Chave *Crossbar* Reconfigurável para Implementação Dinâmica de Topologias em Redes de Interconexão de Dados

Henrique Cota de Freitas[1] Carlos Augusto Paiva da Silva Martins[2]
Instituto de Informática[1,2]
Programa de Pós-graduação em Engenharia Elétrica[2]
Grupo de Sistemas Digitais e Computacionais[1,2]
Pontifícia Universidade Católica de Minas Gerais
{cota, capsm}@pucminas.br

Resumo

O aumento de desempenho e flexibilidade dos equipamentos de rede é uma exigência que cresce a cada dia, decorrente do grande uso dos meios de comunicação de dados, principalmente a Internet. Por este motivo, nos últimos anos, as pesquisas envolvendo os equipamentos de rede têm se tornado freqüentes nas universidades e indústrias. Neste artigo descrevemos o projeto de uma chave crossbar para unidade de chaveamento de dados usando conceitos de computação reconfigurável na implementação dinâmica de topologias. Os resultados obtidos através de modelo em grafos e simulação foram comparados com uma chave crossbar tradicional como forma de verificação do aumento de desempenho e flexibilidade de interconexão e chaveamento de dados.

1. Introdução

Nos últimos anos o grande número de serviços disponibilizados na Internet e nos demais tipos de rede de comunicação de dados [1][20] tem provocado o aumento de pesquisas com o objetivo de melhorar o desempenho e a qualidade de serviço oferecida. Sendo assim, muitos dos equipamentos de redes [9][19] sofreram nos últimos anos avanços consideráveis em sua arquitetura de processamento e chaveamento de pacotes de dados resultando no surgimento de Processadores de Rede [3][5][6][7] e *Switching Fabrics* [3][5][18] específicos e dedicados para trabalhar com o aumento de qualidade e demanda de serviço. Os equipamentos de rede, em específico o roteador, são considerados pontos de concentração e por isso, são muitas vezes os gargalos em uma rede de comunicação de dados. Os *Switching Fabrics* são unidades encarregadas de oferecer o meio pelo qual será realizada a interconexão entre os diversos nós de uma rede de dados. Sendo assim, a escolha da arquitetura de *Switching Fabric* pode influenciar no desempenho [15] final do equipamento de rede.

Atualmente estamos desenvolvendo e implementando uma chave *crossbar* reconfigurável para chaveamento de dados e implementação dinâmica de topologias. A flexibilidade de operação, agregada a um aumento da vazão de dados e diminuição da latência é um fator motivador para o projeto de uma arquitetura de chave *crossbar* usando conceitos de computação reconfigurável [10][14]. Portanto nosso principal objetivo neste artigo é apresentar as principais características e os resultados parciais da Chave *Crossbar* Reconfigurável (RCS: *Reconfigurable Crossbar Switch*) descrita no tópico 3.

Um dos fatores mais relevantes da pesquisa que estamos realizando é o fato da nossa proposta de chave *crossbar* agregar conceitos de reconfiguração. Por se tratar ainda de uma área recente (computação reconfigurável), não encontramos nenhum trabalho correlato que utilizasse a mesma idéia de arquitetura para uma unidade de chaveamento de dados.

2. Unidades de Chaveamento de Pacotes de Dados

O chaveamento de pacotes de dados em uma rede é feito através de redes de interconexão [4]. As redes de interconexão são divididas em estáticas e dinâmicas e a definição destes dois tipos de redes está baseada na estrutura de interconexão (topologia).

Redes Estáticas: Neste tipo de rede a topologia adotada oferece um caminho fixo da origem até o destino. Desta forma, podemos entender como estática a rede de interconexão onde a comunicação se faz por conexões fixas, que não mudam ao longo do tempo.

Redes Dinâmicas: Ao contrário das redes estáticas, as redes dinâmicas oferecem alternativas de conexão. A topologia não possui pontos fixos de conexão. Ou seja, o caminho de comunicação não é fixo. Portanto, podemos considerar que em uma rede dinâmica as conexões são realizadas de acordo com a necessidade de comunicação ou transferência de dados.

Existem unidades ou estruturas de chaveamento chamadas *Switching Fabrics* [3][5][18]. Estas unidades são responsáveis por estabelecer o meio de interconexão para que os pacotes possam ser redirecionados ou roteados. Podemos classificar os *Switching Fabrics* de acordo com os seguintes estágios:

Divisão no tempo: Esta classificação se refere às unidades que utilizam multiplexação de dados no tempo para alcançar o chaveamento entre entrada e saída. A grande vantagem da utilização de *Switching Fabrics* por divisão no tempo é a não necessidade de pontos de cruzamento como alternativas de caminhos para transmissão de dados. Porém a maior desvantagem deste tipo de unidade de chaveamento é o atraso ocorrido na espera do processamento de conexões.

