



Pontifícia Universidade Católica do Rio Grande do Sul  
Faculdade de Informática  
Programa de Pós-Graduação em Ciência da Computação



# **PROPOSTA DE UMA INFRAESTRUTURA DE GERAÇÃO E AVALIAÇÃO PARA REDES INTRACHIP HERMES-G**

RAFFAEL BOTTOLI SCHEMMER

ORIENTADOR: PROF. DR. NEY LAERT VILAR CALAZANS

PORTO ALEGRE, AGOSTO DE 2012



### **Dados Internacionais de Catalogação na Publicação (CIP)**

S323p Schemmer, Raffael Bottoli

Proposta de uma infraestrutura de geração e avaliação para redes intrachip Hermes-G / Raffael Bottoli Schemmer. – Porto Alegre, 2012.

125 p.

Dissertação (Mestrado) – Faculdade de Informática, PUCRS.  
Orientador: Prof. Dr. Ney Laert Vilar Calazans.

1. Informática. 2. Arquitetura de Redes. 3. Redes de Computadores. 4. Circuitos Assíncronos. I. Calazans, Ney Laert Vilar. II. Título.

CDD 004.6

**Ficha Catalográfica elaborada pelo  
Setor de Tratamento da Informação da BC-PUCRS**





Pontifícia Universidade Católica do Rio Grande do Sul  
FACULDADE DE INFORMÁTICA  
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

## TERMO DE APRESENTAÇÃO DE DISSERTAÇÃO DE MESTRADO

Dissertação intitulada "Proposta de uma Infraestrutura de Geração e Avaliação para Redes Intrachip Hermes-G", apresentada por Raffael Bottoli Schemmer como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação, Sistemas Embarcados e Sistemas Digitais, aprovada em 29/08/2012 pela Comissão Examinadora:

  
Prof. Dr. Ney Laert Vilar Calazans - PPGCC/PUCRS  
Orientador

  
Prof. Dr. César Augusto Missio Marcon - PPGCC/PUCRS

  
Prof. Dr. Fernando Gehm Moraes - PPGCC/PUCRS

  
Prof. Dr. Edson Ifarraguirre Moreno - FACIN/PUCRS

Homologada em 26/03/2015, conforme Ata No. 004 pela Comissão Coordenadora.

  
Prof. Dr. Paulo Henrique Lemelle Fernandes  
Coordenador.

**PUCRS**

**Campus Central**

Av. Ipiranga, 6681 - P32- sala 507 - CEP: 90619-900  
Fone: (51) 3320-3611 - Fax (51) 3320-3621  
E-mail: [ppgcc@pucrs.br](mailto:ppgcc@pucrs.br)  
[www.pucrs.br/facin/pos](http://www.pucrs.br/facin/pos)



## AGRADECIMENTOS

Antes tudo, faço deste um espaço de agradecimento as pessoas que de fato contribuíram para a realização deste trabalho. Também, àquelas que melhoraram de alguma forma a qualidade final e relevância impressa deste trabalho. Que a ordem de citação não imprima o grau de importância das pessoas citadas aqui. Por fim, minhas desculpas para aqueles que foram omitidos e/ou não citados.

Primeiramente, agradeço a Deus, pela disponibilidade dada para a escrita e realização deste trabalho. À CAPES e ao CNPQ pela viabilidade e subsídio para realização da pesquisa dentre os 30 meses que consistiram este trabalho. A meus pais, Fatima Bottoli Schemmer e Wilson Irineu Schemmer, pela oportunidade dada antes tudo de escolher minha área de atuação. Também, pela viabilidade quanto a infraestrutura e subsídio durante minha caminhada acadêmica em busca do conhecimento e das titulações. Por fim, pelo apoio e ajuda durante as dificuldades encontradas.

Agradeço à PUCRS pela infraestrutura disponibilizada em específico à biblioteca Irmão José Otão e também aos laboratórios de pesquisa LAD e GAPH que residem no prédio 32. Meu obrigado ao PPGCC e aos discentes que me elegeram para representar a pós graduação como representante discente entre os anos de 2011-II e 2012-I. Meu obrigado também aos sábios ensinamentos que colhi sendo membro da comissão de coordenação do PPGC, em especial aos professores membros das comissões de coordenação do PPGCC e colegiado da FACIN.

Meu obrigado a todos os mestres e educadores que tive oportunidade de ser aluno durante o período que realizei este trabalho. Em especial, meu obrigado aos Profs. Fernando Gehm Moraes, Cesar De Rose, Luiz Gustavo e Ney Calazans. Gostaria também de agradecer ao PGCC da UFRGS e em especial ao professor Sergio Bampi que possibilitou a realização da disciplina de concepção de circuitos VLSI em caráter de aluno especial durante a realização do mestrado em 2010-I.

Deixo por fim, pelo grau de importância maior, meu agradecimento a três pessoas fundamentais para realização deste trabalho. A primeira delas, meu orientador e mentor, Prof. Ney Calazans, do qual sou imensamente grato não apenas pelo aceite e orientação deste trabalho mas também por permitir minha evolução e amadurecimento de conceitos e conhecimentos durante a realização deste trabalho. Se dedico a alguém este trabalho, este alguém é você. Agradeço também ao Prof. Carlos Petry, por estar sempre a meu lado durante o desenvolvimento deste trabalho e de todos os momentos decisivos em minha vida acadêmica. Agradeço também ao Prof. Fabiano Hessel, que no princípio do ingresso ao mestrado, acreditou em minhas qualidades e potencial, possibilitando o ingresso na PUCRS, culminando por fim no desenvolvimento deste trabalho.



# PROPOSTA DE UMA INFRAESTRUTURA DE GERAÇÃO E AVALIAÇÃO PARA REDES INTRACHIP HERMES-G

## RESUMO

Os avanços relacionados à tecnologia de fabricação de circuitos integrados impulsionam a complexidade e o número de funcionalidades dos produtos eletrônicos. A literatura aponta que até 2015 tarefas do nível comportamental ocuparão cerca de 50% do esforço de projeto, o que reforça a necessidade do desenvolvimento de ferramentas de automação e geração automática de circuitos. Além disso, o projeto de circuitos atuais faz uso prioritariamente do paradigma de projeto síncrono, que associado ao crescimento da complexidade dos mesmos impõe restrições importantes com relação ao consumo de energia e à dissipação de potência.

Este trabalho apresenta uma solução alternativa a alguns dos problemas citados, pela proposta de um ambiente de geração e avaliação de redes intrachip. Tais redes permitem, além de conectar módulos de processamento que operem em diferentes frequências, ajudar a garantir o atendimento de restrições temporais impostas pelos requisitos de tráfego destes módulos. Durante a geração da rede, o ambiente permite em tempo de projetos selecionar características da mesma, tais como as frequências de operação dos roteadores, de forma individualizada.

Além da geração da rede, o ambiente ainda habilita avaliar restrições temporais de diferentes modelos de tráfegos, dando suporte à geração parametrizada de tráfego para exercitar a rede. Esta característica oferece alternativas para reduzir o esforço do projeto dos sistemas eletrônicos ainda nas fases de especificação de requisitos do sistema. Isto ocorre por que o ambiente facilita a visualização do comportamento de um modelo de rede, demonstrando se o mesmo atende ou não a requisitos esperados para um cenário de tráfego.

**Palavras chave:** Geração de Redes, Geração de Tráfego, Redes Intrachip Não Síncronas, Avaliação de Tráfego.



# PROPOSAL OF AN INFRASTRUCTURE FOR THE GENERATION AND EVALUATION OF HERMES-G NETWORKS ON CHIP

## ABSTRACT

Advances related to integrated circuit manufacturing technologies push the complexity and the number of functionalities in electronic products. The literature points out that in 2015 behavioral design will demand 50% of the whole design effort, what indicates a major need for developing circuit design automation tools. Besides that, the design of current circuits employs the synchronous design paradigm prioritarilly. However this design paradigma jointly with the increase of complexity imposes relevant restriction with regard to energy consumption and power dissipation design constraints.

This works presents an alternative to some of the cited problems, proposing an environment for the generation and evaluation of intrachip networks. These networks allow interconnect processing modules operating at different operating frequencies, as well as help to guarantee the fulfillment of temporal restrictions temporais imposed by the traffic requirements of such modules. During the network generation step, the proposed environment allows selecting the network characteristics at design time, including individual router operating frequencies.

Besides network generation, the environment also enables evaluating temporal constraints for several distinct traffic models, supporting the parameterized generation of traffic to exercise the network. This characteristic offer new alternatives to reduce the design effort of intrachip network for electronic systems still in the early phases of system specification. This occurs because the environment enables the visualization of the network behavior, demonstrating if this fulfills or not the expected requirements for some give traffic scenario.

**Keywords:** Network Generation, Traffic Generation, Non-Synchronous Networks on Chip, Traffic Evaluation.



## LISTA DE FIGURAS

Figura 1: Topologias suportadas pela ferramenta OPNET: (a) 2D Mesh (b) Fat Tree (c) Big Fat Tree. ....	23
Figura 2: Modelos propostos por Kreutz et al. (a) Modelo ACP, define os custos de computação e dependências entre as tarefas. (b) Modelo CRG, define a arquitetura de comunicação onde a tarefa vai ser mapeada. ....	25
Figura 3: Arquitetura típica de um cluster na rede DSPIN, formada por três IPs, uma arquitetura de interconexão local e um adaptador de rede, que conecta o cluster a um roteador, este formado por duas portas, uma de envio e outra de recebimento de dados. ....	26
Figura 4: Interface gráfica de geração de redes do ambiente MAIA. (1) Permite ao usuário selecionar os parâmetros da rede. (2) Representa uma rede 4x4 Mesh. (3) Dispara o ambiente que gera a rede selecionada. ....	27
Figura 5: Fila bi síncrona utilizada em redes HERMES-G e HERMES-GLP. Nesta arquitetura de fila, a escrita e a leitura podem ou não operar na mesma fase ou na mesma frequência. A codificação de Gray é utilizada nos ponteiros de leitura e de escrita da fila para garantir que ambos os ponteiros estejam sincronizados, mesmo sendo manipulados em diferentes domínios de fase e de frequência. ....	28
Figura 6: Topologias suportadas pelo ambiente de geração: (a) Malha 2D, (b) Anel, (c) Spidergon, (d) Crossbar. ....	29
Figura 7: (a) Categoria de serviço CBR do inglês <i>Constant Bit Rate</i> em que todos os pacotes são injetados na rede na mesma taxa de injeção. (b) Categoria de serviço VBR, do inglês <i>Variable Bit Rate</i> em que a taxa de injeção é variável ao longo do tempo. (c) Distribuição espacial do cenário de tráfego composto por um gerador de vídeo e um gerador de voz. ....	31
Figura 8: Curva exponencial decrescente de 1000 pacotes gerados em um intervalo de 100Mbps a 200Mbps com incremento de 10Mbps e média de 151Mbps. ....	34
Figura 9: Plataforma de emulação AcENoCs, formada pelos ambientes software e hardware. O ambiente de software é responsável pela emulação da geração das frequências, pelas filas de entrada dos roteadores e pela geração e recepção do tráfego. O ambiente de hardware, é responsável pela emulação da NoC e pelo banco de registradores, que armazena os parâmetros da NoC a ser emulada. ....	37
Figura 10: Algoritmo de Strassen descrito como um grafo de dependências. ....	40
Figura 11: Projeto de um SoC, formado por componentes de diferentes características e demandas por conexões. ....	42
Figura 12: Fluxo de projeto de geração de uma rede no ambiente ATLAS, composto por quatro passos (1) Criação de um projeto. (2) Definição das características desejadas para uma rede. (3) Geração da rede. (4) Edição de um projeto de rede gerado. ....	48

Figura 13: Curvas proibidas e permitidas para os algoritmos adaptativos: (a) Negative First proíbe curvas para Oeste se o pacote está indo para o Norte e curvas para o Sul se o pacote estiver indo para Leste. (b) West First proíbe curvas para Oeste. (c) North Last proíbe qualquer curva após o sentido Norte ter sido tomado. ....	50
Figura 14: Diagrama de blocos da fila bi síncrona utilizada pela rede HERMES-G.....	51
Figura 15: Conexões de entrada e saída das arquiteturas (a) Async_Fifo e (b) Fila bi síncrona. ....	52
Figura 16: Máquina de estados de leitura da fila bi síncrona, composta por cinco estados, sendo um deles responsável pela inicialização, dois responsáveis pelo roteamento do pacote e dois pela transmissão dos <i>flits</i> do pacote. ....	53
Figura 17: Interface principal do ambiente de geração de tráfego HERMES-G e suas opções de seleção de características de geração de redes.....	56
Figura 18: Componente da interface gráfica utilizável para gerenciar valores de frequência definidos pelo usuário. Estas são utilizadas para definir as frequências de operação de roteadores e de módulos IP a estes conectados. ....	57
Figura 19: Interface de adição de uma nova frequência. Campos (1) e (2) possibilitam informar um nome para a frequência e um valor para a frequência. ....	58
Figura 20: Interface de remoção de frequências. Esta interface possui apenas um campo, onde se pode selecionar uma frequência a remover.....	58
Figura 21: Interface de edição de frequências. Permite selecionar uma frequência cadastrada e modificar seus campos referentes a nome, valor e unidade de frequência. ....	59
Figura 22: Interface de seleção de frequências cadastradas: (1) Para o roteador, do inglês <i>Router Frequency</i> ; (2) Para o módulo de transmissão de pacotes (Definido na interface gráfica do ambiente como <i>Input Ip Frequency</i> ); (3) Para o módulo de recepção de pacotes (Definido na interface gráfica do ambiente como <i>Output Ip Frequency</i> ).....	59
Figura 23: Combinações existentes com relação ao número de portas utilizadas pelos roteadores em uma rede com topologia malha 2D. ....	62
Figura 24: (a) Descrição de um roteador formado por cinco portas, um <i>Crossbar</i> e um <i>SwitchControl</i> (SWC). (b) Descrição de uma rede malha formada por nove roteadores.....	63
Figura 25: Distribuições espaciais suportadas pelo modelo de tráfego sintético. (a) Aleatória: Cada pacote de um tráfego é enviado aleatoriamente entre os demais endereços da rede. No exemplo (a), o endereço zero envia aleatoriamente pacotes para outros endereços. (b) Destino único: Todos os pacotes de um tráfego são enviados sempre para um mesmo endereço. No exemplo (b) o endereço zero envia apenas pacotes para o endereço dois. (c) Complemento: Todos os pacotes de um tráfego são enviados ao endereço complemento do endereço do transmissor do tráfego. No exemplo (c), os endereços zero e oito enviam pacotes para seu endereço complemento. ....	65
Figura 26: Modelo de variação das taxas de injeção dos pacotes. O tamanho dos pacotes é sempre o mesmo, onde o tempo ocioso é variado conforme a taxa de injeção de cada pacote.....	66

Figura 27: Exemplo de uma distribuição normal de 1000 pacotes, distribuídos em um intervalo com taxa mínima de 1Mbps e máxima de 1000Mbps, média de 500Mbps, desvio padrão de 100Mbps e incremento de 10Mbps.....	68
Figura 28: Exemplo de uma distribuição exponencial de 1000 pacotes distribuídos em um intervalo com taxa mínima de 1Mbps e máxima de 1000Mbps, média de 100Mbps e incremento de 10Mbps. ....	69
Figura 29: Interface principal do gerador de tráfego, que possibilita criar múltiplos cenários de tráfego.....	70
Figura 30: Interface que possibilita parametrizar o tráfego, tais como número de pacotes, tamanho dos pacotes, distribuição espacial e temporal.....	71
Figura 31: Formato intermediário de um pacote de tráfego. Os campos representam: (a) Tempo de transmissão do pacote; (b) Roteador destino do pacote; (c) Tamanho do pacote; (d) Roteador origem do pacote; (e) Tempo de injeção em decimal; (f) Número de sequência do pacote; (g) Carga útil (dados) do pacote. ....	72
Figura 32: Definição de um pacote, formado pelos campos de Header e Payload, onde um representa informação de controle de transmissão e o outro contém os dados do pacote.....	73
Figura 33: Descrição da interface da entidade SC_Input_Module, que implementa o componente transmissor de pacotes para a rede.....	74
Figura 34: Formato de um pacote gerado pelo transmissor de pacotes.....	75
Figura 35: Descrição da interface da entidade SC_Output_Module, que descreve o componente receptor de pacotes. ....	75
Figura 36: Formato do pacote após ser recebido e processado pelo receptor de pacotes utilizado para avaliação do tráfego dos pacotes. ....	76
Figura 37: Interface linha de comando da ferramenta de testes responsável pela geração e verificação de cenários. ....	79
Figura 38: Uso de modelos de aplicações CDCM no ambiente Atlas: (a) Criação e geração de um grafo de aplicações CDCM; (b) Geração do Testbench pela ferramenta ATLAS; (c) Geração dos arquivos de tráfego e o ambiente de transmissão e recepção de pacotes.....	80
Figura 39: Ferramenta de criação de aplicações CDCM através de grafos CDCG no ambiente CAFES. ....	81
Figura 40: Ferramenta de mapeamento do ambiente CAFES. Cada tarefa da aplicação é mapeada em um dos endereços da rede. ....	82
Figura 41: Arquivo que contém as características de um modelo de aplicação CDCM. ....	83
Figura 42: Interface principal do gerador de tráfego, que possibilita carregar arquivos que descrevem modelos de aplicações. ....	83

Figura 43: Formato intermediário de um pacote de tráfego. Os campos são: (a) Origem do pacote; (b) Roteador destino do pacote; (c) Tempo de processamento do pacote em ciclos; (d) Tamanho do pacote; (e) Número de sequência do pacote. ....	84
Figura 44: Descrição das portas de entrada e saída da entidade SC_Input_Module, que descreve o componente transmissor de pacotes da rede para o modelo de Testbench de modelos de aplicações CDCM.....	85
Figura 45: Interface da ferramenta distribuição de vazões do ambiente de avaliação de tráfego. No exemplo, é usado um tráfego uniforme de 400Mbps de dois pacotes transmitidos do endereço 00 ao endereço 11. Na figura, observa-se que dois pacotes foram transmitidos, um pacote com vazão aproximada de 200Mbps, e outro pacote com vazão de 250Mbps. ....	95
Figura 46: Interface da ferramenta distribuição de latências do ambiente de avaliação de tráfego. O exemplo usa um tráfego uniforme de 400Mbps de dois pacotes transmitidos do endereço 00 ao endereço 11. Na figura, observa-se que dois pacotes foram transmitidos, um pacote com latência aproximada de 1020ns, e outro pacote com latência de 1260ns. ....	96
Figura 47: Interface da ferramenta analisador de latências do ambiente de avaliação de tráfego, que permite visualizar latências específicas de pacotes de tráfegos. ....	96
Figura 48: Interface da ferramenta relatório global que permite visualizar latência e a vazão ideais e vazão e a latência medidas de todos os tráfegos. ....	97
Figura 49: Rede 8x8 contendo duas ilhas de frequência. Os retângulos pontilhados (0 e 2) possuem trinta e dois processadores operando a 50MHz, conectados a roteadores que operam a 100MHz. Nestas duas ilhas todos os roteadores se comunicam utilizando filas síncronas (S). O retângulo preto (1) possui dezesseis elementos de memória e dezesseis controladores de entrada e saída que operam a 50MHz, e estão conectados a roteadores que operam a 50MHz. Repare que as filas bi síncronas (BS) são utilizadas para comunicação unicamente entre as ilhas de frequência distintas. ....	99
Figura 50: Rede 8x8 contendo quatro ilhas de frequência, ilustradas pelos quadrados (0/1/2/3). A fila síncrona é utilizada para comunicação entre os roteadores de cada ilha. Os roteadores periféricos as ilhas, fazem uso de filas bi síncronas para garantir a comunicação entre os diferentes domínios de frequências.....	104
Figura 51: Rede 8x8 formada por 64 roteadores, transmissores e receptores que operam a 50MHz exceto o roteador, transmissor e receptor do endereço 44. Este endereço é utilizado para armazenar a tarefa mestre da aplicação, responsável pela transmissão e recepção dos dados da aplicação. ....	114
Figura 52: Aplicação CDCM que descreve uma multiplicação vetorial paralela implementada através do modelo de programação mestre escravo. A aplicação é implementada em duas fases sendo a fase (1) responsável pela transmissão dos blocos do vetor pela tarefa mestre (A) as tarefas escravas (S0 a S63). Na fase (2) os escravos multiplicam o vetor e transmitem os dados processados novamente a tarefa mestre que conclui a execução da aplicação.....	115

## LISTA DE TABELAS

Tabela 1: Resumo do estado da arte nos temas de projeto e geração de redes intrachip.....	44
Tabela 2: Resumo do estado da arte em caracterização, geração e avaliação de tráfego para redes intrachip. ....	46
Tabela 3: <i>TimeStamp</i> para três pacotes de 13 <i>flits</i> de tamanho injetados com taxa uniforme de 100Mbps por um transmissor operando a 100 MHz em uma rede de comprimento de <i>flit</i> igual a 8bits.....	72
Tabela 4: Valores de vazão e latência média de todos os pacotes de cada tráfego obtidos durante a simulação dos tráfegos para os cenários de rede síncrona e não síncrona. ....	101
Tabela 5: Valores de área em número de LUTs de quatro entradas e consumo de energia. O consumo de energia (em nanoJoule/s) é dado em: (i) Consumo de energia dos sinais de relógio;(ii) Lógica combinacional do circuito; (iii) Lógica sequencial do circuito. ....	103
Tabela 6: Valores de vazão e latência média obtidos durante a simulação dos tráfegos propostos. Como cenário de rede é feito uso da fila bi síncrona com codificação de ponteiros Gray e Johnson e da fila síncrona combinada com filas bi síncronas.....	106
Tabela 7: Valores de área em número de LUTs de quatro entradas e consumo de energia. O consumo de energia (em nanoJoule/s) é dado em: (i) Consumo de energia dos sinais de relógio; (ii) Lógica combinacional do circuito; (iii) Lógica sequencial do circuito. ....	107
Tabela 8: Valores de área em número de LUTs de quatro entradas e consumo de energia. O consumo de energia (em nanoJoule/s) é dado em: (i) Consumo de energia dos sinais de relógio; (ii) Lógica combinacional do circuito; (iii) Lógica sequencial do circuito. ....	109
Tabela 9: Valores de vazão e latência média para os cenários de tráfego simulados para os sete cenários de rede que variam o algoritmo de roteamento. ....	110
Tabela 10: Valores de área em número de LUTs de quatro entradas, número de BUFG e consumo de energia. O consumo de energia (em nanoJoule/s) é dado em: (i) Consumo de energia dos sinais de relógio; (ii) Lógica combinacional do circuito; (iii) Lógica sequencial do circuito.....	111
Tabela 11: Valores de vazão e latência média do tráfego obtidos durante a simulação dos cenários de rede síncrona e não síncrona propostos.....	113
Tabela 12: Valores de vazão e latência média dos cenários de tráfego sintético e de um modelo de aplicação CDCM obtidos durante a simulação. ....	116
Tabela 13: Valores de área em número de LUTs de quatro entradas e consumo de energia. O consumo de energia (em nanoJoule/s) é dado em: (i) Consumo de energia dos sinais de relógio; (ii) Lógica combinacional do circuito; (iii) Lógica sequencial do circuito. ....	117

## LISTA DE EQUAÇÕES

Equação 1: Equação que calcula o momento de criação de cada pacote a partir do tamanho do pacote, da capacidade máxima de transmissão e da taxa de injeção do pacote.....	66
Equação 2: Equação que calcula a função de probabilidade seguindo uma distribuição normal de uma taxa de injeção informada. ....	67
Equação 3: Equação que calcula a função de probabilidade seguindo uma distribuição exponencial de uma taxa de injeção informada. ....	68
Equação 4: Equação que realiza o cálculo da latência de um pacote. A latência é calculada a partir da subtração do instante em que o pacote está apto a entrar na rede do instante em que o último flit do pacote deixou a rede. ....	90
Equação 5: Equação que realiza o cálculo da vazão de um pacote. A vazão é calculada a partir da divisão do tamanho do pacote pelo tempo gasto (latência) de transmissão do pacote. Logo após o valor é multiplicado por 1000 para ser expresso em Mbps. ....	90
Equação 6: Equação que calcula a latência média de um conjunto de pacotes a partir do somatório das latências medidas dos pacotes dividido pelo número total de pacotes. ....	92
Equação 7: Equação que calcula o desvio padrão da latência de um conjunto de pacotes. ....	92
Equação 8: Equação que calcula a vazão ideal de um pacote pela divisão do tamanho do pacote em bits pela latência ideal do pacote em tempo absoluto. Depois de calculado o valor é multiplicado por 1000 para ser convertido em Mbps. ....	93
Equação 9: Equação que calcula a vazão média de um conjunto de pacotes, a partir do somatório das vazões calculadas dos pacotes dividido pelo número total de pacotes. ....	93
Equação 10: Equação que calcula o desvio padrão da vazão de um conjunto de pacotes.....	94

## LISTA DE SIGLAS

ACP	APPLICATION COMMUNICATION PATTERN
BUFG	GLOBAL BUFFER
CAFES	COMMUNICATION ANALYSIS FOR EMBEDDED SYSTEMS
CDCG	COMMUNICATION DEPENDENCE AND COMPUTATION GRAPH
CDCM	COMMUNICATION DEPENDENCE AND COMPUTATION MODEL
CRG	COMMUNICATION RESOURCE GRAPH
DRAM	DYNAMIC RANDOM ACCESS MEMORY
EEPROM	ELECTRICALLY ERASABLE PROGRAMMABLE READ-ONLY MEMORY
FPGA	FIELD PROGRAMMABLE GATE ARRAY
GALS	GLOBALLY ASYNCHRONOUS LOCALLY SYNCHRONOUS
HPC	HIGH PERFORMANCE COMPUTING
IP	INTELLECTUAL PROPERTY
LUT	LOOK-UP TABLE
MPEG	MOVING PICTURE EXPERTS GROUP
NoC	NETWORK ON CHIP
OCP	OPEN CORE PROTOCOL
RTL	REGISTER TRANSFER LEVEL
SOC	SYSTEM ON CHIP
SVN	SUBVERSION
TCL	TOOL COMMAND LANGUAGE
TL	TRANSACTION LEVEL
TSMC	TAIWAN SEMICONDUCTOR MANUFACTURING COMPANY
VCD	VALUE CHANGE DUMP
VHDL	VHSIC HARDWARE DESCRIPTION LANGUAGE

VHSIC

VERY HIGH SPEED INTEGRATED CIRCUIT

# SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO .....</b>	<b>17</b>
1.1	MOTIVAÇÃO DO TRABALHO .....	19
1.2	OBJETIVOS DO TRABALHO .....	20
1.3	ORGANIZAÇÃO DO DOCUMENTO .....	20
<b>2</b>	<b>TRABALHOS RELACIONADOS .....</b>	<b>23</b>
2.1	FEN ET AL. ....	23
2.2	KREUTZ ET AL. ....	24
2.3	PANADES ET AL. ....	25
2.4	OST ET AL. ....	26
2.5	PONTES ET AL. ....	28
2.6	BONONI ET AL. ....	29
2.7	TEDESCO ET AL. ....	30
2.8	LIU ET AL. ....	31
2.9	HONG ET AL. ....	32
2.10	BRUCH ET AL. ....	33
2.11	SCHEMMER ET AL. ....	34
2.12	GRATZ ET AL. ....	35
2.13	ROSA ET AL. ....	36
2.14	LOTLIKAR ET AL. ....	37
2.15	PANDA ET AL. ....	39
2.16	CHANG ET AL. ....	40
2.17	ZAYTOUN ET AL. ....	41
2.18	POSICIONAMENTO DO TRABALHO EM RELAÇÃO AO ESTADO DA ARTE .....	43
<b>3</b>	<b>GERAÇÃO DE REDES INTRACHIP NÃO SÍNCRONAS .....</b>	<b>47</b>
3.1	O AMBIENTE ATLAS E A REDE HERMES-G .....	47
3.1.1	<i>O Ambiente ATLAS .....</i>	47
3.1.2	<i>A Rede HERMES-G .....</i>	48
3.2	PROCESSO DE PARAMETRIZAÇÃO DA REDE HERMES-G .....	54
3.3	PROJETO DA INTERFACE GRÁFICA DO AMBIENTE DE GERAÇÃO .....	55
3.3.1	<i>Geração e Definição de Relógios .....</i>	56
3.4	PROCESSO DE GERAÇÃO DA REDE HERMES-G .....	60
3.4.1	<i>Criação dos Diretórios e Arquivos do Projeto da Rede .....</i>	60
3.4.2	<i>Geração dos Roteadores e seus Componentes Internos .....</i>	60
<b>4</b>	<b>GERAÇÃO DE TRÁFEGO PARA REDES INTRACHIP NÃO SÍNCRONAS .....</b>	<b>65</b>
4.1	CARACTERIZAÇÃO DE TRÁFEGO SINTÉTICO .....	65
4.1.1	<i>Distribuição Espacial .....</i>	65
4.1.2	<i>Distribuição Temporal .....</i>	66
4.2	AMBIENTE DE GERAÇÃO DE TRÁFEGO SINTÉTICO .....	69
4.2.1	<i>Definição do Arquivo de Projeto do Tráfego .....</i>	70
4.2.2	<i>Projeto da Interface do Gerador de Tráfego Sintético .....</i>	70
4.2.3	<i>Geração e Formato Intermediário dos Pacotes do Tráfego .....</i>	72
4.2.4	<i>Geração do Testbench para Tráfego Sintético .....</i>	73
4.2.5	<i>Validação da Proposta .....</i>	78

<b>4.3</b>	<b>AMBIENTE DE GERAÇÃO DE TRÁFEGO DE MODELO DE APLICAÇÕES</b>	<b>79</b>
4.3.1	<i>O Ambiente CAFES: Criação e Mapeamento de um Modelo de Aplicação</i>	80
4.3.2	<i>Formato de Representação de um Modelo de Aplicação</i>	82
4.3.3	<i>Adaptação da Interface do Gerador de Tráfego</i>	83
4.3.4	<i>Geração e Formato Intermediário dos Pacotes do Tráfego</i>	84
4.3.5	<i>Geração do Testbench para Modelos de Aplicação</i>	84
<b>5</b>	<b>AMBIENTE DE AVALIAÇÃO DE TRÁFEGO PARA REDES INTRACHIP NÃO SÍNCRONAS</b>	<b>89</b>
<b>5.1</b>	<b>MÉTRICAS DE AVALIAÇÃO DE TRÁFEGO</b>	<b>89</b>
5.1.1	<i>Latência</i>	89
5.1.2	<i>Vazão</i>	90
<b>5.2</b>	<b>MEDIDAS ESTATÍSTICAS NAS MÉTRICAS DE AVALIAÇÃO</b>	<b>90</b>
5.2.1	<i>Latência</i>	90
5.2.2	<i>Vazão</i>	93
<b>5.3</b>	<b>DISTRIBUIÇÃO DE VAZÕES DE UM TRÁFEGO</b>	<b>94</b>
<b>5.4</b>	<b>DISTRIBUIÇÃO DE LATÊNCIAS DE UM TRÁFEGO</b>	<b>95</b>
<b>5.5</b>	<b>ANALISADOR DE LATÊNCIAS</b>	<b>96</b>
<b>5.6</b>	<b>RELATÓRIO GLOBAL</b>	<b>96</b>
<b>6</b>	<b>RESULTADOS OBTIDOS</b>	<b>99</b>
6.1	<b>ESTUDO DE CASO 1</b>	<b>99</b>
6.2	<b>ESTUDO DE CASO 2</b>	<b>104</b>
6.3	<b>ESTUDO DE CASO 3</b>	<b>107</b>
6.4	<b>ESTUDO DE CASO 4</b>	<b>110</b>
6.5	<b>ESTUDO DE CASO 5</b>	<b>113</b>
<b>7</b>	<b>CONCLUSÕES E TRABALHOS FUTUROS</b>	<b>119</b>
7.1	<b>CONTRIBUIÇÕES DO TRABALHO</b>	<b>119</b>
7.2	<b>TRABALHOS FUTUROS</b>	<b>120</b>
	<b>REREFÊNCIAS BIBLIOGRÁFICAS</b>	<b>123</b>

# 1 INTRODUÇÃO

A evolução da tecnologia de fabricação de circuitos integrados afeta diretamente o processo de projeto de produtos eletrônicos. Parte desta evolução relaciona-se ao crescimento do número de transistores de circuitos integrados, componentes fundamentais na concepção dos produtos eletrônicos. Em [MOO65], Moore identificou que a cada 18 a 24 meses dobra-se a quantidade de transistores que se podia construir sobre a mesma área de silício, resultando em uma redução de custos e aumento do desempenho dos circuitos integrados. Este fato continuou a observar-se em décadas posteriores em maior ou menor escala, dando origem à chamada Lei de Moore.

A indústria de semicondutores emprega a Lei de Moore como incentivo ao crescimento da complexidade dos circuitos. O desafio para atender os pressupostos da lei leva indústria e academia a investirem esforço em pesquisa e desenvolvimento. Isto pode ser corroborado citando que uma organização, o *International Technology Roadmap for Semiconductors* ou ITRS, tem como objetivo analisar e prever o crescimento do mercado de semicondutores. Como resultado destes esforços, em [ARD10], Arden et al. demonstraram existirem subsídios tecnológicos para o projeto de circuitos integrados capazes hoje de superar os pressupostos da Lei de Moore. Em [ITR11a], [ITR11b] e [ITR11c] apresentam-se alguns dos desafios a serem vencidos para que processos de fabricação futuros sejam capazes de superar o estado da arte atual. Estes documentos detalham, por exemplo, que até 2016 será possível desenvolver circuitos com 14.2nm de *min-feature size*, fabricados sobre *wafers* de silício com dimensões de até 450 mm de diâmetro.

O reflexo do esforço e da disponibilidade tecnológica oferecida pode ser observado em produtos que fazem uso das tecnologias de fabricação mais recentes. Até onde o Autor pode investigar, o menor processo de fabricação comercial utilizado nos dias atuais é descrito em [INT12] com uso da tecnologia de fabricação com 22 nm de *min-feature-size*, e fazendo uso de *wafers* de silício de 300 mm. Como outro exemplo, em [NVI12] apresenta-se um circuito integrado que contém 7.1 bilhões de transistores, capaz de sustentar uma vazão de 1 Teraflops, ou seja, um trilhão de operações de ponto flutuante em precisão dupla por segundo. Previsões indicam que este circuito estará disponível no terceiro trimestre de 2012 com tecnologia de fabricação 28nm. Em [INT11] apresenta-se um circuito integrado com função de coprocessador, que também apresenta vazão a nível de Teraflops. Este circuito já é fabricado utilizando tecnologia de fabricação 28nm, e aplicado em supercomputadores de propósito específico.

Ainda durante as últimas décadas, o projeto de circuitos integrados evoluiu do desenho manual de transistores ao uso de linguagens de descrição de hardware e ferramentas de síntese, que a partir da descrição abstrata efetuam a geração automática do projeto em nível físico do circuito. Um dos maiores desafios do projeto de sistemas eletrônicos está no atendimento de restrições e pré-requisitos do sistema. Uma das alternativas para superar estes desafios está no desenvolvimento de ferramentas de automação e geração automática de circuitos. Estas devem ser tais que possam utilizar como entrada as restrições e pré-requisitos do circuito a ser gerado. Em [ITR11a] aponta-se que até 2015, o nível comportamental do projeto de sistemas eletrônicos

ocupará cerca de 50% do esforço de projeto, o que reforça a necessidade de desenvolvimento de ferramentas de automação e geração automática de circuitos.

Nas últimas duas décadas, o desenvolvimento de sistemas eletrônicos compostos por múltiplos elementos de processamento implicava sistemas compostos por diversos circuitos integrados distintos. Com a disponibilidade tecnológica atual, é possível desenvolver sistemas eletrônicos completos em um único circuito integrado. Para esse tipo de circuito, se dá o nome de sistema integrado em chip (do inglês *System on Chip* ou SoC) onde todos os componentes básicos do sistema eletrônico estão encapsulados em um mesmo circuito integrado. Tal tipo de sistema reúne uma série de vantagens, algumas delas diretamente relacionadas à redução dos custos de encapsulamento e de manufatura do produto. Esta tendência de projeto tende a reduzir os esforços em tempo de projeto. Por consequência, o sistema se torna mais competitivo. Outra técnica para aumentar a competitividade, e capaz de fazer uso da disponibilidade tecnológica para aumentar o número de funcionalidades dos sistemas eletrônicos, é o reuso de componentes que compõem um sistema eletrônico.

Com o reuso maciço de componentes, o projeto de um SoC passa a consistir na montagem do sistema a partir de uma maioria de módulos pré-validados e pré-caracterizados. Contudo, a interconexão em si de grande número de módulos não é tarefa trivial, sobretudo quando da interconexão depende a viabilidade do sistema como um todo. Uma arquitetura de interconexão deve ser capaz de permitir que componentes comuniquem-se entre si de forma eficiente. Diversos modelos de arquiteturas de interconexão foram propostos e difundidos. Em geral, uma arquitetura de interconexão deve garantir: (i) Eficiência energética e confiabilidade; (ii) Escalabilidade e largura de banda; (iii) Reusabilidade. Um modelo de arquitetura de interconexão capaz de atender a estes requisitos, e que contém atualmente grande apelo comercial e gera muitos esforços em pesquisa são as redes intrachip, também referenciadas na literatura pelo termo *Network on Chip* ou NoC. Assume-se por convenção neste trabalho, que o termo rede intrachip será referenciado unicamente pelo termo *rede*. Nestas, as decisões de arbitragem e roteamento da informação podem ser tratadas de maneira distribuída, conforme as políticas de roteamento/arbitragem específicas adotadas. Arquiteturas de interconexão derivam suas propriedades da adaptação de conceitos provenientes de redes de computadores e/ou de sistemas distribuídos, existindo assim semelhanças, quer na disposição de protocolos, ou no encaminhamento de informação, e até na sincronização da informação.

Por outro lado, o projeto dos circuitos integrados atuais faz uso majoritário do paradigma de projeto síncrono, em que um único sinal sincronizador é usado para coordenar todos os eventos que acontecem no circuito. Este paradigma foi adotado por favorecer a simplicidade e exigir pouca lógica para sincronizar eventos. Devido à crescente redução na escala dos componentes eletrônicos, o número de componentes a serem sincronizados, e por consequência o número de fios necessários para interconectar estes componentes ao sinal sincronizador, aumentam na mesma proporção. Em [AMD05], os autores apontam que 45% do total da potência consumida por um processador de alto desempenho da época, era devida ao processo de distribuição do sinal de sincronização (relógio) do circuito.

Soluções para as restrições impostas pelo modelo síncrono, que estão ligadas diretamente ao consumo de energia e à dissipação de potência, são temas de pesquisa e de interesse crescente. Dentre algumas das soluções apresentadas na literatura, destacam-se os circuitos que independem de sincronizadores, e circuitos que fazem uso de múltiplos sinais de sincronização. O modelo que independe de sincronizadores, referenciado por modelo assíncrono, faz uso de protocolos de comunicação e sincronização locais no lugar da lógica baseada em um sinal de sincronização global. As principais vantagens dessa abordagem estão na eliminação dos problemas causados pelo sinal de sincronização, apresentando maior robustez se comparada a circuitos que fazem uso do sinal de sincronização. Esta técnica é pouco adotada em SoCs hoje em dia, pela carência de ferramentas de projeto e de recursos humanos aptos para dar suporte à tecnologia.

Outra solução para libertar-se do modelo síncrono é aquela que faz uso de múltiplos sinais de sincronização. Nesta abordagem, conhecida em inglês pelo termo *Globally Asynchronous Locally Synchronous* ou GALS, o circuito é particionado, e as partes resultantes costumam ser referenciadas na literatura como ilhas, onde cada ilha possui um sinal de sincronização, podendo estes sinais serem diferentes uns dos outros. Cada sinal de sincronização possui uma árvore de distribuição de relógio própria. Para garantir a comunicação entre as ilhas, usam-se interfaces de sincronização. De acordo com o ITRS [ITR11a], o uso da técnica GALS no projeto de circuitos com foco em baixo consumo de energia, é considerada como a segunda melhor técnica para redução no consumo de energia dinâmica de sistemas complexos, no mesmo nível de qualidade de técnicas de DVFS, do inglês *Dynamic Voltage and Frequency Scaling*, em que a voltagem e a frequência do circuito são ajustadas dinamicamente, conforme a demanda de operação do circuito.

## 1.1 MOTIVAÇÃO DO TRABALHO

A carência de recursos de suporte à construção de sistemas não síncronos dá sentido a um esforço em pesquisa para fazer evoluir as técnicas de especificação, verificação e síntese de SoCs GALS em geral, e de arquiteturas de comunicação para tais SoCs em particular.

A pesquisa em redes de interconexão que utilizam a abordagem GALS permite explorar alternativas no projeto de circuitos em que as restrições ocasionadas pela distribuição e consumo de potência do sinal de relógio possam ser evitadas. Além disso, a abordagem GALS favorece o reuso de componentes. Componentes pré-projetados e pré-validados operando em diferentes domínios de frequência podem garantir que requisitos de projeto como o tempo máximo para que um produto chegue ao mercado, sejam atendidos. Tal realidade reforça a motivação para existir um conjunto de técnicas para gerar e avaliar o tráfego em NoCs que operem em diferentes domínios de frequência, descrevendo as mesmas em nível de abstração mais baixo, como o nível de transferência entre registradores (em inglês *register transfer level* ou RTL), ao contrário da maioria dos trabalhos disponíveis na bibliografia atual.

O ambiente ATLAS [TED05], proposto e mantido pelo grupo de pesquisa o qual o Autor é membro, dá suporte hoje ao projeto de redes síncronas apenas. Assim, existe uma infraestrutura de software extensa da qual se pode partir para desenvolver e validar técnicas de especificação, verificação e síntese de NoCs não-síncronas. Ainda, uma rede com suporte a SoCs GALS já foi proposta e validada pelo grupo de pesquisa do Autor, e denomina-se HERMES-G [PON08]. Ou seja, à infraestrutura de software disponível (ATLAS) se junta um suporte de hardware já previamente projetado e validado (HERMES-G).

## 1.2 OBJETIVOS DO TRABALHO

Como objetivo geral neste trabalho, pretendeu-se desenvolver uma infraestrutura capaz de viabilizar a geração e avaliação da rede GALS HERMES-G. Para que o objetivo geral fosse atingido, um conjunto de objetivos específicos foi desenvolvido. Estes consistem na inserção de suporte na ferramenta ATLAS para:

- ❖ A geração automatizada de instâncias quaisquer de redes GALS HERMES-G.
- ❖ A geração de tráfego sintético para redes GALS HERMES-G.
- ❖ A partir de grafos de aplicação do tipo CDCG [MAR05], possibilitar gerar tráfego para redes GALS HERMES-G.
- ❖ Permitir a avaliação de tráfegos sintético sem redes GALS HERMES-G.
- ❖ Possibilitar a avaliação de tráfegos gerados a partir de grafos CDCG em redes HERMES-G.

