minutos no computador, enquanto o processamento real levaria um dia todo de trabalho. A animação é assistida num curto período de tempo, representando apenas uma fração do tempo global, assim, é arriscado obter conclusões sobre a aleatoriedade do processo baseado numa pequena amostra simulada. O desempenho do sistema real não corresponde diretamente às ocorrências no tempo da animação.

A tendência sobre o tempo

Mesmo que a animação seja muito eficiente em mostrar interações simultâneas ela é pouco eficiente em mostrar a tendência de uma variável específica sob o tempo.

Durante uma animação, diversas interações instantâneas são vistas, e elas são transformadas em variáveis. Para distinguir a tendência sobre o tempo, gráficos convencionais devem ser incluídos como parte da animação mostrando as principais variáveis. Assim, o analista pode diferenciar as interações entre variáveis e o desempenho de variáveis específicas sobre o tempo.

3.7 – Métodos Numéricos

São metodologias utilizadas para obter soluções numéricas de forma aproximada para problemas cujos modelos são complexos e envolve muitos cálculos. O computador ou calculadoras científicas podem auxiliar na tarefa de solução, entretanto, os resultados obtidos são chamados de aproximados em virtude da limitação da representação da resposta que pode sofrer arredondamentos ou erros de interação seqüencial.

3.7.1 – BCA (*Binary Collision Approximation*)

A aproximação por colisão binária é um método utilizado para simulações em física onde é calculado o comportamento da irradiação de íons em um sólido, a profundidade de penetração do íon e a produção de defeitos no material. O método calcula a perda de energia sofrida pelo íon ao penetrar no material através de colisões binárias, onde é estudado o poder de freamento do íon em relação aos núcleos dos átomos do material. Os parâmetros de impacto dividem o método em duas variações principais de código: "Monte-Carlo BCA" e "Crystal-BCA".

O método "Monte-Carlo BCA" consiste em aproximar a distância e parâmetro de impacto da colisão seguinte escolhendo o próximo átomo aleatoriamente para uma distribuição de probabilidade que depende apenas da densidade atômica do material. Esta abordagem essencialmente simula a passagem de íons em material totalmente amorfo.

O método "Crystal-BCA" considera a iteração de íons em material cristalino, o que é mais difícil de calcular porque a posição dos íons na cadeia cristalina pode variar conforme a trajetória hiperbólica de Rutherford [20] ou sofrer tunelamento (figura 2.7).

3.7.2 – Método de Monte Carlo

O método de Monte Carlo (MMC) é um método estatístico utilizado em simulações estocásticas (que envolve a geração de números reais distribuídos aleatoriamente) com diversas aplicações em áreas como biologia, engenharias, física, matemática, química, entre outras. O

objetivo é explorar as possibilidades de um dado fenômeno cujo comportamento possa ser quantificado matematicamente.

A simulação de Monte Carlo é um mecanismo que permite a determinação da função de distribuição de probabilidade de resultados de um modelo baseada na distribuição de parâmetros do modelo. Isto envolve avaliar o mesmo modelo centenas ou milhares de vezes, gerando a cada vez um número aleatórios para cada parâmetro descrito como uma distribuição. Cada avaliação calcula um valor diferente para cada resultado do modelo. Após diversas execuções, estes resultados podem ser organizados em histogramas de freqüências, que são representações gráficas do número de vezes que cada valor foi resultante em uma simulação. Normalizado, de forma que a área sob a curva se reduza a um, o histograma pode ser visto como a função de distribuição de probabilidade dos resultados do modelo.

3.8 – Simulações

No estudo sobre as possíveis implementações para soluções numéricas que empregam a aproximação por colisão binária (BCA) e o método de Monte Carlo (MMC) utilizando CPU ou GPGPU foram avaliados programas com código-fonte aberto ou de uso gratuito. Os resultados relevantes serão apresentados no capítulo 4. Aqui são apresentados os programa que se destacaram e suas funcionalidades.

3.8.1 – TRIM/SRIM

O programa para computador SRIM (*Stopping and Range of Íons in Matter*) é uma interface gráfica atualizada do programa TRIM (*TRansport of Ions in Matter)* desenvolvida para integrar os códigos criados por James F. Ziegler et al [1]. Estes programas utilizam os métodos BCA e ZBL apresentados no segundo capítulo para realizar os cálculos relacionados à interação de íons com a matéria. Os avanços da computação e adequações efetuadas pelo desenvolvedor permitiram a contínua atualização e utilização do pacote de programas nas últimas três décadas (figura 3.24).

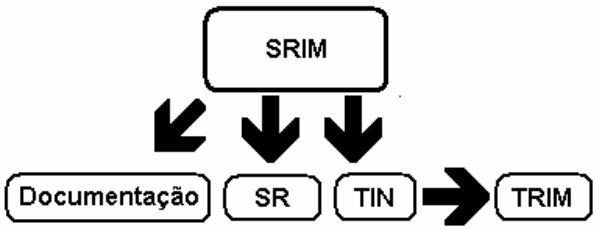
O principal interesse neste programa para uso no CCS/UNICAMP foi a possibilidade de avaliar diversos cenários antes de efetuar os experimentos com nanoestruturas. Além do auxílio com os cálculos, os resultados obtidos no programa permitem avaliar os gráficos e perfis de dopagem (íons primários e secundários, principalmente) que auxiliam na compreensão do estudo e características dos procedimentos.

A importância da simulação neste caso é evitar o desperdício de recursos através de testes empíricos com diversas amostras para compor um documento com os resultados obtidos. Com simulações é possível prever algumas condições antecipando decisões para o projeto de pesquisa e também validando os modelos simulados, criando modelos (*templates*) para o laboratório do CCS/UNICAMP. Esta otimização é muito importante para um ambiente multiusuário, além da elaboração de procedimentos para cursos e treinamentos da universidade.

Os parâmetros necessários para elaborar um perfil mínimo para iniciar uma simulação são o tipo de íon que será lançado na amostra, a energia deste íon (que pode variar entre 10 eV a 2 GeV), o ângulo de incidência na amostra (alvo), o tipo de amostra (dados sobre a composição da amostra, que pode ser composta por uma única camada de um dado material ou ainda ser composta por algumas camadas de materiais distintos) e quais os resultados desejados (*output plots*).

Os resultados contemplam: particionamento de energia entre as perdas nuclear e eletrônica; a taxa de deposição de íons; concentração de vacâncias; a taxa de re-deposição, ionização, produção de fônons na amostra; distribuição tridimensional dos íons no sólido e seus parâmetros, tais como profundidade da penetração, sua extensão ao longo do feixe e íons (straggle) e todas as cascatas de átomos na amostra.

Uma avaliação sobre o funcionamento do programa, variações entre as versões testadas e utilizadas está no anexo B.




```
[+] SRIM
├─ Documentação
├─ Gerador de tabelas Stopping&Range
└─ configuração Trim Input
   └ simulador TRIM
```

Figura 3.24: Associação modular das funcionalidades do SRIM

3.8.2 – MCML

O programa MCML (*Monte Carlo for Multi Layered media*) [2] simula a propagação de um fóton em um dado meio, através da probabilidade de distribuição do fóton e eventualmente seu espalhamento, quando ocorre. Esta simulação pode avaliar diversas grandezas físicas simultaneamente e há demanda de muito tempo computacional para obter os resultados.

Nesta simulação, o número de fótons informado depende da precisão e resolução espacial desejadas. Isso varia conforme o modelo utilizado. Este programa não trata o fóton como um fenômeno de onda e ignora as características de fase e polarização. São baseadas em propriedades ópticas macroscópicas assumindo extensão uniforme em pequenas unidades.

A verificação deste código-fonte neste trabalho tem como interesse analisar o desempenho dos códigos para CPU e GPGPU, em relação ao desenvolvimento de programação específica para avaliação dos resultados obtidos com simulações. Tais resultados serão apresentados no capítulo 4.

O propósito do estudo e avaliação deste programa foi o acesso ao código-fonte para verificar como ajustes ou modificações podem influenciar na redução do tempo necessário para uma simulação. Isto envolve execução dos algoritmos para a interação de partículas e uso do método de Monte Carlo para soluções computacionais.

Capítulo 4

4 – Resultados e Discussões

Dentre as pesquisas e experimentos realizados com CPU e GPGPU, a seguir são apresentados os resultados obtidos com: matrizes esparsas e o método de Monte Carlo, empregado em simulações TRIM/SRIM e avaliação do experimento prático (4.2) e análise de desempenho entre TRIM/SRIM e MCML (4.3); ressaltando as conquistas obtidas no campo da computação com a redução de tempo necessário para os cálculos, além do uso adequado de virtualização para aprimorar resultados com simulações.

Considerando válido o modelo utilizado no programa TRIM/SRIM, foram observados os resultados dos experimentos efetuados com o FIB/SEM no CCS/UNICAMP. Isto permitiu avaliar a possibilidade de utilizar o programa em simulações computacionais para redução de custos com laboratório.

Posteriormente, foram estudadas diversas combinações possíveis para simulação, utilizando máquinas virtuais em computadores com vários núcleos e a avaliação do desempenho do Método de Monte Carlo (MMC) em CPU e GPGPU (para TRIM/SRIM e MCML). Foi verificada a possibilidade de atualização do programa TRIM/SRIM para suportar melhorias significativas para redução do tempo de processamento.

A definição adequada dos parâmetros de entrada para o programa MCML pode melhorar os resultados obtidos em CPU. O desempenho do programa adaptado para uso em GPGPU comprovou que a melhoria no código, que utiliza o método de Monte Carlo, reduziu drasticamente o tempo necessário para simulações, de dias para minutos.

4.1 – Pesquisas realizadas e experiências com GPGPU

No campo de análise numérica, uma matriz esparsa é basicamente uma tabela composta principalmente por zeros [108]. Em contrapartida, quando há poucos elementos nulos na tabela a matriz é denominada densa. Analogamente a dispersão numérica de elementos numa matriz pode ser tratada como base para cálculos em diversos campos da ciência ou engenharias para

verificação da similaridade de contaminantes entre materiais. Através da computação é possível resolver grandes estruturas de dados que consomem grandes quantidades de memória e recursos computacionais para verificar os dados nesse tipo de ambiente.

A aplicação de matrizes esparsas em problemas de engenharia e física, por exemplo, auxilia no método das malhas para resolução de circuitos elétricos ou sistemas de equações lineares. Sua implementação em computador é feita através de um conjunto de listas ligadas que apontam para elementos não nulos, de maneira que os elementos válidos reduzam o consumo de memória, economizando recursos.

Nossos estudos iniciaram com avaliação da implementação do método de gradiente conjugado para matriz esparsa utilizando GPU [109], onde os métodos tradicionais para resolução numérica direta, tais como eliminação Gauss-Jordan [110,111], decomposição LU [110,111] e o método Cramer [112] são ineficientes para matrizes grandes, sendo necessário utilizar métodos iterativos como Gauss-Seidel [110,111], Jacobi [110,111], gradiente conjugado [110,111], entre outros, cujas sucessivas aproximações convergem para a solução desejada. O método ELLPACK-R [113,114] foi utilizado para armazenamento das informações.

Baseado na revisão desses métodos numéricos foi escrito um programa em linguagem C, desenvolvido inicialmente para CPU, que foi revisado diversas vezes para produzir um código eficiente capaz de calcular um grande volume de dados em CPU, utilizando a memória principal do computador, no menor tempo possível.

Posteriormente esse programa foi reescrito para funcionar em CPU e GPGPU, onde a tradução direta do código-fonte manteve o bom desempenho obtido, mas sem redução de tempo de processamento desejada. Inicialmente observou-se que há grande demora com atrasos comuns para uso da GPGPU que consistem em tempo de cópia dos dados entre a memória principal do computador para a memória da placa de vídeo (para carga do programa) e posteriormente, após o processamento de informações pela GPGPU, do retorno dos dados da memória da placa de vídeo para memória principal do sistema, ao término da atividade. Somados os tempos entre as transferências de dados (*host to device e device to host*) e o processamento da GPGPU, o resultado inicial da CPU foi similar ao da GPGPU. Fizemos então, ajustes nos programas para CPU e GPGPU com o objetivo de reduzir o tempo de processamento, onde dentre diversas versões elaboradas foram selecionadas quatro possibilidades de variação do código-fonte do programa, que foram executadas diversas vezes (ao menos vinte e cinco vezes para cada variação

do programa) para verificar o tempo necessário na solução do problema com matrizes esparsas. Dessa avaliação foi elaborada a tabela 4.2 com a média de todos os resultados obtidos. Esses testes definiram os parâmetros para um programa CUDA eficaz, que ficou cerca de três vezes mais rápido que aquele executado exclusivamente em CPU (tempo total global entre todas as transações).

O passo seguinte foi produzir um código para funcionar completamente em GPGPU onde obtivemos como resultado um código quinze mil vezes mais veloz que aquele executado exclusivamente em CPU (tempo de execução). Para a resolução de problemas em GPGPU, encontramos como limitação o domínio a ser usado, cuja dimensão foi de 2048 (linhas e colunas), devido ao limite da memória disponível na placa gráfica NVIDIA modelo GT8800. As características deste modelo podem ser observadas na tabela 4.1. A seguir, a figura 4.1 ilustra o algoritmo utilizado no artigo [109].

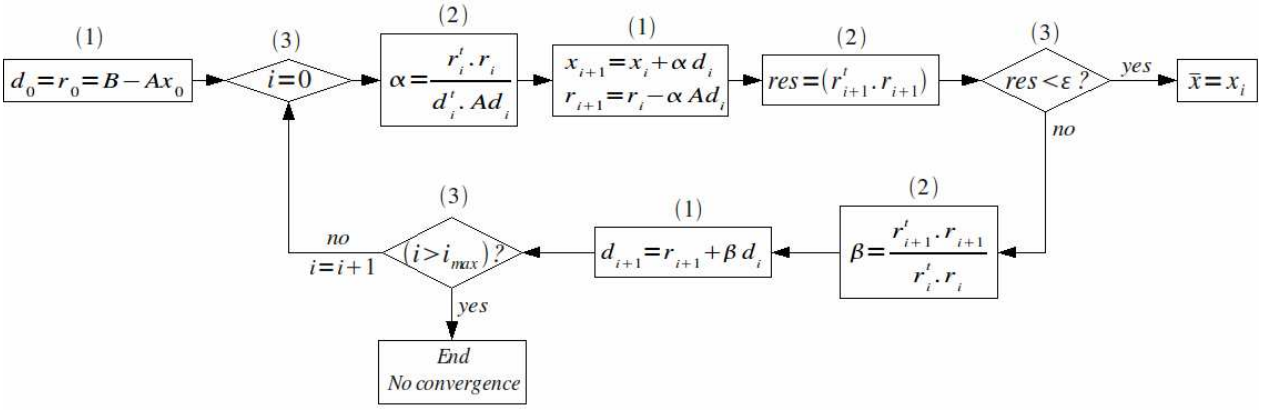


Figura 4.1: Algoritmo do método de gradiente conjugado

Durante o aprendizado foram testadas diversas placas gráficas de uso comum, comercializadas para utilização com jogos em PCs. Esses produtos possuem preços mais acessíveis para aquisição, comparados aos modelos específicos para supercomputação. O custo menor das placas gráficas para jogos permitiu a avaliação de códigos CUDA entre alguns modelos de placas gráficas, onde as principais características estão na tabela 4.1.

Foi possível comparar também as características dos pacotes de desenvolvimento da NVIDIA para programação CUDA (SDK, *Software Development Kit*). Tais ferramentas se tornaram mais fáceis de estudar ao passar do tempo. A maneira adequada de configurar tais dispositivos em sistema operacional GNU/Linux encontram-se no apêndice A.

A experiência adquirida com os diversos pacotes de desenvolvimento estudados possibilitou a migração de códigos-fonte criados em versões mais antigas para a versão mais atual, aprimorando constantemente o material pesquisado.

Tabela 4.1: Placas gráficas utilizadas nas simulações e características relevantes.

Características	**GT 8800**	**GT 9800**	**GT 220**	**GT 520**
Processadores de fluxo (CUDA Cores)	112	112	48	48
Clock do núcleo (MHz)	600	600	625	810
Clock do shader/tester (MHz)	1500	1500	1360	1620
Clock da memória (MHz)	900	900	790	900
Quantidade de memória (MB)	512	512	1024	1024
Interface de memória (bits)	256	256	128	64
Largura de banda da memória (GB/seg)	57.6	57.6	25.3	14.4
Taxa de preenchimento de textura (bilhões/seg)	33.6	33.6	N/D	6.5

*N/D significa não definido.

A avaliação dos resultados obtidos mostra que o desempenho do algoritmo para GPGPU é melhor quando o domínio for maior, ou seja, quando “n” for trinta e dois ou mais, o programa para GPGPU é muito mais eficiente que o mesmo código executado em CPU (tabela 4.2). Dessa forma o tempo de execução em GPGPU pode ser até dezessete mil vezes mais rápido que em CPU e o tempo global de processamento pode ser até quarenta e uma vezes melhor que o tempo necessário em CPU.

A tabela 4.2 utiliza as seguintes siglas para representação dos dados:

GC = Geração de dados em CPU (tempo em milisegundos);
GG = Geração de dados em GPGPU (tempo em milisegundos);
EC = Execução dos cálculos em CPU (tempo em milisegundos);
EG = Execução dos cálculso em GPGPU (tempo em milisegundos);
G2C = Transição de dados entre GPGPU e CPU (e vice-versa; tempo em milisegundos);

Tabela 4.2: Exemplo dos resultados obtidos com a variação três do programa (C3).

Domínio (n)	Matriz (n^2 x n^2)	GC	GG	EC	EG	G2C	Total EC	Total EG	Ganho
		Tempos expressos em milisegundos							
4	16x16	0,0080	0,0275	0,0010	0,0030	0,0200	0,0090	0,0505	0,1782
8	64x64	0,0120	0,0275	0,0010	0,0030	0,0220	0,0130	0,0525	0,2476
16	256x256	0,0220	0,0275	0,0020	0,0030	0,0250	0,0240	0,0555	0,4325
32	1024x1024	0,0840	0,0275	0,0090	0,0030	0,0270	0,0930	0,0575	1,6174
64	4096x4096	0,3700	0,0275	0,0500	0,0030	0,0360	0,4200	0,0665	6,3158
128	16384x16384	1,4790	0,0275	0,1620	0,0030	0,0920	1,6410	0,1225	13,3959
256	65536x65536	6,7010	0,0275	0,9800	0,0030	0,3450	7,6810	0,3755	20,4554
512	262144x262144	27,9620	0,0275	4,7890	0,0030	1,2780	40,4320	1,3085	30,8995
1024	1048576x1048576	113,2870	0,0275	20,1610	0,0030	3,6680	113,4480	3,6985	30,6741
2048	4194304x4194304	464,4150	0,0275	79,5640	0,0030	13,0971	543,9790	13,1276	41,4378

Constatou-se também que durante a execução de uma simulação em GPGPU, quando for possível ocupar o hardware ao máximo (*active threads*, ou seja, subprocessos), melhores serão os resultados em questão de obteção de respostas no menor tempo possível [115], independente da versão do CUDA-SDK utilizado. Isto significa que quanto melhor for o balanceamento de carga, ou seja, subdivisão de uma tarefa (processo) em partes menores (subprocessos) que possam aproveitar ao máximo o poder de processamento de uma GPGPU, mais rápido o resultado será obtido. Esta característica da arquitetura da placa gráfica pode ser revista na figura 3.9, comparando analogamente que enquanto um processo utiliza toda a CPU, esse mesmo processo seria realizado por oito subprocessos na GPU. Como cada uma dessas oito linhas possui a mesma estrutura que uma CPU com diversas unidades lógico-aritméticas (ALU), o poder de cálculo é potencializado.

Para a aplicação de GPGPU, otimizando a utilização do método de Monte Carlo, em processos de nanoengenharia, inicialmente foram verificadas quais os programas gratuitos ou de código-fonte aberto estão à disposição dos pesquisadores.

4.2 – Resultados obtidos com simulações TRIM/SRIM

Iniciamos o estudo de diversas hipóteses sobre a interação de partículas com a matéria utilizando o programa para computador TRIM/SRIM [1] como ferramenta de apoio. O modelo de interação entre partículas empregado pelo programa utilizado nas simulações foi considerado válido em virtude de seu amplo uso em pesquisas e não foi reavaliado neste trabalho.

Este estudo possibilitou a apresentação de um pôster na conferência internacional IBA (*Ion Beam Analysis*) em 2011 [116] avaliando a qualidade dos resultados experimentais obtidos com as simulações de dopagem de substrato (grafeno) por íons de gálio. Verificamos inicialmente o comportamento dos íons primários em diversas simulações, com ajuste do ângulo de incidência e doses diferentes. Uma vez determinada a melhor maneira de dopagem do material, baseada nos resultados das simulações computacionais, foi realizado o experimento em laboratório. Os resultados obtidos com as amostras foram comparados em relação às simulações em computador. Notou-se nas análises preliminares a possibilidade de interação de feixe de íons de gálio em substrato composto por membrana multi-camada de grafeno.

O aprimoramento do trabalho apresentado no IBA [117] empregou membranas mais finas (10-20nm) e melhor resolução do feixe de íons (20-30nm), além da deposição de platina. Considerando o modelo empregado no programa TRIM/SRIM válido, observamos que os resultados dos experimentos produzidos pelo FIB/SEM no CCS/UNICAMP apresentaram perfis semelhantes àqueles observados nas simulações. Isto pode ser notado através da amorfização da região destacada na figura 4.5(d). Dessa forma, há possibilidade de aprimorar os modelos (*templates*) do FIB/SEM através da colaboração obtida com simulações computacionais.

Todos os resultados numéricos e figuras obtidas com as simulações realizadas com TRIM/SRIM encontram-se no apêndice B, onde se pode observar que para cada energia utilizada, o número total de íons não varia muito nos resultados (as dimensões da área processada/amorfizada) a partir de 10^4 íons. Foram efetuadas simulações somente com íons primários (cálculo rápido de colisões e distribuição de íons) e também considerando todas as interações (cálculo detalhado com colisões em cascata).

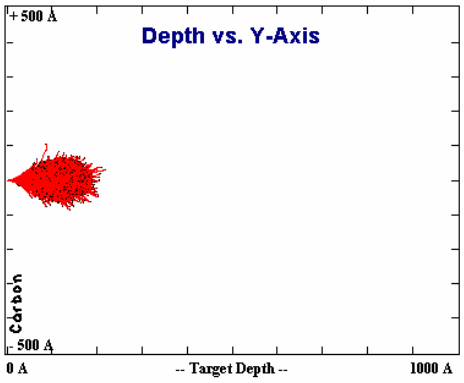
Para nanoestruturas, a escolha adequada do ângulo do feixe de íons (que determina a região afetada) é importante pois pode haver ruptura da amostra quando submetida a altas

energias e grande número total de íons incidentes (dano parcial pode ser observado comparando a membrana da figura 4.5 “a” e “c”).

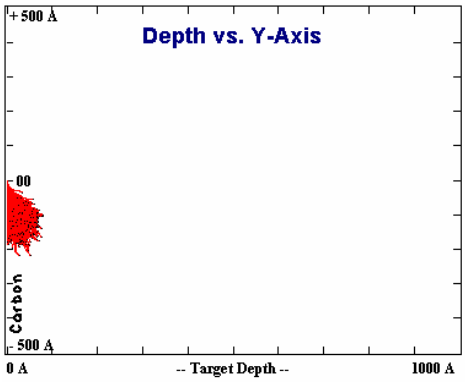
Alguns resultados de cálculos são apresentados na figura 4.2, onde ressaltamos que o diâmetro do feixe de Ga+ nestes cálculos é igual a 0 nm (o diâmetro mínimo do feixe no equipamento FIB é igual a 7 nm para energia 30 keV) , o que permite visualizar melhor a expansão lateral do feixe. A figura 4.2 mostra o efeito da energia de íons sobre o tamanho da área amorfizada por íons primários, para substrato de C, energias de 10, 20 e 30 keV, ângulos de incidência de 0° e 88°. Como pode ser observado, a dimensão da área aumenta de maneira sub-linear com a energia (aumento em duas vezes para energias crescendo de 10 a 30 keV), para os dois ângulos (a incidência normal e rasante).

A figura 4.3 mostra o efeito do número de íons (a dose) sobre a área amorfizada no substrato por íons primários, para substrato de C, energias de 30 keV, ângulos de incidência de 0° , 80° e 88°. Como pode ser observado, acontece expansão significativa da área (principalmente, na direção lateral) para doses de 10 a 10^3, porem, o aumento para a dose de 10^5 já não é tão significativa. Podemos concluir que para avaliação do tamanho da área amorfizada, é suficiente fazer cálculos para doses 10^4 ou 10^5, no máximo. Para ângulos de incidência rasantes (80° a 88°), o tamanho da área amorfizada (e volume do material removido) diminui comparando com a incidência normal. A diminuição da área amorfizada/removida é mais notável para ângulo 88° que é frequentemente utilizando para o polimento das paredes de estruturas por FIB, procurando diminuir a amorfização do material na direção normal.

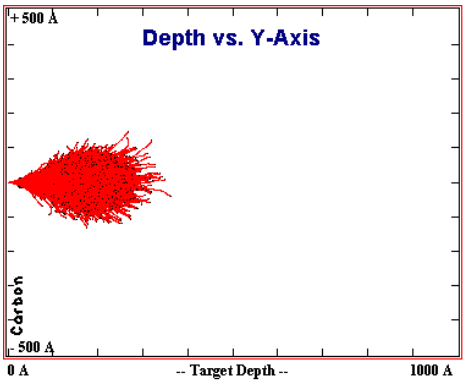
A figura 4.4 mostra o efeito da inclusão dos íons e átomos secundários (deslocados no substrato após as colisões com íons primários), produzidos em cascata de colisões. Como podemos observar, o numero de partículas atômicas secundarias é muito maior e a geração das partículas secundarias, mesmo com energias menores, resulta em um aumento forte da área processada/amorfizada, especialmente na região superficial (entrada) do substrato, o que determina eventualmente a resolução lateral de processamento. A resolução lateral de processamento, para material de C e energia de íons 30 keV, fica em torno de 60 nm, considerando a contribuição dos íons e átomos secundários, como pode ser visto na figura 4.4. Vale ressaltar que esta resolução mínima foi confirmada em nossos experimentos com folhas de grafeno de múltiplas camadas (espessura ~50 nm), conforme exposto na figura 4.5.



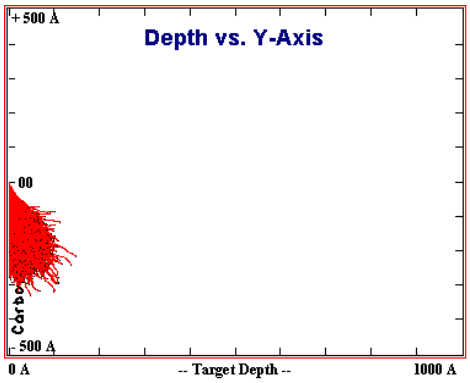
GaC 10 KeV; 0; 1E4



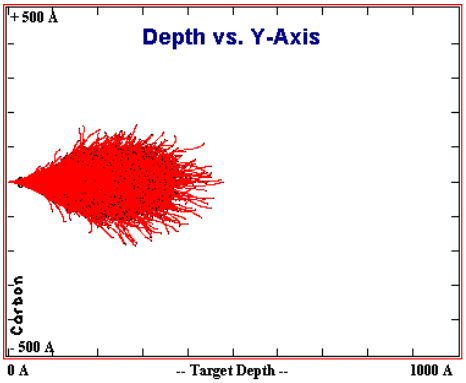
GaC 10 KeV; 88; 1E4



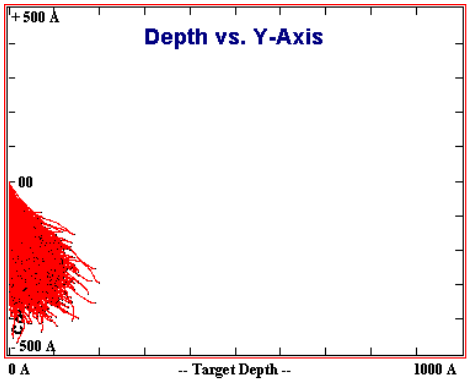
GaC 20 KeV; 0; 1E4



GaC 20 KeV; 88; 1E4



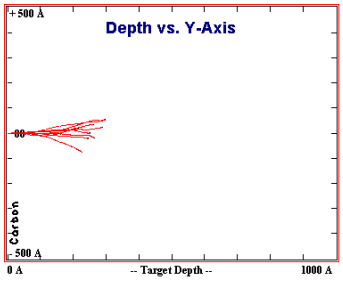
GaC 30 KeV; 0; 1E4



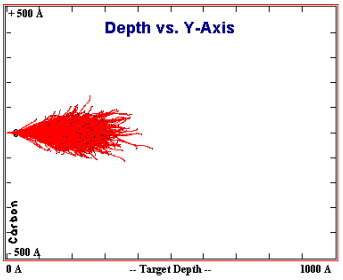
GaC 30 KeV; 88; 1E4

Figura 4.2: Cascata de íons primários dentro do alvo;

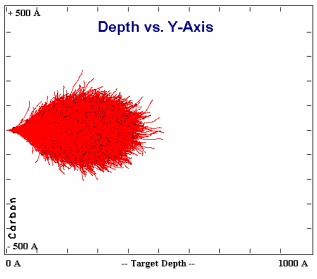
material – C, E_i = 10, 20, 30 keV; $\theta = 0°$ e $88°$.



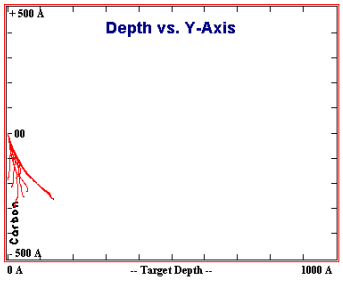
GaC 0; 1E1 QDC



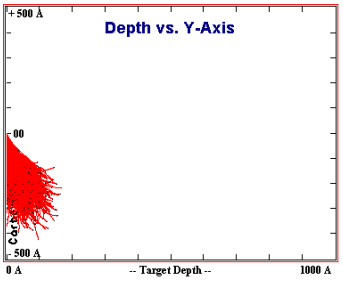
GaC 0; 1E3 QDC



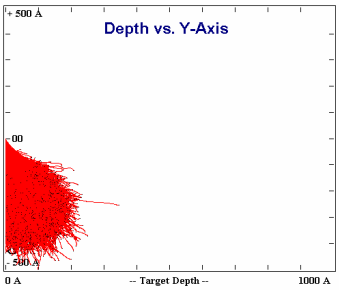
GaC 0; 1E5 QDC



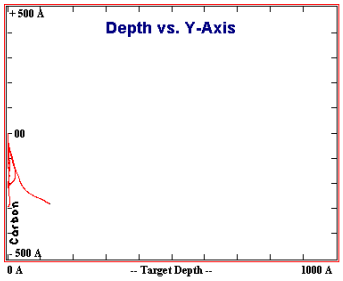
GaC 80; 1E1 QDC



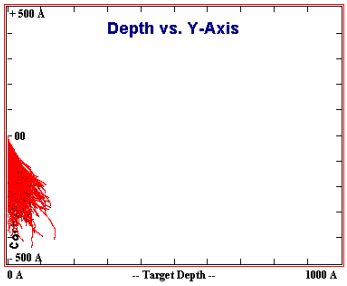
GaC 80; 1E3 QDC



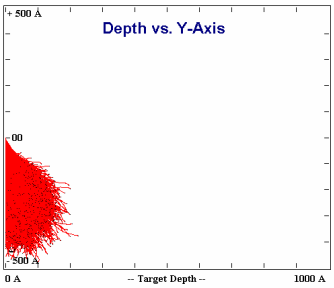
GaC 80; 1E5 QDC



GaC 88; 1E1 QDC



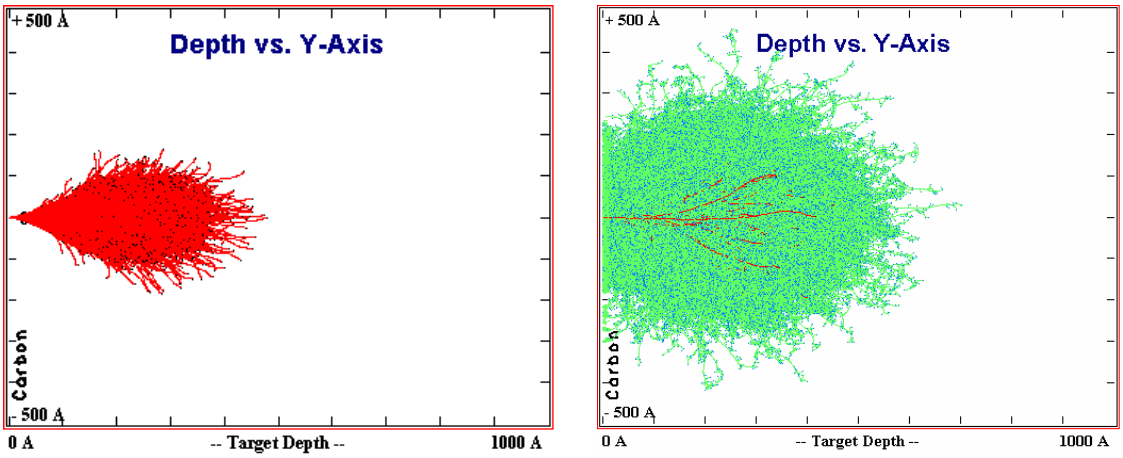
GaC 88; 1E3 QDC



GaC 88; 1E5 QDC

Figura 4.3: Cascata de íons primários dentro do alvo;

material – C, E_i = 30 keV; $\theta = 0°, 80°, 88°$.



GaC 30 KeV; 0; 1E4 QDC GaC 30 KeV; 0; 1E4 FDC

Figura 4.4: Cascata de íons somente primários (esquerda, as trajetórias em vermelho) e primários junto com secundários (direita, as trajetórias em verde). Material – C, E_i = 30 keV; 10^4 íons, θ = 0° .

Na figura 4.5, a largura mínima de corte por FIB é de 50 nm, como é indicado por flechas e está de acordo com os calculos (figura 4.4).

A figura 4.6 mostra o efeito do material sobre a cascata de íons e a expansão da área amorfizada, para os alvos de grafite (C, massa atômica MA=12), silício (Si, MA=28) e platina (Pt, MA=190), sendo MA =69 para Ga. Podemos observar certas diferenças para materiais com átomos mais leves (C, Si) e pesados como Pt. Para o alvo de Pt, a propagação do feixe dentro do material é menor, com rápida expansão lateral, devido a interação mais forte de íons de Ga com átomos de Pt. Já para alvos de C e Si, a interação é menos intensa e a propagação é maior na direção longitudinal, resultando em uma forma especifica (alongada, lembrando a gota de liquido) da área amorfizada.

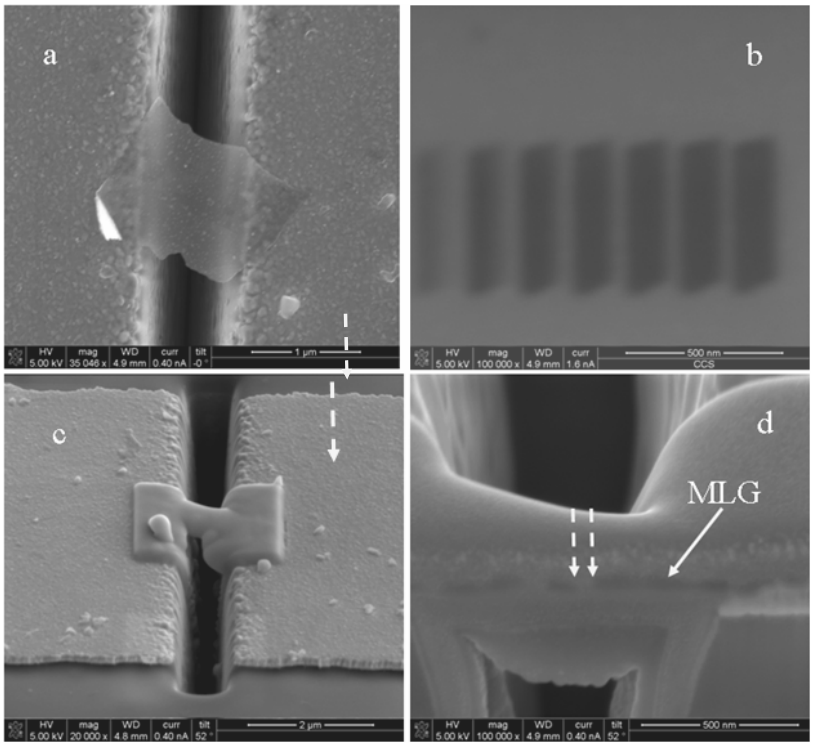
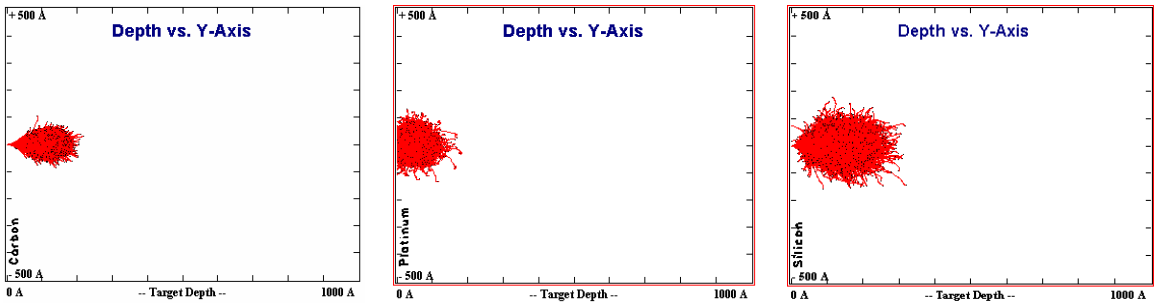
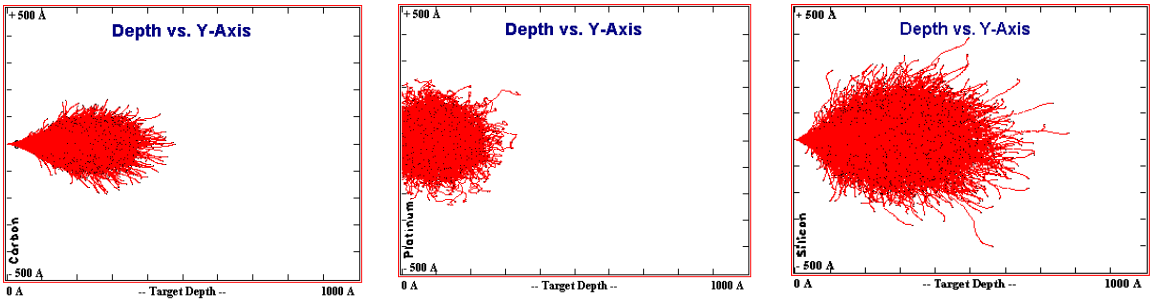


Figura 4.5: Imagens do microscópio eletrônico de varredura (MEV) mostrando uma folha de grafeno depositada por di-eletroforese sobre os eletrodos de tungstênio (a), superfície da folha de grafeno depois de irradiação por feixe de íons com doses aumentando de 0.4 a 3.1×10^{17} ions/cm^2, da esquerda para a direita (b), da folha coberta com camada protetora de Pt-C depositada por feixe de eletrons (c), a vista lateral da folha depois de corte (milling) por FIB (d). Imagens (a) e (b) – vista de cima, (c) e (d) – ângulo 53° . Na imagem "c" a flecha destaca a camada protetora (mudança de cor em relação à imagem "a"); na imagem "d", flechas indicam a região onde ocorreu amorfização (seta tracejada) e a membrana (MLG, *Multi Layer Graphene*).

A figura 4.7 mostra o efeito de colisões em cascata (íons/átomos secundários) e do ângulo de incidência, para dois materiais, C e Pt. O alargamento da área amorfizada é mais visível, me todos os casos, para C, enquanto a dimensão lateral é menor para Pt.



GaC 10 KeV; 0; 1E4 QDC GaPt 10 KeV; 0; 1E4 QDC GaSi 10 KeV; 0; 1E4 QDC



GaC 30 KeV; 0 1E4 QDC GaPt 30 KeV; 0; 1E4 QDC GaSi 30 KeV; 0; 1E4 QDC

Figura 4.6: Cascata de íons primários dentro do alvo, material – C, Pt e Si, E_i = 10 e 30 keV; 10^4 íons, $\theta = 0°$.

As figuras 4.8, 4.9 e 4.10 demonstram a propagação do feixe de íons dentro de membranas com espessura variada (1000, 300, 100 e 50 A°) , para materiais - C, Pt e Si, com energias de 10 e 30 keV. Para espessuras menores de 100 A (10 nm) ou 300 A (30 nm), dependendo da energia de íons (10 e 30 keV, respectivamente), podemos observar a diminuição da área amorfizada na direção lateral, resultando em melhoria da resolução do processamento. A largura da área diminui para :

~13 e 7 nm para 10 keV (membrana 10 / 5 nm), material C,

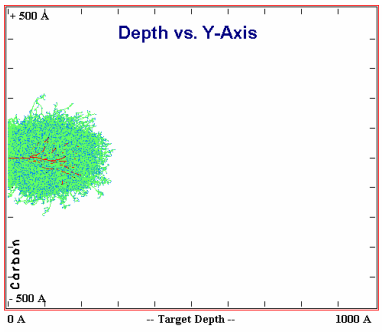
~12 e 6 nm para 30 keV (membrana 10 / 5 nm), material C,

~21 e 16 nm para 10 keV (membrana 10 / 5 nm), material Pt,

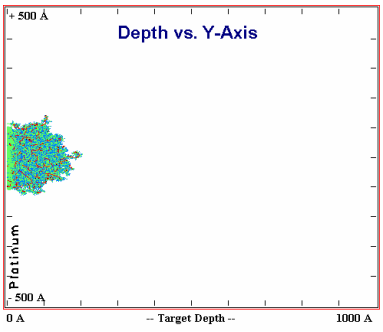
~30 e 22 nm para 30 keV (membrana 10 / 5 nm), material Pt,

~23 e 14 nm para 10 keV (membrana 10 / 5 nm), material Si,

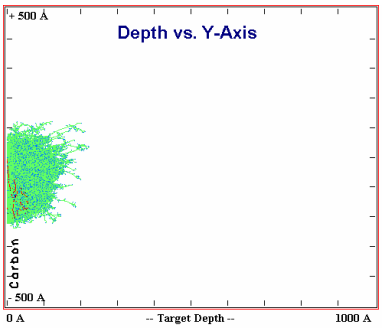
~17 e 8 nm para 30 keV (membrana 10 / 5 nm), material Si.



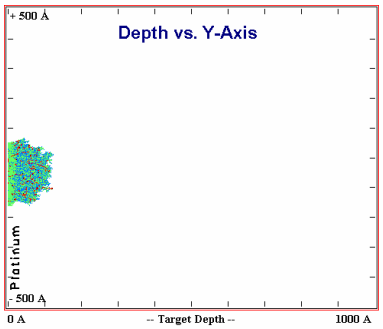
GaC 10 KeV; 0; 1E4 FDC



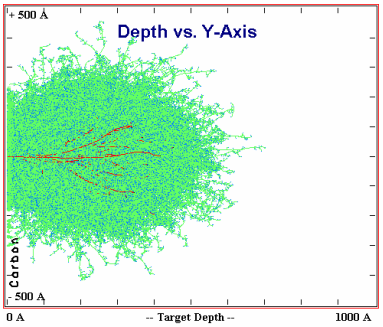
GaPt 10 KeV; 0; 1E4 FDC



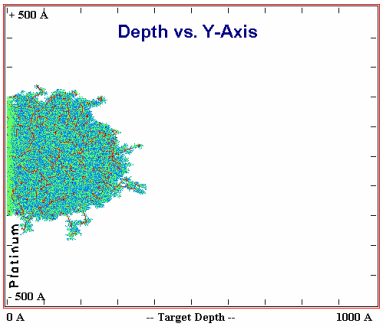
GaC 10 KeV; 88; 1E4 FDC



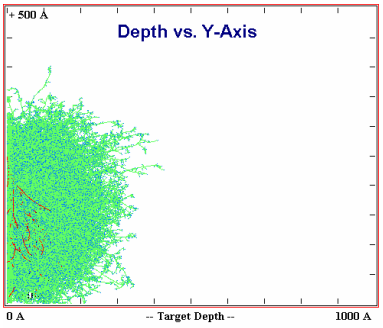
GaPt 10 KeV; 88; 1E4 FDC



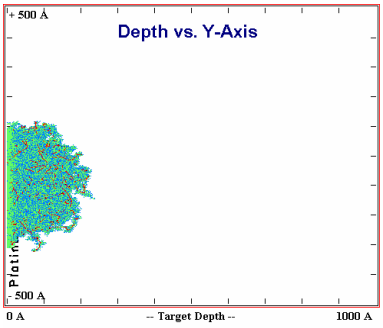
GaC 30 KeV; 0; 1E4 FDC



GaPt 30 KeV; 0; 1E4 FDC



GaC 30 KeV; 88; 1E4 FDC



GaPt 30 KeV; 88; 1E4 FDC

Figura 4.7: Cascata de íons primários e secundários dentro do alvo; material – C e Pt, E_i = 10 e 30 keV; 10^4 íons, $\theta = 0°$ e 88º.

Estes dados demonstram a possibilidade de conseguir a resolução lateral para processos de corte (*milling*) por feixe de íons na faixa de ~10 a 20 nm para membranas com espessura de 5 a 10 nm.

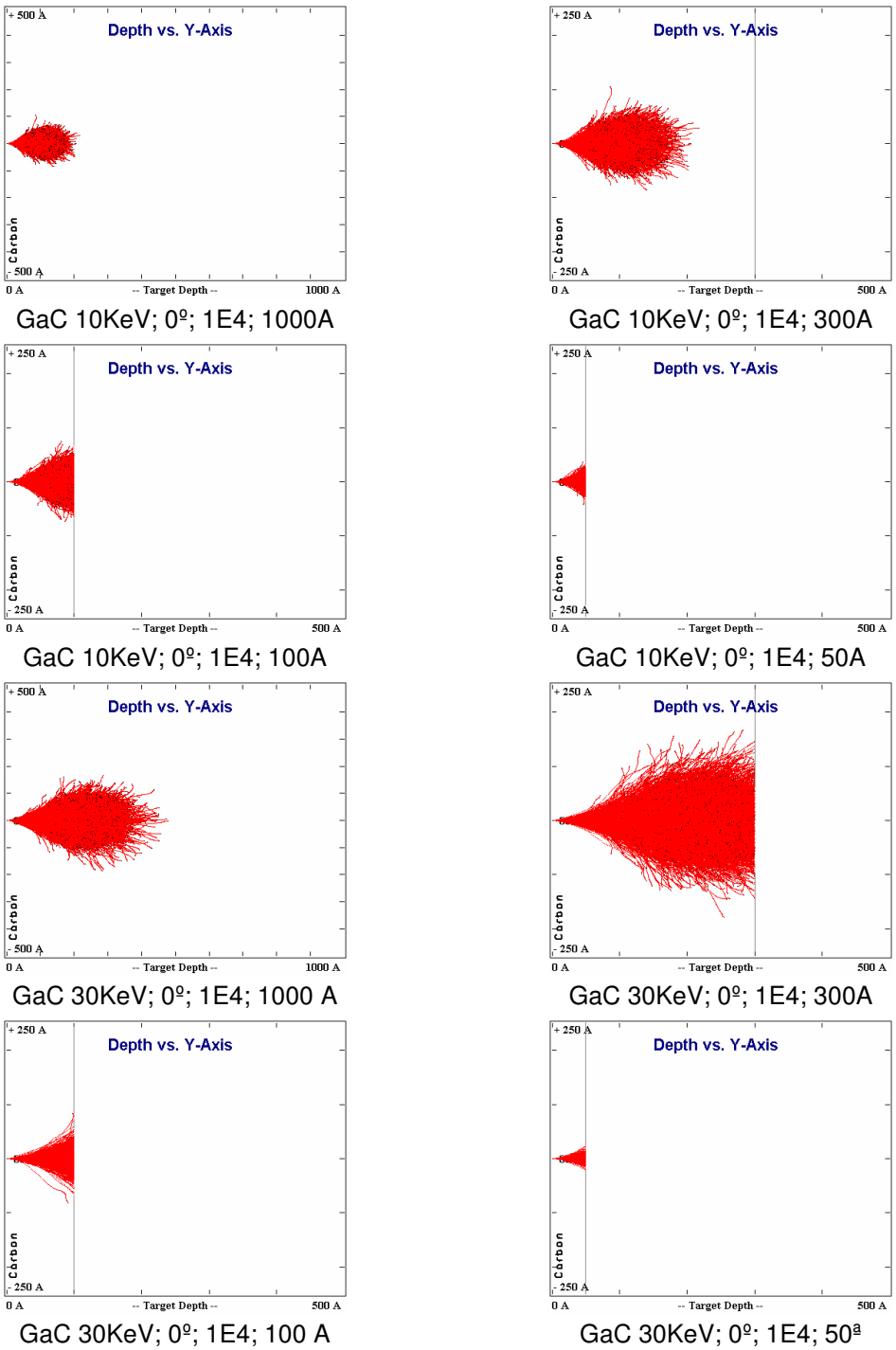
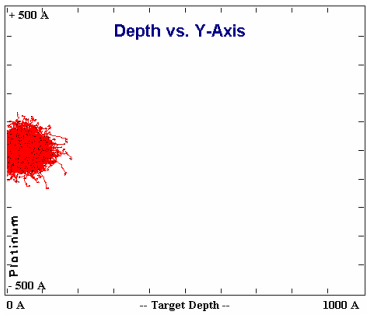
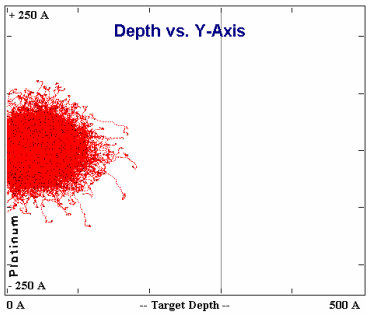


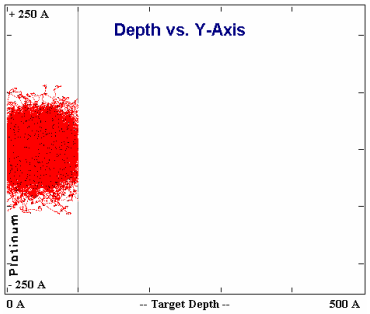
Figura 4.8: Cascata de íons primários dentro de membranas com espessuras 1000, 300, 100, 50A, material – C, E_i = 10 e 30 keV; 10^4 íons, $\theta = 0°$.



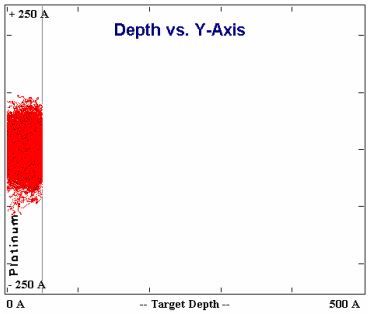
GaPt 10KeV; 0º; 1E4; 1000A



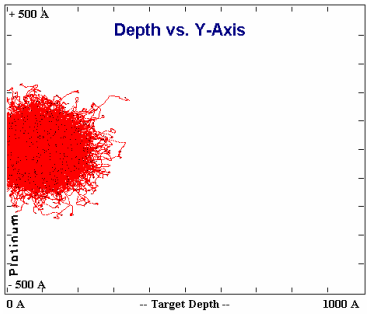
GaPt 10KeV; 0º; 1E4; 300A



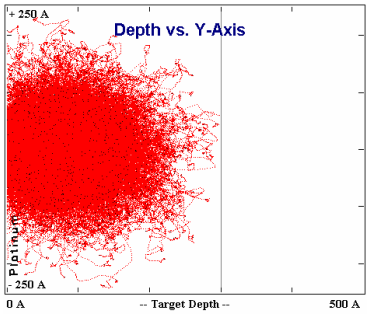
GaPt 10KeV; 0º; 1E4; 100A



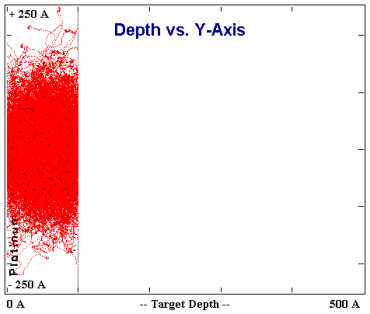
GaPt 10KeV; 0º; 1E4; 50A



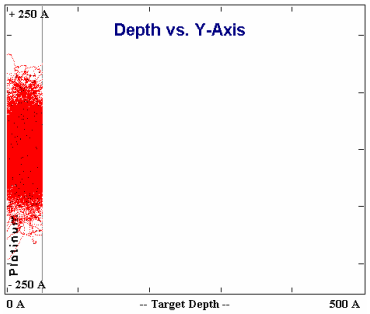
GaPt 30KeV; 0º; 1E4; 1000A



GaPt 30KeV; 0º; 1E4; 300A

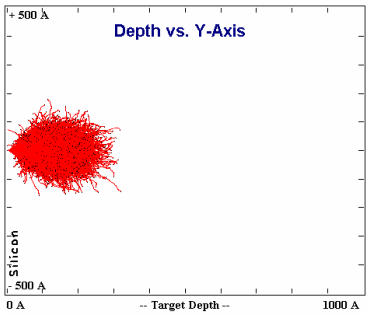


GaPt 30KeV; 0º; 1E4; 100A

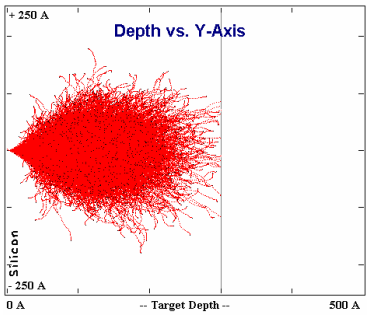


GaPt 30KeV; 0º; 1E4; 50A

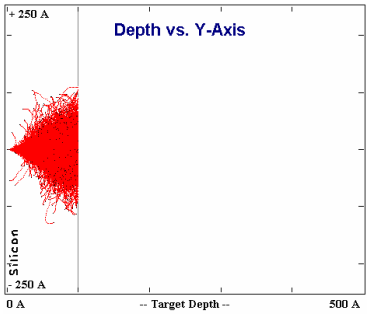
Figura 4.9: Cascata de íons primários dentro de membranas com espessuras 1000, 300, 100, 50A, material – Pt, E_i = 10 e 30 keV; 10^4 íons, $\theta = 0°$.



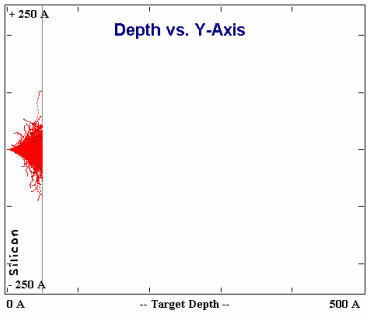
GaSi 10KeV; 0º; 1E4; 1000A



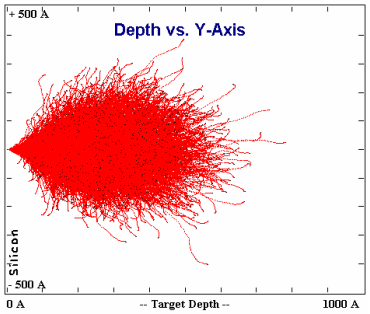
GaSi 10KeV; 0º; 1E4; 300A



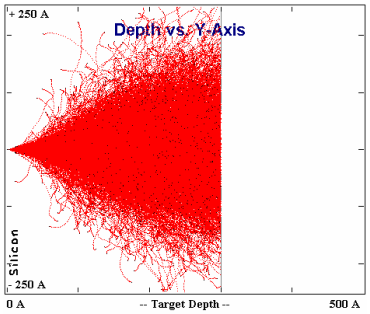
GaSi 10KeV; 0º; 1E4; 100A



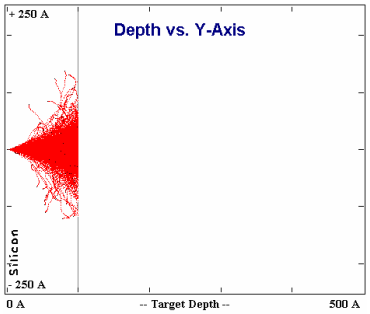
GaSi 10KeV; 0º; 1E4; 50A



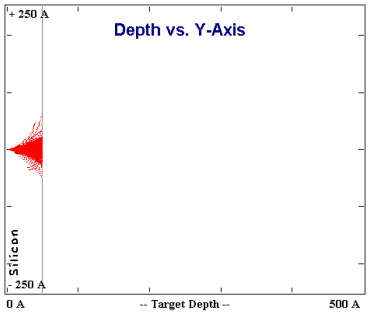
GaSi 30KeV; 0º; 1E4; 1000A



GaSi 30KeV; 0º; 1E4; 300A



GaSi 30KeV; 0º; 1E4; 100A



GaSi 30KeV; 0º; 1E4; 50ª

Figura 4.10: Cascata de íons primários dentro de membranas com espessuras 1000, 300, 100, 50A, energias de íons 10 e 30 keV, material Si.