Divisão no espaço: O chaveamento entre entradas e saídas é feito através de caminhos realizados espacialmente, através de pontos de conexão. A grande vantagem da utilização de unidades de chaveamento no espaço se refere ao desempenho devido à maior vazão de dados, uma vez que não há atraso por espera de processamento. Porém, a maior desvantagem se refere a grande quantidade de pontos de conexão, que podem ocasionar um alto custo e em alguns casos funções bloqueantes.

Uma chave *crossbar* tradicional se enquadra na classificação de redes dinâmicas com chaveamento por divisão no espaço. É baseado nesta estrutura que desenvolvemos nossa proposta apresentada nas seções seguintes.

3. Proposta da Chave *Crossbar* Reconfigurável

Uma chave *crossbar tradicional* (sem suporte a *broadcast* ou *multicast*) é capaz de fechar, em intervalos de tempo, a mesma entrada com diversas saídas e vice-e-versa, ao mesmo tempo não é possível fechar todos os pontos de uma coluna ou de uma linha. Portanto, não é possível implementar uma topologia fixa (no espaço), já que as diversas conexões são abertas ou fechadas ao longo do tempo. Entretanto, a nossa proposta de chave *crossbar* reconfigurável, além de funcionar como a chave *crossbar* tradicional, implementa topologias também no espaço através de seus bits de reconfiguração.

Reconfigurar significa mudar a estrutura ou mudar a forma. Portanto, um dispositivo reconfigurável altera sua forma a cada novo conjunto de bits de reconfiguração.

A arquitetura da chave *crossbar* que estamos desenvolvendo possui dois níveis de reconfiguração (figura 1). No primeiro nível a reconfiguração está baseada na estrutura e tamanho da chave *crossbar*, portanto, é uma reconfiguração dependente do dispositivo de prototipação reconfigurável, como por exemplo, um FPGA (*Field Programmable Gate Array*) [17][21]. Os resultados de reconfiguração da chave *crossbar* apresentados neste artigo e que representam uma contribuição, são referentes ao segundo nível de reconfiguração, que é independente do dispositivo de prototipação reconfigurável, uma vez que a reconfiguração é realizada sobre a chave *crossbar* implementada, no entanto usamos um FPGA para realizar os testes e obter os resultados iniciais.

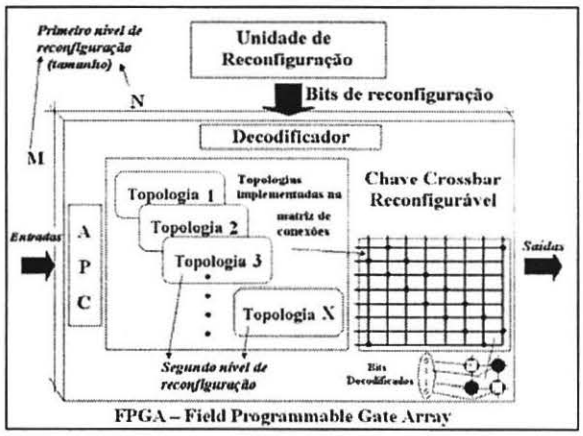

Figura 1. Arquitetura da Chave *Crossbar*

Nesta arquitetura, os bits de reconfiguração (podem representar fechamento de um nó, de uma coluna, de uma linha ou da matriz inteira) são decodificados (Decodificador) para fechar ou abrir um nó da chave *crossbar*. Adotamos o bit 0 como nó aberto e bit 1 como nó fechado. Após a reconfiguração, a topologia desejada está implementada sobre a chave *crossbar* e somente através dos pontos conectados é que poderá haver comunicação entre as entradas e saídas. O bloco APC (Analisador de Pré-cabeçalho) é utilizado em situações em que um processador de rede [3][5][6] insere novas informações no cabeçalho do pacote enviado. Os resultados do bloco APC não são apresentados neste artigo. Apesar de uma chave *crossbar* tradicional usar os bits 0 e 1 para realizar as

conexões entre entrada e saída, ela não implementa topologias. Na chave *crossbar* tradicional existe a sensação, ao longo do tempo, de uma topologia implementada, mas ela não existe fisicamente, ela é temporal e não espacial.

De acordo com a figura 1, a chave *crossbar* reconfigurável possui 8 entradas e 8 saídas ou 64 nós reconfiguráveis. A grande vantagem é poder contar com uma conexão pronta (topologia implementada) para manter a comunicação entre os pontos de origem e destino, sem que se perca tempo de processamento e fechamento de conexões, que poderiam retardar o envio de um pacote de dados. O Processador de Rede de um equipamento irá processar questões mais nobres como, por exemplo, relativas à qualidade de serviço.

A figura 2 ilustra uma topologia anel sendo implementada na chave *crossbar* reconfigurável. Somente os nós que receberam o bit 1 foram fechados os demais ficaram abertos. Neste caso, cada entrada e respectiva saída estão conectadas a computadores, somente haverá comunicação, por exemplo, do computador 1 para os computadores 2 e 8 e vice-e-versa. Nesta topologia o computador 1 somente irá se comunicar com o computador 5 se houver a passagem do pacote entre os caminhos 2, 3 e 4 ou 8, 7 e 6.