## 1.3 ORGANIZAÇÃO DO DOCUMENTO

O restante do documento é detalhado em 6 capítulos, cada um contendo os seguintes assuntos:

- ❖ Capítulo 2: Apresenta uma revisão do estado da arte da literatura relacionada ao presente trabalho. Também se compara a presente proposta com os trabalhos revisados.
- ❖ Capítulo 3: Descreve em detalhes como é feito o processo de geração automática da rede HERMES-G, uma das contribuições do trabalho.
- ❖ Capítulo 4: Descreve o processo de geração de tráfego sintético e tráfego baseado em aplicações para redes do tipo HERMES-G, outra das contribuições do trabalho.
- ❖ Capítulo 5: Descreve o processo de avaliação de tráfego sintético e tráfego baseado em modelos de aplicações para redes do tipo HERMES-G, mais uma das contribuições.
- ❖ Capítulo 6: Apresenta resultados do trabalho. Nele descrevem-se os experimentos realizados, mostrando os ganhos obtidos em desempenho em redes do tipo HERMES-G. Ainda neste Capítulo, mostra-se o impacto de um tráfego sintético gerado pelo ambiente de geração de tráfego da ATLAS comparado a um tráfego baseado em modelo de aplicação real.

- ❖ Capítulo7: Apresenta as conclusões e resume os resultados alcançados. Além disso, o Capítulo descreve um conjunto de trabalhos futuros a desenvolver a partir do que foi apresentado neste trabalho.

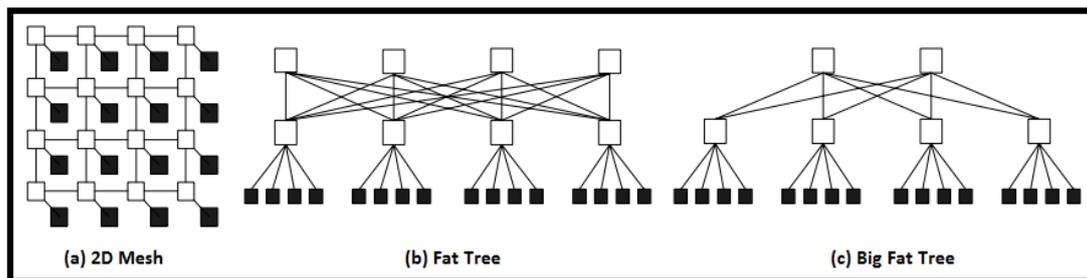


## 2 TRABALHOS RELACIONADOS

Este Capítulo aborda um conjunto de trabalhos relacionados aos temas de pesquisa explorados por este trabalho. Dentre os temas pesquisados pode-se citar: (i) O projeto de redes intrachip capazes de operar simultaneamente com diferentes frequências de operação, e que de alguma forma, sejam passíveis de sofrer parametrização, e que estejam vinculadas a um ambiente de geração automatizada; (ii) A caracterização do tráfego em redes intrachip na literatura disponível; (iii) Métricas de avaliação de tráfego, como são calculadas e como o desempenho de rede é mensurado. O Capítulo conclui com uma compilação e comparação dos temas pesquisados com a proposta deste trabalho.

### 2.1 FEN ET AL.

Fen et al. [FEN07] propõem criar diretrizes para que projetistas de redes intrachip escolham configurações de redescapazes de atender os requisitos de desempenho das aplicações que usam a rede intrachip como meio de comunicação.



**Figura 1: Topologias suportadas pela ferramenta OPNET: (a) 2D Mesh (b) Fat Tree (c) Big Fat Tree.**

Os autores utilizam o ambiente OPNET para geração e simulação de redes intrachip. Durante a geração, o autor descreve a possibilidade de gerar redes com topologias malha 2D, árvore gorda e árvore gorda *butterfly*. A Figura 1 ilustra a disposição dos roteadores e dos elementos de processamento em cada uma das topologias possíveis de serem geradas. Para cada topologia, o gerador permite utilizar duas políticas de chaveamento de pacotes, a *wormhole* e a *virtual cut through*. Cada uma das portas dos roteadores faz uso de uma fila, com capacidade máxima de até dois pacotes, sendo que cada pacote possui tamanho igual a 256bits.

O processo de geração do tráfego proposto leva em consideração a distribuição espacial que define a relação entre a origem e o destino dos pacotes e a distribuição temporal que define o intervalo entre a geração dos pacotes do tráfego. Os autores fazem uso de duas distribuições espaciais, a primeira, define os destinos de maneira aleatória, e a segunda, define os destinos do tráfego seguindo um critério de afinidade, em que os endereços vizinhos mais próximos do endereço gerador têm maior probabilidade de serem os destinos do tráfego. A distribuição temporal não é detalhada, porém, o autor durante a avaliação do trabalho utiliza cenários

variados com taxas de injeção definidas, o que caracteriza tráfego com distribuição temporal uniforme.

O processo de avaliação de tráfego utiliza duas métricas de desempenho. A primeira destas é a vazão da rede, caracterizada pela média de bits recebidos pelos elementos de processamento acoplados a esta. A segunda métrica é a latência de pacote, caracterizada pelo intervalo de tempo entre a entrada e a saída de um pacote da rede, tempo este medido em ciclos de relógio da rede.

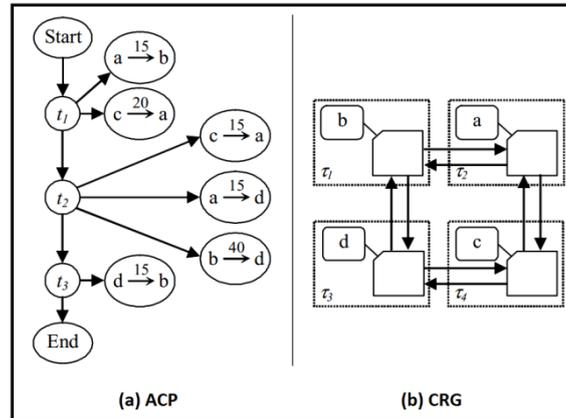
Por fim, após a realização de um conjunto de experimentos, variando as características de rede propostas pelo trabalho, os autores concluem que a arquitetura de rede com topologia árvore gorda utilizando chaveamento de pacotes *wormhole*, foi aquela capaz de atingir a maior vazão de rede e a menor latência de pacote. Os autores destacam ainda que durante a avaliação dos experimentos propostos, as características impostas ao tráfego foram decisivas na obtenção dos resultados.

## 2.2 KREUTZ ET AL.

Kreutz et al. [KRE05], propõem o desenvolvimento de uma técnica capaz de encontrar a arquitetura de rede intrachip ótima para uma aplicação, onde para isso deva ser levado em consideração o compromisso de latência e consumo mínimo de energia.

Os autores descrevem que a proposta do trabalho faz uso de redes com topologias diretas e indiretas. Nas topologias diretas, o trabalho dá suporte a malha 2D e *toro*, usando roteamento determinístico XY. Na topologia indireta, o trabalho dá suporte a *árvore gorda* usando roteamento adaptativo. Em ambas as topologias são utilizadas as técnicas de chaveamento de pacotes *wormhole* e controle de fluxo baseado em créditos. Os autores não descrevem existir suporte tanto a parametrização quanto de uma ferramenta de geração automatizada de redes.

O processo de geração do tráfego é feito através de modelos. O trabalho faz uso de dois modelos, o primeiro denominado *Application Communication Pattern (ACP)*, que define os custos de comunicação e dependências entre as tarefas. O segundo chama-se *Communication Resource Graph (CRG)*, que define a arquitetura de comunicação onde a tarefa deve ser mapeada. A Figura 2 ilustra um exemplo da proposta, contendo um modelo ACP que define uma aplicação e um modelo CRG que define a arquitetura de comunicação, juntamente com a aplicação mapeada.



**Figura 2: Modelos propostos por Kreutz et al. (a) Modelo ACP, define os custos de computação e dependências entre as tarefas. (b) Modelo CRG, define a arquitetura de comunicação onde à tarefa vai ser mapeada.**

No trabalho de Kreutz et al., os autores propõem a utilização de um algoritmo de mapeamento e minimização de caminhos entre tarefas, denominado *Tabu Search*. Este algoritmo busca heurísticamente o melhor mapeamento possível, procurando encontrar um compromisso entre a menor latência e o menor consumo de energia. O processo de avaliação de tráfego utiliza como métrica de desempenho a latência média gasta por todos os pacotes, medida em ciclos de relógio, e o consumo de energia gasto por uma aplicação para ser executada, medida em micro joules.

Por fim, após a realização de um conjunto de experimentos, variando as características de redes propostas, e usando um conjunto definido de aplicações descritas pela proposta do trabalho, os autores concluem que a rede *árvore gorda* foi a que apresentou a menor latência dentre as redes propostas e as aplicações utilizadas, mas que a topologia malha 2D foi a que apresentou o melhor compromisso entre latência e consumo de energia.

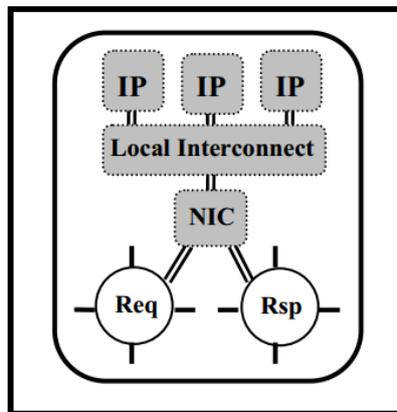
## 2.3 PANADES ET AL.

Panades et al. [PAN06], apresentam uma rede intrachip chamada DSPIN, com suporte a serviços de entrega de pacotes na rede, capazes de sustentar em um fluxo de pacotes um conjunto de restrições como latência máxima e vazão mínima.

A rede proposta pelo autor é projetada como uma arquitetura de rede intrachip voltada para multiprocessadores com memória compartilhada. DSPIN é caracterizada por uma topologia malha 2D, chaveamento de pacotes *wormhole* e algoritmo de roteamento determinístico XY. As principais características da rede DSPIN são: (i) O uso de canais virtuais nas portas de entrada dos roteadores, onde cada canal é responsável por transportar uma classe de serviço. A rede proposta dá suporte a dois tipos de pacote, um considerado como de melhor esforço, do inglês *best effort* e outro com garantia de serviço, do inglês *guaranteed service*; (ii) Suporte a técnica GALS, entre roteadores e IPs. Entre roteadores é possível existir defasagem entre as bordas dos relógios dos roteadores, mas todos os relógios devem trabalhar na mesma frequência. Entre os roteadores e

IPs a relação é inversa, sendo que é possível os relógios operarem em diferentes frequências, mas devendo estar sincronizados na mesma fase.

A rede DSPIN considera um módulo de processamento como um cluster, podendo ser formados por diversos sub módulos, interconectados por sua rede local. Cada cluster é conectado a um único roteador, que possui dois canais físicos, um para enviar e outro para receber dados da rede. A Figura 3 ilustra uma arquitetura de cluster composta por três núcleos de propriedade intelectual (do inglês, *intellectual property core*, ou IP ou *IP core*), uma rede local e um adaptador de rede. Durante o processo de avaliação do trabalho, o autor cita algumas variações nas características da rede DSPIN, mas não detalha em nenhum momento se a rede é passível de parametrização, ou da existência de um ambiente para geração automática da rede.



**Figura 3: Arquitetura típica de um cluster na rede DSPIN, formada por três IPs, uma arquitetura de interconexão local e um adaptador de rede, que conecta o cluster a um roteador, este formado por duas portas, uma de envio e outro de recebimento de dados.**

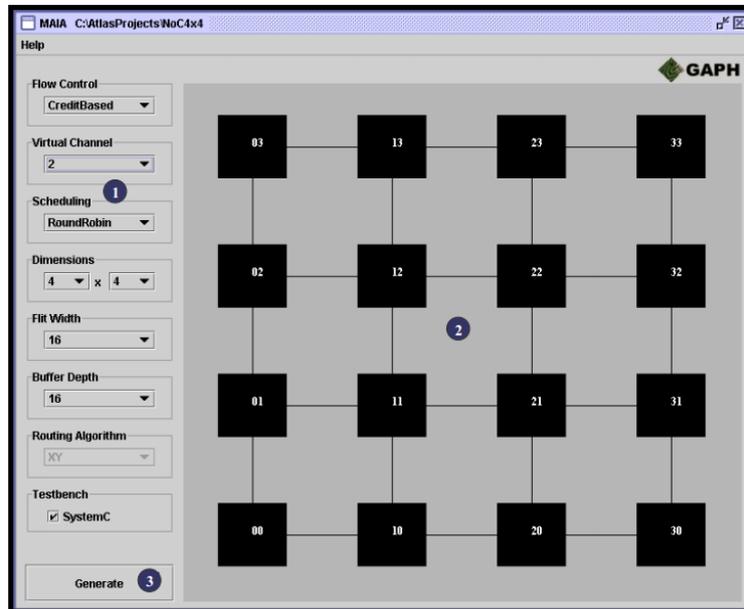
O processo de geração de tráfego é pouco detalhado, os autores citam durante o processo experimental fazer uso de distribuição espacial aleatória e de distribuição temporal uniforme, variando o comprimento dos pacotes em um intervalo de um a dezesseis *flits*, sendo o *flit*, de tamanho fixo em 34 bits. Não é mencionada a maneira como o comprimento dos pacotes é variado. O processo de avaliação de tráfego utiliza como métrica de desempenho a latência medida em ciclos de relógio. Como processos experimentais, são propostos cenários de tráfego com variação nas taxas de injeção. Os autores concluem que durante a utilização do serviço *guaranteed service*, mesmo tráfegos injetados na taxa máxima da rede, não há variação na latência dos pacotes. Em contrapartida, durante a utilização do serviço *best effort*, tráfegos injetados em 25% da capacidade da rede, apresentam variações bruscas no aumento da latência.

## 2.4 OST ET AL.

Ost et al.[OST05], apresentam o ambiente MAIA, que possibilita geração e avaliação de redes intrachip. O trabalho apresenta o ambiente e demonstra algumas das funcionalidades existentes.

O ambiente MAIA, possibilita a geração de redes intrachip, modeladas através de *templates* parametrizáveis da rede HERMES. Esta rede dá suporte a diferentes topologias de redes, comprimentos nas filas de entrada dos roteadores, algoritmos de roteamento determinísticos e

adaptativos e diferentes técnicas de controle de fluxo. Durante o processo de geração da rede, o projetista pode optar pela geração de interfaces externas a rede, existindo duas possibilidades, a interface nativa da rede ou uma interface *open core protocol* (ou OCP). A Figura 4 ilustra a interface gráfica de usuário de geração de redes do ambiente MAIA, e descreve suas funcionalidades.



**Figura 4: Interface gráfica de geração de redes do ambiente MAIA. (1) Permite ao usuário selecionar os parâmetros da rede. (2) Representa uma rede 4x4 Mesh. (3) Dispara o ambiente que gera a rede selecionada.**

O processo de geração de tráfego do ambiente MAIA, é feito a partir de um módulo do ambiente chamado *Traffic Generation*, responsável por implementar a interface gráfica que permite parametrizar o tráfego, e por gerar os arquivos que compõem o tráfego a ser transmitido na rede. O gerador permite variar os parâmetros que definem o tráfego, sendo eles, a taxa de injeção do tráfego, o número de pacotes, o tamanho dos pacotes e os destinos do tráfego, que podem ser fixados em um endereço da rede, ou definidos de maneira aleatória.

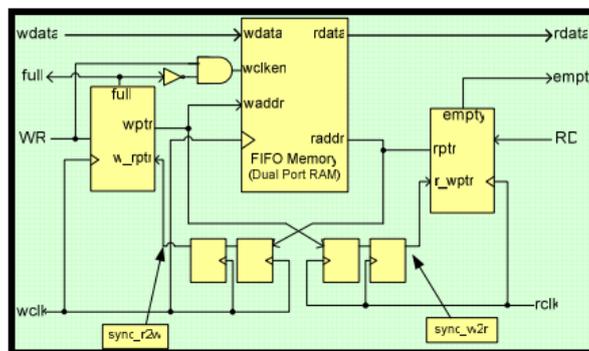
A avaliação do tráfego é feita a partir de um módulo do ambiente MAIA chamado *Traffic Analysis*, que usa arquivos que coletam informações dos pacotes durante a simulação da rede. O processo de avaliação é feito após o término da simulação. Dentre as métricas de desempenho utilizadas para avaliar o tráfego estão o número de pacotes recebidos, o tempo médio de entrega dos pacotes, o tempo total de entrega dos pacotes e o tempo total de simulação da rede, todos eles calculados em ciclos de relógio da rede.

Durante o processo experimental os autores propõem variar as características de algumas redes disponibilizadas pelo ambiente, obtendo algumas conclusões como que o tráfego com menor tamanho de pacote, obtém melhor desempenho em redes com algoritmos de roteamento adaptativos e tráfegos com pacotes de maior tamanho, obtém melhor desempenho em algoritmos de roteamento determinísticos.

## 2.5 PONTES ET AL.

Pontes et al. [PON08], propõem o desenvolvimento de dois roteadores GALS, baseados no roteador HERMES, ambos com objetivo de redução no consumo de energia da rede. As redes propostas por este trabalho são: (i) HERMES-G; (ii) HERMES-GLP. Ambos os roteadores são baseados no roteador HERMES, descrito originalmente por [MOR04]. Ambas as redes são compostas pelas seguintes características, roteamento determinístico XY, controle de fluxo baseado em créditos, chaveamento de pacotes *wormhole* e largura dos canais dos roteadores igual a 16bits.

O principal componente que diferencia as redes propostas da rede HERMES são as filas utilizadas nos roteadores. Na rede HERMES, é feito uso de uma fila síncrona como componente de entrada nos roteadores, onde leituras e escritas entre os roteadores são feitas na mesma fase e na mesma frequência. As redes HERMES-G e HERMES-GLP utilizam uma fila bi síncrona, que permite que escritas e leituras na fila sejam feitas tanto na mesma, como em diferentes frequências, podendo ou não estar na mesma fase. A Figura 5 ilustra a arquitetura da fila, composta por uma memória que armazena os dados. Também, existe na arquitetura uma lógica de codificação de ponteiros de leitura e de escrita, que garante que mesmo que ambos os ponteiros utilizados para ler e escrever na fila operem em diferentes domínios de relógio, sempre estejam sincronizados. Além da fila bi síncrona, o roteador HERMES-GLP possui um componente que ajusta a frequência do roteador, do inglês *clock gating* com base em valores de prioridade definidos e implementados através de um campo de prioridade nos pacotes do tráfego.



**Figura 5: Fila bi síncrona utilizada em redes HERMES-G e HERMES-GLP. Nesta arquitetura de fila, a escrita e a leitura podem ou não operar na mesma fase ou na mesma frequência. A codificação de Gray é utilizada nos ponteiros de leitura e de escrita da fila para garantir que ambos os ponteiros estejam sincronizados, mesmo sendo manipulados em diferentes domínios de fase e de frequência.**

O processo de geração de tráfego é pouco detalhado, uma vez que os autores o citam unicamente durante o processo experimental do trabalho. Em resumo, a distribuição espacial do tráfego não é explorada, os autores apenas utilizam tráfego com origens e destinos estáticos. A distribuição temporal é uniforme, usando a taxa máxima de injeção dos IPs. Como avaliação do tráfego, é feito uso da latência média de pacote como métrica de desempenho. Durante o processo experimental, é comparado o tempo adicional ocasionado pelo mecanismo que varia a frequência dos roteadores com base nas prioridades dos pacotes, e quais são as taxas de ativação dos roteadores em uma rede HERMES-GLP. Os resultados demonstram que não houve acréscimo

significante na latência dos pacotes pelo uso do mecanismo que altera a frequência do roteador com base na prioridade nas redes HERMES-GLP. Além disso, os resultados mostram que existem diferenças significativas nas taxas de ativação entre roteadores (sendo esta uma forma de estimar potência de forma grosseira), o que demonstra a principal vantagem da rede HERMES-GLP na redução de energia comparada à rede HERMES-G.

## 2.6 BONONI ET AL.

Bononi et al. [BON07] , comparam diferentes arquiteturas de redes intrachip utilizando tráfego sintético e tráfego real, demonstrando o impacto do tráfego e da arquitetura utilizada nos resultados obtidos. Os autores utilizam o ambiente OMNeT++ para geração e simulação da rede. Este ambiente permite gerar diferentes topologias. O trabalho faz uso de quatro topologias, ilustradas pela Figura 6 e descritas a seguir: (i) Malha 2D; (ii) Anel; (iii) Spidergon; (iv) Crossbar. Em todas as topologias são utilizados algoritmos de roteamento determinísticos mínimos, livres de situações de impasse, do inglês *deadlock*. O simulador utilizado usa precisão em nível de *flit*, onde todos os roteadores se comunicam de maneira síncrona.

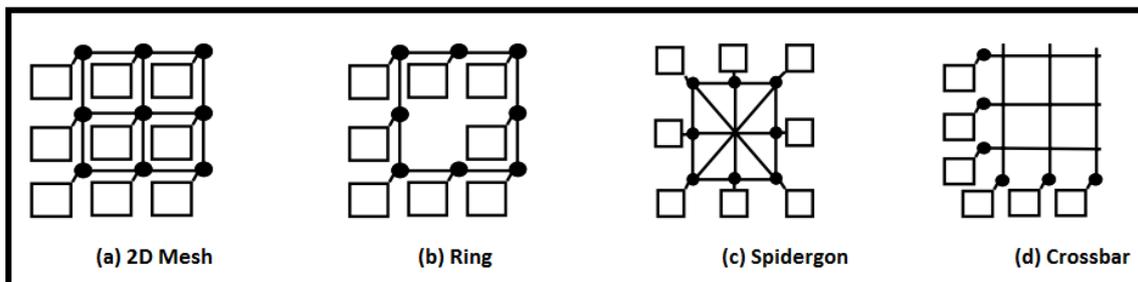


Figura 6: Topologias suportadas pelo ambiente de geração: (a) Malha 2D, (b) Anel, (c) Spidergon, (d) Crossbar.

O processo de geração de tráfego é feito através da ferramenta SCOTCH que a partir de grafos de aplicações, permite modelar tráfego sintético e tráfego real. A ferramenta SCOTCH realiza o particionamento e o mapeamento da aplicação através de dois grafos, um que descreve a aplicação e outro que descreve a topologia e as demais características da rede, e leva em consideração a melhor combinação possível que explore o melhor desempenho da rede.

O processo de avaliação do tráfego utiliza como métrica de desempenho o tempo de processamento de uma aplicação real em ciclos de relógio e a vazão média da rede. Durante os experimentos, os autores avaliam as topologias de rede propostas sobre diferentes cenários de aplicações sintéticas, onde o modelo de particionamento e mapeamento das tarefas é variado. Além disso, os autores fazem uso de um cenário de aplicação real MPEG, utilizando as diferentes topologias de rede propostas. Como resultado, os autores concluem que a topologia de rede *crossbar* foi a que atingiu melhor desempenho dentre as topologias comparadas. Além disso, os autores destacam a importância do particionamento e do mapeamento, sendo eles os critérios mais importantes para obter o melhor desempenho da rede.

## 2.7 TEDESCO ET AL.

Tedesco et al.[TED05a], apresentam métodos para geração e avaliação de tráfego em redes intrachip, onde é proposta uma abordagem alternativa de avaliação de desempenho, que leva em consideração os canais que interligam os roteadores, possibilitando avaliar em quais pontos da rede os requisitos do tráfego não estão sendo atendidos.

Os autores utilizam a rede intrachip HERMES em seu trabalho, esta rede, possui topologia malha 2D, suporte a canais virtuais, roteamento determinístico XY e adaptativo *west-first*, chaveamento de pacotes *wormhole*, controle de fluxo baseado em créditos e *handshake* e suporte a parametrização do comprimento dos canais que interligam os roteadores.

O tráfego proposto faz uso da distribuição espacial complemento para definir os iniciadores e destinatários do tráfego. O intervalo em que os pacotes são colocados na rede é dado de maneira uniforme. Além da taxa de injeção, é possível também a parametrização do número de pacotes em *flits* de cada tráfego injetado por um IP.

O processo de avaliação do tráfego é descrito ao longo do trabalho como a principal contribuição. O modelo proposto permite que o tráfego seja avaliado tanto nos pontos de injeção e coleta de dados da rede, caracterizado como “Abordagem Externa” quanto nos canais que interconectam os roteadores, caracterizado como “Abordagem Interna”. Na abordagem externa, os resultados são obtidos nas interfaces externas da rede, já na abordagem interna, é possível obter detalhes entre os canais que interligam os roteadores, e entender em quais pontos, a rede apresenta maior e menor índice de atividade ou ociosidade. Como métricas de avaliação de desempenho, os autores fazem uso de vazão e de latência, tanto para a abordagem externa, quanto para a abordagem interna.

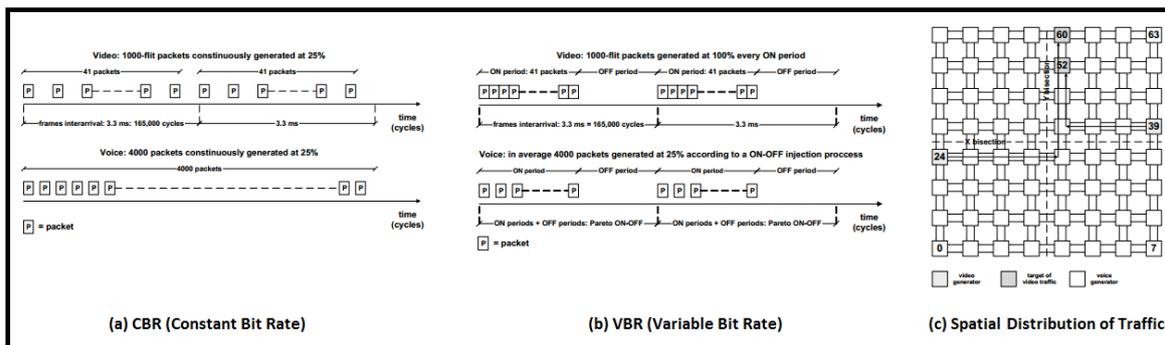
O processo experimental é feito através de cenários de redes HERMES, fazendo uso de algoritmos de roteamento determinísticos e adaptativos, e uso de canais virtuais, que multiplexam um canal físico, o que permite em alguns casos, um melhor aproveitamento da rede. Através da técnica de avaliação interna, o autor demonstra a possibilidade de detectar em quais pontos a rede estava causando a saturação no tempo de transmissão dos pacotes. Além disso, os autores demonstram as vantagens no uso de canais virtuais, capaz de em alguns cenários aumentarem as taxas de tráfego aceito.

Ainda em Tedesco et al. [TED06], os autores desenvolvem um novo trabalho propondo evolução no processo de geração de tráfego, com objetivo de comparar o desempenho de uma rede quando diferentes modelos de tráfegos são utilizados. As questões relacionadas à geração de redes permanecem as mesmas descritas em [TED05a].

O processo de geração de tráfego proposto pelos autores destaca a importância de dirigir a modelagem do tráfego conforme as características de uma aplicação. O trabalho de caracterização do tráfego proposto leva em consideração os requisitos de entrega das aplicações. Os autores propõem um modelo de tráfego capaz de variar os seguintes parâmetros de um tráfego: (i) Tamanho dos pacotes; (ii) Intervalo de geração de um pacote; (iii) Intervalo de ociosidade entre

pacotes. Os autores fazem uso da distribuição de probabilidade normal e Pareto *on-off* para variar os intervalos de geração e ociosidade dos tráfegos.

O ambiente de avaliação de tráfego faz uso das mesmas métricas descritas em [TED05a], sendo que durante o processo experimental, são propostos dois cenários compostos por redes HERMES de tamanho 8x8 com 16bits de largura dos canais entre os roteadores e 8 *flits* de profundidade nas filas de entrada dos roteadores, que operam a 50 MHz. Em ambos os cenários, a distribuição espacial ilustrada pela Figura 7(c) é utilizada, sendo ela um tráfego de voz, gerado pelos roteadores 0 e 7 para o roteador 63, e um tráfego de vídeo, gerado do roteador 24 para o roteador 52, e do roteador 39 para o roteador 60. A diferença entre ambos os cenários está na forma como os pacotes são injetados na rede. O primeiro cenário representado pelo tráfego de voz realiza a injeção de pacotes conforme ilustra a Figura 7(a), seguindo as características de tráfego CBR, do inglês *Constant Bit Rate*, onde a taxa de injeção é contínua ao longo do tempo. O segundo cenário representado pelo tráfego de vídeo realiza a injeção de pacotes conforme ilustra a Figura 7(b), seguindo as características de tráfego VBR, do inglês *Variable Bit Rate* onde a taxa de injeção varia entre períodos *on-off*. A mesma quantidade de pacotes foi utilizada em ambos os tráfegos, injetados sob as mesmas taxas de injeção. Como resultados, os autores concluem que o segundo cenário que utiliza o modelo VBR apresentou melhores resultados em relação à variação da latência média de todos os pacotes.



**Figura 7: (a) Categoria de serviço CBR do inglês *Constant Bit Rate* em que todos os pacotes são injetados na rede na mesma taxa de injeção. (b) Categoria de serviço VBR, do inglês *Variable Bit Rate* em que a taxa de injeção é variável ao longo do tempo. (c) Distribuição espacial do cenário de tráfego composto por um gerador de vídeo e um gerador de voz.**

## 2.8 LIU ET AL.

Liu et al. [LIU07], propõem uma abordagem de caracterização de tráfego em NoCs a partir de aplicações reais. O trabalho tem por objetivo demonstrar a complexidade envolvida no processo de caracterização do tráfego e nos ganhos da proposta comparadas a distribuições de tráfego sintéticas, amplamente utilizadas por trabalhos relacionados.

Os autores utilizam o ambiente MCSL, que dá suporte a simulação de redes com precisão de ciclo, e dá suporte as topologias malha 2D, toro 2D e árvore gorda. O processo de geração de tráfego, descrito pelo autor como uma metodologia de caracterização de tráfego foi proposto a partir de um conjunto de oito aplicações reais, descritas originalmente para *Multiprocessor System*

*on Chip* (MPSoCs). A proposta é dividida em múltiplas partes, que descrevem as características que um tráfego deve assumir. Dentre elas estão o modelo da aplicação e o modelo da arquitetura, ambos descritos através de grafos. O modelo da aplicação descreve o tráfego, no que tangem os custos de comunicação e de computação de uma aplicação. Já o modelo da arquitetura descreve as características desejadas da rede, incluindo os parâmetros dos recursos e a capacidade dos elementos de processamento.

O processo de avaliação de tráfego utiliza como métrica de desempenho a vazão total da rede e os atrasos de todos os pacotes medidos entre o envio e recebimento dos pacotes do tráfego. O processo experimental varia as topologias suportadas pelo trabalho sob diferentes tamanhos de redes, onde tráfego sintético uniforme é comparado com modelos de aplicações descritas através da abordagem proposta. Em geral, os autores concluem avaliando vazão da rede e latência dos pacotes, que o modelo proposto comparado a tráfego sintético uniforme, por ter a capacidade de concentrar tráfego tanto espacialmente quanto temporalmente trouxe maior precisão aos resultados.

## 2.9 HONG ET AL.

Hong et al. [HON08], propõem avaliar o desempenho de uma rede de topologia *toro* com suporte a roteamento *backtracking*, fazendo uso de tráfego sintético orientado a aplicações.

Os autores utilizam o ambiente OMNeT++ para geração e simulação de redes intrachip. O ambiente utilizado possui licença pública, é desenvolvido em C++ e dá suporte a interface gráfica. A rede utilizada possui como características, topologia *toro* 2D, 32bits de largura de dados entre os roteadores e algoritmo de roteamento *backtracking*. O roteamento segue a ideia do menor caminho entre a origem e o destino do tráfego, possuindo protocolo de chaveamento de pacotes de três fases, sendo elas: (i) Chaveamento do circuito; (ii) Transmissão do pacote; (iii) Liberação do circuito. O chaveamento é feito através de um pacote que contém o destino do tráfego, que pode ou não assumir um caminho alternativo ao menor caminho caso encontre bloqueios durante o chaveamento do circuito. A transmissão dos pacotes ocorre *flit a flit*, já a liberação do circuito é feita através de um pacote que libera o caminho.

O processo de geração de tráfego dá suporte a tráfego sintético dirigido a aplicações. Pacotes são gerados conforme três parâmetros, sendo eles a distribuição espacial, a distribuição do intervalo de geração entre os pacotes e o tamanho dos pacotes. A distribuição espacial, que descreve a relação entre a origem e o destino dos pacotes, é variada de três maneiras, sendo elas: (i) Uniforme; (ii) *Locality*; (iii) *Transpose*. Na distribuição uniforme, a quantidade total de pacotes é distribuída de maneira proporcional entre todos os destinos disponíveis para receber tráfego na rede. Na distribuição *Locality*, os pacotes são distribuídos de maneira uniforme, porém, com uma relação de afinidade entre os vizinhos próximos a origem do tráfego. Na distribuição *Transpose*, o destino do tráfego é a sua matriz transposta do endereço origem do tráfego. A distribuição temporal é feita através de uma distribuição de probabilidade Poisson onde a taxa de injeção dos pacotes é fixa e o tempo de intervalo entre pacotes consecutivos é variado. A distribuição de

tamanho de pacote é parametrizável, podendo ser feita de três formas, com tamanho de pacote curto, médio e longo.

O processo de avaliação de tráfego utiliza como métrica de desempenho o atraso médio de pacotes medido em ciclos de relógio e a vazão média da rede. O processo experimental é feito em uma rede *toro* 2D com dimensões 4x4. No modelo de tráfego proposto, cada IP envia 10.000 pacotes, variados em três tamanhos de pacotes, sendo o curto de 128bytes, o médio de 512bytes e o longo de 2048bytes. Como resultados, os autores concluem que o fator de localidade do tráfego trouxe ganhos no atraso e na vazão dos pacotes, o que indica um possível ganho de desempenho quando uma estratégia de mapeamento é utilizada. Além disso, o autor detectou que a distribuição espacial uniforme foi aquela que atingiu os menores resultados envolvendo saturação do tráfego, quando comparadas a distribuições espaciais *Locality* e *Transpose*.

## 2.10 BRUCH ET AL.

Bruch et al.[BRU09], apresentam um ambiente que permite avaliar redes intrachip projetadas em nível de transferência entre registradores (RTL) como no nível de transação (TL), sobre diferentes configurações, utilizando ferramentas com suporte a interface gráfica.

O ambiente proposto, referenciado por BrownPepper, implementado com uso de interface gráfica, e de domínio público, permite a geração parametrizável da rede SoCIN, acrônimo de *SoC Interconnection Network*, que possui topologia malha 2D, composta pelo roteador ParIS, acrônimo de (*Parameterizable Interconnect Network*), formado por 5 portas, onde 4 delas são responsáveis pela comunicação entre os roteadores e uma delas responsável pela comunicação do roteador com um módulo IP. Cada porta do roteador possui dois canais, um responsável por receber e outro pelo envio dos dados. A rede SoCIN permite a transmissão de pacotes de tamanhos ilimitados.

O processo de geração de tráfego é implementado em um dos módulos do ambiente BrownPepper através de um ambiente parametrizável de configuração de tráfego utilizando interface gráfica. O modelo de tráfego proposto segue o mesmo definido em [TED05], com acréscimo de um novo campo aos parâmetros do tráfego, que o classifica conforme sua qualidade de serviço (do inglês *Quality of Service* ou QoS). Através deste valor é possível avaliar se um requisito do tráfego, como um tempo máximo de atraso, por exemplo, foi atendido.

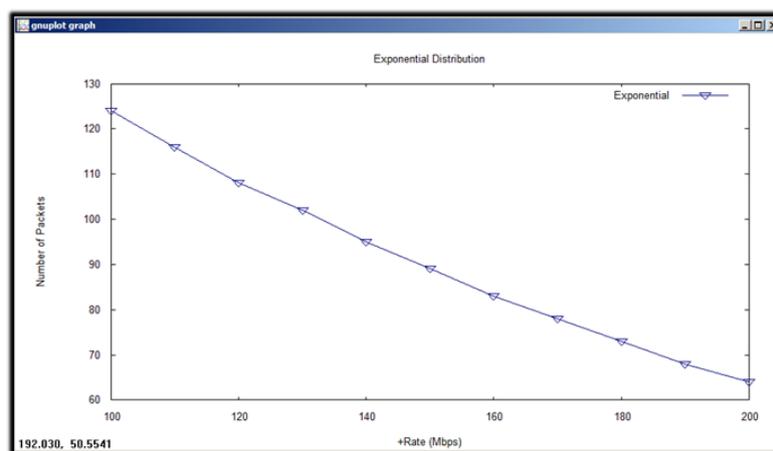
O processo de avaliação de tráfego utiliza como métrica de desempenho a latência média da rede, medida em ciclos de relógio, e o percentual de pacotes que tiveram seu tempo de transmissão ideal atendidos. O ambiente de avaliação é disponibilizado através de uma interface gráfica. Durante o processo experimental o autor avalia dois cenários de tráfego, que demonstram as funcionalidades do ambiente de maneira geral, demonstrando que modificando certos parâmetros da rede, é possível atender restrições impostas pelos tráfegos gerados.

## 2.11 SCHEMMER ET AL.

Schemmer e Calazans [SCH10], apresentam uma proposta para estender o gerador de tráfego do ambiente ATLAS, utilizando a distribuição de probabilidade exponencial decrescente para variar a distribuição temporal de um tráfego de pacotes, que define o intervalo entre a transmissão dos pacotes de um tráfego.

Os autores fazem uso da rede HERMES e da ferramenta de geração de redes presente no ambiente ATLAS, que permite gerar variações da rede HERMES, variando suas dimensões, profundidade das filas nas portas dos roteadores, largura dos canais e número de canais virtuais. Além disso, o ambiente possibilita a seleção de um dentre sete algoritmos de roteamento, sendo 6 deles adaptativos e um determinístico, todos livres de situação de impasse (em inglês *deadlock free*).

O processo de geração de tráfego permite variar os parâmetros deste, no que diz respeito ao número de pacotes, tamanho dos pacotes, destinos do tráfego e intervalos entre a geração dos pacotes. Os destinatários dos pacotes do tráfego podem ser parametrizados de três maneiras: (i) Endereço destino único: Todos os pacotes de um tráfego são enviados para um único destino; (ii) Complemento: Todos os pacotes de um tráfego são enviados para endereço binário negado; (iii) Aleatório: Os destinos são escolhidos de maneira aleatória pelo gerador de tráfego. O intervalo de injeção entre os pacotes segue uma distribuição exponencial decrescente, calculado entre um intervalo de taxas de injeção, passível de parametrização. A Figura 8 descreve através de um exemplo uma das interfaces do ambiente de geração, que permite a visualização gráfica da distribuição dos pacotes para um intervalo informado. No exemplo ilustrado pela figura, é proposto um tráfego de 1000 pacotes variados em um intervalo de 100Mbps a 200Mbps variados em incrementos de 10 Mbps, com valor de média que define a intensidade do decaimento da distribuição definido em 151 Mbps.



**Figura 8: Curva exponencial decrescente de 1000 pacotes gerados em um intervalo de 100Mbps a 200Mbps com incremento de 10Mbps e média de 151Mbps.**

O processo de avaliação do tráfego utiliza a mesma abordagem descrita por [TED05]. Como métricas de avaliação de desempenho, é feito uso de vazão e de latência. Durante o processo experimental os autores comparam a distribuição temporal de tráfego exponencial decrescente

proposta com uma distribuição normal já existente no ambiente. Dois fluxos de 500 pacotes são propostos, um do roteador 00 para o roteador 22, e um do roteador 20 para o roteador 22, ambos concorrentes pelo mesmo caminho da rede. O primeiro tráfego (00-22) é gerado de maneira uniforme, o segundo tráfego (20-22) é gerado em um cenário através de uma distribuição normal, e em outro cenário através de uma distribuição exponencial. O objetivo do experimento é avaliar a capacidade de saturação da distribuição exponencial comparada à distribuição normal, sendo que ambas enviam a mesma quantidade de informação, em intervalos de tempo correlatos para os mesmos destinos.

Como resultados, os autores detectam existir 20% de aumento na latência média do tráfego uniforme (00-22) utilizando tráfego exponencial (20-22), quando comparado a um mesmo cenário, que utiliza de tráfego normal (20-22). A vazão por outro lado diminuiu 9% no tráfego uniforme utilizando tráfego exponencial, quando comparado a tráfego normal. Por fim, o tráfego exponencial apresenta melhores resultados com relação a capacidade de saturar a vazão e a latência dos pacotes, comparado a tráfego normal e uniforme.

## 2.12 GRATZ ET AL.

Gratz et al. [GRA06], apresentam o projeto, implementação e a avaliação da rede intrachip TRIPS OCN, proposta como uma rede intrachip para interconexão de processadores.

Os autores propõem o desenvolvimento da rede TRIPS OCN, que contem como características, tamanho 4x10, topologia malha 2D, chaveamento de pacotes *wormhole*, 4 canais virtuais em cada uma das portas dos roteadores, formados por 5 portas, sendo 4 delas para comunicação entre roteadores e uma delas para comunicação da rede componentes.

O processo de geração de tráfego propõe o uso de tráfego sintético e o uso de aplicações reais. O tráfego sintético é modelado utilizando distribuições *bit complement* e *random* que variam os destinos dos pacotes, já o tempo de injeção dos pacotes é dado por uma distribuição uniforme. Os autores utilizam o SPEC CPU2000 para extrair as aplicações reais, sendo utilizadas 20 aplicações no total.

O processo de avaliação do tráfego utiliza como métrica de desempenho a latência média dos pacotes, medida em ciclos de relógio e a vazão aceita dos tráfegos. Durante o processo experimental, os autores comparam os resultados obtidos entre tráfego sintético e as aplicações reais. Concluem então que o tráfego sintético, além de não reproduzir o mesmo universo da aplicação, no que diz respeito ao momento e a quantidade de informação injetada, não reproduz resultados de saturação próximos da aplicação. Estes, em relação aos cenários avaliados, são justamente os causadores de aumento na latência média dos pacotes, e redução na vazão total da rede.

### 2.13 ROSA ET AL.

Rosa et al. [ROS12], propõem o uso da técnica de DFS, (do inglês *Dynamic Frequency Scaling*) em um MPSoC, sendo que o controle da frequência é aplicado tanto aos elementos de processamento conforme a carga de trabalho quanto a NoC, conforme a carga de comunicação.

O trabalho utiliza um MPSoC homogêneo baseado em uma versão modificada da NoC HERMES. Os elementos de processamento do MPSoC são formados por 4 componentes, sendo eles: (i) Processador de 32bits plasma, que executa um micro sistema operacional que dá suporte a execução de tarefas; (ii) Memória local, que armazena os dados referentes ao sistema operacional e as tarefas; (iii) Controlador DMA, responsável por transferir códigos do adaptador de rede para a memória local; (iv) Adaptador de rede responsável por transmitir e receber dados da porta local de um roteador da NoC. Para que cada roteador e elemento de processamento pudesse operar sobre diferentes frequências de operação, definidas de maneira dinâmica conforme a carga de trabalho de um elemento de processamento, e a carga de comunicação em um roteador, as filas de entrada de cada uma das portas dos roteadores da NoC HERMES foram modificadas. O trabalho utilizou a técnica GALS, através do uso da estratégia de sincronização baseada em dois *flip-flops*, que conforme o trabalho, é livre de falhas. Através desta, as filas de entrada de cada porta local, garantem que um elemento transmita, e outro receba, mesmo que estejam trabalhando em diferentes frequências.

