PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

IMPLEMENTAÇÃO DE UMA INFRA-ESTRUTURA DE MONITORAMENTO PARA AVALIAÇÃO DE PLATAFORMAS MPSOC BASEADAS EM NOC

SAMUEL DOS SANTOS MARCZAK

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Ciência da Computação na Pontifícia Universidade Católica do Rio Grande do Sul.

Orientador: Prof. Dr. Fernando Gehm Moraes

Porto Alegre, Brasil
2010

FICHA CATALOGRÁFICA EMITIDA PELA BIBLIOTECA

TERMO DE APRESENTAÇÃO DE
DISSERTACAO DE MESTRADO
EMITIDA E ASSINADA PELA FACULDADE

IMPLEMENTAÇÃO DE UMA INFRA-ESTRUTURA DE MONITORAMENTO PARA AVALIAÇÃO DE PLATAFORMAS MPSoC BASEADAS EM NoC

RESUMO

MPSoCs (*Multi Processor System on Chips*) são arquiteturas bastante complexas e, por consequência, a verificação do correto funcionamento do sistema bem como a garantia da qualidade de serviço são ações que se tornam cada vez mais difíceis de serem realizadas. Assim sendo, é importante a pesquisa de mecanismos para a verificação da operação do sistema como um todo que visem a captura de informações sobre seu estado a cada instante, obtidas através de monitores adequadamente adicionados à arquitetura. Este trabalho apresenta o desenvolvimento de uma infra-estrutura de monitoramento para MPSoCs baseados em NoC (*Networks on Chip*), sendo realizado através de monitores de tráfego adicionados à NoC. A estrutura de monitoramento é integrada ao *microkernel* do processador que controla o MPSoC. Os resultados demonstram que os monitores não interferem no desempenho global da NoC e que é possível calcular a taxa de recepção de pacotes na rede através das informações coletadas pelos monitores. A integração da estrutura de monitoramento ao MPSoC é validada a partir de matrizes inseridas no *microkernel* do processador de controle, que armazenam os valores de monitoramento dos canais de cada roteador da NoC.

Palavras Chave: redes intra-chip (NoCs); monitores; sistemas de monitoramento; desempenho; MPSoCs.

IMPLEMENTATION OF A MONITORING INFRASTRUCTURE FOR NoC-BASED MPSoC PLATFORMS

ABSTRACT

MPSoCs (Multi Processor System on Chips) are complex architectures. As a consequence, verify the system and assure quality of service constraints become complex tasks. Therefore, the research on mechanisms for verifying the system operation is necessary. These mechanisms aim at capturing data about the system status at each moment. This data is captured through the addition of monitors to the architecture. This work presents the implementation of a monitoring infrastructure for NoC-based MPSoCs. The monitoring is captured through traffic monitors added to the NoC (Network-on-Chip). The monitoring infrastructure is integrated to the microkernel of the MPSoC manager processor. Results show that the monitors do not interfere with the NoC global performance and that is possible to obtain the throughput of the flows in the network through the data collected by the monitors. The integration of the monitoring infrastructure to the MPSoC is validated based on matrices added to the microkernel of the manager processor. These matrices store the monitoring values of each NoC router channel.

Keywords: networks intra-chip (NoCs); monitors; monitoring systems; performance; MPSoCs.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

CCBE	*Congestion Controlled Best Effort*
DMA	*Direct Memory Access*
ECR	*East Channels RCV*
ECT	*East Channels Total*
FPGA	*Field Programmable Gate Array*
GPC	Gerador de Pacotes de Controle
GS	*Guaranteed Service*
IAP	*llegal Access Probe*
ISA	*Instruction Set Architecture*
ISE	*Integrated Software Environment*
LPR	*Local Port RCV*
LPT	*Local Port Total*
LUT	*Look Up Table*
MPC	*Model Predictive Control*
MPSoC	*Multi Processor System on Chip*
MSA	*Monitoring Service Access Point*
NCR	*North Channels RCV*
NCT	*North Channels Total*
NI	*Network Interface*
NoC	*Networks-on-Chip*
PE	*Processing Element*
PSFE	*Protocol-Specific Front End*
SCR	South Channels RCV
SCT	South Channels Total
RTL	*Register Transfer Level*
SoC	*System-on-a-Chip*
VC	*Virtual Channel*
VHDL	*VHSIC Hardware Description Language*
VHSIC	*Very High Speed Integrated Circuits*
WCR	*West Channels RCV*
WCT	*West Channels Total*

SUMÁRIO

1 INTRODUÇÃO

Com a evolução da tecnologia do silício obteve-se um aumento significativo na densidade dos circuitos integrados. Isso levou à criação de dispositivos cada vez mais complexos, que agregam de dezenas a centenas de módulos comunicando-se internamente em um único chip, conduzindo ao conceito de SoC (*System-on-a-Chip*) [MAR01]. O avanço contínuo da tecnologia permite o projeto de sistemas mais complexos, tais como sistemas multiprocessados integrados na forma de um SoC. Um primeiro esforço nesse sentido resultou nos processadores *Dual Core*, que estão substituindo os processadores com um único núcleo de processamento.

Dada a especificidade e elevada quantidade de aplicações dos sistemas embarcados atuais, é necessária a construção desses sistemas utilizando-se MPSoCs (*Multi Processor System on Chips*). MPSoCs são arquiteturas compostas por *n* elementos de processamento (*Processing Element*, PE) que executam instruções específicas e que, em conjunto, devem atender a requisitos de uma dada aplicação ou classe de aplicações [JER05].

MPSoCs são uma tendência atual no mercado de dispositivos eletrônicos voltados à computação móvel. Esses dispositivos utilizam como meio de interconexão, entre seus diversos núcleos de processamento, estruturas do tipo barramentos ou redes intra-chip (*Networks-on-Chip*, NoC) [BEN02]. A comunicação através de barramentos não satisfaz os requisitos de desempenho para os dispositivos compostos por diversos núcleos, visto que os elementos de processamento desses dispositivos comunicam-se entre si em alta velocidade, fazendo com que um MPSoC possua requisitos rígidos de comunicação. Com isso, as NoCs com topologia regulares destacam-se como uma tendência, devido a características intrínsecas ao seu funcionamento, como escalabilidade e paralelismo na comunicação. Dentre algumas vantagens que essa estrutura de comunicação apresenta, citam-se [BEN01]: (*i*) escalabilidade da largura de banda [GUE00]; (*ii*) paralelismo na comunicação; (*iii*) reusabilidade; (*iv*) confiabilidade; e (*v*) eficiência na gerência do consumo de energia.

Sob o ponto de vista do multiprocessamento, um MPSoC é dito homogêneo quando os elementos processadores que o compõe são todos da mesma natureza. Por exemplo, um sistema composto por *n* processadores com a mesma arquitetura e que permite a execução do mesmo conjunto de instruções ISA (*Instruction Set Architecture*) pode ser considerado homogêneo. Por outro lado, quando os elementos processadores são de naturezas distintas, o MPSoC é dito heterogêneo. Isso pode ocorrer, por exemplo, quando os processadores embarcados no sistema

possuem arquiteturas diferentes, como MIPS, ARM e PowerPC, ou propósitos diferentes, como processadores de propósito geral e processadores para processamento de sinais.

Mesmo que o emprego de MPSoCs represente uma tendência em sistemas embarcados, alguns tópicos ainda necessitam de maiores estudos, tais como metodologias de seus projetos e suporte à execução. Algumas funções essenciais de suporte à execução são: (*i*) gerência de memória; (*ii*) escalonamento; e (*iii*) mapeamento de tarefas. Em geral, tais políticas são implementadas por sistemas operacionais. Em MPSoCs com carga dinâmica de tarefas, um elemento de processamento deve realizar essas funções de controle em tempo de execução. Por exemplo, o mapeamento de tarefas pode se beneficiar do emprego de monitores caso considere em sua heurística o estado do sistema, definido pela informação de monitoramento. Caso esse estado fosse obtido por estimativas produzidas em tempo de projeto, a informação do estado do MPSoC pode ser imprecisa em um cenário de carga dinâmica de aplicações. Com monitores é possível substituir estimativas por medidas reais.

Outra aplicação importante dos monitores é o auxílio à validação, um dos pontos críticos no projeto de MPSoCs. Simulação é o estado da arte na verificação do desempenho de MPSoCs, no entanto ela possui desvantagens conceituais que a tornam inviável conforme a complexidade do sistema cresce. Nesse caso, uma possível solução é utilizar emulação, onde monitores de tráfego capturam informações relativas ao MPSoC em hardware enquanto este está em funcionamento. Essa técnica é mais eficiente e mais confiável que simulação, pois obtém informações relativas ao funcionamento real do MPSoC [RIC03] ao custo de elevada complexidade de execução.

Neste trabalho é apresentado o desenvolvimento de uma infra-estrutura de monitoramento para MPSoCs que usam NoC como meio de interconexão, com o objetivo de analisar o tráfego das portas de entrada dos roteadores. A partir dessa análise é possível realizar tomadas de decisões, tais como mapeamento e migração de tarefas em núcleos aptos a receber novas tarefas, para que o MPSoC tenha um melhor desempenho. A partir das informações de tráfego da NoC, obtidas através dos monitores, é montada uma estrutura de dados no processador que controla o MPSoC e que reflete a ocupação do MPSoC como um todo.

1.1 Motivações do Trabalho

MPSoCs estão sendo cada vez mais empregados, tanto na indústria como no meio acadêmico e a tendência é que ainda haja um grande crescimento na utilização desses sistemas [JER07]. Para conectar os diversos núcleos de processamento, MPSoCs utilizam como meio de interconexão

estruturas do tipo barramentos ou redes intra-chip (NoCs). As NoCs destacam-se como uma tendência, pois são mais escaláveis que barramentos, oferecendo maior largura de banda.

Como MPSoCs são arquiteturas bastante complexas, e em geral suas aplicações apresentam naturezas distintas com relação aos requisistos de comunicação, torna-se cada vez mais difícil estimar o comportamento do sistema. Assim, a realização de tarefas fundamentais, tais como a verificação do correto funcionamento do sistema e a garantia da qualidade de serviço tornam-se igualmente complexas. Uma alternativa para auxiliar na realização dessas tarefas, está no uso de monitores de tráfego em NoCs. A avaliação de um conjunto de variáveis, tais como: (*i*) violações no escalonamento dos processos; (*ii*) falhas no atendimento de *deadlines* das aplicações de tempo real; e (*iii*) congestionamento no meio de comunicação; é importante para definir qual o conjunto de informações que deve ser gerado pelos monitores.

1.2 Objetivos do Trabalho

O objetivo principal deste trabalho é descrever a implementação e validação de uma infra-estrutura de monitoramento para MPSoCs baseado em NoC. A validação ocorre através de sua implementação na NoC HERMES [MOR04] utilizada como meio de interconexão dos núcleos da plataforma MPSoC HeMPS [WOS07]. O monitoramento é realizado através de monitores inseridos nas portas de entrada dos roteadores da NoC que geram informações da quantidade de *flits*[1] que passam nas portas de entrada dos roteadores em uma determinada janela de tempo. Essa análise permite, por exemplo, obter a taxa de recepção dos pacotes (vazão). Cada roteador envia pacotes com os dados de monitoramento a um processador responsável pela gerência do MPSoC, o qual monta uma estrutura de dados que reflete a ocupação dos núcleos do MPSoC. A partir dessa infra-estrutura espera-se melhorar o desempenho da plataforma, realizando tomadas de decisões, tais como mapeamento e migração de tarefas em núcleos da plataforma que estejam menos sobrecarregados e aptos a receber novas tarefas.

1.3 Contribuição do Presente Trabalho

O presente trabalho apresenta como contribuições uma estrutura de monitoramento de NoCs e a integração do monitoramento em um MPSoC realizado através de um sistema de recepção e tratamento dos dados monitorados.

---

[1] *Flit* é a menor unidade de transferência de dados.

As contribuições deste trabalho, no que se refere ao monitoramento de NoCs, compreendem o monitoramento da vazão da rede HERMES, através do compartilhamento da rede para dados de aplicação e monitoramento. Destaca-se que o monitoramento pode ser realizado sobre diferentes configurações da rede HERMES.

Em relação à integração do monitoramento, um sistema de recepção e tratamento dos dados monitorados foi incluído no kernel do MPSoC HeMPS. O sistema permite que o processador mestre (gerente) tome decisões relevantes para a melhora de desempenho do MPSoC.

1.4 Organização do Documento

Este trabalho está organizado como segue. O próximo Capítulo apresenta conceitos relacionados ao processo de monitoramento de NoCs, bem como a revisão bibliográfica de trabalhos relacionados ao monitoramento do meio de comunicação em MPSoCs. A arquitetura dos monitores implementados e a nova estrutura da rede intra-chip, criada a partir da inserção dos monitores na rede, é apresentada, em detalhes, no Capítulo 3. Em seguida, o Capítulo 4 apresenta a utilização dos dados de monitoramento na plataforma MPSoC HeMPS enquanto que o Capítulo 5 traz os resultados obtidos com os monitores inseridos na rede HERMES e aqueles obtidos através da integração dos monitores em serviços do microkernel na plataforma HeMPS. Finalmente, o Capítulo 6 apresenta as conclusões do trabalho e diretrizes para futuros estudos.

2 ESTADO DA ARTE

Este Capítulo apresenta o estado da arte em sistemas para monitoramento de NoCs e MPSoCs baseados em NoC, tanto em questões de implementação quanto em requisitos de desempenho e métricas propostas para avaliar o desempenho de plataformas MPSoCs.

A monitoração de tráfego em NoCs tem por objetivo observar o comportamento dos dados que trafegam na rede, de modo a oferecer suporte a algum dispositivo de controle para que se mantenha o funcionamento correto no MPSoC onde a rede está inserida. O número elevado de núcleos em MPSoCs com requisitos rígidos de desempenho, além da própria complexidade inerente de NoCs, faz da monitoração um aspecto crítico no projeto de plataformas MPSoCs.

O processo de monitoração ocorre em duas etapas: (*i*) coleta de informações; e (*ii*) processamento das informações coletadas. O serviço de monitoração em NoCs consiste em se configurar pontos de teste que são integrados a roteadores ou interfaces de rede, onde as informações de tráfego são coletadas [CIO04]. A conexão dos pontos de teste à rede pode ser realizada de maneira externa, onde são acoplados a interfaces de rede dos IPs receptores de tráfego, ou de maneira interna, na qual é acoplado um ponto de teste em cada roteador ou em um subconjunto de roteadores da NoC.

As informações coletadas devem ser transmitidas para um sistema de controle, denominado MSA (*Monitoring Service Access Point*) [CIO06a], para que sejam processadas e para que se tome alguma decisão de acordo com os eventos diagnosticados, tais como mapeamento ou migração de tarefas. As informações de monitoração normalmente incluem valores de variáveis relativas ao tráfego que interferem no desempenho da aplicação. Dentre esses valores pode-se considerar, por exemplo, os parâmetros de contenção e liberação de pacotes, bem como a utilização de filas de roteadores.

A avaliação da contenção de pacotes verifica o atendimento de requisitos relacionados ao tempo de envio de mensagens de uma dada aplicação. Os seguintes parâmetros podem ser monitorados:

- latência, é o intervalo de tempo decorrido entre o início da transmissão de um pacote e sua completa recepção no destino;
- *jitter*, indica o grau de variabilidade dos valores de latência, através do cálculo do seu desvio padrão, e;

- *cpf*, refere-se ao número de ciclos para transmitir um *flit* entre roteadores vizinhos.

Os requisitos de liberação de pacotes são avaliados com o objetivo de verificar o cumprimento ou não de requisitos de taxa de transmissão de dados pela fonte, recepção no destino e ocupação dos canais da rede. O tráfego aceito é a relação existente entre as taxas de injeção e as taxas de recepção, sendo que as informações para análise são coletadas nas interfaces externas dos núcleos geradores e receptores de tráfego. A taxa de utilização de canais especifica o percentual de tempo em que o mesmo está ocupado com a transmissão de dados.

Durante a monitoração de utilização de filas são verificados os *buffers* dos roteadores (*buffers* internos) e/ou aqueles conectados às interfaces de rede - NI (*Network Interface*). Nessa monitoração, salientam-se alguns parâmetros importantes, tais como:

- a quantidade de posições preenchidas do *buffer*;
- a quantidade de dados não inserida no *buffer* da NI pelo fato do mesmo estar cheio;
- o número de vezes que a aplicação quis consumir dados de *buffer* da NI mas o mesmo está vazio, e;
- o número de ocorrências em que o crédito em *buffers* internos de roteadores vizinhos estiver em nível baixo.