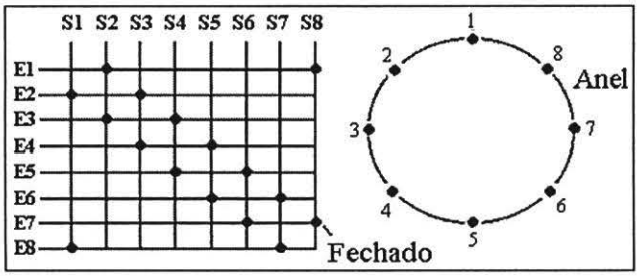

Figura 2. Topologia anel implementada sobre RCS

Uma das características da arquitetura proposta é a flexibilidade de implementação dinâmica (durante tempo de execução) de topologias através de bits de reconfiguração. No entanto, a reconfiguração não é um fator limitante, uma vez que o Processador de Rede pode requisitar uma nova conexão para a unidade de reconfiguração. Em trabalhos futuros mencionamos a RCS-2 (Chave *Crossbar* Reconfigurável – 2 bits) que possibilita a influência direta de instruções do Processador de Rede na alteração das conexões ou bits de reconfiguração.

Procuramos por **trabalhos correlatos** [2][8][13], mas o que encontramos usam o FPGA como base para reconfiguração, o que não corresponde a nossa proposta de implementação dinâmica de topologias, que usa uma reconfiguração de segundo nível (figura 1) independente de plataforma de prototipação, uma vez que a reconfiguração atua em um nível acima do hardware de prototipação, neste caso, um FPGA.

4. Resultados

Neste tópico apresentamos resultados obtidos através do uso de teoria de grafos e simulações comportamentais da chave *crossbar* reconfigurável. Procuramos mostrar as vantagens da RCS em relação a uma chave *crossbar* padrão (tradicional) conforme descrito nos tópicos 4.1 e 4.2.

4.1. Análise de Implementação Dinâmica de Topologias Usando Teoria de Grafos

Existem algumas métricas adotadas para a modelagem de grafos [16][11][12] aplicada às redes de comunicação de dados [1][20]. Iremos adotar duas para analisar o impacto da utilização da chave *crossbar* reconfigurável como unidade de chaveamento de pacotes e implementação dinâmica de topologias em uma rede de interconexão de dados. São elas:

Grau médio do grafo: Podemos considerar como grau médio de um grafo, o quanto os vértices estão conectados entre si através das arestas, proporcionando uma maior possibilidade de caminhos alternativos.

Grau médio = 2*QtdAresta / QtdVértice

Diâmetro do grafo: Podemos considerar como diâmetro de um grafo, o maior entre os menores caminhos (quantidade de saltos) entre qualquer dois vértices do grafo. Desta forma, podemos dizer que o menor diâmetro também é o menor caminho ente dois vértices. Neste caso, não levado em consideração os possíveis e diferentes pesos nas arestas.

Com base nestas duas métricas procuramos apresentar como a flexibilidade de implementação dinâmica de topologias pode ajudar no desempenho de processamento de dados.

A figura 3 apresenta três topologias que poderiam ser implementadas na chave *crossbar* reconfigurável (em tempo de execução) de acordo com a necessidade de processamento de pacotes.

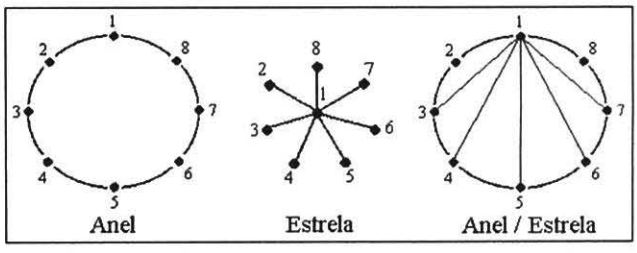

Figura 3. Topologias implementadas

Para modelarmos as topologias de comunicação de dados em grafos adotamos que os nós das redes são os vértices e as linhas de comunicação são as arestas.

Em uma situação hipotética, o seguinte problema poderia ser apresentado (figura 3): uma topologia anel é implementada na chave *crossbar* reconfigurável através dos bits de reconfiguração. Porém, constatou-se um crescente número de envio de pacotes do nó 1 para os demais nós da topologia, tal como um *broadcast*. Neste caso, se apenas o nó 1 estabelecesse comunicação com os demais, a melhor topologia seria estrela. Contudo, as demais comunicações permanecem e por este motivo devem ser mantidas na implementação da nova topologia.

Como a chave *crossbar* é reconfigurável, as conexões diretas que faltavam foram implementadas dinamicamente. Desta foram, diminui-se o número de saltos para a comunicação e conseqüentemente aumenta o desempenho e a vazão de dados do nó 1 para os demais vértices.