O trabalho explica ainda, que cada elemento de processamento e roteador, fazem uso de uma frequência única, que simplifica a árvore de distribuição de frequências e possui baixo custo de área do circuito. O processo de geração de frequências é feito através de uma técnica de omissão de ciclos da frequência original, que permite definir a frequência do roteador ou elemento de processamento, conforme sua respectiva carga de transmissão ou de processamento. Os autores não citam existir nenhum processo de geração automatizado ou suporte a parametrização da rede utilizada pelo trabalho.

Os autores propõem dois modelos de tráfego, sendo um sintético, em que uma aplicação composta de 6 tarefas é utilizada, e outro real, em que duas aplicações reais, uma chamada VOPD e outra MPEG são utilizadas. No cenário sintético os autores variam o tráfego com base na computação das tarefas da aplicação, em cenários onde todas as tarefas operam sobre uma mesma carga, e em outra que as cargas são variadas conforme as tarefas. Nas aplicações reais os autores propõem diferentes cenários de mapeamento das tarefas, no sentido de avaliar o impacto do mapeamento.

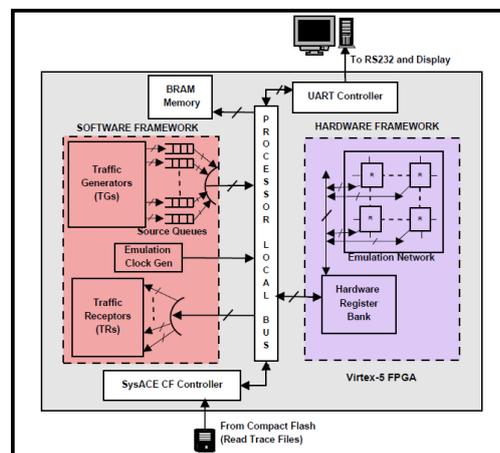
O processo de avaliação do tráfego é feito utilizando métricas de dissipação de potência (em miliwatts), e tempo de execução (em milissegundos). Como resultados, os autores avaliam o impacto da sobrecarga da técnica DFS sobre o tempo de execução e o impacto na dissipação de potência para os diferentes cenários de tráfego propostos. No cenário sintético, conclui-se que sobre cargas fixas o impacto na sobrecarga da técnica de DFS é maior que em cenários de carga variável, uma vez que um menor número de operações é executado pela redução na frequência de operação. Com o uso da técnica de DFS, o cenário sintético demonstrou ganhos na ordem de

20% na dissipação de potência para os elementos de processamento e 72% para a NoC. Já nos cenários reais, os autores concluem que a sobrecarga no tempo de execução é maior utilizando a técnica de DFS do que nos cenários sintéticos, e que a dissipação no consumo de energia demonstrou ganhos parecidos ao cenário sintético, com ganhos na ordem de 25% para os elementos de processamento e 75% para a NoC.

## 2.14 LOTLIKAR ET AL.

Lotlikar et al. [LOT11], propõem uma plataforma de emulação de NoCs em FPGAs, intitulada de AcENoCs, (do inglês *Accelerated Emulation Platform for NoCs*). O objetivo do trabalho é permitir que via emulação, se consiga atingir maior velocidade no processo de emulação, comparada a técnicas tradicionais de simulação.

A plataforma proposta é dividida em duas partes, conforme ilustra a Figura 9. A primeira parte consiste de um ambiente de software, que executa sobre um processador Microblaze, processador este presente no FPGA Virtex-5 (VLX110T). Este processador executa os códigos referentes aos geradores de tráfego, emulador de frequências, filas de entrada dos roteadores e os receptores de tráfego. A segunda parte da plataforma, consiste em um ambiente de hardware, que emula uma NoC e possui um banco de registradores que armazena os valores que a NoC deverá ser configurada durante a emulação. Ambos os ambientes são conectados por um barramento PBL, (do inglês *Processor Local Bus*). Todo o processo de parametrização e configuração do ambiente é feito a partir de um arquivo externo de configuração, que é enviado via *compact flash* e lido por um controlador desenvolvido em nível de software pelo ambiente.



**Figura 9: Plataforma de emulação AcENoCs, formada pelos ambientes software e hardware. O ambiente de software é responsável pela emulação da geração das frequências, pelas filas de entrada dos roteadores e pela geração e recepção do tráfego. O ambiente de hardware, é responsável pela emulação da NoC e pelo banco de registradores, que armazena os parâmetros da NoC a ser emulada.**

A NoC proposta pelo trabalho é formada por roteadores de 5 portas, sendo que cada porta dá suporte a canais virtuais de entrada. Cada roteador é formado por um *crossbar*, uma unidade de roteamento e uma unidade de alocação de canais virtuais. Os autores detalham que a proposta da NoC suporta algoritmo de roteamento XY livre de erros, (inglês *deadlock free*) e que a técnica de escalonamento dos canais virtuais pode ser parametrizada entre *round-robin* ou com prioridades

fixas. O controle de fluxo utilizado pelos canais virtuais faz uso da técnica de *wormhole*. A NoC permite que a frequência dos roteadores seja síncrona ou não síncrona, através do uso da técnica GALS. As filas de entrada dos canais virtuais dos roteadores, são os responsáveis por realizar a sincronização dos transmissores, receptores e dos roteadores quando estiverem trabalhando sobre diferentes frequências de operação.

A proposta do trabalho dá suporte tanto para tráfego sintético como com tráfego real, a partir de traces de aplicações. No tráfego sintético, os autores citam que o ambiente permite variar os destinos dos pacotes utilizando distribuições uniforme, *bit complement*, *bit reversal*, *matrix transpose*, *bit shuffle*, *bit rotation* e *hotspot*. A taxa de injeção dos pacotes é definida via de maneira parametrizável através do arquivo externo de configuração dos parâmetros a serem emulados. O tráfego é gerado utilizando um processador Microblaze, que a partir dos parâmetros definidos irá gerar o tráfego. Para o cenário de tráfego real, o processador Microblaze apenas irá ler, decodificar e enviar cada pacote presente nas linhas do trace da aplicação, informado via arquivo de configuração, presente em uma memória *compact flash*, conforme ilustrado pela Figura 9.

O processo de avaliação do tráfego é feito utilizando as seguintes métricas: (i) Número total de pacotes transmitidos e recebidos; (ii) Número de ciclos gastos pela emulação; (iii) Latência e vazão média dos pacotes transmitidos; (iv) Tempo total de emulação. Nos cenários propostos para avaliar o trabalho, é utilizada uma rede com topologia malha, com dimensões de 5x5, roteamento XY, dois canais virtuais por porta de cada roteador, filas de profundidade de 8 flits e frequências iguais para todos os roteadores, transmissores e receptores. O trabalho utiliza dois tipos de tráfego, sendo o sintético para avaliar o número de ciclos gastos pela emulação, o consumo em área e a velocidade de emulação. O tráfego real é utilizado para avaliar o número de ciclos gastos pela emulação. Na avaliação do tráfego sintético, os autores concluem que o processo de geração dos pacotes é aquele o responsável pelo maior consumo no número de ciclos gastos pela emulação. Na avaliação do consumo de área, o número de LUTs (do inglês *Lookup Table*) disponível no FPGA representa o fator limitante para implementação de redes grandes compostas de muitos roteadores.

Nas questões relacionadas a velocidade de emulação do tráfego sintético, os autores comparam o trabalho a uma proposta de emulação alternativa, chamada *ocin\_tsim*, que para os mesmos cenários avaliados, a proposta AcENoCs, conseguiu atingir um fator de aceleração na ordem de 17X a 47X comparado a proposta *ocin\_tsim*. Além disso, os autores comparam o tempo de emulação da proposta com o mesmo cenário executando em um simulador HDL (do inglês *Hardware Description Language*) onde a proposta ACeNoC conseguiu um fator de aceleração na ordem de 10.000X a 12.000X comparado a proposta simulada. O último cenário avaliado considera um tráfego real, baseado em traces gerados pelo benchmark SPEC CPU2000, sendo que o número de ciclos gastos pela emulação é avaliado. Os resultados são comparados a proposta *ocin\_tsim* configurada com o mesmo cenário de tráfego real, sendo que a proposta AcENoCs apresenta ganhos na redução do número de ciclos gastos na ordem de 9X comparado a proposta de emulação *ocin\_tsim*.

## 2.15 PANDA ET AL.

Panda et al. [PAN11], propõem um modelo de geração de tráfego sintético e de aplicações para arquiteturas do tipo multicore, que utilizem NoCs ou barramentos. No modelo de geração de tráfego sintético, definido como *ConWork*, o tráfego é caracterizado de maneira artificial, a partir de valores parametrizáveis como o número de threads, tamanho dos dados, número de leituras e escritas a memória, dentre outros parâmetros. No modelo de tráfego de aplicações, definido como *CompWork*, é possível parametrizar além dos valores disponíveis para o tráfego sintético, a dependência entre a execução das *threads*. Além disso, a carga de trabalho das *threads* é definida a partir de bibliotecas de multiplicação vetorial ou matricial.

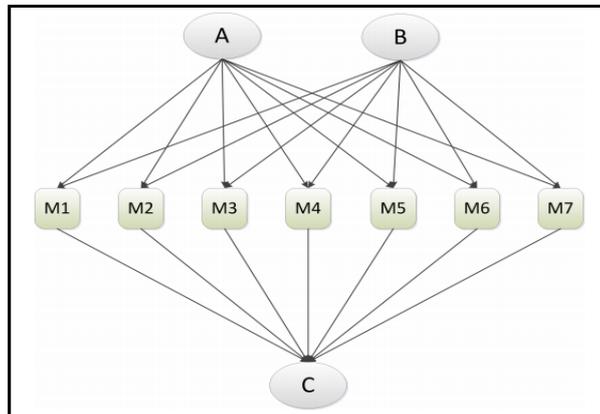
A proposta do trabalho não visa definir um modelo de rede intrachip, porém, os autores fazem uso de dois modelos de NoCs durante a avaliação do trabalho, um híbrido, formado por duas redes de dimensões de 2 linhas por 2 colunas, com topologia malha, interconectadas por um barramento. A outra NoC utilizada nos experimentos, trata-se de uma malha de 4 linhas por 4 colunas. Os autores fazem uso do simulador Simics, que permite a implementação tanto dos processadores quanto dos modelos de NoCs.

No modelo de tráfego sintético *ConWork*, a computação é descrita como um conjunto concorrente de threads em execução, operando sobre conjuntos de dados arbitrários. O modelo assume que um sistema operacional será responsável pelo mapeamento das *threads* sobre os processadores, que também poderá ser feito de maneira manual, caso não exista um mecanismo de mapeamento das threads. Neste modelo, assume-se que os processadores são apenas capazes de executar uma *thread* por vez, sendo que o modelo permite a parametrização do número de threads, quantidade e tamanho dos dados a serem processados, incluindo suporte a parametrização do percentual de leituras e escritas as memórias de cada processador por cada thread.

O modelo *ConWork*, que caracteriza tráfegos sintéticos é avaliado pelos autores do trabalho sobre diferentes cenários de teste, onde são parametrizados valores para os tráfegos, incluindo variação no número de threads e no número e tamanho dos dados de entrada. Como resultados, os autores concluem que a arquitetura de NoC híbrida, foi aquela que apresentou os menores tempos de execução para os cenários de testes sintéticos propostos. Ainda, é feita uma comparação do modelo *ConWork* com aplicações do benchmark PARSEC, no sentido de avaliar como a distribuição das tarefas do benchmark é realizada sobre a proposta de arquitetura multicore implementada. Os autores concluem para este cenário, que o modelo *ConWork*, por permitir maior precisão na descrição e mapeamento das *threads*, algo não existente no PARSEC, é capaz de apresentar redução no tempo de execução das aplicações, dado o efeito do mapeamento realizado.

Além do tráfego sintético, descrito pelo modelo *ConWork*, os autores apresentam o modelo *CompWork*, que permite caracterizar tráfegos de aplicações, onde a carga de computação das *threads* são baseadas em bibliotecas de multiplicação vetorial e matricial. O modelo *CompWork* é baseado na definição dos mesmos parâmetros definidos pelo modelo de tráfego sintético

*ConWork*, com a adição de condições de dependências entre as *threads*, definido através de um grafo de dependências. Neste modelo, a execução é feita a partir da ordem de dependências pré-estabelecida no grafo da aplicação. Através desta funcionalidade, os autores informam ser possível definir o grau de paralelismo da aplicação. A Figura 10 ilustra um exemplo de grafo de aplicação, contendo dependências entre as *threads* A, B e C que devem ser respeitadas para a correta execução da aplicação.



**Figura 10: Algoritmo de Strassen descrito como um grafo de dependências.**

Como experimentos, os autores utilizam a mesma arquitetura baseada no simulador Simics utilizado pelo modelo *ConWork*, sendo que é utilizada uma aplicação baseada no algoritmo de Strassen, conforme ilustrado pelo grafo de dependências da aplicação pela Figura 10. Como resultados, os autores avaliam que a arquitetura de NoC híbrida, para o problema baseado no algoritmo de Strassen foi aquela que levou o menor tempo para executar o problema, comparado a arquitetura de NoC malha de 4 linhas por 4 colunas.

## 2.16 CHANG ET AL.

Chang et al. [CHA10], apresentam um ambiente para simulação de NoCs, intitulada de NOCSEP (do inglês *Network-On-Chip-centric System Exploration Platform*). O objetivo dos autores é disponibilizar um ambiente que permita a exploração eficiente de alternativas quanto ao projeto de NoCs, de maneira a facilitar a tomada de decisão em projetos de circuitos que utilizem NoCs como meio de interconexão de componentes.

A plataforma NOCPSEP, descrita de maneira integral em linguagem SystemC, é dividida em três grandes componentes sendo as *threads*, as responsáveis por definir o tráfego, os nós, por definir os processadores que executam as threads, e os adaptadores, que conectam os nós processadores a NoC. A NoC utilizada pela plataforma NOCSEP é baseada em um modelo orientado a objetos OONoC (do inglês *Object-Oriented NoC*), sendo esta, formada por três grandes blocos: (i) Data Link-Layer; (ii) DC Linker; (iii) NoC Information.

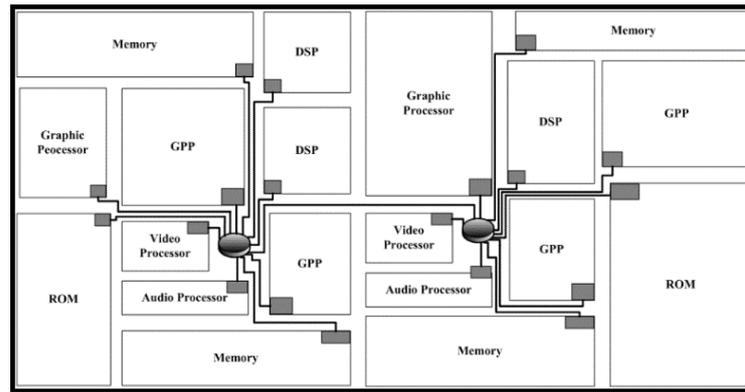
O tráfego é descrito através de grafos de aplicações, capazes de caracterizar comportamentos de aplicações paralelas. O componente thread é o responsável pela caracterização da aplicação, e aquele que realiza a geração do tráfego. Os autores assumem que o tráfego é caracterizado por uma composição de computação e de comunicação, referentes respectivamente ao

processamento do nó e as mensagens transmitidas na rede. Para cada thread, é possível definir como característica de um tráfego de aplicação: (i) Os requisitos de computação e comunicação; (ii) As condições de dependência, agrupamento e mapeamento das tarefas nos nós. Através destes parâmetros, é possível definir diferentes tipos de aplicações paralelas.

O processo de avaliação do tráfego utiliza como métricas de avaliação, a média da vazão e da latência dos pacotes e o tempo total da execução da aplicação. Afim de demonstrar a aplicação da plataforma NOCSEP, os autores sugerem a proposta de três modelos de aplicações, baseadas cada aplicação em um total de oito threads. O primeiro modelo de aplicação é baseado em uma composição de pipeline, o segundo modelo, em um conjunto paralelo de pipelines, e o terceiro, em uma composição de tarefas seguindo um modelo *hot-spot*. Os autores utilizam 4 topologias de NoCs, sendo elas (i) Malha de duas dimensões; (ii) Anel duplo; (iii) Rede binária; (iv) Rede multiestágio. Ainda, são utilizadas quatro estratégias de mapeamento de tarefas, para definir o nó processador na rede onde cada thread irá executar. Os autores avaliam como resultado as diferentes propostas de redes disponibilizadas pela plataforma NOCSEP, demonstrando sua aplicabilidade para detecção da melhor configuração de rede a partir de um aplicação definida.

## 2.17 ZAYTOUN ET AL.

Em Zaytouen et al. [ZAY12], os autores propõem uma nova arquitetura de roteador, que utiliza uma topologia em formato de estrela, como proposta alternativa a redes com topologias em formato de malha de duas dimensões. Conforme comentam os autores, topologias em formato de malha de duas dimensões, poderão no futuro serem proibitivas para interconexão de circuitos do tipo muitos processadores (do inglês *many-cores*), dado o aumento da latência vide o aumento da rede e o número de saltos entre os roteadores. Ainda, são feitas referências a tecnologias do estado da arte na fabricação de semicondutores, nas questões relacionadas a diferença de atrasos entre o chaveamento de um transistor, e a transmissão de um dado em um fio, que tende a aumentar dado o aumento da distância comparada a redução na escala do processo de fabricação. Por fim, na visão do trabalho, redes com topologias não regulares, apresentam não só vantagens quanto as restrições impostas pelo processo de fabricação como por oferecer canais dedicados de transmissão em pontos estratégicos entre roteadores na rede. A Figura 11 ilustra como exemplo, o projeto de um SoC, composto por uma rede intrachip que interliga componentes que demandam diferentes características de interconexão.



**Figura 11: Projeto de um SoC, formado por componentes de diferentes características e demandas por conexões.**

A rede proposta é formada por um roteador, composto de três componentes, sendo eles: (i) Elemento de chaveamento; (ii) Unidade de entrada; (iii) Unidade de arbitragem. Para o elemento de chaveamento, duas propostas foram desenvolvidas, uma delas, onde todas as portas de entrada, são mapeadas diretamente em portas de saída. Esta abordagem utiliza portas lógicas do tipo AND para entrada, e OR para saída, com objetivo de interconectar todas as portas entre si. A segunda abordagem para o elemento de chaveamento, faz uso da técnica Batcher-Banyan onde as portas de entrada são divididas em estágios, conectadas através de pequenos multiplexadores as portas de saída. Para as unidades de entrada, duas propostas foram desenvolvidas. A primeira delas, faz uso de uma fila do tipo FIFO para cada porta de entrada. A segunda, utiliza um mecanismo de divisão da fila de cada porta de entrada, de maneira que cada uma das portas de saída do roteador, esteja vinculada a um número de espaços da fila de entrada. O terceiro e último componente, refere-se a unidade de arbitragem, implementada através de um anel de reservas que através da monitoração das portas de entrada, determina quais pacotes com mesmos endereços de destino, presentes nas filas de entrada irão utilizar as filas de saída. Não é feito nenhum comentário sobre uma ferramenta ou ambiente de geração parametrizável.

O processo de geração e avaliação de tráfego é pouco detalhado, os autores citam utilizar distribuição espacial aleatória, e distribuição temporal uniforme, em que é possível parametrizar a taxa de injeção dos pacotes, e a quantidade de pacotes do tráfego. Como avaliação de tráfego, os autores comentam fazer uso da média e do desvio padrão da latência e da vazão, descrevendo de maneira superficial, como é feito o cálculo do tempo de transmissão dos pacotes para obtenção destes valores. Como resultados do trabalho, os autores avaliam a proposta através de 3 tamanhos de redes, variando o número de entradas e saídas em 32, 64 e 128, e usando tamanho de flit igual a 8bits. Como tráfego, foi utilizado 1000 pacotes por entrada, com distribuição espacial randômica e distribuição temporal uniforme, com variação na taxa de injeção dos pacotes. Os autores concluem que a estratégia de filas de entrada usando a técnica de divisão da fila pelo número de saídas consegue manter a vazão máxima dos pacotes quando exposta a tráfegos com até 97% da taxa de injeção. Os autores consideram para este cenário, tamanho de fila infinitos. Da mesma forma, quando utilizado tamanhos finitos de filas de entrada, o aumento da latência para os cenários avaliados, tende a crescer exponencialmente na medida que as taxas de injeção dos pacotes aumentam, que por consequente reduz a vazão máxima da rede.

## 2.18 POSICIONAMENTO DO TRABALHO EM RELAÇÃO AO ESTADO DA ARTE

Esta Seção resume alguns dos assuntos em comum propostos por este trabalho, coexistentes com os trabalhos pesquisados. A seguir, são feitas comparações, justificando através destas a proposta deste trabalho.

A Tabela 1 apresenta cinco aspectos coletados dos trabalhos pesquisados seguindo o tema de projeto e geração parametrizável de redes intrachip. Os itens coletados partem dos seguintes questionamentos: (i) Trabalho descreve uma ferramenta de geração automática de rede intrachip? (ii) Processo de geração de redes é passível de parametrização? (iii) Quais topologias são suportadas pela rede? (iv) Quais algoritmos de roteamento são suportados pela rede? (v) Rede utilizada é síncrona ou não síncrona? (vi) Em que nível de abstração de projeto a rede é descrita?

A partir da coleta dos dados, constata-se que no item ferramentas de geração de redes e suporte à geração parametrizável, entende-se a existência de seis ferramentas, onde somente em uma delas os autores não descrevem existir suporte a parametrização durante a geração da rede. Dentre as topologias utilizadas, com exceção de um, todos os outros fazem uso da topologia malha 2D, além disso, quase todos dão suporte a topologia *toro*. Com relação ao tipo de roteamento suportado, a maior parte faz uso de roteamento determinístico. Com relação ao nível de abstração do projeto da rede, na grande maioria dos trabalhos é feito uso do *Register Transfer Level* (RTL), utilizando linguagens como Verilog e/ou VHDL para descrever a rede. O último item que pesquisa se as redes utilizadas são síncronas ou não síncronas, demonstra que a grande maioria dos trabalhos pesquisados faz uso de redes síncronas, com exceção de dois deles.

No contexto deste trabalho é apresentada uma ferramenta de geração parametrizável de redes com suporte à definição de múltiplos domínios de frequência para os roteadores, descritos em nível de projeto RTL utilizando linguagem VHDL. Conforme o levantamento feito, nenhum dos trabalhos pesquisados apresenta esta proposta. A rede utilizada por este trabalho faz uso da topologia Malha 2D seguindo a tendência utilizada pela maioria dos trabalhos pesquisados. Por fim, a maior parte dos trabalhos faz uso de algoritmos de roteamento determinísticos, este trabalho, além do uso de algoritmos de roteamento determinístico, dá suporte também a seis algoritmos de roteamento adaptativos, suportados somente por alguns dos trabalhos pesquisados.

Tabela 1: Resumo do estado da arte nos temas de projeto e geração de redes intrachip.

Autor/Ano	Objetivo	Ferramenta de Geração de Rede Intrachip	Suporte à Geração Parametrizável	Topologias Suportadas	Roteamento Suportado	Rede Síncrona ou Não Síncrona (GALS)	Nível de Abstração do Projeto da Rede Intrachip
Fen et al.[FEN07]	Propor diretrizes para gerar redes voltadas para requisitos de aplicações	Ferramenta OPNET	Não se aplica	Malha 2D/ Toro 2D/ Árvore gorda	Não se aplica	Não Disponível	Não Disponível
Kreutz et al.[KRE05]	Sugerir técnica para encontrar arquitetura ótima de rede	Não se aplica	Não se aplica	Malha 2D/ Toro 2D/ Árvore gorda	Determinístico XY/Adaptativo	Não Disponível	Não Disponível
Panades et al.[PAN06]	Propor rede DSPIN com suporte a serviços de rede	Não se aplica	Não	Malha 2D	Determinístico XY	Não Síncrona (GALS)	Não Disponível
Ost et al.[OST05]	Propor ferramenta MAIA para geração e avaliação de redes	Ferramenta MAIA	Sim	Malha 2D	Determinístico XY/Adaptativos <i>West-First, North-Last, Negative-First</i>	Síncrona	RTL (VHDL)
Pontes et al.[PON08]	Propor dois roteadores GALS <i>Low Power</i>	Não se aplica	Não	Malha 2D	Determinístico XY	Não Síncrona (GALS)	RTL (VHDL)
Bononi et al.[BON07]	Comparar tráfego real com tráfego sintético	Ferramenta OMNeT++	Sim	Malha 2D/ Anel/ Spidergon/ <i>crossbar</i>	Determinístico	Síncrona	Não Disponível
Tedesco et al.[TED05a] [TED06]	Apresentar um ambiente de geração e avaliação de tráfego	Não se aplica	Não se aplica	Malha 2D	Determinístico XY/Adaptativo <i>West-First</i>	Síncrona	RTL (VHDL)
Liu et al.[LIU07]	Propor abordagem de caracterização de tráfego	Não se aplica	Não se aplica	Malha 2D/ Toro 2D/ Árvore gorda	Não se aplica	Não Disponível	Não Disponível
Hong et al.[HON08]	Avaliar o desempenho de uma rede com roteamento <i>backtracking</i>	Ferramenta OMNeT++	Sim	Toro 2D	Adaptativo <i>backtracking</i>	Não Disponível	Não Disponível
Brunch e al.[BRU09]	Propor um ambiente de avaliação de redes	Ferramenta BrownPepper	Sim	Malha 2D	Determinístico XY	Síncrona	RTL(SystemC) e TL(SystemC)
Schemmer et al.[SCH10]	Propor uso de tráfego sintético	Ferramenta Atlas	Sim	Malha 2D/ Toro 2D	Determinístico XY	Síncrona	RTL (VHDL)
Gratz et al.[GRA06]	Apresentar o projeto de uma rede intrachip para MPSoCs	Não se aplica	Não se aplica	Malha 2D	Não se aplica	Síncrona	RTL (VHDL)
Rosa et al. [ROS12]	Aplicar técnica de DFS em MPSoCs	Não se aplica	Não se aplica	Malha 2D	Determinístico XY	Não Síncrona (GALS)	RTL (VHDL)
Lotlikar et al. [LOT11]	Propor plataforma de emulação de NoCs em FPGAs	Não se aplica	Sim	Malha 2D	Determinístico XY	Não Síncrona (GALS)	RTL (VHDL)
Panda et al. [PAN11]	Introduzir um modelo de tráfego sintético e de aplicações multicore	Não se aplica	Não se aplica	Malha 2D	Não se aplica	Não Disponível	Não Disponível
Chang et al. [CHA10]	Propor um ambiente para simulação de NoCs	Ambiente NOCSEP	Sim	Malha 2D/Ring	Não se aplica	Não Disponível	SystemC
Zaytoun et al. [ZAY12]	Introduzir uma nova arquitetura de roteador, que utiliza uma topologia estrela	Não se aplica	Não se aplica	Estrela/Árvore gorda	Não se aplica	Não Disponível	RTL (VHDL)
Este trabalho	Introduzir um ambiente de geração parametrizável de redes GALS baseadas na rede HERMES-G	Ferramenta Atlas com suporte à geração de redes HERMES-G	Sim	Malha 2D	Determinístico XY/Adaptativos <i>West-First, West First Minimal, West-First Non Minimal, North-Last, North-Last Minimal, North-Last Non Minimal</i>	Síncrona e não síncrona (GALS)	RTL (VHDL)

A Tabela 2 apresenta sete aspectos coletados dos trabalhos pesquisados seguindo os temas de caracterização de tráfego de rede e métricas de avaliação de tráfego utilizadas. Os itens coletados partem dos seguintes questionamentos: (i) Trabalho descreve ferramenta de geração de tráfego? (ii) Processo de geração de tráfego é passível de parametrização? (iii) Trabalho utiliza grafo de aplicação para descrever tráfego? (iv) Se grafo for utilizado, trabalho descreve alguma técnica de particionamento e/ou mapeamento? (v) Qual é a distribuição espacial do tráfego utilizada pelo trabalho? (vi) Qual é a distribuição temporal do tráfego utilizada pelo trabalho? (vii) Quais são as métricas de avaliação de tráfego utilizadas?

A partir da coleta dos dados, constata-se que no item ferramenta de geração de tráfego com suporte a parametrização, somente alguns trabalhos detalham existir uma ferramenta apta a possibilitar geração de tráfego parametrizável. Em alguns trabalhos, foi detectado o uso de grafos de aplicações para descrever tráfego, sendo que somente em um deles os autores descrevem a existência de uma técnica de particionamento e mapeamento utilizada para sintetizar o grafo que descreve o tráfego na arquitetura da rede. Com relação a distribuição temporal, grande parte dos trabalhos utiliza tráfego aleatório e complemento, existindo em alguns deles, uma proposta evolutiva ao modelo aleatório, em que propriedades estatísticas são utilizadas na definição dos destinos do tráfego. Já no item distribuição temporal, a grande maioria dos autores utilizam distribuição uniforme, existindo alguns casos em que é proposto o uso de modelos estatísticos como uma proposta evolutiva para aumentar a precisão do tráfego. Por fim, o item avaliação de tráfego, resume as métricas utilizadas para avaliar o tráfego na rede, sendo que na grande maioria, os trabalhos utilizam a vazão da rede e a latência média dos pacotes, avaliadas sobre diferentes óticas durante a injeção, transmissão e recebimento dos pacotes.

No contexto deste trabalho, é apresentada uma ferramenta de geração de tráfego parametrizável, que possibilita variar o tráfego, com relação a quantidade e ao tamanho dos pacotes, variando a questão espacial que define os destinos dos pacotes e a questão temporal que define o momento de injeção dos pacotes, utilizando distribuições estatísticas. Além disso, este trabalho dá suporte à geração de tráfego através de grafos de aplicações, que além de permitirem caracterizar a questão espacial e temporal dos pacotes, permitem modelar as dependências existentes entre os tráfegos dos pacotes, que de certa forma caracteriza um maior grau de aproximação ao comportamento de uma aplicação. Conforme o levantamento feito, nenhum dos trabalhos pesquisados descreve um ambiente de geração de tráfego parametrizável, capaz de modelar as características do tráfego conforme é proposto. Por fim, este trabalho apresenta uma ferramenta de avaliação de tráfego, com suporte à interface gráfica, que faz uso das métricas de vazão e latência conforme os demais trabalhos pesquisados. Esta ferramenta leva em consideração o cálculo da latência e da vazão ideal, mesmo quando uma rede intrachip não síncrona é utilizada, uma vez que este trabalho originalmente dá suporte à geração de redes não síncronas. As particularidades referentes a estes detalhes, e a proposta integral deste trabalho será descrita e detalhada em profundidade nos próximos capítulos.

**Tabela 2: Resumo do estado da arte em caracterização, geração e avaliação de tráfego para redes intrachip.**

Autor/Ano	Objetivo	Ferramenta de Geração de tráfego	Suporte à geração de tráfego Parametrizável	Utiliza grafos para descrever tráfego	Descreve técnica de particionamento/ Mapeamento	Distribuição espacial	Distribuição temporal	Avaliação de tráfego
Fen et al.[FEN07]	Apresentar diretrizes para gerar redes voltadas para requisitos de aplicações	Não se aplica	Não se aplica	Não se aplica	Não se aplica	Aleatório/ <i>Locality</i>	Uniforme	Vazão média da rede/Latência de pacote
Kreutz et al.[KRE05]	Técnica para encontrar arquitetura ótima de rede	Sim	Sim	Sim	<i>Sim Tabu Search</i>	Não se aplica	Não se aplica	Latência média dos pacotes
Panades et al.[PAN06]	Apresentar rede DSPIN com suporte a serviços de rede	Não se aplica	Não se aplica	Não se aplica	Não se aplica	Aleatório	Uniforme	Latência total do tráfego
Ost et al.[OST05]	Apresentar ambiente MAIA para geração e avaliação de redes	Ferramenta <i>Traffic Generation</i>	Sim	Não se aplica	Não se aplica	Aleatório	Uniforme	Tempo médio de entrega dos pacotes/ Tempo de entrega dos pacotes
Pontes et al.[PON08]	Apresentar dois roteadores <i>GALS Low Power</i>	Não se aplica	Não se aplica	Não se aplica	Não se aplica	Destino fixo	Uniforme	Latência média de pacote
Bononi et al.[BON07]	Comparar tráfego real com tráfego sintético	Ferramenta <i>Scotch</i>	Sim	Sim	Não se aplica	Não se aplica	Não se aplica	Vazão média da rede
Tedesco et al.[TED05a] [TED06]	Apresentar ambiente de geração e avaliação de tráfego	Ferramenta <i>Traffic Generation</i>	Sim	Não se aplica	Não se aplica	Destino fixo/ Complemento/ Aleatório	Uniforme/ Normal/ Pareto on-off	Latência média de pacote/Vazão média da rede
Liu et al.[LIU07]	Apresentar abordagem de caracterização de tráfego	Não se aplica	Não se aplica	Sim	Não se aplica	Não se aplica	Não se aplica	Vazão da rede/ Latência dos pacotes
Hong et al.[HON08]	Avaliar o desempenho de uma rede com roteamento <i>backtracking</i>	Não se aplica	Não se aplica	Não se aplica	Não se aplica	Uniforme/ <i>Locality/ Transpose</i>	Poisson	Latência média dos pacotes/ Vazão da rede
Brunch et al.[BRU09]	Apresentar um ambiente de avaliação de redes	Sim	Sim	Não se aplica	Não se aplica	Destino fixo/ Complemento/ Aleatório	Uniforme/ Normal/ Pareto on-off	Latência média da rede
Schemmer et al.[SCH10]	Apresentar proposta de tráfego sintético	Ferramenta <i>Traffic Generation</i>	Sim	Não se aplica	Não se aplica	Destino Fixo/ Complemento/ Aleatório	Exponencial	Vazão da rede/ Latência dos pacotes
Gratz et al.[GRA06]	Apresentar o projeto de uma rede intrachip para MPSoCs	Não se aplica	Não se aplica	Não se aplica	Não se aplica	<i>bit complement/ Aleatório</i>	Uniforme	Latência média dos pacotes/ Vazão aceita da rede
Rosa et al.[ROS12]	Propor uso da técnica de DFS em MPSoCs	Não se aplica	Não se aplica	Não se aplica	Não se aplica	Destino fixo	Uniforme	Tempo de entrega dos pacotes
Lotlikar et al.[LOT11]	Propor um a plataforma de emulação de NoCs em FPGAs	Não se aplica	Sim	Não se aplica	Não se aplica	<i>Uniforme, bit complement, bit reversal, matrix transpose, bit shuffle, bit rotation e hotspot</i>	Uniforme	Latência e vazão média dos pacotes transmitidos
Panda et al.[PAN11]	Introduzir um modelo de tráfego sintético e de aplicações multicore	Não se aplica	Sim	Sim	Não se aplica	Destino fixo	Uniforme	Tempo de entrega dos pacotes
Chang et al.[CHA10]	Propor ambiente para simulação de NoCs	Ambiente NOCSEP	Sim	Sim	Não se aplica	Destino fixo	Uniforme	Média da vazão e da latência e tempo de entrada dos pacotes
Zaytoun et al.[ZAY12]	Introduzir uma nova arquitetura de roteador, que utiliza uma topologia estrela	Não se aplica	Não se aplica	Não se aplica	Não se aplica	Aleatório	Uniforme	Média e do desvio padrão da latência e da vazão
Este trabalho	Introduzir um ambiente de geração parametrizável de redes GALS baseadas na rede HERMES-G	Ferramenta Atlas com geração e avaliação de tráfego	Sim	Sim	<i>Sim Tabu Search</i>	Destino Fixo/ Complemento/ Aleatório	Uniforme/ Normal/ Exponencial	Latência média dos pacotes/ Vazão média dos pacotes

### 3 GERAÇÃO DE REDES INTRACHIP NÃO SÍNCRONAS

Este Capítulo descreve o processo de parametrização e projeto da ferramenta de geração de redes GALS, baseadas na arquitetura original da rede HERMES-G. Ele descreve quais foram as dificuldades e as decisões tomadas durante o desenvolvimento do gerador. Este gerador foi completamente integrado ao fluxo de projeto de geração do ambiente ATLAS. Na visão do autor, este Capítulo é considerado com uma das contribuições apresentadas por este trabalho.

#### 3.1 O AMBIENTE ATLAS E A REDE HERMES-G

Esta Seção descreve através de uma abordagem cronológica, quais foram as primeiras atividades desenvolvidas, para obter e entender a ferramenta ATLAS e a rede HERMES-G, para então dar início a parametrização da rede e desenvolvimento da ferramenta de geração. A seguir, serão detalhadas as principais características do processo de geração de redes no ambiente ATLAS, que influenciaram no projeto do gerador de redes não síncronas. Logo após é detalhado como a rede HERMES-G foi obtida, e algumas das principais características e funcionalidades adicionadas durante o desenvolvimento deste trabalho à rede.

##### 3.1.1 O AMBIENTE ATLAS

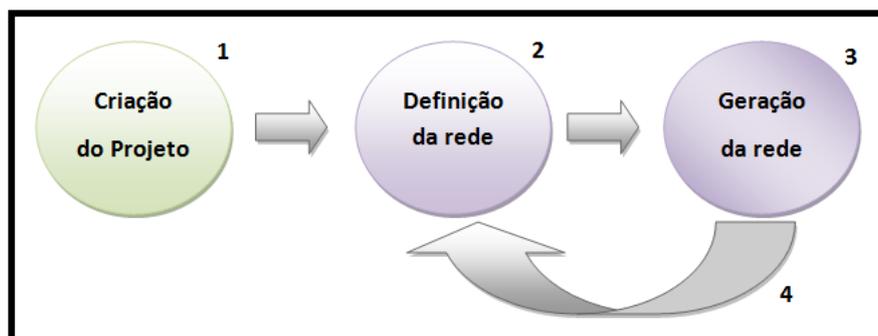
O ambiente ATLAS foi proposto pelo grupo de pesquisa GAPH, ou grupo de apoio ao projeto de hardware, inicialmente com objetivo de integrar as ferramentas de geração e avaliação de tráfego e de energia para redes do tipo HERMES. Hoje, este ambiente dá suporte à geração e avaliação de tráfego para outros seis tipos de redes, originalmente concebidas como uma evolução da rede HERMES, ou levando em consideração algumas das características utilizadas por esta rede. Atualmente o ambiente ATLAS é mantido sobre controle de versionamento via uso do software SVN. Todas as contribuições propostas por este trabalho estão mantidas em um dos inúmeros subdiretórios (em inglês *branch*), derivados a partir do diretório tronco (em inglês *trunk*) do ambiente ATLAS. Para este trabalho foi criado um ramo derivado do tronco, chamado GALS. Todas as contribuições desenvolvidas por este trabalho estão disponíveis publicamente em <https://corfu.pucrs.br/svn/atlas/branches/gals>, incluindo o gerador de redes não síncronas apresentado por este capítulo.

O ambiente ATLAS dá suporte à geração de sete tipos de redes, sendo elas: (i) HERMES; (ii) HERMES-TB; (iii) HERMES-TU; (iv) HERMES-SR; (v) HERMES-CRC; (vi) Mercury. Como resultados do estudo do ambiente de geração, o autor deste trabalho concluiu que:

- ❖ Cada uma das redes suportadas pelo processo de geração de redes do ambiente ATLAS possui seu próprio ambiente de geração, incluindo desde a interface gráfica do gerador até os arquivos que geram a rede intrachip.

- ❖ Todas as redes seguem um padrão, no que diz respeito aos nomes dos diretórios que armazenam o código fonte dos geradores e aos diretórios que armazenam os modelos das redes, utilizados durante o processo de geração de rede.
- ❖ Todas as redes propostas seguem um fluxo de projeto ilustrado pela Figura 12 durante a definição de um projeto e geração da rede, composto por quatro etapas: (1) Criação de um projeto de rede; (2) Definição das características da rede; (3) Geração da rede. (4) Dar suporte à edição de um projeto de rede gerada.
- ❖ Toda rede é vista no ambiente como um modelo universal, durante o processo de geração da rede, os modelos são lidos, manipulados e gerados conforme as características selecionadas pelo usuário.

Partindo destes conhecimentos, o novo ambiente de geração proposto, irá seguir o mesmo modelo de diretórios e nomenclaturas utilizadas para armazenar os arquivos que compõem o ambiente de geração e os modelos da rede. Além disso, o ambiente de geração de redes HERMES-G também irá dar suporte à edição de um projeto de rede.



**Figura 12: Fluxo de projeto de geração de uma rede no ambiente ATLAS, composto por quatro passos (1) Criação de um projeto. (2) Definição das características desejadas para uma rede. (3) Geração da rede. (4) Edição de um projeto de rede gerado.**

### 3.1.2 A REDE HERMES-G

De acordo com [PON08], a rede HERMES-G é uma extensão da rede HERMES, proposta originalmente em [MOR04]. Ambas as redes, são formadas pelas seguintes características, topologia de interconexão malha bidimensional, controle de fluxo utilizando créditos, algoritmo de roteamento determinístico XY e chaveamento de pacotes utilizando a técnica *wormhole*. A principal diferença entre uma rede HERMES de uma rede HERMES-G, está nas filas de entrada dos roteadores. Na rede HERMES, é feito uso de uma fila síncrona, onde a frequência de escrita e a frequência de leitura são feitas sempre, na mesma frequência e na mesma fase. Na rede HERMES-G, é feito uso de uma fila bi síncrona, que possibilita que leituras e escritas, sejam realizadas, tanto na mesma fase e na mesma frequência, como em fases e frequências diferentes.

Da mesma forma como em outros projetos de rede, em uma rede HERMES-G, o principal elemento que compõe a rede é o roteador. Uma rede é composta por diversos roteadores. Cada roteador é composto por três componentes fundamentais. O primeiro deles, chamado de controlador de chaveamento, do inglês *switch control*, é o responsável por realizar a arbitragem

das portas de entrada do roteador, e pelo cálculo do endereço destino de um pacote. O segundo componente, também presente em todos os outros roteadores, do inglês *crossbar*, é o responsável por interconectar todas as portas de entrada com todas as portas de saída do roteador. O terceiro e último componente, presente também em todos os roteadores, chamado de fila de entrada, do inglês *buffer*, é responsável por dar suporte a duas funcionalidades em redes HERMES-G. A primeira delas é garantir que independente a fase, ou a frequência em que o roteador estiver operando, em relação ao roteador ou elemento de processamento ao qual a porta do roteador estiver conectada, a fila presente nesta porta irá garantir que a comunicação ocorra da mesma forma. A segunda funcionalidade é realizar o armazenamento dos *flits* dos pacotes de maneira temporária, quando houver contenção na rede, causado pela ocupação do caminho por um fluxo de pacotes concorrente durante o chaveamento da conexão.

Este trabalho obteve acesso a um projeto de rede HERMES-G, com características definidas, tamanho 3x3, largura de canais igual a 16bits e profundidade das filas de entrada dos roteadores igual a 4*flits* de 16bits cada. Para este projeto foi feito um estudo dirigido, com objetivo de identificar quais eram as estruturas que formavam a rede. Durante este estudo os arquivos da rede foram documentados. Além disso, executou-se um conjunto de experimentos variando algumas das características da rede, com objetivo de validar seu funcionamento.

