Análise de Desempenho de Topologias para Redes em Chip

Nelson A. Gonçalves Junior, Ronaldo A. L. Gonçalves, João Angelo Martini
Departamento de Informática
Universidade Estadual de Maringá – Maringá, PR - Brasil
{njunior, ronaldo, jangelo}@din.uem.br

Resumo

Os avanços nos processos de fabricação de chips têm permitido um constante aumento na quantidade de transistores integrados em uma mesma pastilha de silício, possibilitando a associação de todos os componentes de um computador em um único chip. São os chamados Sistemas em Chip (SoCs – Systems on Chip), cuja complexidade vem aumentando frequentemente com a integração de diversos componentes e exigindo formas eficientes de realizar a comunicação entre esses componentes. Uma abordagem muito discutida atualmente para garantir essa comunicação é a utilização de redes em chip (NoCs – Networks on Chip), que mantêm chaves roteadoras para direcionar os pacotes de dados para seus respectivos destinos e são interligadas de acordo com determinada topologia. Nesse contexto, o presente trabalho busca investigar tais redes, analisando e comparando o desempenho de NoCs com topologias diferentes, para mostrar o impacto que a topologia pode ter no desempenho final de uma rede intra-chip.

1. Introdução

O avanço no processo de encapsulamento de transistores tem permitido a integração de milhões destes em uma única pastilha de silício, sustentando a Lei de Moore [1]. Este avanço possibilita a coexistência de diversos componentes em um chip, permitindo a manutenção de sistemas computacionais completos dentro de um único chip. Tais sistemas são denominados Sistemas em Chip (*Systems on Chip*) ou SoCs [2], comumente utilizados em sistemas embarcados e dispositivos portáveis, uma vez que apresentam baixo consumo de energia e baixo custo final.

Os SoCs são compostos por diversos componentes, como processadores, DSPs, módulos de memória e dispositivos de entrada e saída. Esses componentes necessitam de meios para se comunicarem, de forma a trocarem dados.

A comunicação entre esses componentes é comumente realizada de duas formas: através de canais ponto-a-ponto dedicados ou utilizando canais multiponto, denominados barramentos. A principal vantagem dos canais ponto-a-ponto é que eles proporcionam melhor desempenho e menor latência, pois a comunicação entre dois núcleos ocorre independentemente dos demais, o que possibilita comunicações paralelas. Porém, tal arquitetura de comunicação possui custo de projeto maior, uma vez que necessita de um projeto específico.

Já o uso de barramentos provê um canal multiponto compartilhado entre todos os núcleos do sistema. A grande vantagem de utilizar arquitetura multiponto está relacionada ao custo e ao tempo de projeto, pois tal abordagem garante reusabilidade ao sistema. Alternativas como o uso de múltiplos barramentos e hierarquia de barramentos também são utilizadas, porém acabam sendo limitadas por apresentar baixa escalablidade [3].

O aumento na quantidade de transistores em uma mesma pastilha de silício possibilita a integração de mais componentes em um mesmo sistema em chip, como múltiplos núcleos de processamento, que permitem a execução de diferentes aplicações simultaneamente. Nesse sentido, o problema consistirá na comunicação entre esses componentes, uma vez que os SoCs serão tão complexos em um futuro próximo, a ponto de deixar canais ponto-a-ponto inviáveis, devido a complexidade de projeto, e canais multiponto com desempenho muito reduzido [4][5].

Logo, novas formas de comunicação entre esses núcleos devem ser encontradas de forma a proporcionar performance eficiente a esses sistemas. Nesse âmbito, muitas pesquisas têm sido realizadas visando trazer conceitos de redes de interconexão de multiprocessadores [6] para tratar a comunicação em chip. São as chamadas redes em chip (NoCs –

Networks on Chip) [7][8], que buscam equilibrar o desempenho de arquiteturas ponto-a-ponto com a reusabilidade de canais multiponto.

Tais redes têm como elementos principais enlaces (*links*), canais por onde trafegam os dados, e chaves roteadoras (chaves *crossbar* ou *crossbar switches*), responsáveis pelo redirecionamento dos dados através da rede para chegar a determinado destino. A forma como as chaves de uma rede de interconexão são interligadas é indicada por sua topologia.