Quando as informações são avaliadas após a execução do sistema, e servem como referência para alguma modificação estrutural da arquitetura da rede, o processamento é denomidado *off-line*. Por exemplo, em função dos dados obtidos durante a simulação pode-se dimensionar enlaces, *buffers* e até suprimir roteadores da rede gerando-se uma topologia irregular. Entretanto, quando as informações são avaliadas durante a simulação do sistema, o processamento é denominado *on-line*. O *foco do presente trabalho* está no monitoramento *on-line*. O monitoramento *on-line* permite a tomada de decisões de mapeamento e de roteamento, por exemplo. O processamento *off-line* mostra-se mais apropriado para tratamento de tráfego conhecido, onde é maior a previsibilidade do comportamento da aplicação, isto devido a rotas de tráfego previamente definidas.

Quanto à localização do serviço de processamento dos dados de monitoração, duas abordagens podem ser adotadas: (*i*) monitoração centralizada; e (*ii*) monitoração distribuída [CIO04]. As maneiras como podem ser transmitidas as informações de monitoramento em NoCs são apresentadas abaixo:

(1) interconexão física separada, com uma rede de dados para aplicações e uma rede

física específica, à qual os pontos de teste são conectados para monitorar a rede de dados para aplicações;

(2) interconexão física comum, apenas uma rede é utilizada como estrutura de comunicação, porém dados de monitoração são transmitidos em fios dedicados, e;

(3) interconexão física comum, somente uma rede é utilizada como estrutura de comunicação, onde os enlaces que interligam os roteadores têm sua utilização multiplexada entre tráfego de dados de aplicação e de monitoração.

Neste trabalho empregam-se: (*i*) monitoração distribuída, com controle centralizado; e (*ii*) interconexão física comum multiplexada.

2.1 Marescaux et al. 2005

Marescaux et al. [MAR05] apresentam uma técnica de controle de congestionamento utilizando monitores em roteadores, denominado *monitor de buffer*. O monitor de *buffer* utiliza quatro sinais: *new_packet*, acionado quando um novo pacote chega ao roteador; *source*, que indica a fonte do pacote; *priority*, que especifica a prioridade do pacote e *packets_outqueue*, que informa quantos pacotes existem no *buffer*. O monitor de *buffer* é conectado na porta de saída dos *buffers* dos roteadores da rede de dados. Sua função principal é verificar o congestionamento através de medição da quantidade de pacotes que são armazenados no interior do *buffer* do roteador. Se existirem mais pacotes no *buffer* do que especificado em um nível de *threshold*, significa que o mesmo está congestionado. Uma vez detectado congestionamento, deve ser enviada uma mensagem de congestionamento para o IP de controle.

Para tomar uma decisão, o monitor de *buffer* armazena informações relativas a pacotes recentes em uma fila separada de histórico, denominada *hFIFO*, a qual contém o endereço fonte e a prioridade do pacote. É então invocada a função *find_lowest* para pesquisar todos os campos de prioridade na *hFIFO* para encontrar o endereço fonte do pacote que possui a menor prioridade. Se dois pacotes possuem a mesma prioridade, a fonte do pacote mais recente é selecionada.

Uma mensagem de notificação de congestionamento composta pelo endereço fonte e a prioridade é então reportada ao monitor de congestionamento. A entrada na FIFO de histórico é marcada como *notified* para evitar notificações subsequentes. A marca *notified* é local à *hFIFO* do monitor de *buffer*, portanto o mesmo pacote pode acionar mais notificações de congestão nos próximos *hops* ao longo do caminho de um fluxo. Finalmente, é gerada uma mensagem de controle

que utiliza fios separados para transmitir notificações de congestionamento sobre a rede para as fontes de tráfego relacionadas, as quais ajustam a sua taxa de injeção.

2.2 Brand et al. 2007

Brand et al. [BRA07] propõem uma estratégia de controle de congestionamento com o objetivo de limitar a latência na rede, denominada CCBE (*Congestion Controlled Best Effort*). Os autores definiram a utilização dos enlaces como sendo o parâmetro de medição de congestionamento, pelo fato de o considerarem mais direto que a ocupação em *buffers*. Medidas de congestionamento realizadas no enlace são consideradas mais precisas, por especificar com precisão onde a congestão ocorre.

A técnica MPC (*Model Predictive Control*) combina predições baseadas na modelagem de um sistema com dados de medição. Controladores MPC oferecem suporte para tratar latências com variabilidade (*jitter*), o que é considerado um fator crítico. Parâmetros de referência, como por exemplo utilização de enlaces e cargas oferecidas por conexões de melhor esforço – BE (*Best-Effort*) podem ser especificados ao MPC, o qual toma decisões de controle com base nestes valores para que o sistema não oscile. O controlador MPC é utilizado de forma centralizada em conjunto com o serviço de monitoração centralizado.

Pontos de teste espalhados pela rede são utilizados para obter medidas de utilização dos enlaces. Tais informações são transmitidas ao longo de canais de serviço garantido – GS (*Guaranteed Service*), para atingir o objetivo de se obter um sistema confiável, sem interferência de outros fluxos da rede. A partir dos dados de medição, é realizado o controle de injeção de dados na rede pela fonte, de modo a evitar congestionamento.

2.3 Ciordas et al. 2004

Ciordas et al. em [CIO04] propõem um serviço de monitoramento genérico para NoCs onde os pontos de teste de monitoração são conectados a componentes tais como: roteadores ou interfaces de rede. O serviço de monitoração é configurado em tempo de execução, por um módulo IP conectado a uma interface de rede, denominado MSA (*Monitoring Service Access Point*).

O serviço de processamento dos dados de monitoração pode ser centralizado ou distribuído. Em um serviço de monitoração centralizada, como o apresentado na Figura 1(a), a informação monitorada pelos pontos de teste é coletada em um ponto central, nesse caso por um simples

MSA, destacado na Figura 1(a). Essa configuração é possível e conveniente para redes pequenas. Contudo, o envio de dados monitorados somente para um ponto central pode tornar-se um gargalo em redes maiores. No serviço de monitoramento distribuído, a informação a ser monitorada é coletada de diferentes subconjuntos de componentes de NoC, localizados em diferentes pontos da rede. Nessa configuração, gargalos de desempenho são minimizados adquirindo-se, assim, escalabilidade, o que torna esta estratégia mais adequada para redes de maior tamanho. A Figura 1(b) ilustra o serviço de monitoração distribuído, composto por dois subconjuntos de componentes, *MSA1* e *MSA2*, que controlam diferentes regiões da rede.

O gerenciador de tráfego direciona o tráfego do MSA para os pontos de teste (quando for requisitada a configuração dos pontos de teste) e o tráfego dos pontos de teste para o MSA (quando forem requisitadas informações de monitoramento).

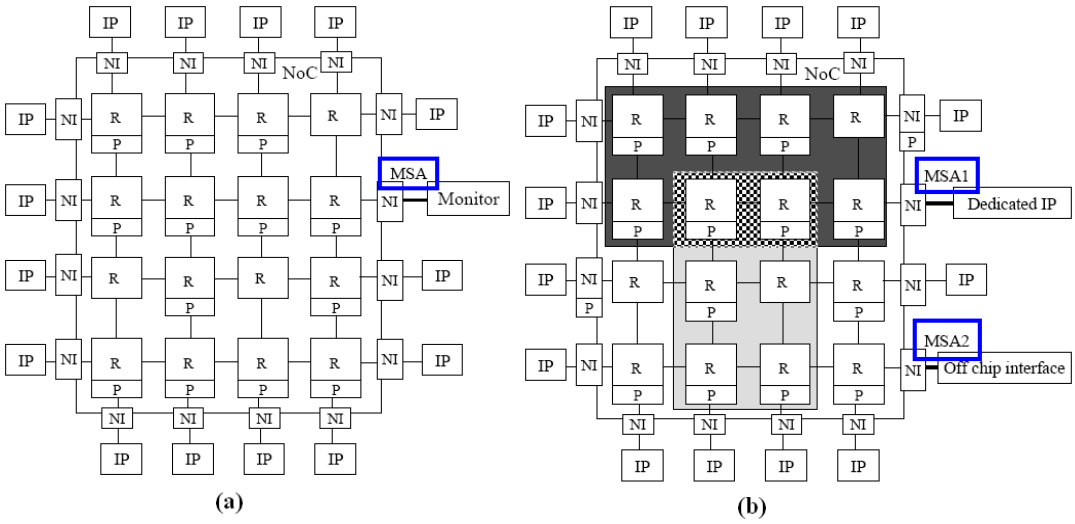


Figura 1 - Configurações do serviço de monitoração dos dados coletados: (a) centralizado e (b) distribuído, onde os pontos de teste são representados por (P) e um sistema de controle é representado por (MSA) [CIO04].

Toda a informação monitorada é modelada no formato de eventos. Cada evento possui os seguintes parâmetros:

- identificador, o qual define o tipo de evento;
- etiqueta de tempo, que define o momento no qual é gerado o evento;
- produtor, que é a entidade que gera o evento, e;
- atributos, que definem valores relacionados aos eventos.

Os pontos de teste de monitoração responsáveis pela captura das informações na NoC não são necessariamente conectados a todos os componentes da NoC, sendo sua localização definida

em tempo de projeto. Para a definição dessa localização é necessário observar o compromisso de seu custo com a quantidade de eventos observados. Eventos monitorados podem ser relacionados:

- à configuração das comunicações de usuário;
- às informações das aplicações da rede, por exemplo, movimentações de dados da memória;
- à configuração do funcionamento da rede;
- a alertas da rede, por exemplo, indisponibilidade em *buffers*, e;
- aos próprios serviços de monitoração, como, por exemplo, perda de dados.

Ciordas et al. [CIO06a] apresentam diversas alternativas escaláveis para a transmissão das informações de monitoramento em NoCs. As alternativas de como podem ser transmitidas as informações de monitoramento são ilustradas na Figura 2. Nessa Figura, podem ser observados exemplos da utilização de um processamento de informações centralizado, onde o MSA é um núcleo IP conectado à rede. É possível, entretanto, a utilização de uma configuração distribuída. No caso ilustrado pela Figura 2(a), outra rede física específica é escolhida para monitorar a rede de dados a qual são conectados os pontos de teste. Essa estrutura é utilizada para transportar as informações monitoradas pelos pontos de teste para o MSA e para monitorar a configuração de seu tráfego para os pontos de teste. A estrutura dessa NoC pode ter topologia diferente da rede de dados. Um número menor de pontos de teste pode ser adotado, dando cobertura a regiões mais críticas da rede. O fato de os dados monitorados trafegarem em uma rede isolada traz como vantagem a ausência de interferência que seria causada por dados de aplicações. As desvantagens dessa abordagem estão relacionadas ao custo em área de silício e à impossibilidade de reuso da estrutura para outros tipos de tráfego. A inserção de mais fios aumenta a probabilidade de aparecimento de *crosstalk*, um fenômeno que pode ocasionar erros na transmissão dos dados.

Na Figura 2(b) é ilustrada uma segunda alternativa de monitoração, onde apenas uma rede é utilizada. Observa-se que são acrescentadas novas portas nos roteadores para fazer a conexão dos pontos de teste e do MSA. O fato de os enlaces serem conectados aos mesmos roteadores que a rede de dados de aplicações traz ao sistema flexibilidade, uma vez que é possível o acionamento da transmissão de dados por essas vias quando as mesmas não estiverem enviando dados de monitoração. A principal desvantagem dessa alternativa é que os roteadores são normalmente limitados a um número máximo de portas.

A terceira alternativa para transmitir as informações de monitoramento é ilustrada na Figura 2(c), onde se pode visualizar a abordagem em que os enlaces que interligam os roteadores têm sua utilização multiplexada entre tráfego de dados de monitoração e de aplicações. Essa abordagem pode causar interferência dos dados de monitoração na transmissão de dados de aplicação, o que pode ser minimizado com a adoção de um mecanismo de escalonamento de comunicação baseado em prioridades. As vantagens dessa abordagem encontram-se no menor custo em área de silício e a reusabilidade da estrutura original da rede.

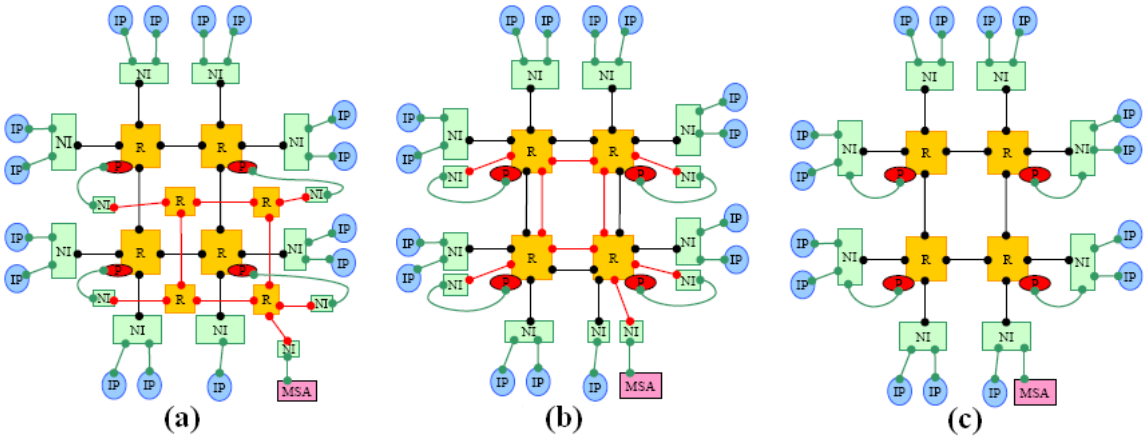


Figura 2 - Alternativas para transmitir informações de monitoramento [CIO06a].

Um breve resumo das vantagens e desvantagens das alternativas para transmitir informações de monitoramento abordadas anteriormente é apresentado na Tabela 1.

Tabela 1 - Comparação das vantagens (+) e desvantagens (-) das alternativas abordadas para transmitir informações de monitoramento.

	Alternativa(1) [Figura 2(a)]	Alternativa(2) [Figura 2(b)]	Alternativa(3) [Figura 2(c)]
Fluxo de Projeto	+	+	-
Não-Intrusiva	+	+	+ e -
Custo de Área	-	-	+

Observa-se na Tabela 1 que o uso de caminhos independentes para a monitoração, alternativas (1) e (2), são opções não-intrusivas (não afetam o fluxo dos dados) e são mais simples de serem acrescidas no fluxo de projeto da rede. A alternativa (3) requer a modificação do roteador, sendo necessário o acréscimo de lógica adicional. Por essa razão o fluxo de projeto é comprometido. Entretanto, a alternativa (3) é a que menor causa impacto na área final, pois não requer uma rede extra (alternativa (1)) nem portas adicionais na interface de rede (alternativa (2)).

2.4 Kim et al. 2007

Kim et al. em [KIM07] propõem um sistema de monitoração de tráfego onde são medidos diversos parâmetros em tempo de execução, com o objetivo de, através dos valores medidos, modificar dois parâmetros da NoC: (*i*) tamanhos de *buffers*; e (*ii*) caminhos de roteamento.

A estrutura do sistema de monitoramento de tráfego consiste de três subsistemas: (*i*) interface hospedeira; (*ii*) controlador central; e (*iii*) unidades de monitoramento. A unidade de monitoramento consiste de um ponto de teste de tráfego, um gerenciador de tráfego e uma memória de tráfego. O ponto de teste é conectado a um roteador ou a uma interface de rede com o objetivo de registrar parâmetros de tráfego em tempo de execução, tais como latência fim-a-fim, preenchimento de *buffers* e utilização de enlaces. O gerenciador de tráfego, então, armazena os dados em sua memória local após anexar um *timestamp* em cada registro, utilizando como referência um contador global conectado a todas as unidades de monitoramento. Durante a execução de uma aplicação, todos os resultados monitorados são armazenados na memória local correspondente. Os resultados monitorados são transferidos pelo módulo *interface hospedeira* para um computador externo (*controlador central*), através de uma conexão do tipo *Ethernet*. O *controlador central* habilita/desabilita cada unidade de monitoramento baseado em um escopo de requisições de monitoramento a partir de regiões da rede e um intervalo de tempo.

O sistema de monitoramento de tráfego proposto pelos autores possui uma arquitetura modular o que permite anexar uma unidade de monitoramento a qualquer componente da NoC em tempo de execução.

2.5 Fiorin et al. 2008

Fiorin et al. em [FIO08] apresentam um sistema de monitoramento para arquiteturas baseadas em NoC, com o objetivo de detectar violações de segurança em dispositivos e ajudar a combatê-las através do monitoramento de acessos a endereços específicos de memória e desvios de comportamentos esperados. O sistema de monitoramento é composto, por três elementos: (*i*) pontos de teste; (*ii*) gerente de segurança da rede; e (*iii*) infra-estrutura de comunicação (roteadores e interfaces de rede).

