MARTHA JOHANNA SEPÚLVEDA FLÓREZ

**ESTIMATIVA DE DESEMPENHO DE UMA *NoC* A PARTIR DE
SEU MODELO EM SYSTEMC – TLM**

Dissertação apresentada à Escola
Politécnica da Universidade de São Paulo
para obtenção do Título de Mestre em
Engenharia Elétrica.

São Paulo
2006

MARTHA JOHANNA SEPÚLVEDA FLÓREZ

**ESTIMATIVA DE DESEMPENHO DE UMA *NoC* A PARTIR DE
SEU MODELO EM SYSTEMC – TLM**

Dissertação apresentada à Escola
Politécnica da Universidade de São Paulo
para obtenção do Título de Mestre em
Engenharia Elétrica.

Área de Concentração:
Microeletrônica

Orientador:
Prof. Dr. Marius Strum

São Paulo
2006

FICHA CATALOGRÁFICA

Sepúlveda Flórez, Martha Johanna

Estimativa de desempenho de uma NoC a partir de seu modelo em SYSTEMC – TLM / M.J. Sepúlveda Flórez. -- São Paulo, 2006.

167 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Sistemas Eletrônicos.

1.System-on-Chip (SoC) 2.Network-on-Chip 3.Metodologia de projeto 4.TLM-SystemC I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Sistemas Eletrônicos II.t.

Dedico esta tesis a mis padres Martha Lucía y Juan de J. por su ejemplo de fortaleza, perseverancia y dedicación.

A mis hermanos Juan Carlos y Daniel Mauricio por creer siempre en mí, darme fuerza y brindarme su amistad y amor.

AGRADECIMENTOS

Quero apresentar meus mais sinceros agradecimentos às pessoas e entidades que tornaram possível a realização deste trabalho. Trabalho que marcou uma etapa importante em minha vida pessoal e profissional.

Inicialmente quero agradecer a minha família, meus pais Martha Lucía e Juan de J. e irmãos Juan Carlos e Daniel Mauricio, pelo apoio sempre incondicional, por serem a razão de meu esforço e luta e pelo constante estímulo nesta caminhada.

Ao meu orientador, Prof. Dr. Marius Strum, agradeço a atenção dispensada no decorrer deste trabalho. Agradeço sua confiança e tempo investido em minha orientação. Sou também grata pelo aprendizado construído ao longo destes anos.

Aos professores Ivan e John, por acreditarem em meu potencial desde a graduação e pela confiança e estímulo. Obrigada pela sua parceria e principalmente pela sua amizade.

Ao Grupo GAPH, do departamento de Engenharia Elétrica da Pontifícia Universidade Católica do Rio Grande do Sul (UFRGS), pela gentileza em disponibilizar o modelo da rede *Hermes* e pelo suporte técnico.

Aos integrantes do Grupo GSEIS, especialmente ao Prof. Dr. Wang pelo auxílio durante minha formação acadêmica e ao colega Gino, pessoa que tenho enorme carinho, respeito e admiração, pela sincera amizade, disponibilidade e atenção.

Ao Daniel, pelo constante apoio e carinho, tornado-se meu confidente e parceiro de tantas alegrias e tristezas. Por nunca me deixar cair ao longo desta jornada, oferecendo-me um motivo a mais para continuar.

À Lucia, Edecio, Paula e Luisa, por terem me acolhido. Obrigada pelos momentos de alegria, pelo cuidado e carinho.

Ao PNM – CNPQ pelo apoio financeiro para a realização deste trabalho.

RESUMO

A existência de uma grande variedade de estruturas de interconexão para sistemas-sobre-silício (*System-on-Chip*, *SoC*), conexões ponto-a-ponto, barramentos e redes intrachip, cada qual com uma diversa gama de parâmetros de configuração, fornece aos projetistas amplas alternativas de projeto. Apesar da estrutura de interconexão ser um dos componentes críticos de um *SoC*, existem poucas ferramentas que permitem definir a estrutura mais adequada para uma determinada aplicação.

Um projeto eficiente de um *SoC* deve realizar uma exploração prematura do espaço de soluções da estrutura de comunicação, já durante as primeiras etapas do fluxo de projeto. Em razão da ausência de ferramentas para essa exploração, torna-se fundamental o juízo do projetista na avaliação dos compromissos envolvidos.

O presente trabalho procura contribuir com o projeto da estrutura de comunicação de um *SoC* implementada através de uma rede *intrachip* (*Network-on-Chip*, *NoC*).

Neste trabalho é proposta uma metodologia que permite definir a configuração de uma rede *intrachip* a partir de seu modelo de alto nível (descrição *SystemC* TLM temporizado).

A metodologia está baseada na análise e avaliação do desempenho da rede *intrachip* diante de uma grande variedade de condições de tráfego e para diferentes configurações da rede. O estudo experimental foi desenvolvido para uma rede representada pelo seu modelo em *SystemC* TLM temporizado (*Hermes_temp*). O tráfego é controlado por geradores paramétricos e pseudo-aleatórios, também descritos pelos seus modelos em *SystemC* - TLM. A análise foi feita através do auxílio de uma ferramenta desenvolvida para esta finalidade, a qual fornece os valores de um conjunto de métricas de desempenho. Foram também desenvolvidas técnicas e métodos para realizar a avaliação do desempenho. Os resultados obtidos permitiram elucidar o comportamento global e interno da rede. Os valores de desempenho são utilizados no projeto de redes *intrachip* homogêneas e heterogêneas nas primeiras etapas do projeto do *SoC*, ampliando assim os resultados até agora obtidos em outros trabalhos científicos que realizam a avaliação de desempenho das redes *intrachip*.

ABSTRACT

The wide variety of interconnection structures presently nowadays for SoC (System-on-Chip), bus and networks-on-Chip NoCs, each of them with a wide set of setup parameters, provides a huge amount of design alternatives. Although the interconnection structure is a key SoC component, there are few design tools in order to set the appropriate configuration parameters for a given application.

An efficient SoC project may comply an exploration stage among the possible solutions for the communication structure, during the first steps of the design process. The absence of appropriate tools for that exploration makes critical the designer's judgment.

The present study aims to enhance the communication SoC structure design area, when a NoC is used.

This work proposes a methodology that allows the establishment of the NoC communication parameters using a high level model (SystemC TLM timed).

Our approach analyzes and evaluates the NoC performance under a wide variety of traffic conditions. The experimental stage was conducted employing a model of a net represented by a SystemC TLM timed (Hermes_Temp). Parametric and pseudo-random generators control the network traffic. The analysis was carried on with a tool designed for these purpose, which generates a group of performance metrics.

The results allow to elucidate the global and inner network behavior. The performance values are useful for the heterogeneous and homogeneous NoC design projects, improving the performance evaluation studies scope.

SUMARIO

Capítulo 1 INTRODUÇÃO

1.1	CONTEXTUALIZAÇÃO DO TRABALHO	14
1.2	OBJETIVO	20
1.3	ORGANIZAÇÃO DO TEXTO	21

Capítulo 2 ESTADO DA ARTE

2.1	PROJETO DE <i>SOC</i>	24
2.2	AVALIAÇÃO DAS REDES INTRACHIP	26
2.2.1	Avaliação de custo das redes <i>intrachip</i>	27
2.2.2	Avaliação de desempenho das redes <i>intrachip</i>	29
2.3	TRABALHO DESENVOLVIDO PELO GRUPO DE PESQUISA <i>GSEIS</i> ...	31
2.4	CONSIDERAÇÕES	33

Capítulo 3 PROJETO DE *SoCs* (*System-on-Chip*)

3.1	FLUXO DE PROJETO DE UM <i>SoC</i>	34
3.2	DESCRIÇÃO DO SISTEMA.....	36
3.3	DEFINIÇÃO DAS PLATAFORMAS DO SISTEMA.....	37
3.4	INTEGRAÇÃO DOS COMPONENTES <i>HW/SW</i>	42
3.5	CONSIDERAÇÕES	43

Capítulo 4 REDES *INTRACHIP* (*Network-on-Chip*)

4.1	CONCEITOS BÁSICOS	49
4.1.1	Parâmetros da Rede	46
4.1.2	Roteadores	48
4.1.3	Enlaces	49
4.2	CARACTERÍSTICAS DE UMA <i>NoC</i>	49
4.2.1	Topologia da Rede	49
4.2.2	Estratégia de Roteamento	51
4.2.3	Controle de Fluxo	54
4.3	PROJETO DAS <i>NoCs</i>	56
4.4	CONSIDERAÇÕES	59

Capítulo 5 MODELAGEM AO NÍVEL DE TRANSAÇÕES

5.1	INTRODUÇÃO	60
5.2	DESCRIÇÃO DO TLM.....	61
5.2.1	Transação	61
5.2.2	Sub-níveis TLM	64
5.3	SYSTEMC	66
5.4	CONSIDERAÇÕES	67

Capítulo 6 DESCRIÇÃO DO SISTEMA EXPERIMENTAL

6.1 SISTEMA PARA A AVALIAÇÃO DE DESEMPENHO	70
6.1.1 Geradores de tráfego	71
6.1.2 Blocos escravos.....	75
6.2 REDE INTRACHIP	76
6.2.1 Rede <i>HERMES</i>	76
6.2.2 Rede <i>HERMES_TEMP</i>	81
6.3 REQUISITOS DE COMUNICAÇÃO.....	92
6.4 MÉTRICAS DE DESEMPENHO	92
6.4.1 Classificação das métricas	93
6.4.2 Métricas adotadas.....	93
6.5 DEFINIÇÃO DO CONJUNTO DE SIMULAÇÕES	96
6.6 QUANTIFICAÇÃO DAS MÉTRICAS	98
6.7 VALIDAÇÃO.....	102
6.8 CONSIDERAÇÕES	105

Capítulo 7 AVALIAÇÃO DE DESEMPENHO

7.1 PADRÕES DE QUALIDADE	107
7.2 PADRÃO TRANSPOSTO	120
7.2.1 Padrões de Qualidade para o Padrão Transposto	121
7.2.2 Resultados do Padrão Transposto	124
7.2.3 Discussão	132
7.3 PADRÃO <i>HOT SPOT</i>	133
7.3.1 Padrões de Qualidade para o Padrão <i>Hot Spot</i>	135
7.3.2 Resultados do Padrão <i>Hot Spot</i>	138
7.4 PADRÃO PSEUDO-ALEATÓRIO	144
7.4.1 Resultados do <i>Pseudo-Aleatório</i>	148
7.5 RESUMO	153
7.6 EXEMPLO DE ESCOLHA DE CONFIGURAÇÃO.....	154

Capítulo 8 CONCLUSÕES

ÍNDICE DE FIGURAS

Figura 1.1. Visão geral resumida do fluxo do projeto de um <i>SoC</i>	16
Figura 2.1 Roteadores implementados por [RIJ03].....	30
Figura 2.2 Sistema implementado por [WOL04]	30
Figura 3.1 Fluxo de projeto de um <i>SoC</i>	35
Figura 3.2 Componentes do <i>SoC</i>	37
Figura 3.3 Projeto da plataforma de comunicação.....	39
Figura 3.4 Fluxo de projeto da plataforma de comunicação incluindo a validação e a avaliação de desempenho.....	41
Figura 4.1 Rede em anel.	45
Figura 4.2 Configuração geral de um (a) terminal ; (b) roteador.....	45
Figura 4.3 Topologias da rede direta ortogonal: (a) Rede grelha; (b) Rede toróide; (c) Rede hipercubo.	50
Figura 4.4 Hierarquia da divisão das mensagens.....	52
Figura 4.5 Fluxo de projeto da <i>NoC</i>	57
Figura 5.1 Níveis de modelagem do sistema	64
Figura 5.2 Exemplo de comunicação de um sistema.....	65
Figura 5.3 Representação gráfica das ferramentas de modelagem	67
Figura 6.1 Projeto da plataforma de comunicação.....	70
Figura 6.2 Sistema empregado para a avaliação do desempenho.....	71
Figura 6.3 Representação do tempo de geração de pacotes.....	72
Figura 6.4 Representação da largura dos pacotes.	73
Figura 6.5 Destino do pacote.	74
Figura 6.6 Três combinações dos parâmetros do gerador de tráfego: (a) largura do pacote e destino do pacote constante e tempo de geração variável; (b) largura do pacote e tempo de geração constante e o destino do pacote variável; (c) largura do pacote variável e tempo de geração e destino do pacote constante	75
Figura 6.7 Camadas do modelo OSI.	76
Figura 6.8 Interconexão entre roteadores.....	77
Figura 6.9 Topologia de uma rede <i>Hermes</i> 3x3.....	78
Figura 6.10 Roteador da <i>NoC Hermes</i>	79
Figura 6.11 Pacote transmitido pela rede.....	79
Figura 6.12 Processos da rede envolvidos na comutação de um pacote.....	82
Figura 6.13 Comutação da informação na rede.	85
Figura 6.14 Intervalos de tempo de uma comutação.	85
Figura 6.15 Roteamento de informação na rede.	88
Figura 6.16 Transação composta de três comutações.	88
Figura 6.17 Diagrama de tempos associadas à comunicação dentro da <i>NoC</i>	89
Figura 6.18 Um dos portos do roteador da rede <i>Hermes_Temp</i>	90
Figura 6.19 Fluxo de captura e análise da avaliação de desempenho.....	99
Figura 6.20 Arquivos gerados nas diferentes etapas da simulação e análise da rede	100
Figura 6.21 Avaliação na rede <i>intrachip</i>	101