De acordo com sua topologia, uma rede de interconexão pode ser classificada em rede direta ou rede indireta. Redes diretas, ou estáticas, mantêm conexões fixas (ponto-a-ponto) entre as unidades de processamento, que são diretamente conectadas por meio de rotadores. Já nas redes indiretas, ou dinâmicas, as conexões passam por chaves roteadoras que indicam para qual caminho o dado será enviado. Enquanto nas redes diretas um canal conecta diretamente dois roteadores que estão associados a um núcleo, nas redes indiretas podem existir canais que interligam estágios de roteadores, sem que estes estejam diretamente ligados a um núcleo. O par núcleo/roteador é comumente denominado nodo.

A escolha da topologia de uma rede é realizada buscando atingir determinadas metas baseadas em alguns parâmetros. Em redes em chip, os principais são: desempenho (latência e vazão), consumo de energia, escalabilidade e custo de implementação [9].

A maioria das propostas de implementação de arquiteturas de comunicação em sistemas em chip é de redes diretas com uma topologia ortogonal [9], na qual é possível organizar os nodos em um espaço ortogonal de dimensão *n* de forma que cada link produza um deslocamento em uma única dimensão.

2. Trabalhos Relacionados

A necessidade de novas arquiteturas de comunicação intra-chip vem abrindo alternativas para a pesquisa de novas formas de comunicação para tais sistemas. A abordagem mais aceita na literatura atualmente é a utilização de conceitos de redes de interconexão de multiprocessadores para realizar a comunicação entre os componentes do sistema. Apesar de utilizar boa parte da teoria de interconexão de computadores paralelos, as redes em chip têm suas características próprias, o que as distingue das anteriores.

Entre essas diferenças, a principal está no fato de que existe certa limitação de área nos chips, exigindo que as chaves roteadoras das NoCs sejam pequenas e rápidas, a ponto de não causar atrasos na comunicação entre os núcleos. Além disso, normalmente sistemas em chip têm restrições quanto ao consumo de energia. Assim, os algoritmos de roteamento e lógica de controle e arbitragem devem manter uma relação entre simplicidade e eficiência nessa nova abordagem de comunicação.

Algumas redes em chip têm sido propostas ultimamente. A rede Octagon [10] apresenta topologia em anel, possuindo vazão mais alta que sistemas baseados em barramentos, custo mais baixo que redes *crossbar* e alta escalabilidade. A Octagon pode operar tanto em modo *circuit* quanto em *packet switching* com tamanhos de pacotes variáveis. A rede ainda pode ser adaptada de acordo com a necessidade do sistema, seja para menor custo ou para alto desempenho.

A rede SPIN (*Scalable, Programmable, Integrated Network*) [3] utiliza uma topologia baseada em árvore. Na SPIN, que utiliza roteamento *wormhole*, os pacotes são enviados como uma sequência de *flits* de 4 bytes. Os pacotes não possuem limite de tamanho. Esta rede possui roteamento em tempo de execução, sendo mais genérica e suportando tanto arquiteturas *data-flow* quanto *control-flow*.

Outros projetos de redes utilizam topologia grelha 2D [11][12][13]. O uso de tal topologia tem origem na facilidade de implementação, uma vez que as tecnologias de implementação de circuitos integrados são planas. Além disso, essa topologia oferece simplicidade no roteamento.

Como base para este trabalho, foram utilizados o roteador e os protocolos de comunicação da infraestrutura Hermes [12], uma vez que esta se encontra em um estágio avançado de implementação e otimização.

3. Hermes

Hermes é uma infraestrutura para a implementação de NoCs com topologia grelha. Consiste de roteadores com cinco portas bidirecionais cada, sendo quatro para a comunicação externa com outros roteadores e uma porta para a comunicação local, com o núcleo ao qual o roteador está associado.

Os canais correspondem às direções Norte, Sul, Leste e Oeste, para as quais os pacotes podem ser enviados, como mostra a Figura 1. Os canais são divididos em módulos de entrada e saída, permitindo o recebimento e envio de dados por um mesmo canal. Internamente, as chaves são estruturadas com *buffers* FIFO (*First In,First Out*) nas portas de entrada.