Os pontos de teste coletam informações sobre o tráfego da NoC e são localizados dentro de interfaces de rede. O fato de adicionar os pontos de teste dentro de NIs permite a análise direta do tráfego quando inserido pelo núcleo. Sendo assim, um tráfego detectado como malicioso pode ter

seu acesso suspenso ou limitado à NoC. Cada ponto de teste gera um evento a ser encaminhado a um gerador de eventos. Nesse, é gerado um pacote utilizado para notificar o gerente de segurança da rede sobre possíveis violações de segurança encontradas. As definições de evento são aplicadas através dos conceitos discutidos em [CIO04], onde um evento pode ser representado como uma tupla, composta de um identificador, uma etiqueta de tempo (*timestamp*), um produtor e vários atributos.

A arquitetura do ponto de teste é constituída por dois módulos principais, são eles: (*i*) ponto de teste de acesso ilegal – IAP (*Illegal Access Probe*) e (*ii*) ponto de teste de negação de serviço - DoSP (*Denial of Service Probe*). O IAP é encarregado por detectar a presença de tentativas de acessos ilegais a blocos de memórias ou faixas de endereços de memórias em sistemas com memória compartilhada. No entanto, esse módulo não fornece proteção contra os ataques visando à criação de negação de serviço em um sistema, por exemplo, através da injeção de pacotes inúteis que tenham como meta a intenção de reduzir os recursos de largura de banda do sistema. O DoSP é responsável por coletar informações sobre o tráfego gerado pelos elementos de processamento que fazem interface com a NI para detectar comportamentos não comuns no tráfego, interpretados como sintomas de ataques de negação de serviço.

O gerente de segurança da rede é um núcleo dedicado do sistema responsável pela coleta de eventos e informações provenientes dos pontos de teste além de analisar esses dados e elaborar medidas adequadas para neutralizar a ataques.

O sistema de monitoramento de segurança consome em torno de 25,6% da área total da interface de rede utilizada por esse sistema. Comparada com a mesma interface de rede, porém sem o sistema de monitoramento, a sobrecarga de área associada é em torno de 34,7%.

2.6 Vermeulen et al. 2009

Vermeulen et al. em [VER09] apresentam uma infra-estrutura de monitoramento para MPSoCs que possui seus elementos de processamento comunicando-se através de uma NoC e de barramentos. A Figura 3 ilustra a infra-estrutura proposta pelo autor. Nota-se, através da Figura que há barramentos externos a NoC, conectados a mais de um roteador. A infra-estrutura é aplicada com o objetivo de analisar o desempenho do MPSoC e validar o seu correto funcionamento. Um modelo (*template*) genérico de monitor é implementado para monitorar barramentos e roteadores, esses monitores são instanciados em tempo de projeto através de um fluxo de projeto de NoC.

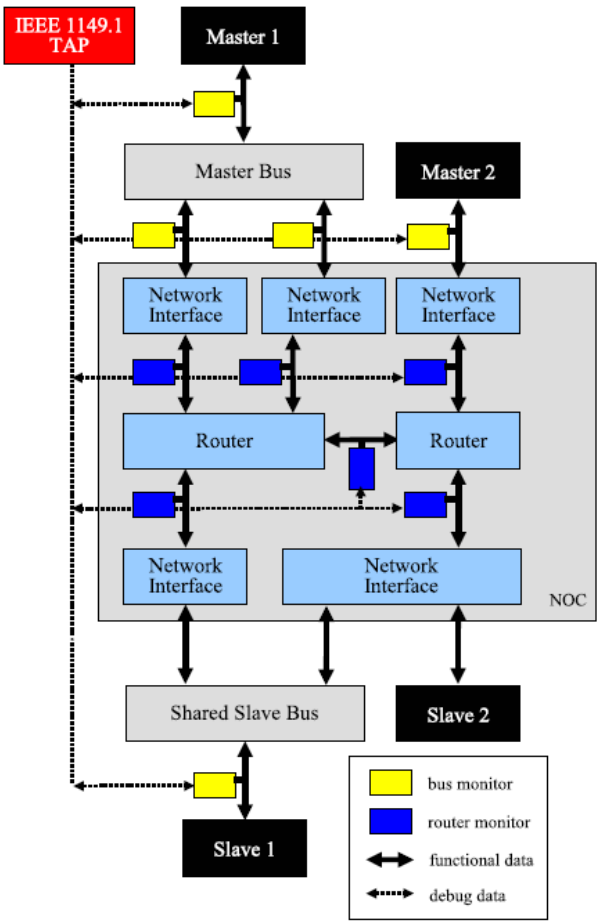


Figura 3 - infra-estrutura de monitoramento proposta por [VER09].

O monitoramento de roteador observa um fluxo de pacote de dados nos canais de comunicação entre roteadores ou entre roteadores e interfaces de rede. No monitoramento dos barramentos externos é observado o fluxo de dados entre as interfaces de rede e barramentos, ou barramentos e elementos de processamento. O *template* do monitor projetado inclui os seguintes componentes:

- protocolo específico – PSFE (*Protocol-Specific Front End*);
- medidor de largura de banda;
- medidor de transição de latência;
- gerador de *trigger*;
- gerador de *checksum*, e;
- registradores de controle e *status*.

O PSFE possui um filtro de transações para restringir o monitoramento para um subconjunto de dados de comunicação sobre um canal. Utilizando o conhecimento do protocolo de

comunicação da NoC, o PSFE fornece dados brutos e informações sobre: qualidade de serviço (*Best Effort*, BE ou *Guaranteed Throughput*, GT), número de palavras em um *flit* do pacote da NoC, e se os dados no canal pertencem a um cabeçalho do pacote, um corpo de pacote e fim de um pacote. Através dessas informações o PSFE pode ser configurado para filtrar as transações observadas pelos monitores.

A utilização de largura de banda para esse trabalho é definida como o número real de ciclos de barramento usado para transportar dados sobre o canal monitorado, normalizados para o número total de ciclos de barramento em um determinado intervalo de tempo. Essa métrica é medida para todos os subconjuntos de tráfego sobre o canal, conforme determinado pela programação do filtro de transações opcionais.

Medições de latência são úteis para os protocolos de barramento, onde um protocolo de comunicação (*handshake*) é usado para transferir cada elemento de dados de uma transação. Em uma NoC, os dados são tipicamente transportados através de canais em um determinado período de tempo, isto é, sem variação na latência. Nesses casos, não é necessário incluir medidas de transação de latência para o monitor.

O componente gerador de *trigger* é usado para gerar um disparo de depuração após um número predefinido de transações específicas que tenham ocorrido sobre o canal monitorado. O componente de gerador de *checksums* é útil para calcular assinaturas compacta de dados brutos sobre o canal de comunicação ou valores de protocolos abstratos. A mesma funcionalidade do filtro de transação é aplicada no componente de somas de verificação. O controle dos monitores e os registros de *status* são acessíveis a partir de uma interface de depuração. Através desses registros um operador do sistema pode programar as métricas de desempenho para medir e definir pontos de disparo para parar o sistema de monitoramento.

Os resultados experimentais desse trabalho apresentam que o custo de área para cada monitor é relativamente pequeno comparado a um projeto de SoC de milhões de portas, habilitando seu uso irrestrito em lugares estratégicos em uma plataforma MPSoC, de forma a ajudar a encontrar erros funcionais e auxiliar na análise de desempenho em tempo execução.

2.7 Fiorin et al. 2009

Fiorin et al. em [FIO09] propõem monitoramento em tempo de execução de atividades de um sistema em uma plataforma MPSoC baseada em NoC através da observação das operações

realizadas sobre um subsistema de comunicação. A principal meta do sistema de monitoramento é recuperar informações úteis em tempo de execução para otimizar e alocar recursos em sistemas adaptativos. As informações são coletadas através de pontos de teste inseridos dentro de interfaces de rede e são enviadas para uma unidade central, encarregada de coletar as informações em tempo de execução. A plataforma referida nesse trabalho possui seus núcleos mapeados em memória e a NoC implementa um protocolo no nível de transação, com núcleos atuando como iniciadores (fontes) ou destinos de transações.

O sistema de monitoramento é composto por três elementos principais: (*i*) pontos de teste; (*ii*) infra-estrutura de comunicação (roteadores e NIs); e (*iii*) gerente. Nos pontos de teste, as transações são analisadas quando solicitadas por um núcleo (pontos de teste em iniciadores NI) ou quando são recebidos (pontos de teste em fontes NI). As informações medidas pelos pontos de teste são enviadas ao gerente (unidade central) através de pacotes de eventos criados por um gerador de eventos, acionado pelos pontos de teste. As definições de evento são aplicadas através dos conceitos discutidos em [CIO04], onde um evento pode ser representado como uma tupla, composta de um identificador, uma etiqueta de tempo (*timestamp*), um produtor e vários atributos.

A partir da análise de possíveis transações solicitadas por um núcleo agindo como iniciador ou destino, foram identificadas três categorias gerais de perfis que podem ser extraídos pelo sistema de monitoramento:

- vazão, quantidade de tráfego gerado e recebido por núcleos, monitorada a fim de medir largura de banda de comunicação;
- tempo/latência, tempo pode ser monitorado em diferentes níveis e diferentes propósitos, pode ser distinguido entre o tempo monitorado envolvendo aspectos de comunicação e computação de aplicações, e;
- ocorrências, contadores para medir a ocorrência de um evento em particular.

A arquitetura do ponto de teste é composta por três diferentes arquiteturas de pontos de teste, são elas: (*i*) ponto de teste de vazão; (*ii*) ponto de teste de tempo; e (*iii*) ponto de teste contador. O ponto de teste de vazão (*Throughput Probe*, THP) fornece medidas sobre a quantidade de tráfego gerado pelos núcleos em uma determinada janela de tempo. No ponto de teste de tempo (*Timing Probe*, TMP) informações sobre medidas de tempo são fornecidas. O ponto de teste contador (*Counter Probe*, CNP) monitora o número de transações, dirigidos a um destino

específico. O comprimento da janela de tempo pode ser definido pelo gerente em tempo de execução ou por um projetista em tempo de projeto. No final da janela de tempo, o gerador de eventos é acionado e cria um pacote para transmitir as informações coletadas para o gerente.

A sobrecarga de área que o sistema de monitoramento implica na NI é de aproximadamente 27% para pontos de teste implementados em iniciadores e de aproximadamente 10% quando implementados em destinos.

2.8 Considerações Finais

A Tabela 2 resume as principais características dos trabalhos identificados na literatura e apresentados neste Capítulo. Observa-se que os trabalhos possuem como métrica de monitoramento o preenchimento de *buffers* e quantidade de dados que trafegam nos enlaces. Os pontos de teste localizam-se em roteadores em grande parte dos trabalhos. Quanto ao processamento de informações de monitoração, a abordagem centralizada é ainda a alternativa mais utilizada. Na transmissão das informações monitoradas, a maioria dos trabalhos utiliza somente uma rede como estrutura de comunicação. Além disso, os sistemas que monitoram o tráfego da rede observam o comportamento do tráfego em tempo de execução, obtendo assim, informações mais precisas do tráfego de dados.

Tabela 2 - Tabela comparativa dos trabalhos relacionados a sistemas de monitoramento de NoCs e MPSoCs apresentados no estado da arte.

Autores	Pontos de Monitoração na Rede	Processamento de Informações de Monitoração	Estrutura de Transmissão de Dados de Monitoração	Localização de Pontos de Teste
Marescaux et al. em [MAR05]	*Buffers*	Distribuído	Canais físicos dedicados	Roteadores
Brand et al. em [BRA07]	Enlaces	Centralizado	Canal virtual de serviço garantido	Roteadores
Ciordas et al. em [CIO04]	*Buffers*	Centralizado e Distribuído	Canal virtual de serviço garantido e melhor esforço	Roteadores
Kim et al. em [KIM07]	*Buffers* e enlaces	Centralizado	Conexão ponto-a-ponto com o roteador central	Roteadores
Fiorin et al. em [FIO08]	Enlaces	Centralizado	Canais físicos multiplexados	Interfaces de rede
Vermeulen et al. em [VER09]	Enlaces e barramentos	Centralizado	Canais físicos dedicados	Entre roteadores, roteadores e NIs, NIs e barramentos, barramentos e PEs
Fiorin et al. em [FIO09]	Enlaces	Centralizado	Canais físicos multiplexados	Interfaces de rede
Trabalho Proposto	**Enlaces**	**Centralizado**	**Canais físicos multiplexados**	**Roteadores**

No trabalho proposto os monitores são adicionados nos roteadores da NoC, para que observem em tempo de execução o comportamento do tráfego de dados nas portas de entrada dos roteadores. O foco do trabalho esta no processamento das informações realizado no processador gerente do MPSoC, que de posse das informações gera uma estrutura de dados que reflete a ocupação dos processadores da plataforma MPSoC. A transmissão dos dados de monitoramento ocorre na mesma rede em que trafegam dados de aplicação, evitando um acréscimo de área.

3 MONITORAMENTO DA REDE HERMES

Baseando-se nos trabalhos revisados, os monitores nos roteadores da rede HERMES[2] [MOR04] foram implementados com o objetivo de monitorar o tráfego da rede, gerando informações úteis para analisar seu desempenho. Esse monitoramento indica a vazão em cada porta de entrada dos roteadores e permite, assim, que se determine a carga da rede.

Este Capítulo corresponde à *principal contribuição* da Dissertação, ou seja, a implementação dos monitores em roteadores da rede HERMES. O restante do Capítulo apresenta as alterações realizadas no roteador da arquitetura alvo para adicionar os monitores na rede bem como o módulo desenvolvido para receber as informações geradas pelos monitores e realizar a montagem de um pacote de dados de monitoramento.

3.1 Rede HERMES

A rede HERMES é uma infra-estrutura parametrizável, especificada em VHDL no nível RTL. Os roteadores possuem uma lógica de roteamento centralizada e cinco portas de comunicação bidirecionais: *Leste*, *Oeste*, *Norte*, *Sul* e *Local*. A porta *Local* é utilizada para estabelecer a conexão entre o roteador e seu módulo de processamento local, enquanto as outras portas são conectadas a roteadores vizinhos. Cada porta possui *buffers* de entrada para armazenamento temporário de dados. Cada porta unidirecional (entrada ou saída) do roteador corresponde a um canal físico. A rede utiliza o modo de chaveamento *wormhole*, no qual um pacote é transmitido entre os roteadores em *flits*. O modo *wormhole* permite que cada canal físico seja multiplexado em *n* canais virtuais (*Virtual Channel*, VC). Para controle de fluxo pode ser utilizada a estratégia *handshake* ou a baseada em créditos. A arquitetura básica da rede utilizada para a implementação dos monitores apresentada possui uma topologia malha 4x4, com tamanho do *flit* de 16 *bits*, profundidade das filas de 8 posições, algoritmo de roteamento XY, controle de fluxo baseado em créditos e não possui canais virtuais.

3.2 Implementação dos Monitores na Rede HERMES

A análise do desempenho da NoC HERMES é realizada através de monitores inseridos nas portas de entrada de todos os roteadores da rede. Os monitores implementados geram informações sobre o tráfego da rede a partir da quantidade de *flits* que passam nas portas de

[2] HERMES realiza a interconexão dos núcleos da plataforma MPSoC alvo desse trabalho.

entrada dos roteadores da rede em uma determinada janela de tempo.

As informações geradas pelos monitores são transmitidas a um módulo denominado de GPC (Gerador de Pacotes de Controle), sendo que a transmissão é realizada através dos próprios monitores. O GPC tem por finalidade enviar periodicamente pacotes de dados de monitoramento a um sistema de controle (MSA), onde as informações de todos os monitores são coletadas. Para a estrutura de monitoramento são inseridos nos roteadores da NoC HERMES monitores em cada porta de entrada dos roteadores, assim como o módulo gerador de pacotes (GPC), como pode ser visto na Figura 4. Como há somente uma rede como estrutura de comunicação, é necessário realizar a multiplexação de alguns sinais provenientes da porta local do roteador e do GPC para que sejam transmitidos ou dados provenientes do IP, ou seja, das aplicações ou dados de monitoramento oriundos do GPC.

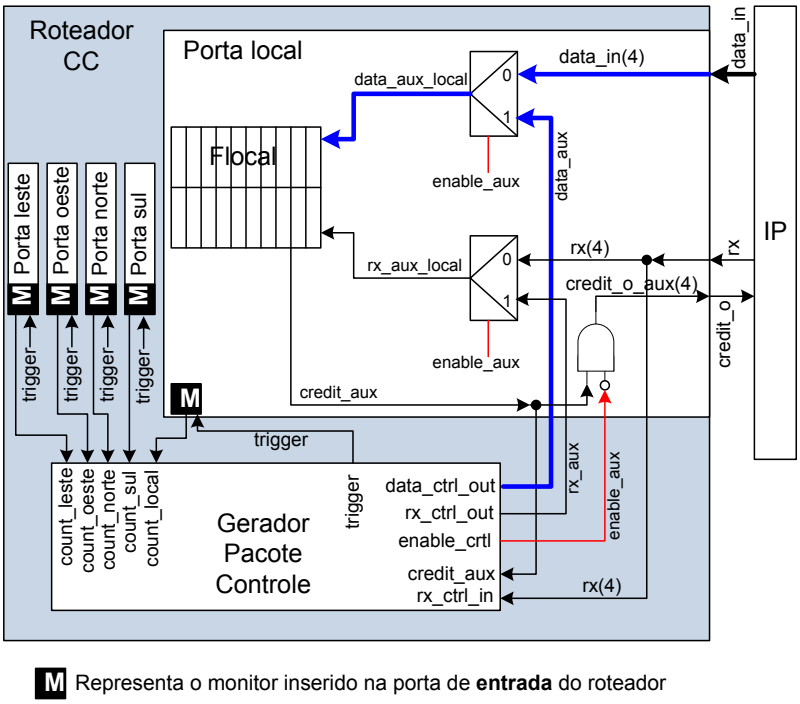


Figura 4 - Nova estrutura do roteador CC da rede HERMES, onde se acrescentou o módulo GPC, uma porta lógica *AND* e dois multiplexadores.