Figura 6.22 Padrão de tráfego complemento.....	103
Figura 6.23 Resultado da simulação de <i>Hermes</i> apresentado em [TED05].	104
Figura 6.24 Resultados da simulação do modelo <i>Hermes_Temp</i>	104
Figura 7.1 Comutação de um pacote de 5 <i>flits</i> de carga útil	109
Figura 7.2 Modelo da comunicação para completar uma transação (até sete comutações)	109
Figura 7.3 Ciclos envolvidos na comutação	114
Figura 7.4 Aumento da latência de arbitragem na comutação de um pacote	114
Figura 7.5 Intervalos de qualidade para TE	115
Figura 7.6 Intervalos de qualidade definidos para NU	117
Figura 7.7 Padrão de tráfego transposto sobre (a) topologia <i>Grelha</i> , (b) topologia <i>Toróide</i> , (c) topologia <i>Anel Cordal</i> e (d) topologia <i>Circular Entrelaçada</i> ...	121
Figura 7.8 Duração média das transações.....	126
Figura 7.9 Vazão das transações.....	127
Figura 7.10 Tempo de espera médio (TE) por roteador para as configurações com topologia <i>Grelha</i>	129
Figura 7.11 Tempo de espera médio (TE) por roteador para topologia <i>Toróide</i>	130
Figura 7.12 Tempo de espera médio (TE) por roteador para a topologia <i>Anel Cordal</i> ..	130
Figura 7.13 Tempo de espera médio (TE) por roteador para a topologia <i>Circular</i> <i>Entrelaçada</i>	131
Figura 7.14 Configuração C1 modificada.....	133
Figura 7.15 Padrão de tráfego <i>hot spot</i> sobre (a) topologia <i>Grelha</i> , (b) topologia <i>Toróide</i> , (c) topologia <i>Anel Cordal</i> e (d) topologia <i>Circular Entrelaçada</i>	134
Figura 7.16 Duração média das transações.....	139
Figura 7.17 Vazão das transações.....	140
Figura 7.18 Tempo de espera médio (TE) para os roteadores com topologia <i>Grelha</i>	141
Figura 7.19 Tempo de espera médio (TE) para os roteadores com topologia <i>Toróide</i> ..	142
Figura 7.20 Tempo de espera médio (TE) para os roteadores com topologia <i>Anel Cordal</i>	142
Figura 7.21 Tempo de espera médio (TE) para os roteadores com topologia <i>anel</i> <i>entrelaçada</i>	143
Figura 7.22 Padrão de tráfego pseudo-aleatório sobre (a) topologia <i>Grelha</i> e (b) topologia <i>Toróide</i> , (c) topologia <i>Anel Cordal</i> e (d) topologia <i>Circular Entrelaçada</i> ...	145
Figura 7.23 Duração média das transações.....	149
Figura 7.24 Vazão das transações.....	149
Figura 7.25 Tempo de espera médio (TE) para os roteadores com topologia <i>Grelha</i>	150
Figura 7.26 Tempo de espera médio (TE) para os roteadores com topologia <i>Toróide</i> ..	151
Figura 7.27 Tempo de espera médio (TE) para os roteadores com topologia <i>Anel Cordal</i>	151
Figura 7.28 Tempo de espera médio (TE) para os roteadores com topologia <i>Circular</i> <i>Entrelaçada</i>	152
Figura 81 Topologia <i>Circular Entrelaçada</i> desde (a) representada em dois dimensões e (b) na sua construção Tridimensional	166

ÍNDICE DE TABELA

Tabela 6.1 Quadro resumo das características da rede <i>Hermes</i>	80
Tabela 6.2 Conjunto de simulações realizadas	98
Tabela 6.3 Informações fornecidas por <i>Apolo</i>	102
Tabela 6.4 Tempo médio gasto na simulação e filtragem	102
Tabela 7.1 Número mínimo de ciclos segundo o número de roteadores percorridos.....	109
Tabela 7.2 Intervalo de qualidade da Duração Média das transações (DT)	110
Tabela 7.3 Intervalo de qualidade da vazão.....	111
Tabela 7.4 Intervalo de qualidade da largura de banda	113
Tabela 7.5 Intervalos de TE definidos para os quatro tamanhos de pacote	116
Tabela 7.6 Intervalo de qualidade do nível de utilização.....	117
Tabela 7.7 Intervalo de qualidade da dispersão	118
Tabela 7.8 Duração ótima das transações	122
Tabela 7.9 Valores ótimos da configuração transposta	123
Tabela 7.10 Características do conjunto de configurações da rede <i>Hermes_Temp</i>	124
Tabela 7.11 Duração média das dez configurações da rede <i>Hermes_Temp</i>	125
Tabela 7.12 Desvio médio do nível de utilização dos roteadores.....	131
Tabela 7.13 Duração ótima das transações	136
Tabela 7.14 Intervalos de qualidade para o padrão <i>hot spot</i>	137
Tabela 7.15 Duração média para o padrão <i>hot spot</i>	138
Tabela 7.16 Duração média das transações das dez configurações para diferentes localizações do <i>hot spot</i>	140
Tabela 7.17 Desvio médio do nível de utilização dos roteadores.....	143
Tabela 7.18 Duração ótima das transações	146
Tabela 7.19 Valores ótimos da configuração aleatoria.....	147
Tabela 7.20 Duração média para o padrão pseudo-aleatório.....	148
Tabela 7.21 Desvio médio do nível de utilização dos roteadores.....	152
Tabela 7.22 Melhores e piores configurações.....	153
Tabela 7.23 Resumo dos resultados das dez configurações que implementam os três padrões de tráfego para os cenários 25 e 100 ciclos	156
Tabela 7.24 Resumo dos resultados das dez configurações que implementam os três padrões de tráfego para os cenários 150 ciclos e exponencial.....	157
Tabela 7.25 Subconjunto de soluções gerado a partir da análise da primeira condição nos cenários 25 e 100 ciclos	158
Tabela 7.26 Subconjunto de soluções gerado a partir da análise da primeira condição nos cenários 25 e 100 ciclos	159
Tabela 7.27 Subconjunto de soluções gerado a partir da análise da segunda condição nos cenários 25 e 100 ciclos	160
Tabela 7.28 Subconjunto de soluções gerado a partir da análise da segunda condição nos cenários 150 ciclos e exponencial.....	161
Tabela 7.29 Subconjunto de soluções gerado a partir da análise da terceira condição nos cenários 25 e 100 ciclos	162
Tabela 7.30 Subconjunto de soluções gerado a partir da análise da terceira condição nos cenários 150 ciclos e exponencial.....	163

LISTA DE ABREVIATURAS E SIGLAS

CAD	<i>Computer aided design</i>
CI	<i>Circuito Integrado</i>
DSP	<i>Digital signal processor</i>
EC	<i>Estrutura de comunicação</i>
EDA	<i>Electronic Design Automation</i>
EP	<i>Elemento de processamento</i>
FCFS	<i>First-come-first-served</i>
FIFO	<i>First-in-First-out</i>
FLIT	<i>Flow control digit</i>
HOL	<i>Head-of-Line blocking</i>
IP	<i>Intellectual Property</i>
ISO	<i>International Organization for Standardization</i>
ITRS	<i>International Technology Roadmap for Semiconductors</i>
LRS	<i>Least recently served</i>
MWD	<i>Multi-Window Displayer</i>
NI	<i>Network Interface</i>
NoC	<i>Network-on-Chip</i>
OSI	<i>Open System Interconnection</i>
PE	<i>Processing Element</i>
RR	<i>Round-Robin</i>
RTL	<i>Register Transfer Level</i>
SoC	<i>System-on-Chip</i>
TLM	<i>Transaction Level Model</i>
VOPD	<i>Video Object Plane Decoder</i>

Capítulo 1

INTRODUÇÃO

Neste capítulo vamos apresentar e contextualizar o trabalho sobre estimativas de desempenho de redes intrachip a partir de modelos de alto nível de abstração desenvolvido neste mestrado. Em seguida uma explicação dos objetivos do trabalho é realizada. O capítulo está dividido em três partes. A primeira parte descreve a importância da plataforma de comunicação no projeto de um sistema digital sobre silício. A segunda parte apresenta os objetivos do trabalho. A terceira parte descreve a organização do texto.

1.1 CONTEXTUALIZAÇÃO DO TRABALHO

O contínuo avanço das tecnologias de fabricação de dispositivos semicondutores e das técnicas de projeto de sistemas integrados, assim como a exigência das indústrias por circuitos com especificações cada vez mais agressivas, caracterizadas por maior funcionalidade, maior frequência de operação e menor dissipação de potência, vêm promovendo um contínuo incremento nos níveis de integração dos modernos circuitos integrados (CIs) e uma diminuição do seu ciclo de vida. Nestas condições, os projetistas são atualmente capazes de desenvolver sistemas eletrônicos embutidos (do inglês *embedded systems*) na forma de um único circuito integrado. **Sistemas embutidos** referem-se a sistemas eletrônicos desenvolvidos para desempenhar um conjunto de funções específicas de um equipamento ou instrumento eletrônico. Sistemas embutidos são desenvolvidos na forma de **sistemas computacionais** personalizados. Um sistema computacional é um sistema *hardware/software* que inclui, por exemplo, um sistema operacional e *drivers* (componentes *software*) e um ou mais microprocessadores¹, sistema de memórias, co-processadores que atuam como aceleradores de *hardware* para funções tais como MPEG, PM3, criptografia, e controladores de E/S de diversos tipos Ethernet,

¹ Podem ser microcontroladores ou DSPs.

USB, Bluetooth controladores (componentes *hardware*). Quando tais sistemas computacionais são implementados na forma de um único circuito integrado, isto é, em um único *chip*, são chamados de sistemas-sobre-silício conhecidos pela sigla *SoC* (do inglês *System-on-Chip*).

Devido a sua complexidade, o projeto de um *SoC* depende criticamente do conceito do “reaproveitamento de componentes” (do inglês *componente reuse*). Tais componentes são disponibilizados na forma de módulos virtuais pré-projetados e pré-testados, conhecidos pelo termo **núcleos de hardware**² ou ainda **módulos IP** (sigla do inglês *Intellectual Property*).

Módulos IP podem ser disponibilizados em três diferentes formas: (i) Módulos IP *soft* é uma descrição comportamental executável e sintetizável independente de tecnologia, (ii) Módulos IP *firm* é uma descrição na forma de um esquema composto de células de uma determinada biblioteca e, portanto já comprometido tecnologicamente ou (iii) Módulos IP *hard* descreve o leiaute e, portanto comprometido com uma tecnologia de fabricação.