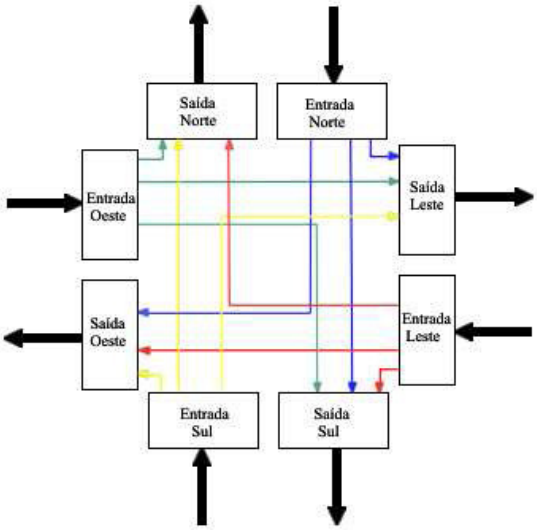

Figura 1. Canais de comunicação externa nos roteadores Hermes

Como estratégia de *Switching*, fator que determina como uma mensagem atravessa a rede, a Hermes utiliza *Packet Switching*. As mensagens são divididas em pacotes que são roteados individualmente da fonte até o destino [14].

Os pacotes são transmitidos por meio do método *Wormhole* [15], não sendo necessário armazenar pacotes inteiros em um roteador caso ele não possa ser imediatamente roteado. Nesta abordagem os pacotes são divididos em *flits*, sendo o cabeçalho enviado antes do restante do pacote. O *flit*, ou unidade de controle de fluxo (*flow control unit*), é a menor unidade de informação que pode ser transferida através de um enlace, sendo que o cabeçalho *flit* de um pacote contém o seu destino [16].

Ao chegar em uma chave roteadora, o cabeçalho *flit* é enviado para sua rota especificada e todos os *flits* restantes do pacote seguem-no. Para que isso possa ocorrer três condições devem ser satisfeitas: o *link* até o próximo nodo deve estar disponível, deve haver espaço para o cabeçalho *flit* no *buffer* do próximo roteador e um *flit* de largura de banda do canal deve estar disponível. Caso o cabeçalho chegue a um nodo e uma dessas condições não seja satisfeita, ele é armazenado em um *buffer* e fica bloqueado. Dessa forma o resto do pacote para de avançar e fica retido na rede, causando atraso na comunicação, mas evitando que o pacote seja perdido.

Em relação ao mecanismo de controle de fluxo, que determina quando uma mensagem ou pacote segue sua rota, a Hermes apresenta duas implementações diferentes.

O mecanismo de controle de fluxo *Ack/Nack* [17], ou *Handshake*, não faz controle do espaço utilizado no *buffer* do nodo receptor. Assim que o dado está disponível para envio, o nodo transmissor envia uma requisição para enviar seus dados. Caso exista espaço no roteador receptor, este envia um sinal de confirmação ao transmissor (*ack*), que transmite seus dados. Porém, se não houver espaço no nodo receptor, ele enviará um sinal informando que o dado não poderá ser recebido (*nack*). O transmissor continua permanentemente enviando sinais de solicitação de envio até receber um sinal *ack* como resposta.

Já o mecanismo de controle de fluxo baseado em créditos [18] permite a transmissão de dados apenas se a rede possuir recursos suficientes (espaço em *buffer* e capacidade nos canais) para que a comunicação ocorra. Tal abordagem funciona sobre cada canal que conecta dois nodos, um nodo transmissor e outro receptor. Cada nodo deve possuir *buffers* separados para cada conexão. Assim, a transmissão é controlada por créditos gerados pelo receptor, que previnem seu *buffer* de receber mais dados do que o suportado. Antes de enviar dados pelo canal, o transmissor deve ser habilitado pelo receptor, que envia os créditos que podem ser utilizados. Esses créditos refletem a capacidade de seu *buffer*. Cada vez que dados são enviados pelo transmissor, os créditos são reduzidos. Caso os créditos cheguem a zero, dados não poderão ser mais transmitidos até que o nodo receptor utilize os recursos recebidos, libere espaço em seu *buffer* e incremente a contagem dos créditos novamente.

A topologia utilizada na infraestrutura Hermes é a grelha. Nela um núcleo se conecta aos seus núcleos adjacentes em até quatro direções diferentes, disponibilizados em forma de grelha, como apresentado na Figura 2.