A arquitetura do monitor é desenvolvida para roteadores de cinco portas, mas esses podem ser aplicados a roteadores com um número menor de portas. No caso de roteadores que possuam um menor número de portas, o monitor irá conter o valor de quantidade de *flits* igual a zero para as portas inexistentes, pois nessas não há tráfego de dados. Por exemplo, para o roteador que possua apenas as portas *oeste*, *norte* e *local*, o valor da quantidade de *flits* que passaram em uma determinada janela de tempo na porta de entrada *sul* e *leste* é igual a zero.

As portas *leste*, *oeste*, *norte*, *sul* e *local* enviam ao GPC o valor referente à quantidade de *flits* recebidos (*count_xxx*). O IP conectado à porta *local* envia ao GPC o sinal de controle *rx(4)*, o qual indica se esse IP está ou não injetando dados na rede. Como os dados de monitoramento (*data_aux*) são multiplexados com os dados do IP (*data_in(4)*) e o sinal de controle *rx(4)* é multiplexado com o sinal de controle *rx_aux,* houve a necessidade de se criar o sinal *enable_aux* para realizar esses controles. Quando o sinal *enable_aux* está ativo, os dados de monitoramento são enviados para a rede.

O controle de fluxo responsável pela transmissão de dados da aplicação é realizado pelo sinal *credit_o*, definido em função dos sinais *credit_aux* e *enable_aux*, provenientes da fila *Flocal* e *GPM,* respectivamente. Quando se está enviando dados de monitoramento (*enable_aux = '1'*), a porta lógica *AND* desabilita o sinal de crédito que vai para o IP conectado à porta *local* (*credit_o = '0'*), evitando, assim, a inserção de novos dados durante a transmissão de pacotes de controle.

Os monitores repassam a quantidade de *flits* que são recebidos nas portas de entrada dos roteadores em uma janela de tempo. No nível físico a transferência de *flits* nos enlaces da NoC HERMES segue duas possíveis estratégias de controle de fluxo: (*i*) crédito e (*ii*) *handshake*. Os monitores funcionam para essas duas estratégias.

Em um controle de fluxo baseado em créditos, a transferência de um *flit* se dá em um ciclo de relógio (na NoC HERMES, sinal *clock_rx* do roteador). A quantidade de *flits* só é computada pelo monitor se houver crédito na porta do roteador. No caso do controle de fluxo baseado no protocolo *handshake*, a transferência de um *flit* se dá em dois ciclos de relógio, ou seja, a quantidade de *flits* é computada pelo monitor a cada dois ciclos de relógio.

3.2.1 Módulo Gerador de Pacote de Controle (GPC)

O módulo Gerador de Pacote de Controle recebe as informações coletadas pelos monitores, ou seja, recebe informações sobre o tráfego provindas das cinco portas do roteador. Em cada um dos dados há a informação da quantidade de *flits* que foram consumidos pelas portas *leste*, *oeste*, *norte*, *sul* e *local* dos roteadores da rede. Tais informações devem ser transmitidas para um sistema de controle. Para isso, o GPC as encapsula em um pacote de monitoramento, denominado pacote de controle. Esse pacote tem um tamanho total de dez *flits*, sendo quatro para o cabeçalho (*header*) e seis para o corpo de dados (*payload*). A Figura 5 ilustra os campos que compõem o pacote de controle.

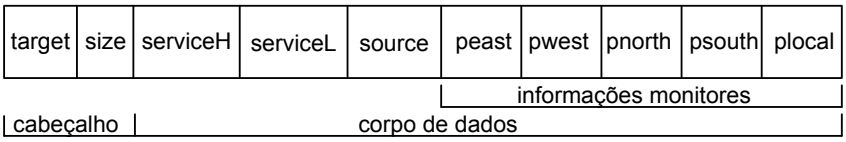


Figura 5 - Pacote de controle gerado pelo GPC. Pacote composto por quatro flits de cabeçalho e seis flits de corpo de dados.

Como visto, além das informações de tráfego tem-se outras informações no pacote. Para compor o cabeçalho do pacote temos o campo *target*, em que é determinado o destino do pacote; o campo *size*, que possui o tamanho total corpo de dados do pacote (constante, igual a oito) e os campos *serviceH* e *serviceL* que possuem a parte alta e a parte baixa da palavra que identifica o serviço de monitoramento. Já para os seis campos que compõem o corpo de dados tem-se: *source,* que identifica qual é o roteador gerador do pacote de controle e *peast, pwest, pnorth, psouth e plocal,* que contém a informação da quantidade de *flits* que passaram pelas portas *leste*, *oeste*, *norte*, *sul* e *local* do roteador gerador do pacote de controle. Esses pacotes são enviados de maneira periódica de cada roteador, sendo este período uma janela de tempo definida pelo projetista da rede.

A Figura 6 ilustra a máquina de transição de estados que controla a transmissão dos pacotes de controle. A máquina é inicializada através do sinal *reset*. O estado inicial chama-se *Sidle*. Nesse estado ocorre a inicialização dos sinais:

- *rx_ctrl_out*, indica o início do envio do pacote de controle;
- *enable_ctrl*, habilitação para o envio de pacote de controle, e;
- *clk_count*, contador para o gerenciamento do envio de um novo pacote de controle.

O segundo estado é o *Sstart*. A máquina permanece nesse estado enquanto a janela de tempo não chegar ao valor pré-definido. No estado seguinte, *Stx*, ocorre a verificação se a porta *local* está ou não ocupada recebendo dados de um IP. Se a porta *local* estiver desocupada *(rx_ctrl_in='0')* a máquina avança para o próximo estado, *Senable*. Nesse estado, o sinal *trigger* é ativado (*trigger = '1'*), através desse sinal os monitores param a contagem de quantidade de *flits*. Ainda em *Senable* o sinal *enable_ctrl* é ativado. No estado seguinte, *Starget*, o envio do pacote é inicializado, ativando-se o sinal *rx_ctrl_out*.

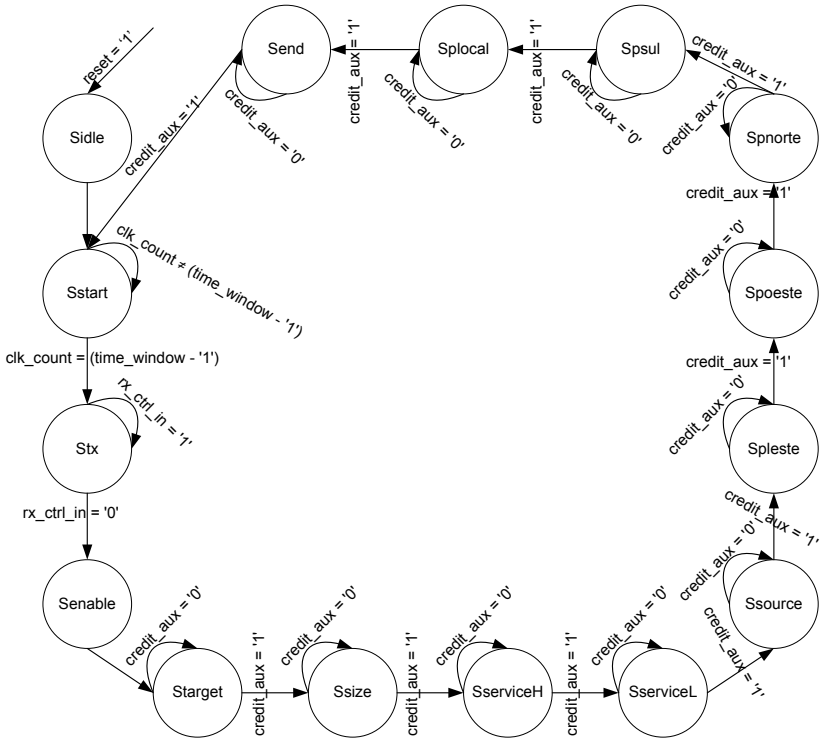


Figura 6 - Máquina de transição de estados que controla a transmissão dos pacotes de controle gerados pelo GPC.

A partir do estado *Starget* até o estado *Send* há um controle de fluxo para garantir que o pacote seja transmitido corretamente, sem perdas de dados ao longo da transmissão. O controle de fluxo é realizado pelo sinal de saída da fila *Flocal*, sinal *credit_aux*. Esse sinal indica se a fila está ou não apta a receber dados. É importante destacar na Figura 6 que esse é o sinal que controla a transição dos estados. Do estado *Starget* até o estado *Splocal* ocorre a transmissão dos dez *flits* que compõem o pacote de controle. Após o estado *Splocal* a máquina avança para o estado *Send*. Nesse estado os sinais de controle de habilitação para o envio do pacote, *rx_ctrl_out* e *enable_ctrl* são desabilitados.

Na forma de onda apresentada na Figura 7 é possível verificar o funcionamento correto da máquina de estados que controla a transmissão dos pacotes de controle. É destacada na forma de onda a transição de nível lógico de três sinais importantes no controle do envio dos pacotes, *rx_ctrl_in*, *enable_ctrl* e *rx_ctrl_out*. Em cada um dos destaques da Figura é possível verificar que:

(1) o sinal *rx_ctrl_in* ao passar do nível lógico um para o zero a máquina passa do estado *Stx* para o *Senable*;

(2) o sinal *enable_ctrl* tem seu nível lógico alterado de zero para um, indicando que o pacote pode ser transmitido, e;

(3) o sinal *rx_ctrl_out* ao ser ativado é inicializado o envio do pacote de controle. O início do envio do pacote pode ser observado através do sinal *data_ctrl_out*. Nota-se que o sinal *data_ctrl_out* estava com valor zero até o momento em que o sinal *rx_ctrl_out* foi ativado.

Após esta ativação o sinal *data_ctrl_out* passou a receber os valores correspondentes a cada campo do pacote.

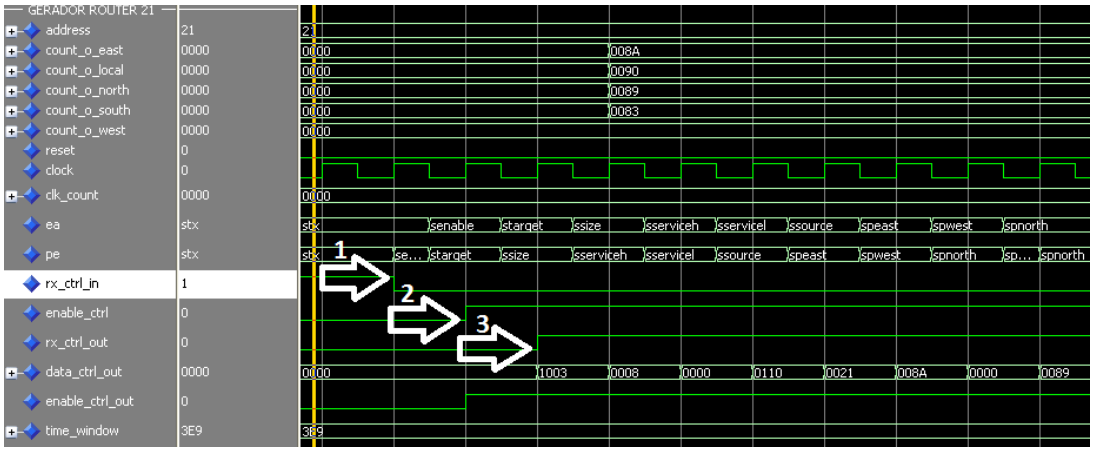


Figura 7 - Funcionamento da máquina de estados que controla a transmissão dos pacotes de controle.

Na Figura 8, observa-se a montagem correta do pacote de controle. Em destaque na Figura, é possível ver os dez campos que compõem o pacote bem como seus respectivos valores conforme apresentado na Figura 5 (página 43). Outro destaque dessa Figura é a quantidade de *flits* registrada nos monitores. Percebe-se a que esses valores preenchem os últimos cinco campos do pacote, ou seja, os valores correspondentes as informações monitoradas.

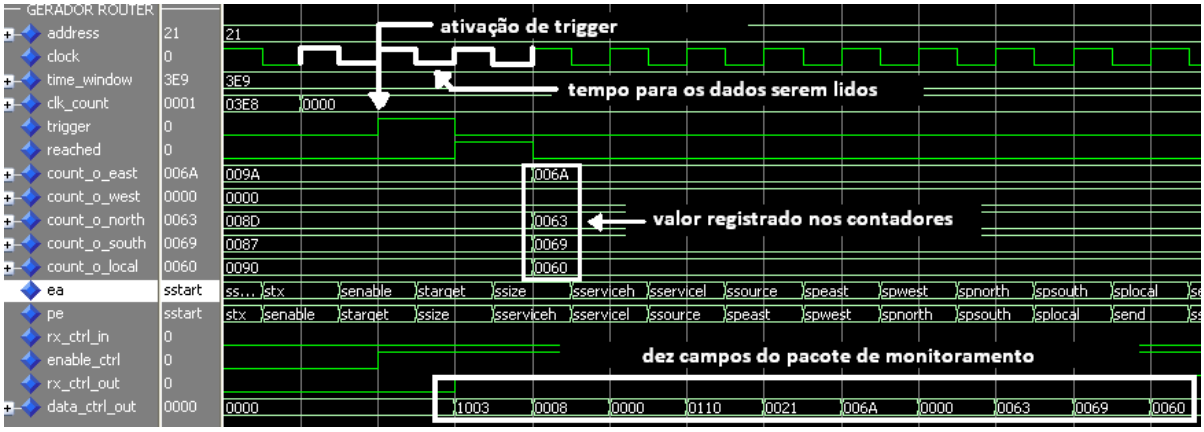


Figura 8 - Dados lidos dos monitores após ativação do sinal trigger e montagem correta dos pacotes de controle.

É importante destacar que os dados de monitoramento são enviados apenas quando a janela de tempo atingir o tempo pré-determinado **e** se o IP conectado à porta local do roteador responsável pela geração dos dados de monitoramento não estiver injetando dados na rede. Caso o IP esteja injetando dados na rede e a janela de tempo tenha atingido o tempo definido, os monitores não param de contar a quantidade de flits. Portanto cria-se o sinal trigger, que é responsável pelo controle de leitura da contagem dos monitores, e o respectivo zeramento dos mesmos. Na Figura 8, pode-se verificar que os dados de monitoramento são enviados após a ativação do sinal *trigger* (*trigger = '1'*).

4 UTILIZAÇÃO DOS DADOS DE MONITORAMENTO NA PLATAFORMA HeMPS

Este Capítulo corresponde à segunda contribuição da Dissertação, ou seja, o desenvolvimento de um sistema para recepção e tratamento dos dados monitorados no MPSoC HeMPS [WOS07]. O restante deste Capítulo apresenta como é realizada a integração da estrutura de monitoramento com a plataforma MPSoC.

4.1 Plataforma HeMPS

Neste trabalho é utilizado o MPSoC HeMPS [WOS07], uma plataforma homogênea que emprega processadores Plasma [PLA01] interconectados pela NoC HERMES [MOR04]. A HeMPS possui uma arquitetura de processadores mestre-escravo, e uma infra-estrutura de hardware e software. A Figura 9 ilustra uma instância do MPSoC HeMPS utilizando uma NoC com dimensões 2 x 3 para interconectar cinco processadores Plasma-IP SL (processador escravo) e um processador Plasma-IP MP (processador mestre).