Projetar um *SoC* é, portanto, freqüentemente designado pelo termo “integrar um sistema”, pois o objetivo do projetista consiste em gerar uma arquitetura (arquitetura significando um esquema composto exclusivamente ou predominantemente de módulos IP) reutilizando módulos IP disponíveis no mercado³. Neste sentido *SoC* são projetados baseados segundo o conceito “*view points meeting*”, conceito que combina duas estratégias de projeto hierárquico: *Top-down* e *Bottom-up* como ilustra a Figura 1.1. O projeto *Top-down* consiste na transformação, mediante uma série de refinamentos sucessivos, de uma descrição abstrata do comportamento do circuito (subsistema ou sistema completo) em uma arquitetura composta de módulos IP (*software e hardware*) de forma a satisfazer os requisitos funcionais e de desempenho do sistema alvo. Por outro

² Assim como módulos de *hardware*, existem também no mercado módulos de *software*.

³ Eventualmente pode ser utilizado algum módulo IP personalizado para o sistema alvo.

lado, o projeto *Bottom-up* consiste em gerar todos os modelos de cada núcleo IP adotado, necessários para cada etapa da integração do *SoC*.

Um dos problemas a ser resolvido durante a integração de um *SoC* é a comunicação entre os módulos IP que, por poderem ter origens de diferentes fornecedores, podem, por sua vez, apresentar diferentes protocolos ou portos (sinais) incompatíveis. Isto acarreta a necessidade de serem projetadas interfaces entre os módulos IP, o que pode se tornar uma tarefa longa e ineficiente [CHA99]. Quando a comunicação, também referida neste texto como interconexão, entre os diferentes módulos IP é feita adotando-se interfaces padrão proprietárias, tais como, por exemplo, CoreConnect [COR] ou AMBA [AMB], ou ainda interfaces padrão abertas tais como OCP [OCP], ou ainda também interfaces personalizadas, o problema é atenuado [CHA99]. A adoção de uma interface padrão conduz à noção de “plataforma”⁴. Nestes casos fala-se em “projeto baseado em plataforma”.

Figura 1.1. Visão geral resumida do fluxo do projeto de um *SoC*

⁴ Adota-se o termo “projeto baseado em plataforma”.

Outra noção associada ao projeto de um *SoC* é o da ortogonalidade entre os projetos das plataformas de processamento e de comunicação, que são realizados separadamente. Uma visão geral do fluxo do projeto de um *SoC* é apresentado na Figura 1.1.

Este fluxo ilustra que o projeto parte da criação de um modelo comportamental de alto nível do sistema completo. A partir da análise deste modelo e de uma lista de requisitos, o sistema é particionado em duas plataformas: processamento (às vezes também designada por computação) e comunicação. O projeto de cada plataforma é feito separadamente. Durante o projeto da plataforma de processamento é feita, entre outras tarefas, a partição do comportamento do sistema em *hardware* e *software*. O projeto da plataforma de comunicação é influenciado pelas decisões tomadas no projeto da plataforma de processamento [GER05]. As arquiteturas virtuais das duas plataformas podem ser co-simuladas para validar o comportamento global do sistema. Finalmente vem a etapa de implementação física que pode adotar diferentes soluções: ser mapeada sobre um dispositivo reconfigurável (FPGA) ou ser fabricado na forma de um circuito integrado de aplicação específica (ASIC). Ao longo de todo o fluxo de projeto são feitas sucessivas etapas de verificação funcional a fim de validar os sucessivos modelos de níveis de abstração cada vez mais baixos que vão sendo criados no final das diversas tarefas de projeto. Isto pode ser feito comparando tais modelos com um modelo padrão chamado *golden model* [AUG05], [MIS04], [SCH04] e [ROM05].

Dentre as decisões de projeto tomadas ao longo deste fluxo está a seleção dos módulos IP para comporem as duas plataformas: processamento (processadores, memórias, módulos de aplicação específica, periféricos, módulos de E/S, etc.) e comunicação (conexões ponto-a-ponto, interfaces, barramentos hierárquicos, redes *intrachip*, árbitros, pontes, etc.). Uma outra decisão de projeto refere-se à definição dos parâmetros de configuração de cada módulo IP (fala-se em instanciar o módulo IP). Estas decisões devem ser tomadas de forma a atingir os índices de desempenho do sistema alvo, conforme especificado na lista inicial de requisitos. A verificação da aderência do resultado do projeto a tais índices de desempenho pode ser feita em três diferentes momentos do fluxo de projeto. O desempenho pode ser avaliado a partir de medidas

feitas sobre um protótipo do sistema. Isto é feito no final do projeto, garantindo a precisão da avaliação. Entretanto, esta estratégia faz com que as decisões de projeto, particularmente as que se referem às configurações dos módulos IP, sejam tomadas sem levar em conta explicitamente esta informação, o que pode elevar o tempo e, portanto o custo do projeto no caso de algum índice de desempenho não ter sido alcançado. A segunda forma é estimar (em contraste com medir) o desempenho do sistema a partir de modelos abstratos de baixo nível, por exemplo, no nível RTL (do inglês *Register Transfer Level*)⁵. Esta estimativa, feita durante o desenvolvimento do projeto, apesar de imprecisa em relação às medidas feitas sobre um protótipo, permite orientar as decisões do projetista sobre as configurações (instâncias) dos módulos IP selecionados para a arquitetura, reduzindo a possibilidade de serem necessárias modificações após a obtenção de um protótipo. A terceira forma é semelhante à segunda, consistindo em estimar o desempenho do sistema a partir de modelos abstratos, por exemplo, modelos no nível de transações (do inglês *Transaction Level Model - TLM*). Neste caso, apesar da imprecisão ser maior do que no caso anterior, devido ao nível de abstração mais alto destes modelos, o projetista antecipa decisões de projeto sobre as configurações dos módulos IP, podendo obter ganhos maiores em tempo de desenvolvimento e, portanto, custo do projeto.

No projeto da plataforma de comunicação de sistemas, cuja arquitetura de processamento requer um grande número de componentes, redes *intrachip* (do inglês *network-on-chip - NoC*) estão se tornando estruturas muito atraentes em substituição a barramentos (simples ou hierárquicos). Uma *NoC* consiste em uma rede integrada⁶ que utiliza nodos de roteamento para permitir a comunicação dos nodos de processamento (núcleos do sistema) [BAN04]. Segundo os autores de [LIN03], as *NoCs* surgiram como uma estrutura alternativa para superar as restrições dos barramentos hierárquicos devido às seguintes características: (i) eficiência de energia e confiabilidade; (ii) escalabilidade da largura de banda; (iii) decisão de roteamento distribuído. Entretanto, *NoCs* apresentam elevada complexidade e alto custo de implementação quando comparadas a barramentos hierárquicos [RIC06]. A escolha entre estas duas alternativas depende, por exemplo, do

⁵ O nível RTL é mencionado por ser atualmente considerado um padrão industrial.

⁶ Implementada em uma pastilha de silício, o que implica em restrições de área e potência.

número de componentes que se comunicam [RIC06]. Se o projeto demanda escalabilidade, uma *NoC* pode ser mais apropriada que um barramento hierárquico [MAR06]. Além disso, o grau de complexidade do sistema e fatores como custo e disponibilidade estão entre outros parâmetros que determinam a escolha de uma *NoC* ou de um barramento hierárquico [RIC06].

Inúmeros trabalhos foram realizados sobre as vantagens e desvantagens do uso de *NoCs* no projeto de *SoCs* [COP06], assim como sobre estimativas de desempenho de tais estruturas a partir de diferentes modelos e utilizando diferentes técnicas [MAR06],[MÄÄ06] e[AHO06].

Trabalhos como os de [JAL04], [NIL04] e [BER05] realizaram o estudo de desempenho do modelo da rede *intrachip* descrita em RTL (nível de transferência de registro) baseado na quantificação de duas informações: latência e vazão da rede. Estes trabalhos evidenciaram a carência de uma metodologia de avaliação de desempenho a partir de um nível abstrato da rede. O conjunto de métricas utilizadas deve permitir elucidar não somente o comportamento global da rede, como estava sendo realizado até então, como também o comportamento dos componentes que a constituem, isto é, o comportamento interno da rede. O conhecimento do valor destas métricas ajuda o projetista a tomar decisões da arquitetura da rede *intrachip* nas primeiras etapas do projeto.

Por outro lado, estes trabalhos quantificaram o conjunto de métricas de desempenho da rede *intrachip* usando as ferramentas até então existentes (*OPNET* e *NS2*). Estas ferramentas não são apropriadas para a simulação e análise de desempenho de redes *intrachip*, uma vez que a grandeza da unidade de tempo é inadequada e não consideram parâmetros como custo da implementação.

Neste trabalho, apresentamos um estudo sobre a estimativa de desempenho de uma rede *intrachip* (do inglês *Network-on-Chip*, *NoC*), realizadas a partir de seu modelo no nível de transações temporizadas descrito em *SystemC*. Neste trabalho foi desenvolvido

um conjunto de ferramentas que permite a simulação e coleta de informações do desempenho global e interno da rede. Estas informações permitem a quantificação de métricas que elucidam o comportamento da rede. O valor das métricas é utilizado no projeto de redes homogêneas e heterogêneas nas primeiras etapas do projeto, ampliando assim, os resultados até agora obtidos nos trabalhos que realizam a avaliação de desempenho das redes *intrachip*.

1.2 OBJETIVO

O objetivo geral do trabalho foi desenvolver uma metodologia para a análise de desempenho da plataforma de comunicação do projeto de um *SoC* baseada em uma *NoC*, de forma a explorar diferentes configurações da *NoC* capazes de satisfazer os requisitos de comunicação estabelecidos no início do projeto (especificação).

O presente trabalho tem por objetivos específicos:

1. Identificar os parâmetros (métricas) de desempenho de uma *NoC* capazes de serem estimados no início do fluxo de projeto a partir de seu modelo de alto nível em *SystemC* TLM temporizado.
2. Desenvolver um modelo da comunicação com o qual é possível calcular tais métricas a partir do modelo de alto nível.
3. Criar um modelo de alto nível de abstração de um *SoC* composto de elementos mestres e escravos da comunicação, representados apenas como geradores e receptores de tráfego interligados por uma rede *intrachip*. O modelo dos elementos mestre foi desenvolvido de forma a permitir simular diferentes condições de tráfego independentes da aplicação através da *NoC* (geradores de tráfego paramétricos e aleatórios). O modelo dos elementos escravo consome as informações geradas pelos elementos mestres da rede.
4. Desenvolver um modelo em *SystemC* temporizado de uma *NoC* (rede *Hermes_Temp*), a partir de um modelo em *SystemC* não temporizado existente (*Hermes_untimed*), em função de seus parâmetros de configuração.

5. Criar uma ferramenta de captura e análise dos dados gerados pelas simulações do *SoC* cuja plataforma de comunicação é a *NoC Hermes_Temp*.
6. Realizar um estudo de desempenho da *NoC Hermes_Temp*⁷ em função de seus parâmetros de configuração para diferentes condições de tráfego do *SoC*, a fim de demonstrar a validade dos modelos e técnicas desenvolvidos.

1.3 ORGANIZAÇÃO DO TEXTO

Este texto está dividido em oito capítulos.

Capítulo 2 – Estado da Arte.

O objetivo deste capítulo é descrever o desenvolvimento científico no campo da avaliação de desempenho de estruturas de comunicação para sistemas-sobre-silício (*SoC*) baseadas em redes *intrachip* (*NoC*). O capítulo fornece uma revisão geral dos principais trabalhos encontrados na literatura dos últimos anos. O capítulo está dividido em quatro partes; a primeira parte descreve a importância da estrutura de comunicação dentro do projeto de um *SoC*, a segunda parte apresenta os trabalhos desenvolvidos para a avaliação de custo e desempenho da rede *intrachip*, a terceira parte descreve o trabalho desenvolvido pelo nosso grupo de investigação⁸ e finalmente na quarta parte é realizada uma discussão da informação apresentada.

Capítulo 3 – Projeto de *SoCs* (*System-on-Chip*).