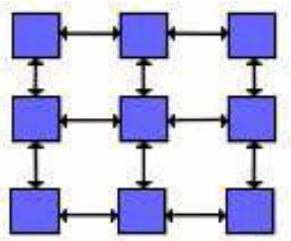

Figura 2. Topologia Grelha

Como algoritmo de roteamento, que indica qual caminho um dado deve seguir baseado no endereço de destino, é utilizado o algoritmo XY. Neste algoritmo, um pacote é roteado pelo eixo X até atingir o mesmo destino do pacote destino neste eixo. A partir de então

o pacote passa a ser roteado pelo eixo Y, até chegar ao seu destino.

4. Testes e Resultados

No presente trabalho foi utilizada a infraestrutura Hermes, e diferentes topologias foram implementadas em busca de uma análise da influência da escolha da topologia em uma NoC.

Os testes realizados utilizaram o mecanismo *Handshake*, por ser de implementação mais simples e, consequentemente, de mais fácil adaptação às outras topologias estudadas.

Esses testes visaram comparar o desempenho de diferentes topologias utilizando configurações distintas. Para isso, a infraestrutura Hermes foi adaptada para suportar, além da topologia grelha, as topologias: malha (também denominada toro), manhattan street e hypercubo. A Figura 3 apresenta tais topologias, na qual a ausência de setas indica canais bidirecionais.

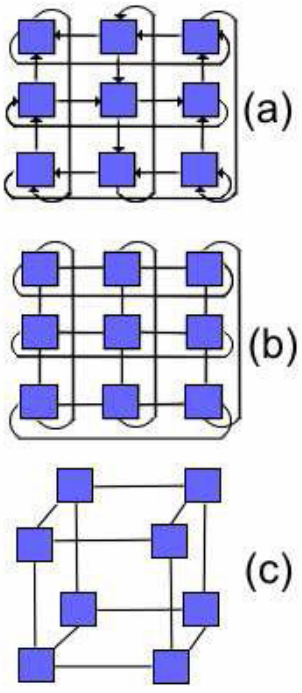

Figura 3. Topologias Manhattan Street (a), Malha (b) e Hypercubo (c)

A rede manhattan street (*MS*) é semelhante à topologia grelha. Porém, entre suas diferenças é importante ressaltar que os núcleos nas extremidades são interligados e cada roteador possui dois *links* de entrada e dois de saída, sendo que canais de entrada e saída em um mesmo eixo devem ser conectados em diferentes direções do roteador. Já a rede malha utiliza portas bidirecionais em todas as direções do roteador, incluindo ainda conexões nas extremidades. Por fim, as redes hypercubo apresentam núcleos conectados de maneira que sua interconexão possui a forma de um cubo. A Figura 3c apresenta uma rede hypercubo de grau 3.

A geração da rede grelha ocorreu com o uso do *framework* MAIA [19], que gera o código RTL de uma NoC utilizando a infraestrutura Hermes. Dados como as dimensões da rede, profundidade dos *buffers* e tamanho dos *flits* são passados como parâmetro e a rede é gerada na linguagem VHDL.

Com base no código gerado, foram feitas adaptações na estrutura da rede e nos algoritmos de roteamento, que tiveram que ser adaptados de acordo com cada topologia a ser analisada.

Na topologia manhattan street, um pacote só pode seguir uma direção por eixo. Assim, o algoritmo de roteamento simplesmente consiste em, primeiramente, enviar o pacote pelo eixo X até encontrar o eixo Y do nodo destino, para então enviar o pacote pelo eixo Y. Por se tratar de uma rede unidirecional, o algoritmo da manhattan street não precisa tomar decisões sobre qual direção um pacote deve ser enviado, já que há apenas uma rota a seguir por eixo. Entretanto, ele deve atentar para o fato de as portas de saída dos nodos de cada eixo apontarem para direções intercaladas. Dessa forma, tomando como exemplo o eixo X, se o roteador estiver em um eixo par, o pacote deve ser encaminhado para a porta oeste. Mas se estiver um eixo ímpar, o algoritmo deve rotear o pacote para a porta leste. O mesmo acontece com o eixo Y e as direções norte e sul.

