Análise do Tráfego Interno em uma Rede-em-Chip por meio de Simulação

Sérgio Vargas Júnior, Eduardo A. da Silva, Cesar A. Zeferino

Laboratório de Sistemas Embarcados e Distribuídos
Universidade do Vale do Itajaí (UNIVALI) – Itajaí, SC – Brasil

{sergiovargas,eduardoalves}@edu.univali.br, zeferino@univali.br

Abstract. *RedScarf is a simulation environment for performance evaluation of Networks-on-Chip for design space exploration. Despite the range of offered features, RedScarf still lacked the means to analyze the internal traffic of the network, making it difficult to identify congestion points. In this context, this paper aims to characterize the internal traffic of the network and identify its congestion points. For this purpose, we implemented monitors in the network links, extracted its generated information, and represented it in the form of textual maps, which can be used to evaluate the traffic between the routers of the network and to identify the congestion points for different configurations of its project parameters.*

Resumo. *O RedScarf é um ambiente de simulação para avaliação de desempenho de Redes-em-Chip para exploração do seu espaço de projeto. Apesar da série de recursos oferecidos, o RedScarf ainda carecia de meios para analisar o tráfego interno da rede, dificultando a identificação de pontos de congestionamento. Nesse contexto, este trabalho busca caracterizar o tráfego interno da rede e identificar seus pontos de congestionamento. Para isso, foram implementados monitores nos enlaces da rede, suas informações geradas foram extraídas e representadas em forma de mapas textuais, que podem ser utilizados para avaliar o tráfego entre os roteadores da rede e identificar os pontos de congestionamento para diferentes configurações de seus parâmetros de projeto.*

1. Introdução

O avanço das tecnologias de fabricação de circuitos eletrônicos viabilizou a integração de dezenas de elementos de processamento em um único chip de silício para a construção de sistemas computacionais integrados, os quais são denominados SoCs (Systems-on-Chip). Para atender as demandas de comunicação de tais sistemas, pesquisadores propuseram, no início dos anos 2000, a adoção de redes de interconexões chaveadas para suportar a comunicação entre os componentes desses sistemas [Guerrier e Greiner 2000; Hemani et al. 2000]. Essas redes, quando implementadas no nível intrachip são denominadas Redes-em-Chip, do inglês NoCs – Networks-on-Chip [Hemani et al. 2000]. As NoCs provêm, dentre outras características, escalabilidade, reusabilidade e paralelismo de comunicação [Guerrier e Greiner 2000; Benini e De Micheli 2002; Jerger e Peh 2009].

O espaço de projeto das NoCs é bastante amplo e vários aspectos arquiteturais precisam ser considerados para atender os requisitos de desempenho e custo da aplicação alvo. Para facilitar e acelerar a exploração desse espaço de projeto, são necessários ambientes e ferramentas especializadas, conforme relatam Benini e De Micheli (2002).

Esses autores também afirmam que ferramentas de análise são importantes para avaliar o desempenho de uma rede e buscar um melhor *tradeoff* por meio do refinamento de parâmetros.

Para facilitar a exploração do espaço de projeto de NoCs, pesquisadores da Univali desenvolveram uma ferramenta de avaliação de desempenho de NoCs denominada RedScarf [Silva et al. 2017]. Essa ferramenta baseia-se em modelos de simulação SystemC [Accelera 2015] de uma NoC e oferece ferramentas e uma interface gráfica que facilitam a configuração, execução e análise de experimentos de forma automática. A primeira versão (1.0) do RedScarf apresenta as seguintes características principais:

- Multiplataforma (Windows, Linux e OS X);
- Interface multilíngue (em português e inglês);
- Execução multi-threading;
- Modelo SystemC parametrizável de NoC;
- Visualização de resultados dos experimentos por meio de gráficos e tabelas; e
- Armazenamento das configurações e dos resultados dos experimentos.

O RedScarf utiliza um modelo de simulação de uma NoC parametrizável descrito em SystemC que oferece as seguintes alternativas arquiteturais (às quais podem ser facilmente adicionadas novas alternativas):

- *Roteamento*: determinístico ou parcialmente adaptativo;
- *Arbitragem*: prioridades fixas ou variáveis;
- *Controle de fluxo*: handshake ou baseado em créditos;
- *Memorização*: FIFOs de profundidade parametrizável nos canais de entrada com opção de FIFOs nos canais de saída;
- *Chaveamento*: por pacotes (do tipo Wormhole) e por circuitos baseados em pacotes; e
- *Canais virtuais*: com ou sem canais virtuais (dois canais).