4.3 – Resultados com simulações: comparando desempenho entre CPUs e GPGPUs

Com o objetivo de reduzir o tempo necessário para a simulação de experimentos com TRIM/SRIM, foram avaliadas diversas possibilidades para execução do programa nas arquiteturas mais comuns, disponíveis para uso regular na universidade.

Dentre as diversas simulações realizadas com o programa TRIM/SRIM, observou-se entretanto, que quando solicitado cálculos com maior número total de íons era necessário maior tempo para processamento dos dados, o que bloqueia a utilização do computador durante algum tempo (que pode ser de algumas horas até alguns dias), havendo a possibilidade de travamentos do sistema operacional (o que requer reinício de todo o processo de simulação) e conseqüente perda de dados.

Para evitar tais restrições, foram estudadas as maneiras de virtualizar o ambiente em que se pudesse efetuar as simulações, sem bloquear o uso do equipamento e sem a necessidade de executar uma única simulação a cada vez.

Para conduzir este experimento foram utilizados computadores fabricados pela empresa AMD (entre 2004 e 2011) e INTEL (entre 2008 e 2010), cujo nome do processador, velocidade em gigahertz (GHz), quantidade de memória RAM, conectividade com rede de dados, instalação de programa gestor para virtualização (VMWare Player), versão do sistema operacional Microsoft Windows e respectivo tempo necessário para simulação; além da aceleração obtida, estão na tabela 4.3. Também é importante não interferir no comportamento da simulação e considerar o menor uso de espaço em disco rígido possível, além dos respectivos ajustes para uso de memória RAM e memória virtual do sistema, que permite a utilização de várias cópias de um mesmo ambiente virtualizado.

Foram elaboradas possíveis combinações para a execução do programa TRIM/SRIM diretamente no host (quantidade de *Virtual Machines*, VMs, indicado por nenhuma) ou em modo hospedado, respeitando a alocação de um núcleo (*core*) para o supervisor (*hypervisor*) e um núcleo para cada máquina virtual, até o limite de núcleos disponível em cada computador utilizado.

Foram utilizados no cenário de testes o sistema operacional Windows (32 bits) nas versões XP (SP2 e SP3, onde SP significa *Service Pack*) e Sete, além do GNU/Linux, distribuição Ubuntu (versão 12.04.4; 32bits) como hospedeiro das máquinas virtuais. O TRIM/SRIM é um programa desenvolvido em Visual Basic 6 e funciona em Windows. Dessa forma foram testados os ambientes para simulação com Windows puro como sistema principal (hospedeiro, *host*) e posteriormente, foi utilizado Ubuntu como sistema principal e máquinas virtuais com Windows (hóspede, *guest*).

Observou-se que o comportamento do sistema operacional não sofreu alteração em possuir ou não memória virtual ativada. O espaço em disco necessário foi aquele utilizado pelo próprio Windows, acrescido de 100MB para trabalho com o programa TRIM/SRIM. Nos testes, os requisitos mínimos foram: com a versão XP, 256MB de memória RAM e 4GB de espaço em disco; com a versão Sete, 1GB de memória RAM e 15GB de espaço em disco. Como o programa foi utilizado em ambiente de testes e em ambiente de uso comum da universidade, foi criada uma configuração para usar o programa de forma portátil, independente de instalação no computador. O tutorial para elaboração do TRIM/SRIM portátil encontra-se no apêndice B.

Com relação a memória RAM disponível para o sistema operacional, foi elaborada a tabela 4.3 que traz a relação de equipamentos utilizados, características dos equipamentos e condições dos testes, assim como o tempo necessário para simulação e aceleração.

Em todas as simulações foram utilizadas máquinas com Windows e memória virtual desabilitada. Foi definido como padrão para referência o uso dos seguintes parâmetros no TRIM/SRIM: distribuição de íons e cálculo rápido de danos; íons de gálio com energia de 30 KeV e ângulo de incidência zero; alvo monocamada de carbono com 100nm (1000 Ang) e número total de íons de 1.0E+04 (10.000 íons).

Avaliando os resultados, observamos que as atualizações trazidas pelo pacote de serviços 3 (atualização SP3) estabilizaram o funcionamento do núcleo (*kernel*) do sistema operacional, em relação ao funcionamento com sistema de arquivos, segurança dos dados; aprimoramento dos mecanismos de cálculo; melhoria no gerenciamento de tarefas além da gestão de conexões de rede; assim como avanços no mecanismo de gestão gráfica, que eliminou gargalos e falhas de vídeo. Nestas simulações observou-se o uso total do processador (núcleo) alocado para máquina virtual que trabalhou com carga máxima (cem por cento) durante todo o tempo necessário para o

processamento numérico. O uso de memória RAM ficou em torno de 105MB para o Windows versão XP, acrescidos de 45MB para uso do TRIM/SRIM.

Com relação ao cenário de computação distribuída para execução de grande número de simulações, principalmente com o melhor resultado possível, observou-se até quatro máquinas virtuais podem ser utilizadas em conjunto, sendo necessário o mínimo de 1024MB de RAM para o sistema operacional hospedeiro gerenciar as máquinas virtuais hospedadas. A melhor opção observada foi o uso do Windows, versão XP-SP3 por apresentar os resultados mais rapidamente e necessitar de menor espaço total em disco. Em relação aos diferentes tipos de processadores (CPUs) pudemos observar que a quantidade de núcleos do processador e subtarefas que podem ser executadas simultaneamente fazem diferença no processamento numérico, já que as máquinas mais atuais (tabela 4.3 itens de 5 a 16) permitiram maior quantidade de resultados numa mesma parcela de tempo.

A aceleração obtida com as máquinas virtuais em arquitetura AMD (tabela 4.3 item 11 comparado com o item 16) foi em média duas vezes maior que a mesma tarefa executada diretamente na máquina (execução efetuada diretamente na máquina, *host*, está indicada pela linha verde; tabela 4.3, itens 1, 3, 5 e 12) atuando como hospedeiro; enquanto em arquitetura INTEL houve ganho real entre quatro e quatorze por cento. Dessa forma, pode-se concluir que houve ganho em processamento principalmente na possibilidade de realizar diversas simulações simultaneamente em ambientes virtualizados, quando os requisistos para simulação limitam a utilização de aplicativo.

Além da avaliação de desempenho do TRIM/SRIM, ajustamos os parâmetros das máquinas virtuais para melhorar a utilização de recursos e avaliar o tempo de simulação massiva do aplicativo, com diversas combinações de cenários (energias, ângulos e materiais). Pesquisamos aplicações análogas ao TRIM/SRIM, para verificar o comportamento de código similar que emprega o Método de Monte Carlo em simulações executadas em CPU e GPGPU. Ressaltamos que não se trata da comparação de código idêntico, mas sim, de condições de processamento equivalentes para avaliar a possibilidade de expansão do código do TRIM/SRIM para oferecer ganho de performance quando houver uma GPGPU presente no equipamento onde a simulação é executada.

Tabela 4.3: Desempenho da execução do TRIM/SRIM em Windows

	Processador	GHz	Memória RAM	Acesso a Rede	Qtde de VMs	Tempo em Segundos	Aceleração
1	AMD Sempron 64 2800+	1.60	256 MB	Sim	Nenhuma	WXP-SP3 323	0
2	AMD Sempron 64 2800+	1.60	2048 MB	Sim	1; é a VM	WXP-SP3 335	0
3	INTEL Core2Duo E4600	2.40	256 MB	Sim	Nenhuma	WXP-SP3 223	0
4	INTEL Core2Duo E4600	2.40	2048 MB	Sim	1; é a VM	WXP-SP3 225	0
5	AMD Phenom IIx6 T1100	3.31	8192 MB	Sim	Nenhuma	WXP-SP2 269	0
6	AMD Phenom IIx6 T1100	3.31	256 MB	Sim	1; é a VM	WXP-SP2 258	1,043
7	AMD Phenom IIx6 T1100	3.31	512 a 3072	Sim	1; é a VM	WXP2 268-272	0 a -0,012
8	AMD Phenom IIx6 T1100	3.31	256 MB	Sim	1; é a VM	WXP-SP3 103	2,612
9	AMD Phenom IIx6 T1100	3.31	512 a 3072	Sim	1; é a VM	WXP-SP3 113	0 a 2,380
10	AMD Phenom IIx6 T1100	3.31	3072 MB	Sim	1; é a VM	W7-SP1 289	-0,070
11	AMD Phenom IIx6 T1100	3.31	256 MB	Sim	4 VMs	WXP-SP3 118	2,279
12	INTEL CoreI5 650 Quad	3.20	8192 MB	Sim	Nenhuma	WXP-SP3 118	0
13	INTEL CoreI5 650 Quad	3.20	256 MB	Sim	1; é a VM	WXP-SP3 103	1,146
14	INTEL CoreI5 650 Quad	3.20	512 a 3072	Sim	1; é a VM	WXP-SP3 113	0 a 1,044
15	INTEL CoreI5 650 Quad	3.20	3072 MB	Sim	1; é a VM	W7-SP1 163	-0,277
16	INTEL Corei5 650 Quad	3.20	256MB	Sim	4 VMs	WXP-SP3 104	1,135

Dentre os programas com código fonte aberto, utilizamos o programa MCML (*Monte Carlo for Multi Layered media*) [2] para CPU e o programa CUDAMCML [4] para execução em GPGPU, devido ao comportamento para simulação ser similar ao aplicado no TRIM/SRIM. O algoritmo do MCML calcula o comportamento de fótons que incidem num tecido com o Método de Monte Carlo (MMC) em ambiente multi-camada, similar a incidência de íons em matéria apresentado pelo TRIM/SRIM. Os resultados numéricos estão na tabela 4.4. As simulações em GPGPU foram efetuadas utilizando a placa gráfica NVIDIA modelo GT9800.

Durante a execução dos testes foi utilizado o mesmo arquivo de entrada para a versão 1.2.0 do MCML e do código atualizado e recompilado do CUDAMCML para versão 5.5 do CUDA. Os ajustes dos parâmetros de entrada exigiram algum tempo para testes e a leitura de artigos e tutoriais para compreender o funcionamento do código e resolver alguns travamentos na execução do programa.

A definição de um arquivo de entrada em comum para ambas as versões (CPU e GPGPU) foi necessária para comparação dos tempos obtidos com os resultados numéricos. Foram testadas

as condições de execução para evitar resultados indesejados como resposta. Dentre as recomendações, temos as seguintes limitações: em CUDA, a primeira camada precisa ser de vidro, antes do tecido avaliado; número máximo de camadas é 100. Número mínimo de fótons 17920 (que é o limite de *threads*, ou seja, subprocessos) e máximo 2^{32} (4.2E+09). A versão disponibilizada para GPGPU não verifica integridade ou relação de dados, o que é feito pela versão para CPU (que ainda recebe ajustes por parte do desenvolvedor).

Os resultados numéricos utilizam precisão simples para números reais de 32bits. O programa sorteia números pseudo-aleatórios utilizando o horário do computador como padrão para aplicar no método de monte carlo e considera a distribuição interna dos fótons no tecido se não houver manifestação do usuário. Ao término dos cálculos os resultados são salvos em um arquivo para verificação posterior e o tempo necessário para a simulação é informado (em segundos). Utilizando o parâmetro "`-S<número>`" é informado um valor inicial para o sorteio de números pseudo-aleatórios, enquanto o parâmetro "`-A`" desativa o cálculo de distribuição de fótons. Isto influencia no tempo necessário para simulação, conforme ilustrado na tabela 4.4.

Tabela 4.4: Resultados das simulações com 1.0E+07 fótons

		INTEL E4600	AMD T1100	INTEL Corel5 650 Quad		
	dr, dz	**MCML**	**MCML**	**MCML**	**CUDAMCML**	**CUDAMCML -A**
A	0.01 x 0.01 cm	14665 seg	8310 seg	7988 seg	56.53 seg (141x)	14.27 seg (559x)
B	0.05 x 0.05 cm	14623 seg	8512 seg	7284 seg	25.96 seg (280x)	14.27 seg (510x)
C	0.10 x 0.10 cm	14642 seg	8587 seg	7233 seg	22.93 seg (315x)	14.27 seg (506x)

A tabela 4.4 apresenta todos os tempos em segundos para equiparar todas as simulações com o mesmo conjunto de dados para entrada (sample.mci); o número total de fótons utilizado foi de 1.0E+07, alterando apenas o tamanho das células da grade (dr,dz) utilizada pelo MMC, conforme figura 4.11. Os números entre parênteses nessa tabela representam a aceleração obtida.

Os dados foram obtidos com diferentes processadores para o código MCML. Observamos neste caso que os resultados obtidos com processador INTEL mais atual foi melhor, reduzindo o tempo de processamento em CPU praticamente pela metade.

Quando comparados os resultados obtidos no computador com Core i5, observamos uma melhora ainda maior. Analisando o tempo de resposta obtido nos cálculos (tabela 4.4, item A), houve redução de 141 vezes no tempo de resposta obtido com GPGPU em relação ao tempo

necessário com CPU. Foi utilizada uma placa gráfica NVIDIA modelo GT 9800 com 512MB de memória dedicada; CUDAMCML utilizando geração automática de números aleatórios considerando a dispersão (cálculo completo por padrão). Quando ignorada a dispersão (utilização do parâmetro `-A`) para processamento com GPGPU, houve redução de 559 vezes no tempo de resposta em comparação ao tempo necessário para CPU.

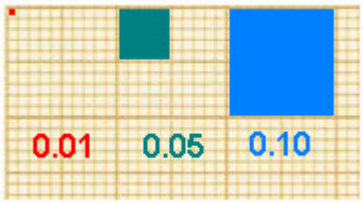


Figura 4.11: Exemplo de tamanho de célula da grade

Os resultados obtidos com os códigos para CPU e GPGPU utilizando MCML e CUDAMCML respectivamente mostram que o tempo das simulações está relacionado com a resolução da grade (dr,dz), principalmente para resolução em GPGPU (tabela 4.4). A resolução da grade é importante para o usuário conforme o refinameno numérico desejado [4]. Considerando que é utilizado o Método de Monte Carlo nestas simulações, observou-se a redução de tempo de processamento em GPGPU, com o uso de grades (dr, dz) maiores, onde o tempo de resposta reduziu entre 141 a 315 vezes.

Observando que a dose mínima com tempo de um segundo no equipamento FIB/SEM é de 1.62E+19 ions/cm2 e o máximo computado pelos programas era 1.0E+07, novas rodadas de simulação com MCML e CUDAMCML foram executadas até o limite de 1.0E+09, que é o atual limite para variáveis do tipo inteiro no programa. Os resultados são apresentados na tabela 4.5.

Tabela 4.5: Resultados com incremento do número total de fótons para região 0.01x0.01 (dr,dz).

dr,dz	Photons	MCML	CUDAMCML	CUDAMCML -A
0.01 x 0.01 cm	1.0E+07	7988 s	56.53 s (141x)	14.27 s (559x)
0.01 x 0.01 cm	1.0E+08	68356 s	559.17 s (122x)	141.82 s (482x)
0.01 x 0.01 cm	1.0E+09	~777600 s	5617 s (138x)	1416.18 s (549x)

Os valores entre parênteses nas tabelas 4.4 e 4.5 significam aceleração obtida. Como exemplo, na tabela 4.5 a GPGPU realizou a simulação 141 vezes mais rápido em relação ao tempo necessário para a mesma simulação em CPU.

Quando realizada uma simulação utilizando apenas CPU para o cálculo numérico, observou-se que são necessários vários dias. No caso de 1.0E+09 foi estimado um prazo de nove dias, que foi praticamente cumprido. O exemplo utilizado na simulação foi monocamada e o tempo total necessário demasiadamente alto para obter uma resposta. Considerando que o processo de fabricação é mais complexo, o tempo necessário para simulação seria ainda maior.

Entretanto, a simulação realizada com GPGPU apresentou resultados para 1.0E+09 em cerca de noventa minutos, o que demonstrou ser viável a simulação quando empregada GPGPU e os modelos otimizados. Tais ganhos de tempo podem ser observados no gráfico gerado (figura 4.12) com os resultados obtidos nas simulações, detalhados na da tabela 4.6.

Tabela 4.6: Tempo em segundos para resultados com MCML e CUDAMCML

Dose (Nro.Total de fótons)	CPU (MCML)	GPGPU (CUDAMCML)	GPGPU (CUDAMCML –A)
1.0E+09	776786,00	5617,00	1416,18
1.0E+08	68356,00	559,17	141,82
1.0E+07	7988,00	56,53	14,27
1.0E+06	1032,00	5,12	1,41
1.0E+05	103,00	0,69	0,26
1.0E+04	11,00	0,00	0,00
1.0E+03	1,00	0,00	0,00
1.0E+02	0,00	0,00	0,00
1.0E+01	0,00	0,00	0,00

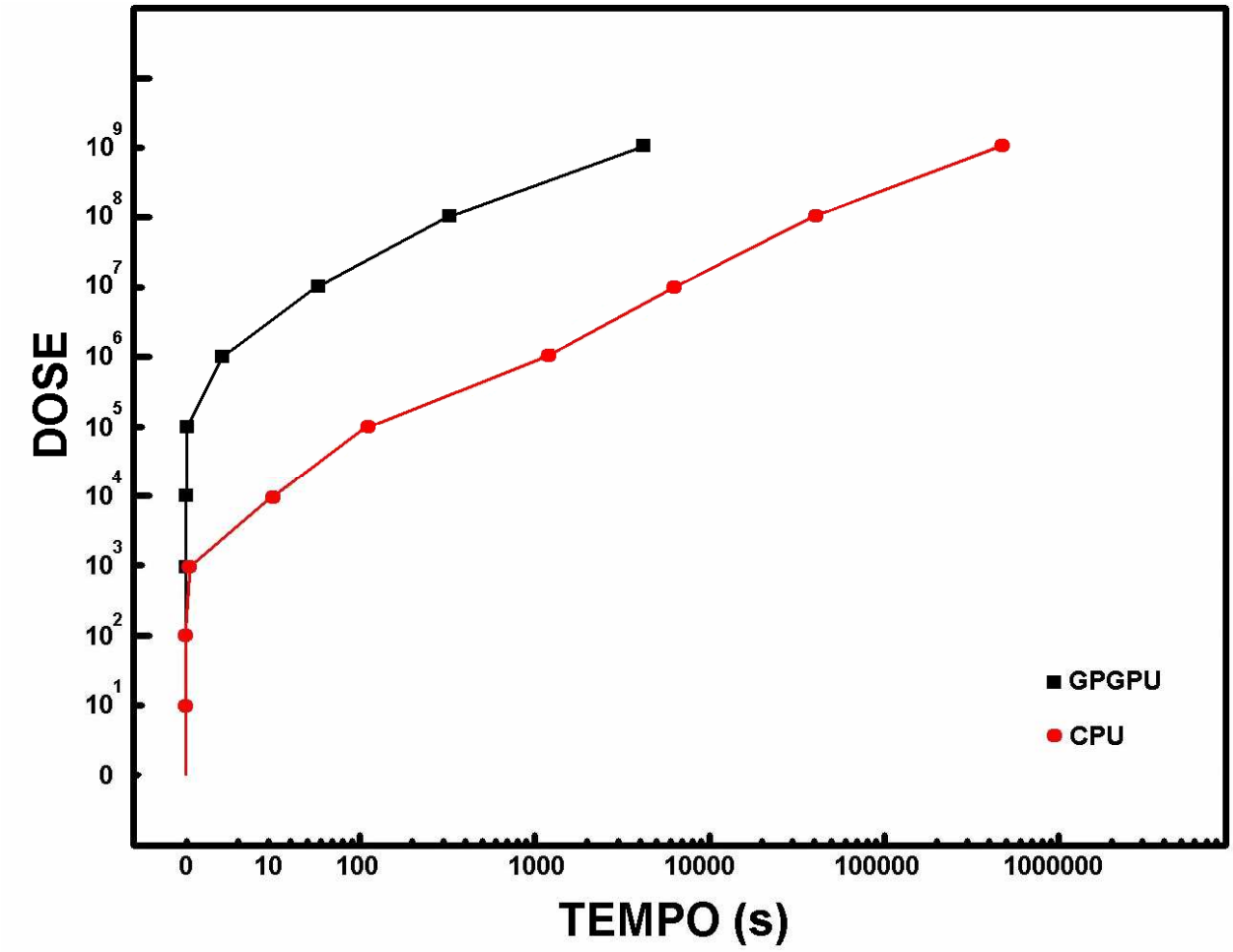


Figura 4.12: Resultados obtidos com MCML (CPU) e CUDAMCML (GPGPU)

Capítulo 5

5 – CONCLUSÕES

Este trabalho multidisciplinar aborda questões de química, física, engenharia elétrica e traz principalmente resultados obtidos com simulações por computador. A computação contribui aqui com aceleração do processo de simulação, reduzindo o tempo necessário para diversas aplicações em execução paralela em ambiente virtualizado e principalmente ganhos com a utilização de placa gráfica (GPGPU) na resolução numérica de problemas que utilizam principalmente o Método de Monte Carlo.

Foram estudadas diversas ferramentas que pudessem aprimorar o trabalho no CCS/UNICAMP (Centro de Componentes Semicondutores da Universidade Estadual de Campinas) cujo custo fosse baixo ou nenhum para manutenção dessas atividades, observando-se apenas o investimento em computadores e ajustes para fácil utilização em um laboratório multiusuário.

Dentre diversas possibilidades de implementação de soluções, observou-se a complexidade para criação de uma nova ferramenta a partir do zero e o alto relacionamento entre conceitos sobre física e interações das partículas. Como todo software que trabalhe com altas energias, energia nuclear ou feixe de íons é altamente amarrado a centros de pesquisa, não foi possível obter qualquer código-fonte de programas que trabalhem com colisões binárias dentre outros conceitos. Todos os reservatórios de programas estão em servidores nos EUA, cuja burocracia para acesso é grande. Alguns poucos programas estão abertos para uso em outras instituições para pesquisa, como o TRIM/SRIM (Transport of Ions in Matter, posteriormente chamado de Stopping and Range of Ions in Matter) [1] e programas utilizados no laboratório Sandia [5] para pesquisa computacional ou pesquisa biomédica [3].

5.1 – Resultados obtidos

Foi possível observar a redução de tempo ao aplicar a simulação em lotes de processamento distribuindo a tarefa em máquinas virtuais; redução de tempo nos cálculos quando é elevado o volume de dados e iterações entre resultados parciais até se obter uma resposta final.

Com o auxílio do TRIM/SRIM foram realizados cálculos de cascatas de colisões de íons de Ga dentro de vários substratos, incluindo materiais leves (C – grafite) e Si bem como pesados como Pt, variando ângulos de incidência, energia de íons, dose de irradiação.

Para energia de íons 30 keV, geralmente utilizada para processamento, a área amorfizada se expande para até 40-50 nm na direção longitudinal e cerca de 20-25 nm na direção lateral, dependendo do material. Para angulos de incidência rasante (80º e 88º), o volume e a extensão lateral da área amorfizada diminuem, comparando com o ângulo de incidência normal. Este tipo de processo com incidência de íons rasante, é utilizado para polimento das paredes de amostras (pois proporciona menor amorfização na direção lateral).

Foram calculadas as cascatas em dois modos: considerando somente íons primários e também incluindo íons e átomos secundários. Os resultados mostraram a importância da inclusão de partículas secundárias que implica em alargamento da região amorfizada, ou em outras palavras, afeta a resolução do processamento pelo feixe de íons de Ga por FIB (*Focused Ion Beam*). Para fins de melhora na resolução, foi considerado o processo em cima de membranas finas dos materiais em questão. Foi demonstrado que a extensão lateral da área amorfizada pode ser diminuída para membranas com espessuras menores de 30 nm, chegando a 6 nm para membranas de C (amorfo) de 5 nm, com energias de íons de Ga de 30 keV. Em experimentos realizados no CCS com camadas finas de grafeno de camadas múltiplas, suspensas entre os eletrodos de metal, foi obtida a resolução de processamento de aproximadamente 50 nm, de acordo com os cálculos considerando as partículas secundarias.

Em relação às simulações, o programa TRIM/SRIM informa em sua documentação o suporte a até (1E7)-1 para o número total de íons incidentes no alvo, entretanto, foi constatado que o programa falha na execução dos cálculos (runtime error 6: *overflow*; erro na execução do programa, onde o resultado do cálculo é maior do que o tamanho máximo suportado pela variável) após aproximadamente 50 horas. Esta falha ocorre devido as características da linguagem em que o programa foi desenvolvido e das características de gestão de recursos do

sistema operacional onde é necessária a execução do aplicativo. Dessa forma, há sucesso nos cálculos de incidência de íons em um material monocamada para dose incidente total de até 1E6 íons. O tempo necessário para essa simulação foi de 18120 segundos, não sendo possível a execução equivalente em GPGPU.

Para análise análoga, o programa MCML (*Monte Carlo for Multi Layered media*) [3] efetuou a simulação para monocamada em CPU em 1032 segundos enquanto o CUDAMCML (*Compute Unified Device Architecture* MCML, ou seja, o mesmo programa ajustado para funcionar em GPGPU) efetuou a mesma simulação em 5,12 segundos; e foi além seguindo com cálculos até o limite da variável utilizada, realizando simulação com dose 1E9 (limite maior em 3 ordens de grandeza) realizada em até 777600 segundos (conforme previsão do algoritmo). Quando executado em GPGPU, efetuou os mesmos cálculos em 5617 segundos.

Estes resultados estão tabulados a seguir (tabela 5.1) onde fica claro que o tempo necessário para simulação com GPGPU é muito menor do que àquele necessário para realizar a mesma tarefa em CPU. Dessa forma é viável realizar simulações porque o tempo empregado no estudo é menor do que àquele necessário para o mesmo processo empírico em laboratório.

Tanto a utilização de virtualização para simulações como a adequação de programas para GPGPU pode melhorar o desempenho dos programas existentes com redução de tempo necessário para simulação. Este trabalho comparou programas (TRIM/SRIM, MCMML e CUDAMCML) que empregam métodos convencionais para calcular o alcance de fotons/íons na matéria utilizando aproximações de colisões binárias (BCA, *Binary Collision Approximation*) [98], onde os cálculos consideram as sucessivas colisões individuais dos íons em relação à matéria (alvo) e os desvios dos átomos, ou seja, uma maneira para avaliar de maneira eficiente a profundidade de penetração de íons em sólidos; assim como o Método de Monte Carlo (MMC) utilizando em CPU e GPGPU.

Tabela 5.1: Resultados em simulação

		Tempo (segundos)		
	Dose	**CPU**	**GPGPU**	**Aceleração**
TRIM/SRIM e N/D	1E6	18120,00	N/D	N/D
MCML e CUDAMCML	1E6	1032,00	5,12	201x
MCML e CUDAMCML	1E9	777600,00	5617,00	138x

*N/D = não disponível

5.2 – Perspectivas e trabalhos futuros

Com relação aos experimentos em laboratório, realizar experimentos com camadas ultrafinas de grafeno (até 5-10 nm de espessura) para obtenção de melhor resolução de processamento por feixe de ions (*milling,* e deposição de camadas de metal ou de SiO_2 induzidos por ions). A expectativa é obter resolução lateral comparavel com a largura do feixe minima, na faixa de 7-10 nm.

No que tange o trabalho com computação de alta performance, fizemos contato com o senhor James F. Ziegler, que concordou com a expansão de seu programa, acrescentando melhorias ao código do TRIM/SRIM (com agendamento do período de interação). Ele não pretende abrir o código-fonte do programa no momento, mas permitirá o acesso para colaboradores que forem até ele para trabalhar em conjunto.

Com o acesso ao código-fonte do TRIM/SRIM será possível expandir o poder de processamento e condições (cenários) para simulações, assim como pré-definir os resultados que se deseja analisar posteriormente, possibilitando continuar os cálculos a partir de determinado ponto (e tempo de processamento realizando operações); disponibilizar resultados numéricos em formatos abertos intermediários (.CSV, por exemplo); integração com web; aplicação de contextos a CPU e GPGPU, entre outras facilidades para simulações em nanoengenharia.

REFERÊNCIAS

[1] ZIEGLER, J.F., ZIEGLER, M.D. e BIERSACK, J.P. “SRIM - The Stopping and Range of Ions in Matter”, Programa para computador, versão 2008.03, (2008). Disponível em: <http://www.srim.org>. Acesso em: 30 set. 2013.

[2] WANG, Lihong, JACQUES, S.L. e ZHENG, L.-Q. “MCML - Monte Carlo modeling of photon transport in multi-layered tissues”. Computer Methods and Programs in Biomedicine 47, 131-146 (1995). Disponível em: <http://labs.seas.wustl.edu/bme/Wang /mc.html>. Acesso em: 30 set. 2013.

[3] Oregon Medical Laser Center. "Monte Carlo Simulations". Disponível em: <http://omlc.ogi.edu/software/mc/>. Acesso em: 30 set. 2013.

[4] ALERSTAM, E., SVENSSON, T. e ANDERSON-ENGELS, S. “GPU Monte Carlo for Graphics Cards (CUDAMCML)”. J. Biomedical Optics Letters 13, 060504 (2008). Disponível em: <http://www.atomic.physics.lu.se/biophotonics/our_research/monte_carlo_ simulations/gpu_monte_carlo/>. Acesso em: 30 set. 2013.

[5] SANDIA National Laboratories. “SPPARKS Kinetic Monte Carlo Simulator”. Disponível em: <http://spparks.sandia.gov>. Acesso em: 30 set. 2013.

[6] Universidade Federal do Pará. “Química Geral > Capítulo I – Estrutura Atômica (PDF)”. Material para licenciatura em química, modalidade à distância. Disponível em: <http://www2.ufpa.br/quimdist/Livros/Qu%EDmia%20Geral%20Def%20PDF/Cap%EDtul o%20I%20Estrutura%20At%F4mica.pdf>. Acesso em: 30 set 2013.

[7] SANTOS, Antonio C.F.. “Interação de partículas carregadas com a matéria”. UFRJ, disciplina FIW474, ministrada no primeiro semestre de 2007. Disponível em: <http://www.if.ufrj.br/~mms/lab4/capitulo_10corrigido.pdf>. Acesso em: 30 set 2013.

[8] BOHR, Niels. “On the theory of decrease of velocity of moving electrified particleson passing through matter”. Philos. Mag. 25, 10-31 (1913).

[9] BETHE, H.A. “The theory of the passage of rapid neutron radiation through matter”. Annalen der Physik 5, 325-400 (1930).

[10] LINDHARD, J., SCHARFF, M. e SCHIOTT, H.E. “Range concepts and heavy ions ranges”. Matematisk-fysiske Meddelelser Det K. Danske Videnskabernes Selskab. 33(14) (1963).

[11] LINDHARD, J. e WINTER, A. “Stopping power of electron gas and equipartition rule”. Matematisk-fysiske Meddelelser Det K. Danske Videnskabernes Selskab. 34(4) (1964).

[12] NORTHCLIFFE, L.C. “Passage of heavy ions through matter. Annual Reviews of Nuclei Science 13, 67-102 (1963).

[13] SCHIWIETZ, G. e GRANDE, P.L. “Unitary convolution approximation for impact-parameter dependent electronic energy loss”. Nuclear Instruments and Methods B 153, 1-9 (1999).

[14] SIGMUND, P. e SCHINNER, A. “Binary stopping theory for swift heavy ions”. European Physics Journal D 12, 425-434 (2000).

[15] NOVAIS, Vera Lúcia Duarte de. "Química, Volume 1- Química Geral e Inorgânica". Editora Atual, 1993. ISBN: 9788570564870.

[16] ARASSE, Daniel. “Leonardo DaVinci”, 1998, ISBN 1568521987.

[17] ALVES, Virginia M. et al. “Histórico da radioatividade”. Universidade Federal de Pelotas, Pelotas/RS, jul. 1999. Disponível em: <http://ifm.ufpel.edu.br/histfis/first.htm>. Acesso em 30 set. 2013.

[18] ZIEGLER, J.F.“SRIM – The Stopping and Range of Ions in Matter”, 2008, ISBN0-9654207-1-X.

[19] BRITANNICA. “Stopping Power (particles radiation)”. Enciclopédia online. Disponível em: <http://www.britannica.com/EBchecked/topic/488507/radiation/28825/The-passage-of-matter-rays#ref398849> Acesso em: 30 set. 2013

[20] MAHAN, B. M.; MYERS, R. J. Química: um curso universitário. Trad. 4a Ed. americana. São Paulo: Blucher, 1995.

[21] TESMER, J.R. e NASTASI, M. A. “Handbook of modern ion beam materials analysis”. Materials Research Society, Julho (1995).

[22] BOHR, N. “The penetration of atomic particles through matter”. Matematisk-fysiske Meddelelser Det K. Danske Videnskabernes Selskab. 18(8), 1-144 (1948).

[23] SIGMUND, P. “Stopping of heavy ions - a theoretical approach”, volume 1. Springer, (2004).

[24] FIRSOV, O.B. “A qualitative interpretation of the mean electron excitation energy in atomic collisions”. Zh. Eksp. Teor. Fiz. 36, 1517-1523 (1959).

[25] LINDHARD, J. e SCHARFF, M. “Energy dissipation by ions in the keV region”. Physical Review 124, 128-130 (1961).

[26] PAUL, Helmut. “Stopping power for light ions”. Disponível em: <http://www-nds.iaea.org/stopping/>. Acesso em: 30 set. 2013.

[27] GRANDE, P.L. e SCHIWIETZ, G. “Impact-parameter dependence of the electronic energy loss of fast ions”. Physical Review A 58, 3796-3801 (1998).

[28] SIGMUND, P. e SCHINNER, A. “Binary theory of electronic stopping”. Nuclear Instruments and Methods B 195, 64-90 (2002).

[29] BETHE, H.A. “Intermediate Quantum Mechanics”. W. A. Benjamin, 1a. ed., (1964).

[30] SIGMUND, P. “Particle penetration and radiation effects: general aspects and stopping of swift point charges”, volume 1 of Solid-State Science. Springer-Verlag, Germany, (2006).

[31] FANO, U. “Penetration of protons, alpha particles and mesons”. Annual Reviews of Nuclei Science 13, 1-66 (1963).

[32] ZIEGLER, J.F. “Stopping of energetic light ions in elemental matter”. Journal of Applied Physics 85(3), 1249-1272 (1999).

[33] LINDHARD, J., NIELSEN, V. e SCHARFF, M. “Approximation method in classical scattering by screened coulomb fields”. Matematisk-fysiske meddelelser udgivet af det kongelige danske videnskabernes selskab 36(10), 3 (1968).

[34] LINDHARD, J. “On the properties of a gas of charged particles”. Matematisk-fysiske meddelelser kongelige danske videnskabernes selskab 28(8), 1-57 (1954).

[35] NORTHCLIFFE, L.C. e SCHILLING, R.F. “Range and stopping power tables for heavy ions”. Nuclear Data Tables A7, 233-463 (1970).

[36] ZIEGLER, J.F., BIERSACK, J.P. e LITTMARK, U. “The stopping and ranges of ions in matter”, volume 1. Pergamon Press, 1a. ed., (1985).

[37] NORTHCLIFFE, N.C. “Energy loss and effective charge of heavy ions in Aluminium”. Physical Reviews 120, 1744-1757 (1961).

[38] BRANDT, W. e KITAGAWA, M. “Effective stopping-power charges of swift ions in condensed matter”. Physical Review B 25, 5631-5637 (1982).

[39] GLAZOV, L.G. e SIGMUND, P. “Nuclear stopping in transmission experiments”. Nuclear Instruments and Methods B 207, 240-256 (2003).

[40] MADOU, M.J. “Fundamentals of microfabrication: the science of miniaturization”, vol. 29, no. 14. CRC Press, 2002, p. 723.

[41] UTKE, R.P., MOSHKALEV, S.A. “Nanofabrication Using Focused Ion and Electron Beams: Principles and Applications”, Oxford University Press, 2012.

[42] TOP500 Supercomputer Sites. Disponível em: <http://www.top500.org>. Acessado em 30 de setembro de 2013.

[43] ARRUDA, Felipe. “A história dos processadores”. Tecmundo, jun. 2011. Disponível em: <http://www.tecmundo.com.br/2157-a-historia-dos-processadores.htm> Acesso em: 30 set. 2013.

[44] BABBO. “Galeria: Evolução dos processadores”. , dez. 2008. Disponível em: <http://www.baboo.com.br/conteudo/modelos/Galeria-Evolucao-dos-processadores_a33780_z0.aspx>. Acesso em: 30 set. 2013.

[45] TORRES, Gabriel. “CISC versus RISC”. Clube do Hardware, jan. 1997. Disponível em: <http://www.clubedohardware.com.br/printpage/857>. Acesso em: 30 set. 2013.

[46] MORIMOTO, Carlo E. “Processadores RISC versus processadores CISC”. Guia do hardware, jul. 2007. Disponível em: <http://www.hardware.com.br/artigos/risc-cisc/> Acesso em: 30 set. 2013.

[47] LANDIM, Wikerson. “Arquitetura ARM” Tecmundo, jan. 2011. Disponível em: <http://www.tecmundo.com.br/7708-por-que-os-processadores-arm-podem-mudar-o-rumo-dos-dispositivos-eletronicos-.htm> Acesso em: 30 set. 2013.

[48] IBM. “IBMi Power Systems Whitepaper”. IBM, jul. 2012. Disponível em: <http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?infotype=SA&subtype=WH&appname=STGE_PO_PO_USEN&htmlfid=POW03032USEN&attachment=POW03032USEN.PDF>. Acesso em: 30 set. 2013.

[49] IBM. “Estratégia com soluções baseadas na linha Power”. IBM, fev. 2011. Disponível em: <http://www-03.ibm.com/systems/power/software/i/strategy.html>. Acesso em: 30 set. 2013.

[50] LONGBOTTOM, Roy. “Computer Speeds From Instruction Mixes pre-1960 to 1971”. PC Benchmark Collection, maio 2013. Disponível em: <http://www.roylongbottom.org.uk>. Acesso em: 30 set. 2013.

[51] JECHLITSCHEK, Christoph. “A Survey Paper on Processor Workloads: MIPS versus FLOPS”. Washington University in St. Louis, Disponível em: <http://www.cs.wustl.edu/~jain/cse567-06/ftp/processor_ workloads/index.html>. Acesso em: 30 set. 2013.

[52] MAPLESON, Ian. “MIPS/MFLOPS and CPU Performance”. Disponível em: <http://www.sgidepot.co.uk/perf.html> Acesso em: 30 set. 2013.

[53] VERICEL, Alfred.“The mother of all CPU charts 2005/2006”. Tom’s hardware, nov. 2005. Disponível em: <http://www.tomshardware.com/reviews/mother-cpu-charts-2005,1175 .html>. Acesso em: 30 set. 2013.

[54] INTEL. “Intel Architectures”. Meld.org, dez. 2011. Disponível em: <http://meld.org/library/education/intel-architectures>; Cópia dos dados disponível em: <http://en.wikipedia.org/w/index.php?title=List_of_Intel_microprocessors>. Acesso em: 30 set. 2013.

[55] GILHEANY, Steve. “Evolution of Intel Microprocessors: 1971-2007”. Disponível em: <www.eie.polyu.edu.hk/~enyhchan/cpu.pdf> Acesso em: 30 set. 2013.

[56] OMENA, Moisés. “Clusters e supercomputação”. Viva o linux, ago. 2004. Disponível em: < http://www.vivaolinux.com.br/artigo/Clusters-e-Supercomputacao>. Acesso em: 30 set. 2013.

[57] ROSE, César A.F., NAVAUX, Philippe O.A. “Arquiteturas Paralelas”. Porto Alegre: Bookman, 2008. 152p. v.15. UFRGS Série Livros didáticos. ISBN: 9788577803095.

[58] DONGARRA, Jack. “Método de referência Linpack”. TOP500, ago. 2007. Disponível em: < http://www.top500.org/project/linpack>. Acesso em: 30 set. 2013.

[59] KOGGE, Peter. “Next-Generation Supercomputers”. IEEE Spectrum, jan. 2011. Disponível em: <http://spectrum.ieee.org/computing/hardware/nextgeneration-supercomputers/0>. Acesso em: 30 set. 2013.

[60] MEUER, Hans. “História sobre o pioneirismo em computação de alta performance (HPC Pioneer)” Supercomp, out. 2010. Disponível em: < http://www.isc-events.com/isc09/About/History >. Acesso em: 30 set. 2013.

[61] XBITLabs. “Computer performance”. Disponível em: <http://forum.xbitlabs.com/ viewtopic.php?f=7&t=15540> Acesso em: 30 set. 2013.

[62] “Grow in supercomputer power”. Disponível em: <http://scalometer.wikispaces.com/singularity> Acesso em: 30 set. 2013.

[63] IEEE Spectrum. “The tops in flops”. IEEE Spectrum, Feb., v.48, issue 2, p. 48-54, 2011.

[64] INPE, Instituto Nacional de Pesquisas Espaciais. “Historia da Supercomputação no CPTEC/INPE”. Disponível em: <http://www.cptec.inpe.br/supercomputador/> Acesso em: 30 set. 2013.

[65] INPE. “Supercomputador”. São Paulo, 2011. Disponível em: <http://top500.org/site/902>. Acesso em: 30 set. 2013.

[66] SINAPAD: Sistema Nacional de Processamento de Alto Desempenho. Disponível em: <http://www.lncc.br/sinapad/> Acesso em: 30 set. 2013.

[67] NVIDIA. “TESLA: Computação de alta performance e supercomputação”. Disponível em: <http://www.nvidia.com.br/object/tesla_computing_solutions_br.html> Acesso em: 30 set. 2013.

[68] LALIN, Matilde N. “Bernoulli Numbers”. University of Texas at Austin, set. 2005. Disponível em: <http://www.dms.umontreal.ca/~mlalin/bernoulli.pdf>. Acesso em: 30 set. 2013.

[69] “The UNIX System home page”. Disponível em: <http://www.unix.org>. Acesso em: 30 set. 2013.

[70] STALLMAN, Richard. “GNU General Public License”. Free Software Foundation Inc., jun. 2007. Disponível em: <http://www.gnu.org/licenses/gpl.html>. Acesso em: 30 set. 2013.

[71] OAK RIDGE National Laboratory. “PVM: Parallel Virtual Machine”. Disponível em: <http://www.csm.ornl.gov/pvm/pvm_home.html>. Acesso em: 30 set. 2013.

[72] CRUZ, Adriano O. e SILVA, Gabriel P. “Programação Paralela e Distribuída – PVM: Um curso prático”. UFRJ, set. 2004. . Disponível em: <http://equipe.nce.ufrj.br/gabriel/progpar/PVM.pdf>. Acesso em: 30 set. 2013.

[73] OPENMP. “Especificações para programação paralela”. Disponível em: <http://openmp.org>. Acesso em: 30 set. 2013.

[74] OPENMPI. “Opensource High Performance Computing”. Disponível em: <www.open-mpi.org>. Acesso em: 30 set. 2013.

[75] NVIDIA. “CUDA: Parallel Programming and Computing Platform”. Disponível em: <http://www.nvidia.com/object/cuda_home_new.html>. Acesso em: 30 set. 2013.

[76] IEEE Standards Association. “POSIX: Padrão 14515-1:2000 /Amd.1-2003 - IEEE ISO/IEC”. Disponível em: <http://posixcertified.ieee.org>. Acesso em: 30 set. 2013.

[77] MALLACH, E.G. “On the Relationship Between Virtual Machines and Emulators”. Proceedings of ACM, SIGARCH-SIGOPS Workshop on virtual computer systems, Cambridge, Massachusetts, United States (1973), 117-126.

[78] GOLDBERG, R.P. “Architectural Principles for Virtual Computer Systems” Thesis, Harvard University, AD-772,809, Feb.1973. Disponível em: <http://www.dtic.mil/cgi-bin/GetTRDoc?AD=AD772809&Location=U2&doc=GetTRDoc.pdf>. Acesso em 30 set. 2013.

[79] LAUREANO, M.A.P. “Máquinas Virtuais e Emuladores: Conceitos, Técnicas e Aplicações”. Editora Novatec, 2006, 184p. ISBN: 85-7522-098-5.

[80] BUTLER, Tim. “BOCHS: Emulador da plataforma IA-32 (x86) PC”. Disponível em: <http://bochs.sourceforge.net>. Acesso em: 30 set. 2013.

[81] Microsoft. “Microsoft Virtual PC”. Disponível em: <http://www.microsoft.com/pt-br/download/details.aspx?id=3702>. Acesso em: 30 set. 2013.

[82] QEMU. “QEMU: Open source processor emulator”. Disponível em: <http://www.qemu.org>. Acesso em: 30 set. 2013.

[83] REYNOLDS, David T. “QEMU Manager”. Disponível em: <http://download.cnet.com/Qemu-Manager/3000-2094_4-75451507.html>. Acesso em: 30 set. 2013.

[84] ORACLE. “VirtualBox: general purpose virtualizer for x86”. Disponível em: < https://www.virtualbox.org>. Acesso em: 30 set. 2013.

[85] VMWARE. “Soluções em virtualização’. Disponível em: <http://www.vmware.com>. Acesso em: 30 set. 2013.

[86] VMWARE. “VMWare Player: download do programa”. Disponível em: <https://my.vmware.com/web/vmware/free#desktop_end_user_computing/vmware_player/6_0>. Acesso em: 30 set. 2013.

[87] PRADA, Rodrigo. “A história dos monitores”. Tecmundo, jul. 2009. Disponível em: <http://www.tecmundo.com.br/2350-a-historia-dos-monitores.htm>. Acesso em: 30 set. 2013.

[88] TORTUGO, John. “CUDA: Modelo de programação paralela”. Blog, dez. 2008. Disponível em: <http://johntortugo.wordpress.com/2008/12/30/cuda-modelo-de-programacao-paralela/>. Acesso em: 30 set. 2013.

[89] JORDÃO, Fabio. “Tudo o que você precisa saber sobre o NVIDIA Tegra 3”. Tecmundo, dez. 2011. Disponível em: <http://www.tecmundo.com.br/processadores/16055-tudo-o-que-voce-precisa-saber-sobre-o-nvidia-tegra-3.htm>. Acesso em: 30 set. 2013.

[90] “CPU and GPU trends over time”. R-Bloggers, jan. 2011. Disponível em: <http://www.r-bloggers.com/cpu-and-gpu-trends-over-time/>. Acesso em: 30 set. 2013.

[91] “CPU Benchmark”. Beyond3D, jan. 2009. Disponível em: <http://forum.beyond3d.com /showthread.php?t=51677/>. Acesso em: 30 set. 2013.

[92] “GPU Benchmark: modelos NVIDIA”. Disponível em: <http://www.hardware-infos.com/grafikkarten_nvidia.php/>. Acesso em: 30 set. 2013.

[93] HUANG, Jen-Hsun.“Exascale: Tegra roadmap 2011-2035”. NVIDIA, jun. 2009. Disponível em: <http://www.nvnews.net/vbulletin/showthread.php?t=168736>. Acesso em: 30 set. 2013.

[94] GUPTA, Sumit. “GPU Supercomputers show exponential growth in TOP500 list”. NVIDIA, nov. 2011. Disponível em: <http://blogs.nvidia.com/2011/11/gpu-supercomputers-show-exponential-growth-in-top500-list/>. Acesso em: 30 set. 2013.

[95] FOSHEIM, Tor. “NVIDIA de olho no futuro”. PCUNLEASH, abr. 2004. Disponível em: <http://www.hardware.no/artikler/nvidia_ser_inn_i_fremtiden/6925>. Acesso em: 30 set. 2013.

[96] NVIDIA. “CUDA Developer Zone”. Disponível em: <http://developer.nvidia.com /category/zone/cuda-zone/>. Acesso em: 30 set. 2013.

[97] PEGDEN, C.D.; SHANNON, R.E. e SADOWSKI, R.P. “Introduction to simulation using SIMAN”. McGraw-Hill, NY, 2nd ed., 1990.

[98] ROBINSON, Mark; Torrens, Ian. “Computer simulation of atomic-displacement cascades in solids in the binary-collision approximation”. Physical Review B 9 (12): 5008. Bibcode 1974PhRvB...9.5008R. doi:10.1103/PhysRevB.9.5008. 1974.

[99] ZIEGLER, J. F.. BIERSACK, J. P.. LITTMARK, U. “In The Stopping and Range of Ions in Matter”. New York: Pergamon, 1985.

[100] BIERSACK, J.; HAGGMARK, L. “A Monte Carlo computer program for the transport of energetic ions in amorphous targets”. Nuclear Instruments and Methods 174: 257. 1980. Disponível em: <http://www.sciencedirect.com/science/article/pii/0029554X80904401>. Acesso em: 30 set. 2013.

[101] ROBINSON, M. “Computer simulation studies of high-energy collision cascades”. Nuclear Instruments and Methods in Physics Research Section B 67: 396. Bibcode 1992NIMPB..67..396R. doi:10.1016/0168-583X(92)95839-J. (1992).

[102] NORDLUND, K, “Molecular dynamics simulation of ion ranges in the 1–100 keV energy range”. Computational Materials Science 3 (4): 448. (1995). Disponível em: <http://beam.acclab.helsinki.fi/~knordlun/mdh/mdh_captions.ps>. Acesso em: 30 set. 2013.

[103] BEARDMORE, Keith; GRONBECH-JENSEN, Niels. Efficient molecular dynamics scheme for the calculation of dopant profiles due to ion implantation. Physical Review E 57 (6): 7278. doi:10.1103/PhysRevE.57.7278. (1998).

[104] SHEPPARD, S. “Applying Software Engineering to Simulation”. Simulation, Vol. 10, No.l, pp:13-19. 1983.

[105] McKAY, K.N. "Software engeneering apphed to discrete event simulation", Winter Simulation Conference, Washington, DC, p. 485-493, 1986.

[106] LAW, A. M., McCOMAS, M.G. and STEPI-IEN, G.V. "The crucial role of input modehng in successfull simulation studies". Industry Engineering, v. 26, n. 9, p. 55-59, July 1994.

[107] JOHNSON, M.E. & POORTE, J.P. "A Hierarchical Approach to Computer Animation in Simulation Modelling". Simulation, 50 (1): 30 -36. (1988).

[108] STOER, J.; BULIRSCH, R.”Introduction to Numerical Analysis”. Terceira edição. Berlin, New York: Springer-Verlag, 2002. 745p. ISBN 978-0-387-95452-3.

[109] POZZO, L.Y. et al. “Implementation of the Conjugate Gradient Method for Sparse Matrix Using GPU”. In: CONEM-2010, 2010, Campina Grande. CD-Proceedings do CONEM-2010. Campina Grande/PB, Brasil: ABCM, 2010. p. 01-10.

[110] COSTA, Therezinha S. "Métodos numéricos para álgebra linear". PUC-Rio, jul 1996. Disponível em: <http://www-di.inf.puc-rio.br/~tcosta/cap2.htm>. Acesso em: 30 set. 2013.

[111] CAMPONOGARA, Eduardo.;CASTELAN NETO, Eugênio B. "Cálculo numérico para controle e automação". UFSC, ago. 2008. Disponível em: <http://www.das.ufsc.br/~camponog/Disciplinas/DAS-5103/LN.pdf>. Acesso em: 30 set. 2013.

[112] Instituto Gauss de Matemática. "Método de Cramer". Disponível em: <http://www.igm.mat.br/aplicativos/index.php?option=com_content&view=article&id=208%3Ametododecramer&catid=41%3Aconteudosal&Itemid=38>. Acesso em: 30 set. 2013.

[113] BELL, Nathan; GARLAND, Michael. "Implementing sparse matrix-vector multiplication on throughput-oriented processors". In Proc. ACM/IEEE Conf. Supercomputing, SC '09, pp. 18. ACM, 2009.

[114] BURKARDT, John. "FORTRAN77: Sparsekit2". Florida State University, out. 2012. Disponível em: <http://people.sc.fsu.edu/~jburkardt/f77_src/sparsekit2/sparsekit2.html>. Acesso em: 30 set. 2013.

[115] TURATTI, L.G. et al. "Performance Improvements in Process Simulation Using GPU". In: SEMINATEC-2011, Abstracts Book. Campinas/SP, Brasil: EDS/IEEE Region 9, 2011. p. 42.

[116] TURATTI, L.G. et al. "Ultimate resolution achievable with focused ion beams: comparing computer simulations with practical process". (poster) In: IBA-2011 (PA-78), 2011, Itapema. Abstracts Book. Itapema/SC, Brasil: IBA, 2011, p. 180.

[117] TURATTI, L.G. et al. "Ultimate resolution achievable with focused ion beams: comparing computer simulations with practical process". Smart Nanocomposites, NY, v.2, n.1, p.85-92, 2011. ISSN:1949-4823.

APÊNDICES

APÊNDICE A – GPGPU

A.1 – Placas gráficas avaliadas

O capítulo 3, item 3.5, apresentou a arquitetura CUDA (*Compute Unified Device Architecture*), da NVIDIA, que permite o desenvolvimento de aplicações que são executadas em GPGPU (*General Purpose Graphics Processing Unit*) aproveitando o poder de processamento destas unidades gráficas para o processamento massivo numérico. As características relevantes dos dispositivos utilizados foram tabuladas a seguir (tabelas A1-A4) e informações sobre outros dispositivos com suporte a CUDA estão disponíveis na página do fabricante [Ra1].

Tabela A1: NVIDIA GeForce 8800 Series – OpenGL 2.1

	GS	**GT**	**GTS**	**GTS**	**GTX**	**Ultra**
Processadores de fluxo	96	112	128	96	128	128
Clock do núcleo (MHz)	550	600	650	500	575	612
Clock do shader/tester (MHz)	1375	1500	1625	1200	1350	1500
Clock da memória (MHz)	800	900	970	800	900	1080
Quantidade de memória (MB)	384	512	512	320	768	768
Interface de memória (bits)	192	256	256	320	384	384
Largura de banda da memória (GB/seg)	38.4	57.6	64	64	86.4	103.7
Taxa de preenchimento de textura (bilhões/seg)	26.4	33.6	41.6	24	36.8	39.2
Suporte a SLI	sim	2-way	sim	sim	sim	sim

Tabela A2: NVIDIA GeForce 9 Series – OpenGL 2.1

	9400 GT	**9600 GT**	**9800 GT**	**9800 GTX+**	**9800 GX2**
Quantidade de GPUs	1	1	1	1	2
Processadores de fluxo	16	64	112	128	256/2
Clock do núcleo (MHz)	550	650	600	738	600
Clock do tester (MHz)	1400	1625	1500	1836	1500
Clock da memória (MHz)	400	900	900	1100	1000
Quantidade de memória (MB)	512	512	512	512	1024
Interface de memória (bits)	128	256	256	256	512
Largura de banda da memória (GB/seg)	12.8	57.6	57.6	70.4	128/2
Taxa de preenchimento de textura (bilhões/seg)	4.4	20.8	33.6	47.2	76.8
Suporte a SLI (n-way)	1	2	2-way	2 e 3	4

Tabela A3: NVIDIA GeForce 200 Series – OpenGL 3.x

	GT 220	**GTS 250**	**GTX 260**	**GTX 280**	**GTX 285**	**GTX 295**
Quantidade de GPUs	1	1	1	1	1	1
Processadores de fluxo (CUDA cores)	48	128	192	240	240	480
Clock do núcleo (MHz)	625	738	576	602	648	576
Clock do tester (MHz)	1360	1836	1242	1296	1476	1242
Clock da memória (MHz)	790	1100	999	1107	1242	999
Quantidade de memória (MB)	1024	1024	896	1024	1024	1792
Interface de memória (bits)	128	256	448	512	512	896
Largura de banda da memória (GB/seg)	25.3	70.4	111.9	141.7	159.0	223.8
Taxa de preenchimento de textura (bilhões/seg)	N/D	47.2	36.9	48.2	51.8	92.2
Suporte a SLI (n-way)	não	2 e 3	2 e 3	2 e 3	2 e 3	4
OpenGL	3.1	3.0	2.1	2.1	2.1	2.1

Tabela A4: NVIDIA GeForce 500 Series (Fermi)

	GT 520	**GTX 560**	**GTX 560 Ti**	**GTX 580**	**GTX 590**
Quantidade de GPUs	1	1	1	1	2
Processadores de fluxo	48	336	384	512	1024/2
Clock do núcleo (MHz)	810	950	822	772	607
Clock do tester (MHz)	1620	1900	1645	1544	1215
Clock da memória (MHz)	900	2200	1900	2004	1707
Quantidade de memória (MB)	1024	1024	1024	1536	3072/2
Interface de memória (bits)	64	256	256	384	768/2
Largura de banda da memória (GB/seg)	14.4	128	128	192.4	327.7
Taxa de preenchimento de textura (bilhões/seg)	6.5	49.8	52.5	49.4	77.7
Suporte a SLI (n-way)	Não	2	2	3	4
OpenGL	4.2	4.1	4.1	4.2	4.2

NVIDIA SLI é uma plataforma que permite aumentar o desempenho gráfico através do uso combinado de dispositivos (placas iguais). Exemplificando, o suporte a SLI 2 significa o uso de duas unidades gráficas que permite até dobrar o desempenho obtido. Para SLI 3 com três

placas, o desempenho pode chegar até a 2.8x em comparação ao mesmo uso com apenas uma placa gráfica. A figura A1 ilustra uma placa mãe com três placas gráficas utilizando SLI (destaque em vermelho). Na página da NVIDIA há maiores informações sobre como obter SLI [Ra2,Ra3].