Antes de descrever com maior grau de detalhamento os componentes que caracterizam uma rede HERMES-G, serão apresentadas algumas atualizações feitas na rede durante o desenvolvimento deste trabalho, umas no sentido de corrigir falhas existentes, outras adicionando novas características a rede. Ao todo, três novas funcionalidades foram adicionadas, e duas correções foram feitas, com objetivo de aumentar a robustez das filas utilizadas nos roteadores. Todas estas modificações feitas na rede são suportadas pelo ambiente de geração de redes HERMES-G.

### 3.1.2.1 ADIÇÃO DE NOVOS ALGORITMOS DE ROTEAMENTO

A implementação original da rede HERMES-G dá suporte a roteamento determinístico XY, em que qualquer pacote segue por um caminho predeterminado a partir da fonte e do destino do mesmo. Ao longo do desenvolvimento da rede HERMES agregaram-se, entre outros, seis novos algoritmos de roteamento adaptativos, sendo estes: (i) *West First*; (ii) *West First Minimal*; (iii) *West First Non Minimal*; (iv) *North Last*; (v) *North Last Minimal*; (vi) *North Last Non Minimal*. Devido às características de semelhança existentes entre as redes HERMES e HERMES-G, os algoritmos de roteamento HERMES são em geral totalmente compatíveis com a HERMES-G. Este trabalho usa roteamento adaptativo de redes HERMES na rede HERMES-G.

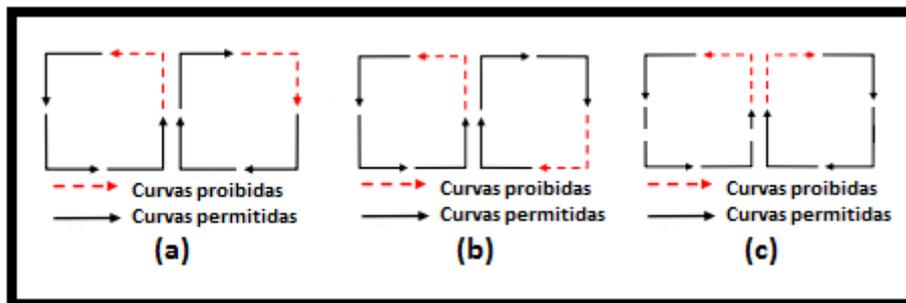
Para garantir o funcionamento destes algoritmos adaptativos, o autor desenvolveu uma ferramenta de testes capaz de gerar, simular e verificar um conjunto arbitrário de redes HERMES e HERMES-G, de maneira automática. Esta ferramenta faz uso dos geradores de rede e de tráfego do ambiente ATLAS, sendo detalhada na seção 4.2.5, tendo sido desenvolvida para ajudar na verificação do funcionamento do gerador de redes HERMES-G.

Um conjunto composto por diversos casos de teste foi proposto para validar os algoritmos adaptativos, nas redes HERMES e HERMES-G, utilizando a dita ferramenta de testes. Durante os experimentos, foram detectados alguns erros em alguns cenários com relação ao roteamento e à transmissão dos tráfegos. Algumas mudanças nos algoritmos adaptativos foram propostas e desenvolvidas por outro membro do grupo de pesquisa do autor. Os mesmos cenários de teste foram novamente executados sob as mudanças propostas, quando então a ferramenta de testes relatou êxito na transmissão dos cenários de tráfegos propostos em todos os cenários testados.

### 3.1.2.2 MINIMIZAÇÃO DO *CROSSBAR*

Outra funcionalidade adicionada à rede HERMES-G, desenvolvida originalmente para redes HERMES, é a minimização do número de portas de saída do *crossbar* para cada um dos seis algoritmos descritos na Seção anterior. Dependendo o algoritmo de roteamento adaptativo utilizado, algumas portas de entrada nunca irão enviar pacotes para algumas das portas de saída do roteador. A Figura 13 mostra para cada um dos algoritmos adaptativos as curvas permitidas e proibidas para pacotes. Estas foram utilizadas para desenvolver as versões minimizadas de *crossbars* para cada um dos algoritmos de roteamento conforme descrito pela Seção 3.1.2.1.

Um conjunto de diversos casos de teste foi proposto para validar as minimizações de *crossbars*, nas redes HERMES e HERMES-G, fazendo uso da ferramenta de testes. Para todos os casos propostos, a ferramenta de testes relatou êxito na transmissão dos tráfegos utilizando diferentes algoritmos de roteamento com suas respectivas versões de *crossbars* minimizados.

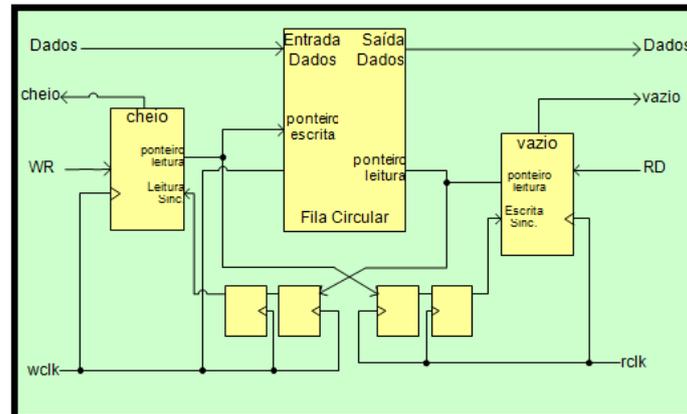


**Figura 13: Curvas proibidas e permitidas para os algoritmos adaptativos: (a) Negative First proíbe curvas para Oeste se o pacote está indo para o Norte e curvas para o Sul se o pacote estiver indo para Leste. (b) West First proíbe curvas para Oeste. (c) North Last proíbe qualquer curva após o sentido Norte ter sido tomado.**

### 3.1.2.3 MODIFICAÇÃO DA CODIFICAÇÃO DE PONTEIROS DO *BUFFER* DOS ROTEADORES

Durante o desenvolvimento do trabalho de mestrado de Heck et al. [HEC11], paralelo ao presente, detectou-se existirem deficiências funcionais no módulo de comparação de ponteiros da fila bi síncrona, componente este utilizado em certas portas de entrada dos roteadores na rede HERMES-G. A Figura 14 ilustra através de um diagrama de blocos os componentes que formam a fila. Dentre eles estão os módulos que implementam os comparadores de ponteiros de fila cheia e fila vazia, utilizados respectivamente como controle de fluxo para escrita na fila, através do sinal de fila cheia e para consumo da fila, através do sinal de fila vazia. Além deles, a fila é composta por

um módulo de memória, que armazena os dados na fila, e por uma lógica de comparadores, que sincroniza a leitura e a escrita, ambas podendo ou não ser feitas em diferentes domínios de frequência e de fase. Através desta funcionalidade é possível configurar os roteadores para operar em diferentes domínios de relógio. Para resolver as condições de exceção encontradas na sincronização de ponteiros, em [HEC11] o autor descreve a proposta de um módulo comparador utilizando três ponteiros, sendo um de leitura, um de escrita e um de escrita anterior.



**Figura 14: Diagrama de blocos da fila bi síncrona utilizada pela rede HERMES-G.**

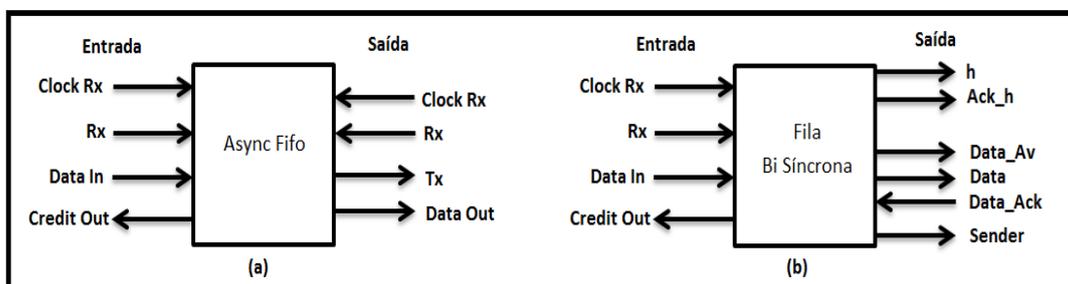
Além disso, no mesmo trabalho detecta-se também que a codificação Gray utilizada originalmente no comparador de ponteiros pela fila bi síncrona [PON08], faz uso de endereços binários, convertido através da operação de ou exclusivo (XOR). Esta conversão possibilita a geração de *glitches*, efeitos indesejáveis que podem levar a fila a um estado inválido. Como solução para este problema, propõe-se o desenvolvimento de um novo comparador de ponteiros utilizando codificação Johnson. Esta faz uso de apenas um registrador de deslocamento e uma porta inversora para realizar o processo de incremento dos ponteiros de leitura e de escrita, eliminando a possibilidade de gerar *glitches* durante o incremento. Em [HEC11], o autor ainda desenvolve um estudo do crescimento da área das filas bi síncronas. Para tanto, compara a codificação de ponteiros Johnson com a codificação Gray originalmente existente na fila bi síncrona. Durante os experimentos varia-se a profundidade das filas em 4, 8, 16, 32, 64 e 128 endereços possíveis. Os resultados são obtidos através do processo de síntese para FPGAs, extraindo o número de LUTs de quatro entradas. Os resultados obtidos apontam que para casos em que 4, 8 e 16 endereços da fila, o número de LUTs gastas por ambas as codificações são aproximadamente iguais. Para profundidades de endereços de filas de 32, 64 e 128 endereços, a codificação Gray, apresenta menor consumo em área que a codificação Johnson.

Por existir ganhos em cada uma das abordagens, o autor deste trabalho adotou a possibilidade de parametrizar a codificação de ponteiros durante a geração de redes HERMES-G, permitindo que o usuário parametrize a geração de arquiteturas de rede que utilizem codificação Johnson e/ou codificação Gray.

### 3.1.2.4 MODIFICAÇÃO DA ARQUITETURA DO COMPONENTE ASYNC\_FIFO

O projeto da rede HERMES-G, por permitir que os roteadores operem em frequências diferentes, cria condições em que os elementos que transmitem e recebem o tráfego da rede necessitem de um componente capaz de sincronizar a transmissão. Com relação ao componente externo transmissor, conectado a uma porta local de um roteador, independente de este operar em uma frequência maior ou menor do que o roteador, a sincronização entre ambos os componentes é garantida pela fila bi síncrona presente na porta local do roteador. Já em um cenário em que o componente externo receptor estiver conectado à porta local, tanto operando em uma frequência de leitura maior ou menor que a frequência do roteador, se faz necessário existir um componente sincronizador entre a saída da porta local do roteador, e o componente de recebimento do tráfego. Na visão de um projeto, este sincronizador é visto como um componente necessário para garantir a recepção do tráfego. Porém ele não faz parte dos módulos que compõem a rede HERMES-G.

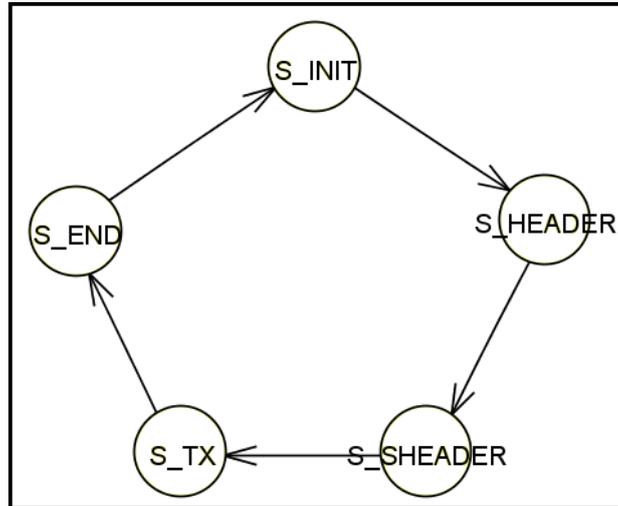
A proposta original descrita em [PON08] propõe a utilização de um componente denominado *Async\_Fifo*, desenvolvido a partir da arquitetura da fila bi síncrona, mas com diferenças no protocolo de leitura da fila. Esta arquitetura compartilha das mesmas deficiências relacionadas à sincronização dos ponteiros detectadas e corrigidas em [HEC11]. Sendo assim, visando corrigir as deficiências existentes, propôs-se o desenvolvimento de uma nova arquitetura *Async\_Fifo*, baseada na arquitetura da fila bi síncrona, que dá suporte às correções propostas em [HEC11], desenvolvida unicamente utilizando a codificação de ponteiros Johnson. A Figura 15 ilustra as conexões de entrada e de saída da fila bi síncrona (na Figura 15(b)), e do componente *Async\_Fifo* (na Figura 15(a)). Em ambos os casos, o protocolo de entrada das filas é o mesmo, pelo motivo de estarem sempre conectadas às portas dos roteadores. Na Figura 15 é possível visualizar que existem diferenças nas interfaces, e estas se refletem em diferenças no protocolo de saída das filas. Na fila bi síncrona, parte de suas saídas são conectadas ao módulo que realiza o roteamento, e parte delas ao *crossbar*. Já na *Async\_Fifo*, a saída da fila possui apenas conexões de transmissão, pelo motivo de não haver necessidade de realizar o roteamento para transmitir os dados.



**Figura 15: Conexões de entrada e saída das arquiteturas (a) *Async\_Fifo* e (b) Fila bi síncrona.**

A Figura 16 descreve a máquina de estados de leitura da fila bi síncrona, composta ao todo por cinco estados, sendo o estado *S\_INIT* responsável por inicializar a fila, os estados *S\_HEADER* e *S\_SHEADER* por realizar o roteamento, e *S\_TX* e *S\_END* pela transmissão do pacote. A nova arquitetura *Async\_Fifo* conta com apenas dois destes cinco estados, sendo eles o *S\_INIT* que

realiza a transmissão do dado, quando existir algum e S\_END, que aguarda e detecta até que exista um dado na fila para ser transmitido.



**Figura 16: Máquina de estados de leitura da fila bi síncrona, composta por cinco estados, sendo um deles responsável pela inicialização, dois responsáveis pelo roteamento do pacote e dois pela transmissão dos *flits* do pacote.**

### 3.1.2.5 EVOLUÇÃO DA ARQUITETURA DA FILA HERMES\_BUFFER

Dentre os diversos tópicos propostos originalmente por este trabalho, um deles propõe utilizar filas síncronas e filas bi síncronas de maneira simultânea em uma rede, uma vez que ambas compartilham do mesmo protocolo de comunicação, e teoricamente, são compatíveis. A ideia por trás do uso de ambas as filas está relacionada ao aumento no tempo de transmissão e na área da fila bi síncrona comparada à fila síncrona. A seleção no uso das filas é feita durante a geração da rede a partir da frequência de operação de cada roteador. Em contextos onde as frequências são diferentes é utilizada a fila bi síncrona, enquanto que em contextos onde a frequência é igual, é utilizada a fila síncrona.

Durante o processo experimental, em que cenários foram gerados de maneira manual, o autor detectou situações de impasse, onde tráfegos de pacotes não eram completamente transmitidos. Em uma inspeção detalhada nas simulações utilizando formas de onda, detectou-se que a arquitetura interna da fila síncrona não era capaz de tratar algumas condições existentes em diferentes cenários de tráfego. As alterações necessárias de serem feitas modificam algumas das propriedades da fila síncrona do roteador HERMES. Sendo assim, propõe-se fazer evoluir a arquitetura síncrona do roteador HERMES, criando uma variação desta. Para classificar a nova variação de arquitetura proposta, assume-se a nomenclatura HERMES-GS, uma vez que a arquitetura é desenvolvida especificamente para trabalhar em redes HERMES-G, possuindo a característica de permitir leituras e escritas unicamente de maneira sincronizada, donde o sufixo GS, acrônimo de *GALS Synchronous*.

A nova proposta soluciona uma deficiência encontrada no processo de leitura de filas síncronas, que assumem depois de feito roteamento do pacote, que a cada ciclo de relógio do roteador, um novo *flit* deve ser transmitido. Esta condição só funciona em dois casos: em um

modelo ideal, onde não há atrasos na transmissão dos *flits* dos pacotes, e quando o chaveamento de pacotes garante que uma vez realizado o chaveamento do primeiro *flit*, os demais *flits* estarão disponíveis para consumo da fila no próximo ciclo da frequência de leitura do roteador. A rede HERMES-G assume, por outro lado, que diferentes frequências podem ser definidas estaticamente a cada roteador, o que na prática pode implicar em diferentes atrasos de transmissão para os fluxos de pacotes. Em outras palavras, a condição de que a cada ciclo de relógio do roteador, um novo *flit* estará disponível na fila e deverá ser transmitido pode não ser atendida, uma vez que o atraso da transmissão pode invalidar esta condição. Para isso, a máquina de estados que dá suporte ao processo de leitura da fila foi reescrita, levando em consideração a condição que o sinal da fila que informa que existe uma transmissão de um *flit* só seja habilitada quando de fato os ponteiros da fila indiquem que um novo dado está presente e disponível na fila para então ser transmitido.

### 3.2 PROCESSO DE PARAMETRIZAÇÃO DA REDE HERMES-G

O projeto de uma rede HERMES-G pode ser decomposto em três grandes partes: (i) os arquivos VHDL que compõem a rede; (ii) os arquivos SystemC que compõem os geradores e receptores do tráfego; (iii) um arquivo VHDL e um TCL, responsáveis por conectar à rede com os geradores e receptores do tráfego e automatizar a compilação e simulação do projeto da rede. No ambiente ATLAS, o processo de geração de redes consiste em gerar de maneira simultânea as três partes. Aqui, por existirem outros fatores que diferenciam o que é rede do que é a estrutura de geração e avaliação de tráfego, preferiu-se optar por desenvolver o gerador de redes com habilidade apenas para gerar a rede em si. A seguir detalha-se como gerar redes HERMES-G via parametrização. Primeiro, descreve-se quais são os componentes que caracterizam a rede, e como eles podem ser parametrizados e adaptados para permitir sua geração. Nos capítulos seguintes, os demais componentes que formam o ambiente de geração de tráfego da rede serão detalhados.

Uma instância de rede HERMES-G é composta ao todo por seis arquivos. Por convenção, renomearam-se os arquivos que compõem a rede, prefixando todos os arquivos com o rótulo HermesG, seguindo a tendência utilizada por outros tipos de redes que o ambiente ATLAS pode gerar. A seguir, detalham-se os arquivos que descrevem a rede, equais foram as decisões de projeto assumidas para permitir a parametrização dos mesmos.

- ❖ HermesG\_Buffer: Arquivo utilizado como modelo para descrever quatro arquiteturas distintas de fila. A parametrização com relação ao tipo e a capacidade da fila têm suporte em outros arquivos. Durante a geração da rede, apenas se copia este arquivo.
- ❖ HermesG\_SwitchControl: Arquivo modelo que descreve seis arquiteturas de arbitragem e roteamento. A parametrização com relação ao algoritmo de roteamento a utilizar tem suporte em outro arquivo. Durante a geração da rede, apenas se faz uma cópia deste.
- ❖ HermesG\_Crossbar: Arquivo modelo, que descreve seis arquiteturas distintas de *crossbar*. A parametrização com relação ao tipo de *crossbar* utilizado novamente é mantida em outro arquivo. Durante a geração da rede, apenas se faz uma cópia deste.

- ❖ Router: Arquivo utilizado como modelo para geração dos roteadores da rede. Este arquivo possui uma série de marcadores, que indicam os locais onde devem ser declaradas as arquiteturas de *SwitchControl*, *crossbar* e *buffer* da rede. Durante a geração da rede, uma cópia deste arquivo é feita, substituindo os marcadores existentes pelas arquiteturas dos componentes *SwitchControl*, *crossbar* e *buffer* selecionados através da interface do ambiente.
- ❖ NOC: Arquivo utilizado como modelo para geração da rede, interliga todos os roteadores. Este arquivo possui uma série de marcadores, que indicam os locais onde devem ser declarados e interconectados os roteadores, como também as portas de entrada e saída do componente. Durante a geração da rede, faz-se uma cópia deste arquivo, substituindo os marcadores pelos roteadores da rede e por suas conexões.
- ❖ HermesG\_Package: Arquivo utilizado como biblioteca de constantes, que armazena, entre outros, os valores do comprimento dos canais e profundidade dos buffers. Durante a geração da rede, uma cópia deste arquivo é feita, substituindo os marcadores existentes pelos valores referentes ao comprimento dos canais e à profundidade dos buffers selecionados através da interface do ambiente.

De maneira resumida, o projeto da interface gráfica do ambiente de geração de redes HERMES-G dá suporte à seleção de oito características da rede, sendo elas:

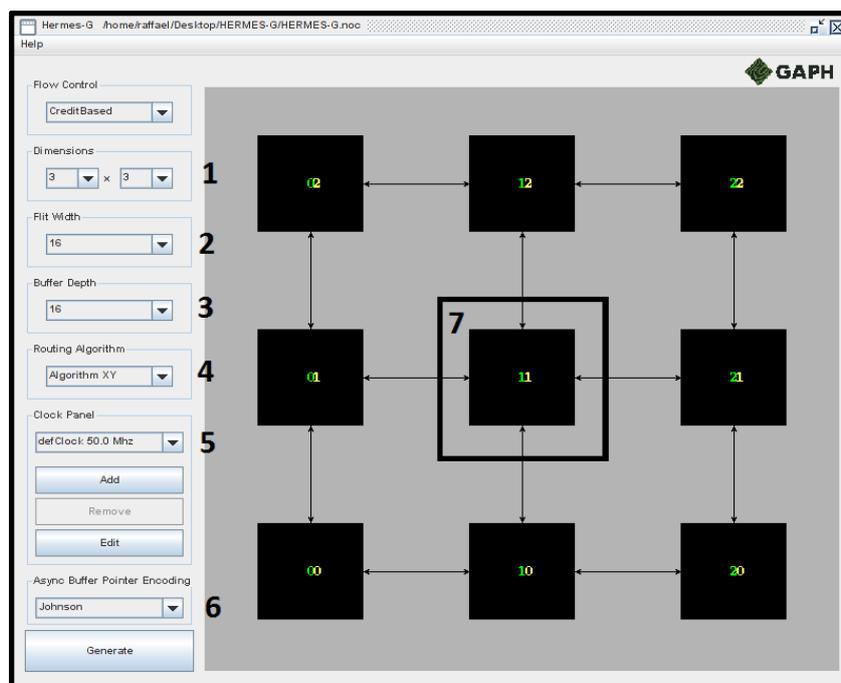
- ❖ Dimensões da rede, em número de linhas e de colunas (assumindo topologia malha 2D).
- ❖ Profundidade das filas de entrada dos roteadores.
- ❖ Largura dos canais que interligam os roteadores.
- ❖ Algoritmo de roteamento utilizado pelos roteadores.
- ❖ Frequência de operação de cada roteador.
- ❖ Frequência de operação de cada gerador e receptor do tráfego.
- ❖ Codificação de ponteiros das filas bi síncronas.

A seguir detalha-se o projeto da interface gráfica do gerador de redes, apresentando uma solução que dá suporte à seleção de valores para todas as características da rede HERMES-G conforme resumidas aqui.

### 3.3 PROJETO DA INTERFACE GRÁFICA DO AMBIENTE DE GERAÇÃO

O projeto da interface gráfica do ambiente de geração de redes HERMES-G segue o mesmo padrão usado em outros tipos de redes no ambiente ATLAS, que estendem a interface gráfica do gerador de redes HERMES. A Figura 17 descreve a interface principal do ambiente de geração, como os campos correspondentes à parametrização das características da rede são numerados de 1 a 7. Os itens 1 a 4 e 6 são detalhados a seguir. Discute-se os demais (5 e 7) na Seção 3.3.1.

- ❖ Dimensões (1): Permite parametrizar as dimensões da rede, suportando redes com tamanho mínimo de dois roteadores em X e dois em Y, e tamanho máximo de dezesseis em X e dezesseis em Y.
- ❖ Comprimento do *flit* (2): Permite parametrizar o comprimento dos canais que interligam os roteadores, em 8, 16, 32 ou 64bits.
- ❖ Profundidade do buffer (3): Permite parametrizar a profundidade das filas utilizadas pelos roteadores, em 4, 8, 16 e 32 *flits*.
- ❖ Algoritmo de roteamento (4): Permite parametrizar o algoritmo de roteamento utilizado pelos roteadores. Os algoritmos de roteamento atualmente disponíveis são: (i) XY; (ii) *West First Minimal*; (iii) *West First Non Minimal*; (iv) *North Last Minimal*; (v) *North Last Non Minimal*; (vi) *Negative First Minimal*; (vii) *Negative First Non Minimal*.
- ❖ Codificação de ponteiros (6): Opção do gerador que permite parametrizar a codificação de ponteiros utilizada pelas filas bi síncronas. Dentre as opções disponíveis estão a codificação Johnson e a codificação Gray.

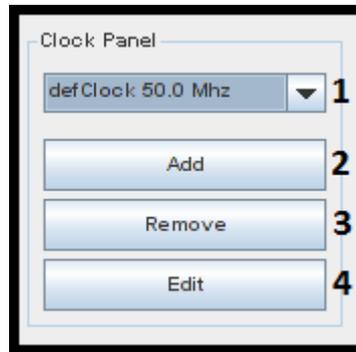


**Figura 17: Interface principal do ambiente de geração de tráfego HERMES-G e suas opções de seleção de características de geração de redes.**

### 3.3.1 GERAÇÃO E DEFINIÇÃO DE RELÓGIOS

O item 5 da Figura 17 dá suporte à criação, edição e remoção de frequências. Estas frequências, depois de cadastradas podem ser usadas (via item 7 da Figura 17) para definir a frequência de operação dos roteadores e seus respectivos módulos de transmissão e recepção de pacotes. Este componente é responsável pela gerência de frequências e possui ao todo quatro funcionalidades, conforme ilustradas na Figura 18.

A funcionalidade 1 permite a visualização dos valores de frequências cadastradas. A funcionalidade 2 permite que novas frequências sejam cadastradas. A funcionalidade 3 permite remover frequências cadastradas, e a funcionalidade 4 permite a edição de frequências cadastradas. Toda frequência é definida por três campos: (i) Nome; (ii) Valor; (iii) Unidade de frequência. Descreve-se a seguir as funcionalidades de adição, remoção e edição de frequências.

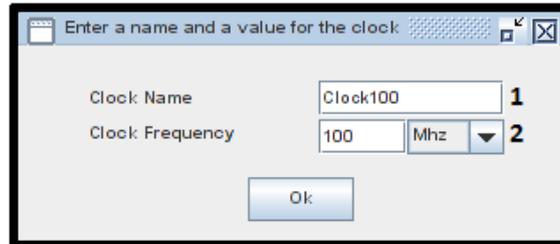


**Figura 18: Componente da interface gráfica utilizável para gerenciar valores de frequência definidos pelo usuário. Estas são utilizadas para definir as frequências de operação de roteadores e de módulos IP a estes conectados.**

#### 3.3.1.1 ADICIONANDO NOVAS FREQUÊNCIAS

A adição de novas frequências permite especificar frequências usadas por roteadores e módulos IP. A Figura 19 ilustra a interface gráfica que dá suporte a esta funcionalidade. A adição de uma nova frequência consiste em definir um nome, um valor e uma unidade de frequência. A funcionalidade de adição de frequências leva em consideração um conjunto de restrições descritas a seguir:

- ❖ Nomes de frequências devem obrigatoriamente iniciar por uma letra do alfabeto, e seus caracteres remanescentes podem ser letras, números ou o caractere ponto. Além disso, palavras reservadas da linguagem VHDL não podem ser usadas como nomes. Estas restrições seguem as restrições impostas para nomes de variáveis e sinais da linguagem VHDL, uma vez que o nome das frequências é utilizado durante a geração do projeto da rede. Duas frequências de nome distinto podem ter exatamente o mesmo valor;
- ❖ Valores para as frequências válidos devem estar entre um intervalo de 0.1MHz (100KHz) e 5,000MHz (5GHz);
- ❖ O ambiente verifica se o nome informado para uma nova frequência já está cadastrado. Não é possível definir duas frequências distintas com o mesmo nome;

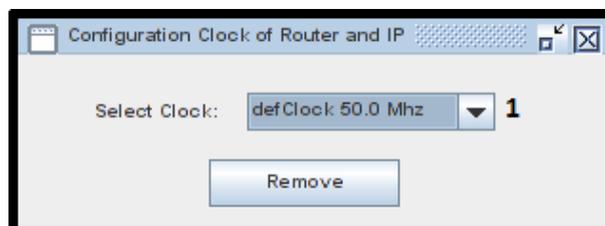


**Figura 19: Interface de adição de uma nova frequência. Campos (1) e (2) possibilitam informar um nome para a frequência e um valor para a frequência.**

### 3.3.1.2 REMOVENDO FREQUÊNCIAS CADASTRADAS

A Figura 20 apresenta a interface de remoção de frequências, que possui um único campo, onde é possível selecionar e remover uma frequência cadastrada. A funcionalidade remoção leva em consideração um conjunto de restrições na remoção de frequências. Para toda rede é gerada uma frequência padrão da rede, que durante a inicialização é definida pela tripla “def Clock/50.0/MHz”. Esta é automaticamente atribuída a todos os roteadores e seus respectivos transmissores e receptores (IPs) no início de um processo de geração de uma instância da rede HERMES-G. Por regra, este valor não pode ser removido, mas pode ser alterado caso desejado.

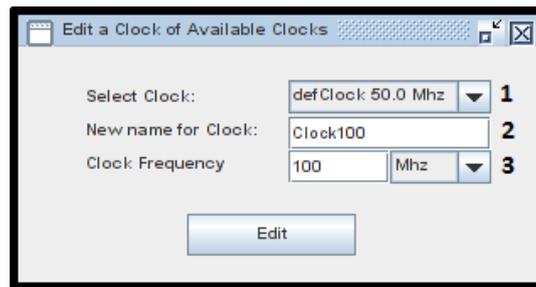
Uma vez removida uma frequência, todos os roteadores e módulos de transmissão e recepção de pacotes que fazem uso desta frequência recebem automaticamente a frequência padrão da rede.



**Figura 20: Interface de remoção de frequências. Esta interface possui apenas um campo, onde se pode selecionar uma frequência a remover.**

### 3.3.1.3 EDITANDO FREQUÊNCIAS CADASTRADAS

A Figura 21 apresenta a interface da funcionalidade de edição, compostas pelos seguintes campos: (1) Selecionar uma das frequências cadastradas; (2) Informar um novo nome para a frequência; (3) Informar um novo valor e selecionar uma unidade de frequência. As regras com relação a nomes e valores de frequências válidas seguem as restrições já definidas na Seção 3.3.1.1. Quando uma frequência é editada, todos os roteadores e módulos de transmissão e recepção que fazem uso desta frequência recebem os novos valores informados para a frequência durante a edição.

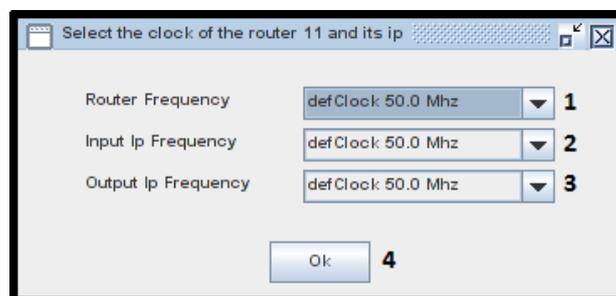


**Figura 21: Interface de edição de frequências. Permite selecionar uma frequência cadastrada e modificar seus campos referentes a nome, valor e unidade de frequência.**

#### 3.3.1.4 SELECIONANDO UMA FREQUÊNCIA CADASTRADA

O item 7 da Figura 17 define a funcionalidade que permite selecionar uma das frequências de operação cadastradas para cada um dos roteadores e seus respectivos módulos de transmissão e recepção de pacotes. Ao se clicar com o *mouse* no roteador 11 da Figura 17, surge a Janela detalhada na Figura 22, que permite selecionar uma das frequências cadastradas, sendo elas detalhadas a seguir:

- ❖ 1 – Frequência do roteador, do inglês *Router Frequency*, responsável por permitir a seleção de qual será a frequência de operação do roteador.
- ❖ 2 – Frequência do transmissor de pacotes, definido no desenvolvimento deste trabalho pelo rótulo na interface gráfica do ambiente por frequência de entrada do Ip, do inglês *Input Ip Frequency*. Este campo é responsável por permitir a seleção da frequência de operação do transmissor de pacotes, vinculado ao respectivo endereço do roteador.
- ❖ 3 – Frequência do receptor de pacotes, definido no desenvolvimento deste trabalho pelo rótulo na interface gráfica do ambiente por frequência de saída do Ip, do inglês *Output Ip Frequency*. Este campo é responsável por permitir a seleção da frequência de operação do receptor de pacotes, vinculado ao respectivo endereço do roteador.



**Figura 22: Interface de seleção de frequências cadastradas: (1) Para o roteador, do inglês *Router Frequency*; (2) Para o módulo de transmissão de pacotes (Definido na interface gráfica do ambiente como *Input Ip Frequency*); (3) Para o módulo de recepção de pacotes (Definido na interface gráfica do ambiente como *Output Ip Frequency*).**

### 3.4 PROCESSO DE GERAÇÃO DA REDE HERMES-G

Detalha-se a geração da rede HERMES-G em duas partes. Primeiro, descrevem-se as questões relacionadas aos diretórios e ao arquivo de projeto da rede, responsáveis respectivamente por armazenar o projeto desta e por permitir a edição da rede gerada. O segundo passo descreve como se dá a geração dos arquivos da rede, detalhando algumas particularidades na escolha das arquiteturas dos componentes.

#### 3.4.1 CRIAÇÃO DOS DIRETÓRIOS E ARQUIVOS DO PROJETO DA REDE

O diretório onde a rede é criada é escolhido no momento da criação do projeto da rede. Neste diretório a pasta “NOC” irá armazenar os arquivos que definem a rede. Um arquivo com extensão “noc” irá armazenar as características da rede gerada, incluindo as frequências definidas na Seção 3.3.1, e as frequências cadastradas para os roteadores e seus respectivos módulos de transmissão e recepção de pacotes, bem como a codificação de ponteiros definida para a rede. A função deste arquivo é também permitir a edição de um projeto de rede previamente gerado. Uma vez gerado o diretório da rede e o arquivo de projeto, os arquivos que compõem a rede são produzidos, conforme descrito a seguir.

#### 3.4.2 GERAÇÃO DOS ROTEADORES E SEUS COMPONENTES INTERNOS

O processo de geração da rede é obtido através da cópia dos arquivos utilizados como modelos parametrizáveis dos componentes da rede, detalhados na Seção 3.2. Os componentes *HermesG\_Buffer*, *HermesG\_Crossbar* e *HermesG\_SwitchControl*, que descrevem respectivamente as arquiteturas das filas, as arquiteturas de *crossbar* e as arquiteturas de roteamento utilizadas para a geração dos roteadores são unicamente copiados do diretório do ambiente ATLAS, que armazena os modelos para o diretório destino do projeto da rede. A seguir são detalhados como os roteadores são gerados.

##### 3.4.2.1 GERAÇÃO DO COMPONENTE HERMESG\_PACKAGE

O componente *HermesG\_Package* é copiado do diretório do ambiente ATLAS que armazena o modelo para o diretório destino do projeto. Durante a cópia do arquivo é feita uma busca e substituição dos marcadores “\$tam\_flit\$” e “\$tam\_buffer\$”, pelos valores referentes ao comprimento dos canais e pela profundidade das filas informada pela interface gráfica durante a definição dos parâmetros da rede. O componente *HermesG\_Package* é utilizado como um arquivo que armazena constantes responsáveis pelos valores dos componentes utilizados.

##### 3.4.2.2 GERAÇÃO DO COMPONENTE ROUTER

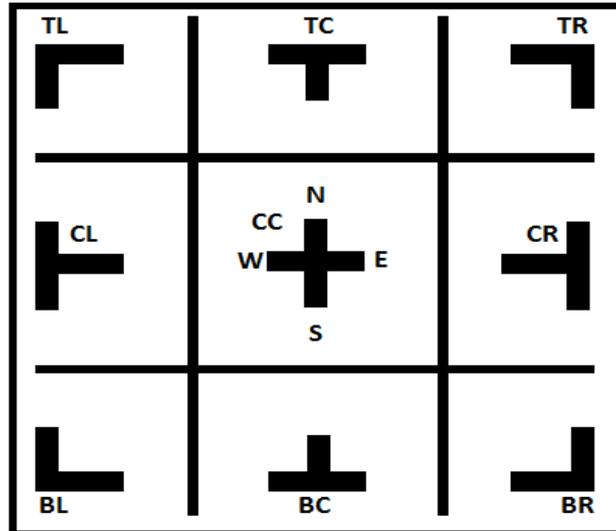
O componente Router contém a descrição de um roteador. Este projeto assume que cada roteador em uma rede com N roteadores será implementado por um arquivo VHDL. Sendo assim, um projeto com N roteadores terá na pasta da rede N cópias do componente Router. Esta estratégia foi adotada por conveniência e facilidade de geração dos roteadores, e devido à

flexibilidade necessária para definir os diferentes domínios de frequência do projeto. Durante a realização de N cópias do arquivo Router, faz-se uma busca e substituição de cinco marcadores, explicados a seguir.

O primeiro marcador \$router\_name\$ indica o local onde o nome do roteador deve ser escrito. A estratégia adotada assume que para cada roteador deva existir um nome único. Para tanto, adotou-se a seguinte nomenclatura como nomes dos roteadores. Todo nome contém como prefixo a palavra Router que indica que o arquivo implementa um roteador, seguido do endereço do roteador em hexadecimal, tendo como sufixo o tipo de fila utilizado em cada uma das portas do roteador. O sufixo parte da seguinte proposta. Todo roteador possui cinco portas, denominadas leste, oeste, norte, sul e local. Nas redes HERMES-G uma porta pode possuir uma fila síncrona (HERMESGS), uma fila bi síncrona (HERMESG), ou estar desconectada. Adotou-se o caractere (0) para porta desconectada, o caractere (A) para fila bi síncrona e o caractere (S) para fila síncrona. O sufixo deverá descrever o tipo de fila, ou se a porta estiver desconectada para cada uma das portas do roteador, seguindo a sequência na ordem leste, oeste, norte, sul, local, segundo convenção de ordem já existente no ambiente ATLAS. Por fim, um exemplo de nome gerado para um roteador seria "Router00S0A0S", onde seu prefixo indica que é o roteador das coordenadas X=0, Y=0, e que possui três portas conectadas, sendo duas delas com filas síncronas, sendo elas a porta leste e local, e possuindo a porta norte conectada a uma fila bi síncrona.

O segundo marcador \$Algorithm\_type\$ indica o local onde o algoritmo de roteamento escolhido deve ser declarado. O terceiro marcador \$Crossbar\_type\$ indica o local onde a arquitetura do *crossbar* deverá ser escrita. A arquitetura do *crossbar* escolhida pela ferramenta de geração de redes é sempre aquela compatível com a arquitetura de roteamento escolhida, uma vez que como explicado na Seção 3.1.2.2, hoje cada arquitetura de *crossbar* é otimizada, ou tem suas portas minimizadas especificamente para cada algoritmo de roteamento.

O quarto marcador \$pin\_ground\$ indica o local onde as portas não utilizadas pelo roteador devem ser desconectadas. Em uma topologia malha 2D, podem existir nove combinações de roteadores com relação ao número de portas dos roteadores. A Figura 23 ilustra as combinações, sendo possível existirem roteadores com 3, 4 e 5 portas. O gerador de redes detecta a partir do endereço do roteador durante a geração quais portas estarão desconectadas.



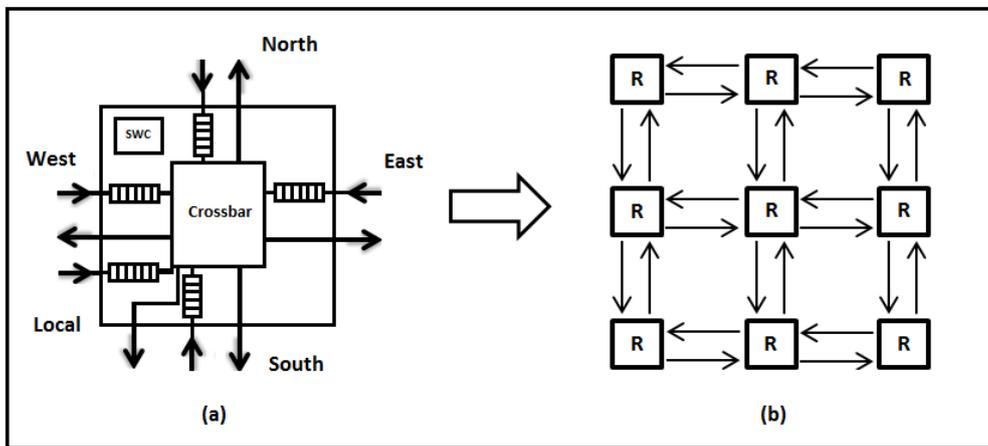
**Figura 23: Combinações existentes com relação ao número de portas utilizadas pelos roteadores em uma rede com topologia malha 2D.**

O quinto e último marcador `$port_type$` indica o local onde as arquiteturas que implementam as filas dos roteadores devem ser declaradas e interconectadas aos demais componentes do roteador. O componente `HermesG_Buffer` dá suporte a quatro arquiteturas de fila, selecionadas pelo gerador a partir das seguintes condições:

- ❖ Quando somente uma frequência é definida para todos os roteadores e módulos de geração e recepção de pacotes, todos os roteadores irão fazer uso da arquitetura da fila do tipo HERMES. Esta rede, além de gastar menos tempo para rotear e transmitir um pacote ocupa menos área que a fila bi síncrona. Como detalhado na Seção 3.1.2, a rede HERMES-G só difere da rede HERMES pelo uso das filas bi síncronas. Dando suporte ao uso de filas do tipo HERMES quando uma única frequência é utilizada este trabalho permite estender as ferramentas de geração e de avaliação de tráfego propostas também para redes HERMES.
- ❖ Quando mais de uma frequência é utilizada, seja por roteadores ou por módulos de transmissão ou recepção, o ambiente de geração faz uso das filas síncronas (HERMESGS) e bi síncronas (HERMESG). Este utiliza a respectiva codificação de ponteiros, com base no que foi definido durante a escolha dos valores da rede. A escolha de qual fila será utilizada em cada porta é feita pela comparação da frequência do roteador com a frequência do roteador ou módulo transmissor/receptor conectado à porta em questão. Se as frequências forem iguais utiliza-se uma fila síncrona (HERMESGS), se as frequências forem diferentes é utiliza-se fila bi síncrona (HERMESG). Esta pode garantir a comunicação, independente se o transmissor conectado à fila é mais lento ou mais rápido que a frequência de operação desta. Note-se que duas frequências idênticas com nomes distintos são tratadas como frequências diferentes, para permitir simular frequências distintas apenas em fase e/ou ciclo de serviço.