Um usuário pode definir diferentes configurações da rede e executar experimentos baseados na simulação da aplicação de um determinado padrão espacial de tráfego sobre a rede executando em diferentes frequências de operação (cada frequência resulta em uma taxa de injeção ou tráfego oferecido diferente). Por exemplo, pode-se comparar diferentes topologias, algoritmos de roteamento e técnicas de arbitragem, entre outros atributos arquiteturais, como demonstrado em [Silva e Zeferino 2017].

Após a execução dos experimentos, o RedScarf oferece ferramentas para análise dos resultados que permitem comparar as diferentes configurações de rede usando diferentes métricas, como por exemplo:

- *Latência* x *tráfego oferecido*: variação da latência média em relação à frequência de operação;
- *Tráfego aceito* x *tráfego oferecido*: variação da vazão em relação à frequência de operação;
- *Cumprimento de prazos* x *tráfego oferecido*: taxa de atendimento de prazos de entrega dos pacotes em relação à frequência de operação; e
- *Distribuição da latência*: histograma das latências dos pacotes transferidos.

Apesar do conjunto de facilidades oferecidas pela versão 1.0 do RedScarf, ela apresenta limitações que devem ser resolvidas para permitir uma análise mais detalhada do desempenho da rede. Uma dessas limitações consiste na falta de recursos para medir e analisar o tráfego interno da rede de modo a identificar pontos de congestionamento. Tais recursos facilitariam o entendimento mais aprofundado da razão de uma configuração de rede ter desempenho melhor que outra.

Por exemplo, a rede utilizada no RedScarf pode ser configurada para operar com algoritmos de roteamento determinístico (ex. XY) ou parcialmente adaptativo (ex. WF – West-First). No primeiro, cada pacote segue sempre um determinado caminho. No segundo, conforme a direção percorrida, um pacote pode utilizar um caminho alternativo, caso o caminho prioritário não esteja disponível quando necessário. Com base nessas definições, em um primeiro momento, pode-se supor que o roteamento parcialmente adaptativo resulta em um melhor desempenho pelo fato dos pacotes poderem utilizar caminhos alternativos. Porém, não é o que se observa.

A Figura 1 apresenta o tráfego aceito por uma rede em malha 2D 4x4 operando sob um padrão de tráfego com distribuição espacial uniforme em que todos os nodos injetam pacotes de 128 bits a 320 Mbps. A frequência de operação da rede é variada de 10 a 100 MHz, em passos de 10 MHz, para variar a capacidade do canal de 320 Mbps a 3200 Mbps e, portanto, a demanda de uso do canal (de 100% a 10%, respectivamente). O gráfico mostra que o roteamento XY aceita um maior tráfego que o roteamento WF. Enquanto o primeiro aceita até 33% da demanda oferecida, o segundo aceita no máximo 23%. Já a Figura 2 ilustra a distribuição das latências dos pacotes transferidos quando a rede está operando a 40 MHz. Nota-se que o roteamento WF apresenta uma grande dispersão nos valores das latências dos pacotes em virtude do fato de a rede estar saturada. Já o roteamento XY apresenta uma menor dispersão, pois a rede não está saturada para essa frequência de operação. Com relação à latência média abaixo do ponto de saturação, o roteamento XY resultou em uma latência de 19 ciclos, enquanto o roteamento WF apresentou uma latência média maior (de 19,8 ciclos) quando a rede está operando a 100 MHz.