Figura A1: Sistema com suporte a SLI 3.

No início do estudo e desenvolvimento de aplicativos que utilizam a GPGPU como ferramenta de pesquisa, a limitação encontrada foi a quantidade de computadores equipados com GPGPU com suporte a CUDA. Para contornar esta limitação de recursos o kit de desenvolvimento CUDA 2.0, permitia o trabalho em modo emulado. Isto servia para verifica a sintaxe do programa (executado posteriormente em GPGPU) para verificar se não haviam erros de compilação. Nas versões seguintes, o modo emulado desapareceu e a NVIDIA iniciou o trabalho de otimização dos recursos disponíveis com CUDA para tornar o uso com a linguagem C mais didático. A experiência de uso em ambiente GNU/Linux surgiu da necessidade de acesso multiusuário a um recurso limitado naquela época.

Para compartilhar com a comunidade de pesquisadores os roteiros para instalação adequada do CUDA em equipamentos de mesa (*desktops*) foi criado um website com conteúdo em português [Ra4]. Tais tutoriais são o ponto de partida para o estudo da sintaxe modificada da linguagem C para aproveitar os recursos disponibilizados com CUDA. O sistema operacional utilizado foi GNU/Linux, distribuição Ubuntu.

A.2 – Roteiro para instalação do NVIDIA CUDA

Para versões mais antigas do CUDA e do Ubuntu Linux, estes roteiros foram adicionados para registro histórico das atividades desenvolvidas. A figura A2 ilustra as versões lançadas aos usuários, datas de lançamento e período de suporte. Devido a atualizações das páginas da NVIDIA na Internet, documentos mais antigos e versões anteriores não estão mais acessíveis online e foram arquivadas na mídia que acompanha esta obra.

Version	Code name	Release date	Supported until	
			Desktop	Server
4.10	Warty Warthog	2004-10-20	2006-04-30	
5.04	Hoary Hedgehog	2005-04-08	2006-10-31	
5.10	Breezy Badger	2005-10-13	2007-04-13	
6.06 LTS	Dapper Drake	2006-06-01	2009-07-14	2011-06-01
6.10	Edgy Eft	2006-10-26	2008-04-25	
7.04	Feisty Fawn	2007-04-19	2008-10-19	
7.10	Gutsy Gibbon	2007-10-18	2009-04-18	
8.04 LTS	Hardy Heron	2008-04-24	2011-05-12	2013-04
8.10	Intrepid Ibex	2008-10-30	2010-04-30	
9.04	Jaunty Jackalope	2009-04-23	2010-10-23	
9.10	Karmic Koala	2009-10-29	2011-04-30	
10.04 LTS	Lucid Lynx	2010-04-29	2013-04	2015-04
10.10	Maverick Meerkat	2010-10-10	2012-04-10	
11.04	Natty Narwhal	2011-04-28	2012-10-28	
11.10	Oneiric Ocelot	2011-10-13	2013-04	
12.04 LTS	Precise Pangolin	2012-04-26	2017-04	
12.10	Quantal Quetzal	2012-10-18	2014-04	
13.04	Raring Ringtail	2013-04-18	2015-10	

Color	Meaning
Red	Release no longer supported
Green	Release still supported
Blue	Future release

Figura A2: Histórico das versões da distribuição Ubuntu Linux [Ra5]

Roteiro para instalação em Ubuntu 7.10 do CUDA 2.0 (Em Modo Emulado)

LABAKI, J. "Roteiro para instalação do NVidia CUDA". GPUBrasil, em 10 fev. 2009.

Devido ao alto custo de uma placa gráfica com suporte a CUDA, é possível aprender a programar, verificar a sintaxe e executar seus programas em modo emulado. Estas orientações valem para Ubuntu 7.10 (Gutsy Gibbon) ou 8.04 (Hardy Heron) para 32/64 bits.

1. Baixar os seguintes arquivos do site da NVIDIA:

```
a) NVIDIA_CUDA_Toolkit2.0_Ubuntu7.10x86.run
b) NVIDIA_CUDA_SDK_2.02.0807.1535_linux.run
```

2. Instale os pacotes de compilação em linguagem C necessários com o seguinte comando:

```
$ sudo apt-get install build-essential libglut3-dev libc6-dev-i386 libstdc++6 -y
```

3. A versão 2.x do CUDA requer o gcc 4.1. Caso não seja a versão principal de seu sistema é necessário configurá-lo como alternativa de mecanismo de compilação, através dos comandos:

```
$ sudo apt-get install gcc-4.1 g++-4.1 -y

$ sudo update-alternatives -install /usr/bin/gcc gcc /usr/bin/gcc-4.1 60 -slave
/usr/bin/g++ g++ /usr/bin/g++-4.1
```

4. Para proceder com a instalação do toolkit, altere a propriedade do arquivo .run para que seja interpretado como arquivo executável e invoque sua execução através do shell:

```
$ sudo chmod +x NVIDIA_CUDA_Toolkit2.0_Ubuntu7.10 x86.run

$ sudo ./NVIDIA_CUDA_Toolkit2.0_Ubuntu7.10x86.run auto
```

5. Adicione ao final do arquivo .bashrc as linhas com export, salve o arquivo, saia do editor e feche o terminal para que as alterações tenham efeito.

```
$ gedit .bashrc
...
export PATH=$PATH:/usr/local/cuda/lib
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr /local / cuda/lib
$ exit
```

6. Para instalar o SDK, abra novamente o terminal de texto e digite:

```
$ sudo chmod +x NVIDIA_CUDA_sdk_2.02.0807.1535_linux.run

$ sudo ./NVIDIA_CUDA_sdk_2.02.0807.1535_linux.run
```

7. Entre na pasta do SDK:

```
$ cd NVIDIA_CUDA_SDK
```

8. Compile o SDK utilizando a diretiva de emulação (emu = 1)

```
$ sudo make emu=1
```

9. Entre na pasta dos programas-exemplo compilados

```
$ cd bin/linux/emurelease
```

10. Execute os programas de sua preferência. Por exemplo:

```
./clock
```

Roteiro para instalação em Ubunto 9.10 do CUDA 2.3

FERREIRA, L.O.S. em outubro de 2009.

```
1) Faça o download:
http://www.ubuntu.com/getubuntu/download#
Usar os pacotes NVIDIA para versão Ubuntu 9.04
http://developer.nvidia.com/object/gpucomputing.html (CUDA 3.0beta)
http://developer.nvidia.com/object/cuda_2_3_downloads.html#Linux

2) Ubuntu 9.10 Instalado e pacotes: Driver de vídeo (190.53), Toolkit e SDK em mãos;
2. Parar o GDM (gdm stop) e instalar o driver de video em modo console (texto);
3. Instalar o toolkit;
4. Ajustar e sanar todas as dependencias e links faltantes (apt-get install): linux-
source-2.6.X; linux-headers-2.6.X; gcc-4.1 (ou 3.3, se necessário); g++; mesa-common-
dev; libgl1-mesa-dev; libglu1-mesa-dev; libxi-dev; libxmu-dev; libglut-dev; libxm;
libxi...

ln -s /usr/lib/libglut.so.3 /usr/lib/libglut.so
ln -s /usr/lib/libGLU.so /usr/lib/libGLU.so.1
ln -s /usr/lib/libX11.so.6 /usr/lib/libX11.so
ln -s /usr/lib/libXi.so.6  /usr/lib/libXi.so
ln -s /usr/lib/libXm.so.6  /usr/lib/libXm.so

5. (opcional; não fiz e funcionou) Place a file named cuda.conf in /etc/ld.so.conf.d
with this content: /usr/local/cuda/lib64 (check the directory, if you choosed another
for the installation and for a 32-bit version clear the 64 after lib)

6. Para finalizar, make no diretorio do SDK.



Posteriormente um ajuste único para todos os usuários de um computador.
# CUDA PATHS

Descobrí que instalando os paths no arquivo /etc/profile ao invés do arquivo
.bashrc a vida fica muito mais fácil na hora de usar um IDE, pois do jeito
antigo a gente precisa "ensinar" ao IDE os paths das bibliotecas CUDA todas as
vezes que cria um novo projeto.

Dessa nova maneira esse procedimento fica desnecessário.
=========================================================================
Rode o editor gedit no modo superusuário para acrescentar as seguintes linha
ao final do arquivo de configuração /etc/profile
       sudo gedit /etc/profile
       Após o editor abrir o arquivo, mova o cursor até a última linha e digite
              # CUDA stuff
              PATH=$PATH:/usr/local/cuda/bin
              LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/cuda/lib
              export PATH
              export LD_LIBRARY_PATH
       Salve o arquivo e saia do editor.
```

Roteiro para instalação em Ubuntu 10.04 do CUDA 3.0 (32/64 bits)
TURATTI, L.G. em abril de 2010.

1. Verifique qual a versão de seu sistema operacional com o comando a seguir:
`$ uname –a` Se na resposta tiver i386 é 32 bits; Se tiver x86_64 é 64 bits.

2. Baixar o driver de video 195.36.24 para 32 ou 64 bits;

3. Remover os drivers nvidia instalados

```
$ sudo apt-get –purge remove nvidia-*
```

4. Crie o arquivo `/etc/modprobe.d/nvidia-graphics-drivers.conf` com o seguinte conteudo:

```
blacklist vga16fb
blacklist nouveau
blacklist lbm-nouveau
blacklist nvidia-173
blacklist nvidia-96
blacklist nvidia-current
blacklist nvidiafb
```

5. Crie ou ajuste `/etx/X11/xorg.conf` com o seguinte conteúdo:

```
Section    "Device"
   Identifier "Device0"
   Driver     "nvidia"
   VendorName "NVIDIA Corporation"
EndSection
```

6. Reinicie o computador (reboot);

7. Aparecerá uma tela de erro. Dê “OK” e a seguir, saia para modo console;

8. Faça o login; instale o driver e permita a configuração sugerida

```
$ sudo sh NVIDIA-Linux-<versao>.run
```

9. Reinicie o computador (reboot);

10. Instale o Toolkit para 32 ou 64 bits

```
$ sudo sh cudatoolkit_3.0_linux_<versao>.run
```

11. Adicione no final do arquivo o caminho dos arquivos CUDA para o usuário:

```
$ gedit .bashrc

#### CUDA Settings - Toolkit/SDK ####
export PATH=$PATH:/usr/local/cuda/bin
```

Acrescente uma das linhas abaixo, conforme sua plataforma 32/64 bits

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/cuda/lib      # 32 bits
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/cuda/lib64    # 64 bits
```

Salve e saia do editor.

12. Uma alternativa é seguir a sugestão do toolkit é adicionar as linhas acima em `/etc/ld.so.conf`.

13. Instalando o SDK:

```
$ chmod +x gpucomputingsdk_3.0_linux.run
$ ./gpucomputingsdk_3.0_linux.run
```

14. Satisfazendo dependências do SDK:

```
$ sudo apt-get install libgl1-mesa-dev libglu1-mesa-dev mesa-commondev
libgl1-mesa-dev libglu1-mesa-dev libxi-dev libxmu-dev freeglut3 libmotif3

$ sudo ln -s /usr/lib/libglut.so.3 /usr/lib/libglut.so

$ sudo ln -s /usr/lib/libXm.so.6 /usr/lib/libXm.so
```

15. Para adequar a libGL do pacote atual do driver NVidia 195.36.24

```
$ sudo rm /usr/lib/libGL.so

$ sudo ln -s /usr/lib/libGL.so.1 /usr/lib/libGL.so
```

16. Instalar gcc 4.3 (este é o padrão utilizado pelo CUDA)

```
$ sudo mkdir /usr/bin/gcc-4.3

$ sudo mkdir /usr/bin/g++-4.3

$ sudo mkdir /usr/bin/g++-4.4

$ sudo apt-get install gcc-4.3 g++-4.3

$ sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-4.3 40 --
slave /usr/bin/g++ g++ /usr/bin/g++-4.3

$ sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-4.4 60 --
slave /usr/bin/g++ g++ /usr/bin/g++-4.4
```

17. Configurar para utilizar o gcc4.3

```
$sudo update-alternatives -config gcc
```

18. Compilando os exemplos do SDK:

```
$ cd <diretório do usuário>/NVIDIA_GPU_Computing_SDK/C
$ make
```

19. Após algum tempo a compilação terminará sem mensagens de erros. Entre no diretório e execute os exemplos para testar:

```
$ cd bin/linux/release
$ ./fluidsGL
ou
$ ./particles
```

Pronto! Se os exemplos apareceram, a instalação esta concluída!

Roteiro para instalação do CUDA 5.0 no Ubuntu 12.04.1 LTS

TURATTI, L.G., "Guia de instalação do CUDA 5.0 no Ubuntu 12.04.1 LTS", de 16/out/2012

ATENÇÃO: As informações a seguir documentam a instalação do CUDA 5.0 em um desktop com o sistema operacional Linux Ubuntu versão 12.04.1 LTS instalado e atualizado até 19 de outubro de 2012. A placa utilizada é uma NVidia GeForce GT 9800 (512MB).

1) O sistema operacional pode ser obtido no espelho da Canonical disponibilizado pelo Laboratório de Administração e Segurança de Sistemas (LAS) no endereço a seguir:

```
http://www.las.ic.unicamp.br/pub/ubuntu-releases/12.04/
```

A versão desktop é suficiente para o estudo de CUDA.

2) O pacote do CUDA 5 pode ser obtido diretamente no website da NVidia. Faça o download da versão para Ubuntu 11.10 (32 ou 64 bits) conforme seu sistema operacional. Para verificar a plataforma de seu sistema:

```
$ uname -m          Se a resposta for  i686 = 32 bits; se x86_64 = 64 bits
```

O pacote pode ser baixado daqui: http://developer.nvidia.com/cuda/cuda-downloads

32 bits:

```
http://developer.download.nvidia.com/compute/cuda/5_0/rel/installers/cuda_5.0.35_linux_32_ubuntu11.10.run
```

64 bits:

```
http://developer.download.nvidia.com/compute/cuda/5_0/rel/installers/cuda_5.0.35_linux_64_ubuntu11.10.run
```

3) Nesta versão a NVidia distribuiu o CUDA em um único arquivo com: a) driver de vídeo; b) ferramentas (toolkit) e c) pacote de desenvolvimento (SDK, Software Development Kit) com exemplos de programas. A instalação do driver de vídeo é opcional, pois a versão do driver no sistema Ubuntu 12.04.1 atualizado é o driver 304.51. Caso queira instalar a versão que vem no pacote CUDA 5.0.35 é necessário desativar a interface gráfica e iniciar o sistema em modo textual (console). Dessa forma o driver de vídeo mais recente (versão 304.54) será instalado. Siga os passos abaixo para efetuar a instalação completa do pacote 5.0.35. Se não quiser mexer no driver de vídeo, siga a partir da quarta etapa.

```
$ sudo gedit /etc/default/grub
```

Mude na linha:

```
De: GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
Para: GRUB_CMDLINE_LINUX_DEFAULT="text"
```

Salve o arquivo e feche o editor

Atualize o índice do gerenciador de inicialização do sistema:

```
$ sudo update-grub
```

Reinicie o computador para operar o sistema em modo texto:

```
$ sudo shutdown -r now
```

Faça o login informando usuário e senha.

4) Feito o login, no diretório onde o pacote CUDA foi baixado, ajuste a permissão do arquivo:

```
$ chmod +x cuda_5.0.35_linux_<versão>_ubuntu11.10.run
```

Execute o instalador:

```
$ sudo ./cuda_5.0.35_linux_<versão>_ubuntu11.10.run
```

Aparecerá um aviso sobre o termo de licença de três produtos:

```
a) CUDA Toolkit
>>> Linux: /usr/local/cuda-#.#
b) CUDA Samples
>>> Linux: /usr/local/cuda-#.#   e   $HOME/NVIDIA_CUDA-#.#_Samples
c) NVidia Display Driver
```

Serão apresentadas informações sobre as bibliotecas que serão instaladas:

Componente: CUDA Runtime = libcudart.so
Componente: CUDA FFT Library = libcufft.so
Componente: CUDA BLAS Library = libcublas.so
Componente: CUDA Sparse Matrix Library = libcusparse.so
Componente: CUDA Random Number Generation Library = libcurand.so
Componente: CUDA Performance Primitives Library = libnpp.so

A seguir o termo de licenciamento é mostrado. Aceite e prossiga…

```
Do you accept the previously read EULA (accept/decline/quit): accept
Install NVIDIA Accelerated Graphics Driver for Linux-x86 304.54? ((Y)es/(N)o/(Q)uit): Y
Install the CUDA 5.0 Toolkit? ((Y)es/(N)o/(Q)uit): Y
Enter Toolkit Location [ default is /usr/local/cuda-5.0 ]: <ENTER>
Install the CUDA 5.0 Samples? ((y)es/(n)o/(q)uit): Y
Enter CUDA Samples Location [ default is /usr/local/cuda-5.0/samples ]: <ENTER>
```

Aguarde o término da instalação.

```
Installing the NVIDIA display driver…
Installing the CUDA Toolkit in /usr/local/cuda-5.0 …
Installing the CUDA Samples in /usr/local/cuda-5.0/samples …
```

Quando terminar, aparecerá um resumo da instalação:

```
===========
= Summary =
===========
Driver:   Installation Failed (se pulou etapa 2)
Toolkit:  Installed in /usr/local/cuda-5.0
Samples:  Installed in /usr/local/cuda-5.0/samples (pristine) and
                       /home/<user>/NVIDIA_CUDA-5.0_Samples (writable)
* Please make sure your PATH includes /usr/local/cuda-5.0/bin
* Please make sure your LD_LIBRARY_PATH
```

```
*   for 32-bit Linux distributions includes /usr/local/cuda-5.0/lib
*   for 64-bit Linux distributions includes /usr/local/cuda-5.0/lib64:/lib
* OR
*   for 32-bit Linux distributions add /usr/local/cuda-5.0/lib
*   for 64-bit Linux distributions add /usr/local/cuda-5.0/lib64 and /lib
* to /etc/ld.so.conf and run ldconfig as root
* To uninstall CUDA, remove the CUDA files in /usr/local/cuda-5.0
* Installation Complete
Logfile is /tmp/cuda_install_<numero>.log
```

Caso não tenha instalado o driver de vídeo na etapa 3, siga para a etapa 6.

5) Feita a instalação, a interface gráfica (modo gráfico) será reativado:

```
$ sudo nano /etc/default/grub

Ajuste a linha:
De:GRUB_CMDLINE_LINUX_DEFAULT="text"
Para: GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
```

Pressione CTRL+O para salvar
Pressione CTRL+X para sair

Atualize o índice do gerenciador de inicialização do sistema:

```
$ sudo update-grub
```

Reinicie o computador para operar o sistema em modo texto:

```
$ sudo shutdown -r now
```

Faça o login informando usuário e senha.

6) Agora é necessário ajustar parâmetros e testar os exemplos.

Edite o arquivo .bashrc

```
$ gedit .bashrc
```

Ao final do arquivo insira as linhas:

```
## Configuracao CUDA
export PATH=$PATH:/usr/local/cuda/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/cuda/lib
```

Salve o arquivo e feche o terminal. Quando abrir novo terminal os ajustes serão carregados.
Feito isto será possível testar os exemplos:

```
$ cd ~/NVIDIA_CUDA-5.0_Samples/
```

Escolher um exemplo; compilar (make) e executar. Uma sugestão para testes:

```
$ cd ~/NVIDIA_CUDA-5.0_Samples/2_Graphics/simpleGL/
```

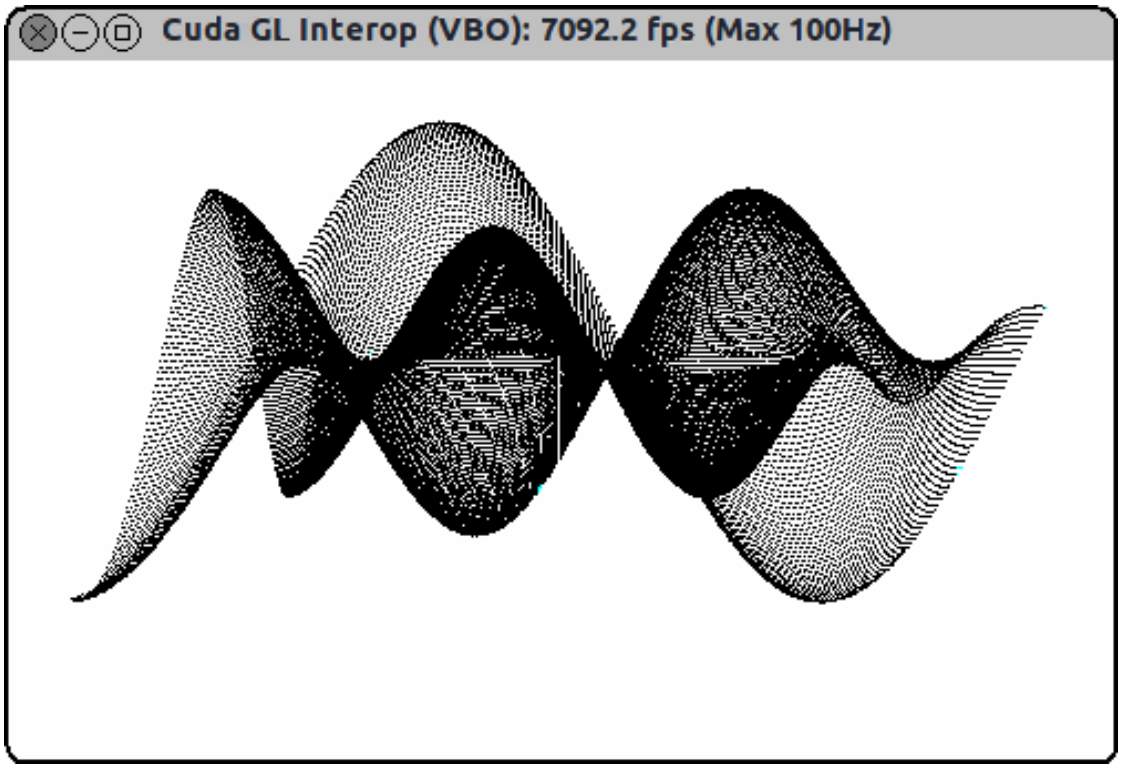
Compile o programa

```
$ make
```

Execute o programa

```
$ ./simpleGL
```

Abrirá uma janela como esta com o exemplo em movimento.



Pronto!

Foi observado que é necessária maior atenção antes de autorizar e aplicar atualizações em sistemas onde se utiliza driver de vídeo diferente do padrão instalado na distribuições Linux escolhida. A partir da versão 10.10 do Ubuntu Linux, foi modificado o gerenciador da interface gráfica. O comando para parar o modo gráfico foi modificado. Agora não é mais necessário efetuar diversos ajustes em arquivos do sistema e várias reinicializações. Basta interromper o funcionamento do gerenciador de interface, aplicar as modificações, reiniciar o gerenciador de interface gráfica e eventualmente reiniciar o sistema.

Roteiro para instalação do CUDA 5.5 no Ubuntu 12.04.3 LTS
TURATTI, L.G., "Guia de instalação do CUDA 5.5". Atualizado em: jul 2013.

Em complemento ao guia para versão 5.0:

Item 2: Obtenha a versão mais recente em: `https://developer.nvidia.com/cuda-downloads`

Item 3: Não é mais necessário modificar o ambiente de trabalho. Basta desativar o gerenciador da interface gráfica para instalação do novo driver de vídeo; reativar a interface ao reiniciar.

```
$ sudo bash
# service lightdm stop
# apt-get remove --purge nvidia*
# apt-get remove --purge xserver-xorg-...-nouveau
# echo "blacklist nouveau" >> /etc/modprobe.d/blacklist.conf
# ./NVIDIA-<driver de video 319.37>.run  :::NAO USE DKMS!:::
# ./cuda-linux<toolkit>
# ./cuda-samples<pack>
# apt-get install g++ g++-4.4
```

Satisfazendo dependencias do SDK:

```
$ sudo apt-get install libgl1-mesa-dev libglu1-mesa-dev mesa-commondev libgl1-mesa-dev
libglu1-mesa-dev libxi-dev libxmu-dev freeglut3 libmotif3
$ sudo ln -s /usr/lib/libglut.so.3 /usr/lib/libglut.so
$ sudo ln -s /usr/lib/libXm.so.6 /usr/lib/libXm.so
```

Para copiar os exemplos, crie um diretório (pasta) e copie os arquivos:

```
$ mkdir NV55SDK
$ cp /usr/local/cuda/samples/* . -R
```

Escolha uma pasta com o exemplo desejado e compile

```
$ make
```

PRONTO!

Referências:

[Ra1] NVIDIA. "GeForce Graphics Processors (GPU)". Disponível em: <http://www.nvidia.com/object/geforce_family.html>. Acesso em: 30 set. 2013.

[Ra2] NVIDIA. "SLI – FAQ GeForce". Disponível em: <http://www.geforce.com/hardware/technology/sli/faq>. Acesso em: 30 set. 2013.

[Ra3] SpaceClockers. Disponível em: <http://spaceclockers7.blogspot.com.br/2012/05/multi-gpu-solution.html>. Acesso em: 30 set. 2013.

[Ra4] GPUBrasil. Disponível em: <http://gpubrasil.com>. Acesso em: 30 set. 2013.

[Ra5] Ubuntu. Disponível em: <http://en.wikipedia.org/wiki/Ubuntu_linux>. Acesso em: 30 set. 2013.

APÊNDICE B – Documentação sobre o programa TRIM/SRIM

SRIM (*Stopping and Range of Ions in Matter*) é uma interface atualizada do TRIM (*Transport of Ions in Matter*) para integrar os programas para computador desenvolvidos por James F. Ziegler et al. Esses programas permitem realizar cálculos relacionados à interação entre átomos, onde o átomo incidente é tratado como íon, enquanto os átomos que compõe a matéria são tratados como alvo. Os cálculos para estimativa do freamento de íons na matéria consideram a interação de íons com a rede cristalina de um determinado material, cujos parâmetros iniciais da simulação são informados pelo usuário do programa. Há possibilidade de avaliar a interação de íons num material constituído por monocamada (somente um tipo de material) ou composto (diversas camadas de materiais). O programa utiliza diversas aproximações físicas e numéricas para obter eficiência elevada, mantendo a precisão dos resultados. Para isto emprega uma fórmula analítica para determinar colisões átomo-átomo (conhecida com "fórmula mágica") e o conceito de caminho livre entre colisões, onde apenas as colisões significativas são avaliadas. São utilizadas aproximações para colisões binárias para permitir o cálculo da profundidade de penetração dos íons além do Método de Monte Carlo para avaliar a distribuição de íons no alvo.

Sua utilização é importante para avaliar os efeitos de íons na matéria. Conforme o interesse do usuário do programa é possível definir o tipo de cálculos que serão realizados. Na distribuição de íos e cálculo rápido de danos (*Íon Distribution and Quick Calculation of Damage*), por exemplo, não são considerados os danos causados ao alvo, ou eventual remoção (*sputtering*) de material do alvo. Neste caso as estimativas da simulação utilizam o formalismo de Kinchin-Pease, onde não há diferença nos resultados, mas apenas a ilustração com os íons primários. Todos os resultados estão disponíveis (como pode ser observado nas tabelas de resultados presentes neste apêndice) e os campos com informação nula (zero) não foram calculados ou estimados nesta opção de cálculo para simulação.

As tabelas de dados foram obtidas com as simulações utilizando os principais tipos de cálculo, o cálculo para distribuição rápida e o cálculo detalhado com colisões em cascata (que contempla também os íons secundários e redistribuição após inúmeras colisões. Esses gráficos e resultados numérico são úteis ao CCS/UNICAMP para auxiliar os pesquisadores a visualizar o comportamento dos íons na matéria e obter estimativas para análise dos resultados experimentais.

Foram selecionados os íons de gálio (Ga) e os materiais carbono (C), platina (Pt) e silício (Si) por serem os mais utilizados nas pesquisas em execução no momento.

A ampliação do código atual do TRIM/SRIM com suporte a GPGPU, pode além de reduzir o tempo necessário para as simulações, estimar melhor os resultados para elaboração de novos modelos para utilização no FIB/SEM, por exemplo.

B.1 – Estrutura do programa

O programa TRIM/SRIM atua como interface principal de diversos módulos desenvolvidos e utilizados desde o surgimento do TRIM. Através dessa interface é possível acessar a documentação para esclarecimento de dúvidas sobre o funcionamento do programa, características e limitações do programa; o módulo SR (*Stopping and Range*) é o gerador de tabelas, capaz de salvar em arquivo texto todas as informações calculadas para o freamento de íons num dado alvo (mono ou multicamada). O módulo TIN é o responsável por gerar o arquivo "`TRIM.IN`" que possui todos os parâmetros necessários para executar a simulação com o módulo TRIM. O fluxograma com essas informações é representado pela figura B.1-1:

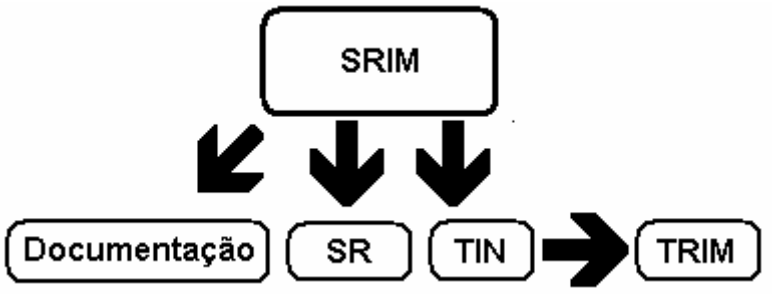


Figura B.1-1: Associação modular das funcionalidades do SRIM

B.2 – Análise do funcionamento do programa

Para utilização essencial do programa, foram testadas as conexões dos módulos e demais arquivos presentes no conjunto de arquivos distribuídos gratuitamente. Esta avaliação foi iniciada para verificar a possibilidade de transformar o programa em um aplicativo portátil.

Foi efetuada uma instalação regular do programa e em uma cópia do diretório onde o programa foi instalado, os arquivos foram modificados ou removidos para avaliar a necessidade e interdependência entre as parte do programa. A figura B.1-2 apresenta a tela inicial do programa nas versões 2008 e 2013. As funções da tela principal estão mapeadas na figura B.1-3.

Figura B.1-2: Tela inicial da versão 2008.04 e 2013 respectivamente

A execução do "`SRIM<versão>.EXE`" apresenta uma tela que é uma interface com atalhos para os demais componentes do pacote SRIM. Sua chamada direta executa as funcionalidades básicas a que o programa se propõe, sem mensagens de erro.

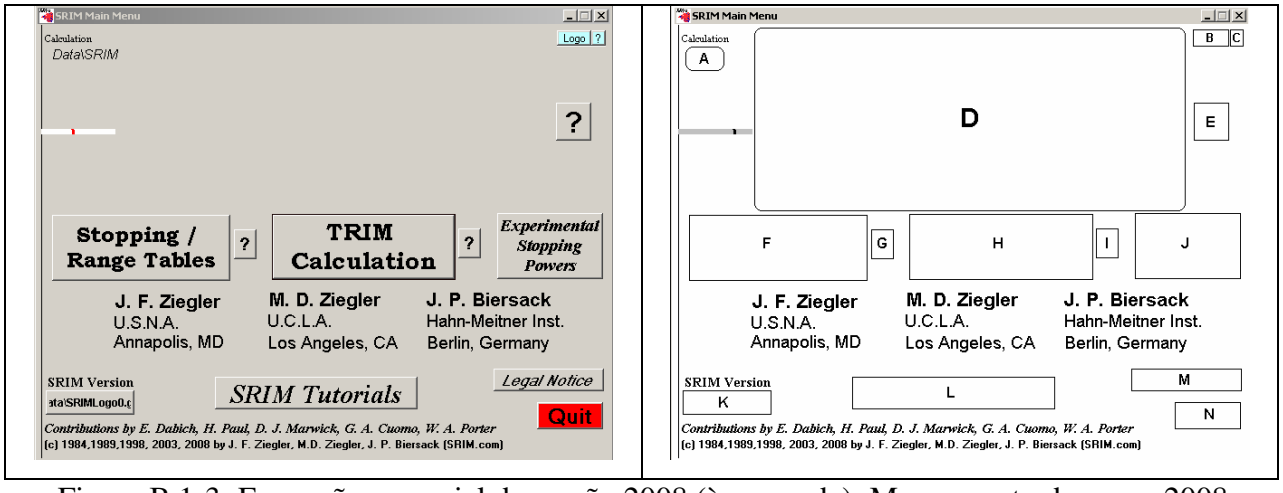


Figura B.1-3: Execução essencial da versão 2008 (à esquerda); Mapeamento do menu 2008.

Analisando o conteúdo da tela inicial, que atua como menu principal do programa, foram mapeadas as funções, representadas na figura B.1-3 (à direita) identificadas por regiões. São elas:

A) Exibe a quantidade de vezes que o programa foi executado (informação armazenada na pasta "`\data\`" no arquivo "`SRIMRun.dat`");

B) O botão "Logo" alterna imagens de logotipo do programa TRIM/SRIM (que estão armazenadas na pasta "`\data\`" com arquivos nomeados sequencialmente "`SRIMLogoX.gif`" onde X é um índice que varia entre 0 e 4). Caso o programa não encontre um desses arquivos é apresentada mensagem de erro (ERRO 53, que significa que o arquivo não foi encontrado ou biblioteca ausente no sistema);

C) Informação sobre a possibilidade de sugerir um novo logotipo para o programa ("*SRIM Logo Contest*" disponível no arquivo "`help22.rtf`" da pasta "`\SRIMHelp\`");

D) Logotipo do programa (imagem obtida na pasta "`\data\`" arquivo "`SRIMLogoX.gif`");

E) Exibe informação sobre SRIM (`\SRIMHelp\help0.rtf`)

F) Fecha este programa e executa "`SR.EXE`"

G) Exibe informação sobre tabelas de freamento (`\SRIMHelp\help17.rtf`)

H) Fecha este programa e executa "`TIN.EXE`"

I) Exibe informação sobre o TRIM (`\SRIMHelp\help8.rtf`);

J) Experimental Stopping Plots exibe imagens de "`\SRIMPICS`" (exibe tudo somente se for a versão completa!);

K) Mostra a versão do programa (`\data\VERSION`) e quando clicado exibe histórico das versões (`\data\Version.rtf`);

L) SRIM Tutorial recomenda a leitura do livro para melhor compreensão (`\SRIM Tutorials\SRIM Tutorials.rtf`);

M) Legal Notice informa sobre a distribuição do programa (`\data\Legal.rtf`);

N) Encerra a execução deste programa.

Os testes com execução do arquivo binário `SRIM.EXE` isoladamente demonstraram algumas interdependências de arquivos. Há necessidade do diretório (pasta) "`\data`" responsável por controlar a quantidade de vezes que o programa foi utilizado (arquivo "`SRIMRun.dat`" armazena o número de utilizações. Quando o conteúdo desse arquivo for o número zero ou um, o arquivo "ReadMe.rtf" é apresentado); armazena informação sobre a versão do programa; armazena dados

sobre átomos para geração das tabelas, solicitado pelo SR.EXE (ATOMDATA e SCOEF3.dat) que fica residente em memória mesmo após o término de sua execução (e gera o arquivo SR.IN); definições da interface da simulação, além de imagens utilizadas pelo programa.

A utilização do "TIN.EXE" gera os arquivos "TRIM.IN" e "TRIMAUTO", que servem com parâmetros para execução do "TRIM.EXE" operando com arquivo em lote (modo console).

Como resultado da execução do "TRIM.EXE", os arquivos "FxxNEW.BIN" são criados em "\data". Toda e qualquer alteração de configurações durante a simulação é salva. Ajustes nas cores dos materiais (íons em movimento e íons parados) são mantidos como configurações para as próximas execuções no arquivo "\data\TRIM.cfg", que possui uma relação pré-definida de cores (em decimal) dos elementos que compõe os gráficos, assim como alguns padrões utilizados comumente, como pode ser observado no exemplo de arquivo de configuração abaixo.

Arquivo: TRIM.cfg

```
01  ----TRIM.cfg : TRIM Color Configuration----
02
03  [Ion Colors - MOVE, STOP]
04   255           0
05  ---http://cores.gratuita.com.br
06  [12 Recoil Colors - MOVE, STOP]
07   1             6487906       837122
08   2             15982183      16417654
09   3             16744703      12519886
10   4             16777088      13814035
11   5             13159423      9673727
12   6             16760769      16711680
13   7             8421376       65280
14   8             6422724       12615935
15   9             8388736       16716287
16   10            4755857       8700353
17   11            9408399       12566463
18   12            255           0
19
20  [Distribution Plot Labels - SIZE, COLOR]
21   6             8388608
22
23  [4 Distribution Plot Colors]
24   255           8388608       65280         16776960
25
26  [Plot Background, Outer Box, Axes, Layer Lines, Layer Names]
27  16777215  255  0  0  0
28
29  Maximum number of secondary recoils (default=20000)
30   20000
31  Note: 100k recoils requires 2MB of memory, 1M recoils requires 35MB.
32
```

A seguir um exemplo de um elemento do arquivo de configurações onde os valores foram trazidos para o padrão RGB (red, green, blue) e suas respectivas representações no gráfico (verde: tom mais claro e escuro, respectivamente; utilizados para “moving atoms” e “stopped atoms” respectivamente).

```
07    1              #62FF62         #0CC602
```

Assim como representado na linha sete do arquivo “`TRIM.cfg`”, existem pré-configurações para cores que irão compor os gráficos, em até doze camadas.

A figura B.1-4 mostra a simulação em execução em diferentes versões do programa. Quando padrão é um fundo escuro, fica mais difícil ajustar os parâmetros da imagem da simulação e prejudica a obtenção da figura para utilização em publicações, entretanto, a versão mais recente traz alguns recursos de geração de gráficos em 3D.

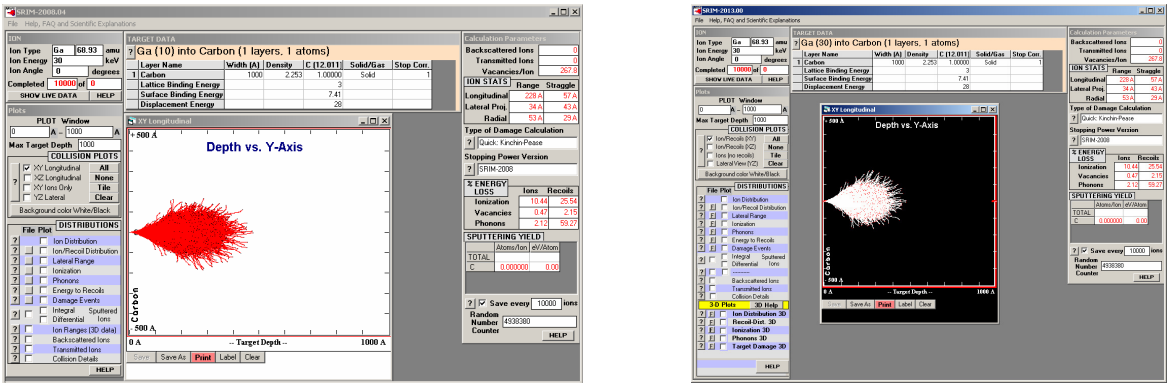


Figura B.1-4: Simulação em TRIM 2008 e 2013, respectivamente.

B.3 – Estrutura de diretórios do programa SRIM

```
[SRIM]            (Pasta principal com os arquivos do SRIM)
+-[Data]
| +- ATOMDATA     (dados da tabela periódica [PT] usada no TRIM)
| +- Clouds.bmp   (imagem de fundo do programa)
| +- Compound.dat (tabela de dados compostos usados no TRIM)
| +- Fxxnew.bin   (dados binários ???)
| +- Legal.rtf    (informações sobre o programa, distribuição,etc)
| +- LineDraw.ttf (fonte de texto utilizada no programa)
| +- S.bmp        (imagem estrela, usada no TIN > Compound Materials)
| +- SCOEF03.dat  (valores ajustados dos elementos [PT])
| +- SNUC03.dat   (relação de valores de possíveis combinações [PT])
| +- SRIM.ico     (ícone do programa)
| +- SRIMLogoX.gif(logotipos)
| +- SRIMRun.dat  (Qtde de vezes que o TRIM foi executado)
| +- TRIM.cfg     (configurações de layout de saída de dados; cores)
| +- VERSION      (informações sobre a versão do programa)
| +- Version.rtf  (detalhes sobre as versões do programa; bugfixes)
|
+-[Demo]          (arquivos para demonstração do TRIM)
| +- Trim.d00 a Trim.d11
|
+-[SR Module]     (programa SR para modo console)
| +- HELP-SEM.rtf (breve manual de operação)
| +- SCOEF03.dat
| +- SNUC03.dat
| +- SR.IN        (exemplo de arquivos para geração de tabelas)
| +- SRModule.exe (programa para modo console em MS-DOS)
| +- VERSION
|
+-[SRIM Outputs]  (local padrão para armazenar resultados)
| +- SRIM Outputs.rtf (ajuda sobre arquivos gerados aqui)
|
+-[SRIM Restore]  (local padrão para salvar dados da última simulação realizada)
| +- SRIM Restore.rtf (ajuda sobre arquivos gerados aqui)
|
+-[SRIM Tools]    (complementos para TRIM)
| +- TRIM*.DAT (exemplos de uso de outros tipos de cálculos para simulação)
| +- *.BAS (componentes para leitura de parâmetros para simulação em basic)
|
+-[SRIM Tutorials] (material didático para aprendizado sobre o simulador)
| +- SRIM Tutorial 1.pdf (Ion Ranges, Doses and Damage)
| +- SRIM Tutorial 2.pdf (Mixing and Sputtering)
| +- SRIM Tutorial 3.pdf (Building Complex Targets)
| +- SRIM Tutorial 4.pdf (Target Damage)
| +- SRIM Tutorials.rtf (Indice com informações sobre os tutoriais)
|
+-[SRIM Help]     (Arquivos de ajuda do programa)
| +- Help*.rtf (Informações para ajuda; o que é o programa; o que faz; comandos)
|
+-[SRIM PICS]     (Imagens com resultados de experimentos)
| +-[STOP01]- STOP0113.GIF (Hydrogen into Aluminum)
| +-[STOP02]- STOP0213.GIF (Helium into Aluminum)
| +-[STOP03]- STOP0313.GIF (Lithium into Aluminum)
| +-[STOP-IONS]- STOP13XX.GIF (Aluminum into ALL SOLIDS)
| +-[STOP-TGTS]- STOPXX13.GIF (Be(4) thru U(92) into Aluminum)
|
+- SRIM ReadMe.*  (Arquivo para orientação inicial em alguns idiomas)
+- *.EXE          (Programas que compõe o SRIM)
+- HELP-TRIM Output.pdf (Capítulo 9 do livro, Informações sobre resultados simulados)
+- HELP TRIM Setup and Input.pdf (Capítulo 8 do livro, Introdução sobre simulações)
```

B.4 – Ajustando o SRIM para uso como programa portátil.

Com o objetivo de executar o programa em qualquer computador para verificar o tempo necessário com as simulações e ter um parâmetro entre diferentes processadores, foi preciso ajustar a maneira de como o programa funciona, contornando a necessidade de privilégio administrativo para instalação em diferentes computadores. Os resultados obtidos e as condições utilizadas nas simulações estão descritos na tabela 4.2.

Foi criado um diretório (pasta) temporário (`\tmp`) para extrair o arquivo de distribuição (SFX) do programa antes da instalação, onde o arquivo obtido na Internet foi copiado para o diretório temporário e foi renomeado de "`srim2008.e`" para "`srim2008.exe`". A execução deste arquivo extraiu todos os componentes necessários para instalação no diretório temporário. O passo seguinte foi instalar de fato o SRIM versão 2008 no diretório raiz do sistema ("`C:\SRIM_Portable`"). Este diretório serviu como base para preparação da versão portátil.

O passo seguinte a instalação foi a expansão das bibliotecas e demais componentes vinculados ao módulo executável (que estão no diretório temporário) e copiar tais arquivos para o diretório onde o programa foi instalado. A expansão desses componentes utilizou o programa "`expand`", do próprio Windows. A descompressão das bibliotecas foi feita informando o arquivo de origem e o arquivo destino. Esta operação tem a seguinte sintaxe:

```
C:\>expand  arquivo_origem.ex_  arquivo_destino.ext
```

Cuja execução real seria, por exemplo:

```
C:\tmp\expand asycfilt.dl_ asycfilt.dll
```

A relação de arquivos necessários para versão portátil:

```
\SRIM_Portable
\SRIM_Portable\asycfilt.dll           120.592 15/01/1997 02:00
\SRIM_Portable\comcat.dll              22.288 31/10/1996 02:00
\SRIM_Portable\comctl32.ocx           608.448 22/05/2000 20:58
\SRIM_Portable\comdlg32.ocx           140.488 22/05/2000 18:58
\SRIM_Portable\ctl3d32.dll             27.136 21/08/1996 02:00
\SRIM_Portable\linedraw.ttf            80.008 18/10/1993 03:00
\SRIM_Portable\msflxgrd.ocx           244.416 22/05/2000 18:58
\SRIM_Portable\msvbvm50.dll         1.355.776 18/08/2001 11:00
\SRIM_Portable\oleaut32.dll           491.792 15/01/1997 02:00
\SRIM_Portable\olepro32.dll            32.528 15/01/1997 02:00
\SRIM_Portable\richtx32.ocx           212.240 09/03/2004 19:45
\SRIM_Portable\SRIM 2008.exe.local          0 26/09/2013 18:07
\SRIM_Portable\stdole2.tlb             16.896 15/01/1997 02:00
\SRIM_Portable\tabctl32.ocx           209.192 30/08/2001 18:43
```

Onde além da expansão das bibliotecas foi necessário criar um arquivo vazio com o nome da interface principal acrescida da extensão "`.local`". Isto faz o programa procurar pelas bibliotecas no mesmo diretório onde é executado, antes de procurar em outros diretórios do sistema operacional. Concluída a preparação, basta copiar o diretório "\SRIM_Portable" para um dispositivo móvel.

B.5 – Características das versões testadas

```
versao 2008.04
- Disponivel até hoje para download
- Completa e estável (revisada em 2010 e 2012)
- Graficos com fundo branco, ions em vermelho (QCD)
- Requer instalação no Windows
- Não traz bibliotecas complementares (instalação manual mais dificil)
- Versao padrao 200MB


versao 2010
SRIM2010_Setup.exe... SFX  9.930.501bytes  9.9MB
Pasta SRIM2010..... ... 10.348.983bytes  9.9MB
SRIM2010 Instalado... ...  9.047.513bytes 10.0MB
StoppingPlots-ALL.exe SFX 24.600.032bytes 23.5MB
Pasta SRIMPICS..... ... 24.429.192bytes 23.2MB


versao 2011.02 obtida no IBA2011 em junho/julho (Full 72MB)
--Nao requer instalacao; basta descompactar pacote.
--mudanca de opcoes do menu damage
--a partir desta versao foram invertidas cores e graficos;
--fundo em preto; ions em branco; sao piores para utilizar.
--nao salva preferencia de outputs dos graficos;
--trabalhosa conversao das imagens para uso
--introducao de 3D-Plots
--trouxe mais documentacao sobre uso
--trouxe pasta SRPICS com 23MB de resultados
--trouxe pasta SR-Setup com 5MB para XP e W7


versao 2012.03
--ampliada quantidade de tutoriais
--Versao padrao 19MB
--Versao completa 41MB


versao 2013.00 (atual disponivel no website) 24.jun.2013
--Versao padrao 20MB
--Versao completa 46MB
```

B.6 – Tabelas com resultados numéricos e gráficos obtidos com SRIM

A definição da maneira como será conduzido um experimento pratico pode ser orientada com a analise dos resultados de diversas simulações numericas. Para efetuar uma simulação utilizando o programa TRIM/SRIM [13] é necessário informar alguns parâmetros de entrada no programa, que serão utilizados para realização dos cálculos desejados. A figura A.6-1 ilustra a tela de configurações do programa "`TRIM.EXE`".

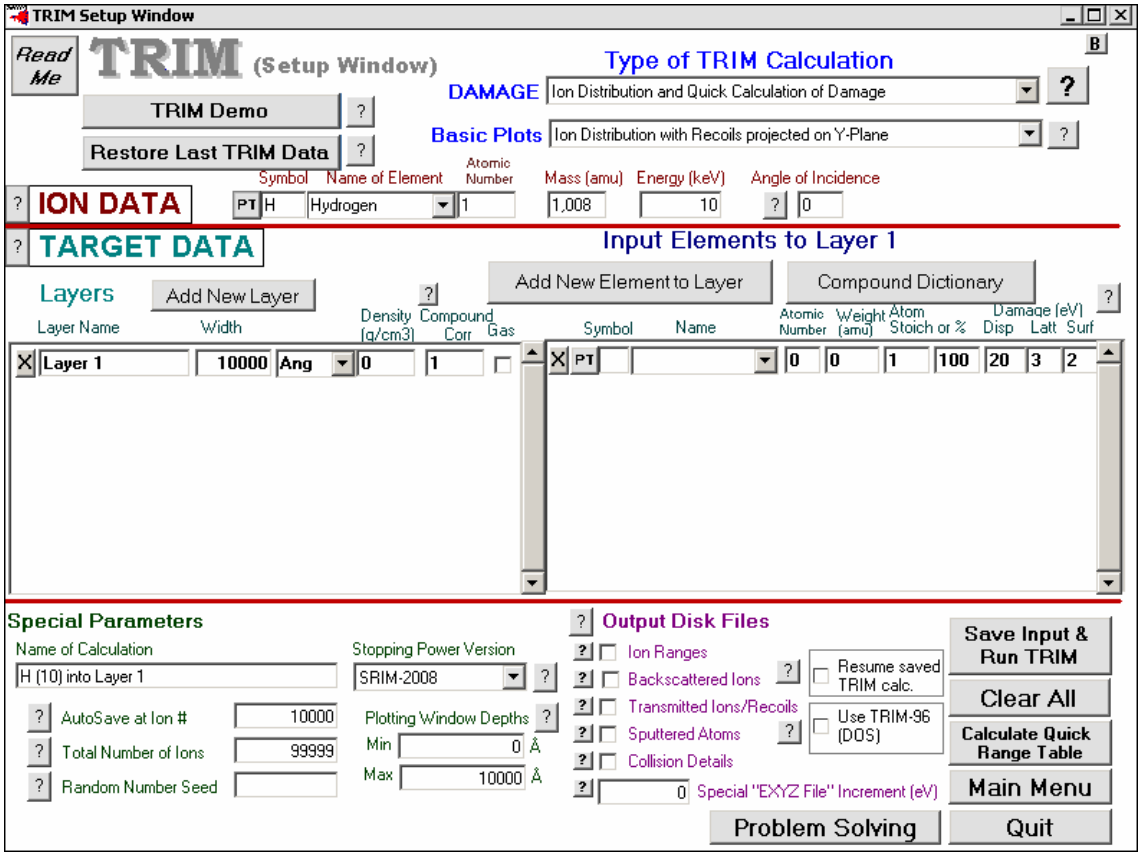


Figura A.6-1: Configuração dos parâmetros da simulação

Nesta tela do programa são definidas algumas condições de entrada, tais como:

DAMAGE (Dano)

1) Ion Distribution and Quick Calculation of Damage (Cálculo rápido de colisões e distribuição de íons)

2) Detailed Calculation with full Damage Cascades (Cálculo detalhado com colisões em cascata)

3) Surface Sputtering / Monolayer Collision Steps (similar ao 2 para camada única de um mesmo material)

As demais opções precisam do arquivo "TRIM.DAT" que possui informações sobre a cinética envolvida na interação entre os átomos do feixe e do alvo. Esse arquivo pode ser gerado utilizando o TRIM com uma das três opções mencionadas e adequando o arquivo "TRIMOUT.txt" gerado pela simulação. Na pasta "\SRIM\SRIM Tools" existem dois modelos para uso em testes. Para usar o arquivo "TRIM.DAT", ele deve estar na mesma pasta (diretório) onde o estiver o programa "TRIM.EXE".

Arquivo: TRIM.DAT