### 3.4.2.3 GERAÇÃO DO COMPONENTE NOC

Uma vez gerados os roteadores, o componente NOC, que interconecta todos os roteadores formando a topologia malha da rede é copiado do diretório do ambiente ATLAS que armazena o modelo parametrizável em questão para o diretório destino do projeto. Durante a cópia do arquivo é feita uma busca e substituição de três marcadores. O primeiro, `$clock_noc$` indica o local onde as conexões de entrada e saída do componente NOC devem ser declaradas. Estas conexões permitem que componentes externos transmitam e recebam dados da rede. O segundo e o terceiro marcadores, `$map_router$` e `$port_router$` são substituídos pelas instâncias dos roteadores detalhados na Seção 3.4.2.2 e pela interligação das portas de entrada com as portas de saída dos roteadores, formando a topologia malha da rede. A Figura 24(a) ilustra um roteador formado por cinco portas, gerado através do processo descrito na Seção 3.4.2.2. A Figura 24(b) mostra uma rede composta por nove roteadores através do processo descrito aqui.



**Figura 24:** (a) Descrição de um roteador formado por cinco portas, um *Crossbar* e um *SwitchControl* (SWC). (b) Descrição de uma rede malha formada por nove roteadores.



## 4 GERAÇÃO DE TRÁFEGO PARA REDES INTRACHIP NÃO SÍNCRONAS

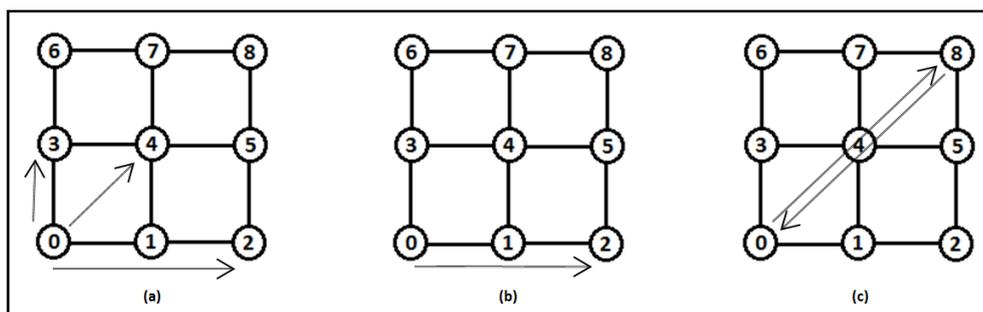
Este Capítulo descreve a geração de tráfego sintético e de tráfego gerado a partir de modelos de aplicações CDCM para redes do tipo HERMES-G. Dentre os tópicos explorados estão a caracterização dos tipos de tráfegos, a descrição da implementação dos geradores de tráfego, e o projeto e geração parametrizável dos componentes responsáveis pela transmissão, recebimento e automação do processo de simulação da rede. Este Capítulo traz mais algumas das contribuições deste trabalho.

### 4.1 CARACTERIZAÇÃO DE TRÁFEGO SINTÉTICO

No escopo desse trabalho, tráfego sintético é caracterizado como aquele que não considera aspectos específicos de aplicações como, por exemplo, a dependência na transmissão de um conjunto de pacotes, ou a variação no comprimento das informações transmitidas. O modelo de tráfego sintético permite que cada módulo transmissor conectado a um roteador, transmita uma quantidade ilimitada de pacotes, onde se varia o tamanho dos pacotes em número de *flits*. Cada transmissor pode enviar diferentes quantidades de pacotes com diferentes tamanhos. A relação entre destinos dos pacotes e intervalos entre a geração de cada um dos pacotes é dada por distribuições espacial e temporal, descritas a seguir.

#### 4.1.1 DISTRIBUIÇÃO ESPACIAL

A Distribuição espacial define a relação entre origens e destinos de um tráfego. Distribuições espaciais suportadas pelo modelo de geração de tráfego sintético utilizado são as propostas por [TED05], e descritas na Figura 25. Ao todo, dá-se suporte a três distribuições:



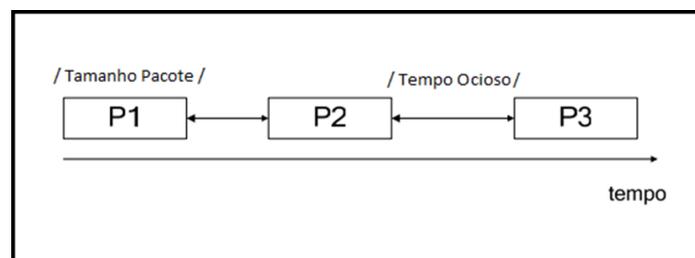
**Figura 25: Distribuições espaciais suportadas pelo modelo de tráfego sintético. (a) Aleatória:** Cada pacote de um tráfego é enviado aleatoriamente entre os demais endereços da rede. No exemplo (a), o endereço zero envia aleatoriamente pacotes para outros endereços. **(b) Destino único:** Todos os pacotes de um tráfego são enviados sempre para um mesmo endereço. No exemplo (b) o endereço zero envia apenas pacotes para o endereço dois. **(c) Complemento:** Todos os pacotes de um tráfego são enviados ao endereço complemento do endereço do transmissor do tráfego. No exemplo (c), os endereços zero e oito enviam pacotes para seu endereço complemento.

- ❖ Aleatória: Na distribuição aleatória, cada pacote de um tráfego terá um endereço destino escolhido de maneira aleatória pelo gerador.

- ❖ Destino único: Na distribuição destino único, todos os pacotes de um tráfego terão um único e mesmo endereço de destino.
- ❖ Complemento: Na distribuição complemento, todos os pacotes de um tráfego terão como destino o roteador cujo endereço é obtido pelo complemento do endereço em representação binária do roteador origem.

#### 4.1.2 DISTRIBUIÇÃO TEMPORAL

A distribuição temporal define a variação das taxas de injeção de um tráfego. Este trabalho faz uso do modelo de variação de taxas de injeção descrito pela Figura 26, onde todos os pacotes do tráfego têm tamanho fixo. Aqui, a variação do intervalo de geração entre os pacotes, conforme descrito na Figura 26 pelo tempo ocioso, é quem define a taxa de injeção de cada pacote.



**Figura 26: Modelo de variação das taxas de injeção dos pacotes. O tamanho dos pacotes é sempre o mesmo, onde o tempo ocioso é variado conforme a taxa de injeção de cada pacote.**

A Equação que calcula o Momento de criação de cada pacote é detalhada na Equação 1. O cálculo é feito a partir do tamanho do pacote a ser transmitido, multiplicado pela razão entre a taxa máxima do transmissor e a taxa de injeção do pacote. A taxa de injeção de um pacote pode ser fixa para todos os pacotes, ou pode ser variada conforme uma distribuição de probabilidade.

$$\text{Momento de criação do pacote} = \text{TamanhoPacote} * \text{NumCiclosFlit} * \left( \frac{\text{TaxaMaxIp}}{\text{TaxaInjPacote}} \right)$$

**Equação 1: Equação que calcula o momento de criação de cada pacote a partir do tamanho do pacote, da capacidade máxima de transmissão e da taxa de injeção do pacote.**

Onde:

*Momento de criação do pacote: Período de transmissão e ociosidade de um pacote*

*TamanhoPacote: Tamanho do pacote (em número de flits)*

*NumCiclosFlit: Número de ciclos gastos para transmissão de um flit*

*TaxaMaxIp: Capacidade máxima de transmissão do gerador de pacotes*

*TaxaInjPacote: Taxa de transmissão de um pacote*

Este trabalho dá suporte a três distribuições de probabilidade: distribuição uniforme [MOR04], distribuição normal [TED05] e distribuição exponencial [SCH10]. Com exceção da distribuição uniforme onde todos os pacotes são transmitidos a uma mesma taxa de injeção, as demais distribuições são detalhadas a seguir.

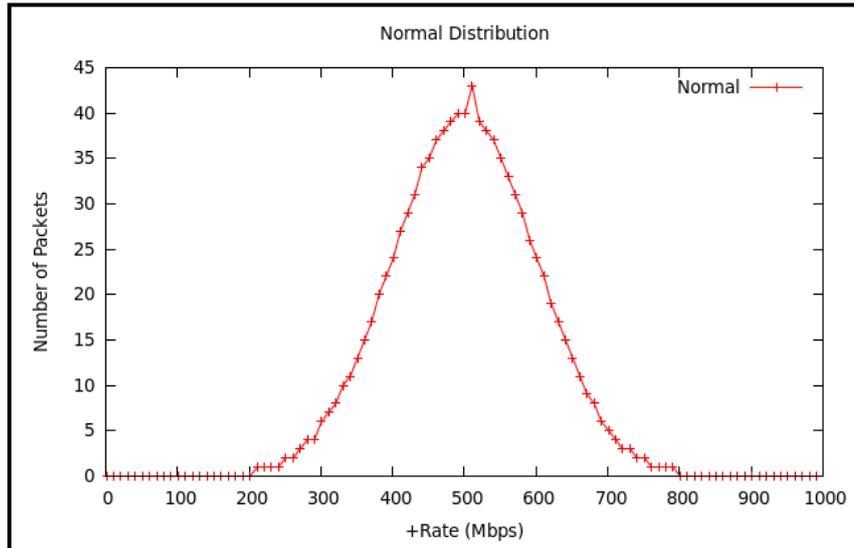
#### 4.1.2.1 DISTRIBUIÇÃO NORMAL

A distribuição normal [TED05] é calculada para um número de pacotes a partir de cinco valores, sendo eles: (i) Taxa de injeção mínima; (ii) Taxa de injeção máxima; (iii) Taxa de injeção média; (iv) Desvio padrão; (v) Incremento. A Equação que calcula a função de probabilidade de cada uma das taxas de injeção de um intervalo é ilustrada na Equação 2. A Equação é formada pelos valores da média da curva ( $\mu$ ), pela taxa de injeção (rate), pelo número de Euler (e) e pelo desvio padrão ( $\sigma$ ).

$$f(Rate) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(Rate-\mu)^2}{2\sigma^2}}$$

**Equação 2: Equação que calcula a função de probabilidade seguindo uma distribuição normal de uma taxa de injeção informada.**

Por conveniência, detalha-se como o cálculo da distribuição normal é realizado através de um estudo de caso. Assume-se um cenário com 10 pacotes com taxa de injeção mínima de 100Mbps, taxa de injeção máxima de 200Mbps, taxa média de injeção de 150Mbps, desvio padrão de 10Mbps e incremento de 10Mbps. A partir do intervalo mínimo de 100Mbps e máximo de 200Mbps e pelo valor de incremento de 10Mbps calculam-se 10 taxas de injeção no intervalo. Para cada uma das taxas de injeção do intervalo, calcula-se o fator de probabilidade através da Equação 2. O somatório de todos os fatores será sempre igual a um, ou seja, o total de pacotes informado para o tráfego. Cada fator indica um percentual do número de pacotes a serem transmitidos. Em uma distribuição normal, a Equação que calcula o fator de probabilidade tende a criar uma curva em formato de sino, onde todas as taxas de injeção entre o intervalo mínimo e máximo de taxas tendem a formar uma relação de simetria em torno da média. Durante o cálculo da distribuição normal, poderão ocorrer casos, como o mostrado na Figura 27 que devido ao arredondamento no número de pacotes durante o cálculo do número de taxas de injeção, ocorrerão sobras de pacotes.



**Figura 27: Exemplo de uma distribuição normal de 1000 pacotes, distribuídos em um intervalo com taxa mínima de 1Mbps e máxima de 1000Mbps, média de 500Mbps, desvio padrão de 100Mbps e incremento de 10Mbps.**

Estes pacotes não calculados deverão, pelas regras definidas pela distribuição normal, ser acrescidas a taxa de injeção referente à média da curva ou próximo desta. A implementação original da distribuição normal descrita em [TED05] apresentou deficiências no cálculo da distribuição normal para algumas variações nos intervalos das taxas de injeção. Em [SCH10], o autor apresentou um conjunto de correções para o cálculo da distribuição normal, eliminando as falhas existentes. A Figura 27 mostra que o cálculo no exemplo utilizado resultou em sobra de pacotes, acrescentados em uma taxa de injeção próxima ao valor da média de 500Mbps.

#### 4.1.2.2 DISTRIBUIÇÃO EXPONENCIAL DECRESCENTE

A distribuição exponencial decrescente é calculada para um número de pacotes a partir de quatro valores, sendo eles: (i) Taxa de injeção mínima; (ii) Taxa de injeção máxima; (iii) Taxa de injeção média; (iv) Incremento. A Equação que calcula a função de probabilidade de cada uma das taxas de injeção de um intervalo é ilustrada pela Equação 3. A Equação é formada pelos valores da média da curva ( $\mu$ ), pela taxa de injeção (rate) e pelo número de Euler (e).

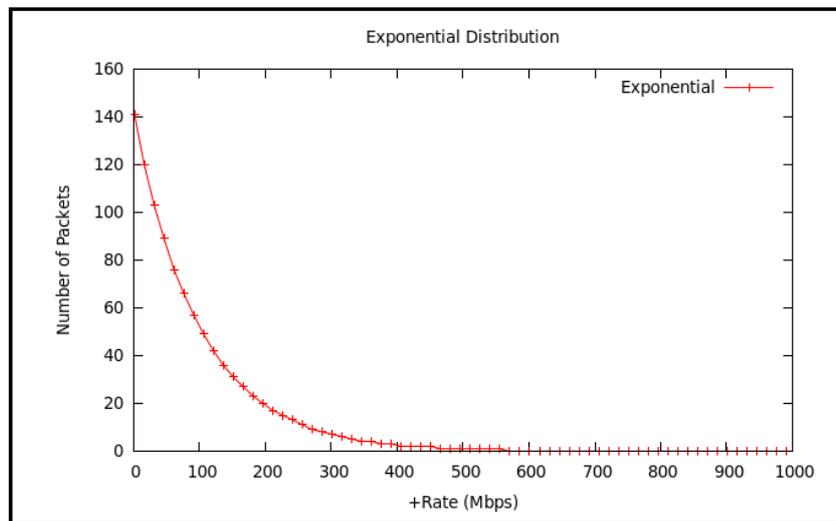
$$f(\text{Rate}) = \frac{1}{(\mu)} e^{-\frac{\text{Rate}}{(\mu)}}$$

**Equação 3: Equação que calcula a função de probabilidade seguindo uma distribuição exponencial de uma taxa de injeção informada.**

Por conveniência, detalha-se como no caso anterior, o cálculo da distribuição exponencial através de um estudo de caso. Assume-se um cenário com 10 pacotes e taxa de injeção mínima de 100Mbps, taxa de injeção máxima de 200Mbps, taxa média de injeção de 150Mbps e incremento de 10Mbps. A partir do intervalo mínimo de 100Mbps e máximo de 200Mbps e pelo valor de incremento de 10Mbps calculam-se 10 taxas de injeção no intervalo. Para cada uma das taxas de injeção do intervalo é calculado o fator de probabilidade através da Equação 3. O somatório de

todos os fatores será sempre igual a um, ou seja, o total de pacotes informado para o tráfego. Cada fator indica um percentual do número de pacotes a serem transmitidos. Em uma distribuição exponencial decrescente, a Equação que calcula o fator de probabilidade cria uma curva decrescente. Conforme aumenta a taxa de injeção, menor a quantidade de pacotes gerados.

O mesmo tratamento com relação ao cálculo do número de taxas e o arredondamento de pacotes detalhado para a distribuição normal se aplica à distribuição exponencial. Seguindo o que diz a regra [SCH10], o excesso de pacotes calculado é colocado na taxa de injeção mínima da curva, que representa o ponto onde o maior número de pacotes é gerado. A Figura 28 apresenta um exemplo de uma distribuição exponencial decrescente de 1000 pacotes distribuídos em uma taxa mínima de 1Mbps e uma taxa máxima de 1000Mbps com média de 100Mbps e incremento de 15Mbps.



**Figura 28: Exemplo de uma distribuição exponencial de 1000 pacotes distribuídos em um intervalo com taxa mínima de 1Mbps e máxima de 1000Mbps, média de 100Mbps e incremento de 10Mbps.**

## 4.2 AMBIENTE DE GERAÇÃO DE TRÁFEGO SINTÉTICO

O ambiente de geração de tráfego sintético para redes HERMES-G diferentemente da abordagem utilizada na Seção 3.1, propõe evoluir o ambiente de geração de tráfego de [TED05]. Modificações foram introduzidas neste ambiente de geração, tanto nos aspectos relacionados à parametrização de um tráfego, como na geração dos arquivos do tráfego. Além disso, o ambiente de geração de tráfego para redes HERMES-G também é responsável pela geração dos arquivos do projeto referentes à transmissão, recepção e simulação da rede, produzidos em conjunto com os arquivos do tráfego. Esta abordagem difere da proposta original, onde os arquivos referentes à transmissão, recepção e simulação da rede são gerados em conjunto com a rede, pela ferramenta de geração de rede. Na visão do autor deste trabalho, os arquivos referentes à transmissão, recepção e simulação, fazem parte do processo de geração de tráfego, razão esta de realizar a geração destes arquivos pela ferramenta de geração de tráfego, e não pela ferramenta de geração de redes, como originalmente feito pelas demais propostas de redes do ambiente ATLAS.

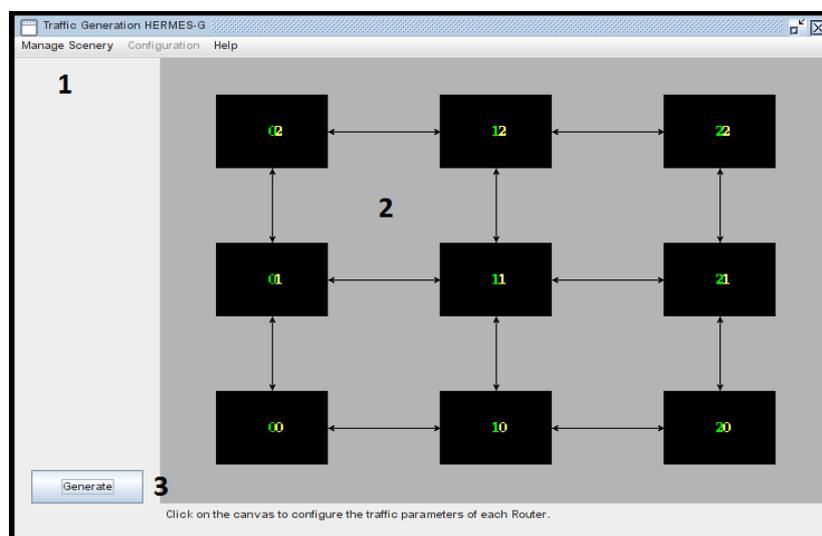
#### 4.2.1 DEFINIÇÃO DO ARQUIVO DE PROJETO DO TRÁFEGO

Antes de detalhar quais são as características da interface gráfica do gerador e o processo de geração de tráfego, detalham-se algumas questões fundamentais do processo de geração de tráfego. O gerador de tráfego permite produzir diversos cenários de tráfego para um mesmo projeto de rede existente. Cada projeto de tráfego possui um identificador ou nome. Este nome é utilizado para geração de um diretório e um arquivo de cenário durante a geração do tráfego, sendo no diretório armazenados os arquivos do tráfego e o arquivo cenário, com os parâmetros definidos do tráfego.

O arquivo de cenário também é usado durante a simulação e avaliação da rede. Neste trabalho, o gerador de tráfego com as modificações feitas para dar suporte a redes HERMES-G irá utilizar a mesma hierarquia de diretórios onde o tráfego é gerado. O mesmo arquivo de cenário possibilita a criação e a edição de mais de um cenário de tráfego para um mesmo projeto de rede.

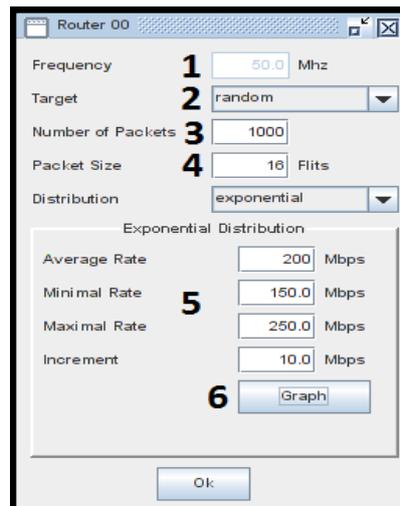
#### 4.2.2 PROJETO DA INTERFACE DO GERADOR DE TRÁFEGO SINTÉTICO

A interface gráfica do gerador de tráfego sintético é composta pela interface principal ilustrada na Figura 29 que apresenta ao todo três funcionalidades: (1) Menu de criação e edição de cenários salvos; (2) Interface de configuração de tráfegos para os módulos geradores; (3) Botão que dispara o processo de criação de um tráfego.



**Figura 29: Interface principal do gerador de tráfego, que possibilita criar múltiplos cenários de tráfego.**

O item (2) da Figura 29 permite, ao clicar sobre um dos quadrados que representam os transmissores de cada roteador com mesmo endereço XY, selecionar o tipo de tráfego para cada um dos geradores. A Figura 30 descreve a interface que possibilita a seleção dos parâmetros do tráfego para cada um dos geradores.



**Figura 30: Interface que possibilita parametrizar o tráfego, tais como número de pacotes, tamanho dos pacotes, distribuição espacial e temporal.**

No gerador de tráfego HERMES-G, a frequência de operação de cada um dos módulos transmissores é conhecida, uma vez que durante a geração da rede, este valor teve de ser informado durante a geração da rede. Sendo assim, na Figura 30 (1) o campo *Frequency* apresenta a frequência do módulo gerador de maneira sombreada, que não pode ser alterada. Na Figura 30 o campo *Target* define a distribuição espacial dos pacotes do tráfego, podendo como já descrito na Seção 4.1.1 ser aleatório, destino único ou complemento. Os campos (3) e (4) da Figura 30 permitem que o número de pacotes e o tamanho do pacote de um gerador sejam variados. O campo (5) da Figura 30 permite configurar as características temporais dos pacotes, através das distribuições normal, uniforme e exponencial descritas na Seção 4.1.2. O campo (6) da Figura 30 permite visualizar graficamente uma distribuição temporal. Durante a adaptação da interface do gerador que possibilita parametrizar o tráfego, a não existência de um tratamento de exceções nos campos de entrada levou o autor a estudar e propor condições mínimas de entrada de valores baseado em alguns critérios, descritos a seguir:

- ❖ Para qualquer uma das três distribuições temporais, a taxa máxima de injeção deve ser sempre a frequência do gerador multiplicada pelo comprimento do *flit*. Por exemplo, em uma rede com comprimento de *flit* igual a 8bits e um gerador operando a 50 MHz, a taxa de injeção máxima é de 400Mbps.
- ❖ Em redes HERMES-G, o tamanho mínimo de pacote suportado para geração de tráfego é de 13 *flits*.
- ❖ O tamanho máximo do pacote em número de *flits* é limitado pela equação  $2^{(\text{Comprimento do flit})}$ . Por exemplo, em uma rede com comprimento de *flit* igual a 8bits, o tamanho máximo em *flits* de um pacote é de 256 *flits*.
- ❖ O número máximo de pacotes de um tráfego é limitado pela equação  $2^{(\text{Comprimento do flit} * 2)}$ . Por exemplo, em uma rede com comprimento de *flit* igual a 8bits, o número máximo de pacotes é de 65535 pacotes.

### 4.2.3 GERAÇÃO E FORMATO INTERMEDIÁRIO DOS PACOTES DO TRÁFEGO

A partir dos valores informados através da interface gráfica é produzido o tráfego. No modelo proposto em [TED05], o tráfego de cada transmissor é representado em arquivos texto, utilizados como entrada por estes módulos conectados nos roteadores. Em cada uma das linhas destes arquivos texto, existem informações sobre um pacote, descritas através de um formato intermediário de pacote a ser transmitido na rede. A Figura 31 descreve o formato intermediário do pacote produzido pelo gerador de tráfego em um arquivo texto que representa o tráfego de um módulo transmissor. A seguir detalha-se cada um dos campos da Figura 31 que compõem um pacote.

TimeStamp (1 flit)	Destino (1 flit)	Tam Payload (1 flit)	Origem (1 flit)	TimeStampDecimal (4 flits)	NúmeroSequência (2 flits)	Payload (.....)
(a)	(b)	(c)	(d)	(e)	(f)	(g)

**Figura 31: Formato intermediário de um pacote de tráfego. Os campos representam: (a) Tempo de transmissão do pacote; (b) Roteador destino do pacote; (c) Tamanho do pacote; (d) Roteador origem do pacote; (e) Tempo de injeção em decimal; (f) Número de sequência do pacote; (g) Carga útil (dados) do pacote.**

#### 4.2.3.1 *TIMESTAMP* DE UM PACOTE

O *TimeStamp* define o momento ideal de injeção de um pacote em ciclos da frequência de operação do módulo transmissor. A partir de uma distribuição temporal, cada um dos pacotes assume uma taxa de injeção. De maneira incremental, o tempo de injeção de cada pacote é calculado a partir da Equação 1. A seguir detalha-se através de um exemplo o cálculo do tempo de injeção de três pacotes de 13 *flits* cada, com taxa de injeção uniforme de 100Mbps, calculado para uma rede com *flits* de 8bits e um módulo transmissor de 100 MHz. Aplicando a Equação 1a cada um dos três pacotes obtêm-se os valores descritos na Tabela 3.

**Tabela 3: *TimeStamp* para três pacotes de 13 *flits* de tamanho injetados com taxa uniforme de 100Mbps por um transmissor operando a 100 MHz em uma rede de comprimento de *flit* igual a 8bits.**

Pacote	<i>TimeStamp</i>
Pacote número 0	1 ciclo
Pacote número 1	105 ciclos
Pacote número 2	209 ciclos

Com base nos valores da Tabela 3 observa-se que o intervalo entre os pacotes é o mesmo de 104 ciclos, pelo motivo de possuírem a mesma taxa de injeção uniforme de 100Mbps. Além disso, o cálculo do *TimeStamp* de cada pacote é dado de maneira incremental, uma vez que cada pacote demanda um tempo para ser transmitido.

#### 4.2.3.2 CAMPOS REMANESCENTES DO PACOTE

Além do *TimeStamp* ilustrado na Figura 31 outros seis campos compõem o pacote do tráfego. A seguir descreve-se cada um destes:

- ❖ Destino: O campo destino informa para qual roteador o pacote deve ser transmitido. Este é utilizado pelos roteadores da rede para calcular a rota que o pacote deve seguir.
- ❖ Tamanho do *Payload*: Por definição, um pacote na rede HERMES-G é formado por dois campos, conforme ilustrado pela Figura 32, sendo eles o *Header* que representa o cabeçalho do pacote e contém o destino e o tamanho do *Payload*. O cabeçalho é responsável por realizar a conexão entre origem e destino. Já o campo *Payload* contém os dados do pacote. A função do campo Tamanho do Payload é enviar ao transmissor o tamanho do pacote, utilizado durante a leitura para montagem do pacote final a ser transmitido na rede.



**Figura 32: Definição de um pacote, formado pelos campos de Header e Payload, onde um representa informação de controle de transmissão e o outro contém os dados do pacote.**

- ❖ Origem: O campo origem informa de qual roteador o pacote está sendo transmitido.
- ❖ *TimeStamp* em decimal: Este campo define o momento em que a transmissão do pacote deve iniciar, expresso em ciclos de relógio desde o início da simulação em notação decimal, utilizado pelo módulo transmissor durante a transmissão do pacote.
- ❖ Número de sequência: O número de sequência é utilizado para garantir que durante a avaliação do tráfego seja possível identificar pacotes a partir de um identificador único. Assim, durante a simulação é possível identificar se pacotes de um fluxo de tráfego foram perdidos ou chegaram ao destino em ordem diversa da ordem de envio de pacotes (particularmente útil quando se emprega algoritmos de roteamento não-determinísticos).
- ❖ *Payload*: O *Payload* contém os dados de uma aplicação em uma transmissão. Em tráfegos sintéticos gerados aqui, este campo é normalmente preenchido com valores aleatórios gerados por convenção a partir do número dez (em decimal).

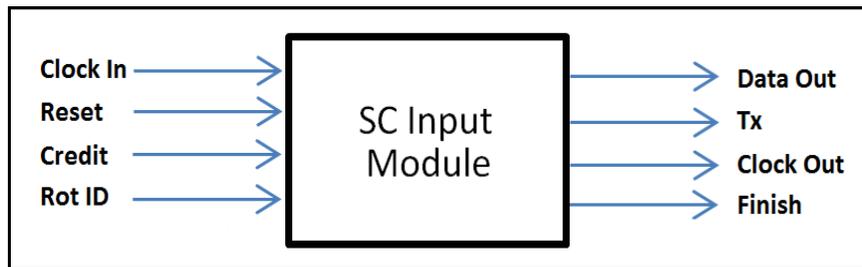
#### 4.2.4 GERAÇÃO DO TESTBENCH PARA TRÁFEGO SINTÉTICO

Além dos arquivos que definem o tráfego sintético, o ambiente de geração de tráfego HERMES-G gera arquivos que definem o ambiente de transmissão e recepção de pacotes, denominado doravante de Testbench. O diretório onde estes arquivos são criados é o mesmo do projeto da rede. Neste diretório o subdiretório SC\_NOC armazena o Testbench, formado por quatro arquivos: (i) SC\_Input\_Module: Responsável pela transmissão dos pacotes; (ii) SC\_Output\_Module: Responsável pela recepção dos pacotes; (iii) HermesG\_Fifo\_Output: Responsável por sincronizar a recepção dos pacotes, quando o receptor opera em uma frequência

diferente do roteador; (iv) TopNoC: Responsável por conectar os transmissores e os receptores a uma entidade de rede. As próximas seções descrevem o processo de geração destes arquivos.

#### 4.2.4.1 O TRANSMISSOR DO TRÁFEGO: SC\_INPUT\_MODULE

O componente responsável pela leitura e transmissão do tráfego é implementado pela entidade SC\_Input\_Module. A versão original do componente, conforme descrito na Seção 3.1.2 foi modificada, tanto com relação às portas de entrada e saída do componente quanto a sua arquitetura, uma vez que um novo formato de pacote de tráfego contendo novos campos foi proposto. A Figura 33 ilustra quais são as portas de entrada e saída da entidade SC\_Input\_Module.



**Figura 33: Descrição da interface da entidade SC\_Input\_Module, que implementa o componente transmissor de pacotes para a rede.**

As portas de saída do componente são:

- ❖ Data Out: Transporta os dados do componente SC\_Input\_Module para o roteador. A largura desta porta é parametrizável, igual à do *flit* da rede.
- ❖ Tx: Indica à porta local do roteador que o transmissor deseja realizar uma comunicação com a rede.
- ❖ Clock Out: Repassa à porta local do roteador a frequência de operação do transmissor.
- ❖ Finish: Indica a finalização da injeção de tráfego na rede, explorado em mais detalhe na Seção 4.2.4.3.

As portas de entrada do componente são:

- ❖ Clock In: Repassa a frequência de operação do transmissor.
- ❖ Reset: É o sinal de inicialização do circuito.
- ❖ Credit: Informa se a porta local do roteador ao qual o transmissor está conectado têm disponibilidade para receber flits de um pacote.
- ❖ RotID: Informa o endereço do roteador, utilizado para leitura dos arquivos do tráfego para o respectivo transmissor.

A arquitetura da entidade SC\_Input\_Module foi basicamente reescrita para comportar as condições de fim de transmissão e para possibilitar a leitura e a transmissão dos novos campos do pacote. Ao todo, o componente opera em três fases, descritas resumidamente a seguir:

- ❖ (1) – Realiza a leitura de um pacote do arquivo de tráfego. Se existir um pacote, avança para o item (2), caso contrário, informa que transmitiu todos os pacotes do tráfego via sinal de Finish a entidade TopNoC descrita na seção 4.2.4.3.
- ❖ (2) – Aguarda enquanto o *TimeStamp* do pacote for menor que o contador de ciclos do relógio do transmissor. Quando o contador for maior ou igual ao *TimeStamp*, avança para o item (3).
- ❖ (3) – Anota o tempo em ciclos do relógio do transmissor, monta o pacote ilustrado na Figura 34 e permanece em uma condição de envio até que todos os *flits* do pacote sejam transmitidos. Quando todos os pacotes forem transmitidos, volta novamente para o item (1).

O pacote transmitido possui sete campos conforme ilustra a Figura 34. Estes incluem o destino (a), utilizado pelos roteadores para encaminhar o pacote e o tamanho do pacote (b) utilizado pelas filas dos roteadores para garantir o chaveamento da conexão a partir do tamanho do pacote. Os campos de origem (c) e número de sequência (e) são utilizados pela ferramenta de avaliação de tráfego para identificação dos pacotes de um tráfego. Os valores referentes ao *TimeStamp* (d) e ao tempo de transmissão (f) são utilizados para os cálculos de vazão e latência dos pacotes, descritos nas seções 5.1.1 e 5.1.2, respectivamente. O item *Payload* (g) completa o pacote de tráfego.

Destino (1 flit) (a)	Tam Payload (1 flit) (b)	Origem (1 flit) (c)	TimeStampDecimal (4 flits) (d)	NúmeroSequência (2 flits) (e)	Tempo Trans (4 flits) (f)	Payload (...) (g)
----------------------------	--------------------------------	---------------------------	--------------------------------------	-------------------------------------	---------------------------------	-------------------------

Figura 34: Formato de um pacote gerado pelo transmissor de pacotes.

#### 4.2.4.2 O RECEPTOR DO TRÁFEGO: SC\_OUTPUT\_MODULE

O componente responsável pelo recebimento dos pacotes da rede e pela escrita dos arquivos texto de saída do tráfego é descrito pela entidade *SC\_Output\_Module*. A versão original do componente, conforme descreve a Seção 3.1.2 foi modificada, tanto com relação as portas de entrada e saída do componente quanto em sua arquitetura, uma vez que um novo formato de pacote gerado pelo transmissor, contendo novos campos, foi proposto. A Figura 35 ilustra quais são as portas de entrada e saída da entidade “*SC\_Output\_Module*”.

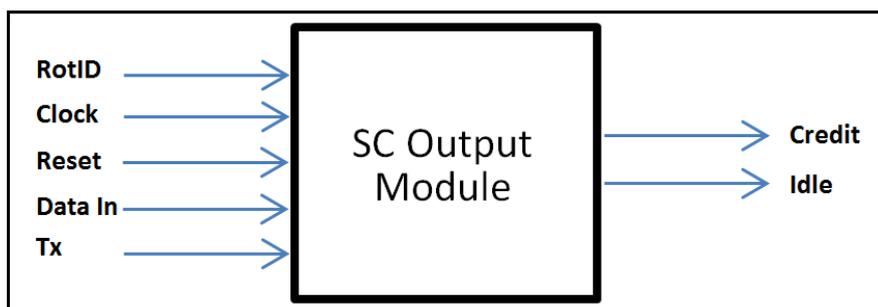


Figura 35: Descrição da interface da entidade *SC\_Output\_Module*, que descreve o componente receptor de pacotes.

As portas de entrada do componente são:

- ❖ RotID: Informa o número do roteador, utilizado para escrita dos arquivos de saída do tráfego, recebidos pelo receptor.
- ❖ Clock: Recebe a frequência de operação do receptor.
- ❖ Reset: É o sinal de inicialização do circuito.
- ❖ Data In: Recebe os dados da porta local dos roteadores para o componente SC\_Output\_Module. A largura desta porta é igual à do *flit* da rede.
- ❖ Tx: Indica ao componente SC\_Output\_Module que a porta local do roteador deseja realizar uma comunicação.

As portas de saída do componente são:

- ❖ Credit: Indica à porta local do roteador se o componente SC\_Output\_Module está apto a receber ou não um *flit* de um pacote.
- ❖ Idle: Indica que o componente receptor não está recebendo flits de nenhum pacote, e que não está aguardando flits de nenhum pacote incompleto. O uso desta porta é detalhado em maior profundidade na Seção 4.2.4.3.

A arquitetura da entidade SC\_Output\_Module foi basicamente reescrita para comportar os novos campos do pacote. Em resumo, ela aguarda a chegada dos pacotes, contendo uma máquina de estados capaz de detectar quais campos está recebendo, escrevendo estes valores no arquivo de saída de tráfego ao qual estiver conectado.

Os pacotes escritos no arquivo de saída são utilizados posteriormente para avaliação de tráfego. O formato do pacote após ser recebido e processado é ilustrado na Figura 36, sendo composto por nove campos. Entre os campos que compõem um pacote estão o destino (a), a origem (c) e o número de sequência (e) utilizados pela ferramenta de avaliação descrita no Capítulo 5 para identificar os tráfegos. Os campos TimeStampDecimal (d), Tempo Transmitido (f), TempoChegadaPF (h) e TempoChegadaUF (i) são utilizados para os cálculos da vazão e da latência dos pacotes descritos nas Seções 5.1.1 e 5.1.2.

Destino (1 flit) (a)	Tam Payload (1 flit) (b)	Origem (1 flit) (c)	TimeStampDecimal (4 flits) (d)	NúmeroSequência (2 flits) (e)	Tempo Transmitido (4 flits) (f)	Payload (...) (g)	TempoChegadaPF (Decimal) (h)	TempoChegadaUF (Decimal) (i)
----------------------------	--------------------------------	---------------------------	--------------------------------------	-------------------------------------	---------------------------------------	-------------------------	------------------------------------	------------------------------------

**Figura 36: Formato do pacote após ser recebido e processado pelo receptor de pacotes utilizado para avaliação do tráfego dos pacotes.**

#### 4.2.4.3 O COMPONENTE TOPNOC

O componente TopNoC é responsável por declarar e conectar transmissores e receptores a uma instância de rede. A seguir descreve-se algumas das estruturas que o componente TopNoC é responsável por gerar e conectar:

- ❖ (1) Realiza a geração das estruturas que descrevem os geradores de frequência para cada uma das frequências definidas para os roteadores e para os módulos de transmissão e recepção de pacotes criados durante a geração da rede.
- ❖ (2) Declara N instâncias de arquivos transmissores e receptores, onde N é o número de roteadores da rede, e uma instância da rede gerada.
- ❖ (3) Atribui a cada transmissor um endereço e conecta suas portas de entrada e saída a cada uma das portas locais dos roteadores da rede com o mesmo endereço. Além disso, a partir da frequência definida para os módulos transmissores durante a geração da rede, atribui a cada transmissor seu respectivo gerador de frequência.
- ❖ (4) A partir dos valores de frequência definidos para os roteadores, conecta os geradores de frequência às entradas de relógio dos roteadores e de suas portas.
- ❖ (5) Verifica em quais casos a frequência do roteador é diferente da frequência do componente receptor do tráfego. Caso haja pelo menos uma ocorrência dessa condição, o gerador do componente TopNoC realiza uma cópia do arquivo modelo `HermesG_Fifo_Output` para o diretório de projeto, no subdiretório NOC. Esta entidade é detalhada na Seção 3.1.2.4, que descreve o componente `Async_Fifo`. O nome `HermesG_Fifo_Output` foi escolhido para padronizar a nomenclatura com os demais arquivos da rede.
- ❖ (6) Atribui a cada receptor um endereço, sendo que se a frequência do receptor for igual a do roteador correspondente ao endereço, conecta diretamente o receptor ao roteador. Se a frequência do receptor for diferente do roteador correspondente ao endereço, conecta o receptor a uma fila `HermesG_Fifo_Output` e então conecta este a porta local do respectivo roteador. Esta fila garante a comunicação nos casos em que as frequências do roteador e do receptor são diferentes.

Além destas condições, o componente TopNoC é dotado de uma estrutura que finaliza a simulação de maneira automática uma vez que não exista mais nenhum tráfego ativo na rede. Este trabalho implementou esta funcionalidade dado que o processo de simulação do ambiente ATLAS necessita que o usuário estime e defina estaticamente um tempo de duração da simulação. Redes do tipo HERMES-G, por permitirem fazer uso de diferentes frequências de operação, tendem a dificultar a predição de um tempo ideal de simulação, capaz de garantir que todos os pacotes de um tráfego sejam transmitidos. Em muitas situações, a predição de um tempo de simulação pode tanto não ser suficiente para que o tráfego seja transmitido, como também ser muito maior do que o necessário, o que leva a um aumento não desejado no tempo de simulação. Pelas limitações e deficiências existentes, a estrutura que finaliza de maneira automática a simulação foi proposta, resolvendo as limitações e facilitando o processo de simulação de redes HERMES-G.

A estrutura que finaliza a simulação de maneira automática faz uso de três valores. O primeiro deles referente aos sinais `Finish` de todos os componentes transmissores, indicam quando um transmissor terminou de transmitir seus tráfegos. O segundo são os sinais `Idle` de todos os

receptores, indicam se o receptor está recebendo tráfego ou se está ocioso. Finalmente usa-se a condição de fila vazia de todas as filas de todos os roteadores da rede, inclusive as filas utilizadas entre os roteadores e os receptores no sentido de identificar se existem tráfegos sendo transmitidos ao longo da rede. Uma vez que todos os transmissores tenham transmitido seus pacotes, e os receptores estejam em estado ocioso, e todas as filas da rede estiverem vazias, a estrutura produzirá um evento externo ao simulador. Este último irá retornar a chamada ao arquivo que automatiza a simulação, e a seguir irá finalizar automaticamente a simulação.

#### 4.2.4.4 ARQUIVO DE AUTOMAÇÃO DA SIMULAÇÃO

O arquivo Simulate.do é gerado para um projeto de rede com os comandos de compilação e simulação da rede, de maneira que este arquivo, uma vez disparado, automatiza o exercício da rede. Este arquivo contém uma condição que após disparar o comando de simulação da rede, aguarda que um evento externo do simulador que informe o fim da simulação, para então finalizar sua execução. A simulação de um tráfego sintético e/ou de uma aplicação é feita a partir da leitura dos arquivos do tráfego e da transmissão dos pacotes na rede. Por fim, os receptores coletam os pacotes e escrevem os arquivos de saída, contendo os dados dos pacotes e os tempos gastos durante a transmissão.

#### 4.2.5 VALIDAÇÃO DA PROPOSTA

Como já descrito em seções anteriores, desenvolveu-se uma ferramenta de testes em caráter experimental, com objetivo de verificar se as características funcionais da rede e do ambiente de Testbench desta estão operacionais, conforme os critérios do desenvolvimento. A ferramenta de testes permite, através da interface de linha de comando, mostrada na Figura 37, que diferentes abordagens sejam utilizadas para definir cenários de teste, estes sendo formados por arquivos de rede e de tráfego.