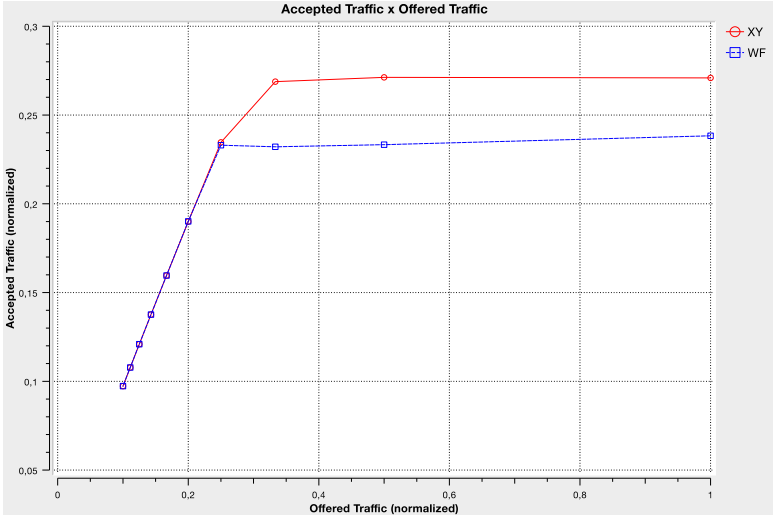

Figura 1. Comparativo dos roteamentos determinístico (XY) e parcialmente adaptativo (WF): tráfego aceito normalizado x tráfego oferecido normalizado

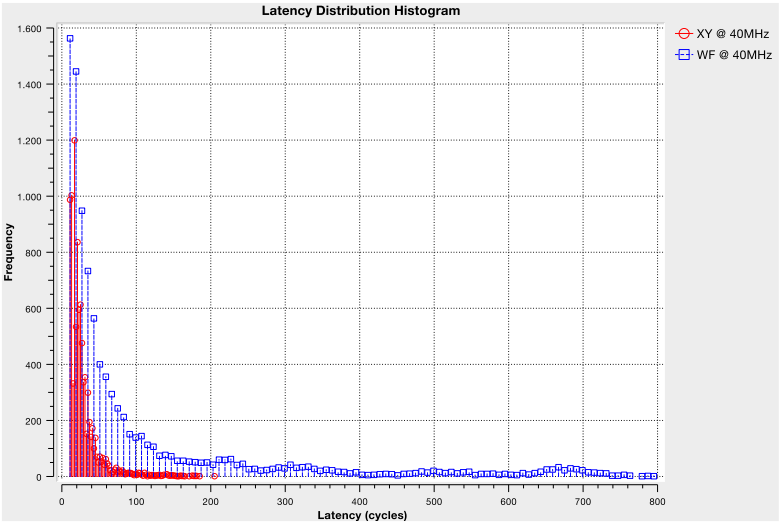

Figura 2. Comparativo dos roteamentos determinístico (XY) e parcialmente adaptativo (WF): histograma das latências

Ou seja, os resultados experimentais contrariam o entendimento inicial de que o roteamento parcialmente adaptative teria um desempenho superior ao determinístico. Pela análise dos resultados e do comportamento dos algoritmos de roteamento, acredita-se que o menor desempenho do roteamento parcialmente adaptativo deve-se ao fato de que o desvio realizado por alguns pacotes leve a uma concentração de tráfego em determinados pontos da rede levando esses e outros pacotes a sofrerem uma latência maior do que no roteamento determinístico. Entretanto, para essa hipótese ser confirmada, é necessário que o RedScarf ofereça recursos para identificar a concorrência entre os fluxos de comunicação e os pontos de congestionamento da rede.

Para resolver o problema supracitado, neste trabalho, foram desenvolvidos monitores de tráfego conectados nos enlaces internos da rede para registrar o tráfego dos pacotes nesses enlaces, e foram implementados métodos de leitura no RedScarf para o processamento das informações geradas pelos monitores.

O restante deste artigo é organizado da seguinte forma. A Seção 2 discute os trabalhos relacionados. A Seção 3 apresenta a solução desenvolvida para tratar o problema supracitado. Continuando, a Seção 4 apresenta e discute os resultados obtidos. Concluindo, a Seção 5 apresenta as considerações finais.

2. Trabalhos Relacionados

Achballah e Saoud (2013) realizaram uma busca não exaustiva sobre ferramentas para auxiliar na exploração do espaço de projeto de NoCs. Esse estudo foi complementado por uma pesquisa bibliográfica realizada por Silva e Zeferino (2015). Dentre as ferramentas analisadas por esses autores, foi observado que a maioria delas realiza a configuração dos atributos da NoC (topologia, arquitetura do roteador, árbitros, dimensionamento de *buffers*) e é capaz de criar modelos de tráfegos sintéticos ou realizar a alocação de tarefas para estabelecer fluxos de comunicação. Em todos os trabalhos analisados, o desempenho da rede é medido com base em métricas de vazão e latência calculadas a partir do monitoramento do tráfego externo da rede (entre os elementos de processamento e a rede).