```
01  ########### TRIM with various Incident Ion Energies/Angles and Depths #########
02  # Top  10 lines are user comments,  with  line  #8 describing experiment.     #
03  # Line #8 will be written into all TRIM output files ( various files:  *.TXT).#
04  # Data Table line consist of: EventName(5 char)+8 numbers separated by spaces.#
05  # The Event Name consists of any 5  characters  to identify that line.        #
06  # Cos(X) = 1 for normal incidence, and Cos(X) = -1  for backwards.            #
07  ################ Typical Data File is shown below  ############################
08  ## Ar Plasma Ions into Si (1000A thick) (Energies 20-80 keV, Various Angles) ##
09  Event  Atom  Energy  Depth   Lateral-Position   ----- Atom Direction ----
10  Name   Numb   (eV)    _X_(A)   _Y_(A)  _Z_(A)   Cos(X)   Cos(Y)   Cos(Z)
11  A-1    18    12345    0        0       0        1.00000  .000000 -.000000
12  abcde  18    54321    0        0       0        0.62344 -.295513  .003415
13  AA#1   18    1.31E4   123      0      -154      0.34234 -.336437 -.017437
14  C-3    18    123.55   1230     432     12.3E2  -0.23258 -.543453  .443483
15  A-1    18    0.123E2  0       -10     -12       0.99998  .000012 -.000017
16
```

Basic Plots (Gráficos):

1) Ion Distribution with Recoils projected on Y-Plane (Distribuição de íons primários, secundários e resultados projetados no plano Y - usado como padrão). As demais opções podem complementar a visualização dos resultados. Estes gráficos são gerados durante a simulação. Podem ser salvos como figuras para uso posterior, em análise de resultados como utilizado após as tabelas com resultados das simulações a seguir.

ION DATA (íon incidente):

Galio (Ga+), por exemplo. Pode ser informada a sigla do elemento químico ou através da escolha por nome do elemento, informada pelo usuário.

Energy (Energia em KeV): 30

Angle of incidence (Angulo de incidência): 0

O ângulo informado é o do feixe de íons em relação ao Alvo, cujas referências são ilustradas nas figuras A.6-2, A.6-3 e A.6-4. O programa utiliza como referência o ajuste de ângulo em sentido horário para incidência dos histórico de íons (feixe). Quando indicado zero graus, equivale à incidência perpendicular a amostra (figura A.6-2), assim como os ângulos utilizados (80º e 88º, figuras A.6-3 e A.6-4, respectivamente).

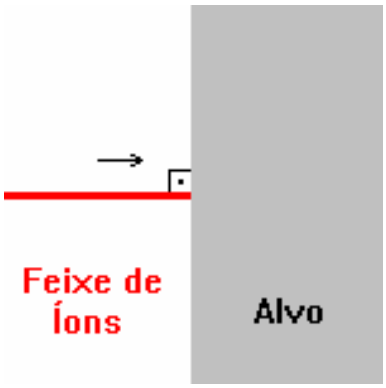


Figura A.6-2: Feixe de íons em ângulo 0º

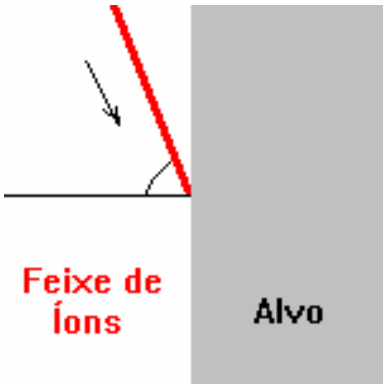


Figura A.6-3: Feixe de íons em ângulo 80º

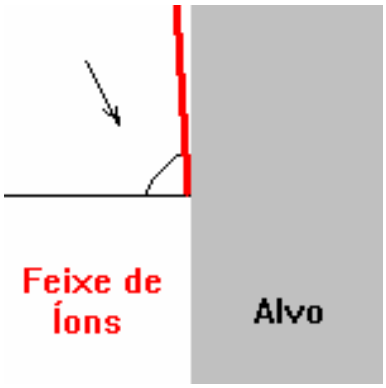


Figura A.6-4: Feixe de íons em ângulo 88º

TARGET DATA (Alvo):

São as informações que compõe o material atingido pelo feixe de íons. Esse material pode ser monocamada ou possuir diversas camadas. Cada camada pode ser um elemento simples (Si) ou composto (SiO_2, por exemplo). O preenchimento de camadas é feito em dois quadros. O quadro da esquerda recebe um nome dado para camada (*alias*), seguido da espessura (*width*, dada pelo intervalo que pode ser entre Angstrons até Km), enquantro no quadro a direita é especificada a composição da camada.

Utilizamos nas simulações monocamada de: Carbono, Silício e Platina.

Total Number of Íons (Número total de íons): 100000 (1.0E+05).

Plotting Window Depths (ajuste da janela de resultados gráficos):

Aqui é possível ajusta o tamanho da janela onde há projeção dos resultados, permitindo melhorara a visualização para camadas mais finas (usadas no capítulo 4, figuras 4.8, 4.9 e 4.10).

Output Disk Files (arquivos com resultados):

Permite definir quais arquivos com resultados numéricos serão salvos ao final dos cálculos. Caso não seja especificado, é criado um conjunto de arquivos mínimo, capaz de continuar uma simulação que foi salva.

Feito esse preenchimento, o usuário deve clicar em "*Save Input & Run TRIM*" para iniciar a simulação (figura A.6-1).

Conforme as condições desejadas para simulação, após algum tempo, os resultados obtidos estarão disponíveis em uma janela como ilustrado na figura A.6-5, ao término da simulação.

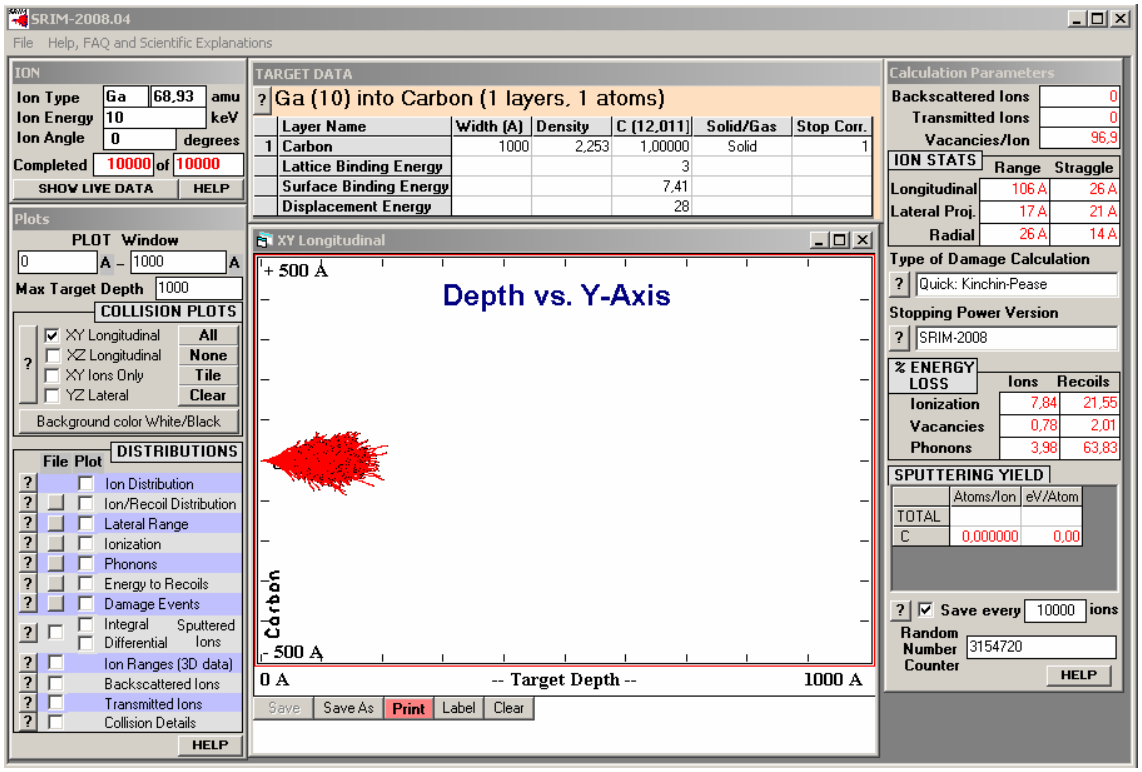


Figura A.6-5: Resultados de uma simulação com TRIM.

O usuário será questionado ao fina da simulação ("*End of calculation*") sobre o que deseja fazer, onde é permitido salvar os dados (*Yes*) ou aumentar o número de íons informado (*Increase number of ions*). Caso o usuário escolha não, é possível informar o novo número total de íons para simulação. "*OK*" continua a simulação, enquanto "*Cancel*" não efetua modificações. A seguir uma janela permite armazenar os resultados da simulação. "*Store in SRIM directory (default)*" armazenará os arquivos na pasta "`\SRIM\SRIM Outputs`", a opção "*Store in another Directory/Disk*" permite ao usuário indicar aonde salvar os arquivos ou então "*Continue without*

Saving TRIM" ignora o salvamento de dados. Há ainda a opção de salvar informações da simulação ao fechar a janela do TRIM.

Os resultados tabulados nas próximas páginas possuem os parâmetros de cada simulação realizada, com as seguintes informações para cada conjunto de dados:

ION STATS, que são os parâmetros de espalhamento: **longitudinal**, que refere-se a maior distância média alcançada pela maioria dos íons (dados sobre alcance e desvio no eixo X do gráfico 3D, ilustrado pela figura A.6-6); **lateral**, que refere-se a distância média inicial onde ocorre a abertura do feixe (dados sobre alcance e desvio no eixo Y do gráfico 3D, ilustrado pela figura A.6-7) e **radial**, que refere-se a distância inicial média da abertura do feixe (dados sobre alcance e desvio da abertura do "cone", ilustrado pela figura A.6-8).

ENERGY LOSS, que são as perdas de energias, conforme o modelo de Bohr, ou seja, são os efeitos do íon incidente sobre um dado material (freamento). A ionização é a formação de íons como resultado de colisões de partículas, ou seja, um átomo cuja quantidade de elétrons é diferente do total de prótons ou ainda um *recoil* é o íon que "ricocheteia" através do movimento adquirido pela partícula que emite um fóton; vacâncias são os defeitos causados na rede cristalina pela perda de um átomo; e um *phonon* é a excitação de partículas de um material; Movimento/Vibração causada enquanto não encontra equilibrio da rede cristalina.

SPUTTERING YIELD, é o número de átomos removidos por íon incidente que atinge o material. Quando a simulação completa é realizada (*damage,* ou seja, opção de danos do tipo 2, *Full Damage Cascades*), estes dados aparecerão nos resultados. Para o parâmetro "*Atoms/íon*" e "*eV/Atom*" teremos zero como resultado quando a simulação utilizar o cálculo rápido (opção de danos do tipo 1, ou seja, somente íons primários).

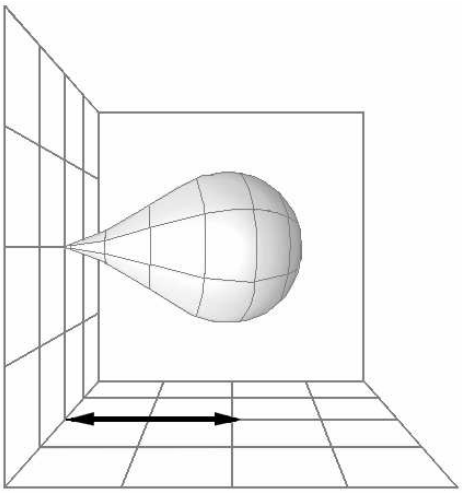


Figura A.6-6: espalhamento longitudinal

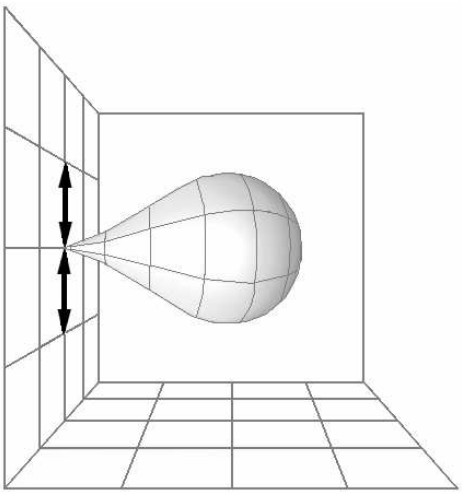


Figura A.6-7: espalhamento lateral

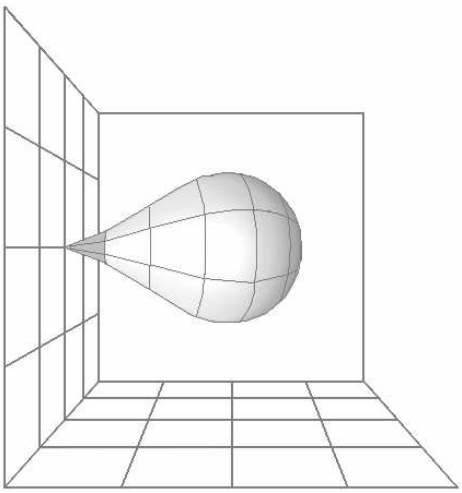


Figura A.6-8: espalhamento radial

SRIM/TRIM: Cálculo rápido de colisões e distribuição de Íons

MATERIAL ION: Galio (Gallium) TARGET: Carbono (Carbon) WIDTH: 1000A = 100nm

Energia	Número Total de Íons	Ângulo	Parâmetros de espalhamento						Perdas de energia						Resultados			
			Longitudinal		Lateral		Radial		Ionização		Vacâncias		Phonons		Sputtering Yield		Figura	Tempo
			Alcance	Desvio	Alcance	Desvio	Alcance	Desvio	Íons	Recoils	Íons	Recoils	Íons	Recoils	Atoms/Ion	eV/Atom		
10KeV	10E1	0	103A	25A	17A	21A	24A	17A	7,72	21,35	0,78	2,00	4,24	63,90	0,00	0,00	GaC 01	0'01"
		80	37A	21A	101A	104A	102A	26A	8,48	21,31	0,76	2,00	4,17	63,28	0,00	0,00	GaC 02	0'01"
		88	25A	18A	116A	122A	118A	37A	8,10	21,34	0,78	2,01	3,97	63,79	0,00	0,00	GaC 03	0'01"
	10E2	0	105A	27A	19A	24A	27A	16A	7,80	21,58	0,78	2,01	4,08	63,76	0,00	0,00	GaC 04	0'02"
		80	24A	16A	106A	108A	108A	24A	8,02	21,46	0,77	2,03	3,85	63,87	0,00	0,00	GaC 05	0'02"
		88	22A	17A	103A	108A	105A	31A	8,12	22,31	0,71	2,09	3,31	63,46	0,00	0,00	GaC 06	0'02"
	10E3	0	105A	26A	16A	21A	25A	14A	7,82	21,58	0,78	2,01	3,96	63,84	0,00	0,00	GaC 07	0'19"
		80	27A	17A	105A	108A	107A	25A	7,97	21,73	0,76	2,03	3,74	63,76	0,00	0,00	GaC 08	0'16"
		88	22A	15A	105A	109A	108A	26A	8,06	22,20	0,71	2,08	3,34	63,60	0,00	0,00	GaC 09	0'11"
	10E4	0	106A	26A	17A	21A	26A	14A	7,84	21,55	0,78	2,01	3,98	63,83	0,00	0,00	GaC 10	3'00"
		80	27A	16A	105A	108A	108A	25A	7,97	21,71	0,76	2,03	3,73	63,79	0,00	0,00	GaC 11	2'30"
		88	22A	14A	106A	109A	108A	26A	8,16	22,21	0,71	2,08	3,33	63,50	0,00	0,00	GaC 12	1'38"
	10E5	0	106A	26A	17A	21A	26A	14A	7,90	21,51	0,78	2,01	3,98	63,82	0,00	0,00	GaC 13	27'15"
		80	26A	16A	105A	109A	108A	26A	7,98	21,72	0,77	2,03	3,72	63,78	0,00	0,00	GaC 14	25'08"
		88	21A	14A	107A	110A	109A	26A	8,18	22,19	0,72	2,08	3,34	63,50	0,00	0,00	GaC 15	16'42"
20KeV	10E1	0	172A	34A	36A	41A	45A	18A	10,12	22,92	0,61	2,11	2,62	61,61	0,00	0,00	GaC 16	0'01"
		80	24A	13A	166A	170A	170A	37A	9,72	24,00	0,58	2,10	2,65	60,95	0,00	0,00	GaC 17	0'01"
		88	27A	29A	189A	192A	190A	38A	10,52	23,76	0,56	2,14	2,13	60,89	0,00	0,00	GaC 18	0'01"
	10E2	0	167A	39A	30A	39A	44A	23A	9,28	23,95	0,57	2,14	2,69	61,37	0,00	0,00	GaC 19	0'03"
		80	34A	25A	172A	178A	176A	45A	9,67	23,78	0,58	2,12	2,69	61,16	0,00	0,00	GaC 20	0'03"
		88	27A	20A	177A	182A	180A	40A	9,81	24,47	0,53	2,15	2,38	60,66	0,00	0,00	GaC 21	0'02"
	10E3	0	169A	40A	26A	33A	39A	22A	9,24	24,06	0,57	2,13	2,66	61,33	0,00	0,00	GaC 22	0'21"
		80	43A	27A	166A	171A	170A	40A	9,33	24,29	0,55	2,15	2,51	61,17	0,00	0,00	GaC 23	0'19"
		88	31A	22A	172A	177A	175A	42A	9,69	24,66	0,51	2,17	2,30	60,67	0,00	0,00	GaC 24	0'14"
	10E4	0	170A	42A	26A	33A	40A	22A	9,31	23,99	0,57	2,13	2,67	61,33	0,00	0,00	GaC 25	3'42"
		80	42A	26A	168A	173A	171A	42A	9,42	24,18	0,55	2,15	2,52	61,18	0,00	0,00	GaC 26	3'16"
		88	32A	21A	170A	175A	174A	42A	9,73	24,68	0,51	2,17	2,27	60,64	0,00	0,00	GaC 27	2'04"
	10E5	0	170A	42A	26A	33A	41A	22A	9,33	24,00	0,57	2,13	2,67	61,29	0,00	0,00	GaC 28	33'27
		80	41A	26A	168A	173A	171A	42A	9,44	24,20	0,56	2,14	2,51	61,15	0,00	0,00	GaC 29	29'10"
		88	33A	22A	170A	175A	173A	42A	9,72	24,70	0,52	2,16	2,28	60,63	0,00	0,00	GaC 30	18'20"
30KeV	10E1	0	251A	40A	31A	39A	54A	24A	12,77	23,65	0,53	2,07	2,17	58,81	0,00	0,00	GaC 31	0'01"
		80	60A	43A	228A	231A	235A	41A	11,10	25,12	0,48	2,12	2,24	58,94	0,00	0,00	GaC 32	0'01"
		88	128A	- A	233A	- A	294A	- A	13,68	23,98	0,45	2,10	1,97	57,81	0,00	0,00	GaC 33	0'01"
	10E2	0	227A	56A	40A	48A	59A	33A	10,62	25,26	0,48	2,15	2,14	59,35	0,00	0,00	GaC 34	0'03"
		80	55A	32A	239A	245A	243A	56A	11,15	25,25	0,47	2,13	2,10	58,89	0,00	0,00	GaC 35	0'03"
		88	45A	26A	232A	239A	236A	57A	11,05	26,29	0,43	2,16	1,77	58,30	0,00	0,00	GaC 36	0'02"
	10E3	0	227A	55A	32A	40A	51A	29A	10,36	25,61	0,47	2,15	2,12	59,28	0,00	0,00	GaC 37	0'26"
		80	56A	34A	225A	232A	229A	56A	10,46	25,88	0,46	2,16	2,01	59,04	0,00	0,00	GaC 38	0'22"
		88	45A	28A	232A	240A	237A	59A	11,08	26,24	0,46	2,15	1,84	58,26	0,00	0,00	GaC 39	0'15"
	10E4	0	228A	57A	34A	43A	53A	29A	10,44	25,54	0,47	2,15	2,12	59,27	0,00	0,00	GaC 40	4'02"
		80	55A	35A	226A	233A	230A	56A	10,64	25,67	0,46	2,15	2,02	59,05	0,00	0,00	GaC 41	3'42"
		88	44A	29A	229A	236A	234A	57A	10,98	26,18	0,42	2,16	1,84	58,41	0,00	0,00	GaC 42	2'36"
	10E5	0	228A	57A	34A	43A	53A	29A	10,47	25,53	0,47	2,15	2,13	59,25	0,00	0,00	GaC 43	39'00"
		80	55A	34A	226A	233A	230A	57A	10,61	25,72	0,46	2,15	2,02	59,04	0,00	0,00	GaC 44	36'20"
		88	43A	29A	229A	236A	233A	58A	11,00	26,20	0,43	2,16	1,84	58,38	0,00	0,00	GaC 45	21'40"

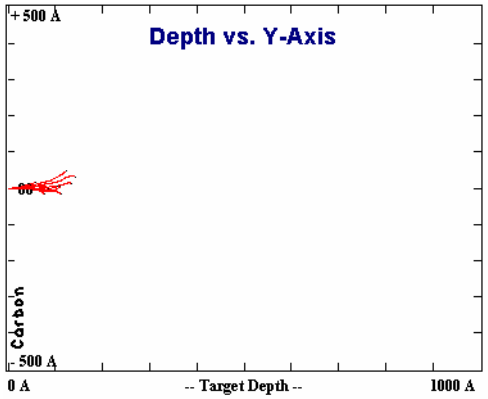


Figura GaC01: 1000A 10KeV 10E1 0º

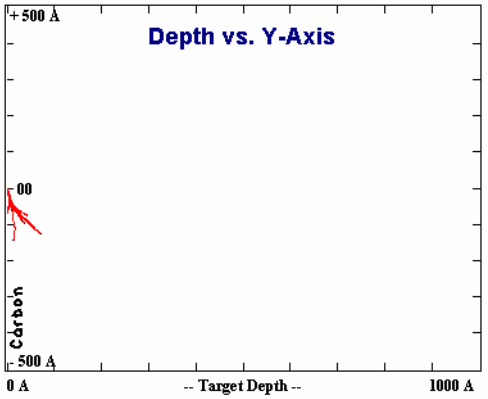


Figura GaC02: 1000A 10KeV 10E1 80º

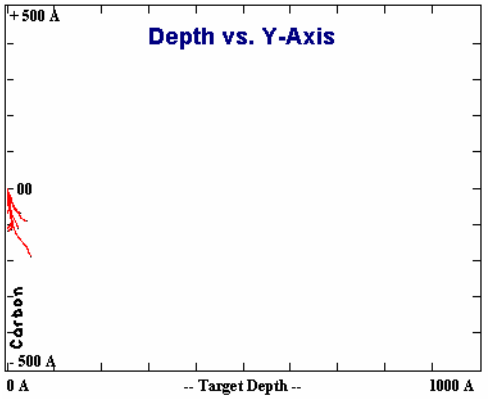


Figura GaC03: 1000A 10KeV 10E1 88º

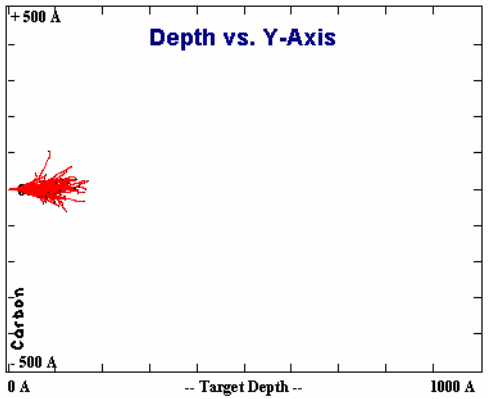


Figura GaC04: 1000A 10KeV 10E2 0º

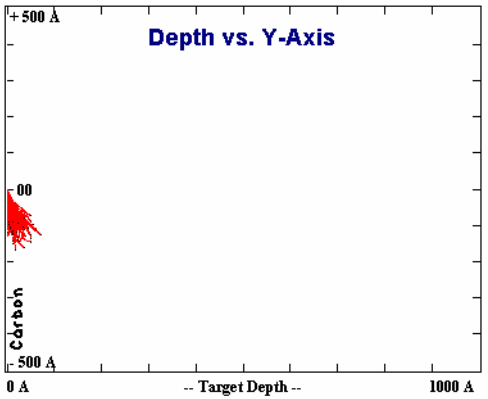


Figura GaC05: 1000A 10KeV 10E2 80º

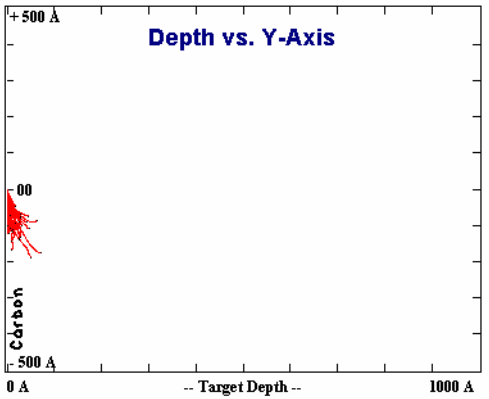


Figura GaC06: 1000A 10KeV 10E2 88º

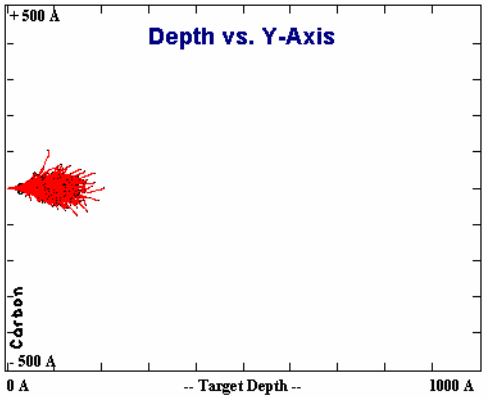


Figura GaC07: 1000A 10KeV 10E3 0º

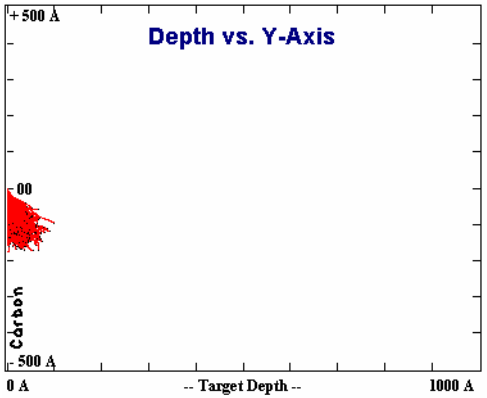


Figura GaC08: 1000A 10KeV 10E3 80º

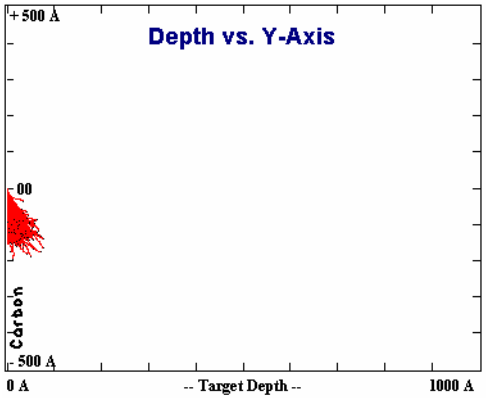


Figura GaC09: 1000A 10KeV 10E3 88º

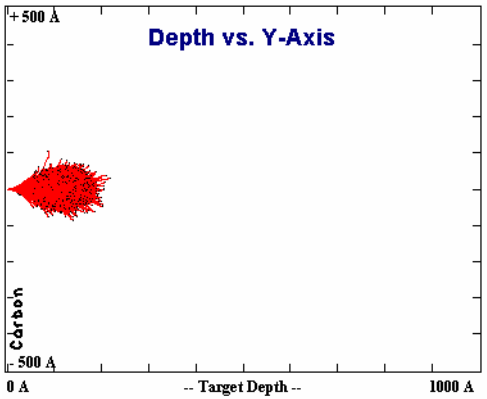


Figura GaC10: 1000A 10KeV 10E4 0º

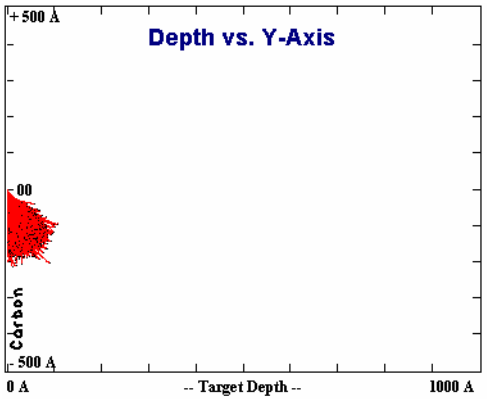


Figura GaC11: 1000A 10KeV 10E4 80º

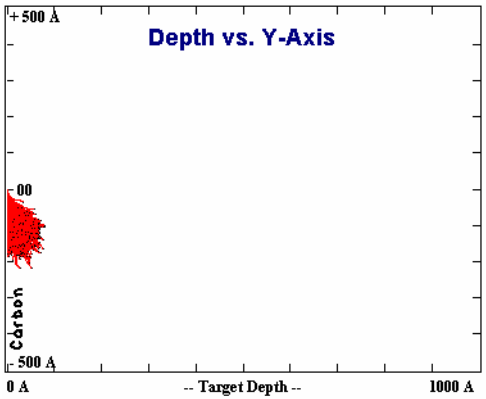


Figura GaC12: 1000A 10KeV 10E4 88º

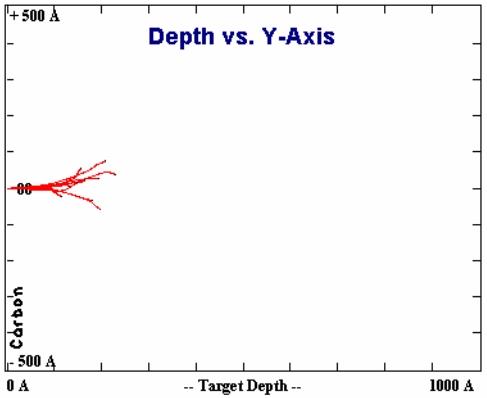


Figura GaC16: 1000A 20KeV 10E1 0º

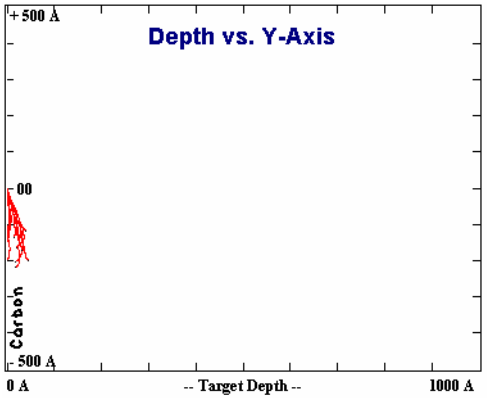


Figura GaC17: 1000A 20KeV 10E1 80º

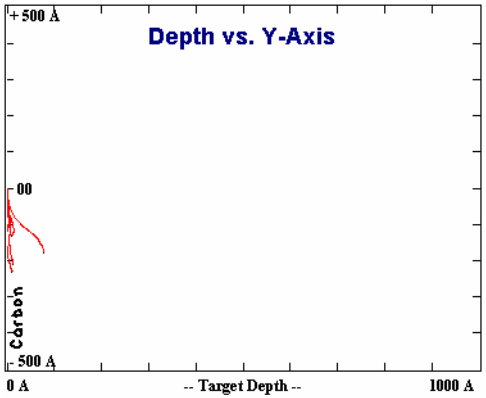


Figura GaC18: 1000A 20KeV 10E1 88º

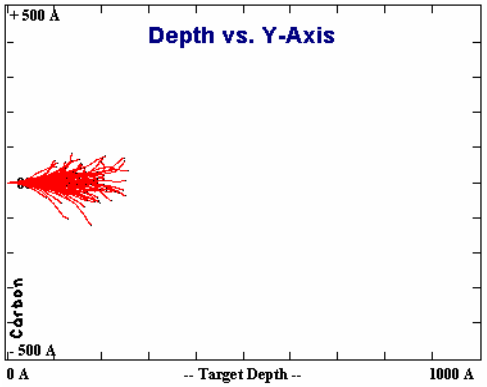


Figura GaC19: 1000A 20KeV 10E2 0º

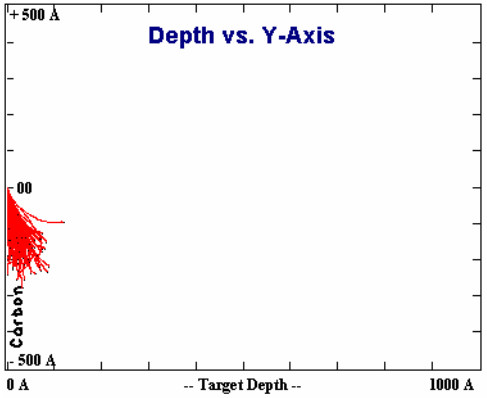


Figura GaC20: 1000A 20KeV 10E2 80º

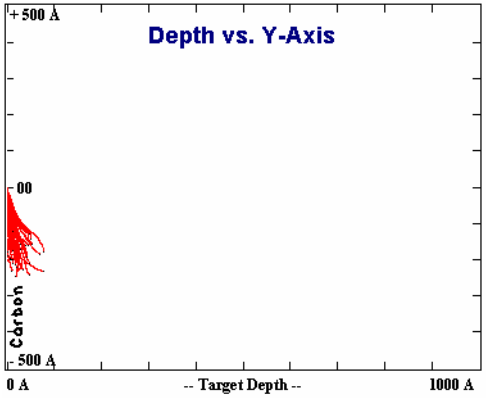


Figura GaC21: 1000A 20KeV 10E2 88º

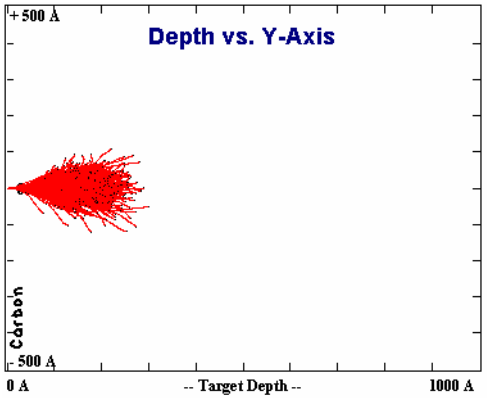


Figura GaC22: 1000A 20KeV 10E3 0º

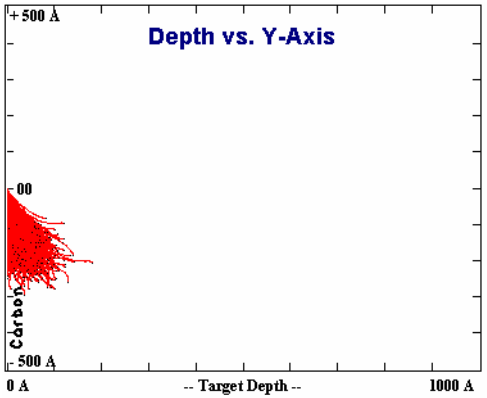


Figura GaC23: 1000A 20KeV 10E3 80º

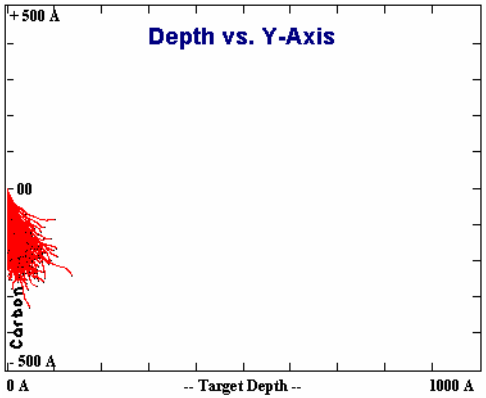


Figura GaC24: 1000A 20KeV 10E3 88º

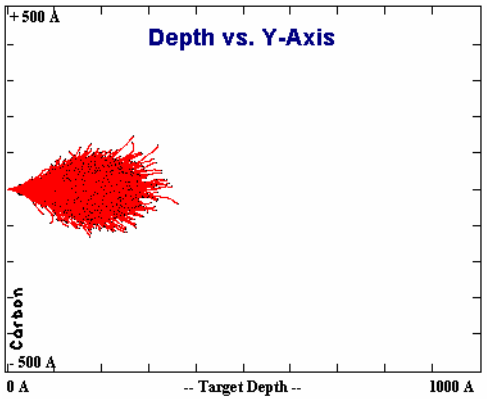


Figura GaC25: 1000A 20KeV 10E4 0º

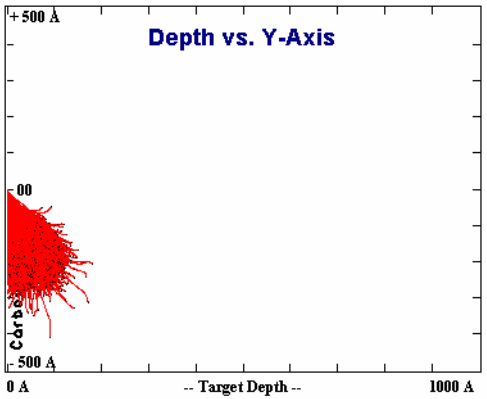


Figura GaC26: 1000A 20KeV 10E4 80º

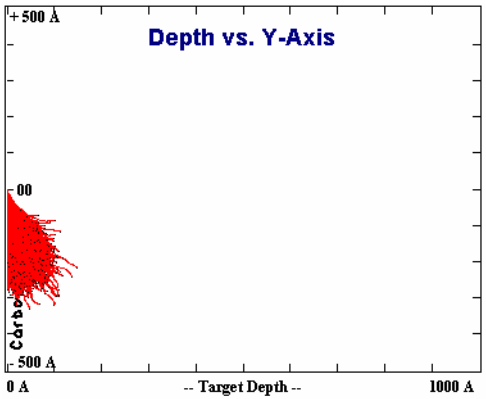


Figura GaC27: 1000A 20KeV 10E4 88º

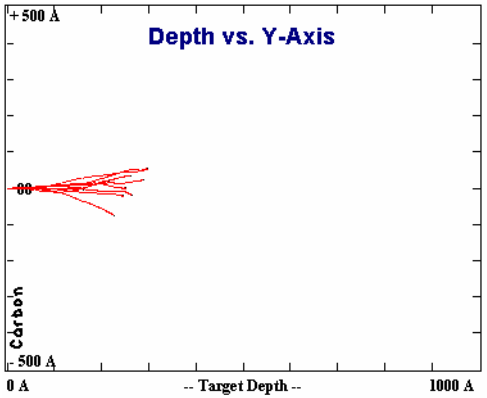


Figura GaC31: 1000A 30KeV 10E1 0º

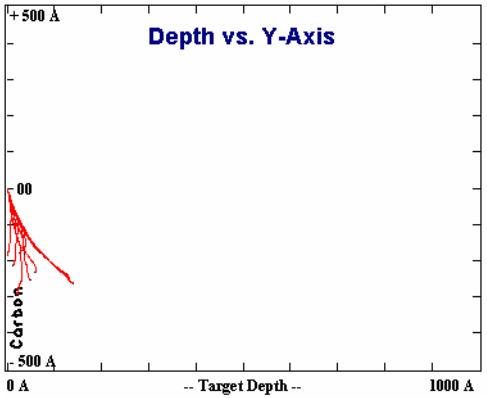


Figura GaC32: 1000A 30KeV 10E1 80º

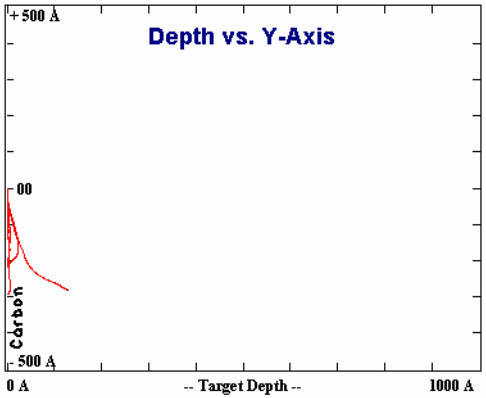


Figura GaC33: 1000A 30KeV 10E1 88º

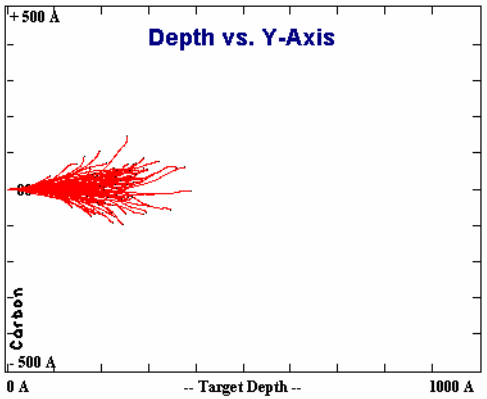


Figura GaC34: 1000A 30KeV 10E2 0º

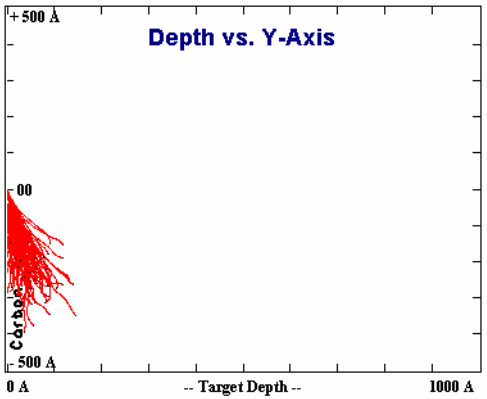


Figura GaC35: 1000A 30KeV 10E2 80º

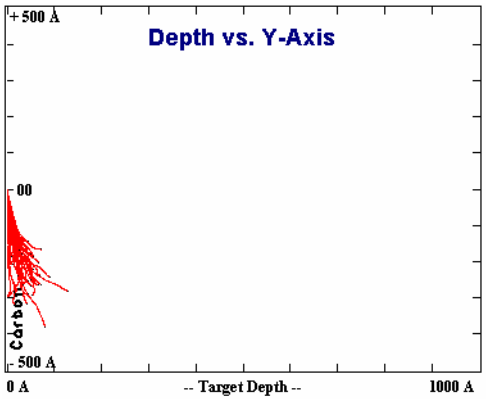


Figura GaC36: 1000A 30KeV 10E2 88º

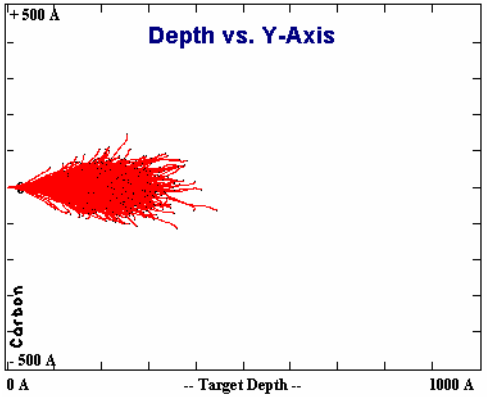


Figura GaC37: 1000A 30KeV 10E3 0º

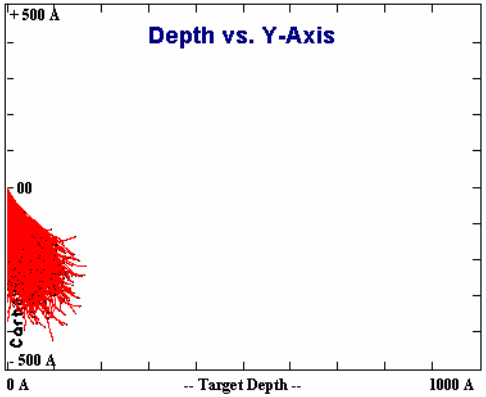


Figura GaC38: 1000A 30KeV 10E3 80º

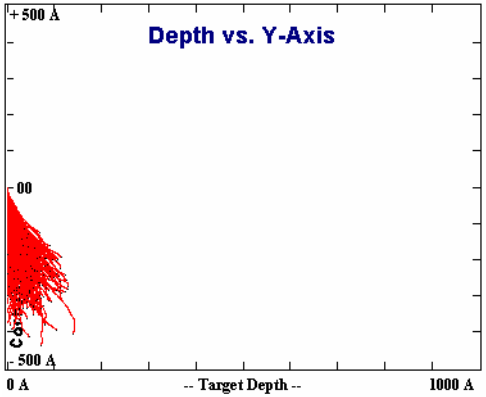


Figura GaC39: 1000A 30KeV 10E3 88º

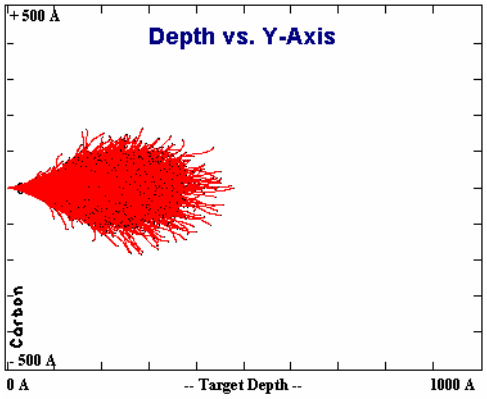


Figura GaC40: 1000A 30KeV 10E4 0º

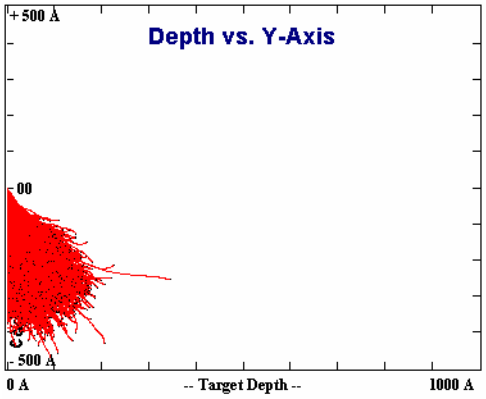


Figura GaC41: 1000A 30KeV 10E4 80º

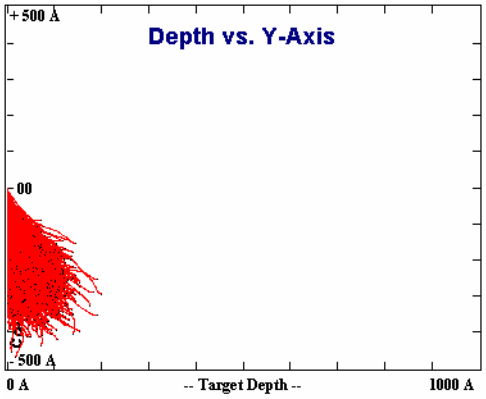