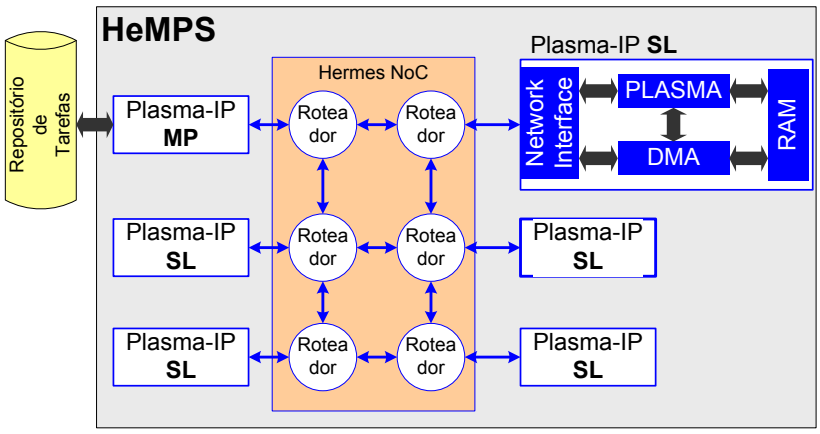


Figura 9 - Instância da HeMPS utilizando uma NoC com dimensões 2 x 3 para interconectar seis processadores Plasma-IP .

Diversas aplicações, sejam elas com a mesma ou diferentes finalidades podem executar simultaneamente em um MPSoC. Caso novas aplicações possam ser inseridas no sistema em tempo de execução, caracteriza-se uma *carga dinâmica de trabalho*. Uma forma do MPSoC suportar novas aplicações em tempo de execução é ter um número de recursos de processamento igual ao somatório das tarefas de todas as aplicações, o que freqüentemente não ocorre. Uma forma de minimizar o custo do sistema é através do uso de mapeamento dinâmico de tarefas, onde apenas a tarefa inicial de cada aplicação é mapeada no sistema [CAR09]. A plataforma HeMPS assume que todas as aplicações são modeladas através de um grafo de tarefas, em que somente as tarefas iniciais são carregadas no sistema no momento da inicialização do mesmo. As demais tarefas são

armazenadas em uma memória externa, chamada de *repositório de tarefas*, sendo inseridas dinamicamente no sistema em função das requisições de comunicação e dos recursos disponíveis.

Os processadores escravos (Plasma-IP SL) executam um *microkernel* que suporta os serviços de execução multi-tarefa e comunicação entre tarefas. O *microkernel* segmenta a memória em páginas, alocando a si próprio a primeira página e utilizando as demais para a alocação de tarefas. Cada Plasma-IP tem uma tabela de tarefas com a localização das tarefas locais e remotas.

O processador mestre (Plasma-IP MP) é responsável pelo gerenciamento dos recursos do sistema. Este processador não executa tarefas de aplicações, e é o único a ter acesso ao repositório de tarefas, visando reduzir o tráfego de dados na NoC e aumentar o desempenho do sistema. Suas principais funções são a alocação das tarefas iniciais e o mapeamento de novas tarefas demandadas em tempo de execução. Além disso, também verifica o fim da execução de uma tarefa e a liberação de recursos no sistema, bem como recepciona mensagens de controle, tais como finalização de tarefas e *debug* de pacotes. As informações de monitoramento são tratadas neste processador.

4.2 Detalhes do Módulo Plasma-IP

O módulo DMA (*Direct Memory Access*), contido no Plasma-IP é responsável unicamente por transferir os códigos objeto das tarefas que chegam à NI para a memória do processador escravo local, permitindo que o processador realize a execução das suas tarefas em paralelo com a recepção de novas tarefas.

Visando facilitar a avaliação de desempenho do sistema, foi criado um registrador denominado *tick_counter*. Este registrador acumula ciclos de relógio durante a execução do sistema e pode ser lido pelo *microkernel* ou por aplicações através de uma chamada de sistema. Um contador, denominado *time_slice*, é criado para controlar uma determinada fatia de tempo durante a qual uma tarefa é executada.

Dentro do módulo Plasma-IP é incluída uma interface de rede, que realiza a interconexão entre o processador Plasma e a NoC, sendo responsável por: (*i*) enviar e receber pacotes da rede; e (*ii*) repassar o código objeto de tarefas recebido da rede, através do DMA, para a memória e informar ao *microkernel* qual a sua localização na rede.

Ao receber um pacote da rede, a NI interrompe o processador para que este receba os dados. Um pacote que trafega na rede possui o formato: <*target*><*size*><*payload*>, onde *target* indica o

destino do pacote e *size* indica o tamanho do conteúdo do pacote. O campo *payload* é formado por: <*service*><*service_parameters*>, onde *service* indica o serviço solicitado e *service_parameters* são os parâmetros necessários ao serviço, podendo ser diferentes em número e forma para cada serviço. Através do código do serviço o *microkernel* decide qual ação tomar após ter recebido o pacote. Os serviços que um pacote pode carregar juntamente com seus códigos de identificação são descritos na Tabela 3.

A Tabela 3 lista os serviços de comunicação entre processador mestre e escravo, para:

- alocação estática e dinâmica de tarefas;
- término da alocação estática de tarefas;
- comunicação entre tarefas;
- término de execução de tarefa, e;
- depuração.

Tabela 3 - Descrição dos serviços que um pacote pode carregar.

Serviço	Código
MESSAGE_REQUEST: solicitação de uma mensagem por parte do processador escravo	0x00000010
MESSAGE_DELIVERY: entrega de uma mensagem previamente solicitada	0x00000020
MESSAGE_UNAVAILABLE: aviso de que a mensagem solicitada não existe	0x00000030
TASK_ALLOCATION: alocação de tarefa	0x00000040
TASK_ ALLOCATED: aviso de que uma nova tarefa está alocada no sistema	0x00000050
TASK_ REQUEST: requisição de uma tarefa por parte do processador escravo para o nodo mestre (alocação dinâmica)	0x00000060
TASK_ TERMINATED: aviso por parte do processador escravo para o nodo mestre de que uma tarefa terminou a sua execução	0x00000070
TASK_ DEALLOCATED: aviso por parte do nodo mestre de que uma tarefa terminou a sua execução e pode ser liberada	0x00000080
FINISHED_ALLOCATION: aviso que o nodo mestre terminou a alocação inicial das tarefas (alocação estática)	0x00000090
DEBUG_MESSAGE: mensagem de depuração	0x00000100

Em uma alocação estática, o processador mestre, através do serviço TASK_ALLOCATION, informa a um determinado processador escravo que uma tarefa deve ser alocada no mesmo. Após a alocação da tarefa, o processador mestre informa através do serviço TASK_ ALLOCATED os outros processadores escravos que fazem parte do sistema, exceto o escravo com a tarefa alocada, de que a tarefa foi alocada em um determinado processador (Figura 10(a)). Na alocação dinâmica diferentemente da alocação estática, é usado um serviço adicional, TASK_ REQUEST, este enviado pelo processador escravo ao mestre solicitando uma tarefa (Figura 10(b)). Através do serviço

FINISHED_ALLOCATION o processador mestre informa aos outros processadores do sistema que a alocação estática de todas as tarefas terminou (Figura 10(c)). Os serviços MESSAGE_UNAVAILABLE e MESSAGE_DELIVERY são respostas do serviço MESSAGE_REQUEST, e são utilizados na comunicação entre tarefas remotas (Figura 10(d)). Quando um processador escravo termina a execução de uma tarefa é necessário que um aviso informando o término da execução seja feito ao processador mestre, aviso este feito através do serviço TASK_TERMINATED. O mesmo ocorre por parte do mestre, que deve informar aos escravos, exceto ao que lhe enviou o serviço, que a tarefa pode ser liberada, informação essa transmitida pelo serviço TASK_DEALLOCATED (Figura 10(e)). O serviço DEBUG_MESSAGE permite aos usuários da plataforma HeMPS analisar simulações de aplicações que utilizem este serviço (Figura 10(f)).

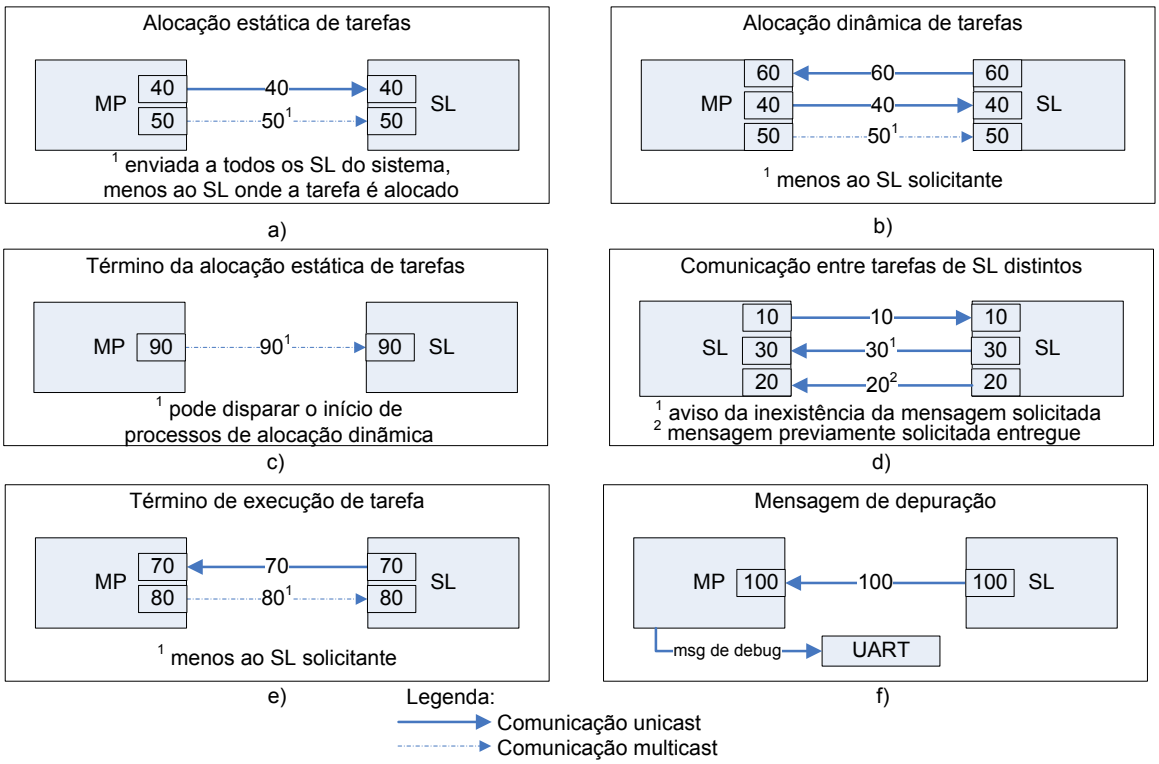


Figura 10 - Comunicação entre processador mestre e escravo através de serviços. Os números indicados nas setas de comunicação correspondem aos serviços definidos na Tabela 3.

4.3 Integração da Estrutura de Monitoramento no Microkernel do Plasma-IP MP

A estrutura de monitoramento implementada no presente trabalho está inserida no *microkernel* do processador mestre do MPSoC HeMPS na forma de um serviço que recebe e identifica pacotes de controle, coletando seus dados e os armazenando em estruturas de controle. Uma nova função é destinada ao processador mestre, na qual este é o responsável pela montagem das estruturas de dados que armazenam o estado do sistema, sendo utilizadas para o controle da

ocupação dos canais da rede e dos processadores. Portanto, para o processador mestre realizar a nova função, fez-se alterações em sua estrutura. O Plasma-IP MP é o responsável, assim, pelo sistema de controle (MSA).

A funcionalidade do novo serviço adicionado no *microkernel* do processador mestre é detectar e tratar os pacotes de controle provenientes dos processadores escravos. Este serviço é denominado *DEBUG_MONITORING* e possui o seguinte código de identificação: 0x00000110. Ao detectar o recebimento de um pacote de controle, através do campo *service* que contém o código do serviço *DEBUG_MONITORING*, o *microkernel* deve tratar as informações de monitoramento contidas nos pacotes. Lembrar, que o campo *payload* dos pacotes que trafegam pela rede, é formado pelos campos: <*service*><*service_parameters*>. As informações a serem tratadas estão contidas no campo *service_parameters*, que é preenchido com o *payload* do pacote de controle, ilustrado na Figura 5 (página 36). O tratamento dessas informações é realizado através da função *DebugMonitoring( )* implementada no ***microkernel*** do processador mestre, sendo o pseudo-código da função apresentado na Figura 11.

```
1. x = NI_Read();
2. east = x & 0x0000FFFF;
3. source = x>>16;
4. l = source & 0x000F;
5. c = (source & 0x00FF)>>4;
6. x = NI_Read();
7. north = x & 0x0000FFFF;
8. west = x>>16;
9. x = NI_Read();
10.local = x & 0x0000FFFF;
11.south = x>>16;
```

Figura 11 – Pseudo-código da função *DebugMonitoring( )*, a qual realiza o tratamento das informações de monitoramento.

A interface de rede (NI) ao receber um pacote realiza as seguintes ações: montagem das palavras de 32 *bits* a partir dos *flits* de 16 *bits*, e geração de uma interrupção para o processador. O processador ao ser interrompido pela NI executa a função responsável por tratar o pacote recebido. A primeira ação é identificar o serviço contido no pacote. Caso o serviço seja de monitoramento, a função contida no pseudo-código da Figura 11 é executada.

A linha 1 realiza a primeira leitura de palavra da NI, correspondendo aos *flits source* e *peast* (a NI os agrupa em uma palavra de 32 *bits*). A linha 2 isola o número de *flits* da porta leste (*east*). Na linha 3 é obtido o endereço do roteador que gerou o pacote recebido (*source*). As linhas 4 e 5

definem a partir da variável *source*, o valor para a linha (*l*) e coluna (*c*), que serão utlizados para identificar o roteador que gera o pacote durante o preenchimento da estrutura de dados. Na sequência há duas novas leituras de palavra da NI (linhas 6 e 9), para se obter os valores dos demais monitores (linhas 7-8 e 10-11).

4.4 Tratamento dos Pacotes de Monitoramento

As estruturas de dados geradas para tratar os dados monitorados é implementada através de dois conjuntos de matrizes, baseadas em [CAR09].

O primeiro conjunto de matrizes armazena a **quantidade total** de *flits* recebida em cada canal ao longo da execução das aplicações. O conteúdo destas matrizes é incrementado a cada novo pacote de monitoramento recebido. Cinco matrizes compõem este conjunto: *East Channels Total* (ECT), matriz contendo a ocupação dos canais com direção para leste; *West Channels Total* (WCT); *North Channels Total* (NCT); *South Channels Total* (SCT), e; *Local Port Total (*LPT).

O segundo conjunto de matrizes armazena a **quantidade instantânea** de *flits* recebida em cada canal, a cada novo pacote de monitoramento recebido. Cinco matrizes compõem este conjunto: *East Channels RCV* (ECR), matriz contendo a ocupação dos canais com direção para leste; *West Channels RCV* (WCR); *North Channels RCV* (NCR); *South Channels RCV* (SCR), e; *Local Port RCV (*LPR).

Nestas matrizes, as posições de coluna *c* e linha *i* são obtidas pelo pseudo-código apresentado na Figura 11. Essas 10 matrizes são inseridas no *microkernel* do processador mestre, e devem ser atualizadas em tempo de execução para que, baseado nas informações de ocupação dos recursos, o processador mestre possa definir, por exemplo, a melhor posição de mapeamento de tarefas. A Tabela 4 ilustra as matrizes para armazenamento total de *flits*. As matrizes RCV possuem estrutura análoga.

A atualização das matrizes é implementada na função *DebugMonitoring( )*. O pseudo-código apresentado na Figura 12, continuação do pseudo-código apresentado na Figura 11, demonstra como os dados monitorados são armazenados nas cinco matrizes Total e RCV.

Tabela 4 - Matrizes que representam a ocupação dos canais de cada porta do roteador.

Matriz		Informação de ocupação
East Channels Total (ECT)	$\begin{bmatrix} EC_{0,l-1} & \cdots & EC_{c-2,l-1} \\ \vdots & & \vdots \\ EC_{0,1} & \cdots & EC_{c-2,1} \\ EC_{0,0} & \cdots & EC_{c-2,0} \end{bmatrix}$	Canais para Leste (LC)
West Channels Total (WCT)	$\begin{bmatrix} WC_{0,l-1} & \cdots & WC_{c-2,l-1} \\ \vdots & & \vdots \\ WC_{0,1} & \cdots & WC_{c-2,1} \\ WC_{0,0} & \cdots & WC_{c-2,0} \end{bmatrix}$	Canais para Oeste (WC)
North Channels Total (NCT)	$\begin{bmatrix} NC_{0,l-2} & NC_{1,l-2} & \cdots & NC_{c-1,l-2} \\ \vdots & \vdots & & \vdots \\ NC_{00} & NC_{10} & \cdots & NC_{c-10} \end{bmatrix}$	Canais para Norte (NC)
South Channels Total (SCT)	$\begin{bmatrix} SC_{0,l-2} & SC_{1,l-2} & \cdots & SC_{c-1,l-2} \\ \vdots & \vdots & & \vdots \\ SC_{00} & SC_{10} & \cdots & SC_{c-10} \end{bmatrix}$	Canais para Sul (SC)
Local Port Total (LPT)	$\begin{bmatrix} LP_{0,l-1} & LP_{1,l-1} & \cdots & LP_{c-1,l-1} \\ \vdots & \vdots & & \vdots \\ LP_{0,1} & LP_{1,1} & \cdots & LP_{c-1,1} \\ LP_{0,0} & LP_{1,0} & \cdots & LP_{c-1,0} \end{bmatrix}$	Porta Local (LP)

```
1. east_channel_RCV[c][l] = east;
2. west_channel_RCV[c][l] = west;
3. north_channel_RCV[c][l] = north;
4. south_channel_RCV[c][l] = south;
5. local_channel_RCV[c][l] = local;
6. east_channel_TOTAL[c][l] = east_channel_TOTAL[c][l] + east;
7. west_channel_TOTAL[c][l] = west_channel_TOTAL[c][l] + west;
8. north_channel_TOTAL[c][l] = north_channel_TOTAL[c][l] + north;
9. south_channel_TOTAL[c][l] = south_channel_TOTAL[c][l] + south;
10.local_channel_TOTAL[c][l] = local_channel_TOTAL[c][l] + local;
```

Figura 12 - Pseudo-código demonstrando o armazenamento dos dados monitorados em cada uma das cinco matrizes Total e RCV.