Para a topologia apresentada no início (anel) e para a topologia final (anel/estrela), os seguintes resultados com base nas duas métricas de modelagem em grafos são:

Topologia Anel (8 Vértices e 8 Arestas):

Grau médio do grafo = 2*8/8 = **2**

Menor Diâmetro do vértice 1 ao 5 = **4**

Topologia Anel/Estrela (8 Vértices e 13 Arestas):

Grau médio do grafo = 2*13/8 = **3,25 = 3**

Menor Diâmetro do vértice 1 ao 5 = **1**

Portanto, as métricas baseadas em modelos de grafos mostraram o quanto é importante uma topologia específica ou mais otimizada. Há um aumento da capacidade de estabelecer a comunicação (grau médio do grafo) com os demais vértices e por este motivo, pode haver uma diminuição no tempo de resposta de comunicação (menor diâmetro – caminho entre vértices) e por conseqüência uma melhoria no desempenho de processamento e vazão de pacotes.

É importante ressaltar, que o cálculo do diâmetro com base apenas no número de saltos pode não ser satisfatório, uma vez que as arestas podem ter pesos diferentes (linhas de comunicação com custos – tráfegos – ou latências diferentes). Porém, em casos onde estes pesos são iguais em todas as arestas, esta análise é suficiente para o cálculo de menor caminho. No nosso caso, todas as arestas possuem o mesmo peso, já que cada aresta representa uma conexão realizada pela chave *crossbar* (entre entradas e saídas). Desta forma, o resultado do cálculo de menor diâmetro reflete a realidade da chave *crossbar* reconfigurável proposta.

No entanto, podemos considerar que existe a influência do tráfego, tempo de transmissão ou latência para que a mensagem enviada para uma saída chegue ao seu destino, em algum outro ponto da rede. Ou seja, todas as saídas da chave *crossbar* levam a outras redes ou outros destinos, coincidentes ou não. Sendo assim, vamos considerar que somente a latência influencia no cálculo de menor caminho e que toda a influência externa está modelada no grafo da figura 4, onde cada nó representa um roteador. Porém, para o roteador R-1 (nó 1) a sua RCS possui sete entradas e sete saídas conectadas aos demais nós ou roteadores da rede e os demais possuem 2 ou 3 entradas e saídas conectadas aos demais.

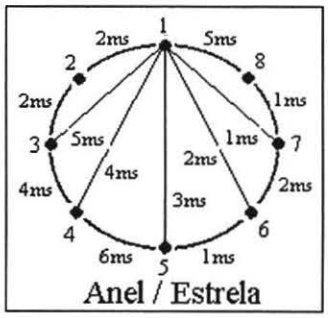

Figura 4. Arestas do grafo com peso (latência em ms)

Neste grafo, as latências entre os nós são dadas em milisegundos (ms). Usaremos o algoritmo conhecido como roteamento com vetor de distância [20] para definição da melhor rota para envio dos pacotes de dados.

Neste algoritmo, cada roteador (presente em cada nó da topologia) mantém uma tabela (tabela de roteamento resultante – figura 5) que fornece a menor distância entre a origem e cada destino da rede. Estas tabelas são atualizadas pelos roteadores vizinhos (ou nós adjacentes) informando a distância entre estes roteadores vizinhos e cada outro roteador da rede. Desta forma, com as informações enviadas pelos vizinhos é construída uma nova tabela com a menor distância resultante para se chegar a qualquer roteador da rede, usando como primeiro passo o roteador vizinho que oferecer o menor caminho.

A figura 5 ilustra as duas tabelas (enviada pelos vizinhos e resultante de distância) com um exemplo evidenciado: Se for necessário enviar um pacote do nó 5 para o nó 3, qual seria o melhor caminho?

Se as novas conexões entre o nó 1 e todos os outros nós da rede não fossem realizadas (figura 3 – Anel), a resposta seria do nó 5 para o 4 finalizando nó 3, com uma latência total de 10ms. Porém com as novas conexões (figura 3 – Anel/Estrela), verificamos que existem 3 caminhos para se chegar ao destino, possuímos uma maior opção de roteamento. De acordo com as informações enviadas pelos nós adjacentes

(figura 5) a melhor opção de roteamento seria através do nó 1 com latência total de 7ms. Ou seja, o pacote de dados seria 3ms mais rápido se comparado à topologia anterior (anel) onde haveria apenas dois caminhos alternativos (pelo nó 4 ou pelo nó 6).

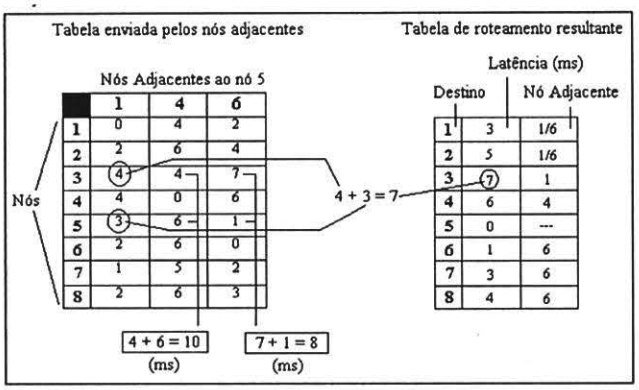

Figura 5. Tabelas de roteamento para o nó 5

Neste tópico mostramos a influência e o impacto da implementação dinâmica de topologias para definição de rotas. Independente da influência externa, o aumento do grau médio e a diminuição do diâmetro entre dois nós (métricas alteradas pela reconfigurabilidade da chave *crossbar*), foram responsáveis diretos pelos números obtidos na tabela de roteamento resultante ilustrado na figura 5. Outras análises usando algoritmos diferentes podem ser obtidas através das referências [16] e [20].