Para esta ferramenta, propôs-se uma distribuição de tráfego alternativa às demais existentes no gerador de tráfego do ambiente ATLAS, onde todos os elementos transmissores da rede geram tráfegos para todos os endereços da rede, com exceção dele mesmo. A distribuição temporal utilizada é uniforme, que assume a taxa máxima de transmissão do transmissor. O modelo de tráfego varia o número e o tamanho dos pacotes de diversas maneiras, seja a partir de parâmetros informados pelo usuário durante a geração dos cenários ou de maneira aleatória. Considera-se que esta abordagem permite que todas as combinações possíveis de caminhos sejam utilizados pelos pacotes.

```

||||| Gerador e verificador de tráfego sintético para redes HERMES e HERMES-G |||||
-----
Java Gerador : TrafficOp | NumMaxX | NumMaxY | FlitWidth | TamPckt | NoCType | BufDepth | BufType | AlgoType | ClockType
-----
Example : java Gerador 2 2 16 16 0 4 J 1 1
Obs :
TrafficOp : (0) Gen Traffic (1) Ver Traffic (2) Gen NoCs (3) Gen NoCs + Traffic (4) Ver Traffic Ger (5) Gen Auto Cenery
FlitWidth : (8) bits (16) bits (32) bits (64) bits
NoCType : (0) HERMES-G (1) HERMES
BufDepth : (8) flits (16) flits (32) flits
BufType : (J) Johnson Coding (G) Gray Coding
ClockType : (0) DefClock 50 Mhz (1) Random 0Mhz - 5Ghz
AlgoType : (0) XY (1) WFM (2) WFNM (3) NLM (4) NLNM (5) NFM (6) NFNM

***** Warning : This Software dont verifies wrong values !!! *****
|||||

```

**Figura 37: Interface linha de comando da ferramenta de testes responsável pela geração e verificação de cenários.**

Utilizando uma opção do ambiente que permite gerar todas as combinações de redes em um intervalo de parâmetros, obteve-se ao todo 50.400 cenários possíveis de redes do tipo HERMES-G, onde todos os parâmetros que definem as características da rede foram variados. Estes cenários variam opções de dimensões da rede, profundidade de filas, largura de *flit*, algoritmo de roteamento e codificação de ponteiro das filas. Como já descrito, a ferramenta, além de possibilitar que a simulação seja feita de maneira automática, é finalizada quando os tráfegos de um cenário forem transmitidos. Além da simulação, a ferramenta também verifica o tráfego gerado e compara se os pacotes do tráfego foram corretamente recebidos nos destinos. Para os 50.400 cenários de testes simulados, em todos os casos houve sucesso na transmissão e recepção dos pacotes.

Também se adicionaram diretivas na ferramenta para detectar e usar a capacidade do computador hospedeiro para *multithreading*, quando existirem. Durante o desenvolvimento do trabalho, o autor detectou que o simulador RTL utilizado nas simulações, não fazia uso, ou não tinha por comportamento fazer uso da capacidade de execução *multithreading* do computador hospedeiro. Ainda, pela ferramenta de teste possibilitar que um conjunto de testes formados por um ou mais cenários de rede, foi adicionada a ferramenta de teste, a capacidade de realizar o processo de simulação de vários cenários de rede de forma simultânea. O número de simulações simultâneas é definido pelo número de elementos de processamento presentes no computador hospedeiro. Esta técnica foi proposta, uma vez detectado que a maior parte do tempo gasto pela ferramenta de teste envolvia o processo de simulação. Além disso, por detectar que o simulador RTL não era capaz de extrair a capacidade *multithreading* do computador hospedeiro, e também, por existirem cenários de teste capazes de serem processados simultaneamente. Durante o uso da técnica, foram detectados ganhos significativos no tempo de execução dos cenários de teste com o uso desta técnica.

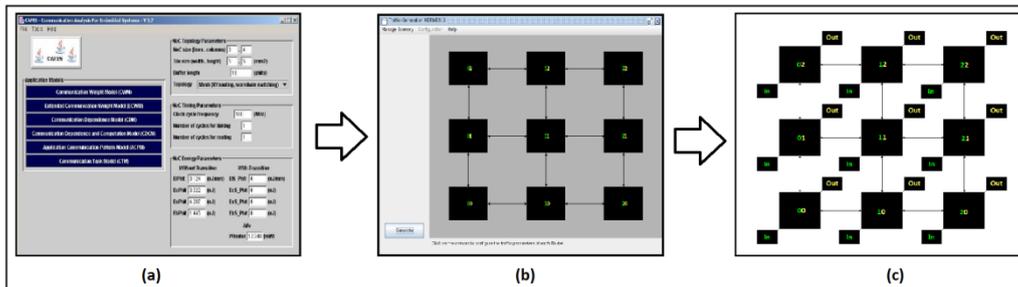
### 4.3 AMBIENTE DE GERAÇÃO DE TRÁFEGO DE MODELO DE APLICAÇÕES

Além da proposta de um modelo de tráfego sintético, este trabalho também propõe fazer uso de modelos de aplicações para descrever tráfego para uma rede HERMES-G. Usa-se o modelo de aplicações CDCM, do inglês *Communication Dependence and Computation Model*, proposto por

Marcon [MAR05], que representa uma aplicação pelas suas dependências de computação e comunicação. De acordo com Marcon [MAR05], o modelo permite capturar o tempo de computação, o tamanho da mensagem e suas dependências. O modelo CDCM é representado através de um grafo CDCG, do inglês *Communication Dependence and Computation Graph*, sendo utilizado para descrever as aplicações.

Marcon propôs o ambiente denominado *Communication Analysis for Embedded Systems* (CAFES), que permite a geração e o mapeamento de modelos CDCM sobre um MPSoC baseado em NoCs. Dentre os modelos de aplicações e grafos de tarefas suportados pelo ambiente CAFES, o modelo que na visão do autor deste trabalho melhor representa as características de uma aplicação, são respectivamente o modelo de aplicação CDCM e o grafo de tarefas CDCG.

O processo de criação e geração de um modelo de aplicação no escopo deste trabalho é conduzido em três fases, ilustradas na Figura 38.



**Figura 38: Uso de modelos de aplicações CDCM no ambiente Atlas: (a) Criação e geração de um grafo de aplicações CDCM; (b) Geração do Testbench pela ferramenta ATLAS; (c) Geração dos arquivos de tráfego e do ambiente de transmissão e recepção de pacotes.**

O ambiente CAFES da Figura 38(a) é usado nas etapas de criação e mapeamento de um modelo de aplicação sobre uma topologia de NoC. Esta etapa gera um arquivo intermediário, que servirá de entrada para a ferramenta de geração de tráfego do ambiente ATLAS, ilustrado na Figura 38(b). A partir das definições contidas neste arquivo gera-se o *Testbench*, contendo os arquivos de tráfego e o ambiente de transmissão e recepção adaptado para as características do modelo de aplicação CDCM informado.

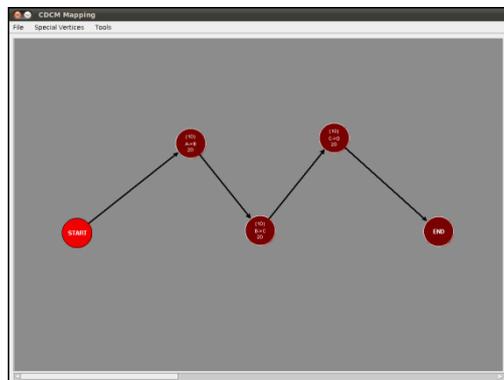
#### 4.3.1 O AMBIENTE CAFES: CRIAÇÃO E MAPEAMENTO DE UM MODELO DE APLICAÇÃO

O ambiente CAFES permite descrever modelos de aplicação CDCM a partir de grafos CDCG. Em um grafo CDCG, ilustrado na Figura 39, vértices correspondem a tarefas. Toda aplicação possui um início, descrito no grafo pelo vértice especial START e um fim, descrito no grafo pelo vértice especial END. Cada tarefa é descrita pelos seguintes campos:

- ❖ Tempo de processamento: Define um tempo de processamento da tarefa expressa em nano segundos.
- ❖ Identificador origem e destino: Define quem são a origem e o destino da comunicação realizada durante a execução da tarefa. Durante o mapeamento estes identificadores são traduzidos em endereços da rede.

- ❖ Mensagem de transmissão: Define um tamanho da mensagem a ser transmitida, logo após o término do tempo de processamento da tarefa. Toda mensagem deve ser expressa em número de *flits*.
- ❖ Dependências de entrada: Definem os eventos que devem acontecer para que a tarefa seja executada.
- ❖ Dependências de saída: Definem que outras tarefas devem ser disparadas após a execução da tarefa.

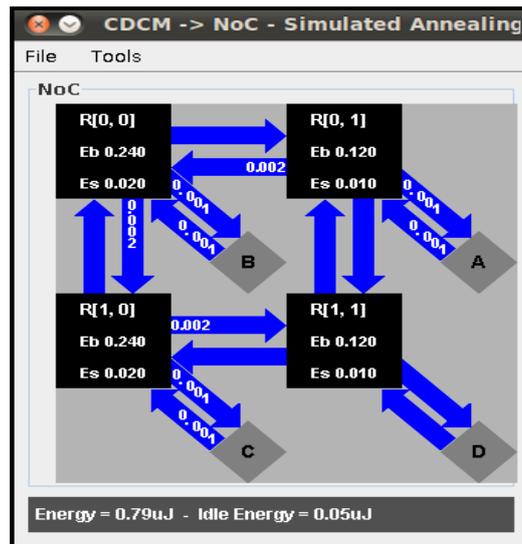
O grafo CDCG permite definir as características de tempo de processamento e quantidade de comunicação. Ele também permite particionar a aplicação em tarefas e definir quais são os eventos que devem acontecer antes que cada tarefa seja executada, e quais eventos devem ser disparados após sua execução.



**Figura 39: Ferramenta de criação de aplicações CDCM através de grafos CDCG no ambiente CAFES.**

Uma vez descrita a aplicação, o ambiente CAFES permite realizar o mapeamento das tarefas para um modelo de rede de dimensões definidas. O mapeamento considera um conjunto de requisitos, visando minimizar o consumo de energia e a redução no tempo de execução da aplicação na rede. Ao todo, o ambiente permite três estratégias de mapeamento, sendo que no escopo deste trabalho, o mapeamento só é utilizado para conhecer em quais endereços tarefas que compõem uma aplicação devem ser mapeadas, não sendo detalhado. Na Seção 7.2, comenta-se uma adaptação do processo de mapeamento, uma vez que o mapeamento como é feito atualmente assume uma rede síncrona. A Figura 40 descreve a ferramenta de mapeamento do ambiente CAFES para o grafo ilustrado pela Figura 39, onde cada endereço da rede comporta uma tarefa.

Uma vez realizado o mapeamento, o ambiente CAFES foi adaptado para gerar um arquivo contendo os valores referentes à aplicação e ao mapeamento, utilizadas pela ferramenta de geração de tráfego do ambiente ATLAS, conforme ilustrado na Figura 38(b) para geração dos arquivos de tráfego e do Testbench. A seguir, detalham-se as características da aplicação adicionadas a este arquivo.



**Figura 40: Ferramenta de mapeamento do ambiente CAFES. Cada tarefa da aplicação é mapeada em um dos endereços da rede.**

#### 4.3.2 FORMATO DE REPRESENTAÇÃO DE UM MODELO DE APLICAÇÃO

O arquivo do modelo de aplicação é gerado em um diretório informado pelo usuário durante o processo do mapeamento, e contém o nome do projeto seguido de uma extensão de arquivo do tipo *model*. Este contém quatro campos que representam os dados da aplicação, e são utilizados pela ferramenta de geração de tráfego do ambiente ATLAS para gerar os arquivos de tráfego e o Testbench para o modelo de aplicação CDCM. A Figura 41 mostra um exemplo de tal arquivo, onde os quatro campos que representam os dados da aplicação são interpretados da seguinte forma:

- ❖ **Noc Size:** Informa o tamanho da rede utilizada pela aplicação. A ferramenta de geração utiliza este valor para verificar se a rede gerada pelo usuário irá comportar o número de tarefas da aplicação.
- ❖ **Cost of Applications:** Informam a origem e o destino das tarefas, bem como o tempo de processamento em nano segundos, a quantidade de *flits* a serem transmitidos e o tempo máximo de transmissão dos dados, sempre em nano segundos.
- ❖ **Dependence of Applications:** Informam as dependências entre as tarefas, e quais delas estão relacionadas ao processo de inicialização e término da simulação.
- ❖ **Mapping of Applications:** Informam em quais roteadores as tarefas devem ser mapeadas.

A partir destes campos a ferramenta de geração de tráfego realiza a geração dos arquivos que compõem as mensagens a serem transmitidas e o Testbench, formado pelos arquivos transmissores e receptores do tráfego.

```

1 #
2 Noc Size
3 #
4 2 2
5 #
6 Cost of Applications
7 0 A - B 20 : 60 : 0
8 1 B - C 40 : 40 : 0
9 2 C - D 60 : 20 : 0
10 #
11 Dependence of Applications
12 START 0
13 END
14 0 1
15 1 2
16 2 END
17 #
18 Mapping of Applications
19 A 0 0
20 B 1 0
21 C 0 1
22 D 1 1

```

Figura 41: Arquivo que contém as características de um modelo de aplicação CDCM.

### 4.3.3 ADAPTAÇÃO DA INTERFACE DO GERADOR DE TRÁFEGO

A interface gráfica do gerador de tráfego foi adaptada para dar suporte à leitura do arquivo que descreve o modelo de uma aplicação, conforme ilustrado na Figura 42. Após o modelo ser carregado verifica-se se a extensão do arquivo é válida, e então se efetua a leitura dos campos do arquivo. Durante a leitura do arquivo, o ambiente verifica se as dimensões da rede utilizada pela aplicação são iguais ao projeto da rede gerada, bem como outros valores da aplicação relacionada às capacidades da rede. Entre estes, pode-se citar, por exemplo, o tamanho das mensagens ou o tempo de processamento, que variam conforme o comprimento do *flit* utilizado na rede.

Durante a leitura, o ambiente informa se o arquivo que descreve a aplicação está de acordo com as restrições impostas por um projeto de rede. Uma vez que estão de acordo, apresenta-se a mensagem de sucesso conforme ilustrado pela Figura 42, sendo que o botão *Generate* é então liberado. A ativação deste gera os arquivos do tráfego e o Testbench composto pelos transmissores, receptores e pelo arquivo que interconecta a rede aos transmissores e os receptores. Durante a geração destes arquivos o ambiente produz também um arquivo cenário de tráfego, que serve como entrada para as ferramentas de simulação de redes e de avaliação de tráfego.

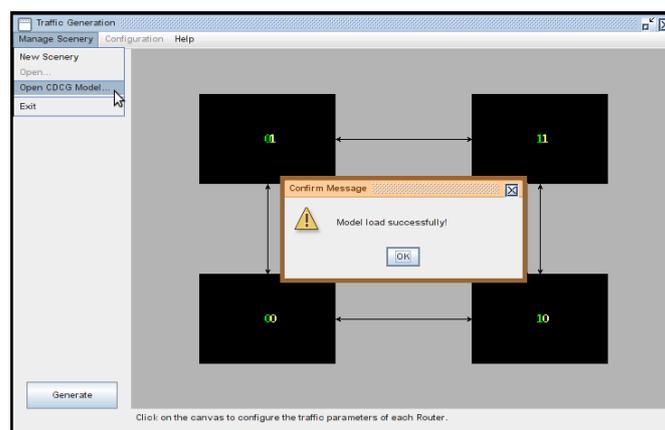


Figura 42: Interface principal do gerador de tráfego, que possibilita carregar arquivos que descrevem modelos de aplicações.

#### 4.3.4 GERAÇÃO E FORMATO INTERMEDIÁRIO DOS PACOTES DO TRÁFEGO

A partir dos valores expressos nas descrições de tarefas faz-se a geração do tráfego. Utiliza-se aqui a mesma abordagem do tráfego sintético, descrito na Seção 4.2.3, onde o tráfego de cada transmissor é produzido em arquivo texto usado como entrada por estes módulos, cada conectado a um roteador. Cada tarefa possui uma mensagem com um conjunto de informações a ser transmitida. Definiu-se que cada mensagem seria transmitida como um único pacote. Sendo assim, as mensagens de cada tarefa mapeada para um endereço da rede são vistas como um pacote descrito em um arquivo de texto que caracteriza o tráfego.

O formato intermediário dos pacotes presentes nos arquivos texto é descrito conforme ilustrado na Figura 43, que representa o modelo de um pacote do tráfego, com cinco campos, sendo estes:

- ❖ Origem: O campo origem informa de qual endereço o pacote está sendo transmitido. Ele é obtido a partir do identificador origem da tarefa, associado a um endereço da rede.
- ❖ Destino: O campo destino informa para qual roteador o pacote deve ser encaminhado. É obtido a partir do identificador destino da tarefa, associado a um endereço da rede.
- ❖ Tempo de processamento: O tempo de processamento é definido em ciclos de relógio da frequência de cada transmissor, calculado a partir do tempo de transmissão de cada tarefa.
- ❖ Tamanho do pacote: Define o tamanho do pacote em *flits*, calculado a partir do tamanho da mensagem definido para cada tarefa.
- ❖ Número de sequência: Utilizado para garantir que durante a avaliação do tráfego seja possível identificar os pacotes a partir de um identificador único em relação a todos os pacotes que trafegam na rede.

Origem (1 flit) (a)	Destino (1 flit) (b)	Tempo Processamento (1 flit) (c)	Tamanho Pacote (1 flit) (d)	Número Sequência (1 flit) (e)
---------------------------	----------------------------	--	-----------------------------------	-------------------------------------

**Figura 43: Formato intermediário de um pacote de tráfego. Os campos são: (a) Origem do pacote; (b) Roteador destino do pacote; (c) Tempo de processamento do pacote em ciclos; (d) Tamanho do pacote; (e) Número de sequência do pacote.**

#### 4.3.5 GERAÇÃO DO TESTBENCH PARA MODELOS DE APLICAÇÃO

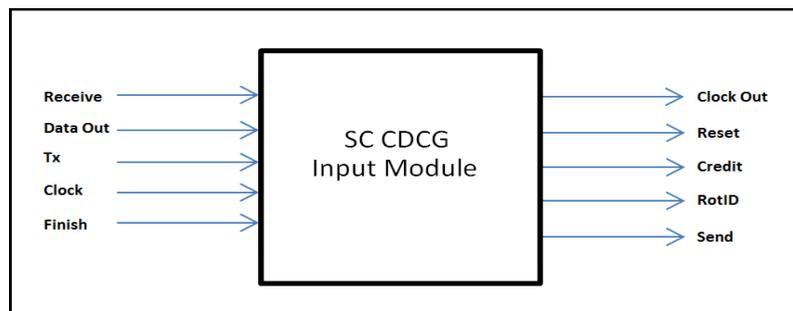
A partir dos valores informados pelo arquivo da aplicação, geram-se os respectivos transmissores, receptores e um componente que interliga a rede aos transmissores e receptores. Por existirem mudanças com relação à maneira como a transmissão é feita, um novo componente transmissor foi desenvolvido. O componente CDCG\_TopNoC que interliga a rede aos transmissores e receptores sofreu uma revisão em suas estruturas, uma vez que agora este arquivo coordena a transmissão das tarefas, considerando as dependências existentes.

A fim de garantir que a ferramenta de avaliação e as técnicas associadas tenham suporte para este modelo de tráfego, o formato do pacote, a arquitetura do receptor de tráfego e o formato

dos pacotes de saída do tráfego permanecem os mesmos. Isto permite que o ambiente de avaliação descrito no Capítulo 5 dê suporte à avaliação de tráfego de modelos de aplicações CDCM em redes HERMES-G.

#### 4.3.5.1 MÓDULO TRANSMISSOR DE TRÁFEGO: SC\_CDCG\_INPUT\_MODULE

O componente responsável pela leitura e transmissão do tráfego de modelos de aplicações CDCM tem suporte na entidade `SC_CDCG_Input_Module`. Com relação à variação nas portas de entrada e saída, comparado à entidade `SC_Input_Module`, utilizada para transmissão de tráfego sintético e descrita na Seção 4.2.4.1, o componente `SC_CDCG_Input_Module` possui uma porta de entrada *Send* e uma porta de saída *Receive*, conforme ilustra a Figura 44, utilizadas para comunicação entre o transmissor e a entidade `CDCG_TopNoC`. A porta de entrada *Send* indica ao componente transmissor que um pacote deve ser lido e transmitido. Já a porta *Receive* é utilizada pelo transmissor para confirmar a entidade `CDCG_TopNoC` quando o pacote foi transmitido à rede. A entidade `CDCG_TopNoC` através de uma estrutura interna, que faz uso das portas *Send* e *Receive* de cada um dos transmissores, realiza a coordenação das transmissões de um grafo, conforme ilustrado pelo grafo exemplo na Figura 39, preservando a ordem das transmissões. A Seção 4.3.5.2 descreve em detalhes a entidade `CDCG_TopNoC`, e o funcionamento da estrutura que realiza a coordenação das transmissões.



**Figura 44:** Descrição das portas de entrada e saída da entidade `SC_Input_Module`, que descreve o componente transmissor de pacotes da rede para o modelo de Testbench de modelos de aplicações CDCM.

A arquitetura da entidade `SC_CDCG_Input_Module` foi projetada para ter o seguinte comportamento:

- ❖ (1) – Aguarda até que o sinal de entrada *Send* esteja em '1', o que indica que um pacote deve ser transmitido, e o estado deve avançar para o item (2).
- ❖ (2) – Realiza a leitura de um pacote do arquivo de tráfego, captura o tempo de processamento em ciclos do relógio do transmissor em um contador e avança para o item (3). Se o arquivo de tráfego estiver vazio, o estado deve ser atualizado para o item (6).
- ❖ (3) – Permanece neste estado, decrementando em um ciclo o contador criado no item (2) a cada ciclo do relógio do transmissor. Quando o contador atingir zero, o que indica que o tempo de processamento da tarefa foi atingido, avança para o item (4).

- ❖ (4) – Monta o pacote e tenta transmiti-lo. Neste instante, anota o tempo que será adicionado (como Tempo Ideal do pacote), ou o momento em que a tarefa após ser executada quis ganhar acesso à rede. No momento em que o pacote ganhar acesso à rede o tempo deve ser anotado (como Tempo Transmitido), ou seja, o momento em que o pacote entrou na rede. Uma vez transmitido, avança-se para o item (5).
- ❖ (5) – Informar o componente CDCG\_TopNoC que o pacote foi transmitido através do sinal de saída *Receive* trocando seu estado de '0' para '1' e deve avançar o estado para o item (1).
- ❖ (6) – Informar o componente CDCG\_TopNoC através do sinal de saída *Finish* que o transmissor de um determinado endereço já enviou todos os pacotes referentes a seu arquivo de tráfego.

#### 4.3.5.2 MÓDULO CDCG\_TopNoC

O componente CDCG\_TopNoC é responsável por declarar e conectar os transmissores e receptores a uma instância de uma rede. A seguir descrevem-se algumas das estruturas que o componente CDCG\_TopNoC é responsável por gerar e conectar:

- ❖ (1) Produz as estruturas que descrevem os geradores de frequência para cada frequência de roteador definida e para os módulos de transmissão e recepção de pacotes criados.
- ❖ (2) Declara N instâncias de arquivos transmissores e receptores, onde N é o número de roteadores da rede. Gera ainda uma instância da rede.
- ❖ (3) Atribui a cada transmissor um endereço e conecta suas portas de entrada e saída a cada uma das portas locais dos roteadores da rede com o mesmo endereço. Além disso, a partir da frequência definida para os módulos transmissores durante a geração da rede, atribui a cada transmissor seu respectivo gerador de frequência.
- ❖ (4) A partir dos valores de frequência dos roteadores, conecta os geradores de frequência dos roteadores às portas da rede que conectam as respectivas frequências aos roteadores.
- ❖ (5) Verifica em quais casos a frequência do roteador é diferente da frequência do componente receptor do tráfego. Caso haja pelo menos uma ocorrência dessa situação, o gerador do componente CDCG\_TopNoC realiza uma cópia do arquivo modelo *HermesG\_Fifo\_Output* para o diretório de projeto, subdiretório NOC. Esta entidade é detalhada na Seção 3.1.2.4, que descreve o componente *Async\_Fifo*. O nome *HermesG\_Fifo\_Output* foi escolhido para manter a padronização de nomenclatura com os demais arquivos da rede.
- ❖ (6) Atribui a cada receptor um endereço, sendo que se a frequência do receptor for igual à do roteador correspondente ao endereço, conecta diretamente o receptor ao roteador. Se a frequência do receptor for diferente a do roteador correspondente ao endereço, conecta o receptor a uma fila *HermesG\_Fifo\_Output* e então conecta a fila ao roteador. Esta fila garante a comunicação nos casos em que as frequências do roteador e do receptor são diferentes.

Além destas condições, todo componente CDCG\_TopNoC possui uma estrutura responsável por controlar a transmissão das tarefas, de maneira que as dependências existentes entre as tarefas sejam levadas em consideração. O componente CDCG\_TopNoC, através dos sinais *Send* e *Receive* coordena a transmissão das tarefas, possuindo uma estrutura interna, formada por uma memória que armazena quais são as dependências de cada tarefa. Para demonstrar o funcionamento da estrutura que coordena a transmissão das tarefas, é proposto um estudo de caso, a partir do grafo CDCG da Figura 39. O grafo sugerido é composto por cinco vértices, sendo dois vértices responsáveis por definir o início *START* e o fim *END* do grafo, e três vértices que definem as tarefas. Cada uma das tarefas contém três argumentos, sendo eles o tempo de processamento, a origem e o destino e o tamanho do tráfego de pacotes. Os identificadores que descrevem a origem e destino são traduzidos em endereços de rede durante o mapeamento de tarefas, realizado pelo ambiente CAFES. Pelo escopo do estudo, estes valores serão desconsiderados, uma vez que não são necessários para o entendimento do mesmo.

Conforme ilustrado pela Figura 39, os três vértices que correspondem as tarefas, possuem uma dependência sequencial, em que a primeira tarefa, depende única e exclusivamente do vértice que representa o início *START*, e as demais, dependem da tarefa anterior. A primeira tarefa, que depende unicamente do início *START*, será a primeira tarefa a executar. Esta tarefa, conforme detalhado na Seção 4.3.5.1, irá receber na respectiva entidade *SC\_CDCG\_Input\_Module* onde estiver mapeada, uma requisição de transmissão na porta de entrada *Send*, enviada pela entidade CDCG\_TopNoC, que realiza a coordenação da transmissão. Uma vez que a tarefa realizar seu processamento, e transmitir seu tráfego de pacotes, a entidade *SC\_CDCG\_Input\_Module* irá informar a entidade CDCG\_TopNoC através de uma confirmação de transmissão na porta de saída *Receive* que concluiu o processamento e a transmissão do tráfego de pacotes da tarefa, em outras palavras, que realizou sua execução. Dessa forma, a entidade CDCG\_TopNoC recebe autorização para liberar a próxima tarefa que conforme descrito pelo grafo, possui uma dependência, e só deve ser executada quando a tarefa anterior confirmar sua execução. O mesmo processo que envolve enviar uma requisição de transmissão, e aguardar a confirmação da transmissão, se repete as demais tarefas, até que todas elas processem e enviem seu respectivo tráfego.

Todo componente CDCG\_TopNoC é dotado de uma estrutura que finaliza a simulação de maneira automática, uma vez que não exista mais tráfego na rede, nem haja qualquer possibilidade de tráfego existir no futuro. Uma vez que todos os transmissores tenham transmitido seus pacotes, e os receptores estejam em estado ocioso, e todas as filas da rede estiverem vazias, a estrutura produzirá um evento externo ao simulador. Este último irá retornar a chamada ao arquivo que automatiza a simulação, e a seguir irá finalizar automaticamente a simulação.

#### 4.3.5.3 ARQUIVO DE AUTOMAÇÃO DA SIMULAÇÃO

O arquivo *Simulate.do* é gerado para um projeto de rede contendo os comandos de compilação e simulação da rede, de maneira que este arquivo uma vez disparado automatiza a simulação da rede. Este arquivo contém uma condição que após disparar o comando de simulação da rede, aguarda que um evento externo do simulador informe o fim da simulação, para então

finalizar sua execução. A simulação de um tráfego de modelos de aplicação CDCM é feita a partir da leitura dos arquivos do tráfego e da transmissão dos pacotes na rede. Por fim, os receptores coletam os pacotes e escrevem os arquivos de saída, que conterão os dados dos pacotes e os tempos gastos durante a transmissão.

## 5 AMBIENTE DE AVALIAÇÃO DE TRÁFEGO PARA REDES INTRACHIP NÃO SÍNCRONAS

Este Capítulo apresenta a avaliação de tráfego relacionada a este trabalho. Nele descrevem-se as métricas utilizadas para avaliar o tráfego na rede HERMES-G, e como estas são calculadas. Além disso, descreve-se o ambiente de avaliação, comparando a proposta do ambiente adaptado com relação à versão original proposta por Tedesco em [TED05]. Este Capítulo contém mais uma das contribuições deste trabalho.

### 5.1 MÉTRICAS DE AVALIAÇÃO DE TRÁFEGO

Faz-se uso aqui de duas métricas para avaliar o tráfego. A primeira delas é a latência e a segunda a vazão. A seguir, detalha-se como estes valores são expressos, obtidos e calculados.

#### 5.1.1 LATÊNCIA

A latência é utilizada por este trabalho para calcular o tempo gasto por um pacote para ser transmitido. No escopo deste trabalho o tempo é expresso em nanosegundos (ns) ao invés de ciclos de relógio, uma vez que um ou mais roteadores, que realizam o enlace entre a origem (transmissor) e o destino (receptor) de um ou mais pacotes, podem ou não estarem operando em diferentes ciclos de relógio. Desta forma, a utilização do tempo em nanosegundo, é utilizada como uma das maneiras de representar a latência de um pacote.

Este trabalho utiliza como base de cálculo a latência de pacote, calculada a partir do intervalo entre o momento em que o primeiro *flit* do pacote está apto a entrar na rede até o momento em que o último *flit* do pacote sai da rede. Com base nas explicações das Seções 4.2.4.1 e 4.2.4.2, transmissor e receptor durante a transmissão de cada pacote anotam os momentos em que cada um dos pacotes tentou entrar e saiu da rede. Estes valores podem ser obtidos nos arquivos de saída do tráfego gerados pelos receptores. A Figura 36 descreve o formato de informações sobre cada pacote nos arquivos de saída do tráfego. O item (f) na Figura 36 representa o instante, em ciclos de relógio da frequência do transmissor (a partir do início da simulação) quando o primeiro *flit* do pacote estava apto entrar na rede. O item (i) da Figura 36 define o instante, em ciclos de relógio da frequência do receptor quando o último *flit* do pacote deixou a rede. O cálculo da latência de um pacote ocorre durante a avaliação de tráfego. Primeiro, os valores que descrevem o momento em que cada pacote ficou pronto para entrar na rede, e o instante em que cada pacote saiu da rede são convertidos de ciclos de relógio para tempo absoluto após o início da simulação. As frequências do transmissor e do receptor podem ser diferentes. Por esta razão, o cálculo da latência ocorre durante a avaliação de tráfego, quando o ambiente conhece as frequências de ambos os componentes. Feito isso, a Equação 4 é usada, nela do instante de chegada do último *flit* do pacote no receptor subtrai-se o instante em que o pacote ficou apto a

ser transmitido, ambos valores em tempo absoluto. O valor resultante é o tempo gasto pelo pacote para ser transmitido, ou seja, a latência do mesmo, representada em nanosegundos.

$$\text{Latência pacote} = \text{Tempo chegada último flit pacote no receptor} - \text{Tempo ideal para transmissão do pacote}$$

**Equação 4: Equação que realiza o cálculo da latência de um pacote. A latência é calculada a partir da subtração do instante em que o pacote está apto a entrar na rede do instante em que o último flit do pacote deixou a rede.**

### 5.1.2 VAZÃO

A vazão mede a quantidade de informação transmitida entre dois pontos da rede em um intervalo de tempo. Aqui vazão é expressa em Megabits por segundo (Mbps).

Do ponto de vista da rede, a vazão é a quantidade de bits transmitidos divididos pelo tempo gasto para transmissão do pacote. A quantidade de bits transmitida refere-se ao tamanho do pacote, obtido através de campos do pacote no arquivo de saída do tráfego. Já o tempo de transmissão corresponde à latência do pacote, obtida pelo cálculo descrito na Seção 5.1.1.

O cálculo da vazão de um pacote é realizado durante o processo de avaliação de tráfego. A Equação 5 denota o cálculo da vazão, pela divisão do tamanho do pacote em bits pela latência do pacote em tempo absoluto (nanosegundos). A vazão calculada é então multiplicada por 1000 para ser expressa em Mbps. Esta multiplicação claramente converte bits/ns em Mbps.

$$\text{Vazão do pacote (Mbps)} = \left( \frac{\text{Tamanho do pacote (bits)}}{\text{Latência do pacote (ns)}} * 1000 \right)$$

**Equação 5: Equação que realiza o cálculo da vazão de um pacote. A vazão é calculada a partir da divisão do tamanho do pacote pelo tempo gasto (latência) de transmissão do pacote. Logo após o valor é multiplicado por 1000 para ser expresso em Mbps.**

## 5.2 MEDIDAS ESTATÍSTICAS NAS MÉTRICAS DE AVALIAÇÃO

Além dos valores de vazão e latência medidos durante a transmissão de um pacote, o ambiente de avaliação de tráfego proposto por Tedesco [TED05] propõe a utilização de outras medidas durante a avaliação do tráfego. A seguir detalham-se as medidas utilizadas e como seu cálculo é feito.

### 5.2.1 LATÊNCIA

O ambiente de avaliação de tráfego faz uso de duas medidas de latência para um conjunto de pacotes: (i) A latência média; (ii) O desvio padrão da latência. Além disso, o ambiente também calcula a latência ideal de transmissão de um pacote na rede, a partir de características da rede e do caminho utilizado durante o roteamento.

### 5.2.1.1 LATÊNCIA IDEAL DE UM PACOTE

A latência ideal de um pacote informa o tempo mínimo de transmissão de um pacote da origem ao destino, calculado a partir das características da rede. No escopo do ambiente de avaliação de tráfego, a latência ideal é utilizada para comparar a latência medida de um pacote com um tempo mínimo de transmissão, que pode em si ser uma medida otimista, levando em consideração que o caminho entre o transmissor e o receptor esteja sempre disponível, não existindo contenção de pacotes concorrentes na rede.

O cálculo da latência ideal considera uma série de características da rede, entre estas o número de roteadores existentes entre a origem e o destino, a frequência de operação de cada um dos roteadores e o algoritmo de roteamento utilizado. Devido à necessidade de se conhecer durante a avaliação os roteadores do caminho, o cálculo da latência ideal só pode ser feito com precisão para roteamento determinístico, uma vez que é impossível conhecer exatamente o caminho usado em roteamento adaptativo, sendo esta uma opção de roteamento que redes do tipo HERMES-G dão suporte durante a geração parametrizável de redes. O tempo de transmissão de um pacote em cada roteador é composto pelo tempo gasto pelo primeiro *flit* do pacote para realizar o roteamento, que deve ser somado ao tempo de transmissão dos demais *flits* do pacote. Primeiramente detalha-se o cálculo do tempo de roteamento, o processo mais complexo de ser calculado. O primeiro passo executado no cálculo é verificar que roteadores encontram-se entre a origem e o destino do pacote. Em seguida, verificam-se quais são as filas síncronas (HERMESGS) e quais são as filas bi síncronas (HERMESG) utilizadas entre os roteadores. Ao todo podem existir duas combinações de roteadores, sendo que cada uma delas consome um tempo diferente para realizar a escrita, a sincronização do dado e a requisição de roteamento.

Uma primeira combinação faz uso da fila síncrona (HERMESGS) conectada a outra fila síncrona (HERMESG), o que consome cinco ciclos de relógio da frequência do roteador para escrever, sincronizar e realizar o roteamento do primeiro *flit* do pacote. Este valor em ciclos é convertido para tempo absoluto, utilizando o valor da frequência do roteador.

A segunda combinação faz uso da fila bi síncrona (HERMESG), que pode estar conectada a uma fila síncrona (HERMESGS) ou a uma fila bi síncrona (HERMESG). Este caso consome um tempo não determinístico entre a escrita do primeiro *flit* e a sincronização da fila. Por regra, a fila consome meio ciclo de relógio da frequência de escrita para escrever o primeiro *flit* do pacote, e gasta um tempo não determinístico, que no pior caso pode levar até um ciclo de relógio da frequência de leitura para sincronização da fila. Sendo assim, este trabalho assume o pior caso para o cálculo da latência ideal, considerando os efeitos não determinísticos causados pela sincronização da fila. Podem assim existir cenários em que o tempo medido para transmissão de um pacote seja menor que a latência ideal calculada. No total, o tempo gasto para escrever, sincronizar e rotar o primeiro *flit* do pacote nesta combinação de roteadores é no pior caso, de meio ciclo de relógio da frequência de escrita e sete ciclos de relógio da frequência de leitura. Quatro ciclos são gastos para o roteamento e três para a sincronização dos ponteiros da fila. Este valor em ciclos é convertido para tempo absoluto, utilizando os valores das frequências de leitura e de escrita do roteador.

Uma vez conhecidos os roteadores e seus respectivos tempos gastos para transmitir o primeiro *flit* do pacote, é feito um somatório dos tempos gastos em cada um dos roteadores existentes onde é obtido o tempo ideal de transmissão do primeiro *flit*. Para os demais *flits* que compõem um pacote é feita uma busca pelo componente com a menor frequência entre a origem e o destino, sendo que nesta busca, são considerados tanto os roteadores quanto o transmissor e o receptor do caminho. A menor frequência encontrada é convertida em tempo absoluto e multiplicada pelos demais *flits* que compõem um pacote, uma vez que conforme definido pelo protocolo de transmissão da rede, depois de realizada a conexão entre a origem e o destino, cada *flit* irá gastar um ciclo da frequência mais lenta entre a origem e o destino para ser transmitida. O tempo obtido para os demais *flits* do pacote é somado ao tempo do primeiro *flit* do pacote, obtendo assim a latência ideal total do pacote.

#### 5.2.1.2 LATÊNCIA MÉDIA DE UM FLUXO DE PACOTES

A latência média de um conjunto de pacotes é uma medida que representa o tempo médio gasto por um conjunto de pacotes para ser transmitido. O cálculo da média, ilustrado na Equação 6 é feito a partir da divisão do somatório de todos os valores de cada uma das latências dos pacotes medidas durante a transmissão de um determinado fluxo de pacotes origem-destino do tráfego pelo número total de pacotes.

$$\text{Latência média} = \frac{\sum \text{Latência de pacote}}{\text{Número de Pacotes}}$$

**Equação 6: Equação que calcula a latência média de um conjunto de pacotes a partir do somatório das latências medidas dos pacotes dividido pelo número total de pacotes.**

#### 5.2.1.3 DESVIO PADRÃO DA LATÊNCIA DE UM FLUXO DE PACOTES

O desvio padrão da latência de um conjunto de pacotes é uma medida que representa a dispersão das latências dos pacotes em relação à média. Quanto menor for este valor, menor será a variação nos tempos de transmissão dos pacotes, indicando a uniformidade (ou não) da transmissão de um conjunto de pacotes. O cálculo do desvio padrão é ilustrado na Equação 7. Ele é feito a partir da raiz quadrada da divisão de um somatório. O somatório é feito sobre as latências dos pacotes, cada uma sendo previamente subtraído do valor da latência média dos pacotes, valor este elevado ao quadrado antes de ser acumulado. Após o somatório é dividido pelo número total de pacotes, antes de se aplicar a raiz quadrada.

$$\text{Desvio Padrão da Latência} = \sqrt{\frac{\sum (\text{Latência pacote} - \text{média})^2}{\text{Número de Pacotes}}}$$

**Equação 7: Equação que calcula o desvio padrão da latência de um conjunto de pacotes.**

## 5.2.2 VAZÃO

O ambiente de avaliação de tráfego faz uso de duas medidas no cálculo da vazão de um conjunto de pacotes: (i) Avazão média ;(ii) O desvio padrão da vazão. Além disso, o ambiente também calcula a vazão ideal de transmissão de um pacote na rede, a partir das características de uma rede e do caminho utilizado durante o roteamento.

### 5.2.2.1 VAZÃO IDEAL DE UM PACOTE

A vazão ideal informa a quantidade de informação possível de ser transmitida entre um caminho da rede em um intervalo de tempo, calculado a partir das características da rede. No escopo deste trabalho, a vazão ideal é utilizada para avaliar a vazão medida de um conjunto de pacotes contra a vazão ideal do caminho, que representa a quantidade máxima de informação possível de ser transmitida na unidade de tempo, levando em consideração que o caminho entre o transmissor e o receptor esteja sempre disponível, não existindo contenção de pacotes concorrentes na rede.

A vazão ideal de um pacote é calculada conforme a Equação 8, pela divisão do tamanho do pacote em bits pela latência ideal do pacote calculada conforme detalhado na Seção 5.1.1.1. O valor depois de calculado é multiplicado por 1000 para ser convertido em Mbps.

$$\text{Vazão ideal do pacote (Mbps)} = \left( \frac{\text{Tamanho do pacote (bits)}}{\text{Latência ideal do pacote (ns)}} \right) * 1000$$

**Equação 8: Equação que calcula a vazão ideal de um pacote pela divisão do tamanho do pacote em bits pela latência ideal do pacote em tempo absoluto. Depois de calculado o valor é multiplicado por 1000 para ser convertido em Mbps.**

### 5.2.2.2 VAZÃO MÉDIA DE UM FLUXO DE PACOTES

A vazão média de um conjunto de pacotes é uma medida que representa a quantidade média de informação transmitida por um conjunto de pacotes. O cálculo da vazão média é feito usando a Equação 9, a partir da divisão do somatório de todos os valores das vazões calculadas de um conjunto de pacotes pelo número total de pacotes. O cálculo da vazão de cada pacote é feito conforme detalha a Seção 5.1.2, a partir da latência medida de cada pacote.

$$\text{Vazão média} = \frac{\sum \text{Vazão de pacote}}{\text{Número de Pacotes}}$$

**Equação 9: Equação que calcula a vazão média de um conjunto de pacotes, a partir do somatório das vazões calculadas dos pacotes dividido pelo número total de pacotes.**

### 5.2.2.3 DESVIO PADRÃO DA VAZÃO DE UM FLUXO DE PACOTES

O desvio padrão da vazão de um conjunto de pacotes é uma medida que representa a dispersão das vazões dos pacotes em relação à média. Quanto menor for este valor, menor será a variação da quantidade de informação transmitida, indicando a uniformidade (ou não) da transmissão de um conjunto de pacotes. O cálculo do desvio padrão é ilustrado na Equação 10. Ele é feito a partir da raiz quadrada da divisão de um somatório. O somatório é feito sobre as vazões dos pacotes, cada uma sendo previamente subtraído do valor da vazão média dos pacotes, valor este elevado ao quadrado antes de ser acumulado. Após o somatório é dividido pelo número total de pacotes, antes de se aplicar a raiz quadrada.