A posição da coluna e linha identifica qual é o roteador que enviou o pacote de controle que esta sendo tratado e as variáveis *east*, *west*, *north*, *south* e *local* armazenam o número de *flits* das cinco portas desse roteador. Da linha 1 até a linha 5 é realizado o preenchimento das matrizes RCV, de acordo com a posição coluna *c* e linha *l*. O preenchimento das matrizes Total com os dados monitorados é realizado das linhas 6 até a linha 10.

A janela de tempo que é definida pelo projetista da rede (t_{janela}) para geração dos pacotes de controle deve ser calculada de tal forma que o processador mestre não seja sobrecarregado com

processamento de pacotes de controle. Assume-se como carga máxima um valor de 10% da carga do processador mestre para tratamento de pacotes de monitoramento. A Equação 1 pode ser utilizada para definir o valor da janela de tempo em ciclos de relógio.

$$t_{janela} = \left(nucleos_{qtd} - 1\right) * t_{tratamento} * \frac{100}{ocup} \qquad (1)$$

Onde: $nucleos_{qtd}$ corresponde à quantidade de núcleos que compõem o MPSoC; $t_{tratamento}$ é o tempo consumido para tratar o recebimento de pacotes de controle, em ciclos de relógio; *ocup* é o valor de ocupação do processador mestre para tratamento de pacotes de controle. Por exemplo, assumindo-se:

- *ocup* = 10%, ou seja, 10% de carga máxima do processador mestre;
- um MPSoC com 9 núcleos, ou seja, rede 3 x 3, sendo que um dos núcleos é o responsável por tratar as informações monitoradas e portando não é monitorado, e;
- $t_{tratamento}$ = 105, ou seja, os pseudo-códigos da Figura 11 e Figura 12 consomem 105 ciclos de relógio para execução, incluindo o tratamento de interrupções.

Ao aplicar os valores mencionados na Equação 1 obtém-se uma janela de tempo igual a 8400 ciclos de relógio ($t_{janela} = (9 - 1) * 105 * \frac{100}{10} = 8400$). Ao se calcular o valor da janela, cabe ao projetista da rede informar, em tempo de projeto, seu valor.

5 RESULTADOS

Neste Capítulo são apresentados os experimentos realizados para validar o correto funcionamento dos monitores implementados, bem como a estrutura de monitoramento integrada com o *microkernel* do processador mestre do MPSoC.

5.1 Avaliação da Rede HERMES

Os monitores implementados nas portas de entradas dos roteadores da NoC HERMES foram validados através de simulação. Três cenários de teste foram gerados utilizando a ferramenta ATLAS [ATL07], uma plataforma desenvolvida para geração automática da rede HERMES através de parâmetros definidos pelo usuário, tais como dimensões da rede, tamanho dos *buffers*, largura dos *flits*, algoritmo de roteamento, entre outros. Ao final da simulação de cada um dos três cenários espera-se calcular as taxas de recepção dos pacotes na rede. A Figura 13 apresenta a interface da plataforma ATLAS para a geração das redes dos cenários de teste. Os parâmetros adotados para a rede dos cenários são:

- Dimensão da rede: 4x4 roteadores;
- Tamanho do *flit* de 16 *bits*;
- Controle de fluxo baseado em créditos;
- Sem canais virtuais;
- Profundidade das filas de 8 *bits*, e;
- Algoritmo de roteamento XY.

Os três cenários de teste utilizam a mesma configuração de rede e freqüência de operação de 50 MHz. Cada um dos cenários é gerado de modo que as taxas de injeção atinjam 10%, 20% e 25% da taxa máxima da rede. O cálculo da taxa máxima (*taxamax*) de injeção da rede é dado pela Equação 2.

$$taxamax = \frac{tamflit}{perclock} \qquad (2)$$

Onde *tamflit* é o tamanho do *flit* em *bits* e *perclock* é o período do relógio da rede em segundos. Aplicando-se na Equação 2 os valores do tamanho do *flit* e o período do relógio obtém-se uma taxa máxima de injeção por canal igual a 800 Mbps.

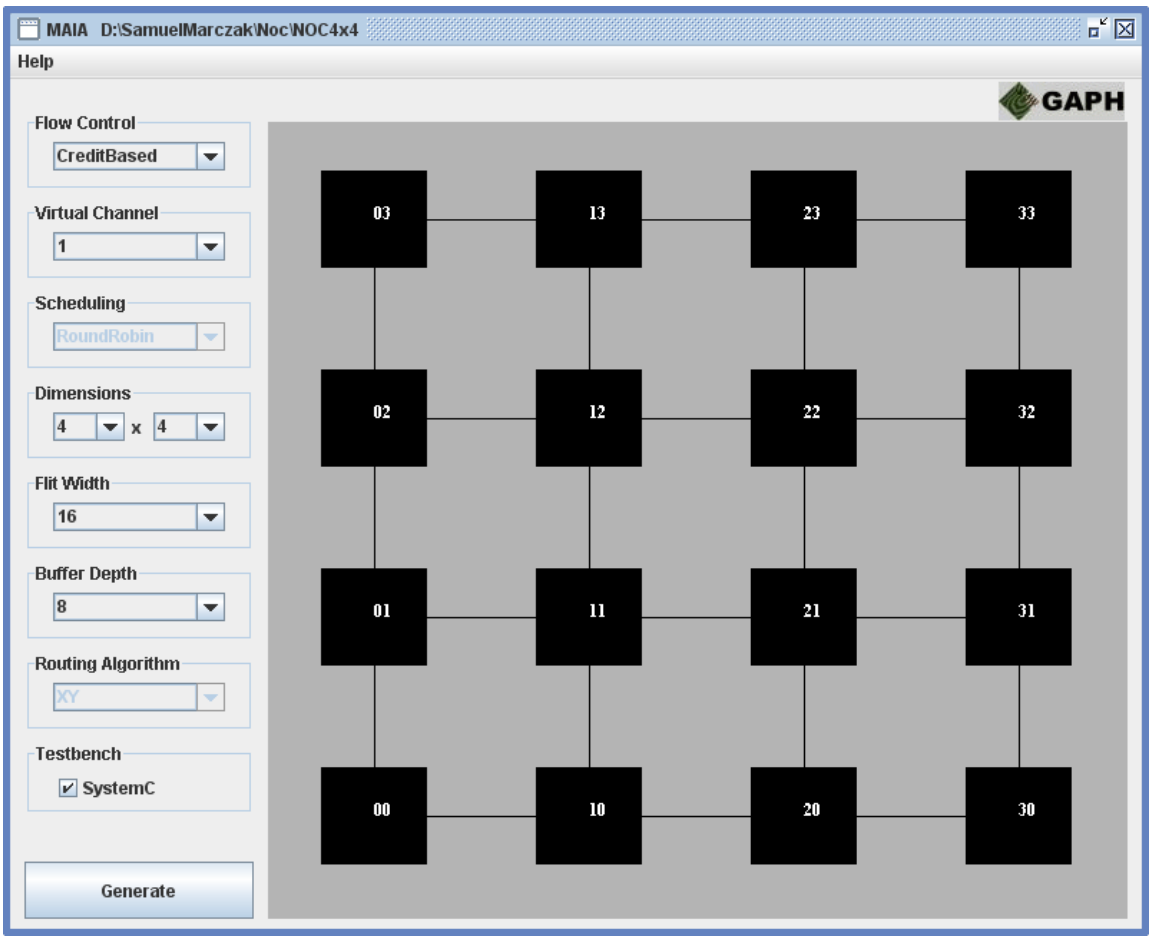


Figura 13 - Interface da plataforma ATLAS para a geração das redes utilizadas na validação dos monitores.

Através das informações coletadas pelos monitores é possível calcular as taxas de recepção dos pacotes para cada uma das cinco portas de entrada dos roteadores. A taxa de recepção dos pacotes (*taxarecep*) é dada pela Equação 3.

$$taxarecep = \frac{qtdflits * perclock}{t_{amostragem}} \qquad (3)$$

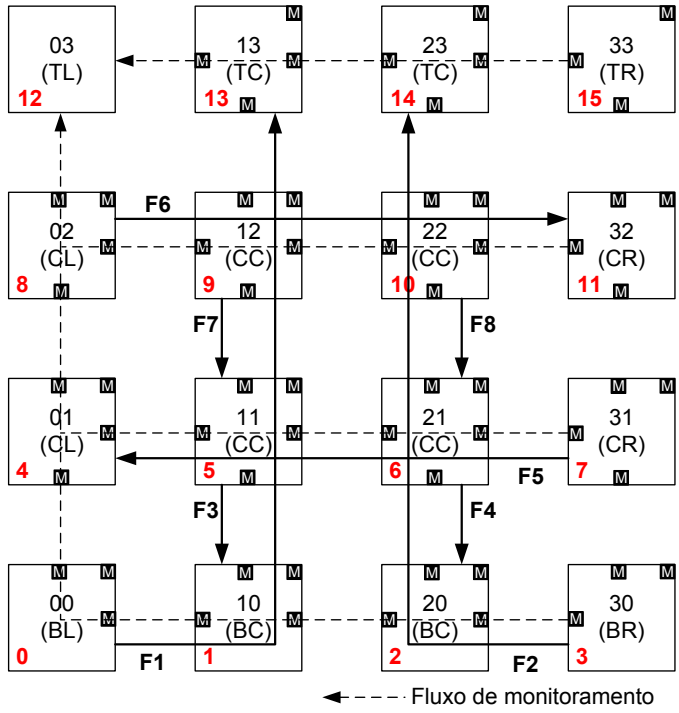
Onde *qtdflits* é a quantidade de *flits* computada pelo monitor em um determinado intervalo de tempo de simulação, *perclock* é o período do relógio da rede e $t_{amostragem}$ é a janela de tempo de amostragem.

O Cenário1 e o Cenário2 possuem a mesma distribuição espacial (apresentada na Figura 14(a)), diferenciando-se apenas na taxa de injeção, de 10% e 20% respectivamente. O Cenário3 possui uma distribuição espacial (apresentada na Figura 15(b)) diferente dos Cenários1 e 2 e uma taxa de injeção de 25%. Para os três cenários utilizam-se 2000 pacotes para cada fluxo, e tamanho do pacote de 48 *flits* e a distribuição temporal uniforme.

A Figura 14(b) ilustra o tráfego dos Cenário1 e 2. Em ambos cenários há monitores inseridos

em todos os roteadores da rede, exceto no roteador responsável por receber os dados monitorados, denominado TL (*Top Left*). A Figura 14(a) detalha as origens e destinos de cada fluxo para estes cenários. Nestes dois cenários não há fluxos de pacotes concorrentes, logo as taxas de recepção dos pacotes medidas devem ser iguais as taxas de injeção.

Fluxo Pacotes	Roteador	
	Origem	Destino
F1	00	13
F2	30	23
F3	11	10
F4	21	20
F5	31	01
F6	02	32
F7	12	11
F8	22	21



(a) (b)

Figura 14 – (a) Fluxo de pacotes do Cenário1 e Cenário2; (b)Tráfegos do Cenário1 e Cenário2.

Dado que o objetivo dos experimentos propostos é de validar os monitores, o intervalo entre pacotes de controle é fixo em 1ms, sem preocupação com a carga de tratamento destes pacotes no processador mestre (roteador TL).

A Tabela 5 apresenta as taxas de recepção dos pacotes para o Cenário1, obtidas ao aplicar-se a Equação 3 para um intervalo de tempo de simulação igual a 1ms ($t = 1ms$). Os resultados apresentados na Tabela demonstram que os monitores computaram corretamente as taxas de injeção dos pacotes igual a 80 Mbps, 10% da taxa máxima da rede. Nota-se ainda nos resultados apresentados que é possível observar a carga induzida pelos monitores. Por exemplo, a porta leste do roteador 9 recebe apenas dados de monitoramento provenientes do roteador 10. A taxa medida neste monitor é igual a 1,1%, essa a taxa referente aos pacotes de controle.

Tabela 5 - Taxa de recepção dos pacotes para o Cenário1.

Taxa de Recepção dos Pacotes (%)																				
Roteador	Porta	Tempo de Simulação (ms)																		
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
5	Leste	11,1	11,1	11,1	11,1	11,1	11,1	11,1	11,1	11,1	11,1	11,1	11,1	11,1	11,1	11,1	11,1	11,1	11,1	9,43
	Oeste	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Norte	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	8,35
	Sul	10	9,98	9,98	9,98	10	9,98	9,98	9,98	10	9,98	9,98	9,98	10	9,98	9,98	9,98	10	9,98	8,35
	Local	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	8,35
6	Leste	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	8,35
	Oeste	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Norte	10	9,98	9,98	9,98	10	9,98	9,98	9,98	10	9,98	9,98	9,98	10	9,98	9,98	9,98	10	9,98	8,35
	Sul	10	9,98	9,98	9,98	10	9,98	9,98	9,98	10	9,98	9,98	9,98	10	9,98	9,98	9,98	10	9,98	8,35
	Local	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	8,35
9	Leste	**1,08**	**1,09**	**1,1**	**1,09**	**1,08**	**1,09**	**1,1**	**1,09**	**1,08**	**1,09**	**1,1**	**1,09**	**1,08**	**1,09**	**1,1**	**1,09**	**1,08**	**1,09**	**1,1**
	Oeste	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	8,35
	Norte	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Sul	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	8,35
	Local	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	8,35
10	Leste	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Oeste	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	8,35
	Norte	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Sul	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	8,35
	Local	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	9,98	9,98	10,1	9,98	8,35

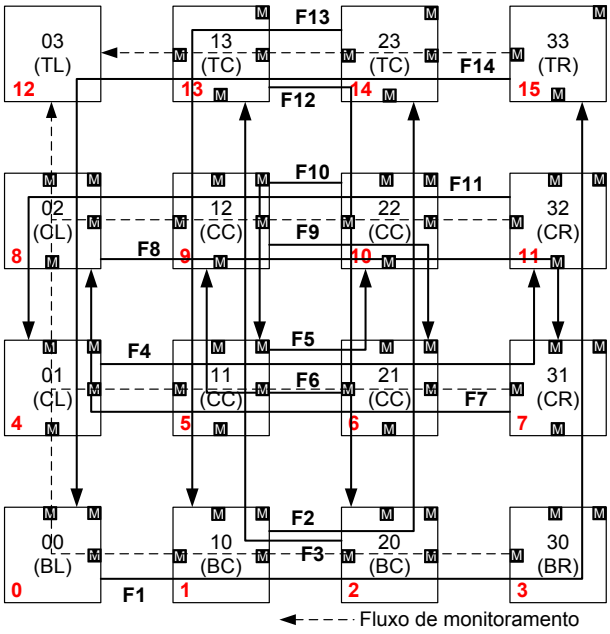
A Tabela 6 apresenta os resultados das taxas de recepção dos pacotes obtidas na simulação do Cenário2. Nesse cenário a taxa de injeção dos pacotes é igual a 20% da taxa máxima da rede, ou seja, 160 Mbps. Os resultados apresentados na Tabela demonstram que através dos monitores é possível calcular as taxas de recepção dos pacotes. Nota-se que o tempo de simulação para esse cenário é a metade do tempo em relação ao Cenário1, isso ocorre em função do Cenário2 ter a taxa de injeção dos pacotes duas vezes maior que a do Cenário1. A carga inserida pelos monitores manteve-se a mesma, em torno de 1%.

Tabela 6 - Taxa de recepção dos pacotes para o Cenário2.