Em uma chave *crossbar* tradicional não teríamos uma topologia implementada, haveria um acréscimo de tempo de comutação para o fechamento de um nó e, portanto, para o roteamento de cada pacote de dados. Este atraso de comutação pode ser percebido pelos resultados de simulação apresentados no tópico 4.2.

4.2. Simulação Comportamental da Chave *Crossbar* Reconfigurável

Neste tópico, apresentamos duas simulações comportamentais realizadas em ambiente de descrição e simulação de hardware da Xilinx [21]: i) chave *crossbar* reconfigurável e ii) chave *crossbar* tradicional.

A RCS foi simulada usando um modelo conceitual de arquitetura (matriz 4x4 para facilitar a apresentação das figuras) com as seguintes entradas e saídas:

Entrada0, Entrada1, Entrada2, Entrada3: Representam as quatro entradas de dados da chave *crossbar* por onde chegam os pacotes de dados. No teste de simulação estas entradas foram agrupadas em um vetor de 4 bits.

Nó_00 até Nó_33: Representam os dezesseis pontos de conexão ou bits responsáveis pela reconfiguração da chave *crossbar*. As numerações que seguem as palavras "Nó" são referentes aos índices de uma matriz 4x4.

RegHabilitaEntrada: Na nossa proposta de chave *crossbar* reconfigurável é possível fechar uma conexão entre uma entrada e várias saídas, ou vice-e-versa. Por este motivo é necessário que se tenha um controle contra conflitos. Optamos por acrescentar um registrador ("RegHabilitaEntrada") que habilita o envio de dados para simplificar a arquitetura simulada.

Saída0, Saída1, Saída2, Saída3: Representam as quatro saídas de dados da chave *crossbar*. No teste de simulação estas entradas foram agrupadas em um vetor de 4 bits.

Nos resultados de simulação apresentados na figura 7, podemos observar três diferentes topologias implementadas ao longo do tempo, conforme ilustrado pela figura 6.

Nos primeiros 400ns a topologia anel bidirecional foi implementada através dos bits de reconfiguração. Neste intervalo de tempo, nós podemos observar que o registrador "RegHabilitaEntrada" habilita a Entrada0. Sendo assim, a Saída1 e Saída3 que foram conectadas à Entrada0 através dos pontos de conexão Nó_01 e Nó_03, recebem os sinais da Entrada0 durante 400ns.

No intervalo de 400ns a 800ns a topologia foi reconfigurada para árvore bidirecional. Neste instante o registrador "RegHabilitaEntrada" habilita a Entrada2, mas não há nenhum dado a ser enviado. Por este motivo nos primeiros 100ns as portas de saídas não receberam nenhum sinal. Após os 500ns as entradas 0 e 2 enviam sinal para a chave *crossbar*, porém o registrador está habilitando apenas a Entrada2, e por conseqüência a Saída0 é a única a receber o sinal vindo da Entrada2.

Nos últimos 200ns foi implementada a topologia estrela bidirecional. Nos primeiros 100ns nenhuma entrada da *crossbar* foi habilitada. Após estes 100ns, o registrador "RegHabilitaEntrada" habilita a Entrada3 para enviar dados. De acordo com a topologia estrela, a entrada três está conectada através do Nó_30 à Saída0, por este motivo somente esta saída irá receber o sinal enviando por Entrada3.

Através dos resultados obtidos desta simulação, observamos que o funcionamento da RCS corresponde à proposta apresentada de flexibilidade. Ou seja, ao longo do tempo e durante o período de trabalho da chave, é possível alterar a topologia implementada, sem que haja a necessidade de parar o funcionamento, já que esta implementação (reconfiguração) ocorre em tempo de execução.