O objetivo deste capítulo é contextualizar a tarefa de estimar o desempenho de uma plataforma de comunicação dentro do fluxo do projeto de um *SoC*. O capítulo apresenta os fundamentos e principais conceitos do projeto de um *SoC* com ênfase no projeto da plataforma de comunicação e na estimativa de seu desempenho. O capítulo está dividido em cinco partes. A primeira parte apresenta as generalidades do fluxo de projeto de um *SoC*. As seguintes três seções descrevem cada uma das três etapas que compõem o fluxo de projeto: *descrição do sistema*, *definição das plataformas* (de processamento e

⁷ Modelo temporizado criado a partir de um modelo *untimed* existente (Hermes).

⁸ GSEIS do LME-EPUSP [GSE].

comunicação) e *componentes HW/SW*. Por último, na quinta parte, será realizada uma discussão das informações aqui apresentadas.

Capítulo 4 – Redes *Intrachip* (*Network-on-Chip*)

O objetivo deste capítulo é apresentar os fundamentos e principais conceitos relativos às redes de interconexão e identificar as principais características que definem uma rede de interconexão *intrachip*, que denominaremos resumidamente de redes *intrachip* (do inglês *Network-on-Chip*, *NoC*). O capítulo está dividido em quatro partes. A primeira parte apresenta os conceitos básicos das redes *intrachip*. A segunda parte lista os parâmetros de configuração de uma rede. A terceira parte apresenta algumas das aplicações encontradas na literatura que empregam uma rede *intrachip* como estrutura de comunicação. Na última parte é realizada uma discussão do capítulo.

Capítulo 5 – Modelagem ao nível de transações

O objetivo principal deste capítulo é identificar as características necessárias para realizar uma modelagem da comunicação no nível de transações. O capítulo está dividido em quatro partes. A primeira parte introduz os diferentes tipos de modelagem de um sistema. A segunda parte descreve o conjunto de subníveis que constituem a modelagem no nível de transações. A terceira parte apresenta o conjunto de ferramentas associadas a *SystemC*, uma biblioteca de C++ que permite a modelagem em TLM dos sistemas. Na quarta parte é realizada uma discussão do conteúdo do capítulo.

Capítulo 6 – Descrição do sistema experimental

O objetivo do capítulo é apresentar a metodologia adotada para realizar a estimativa de desempenho de uma rede *intrachip* (*NoC*) a partir de seu modelo em TLM (*Transaction Level Modeling*) temporizado. O capítulo está dividido em seis partes. A primeira parte apresenta o sistema utilizado na avaliação de desempenho da rede *intrachip*. Descreve as características dos geradores de tráfego e o modelo da rede *Hermes_Temp* desenvolvidos. A segunda parte apresenta os requisitos de comunicação do projeto. A terceira parte descreve o conjunto de métricas adotadas para a avaliação de desempenho. Na quarta

parte é definido o conjunto de simulações realizadas. A quinta parte apresenta a forma como as métricas são quantificadas e na sexta parte é realizada uma discussão das informações aqui apresentadas.

Capítulo 7 – Avaliação de desempenho

O objetivo deste capítulo é apresentar a avaliação de desempenho da rede *intrachip*. São apresentados os experimentos e os resultados obtidos a partir da análise do conjunto de seis métricas definidas no capítulo 6. São definidos os parâmetros de qualidade utilizados na avaliação de desempenho. O capítulo está dividido em cinco partes. A primeira parte apresenta os padrões de qualidade utilizados para avaliar o desempenho das configurações da rede. As três seções seguintes apresentam os resultados e a avaliação das simulações de 3 padrões de tráfego: transposto, *hot spot* e pseudo-aleatório. Na quinta parte é apresentado um resumo dos resultados obtidos.

Capítulo 8 – Conclusões e Trabalhos Futuros

Neste capítulo é apresentado um resumo das principais contribuições do trabalho aqui desenvolvido e os possíveis caminhos de um desenvolvimento futuro.

Capítulo 2

ESTADO DA ARTE

O objetivo deste capítulo é apresentar o desenvolvimento científico no campo da avaliação de desempenho de estruturas de comunicação para sistemas-sobre-silício (SoC) baseadas em redes intrachip (NoC). O capítulo fornece uma revisão não detalhada dos principais trabalhos encontrados na literatura dos últimos anos. O capítulo está dividido em quatro partes; a primeira parte descreve a importância da estrutura de comunicação dentro do projeto de um SoC, a segunda parte apresenta os trabalhos desenvolvidos para a avaliação de custo e desempenho da rede intrachip, a terceira parte descreve o trabalho desenvolvido pelo nosso grupo de investigação e finalmente na quarta parte é realizada uma discussão da informação apresentada.

2.1 PROJETO DE SOC

O êxito do projeto de um SoC consiste, entre outros fatores, na adequada escolha⁹ e dimensionamento¹⁰ dos componentes HW/SW das plataformas de processamento e de comunicação. Além de cumprir com um conjunto de requisitos funcionais e temporais, as plataformas devem também atingir ou otimizar algumas métricas de desempenho, obtidas a partir dos requisitos temporais do projeto, e custo¹¹. Com o aumento da complexidade dos SoCs, a reutilização de componentes *software* e/ou *hardware* previamente projetados tem se tornado uma tendência, o que facilita a tarefa de projeto das plataformas [POL03]. Como escolher estes componentes? Como configurar ou dimensionar os componentes escolhidos¹²? Quando deve ser projetado um componente específico para o projeto? Estes são alguns dos desafios que o projetista deve enfrentar. Ao responder estas perguntas, foi demonstrado um forte impacto da plataforma de comunicação no desempenho do

⁹ No caso de reaproveitamento de núcleos IP existentes.

¹⁰ No caso do projeto ou configuração das características de algum componente.

¹¹ Relacionado à área no silício ocupada e com a potência dissipada.

¹² Os componentes costumam ser especificados em função de parâmetros cujos valores são personalizados para cada projeto.

sistema, devido à sua dependência com a aplicação [LAH99]. Isto tornou relevante o estudo de desempenho da plataforma de comunicação dentro do sistema, quando este é submetido a diferentes condições de tráfego¹³

Os primeiros estudos de desempenho de estruturas de comunicação *intrachip* foram realizados para barramentos. Em [LAH01], foi apresentada uma análise de desempenho de três configurações de um barramento. Em cada um dos casos, os componentes mestres foram substituídos por geradores de tráfego aleatórios. Além disso, a metodologia empregada faz uso de uma implementação que independe da abstração da interface de comunicação, de forma a permitir uma avaliação *plug-and-play* das diferentes estruturas de comunicação. A métrica considerada foi o *throughput* de dados.

Em [POL03], foi analisado o impacto de três políticas de arbitragem para o acesso ao barramento *AMBA* sobre o desempenho de um *SoC*. As políticas de arbitragem utilizadas foram: round robin, *TDMA*, e faixa reservada. Neste trabalho são identificados três cenários de comunicação: tarefas mutuamente dependentes, tarefas independentes, e tarefas em *pipeline*. As métricas consideradas foram: tempo de execução, tempo de espera médio, e *throughput* de dados.

Em [SUH04], foi proposto um método para estimar o desempenho de um barramento. O método utiliza geradores de tráfego aleatórios que substituem os elementos mestres. O estudo pretendeu analisar o impacto da política de arbitragem e do número de elementos mestres no desempenho do barramento. As métricas consideradas foram: nível de utilização e latência de arbitragem.

Estes estudos contribuíram para a definição do conjunto de métricas de interesse no projeto de uma estrutura de comunicação. Permitiram também identificar a necessidade da estimativa de desempenho, independente da aplicação, e a necessidade do uso de componentes paramétricos que permitam simular diferentes condições.

¹³ Fluxo de dados dentro de uma estrutura de comunicação.

2.2 AVALIAÇÃO DAS REDES INTRACHIP

A rede *intrachip* foi proposta como uma nova forma de interconexão para os diferentes componentes de um *SoC* em [BEN01]. Desde então, diversos estudos foram publicados sobre a definição dos conceitos básicos da rede *intrachip*.

[DAL01] propôs definições para os componentes que constituem a rede *intrachip* (roteadores e enlaces), assim como para a influência exercida pelos parâmetros de configuração da rede (topologia, estratégia de roteamento e instanciação dos roteadores). Embora nenhum modelo de uma rede tenha sido apresentado, a importância deste trabalho foi a identificação do dimensionamento e a avaliação do custo e desempenho como os desafios do projeto de uma rede *intrachip*.

Uma vez definidos os componentes da rede, o esforço da pesquisa nesta área passou a focalizar o desenvolvimento destes componentes, gerando diversas arquiteturas de roteadores e topologias.

Uma topologia de árvore gorda (*butterfly fat tree*) foi implementada por [PAN03]. Esta incluía um controle adicional para conectar mais de um nodo de processamento ao um mesmo roteador, e algoritmos de roteamento complexos, modelados em RTL (*Register Transfer Level*). O objetivo do trabalho foi projetar uma rede que conectasse o maior número de nodos de processamento utilizando um número menor de roteadores. O projeto de sistemas complexos (algoritmos de roteamento e controles adicionais) envolve altos custos em termos de área ocupada no silício e dissipação de potência. Estes fatores devem ser avaliados em relação ao desempenho do sistema. Em [PAN03] inexistia qualquer tipo de avaliação nesse sentido.

Na busca de uma configuração adequada, componentes paramétricos substituíram os componentes fixos da rede.

Um modelo parametrizável de um roteador foi descrito em baixo nível (RTL) em [ZEF04]. Os parâmetros do roteador foram: a largura do canal, a estrutura de informação e a largura do *buffer*. A área da implementação foi expressa na forma de unidades funcionais (células lógicas, registros e memórias). O trabalho, entretanto, não apresentou uma análise de custo em termos da potência dissipada e não realizou um estudo do desempenho da rede. A ausência dessas informações impede quantificar as vantagens da funcionalidade dos componentes paramétricos no dimensionamento da rede.

2.2.1 Avaliação de custo das redes *intrachip*

O desenvolvimento das redes *intrachip* forneceu ao projetista um amplo conjunto de opções de configuração desta estrutura de comunicação. A preocupação em criar plataformas de comunicação de baixo custo, em termos do espaço ocupado no silício e da potencia dissipada, foi discutida nos trabalhos [NUR03] e [BAN04].

O número de configurações da rede que satisfazem as especificações do projeto pode ser amplo. Como escolher uma dessas redes? Qual se adapta melhor ao projeto? Estas são algumas das perguntas que são feitas em alguma etapa do fluxo do projeto de um *SoC* (vide capítulo 3). Para tentar respondê-las, foram investigados diferentes métodos que permitem a avaliação das diferentes configurações da rede. Os primeiros trabalhos avaliaram as redes através da quantificação de seu custo.

Em [PES04],[EIS04] e [BHO05] a avaliação da rede foi feita em razão da área ocupada pelos roteadores que a constituíam. Em [RHE04], a avaliação da rede foi realizada levando-se em consideração a potência consumida pelos roteadores, através de um modelo de dissipação constituído por dois fatores: a potência dinâmica e a potência estática. Estes trabalhos utilizaram uma descrição de baixo nível da rede (RTL), pois, segundo os autores, é o grau de detalhamento necessário para realizar a quantificação do seu custo.

Os trabalhos de [HU03], [HU04b] e [SRI05] implementaram algoritmos de dimensionamento das redes a fim de encontrar a solução com menor custo.

[HU03] apresentou um algoritmo que mapeia¹⁴ os componentes do sistema a uma rede *intrachip* grelha homogênea, de forma que a configuração final minimiza os custos da rede (expressados na forma de potência dissipada) atingindo os requisitos de comunicação. O algoritmo representa as características da rede e as características da aplicação. As características da rede são: os nodos de roteamento (configuração dos *buffers*) e o algoritmo de roteamento. As características da aplicação são: o volume da comunicação entre os componentes da plataforma de processamento e a largura de banda. Este trabalho demonstrou pela primeira vez como um modelo abstrato da rede (ao nível de algoritmo) pode ser utilizado para otimizar o seu custo, atingindo os requisitos temporais e funcionais do sistema. Entretanto, o modelo exige um conhecimento exaustivo do tráfego da aplicação e utiliza poucos parâmetros de configuração da rede.