Figura GaC42: 1000A 30KeV 10E4 88º

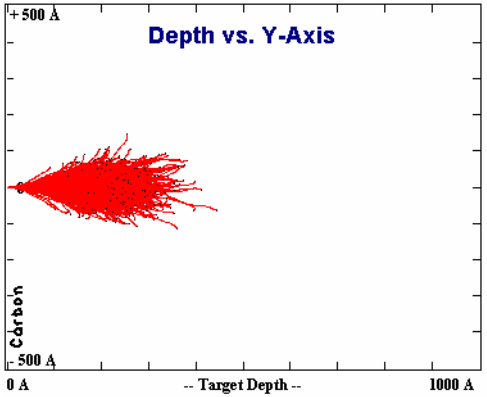


Figura GaC37: 1000A 30KeV 10E3 0º

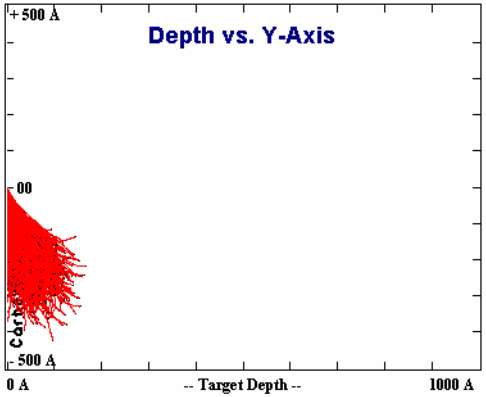


Figura GaC38: 1000A 30KeV 10E3 80º

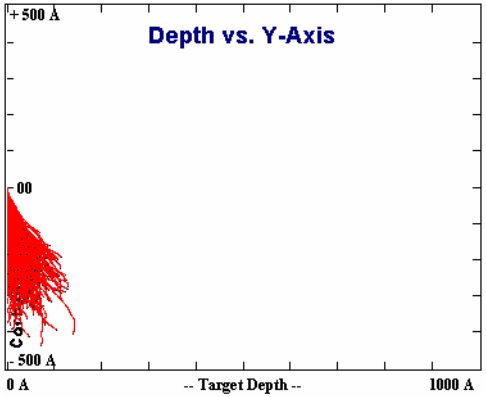


Figura GaC39: 1000A 30KeV 10E3 88º

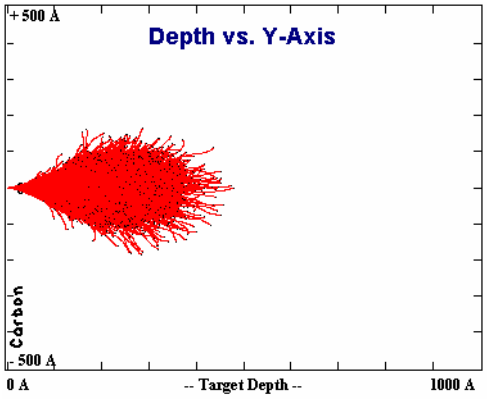


Figura GaC40: 1000A 30KeV 10E4 0º

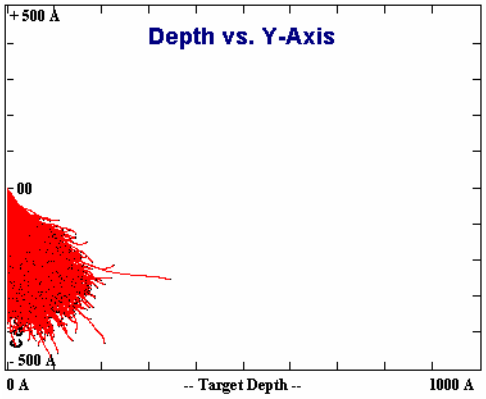


Figura GaC41: 1000A 30KeV 10E4 80º

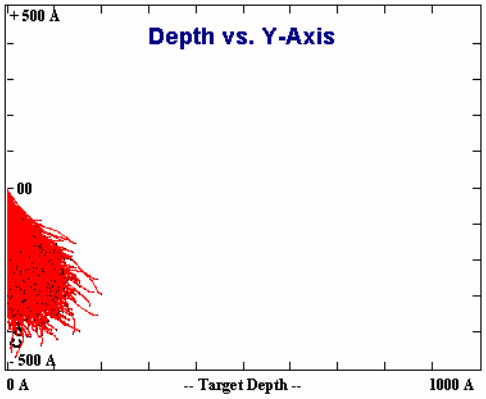


Figura GaC42: 1000A 30KeV 10E4 88º

SRIM/TRIM: Cálculo rápido de colisões e distribuição de íons

MATERIAL ION: Galio (Gallium) TARGET: Carbono (Carbon) WIDTH: 1000A = 100nm

Energia	Número Total de Íons	Ângulo	Parâmetros de espalhamento						Perdas de energia						Resultados			
			Longitudinal		Lateral		Radial		Ionização		Vacâncias		Phonons		Sputtering Yield		Figura	Tempo
			Alcance	Desvio	Alcance	Desvio	Alcance	Desvio	Íons	Recoils	Íons	Recoils	Íons	Recoils	Atoms/Ion	eV/Atom		
30KeV	10E4	0	228A	57A	34A	43A	53A	29A	10,44	25,54	0,47	2,15	2,12	59,27	0,00	0,00	GaC 46	4'02"
		80	55A	35A	226A	233A	230A	56A	10,64	25,67	0,46	2,15	2,02	59,05	0,00	0,00	GaC 47	3'42"
		88	44A	29A	229A	236A	234A	57A	10,98	26,18	0,42	2,16	1,84	58,41	0,00	0,00	GaC 48	2'36"
	10E5	0	228A	57A	34A	43A	53A	29A	10,47	25,53	0,47	2,15	2,13	59,25	0,00	0,00	GaC 49	39'00"
		80	55A	34A	226A	233A	230A	57A	10,61	25,72	0,46	2,15	2,02	59,04	0,00	0,00	GaC 50	36'20"
		88	43A	29A	229A	236A	233A	58A	11,00	26,20	0,43	2,16	1,84	58,38	0,00	0,00	GaC 51	21'40"
	10E6	0	229A	57A	34A	43A	53A	29A	10,49	25,51	0,47	2,15	2,13	59,25	0,00	0,00	GaC 52	302'00"
		80	55A	34A	226A	233A	230A	57A	10,62	25,71	0,48	2,13	2,02	59,04	0,00	0,00	GaC 53	284'00"
		88	43A	29A	229A	236A	233A	57A	10,98	26,20	0,45	2,14	1,84	58,39	0,00	0,00	GaC 54	270'00"
	10E7	0	229A	57A	34A	43A	53A	29A	10,49	25,51	0,46	2,16	2,13	59,25	0,00	0,00	ERRO6	50 horas
		80																
		88																

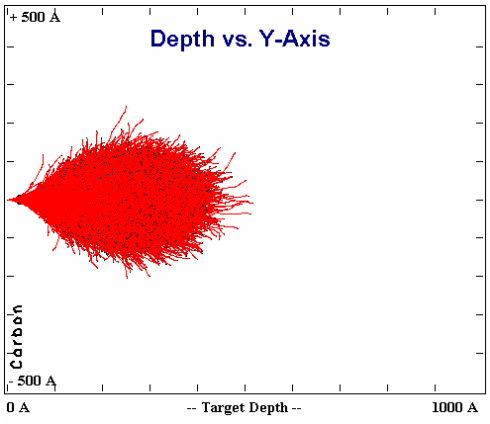


Figura GaC49: 1000A 30KeV 10E5 0º

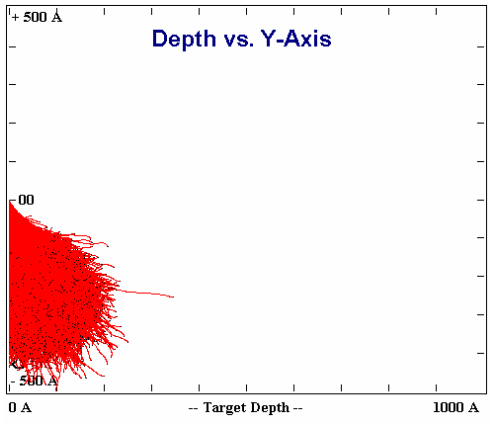


Figura GaC50: 1000A 30KeV 10E5 80º

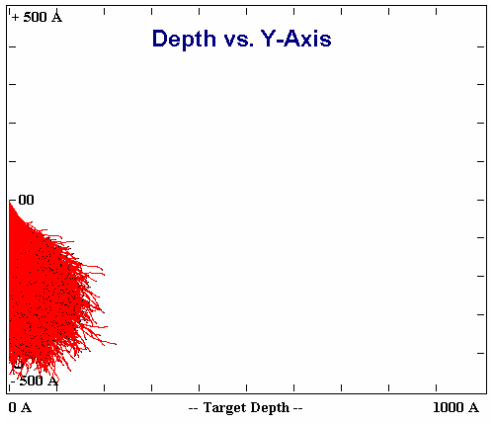


Figura GaC51: 1000A 30KeV 10E5 88º

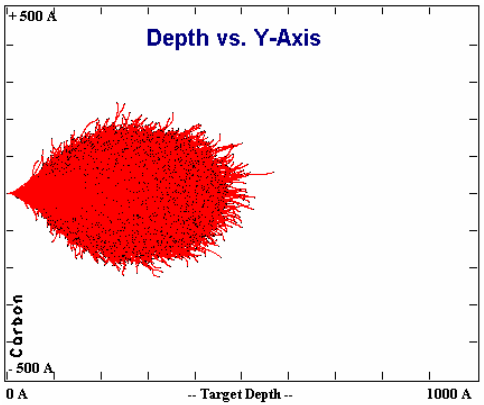


Figura GaC52: 1000A 30KeV 10E6 0º

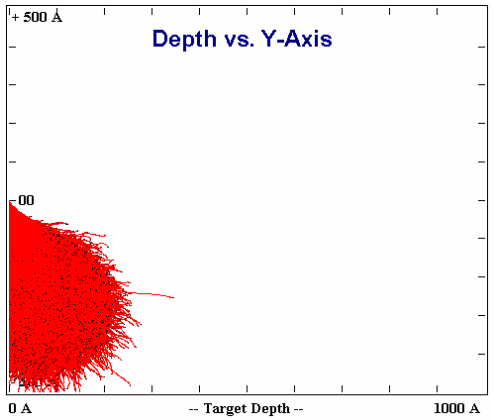


Figura GaC53: 1000A 30KeV 10E6 80º

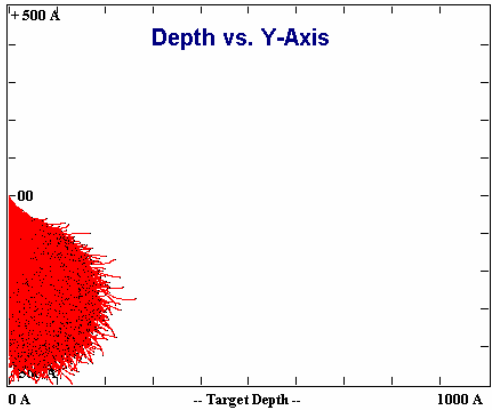


Figura GaC54: 1000A 30KeV 10E6 88º

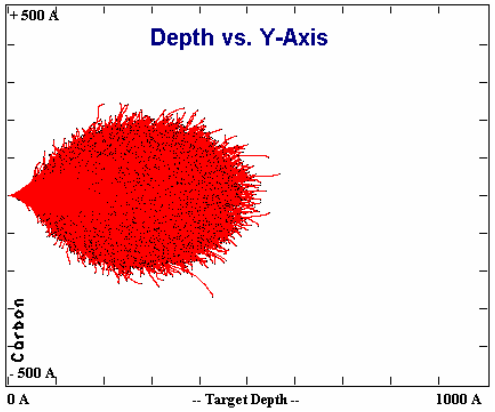


Figura ERRO6: 1000A 30KeV 10E7 0º

SRIM/TRIM: Cálculo detalhado com colisões em cascata

MATERIAL ION: Galio (Gallium) TARGET: Carbono (Carbon) WIDTH: 1000A = 100nm

Energia	Número Total de Ions	Ângulo	Parâmetros de espalhamento						Perdas de energia						Resultados			
			Longitudinal		Lateral		Radial		Ionização		Vacâncias		Phonons		Sputtering Yield		Figura	Tempo
			Alcance	Desvio	Alcance	Desvio	Alcance	Desvio	Ions	Recoils	Ions	Recoils	Ions	Recoils	Atoms/Ion	eV/Atom		
10KeV	10E1	0	121A	17A	19A	26A	35A	19A	8,85	28,85	0,93	2,75	3,82	54,79	1,70	40,93	GaC 55	0'02"
		80	32A	19A	115A	116A	116A	9A	10,21	33,11	1,06	3,43	3,51	48,67	3,51	48,67	GaC 56	0'02"
		88	10A	9A	109A	110A	113A	7A	15,52	30,30	1,31	3,54	4,31	45,01	11,90	166,95	GaC 57	0'01"
	10E2	0	118A	30A	20A	24A	33A	16A	8,37	29,91	0,94	2,75	3,38	54,65	1,31	31,40	GaC 58	0'09"
		80	28A	19A	116A	119A	118A	26A	10,31	33,27	1,11	3,32	3,86	48,31	18,56	106,14	GaC 59	0'09"
		88	22A	16A	122A	124A	124A	24A	13,72	32,46	1,13	3,50	4,47	44,70	11,63	132,30	GaC 60	0'06"
	10E3	0	113A	26A	20A	25A	30A	16A	8,00	30,41	0,88	2,77	3,27	54,67	0,96	29,88	GaC 61	3'00"
		80	29A	18A	112A	115A	115A	26A	9,92	33,56	1,05	3,43	3,81	48,23	19,08	102,10	GaC 62	1'38"
		88	25A	16A	113A	116A	116A	26A	11,98	33,11	1,14	3,54	4,12	46,11	13,23	153,19	GaC 63	0'70"
	10E4	0	113A	26A	19A	24A	30A	16A	8,10	30,25	0,89	2,78	3,28	54,70	0,99	29,18	GaC 64	15'10"
		80	30A	18A	113A	116A	115A	26A	10,13	33,46	1,06	3,41	3,86	48,08	18,82	104,72	GaC 65	17'06"
		88	24A	16A	114A	117A	116A	26A	11,84	33,33	1,12	3,56	4,09	46,06	12,95	151,79	GaC 66	9'00"
	10E5	0	114A	26A	19A	24A	30A	16A	8,13	30,26	0,89	2,77	3,30	54,66	0,99	28,80	GaC 67	115'00"
		80	29A	18A	113A	116A	115A	26A	10,11	33,47	1,06	3,41	3,86	48,08	18,89	104,83	GaC 68	100'00"
		88	24A	16A	114A	117A	117A	26A	11,84	33,33	1,13	3,55	4,07	46,08	12,92	151,39	GaC 69	135'00"
20KeV	10E1	0	191A	47A	26A	33A	45A	28A	11,04	31,18	0,68	2,66	2,58	51,85	1,11	32,67	GaC 70	0'10"
		80	81A	29A	168A	169A	174A	14A	10,89	36,34	0,77	3,22	2,83	45,95	28,40	137,59	GaC 71	0'04"
		88	52A	40A	194A	194A	198A	11A	15,93	32,97	0,92	3,19	3,06	43,93	13,80	169,57	GaC 72	0'03"
	10E2	0	180A	44A	30A	37A	48A	24A	9,52	33,45	0,65	2,67	2,47	51,25	1,45	31,19	GaC 73	0'16"
		80	46A	30A	184A	190A	189A	47A	11,88	35,07	0,74	3,11	2,87	46,32	25,81	127,12	GaC 74	0'15"
		88	32A	20A	185A	189A	188A	38A	14,23	34,88	0,77	3,16	2,92	44,03	17,69	191,24	GaC 75	0'10"
	10E3	0	173A	43A	28A	36A	45A	24A	9,20	34,04	0,63	2,69	2,33	51,12	1,37	41,17	GaC 76	2'30"
		80	42A	27A	175A	179A	178A	40A	11,47	35,35	0,74	3,11	2,75	46,59	25,42	141,88	GaC 77	2'21"
		88	33A	22A	180A	184A	183A	41A	13,78	34,66	0,79	3,21	2,95	44,61	18,55	197,93	GaC 78	2'20"
	10E4	0	178A	43A	28A	36A	44A	24A	9,48	33,75	0,63	2,69	2,39	51,06	1,24	38,76	GaC 79	21'02"
		80	44A	27A	176A	181A	179A	42A	11,62	35,24	0,74	3,09	2,77	46,54	25,03	142,80	GaC 80	30'00"
		88	35A	23A	177A	182A	181A	43A	13,69	34,75	0,78	3,22	2,93	44,63	18,31	199,85	GaC 81	16'00"
	10E5	0	177A	43A	28A	36A	44A	24A	9,47	33,80	0,63	2,69	2,38	51,02	1,23	39,00	GaC 82	217'00"
		80	44A	28A	175A	181A	179A	43A	11,59	35,33	0,74	3,08	2,77	46,49	25,01	142,95	GaC 83	126'06"
		88	35A	23A	177A	182A	181A	43A	13,71	34,85	0,79	3,21	2,94	44,51	18,47	198,28	GaC 84	92'10"
30KeV	10E1	0	231A	86A	47A	59A	58A	32A	10,69	34,91	0,52	2,63	2,04	49,21	1,90	65,13	GaC 85	0'03"
		80	71A	32A	236A	243A	245A	53A	13,89	33,27	0,70	3,00	2,62	46,52	28,30	237,13	GaC 86	0'03"
		88	42A	42A	251A	253A	254A	25A	17,12	32,62	0,67	3,01	2,34	44,24	14,90	240,37	GaC 87	0'02"
	10E2	0	236A	65A	36A	44A	58A	28A	10,63	35,66	0,53	2,61	2,01	48,58	1,69	62,42	GaC 88	0'20"
		80	56A	33A	233A	239A	238A	53A	12,48	36,52	0,61	2,93	2,21	45,25	29,41	175,62	GaC 89	0'23"
		88	45A	35A	245A	251A	249A	56A	15,87	35,62	0,63	2,95	2,50	42,42	22,93	213,43	GaC 90	0'13"
	10E3	0	237A	60A	36A	45A	56A	30A	10,66	35,83	0,52	2,59	1,99	48,41	1,33	48,28	GaC 91	4'00"
		80	58A	36A	235A	241A	239A	56A	12,67	36,68	0,60	2,89	2,26	44,91	28,43	169,56	GaC 92	5'00"
		88	44A	29A	235A	241A	240A	55A	15,03	36,05	0,63	2,98	2,39	42,92	22,54	243,20	GaC 93	3'00"
	10E4	0	236A	58A	36A	46A	57A	31A	10,59	35,82	0,52	2,60	1,98	48,49	1,32	45,58	GaC 94	30'00"
		80	57A	36A	233A	240A	238A	57A	12,76	36,56	0,60	2,89	2,28	44,92	28,41	175,48	GaC 95	33'26"
		88	46A	30A	235A	242A	239A	58A	15,14	35,89	0,63	2,99	2,43	42,93	22,01	236,09	GaC 96	30'00"
	10E5	0	236A	58A	36A	46A	57A	31A	10,57	35,87	0,52	2,60	1,98	48,46	1,35	45,95	GaC 97	162'00"
		80	57A	36A	233A	240A	238A	58A	12,81	36,59	0,60	2,88	2,28	44,84	28,55	174,79	GaC 98	180'00"
		88	46A	30A	235A	242A	240A	58A	15,16	35,94	0,63	2,98	2,43	42,86	22,18	235,03	GaC 99	210'00"

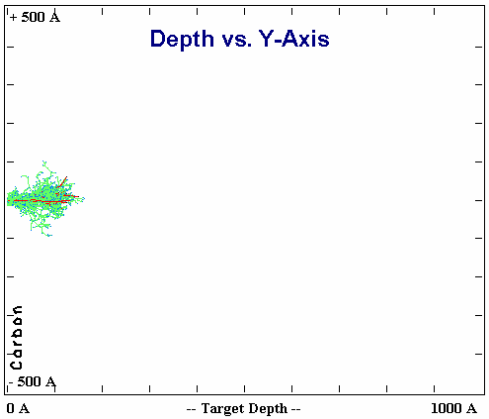


Figura GaC55: 1000A 10KeV 10E1 0º

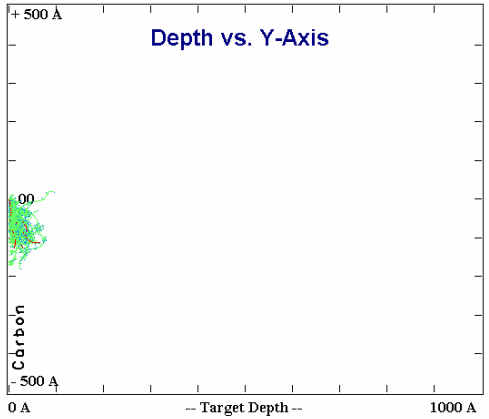


Figura GaC56: 1000A 10KeV 10E1 80º

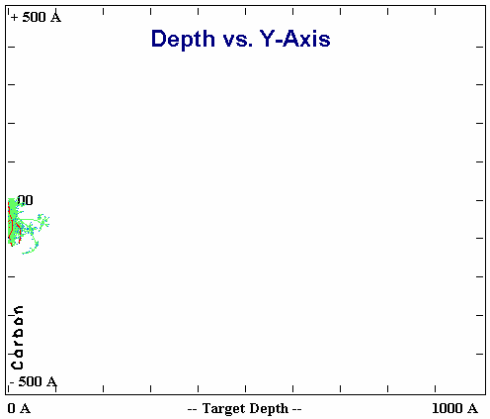


Figura GaC57: 1000A 10KeV 10E1 88º

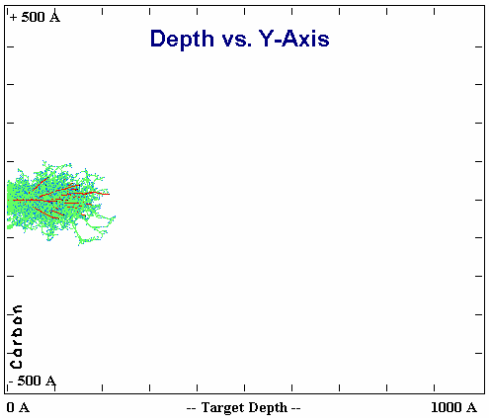


Figura GaC58: 1000A 10KeV 10E2 0º

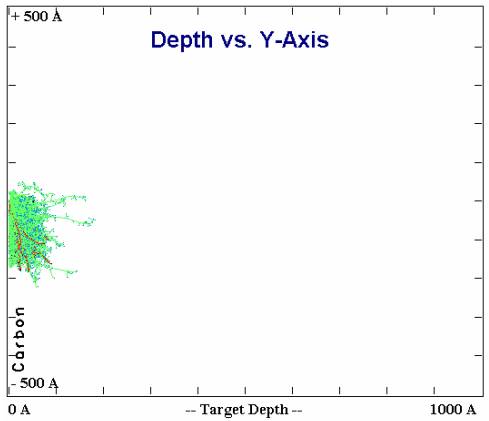


Figura GaC59: 1000A 10KeV 10E2 80º

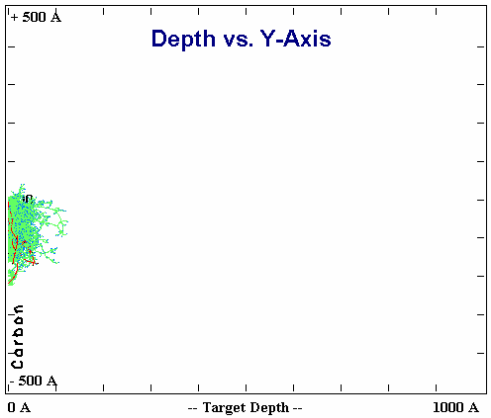


Figura GaC60: 1000A 10KeV 10E2 88º

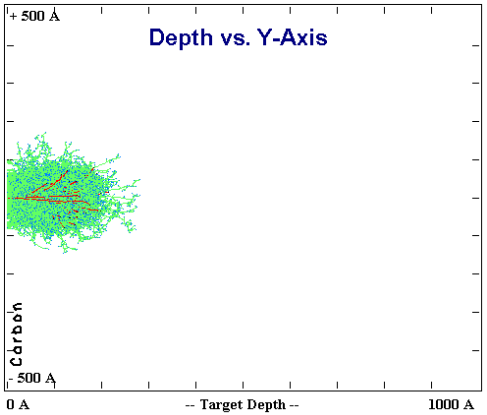


Figura GaC61: 1000A 10KeV 10E3 0º

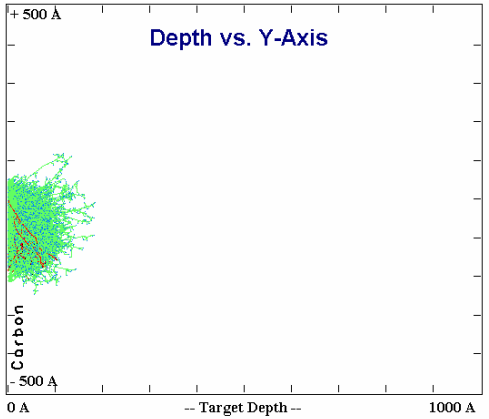


Figura GaC62: 1000A 10KeV 10E3 80º

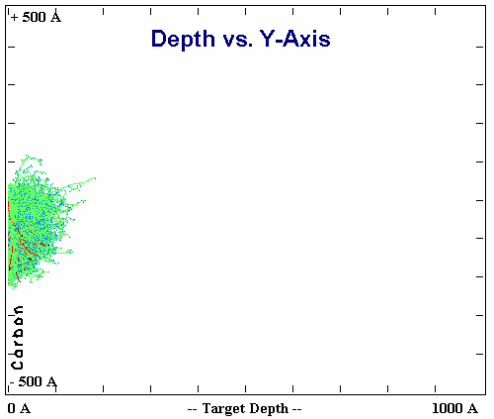


Figura GaC63: 1000A 10KeV 10E3 88º

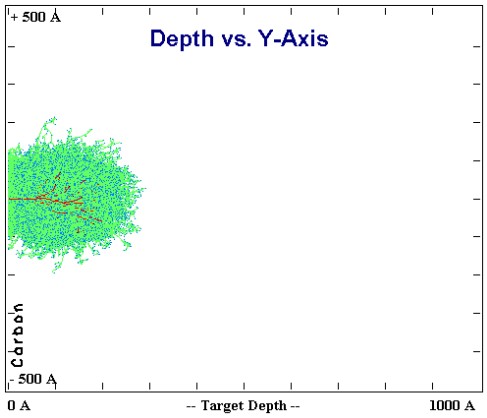


Figura GaC64: 1000A 10KeV 10E4 0º

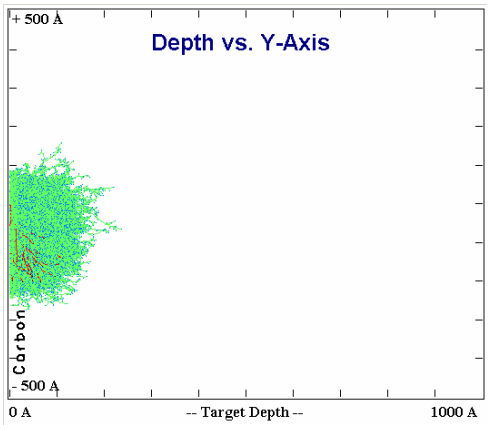


Figura GaC65: 1000A 10KeV 10E4 80º

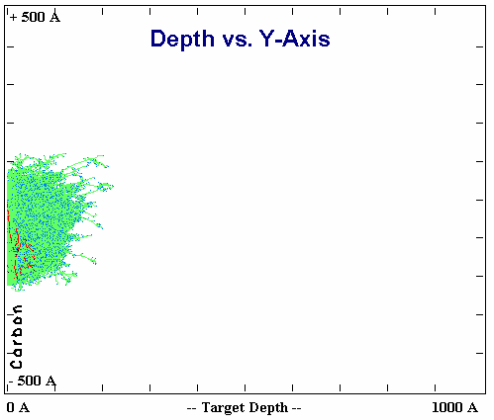


Figura GaC66: 1000A 10KeV 10E4 88º

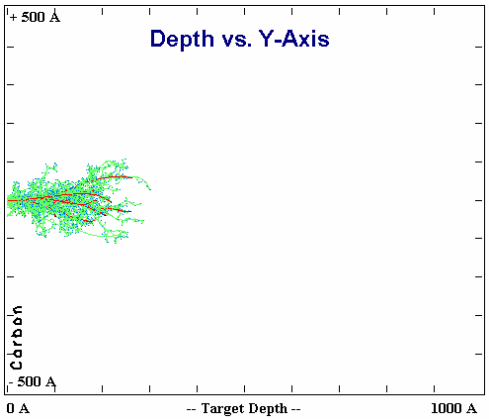


Figura GaC70: 1000A 20KeV 10E1 0º

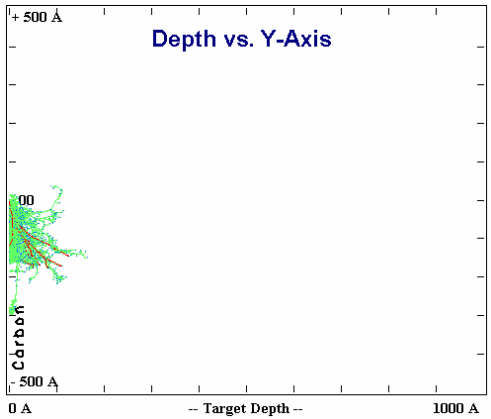


Figura GaC71: 1000A 20KeV 10E1 80º

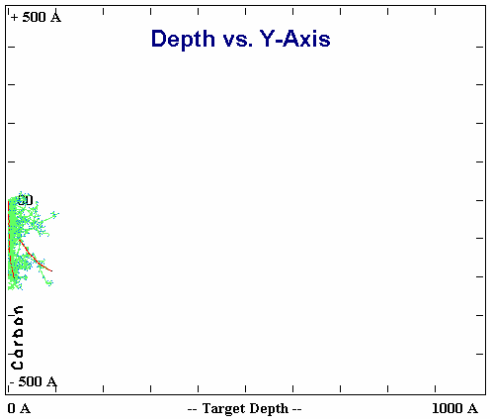


Figura GaC72: 1000A 20KeV 10E1 88º

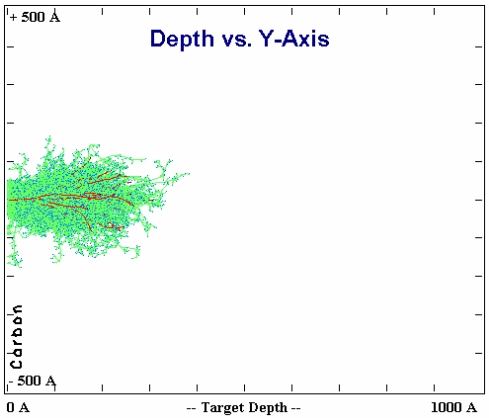


Figura GaC73: 1000A 20KeV 10E2 0º

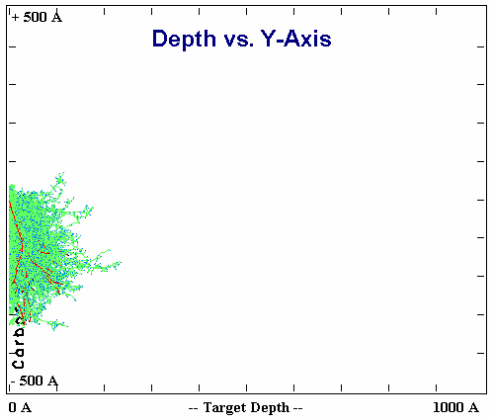


Figura GaC74: 1000A 20KeV 10E2 80º

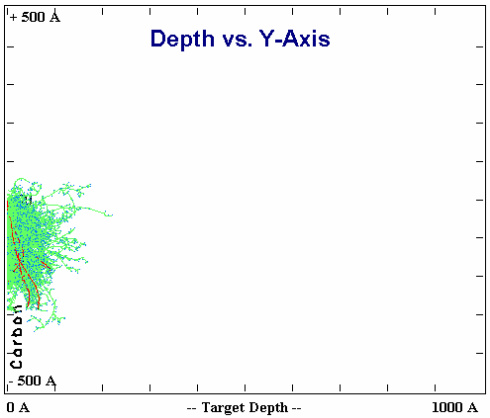


Figura GaC75: 1000A 20KeV 10E2 88º

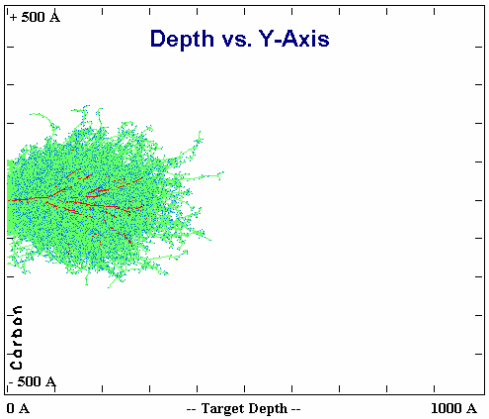


Figura GaC76: 1000A 20KeV 10E3 0º

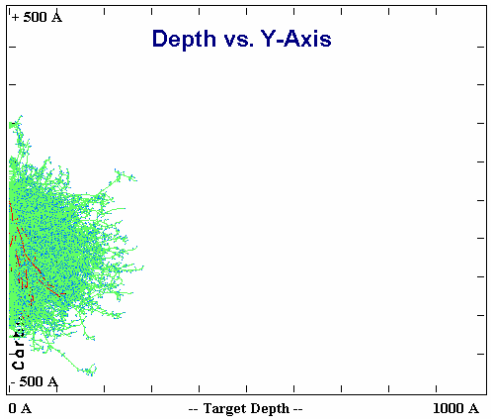


Figura GaC77: 1000A 20KeV 10E3 80º

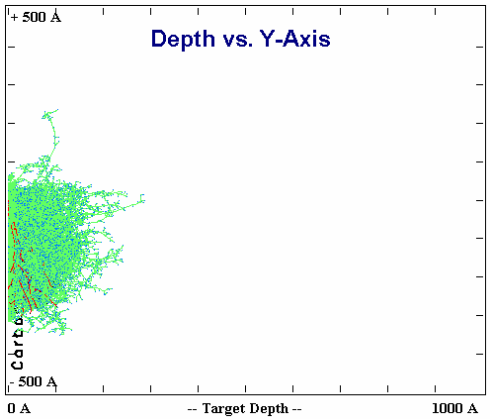


Figura GaC78: 1000A 20KeV 10E3 88º

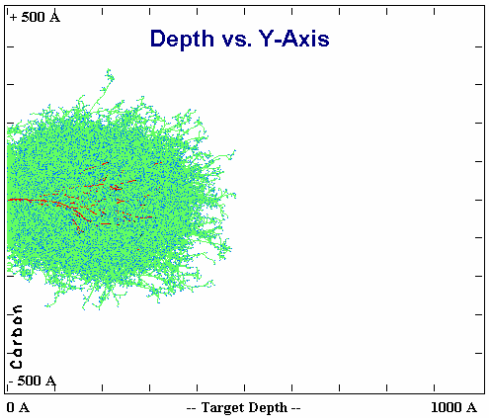


Figura GaC79: 1000A 20KeV 10E4 0º

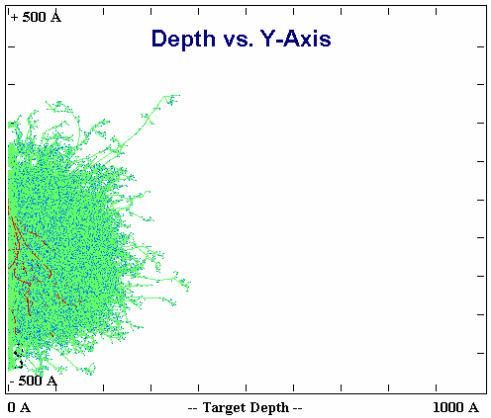


Figura GaC80: 1000A 20KeV 10E4 80º

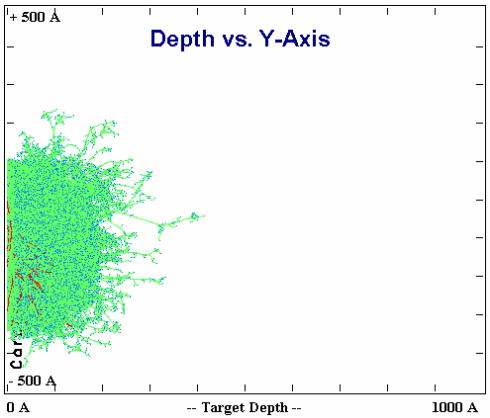


Figura GaC81: 1000A 20KeV 10E4 88º

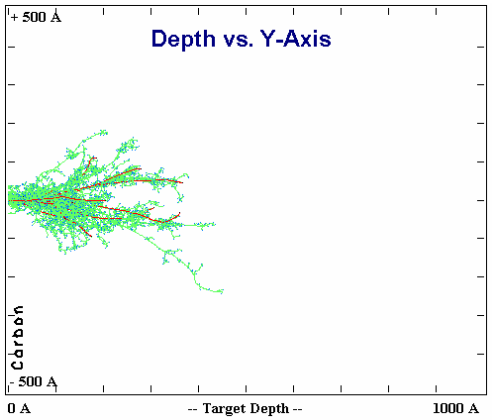


Figura GaC85: 1000A 30KeV 10E1 0º

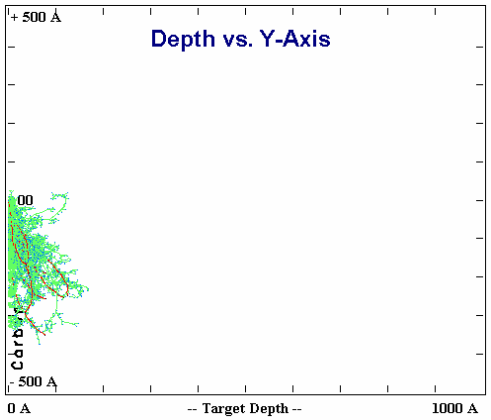


Figura GaC86: 1000A 30KeV 10E1 80º

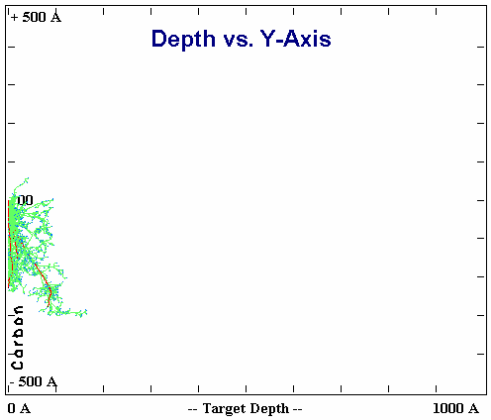


Figura GaC87: 1000A 30KeV 10E1 88º

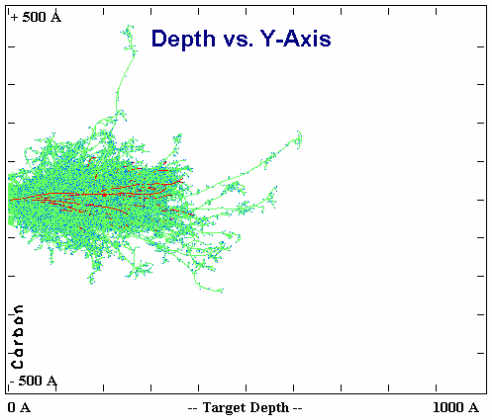


Figura GaC88: 1000A 30KeV 10E2 0º

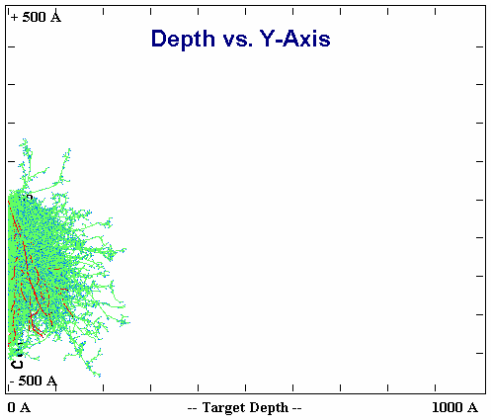


Figura GaC89: 1000A 30KeV 10E2 80º

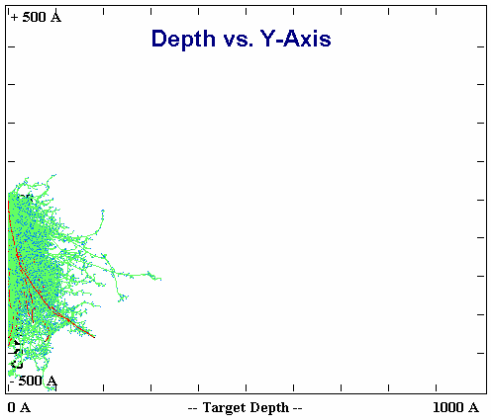


Figura GaC90: 1000A 30KeV 10E2 88º

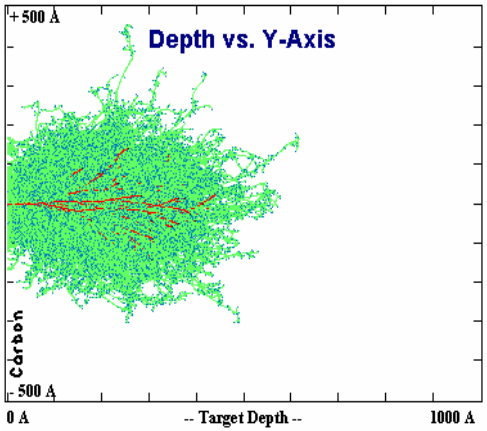


Figura GaC91: 1000A 30KeV 10E3 0º

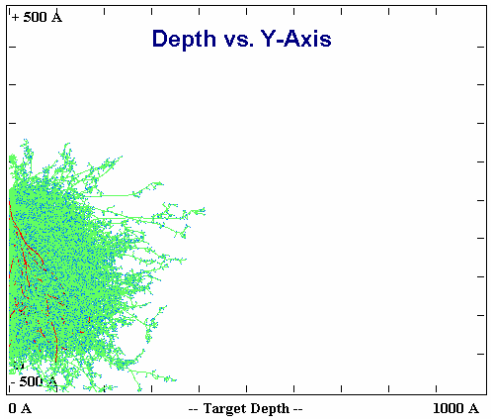


Figura GaC92: 1000A 30KeV 10E3 80º

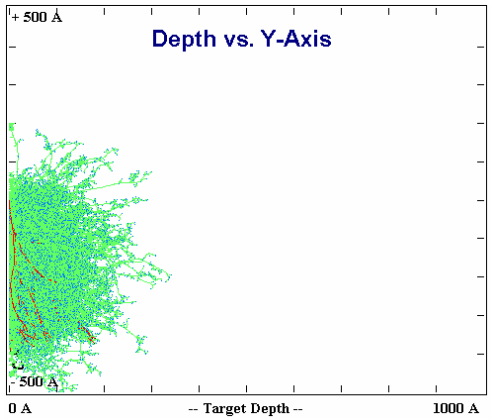


Figura GaC93: 1000A 30KeV 10E3 88º

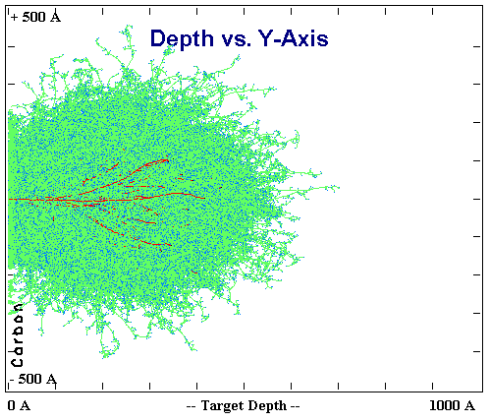


Figura GaC94: 1000A 30KeV 10E4 0º

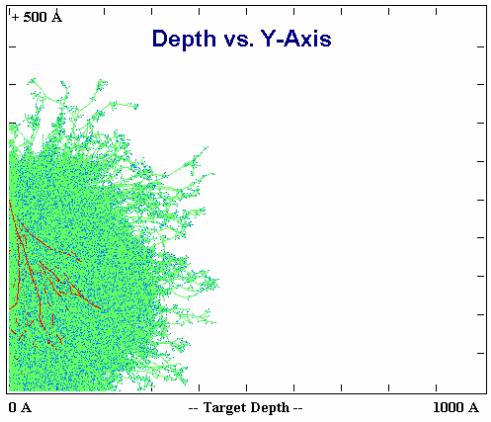


Figura GaC95: 1000A 30KeV 10E4 80º

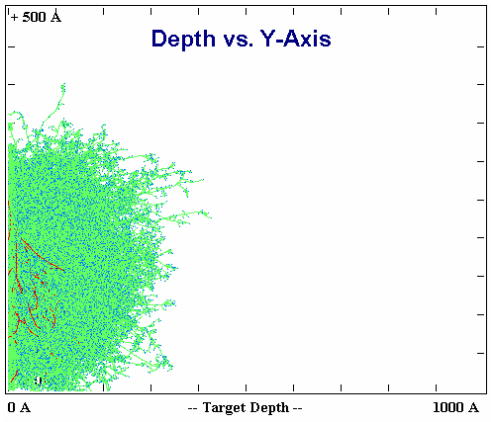


Figura GaC96: 1000A 30KeV 10E4 88º

SRIM/TRIM: Cálculo rápido de colisões e distribuição de íons

MATERIAL ION: Galio (Gallium) TARGET: Platina (Platinum) WIDTH: 1000A = 100nm

Energia	Número Total de Íons	Ângulo	Parâmetros de espalhamento						Perdas de energia						Resultados		Figura	Tempo
			Longitudinal		Lateral		Radial		Ionização		Vacâncias		Phonons		Sputtering Yield			
			Alcance	Desvio	Alcance	Desvio	Alcance	Desvio	Íons	Recoils	Íons	Recoils	Íons	Recoils	Atoms/Ion	eV/Atom		
10KeV	10E1	0	50A	27A	31A	34A	50A	17A	7,80	12,13	0,74	2,98	2,21	74,15	0,00	0,00	GaPt 01	0'01"
		80	14A	8A	29A	33A	34A	15A	6,78	12,96	0,56	3,23	1,31	75,17	0,00	0,00	GaPt 02	0'01"
		88	n/a	n/a	n/a	n/a	n/a	n/a	8,33	12,88	0,46	3,22	1,41	73,70	0,00	0,00	GaPt 03	0'01"
	10E2	0	39A	25A	23A	29A	33A	20A	5,91	13,17	0,57	3,19	1,89	75,26	0,00	0,00	GaPt 04	0'02"
		80	28A	23A	33A	41A	43A	26A	5,83	13,44	0,52	3,27	1,46	75,48	0,00	0,00	GaPt 05	0'02"
		88	37A	24A	41A	50A	52A	26A	6,87	13,07	0,54	3,20	1,62	74,71	0,00	0,00	GaPt 06	0'01"
	10E3	0	39A	24A	22A	28A	32A	20A	5,75	13,20	0,57	3,21	1,82	75,46	0,00	0,00	GaPt07	0'12"
		80	30A	19A	38A	46A	48A	25A	6,14	13,30	0,52	3,25	1,57	75,21	0,00	0,00	GaPt 08	0'08"
		88	29A	19A	36A	43A	44A	25A	6,15	13,38	0,52	3,25	1,49	75,20	0,00	0,00	GaPt 09	0'06"
	10E4	0	40A	25A	21A	27A	33A	20A	5,87	13,16	0,57	3,20	1,83	75,37	0,00	0,00	GaPt 10	1'50"
		80	28A	19A	37A	45A	46A	25A	6,04	13,41	0,52	3,26	1,56	75,22	0,00	0,00	GaPt 11	1'16"
		88	28A	18A	35A	43A	44A	25A	6,03	13,48	0,50	3,27	1,52	75,19	0,00	0,00	GaPt 12	0'52"
	10E5	0	40A	25A	21A	27A	33A	20A	5,92	13,14	0,57	3,20	1,83	75,34	0,00	0,00	GaPt 13	10'41"
		80	28A	19A	36A	44A	45A	25A	6,03	13,41	0,52	3,25	1,57	75,23	0,00	0,00	GaPt 14	7'08"
		88	27A	18A	35A	43A	44A	25A	6,03	13,50	0,50	3,27	1,51	75,20	0,00	0,00	GaPt 15	5'36"
20KeV	10E1	0	55A	32A	40A	48A	54A	36A	7,10	13,89	0,41	3,32	1,18	74,10	0,00	0,00	GaPt 16	0'01"
		80	24A	21A	78A	109A	96A	72A	6,00	14,77	0,38	3,38	0,86	74,61	0,00	0,00	GaPt 17	0'01"
		88	32A	24A	46A	51A	87A	28A	7,43	14,58	0,38	3,29	1,22	73,10	0,00	0,00	GaPt 18	0'01"
	10E2	0	64A	37A	35A	45A	51A	30A	6,72	14,19	0,41	3,31	1,20	74,17	0,00	0,00	GaPt 19	0'02"
		80	31A	20A	51A	65A	66A	43A	6,44	14,67	0,36	3,37	0,97	74,19	0,00	0,00	GaPt 20	0'02"
		88	41A	29A	44A	53A	62A	36A	7,32	14,44	0,36	3,33	1,02	73,52	0,00	0,00	GaPt 21	0'01"
	10E3	0	65A	40A	33A	42A	51A	31A	6,55	14,35	0,40	3,33	1,18	74,20	0,00	0,00	GaPt 22	0'14"
		80	43A	30A	56A	67A	69A	38A	6,72	14,55	0,36	3,36	1,01	74,01	0,00	0,00	GaPt 23	0'10"
		88	40A	26A	57A	68A	71A	38A	7,00	14,54	0,35	3,35	1,01	73,75	0,00	0,00	GaPt 24	0'08"
	10E4	0	63A	38A	33A	43A	52A	30A	6,55	14,31	0,40	3,33	1,19	74,22	0,00	0,00	GaPt 25	2'10"
		80	43A	29A	57A	69A	70A	39A	6,67	14,61	0,36	3,36	1,02	73,98	0,00	0,00	GaPt 26	1'30"
		88	42A	28A	55A	67A	68A	38A	6,73	14,68	0,34	3,37	0,99	73,89	0,00	0,00	GaPt 27	1'08"
	10E5	0	63A	39A	33A	42A	51A	30A	6,54	14,32	0,40	3,33	1,19	74,22	0,00	0,00	GaPt 28	13'15"
		80	42A	29A	56A	69A	70A	39A	6,69	14,60	0,36	3,36	1,02	73,98	0,00	0,00	GaPt 29	9'06"
		88	41A	28A	55A	68A	69A	39A	6,73	14,67	0,35	3,37	0,99	73,89	0,00	0,00	GaPt 30	6'32"
30KeV	10E1	0	57A	44A	39A	52A	59A	40A	5,84	15,32	0,31	3,42	0,91	74,19	0,00	0,00	GaPt 31	0'01"
		80	64A	74A	48A	51A	93A	38A	8,50	15,13	0,31	3,31	0,75	71,98	0,00	0,00	GaPt 32	0'01"
		88	161A	- A	99A	- A	103A	- A	8,07	15,02	0,31	3,35	0,78	72,48	0,00	0,00	GaPt 33	0'01"
	10E2	0	79A	45A	43A	55A	69A	36A	7,04	14,94	0,33	3,35	0,99	73,34	0,00	0,00	GaPt 34	0'02"
		80	54A	44A	71A	88A	88A	50A	7,68	15,15	0,30	3,36	0,82	72,69	0,00	0,00	GaPt 35	0'02"
		88	50A	40A	60A	74A	76A	42A	7,74	15,33	0,26	3,38	0,72	72,56	0,00	0,00	GaPt 36	0'01"
	10E3	0	80A	48A	43A	56A	66A	39A	6,86	15,04	0,32	3,37	0,92	73,48	0,00	0,00	GaPt 37	0'15"
		80	55A	38A	72A	88A	90A	51A	7,04	15,29	0,29	3,39	0,79	73,20	0,00	0,00	GaPt 38	0'10"
		88	51A	33A	72A	89A	90A	53A	7,30	15,33	0,28	3,39	0,78	72,92	0,00	0,00	GaPt 39	0'07"
	10E4	0	83A	50A	43A	56A	67A	40A	7,03	15,01	0,32	3,36	0,93	73,35	0,00	0,00	GaPt 40	2'25"
		80	55A	38A	75A	91A	93A	51A	7,19	15,29	0,29	3,38	0,81	73,04	0,00	0,00	GaPt 41	1'34"
		88	54A	36A	75A	91A	92A	52A	7,28	15,36	0,28	3,39	0,78	72,01	0,00	0,00	GaPt 42	1'13"
	10E5	0	83A	51A	43A	55A	67A	40A	7,05	15,01	0,32	3,36	0,93	73,33	0,00	0,00	GaPt 43	15'40"
		80	56A	38A	75A	91A	93A	52A	7,21	15,29	0,29	3,38	0,81	73,02	0,00	0,00	GaPt 44	10'18"
		88	54A	37A	73A	90A	91A	51A	7,29	15,35	0,28	3,38	0,79	72,91	0,00	0,00	GaPt 45	7'28"