Taxa de Recepção dos Pacotes (%)																	
Roteador	Porta	Tempo de Simulação (ms)															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
5	Leste	21,12	21,01	21,12	21,01	21,12	21,01	21,12	21,01	21,12	9,45	1,08	1,08	1,08	1,1	1,08	1,08
	Oeste	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Norte	20,06	19,94	20,06	19,94	20,06	19,94	20,06	19,94	20,06	8,35	0	0	0	0	0	0
	Sul	20,06	19,93	20,06	19,93	20,06	19,93	20,06	19,93	20,06	8,35	0	0	0	0	0	0
	Local	20,06	19,92	20,06	19,92	20,06	19,92	20,06	19,92	20,06	8,35	0	0	0	0	0	0
6	Leste	20,06	19,97	20,06	19,97	20,06	19,97	20,06	19,97	20,06	8,35	0	0	0	0	0	0
	Oeste	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Norte	20,03	19,97	20,03	19,97	20,03	19,97	20,03	19,97	20,03	8,35	0	0	0	0	0	0
	Sul	20,02	19,97	20,02	19,97	20,02	19,97	20,02	19,97	20,02	8,35	0	0	0	0	0	0
	Local	20,05	19,97	20,05	19,97	20,05	19,97	20,05	19,97	20,05	8,35	0	0	0	0	0	0
9	Leste	1,08	1,09	1,08	1,09	1,08	1,09	1,08	1,09	1,08	1,09	1,10	1,09	1,1	1,09	1,09	1,09
	Oeste	20,06	19,97	20,07	19,97	20,06	19,97	20,06	19,97	20,06	8,35	0	0	0	0	0	0
	Norte	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Sul	20,06	19,97	20,07	19,97	20,06	19,97	20,06	19,97	20,06	8,35	0	0	0	0	0	0
	Local	20,06	19,97	20,07	19,97	20,06	19,97	20,06	19,97	20,06	8,35	0	0	0	0	0	0
10	Leste	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Oeste	20,06	19,97	20,07	19,97	20,06	19,97	20,06	19,97	20,06	8,35	0	0	0	0	0	0
	Norte	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Sul	20,06	19,97	20,07	19,97	20,06	19,97	20,06	19,97	20,06	8,35	0	0	0	0	0	0
	Local	20,04	19,97	20,04	19,97	20,04	19,97	20,04	19,97	20,04	8,35	0	0	0	0	0	0

Na Figura 15(b) é ilustrado o tráfego do Cenário3 e a Figura 15(a) detalha as origens e destinos de cada fluxo para este cenário. Em todos os roteadores da rede são inseridos monitores, exceto no roteador TL, responsável por receber os dados monitorados. Nesse cenário há fluxos de pacotes concorrentes, portanto as taxas de recepção dos pacotes medida devem ser iguais ao somatório das taxas dos fluxos concorrentes. Um exemplo de fluxos concorrentes neste cenário pode ser observado na porta leste do roteador 9, fluxos F10 e F11.

Fluxo Pacotes	Roteador	
	Origem	Destino
F1	00	33
F2	10	23
F3	20	13
F4	01	32
F5	11	22
F6	21	12
F7	31	02
F8	02	31
F9	12	21
F10	22	11
F11	32	01
F12	13	20
F13	23	10
F14	33	33



(a) **(b)**

Figura 15 – (a) Fluxo de pacotes do Cenário3; (b)Tráfegos do Cenário3.

Os resultados obtidos a partir da simulação do Cenário3, aplicados na Equação 3 com um intervalo de tempo de simulação igual a 1 ms, são apresentados na Tabela 7. Nesse cenário a taxa de injeção dos pacotes é igual a 25% da taxa máxima da rede, ou seja, é igual a 200 Mbps. Analisando-se em especial a porta leste do roteador 9, nota-se que os valores das taxas são em torno de 50%. Esses valores comprovam a recuperação das taxas para o cenário, visto que nesse há concorrência entre os fluxos de pacotes F10 e F11.

Os experimentos realizados demonstram que a implementação dos monitores está correta. Através das simulações realizadas para cada um dos cenários apresentados, foi possível calcular as taxas de recepção dos pacotes, sendo assim tem-se a validação dos monitores. Nas simulações também foi possível verificar que a vazão original dos pacotes da rede sofre um acréscimo em torno de 1,1%, esse referente aos pacotes de controle.

Tabela 7 - Taxa de recepção dos pacotes para o Cenário3.

Taxa de Recepção dos Pacotes (%)																	
Roteador	Porta	Tempo de Simulação (ms)															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
5	Leste	51,17	51	51,03	51,06	51,01	51,07	51,18	26,04	1,08	1,1	1,08	1,08	1,08	1,1	1,08	1,08
	Oeste	25,06	24,95	25	25,01	24,97	25,03	25,05	12,48	0	0	0	0	0	0	0	0
	Norte	50,11	49,94	49,97	50	50	50	50	24,98	0	0	0	0	0	0	0	0
	Sul	25,06	24,93	24,99	25,01	24,97	25,03	25,05	12,48	0	0	0	0	0	0	0	0
	Local	25,06	25,05	24,95	25,03	24,96	24,93	25,04	12,48	0	0	0	0	0	0	0	0
6	Leste	25,06	24,97	25,04	24,99	24,96	24,96	25,06	12,48	0	0	0	0	0	0	0	0
	Oeste	50,1	49,99	50,03	50,08	50	49,95	50,04	24,89	0	0	0	0	0	0	0	0
	Norte	50,1	49,99	50,04	50,07	50	49,94	50,03	24,9	0	0	0	0	0	0	0	0
	Sul	25,04	24,97	25,04	25	24,96	24,96	25,05	12,48	0	0	0	0	0	0	0	0
	Local	25,02	25,03	24,97	25,06	25,06	24,94	25,02	12,44	0	0	0	0	0	0	0	0
9	Leste	**51,2**	**50,1**	**51,01**	**51,07**	**51,03**	**51,04**	**51,18**	**26,04**	**1,08**	**1,1**	**1,08**	**1,08**	**1,1**	**1,08**	**1,08**	**1,1**
	Oeste	25,03	24,93	24,99	25,04	25,02	25,03	25,06	12,48	0	0	0	0	0	0	0	0
	Norte	25,03	24,93	24,98	25,01	25,01	25,03	25,05	12,48	0	0	0	0	0	0	0	0
	Sul	50,08	49,93	49,98	50	49,9	50	50,1	24,97	0	0	0	0	0	0	0	0
	Local	25,06	25,05	24,96	24,97	24,97	24,95	25,05	12,48	0	0	0	0	0	0	0	0
10	Leste	25,06	24,97	24,96	24,95	24,99	25,04	25,05	12,96	0	0	0	0	0	0	0	0
	Oeste	50,11	50	49,96	50	49,88	50,01	50,12	25,92	0	0	0	0	0	0	0	0
	Norte	25,06	24,97	24,99	24,96	25,03	25,04	25,05	12,96	0	0	0	0	0	0	0	0
	Sul	50,11	49,99	49,97	49,97	49,92	50	50,1	25,94	0	0	0	0	0	0	0	0
	Local	25,06	25,05	24,96	25,04	24,98	24,93	25,04	12,96	0	0	0	0	0	0	0	0

5.1.1 Avaliação Preliminar de Área

Para determinar a área adicional utilizada pela rede com os monitores, inicialmente sintetiza-se uma rede sem alterações, com dimensões de 3 x 3 roteadores, utilizando o ambiente de desenvolvimento da Xilinx ISE versão 10.1, fazendo uso da ferramenta de síntese XST (*Xilinx Synthesis Technology*), tendo como plataforma alvo o dispositivo FPGA Xilinx Virtex5 XC5VLX50T, com 28800 blocos lógicos (LUTs). Para verificar o impacto dos monitores apresentados no Capítulo 3, sintetizou-se a rede com os monitores com as mesmas condições utilizadas na geração da rede original. A Tabela 8 mostra os resultados de área para a rede original, ou seja, sem os monitores inseridos nos roteadores da rede e para a rede com os monitores, bem como os resultados de área para roteadores de três, quatro e cinco portas.

Tabela 8 - Consumo de área da rede HERMES 3 x 3 sem e com monitores e de roteadores de 3, 4 e 5 portas.

	Arquitetura sem monitores		Arquitetura com monitores		Acréscimo de área em relação à rede original
	LUTs	Flip-Flops	LUTs	Flip-Flops	
Rede HERMES (3 x 3)	3965	1290	5688	2527	43%
Roteador (3 portas)	408	127	593	281	45%
Roteador (4 portas)	511	161	724	321	41%
Roteador (5 portas)	648	192	867	353	33%

Como se pode observar, a adição dos monitores nos roteadores da rede possui um acréscimo de área elevado em relação à rede original, a ocupação de blocos lógicos em FPGA aumenta cerca de 43% para a rede com os monitores. Visto que a estrutura de monitoramento é implementada para roteadores de cinco portas, e é utilizada nos roteadores com um número de portas inferior a cinco, tem-se um *hardware* subutilizado nesses roteadores. Na rede sintetizada, há quatro roteadores de três e quatro portas e apenas um de cinco portas, com isso justifica-se o elevado valor da sobrecarga de área que a estrutura de monitoramento implica na rede original.

5.2 Avaliação da Plataforma HeMPS

Utilizando as estruturas de dados para armazenar os valores de monitoramento dos canais de cada roteador da NoC, realizou-se uma validação da integração da estrutura de monitoramento com a plataforma HeMPS. A validação ocorre através da simulação de um cenário de teste, Cenário1. No Cenário1, são utilizadas duas aplicações *communication* executando ao mesmo tempo no MPSoC HeMPS com uma configuração de tamanho 3 x 3, com monitores inseridos em todos os nodos da rede, exceto no nodo mestre. As tarefas das aplicações são mapeadas estaticamente como representado na Figura 16, sendo que o número ao fim do seu nome se refere à aplicação. Por exemplo, *taskA_1* corresponde a tarefa A da aplicação *communication1*. O fluxo das comunicações entre as tarefas são ilustrados da Figura, sendo representados por F1 os fluxos da aplicação *communication1* e F2 os fluxos da aplicação *communication2*.

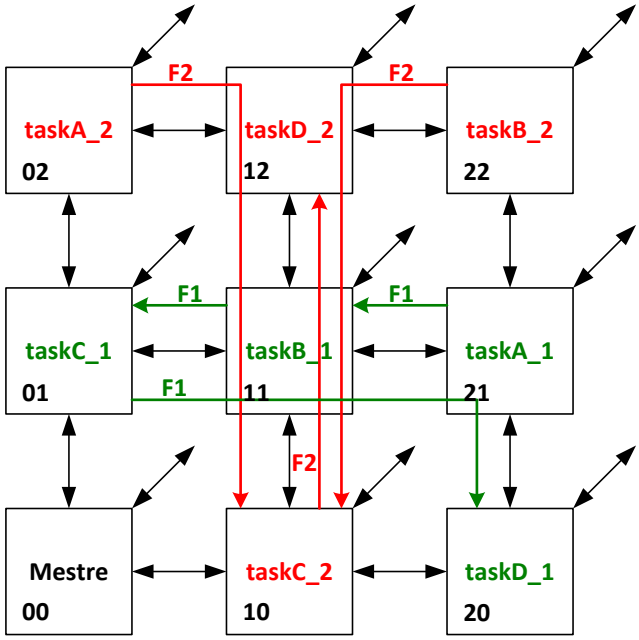


Figura 16 - Mapeamento estático das tarefas de duas aplicações communication no MPSoC HeMPS 3 x 3.

A taxa de comunicação da aplicação *communication* é apresentada na Figura 17. Nesta

aplicação, são enviadas 10 mensagens *msg1*, de tamanho 60 *flits* (30 palavras), da *taskA* e da *taskB* para a *taskC*. A *taskC* recebe essas 20 mensagens e repassa para a *taskD*. Sabendo-se que o cabeçalho de um pacote de dados que trafega na rede tem 12 *flits,* o tamanho total de um pacote para cada mensagem enviada é igual a 72 *flits* (*msg1* = 72 *flits*).

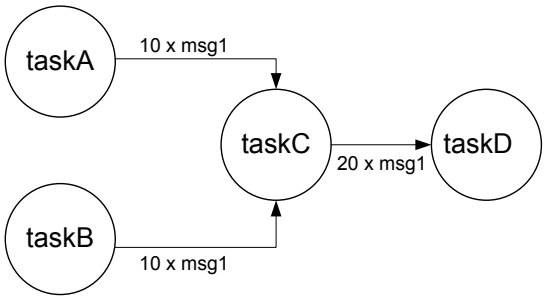


Figura 17 - Taxa de comunicação utilizada na aplicação communication.

De acordo com a comunicação e a taxa de comunicação entre as tarefas, podemos estimar a quantidade de tráfego que será observado ao final da execução das aplicações em cada porta dos roteadores. A Figura 18 ilustra, por exemplo, a estimativa para roteador central (roteador CC). Somente são apresentadas na Figura as mensagens que passam pelas portas do roteador CC, visto que é utilizado um roteamento XY para comunicação entre mensagens. Lembra-se ainda que os monitores contam apenas o número de *flits* de recebimento e não de envio.

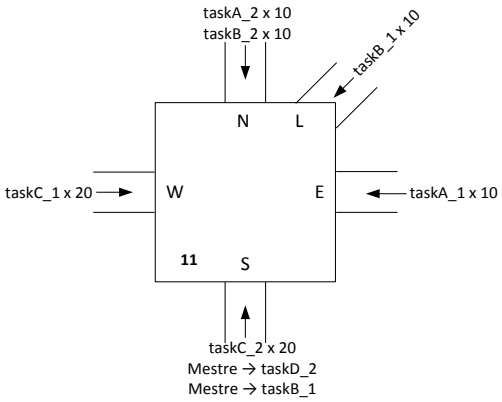


Figura 18 - Estimativa de tráfego nas portas do roteador CC.

Na Figura 18, a notação *taskA_2 x 10*, por exemplo, significa que 10 mensagens provenientes da tarefa *taskA_2* passam pela porta norte (N) do roteador. Pode-se observar que além de pacotes de envio das mensagens trafegando pela rede, tem-se as mensagens utilizadas pelo mapeamento efetuado pelo mestre das tarefas *taskD_2* (*Mestre→taskD_2*) e *taskB_1* (*Mestre→taskB_1*). Outros pacotes, não apresentados na figura, também trafegam pela rede. Como por exemplo, os referentes ao *debug* do sistema gerados pelo monitoramento, assim como pacotes enviados ao

mestre informando início e término de tarefas.

Com base na taxa de comunicação entre as tarefas da aplicação *communication*, ilustrada na Figura 17 e sabendo que *msg1* = 72 *flits*, tem-se a quantidade total de *flits* que são enviados pelas tarefas (Tabela 9). Na Tabela 10, é apresentado o tamanho, em *flits*, das mensagens de mapeamento efetuado pelo mestre para cada uma das tarefas alocadas no MPSoC.

Tabela 9 – Quantidade total de *flits* enviados pelas tarefas A, B e C das aplicações *communication1* e 2.

Tarefas	Taxa de comunicação	Quantidade total de flits enviados
taskA_1 (A1) e taskA_2 (A2)	10 x *msg1*	720
taskB_1 (B1) e taskB_2 (B2)	10 x *msg1*	720
taskC_1 (C1) e taksC_2 (C2)	20 x *msg1*	1440

Tabela 10 – Tamanho das mensagens de mapeamento enviadas pelo mestre para as tarefas.

Mensagens (mestre para tarefas)	Tamanho (*flits*)
Mestre→taskA_1 (MA) Mestre→taskA_2 (MA)	718
Mestre→taskB_1 (MB) Mestre→taskB_2 (MB)	846
Mestre→taskC_1 (MC) Mestre→taskC_2 (MC)	388
Mestre→taskD_1 (MD) Mestre→taskD_2 (MD)	342

A estimativa de tráfego nas portas dos roteadores de toda a rede é apresentada na Figura 19. Com base nessa estimativa, dos valores de quantidade total de *flits* (Tabela 9) e do tamanho das mensagens de mapeamento (Tabela 10), é possível estimar a quantidade de dados que trafegam nas portas de cada um dos roteadores monitorados da rede, essa estimativa é apresentada na Tabela 11.

Tabela 11 – Estimativa da quantidade total de dados que trafegam nos roteadores monitorados.