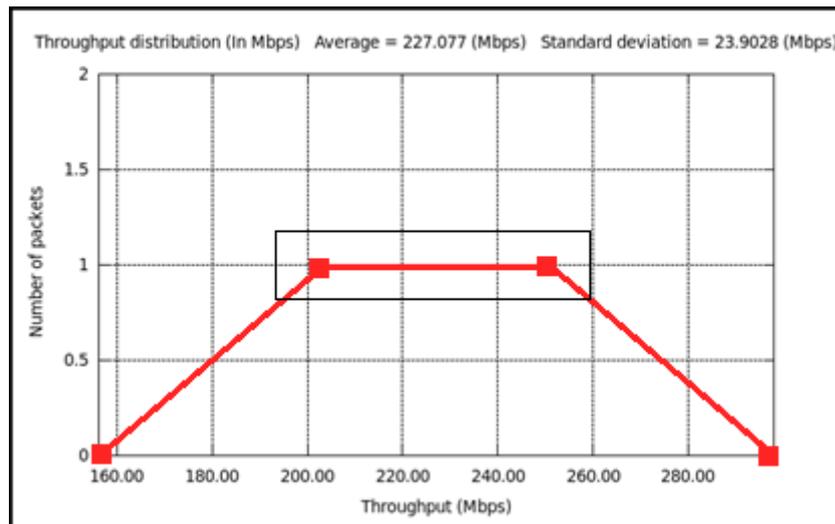
$$\text{Desvio Padrão da Vazão} = \sqrt{\frac{\sum(\text{Vazão de pacote} - \text{média})^2}{\text{Número de Pacotes}}}$$

**Equação 10: Equação que calcula o desvio padrão da vazão de um conjunto de pacotes.**

A partir das métricas e das medidas descritas pelas Seções 5.1 e 5.2 o ambiente de avaliação de tráfego originalmente proposto Tedesco [TED05] foi adaptado para dar suporte aos cálculos e à avaliação de tráfego para redes HERMES-G. O ambiente de avaliação de tráfego faz uso do arquivo de cenário do tráfego, definido na Seção 4.2.1, como arquivo de entrada. Este permite que para o mesmo projeto de rede haja vários cenários já simulados, sendo possível avaliar cada um de maneira independente a partir do cenário de tráfego. Ao todo quatro ferramentas que permitem avaliar o tráfego foram modificadas: (i) Distribuição de vazões; (ii) Distribuição de latências; (iii) Analisador de latências; (iv) Relatório global. Estas ferramentas foram adaptadas para dar suporte ao cálculo do tráfego para redes HERMES-G, e serão detalhadas a seguir.

## 5.3 DISTRIBUIÇÃO DE VAZÕES DE UM TRÁFEGO

A ferramenta de distribuição de vazões através de gráficos como o ilustrado na Figura 45 disponibiliza a visualização dos valores de vazão medidas de um determinado conjunto de pacotes entre um transmissor e um receptor.

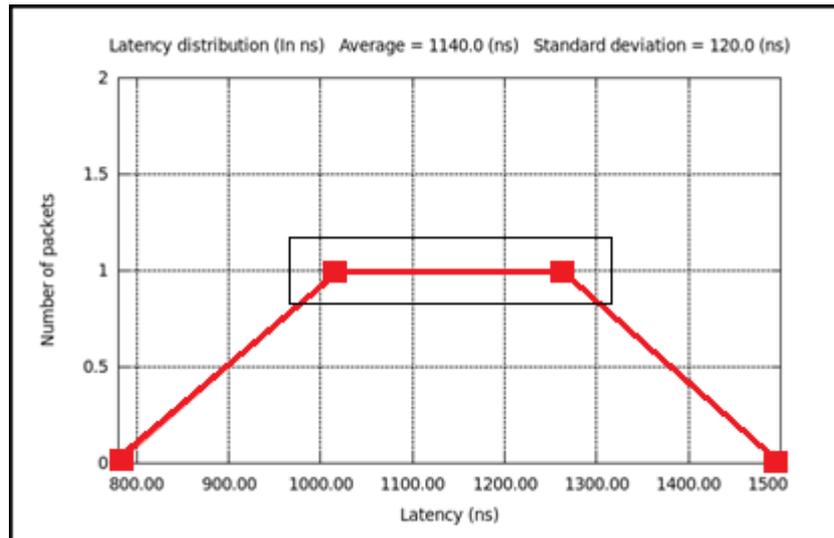


**Figura 45: Interface da ferramenta distribuição de vazões do ambiente de avaliação de tráfego. No exemplo, é usado um tráfego uniforme de 400Mbps de dois pacotes transmitidos do endereço 00 ao endereço 11. Na figura, observa-se que dois pacotes foram transmitidos, um pacote com vazão aproximada de 200Mbps, e outro pacote com vazão de 250Mbps.**

Dentre os valores relacionados estão a vazão de cada um dos pacotes, calculada a partir do método descrito na Seção 5.1.2, a média das vazões dos pacotes, calculada a partir do método descrito na Seção 5.2.2.2 e o desvio padrão da vazão, calculado a partir do método descrito na Seção 5.2.2.3. A Figura 45 descreve um tráfego uniforme injetado a 400Mbps de dois pacotes enviados do endereço 00 ao endereço 11. A partir dos campos disponibilizados pela ferramenta de distribuição de vazões é possível avaliar que para o tráfego exemplo utilizado, um pacote teve vazão de aproximadamente 200Mbps e o outro pacote vazão de 250Mbps. Além disso, a média dos valores foi de 227.0Mbps e o desvio padrão de 23.9Mbps.

#### 5.4 DISTRIBUIÇÃO DE LATÊNCIAS DE UM TRÁFEGO

A ferramenta distribuição de latências através de gráficos como o ilustrado na Figura 46 disponibiliza a visualização dos valores de latência de um determinado conjunto de pacotes enviados entre um transmissor e um receptor. Dentre os valores relacionados estão a latência de cada um dos pacotes, calculada a partir do método descrito na Seção 5.1.1, a média das latências, calculada a partir do método descrito na Seção 5.2.1.2 e o desvio padrão da vazão, calculado a partir do método descrito na Seção 5.2.1.3. A Figura 46 descreve um tráfego uniforme injetado a 400Mbps de dois pacotes do endereço 00 ao endereço 11. A partir dos campos disponibilizados pela ferramenta de distribuição de latência é possível observar que para o tráfego exemplo utilizado, um pacote teve latência de aproximadamente 1020ns e o outro pacote latência de 1260ns. Além disso, a média dos valores foi de 1140ns e o desvio padrão de 120ns.



**Figura 46:** Interface da ferramenta distribuição de latências do ambiente de avaliação de tráfego. O exemplo usa um tráfego uniforme de 400Mbps de dois pacotes transmitidos do endereço 00 ao endereço 11. Na figura, observa-se que dois pacotes foram transmitidos, um pacote com latência aproximada de 1020ns, e outro pacote com latência de 1260ns.

## 5.5 ANALISADOR DE LATÊNCIAS

A ferramenta analisadora de latências gera saída como a ilustrada na Figura 47. Esta permite a visualização de um conjunto de pacotes de todos os tráfegos que levaram mais ou menos tempo para serem transmitidos. Para todo valor de latência informa-se também o número de sequência e o endereço de origem e de destino do pacote. A latência dos pacotes é calculada a partir do método descrito na Seção 5.1.1. Os valores de número de sequência, origem e destino são capturados diretamente dos pacotes nos arquivos do tráfego.

Global Latency Analysis Report				
	Latency	Sequency Number	Source	Target
1	2540.0	12	10	20
2	2420.0	11	10	20
3	2300.0	10	10	20
4	2180.0	9	10	20
5	2060.0	8	10	20
6	1940.0	7	10	20
7	1820.0	6	10	20
8	1700.0	5	10	20
9	1400.0	2	00	20
10	1140.0	4	10	20

**Figura 47:** Interface da ferramenta analisador de latências do ambiente de avaliação de tráfego, que permite visualizar latências específicas de pacotes de tráfegos.

## 5.6 RELATÓRIO GLOBAL

A ferramenta de relatório global, ilustrada na Figura 48, permite avaliar todos os tráfegos transmitidos entre cada par de endereços origem e destino na rede de duas formas. A primeira delas apresenta os valores de vazão e latência ideal de cada tráfego, calculados conforme os métodos descritos nas Seções 5.2.1.1 e 5.2.2.1. A segunda apresenta valores medidos de cada

tráfego respectivo: (i) A média da vazão e da latência; (ii) O desvio padrão da vazão e da latência; (iii) A vazão e latência mínima e máxima de cada tráfego. Além disso, o ambiente disponibiliza para cada tráfego dois botões que permitem visualizar gráficos de distribuição de vazões e latências detalhadas nas Seções 5.4 e 5.3.

HermesG NoC Flow Control: CreditBased (1 cycle per flit) Virtual Channels: 0 Scheduling: RoundRobin Dimensions: 3 x 3 (9 routers) Flit Width: 16 Buffer Depth: 16 Routing Algorithm: Algorithm XY															
Generated					Measured								Graphs		
Source	Target	Packets	Throughput In (Mbps)		Packets	Throughput In (Mbps)				Latency In (ns)				Latency Distribution	Throughput Distribution
			Ideal(WC)	Ideal(WC)		Average	Standard Deviation	Minimum	Maximal	Average	Standard Deviation	Minimum	Maximal		
00	20	2	341.33	750.0	2	216.91	34.06	182.85	250.98	1210.0	190.0	1020.0	1400.0	<b>VIEW</b>	<b>VIEW</b>
10	20	10	426.66	600.0	10	164.87	98.11	100.78	441.37	1868.0	574.60	580.0	2540.0	<b>VIEW</b>	<b>VIEW</b>
		12			12	190.89				1539.0					

Report generated on 16:6:37 Tuesday, July 24, 2012. AD by TrafficMeasurer

**Figura 48: Interface da ferramenta relatório global que permite visualizar latência e a vazão ideais e vazão e a latência medidas de todos os tráfegos.**



## 6 RESULTADOS OBTIDOS

Este capítulo apresenta um conjunto de experimentos desenvolvidos com objetivo de validar os conceitos propostos por este trabalho e avaliar de maneira resumida alguns cenários de rede e de tráfego apresentados nos capítulos anteriores referentes à proposta deste trabalho. Dentre os estudos de caso desenvolvidos, o autor pretende demonstrar como é possível aumentar o desempenho de um cenário de tráfego proposto variando as características da rede como o tipo de roteamento utilizado e a frequência de operação. Além disso, são feitas comparações em área e consumo de energia usando diferentes variações da rede.

### 6.1 ESTUDO DE CASO 1

Este estudo de caso avalia impactos em desempenho, área e consumo de energia de tráfegos concorrentes em dois cenários de rede, um síncrono e outro não síncrono com 3 ilhas de frequência. O cenário usa uma rede 8x8, com 64 roteadores. Destes, 32 conectam processadores, 16 conectam elementos de memória, e 16 elementos de entrada e saída. Segue-se aqui tendências no projeto de MPSoC atuais, 32 processadores operando em duas ilhas com maior frequência de operação, diferente dos componentes de memória e entrada e saída que operam em outra ilha de frequência. A Figura 49 mostra detalhes deste estudo de caso.

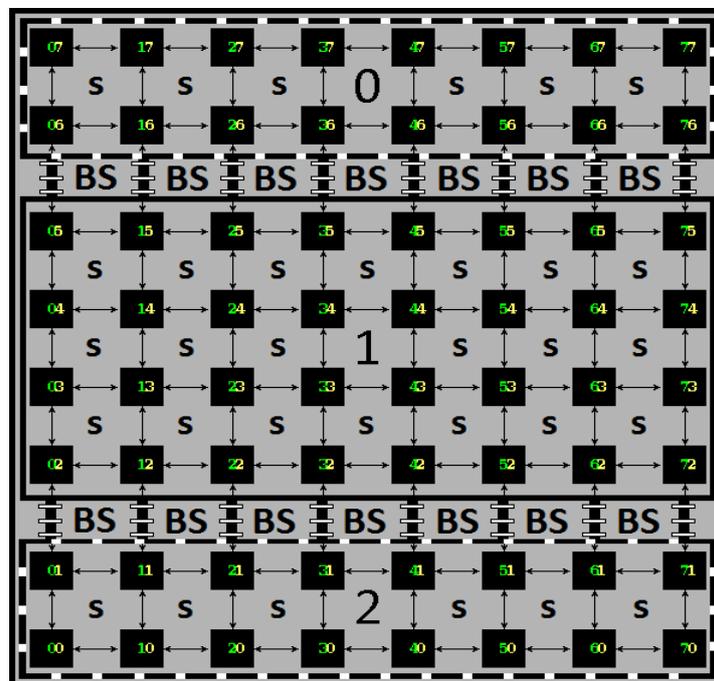


Figura 49: Rede 8x8 contendo duas ilhas de frequência. Os retângulos pontilhados (0 e 2) possuem trinta e dois processadores operando a 50MHz, conectados a roteadores que operam a 100MHz. Nestas duas ilhas todos os roteadores se comunicam utilizando filas síncronas (S). O retângulo preto (1) possui dezesseis elementos de memória e dezesseis controladores de entrada e saída que operam a 50MHz, e estão conectados a roteadores que operam a 50MHz. Repare que as filas bi-síncronas (BS) são utilizadas para comunicação unicamente entre as ilhas de frequência distintas.

Em ambos os cenários de rede foram utilizadas configurações contendo comprimento de canais igual a 16bits, profundidade das filas igual a 16flits, algoritmo de roteamento *Negative First Non Minimal* e codificação de ponteiros das filas Gray. Para este estudo de caso optou-se por trabalhar unicamente com filas síncronas do tipo HERMES-GS, uma vez que utilizando um cenário síncrono contendo apenas filas síncronas do tipo HERMES, o tempo de sincronização e transmissão da fila influenciaria nos resultados. Os processadores serão mapeados nos endereços contidos nos retângulos 0 e 2 ilustrados na Figura 49. Os roteadores destas duas ilhas representadas pelos retângulos 0 e 2 irão operar a frequência de 100MHz no cenário não síncrono e 50MHz no cenário síncrono. Já os elementos de memória e de entrada e saída, serão mapeados nos endereços contidos no retângulo 1. Os roteadores desta ilha representada pelo retângulo 1 irão operar a frequência de 50MHz tanto no cenário de rede síncrono como não síncrono. Todos os elementos transmissores e receptores irão operar a frequência de 50MHz.

O cenário de tráfego propõe fazer uso de três tipos de transmissões. Na primeira transmissão, quatro processadores mapeados respectivamente nos endereços (00/07/40/47) irão enviar a outros quatro processadores mapeados nos endereços (31/36/71/76) 1000 pacotes de 16flits cada ou seja, 256000bits (250KBytes) transmitidos a uma taxa de injeção uniforme de 400Mbps o que representa 50% da capacidade de transmissão do processador. Esta transmissão em um cenário real assemelha-se a uma migração de tarefa entre dois processadores, realizada dentro de uma mesma ilha de frequência. O processo de migração que interrompe momentaneamente a execução da aplicação, demanda que o tempo despendido entre a transmissão da aplicação seja o menor possível de maneira a evitar que aumente o tempo de execução da aplicação, causado pela migração da tarefa. Sendo assim, o tráfego proposto pretende fazer uso de 50% da capacidade de transmissão do processador tendo como objetivo reproduzir este comportamento.

Na segunda transmissão, quatro elementos de memória mapeados nos endereços (62/12/14/64) irão enviar a quatro dos processadores mapeados nos endereços (17/67/60/10) 10000 pacotes de 16flits cada ou seja, 2560000bits (2.5MBytes) transmitidos a uma taxa de injeção exponencial decrescente que varia em um intervalo de 200Mbps a 600Mbps com taxa média de 400Mbps, o que representa respectivamente um intervalo de 25% a 75% com média de 50% da taxa máxima de transmissão de cada uma das memórias. Esta transmissão em um cenário real assemelha-se a um elemento de memória transmitindo de forma concorrente valores de seus respectivos endereços da memória a um ou mais processadores. Por existir concorrência no acesso à memória por outros processadores, a distribuição temporal do tráfego é dada conforme a distribuição exponencial decrescente, uma vez que uma grande parte do total de pacotes é transmitida a uma taxa de injeção mais baixa, e poucos pacotes transmitidos a uma taxa de injeção mais alta.

Na terceira transmissão, quatro elementos de memória mapeados nos endereços (04/74/03/73) irão enviar a quatro elementos de entrada e saída, mapeados nos endereços (75/05/72/02), 1000 pacotes de 16flits cada ou seja, 256000bits (250KBytes) transmitidos a uma taxa de injeção normal variada em um intervalo de 200Mbps a 600Mbps com taxa média de 400Mbps, o que representa um intervalo de 25% a 75% com média de 50% da taxa máxima de transmissão de cada um dos elementos de memória. Esta transmissão em um cenário real

assemelha-se a uma transmissão de uma memória volátil do tipo DRAM, (do inglês *Dynamic random-access memory*) a uma memória não volátil do tipo EEPROM, (do inglês *Electrically Erasable Programmable Read-Only Memory*) responsável pelo armazenamento permanente de uma informação já computada presente em uma das memórias do circuito. A distribuição normal é utilizada na caracterização do tráfego, onde a maior parte dos pacotes é transmitida próxima à média uma vez que atrasos de sincronização e início da recepção reduzem a velocidade da transmissão.

Os resultados com relação ao desempenho obtido após realizar a simulação dos cenários de rede síncrona e não síncrona são ilustrados pela Tabela 4. Com relação a vazão e a latência média da transmissão do tráfego referente a migração de uma tarefa entre quatro processadores, representado na Tabela 4 pelo item “(P - P) Média”, o cenário não síncrono obteve ganho de 37.19% na redução da latência que representa o tempo de migração da tarefa e aumento na vazão de 37.13% comparado ao cenário síncrono. Como no cenário não síncrono os processadores operam em uma ilha de frequência duas vezes mais rápida que no cenário síncrono, foi possível assim obter melhor desempenho no tráfego referente a migração de tarefas.

Os resultados para o tráfego referente a transmissão de dados dos elementos de memória para os processadores, é ilustrado pela Tabela 4 através do item “(M-P) Média”. Com relação a vazão e latência média da transmissão dos tráfegos, o cenário não síncrono apresentou 29.31% redução na latência que representa o tempo de transmissão e 14.94% de aumento na vazão comparado ao cenário síncrono. O ganho de desempenho é obtido graças aos roteadores das ilhas de frequência que operam com frequência duas vezes maior na rede não síncrona. Uma vez que o tráfego faz uso de caminhos compostos por roteadores presentes nas ilhas que operam ao dobro da frequência, e possuem caminhos mais rápidos, a tendência natural tende a obtenção de melhores resultados.

O terceiro e último cenário de tráfego, referente a transmissão dos elementos de memória aos elementos de entrada e saída, é ilustrado pela Tabela 4 através do item “(M - I/O) 20-22”.

**Tabela 4: Valores de vazão e latência média de todos os pacotes de cada tráfego obtidos durante a simulação dos tráfegos para os cenários de rede síncrona e não síncrona.**

Cenário de tráfego	Latência Média Síncrona (ns)	Vazão Média Síncrona (Mbps)	Latência Média Não Síncrona (ns)	Vazão Média Não Síncrona (Mbps)
(P – P) Média	1015.39 ns	252.41 Mbps	637.71 ns	401.52 Mbps
(M – P) Média	88825.83 ns	58.58 Mbps	62783.35 ns	68.87 Mbps
(M – I/O) Média	11613.92 ns	85.36 Mbps	16113.86 ns	81.41 Mbps
Média Global	33818.38 ns	132.12 Mbps	26511.64 ns	183.94 Mbps

Com relação a vazão e latência média da transmissão dos tráfegos, o cenário não síncrono apresentou 4.85% maior latência de transmissão e 38.74% menor vazão comparado ao cenário síncrono. Esta transmissão demonstra uma desvantagem no uso de redes não síncronas. Uma vez que os elementos de memória e entrada e saída estão presentes em uma mesma ilha de

frequência o tempo necessário para sincronização dos ponteiros de leitura e escrita da fila bi síncrona é maior o que aumenta a latência de transmissão e reduz a vazão dos pacotes do tráfego.

A média global calculada a partir das médias de todos os tráfegos demonstra que o cenário de rede não síncrona foi aquele que obteve 28.17% maior vazão e 21.60% menor latência quando comparado ao cenário de rede síncrona. Além do desempenho, este trabalho avalia o impacto na área e no consumo de energia gerado pelo uso das filas bi síncronas e do uso das ilhas de maior frequência de operação em alguns roteadores no cenário de rede não síncrona. Para obtenção da área, este trabalho realizou a síntese dos cenários de rede propostos através do uso da ferramenta Xilinx ISE 14.6, utilizando um dispositivo de prototipação FPGA Virtex5 XC5VLX330T. Este FPGA foi escolhido por existirem disponíveis placas de prototipação com o respectivo FPGA no laboratório de pesquisa do grupo do autor. Os resultados de área são obtidos em número de LUTs de quatro entradas utilizadas pelo FPGA para representar o projeto de cada rede. A Tabela 5 apresenta os resultados da área obtida na síntese para os cenários de rede propostos. Observa-se nos resultados que o cenário de rede não síncrona fez uso de 10.63% maior área que o cenário síncrono.

Além da área, o consumo de energia dos cenários de rede e seus respectivos tráfegos utilizados foi avaliado. Para obtenção da energia este trabalho fez uso no total de três ferramentas sendo elas: (i) Mentor Graphics Modelsim 10.0c; (ii) Synopsys Design Compiler Graphical; (iii) Synopsys PrimeTime. Durante a simulação RTL dos cenários de rede feita através da ferramenta Modelsim, a atividade de chaveamento da rede foi anotada em um arquivo do tipo VCD (do inglês *Value change dump*). A ferramenta Design Compiler Graphical foi utilizada para compilação e síntese dos cenários de rede em VHDL utilizando as regras de projeto do processo de fabricação TSMC 180nm. Por fim, a saída da síntese e a atividade de chaveamento em VCD foram entrada para a ferramenta PrimeTime, responsável pela simulação e cálculo do consumo de energia. Ainda, foi necessário definir durante o uso da ferramenta PrimeTime alguns parâmetros sendo eles: (i) Atividade de chaveamento; (ii) Unidade de tempo utilizada pela simulação; (iii) Valores de frequência utilizados pelos sinais de relógio dos cenários de rede avaliados. O consumo de energia foi aferido sob diferentes aspectos do circuito sendo eles: (i) Somatório do consumo de energia dos sinais de relógio, (do inglês *clock*); (ii) Consumo de energia da lógica combinacional do circuito; (iii) Consumo de energia da lógica sequencial do circuito. Através destas métricas é possível entender sob quais perspectivas o consumo de energia dos cenários de rede avaliados foram diretamente influenciados pela variação das frequências utilizadas.

A Tabela 5 apresenta os resultados referentes ao consumo de energia dos cenários de rede avaliados. Verifica-se que com relação ao item (i), que apresenta o somatório do consumo de energia dos sinais de relógio que o cenário não síncrono apresentou 31.8% maior consumo de energia nos sinais referentes aos relógios comparado ao cenário síncrono uma vez que 50% de seus roteadores operavam ao dobro da frequência comparado ao cenário síncrono. Com relação ao item (ii) que apresenta o consumo de energia da lógica combinacional, o cenário não síncrono apresentou 7.8% maior consumo que o cenário síncrono. Com relação ao item (iii), ambos os cenários de rede apresentaram o mesmo consumo de energia da lógica sequencial. Por fim, no somatório de todas as métricas referentes ao consumo de energia, o cenário não síncrono

apresentou 31.8% maior consumo de energia que o cenário síncrono sendo os sinais de relógio os maiores causadores do aumento do consumo.

**Tabela 5: Valores de área em número de LUTs de quatro entradas e consumo de energia. O consumo de energia (em nanoJoule/s) é dado em: (i) Consumo de energia dos sinais de relógio;(ii) Lógica combinacional do circuito; (iii) Lógica sequencial do circuito.**

Modelo de rede	Área em número de LUTs de quatro entradas	Consumo de Energia (Clocks)	Consumo de Energia (Combinacional)	Consumo de Energia (Sequencial)
Rede Síncrona	11832	0.448 nJ/s	$2.104^{e-4}$ nJ/s	0.0995 nJ/s
Rede Não Síncrona	13240	0.657 nJ/s	$2.284^{e-4}$ nJ/s	0.0995 nJ/s

Este estudo de caso demonstra que o uso de redes não síncronas permite a obtenção de expressivo ganho de desempenho, como demonstrado pelo cenário de tráfego “(P - P) Média” em que um conjunto de processadores trocava mensagens em alusão a um possível cenário de migração de tarefas. Estes resultados apontam as vantagens no uso de ilhas de frequência oportunas, utilizadas para alguns componentes em específico onde existam restrições quanto a atrasos e/ou tempo máximo de transmissão de mensagens. Também, o cenário “(M-P) Média” em que um conjunto de memórias e processadores trocam mensagens em alusão a transmissão de dados entre memórias e processadores aponta melhora no desempenho da transmissão. Este resultado aponta uma vantagem no uso de ilhas de frequência uma vez que o tráfego tende por natureza do roteamento a fazer uso de caminhos contendo roteadores que operem em diferentes frequências de operação podendo assim obter melhores tempos de transmissão.

A métrica que trata do consumo de área, aponta existir um compromisso entre o ganho de desempenho e a demanda excedente na área. O consumo adicional de área pode também resultar em um não atendimento as restrições e requisitos de um projeto, como por exemplo o aumento no custo de fabricação do circuito, devendo o desempenho adicional ser avaliado sobre esta perspectiva. Em uma relação entre o consumo de área e desempenho, considerando a área dos cenários avaliados e a média global da vazão e da latência, o cenário de rede não síncrono comparado ao síncrono apresentou 19.62% melhor eficiência no uso de área por vazão apresentada e 10.63% melhor eficiência no uso de área por latência apresentada. De forma resumida, o projeto não síncrono foi aquele que apresentou a melhor eficiência no uso da área para os cenários de rede e de tráfego avaliados. De acordo com os resultados deste estudo de caso, o maior consumo de área deve apenas servir como métrica para avaliar o custo adicional de fabricação do circuito frente as restrições de projeto existentes. Por fim, mesmo consumindo mais área, o projeto não síncrono foi aquele que reportou a melhor eficiência no uso da área frente ao desempenho atingido.

Em relação a métrica que trata do consumo de energia, o projeto não síncrono apresentou excedente comparado ao síncrono de 31.8% maior consumo de energia. Em uma relação que trata o consumo de energia, considerando a média global da vazão e da latência obtida, o cenário de rede não síncrono comparado ao síncrono apresentou 0.76% melhor eficiência no uso de energia

por vazão apresentada e 7.67% melhor relação de consumo de energia por latência. Da mesma forma que no uso da área, o projeto não síncrono foi aquele que apresentou a melhor eficiência no uso da energia. Da mesma maneira que a área, os resultados deste estudo de caso apontam que o maior consumo de energia demandado pelo projeto não síncrono deve apenas servir como métrica para avaliar o custo adicional de fabricação do circuito frente as restrições de projeto existentes. Por fim, mesmo consumindo mais energia, o projeto não síncrono foi aquele que reportou a melhor eficiência no uso da energia comparado ao desempenho atingido.

## 6.2 ESTUDO DE CASO 2

Este estudo de caso propõe estudar o impacto no tempo de transmissão, custo de área e consumo de energia variando a codificação dos ponteiros das filas bi síncronas e no uso da fila síncrona descrita pela seção 3.1.2.5, utilizada para geração de redes não síncronas. Propõe-se quatro cenários de rede ao todo, sendo que em todos eles são utilizadas as mesmas características, quais sejam dimensões de redes 8x8, totalizando sessenta e quatro roteadores, largura de canais de 16bits, profundidade das filas de 16flits e algoritmo de roteamento *West First Non Minimal*. A frequência de operação dos roteadores apresenta uma topologia de quatro ilhas de frequência conforme ilustra a Figura 50, onde cada ilha faz uso de uma frequência de operação diferente para os roteadores. As respectivas ilhas de roteadores são representadas através de quadrados, conforme ilustra a Figura 50 e operam as frequências de: (i) Quadrado (0), 40MHz; (ii) Quadrado (1), 60MHz; (iii) Quadrado (3), 80MHz; (iv) Quadrado (4), 100MHz. Todos os elementos transmissores e receptores operam em uma frequência de 50MHz.

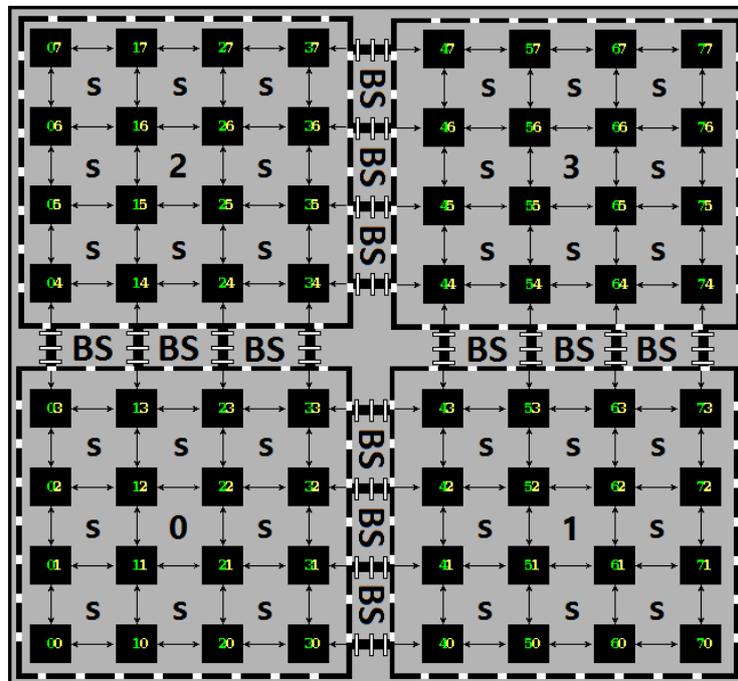


Figura 50: Rede 8x8 contendo quatro ilhas de frequência, ilustradas pelos quadrados (0/1/2/3). A fila síncrona é utilizada para comunicação entre os roteadores de cada ilha. Os roteadores periféricos as ilhas, fazem uso de filas bi síncronas para garantir a comunicação entre os diferentes domínios de frequências.

No total, quatro variações da codificação de ponteiros das filas bi síncronas são propostas. No primeiro cenário (8x8G), todas as filas sendo elas bi síncronas irão possuir codificação de ponteiros Gray. O segundo cenário (8x8J), todas as filas sendo elas bi síncronas irão possuir codificação de ponteiros Johnson. No terceiro e quarto cenários serão utilizadas filas síncronas e bi síncronas conforme ilustra a Figura 50. No terceiro cenário será utilizada a combinação de filas síncronas e bi síncronas com uso de codificação de ponteiros Johnson (8x8JS). No quarto cenário será utilizada a combinação de filas síncronas e bi síncronas com uso de codificação de ponteiros Gray (8x8GS).

O cenário de tráfego sugerido para os quatro cenários avaliados visa explorar o tempo gasto quando a combinação de filas síncronas e bi síncronas são utilizadas na transmissão de pacotes. No total oito cenários de tráfego são propostos sendo quatro deles mapeados nos endereços (00/70/07/77) responsáveis pelo envio de 1000 pacotes de 16flits cada, totalizando um tráfego de 256000bits (250KBytes) distribuído de maneira uniforme a 160Mbps, ou 20% da capacidade do transmissor. Os destinos dos tráfegos são respectivamente os endereços (77/07/70/00). Estes tráfegos têm como objetivo servir para verificação de perdas causadas no desempenho do tráfego quando diferentes ilhas de frequência são utilizadas para transmissão de pacotes. Os demais 4 cenários de tráfego, mapeados nos endereços (30/40/37/47) são responsáveis pelo envio de 1000 pacotes de 16flits cada totalizando um tráfego de 256000bits (250KBytes) distribuídos de maneira uniforme a 160Mbps, ou 20% da capacidade do transmissor. Os destinos dos tráfegos são respectivamente os endereços (01/71/06/76).

A Tabela 6 descreve os resultados obtidos após realizar a simulação dos tráfegos para os cenários de rede propostos. Como as filas com codificação Gray e codificação Johnson gastam sempre o mesmo tempo para transmissão, os resultados obtidos quando utilizadas ambas as filas foi equivalente, conforme ilustra a Tabela 6. A variação no desempenho foi detectada no uso de filas síncronas e bi síncronas tanto para a codificação Gray como codificação Johnson. Com relação aos resultados obtidos, observa-se que para os cenários de tráfego mapeados nos endereços (30/40/37/47) tendo como destino os endereços (01/71/06/76) respectivamente, o desempenho obtido para cada tráfego é diretamente proporcional a frequência de operação dos roteadores e suas respectivas ilhas de frequências. Estes tráfegos foram propostos com objetivo de criar contenção nos caminhos dos tráfegos mapeados nos endereços (00/70/07/77) tendo como destino os endereços (77/07/70/00). O objetivo da contenção faz com que o roteamento adaptativo utilizado busque diferentes caminhos para transmissão dos pacotes podendo assim escolher diferentes ilhas de frequência como caminho e gerar diferentes variações resultados dos pacotes que compõem os tráfegos propostos.

Dos tráfegos propostos para geração de contenção na rede, mapeados nos endereços (30/40/37/47), o que obteve o menor desempenho foi o tráfego (30 – 01) com latência de 1936.02 nanosegundos e vazão de 132.76Mbps. Este tráfego fez uso da ilha representada pelo quadrado (0) ilustrada pela Figura 50, que opera a frequência de 40MHz sendo esta a ilha com menor frequência de operação. Por consequente, o cenário que obteve o melhor desempenho foi o tráfego (47 – 76) com latência de 1145 nanosegundos e vazão de 223.58Mbps. Este tráfego fez uso da ilha representada pelo quadrado (3) ilustrada pela Figura 50, que opera a frequência de 100MHz sendo esta a ilha com maior frequência de operação. Com relação ao uso de filas

síncronas e bi síncronas, a combinação no uso de ambas as filas foi aquele que resultou na geração dos cenários de melhor desempenho. Para ambas as codificações de ponteiros, obteve-se melhora na vazão média de 9.45% no uso de ambas as filas síncronas e bi síncronas e redução de 10.15% na latência média de todos os tráfegos. Uma vez que a fila síncrona consome menos tempo para alinhamento dos ponteiros de leitura e escrita da fila, quando utilizada em combinação com filas bi síncronas, permite reduzir o tempo de transmissão e melhora do desempenho da rede.

**Tabela 6: Valores de vazão e latência média obtidos durante a simulação dos tráfegos propostos. Como cenário de rede é feito uso da fila bi síncrona com codificação de ponteiros Gray e Johnson e da fila síncrona combinada com filas bi síncronas.**

Cenário de tráfego	8x8G		8x8GS		8x8J		8x8JS	
	Latência Média (ns)	Vazão Média (Mbps)						
00 – 77	3344.75	76.63	2661.06	96.21	3344.75	76.63	2661.06	96.21
30 – 01	1936.02	132.76	1681.03	152.53	1936.02	132.76	1681.03	152.53
40 – 71	1438.71	177.94	1291.66	198.19	1438.71	177.94	1291.66	198.19
70 – 07	3586.05	71.55	3173.65	80.72	3586.05	71.55	3173.65	80.72
07 – 70	3572.21	71.77	3447.11	74.26	3572.21	71.77	3447.11	74.26
37 – 06	1677.02	152.66	1639.04	156.19	1677.02	152.66	1639.04	156.19
47 – 76	1145	223.58	1046.35	244.67	1145	223.58	1046.35	244.67
77 – 00	3150.07	81.25	2894	88.51	3150.07	81.25	2894	88.51
Média	2481.35	123.51	2229.38	136.41	2481.35	123.51	2229.38	136.41

A Tabela 7 apresenta quais foram os resultados de área obtidos para cada cenário de rede proposto através da síntese feita utilizando a ferramenta Xilinx ISE 10.1 para um dispositivo de prototipação FPGA Virtex5 XC5VLX330T. Este dispositivo foi escolhido por existirem disponíveis placas de prototipação com o respectivo FPGA no laboratório de pesquisa do grupo do autor. Os resultados de área são obtidos em número de LUTs de quatro entradas utilizadas pelo FPGA para representar o projeto de cada rede.

A partir dos resultados de área conforme ilustra a Tabela 7 é possível concluir que a codificação Johnson conforme descrito pelo item (8x8J) apresenta aumento de 29.18% no consumo de área em relação a codificação Gray, descrito pelo item (8x8G). Estes resultados se assemelham aos obtidos por [HEC11] conforme descrito pela seção 3.1.2.3, onde é comparada a área de cada uma das codificações propostas. Já para os cenários de rede que utilizam a combinação das filas síncronas e bi síncronas, no cenário Johnson conforme descrito pelos itens (8x8J) e (8x8JS), houve uma redução de 35.93% na área quando utilizadas filas síncronas e bi síncronas simultaneamente comparada a versão que apenas faz uso de filas bi síncronas. No cenário Gray conforme descrito pelos itens (8x8G) e (8x8GS), houve uma redução de 23.32% na

área da versão que utiliza filas síncronas e bi síncronas simultaneamente comparada a versão que apenas faz uso de filas bi síncronas. O autor esperava obter resultados de redução na área utilizando combinações de filas síncronas e bi síncronas, uma vez que [PON08] afirma existir um aumento na área da fila bi síncrona comparada a uma fila síncrona.

**Tabela 7: Valores de área em número de LUTs de quatro entradas e consumo de energia. O consumo de energia (em nanoJoule/s) é dado em: (i) Consumo de energia dos sinais de relógio; (ii) Lógica combinacional do circuito; (iii) Lógica sequencial do circuito.**

Cenário de rede	Área em número de LUTs de quatro entradas	Consumo de Energia (Clocks)	Consumo de Energia (Combinacional)	Consumo de Energia (Sequencial)
8x8G	18098	0.6177 nJ/s	$2.678^{e-4}$ nJ/s	0.0398 nJ/s
8x8GS	13876	0.6177 nJ/s	$2.661^{e-4}$ nJ/s	0.0399nJ/s
8x8J	25557	0.6177 nJ/s	$2.678^{e-4}$ nJ/s	0.0398 nJ/s
8x8JS	16374	0.6177 nJ/s	$2.677^{e-4}$ nJ/s	0.0398 nJ/s

Além da área, o consumo de energia dos cenários de rede e seus respectivos tráfegos utilizados foi avaliado. A mesma metodologia utilizada para obtenção do consumo de energia do estudo de caso anterior foi adotada para este estudo de caso. A Tabela 7 apresenta os resultados referentes ao consumo de energia dos cenários de rede avaliados. Verifica-se que com relação ao item (i), que o consumo de energia do sinal de relógio foi idêntico para os cenários avaliados uma vez que todos eles operam a mesma frequência. Com relação aos itens (ii) e (iii) que apresentam respectivamente o consumo de energia da lógica combinacional e sequencial, detecta-se existir uma pequena variação do consumo de energia sendo os cenários que fazem uso da combinação de filas síncronas e bi síncronas aqueles que apresentam menor consumo de energia. Pelas conclusões obtidas no estudo de caso anterior, o sinal de relógio é aquele responsável pelo maior consumo de energia. Sendo assim, as pequenas variações no consumo da lógica combinacional e sequencial não representam impacto no consumo de energia total das métricas avaliadas.

Avaliando os resultados pelo tempo de execução, a técnica proposta por este trabalho que permite fazer uso de filas síncronas e não síncronas obteve melhora na média dos resultados em um aumento de 9.45% na vazão e redução de 10.15% na latência. A métrica referente ao consumo da área é aquela que melhor representa os ganhos no uso da técnica que faz uso de filas síncronas e bi síncronas de forma simultânea. Os resultados para os cenários de rede propostos apontam uma redução de 23.32% na área utilizando filas síncronas e filas bi síncronas com codificação Gray e 35.93% utilizando filas síncronas e filas bi síncronas com codificação Johnson.

### 6.3 ESTUDO DE CASO 3

Este estudo de caso propõe estudar o impacto no desempenho, na área e no consumo de energia ao utilizar diferentes algoritmos de roteamento disponíveis no ambiente de geração de redes. Ainda, conforme detalhado na seção 3.1.2.2, cada roteamento disponibilizado apresenta

uma versão do componente “*Crossbar*” contendo minimizações nas portas de saída deste componente. Este estudo de caso tem como objetivo avaliar o ganho em área disponibilizado pelo uso da técnica, suportada pela ferramenta de geração de redes implementada por este trabalho. Ao todo são utilizadas sete redes, todas compostas pelas seguintes características, dimensões de redes 8x8, largura dos canais que interligam os roteadores de 32bits, profundidade das filas igual a 8flits e frequência de operação igual a 50MHz para os roteadores, transmissores e receptores. Em cada uma das sete redes propostas é variado o algoritmo de roteamento, sendo eles: (i) XY; (ii) *Negative First Minimal*; (iii) *Negative First Non Minimal*; (iv) *North Last Minimal*; (v) *North Last Non Minimal*; (vi) *West First Minimal*; (vii) *West First Non Minimal*.

O cenário de tráfego sugerido para os sete cenários avaliados visa explorar a capacidade do algoritmo de roteamento em assumir diferentes caminhos para o tráfego uma vez exista congestionamento na rede. No total oito cenários de tráfego são propostos sendo quatro deles mapeados nos endereços (00/70/07/77) responsáveis pelo envio de 1000 pacotes de 32flits cada, totalizando um tráfego de 1024000bits (1000KBytes) distribuídos de maneira uniforme a 400Mbps, ou 25% da capacidade do transmissor. Os destinos dos tráfegos são respectivamente os endereços (77/07/70/00). Estes tráfegos têm como objetivo servir para verificação de perdas causadas no desempenho do tráfego quando diferentes ilhas de frequência são utilizadas para transmissão de pacotes. Os demais 4 cenários de tráfego, mapeados nos endereços (30/40/37/47) são responsáveis pelo envio de 1000 pacotes de 32flits cada totalizando um tráfego de 1024000bits (1000KBytes) distribuídos de maneira uniforme a 400Mbps, ou 25% da capacidade do transmissor. Os destinos dos tráfegos são respectivamente os endereços (01/71/06/76).

A Tabela 8 apresenta os resultados de área obtidos para cada um dos sete cenários de rede através da síntese feita utilizando a ferramenta Xilinx ISE 10.1 para um dispositivo de prototipação FPGA Virtex5 XC5VLX330T. Os resultados de área são obtidos em número de LUTs de quatro entradas utilizadas pelo FPGA para representar o projeto de cada rede. A partir dos resultados de área obtidos para cada um dos sete cenários de rede propostos conforme ilustra a Tabela 8, o algoritmo de roteamento que apresentou o maior consumo de área foi o *North Last Non Minimal* e o que apresentou o menor consumo de área foi o algoritmo XY. A diferença em relação ao consumo de área entre os algoritmos de roteamento que mais e menos fizeram uso de área é de 3.5%. Em uma relação que assume o consumo de área pelo desempenho obtido, o roteamento *West First Non Minimal* foi aquele que apresentou a menor latência por área consumida. O roteamento XY foi aquele que apresentou a maior vazão por área consumida. Considerando a combinação das métricas de vazão e latência média, o roteamento *West First Non Minimal* foi aquele que apresentou a melhor eficiência entre o uso da área e o desempenho obtido.

Além da área, o consumo de energia dos cenários de rede e seus respectivos tráfegos utilizados foi avaliado. A mesma metodologia utilizada para obtenção do consumo de energia do estudo de caso anterior foi adotada para este estudo de caso. A Tabela 8 apresenta os resultados referentes ao consumo de energia dos cenários de rede avaliados.