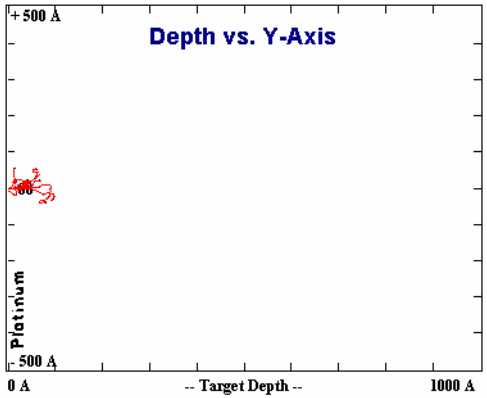


Figura GaPt01: 1000A 10KeV 10E1 0º

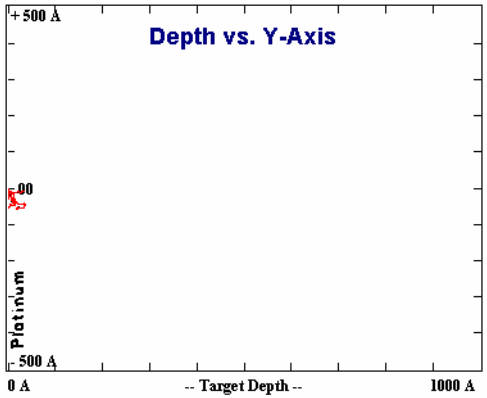


Figura GaPt02: 1000A 10KeV 10E1 80º

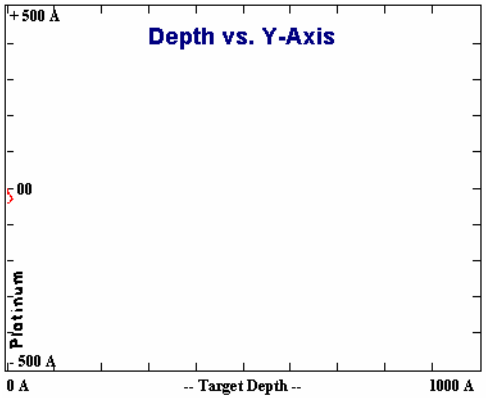


Figura GaPt03: 1000A 10KeV 10E1 88º

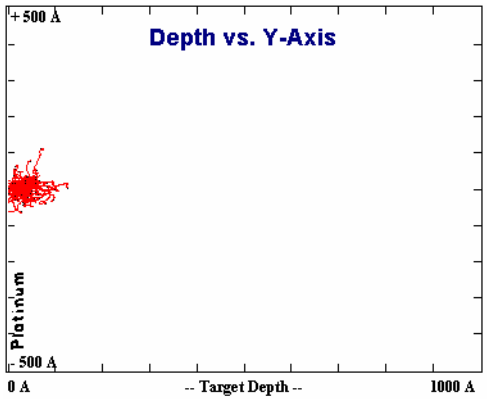


Figura GaPt04: 1000A 10KeV 10E2 0º

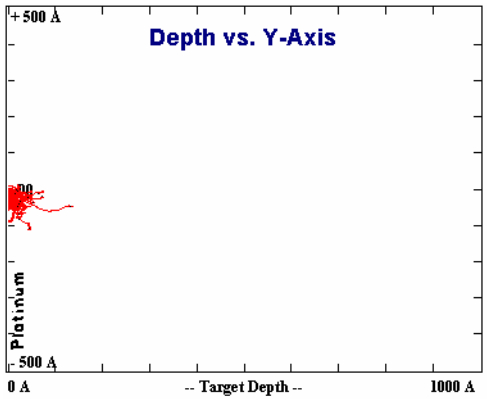


Figura GaPt05: 1000A 10KeV 10E2 80º

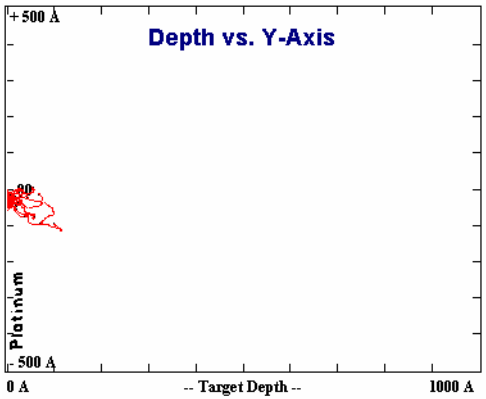


Figura GaPt06: 1000A 10KeV 10E2 88º

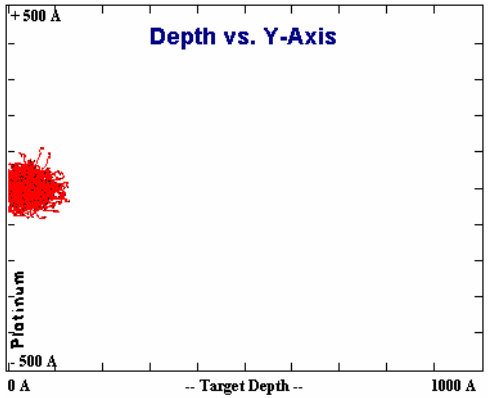


Figura GaPt07: 1000A 10KeV 10E3 0º

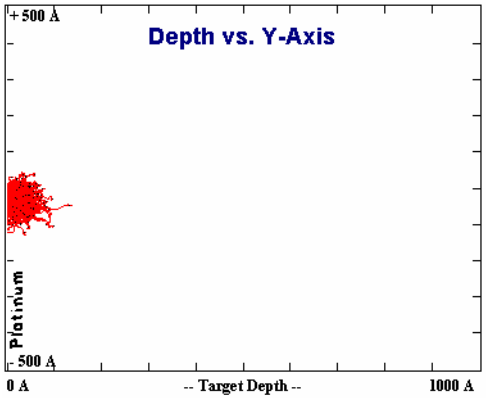


Figura GaPt08: 1000A 10KeV 10E3 80º

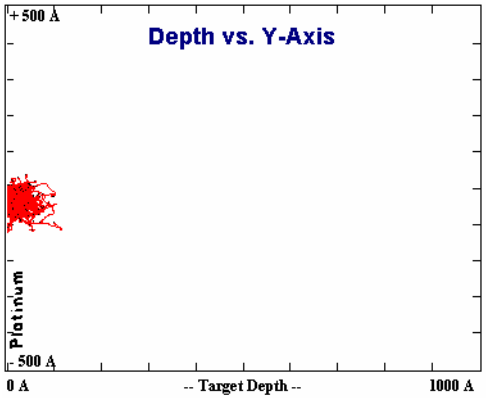


Figura GaPt09: 1000A 10KeV 10E3 88º

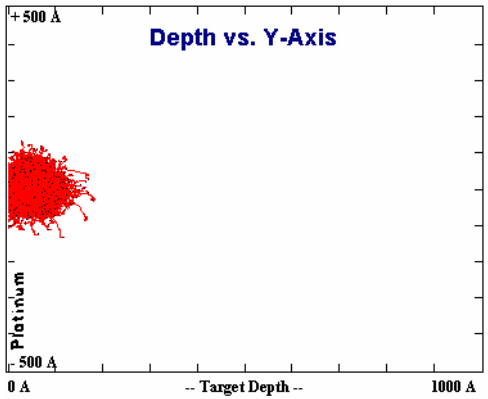


Figura GaPt10: 1000A 10KeV 10E4 0º

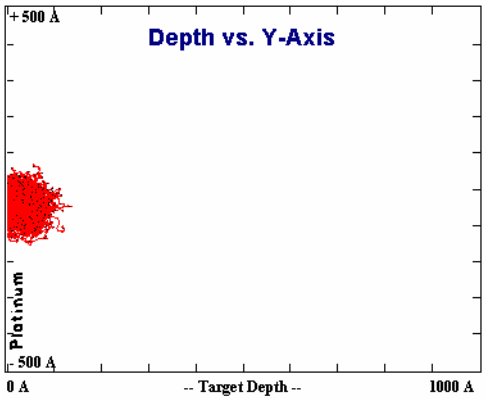


Figura GaPt11: 1000A 10KeV 10E4 80º

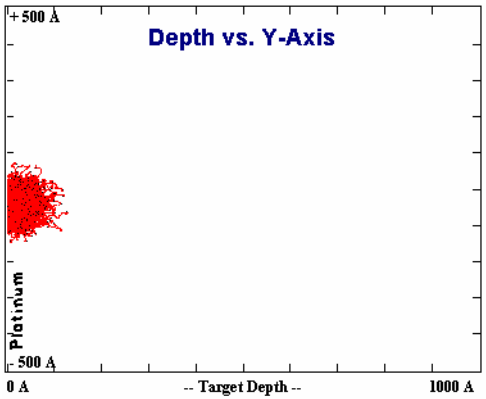


Figura GaPt12: 1000A 10KeV 10E4 88º

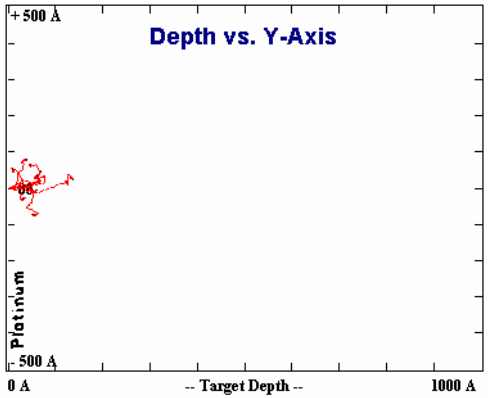


Figura GaPt16: 1000A 20KeV 10E1 0º

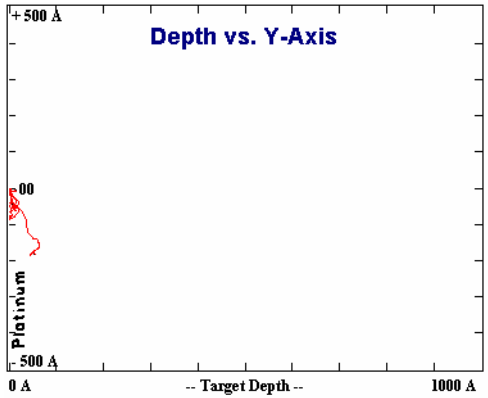


Figura GaPt17: 1000A 20KeV 10E1 80º

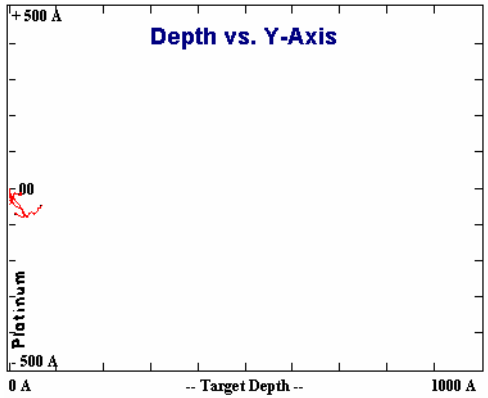


Figura GaPt18: 1000A 20KeV 10E1 88º

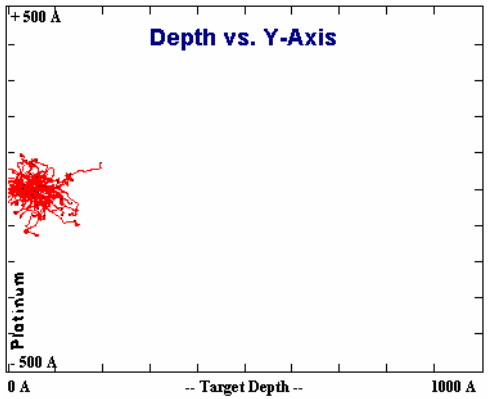


Figura GaPt19: 1000A 20KeV 10E2 0º

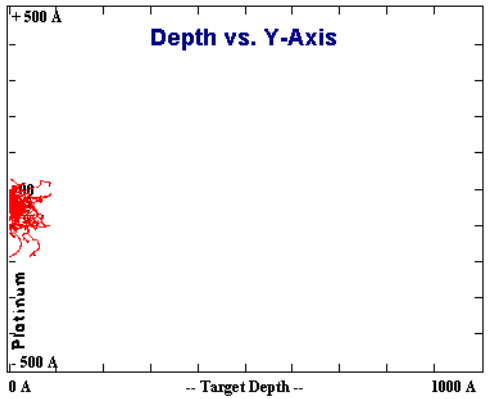


Figura GaPt20: 1000A 20KeV 10E2 80º

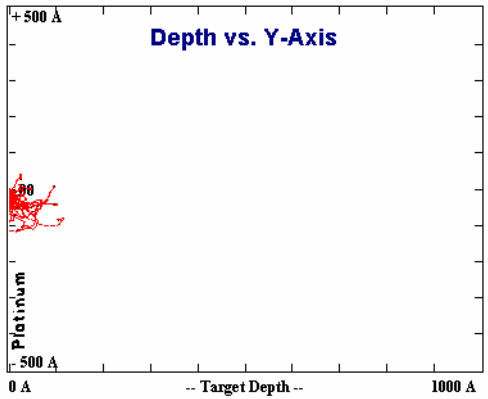


Figura GaPt21: 1000A 20KeV 10E2 88º

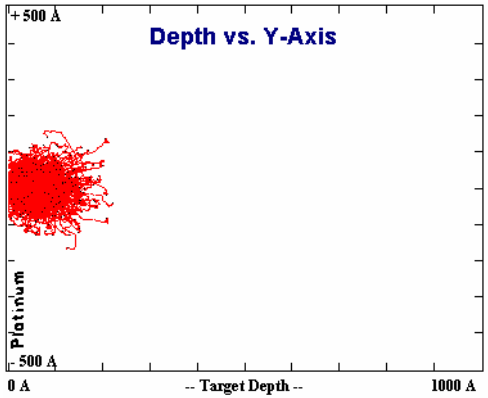


Figura GaPt22: 1000A 20KeV 10E3 0º

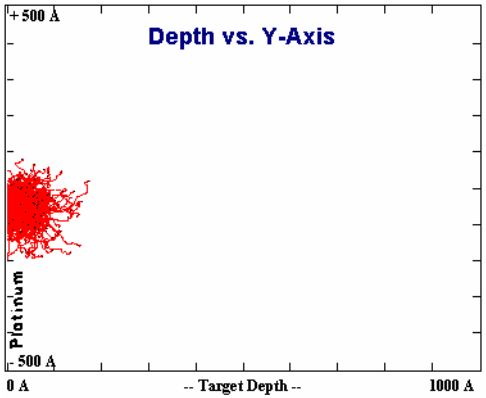


Figura GaPt23: 1000A 20KeV 10E3 80º

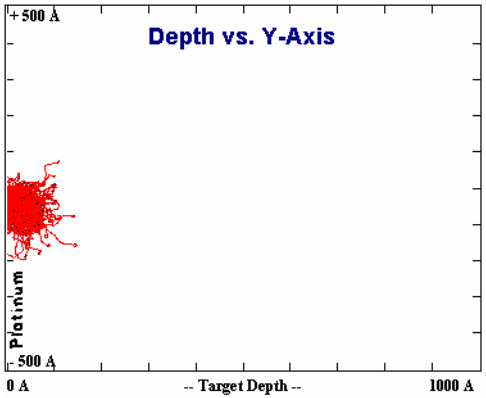


Figura GaPt24: 1000A 20KeV 10E3 88º

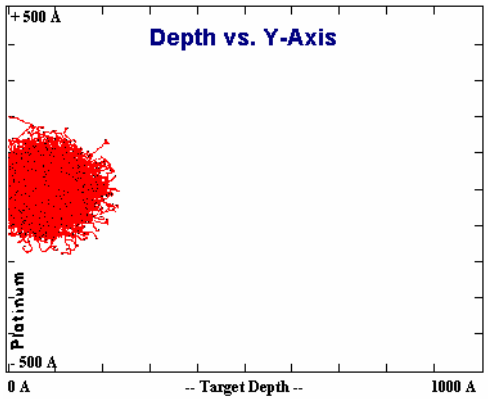


Figura GaPt25: 1000A 20KeV 10E4 0º

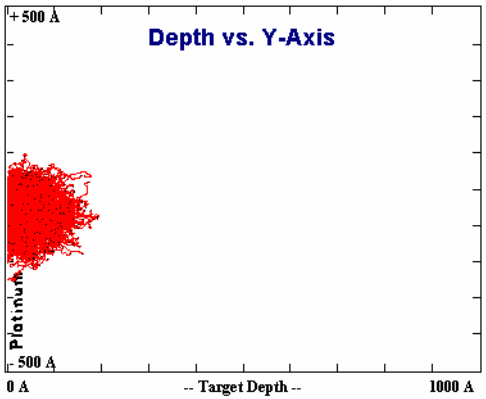


Figura GaPt26: 1000A 20KeV 10E4 80º

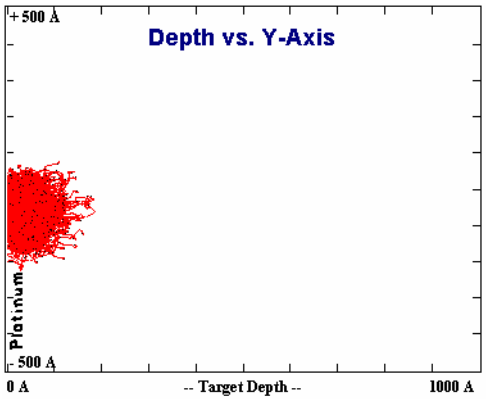


Figura GaPt27 1000A 20KeV 10E4 88º

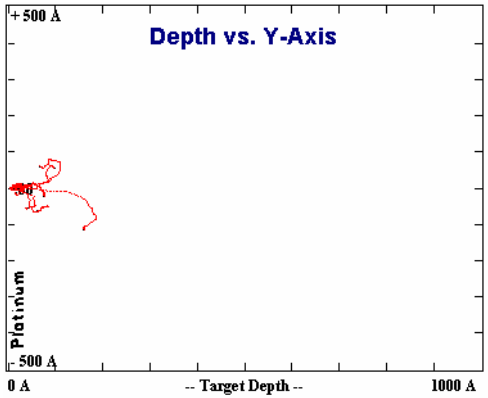


Figura GaPt31: 1000A 30KeV 10E1 0º

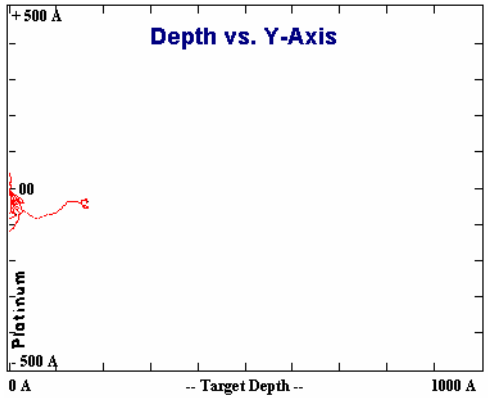


Figura GaPt32: 1000A 30KeV 10E1 80º

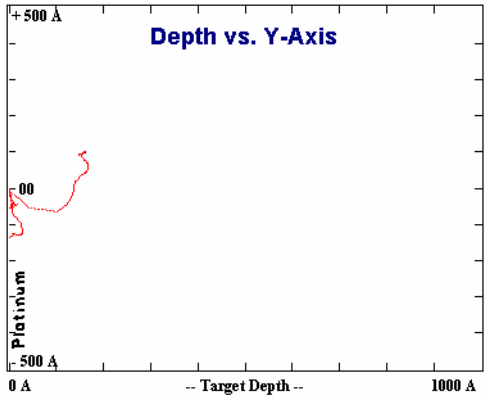


Figura GaPt33: 1000A 30KeV 10E1 88º

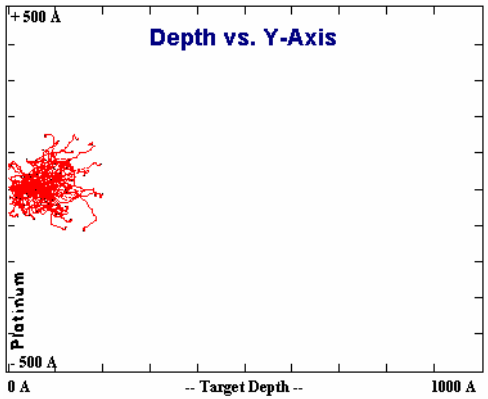


Figura GaPt34: 1000A 30KeV 10E2 0º

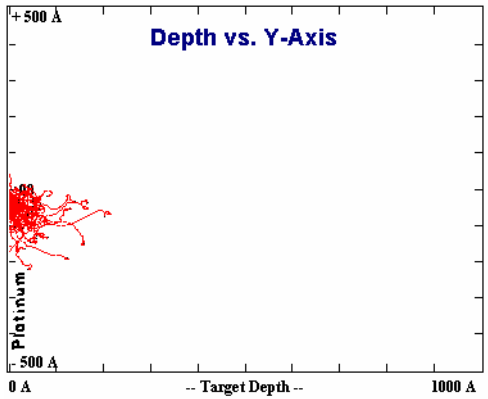


Figura GaPt35: 1000A 30KeV 10E2 80º

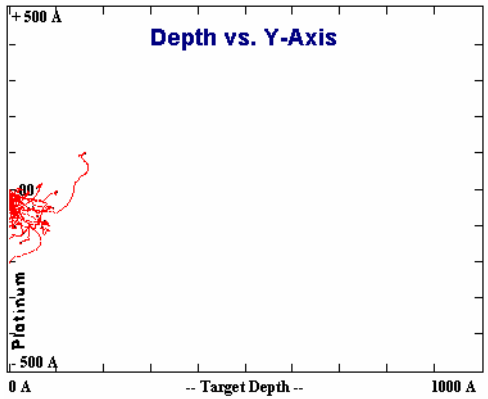


Figura GaPt36:C 1000A 30KeV 10E2 88º

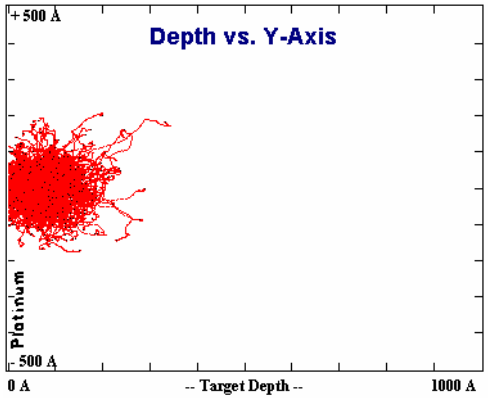


Figura GaPt37: 1000A 30KeV 10E3 0º

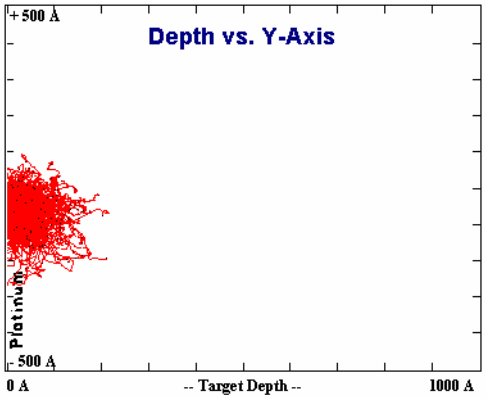


Figura GaPt38: 1000A 30KeV 10E3 80º

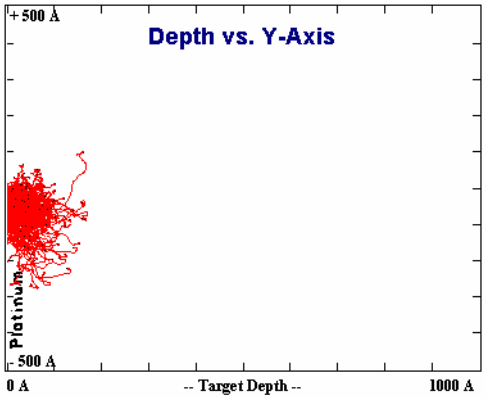


Figura GaPt39: 1000A 30KeV 10E3 88º

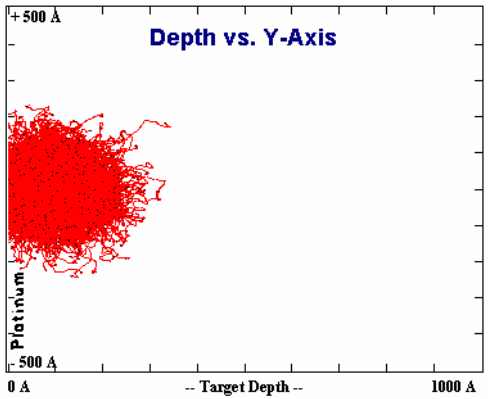


Figura GaPt40: 1000A 30KeV 10E4 0º

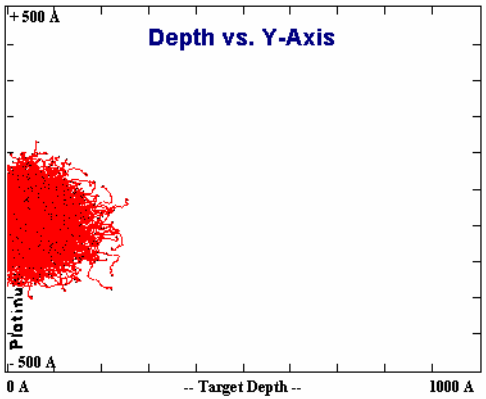


Figura GaPt41: 1000A 30KeV 10E4 80º

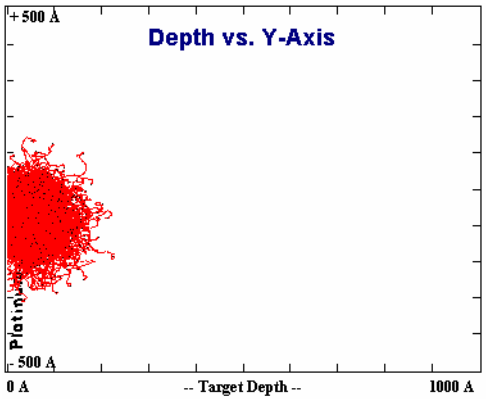


Figura GaPt42: 1000A 30KeV 10E4 88º

SRIM/TRIM: Cálculo detalhado com colisões em cascata

MATERIAL ION: Galio (Gallium) TARGET: Platina (Platinum) WIDTH: 1000A = 100nm

Energia	Número Total de Ions	Ângulo	Parâmetros de espalhamento						Perdas de energia						Resultados			
			Longitudinal		Lateral		Radial		Ionização		Vacâncias		Phonons		Sputtering Yield		Figura	Tempo
			Alcance	Desvio	Alcance	Desvio	Alcance	Desvio	Ions	Recoils	Ions	Recoils	Ions	Recoils	Atoms/Ion	eV/Atom		
10KeV	10E1	0	48A	16A	22A	28A	23A	16A	5,57	36,45	0,55	7,04	1,81	48,58	15,20	67,26	GaPt 46	0'02"
		80	20A	15A	55A	56A	61A	5A	8,89	36,28	0,73	6,96	2,14	45,00	11,10	73,34	GaPt 47	0'02"
		88	n/a	n/a	n/a	n/a	n/a	n/a	12,08	39,76	0,58	6,67	1,80	39,11	3,10	278,76	GaPt 48	0'01"
	10E2	0	44A	26A	22A	28A	32A	20A	6,12	33,19	0,59	6,85	1,79	51,45	8,98	67,57	GaPt 49	0'09"
		80	22A	17A	40A	46A	49A	32A	7,20	38,21	0,65	6,88	1,96	45,10	13,71	99,40	GaPt 50	0'07"
		88	23A	15A	30A	37A	42A	35A	6,55	40,98	0,59	7,12	1,64	43,11	9,14	147,61	GaPt 51	0'04"
	10E3	0	42A	26A	21A	28A	33A	20A	5,84	33,62	0,60	6,86	1,77	51,31	9,82	69,35	GaPt 52	1'25"
		80	28A	19A	37A	44A	47A	26A	7,02	38,58	0,62	6,93	1,81	45,04	14,41	115,73	GaPt 53	1'05"
		88	26A	18A	35A	44A	44A	28A	7,41	39,41	0,64	6,89	1,81	43,84	9,42	138,22	GaPt 54	0'43"
	10E4	0	41A	25A	21A	27A	33A	20A	5,92	33,74	0,60	6,87	1,76	51,11	10,21	70,23	GaPt 55	14'00"
		80	28A	19A	36A	44A	45A	26A	7,05	38,82	0,62	6,93	1,79	44,78	14,90	101,11	GaPt 56	10'56"
		88	28A	19A	35A	44A	45A	25A	7,60	38,87	0,65	6,84	1,88	44,16	9,39	136,12	GaPt 57	6'54"
	10E5	0	41A	25A	21A	27A	33A	20A	5,93	33,70	0,60	6,87	1,76	51,15	10,19	68,78	GaPt 58	121'50"
		80	28A	19A	36A	44A	45A	25A	7,03	38,89	0,62	6,92	1,79	44,75	14,85	104,88	GaPt 59	74'40"
		88	27A	19A	35A	44A	44A	25A	7,55	39,04	0,65	6,85	1,85	44,07	9,51	134,89	GaPt 60	49'53"
20KeV	10E1	0	44A	21A	42A	50A	60A	22A	6,86	32,58	0,44	6,84	1,17	52,11	10,70	51,72	GaPt 61	0'02"
		80	43A	27A	66A	74A	81A	32A	8,76	36,05	0,44	7,00	1,29	46,46	18,30	153,74	GaPt 62	0'02"
		88	55A	51A	69A	70A	86A	21A	9,59	37,98	0,58	6,60	1,70	43,55	20,20	102,92	GaPt 63	0'01"
	10E2	0	62A	37A	35A	46A	57A	31A	6,98	32,89	0,43	6,90	1,26	51,54	13,47	71,62	GaPt 64	0'15"
		80	41A	30A	57A	70A	75A	40A	7,81	36,84	0,44	6,91	1,24	46,76	23,07	98,93	GaPt 65	0'13"
		88	36A	31A	59A	67A	72A	31A	8,27	38,46	0,43	6,90	1,21	44,72	15,40	143,21	GaPt 66	0'08"
	10E3	0	63A	37A	33A	42A	52A	31A	6,54	33,00	0,42	6,92	1,19	51,93	11,62	94,10	GaPt 67	2'18"
		80	42A	29A	59A	72A	72A	39A	7,64	37,29	0,43	6,95	1,22	46,48	21,94	121,33	GaPt 68	1'53"
		88	42A	31A	56A	68A	68A	39A	8,19	37,43	0,44	6,91	1,20	45,83	14,94	145,56	GaPt 69	1'15
	10E4	0	63A	38A	32A	42A	51A	31A	6,48	33,18	0,41	6,93	1,18	51,83	11,79	90,70	GaPt 70	23'09"
		80	43A	29A	57A	70A	71A	40A	7,72	37,50	0,42	6,93	1,21	46,22	21,75	127,06	GaPt 71	18'08"
		88	41A	28A	55A	68A	69A	40A	8,30	37,70	0,44	6,88	1,25	45,43	14,11	167,84	GaPt 72	10'46"
	10E5	0	63A	39A	33A	42A	51A	31A	6,50	33,21	0,41	6,92	1,19	51,77	12,02	91,71	GaPt 73	200'32"
		80	43A	29A	57A	70A	71A	40A	7,70	37,53	0,43	6,93	1,21	46,20	21,81	128,59	GaPt 74	120'14"
		88	41A	28A	55A	68A	69A	40A	8,32	37,67	0,44	6,87	1,26	45,44	13,94	166,55	GaPt 75	81'12"
30KeV	10E1	0	100A	60A	59A	80A	104A	54A	8,31	31,97	0,40	6,85	1,06	51,41	12,40	127,35	GaPt 76	0'02"
		80	76A	53A	87A	88A	100A	37A	8,14	37,15	0,36	6,93	0,96	46,46	33,60	105,37	GaPt 77	0'02"
		88	85A	75A	50A	57A	58A	20A	9,46	34,18	0,36	6,80	1,31	47,89	6,40	140,16	GaPt 78	0'01"
	10E2	0	84A	53A	43A	55A	69A	39A	7,06	32,57	0,34	6,90	0,94	52,19	9,76	69,86	GaPt 79	0'20"
		80	59A	47A	87A	104A	101A	59A	8,58	37,21	0,34	6,95	0,92	46,00	30,13	123,18	GaPt 80	0'18"
		88	66A	43A	84A	98A	111A	63A	9,78	37,29	0,36	6,69	1,11	44,77	15,88	199,72	GaPt 81	0'09"
	10E3	0	87A	53A	44A	57A	67A	40A	7,06	32,88	0,33	6,91	0,96	51,86	12,19	110,30	GaPt 82	2'12"
		80	57A	36A	76A	93A	92A	52A	8,12	36,58	0,34	6,91	0,97	47,08	26,11	139,85	GaPt 83	2'30"
		88	55A	36A	79A	94A	96A	53A	8,63	37,01	0,35	6,87	1,01	46,12	18,43	178,90	GaPt 84	1'40"
	10E4	0	84A	51A	43A	55A	67A	40A	6,97	33,03	0,33	6,91	0,95	51,81	12,45	105,85	GaPt 85	31'54"
		80	56A	38A	75A	91A	92A	52A	8,23	36,76	0,34	6,90	0,97	46,81	26,34	143,77	GaPt 86	24'52"
		88	53A	35A	75A	92A	92A	53A	8,87	37,00	0,35	6,84	1,00	45,94	17,48	186,49	GaPt 87	15"54"
	10E5	0	83A	51A	43A	55A	67A	40A	6,96	33,10	0,33	6,91	0,95	51,75	12,80	109,63	GaPt 88	211'46"
		80	56A	38A	75A	91A	93A	52A	8,24	36,80	0,34	6,90	0,97	46,76	26,67	144,47	GaPt 89	217'20"
		88	54A	37A	73A	90A	91A	52A	8,90	36,95	0,35	6,83	1,01	45,95	17,11	189,40	GaPt 90	102'52"

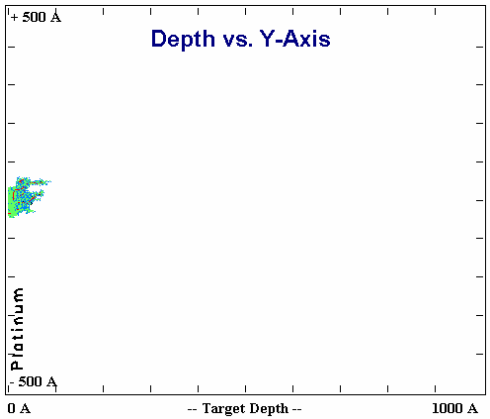


Figura GaPt46: 1000A 10KeV 10E1 0º

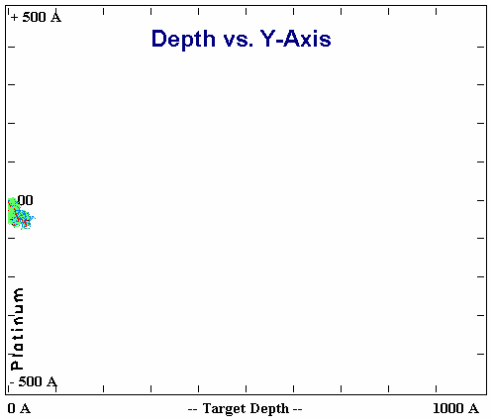


Figura GaPt47: 1000A 10KeV 10E1 80º

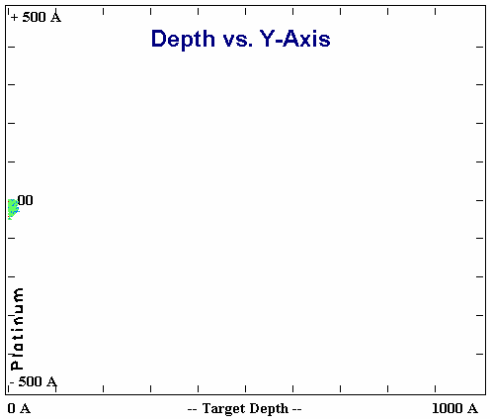


Figura GaPt48: 1000A 10KeV 10E1 88º

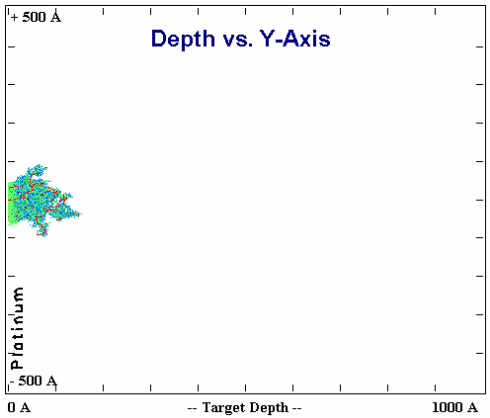


Figura GaPt49: 1000A 10KeV 10E2 0º

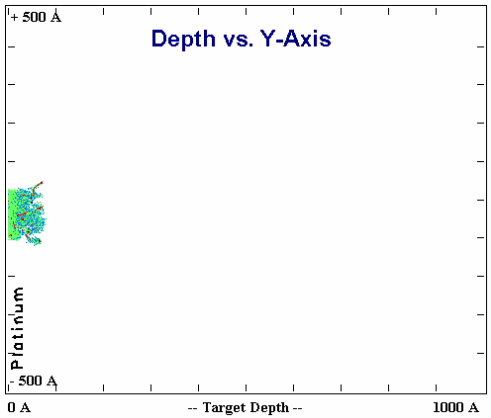


Figura GaPt50: 1000A 10KeV 10E2 80º

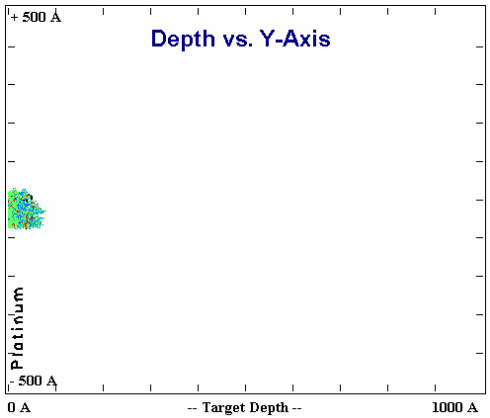


Figura GaPt51: 1000A 10KeV 10E2 88º

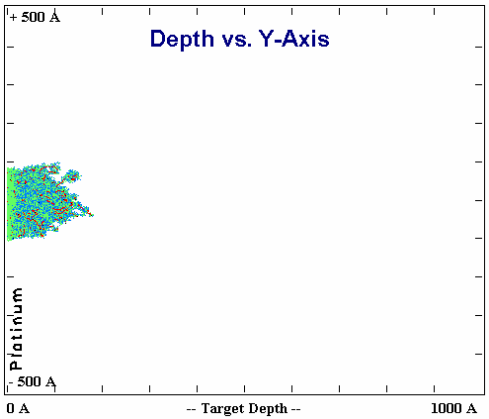


Figura GaPt52: 1000A 10KeV 10E3 0º

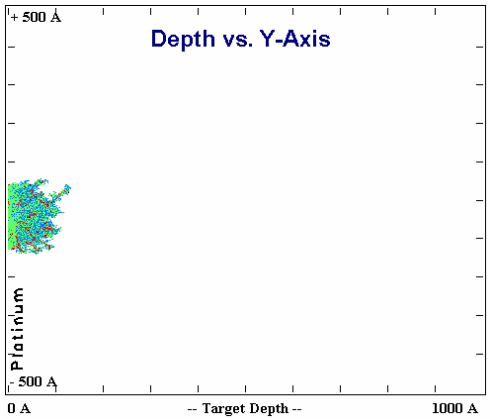


Figura GaPt53: 1000A 10KeV 10E3 80º

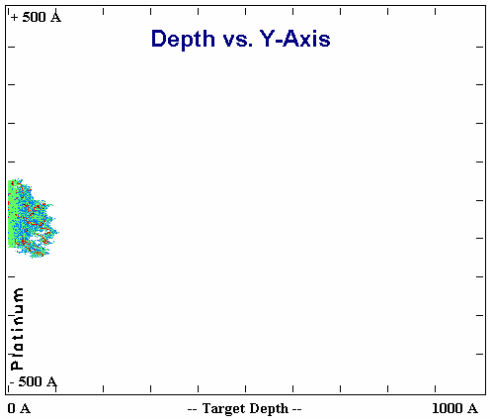


Figura GaPt54: 1000A 10KeV 10E3 88º

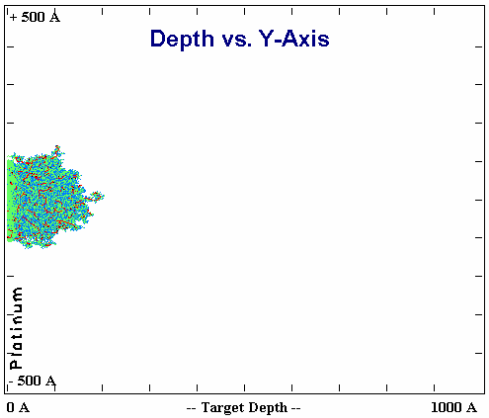


Figura GaPt55: 1000A 10KeV 10E4 0º

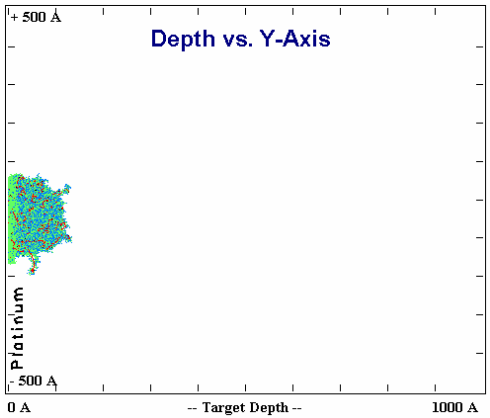


Figura GaPt56: 1000A 10KeV 10E4 80º

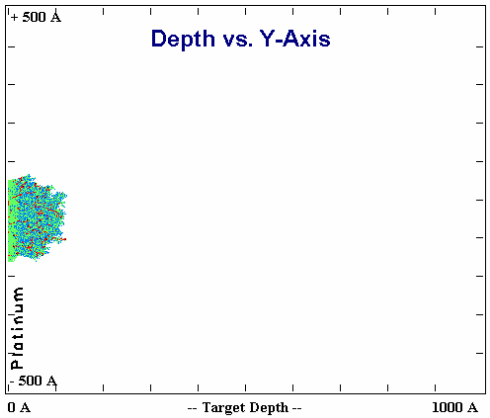


Figura GaPt57: 1000A 10KeV 10E4 88º

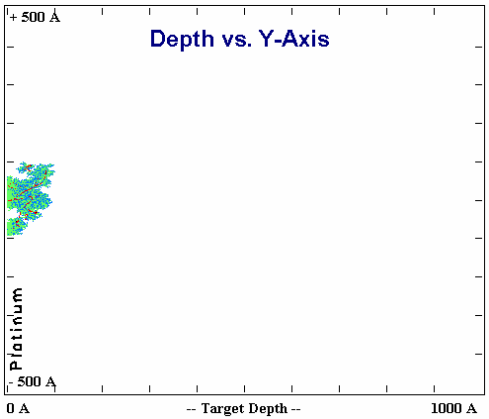


Figura GaPt61: 1000A 20KeV 10E1 0º

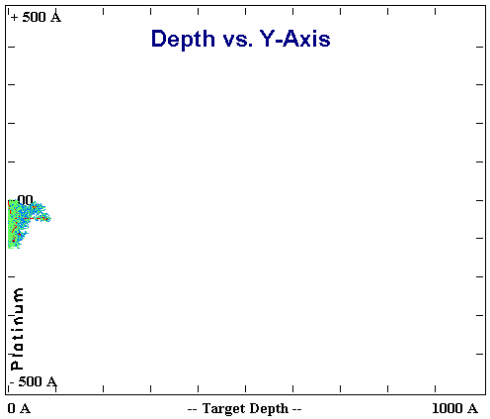


Figura GaPt62: 1000A 20KeV 10E1 80º

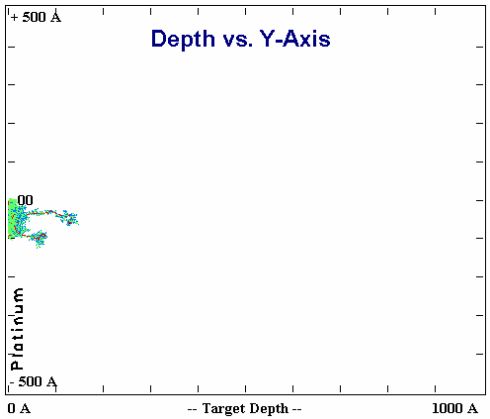


Figura GaPt63: 1000A 20KeV 10E1 88º

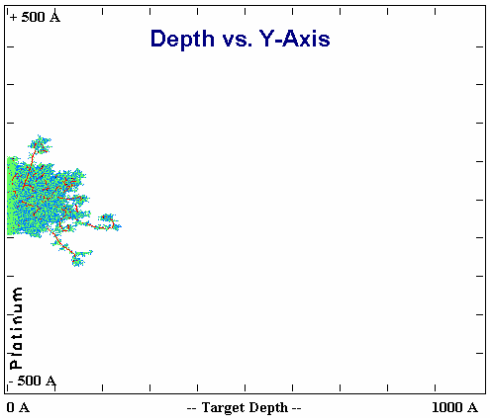


Figura GaPt64: 1000A 20KeV 10E2 0º

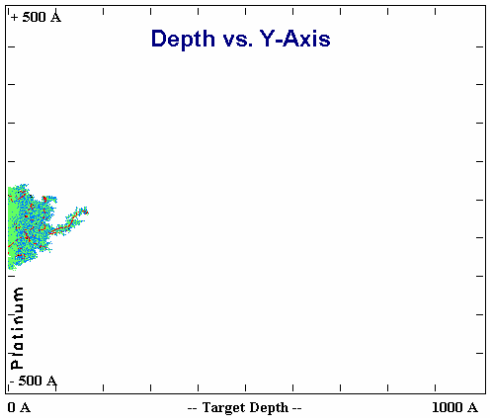


Figura GaPt65: 1000A 20KeV 10E2 80º

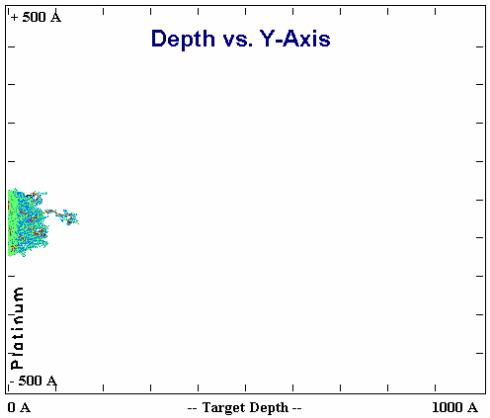


Figura GaPt66: 1000A 20KeV 10E2 88º

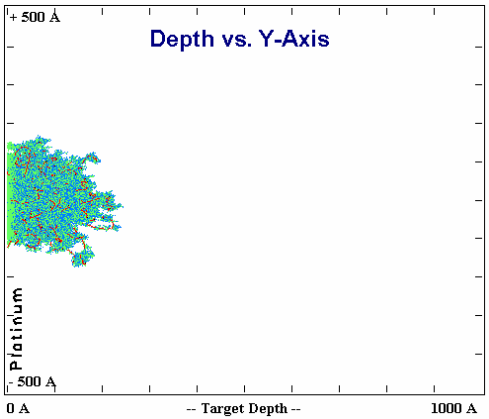


Figura GaPt67: 1000A 20KeV 10E3 0º

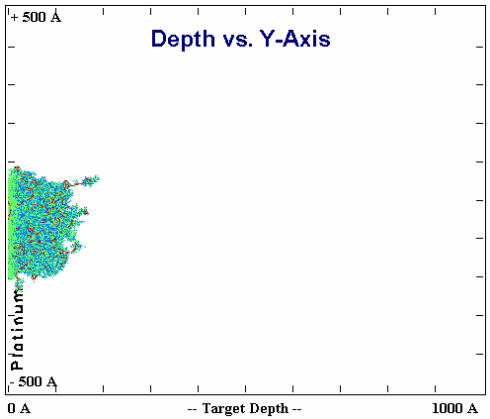


Figura GaPt68: 1000A 20KeV 10E3 80º

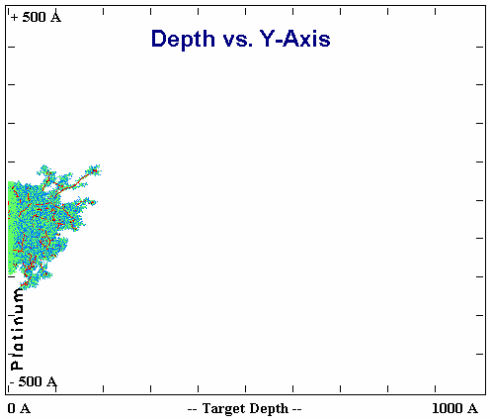


Figura GaPt69: 1000A 20KeV 10E3 88º

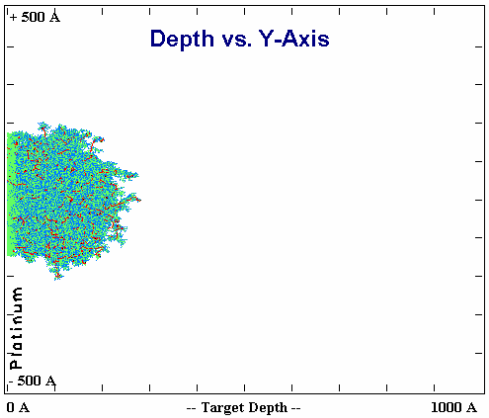


Figura GaPt70: 1000A 20KeV 10E4 0º

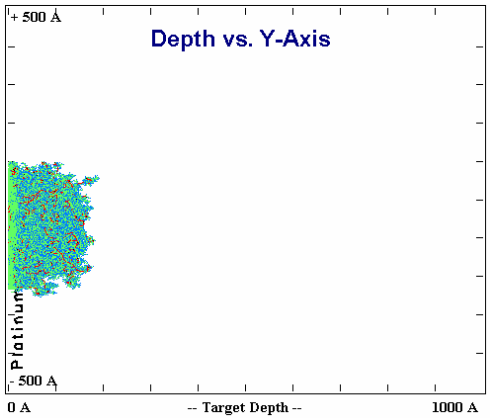


Figura GaPt71: 1000A 20KeV 10E4 80º

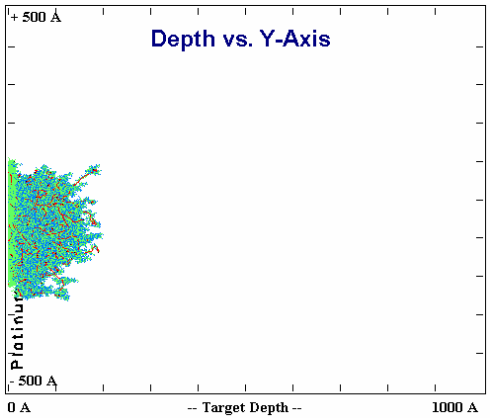


Figura GaPt72: 1000A 20KeV 10E4 88º

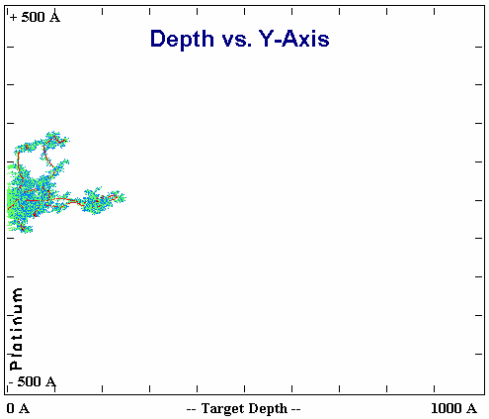


Figura GaPt76: 1000A 30KeV 10E1 0º

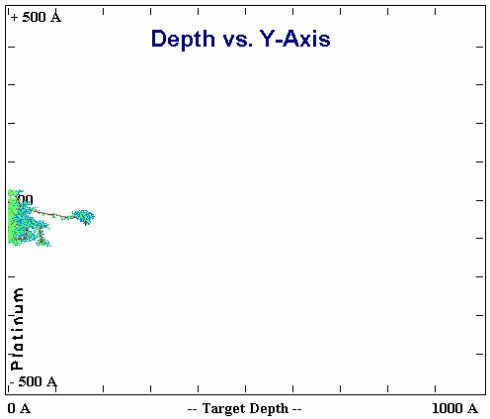


Figura GaPt77: 1000A 30KeV 10E1 80º

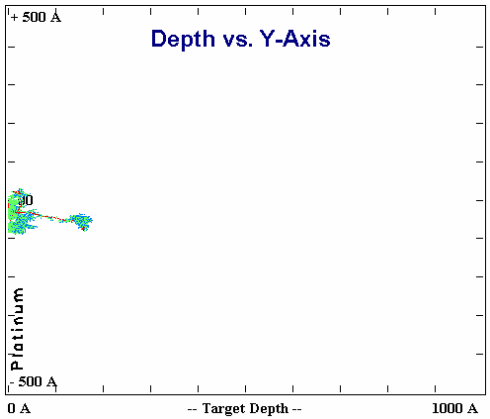


Figura GaPt78: 1000A 30KeV 10E1 88º

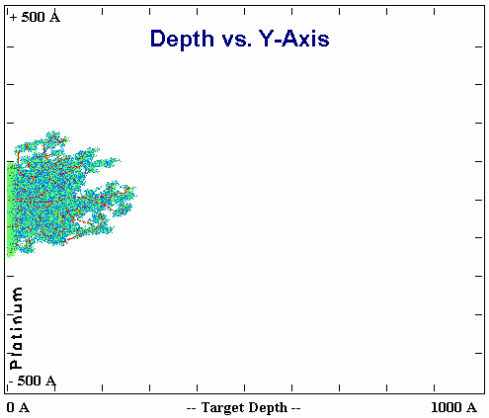


Figura GaPt79: 1000A 30KeV 10E2 0º

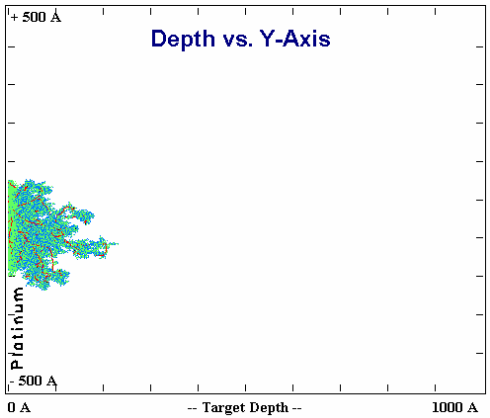


Figura GaPt80: 1000A 30KeV 10E2 80º

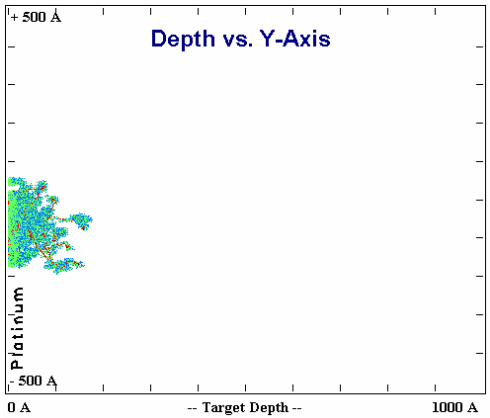


Figura GaPt81: 1000A 30KeV 10E2 88º

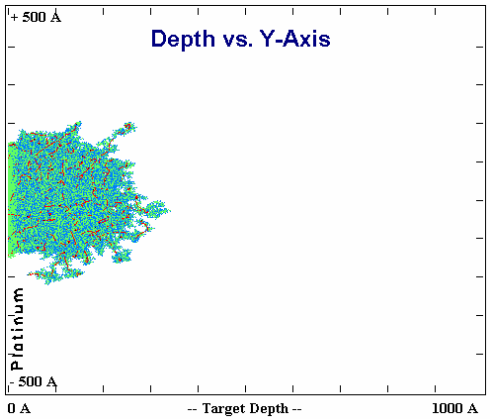


Figura GaPt82: 1000A 30KeV 10E3 0º

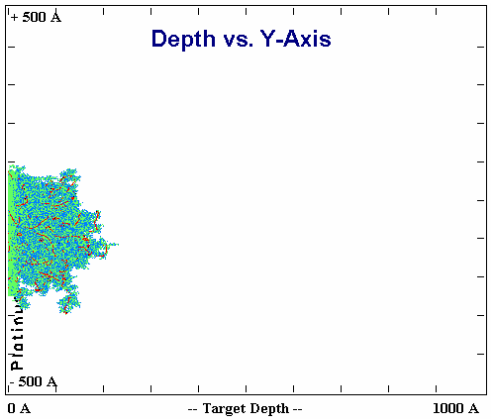


Figura GaPt83: 1000A 30KeV 10E3 80º

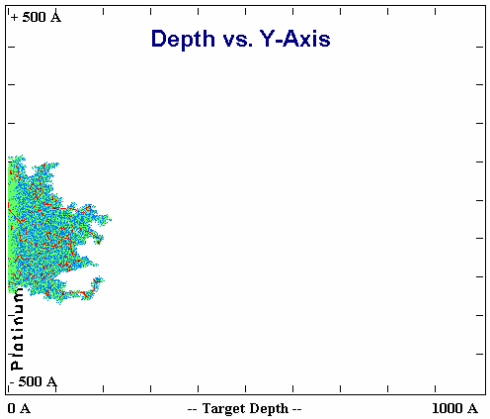


Figura GaPt84: 1000A 30KeV 10E3 88º

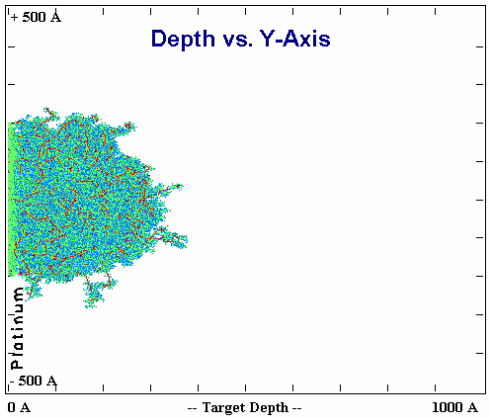


Figura GaPt85: 1000A 30KeV 10E4 0º

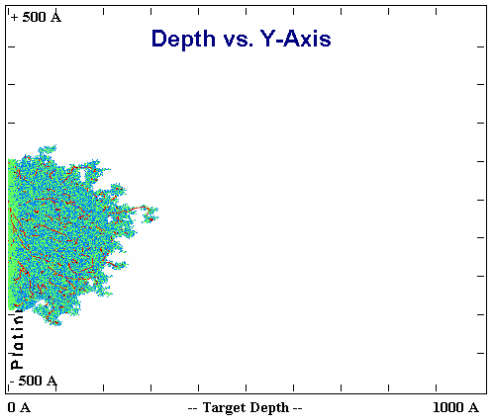


Figura GaPt86: 1000A 30KeV 10E4 80º

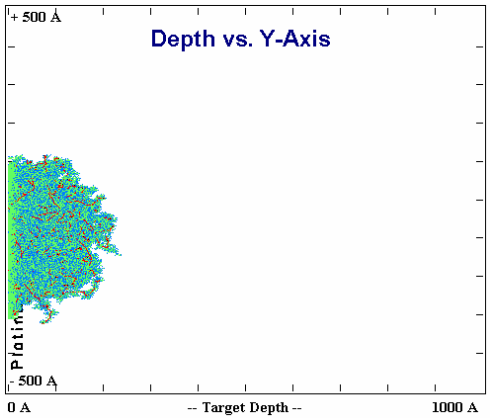


Figura GaPt87: 1000A 30KeV 10E4 88º

SRIM/TRIM: Cálculo rápido de colisões e distribuição de íons

MATERIAL ION: Galio (Gallium) TARGET: Silício (Silicon) WIDTH: 1000A = 100nm

Energia	Número Total de Ions	Ângulo	Parâmetros de espalhamento						Perdas de energia						Resultados			
			Longitudinal		Lateral		Radial		Ionização		Vacâncias		Phonons		Sputtering Yield		Figura	Tempo
			Alcance	Desvio	Alcance	Desvio	Alcance	Desvio	Ions	Recoils	Ions	Recoils	Ions	Recoils	Atoms/Ion	eV/Atom		
10KeV	10E1	0	135A	36A	22A	30A	56A	31A	6,91	19,31	0,53	3,31	1,38	68,55	0,00	0,00	GaSi 01	0'01"
		80	48A	31A	116A	127A	122A	50A	6,01	19,92	0,45	3,42	1,21	68,99	0,00	0,00	GaSi 02	0'01"
		88	42A	35A	162A	166A	171A	30A	9,00	18,82	0,55	3,21	1,30	67,12	0,00	0,00	GaSi 03	0'01"
	10E2	0	125A	44A	31A	40A	50A	28A	6,15	20,23	0,48	3,35	1,39	68,40	0,00	0,00	GaSi 04	0'02"
		80	39A	27A	120A	130A	127A	49A	5,95	20,66	0,45	3,40	1,11	68,43	0,00	0,00	GaSi 05	0'02"
		88	36A	31A	127A	134A	136A	42A	6,10	20,74	0,44	3,39	1,11	68,22	0,00	0,00	GaSi 06	0'02"
	10E3	0	129A	46A	31A	39A	48A	27A	6,24	20,02	0,50	3,34	1,33	68,57	0,00	0,00	GaSi 07	0'12"
		80	44A	29A	127A	136A	133A	49A	6,23	20,32	0,47	3,37	1,20	68,41	0,00	0,00	GaSi 08	0'10"
		88	37A	25A	127A	136A	133A	49A	6,30	20,70	0,43	3,39	1,10	68,09	0,00	0,00	GaSi 09	0'08"
	10E4	0	128A	47A	31A	40A	49A	27A	6,15	20,11	0,50	3,35	1,33	68,57	0,00	0,00	GaSi 10	1'56"
		80	44A	29A	128A	136A	134A	47A	6,25	20,35	0,46	3,37	1,19	68,38	0,00	0,00	GaSi 11	1'40"
		88	39A	27A	128A	136A	134A	47A	6,33	20,63	0,43	3,39	1,10	68,12	0,00	0,00	GaSi 12	1'18"
	10E5	0	129A	47A	31A	40A	49A	28A	6,18	20,08	0,50	3,35	1,33	68,56	0,00	0,00	GaSi 13	20'56"
		80	44A	29A	128A	136A	134A	47A	6,28	20,33	0,47	3,37	1,19	68,37	0,00	0,00	GaSi 14	16'00"
		88	39A	26A	128A	136A	134A	47A	6,39	20,57	0,44	3,39	1,10	68,11	0,00	0,00	GaSi 15	11'35"
20KeV	10E1	0	185A	75A	43A	51A	74A	31A	6,63	22,32	0,34	3,40	0,80	66,51	0,00	0,00	GaSi 16	0'01"
		80	78A	59A	181A	195A	185A	69A	6,56	22,34	0,31	3,43	0,79	66,57	0,00	0,00	GaSi 17	0'01"
		88	112A	10A	223A	224A	228A	18A	10,44	20,43	0,41	3,23	0,84	64,66	0,00	0,00	GaSi 18	0'01"
	10E2	0	196A	77A	54A	66A	82A	43A	7,06	22,04	0,35	3,38	0,89	66,29	0,00	0,00	GaSi 19	0'02"
		80	63A	43A	192A	206A	198A	75A	6,69	22,73	0,31	3,40	0,80	66,07	0,00	0,00	GaSi 20	0'02"
		88	65A	37A	204A	218A	210A	79A	7,50	22,19	0,31	3,39	0,74	65,87	0,00	0,00	GaSi 21	0'02"
	10E3	0	203A	74A	47A	60A	74A	41A	7,03	21,99	0,35	3,38	0,87	66,38	0,00	0,00	GaSi 22	0'15"
		80	71A	47A	205A	218A	214A	75A	7,35	22,05	0,33	3,38	0,80	66,09	0,00	0,00	GaSi 23	0'10"
		88	63A	48A	201A	215A	209A	77A	7,35	22,49	0,30	3,39	0,73	65,73	0,00	0,00	GaSi 24	0'09"
	10E4	0	207A	76A	48A	62A	76A	44A	7,20	21,85	0,35	3,38	0,87	66,35	0,00	0,00	GaSi 25	2'26"
		80	69A	47A	204A	217A	213A	75A	7,30	22,14	0,33	3,39	0,79	66,05	0,00	0,00	GaSi 26	2'03"
		88	61A	42A	206A	220A	215A	77A	4,45	22,38	0,31	3,39	0,74	65,73	0,00	0,00	GaSi 27	1'27"
	10E5	0	207A	77A	48A	62A	76A	44A	7,20	21,86	0,35	3,38	0,87	66,34	0,00	0,00	GaSi 28	24'25"
		80	68A	46A	204A	218A	213A	77A	7,33	22,12	0,33	3,38	0,79	66,05	0,00	0,00	GaSi 29	20'10"
		88	60A	41A	205A	219A	214A	77A	7,49	22,37	0,31	3,39	0,74	65,71	0,00	0,00	GaSi 30	17'45"
30KeV	10E1	0	265A	77A	77A	91A	106A	52A	8,00	22,25	0,31	3,37	0,68	65,40	0,00	0,00	GaSi 31	0'01"
		80	54A	49A	272A	288A	279A	100A	8,34	22,76	0,28	3,35	0,61	64,66	0,00	0,00	GaSi 32	0'01"
		88	97A	72A	295A	296A	306A	28A	9,30	22,44	0,28	3,33	0,53	64,11	0,00	0,00	GaSi 33	0'01"
	10E2	0	266A	87A	66A	84A	104A	56A	7,95	22,84	0,28	3,36	0,70	64,86	0,00	0,00	GaSi 34	0'02"
		80	77A	56A	281A	297A	295A	92A	8,26	23,11	0,26	3,36	0,61	64,39	0,00	0,00	GaSi 35	0'02"
		88	86A	62A	276A	294A	287A	99A	8,26	23,49	0,24	3,36	0,58	64,07	0,00	0,00	GaSi 36	0'02"
	10E3	0	279A	99A	64A	81A	98A	54A	8,03	22,83	0,29	3,35	0,69	64,80	0,00	0,00	GaSi 37	0'17"
		80	92A	61A	274A	291A	287A	99A	8,21	23,14	0,27	3,35	0,63	64,40	0,00	0,00	GaSi 38	0'15"
		88	79A	55A	273A	291A	284A	100A	8,29	23,42	0,25	3,35	0,60	64,09	0,00	0,00	GaSi 39	0'10"