Na literatura, apenas alguns trabalhos reportaram a realização do tráfego interno (foco deste projeto), incluindo as seguintes ferramentas: NoCScope [Möller, Indrusiak e

Glesner 2009], a GSNOC UI [Gottschling, Ying e Hofmann 2012] e o Trace Monitor [Alhonen et al. 2012]. A ferramenta NoCScope utiliza monitores nos roteadores para adquirir informações das portas e buffers dos roteadores, e as exibe em sua interface gráfica. A GSNOC UI permite a visualização do uso dos enlaces e dos buffers durante e após a simulação. Finalmente, o Trace Monitor implementa um monitor que coleta os dados de todos os enlaces de NoC em FPGA, e transmite os dados para um computador para ser exibido.

3. Monitor de Tráfego Interno

O monitor de tráfego interno proposto neste trabalho é um módulo de coleta de informações sobre pacotes transferidos pelos enlaces que interligam os roteadores da rede. Cada um desses enlaces é constituído de dois canais unidirecionais em oposição e, para cada canal é instanciado um monitor. A Figura 3 ilustra um exemplo de NoC com topologia em Malha 2-D 2x2 com um uma instância de monitor conectada a um canal.

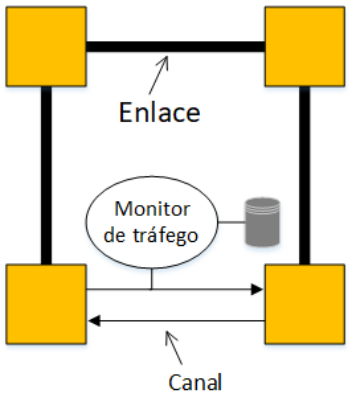

Figura 3. Arquitetura da NoC com o monitor de tráfego interno em um canal

O monitor proposto foi implementado em SystemC e integrado ao modelo de simulação da rede do RedScarf. Também foram implementados os procedimentos de leitura e processamento dessa informação no RedScarf.

As métricas extraídas de cada enlace foram: (i) largura de banda média ou abw (average bandwidth); (ii) vazão ou thr (throughput), que representa a taxa efetiva de transmissão de dados; e (iii) quantidade média de ciclos por flit ou avcpf (average cycles per flit) – sendo que um flit é a unidade de transferência de dados sobre a qual é realizado o controle de fluxo. As métricas abw e thr são aplicadas para avaliar a utilização média dos canais, enquanto o avcpf permite avaliar o tempo médio para transmissão de flits [Tedesco 2005].

O método de visualização das métricas foi por mapas textuais, conforme utilizado por Tedesco (2005). Os mapas textuais representam os nodos da rede como números entre parênteses, a medição do tráfego nos canais de saída do nodo são exibidos ao seu redor, e os valores próximos a esses são dos canais de entrada. Com esses mapas textuais é possível esclarecer o exemplo citado anteriormente.

A Figura 4 ilustra como os dados coletados são apresentados e como podem ser lidos em uma análise. A figura mostra uma NoC com topologia em Malha 2-D 2x2 em que cada roteador é identificado por um número de três dígitos entre parênteses – ex.

(001). Ao redor do nodo, são apresentadas as taxas de ocupação dos seus canais de saída. Já a taxa de ocupação de um canal de entrada é determinada pela taxa de ocupação do canal de saída do roteador vizinho. Por exemplo, para a porta leste do roteador (000), o canal de saída possui uma taxa de ocupação igual a 0.66 (ou seja, 66% da sua capacidade), enquanto o seu canal de entrada possui uma taxa de 0.34 (este é o canal de saída da porta oeste do roteador (001)). No exemplo, o mapa evidencia os pontos de maior ocupação na rede, bem como os que canais menos utilizados para o cenário de tráfego considerado.

```
(002)  0.61  0.35  (003)
0.26                0.29

0.32                0.36
(000)  0.66  0.34  (001)
```