Roteador	Quantidade de dados nas portas (*flits*)				
	Leste (E)	**Oeste** (W)	**Norte** (N)	**Sul** (S)	**Local** (L)
00	0	0	0	0	0
01	1440	0	0	1106	1440
02	0	0	0	718	720
10	0	3482	1440	0	1440
11	720	1440	1440	2628	720
12	720	720	0	1782	0
20	0	1906	1440	0	0
21	0	1440	0	1564	720
22	0	0	0	846	720

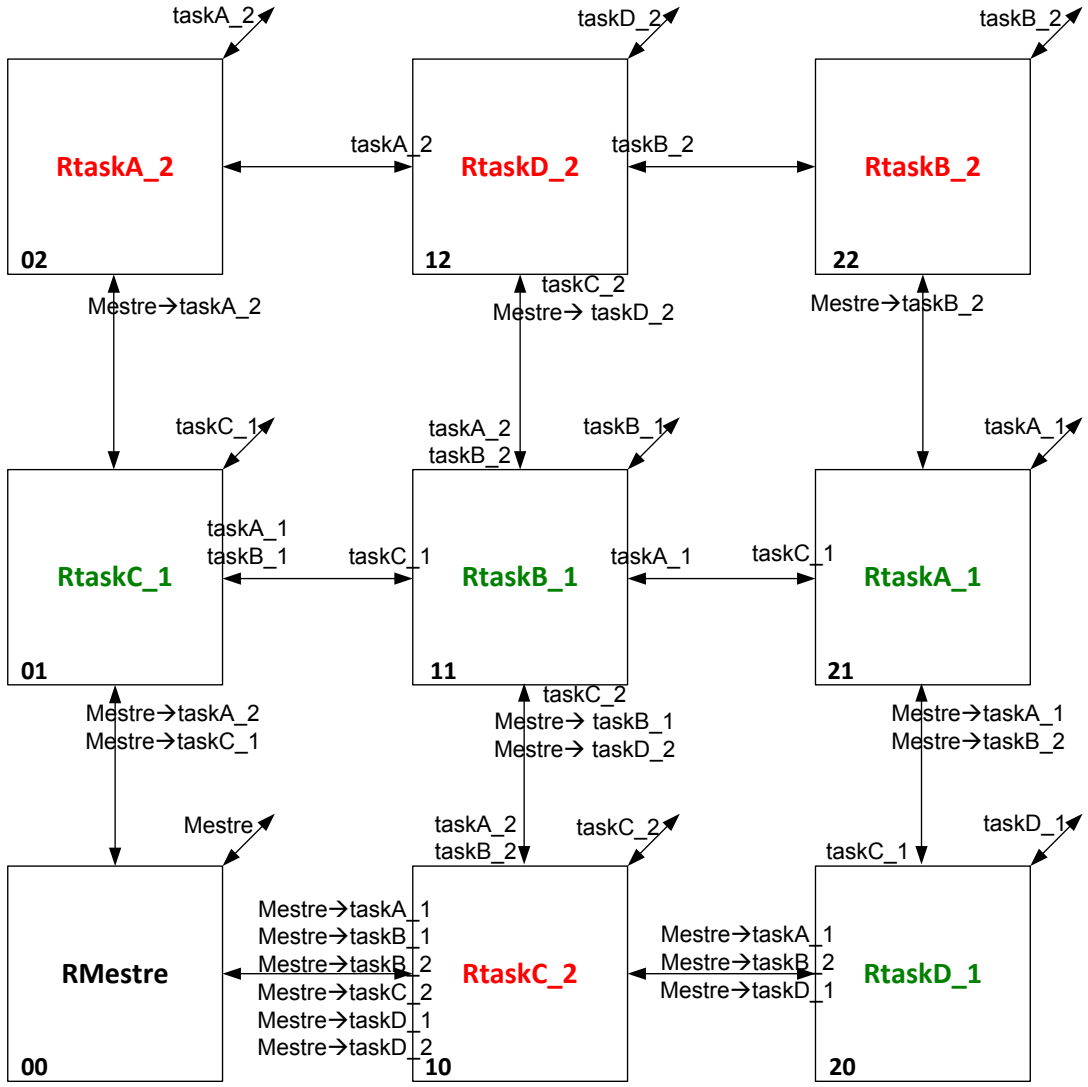


Figura 19 - Estimativa de tráfego nas portas de todos os roteadores monitorados.

Ao fim da execução das aplicações, tem-se a quantidade total de dados que trafegam nas portas dos roteadores monitorados da rede, ou seja, a quantidade **total** de *flits* recebidos em cada porta do roteador. Esses valores encontram-se armazenados nas cinco matrizes que compõem a estrutura matriz Total, as matrizes ilustradas na Tabela 12 apresentam os valores. Esses valores também são mostrados na Tabela 13.

Ao fazer uma análise nos resultados apresentados na Tabela 13, especificamente no roteador 11, por esse ser o único roteador a possuir cinco portas no cenário proposto, pode-se perceber que as portas leste (E) e local (L) que recebem 10 mensagens têm um valor muito próximo do número de *flits* ao final da execução das aplicações, sendo 956 *flits* medidos na porta leste e 816 *flits* na porta local. Na porta norte (N) e na porta oeste (W) que recebem 20 mensagens é contabilizado um valor igual do número de *flits*, sendo 1640 *flits* medidos em ambas as portas. Isto comprova um correto funcionamento dos monitores, pois mostra que portas que tem um número igual de

mensagens a ser recebidas apresentam um comportamento muito parecido. Além disso, a porta sul (S) que recebe além das 20 mensagens de dados, pacotes referentes ao mapeamento, possui um valor muito maior do que as portas que recebem só as 20 mensagens.

Tabela 12 – Valores das matrizes da estrutura matriz total.

Matriz	
East Channels Total (ECT)	$\begin{bmatrix} 404 & 941 & 0 \\ 1847 & 956 & 0 \\ 0 & 432 & 0 \end{bmatrix}$
West Channels Total (WCT)	$\begin{bmatrix} 0 & 720 & 0 \\ 0 & 1640 & 1324 \\ 0 & 4010 & 2252 \end{bmatrix}$
North Channels Total (NCT)	$\begin{bmatrix} 0 & 0 & 0 \\ 595 & 1640 & 0 \\ 0 & 1640 & 1224 \end{bmatrix}$
South Channels Total (SCT)	$\begin{bmatrix} 892 & 1862 & 1020 \\ 1564 & 2806 & 1812 \\ 0 & 0 & 0 \end{bmatrix}$
Local Port Total (LPT)	$\begin{bmatrix} 816 & 296 & 816 \\ 1510 & 816 & 816 \\ 0 & 1734 & 222 \end{bmatrix}$

Tabela 13 – Quantidade total de dados que trafegam nos roteadores monitorados.

Roteador	**Quantidade de dados nas portas (*flits*)**				
	Leste (E)	**Oeste** (W)	**Norte** (N)	**Sul** (S)	**Local** (L)
00	0	0	0	0	0
01	1847	0	595	1564	1510
02	404	0	0	892	816
10	432	4010	1640	0	1734
11	**956**	**1640**	**1640**	**2806**	**816**
12	941	720	0	1862	296
20	0	2252	1224	0	222
21	0	1324	0	1812	816
22	0	0	0	1020	816

Fazendo um cálculo de número de *flits* para 10 mensagens, teríamos 10 vezes 72 (número de *flits* de um pacote de dados para a mensagem msg1) resultando 720 *flits*, valor este que está próximo aos 816 e 956 *flits* encontrados nas portas que recebem este número de mensagens, tais como porta leste e local. Vale ressaltar que este número é maior devido aos pacotes não tomados em consideração, como relatado anteriormente. O mesmo acontece com portas que recebem 20 mensagens, em que o cálculo resulta 1440 *flits* (20 vezes 72) e o valor medido é 1640 *flits*.

O experimento realizado demonstra que as estruturas de dados geradas para tratar os dados monitorados estão funcionando corretamente, tem-se assim a correta integração da estrutura de monitoramento com a plataforma MPSoC.

6 CONCLUSÕES E TRABALHOS FUTUROS

Esse trabalho apresentou uma infra-estrutura de monitoramento para plataformas MPSoCs que utilizam NoC como meio interconexão. O monitoramento é realizado através de monitores inseridos em roteadores da NoC.

Uma estrutura para monitorar a NoC [MOR04] foi implementada. Para que fosse possível inserir os monitores nos roteadores, foi necessário realizar alterações na arquitetura original dos roteadores da rede. Os monitores estão inseridos nas portas de entrada dos roteadores da NoC e geram informações da quantidade de *flits* que passam nessas portas em uma determinada janela de tempo. A partir dessas informações é possível calcular a taxa de recepção de pacotes na rede (vazão). Cada roteador envia pacotes com os dados de monitoramento a um processador responsável pela gerência do MPSoC. Esse processador recebe os dados e monta uma estrutura de dados para armazenar os valores de monitoramento dos canais de cada roteador. Essa estrutura reflete a ocupação dos núcleos do MPSoC. A partir de uma análise realizada na estrutura de dados, o processador gerente pode tomar decisões relevantes para a melhora de desempenho do MPSoC.

No sentido de validar os monitores, realizaram-se simulações de três cenários de teste. Em todos os cenários esperou-se que os monitores recuperem a taxa de recepção dos pacotes. O primeiro e o segundo cenário possuem respectivamente uma taxa de injeção dos pacotes igual a 80 Mbps (10% da taxa máxima da rede) e 160 Mbps (20% da taxa máxima da rede). Como nesses dois cenários não há fluxo de pacotes concorrentes, as taxas de recepção dos pacotes medidas devem ser iguais as taxas de injeção. Nos dois primeiros cenários foi possível observar que os monitores computaram corretamente as taxas de injeção dos pacotes, igual a 10% e 20%. O terceiro cenário apresenta fluxos concorrentes, portanto as taxas de recepção dos pacotes medidas devem ser iguais ao somatório das taxas dos fluxos concorrentes. Nesse cenário verificou-se que os monitores calcularam a taxa de 25%, esta correspondendo à taxa de injeção dos pacotes para esse cenário. Ainda foi possível observar que algumas taxas recuperadas foram iguais a 50%, nesse caso há concorrência entre fluxos de pacotes. Através dessas simulações sabe-se que os monitores geram um acréscimo em torno de 1,1% na vazão original dos pacotes na rede.

Utilizando as estruturas de dados para armazenar os valores de monitoramento dos canais de cada roteador da NoC, realizou-se uma validação da integração da estrutura de monitoramento com a plataforma MPSoC HeMPS [WOS07]. Foi gerado um cenário onde são utilizadas duas aplicações executando ao mesmo tempo no MPSoC e com monitores inseridos em todos os nodos

da rede, exceto no nodo mestre. De acordo com a comunicação e a taxa de comunicação entre as tarefas, foi possível fazer uma estimativa de tráfego em cada porta dos roteadores que será observado ao final da execução das aplicações. A partir da estimativa de tráfego realizada, tem-se a estimativa da quantidade total de dados que trafegam nos roteadores monitorados. Ao final da execução das aplicações, obteve-se a quantidade total de dados que trafegam em cada roteador monitorado. Essa quantidade é obtida através da estrutura de dados que é gerada pelo processador mestre. Ao fazer uma análise nos valores medidos, pode-se perceber que esses possuem um valor muito próximo aos valores estimados, demonstrando assim o correto funcionamento de toda a infra-estrutura de monitoramento proposta nesse trabalho.

Sendo assim, este trabalho atingiu seu objetivo de monitorar a plataforma MPSoC HeMPS visto que os monitores inseridos nos roteadores da rede HERMES recuperam corretamente a taxa de recepção dos pacotes na rede. Com isso, o processador responsável pelo gerenciamento da rede monta corretamente a estrutura de dados que reflete a ocupação dos núcleos do MPSoC em tempo de execução, podendo tomar decisões relevantes para a melhora do desempenho da plataforma.

6.1 Trabalhos Futuros

É importante destacar que a pesquisa realizada deixa margens para que trabalhos futuros possam ser realizados. Inicialmente, a parametrização dos monitores de acordo com a quantidade de portas dos roteadores, visando uma redução da sobrecarga de área que os monitores implicam é um trabalho a ser realizado. Após a parametrização dos monitores, sugere-se automatizar a inclusão dos monitores na rede no *framework* ATLAS.

Destaca-se ainda como trabalho futuro a ser realizado, aplicar a infra-estrutura de monitoramento para MPSoC nas tomadas de decisão para uma melhora no desempenho do mesmo, por exemplo, o mapeamento de ajuste dinâmico de freqüência, onde o processador responsável pelo gerenciamento do sistema possa recomendar que um determinado roteador da rede seja desligado ou tenha sua frequência de operação diminuída. Ainda no sentido quanto a tomadas de decisão, destaca-se o mapeamento dinâmico e a migração de tarefas em núcleos que estejam com uma menor carga de trabalho ou com o caminho que tenha o menor tráfego de comunicação entre tarefas. A inclusão de novas métricas nos monitores, como latência e *jitter*, podem contribuir para o melhor controle dos recursos do sistema em tempo execução.

REFERÊNCIAS

[ATL07] Atlas - An Environment for NoC Generation and Evaluation. Capturado em: https://corfu.pucrs.br/redmine/projects/atlas, Dezembro 2010.

[BEN01] Benini, L.; Micheli, G. “Powering networks on chips: energy-efficient and reliable interconnect design for SoCs”. In: International Symposium on System Synthesis – (ISSS), 2001, pp. 33 -38.

[BEN02] Benini, L.; De Micheli, G. “Networks on chips: a new SoC paradigm”. IEEE Computer Magazine, v.35-1, Janeiro 2002, pp. 70-78.

[BRA07] Brand, J. W.; Ciordas, C.; Goossens, K.; Basten, T. “Congestion-Controlled Best-Effort Communication for Networks-on-Chip”. In: Design Automation and Test in Europe Conference & Exhibition (DATE), 2007, 6p.

[CAR09] Carvalho, E. L. S. “Mapeamento Dinâmico de Tarefas em MPSoCs Heterogêneos baseados em NoC”. Tese de Doutorado, Programa de Pós-Graduação em Ciências da Computação, PUCRS, 2009, 168p.

[CIO04] Ciordas, C.; Basten, T.; Radulescu, A.; Goossens, K.; Meerbergen, J. “An Event-based Network-on-Chip Monitoring Service”. In: IEEE International High-Level Design Validation and Test Workshop (HLDVT), 2004, pp. 149-154.

[CIO06a] Ciordas, C.; Goossens, K.; Radulescu, A.; and Basten, T. “NoC monitoring: Impact on the design flow”. In: IEEE International Symposium on Circuits and Systems (ISCAS), 2006, 4p.

[FIO08] Fiorin, L.; Silvano, C.; Palermo, G. “A Security Monitoring Service for NoCs”. In: International Symposium on System Synthesis (ISSS), 2008, pp. 197-202.

[FIO09] Fiorin, L.; Silvano, C.; Palermo, G. “MPSoCs Run-Time Monitoring through Network-on-Chip”. In: Design, Automation and Test in Europe Conference & Exhibition (DATE), 2009, pp. 558-561.

[GUE00] Guerrier, P.; Greiner, A. “A Generic Architecture for on Chip Packet-Switched Interconnections”. In: Design Automation and Test in Europe Conference & Exhibition (DATE), 2000, pp. 250-256.

[JER05] Jerraya, A.; Tenhunen, H.; Wolf, W. “Guest Editors' Introduction: Multiprocessor Systems-on-Chips”. IEEE Computer, v.38-7, Julho 2005, pp. 36-40.

[JER07] Jerraya, A.; Franza, O.; Levy, M.; Nakaya, M.; Paulin, P.; Ramacher, U.; Talla, D.; Wolf, W. “Roundtable: Envisioning the Future for Multiprocessor SoC“. IEEE Design & Test of Computers, v.24-2, Março. 2007, pp. 174–183.

[KIM07] Kim, K.; Kim, D.; Lee, D.; Yoo, H. “Cost-efficient Network-on-Chip Design Using Traffic Monitoring System”. In: Design, Automation and Test in Europe Conference & Exhibition (DATE), 2007, pp 301-307.

[MAR01] Martin, G.; Chang, H. “System-on-Chip design”. In: International Conference on ASIC (ASIC), 2001, pp. 12-17.

[MAR05] Marescaux, T.; Rangevall, A.; Nollet, V.; Bartic, A.; Corporaal, H. “Distributed Congestion Control for Packet Switched Networks on Chip”. In: International Parallel Computing conference (ParCo), 2005, pp. 761-768.

[MOR04] Moraes, F.; Calazans, N.; Mello, A.; Möller, L.; Ost, L. “HERMES: an Infrastructure for Low Area Overhead Packet-switching Networks on Chip”. Integration, the VLSI Journal, v.38-1, Outubro 2004, pp. 69-93.

[PLA01] Plasma - most MIPS I(TM) opcodes :: Overview. Capturado em: http://opencores.org/project,plasma, dezembro 2010.

[RIC03] Richter, K.; Jersak, M.; Ernst, R. “A Formal Approach to MPSoC Performance Verification”. IEEE Computer, v.36-4, Abril 2003, pp.60-67.

[VER09] Vermeulen, B.; Goossens, K. “A Network-on-Chip Monitoring Infrastructure for Communication-centric Debug of Embedded Multi-Processor SoCs”. In: International Symposium on VLSI Design, Automation and Test (VLSI-DAT), 2009, pp. 183-186.

[WOS07] Woszezenki, C. “Alocação de Tarefas e Comunicação entre Tarefas em MPSoCs”. Dissertação de Mestrado, PPGCC-PUCRS. Fevereiro 2007. 121p. Disponível em: http://www.inf.pucrs.br/~moraes/my_pubs/papers/dissertacao_cris.pdf.