	10E4	0	278A	103A	64A	81A	99A	57A	8,00	22,92	0,29	3,35	0,69	64,75	0,00	0,00	GaSi 40	2'52"
		80	91A	62A	275A	293A	287A	101A	8,14	23,19	0,27	3,35	0,63	64,42	0,00	0,00	GaSi 41	2'17"
		88	79A	56A	275A	294A	286A	104A	8,33	23,45	0,25	3,35	0,59	64,03	0,00	0,00	GaSi 42	1'35"
	10E5	0	279A	103A	63A	81A	100A	58A	8,02	22,92	0,29	3,35	0,70	64,73	0,00	0,00	GaSi 43	26'09"
		80	90A	61A	275A	293A	287A	103A	8,18	23,17	0,27	3,35	0,63	64,39	0,00	0,00	GaSi 44	21'26"
		88	79A	55A	276A	294A	287A	104A	8,40	23,43	0,25	3,35	0,60	63,99	0,00	0,00	GaSi 45	14'27"

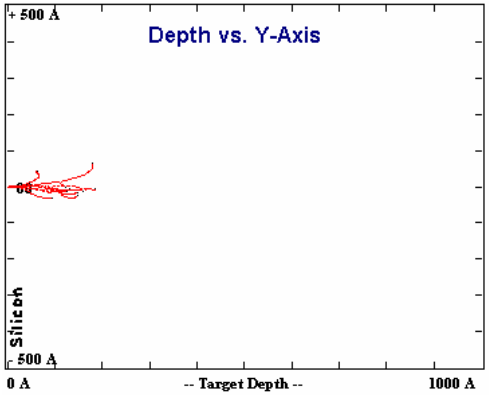


Figura GaSi01: 1000A 10KeV 10E1 0º

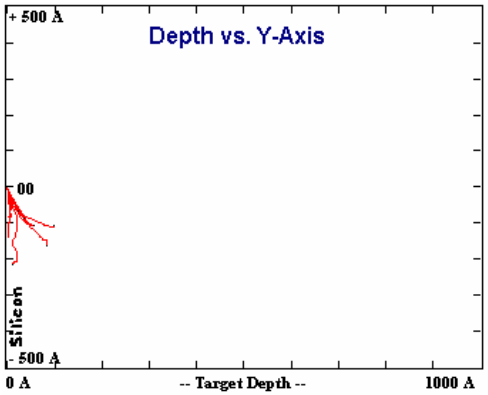


Figura GaSi02: 1000A 10KeV 10E1 80º

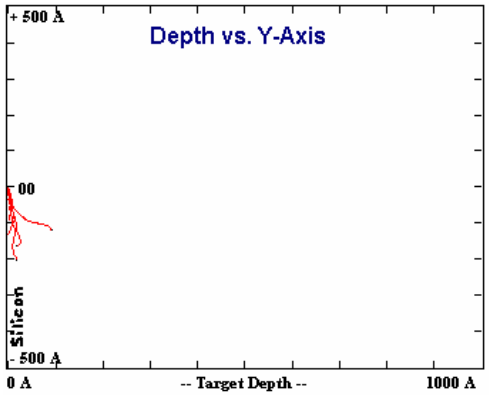


Figura GaSi03: 1000A 10KeV 10E1 88º

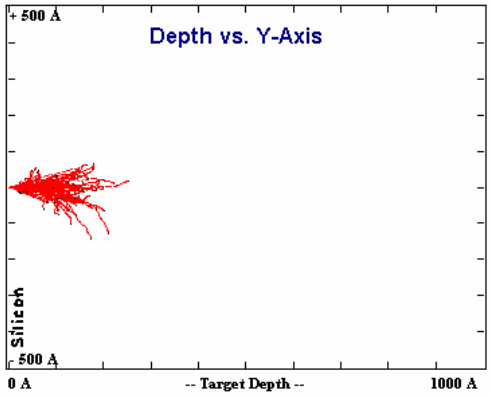


Figura GaSi04: 1000A 10KeV 10E2 0º

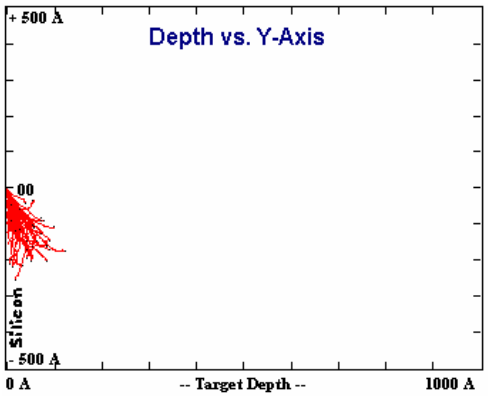


Figura GaSi05: 1000A 10KeV 10E2 80º

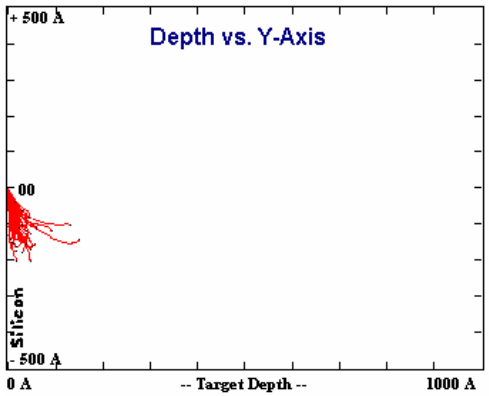


Figura GaSi06: 1000A 10KeV 10E2 88º

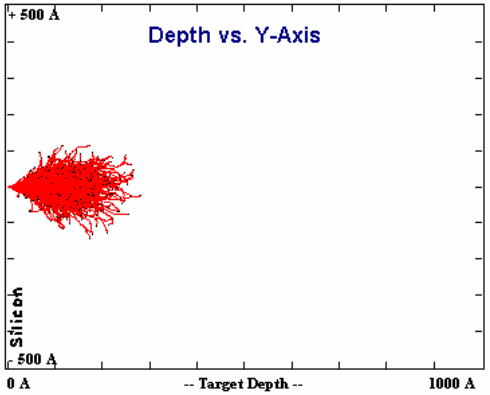


Figura GaSi07: 1000A 10KeV 10E3 0º

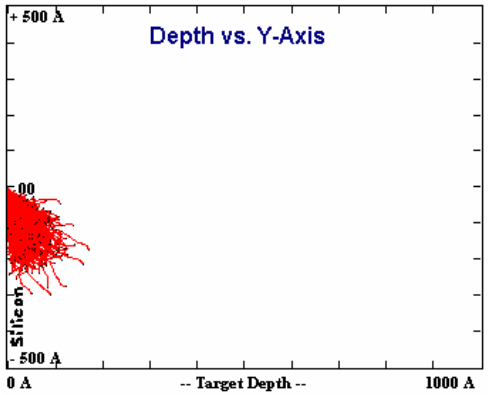


Figura GaSi08: 1000A 10KeV 10E3 80º

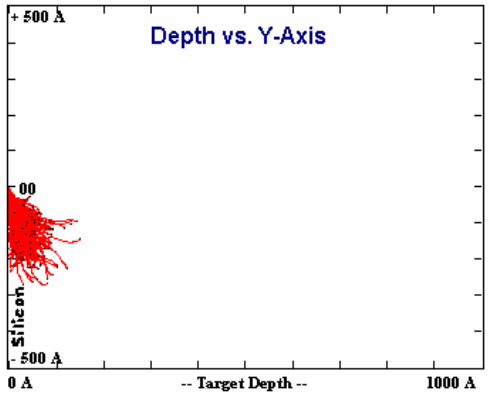


Figura GaSi09: 1000A 10KeV 10E3 88º

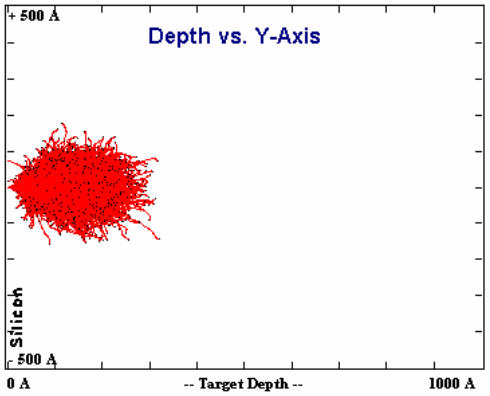


Figura GaSi10: 1000A 10KeV 10E4 0º

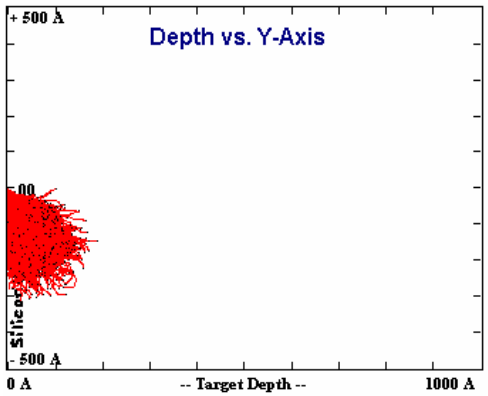


Figura GaSi11: 1000A 10KeV 10E4 80º

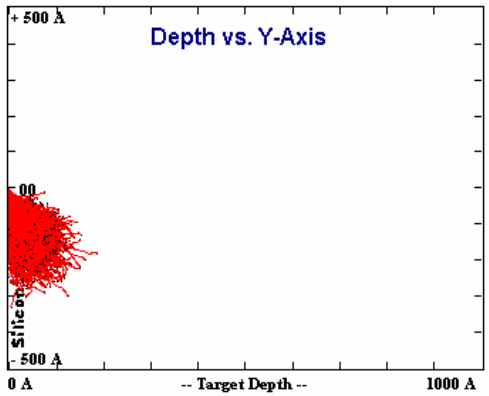


Figura GaSi12: 1000A 10KeV 10E4 88º

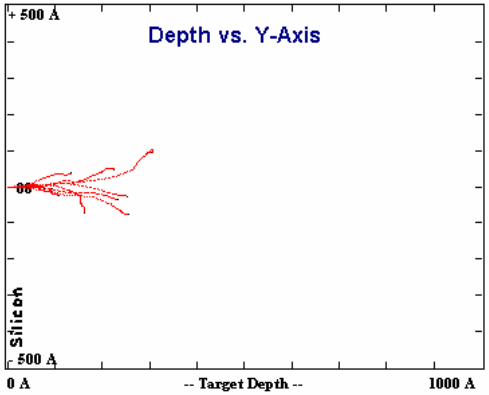


Figura GaSi16: 1000A 20KeV 10E1 0º

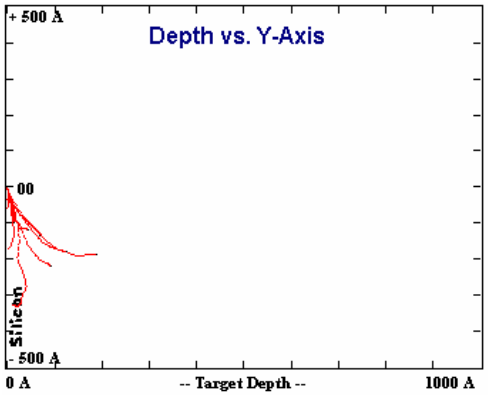


Figura GaSi17: 1000A 20KeV 10E1 80º

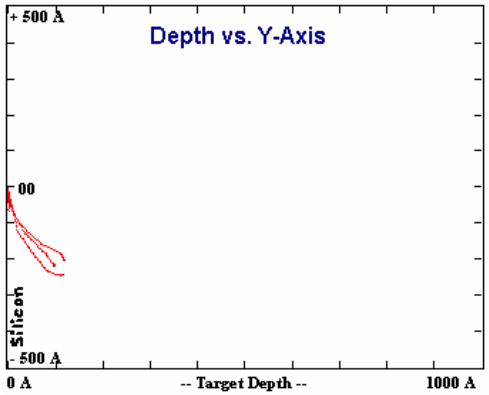


Figura GaSi18: 1000A 20KeV 10E1 88º

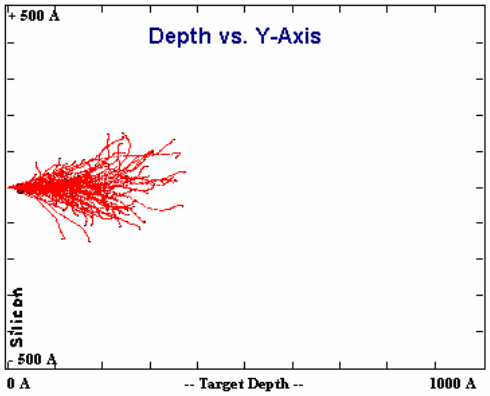


Figura GaSi19: 1000A 20KeV 10E2 0º

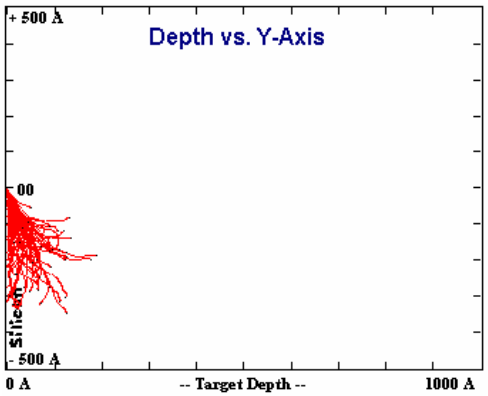


Figura GaSi20: 1000A 20KeV 10E2 80º

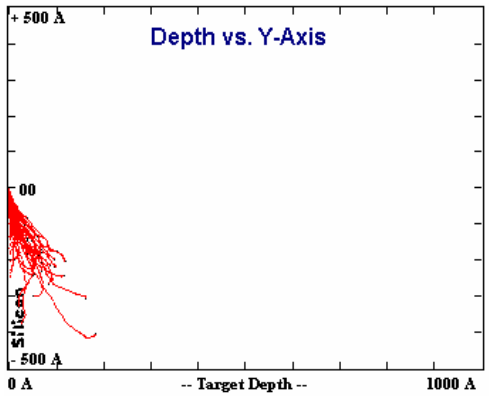


Figura GaSi21: 1000A 20KeV 10E2 88º

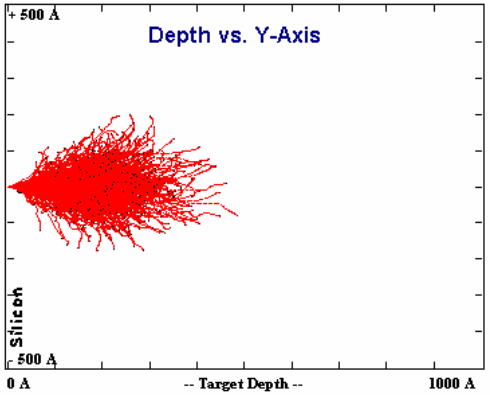


Figura GaSi22: 1000A 20KeV 10E3 0º

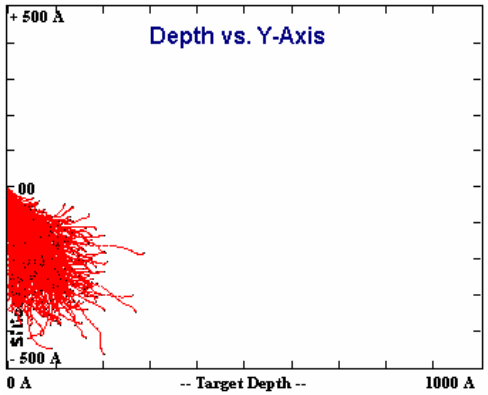


Figura GaSi23: 1000A 20KeV 10E3 80º

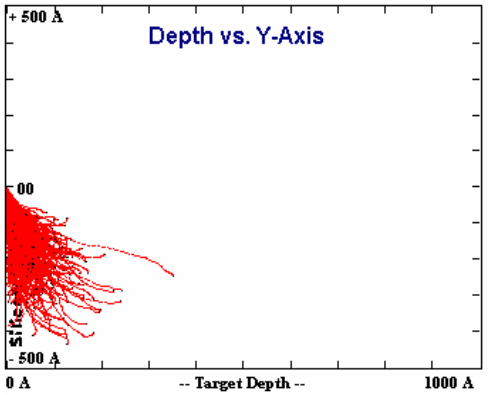


Figura GaSi24: 1000A 20KeV 10E3 88º

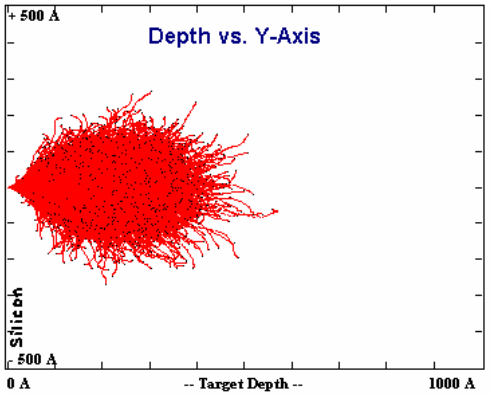


Figura GaSi25: 1000A 20KeV 10E4 0º

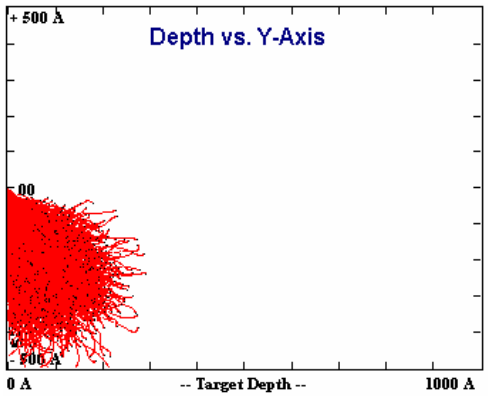


Figura GaSi26: 1000A 20KeV 10E4 80º

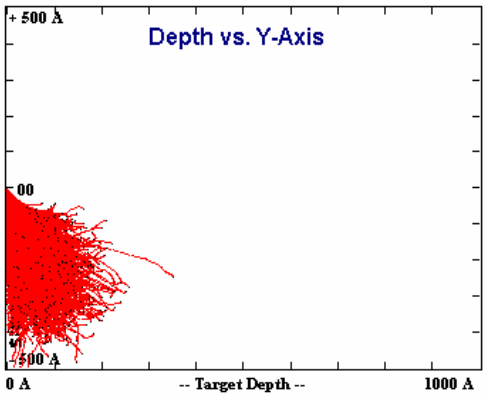


Figura GaSi27: 1000A 20KeV 10E4 88º

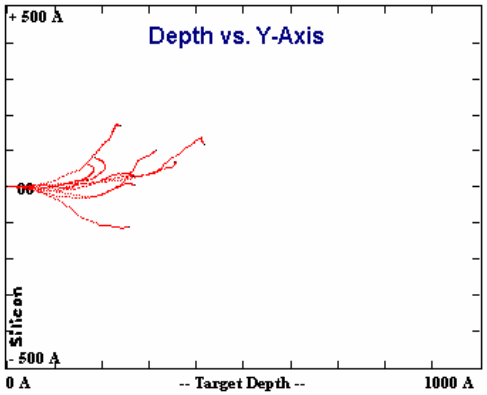


Figura GaSi31: 1000A 30KeV 10E1 0º

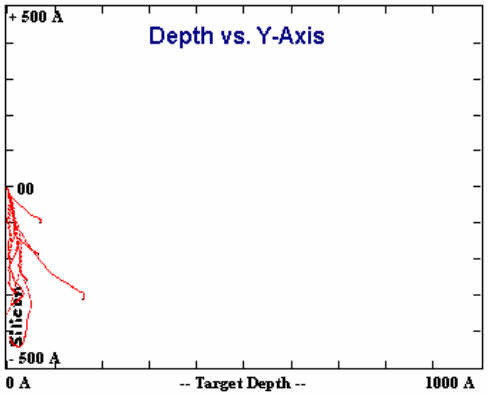


Figura GaSi32: 1000A 30KeV 10E1 80º

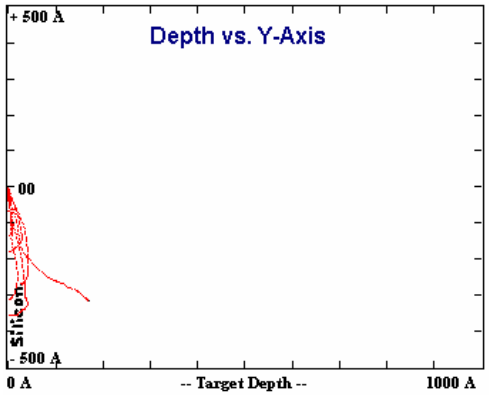


Figura GaSi33: 1000A 30KeV 10E1 88º

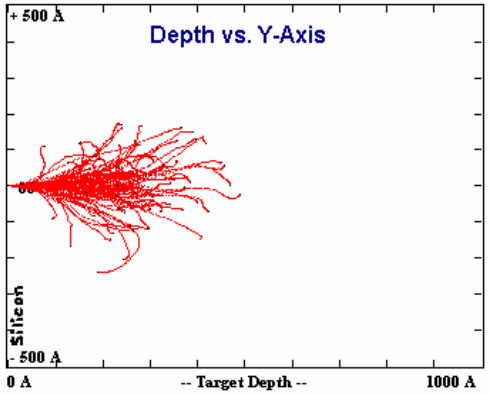


Figura GaSi34: 1000A 30KeV 10E2 0º

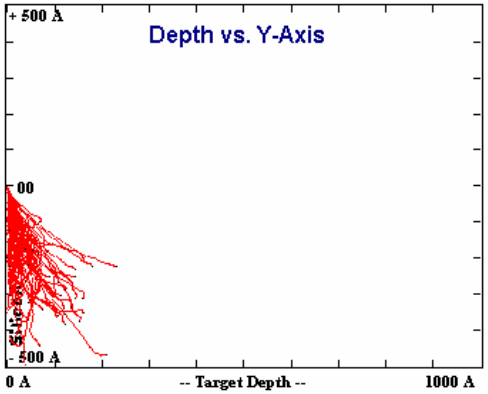


Figura GaSi35: 1000A 30KeV 10E2 80º

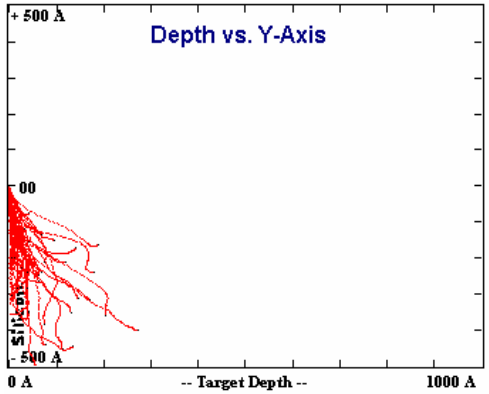


Figura GaSi36: 1000A 30KeV 10E2 88º

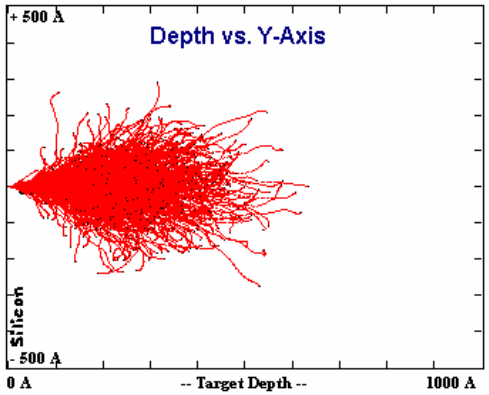


Figura GaSi37: 1000A 30KeV 10E3 0º

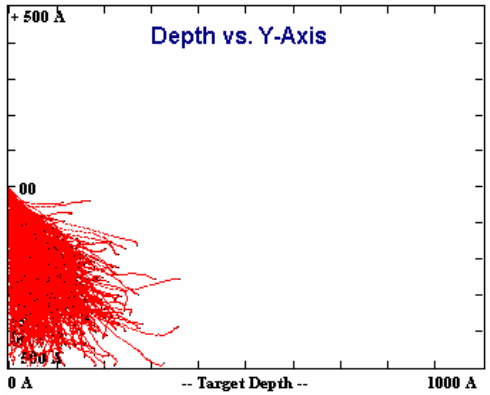


Figura GaSi38: 1000A 30KeV 10E3 80º

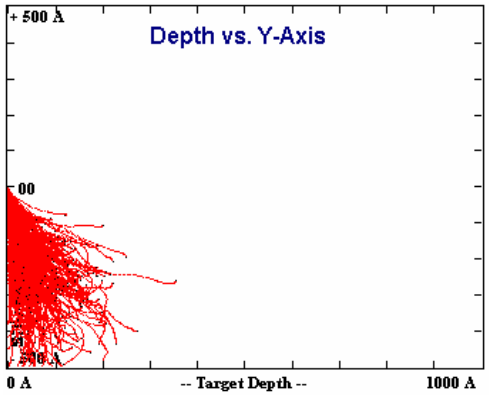


Figura GaSi39: 1000A 30KeV 10E3 88º

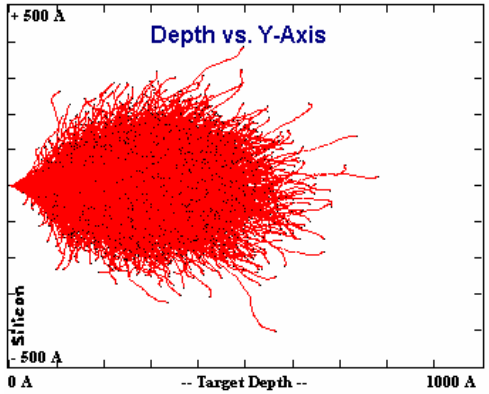


Figura GaSi40: 1000A 30KeV 10E4 0º

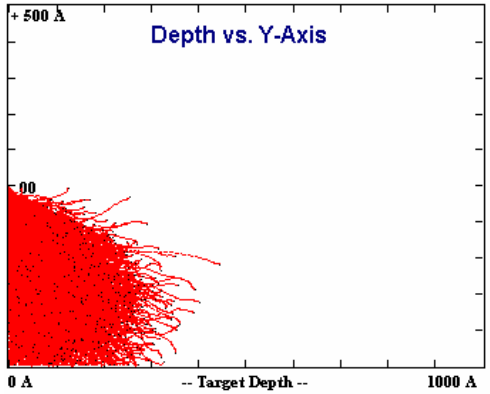


Figura GaSi41: 1000A 30KeV 10E4 80º

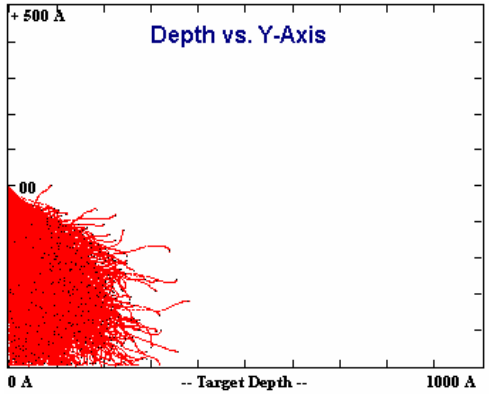


Figura GaSi42: 1000A 30KeV 10E4 88º

SRIM/TRIM: Cálculo detalhado com colisões em cascata

MATERIAL ION: Galio (Gallium) TARGET: Silício (Silicon) WIDTH: 1000A = 100nm

Energia	Número Total de Ions	Ângulo	Parâmetros de espalhamento						Perdas de energia						Resultados			
			Longitudinal		Lateral		Radial		Ionização		Vacâncias		Phonons		Sputtering Yield		Figura	Tempo
			Alcance	Desvio	Alcance	Desvio	Alcance	Desvio	Ions	Recoils	Ions	Recoils	Ions	Recoils	Atoms/Ion	eV/Atom		
10KeV	10E1	0	137A	33A	42A	54A	62A	29A	6,44	28,12	0,57	4,93	1,54	58,40	2,20	29,98	GaSi 46	0'02"
		80	52A	41A	134A	140A	139A	36A	8,32	33,03	0,61	5,36	1,64	54,05	20,70	74,86	GaSi 47	0'02"
		88	50A	29A	97A	99A	107A	15A	10,87	30,98	0,86	5,10	1,86	50,32	8,50	260,23	GaSi 48	0'01"
	10E2	0	135A	50A	38A	47A	55A	29A	6,38	29,51	0,54	4,81	1,43	57,34	1,81	20,41	GaSi 49	0'11"
		80	44A	30A	140A	150A	146A	54A	8,32	32,94	0,60	5,24	1,58	51,32	19,05	87,51	GaSi 50	0'11"
		88	37A	21A	123A	130A	129A	42A	8,14	34,04	0,62	5,37	1,53	50,29	11,74	140,15	GaSi 51	0'07"
	10E3	0	130A	49A	32A	41A	50A	29A	6,16	29,57	0,53	4,85	1,38	57,51	1,82	26,05	GaSi 52	1'42"
		80	45A	30A	128A	137A	135A	48A	7,77	33,15	0,60	5,25	1,56	51,68	18,08	99,82	GaSi 53	1'42"
		88	41A	30A	125A	135A	131A	51A	8,94	33,27	0,65	5,34	1,68	50,13	12,48	149,25	GaSi 54	1'16"
	10E4	0	130A	48A	32A	41A	50A	29A	6,18	29,59	0,52	4,86	1,39	57,46	1,72	27,79	GaSi 55	16'50"
		80	45A	30A	129A	138A	136A	49A	7,77	33,08	0,60	5,26	1,57	51,72	18,29	100,15	GaSi 56	17'20"
		88	40A	28A	129A	137A	135A	48A	8,97	33,22	0,64	5,31	1,65	50,20	12,63	144,12	GaSi 57	11'19"
	10E5	0	130A	48A	32A	41A	51A	28A	6,17	29,59	0,52	4,86	1,39	57,48	1,71	28,74	GaSi 58	176'35"
		80	44A	30A	130A	138A	136A	48A	7,78	33,09	0,60	5,26	1,57	51,71	18,42	99,60	GaSi 59	181'32"
		88	40A	27A	129A	138A	135A	48A	9,00	33,20	0,65	5,31	1,66	50,18	12,52	143,42	GaSi 60	157'19"
20KeV	10E1	0	240A	102A	72A	81A	85A	43A	8,71	29,01	0,43	4,85	1,02	55,97	0,90	20,28	GaSi 61	0'02"
		80	68A	52A	202A	211A	212A	61A	10,94	31,57	0,47	4,98	1,22	50,82	25,60	140,14	GaSi 62	0'02"
		88	28A	- A	244A	- A	244A	- A	13,56	33,29	0,48	5,14	1,37	46,17	16,00	144,51	GaSi 63	0'01"
	10E2	0	220A	82A	53A	68A	82A	47A	7,73	31,00	0,39	4,73	0,99	55,17	1,73	30,29	GaSi 64	0'19"
		80	70A	43A	228A	244A	237A	86A	9,51	33,16	0,44	4,92	1,19	50,78	24,66	133,09	GaSi 65	0'18"
		88	70A	47A	225A	246A	232A	100A	11,26	32,88	0,47	4,98	1,28	49,13	17,85	194,85	GaSi 66	0'12"
	10E3	0	211A	77A	49A	63A	76A	43A	7,27	31,67	0,37	4,73	0,97	55,00	1,87	41,14	GaSi 67	2'56"
		80	69A	46A	206A	221A	216A	79A	8,83	33,94	0,42	4,94	1,09	50,79	24,38	138,48	GaSi 68	2'54"
		88	60A	42A	211A	225A	220A	81A	10,45	33,67	0,45	4,96	1,19	49,28	18,19	185,99	GaSi 69	1'55"
	10E4	0	208A	77A	49A	63A	77A	44A	7,16	31,79	0,37	4,73	0,96	54,98	2,09	41,20	GaSi 70	30'12"
		80	69A	46A	206A	220A	215A	78A	8,89	33,99	0,42	4,93	1,09	50,69	24,23	140,26	GaSi 71	28'48"
		88	60A	42A	206A	220A	215A	79A	10,36	33,84	0,45	4,96	1,16	49,23	17,27	193,03	GaSi 72	18'00"
	10E5	0	209A	78A	49A	63A	77A	44A	7,18	31,81	0,37	4,73	0,96	54,95	2,06	40,21	GaSi 73	259'35"
		80	69A	47A	206A	221A	216A	78A	8,92	33,97	0,42	4,93	1,09	50,67	24,35	139,46	GaSi 74	257'28"
		88	60A	42A	206A	220A	215A	78A	10,37	33,90	0,45	4,96	1,16	49,17	17,24	194,14	GaSi 75	183'00"
30KeV	10E1	0	324A	144A	98A	114A	116A	57A	10,09	29,98	0,33	4,64	0,91	54,05	1,30	11,13	GaSi 76	0'03"
		80	87A	47A	352A	373A	354A	122A	13,15	32,31	0,40	4,60	1,13	48,42	24,50	172,13	GaSi 77	0'03"
		88	87A	68A	284A	294A	298A	73A	10,88	33,80	0,39	5,01	0,97	48,95	38,10	132,33	GaSi 78	0'02"
	10E2	0	286A	111A	68A	90A	110A	64A	8,29	32,67	0,31	4,63	0,81	53,31	2,52	54,57	GaSi 79	0'26"
		80	91A	59A	291A	309A	304A	106A	10,43	34,21	0,35	4,71	0,94	49,36	28,70	158,25	GaSi 80	0'23"
		88	81A	55A	278A	303A	298A	124A	11,87	33,96	0,39	4,74	1,02	48,02	24,26	210,37	GaSi 81	0'20"
	10E3	0	279A	101A	63A	81A	100A	57A	7,97	33,17	0,30	4,61	0,79	53,17	2,09	43,94	GaSi 82	4'00"
		80	87A	59A	278A	295A	289A	99A	9,79	34,70	0,34	4,73	0,88	49,56	28,45	167,68	GaSi 83	3'55"
		88	74A	53A	267A	288A	280A	108A	11,08	34,71	0,35	4,79	0,92	48,15	21,97	226,15	GaSi 84	2'28"
	10E4	0	280A	103A	65A	83A	102A	59A	7,98	33,10	0,30	4,61	0,79	53,21	2,19	45,42	GaSi 85	42'00"
		80	90A	61A	276A	295A	288A	103A	9,88	34,63	0,34	4,73	0,88	49,54	27,74	174,41	GaSi 86	39'10"
		88	80A	55A	274A	294A	286A	106A	11,38	34,37	0,36	4,75	0,94	48,20	20,17	241,05	GaSi 87	24'10"
	10E5	0	279A	104A	64A	82A	101A	58A	7,96	33,15	0,30	4,61	0,79	53,19	2,18	49,36	GaSi 88	354'20"
		80	90A	62A	276A	295A	287A	104A	9,85	34,64	0,34	4,73	0,88	49,55	27,76	174,48	GaSi 89	336'00"
		88	79A	55A	276A	295A	288A	105A	11,46	34,40	0,36	4,73	0,95	48,09	20,25	236,71	GaSi 90	200'23"

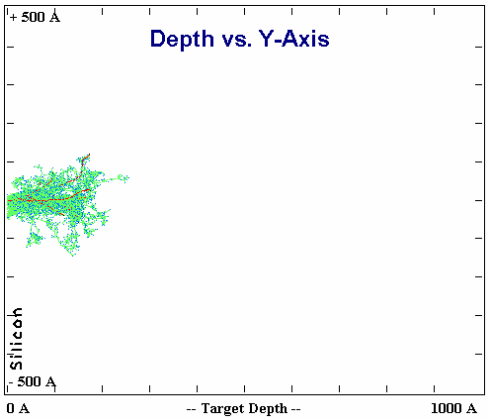


Figura GaSi46: 1000A 10KeV 10E1 0º

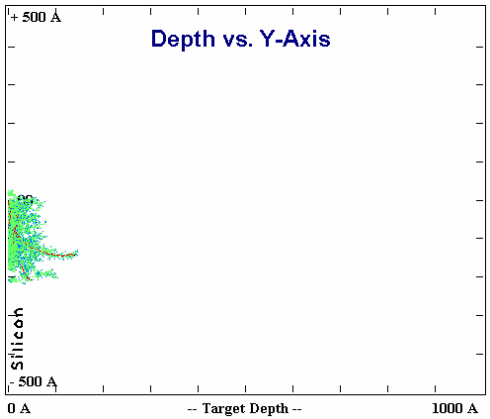


Figura GaSi 47: 1000A 10KeV 10E1 80º

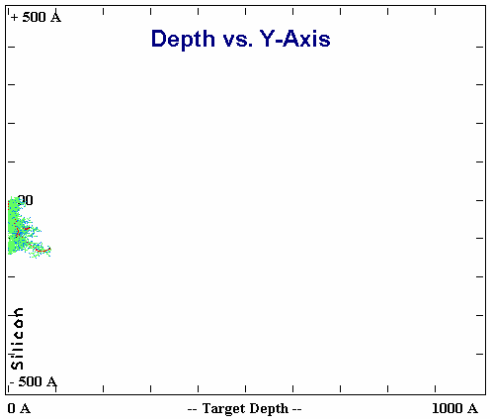


Figura GaSi48: 1000A 10KeV 10E1 88º

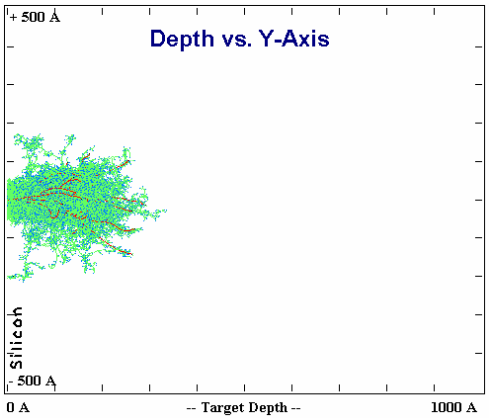


Figura GaSi49: 1000A 10KeV 10E2 0º

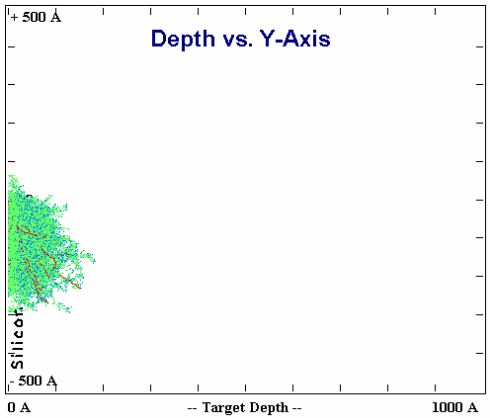


Figura GaSi50: 1000A 10KeV 10E2 80º

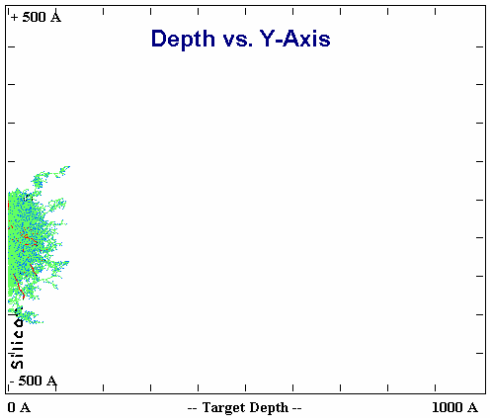


Figura GaSi51: 1000A 10KeV 10E2 88º

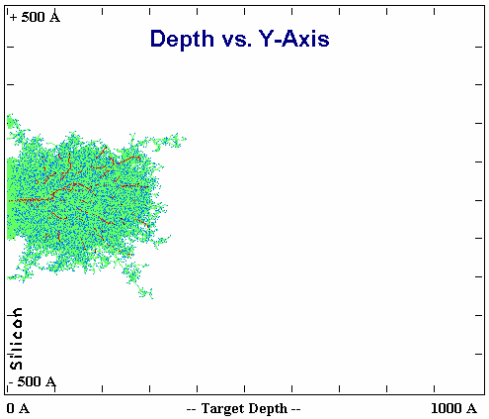


Figura GaSi52: 1000A 10KeV 10E3 0º

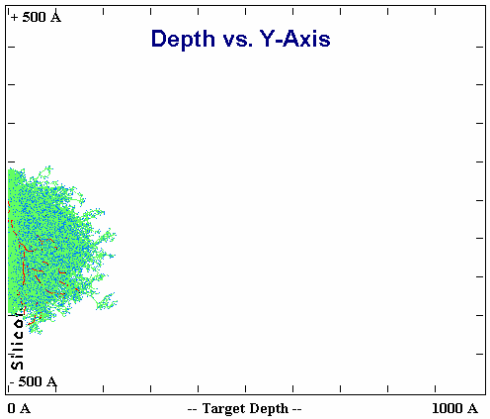


Figura GaSi53: 1000A 10KeV 10E3 80º

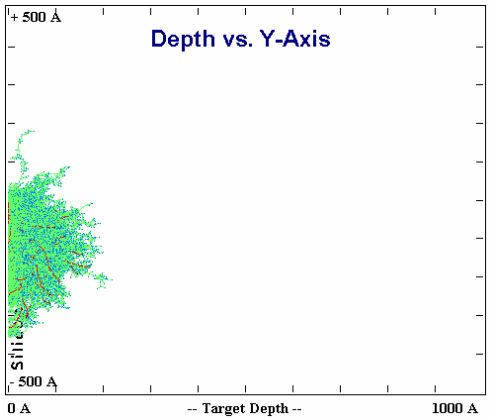


Figura GaSi54:C 1000A 10KeV 10E3 88º

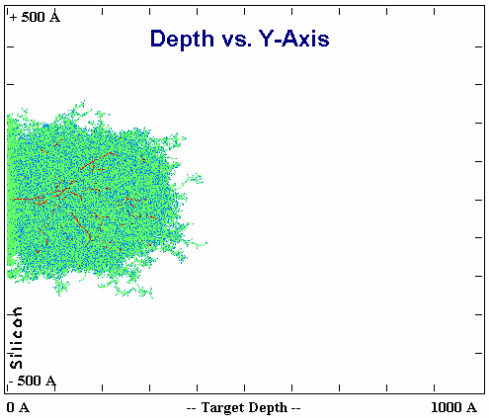


Figura GaSi55: 1000A 10KeV 10E4 0º

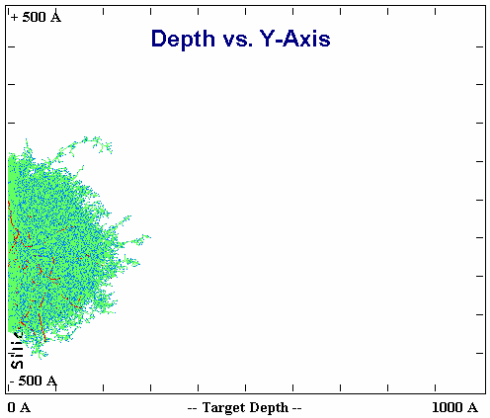


Figura GaSi56: 1000A 10KeV 10E4 80º

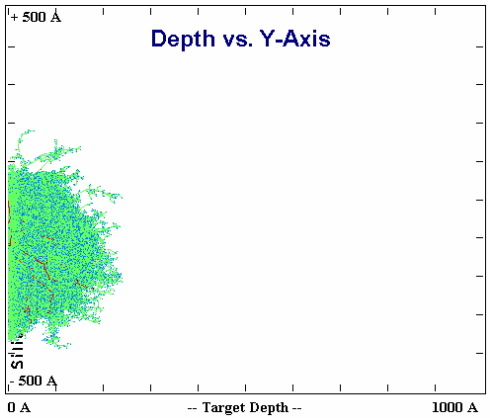


Figura GaSi57: 1000A 10KeV 10E4 88º

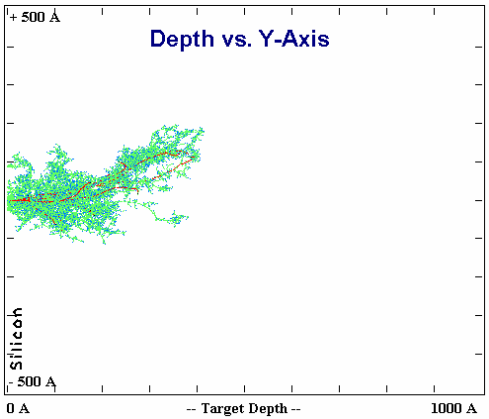


Figura GaSi61: 1000A 20KeV 10E1 0º

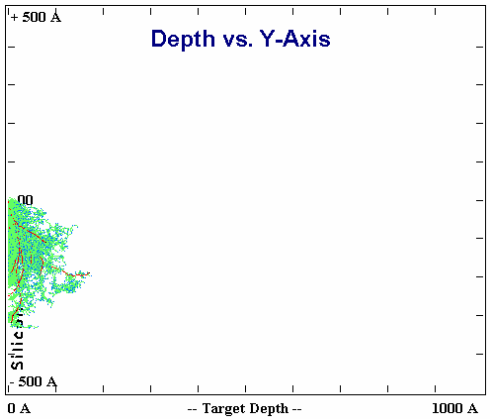


Figura GaSi62: 1000A 20KeV 10E1 80º

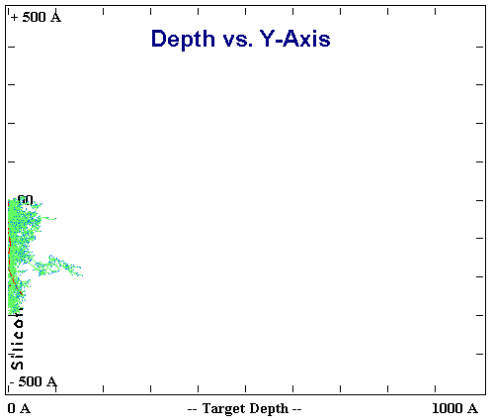


Figura GaSi63: 1000A 20KeV 10E1 88º

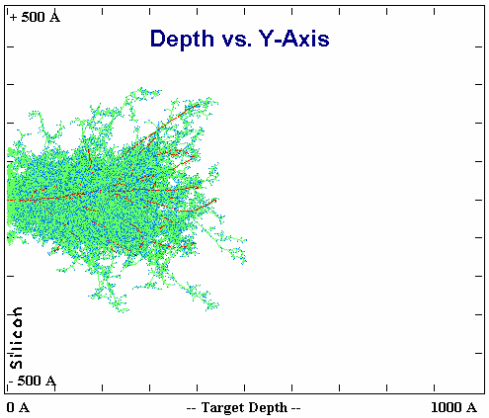


Figura GaSi64: 1000A 20KeV 10E2 0º

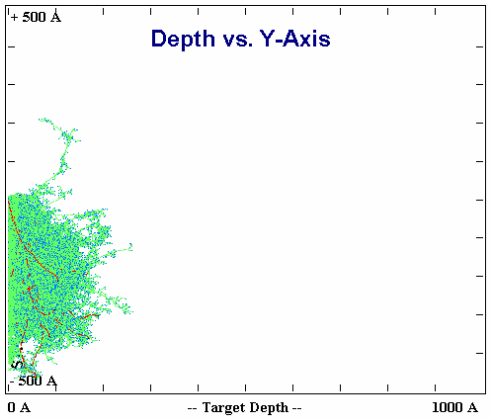


Figura GaSi65: 1000A 20KeV 10E2 80º

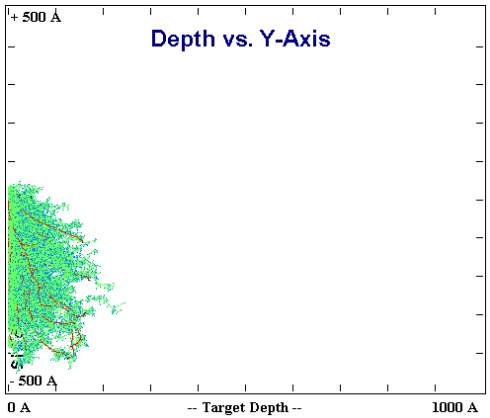


Figura GaSi66: 1000A 20KeV 10E2 88º

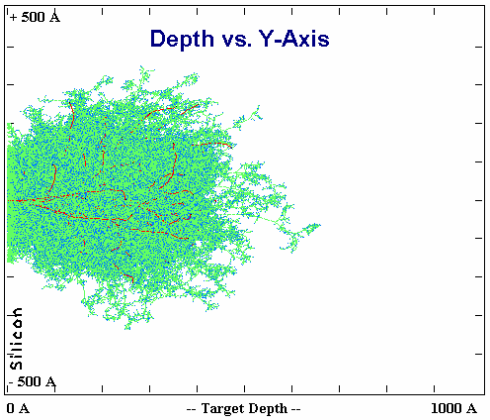


Figura GaSi67: 1000A 20KeV 10E3 0º

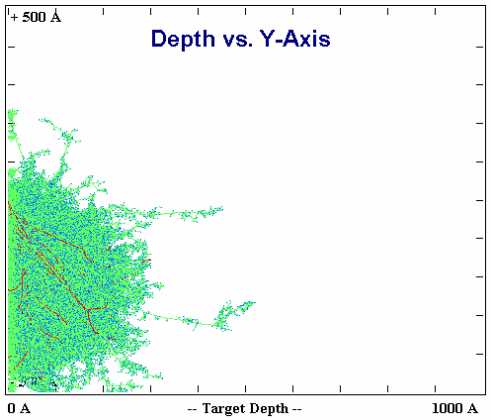


Figura GaSi68:1000A 20KeV 10E3 80º

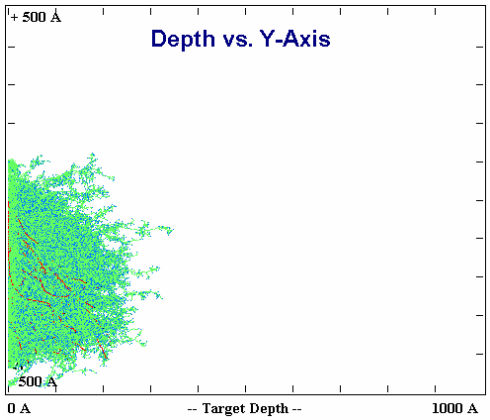


Figura GaSi69: 1000A 20KeV 10E3 88º

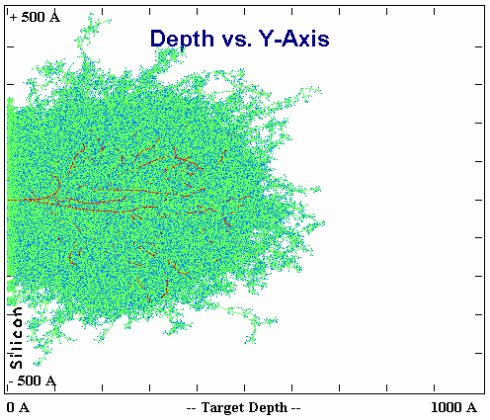


Figura GaSi70: 1000A 20KeV 10E4 0º

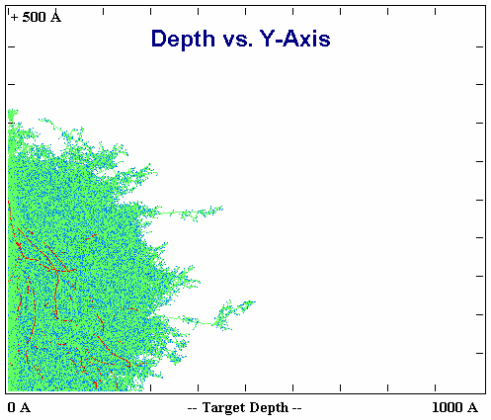


Figura GaSi71: 1000A 20KeV 10E4 80º

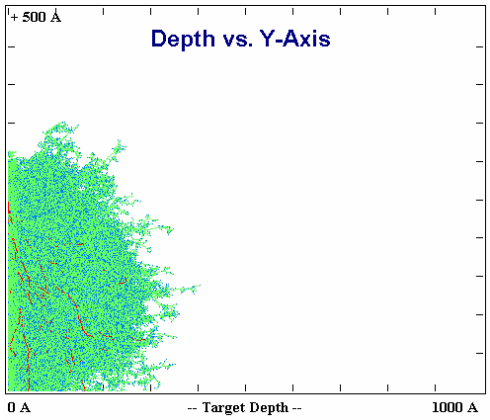


Figura GaSi72: 1000A 20KeV 10E4 88º

SRIM/TRIM: Cálculo detalhado com colisões em cascata

MATERIAL ION: Galio (Gallium) TARGET: Silício (Silicon) WIDTH: 1000A = 100nm

Energia	Número Total de Ions	Ângulo	Parâmetros de espalhamento						Perdas de energia						Resultados			
			Longitudinal		Lateral		Radial		Ionização		Vacâncias		Phonons		Sputtering Yield		Figura	Tempo
			Alcance	Desvio	Alcance	Desvio	Alcance	Desvio	Ions	Recoils	Ions	Recoils	Ions	Recoils	Atoms/Ion	eV/Atom		
10KeV	10E1	0	137A	33A	42A	54A	62A	29A	6,44	28,12	0,57	4,93	1,54	58,40	2,20	29,98	GaSi 46	0'02"
		80	52A	41A	134A	140A	139A	36A	8,32	33,03	0,61	5,36	1,64	54,05	20,70	74,86	GaSi 47	0'02"
		88	50A	29A	97A	99A	107A	15A	10,87	30,98	0,86	5,10	1,86	50,32	8,50	260,23	GaSi 48	0'01"
	10E2	0	135A	50A	38A	47A	55A	29A	6,38	29,51	0,54	4,81	1,43	57,34	1,81	20,41	GaSi 49	0'11"
		80	44A	30A	140A	150A	146A	54A	8,32	32,94	0,60	5,24	1,58	51,32	19,05	87,51	GaSi 50	0'11"
		88	37A	21A	123A	130A	129A	42A	8,14	34,04	0,62	5,37	1,53	50,29	11,74	140,15	GaSi 51	0'07"
	10E3	0	130A	49A	32A	41A	50A	29A	6,16	29,57	0,53	4,85	1,38	57,51	1,82	26,05	GaSi 52	1'42"
		80	45A	30A	128A	137A	135A	48A	7,77	33,15	0,60	5,25	1,56	51,68	18,08	99,82	GaSi 53	1'42"
		88	41A	30A	125A	135A	131A	51A	8,94	33,27	0,65	5,34	1,68	50,13	12,48	149,25	GaSi 54	1'16"
	10E4	0	130A	48A	32A	41A	50A	29A	6,18	29,59	0,52	4,86	1,39	57,46	1,72	27,79	GaSi 55	16'50"
		80	45A	30A	129A	138A	136A	49A	7,77	33,08	0,60	5,26	1,57	51,72	18,29	100,15	GaSi 56	17'20"
		88	40A	28A	129A	137A	135A	48A	8,97	33,22	0,64	5,31	1,65	50,20	12,63	144,12	GaSi 57	11'19"
	10E5	0	130A	48A	32A	41A	51A	28A	6,17	29,59	0,52	4,86	1,39	57,48	1,71	28,74	GaSi 58	176'35"
		80	44A	30A	130A	138A	136A	48A	7,78	33,09	0,60	5,26	1,57	51,71	18,42	99,60	GaSi 59	181'32"
		88	40A	27A	129A	138A	135A	48A	9,00	33,20	0,65	5,31	1,66	50,18	12,52	143,42	GaSi 60	157'19"
20KeV	10E1	0	240A	102A	72A	81A	85A	43A	8,71	29,01	0,43	4,85	1,02	55,97	0,90	20,28	GaSi 61	0'02"
		80	68A	52A	202A	211A	212A	61A	10,94	31,57	0,47	4,98	1,22	50,82	25,60	140,14	GaSi 62	0'02"
		88	28A	- A	244A	- A	244A	- A	13,56	33,29	0,48	5,14	1,37	46,17	16,00	144,51	GaSi 63	0'01"
	10E2	0	220A	82A	53A	68A	82A	47A	7,73	31,00	0,39	4,73	0,99	55,17	1,73	30,29	GaSi 64	0'19"
		80	70A	43A	228A	244A	237A	86A	9,51	33,16	0,44	4,92	1,19	50,78	24,66	133,09	GaSi 65	0'18"
		88	70A	47A	225A	246A	232A	100A	11,26	32,88	0,47	4,98	1,28	49,13	17,85	194,85	GaSi 66	0'12"
	10E3	0	211A	77A	49A	63A	76A	43A	7,27	31,67	0,37	4,73	0,97	55,00	1,87	41,14	GaSi 67	2'56"
		80	69A	46A	206A	221A	216A	79A	8,83	33,94	0,42	4,94	1,09	50,79	24,38	138,48	GaSi 68	2'54"
		88	60A	42A	211A	225A	220A	81A	10,45	33,67	0,45	4,96	1,19	49,28	18,19	185,99	GaSi 69	1'55"
	10E4	0	208A	77A	49A	63A	77A	44A	7,16	31,79	0,37	4,73	0,96	54,98	2,09	41,20	GaSi 70	30'12"
		80	69A	46A	206A	220A	215A	78A	8,89	33,99	0,42	4,93	1,09	50,69	24,23	140,26	GaSi 71	28'48"
		88	60A	42A	206A	220A	215A	79A	10,36	33,84	0,45	4,96	1,16	49,23	17,27	193,03	GaSi 72	18'00"
	10E5	0	209A	78A	49A	63A	77A	44A	7,18	31,81	0,37	4,73	0,96	54,95	2,06	40,21	GaSi 73	259'35"
		80	69A	47A	206A	221A	216A	78A	8,92	33,97	0,42	4,93	1,09	50,67	24,35	139,46	GaSi 74	257'28"
		88	60A	42A	206A	220A	215A	78A	10,37	33,90	0,45	4,96	1,16	49,17	17,24	194,14	GaSi 75	183'00"
30KeV	10E1	0	324A	144A	98A	114A	116A	57A	10,09	29,98	0,33	4,64	0,91	54,05	1,30	11,13	GaSi 76	0'03"
		80	87A	47A	352A	373A	354A	122A	13,15	32,31	0,40	4,60	1,13	48,42	24,50	172,13	GaSi 77	0'03"
		88	87A	68A	284A	294A	298A	73A	10,88	33,80	0,39	5,01	0,97	48,95	38,10	132,33	GaSi 78	0'02"
	10E2	0	286A	111A	68A	90A	110A	64A	8,29	32,67	0,31	4,63	0,81	53,31	2,52	54,57	GaSi 79	0'26"
		80	91A	59A	291A	309A	304A	106A	10,43	34,21	0,35	4,71	0,94	49,36	28,70	158,25	GaSi 80	0'23"
		88	81A	55A	278A	303A	298A	124A	11,87	33,96	0,39	4,74	1,02	48,02	24,26	210,37	GaSi 81	0'20"
	10E3	0	279A	101A	63A	81A	100A	57A	7,97	33,17	0,30	4,61	0,79	53,17	2,09	43,94	GaSi 82	4'00"
		80	87A	59A	278A	295A	289A	99A	9,79	34,70	0,34	4,73	0,88	49,56	28,45	167,68	GaSi 83	3'55"
		88	74A	53A	267A	288A	280A	108A	11,08	34,71	0,35	4,79	0,92	48,15	21,97	226,15	GaSi 84	2'28"
	10E4	0	280A	103A	65A	83A	102A	59A	7,98	33,10	0,30	4,61	0,79	53,21	2,19	45,42	GaSi 85	42'00"
		80	90A	61A	276A	295A	288A	103A	9,88	34,63	0,34	4,73	0,88	49,54	27,74	174,41	GaSi 86	39'10"