Figura 4. Exemplo de mapa textual simplificado

4. Resultados

4.1 Largura de banda média

A Figura 5 permite verificar que, em uma rede com topologia em Malha 2-D 4x4, no roteamento XY com tráfego uniforme, os canais horizontais (valores destacados com sublinha) possuem uma taxa de ocupação maior do que os verticais.

```
Average Bandwidth
(012) 0.65  0.33 (013) 0.54  0.60 (014) 0.32  0.67 (015)
0.34             0.38             0.34             0.31

0.21             0.21             0.21             0.21
(008) 0.58  0.38 (009) 0.55  0.61 (010) 0.34  0.63 (011)
0.34             0.36             0.35             0.32

0.32             0.35             0.33             0.32
(004) 0.61  0.35 (005) 0.55  0.60 (006) 0.35  0.63 (007)
0.26             0.29             0.24             0.22

0.32             0.36             0.34             0.31
(000) 0.66  0.34 (001) 0.54  0.60 (002) 0.32  0.68 (003)
```

Figura 5. Taxa de ocupação dos canais com o roteamento determinístico (XY)

Na Figura 6, para o roteamento WF (West-First), observa-se uma concentração de uso nos canais direcionados ao oeste, assim como nos canais verticais da coluna mais à esquerda. Isso ocorre devido à característica do roteamento de priorizar o encaminhamento dos pacotes para o oeste.

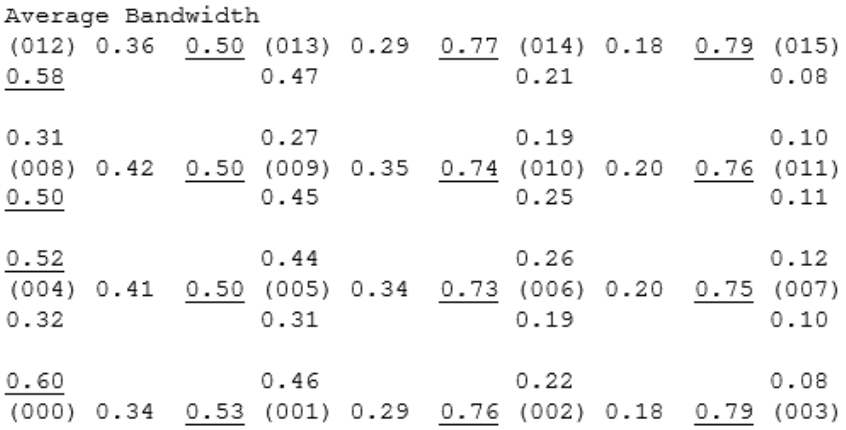

Figura 6. Largura de banda média do roteamento parcialmente adaptativo (WF)

4.2 Vazão

Na Figura 7, para o roteamento XY, é possível observar um tráfego maior nos enlaces das bisecções horizontal e vertical, o que não ocorre no roteamento WF, ilustrado na Figura 8. Neste, é observado uma uma grande concentração de tráfego nos canais verticais na região oeste da rede.

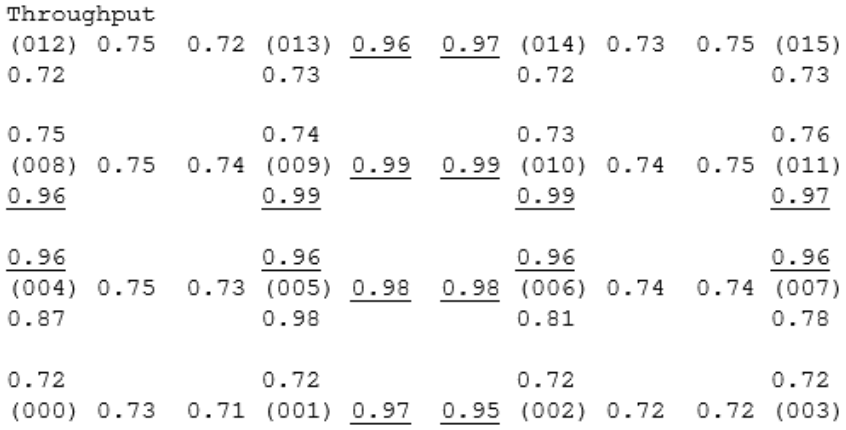

Figura 7. Vazão do roteamento determinístico (XY)