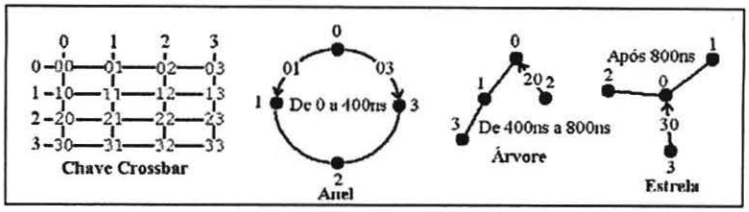

Figura 6. Topologias simuladas sobre a RCS

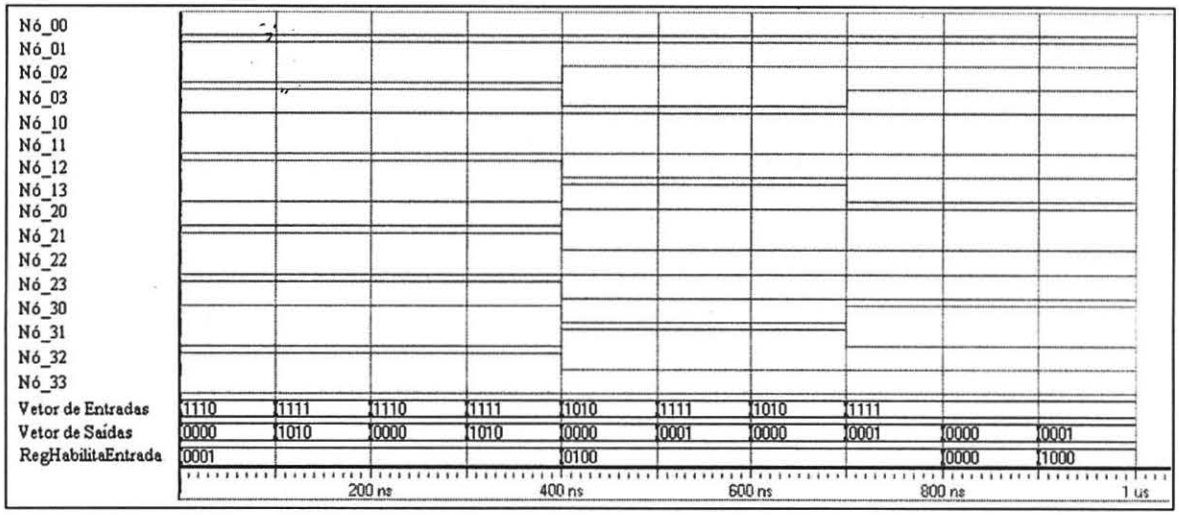

Figura 7. Resultados da simulação da RCS

Realizamos também uma simulação para uma chave *crossbar* tradicional como forma de obtermos uma comparação entre as duas. Neste caso, há apenas um fechamento de conexão entre entrada e saída no intervalo de tempo. Não podemos, portanto, ter uma mesma entrada conectada a várias saídas ao mesmo tempo (simultaneamente).

Como as conexões (ou topologias) não estão implementadas espacialmente em uma chave *crossbar* tradicional, é necessário indicar qual entrada e qual saída deverão efetuar a comunicação. Para isto, usamos dois registradores:

Regin: registrador que indica qual a entrada.

Regout: registrador que indica qual a saída.

Estes registradores definem qual entrada e saída irão se comunicar, e, portanto, qual nó da chave *crossbar* irá fechar. Exemplo:

Regin = 0001 (entrada 0)

Regout = 0010 (saída 1)

Nó fechado = 01 (A_{01} de uma matriz – figura 8)

A figura 8 ilustra as conexões fechadas na simulação e qual topologia poderia ser representada no tempo. Iniciamos a simulação da topologia estrela até a topologia pirâmide como forma de mostrar uma seqüência de conexões que representam topologias não implementadas no espaço e sim no tempo.

A figura 9 ilustra o resultado de simulação da chave *crossbar* tradicional. Podemos verificar ao final dos 300ns que a Entrada0 enviou dados para as demais saídas da chave *crossbar* (1,2 e 3). Sendo assim, a topologia estrela foi implementada no tempo (usando técnicas de multiplexação) e não no espaço (uma topologia fixa).

Ao fim dos 600ns verificamos que os nós conectados ao nó central (nó 0) também começaram a se conectar, formando juntamente com as conexões realizadas de 0 a 300ns uma topologia temporal chamada pirâmide.