		88	80A	55A	274A	294A	286A	106A	11,38	34,37	0,36	4,75	0,94	48,20	20,17	241,05	GaSi 87	24'10"
	10E5	0	279A	104A	64A	82A	101A	58A	7,96	33,15	0,30	4,61	0,79	53,19	2,18	49,36	GaSi 88	354'20"
		80	90A	62A	276A	295A	287A	104A	9,85	34,64	0,34	4,73	0,88	49,55	27,76	174,48	GaSi 89	336'00"
		88	79A	55A	276A	295A	288A	105A	11,46	34,40	0,36	4,73	0,95	48,09	20,25	236,71	GaSi 90	200'23"

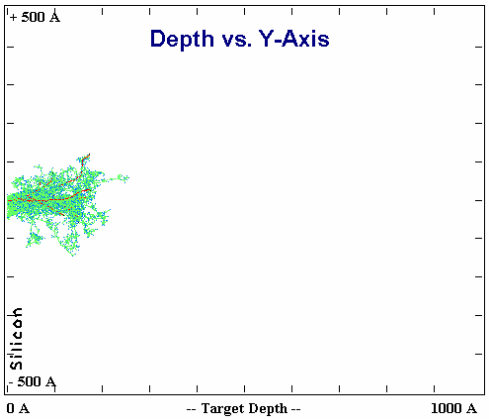


Figura GaSi46: 1000A 10KeV 10E1 0º

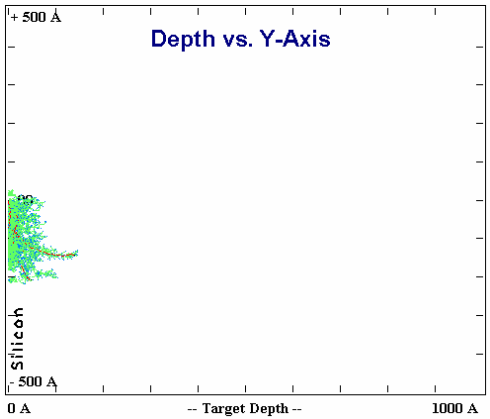


Figura GaSi 47: 1000A 10KeV 10E1 80º

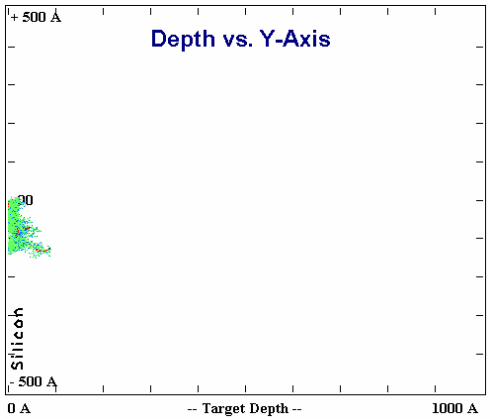


Figura GaSi48: 1000A 10KeV 10E1 88º

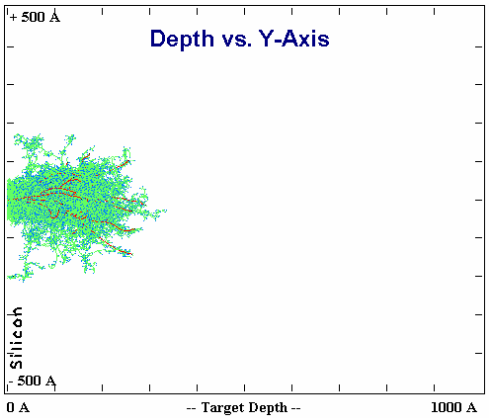


Figura GaSi49: 1000A 10KeV 10E2 0º

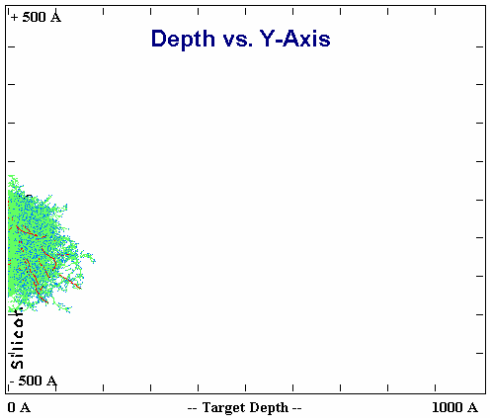


Figura GaSi50: 1000A 10KeV 10E2 80º

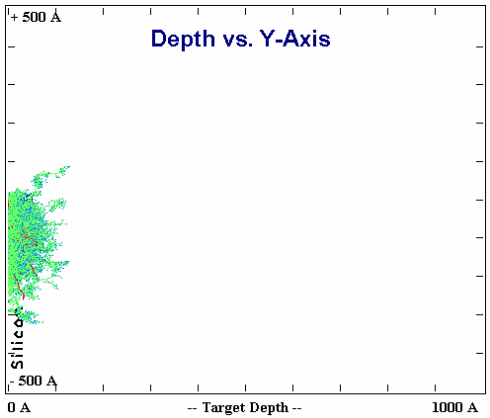


Figura GaSi51: 1000A 10KeV 10E2 88º

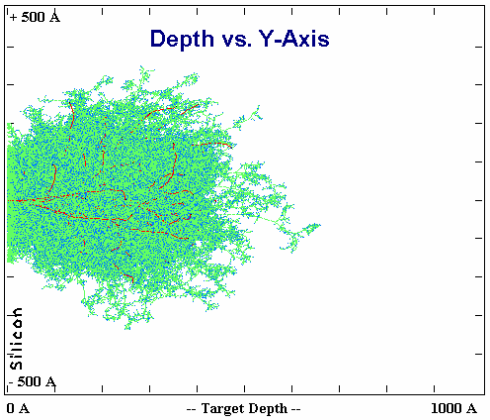


Figura GaSi67: 1000A 20KeV 10E3 0º

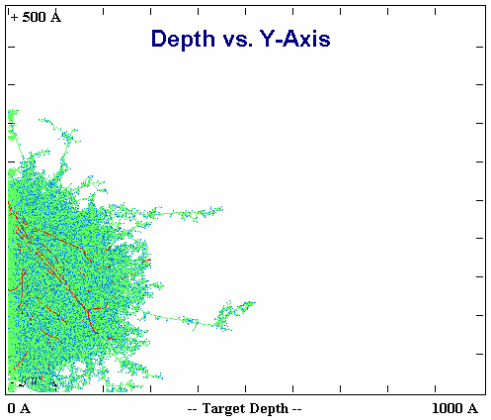


Figura GaSi68:1000A 20KeV 10E3 80º

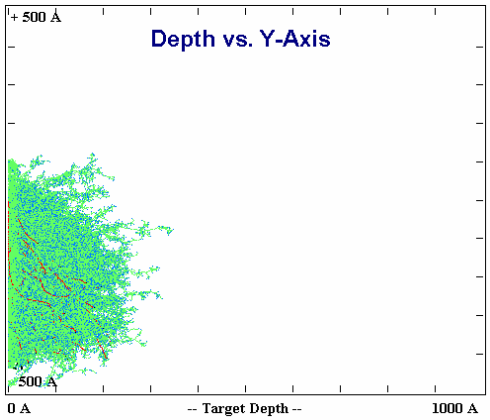


Figura GaSi69: 1000A 20KeV 10E3 88º

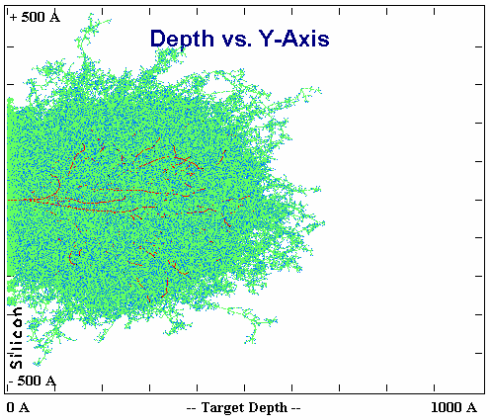


Figura GaSi70: 1000A 20KeV 10E4 0º

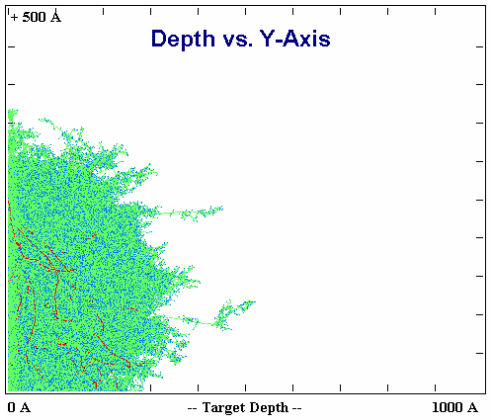


Figura GaSi71: 1000A 20KeV 10E4 80º

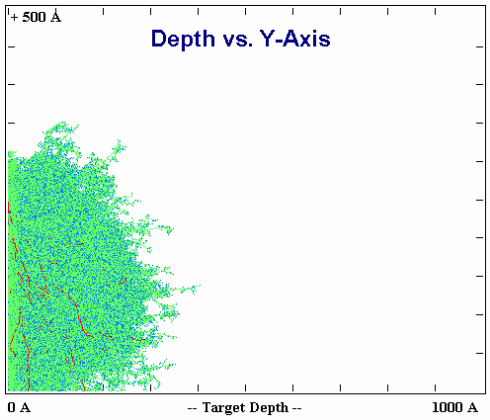


Figura GaSi72: 1000A 20KeV 10E4 88º

SRIM/TRIM: Cálculo detalhado com colisões em cascata

MATERIAL ION: Galio (Gallium) TARGET: Silício (Silicon) WIDTH: 1000A = 100nm

Energia	Número Total de Ions	Ângulo	Parâmetros de espalhamento						Perdas de energia						Resultados			
			Longitudinal		Lateral		Radial		Ionização		Vacâncias		Phonons		Sputtering Yield		Figura	Tempo
			Alcance	Desvio	Alcance	Desvio	Alcance	Desvio	Ions	Recoils	Ions	Recoils	Ions	Recoils	Atoms/Ion	eV/Atom		
10KeV	10E1	0	137A	33A	42A	54A	62A	29A	6,44	28,12	0,57	4,93	1,54	58,40	2,20	29,98	GaSi 46	0'02"
		80	52A	41A	134A	140A	139A	36A	8,32	33,03	0,61	5,36	1,64	54,05	20,70	74,86	GaSi 47	0'02"
		88	50A	29A	97A	99A	107A	15A	10,87	30,98	0,86	5,10	1,86	50,32	8,50	260,23	GaSi 48	0'01"
	10E2	0	135A	50A	38A	47A	55A	29A	6,38	29,51	0,54	4,81	1,43	57,34	1,81	20,41	GaSi 49	0'11"
		80	44A	30A	140A	150A	146A	54A	8,32	32,94	0,60	5,24	1,58	51,32	19,05	87,51	GaSi 50	0'11"
		88	37A	21A	123A	130A	129A	42A	8,14	34,04	0,62	5,37	1,53	50,29	11,74	140,15	GaSi 51	0'07"
	10E3	0	130A	49A	32A	41A	50A	29A	6,16	29,57	0,53	4,85	1,38	57,51	1,82	26,05	GaSi 52	1'42"
		80	45A	30A	128A	137A	135A	48A	7,77	33,15	0,60	5,25	1,56	51,68	18,08	99,82	GaSi 53	1'42"
		88	41A	30A	125A	135A	131A	51A	8,94	33,27	0,65	5,34	1,68	50,13	12,48	149,25	GaSi 54	1'16"
	10E4	0	130A	48A	32A	41A	50A	29A	6,18	29,59	0,52	4,86	1,39	57,46	1,72	27,79	GaSi 55	16'50"
		80	45A	30A	129A	138A	136A	49A	7,77	33,08	0,60	5,26	1,57	51,72	18,29	100,15	GaSi 56	17'20"
		88	40A	28A	129A	137A	135A	48A	8,97	33,22	0,64	5,31	1,65	50,20	12,63	144,12	GaSi 57	11'19"
	10E5	0	130A	48A	32A	41A	51A	28A	6,17	29,59	0,52	4,86	1,39	57,48	1,71	28,74	GaSi 58	176'35"
		80	44A	30A	130A	138A	136A	48A	7,78	33,09	0,60	5,26	1,57	51,71	18,42	99,60	GaSi 59	181'32"
		88	40A	27A	129A	138A	135A	48A	9,00	33,20	0,65	5,31	1,66	50,18	12,52	143,42	GaSi 60	157'19"
20KeV	10E1	0	240A	102A	72A	81A	85A	43A	8,71	29,01	0,43	4,85	1,02	55,97	0,90	20,28	GaSi 61	0'02"
		80	68A	52A	202A	211A	212A	61A	10,94	31,57	0,47	4,98	1,22	50,82	25,60	140,14	GaSi 62	0'02"
		88	28A	- A	244A	- A	244A	- A	13,56	33,29	0,48	5,14	1,37	46,17	16,00	144,51	GaSi 63	0'01"
	10E2	0	220A	82A	53A	68A	82A	47A	7,73	31,00	0,39	4,73	0,99	55,17	1,73	30,29	GaSi 64	0'19"
		80	70A	43A	228A	244A	237A	86A	9,51	33,16	0,44	4,92	1,19	50,78	24,66	133,09	GaSi 65	0'18"
		88	70A	47A	225A	246A	232A	100A	11,26	32,88	0,47	4,98	1,28	49,13	17,85	194,85	GaSi 66	0'12"
	10E3	0	211A	77A	49A	63A	76A	43A	7,27	31,67	0,37	4,73	0,97	55,00	1,87	41,14	GaSi 67	2'56"
		80	69A	46A	206A	221A	216A	79A	8,83	33,94	0,42	4,94	1,09	50,79	24,38	138,48	GaSi 68	2'54"
		88	60A	42A	211A	225A	220A	81A	10,45	33,67	0,45	4,96	1,19	49,28	18,19	185,99	GaSi 69	1'55"
	10E4	0	208A	77A	49A	63A	77A	44A	7,16	31,79	0,37	4,73	0,96	54,98	2,09	41,20	GaSi 70	30'12"
		80	69A	46A	206A	220A	215A	78A	8,89	33,99	0,42	4,93	1,09	50,69	24,23	140,26	GaSi 71	28'48"
		88	60A	42A	206A	220A	215A	79A	10,36	33,84	0,45	4,96	1,16	49,23	17,27	193,03	GaSi 72	18'00"
	10E5	0	209A	78A	49A	63A	77A	44A	7,18	31,81	0,37	4,73	0,96	54,95	2,06	40,21	GaSi 73	259'35"
		80	69A	47A	206A	221A	216A	78A	8,92	33,97	0,42	4,93	1,09	50,67	24,35	139,46	GaSi 74	257'28"
		88	60A	42A	206A	220A	215A	78A	10,37	33,90	0,45	4,96	1,16	49,17	17,24	194,14	GaSi 75	183'00"
30KeV	10E1	0	324A	144A	98A	114A	116A	57A	10,09	29,98	0,33	4,64	0,91	54,05	1,30	11,13	GaSi 76	0'03"
		80	87A	47A	352A	373A	354A	122A	13,15	32,31	0,40	4,60	1,13	48,42	24,50	172,13	GaSi 77	0'03"
		88	87A	68A	284A	294A	298A	73A	10,88	33,80	0,39	5,01	0,97	48,95	38,10	132,33	GaSi 78	0'02"
	10E2	0	286A	111A	68A	90A	110A	64A	8,29	32,67	0,31	4,63	0,81	53,31	2,52	54,57	GaSi 79	0'26"
		80	91A	59A	291A	309A	304A	106A	10,43	34,21	0,35	4,71	0,94	49,36	28,70	158,25	GaSi 80	0'23"
		88	81A	55A	278A	303A	298A	124A	11,87	33,96	0,39	4,74	1,02	48,02	24,26	210,37	GaSi 81	0'20"
	10E3	0	279A	101A	63A	81A	100A	57A	7,97	33,17	0,30	4,61	0,79	53,17	2,09	43,94	GaSi 82	4'00"
		80	87A	59A	278A	295A	289A	99A	9,79	34,70	0,34	4,73	0,88	49,56	28,45	167,68	GaSi 83	3'55"
		88	74A	53A	267A	288A	280A	108A	11,08	34,71	0,35	4,79	0,92	48,15	21,97	226,15	GaSi 84	2'28"
	10E4	0	280A	103A	65A	83A	102A	59A	7,98	33,10	0,30	4,61	0,79	53,21	2,19	45,42	GaSi 85	42'00"
		80	90A	61A	276A	295A	288A	103A	9,88	34,63	0,34	4,73	0,88	49,54	27,74	174,41	GaSi 86	39'10"
		88	80A	55A	274A	294A	286A	106A	11,38	34,37	0,36	4,75	0,94	48,20	20,17	241,05	GaSi 87	24'10"
	10E5	0	279A	104A	64A	82A	101A	58A	7,96	33,15	0,30	4,61	0,79	53,19	2,18	49,36	GaSi 88	354'20"
		80	90A	62A	276A	295A	287A	104A	9,85	34,64	0,34	4,73	0,88	49,55	27,76	174,48	GaSi 89	336'00"
		88	79A	55A	276A	295A	288A	105A	11,46	34,40	0,36	4,73	0,95	48,09	20,25	236,71	GaSi 90	200'23"

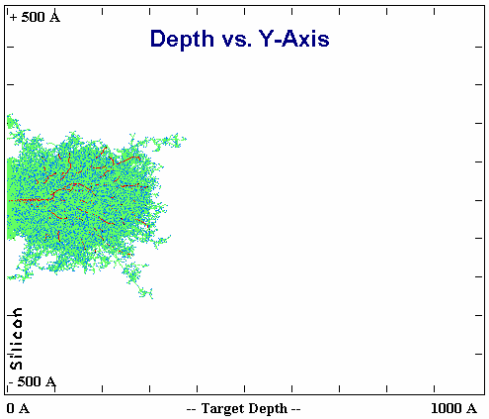


Figura GaSi52: 1000A 10KeV 10E3 0º

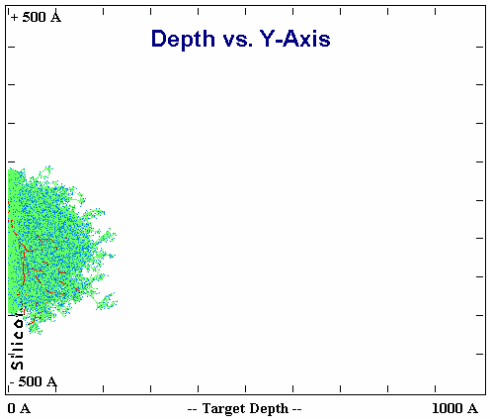


Figura GaSi53: 1000A 10KeV 10E3 80º

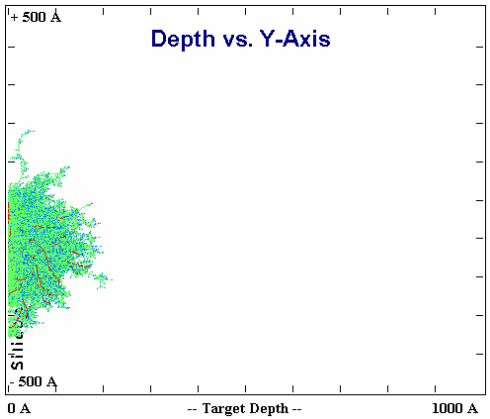


Figura GaSi54:C 1000A 10KeV 10E3 88º

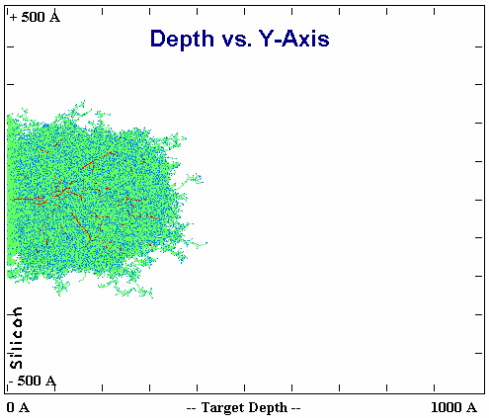


Figura GaSi55: 1000A 10KeV 10E4 0º

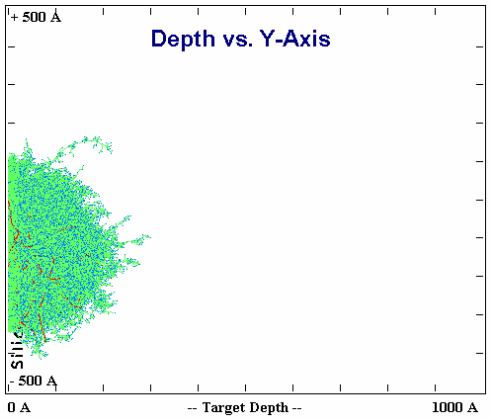


Figura GaSi56: 1000A 10KeV 10E4 80º

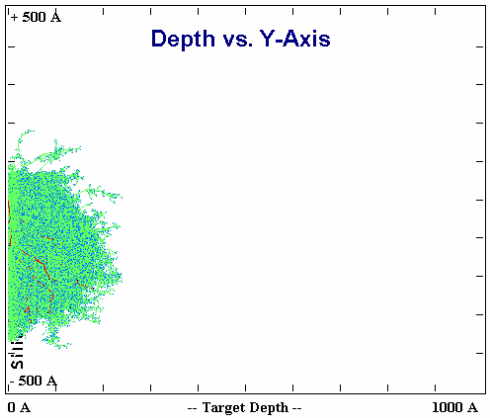


Figura GaSi57: 1000A 10KeV 10E4 88º

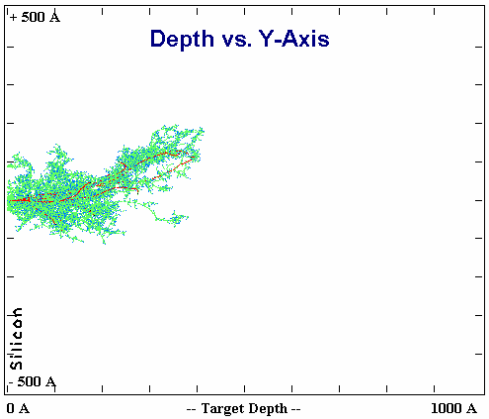


Figura GaSi61: 1000A 20KeV 10E1 0º

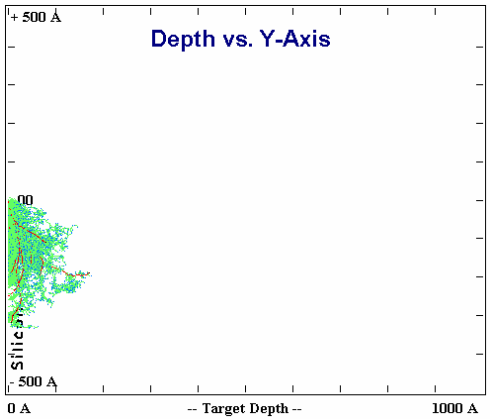


Figura GaSi62: 1000A 20KeV 10E1 80º

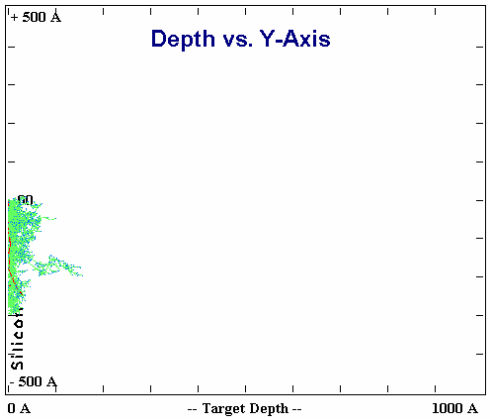


Figura GaSi63: 1000A 20KeV 10E1 88º

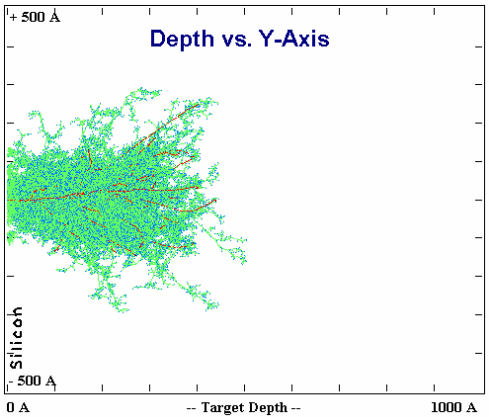


Figura GaSi64: 1000A 20KeV 10E2 0º

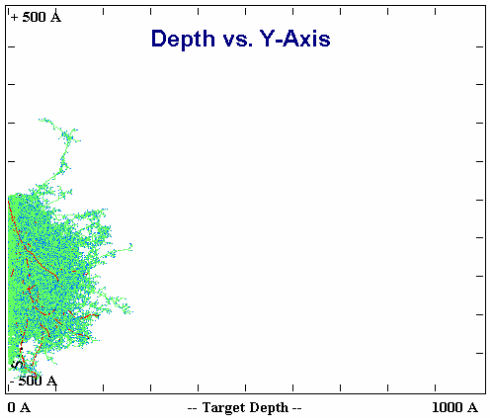


Figura GaSi65: 1000A 20KeV 10E2 80º

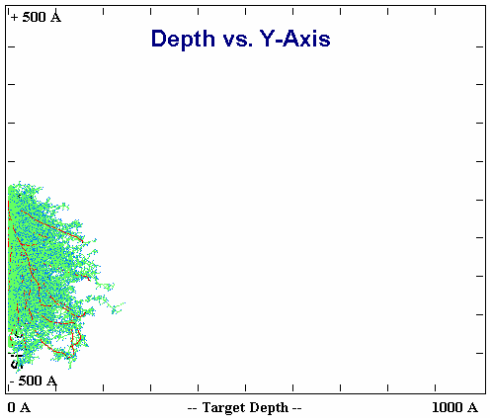


Figura GaSi66: 1000A 20KeV 10E2 88º

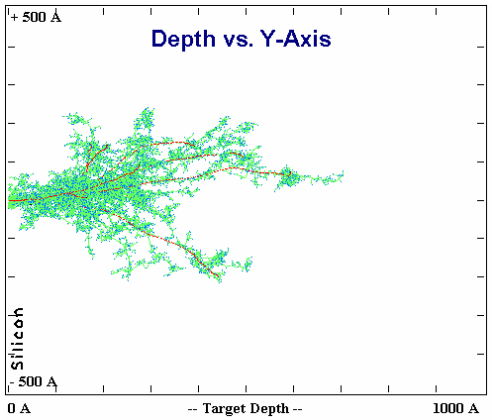


Figura GaSi76: 1000A 30KeV 10E1 0º

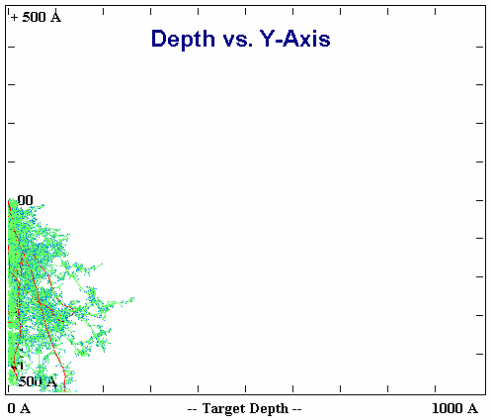


Figura GaSi77: 1000A 30KeV 10E1 80º

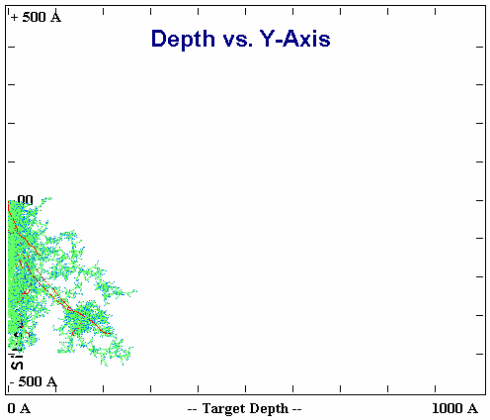


Figura GaSi78: 1000A 30KeV 10E1 88º

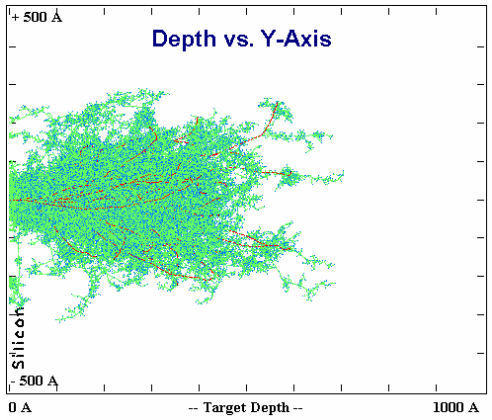


Figura GaSi79: 1000A 30KeV 10E2 0º

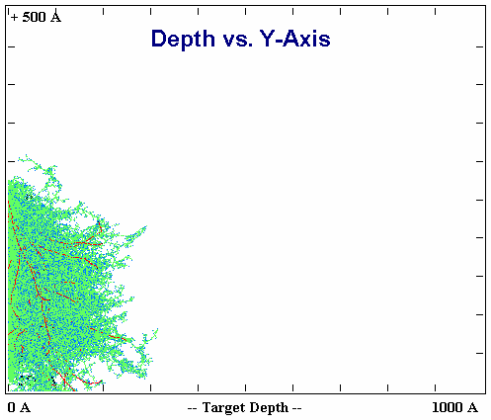


Figura GaSi80: 1000A 30KeV 10E2 80º

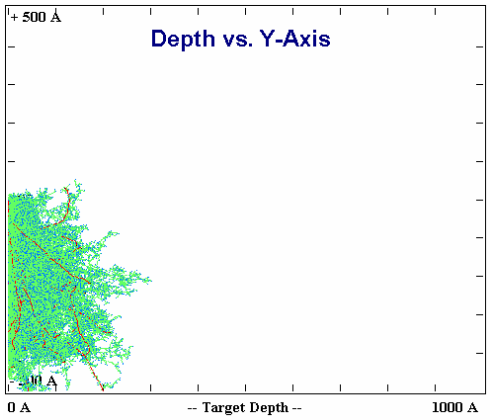


Figura GaSi81: 1000A 30KeV 10E2 88º

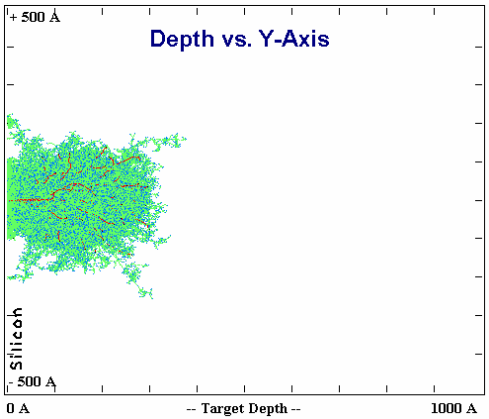


Figura GaSi52: 1000A 10KeV 10E3 0º

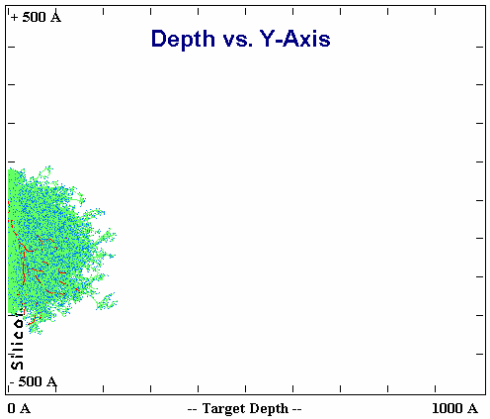


Figura GaSi53: 1000A 10KeV 10E3 80º

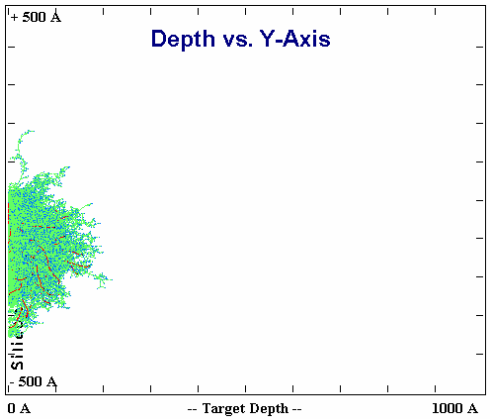


Figura GaSi54:C 1000A 10KeV 10E3 88º

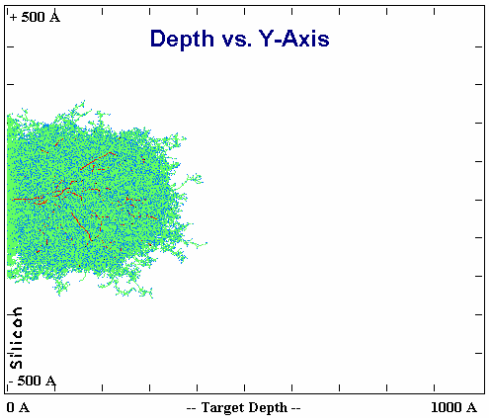


Figura GaSi55: 1000A 10KeV 10E4 0º

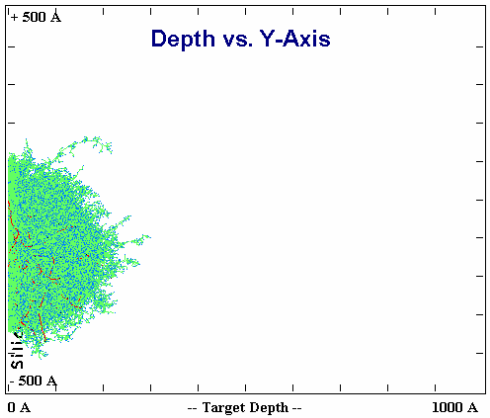


Figura GaSi56: 1000A 10KeV 10E4 80º

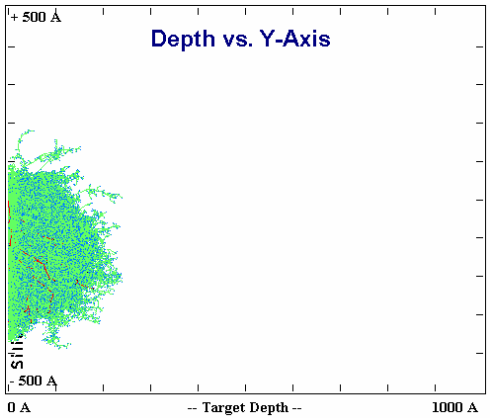


Figura GaSi57: 1000A 10KeV 10E4 88º

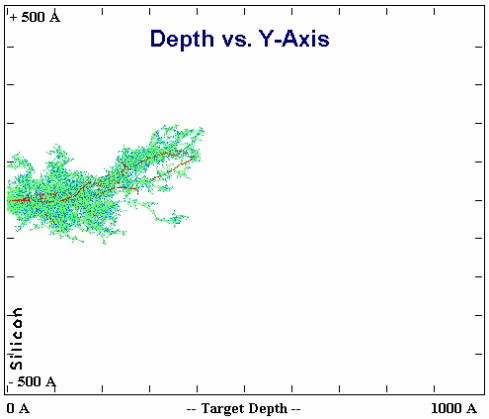


Figura GaSi61: 1000A 20KeV 10E1 0º

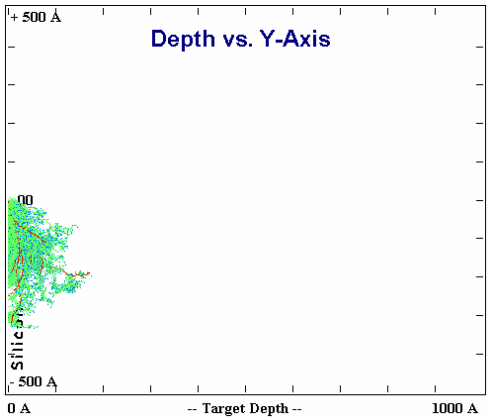


Figura GaSi62: 1000A 20KeV 10E1 80º

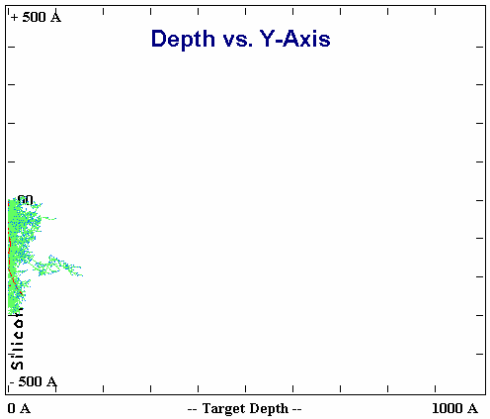


Figura GaSi63: 1000A 20KeV 10E1 88º

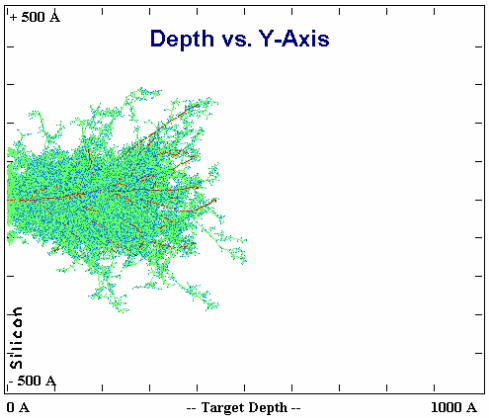


Figura GaSi64: 1000A 20KeV 10E2 0º

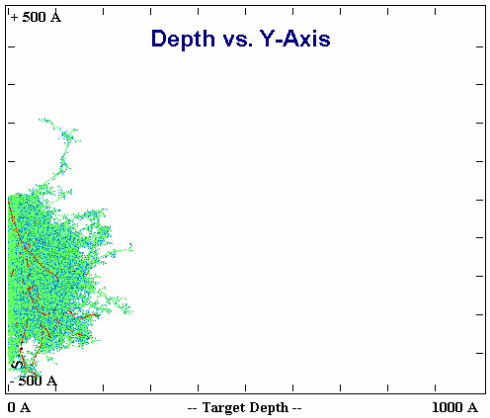


Figura GaSi65: 1000A 20KeV 10E2 80º

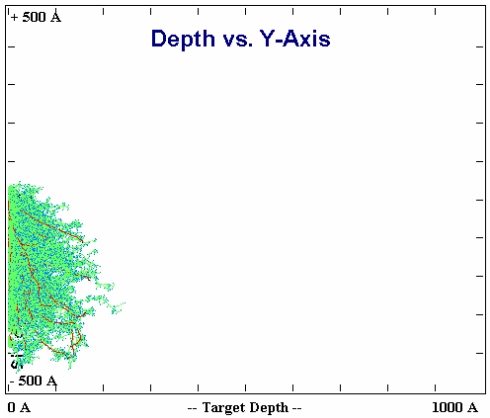


Figura GaSi66: 1000A 20KeV 10E2 88º

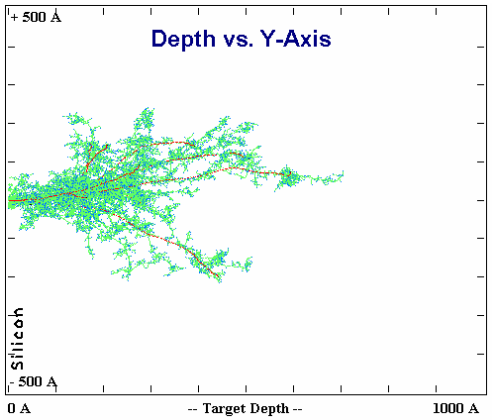


Figura GaSi76: 1000A 30KeV 10E1 0º

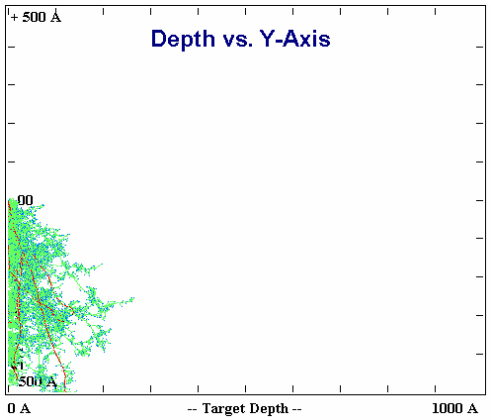


Figura GaSi77: 1000A 30KeV 10E1 80º

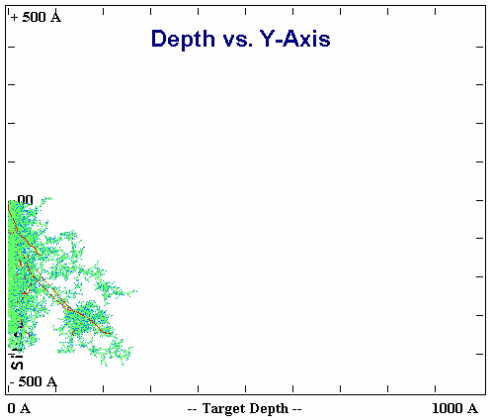


Figura GaSi78: 1000A 30KeV 10E1 88º

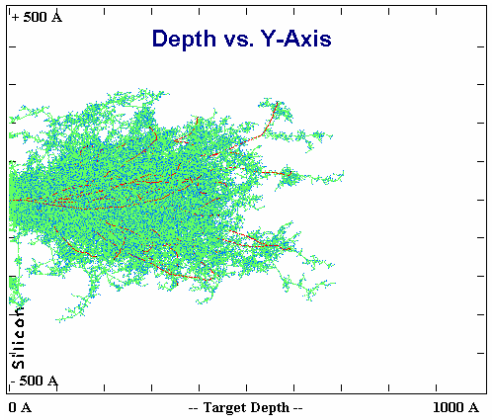


Figura GaSi79: 1000A 30KeV 10E2 0º

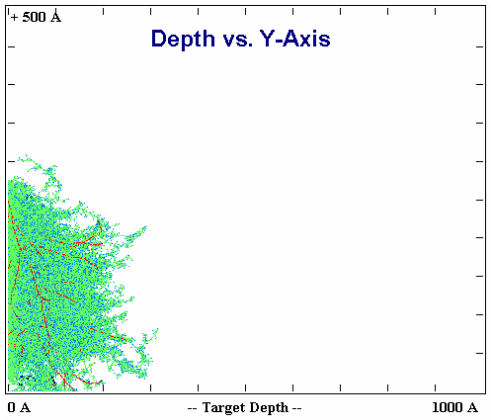


Figura GaSi80: 1000A 30KeV 10E2 80º

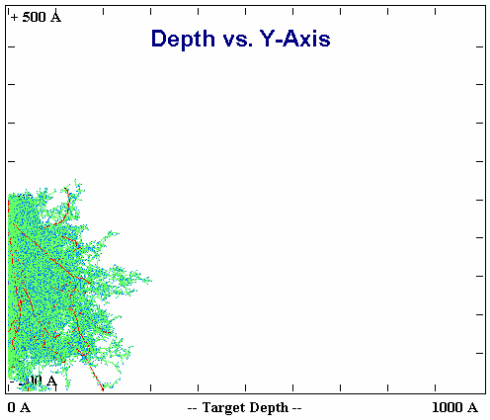


Figura GaSi81: 1000A 30KeV 10E2 88º

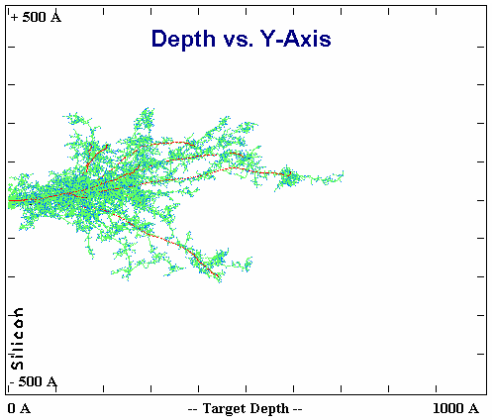


Figura GaSi76: 1000A 30KeV 10E1 0º

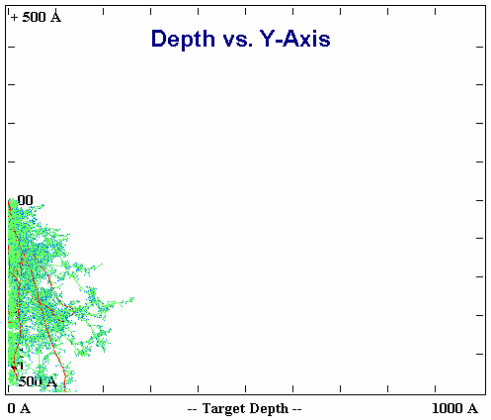


Figura GaSi77: 1000A 30KeV 10E1 80º

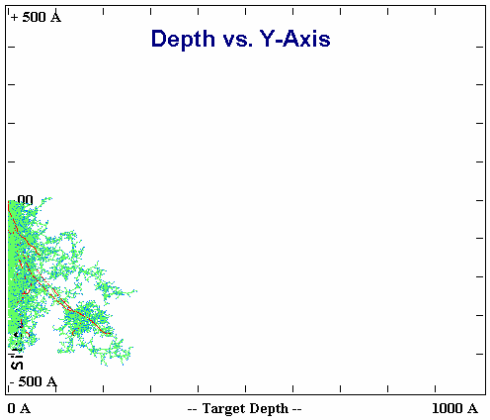


Figura GaSi78: 1000A 30KeV 10E1 88º

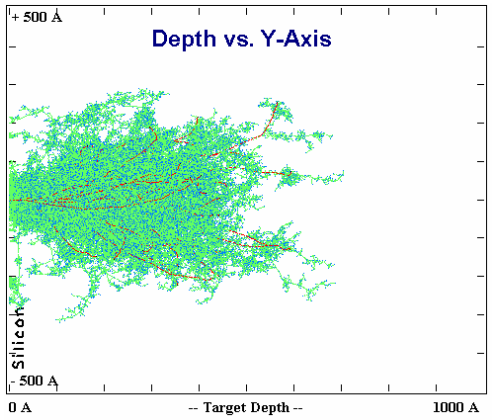


Figura GaSi79: 1000A 30KeV 10E2 0º

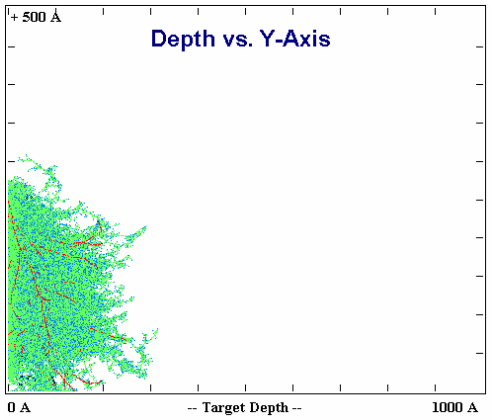


Figura GaSi80: 1000A 30KeV 10E2 80º

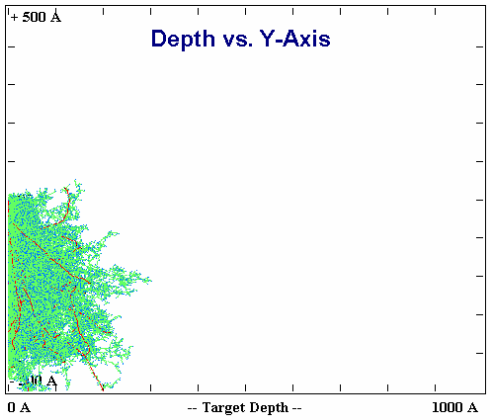


Figura GaSi81: 1000A 30KeV 10E2 88º

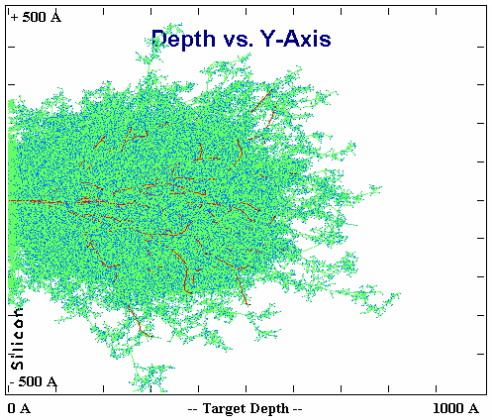


Figura GaSi82: 1000A 30KeV 10E3 0º

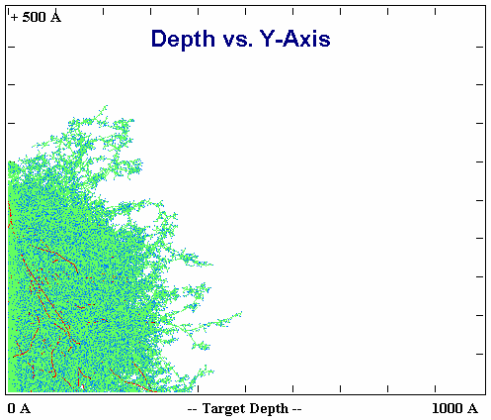


Figura GaSi83: 1000A 30KeV 10E3 80º

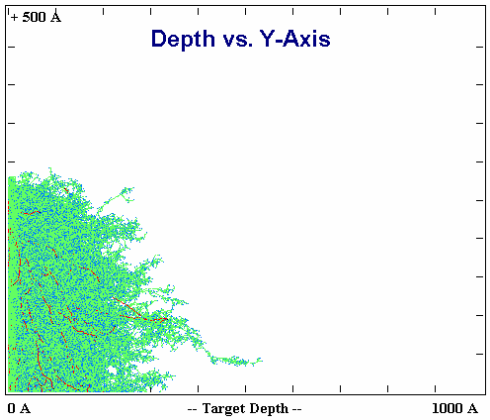


Figura GaSi84: 1000A 30KeV 10E3 88º

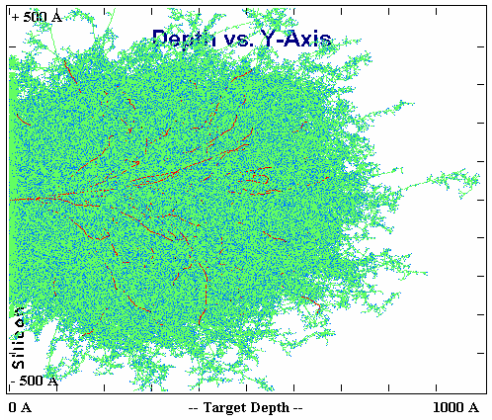


Figura GaSi85: 1000A 30KeV 10E4 0º

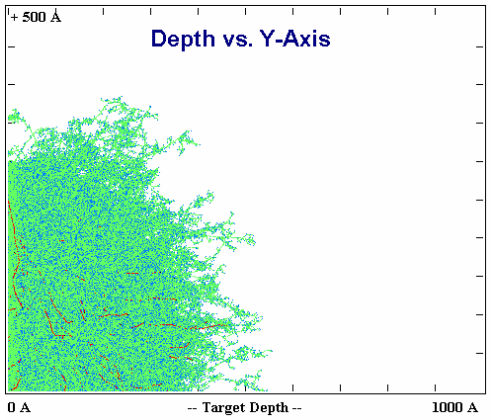


Figura GaSi86: 1000A 30KeV 10E4 80º

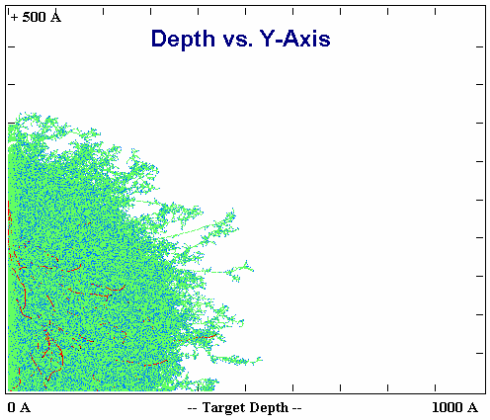
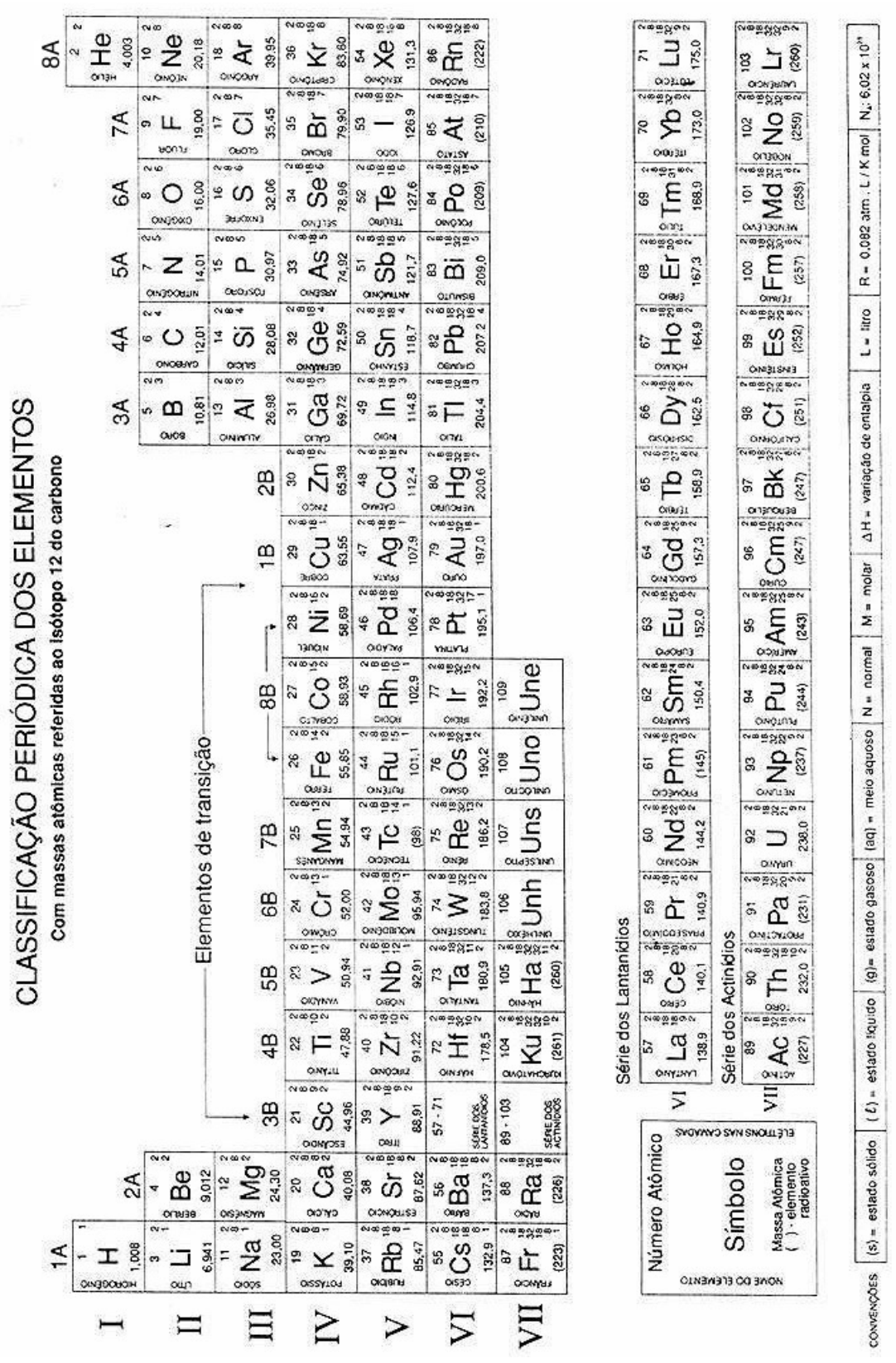


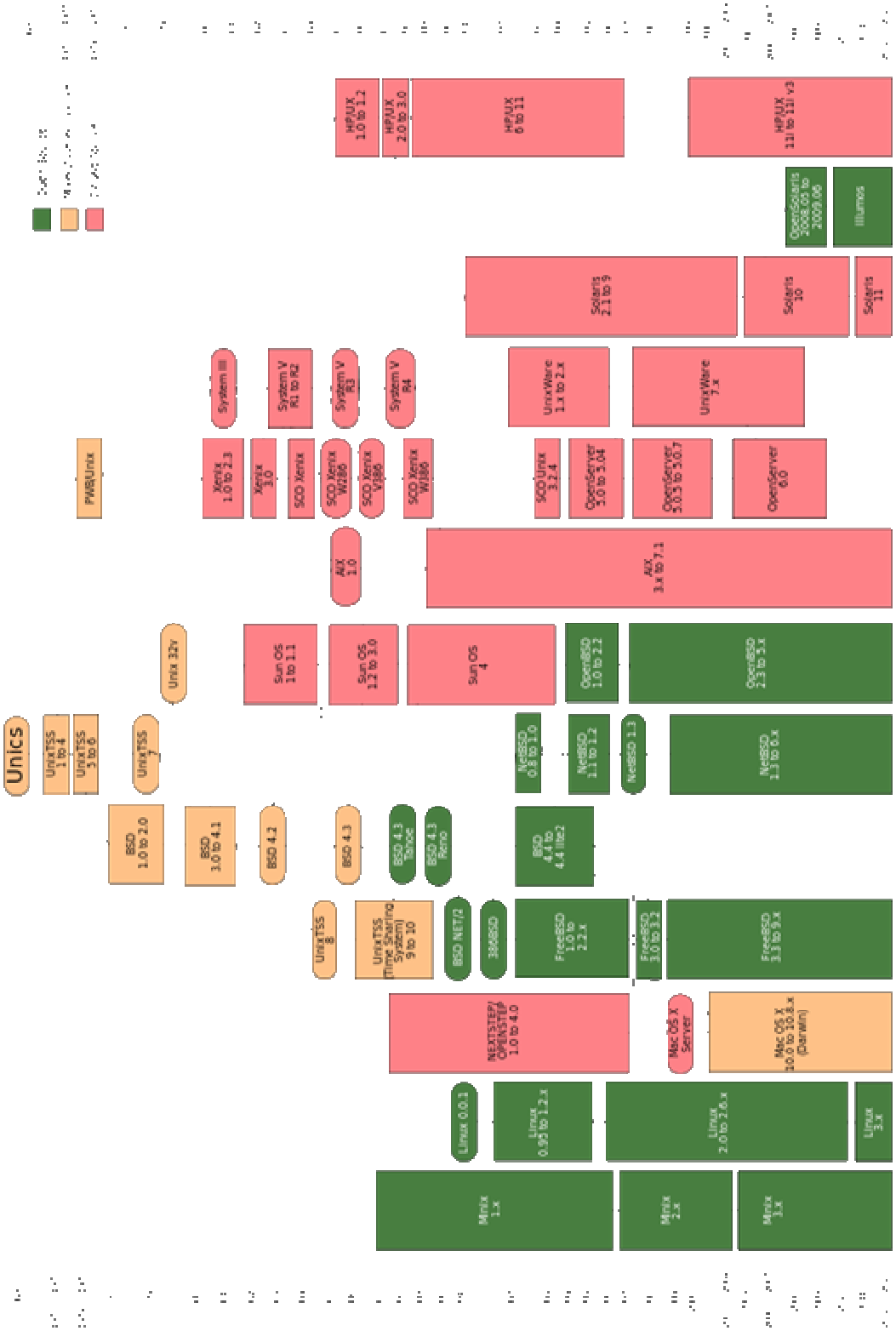
Figura GaSi87: 1000A 30KeV 10E4 88º

ANEXOS

Anexo A – Tabela Periódica



Anexo B – História do UNIX/Linux



Anexo C – Parâmetros de funcionamento do FIB/SEM Nova 2000

As tabelas a seguir possuem os parâmetros de remoção (templates) do FIB/SEM. São os dados necessários para calcular a remoção de 1 μm^3 de Si, utilizando o ângulo padrão do feixe (que é de 90 graus). A dose é definida pelo volume total em função da taxa de remoção (*sputtering*); e o feixe desta tabela é o diâmetro da abertura do feixe de íons do equipamento.

Tabela C1: Parâmetros para remoção de silício com FIB/SEM (5kV e 10kV)

Parâmetros de remoção do FIB/SEM					
ID	Energia	Corrente	Dose	Feixe	Tempo para corrosão
1	5 kV	0.15 pA	2.250E4 pm^3/nC	21.0 nm	12:32:48
2	5 kV	2.00 pA	3.000E5 pm^3/nC	29.0 nm	0:55:38
3	5 kV	10.00 pA	1.500E6 pm^3/nC	46.0 nm	0:11:23
4	5 kV	14.00 pA	2.100E6 pm^3/nC	49.2 nm	0:08:05
5	5 kV	29.00 pA	4.350E6 pm^3/nC	61.2 nm	0:03:55
6	5 kV	70.00 pA	1.050E-2 nm^3/nC	78.8 nm	0:01:40
7	5 kV	0.12 nA	1.800E-2 nm^3/nC	93.0 nm	0:00:58
8	5 kV	0.23 nA	3.450E-2 nm^3/nC	115 nm	0:00:31
9	5 kV	0.60 nA	9.000E-2 nm^3/nC	117 nm	0:00:11
10	5 kV	1.00 nA	1.455E-1 nm^3/nC	207 nm	0:00:08
11	5 kV	1.40 nA	2.100E-1 nm^3/nC	246 nm	0:00:06
12	5 kV	8.40 nA	1.260 nm^3/nC	943.2 nm	0:00:02
1	10 kV	0.30 pA	4.500E4 pm^3/nC	13.0 nm	6:11:09
2	10 kV	3.00 pA	4.500E5 pm^3/nC	20.0 nm	0:37:47
3	10 kV	16.00 pA	2.400E6 pm^3/nC	29.7 nm	0:07:05
4	10 kV	23.00 pA	3.450E6 pm^3/nC	32.9 nm	0:04:51
5	10 kV	50.00 pA	7.500E6 pm^3/nC	41.0 nm	0:02:15
6	10 kV	0.12 nA	1.800E-2 nm^3/nC	51.0 nm	0:00:58
7	10 kV	0.21 nA	3.150E-2 nm^3/nC	60.0 nm	0:00:33
8	10 kV	0.41 nA	6.150E-2 nm^3/nC	75.6 nm	0:00:17
9	10 kV	1.10 nA	1.650E-1 nm^3/nC	107.6 nm	0:00:06
10	10 kV	1.80 nA	2.700E-1 nm^3/nC	132.8 nm	0:00:04
11	10 kV	2.60 nA	3.900E-1 nm^3/nC	161.6 nm	0:00:03
12	10 kV	15.00 nA	2.250 nm^3/nC	821.0 nm	823.9 ms

Tabela C2: Parâmetros para remoção de silício com FIB/SEM (20kV e 30kV)

Parâmetros de remoção do FIB/SEM					
ID	Energia	Corrente	Dose	Feixe	Tempo para corrosão
1	20 kV	0.70 pA	1.050E5 pm^3/nC	8.4 nm	2:39:57
2	20 kV	4.00 pA	6.000E5 pm^3/nC	12.5 nm	0:28:08
3	20 kV	23.00 pA	3.450E6 pm^3/nC	19.3 nm	0:04:50
4	20 kV	37.00 pA	5.550E6 pm^3/nC	22.4 nm	0:03:03
5	20 kV	81.00 pA	1.215E-2 nm^3/nC	28.1 nm	0:01:24
6	20 kV	0.21 nA	3.150E-2 nm^3/nC	36.6 nm	0:00:32
7	20 kV	0.38 nA	5.700E-2 nm^3/nC	44.8 nm	0:00:18
8	20 kV	0.76 nA	1.140E-1 nm^3/nC	56.3 nm	0:00:09
9	20 kV	2.10 nA	3.150E-1 nm^3/nC	84.5 nm	0:00:03
10	20 kV	3.50 nA	5.250E-1 nm^3/nC	111.0 nm	0:00:02
11	20 kV	5.30 nA	7.950E-1 nm^3/nC	143.8 nm	0:00:01
12	20 kV	20.00 nA	3.000 nm^3/nC	533.0 nm	426.2ms
1	30 kV	1.00 pA	1.500E5 pm^3/nC	7.0 nm	1:51:20
2	30 kV	10.00 pA	1.500E6 pm^3/nC	12.0 nm	0:11:09
3	30 kV	30.00 pA	4.500E6 pm^3/nC	16.0 nm	0:11:09
4	30 kV	50.00 pA	7.500E6 pm^3/nC	19.0 nm	0:02:15
5	30 kV	0.10 nA	1.500E-2 nm^3/nC	23.0 nm	0:01:07
6	30 kV	0.30 nA	4.500E-2 nm^3/nC	33.0 nm	0:00:23
7	30 kV	0.50 nA	7.500E-2 nm^3/nC	39.0 nm	0:00:14
8	30 kV	1.00 nA	1.500E-1 nm^3/nC	50.0 nm	0:00:07
9	30 kV	3.00 nA	4.500E-1 nm^3/nC	81.0 nm	0:00:02
10	30 kV	5.00 nA	7.500E-1 nm^3/nC	110.0 nm	0:00:01
11	30 kV	7.00 nA	1.050 nm^3/nC	141.0 nm	0:00:01
12	30 kV	20.00 nA	3.000 nm^3/nC	427.0 nm	410.3ms

Cálculo de dose (íons/cm^2)

Dose = (Corrente * tempo) / área do spot = coloumb/cm^2

D = (I * t) / A

A = (pi*feixe^2)/4

Calculando a dose no FIB/SEM com os seguintes parâmetros:

Energia: 30kV

Spot-size (feixe) = 7nm

Tempo = 1 segundo

A = (3.1416 * (7E-9*7E-9)) / 4 = (3.1416 * (49E-18)) / 4 = (153,934) / 4 = 3.84*10E-13 cm^2

Corrente de 1 picoampere (pA) = 1E-12 ampere

D = (1E-12*1)/3.84*10E-13 = 2.6 coloumb/cm^2

Ions/cm^2 é obtido com D dividido pela carga de 1 eletron (e = 1,602E-19 coloumbs)

Portanto D = 2.6 / 1.602E-19 = 1.623E19 ions/cm^2 que é a dose mínima do FIB/SEM

Como parâmetro para comparação, no implantador EATON GA4204, a dose mínima para utilização no equipamento com as mesmas condições é de 10E11 ions/cm^2.

Tabela C3: Volume removido por dose para alguns materiais a 30kV

TABLE 6-23 MATERIAL SPUTTER RATES AT 30kV			
Material	**Volume per Dose (μm^3/nC)**	**Pt-XML**	**Volume per Dose (μm^3/nC)**
C	0.18	Au	1.50
Si	0.27	MgO	0.15
Al	0.30	SiO_2	0.24
Ti	0.37	Al_2O_3	0.08
Cr	0.10	TiO	0.15
Fé	0.29	Si_3N_4	0.20
Ni	0.14	TiN	0.15
Cu	0.25	Fe_2O_3	0.25
Mo	0.12	GaAs	0.61
Ta	0.32	Pt	0.23
W	0.12	PMMA	0.40

*Fonte: Manual do FIB/SEM FEI Nanolab 200