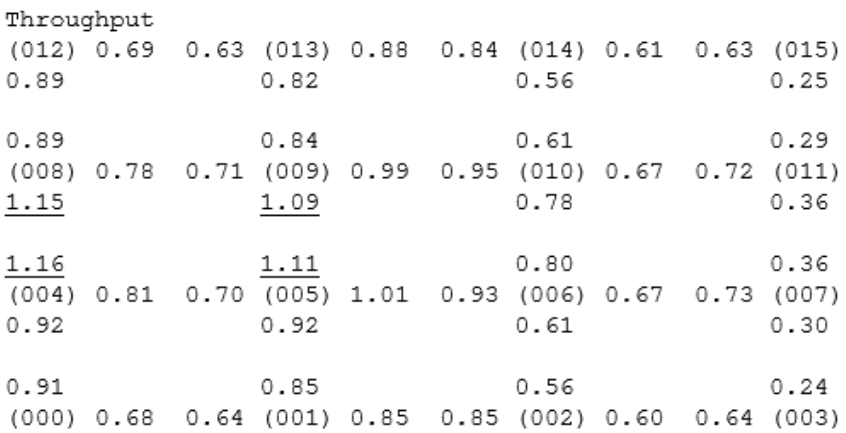

Figura 8. Vazão do roteamento parcialmente adaptativo (WF)

4.3 Ciclos por flit

Na Figura 9, para o algoritmo XY, percebe-se uma contenção maior nos canais horizontais das extremidades em direção ao centro, ou seja, mais ciclos são gastos para transmitir um flit nessas regiões. Já na Figura 10, para o algoritmo WF, as maiores contenções são observadas nos canais com direção ao oeste, mais especificamente na extremidade direita.

```
Average Cycles per Flit
(012) 3.51  1.83 (013) 2.26  2.46 (014) 1.73  3.60 (015)
1.88             2.07             1.87             1.71

1.12             1.17             1.14             1.11
(008) 3.09  2.07 (009) 2.23  2.45 (010) 1.84  3.36 (011)
1.43             1.46             1.40             1.32

1.35             1.45             1.37             1.33
(004) 3.22  1.93 (005) 2.27  2.44 (006) 1.91  3.41 (007)
1.20             1.19             1.18             1.14

1.76             2.01             1.86             1.74
(000) 3.62  1.89 (001) 2.25  2.53 (002) 1.76  3.76 (003)
```

Figura 9. Ciclos por flit médio do roteamento determinístico (XY)

```
Average Cycles per Flit
(012) 2.06  3.19 (013) 1.33  3.68 (014) 1.18  5.04 (015)
2.63             2.29             1.48             1.26

1.42             1.30             1.24             1.33
(008) 2.16  2.82 (009) 1.41  3.11 (010) 1.21  4.24 (011)
1.75             1.65             1.30             1.28

1.80             1.59             1.30             1.31
(004) 2.02  2.87 (005) 1.36  3.15 (006) 1.19  4.14 (007)
1.40             1.35             1.27             1.30

2.65             2.18             1.53             1.30
(000) 2.02  3.30 (001) 1.38  3.58 (002) 1.19  4.91 (003)
```

Figura 10. Ciclos por flit médio do roteamento parcialmente adaptativo (WF)

4.4 Análise

Para analisar a rede como um todo, foram calculados a média e o desvio padrão dessas métricas para todos os canais, listados na Tabela 1. É observado que o roteamento determinístico resulta em uma ocupação maior e mais uniforme da banda dos canais do que o parcialmente adaptativo, assim como a vazão. Em média, a contenção dos dois roteamentos é similar, porém o determinístico possui uma variação menor.

Tabela 1 – Comparativo geral dos roteamentos XY e WF

Métricas		Roteamentos	
		Determinístico (XY)	Parcialmente Adaptativo (WF)
Largura de banda	**Média**	0,41	0,39
	Desvio padrão	0,15	0,21
Vazão	**Média**	0,82	0,74
	Desvio padrão	0,11	0,22
Ciclos por *flit*	**Média**	2,01	2,09
	Desvio padrão	0,76	1,04

Logo, essas informações confirmam a hipótese que o algoritmo de roteamento parcialmente adaptativo concentra o tráfego em certos pontos da rede levando a um desempenho inferior no cenário uniforme, enquanto o roteamento determinístico consegue distribuir melhor a carga pela rede.