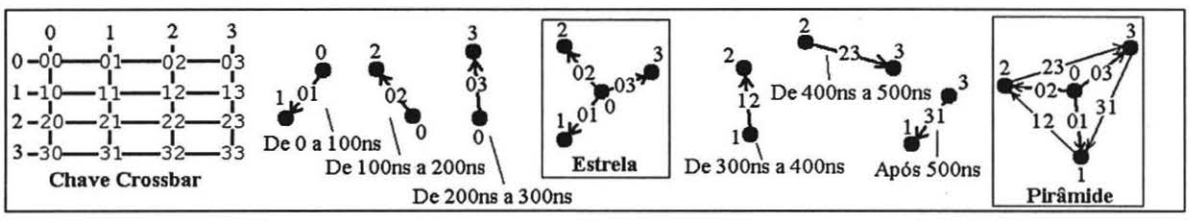

Figura 8. Conexões simuladas na chave *crossbar* tradicional

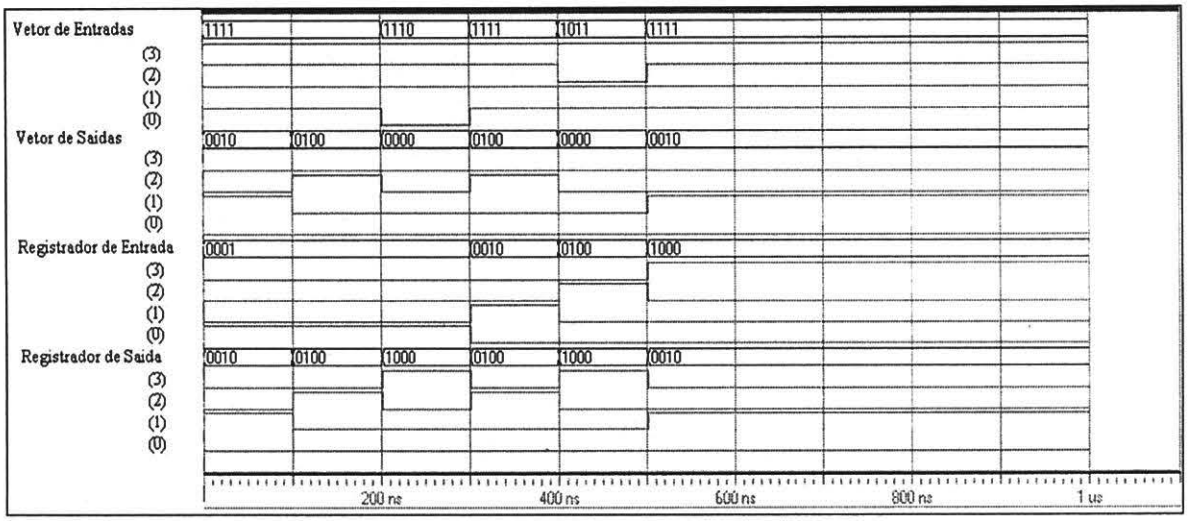

Figura 9. Resultado de simulação da chave *crossbar* tradicional

Portanto, a chave *crossbar* tradicional é capaz de realizar conexões espaciais, mas realiza a implementação de topologias de forma temporal. Se fosse necessário realizar um *broadcast* neste exemplo (Entrada 0 para Saídas 1, 2 e 3), e os intervalos de tempo de simulação fossem considerados intervalos de chaveamento, a RCS levaria duas unidades de tempo a menos para realizar o *broadcast*. Em uma matriz 8x8 a RCS levaria seis unidades de tempo a menos do que uma chave *crossbar* tradicional para realizar o *broadcast*. Ou seja, quanto maior a matriz de conexões da chave *crossbar*, maior será o ganho favorável a RCS, devido principalmente ao tempo de chaveamento (conexão) ser apenas um para implementação dinâmica (tempo de execução) de topologias.

5. Conclusões

Aplicando as métricas descritas para modelagem de grafos, que são o diâmetro e o grau médio de um grafo, foi possível verificar os resultados da utilização da computação reconfigurável, que é o aumento de flexibilidade e desempenho para chaveamento de pacotes de dados e a implementação dinâmica de topologias. Estas duas métricas representam uma maior vazão de dados ou uma maior possibilidade de caminhos alternativos, já que aumenta a quantidade de arestas ou *links* presentes. Isto ficou evidenciado quando usamos o algoritmo de vetor de distância para verificar a escolha da melhor rota com base em influências de latência dos meios externos.

Os resultados de simulação comportamental verificam o funcionamento da chave *crossbar* reconfigurável. Nestas simulações foi possível constatar que a implementação dinâmica de topologias facilita a comunicação entre os diversos pontos conectados na RCS e possibilita a utilização de uma topologia mais específica ou dedicada de acordo com as necessidades das aplicações. Em relação à chave *crossbar* tradicional, verificamos situações onde foi necessário fechar 3 conexões (uma após a outra) ao longo do tempo. Na RCS, com a topologia implementada, estas 3 conexões já estão fechadas e, portanto, ao contrário de 3 unidades de tempo para enviar um pacote para 3 destinos diferentes, gastaríamos apenas 1 unidade de tempo. Ou seja, um ganho de 3 vezes em relação a chave *crossbar* tradicional.

Uma chave *crossbar* tradicional é uma unidade de chaveamento espacial. Ou seja, ela não realiza a comunicação entre dois pontos de forma temporal utilizando técnicas de multiplexação (meio compartilhado – barramento, por exemplo). No entanto, quando uma determinada coluna da chave já está conectada, nenhum outro ponto desta coluna poderá ser fechado. Podemos considerar então que cada topologia ocorre no tempo e, portanto, uma comunicação pode ter que esperar que um outro chaveamento tenha que ser desfeito se a mesma coluna tiver que ser usada. Concluímos então, que uma chave *crossbar* tradicional é uma unidade de chaveamento espacial e de topologias temporais.

A chave *crossbar* reconfigurável (RCS) proposta é uma unidade de chaveamento espacial e de topologias espaciais. O motivo pelo qual classificamos assim, se deve ao fato de existir a implementação dinâmica de topologias na chave *crossbar* reconfigurável. Estas topologias são representações de interconexões espaciais ou de chaveamentos espaciais, não é necessário esperar um encerramento de conexão ou um fechamento de conexão para que outro chaveamento seja realizado, pois ele já está pronto.