Para a rede malha, o algoritmo XY foi adaptado considerando as ligações nos nodos situados nas extremidades. Assim, o algoritmo verifica a menor distância entre o nodo atual e o nodo destino no eixo especificado, partindo do princípio de comparar a quantidade de roteadores entre o roteador local e o roteador alvo. Se entre eles existirem mais da metade dos roteadores do mesmo eixo, significa que o caminho mais curto é utilizando os *links* das extremidades.

Já na rede hypercubo, a variação do algoritmo considera um eixo Z adicional. Assim, o algoritmo faz o roteamento inicialmente pelo eixo X até encontrar o roteador com o mesmo endereço X do roteador destino. Então o pacote passa a ser roteado pelo eixo Y. Ao encontrar o mesmo endereço Y do roteador destino o pacote passa a ser roteado pelo eixo Z.

Para possibilitar a implementação deste algoritmo foi necessária também a alteração no modo de

endereçamento de cada roteador, uma vez que além dos eixos X e Y, é necessário armazenar o endereço do roteador no eixo Z.

Para as simulações, foram analisadas a latência e a vazão médias, dois importantes parâmetros de desempenho de uma rede de interconexão. A latência indica o tempo gasto para um pacote chegar ao seu destino, iniciando do momento que o cabeçalho entra na rede e terminando assim que o último *flit* do pacote chega a seu destino. Nos testes realizados a latência foi medida em ciclos de *clock*.

Já a vazão trata da quantidade de dados transmitidos durante determinado intervalo de tempo. Nas simulações realizadas a vazão é indicada pela taxa média de *flits* enviados durante o tempo de simulação.

Para a realização dos testes foram gerados tráfegos com 100 e 1000 pacotes de forma a verificar o comportamento das redes com pouco tráfego e com uma maior quantidade de dados inseridos na rede. Para cada um desses valores foram gerados três tráfegos distintos e os resultados aqui apresentados mostram a média dos resultados. Cada pacote era constituído por 16 *flits*. A capacidade de transmissão dos canais também foi alterada em cada simulação, variando entre 10, 50 e 90% da capacidade de transmissão dos canais. Tal capacidade indica o percentual de banda utilizada por cada roteador ao iniciar uma comunicação. Uma taxa de 100% indicaria que os pacotes seriam inseridos na rede em seqüência, sem qualquer intervalo entre eles. Já uma taxa de 50% indica que existe um atraso entre a inserção de dois pacotes, levando o dobro de tempo para inseri-los em relação a uma taxa de 100%.

Para a criação dos tráfegos foi utilizada a ferramenta Traffic Mbps, também presente no *framework* MAIA. A ferramenta permite a criação de tráfegos específicos ou aleatórios, indicando a quantidade de pacotes a serem enviados por cada roteador, o tamanho dos pacotes e a taxa de transmissão dos canais, entre outros fatores. Os tráfegos gerados continham destinos aleatórios, porém os mesmos tráfegos foram utilizados nas simulações de cada tipo de rede.

Alterações foram necessárias para a geração dos pacotes na rede com topologia hypercubo, uma vez que o endereço dos roteadores nessa rede não era o mesmo que nas outras redes. Assim, um mapeamento dos endereços correspondentes foi realizado para que os tráfegos gerados tivessem comportamento semelhante em todos os testes.

As redes implementadas foram baseadas em uma rede grelha 3x4. Assim, todos os testes foram realizados em redes que continham um total de 12 pacotes. Os *buffers* foram configurados para receber 16 *flits* cada.

A análise dos dados foi realizada com uma ferramenta também presente no *framework* MAIA, o Traffic Measure. Essa ferramenta lê os dados conforme eles passam pelos roteadores e fornece um relatório de acordo com esses dados.

O Traffic Measure também teve que ser adaptado para que fosse possível realizar a leitura dos dados que trafegavam na rede. A ferramenta original gera arquivos que lêem somente as portas conectadas pela rede de topologia grelha. As portas que não apresentavam *links* não eram consideradas no momento de medir o tráfego. Assim, adaptações foram realizadas para que as portas utilizadas em cada topologia fizessem a contabilização dos dados que por ali passavam.

A Tabela 1 apresenta os resultados da vazão média dos testes, considerando a injeção de 100 pacotes por roteador.