5. Conclusões

Este trabalho teve como objetivo geral identificar os pontos de congestionamento de uma NoC. Para isso, foi implementado um monitor de tráfego nos enlaces da rede, assim como procedimentos de leitura dos dados obtidos por esses monitores e exibição em formato de mapa textual.

As métricas obtidas permitem identificar o nível de ocupação dos enlaces, mas ainda não permitem identificar a ocupação dos buffers dos roteadores. Também não é possível identificar os caminhos percorridos pelos fluxos de comunicação, assim como a visualização gráfica das informações para aprimorar a análise interna.

Foi observada neste trabalho a importância dos dados do tráfego interno da NoC para a análise do comportamento e do desempenho de algoritmos de roteamento, principalmente em situações como o cenário abordado, em que os resultados contradizem o entendimento inicial.

Como trabalhos futuros propõe-se a implementação de um monitor nos buffers dos roteadores, uma modificação do simulador para identificar os fluxos de comunicação de cada pacote, e a implementação de uma ferramenta no RedScarf que represente graficamente a informação dos mapas textuais produzidos a partir deste projeto.

6. Referências

Accellera. (2015) “SystemC”, http://www.accellera.org/downloads/standards/systemc. Acesso em: 15 jun. 2015.

Achballah, A. B. and Saoud, S. B. (2013). A survey of network-on-chip tools. *arXiv preprint arXiv:1312.2976*.

Alhonen, A., Salminen, E., Nieminen, J., and Hämäläinen, T. D. (2010). A scalable, non-interfering, synthesizable network-on-chip monitor. In *NORCHIP, 2010*, p. 1–6. IEEE.

Benini, L. and De Micheli, G. (2002). Networks on chips: A new soc paradigm. *computer*, 35(1):70–78.

Gottschling, P., Ying, H., and Hofmann, K. (2012). Gsnoc ui—a comfortable graphical user interface for advanced design and evaluation of 3-dimensional scalable networkson-chip. In *International Conference on High Performance Computing and Simulation (HPCS), 2012*, p. 261–267. IEEE.

Guerrier, P. and Greiner, A. (2000). A generic architecture for on-chip packet-switched interconnections. In *Proceedings of the conference on Design, automation and test in Europe*, p. 250–256. ACM.

Hemani, A., Jantsch, A., Kumar, S., Postula, A., Oberg, J., Millberg, M., and Lindqvist, D. (2000). Network on chip: An architecture for billion transistor era. In *Proceeding of the 31st IEEE NorChip Conference*, p. 1-8.

Jerger, N. E. and Peh, L.-S. (2009). On-chip networks. *Synthesis Lectures on Computer Architecture*, 4(1):1–141.

Moller, L., Indrusiak, L. S., and Glesner, M. (2009). Nocscope: A graphical interface to improve networks-on-chip monitoring and design space exploration. In *Design and Test Workshop (IDT), 2009 4th International*, p. 1–6. IEEE.

Silva, E. A., Menegasso, D., Vargas Júnior, S., Zeferino, C. A. (2017). RedScarf: A User-Friendly Multi-Platform Network-on-Chip Simulator. In *VII Brazilian Symposium on Computing Systems Engineering (SBESC), 2017*, p. 71-78. SBC.

Silva, E. A., Zeferino, C. A. (2015). Uma Análise sobre Ferramentas de Redes-em-Chip e seus recursos para Uso no Ensino. In *International Journal of Computer Architecture Education (IJCAE), 2015*, v. 4, p. 29-32. SBC.

Silva, E. A., Zeferino, C. A. (2017). Análise arquitetural comparativa do desempenho de Redes-em-Chip baseada em simulação. In *Simpósio de Sistemas Computacionais de Auto Desempenho (WSCAD), 2017*, p. 268-279. SBC.

Tedesco, L. (2005). Uma proposta para geração de tráfego e avaliação de desempenho para nocs. *Pontificia Universidade Catolica Do Rio Grande Do Sul: Porto Alegre*.