A principal contribuição deste artigo é a proposta de chave *crossbar* usando conceitos de computação reconfigurável em dois níveis diferentes. O segundo nível, que foi apresentado neste artigo, aborda reconfiguração sobre a chave *crossbar* (matriz de conexões), tornando este tipo de reconfiguração independente de plataforma de prototipação. É importante ressaltar, que a utilização de dois níveis de reconfiguração aumenta as possibilidades e combinações de topologias que podem ser implementadas na chave *crossbar*.

Os trabalhos futuros são: finalizar os testes da nova versão da RCS com 2 bits de reconfiguração por ponto de conexão; analisar a escalabilidade, custos de implementação e velocidade do circuito e prototipar as duas versões da RCS juntamente com a arquitetura do Processador de Rede (R2NP: *Reconfigurable RISC Network Processor*) [5][7] que também esta em fase de desenvolvimento.

6. Agradecimentos

Aos colegas do Grupo de Sistemas Digitais e Computacionais (GSDC), em especial Amanda R. M. Diniz e a Pró-Reitoria de Pesquisa e de Pós-graduação da PUC Minas.

7. Referências

[1] Buyya, R., "High Performance Cluster Computing", Volume 1, Prentice Hall, 1999

[2] C. Fewer, et al., "A High I/O Reconfigurable Crossbar Switch", 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, April 09-13, 2003, Napa, California, pp.3-10, 2003

[3] Comer, D. E., "Network Systems Design Using Network Processors", Prentice Hall, 2003

[4] De Rose, C. A. F. e P. O. A. Navaux, "Arquiteturas Paralelas", Série Livros Didáticos, Editora Sagra Luzzatto, 2003

[5] Freitas, H. C., "Proposta e Desenvolvimento de Processador de Rede com Chave *Crossbar* Reconfigurável", dissertação de mestrado apresentada em 04 de novembro de 2003

[6] Freitas, H. C., C. A. P. S. Martins, "Processadores de Rede: Conceitos, Arquiteturas e Aplicações" (Capítulo de Livro), Minicurso apresentado na III Escola Regional de Informática RJ/ES, Vitória, 07 de outubro, 2003, pp.127-166

[7] H. C. Freitas, C. A. P. S. Martins, "Didactic Architectures and Simulator for Network Processor Learning", Workshop on Computer Architecture Education, WCAE'2003, San Diego, CA, USA, 2003, pp.86-95

[8] H. Eggers, P. Lysaght, H. Dick, G. McGregor, "Fast Reconfigurable Crossbar Switching in FPGAs", Proceedings of 6th International Workshop on Field-Programmable Logic and Applications, Springer-Verlag LNCS 1142, 1996, pp.297-306

[9] H. J. Chao, 'Next Generation Routers", Proceedings of the IEEE, Vol. 90, No. 9, September 2002, pp.1518-1558

[10] K. Compton, S. Hauck, "Reconfigurable Computing: A Survey of Systems and Software", ACM Computing Surveys, Vol. 34, No. 2, June 2002, pp. 171-210

[11] K. L. Calvert, M. B. Doar, E. W. Zegura, "A Quantitative Comparison of Graph-based Models for Internet Topology", IEEE/ACM Transactions on Networking, Vol. 5, No. 6, December 1992, pp.770-783

[12] K. L. Calvert, M. B. Doar, E. W. Zegura, "Modeling Internet Topology", IEEE Communications Magazine, June 1997, pp.160-163

[13] M. Keyvani, "VHDL Implementation of a High-Speed Symmetric Crossbar Switch", Dissertação de Mestrado, School of Engineering Science Communication Networks Laboratory, Simon Fraser University, August, 2001

[14] Martins, C. A. P. S., E. D. M. Ordonez, J. B. T. Corrêa e M. B. Carvalho, "Computação Reconfigurável: conceitos, tendências e aplicações", Jornada de Informática 2003, Congresso da Sociedade Brasileira de Computação, Capítulo 8, 2003

[15] Menascé, D. A., A. F. Almeida, "Planejamento de Capacidade para Serviços na Web, Métricas, Modelos e Métodos", Editora Campus, 2003

[16] Netto, P. O. B., "Grafos: Teoria, Modelos e Algoritmos", Segunda Edição, Editora Edgard Blücher, 2001

[17] Ordonez, E. D. M., F. D. Pereira, C. G. Penteado, R. A. Pericini, "Projeto, Desempenho e Aplicações de Sistemas Digitais em Circuitos Programáveis (FPGAs)", Bless Gráfica e Editora Ltda, 2003

[18] P. B. Thakker, "Survey of Switch Architectures", Relatório Técnico, ECE 412 – University of Illinois at Urbana-Champaign, December 11, 1998

[19] S. Keshav, R. Sharma, "Issues and Trends in Router Design", IEEE Communications Magazine, Vol. 36, No. 5, May 1998, pp.144-151

[20] Tanembaum, A. S., "Computer Networks", Prentice Hall, 4ª Edição, 2003

[21] Xilinx Development System, "Synthesis and Simulation Design Guide", 2002