Tabela 1. Vazão com inserção de 100 pacotes por roteador

Taxa de transmissão/ Topologia	10%	50%	90%
Grelha	1,97	11,57	11,66
MS	1,70	9,35	10,18
Malha	2,01	12,53	12,44
Hypercubo	2,00	12,36	13,13

A vazão média mostra que, mesmo com uma taxa de inserção baixa por roteador, 100 pacotes, algumas topologias se mostraram mais eficientes em relação à vazão. A rede MS, por exemplo, mostrou-se pouco eficiente nesse quesito mesmo com a taxa de transmissão em 10%. Já as outras topologias apresentaram resultados semelhantes com essa taxa.

Com taxa de transmissão de 50%, a rede malha apresentou os melhores resultados, enquanto a rede hypercubo obteve melhor desempenho com a taxa de transmissão em 90%. Vale ressaltar que as redes grelha e malha obtiveram resultados próximos se comparados os resultados delas com taxas de 50 e 90%. Já a rede hypercubo obteve maior aproveitamento com uma alta taxa de transmissão. A Tabela 2 apresenta os resultados das simulações, considerando a inserção de 1.000 pacotes por roteador. Assim como na Tabela 1, o parâmetro aqui analisado é a vazão.

Tabela 2. Vazão com inserção de 1.000 pacotes por roteador

Taxa de transmissão/ Topologia	10%	50%	90%
Grelha	1,96	12,00	12,05
MS	1,71	10,15	10,27
Malha	1,98	12,87	13,14
Hypercubo	1,97	12,72	12,71

Por meio da Tabela 2 é possível notar que, com uma taxa de transmissão de 10%, os valores se mantiveram muito próximos quando comparada a vazão com 100 pacotes por roteador. A rede manhattan street continuou com pior desempenho em relação às demais, mostrando-se não adequada para as configurações de simulação utilizadas. Já as redes malha e hypercubo obtiveram ganhos de desempenho quando comparadas à rede original, com topologia grelha. Com taxa de transmissão de 10% as três topologias apresentaram resultados semelhantes. Porém, com taxas mais altas, a alteração da topologia traz ganhos em relação a vazão geral da rede.

Comparando ambas as tabelas, percebe-se que a alteração da topologia grelha para malha e hypercubo apresentou um ganho de desempenho em relação à vazão. Isso ocorre porque tais topologias apresentam mais *links* e, consequentemente, mais possibilidades de conexão entre dois roteadores. Isso faz com que pacotes, que antes deveriam aguardar para serem enviados, possam utilizar os canais adicionais e melhor aproveitar a estrutura da rede.

Os dados de latência são apresentados na Tabela 3, para simulações com 100 pacotes inseridos por roteador. Os dados são apresentados em ciclos de *clock*.

Tabela 3. Latência média com inserção de 100 pacotes por roteador

	Taxa de transmissão		
Topologia	**10%**	**50%**	**90%**
Grelha	75,02	378,05	1.921,20
MS	75,07	551,79	2.322,14
Malha	68,39	187,49	1.401,11
Hypercubo	70,82	181,92	1.462,21

Como pode ser observado na Tabela 3, a topologia manhattan street também não apresentou bons resultados se comparada com as outras topologias em relação à latência. A utilização de canais unidirecionais acaba se mostrando menos eficiente quando comparada às topologias que utilizam *links* bidirecionais aqui analisadas. Mesmo possuindo estrutura semelhante à topologia malha, a possibilidade de rotear pacotes em ambas as direções proporciona significativa melhora no tempo de entrega dos pacotes.

Já as utilizações das topologias Hypercubo e Malha apresentaram um ganho de desempenho. Com uma taxa de transmissão de 10%, essas topologias obtiveram ganhos de 5,93% e 9,69% em relação à topologia grelha, respectivamente.

Esse ganho é acentuado se utilizadas taxas de transmissão maiores. Com taxa de transmissão de 50%, os ganhos apresentados foram de 101,63% e 107,81% para as redes malha e hypercubo. As maiores diferenças de desempenho ocorreram justamente com taxa de transmissão em 50%, na qual os pacotes foram entregues, em média, com metade dos ciclos de *clock* gastos pela rede com topologia Grelha.