O trabalho desenvolvido em [HU04b] propôs um algoritmo para customizar o projeto do roteador nas redes *intrachip* através da alocação dos *buffers* em cada nodo de roteamento. O algoritmo tem como parâmetros de entrada as características do tráfego da aplicação e as restrições de espaço do projeto. A partir das soluções obtidas, foi realizada a avaliação de desempenho em termos de tempo de serviço. A importância deste estudo foi introduzir o conceito de redes heterogêneas, onde os roteadores que a constituem não são necessariamente idênticos. O trabalho, porém, não apresenta um estudo de custo das diferentes configurações da rede. Além disso, os parâmetros da rede empregados são limitados.

[SRI05] empregou *ISIS*, um algoritmo genético que seleciona a configuração da rede com o menor custo. A utilização do algoritmo resulta no número de roteadores e conexões da rede. A inovação do estudo foi apresentar um algoritmo genético para o dimensionamento da rede, mas a incapacidade de abstrair outras características da rede, limita o conjunto de possíveis soluções.

¹⁴ Alocação dos núcleos IP do sistema.

2.2.2 Avaliação de desempenho das redes *intrachip*

Recentemente o estudo de desempenho ganhou importância dentro da avaliação da rede. Métricas como latência [JAL04], [NIL04] e vazão [BER05] foram propostas para avaliar esse desempenho.

A diminuição da latência da rede foi o objetivo do trabalho apresentado em [RAM05]. O autor projetou uma rede com enlaces tolerantes a falhas, isto é, verifica a correta transmissão da informação ao longo do caminho percorrido, a fim de reduzir a latência causada pela retransmissão. A rede inclui um circuito para detecção de erros ao longo do enlace. Embora tenha sido comprovada a diminuição da penalidade de tempo, [RAM05] não realizou uma análise do custo adicional causado pela adição dos circuitos ao longo dos enlaces. A análise desse custo é importante, pois garante que a melhoria do desempenho da rede não represente custos elevados.

O estudo apresentado em [BER05] comparou o desempenho de cinco topologias da rede (malha, toróide, hipercubo, borboleta e indireta de três estágios) utilizando duas métricas: vazão e latência.

Poucos trabalhos apresentaram um estudo do custo e do desempenho de uma rede. [RIJ03] realizou um dos primeiros estudos sobre a relação entre a complexidade do *hardware* do sistema e a sua eficiência. Neste trabalho foi projetado um roteador que garantia os serviços de vazão e do melhor esforço. Foi implementada uma rede 5x5 que atingiu uma largura de banda de 80Gbits/s. A Figura 2.1 apresenta o esquema dos roteadores implementados.

O serviço de vazão emprega de maneira ineficiente os recursos de *hardware* e o serviço do melhor esforço não garante o seu desempenho. Este trabalho contribuiu para o desafio do projetista na tarefa de atingir um ótimo desempenho com o menor custo no desenvolvimento da rede.

Figura 2.1 Roteadores implementados por [RIJ03].

A partir do momento que os conceitos foram desenvolvidos e a avaliação do custo e desempenho foi satisfatório, as redes *intrachip* passaram a ter utilização prática.

Em [WOL04] foi projetada uma aplicação para vídeo embarcada em um *SoC*. O projeto utilizou uma rede *intrachip crossbar* (vide capítulo 4) como estrutura de comunicação. A rede foi dimensionada em 3x3 com portas de 16 bits. Neste trabalho o desempenho da rede foi comparado com o de um barramento, e ficou demonstrado que a eficiência da rede superava em 132% a do barramento. Para realizar a avaliação de desempenho, [WOL04] quantificou a utilização da rede no ambiente de simulação OPNET. O sistema final implementado por [WOL04] é apresentado na Figura 2.2 abaixo.

Figura 2.2 Sistema implementado por [WOL04]

Este trabalho utilizou pela primeira vez uma rede *intrachip* em uma aplicação real. Na avaliação da rede, ficou evidente que as ferramentas de simulação até então existentes (OPNET e NS2) não eram as mais adequadas para esse tipo de análise, uma vez que, a menor unidade de tempo utilizada pelas ferramentas é o segundo e nas redes *intrachip* a unidade de grandeza é o nanosegundo. Ambas as ferramentas disponíveis também não levam em conta o custo da implementação da rede. As limitações que as ferramentas disponíveis (OPNET e NS2) apresentaram e a importância da simulação do comportamento da rede, que é dependente das diferentes condições de tráfego, motivaram o desenvolvimento de ferramentas de simulação que permitissem observar o comportamento da rede como um todo.

Atualmente, os grupos de investigação desenvolvem ambientes de projeto da rede que constam de bibliotecas de componentes previamente projetados e ferramentas de simulação que permitam encontrar a melhor configuração, refletida em baixo custo e bom desempenho.

Em [BER04] foi utilizada XPIPES, uma rede construída a partir de componentes previamente projetados e caracterizados. O objetivo foi criar uma rede com *QoS* (do inglês *Quality of Service*). No estudo foram também empregadas a latência e a largura de banda como métricas para avaliar as configurações da rede.

2.3 TRABALHO DESENVOLVIDO PELO GRUPO DE PESQUISA *GSEIS*

A avaliação de desempenho das redes *intrachip* é o objeto de algumas das pesquisas do grupo GSEIS (*Sistemas Eletrônicos Integrados e Software Aplicado*) da USP.

[ESL03] estimou o desempenho de um barramento denominado *Simple_Bus*. Para isto, foi simulado um *SoC* cujos componentes mestre do sistema foram substituídos por geradores de tráfego determinísticos e parametrizáveis. A avaliação de desempenho baseou-se na quantificação de cinco métricas (duração média das transações, *throughput* de transações, *throughput* de dados, nível de utilização do barramento, e nível de utilização por elemento) obtidas a partir das diferentes simulações geradas.

Em [LOZ05] estes componentes mestres do *Simple_Bus* foram substituídos por geradores de tráfego aleatórios. Foram também desenvolvidas ferramentas de captura e análise de dados. O modelo do barramento foi incrementado com novas políticas de arbitragem: de prioridades fixas e dois níveis.

Atualmente há um trabalho em andamento que pretende avaliar o desempenho de microprocessadores, mestres da comunicação, que permitirá no futuro modelar adequadamente os componentes mestres da comunicação.

O presente trabalho realiza a exploração do espaço do projeto das redes *intrachip* a partir de modelos de alto nível de abstração (*Transaction Level Model*), além de apresentar uma metodologia para a avaliação de desempenho da rede. O estudo utiliza a simulação de um conjunto de configurações (instâncias) da rede e diferentes condições de tráfego. A avaliação é feita pela quantificação de seis métricas: Duração média das transações, vazão das transações, largura de banda, tempo de espera médio por roteador, nível de utilização por roteador e dispersão. Como estudo de caso, a avaliação de desempenho de diferentes configurações da *NoC Hermes_Temp*¹⁵ foi realizada. Em contraste com os estudos prévios de avaliação de desempenho em alto nível, neste trabalho são desenvolvidas ferramentas de aquisição e análise de dados que permitem calcular métricas de desempenho e elucidar o que ocorre no interior da rede. A quantificação do comportamento global da rede como vem sendo feito até então, descrevia a *NOC* como uma caixa preta que fornecia um determinado valor de vazão e latência. Por outro lado, a avaliação dos componentes que constituem a rede, como foi

¹⁵ Modelo temporizado criado a partir de um modelo *untimed* existente (*Hermes*).

realizada neste trabalho, permite que o projetista possa detectar pontos de contenção dentro da rede e facilita o projeto de redes homogêneas e heterogêneas.

2.4 CONSIDERAÇÕES

Neste capítulo foi apresentado o desenvolvimento da pesquisa na área da avaliação de desempenho das redes *intrachip*, desde a sua proposta como estrutura de comunicação *intrachip*, em 2001, até o momento. A maioria dos estudos encontrados na literatura avalia e otimiza o custo da rede. Os trabalhos que avaliam o desempenho estão baseados na quantificação da latência e vazão.

O presente trabalho propõe uma metodologia para a avaliação de desempenho da rede a partir de uma descrição de alto nível (TLM), possível nas primeiras etapas do projeto. São empregadas seis métricas (duração média das transações, vazão das transações, largura de banda, tempo de espera médio por roteador, nível de utilização por roteador e dispersão) que permitem descrever o comportamento da rede não apenas na sua totalidade, mas em cada um de seus componentes. Essa avaliação mais detalhada permite uma melhor compreensão do comportamento da rede e ajuda a identificar as suas melhores configurações. O alto nível de abstração alcançado nesta metodologia significa fidelidade nas estimativas, porém perda na precisão da quantificação das métricas.

Capítulo 3

PROJETO DE SoCs (SYSTEM-ON-CHIP)

O objetivo deste capítulo é contextualizar a tarefa de estimar o desempenho de uma plataforma de comunicação dentro do fluxo do projeto de um SoC. O capítulo apresenta os fundamentos e principais conceitos do projeto de um SoC com ênfase no projeto da plataforma de comunicação e na estimativa de seu desempenho. O capítulo está dividido em cinco partes. A primeira parte apresenta as generalidades do fluxo de projeto de um SoC. As seguintes três seções descrevem cada uma das três etapas que compõem o fluxo de projeto: descrição do sistema, definição das plataformas (de processamento e comunicação) e integração dos componentes HW/SW. Por último, na quinta parte, será realizada uma discussão das informações aqui apresentadas.

3.1 FLUXO DE PROJETO DE UM SoC

O fluxo do projeto de um sistema-sobre-silício - SoC é apresentado (de forma hierárquica) na Figura 3.1. É uma copia da Figura 1.1 apresentada no capítulo 1 e aparece aqui novamente para facilitar a leitura do texto.

Este fluxo é dividido em três grandes etapas: descrição do sistema, definição das plataformas do sistema e integração dos componentes HW/SW. O projeto é baseado no conceito “*view points meeting*”, que mistura duas estratégias de projeto hierárquico: *top-down* e *bottom-up*. Em outras palavras, o projeto de um sistema é baseado na convergência entre dois projetos, o do sistema propriamente dito e o de seus componentes.

A estratégia *top-down* se caracteriza pelo fato do projeto do sistema ser desenvolvido através da transformação de uma descrição abstrata do comportamento (e de outras características) do sistema em uma descrição estrutural, composta de componentes de

hardware e software (compreende as etapas de “*descrição do sistema*” e “*definição das plataformas do sistema*”).

Figura 3.1 Fluxo de projeto de um SoC

Na estratégia *bottom-up*, por sua vez, o projeto de um SoC exige o reaproveitamento intensivo de componentes (HW/SW) previamente projetados (pela própria equipe ou por terceiros), denominados de núcleos IP. O aspecto *bottom-up* está relacionado com a pré-existência de todos os modelos (dos componentes) necessários para realizar todas as tarefas do projeto *top-down*.

A etapa da “*integração dos componentes de hardware e software*” compreende a adoção de três tipos de componentes:

Os *personalizados*, ou seja projetados especificamente para o projeto (identificados na Figura 3.1 pelas siglas *ASIC* e *AS_SOFT*).

Os *reaproveitados de outros projetos* ou *reaproveitados de bibliotecas* (identificados na Figura 3.1 pelas siglas *HARD_IP* e *SOFT_IP*). Trata-se de componentes pré-projetados e pré-caracterizados disponibilizados em bibliotecas (do inglês *design reuse*) conhecidos por módulos de propriedade intelectual (*IP modules*).

Adicionalmente, o sistema contém interfaces, componentes que permitem a comunicação entre cada módulo e os demais módulos com os quais troca informações (*intrachip* ou *extrachip*). Para isto, o projeto adota cada vez mais a técnica denominada *platform based design*, baseada na adoção de alguma interface padrão que garante a comunicação entre os núcleos IP selecionados.

Cada uma das três etapas do fluxo de projeto da Figura 3.1 (descrição do sistema, definição das plataformas do sistema, integração dos componentes HW/SW) é composta por um conjunto de tarefas que podem ser realizadas de forma manual (pelo projetista) ou de forma automática (empregando ferramentas). Nas próximas três seções as três etapas do fluxo de projeto são detalhadas.