**Tabela 8: Valores de área em número de LUTs de quatro entradas e consumo de energia. O consumo de energia (em nanoJoule/s) é dado em: (i) Consumo de energia dos sinais de relógio; (ii) Lógica combinacional do circuito; (iii) Lógica sequencial do circuito.**

Algoritmo de roteamento	XY	NFM	NFNM	NLM	NLNM	WFM	WFNM
Área em número de LUTs de quatro entradas	11733	11816	11832	11820	12159	11824	12140
Consumo de Energia (Clocks)	0.4709 nJ/s						
Consumo de Energia (Combinacional)	4.123e <sup>-4</sup> nJ/s	4.094e <sup>-4</sup> nJ/s	4.094e <sup>-4</sup> nJ/s	4.054e <sup>-4</sup> nJ/s	4.727e <sup>-4</sup> nJ/s	3.932e <sup>-4</sup> nJ/s	4.727e <sup>-4</sup> nJ/s
Consumo de Energia (Sequencial)	0.0975 nJ/s						

Verifica-se que com relação aos itens (i) e (iii), que tratam respectivamente do consumo de energia do sinal de relógio e da lógica sequencial, não existem variações no consumo de energia nos cenários de rede avaliados uma vez que todos operam na mesma frequência. O item (ii) que trata do consumo de energia da lógica combinacional, apresenta pequenas variações no consumo sendo o cenário *North Last Non Minimal*, aquele que apresenta o maior consumo de energia quanto a lógica combinacional. O cenário de rede que *West First Minimal* apresenta o menor consumo de energia na lógica combinacional. A diferença no consumo de energia combinacional entre os algoritmos de roteamento avaliados foi de 16.8% entre o *North Last Non Minimal* que apresenta o maior consumo e o *West First Minimal* que apresenta o menor consumo. Assumindo uma relação entre o consumo de energia pelo desempenho obtido a partir da vazão e latência média dos tráfegos, o roteamento *West First Minimal* foi aquele que apresentou a melhor eficiência no uso da energia da lógica combinacional em relação ao desempenho obtido.

A Tabela 9 descreve os resultados referentes ao desempenho da simulação dos tráfegos para ambos os cenários de rede propostos. O campo (Lat) representa a latência média em (ns) para cada um dos tráfegos, o campo (Vaz) representa a vazão média em (Mbps) para cada um dos tráfegos avaliados. A partir dos modelos de rede propostos, o uso do algoritmo de roteamento *West First Non Minimal* para a média de todos os tráfegos foi aquele que atingiu o melhor desempenho. O roteamento *North Last Minimal* foi aquele que reportou o pior desempenho. As diferenças no desempenho entre o melhor e o pior algoritmo são de 41.5 vezes para a latência média e 18.39% para a vazão média. Observa-se na avaliação do desempenho um baixo desvio padrão da média da vazão e da latência para os algoritmos de roteamento avaliados, exceto para as variações mínimas e não mínimas do roteamento *Negative First*.

Em uma avaliação mais apurada do desempenho do roteamento do roteamento *Negative First*, os tráfegos (00 – 77) e (07 – 70) foram os responsáveis por apresentar baixo desempenho favorecendo para a redução da média da vazão e da latência do somatório dos tráfegos deste

cenário avaliado. As diferenças no desempenho apresentadas para as versões mínimas e não mínimas destes tráfegos, demonstram uma possível deficiência do roteamento *Negative First* em realizar o roteamento dos tráfegos sugeridos para este estudo de caso. Com exceção deste algoritmo de roteamento em específico, os demais roteamentos apresentaram desempenhos próximos entre si, sendo o algoritmo XY o mais eficiente no consumo da área pelo desempenho obtido e o *West First Minimal*, o mais eficiente no consumo de energia pelo desempenho obtido.

**Tabela 9: Valores de vazão e latência média para os cenários de tráfego simulados para os sete cenários de rede que variam o algoritmo de roteamento.**

	XY		NFM		NFMN		NLM		NLNM		WFM		WFNM	
	Lat	Vaz	Lat	Vaz	Lat	Vaz	Lat	Vaz	Lat	Vaz	Lat	Vaz	Lat	Vaz
00 – 77	2400	213.33	2570	199.3	2570	199.31	172934	8.11	36236	199.3	2500	204.79	2499	204.8
30 – 01	1079	474.07	1159	447.6	1154	447.67	1049.7	489.24	252053	447.6	1079	474.14	1079	474.1
40 – 71	1079	474.07	1000	512	1000	512	1408.98	385.73	241336	512	1000	512	1000	512
70 – 07	2400	213.33	2540	201.9	2540	201.93	2489.02	205.94	17251	201.9	2420	211.56	2400	213.3
07 – 70	2400	213.33	2764	186.9	2764	186.90	5741.3	99.35	10779	186.9	2679	191.08	2679	191
37 – 06	1079	474.11	1359	378.2	1359	378.28	1292.54	470.08	1013	378.2	1099	465.54	1079	474.3
47 – 76	1079	474.11	995	514.5	995	514.56	997.92	513.09	1003	514.5	980	522.37	980	522.3
77 – 00	2400	213.33	2689	192.6	2689	192.69	5776.86	118.11	17977	192.6	2400	213.32	2400	213.3
Média	1739	343.7	1884	329.1	1884	329.1	23961	286.1	72206	329.1	1769	349.3	1764	350.6

#### 6.4 ESTUDO DE CASO 4

Este estudo de caso propõe estudar a possibilidade de gerar redes menores com relação as suas características como a profundidade das filas que interconectam os roteadores, capazes de consumir menos área e que mesmo assim atingir desempenho equivalente ou superior em alguns cenários de tráfego através do uso de diferentes frequências de operação. Neste estudo de caso duas redes são propostas, a primeira delas possuindo dimensões de rede 8x8, comprimento de canais igual a 32bits, profundidade das filas igual a 32flits, algoritmo de roteamento XY, codificação de ponteiros Gray para a fila bi síncrona e frequência de operação de 50MHz para todos os roteadores, transmissores e receptores do tráfego. A segunda rede foi gerada com dimensões de rede 8x8, comprimento de canais igual a 32bits, profundidade das filas igual a 4flits, algoritmo de roteamento XY e codificação de ponteiros Gray para a fila bi síncrona. A frequência de operação para os roteadores (01/00/10/20/30/40/50/60/70/71) e roteadores (06/07/17/27/37/47/57/67/77/76) foi definida em 100MHz, os demais roteadores, transmissores e receptores receberam frequência de operação de 50MHz. Este cenário de teste propõe que parte dos roteadores que formam o caminho do tráfego, utilizado pelo roteamento XY para transmissão dos pacotes operem ao dobro da frequência de transmissão sendo esta a estratégia

utilizada pela rede não síncrona na tentativa de adquirir desempenho equivalente e/ou superior ao cenário síncrono que possui quatro vezes maior capacidade de armazenamento nas filas das portas dos roteadores e por consequente possui maior capacidade de transmissão.

Como cenário de tráfego são propostos 8 tráfegos, 4 deles mapeados nos endereços (00/70/07/77) responsáveis pelo envio de 1000 pacotes de 16flits cada, distribuídos de maneira uniforme a 400Mbps, ou 25% da capacidade do transmissor. Os destinos dos tráfegos são respectivamente os endereços (77/07/70/00). Os demais 4 cenários de tráfego, mapeados nos endereços (30/40/37/47) são responsáveis pelo envio de 1000 pacotes de 16flits distribuídos de maneira uniforme a 400Mbps, ou 25% da capacidade do transmissor. Os destinos dos tráfegos são respectivamente os endereços (01/71/06/76). Estes 4 tráfegos em específico têm como objetivo criar concorrência no uso da rede aos demais 4 tráfegos apresentados anteriormente. Uma vez que múltiplos tráfegos irão competir pelo menos caminho, a profundidade das filas de entrada dos roteadores será de fundamental importância para aumentar a vazão dos canais, evitando a contenção que reduz a vazão e aumenta a latência dos pacotes. Uma vez que a rede não síncrona possui quatro vezes menos capacidade de armazenamento das filas de entrada dos roteadores, o cenário de tráfego proposto apresenta um desafio para a rede não síncrona, que possui como vantagem 50% dos roteadores que formam o caminho dos tráfegos operando ao dobro da frequência do cenário síncrono. A Tabela 10 apresenta os resultados obtidos para cada modelo de rede proposto através da síntese feita utilizando a ferramenta Xilinx ISE 10.1 para um dispositivo de prototipação FPGA Virtex5 XC5VLX330T.

**Tabela 10: Valores de área em número de LUTs de quatro entradas, número de BUFG e consumo de energia. O consumo de energia (em nanoJoule/s) é dado em: (i) Consumo de energia dos sinais de relógio; (ii) Lógica combinacional do circuito; (iii) Lógica sequencial do circuito.**

Modelo da rede	Área (LUT4)	Área (BUFGs)	Consumo de Energia (Clocks)	Consumo de Energia (Combinacional)	Consumo de Energia (Sequencial)
Rede Síncrona	17018	16	1.4514 nJ/s	$8.251 \times 10^{-3}$ nJ/s	0.3737 nJ/s
Rede Não-Síncrona	15988	2	0.3752 nJ/s	$5.709 \times 10^{-4}$ nJ/s	0.0501 nJ/s

Os resultados de área são obtidos em número de LUTs de quatro entradas utilizado pelo FPGA para síntese do circuito e número de BUFG utilizados para sincronização e distribuição dos sinais referentes a árvore de relógio do projeto de cada rede. A partir dos resultados de área obtidos para cada um dos cenários de rede propostos é possível verificar um aumento de 6.05% na área do cenário de rede síncrona em LUTs de quatro entradas e uma diferença de oito vezes no consumo de estruturas de armazenamento do tipo BUFG (do inglês *Global Buffer*) na área do cenário de rede síncrona. O maior consumo de área para a rede síncrona tem relação direta com a profundidade das filas de entrada dos roteadores. A diferença no uso de estruturas do tipo BUFG, têm relação com o tamanho do circuito e os custos necessários na sincronização e distribuição do sinal de relógio entre todas as estruturas do circuito. O uso de apenas um sinal de relógio síncrono dificulta a síntese do circuito que necessita por via de regra de várias estruturas BUFG para transmissão e sincronização do sinal de relógio.

Além da área, o consumo de energia dos cenários de rede e seus respectivos tráfegos utilizados foi avaliado. A mesma metodologia utilizada para obtenção do consumo de energia do estudo de caso anterior foi adotada para este estudo de caso. Na Tabela 10 verifica-se que com relação ao item (i) que trata do consumo de energia do sinal de relógio, que a rede síncrona apresentou 3.86 vezes maior consumo de energia que a rede não síncrona. Conforme detectado na avaliação da área, o projeto de rede síncrona além de exigir maior área, utilizou um maior número de estruturas do tipo BUFG para sincronização e distribuição do sinal de relógio, tendo como consequência um maior consumo de energia para o sinal de relógio. Com relação ao item (ii) que apresenta o consumo de energia da lógica combinacional, o cenário de rede síncrona apresentou 14.4 vezes maior consumo de energia que o cenário não síncrono. Já para o item (iii) que apresenta o consumo de energia da lógica sequencial, o cenário de rede síncrona apresentou 7.45 vezes maior consumo de energia que o cenário não síncrono. Mesmo a rede não síncrona operando em 20 dos 64 roteadores ao dobro da frequência, a rede síncrona apresentou maior consumo de energia.

As métricas que tratam da área e do consumo de energia, apontam que o cenário não síncrono foi aquele que fez uso do menor número de recursos. Também, aquele que reportou os melhores resultados quanto ao desempenho sendo assim o cenário mais eficiente na área e no consumo de energia. Este estudo de caso demonstra a existência de uma relação de compromisso entre as características da rede como a profundidade das filas e/ou a frequência de operação dos roteadores. Os resultados apontam as dificuldades existentes no roteamento e distribuição de sinais de relógio em circuitos que demandam grande quantidade de área, que acarretam em um maior consumo de energia. O uso de redes não síncronas, capazes de definir roteadores para operar em diferentes frequências de operação, permitem a geração de redes menores em número de recursos, que demandam menor área para o projeto de rede e consumo de energia para o roteamento e distribuição do sinal de relógio. Este projeto de rede permite por fim, a geração de circuitos mais eficientes, que consomem menor quantidade de área e energia, possuindo ainda desempenho equivalente e/ou superior.

A Tabela 11 apresenta resultados referentes ao desempenho da simulação dos tráfegos para ambos os cenários de rede propostos. Observa-se nos resultados que a rede não síncrona obteve na média dos resultados 27.9% maior vazão e 21.5% menor latência comparado ao cenário síncrono. Em todos os tráfegos conforme apresenta a Tabela 11 o cenário não síncrono apresenta melhores resultados tanto em vazão como em latência. Verifica-se que as maiores diferenças no desempenho acontecem nos tráfegos mapeados nos endereços (30/40/37/47) tendo como destino de tráfego os endereços (01/71/06/76). Na rede não síncrona o caminho entre estes endereços faz uso de roteadores que operam a frequência de 100MHz o que favorece a obtenção de melhores resultados nos tráfegos referente a aumento de vazão e redução na latência.

As métricas que tratam da área e do consumo de energia, apontam que o cenário não síncrono foi aquele que fez uso do menor número de recursos. Também, aquele que reportou os melhores resultados quanto ao desempenho sendo assim o cenário mais eficiente na área e no consumo de energia. Este estudo de caso demonstra a existência de uma relação de compromisso entre as características da rede como a profundidade das filas e/ou a frequência de operação dos

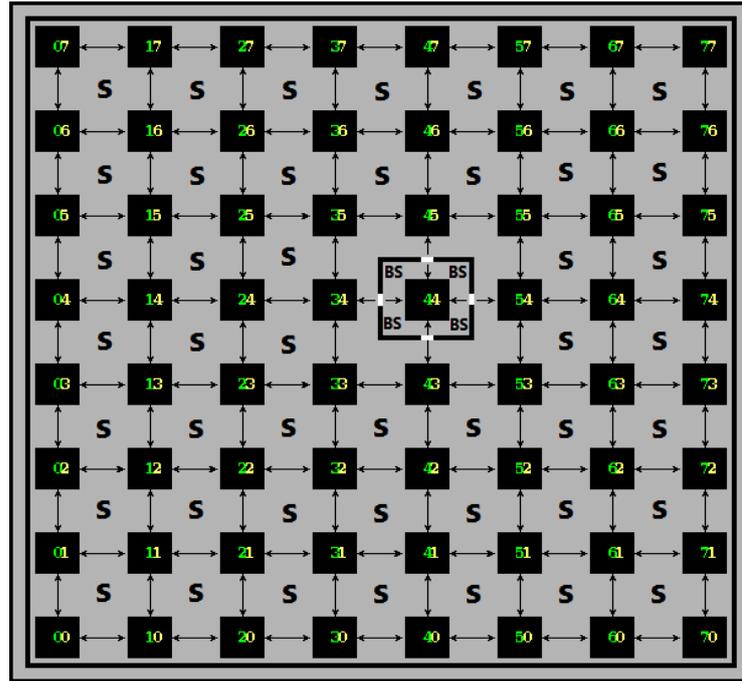
roteadores. Os resultados apontam as dificuldades existentes no roteamento e distribuição de sinais de relógio em circuitos que demandam grande quantidade de área, que acarretam em um maior consumo de energia. O uso de redes não síncronas, capazes de definir roteadores para operar em diferentes frequências de operação, permitem a geração de redes menores em número de recursos, que demandam menor área para o projeto de rede e consumo de energia para o roteamento e distribuição do sinal de relógio. Este projeto de rede permite por fim, a geração de circuitos mais eficientes, que consumam menor quantidade de área e energia, possuindo ainda desempenho equivalente e/ou superior.

**Tabela 11: Valores de vazão e latência média do tráfego obtidos durante a simulação dos cenários de rede síncrona e não síncrona propostos.**

Cenário de tráfego	Rede Síncrona		Rede Não Síncrona	
	Latência Média (ns)	Vazão Média (Mbps)	Latência Média (ns)	Vazão Média (Mbps)
00 – 77	2400 ns	213.33 Mbps	1994.96 ns	256.64 Mbps
30 – 01	1079.92 ns	474.11 Mbps	735 ns	696.59 Mbps
40 – 71	1079.92 ns	474.11 Mbps	735 ns	696.59 Mbps
70 – 07	2400 ns	213.33 Mbps	1994.96 ns	256.64 Mbps
07 – 70	2400 ns	213.33 Mbps	1994.96 ns	256.64 Mbps
37 – 06	1079.92 ns	474.11 Mbps	735 ns	696.59 Mbps
47 – 76	1079.92 ns	474.11 Mbps	735 ns	696.59 Mbps
77 – 00	2400 ns	213.33 Mbps	1994.96 ns	256.64 Mbps
Média Global	1739.96 ns	343.72 Mbps	1364.98 ns	476.76 Mbps

## 6.5 ESTUDO DE CASO 5

Este estudo de caso propõe estudar dois cenários de tráfego, um gerado a partir de um modelo de aplicação CDCM e outro através do modelo de tráfego sintético a partir da ferramenta de geração de tráfego do ambiente ATLAS. Para ambos os cenários de tráfego utilizados foi definido um cenário de rede 8x8, largura de canais igual a 32bits, profundidade das filas igual a 16flits, algoritmo de roteamento *West First Non Minimal* e codificação de ponteiro Gray para a fila bi síncrona. Optou-se por gerar uma configuração de rede formada por duas frequências de operação sendo elas de 50MHz e 100MHz. Conforme ilustra a Figura 51, todos os roteadores, transmissores e receptores, com exceção do endereço 44 operam em 50MHz. O roteador e os componentes de transmissão e recepção do endereço 44 operam em 100MHz. A Figura 51 ilustra o uso das filas síncronas e não síncronas onde observa-se que apenas no entorno do endereço 44 é feito uso da fila bi síncrona uma vez que os componentes que compõem o respectivo endereço operam em uma frequência diferente dos demais componentes que formam a rede.



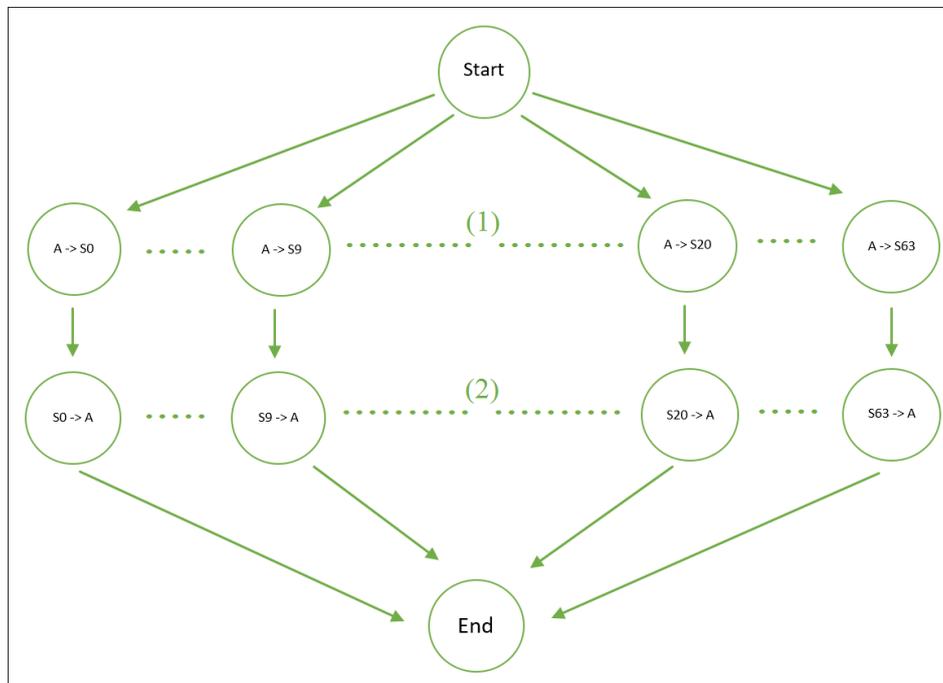
**Figura 51: Rede 8x8 formada por 64 roteadores, transmissores e receptores que operam a 50MHz exceto o roteador, transmissor e receptor do endereço 44. Este endereço é utilizado para armazenar a tarefa mestre da aplicação, responsável pela transmissão e recepção dos dados da aplicação.**

O cenário de tráfego proposto ilustrado pela Figura 52 apresenta uma aplicação de multiplicação vetorial paralela, projetada seguindo o modelo de programação paralelo mestre escravo. O modelo de programação assume por convenção que uma tarefa mestre inicialmente ficará encarregada de particionar e distribuir os dados a serem processados de maneira paralela por tarefas escravas. Conforme ilustra a Figura 52 a aplicação utilizada é composta por 64 tarefas no total sendo a tarefa mestre (A) responsável por particionar o vetor que será multiplicado de forma paralela em 64 blocos e realizar a transmissão das partes do vetor as tarefas escravas. A aplicação é dividida no total em duas fases. Na primeira fase (1), o mestre particiona o vetor presente em sua memória local em blocos e transmite cada bloco para uma cada tarefa escrava em específico. Na segunda fase (2), as tarefas escravas realizam a multiplicação dos elementos que formam o bloco recebido pela etapa anterior. Uma vez concluída a multiplicação do bloco, enviam os dados processados novamente a tarefa mestre (A), que finaliza a execução.

Uma vez que o modelo de aplicação CDCM exige que para cada tarefa do grafo seja definido o tempo de processamento e a quantidade de informação a ser transmitida por cada tarefa, este trabalho implementou a aplicação de multiplicação vetorial utilizando a linguagem de programação C. As tarefas da aplicação foram implementadas na forma de processos onde cada processo simula um elemento de processamento da rede, similar ao conceito de arquitetura de processamento distribuída implementadas pelos cenários de rede utilizadas por este estudo de caso. Os tempos de processamento e quantidade de informação transmitidas por cada processo foram anotadas por monitores adicionados ao código em linguagem C desenvolvido. Como tamanho de entrada foi utilizado um vetor de 1 milhão de inteiros positivos de 32 bits de tamanho cada. Uma vez que o volume de informação capturado extrapola as capacidades de geração de pacotes do ambiente de geração de tráfego utilizado para redes HERMES-G, os resultados

capturados foram reduzidos em uma ordem de mil vezes tanto para o tempo de processamento como para quantidade de informação transmitida.

Conforme ilustra a Figura 52, a primeira fase (1) referente a transmissão dos blocos do vetor pelo mestre aos escravos, transmitiu 65KBytes em um tempo de processamento de 1818 nanosegundos para cada tarefa escrava. A segunda fase (2) referente a a multiplicação dos blocos pelos escravos consumiu 19027 nanosegundos de tempo de processamento e transmitiu 65KBytes para cada tarefa escrava. O mapeamento das tarefas foi feito de forma mutual onde a tarefa mestre (A) foi mapeada no endereço 44, que possui o dobro da frequência de transmissão, recepção e roteamento dos demais componentes da rede. As tarefas escravas foram mapeadas nos demais seguindo uma contagem incremental que começa pelo escravo endereço número 0 para a tarefa escrava número 0 ao endereço número 63 para a tarefa número 62. A partir dos mesmos valores definidos para o tráfego de aplicação CDCM foi então criado um tráfego sintético similar.



**Figura 52: Aplicação CDCM que descreve uma multiplicação vetorial paralela implementada através do modelo de programação mestre escravo. A aplicação é implementada em duas fases sendo a fase (1) responsável pela transmissão dos blocos do vetor pela tarefa mestre (A) as tarefas escravas (S0 a S63). Na fase (2) os escravos multiplicam o vetor e transmitem os dados processados novamente a tarefa mestre que conclui a execução da aplicação.**

O processo de criação e adaptação dos tráfegos é descrito a seguir. O ambiente de geração de tráfego sintético proposto permite originalmente variar o tempo de transmissão, o tamanho do pacote e o destino do tráfego. Foi definido para o tráfego sintético, tráfegos de pacotes com o mesmo tamanho utilizado pela aplicação real, seguido dos mesmos destinos para cada tarefa. Na tarefa referente ao mestre, mapeada no endereço 44, o tráfego sintético foi adaptado manualmente de forma que o mestre envie para todos os endereços escravos os blocos do vetor a ser multiplicado. A transmissão é feita por blocos seguindo uma contagem incremental que começa pelo escravo endereço número 0 ao endereço número 63. No modelo de tráfego sintético,

todas as transmissões iniciam uma vez que comece a simulação. Também, não é possível definir um tempo de processamento para cada pacote. Para todos os tráfegos de pacotes referentes as tarefas, o tráfego sintético foi adaptado manualmente para ser transmitido a 50% da capacidade do transmissor. A transmissão a 50% da capacidade da rede evita que múltiplos tráfegos concorrentes saturam a capacidade de transmissão dos roteadores. O tempo de processamento aferido durante a execução da aplicação real foi acrescido ao tempo de transmissão de cada pacote. Desta forma, o tráfego sintético gerado irá considerar um tempo de processamento antes da transmissão de cada pacote o que aumenta o intervalo entre a geração de pacotes e aumenta a precisão do tráfego utilizado. O tráfego de aplicação CDCM foi implementado seguindo os mesmos conceitos e restrições utilizados no tráfego sintético.

A Tabela 12 descreve os resultados obtidos após realizar a simulação dos tráfegos propostos. Com relação ao desempenho aferido, a média global de todos os tráfegos demonstram que o cenário sintético apresentou 4.09 vezes menor latência e 7.47 vezes maior vazão. Uma vez que o cenário sintético não dá suporte e leva em consideração as dependências entre as transmissões e também, implementa um mecanismo que simula o tempo de processamento gasto pelos transmissores é possível detectar através deste estudo de caso que o modelo de tráfego sintético não é capaz de simular o comportamento de uma aplicação real de maneira a saturar os recursos da rede.

**Tabela 12: Valores de vazão e latência média dos cenários de tráfego sintético e de um modelo de aplicação CDCM obtidos durante a simulação.**

Cenário de tráfego	Multiplicação Vetorial Sintética		Multiplicação Vetorial CDCM	
	Latência Média (ns)	Vazão Média (Mbps)	Latência Média (ns)	Vazão Média (Mbps)
Mestre aos Escravos	210911	21.70	750273	2.78
Escravos ao Mestre	157839	21.48	759186	3.01
Média Global	184375	21.59	754729	2.89

Além do desempenho, a área o consumo de energia dos cenários de rede e seus respectivos tráfegos utilizados foram avaliados. A Tabela 13 ilustra a área consumida, sendo ela a mesma para ambos os cenários de rede uma vez que são utilizadas redes iguais para ambos os cenários avaliados. Com relação ao consumo de energia a mesma metodologia utilizada para obtenção do consumo de energia do estudo de caso anterior foi adotada para este estudo de caso. Na Tabela 13 verifica-se que com relação ao item (i) que trata do consumo de energia do sinal de relógio, ambos os cenários de tráfego possuem os mesmos resultados uma vez que a árvore de relógios é a mesma para ambos os cenários de rede utilizados. Com relação ao item (ii) que apresenta o consumo de energia da lógica combinacional, o cenário de tráfego sintético apresentou 7.04 vezes maior consumo de energia que o cenário de tráfego CDCM. Já para o item (iii) que apresenta o consumo de energia da lógica sequencial, o cenário de tráfego CDCM apresentou 1.45% maior consumo de energia que o cenário de tráfego sintético.

Os resultados apontam que o tráfego sintético consumiu maior quantidade de energia e também, reportou melhores resultados quanto a tempo de execução e vazão atingida. De maneira contrária, o tráfego CDCM reportou resultados inferiores quanto ao consumo de energia porém, demonstrou conseguir maior saturação da vazão média e da latência média dos pacotes transmitidos. Conclui-se que o tráfego CDCM dá suporte a características importantes da aplicação como o tempo de processamento e a dependência entre tarefas sendo estas características importantes e que devem ser consideradas uma vez que permitem melhor saturar a capacidade da rede e ainda, gerar diferentes resultados para o consumo de energia da rede quando comparado ao uso de tráfegos sintéticos.

**Tabela 13: Valores de área em número de LUTs de quatro entradas e consumo de energia. O consumo de energia (em nanoJoule/s) é dado em: (i) Consumo de energia dos sinais de relógio; (ii) Lógica combinacional do circuito; (iii) Lógica sequencial do circuito.**

Cenário de tráfego	Área em número de LUTs de quatro entradas	Consumo de Energia (Clocks)	Consumo de Energia (Combinacional)	Consumo de Energia (Sequencial)
Multiplicação Vetorial Sintética	16980	0.8167 nJ/s	$9.820 \times 10^{-4}$ nJ/s	0.1903 nJ/s
Multiplicação Vetorial CDCM	16980	0.8167 nJ/s	$1.394 \times 10^{-4}$ nJ/s	0.1931 nJ/s



## 7 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho propôs e desenvolveu um ambiente de geração e avaliação de redes intrachip não síncronas integrado ao ambiente ATLAS. O ambiente de geração de redes foi desenvolvido para permitir a geração parametrizável de instâncias da rede HERMES-G. O ambiente de geração e avaliação de tráfego foi desenvolvido a partir do modelo de tráfego sintético proposto por Tedesco [TED05] e de um dos modelos de tráfego de aplicações proposto por Marcon[MAR05].

O estado da arte carece de referências a trabalhos com a proposta de redes intrachip não síncronas em nível RTL, descritas em linguagem VHDL que de alguma forma permitam a parametrização das características da rede. Este trabalho, além de disponibilizar um projeto de rede não síncrona passível de parametrização de suas características, disponibiliza um ambiente gráfico para geração de redes intrachip não síncronas de maneira intuitiva. Ainda, com relação ao estado da arte, no tema relacionado a caracterização de tráfego, diversos trabalhos pesquisados concluem que a maneira como o tráfego é gerado e mapeado tem relação direta com o desempenho da rede, e ainda, que o tráfego sintético utilizado por grande parte dos trabalhos pesquisados está distante de conseguir representar o comportamento de aplicações de forma precisa. Durante a investigação do estado da arte, diversas propostas que promovem a evolução de modelos de tráfego sintético foram encontradas. Neste trabalho, o autor além de propor novas características ao modelo de tráfego sintético do ambiente ATLAS, através da distribuição temporal exponencial decrescente, propõe também gerar tráfego a partir de modelos de aplicações. Além da proposta dos modelos de tráfego, este trabalho disponibiliza ainda um ambiente gráfico de configuração e geração de tráfego parametrizável para redes não síncronas.

Por fim, os trabalhos do estado da arte no tema de avaliação de tráfego para redes detalham o uso de métricas de vazão e latência de pacotes. Na grande maioria dos trabalhos, a latência é medida em ciclos, e a vazão medida através de tráfego aceito. Neste trabalho, uma vez que uma rede opera com diferentes frequências, a latência, ou tempo de transmissão do pacote deve ser avaliada em tempo absoluto e não em ciclos. Uma vez que se possa conhecer o caminho do tráfego e as frequências dos roteadores, é possível mensurar o tempo gasto por um pacote com precisão. Optou-se aqui por avaliar a vazão em quantidade de informação transmitida (Mbps), e não em tráfego aceito, uma vez que em uma rede não síncrona, a taxa de injeção de um pacote pode variar dependendo a frequência de operação da rede utilizada.

### 7.1 CONTRIBUIÇÕES DO TRABALHO

Este trabalho contribui com um ambiente de geração de redes intrachip não síncronas em nível VHDL passíveis de parametrização através de uma interface gráfica intuitiva. Em nenhum trabalho revisado é apresentado uma proposta semelhante.

Com relação à geração de tráfego, este trabalho promove a evolução do gerador de tráfego proposto por Tedesco [TED05] nas questões relacionadas ao modelo de tráfego sintético,

introduzindo a distribuição exponencial decrescente e propondo algumas correções nos cálculos e nas interfaces do gerador. Além disso, introduz-se um modelo de geração de tráfego que faz uso de modelos de aplicações. A utilização deste novo modelo traz ao grupo de pesquisa do autor uma série de novas possibilidades de uso.

Com relação à avaliação do tráfego, este trabalho propôs uma evolução da ferramenta de avaliação de tráfego de Tedesco [TED05], adaptando as técnicas e as métricas utilizadas para avaliação do tráfego gerado para redes não síncronas. Durante a revisão do estado da arte, o autor não encontrou nenhum trabalho propondo uma ferramenta de avaliação de tráfego para redes não síncronas. Em todos aqueles que de alguma forma demonstraram resultados para redes não síncronas, nenhum apresentou uma proposta similar à proposta por este trabalho para avaliar o tráfego.

Por fim, o autor ainda cita como contribuição os esforços desenvolvidos e pouco documentados neste trabalho relacionados à integração e verificação das ferramentas propostas ao fluxo de desenvolvimento de redes do ambiente ATLAS. Por este ambiente ter ampla aceitação da comunidade acadêmica no escopo de geração e avaliação de redes intrachip, ele poderá ser utilizado no futuro por outros trabalhos. Na visão do autor, a maior contribuição deste trabalho é reunir outros trabalhos, utilizando os ambientes descritos em [OST05] e [TED05], e a rede intrachip não síncrona proposta por Pontes et al. em [PON08] em algo maior, criando novos tópicos de pesquisa a serem explorados pelo grupo de pesquisa do autor.

## 7.2 TRABALHOS FUTUROS

A partir da proposta descrita por este trabalho, é possível definir um conjunto de trabalhos futuros a realizar:

- ❖ Desenvolvimento de um componente de avaliação interna, responsável por coletar o tráfego entre os canais que interligam os roteadores. Este componente deve levar em consideração que cada canal irá transmitir informação em uma determinada frequência, e que as características da rede por serem parametrizáveis irão influenciar na geração da sua estrutura.
- ❖ Adaptação do ambiente de avaliação de tráfego, para levar em consideração os valores coletados pelo componente de avaliação interna conforme proposto por Tedesco [TED05]. As métricas utilizadas para avaliação do tráfego de maneira interna em redes síncronas levam em consideração o ciclo de relógio para realizar os cálculos de vazão e latência dos canais, devendo ser adaptadas para tempo absoluto, levando em consideração a frequência de operação dos roteadores.
- ❖ Evolução das técnicas de mapeamento propostas por Marcon [MAR05] e utilizadas no ambiente CAFES, para considerar a frequência de operação dos roteadores e um tempo de transmissão máximo de cada uma das mensagens das tarefas da aplicação. A modificação implica que o ambiente CAFES reconheça a frequência dos roteadores para realizar o mapeamento. Sugere-se que o arquivo de projeto da rede seja utilizado como entrada pelo

ambiente CAFES durante o mapeamento, uma vez que contém os valores das frequências dos roteadores definidas no projeto da rede.

- ❖ Evoluir o modelo de aplicações CDCM, permitindo que durante a geração de um modelo seja possível definir um número de pacotes para uma mensagem em *flits*. Além disso, permitir que os pacotes de uma mensagem sejam transmitidos na rede conforme uma distribuição temporal a partir das distribuições suportadas pelo ambiente de geração de tráfego do ambiente ATLAS, uma vez que hoje uma mensagem é transmitida a taxa máxima de injeção do transmissor.



## REREFÊNCIAS BIBLIOGRÁFICAS

- [AMD05] Amde, M.; Felicijan, T.; Efthymiou, A.; Edwards, D.; Lavagno, L. "Asynchronous on-chip networks". IEEE Proceedings - Computers and Digital Techniques, 152(2), March 2005, pp. 273-283.
- [ARD10] Arden, W.; Brillouët, M.; Cogez, P.; Graef, M.; Huizing, B.; Mahnkopf, R. "More-than-Moore" White Paper". Capturado em: <http://www.itrs.net/Links/2010ITRS/IRC-ITRS-MtM-v2%203.pdf>, June 2012.
- [BON07] Bononi, L.; Concer, N.; Grammatikakis, M.; Coppola, M.; Locatelli, R. "NoC Topologies Exploration based on Mapping and Simulation Models". In: 10th Euromicro Conference on Digital System Design Architectures (DSD'07), 2007, pp. 543-546.
- [BRU09] Bruch, J.; Pizzoni, M.; Zeferino, C. "BrownPepper: a SystemC-based Simulator for Performance Evaluation of Networks-on-Chip". In: 17th Annual International Conference on Very Large Scale Integration (VLSI-SoC'09), 2009, pp. 223-226.
- [CHA10] Chang, C.; Hsu, Y. "A System Exploration Platform for Network-on-Chip". In: International Symposium on Parallel and Distributed Processing with Applications (ISPA'10), 2010, pp. 359-366.
- [FEN07] Fen, G.; Ning, W.; Qi, W. "Simulation and Performance Evaluation for Network-on-Chip Design Using OPNET". In: Taipei International Convention Center (TENCON'07), 2007, pp. 1-4.
- [GRA06] Gratz, P.; Kim, C.; McDonald, R.; Keckler, S.; Burger, D. "Implementation and Evaluation of On-Chip Network Architectures". In: 24th International Conference on Computer Design (ICCD'06), 2006, pp. 477-484.
- [HEC11] Heck, G. "Um MPSoC GALS Baseado em Redes Intrachip com Geração Local de Relógio". Seminário de andamento de Mestrado, PPGCC-FACIN-PUCRS. January 2011. 20p.
- [HON08] Hong, P.; Pham, P.; Tran, X.; Kim, C. "Analysis and Evaluation of Traffic-Performance in a Backtracked Routing Network-on-Chip". In: 2<sup>nd</sup> International Conference on Communications and Electronics (ICCE'08), 2008, pp. 13-17.
- [INT11] INTEL INC. "Teraflops Research Chip: Project". Capturado em: <http://www.intel.com/content/www/us/en/research/intel-labs-teraflops-research-chip.html>, June 2012.
- [INT12] INTEL FOUNDRIES INC. "Introducing the World's First 3-D Transistor Ready for High-Volume Manufacturing". Capturado em: <http://www.intel.com/content/www/us/en/silicon-innovations/intel-22nm-technology.html>, June 2012.
- [ITR11a] ITRS INC. "International Technology Roadmap for Semiconductors 2011 Edition Design". Capturado em: <http://www.itrs.net/Links/2011ITRS/2011Chapters/2011Design.pdf>, June 2012.
- [ITR11b] ITRS INC. "International Technology Roadmap for Semiconductors 2011 Edition Lithography". Capturado em: <http://www.itrs.net/Links/2011ITRS/2011Chapters/2011Lithography.pdf>, June 2012.

2012.

- [ITR11c] ITRS INC. "International Technology Roadmap for Semiconductors 2011 Edition Yield Enhancement". Capturado em: <http://www.itrs.net/Links/2011ITRS/2011Chapters/2011Yield.pdf>, Junho 2012.
- [GLO12] GLOBAL FOUNDRIES INC. "GLOBALFOUNDRIES Fab 8 Adds Tools To Enable 3D Chip Stacking at 20nm and Beyond". Capturado em: <http://www.globalfoundries.com/newsroom/2012/20120426.aspx>, Junho 2012.
- [JAN05] Lu, Z.; Jantsch A. "Traffic Configuration for Evaluating Networks on Chips". In: Fifth International Workshop on System-on-Chip for Real-Time Applications (IWSOC '05), 2005, pp. 535-540.
- [KRE05] Kreutz, M.; Marcon, C.; Carro, L.; Calazans, N.; Susin, A. "Energy and Latency Evaluation of NoC Topologies". In: International Symposium on Circuits and Systems (ISCAS'05), 2005, pp. 5866-5869.
- [LIU07] Liu, W.; Xu, J.; Wu, X.; Ye, Y.; Wang, X.; Zhang, W.; Nikdast, M.; Wang, Z. "A NoC Traffic Suite Based on Real Applications". In: Computer Society Annual Symposium on VLSI (ISVLSI'07), 2007, pp. 66-71.
- [LOT11] Lotlikar, S.; Pai, V.; Gratz, P.V. "AcENoCs: A Configurable HW/SW Platform for FPGA Accelerated NoC Emulation". In: 24th International Conference on VLSI Design (VLSI Design'11), 2011, pp. 147-152.
- [MAR05] Marcon, C. "Modelos para o Mapeamento de Aplicações em Infra-estruturas de Comunicação Intrachip". Tese de Doutorado, PPGCC-FACIN-PUCRS. Dezembro 2005. 192p.
- [MOO65] Moore, G. "Cramming More Components Onto Integrated Circuits". Electronics, 38(8), April 1965, pp.114-117.
- [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, 38(1), October 2004, pp. 69-93.
- [NVI12] NVIDIA INC. "NVIDIA's Next Generation CUDA Compute Architecture: Kepler GK110". Capturado em: <http://www.nvidia.com/content/PDF/kepler/NVIDIA-Kepler-GK110-Architecture-Whitepaper.pdf>, Junho 2012.
- [OST05] Ost, L.; Mello, A.; Palma, J.; Moraes, F.; Calazans, N. "MAIA: a framework for networks on chip generation and verification". In: 10th Annual Asia and South Pacific Design Automation Conference (ASP-DAC'05), 2005, pp. 49-52.
- [PAN06] Miro Panades, I.; Greiner, A.; Sheibanyrad, A. "A Low Cost Network-on-Chip with Guaranteed Service Well Suited to the GALS Approach". In: NanoNet, Sep 2006, pp.1-5.
- [PAN11] Panda, A.; Avakian, A.; Vemuri, R. "Configurable Workload Generators for Multicore Architectures". In: International SOC Conference. (SOCC'11), 2011, pp. 179-184.
- [PON08] Pontes, J.; Moreira, M.; Soares, R.; Calazans, N. "Hermes-GLP: A GALS Network on Chip Router with Power Control Techniques". In: IEEE Computer Society Annual Symposium on VLSI (ISVLSI '08), April 2008, pp. 347-352.
- [ROS12] da Rosa, T.R.; Larrea, V.; Calazans, N.; Moraes, F.G. "Power Consumption Reduction in MPSoCs

through DFS”. In: 25th Symposium on Integrated Circuits and Systems Design (SBCCI'12), 2012, pp. 1-6.

- [SCH10] Schemmer, R.; Calazans, N. “Proposta e Implementação de um Novo Modelo de Geração de Tráfego Para a Plataforma Atlas”. Introdução a Pesquisa II, PPGCC-FACIN-PUCRS. Dezembro 2010. 60p.
- [SOT06] Soteriou, V.; Wang, H.; Peh, L.-S. “A Statistical Traffic Model for On-Chip Interconnection Networks”. In: 14th IEEE International Symposium on Modeling, Analysis, and Simulation (MASCOTS '06), September 2006, pp. 104-116.
- [TED05] Tedesco, L. “Uma proposta para Geração de Tráfego e Avaliação de Desempenho para NoCs”. Dissertação de Mestrado, PPGCC-FACIN-PUCRS. Dezembro 2005. 126p.
- [TED05a] Tedesco, L.; Mello, A.; Garibotti, D.; Calazans, N.; Moraes, F. “Traffic Generation and Performance Evaluation for Mesh-based NoCs”. In: 19th Annual Symposium on Integrated Circuits and Systems design (SBCCI'05), 2005, pp. 184-189.
- [TED06] Tedesco, L.; Mello, A.; Giacomet, L.; Calazans, N.; Moraes, F. “Application driven traffic modeling for NoCs”. In: 19th Annual Symposium on Integrated Circuits and Systems design (SBCCI'06), 2006, pp. 62-67.
- [ZAY12] Zaytoun, A.H.M.; Fahmy, A.H.; Elsayed, K.M.F. “Implementation and Evaluation of Large Interconnection Routers for Future Many-Core Networks on Chip”. In: 14th International Conference on High Performance Computing and Communication. (HPCC-ICISS'12), 2012, pp. 524-531.