Considerando uma taxa de transmissão de 90%, o ganho de desempenho obtido pelas topologias Malha e Hypercubo foi de 37,12% e 33,85%. Um ganho significativo de desempenho, uma vez que cerca de um terço de tempo é ganho com utilização dessas topologias.

Já a Tabela 4 apresenta os resultados da latência média de entrega de um pacote considerando a inserção de 12.000 pacotes na rede, sendo 1.000 pacotes por roteador.

Tabela 4. Latência média com inserção de 1.000 pacotes por roteador

	Taxa de transmissão		
Topologia	**10%**	**50%**	**90%**
Grelha	76,55	4.860,57	17.465,23
MS	77,87	6.523,22	20.456,87
Malha	69,72	595,51	13.289,84
Hypercubo	72,30	1.331,84	14.328,14

Com a inserção de mais pacotes, o que ocasiona maior tráfego na rede, a diferença no desempenho é acentuada. A principal diferença de desempenho ocorre quando utilizada a taxa de transmissão de 50%. A rede malha proporcionou uma melhora de cerca de 716% no desempenho, fazendo com que a média de tempo de entrega dos pacotes fosse muito menor. Já com a taxa de transmissão em 90% o ganho foi de 31%.

A rede hypercubo também foi capaz de melhorar o desempenho em relação à rede grelha, com ganhos de 265% e 21% quando utilizadas taxas de transmissão de 50% e 90%.

Tais resultados mostram que os *links* adicionais inseridos pelas topologias Malha e Hypercubo proporcionam significativa melhora no tempo de transmissão de um pacote. Isso ocorre porque a adição desses canais faz com que a distância entre quaisquer dois roteadores seja igual ou menor que a distância

desses mesmos roteadores em uma topologia Grelha. Na maioria dos casos a distância é reduzida.

Os testes aqui apresentados mostram a análise de desempenho comparativa entre as quatro topologias, sendo que duas delas são bastante discutidas na literatura (malha e grelha) e as outras duas são pouco exploradas (hypercubo e manhattan street).

O ganho de desempenho ficou evidente quando utilizadas topologias malha e hypercubo, porém um aumento no custo de implementação deve ser considerado ao utilizar uma dessas topologias em uma NoC, uma vez que elas apresentam maiores quantidades de canais e consequentemente mais *buffers* ligados a esses canais.

5. Conclusões

O advento dos processos de fabricação de chips tem possibilitado a integração de milhões de transistores em uma mesma pastilha de silício, possibilitando o agrupamento de todos os componentes de um computador em um único chip e de vários núcleos de processamento para execução paralela de aplicações.

A complexidade de tais sistemas tem crescido tanto, que as arquiteturas de comunicação atuais não suportarão os requisitos dos sistemas em um futuro próximo. Nesse sentido, muitas pesquisas focam redes de interconexão em chip, sistemas de comunicação, cujos elementos básicos são *links* e roteadores.

Nesse sentindo, o presente trabalho procurou investigar a forma como esses roteadores são interligados. Quatro diferentes topologias foram analisadas. As implementações partiram de uma rede grelha e as alterações necessárias foram realizadas em sua estrutura e algoritmo de roteamento para compará-las. Os melhores resultados foram obtidos com redes utilizando as topologias hypercubo e malha, que se alternaram entre os melhores desempenhos. Isso mostra que uma maior variedade de caminhos, encurtando a distância entre dois roteadores e permitindo rotas alternativas podem melhorar significativamente o desempenho de uma NoC.

Os testes também mostraram que a topologia manhattan street, com *links* unidirecionais, perde muito desempenho se comparada às redes bidirecionais. Isso porque, muitas vezes, roteadores que estão fisicamente próximos não podem estabelecer uma comunicação direta e acabam gerando tráfego desnecessário na rede. A topologia não se mostra interessante para propósitos gerais, podendo ser mais eficiente em redes cujo tráfego é bem definido.

A próxima etapa da pesquisa consistirá na variação de alguns parâmetros para verificar o impacto que eles proporcionam no desempenho da rede, como os tamanhos dos *buffers* e sua organização interna, além de um estudo mais detalhado da diferença de custo de implementação das topologias aqui analisadas. Na continuidade da pesquisa pretende-se ainda abordar outros fatores que influenciam o desempenho de uma rede em chip. Caso dos algoritmos de roteamento, com teste em algoritmos adaptativos..