3.2 DESCRIÇÃO DO SISTEMA

O fluxo de projeto tem início com uma especificação em linguagem natural do sistema, a qual fornece uma descrição detalhada dos serviços que o sistema proverá (requisitos funcionais) e as restrições a que deve obedecer, por exemplo, restrições da partição HW/SW, restrições temporais, uso de *SOFT_IP* ou *HARD_IP* de determinados fornecedores, etc. A partir destes requisitos, o projetista desenvolve a especificação funcional do sistema em uma linguagem formal, por exemplo, *SystemC* ou Java. Esta descrição pode ser temporizada ou não. A descrição corresponde a um modelo executável do projeto, que é usado para verificar (validar) a funcionalidade e outras especificações

através de simulações. Um problema associado a estas simulações é a criação de um *testbench* que forneça uma cobertura¹⁶ ótima para a validação do projeto.

3.3 DEFINIÇÃO DAS PLATAFORMAS DO SISTEMA

A definição das plataformas do sistema compreende inicialmente a análise da especificação funcional. Esta etapa inicial ocorre através da identificação e listagem dos três tipos de objetos funcionais da especificação: as ***variáveis*** (armazenam dados), os ***comportamentos*** (transformam os dados) e os ***canais*** (transferem dados) junto aos seus ***atributos*** (por exemplo, a conectividade, a frequência de troca de informação, os tipos de dados e os métodos implementados). Estes objetos funcionais fornecem parâmetros para o projeto das duas plataformas que irão constituir o *SoC*: a ***plataforma de processamento*** e a ***plataforma de comunicação*** (vide Figura 3.2).

A listagem das variáveis e dos comportamentos é empregada para projetar a plataforma de processamento. Por sua vez, a listagem dos canais e seus atributos, fornecem uma primeira abstração do tráfego que deverá ocorrer durante a operação do sistema. Esta informação é empregada para iniciar o projeto da plataforma de comunicação.

Figura 3.2 Componentes do SoC

¹⁶ Este termo é utilizado pelo seu significado intuitivo (qualidade de exploração do espaço de soluções). Para mais informações consultar [SHE05] e [EMA95].

Nesta etapa do fluxo do projeto são incluídos os requisitos da especificação¹⁷ e a biblioteca de componentes. Finalizado o projeto das duas plataformas do *SoC*, estas são integradas para realizar o processo de co-simulação¹⁸ do sistema.

3.3.1 *Projeto da Plataforma de processamento*

O objetivo do projeto da plataforma de processamento é escolher os componentes, tanto *hardware* quanto *software*, que satisfazem os requisitos de armazenamento de informação e de processamento do sistema. Para isto, é utilizada a listagem das variáveis e dos comportamentos, objetos funcionais obtidos a partir da análise da especificação funcional. O projeto da plataforma de processamento inclui tarefas tais como a partição HW/SW e a definição da arquitetura através da seleção de módulos HW/SW pré-existent e do projeto de módulos de HW/SW personalizados [KEM05],[COU02],[CHI04]e [WIE04].

3.3.2 *Projeto da Plataforma de comunicação*

O fluxo de projeto da plataforma de comunicação é apresentado na Figura 3.3. O objetivo do projeto da plataforma de comunicação é encontrar uma configuração que satisfaça as exigências da comunicação e otimize um ou mais indicadores de desempenho. O projeto parte da análise da comunicação a partir dos atributos dos canais, estabelecendo uma lista exaustiva de todos os objetivos a atingir (em termos de comunicação) que serão avaliados mediante a estimativa das métricas de desempenho. A tarefa seguinte consiste na seleção do tipo de plataforma de comunicação projetada. Esta pode ser um barramento (hierárquico) ou uma rede de comunicação *intrachip* (*NoC*).

Uma vez selecionado o tipo de plataforma de comunicação, esta é instanciada¹⁹ na fase de alocação dos recursos, composta por duas etapas: a **etapa física** e a **etapa lógica**.

¹⁷ Dentro do projeto existem dois tipos de requisitos: os da especificação e os que decorrem da especificação.

¹⁸ Referente à união das descrições RTL das plataformas de processamento e comunicação.

Figura 3.3 Projeto da plataforma de comunicação

Na primeira etapa são configuradas as características físicas da estrutura de comunicação²⁰ (forma de conexão dos componentes da rede, estrutura dos canais, estrutura dos roteadores do sistema) e são mapeados²¹ os *núcleos hardware* da estrutura de processamento. Na segunda etapa é definido o fluxo de informação dentro da estrutura

¹⁹ Termo empregado no texto como o processo de fixar os parâmetros que definem as características da plataforma de comunicação.

²⁰ Características físicas que podem ser representadas em um alto nível de abstração.

²¹ É estabelecida a localização física do componente *hardware* dentro da estrutura de comunicação.

de comunicação. São definidas as técnicas de arbitragem e de roteamento [BAN04] [JAL04]. A fase da alocação tem um efeito direto no desempenho do sistema [BHO05].

Uma vez instanciada a plataforma, é realizado o processo de verificação da aderência do desempenho da plataforma aos requisitos de comunicação²². Nesta fase do projeto isto pode ser feito a partir de uma análise de desempenho do sistema. Esta análise é baseada na estimativa de métricas de desempenho (que descrevem a qualidade da plataforma de comunicação), estimativas que podem ser feitas a partir do modelo da plataforma no nível de abstração correspondente à fase do fluxo em que o projeto se encontra.

A importância da análise de desempenho consiste na possibilidade do projetista escolher uma plataforma de comunicação dentre uma variedade de alternativas²³ (determinadas pelos valores dos parâmetros). Esta análise permite otimizar a plataforma de comunicação do sistema mediante a avaliação de suas possíveis configurações. [GON04].

A análise de desempenho, necessária para verificar o cumprimento dos requisitos de comunicação, pode ser realizada em diferentes etapas ao longo do fluxo do projeto (vide Figura 3.4). A análise feita nas etapas próximas à implementação final apresenta resultados mais precisos, entretanto, as possíveis modificações da plataforma resultantes da análise nesta etapa final são mais custosas. Em contrapartida, a análise feita nas primeiras etapas do projeto apresenta resultados menos precisos, entretanto, as modificações resultam menos custosas. A realização da avaliação do desempenho nas primeiras etapas do projeto do sistema, a partir do modelo em alto nível da plataforma de comunicação é, portanto, vantajosa. A falta de precisão pode ser compensada pelo fornecimento de uma avaliação fiel²⁴.

²² No fluxo não se incluiu a verificação funcional da estrutura de comunicação realizada depois de cada tarefa de projeto.

²³ Instancias da plataforma de comunicação.

²⁴ Termo empregado para descrever o caráter verdadeiro da avaliação do modelo.

A Figura 3.4 inclui o fluxo do projeto da Figura 3.3 juntamente com as etapas necessárias para a implementação física da plataforma de comunicação.

Figura 3.4 Fluxo de projeto da plataforma de comunicação incluindo a validação e a avaliação de desempenho

A análise de desempenho da estrutura de comunicação envolve a criação de diferentes cenários de tráfego pelo projetista. As características do tráfego podem ser definidas de quatro formas diferentes:

- (i) Simulação do *modelo da aplicação* (tráfego real na rede).
- (ii) A partir de geradores de tráfego *paramétricos* (o projetista controla a intensidade e as características do tráfego).
- (iii) A partir de geradores de tráfego *aleatórios* (a fim de obter uma grande diversidade de situações).
- (iv) Imposição de condições extremas (*corner cases*) do tráfego na rede.

A primeira forma (tráfego real) nem sempre pode ser realizada, pois esta informação frequentemente não está disponível (o proprietário do *SoC* pode desejar manter sigilo sobre sua aplicação) ou a informação simplesmente não é conhecida nas primeiras etapas do projeto. A simulação baseada em geradores paramétricos permite que o projetista ajuste os diferentes parâmetros de tal forma que condições críticas (por exemplo, tráfego intenso e tráfego por rajadas) possam ser avaliadas. A simulação baseada em geradores pseudo-aleatórios, geradores que expressam por meio de variáveis aleatórias os parâmetros previamente ajustados pelo projetista, permite avaliar situações talvez não previstas pelas simulações anteriores. A simulação sob condições extremas (quando conhecidas) permite avaliar o desempenho do sistema para estas situações particulares.

Cada uma das quatro formas descritas anteriormente oferece um nível diferente de cobertura, o que permite que estes quatro cenários não sejam excludentes, e sim complementares, isto é, cada um enriquece o estudo de desempenho da estrutura de comunicação.

3.4 INTEGRAÇÃO DOS COMPONENTES *HW/SW*

Uma das tarefas importantes do projeto do sistema (vide Figura 3.1) é a escolha do conjunto de seus componentes²⁵: núcleos de *hardware* pré-existentes de terceiros (IP) ou próprios e/ou os de aplicação específica (ASIC). Os núcleos de *hardware* pré-existentes podem ser disponibilizados de três formas distintas: *soft* (por exemplo, código VHDL-RTL), *firm* (por exemplo, esquema composto por células de uma biblioteca) e *hard* (leiaute). Os componentes ASIC precisam ser sintetizados a partir de sua descrição em alto nível (comportamental, RTL ou mesmo esquemático). O componente *software* é constituído pelo seu código, por exemplo, em linguagem C.

A tarefa do integrador compreende a adequação dos componentes do sistema. Esta consiste no conjunto de alterações necessárias para atingir os requisitos de integração ao

²⁵ Suas características já estão fixadas porque foi uma das tarefas do projeto.

projeto. O projeto de um *SoC* frequentemente é tratado como um problema de integração de módulos IP.

3.5 CONSIDERAÇÕES

Neste capítulo foi apresentado o fluxo de projeto de um *SoC*. Este foi dividido em três etapas. Na primeira etapa, denominada *descrição do sistema* é desenvolvida a especificação funcional do projeto, usada para criar um *testbench* e fazer validações funcionais. Na segunda etapa, são projetadas: 1 - a plataforma de processamento, de forma a satisfazer os requisitos de armazenamento e de processamento do sistema e; 2 - a plataforma de comunicação, de forma a satisfazer os requisitos de troca de informação entre os componentes do sistema. Simultaneamente, na etapa denominada de *integração dos componentes de hardware e software*, os componentes são escolhidos e configurados, isto é, ajuste dos parâmetros do componente para serem integrados ao sistema.

No capítulo também foi detalhado o fluxo de projeto da estrutura de comunicação. Sua dependência da aplicação e seu efeito no desempenho total do sistema faz com que uma configuração adequada de seus parâmetros seja um dos aspectos críticos do projeto. Isto motivou o desenvolvimento de uma metodologia de avaliação de desempenho de redes *intrachip* (estrutura de comunicação de um *SoC* adotado como objetivo deste trabalho) a partir de seu modelo em alto nível de abstração.

Capítulo 4

REDES INTRACHIP (Network-on-Chip NoC)

O objetivo deste capítulo é apresentar os fundamentos e principais conceitos relativos às redes de interconexão e identificar as principais características que definem uma rede de interconexão intrachip, que denominaremos resumidamente de redes intrachip (do inglês Network-on-Chip, NoC). O capítulo está dividido em quatro partes. A primeira parte apresenta os conceitos básicos das redes intrachip. A segunda parte lista os parâmetros de configuração de uma rede. A terceira parte apresenta algumas das aplicações encontradas na literatura que empregam uma rede intrachip como estrutura de comunicação. Na última parte é realizada uma discussão do capítulo.

4.1 CONCEITOS BÁSICOS

As redes *intrachip* surgiram como uma alternativa para superar as limitações da arquitetura de comunicação baseada em barramentos hierárquicos. Estas redes apresentam às seguintes características: (i) eficiência de energia e confiabilidade; (ii) escalabilidade da largura de banda; (iii) reutilização; (iv) decisão de roteamento distribuído [LIN03].

As redes *intrachip* são utilizadas para interconectar os *nodos terminais*²⁶ de um sistema por meio de *nodos de roteamento*²⁷. A ligação física entre dois nodos quaisquer (terminal ou de roteamento) de um sistema-sobre-silício (*SoC*) é denominada enlace (*link*). Um enlace possui um ou dois canais físicos de comunicação.

²⁶ Termo empregado no texto para denominar os componentes que fazem parte da arquitetura de processamento de um *SoC* como por exemplo processadores, memórias e componentes I/O.

²⁷ O termo nodo de roteamento é empregado para denominar os roteadores dentro da rede. O roteador é o componente que realiza a comutação dos pacotes dentro da rede (Vide página 84).

Na Figura 4.1 é apresentada uma rede em anel, onde cada nodo de roteamento possui ligações para dois nodos de roteamento vizinhos e para um nodo de terminal local.

Figura 4.1 Rede em anel.

As redes de interconexão são constituídas basicamente por roteadores e enlaces, estes últimos ligando tanto roteadores entre si como roteadores aos nodos terminais do sistema (componentes mestres ou escravos). Os roteadores estabelecem o caminho necessário para a transferência de informação pela rede. A informação é transferida sob a forma de mensagens, as quais podem ser divididas em unidades menores chamadas de pacotes. Estas características serão detalhadas nas seções 4.1.2 e 4.1.3.

A Figura 4.2 apresenta a configuração geral dos componentes da rede (terminais e roteadores).

Figura 4.2 Configuração geral de um (a) terminal ; (b) roteador.

4.1.1 Parâmetros da rede

O projeto da plataforma de comunicação (vide capítulo 2) baseia-se na seleção de um conjunto de parâmetros²⁸ que definem suas características físicas e lógicas. A combinação destas características determina o desempenho da plataforma de comunicação.

Os parâmetros da etapa física compreendem as características de dimensionamento geral da rede, isto é, a topologia (referente à geometria das ligações dos roteadores da rede), as conexões da rede com os diferentes núcleos de *hardware* da arquitetura de processamento, o número de *bits* dos *flits*²⁹, a quantidade e as características físicas dos roteadores e a alocação dos *buffers* (número e tamanho). As características da etapa lógica definem o fluxo da informação dentro da rede: arbitragem e roteamento [JAL04] [BAN04].

Em seguida são identificados e classificados os diferentes parâmetros que definem a plataforma de comunicação.

Parâmetros da etapa física da rede:

- ***Roteadores:***

- Conectividade

- * Quantidade de Roteadores

- * Vizinhança

- Interconexão

- * Redes diretas (Grelha, Toróide, Árvore gorda, Hipercubo)

- * Redes indiretas (Barras cruzadas, Multi-estágio)

²⁸ Termo empregado no texto para denominar o conjunto de atributos que podem ser escolhidos para configurar (instanciar) uma plataforma.

²⁹ Do inglês *flow control units*. Menor unidade de informação sobre a qual é realizado o controle de fluxo. Referir-se à seção 4.2.3.

- **Buffers:**
Alocação dos *buffers*
 - * Tamanho
 - * Quantidade
 - * Localização (Portos de entrada, saída)
 - * Tipo (FIFO, SAFC, SAMQ, DAMQ)
Tipo do *buffer*
 - * *Slack buffer*
 - * Canais virtuais
 - * Crédito
- **Enlaces:**
Enlaces e comunicação
 - * Largura de Banda
Divisão da mensagem
 - * Tamanho dos pacotes
 - * Tamanho dos *flits*

Parâmetros da etapa lógica da rede:

- **Roteamento:**
Tipo de roteamento
 - * Circuito
 - * Pacote (*Store-and-forward, Virtual cut-through*)
Adaptividade (**determinístico, adaptativo**)
Momento de realização do roteamento (**dinâmico, estático**)
Quantidade de destinos (**unicast, multicast**)
Tipo de implementação (**centralizado, fonte, distribuído**)
Forma de implementação (**tabela, máquina de estado**)
- **Arbitragem:**
Estática
Dinâmica (**deadline, FCFS, LRS, round robin**)

- **Níveis de serviço:**

Estratégias para evitar erro

- * Prevenir
- * Evitar
- * Recuperar

Qualidade do serviço

- * Vazão Garantida (*Guarantee throughput*)
- * Melhor esforço (*Best effort*)

No decorrer do capítulo serão explicadas em detalhe as características de maior relevância para o estudo de caso da *NoC Hermes*³⁰. Para mais informações sobre os outros parâmetros consultar [GOO05],[GRU04],[HU03],[KUM02] e [RIP03].

4.1.2 Roteadores

Roteador é o componente encarregado de comutar os pacotes dentro da rede. É constituído de:

- 1) **Núcleo de roteamento:** Matriz de interruptores que ligam os canais de entrada e saída dos diferentes portos do roteador.
- 2) **Lógica para roteamento e lógica para arbitragem:** Implementa o mecanismo pelo qual é definido a forma como é realizada a transferência de informação entre um roteador origem da rede e um roteador destino.
- 3) **Portos:** Pontos de acesso para outros nodos de roteamento e, dependendo da organização da rede, para seus nodos terminais.
- 4) **Buffers:** Servem para armazenamento temporário da informação que se dirige aos portos de entrada/saída. Podem possuir controladores de enlace que permitem a implementação do protocolo de comunicação.

³⁰ Modelo da *NoC* adotada para nosso estudo de caso [MOE05].

4.1.3 Enlaces

Um enlace pode possuir um ou dois canais físicos de comunicação. Os enlaces utilizados em redes de interconexão podem ser *full-duplex* ou *half-duplex*. O primeiro tipo é constituído por dois canais unidireccionais opostos, de modo a permitir a transferência simultânea de informação nas duas direcções do enlace. O segundo tipo é constituído por um canal, permitindo realizar transferências em duas direcções, mas não de forma simultânea.

4.2 CARACTERÍSTICAS DE UMA *NoC*

Existem três características que definem uma *NoC* [BEN02]:

1. **Topologia** da rede: Forma de interconexão entre os roteadores da rede. Por exemplo, rede grelha 2D.
2. **Estratégia de roteamento** (configuração lógica dos roteadores da rede): Algoritmo que define o caminho a ser percorrido por um pacote através dos canais da rede. Por exemplo, roteamento XY.
3. **Controle de fluxo** (configuração física dos roteadores da rede): Política de alocação de canais e *buffers*. Por exemplo, quantidade e tipo dos canais, tipo de *buffer* (fila), tamanho dos *buffers*.

Estas características são detalhadas nas secções 4.2.1, 4.2.2 e 4.2.3.

4.2.1 Topologia da rede

A arquitetura de uma *NoC* é composta por um conjunto de nodos conectados entre si através dos canais de comunicação. A forma como os roteadores estão conectados define a topologia da rede. A arquitetura da rede pode ser definida por meio de uma função $G(N,C)$, onde N representa o conjunto de roteadores da rede, e C representa o conjunto de

canais de comunicação. De acordo com a sua topologia, as redes podem ser classificadas em diretas e indiretas.

Nas redes diretas, cada componente é constituído por um roteador que está associado a um nodo terminal. Estes nodos possuem ligações ponto-a-ponto diretas para um determinado número de nodos vizinhos. Em termos de conectividade, a rede direta ideal é a completamente conectada, porém, sua escalabilidade é restrita e seu custo é proibitivo para um grande número de nodos. Como solução às dificuldades e limitações deste tipo de topologia, diferentes alternativas foram propostas na tentativa de obter uma boa relação entre desempenho e custo. Os resultados foram as redes diretas ortogonais, tais como a grelha (*mesh*) n-dimensional, o toróide (*k-ary n-cube*) e o hipercubo. Estas topologias são apresentadas na Figura 4.3.

Figura 4.3 Topologias da rede direta ortogonal: (a) Rede grelha; (b) Rede toróide; (c) Rede hipercubo.

Nas redes indiretas, os nodos terminais são conectados a uma rede de roteadores. Cada roteador por sua vez, possui um conjunto de portas bidirecionais para ligações com outros roteadores e/ou com os nodos da máquina. A interconexão destes roteadores define a topologia da rede. Duas topologias se destacam: barras cruzadas (*crossbar*) e as redes multi-estágio.

De acordo com a natureza das conexões, a topologia da rede também pode ser classificada como dinâmica e estática [MIL04]. Nas redes estáticas, cada nodo tem conexões fixas ponto a ponto a um certo número de nodos. Pertencem a esta categoria as

topologias hipercubo, grelha, e toróide. As redes dinâmicas empregam canais roteados dinamicamente configurados.

4.2.2 Estratégia de roteamento

Para garantir a comunicação entre os diferentes módulos, a rede possui um mecanismo que escolhe o caminho a ser percorrido por cada pacote através dos canais da rede. Este conjunto de mecanismos é implementado sob a forma de algoritmos. Na literatura existem diferentes algoritmos que visam atender os distintos requisitos de comunicação [DAL01].

Comunicação

As informações trocadas entre os componentes de um *SoC* são organizadas sob a forma de mensagens, as quais circulam na rede *intrachip* através dos roteadores, partindo do componente fonte ao componente destino da comunicação.

Em geral, uma mensagem possui três partes: Um cabeçalho (*header*), a carga útil (*payload*) e um terminador (*trailer*). O cabeçalho inclui informações de roteamento e de controle utilizadas pelos roteadores para propagar a mensagem em direção ao nodo destino da comunicação. O terminador inclui informações usadas para a detecção de erros e para a sinalização do fim da mensagem. Ambos formam um envelope ao redor da carga útil da mensagem.

As mensagens são divididas em pacotes para transmissão. Um pacote mantém uma estrutura semelhante à de uma mensagem (tem cabeçalho, carga útil e terminador) e é a menor unidade de informação que contém detalhes sobre o roteamento e sequência da informação. O pacote é ainda constituído por unidades menores denominadas *flits* (do inglês *flow control units*). Um *flit* é a menor unidade de informação sobre a qual é realizado o controle de fluxo (vide seção 4.2.3). Um *phit* (do inglês *physical unit*) é a unidade cuja largura é igual à largura física do canal (vide Figura 4.4).

Figura 4.4 Hierarquia da divisão das mensagens.

A função de uma rede é fornecer suporte físico à comunicação entre os processos alocados nos nodos. A rede transporta pacotes através de seus canais físicos e dos *buffers* dos roteadores. Uma comunicação é realizada com sucesso quando a informação enviada é corretamente recebida pelo destinatário.

Para garantir um correto funcionamento na transferência da mensagem, a *NoC* deve evitar os fenômenos de *deadlock*, *livelock* e *starvation*.

Deadlok: Esta situação ocorre quando existe uma dependência cíclica entre os roteadores da rede. Decorrem das características do tráfego e do algoritmo de roteamento adotado. Para evitar este estado de erro, os recursos necessários para a comunicação são garantidos antes do início da transmissão.

Livelock: Esta situação ocorre quando um pacote se mantém trafegando permanentemente na rede porque os canais necessários para chegar ao seu destino nunca se encontram disponíveis. Quando isto ocorre é necessário limitar o número de operações de desvio [DUA02].

Starvation: Esta situação ocorre quando dois ou mais *buffers* de entrada contêm pacotes destinados a uma mesma saída. Quando isto ocorre, uma política de arbitragem é necessária para seleccionar qual dos *buffers* de entrada deve ser conectado a uma saída. Exemplos de política de arbitragem são: prioridades fixas, prioridades dinâmicas, escalonamento por idade (*deadline*), FCFS (*First-Come-First-Served*), LRS (*Least Recently Served*) e RR (*Round Robin*)³¹. De acordo com o esquema de arbitragem adotado, um pacote armazenado em um *buffer* de entrada ao competir com outros pela mesma saída pode ser permanentemente bloqueado se o recurso requisitado é sempre concedido a outros pacotes. Esta situação, conhecida como *starvation*, pode ser evitada com o emprego de um mecanismo de arbitragem adequado³².

Roteamento

Existem dois métodos para a transferência das mensagens: Roteamento por circuito e roteamento por pacote [LIN03]. No roteamento por circuito, o caminho é estabelecido antes da mensagem ser enviada, negando permissão a outra comunicação que empregue qualquer canal associado à transmissão em progresso. Seu uso se justifica apenas nos casos em que as mensagens são longas e pouco freqüentes. No roteamento por pacote, uma mensagem é dividida em pacotes que serão transmitidos pela rede.