6. Agradecimentos

Os autores agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e à Fundação Araucária pelo apoio.

7. Referências

[1] G. E. Moore, “Cramming More Components onto Integrated Circuits”. *Proceedings of the IEEE*, v. 86, n. 1, p. 82-84, jan. 1998.

[2] K. Mori, Y. Yamada, e S. Takizawa, “System on Chip Age”. *Proceedings of the 1st International Symposium on VLSI Technology, Systems and* Applications. Taipei, Taiwan, 1993. p. k15-k20.

[3] P. Guerrier, e A. Greiner. “A Generic Architecture for on-Chip Packet-Switched Interconnections”. *Proceedings of Conference on Design, Automation and Test in Europe*. Paris, França, 2000. p. 250-256.

[4] C. A. Zeferino. *Redes em-Chip: Arquiteturas e Modelos para Avaliação de Área e Desempenho*. 2003. 242 p. Dissertação (Doutorado em Ciência da Computação) – Universidade Federal do Rio Grande do Sul, Porto Alegre, 2003.

[5] Y. R. Sum, S. Kumar, e A. Jantsch. “Simulation and Evaluation for a Network on Chip”. *Proceedings of NORCHIP Conference*. Copenhagen, Dinamarca, 2002. p. 7-12.

[6] W. J. Dally, e B. Towles. *Principles and Practices of Interconnection Networks*. São Francisco, EUA: Morgan Kaufmann, 2003. 550 p.

[7] W. J. Dally, e B. Towles. Route Packets, Not Wires: On-Chip Interconnection Networks. *Proceedings of Design Automation Conference*. Las Vegas, EUA, 2001. p. 684-689.

[8] L. Benini, e G. De Micheli. "Networks on Chip: A New SoC Paradigm". *Computer*, v. 35, n. 1, p. 70-78, jan. 2002.

[9] G. De Michele e L. Benini. *Networks On Chips*, São Francisco, EUA: Morgan Kaufmann, 2008. 395 p.

[10] F. Karin, A. Nguyen, e S. Dey. "An Interconnect Architecture for Networking Systems on Chip". IEEE Micro, v. 22, n. 5, p. 36-45, set./out. 2002.

[11] C. A. Zeferino, e A. A. Susin. SoCIN: A Parametric and Scalable Network-on-Chip. Proceedings of 16th Symposium on Integrated Circuits and Systems Desing. São Paulo, 2003. p. 169-174.

[12] F. G. Moraes et al. *HERMES: An Infrastructure for Low Area Overhead Packet-Switching Networks On Chip*. Porto Alegre: PUCRS, 2003. 26 p. (Technical Report Series 034).

[13] M. P. Véstias, e H. C. Neto. "A Generic Network-on-Chip Architecture for Reconfigurable Systems: Implementation and Evaluation". *Proceedings of the 16th* International Conference on Field Programmable Logic and Applications. Madrid, Espanha, 2006. p. 1-4.

[14] A. R. Tripathi e G. J. Lipovski. "Switching in Banyan Networks". *Proceedings of the 6th International Symposium in Computer Architecture*, 1979. p. 160-167.

[15] W. J. Dally e C. L. Seitz. "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks". IEEE Transactions on Computers, v. 36, n. 5, p. 547-553, maio 1987.

[16] L. S. Peh e W. J. Dally. "Flit-Reservation Flow Control". Proceedings of the 6th International Symposium on High-Performance Computer Architecture. Toulouse, França, 2000. p. 73-84.

[17] M. Niswar e A. H. Thamrin. "Rate-based Congestion Control Mechanism for Multicast Communication". *Proceedgins of the 4th International Conference on Telecommunications and Informatics*. Praga, República Checa, 2005. artigo nº 14.

[18] S. Yan, G. Min e I. Awan, "Performance Analysis of Credit-Based Flow Control in InfiniBand Interconnection Networks. *Journal of Interconnection Networks*, v. 7, n. 4, p.535-548, 2008.

[19] L. Ost et al. "MAIA – A Framework for Networks on Chip Generation and Verification". *Proceedings of the 2005 Asia and South Pacific Design Automation Conference*, jan. 2007, p. 49-52.