A principal diferença entre estas duas formas de roteamento é que na comunicação por roteamento via pacote não é realizada uma reserva do canal, isto é, o caminho é estabelecido dinamicamente. Este método resulta no uso de modos de recebimento e envio de pacotes, que definem como o pacote se movimenta pela rede. Os mais populares são *store-and-forward*, *virtual cut-through* e *wormhole* [MOR03]. No primeiro modo, o roteador envia um novo pacote se o anterior foi completamente recebido. Em *virtual cut-through*, o roteador envia o pacote assim que o roteador seguinte garanta que o pacote

³¹ Existem muitas políticas de arbitragem [DAL01].

³² A situação de *starvation* pode ser detectada pela simulação de situações de tráfego da rede.

todo seja aceito. O *wormhole* é uma variação do virtual *cut-through* onde um pacote é transmitido entre os roteadores em *flits*. Somente o *flit* cabeçalho contém a informação do roteamento, os *flits* restantes, que compõem o pacote, devem seguir o mesmo caminho reservado para o cabeçalho. A vantagem do *wormhole* é que a latência não depende diretamente da distância, como nos métodos anteriores, mas sim do tráfego entre os roteadores origem e destino. Outra vantagem deste método é a redução das filas nos roteadores intermediários, que não precisam armazenar o pacote inteiro. A desvantagem do método é a contenção de recursos causada pelo bloqueio do pacote.

4.2.3 Controle de fluxo

Os pacotes que trafegam pela rede competem pelos canais e pelos *buffers* para avançarem em direção aos seus destinos. Uma colisão ocorre, quando um pacote não pode prosseguir, uma vez que os recursos necessários já estão sendo utilizados. Neste caso, é empregado o controle de fluxo, uma política para decidir se o pacote é descartado, bloqueado (recebido e armazenado no lugar onde se encontra) ou desviado para outro caminho. Esta política lida com a alocação de canais e *buffers* para um pacote enquanto este trafega pela rede.

As redes *intrachip* realizam controle de fluxo ao nível de enlace. Cada porto do nodo de roteamento da rede pode possuir um *buffer* para o armazenamento do pacote sendo transferido. Se o *buffer* de entrada do receptor estiver cheio, o transmissor deve manter a informação armazenada em um *buffer* local até que os recursos estejam prontos para completar a transmissão. O mecanismo de controle deve bloquear a saída da informação do transmissor e o receptor deve enviar uma informação de controle ao transmissor, notificando se está pronto ou não para receber nova informação [MOR03].

Por exemplo, um protocolo do tipo *handshake* implementa o controle de fluxo. Uma linha de validação é empregada para que o emissor informe ao receptor sua intenção de

enviar um pacote para que este confirme a disponibilidade de espaço em seu *buffer* para receber a informação.

Entre os controles de fluxo mais conhecidos estão os baseados em *slack buffer* e os baseados em canais virtuais. No primeiro, ao conteúdo do *buffer* são impostos níveis inferiores e superiores. Uma mensagem de controle é enviada quando algum dos limites for infringido, provocando a suspensão ou o reestabelecimento do envio da informação pelo transmissor. No controle de fluxo baseado em canais virtuais, o *buffer* de entrada do canal físico é organizado em filas de profundidade menor, denominados canais virtuais. Quando um canal virtual fica bloqueado, o canal físico poderá ser usado por outro canal virtual, evitando o fenômeno HOL (*Head-of-Line blocking*) onde o canal físico permanece bloqueado [NUR03].

Arbitragem

A arbitragem é fundamental para a resolução de conflitos decorrentes da existência de múltiplos pacotes competindo por um mesmo caminho através dos canais da rede. O mecanismo de arbitragem deve ser capaz de resolver esses conflitos, selecionando um dos pacotes com base em algum critério e sem levar qualquer pacote a sofrer *starvation*.

O árbitro de um roteador pode ser implementado de forma centralizada ou distribuída. No primeiro caso, os mecanismos de roteamento e arbitragem são implementados em um mesmo módulo, o qual recebe os cabeçalhos dos pacotes, executa o roteamento e determina o porto de saída a ser utilizado por cada pacote. Em seguida, a arbitragem é feita, selecionando os pacotes a serem colocados em cada saída, configurando o roteador e habilitando os *buffers* selecionados para os mesmos efetuarem a transferência. Na implementação distribuída, o roteamento e a arbitragem são realizados de forma independente para cada porto bidirecional do roteador. Cada um dos portos possui um

módulo de roteamento associado ao seu porto de entrada e um módulo de arbitragem associado ao seu porto de saída.

O mecanismo de arbitragem pode ser baseado em diferentes critérios como, por exemplo, prioridade estática, prioridade dinâmica, escalonamento por *deadline*, FCFS (*First-come-first-served*), LRS (*Least recently served*), RR (*Round-Robin*) entre outros.

4.3 PROJETO DAS *NoCs*

Um sistema que implementa uma aplicação empregando uma rede *intrachip* como plataforma de comunicação exige uma exploração³³ do espaço de soluções da *NoC*. Esta análise surge da possibilidade de se criar diferentes configurações (instâncias) de uma *NoC*. No caso mais geral algumas configurações são capazes de atingir as especificações que satisfazem as exigências de comunicação dos componentes que constituem o *SoC*.

Na etapa do projeto da *NoC* surgem as seguintes perguntas:

- i) A qual dos roteadores da rede devem ser alocados os diferentes núcleos de *hardware*³⁴ do sistema?
- ii) Quais são os parâmetros de configuração da rede mais apropriadas para realizar a comunicação do sistema, de tal forma a garantir as especificações de desempenho (globais) do sistema e otimizar as métricas de interesse (definidas pelo usuário durante a fase de especificação do *SoC*)? [HU03] [HEN04].

Responder a estas perguntas equivale a encontrar valores adequados para os parâmetros que definem a plataforma das *NoC*, ou seja, encontrar uma configuração adequada para a *NoC*.

³³ Pela quantidade de parâmetros que afetam as redes *intrachip*, esta exploração não é exaustiva.

³⁴ Componentes HARD_IP e ASIC (vide capítulo 3).

A Figura 4.5 apresenta o fluxo de projeto da rede *intrachip*. Esta figura é compatível com o fluxo de projeto da plataforma de comunicação apresentado no capítulo 3 (vide figura 3.3) uma vez escolhida a *NoC* como estrutura de comunicação. Os círculos da figura apresentam as características próprias do projeto das *NoC*.

Figura 4.5 Fluxo de projeto da *NoC*

Qualidade do serviço (QoS)

O objetivo da rede *intrachip* é realizar a oportuna comunicação dos componentes da plataforma de processamento que estão ligadas à rede. Os serviços que a rede fornece incluem: *integridade dos dados* (os dados são transmitidos sem serem modificados), *entrega de dados sem perda* (todos os dados que transitam pela rede são entregues), *ordem dos dados* (a ordem de transmissão dos dados é estabelecida anteriormente), *vazão* e *latência*.

QoS (do inglês *Quality of Service*) foi criado para garantir o desempenho de uma estrutura de comunicação baseada em rede. Este índice faz referência à capacidade de

uma rede de fornecer um melhor serviço a um tipo específico de tráfego de uma rede. A aplicação deste conceito às redes *intrachip* surgiu da necessidade de combinar diferentes tipos de serviço empregados na comunicação dos componentes da plataforma de processamento, como os do **melhor esforço** (*best effort*) e os **serviços garantidos** (*guaranteed throughput*, por exemplo) assim como suportar diferentes paradigmas de acesso (requisição-resposta)[BOL04].

Serviços garantidos: Estes tipos de serviços reservam recursos para garantir o desempenho da rede mesmo sob as piores condições de tráfego. Como consequência os elementos da rede podem ser subutilizados.

Melhor esforço: Este tipo de serviços satisfaz as condições de desempenho da rede sob condições normais³⁵ de tráfego. Não requer reserva de recursos, o que diminui o custo da rede.

O índice *QoS* explora o compromisso existente entre a complexidade do *hardware* e o desempenho da rede. Algumas das modificações feitas na etapa de alocação (vide Figura 4.5) do fluxo do projeto da estrutura de comunicação para aplicar *QoS* são: aumento da largura de banda e aumento das dimensões dos *buffers* de entrada do roteador (na etapa física), adição de prioridade e identificação das mensagens (na etapa lógica).

Neste trabalho é utilizada a rede *intrachip Hermes_Temp*, um modelo temporizado descrito em alto nível e desenvolvido a partir da rede *Hermes*. Esta estrutura de comunicação não oferece suporte de *QoS*. Os pacotes são tratados da mesma maneira, sendo encaminhados na medida do possível (nível de serviço *melhor esforço*). Neste trabalho não será implementado o índice *QoS*. Para um melhor entendimento do *QoS* e a forma de implementação nas redes *intrachip*, consultar [BOL04].

³⁵ Condições de operação definidas pelo projetista dentro de um intervalo de tempo.

4.4 CONSIDERAÇÕES

Neste capítulo foram apresentados os conceitos básicos das redes *intrachip*. Esta estrutura de comunicação é constituída por roteadores, que encaminham as mensagens que transitam pela rede, e enlaces, que ligam os roteadores da rede. A rede é caracterizada pela sua topologia (forma de conexão dos roteadores), estratégia de roteamento (conjunto de mecanismos adotados para realizar a comunicação) e instanciação dos nodos de roteamento (escolha das características dos roteadores). A configuração destes elementos dá origem às diferentes instâncias de uma rede *intrachip*. Sua adequada escolha permite um melhor desempenho da rede sob condições de tráfego de uma aplicação. O capítulo 6 apresenta as características da *NoC Hermes*, adotada neste trabalho e os acréscimos feitos à rede ao longo deste trabalho para o desenvolvimento de *Hermes_Temp*.

Capítulo 5

MODELAGEM AO NÍVEL DE TRANSAÇÕES

O objetivo deste capítulo é identificar as características necessárias para realizar uma modelagem da comunicação ao nível de transações. O capítulo está dividido em quatro partes. A primeira parte introduz os diferentes tipos de modelagem de um sistema. A segunda parte descreve o conjunto de subníveis que constituem a modelagem no nível de transações. A terceira parte apresenta o conjunto de ferramentas associadas a SystemC, uma biblioteca de C++ que permite a modelagem em TLM dos sistemas. Na quarta parte é realizada uma discussão do conteúdo do capítulo.

5.1 INTRODUÇÃO

A complexidade dos sistemas que vêm sendo integrados em uma única pastilha de silício (sistema-sobre-silício - *SoC*) e a necessidade de se reduzir seu tempo de projeto, requer não apenas avanços tecnológicos como também métodos de projeto mais eficientes. Um dos recursos adotados para este fim foi incluir níveis mais abstratos do que RTL (*Register Transfer Level*) para modelar o sistema (fala-se em capturar as especificações funcionais e temporais do sistema em alto nível de abstração). Assim surgiu o nível de abstração denominado TLM (do inglês *Transaction Level Model*). Neste nível a **comunicação** entre os diferentes componentes (blocos) do sistema é separada da descrição do **comportamento** de cada um destes blocos [CAL03].

O uso de TLM reduz o esforço na modelagem do sistema e a velocidade de sua simulação é incrementada. TLM se propõe a ser uma ponte sobre a lacuna nos métodos disponíveis atualmente para integração de *hardware* e *software*, o que inclui as cinco tarefas de projeto [DON04]:

- (i) Desenvolvimento de *software*
- (ii) Exploração do espaço do projeto
- (iii) Especificação executável
- (iv) Avaliação de desempenho
- (v) Verificação funcional.

Em um *SoC*, os componentes de processamento que constituem a plataforma de processamento são conectados pela plataforma de comunicação. Esta pode ser implementada através de interconexões dedicadas, barramentos ou rede *intrachip*. Para maior informação vide capítulo 3, referente ao projeto de um *SoC*.

5.2 DESCRIÇÃO DO TLM

O nível TLM se constitui em uma abstração intermediária entre o nível SAM (*System Architectural Model*) [BLA04] e o RTL (*Register transfer level*). O nível SAM modela a funcionalidade do sistema, sem considerar os detalhes da arquitetura ou da implementação. No SAM, o sistema é descrito como um conjunto de eventos que incluem informações temporais. O nível RTL modela com precisão a implementação e a arquitetura do sistema. São utilizadas linguagens de descrição de *hardware* tais como VHDL e VERILOG.

TLM permite a descrição de plataformas separando o comportamento da comunicação do comportamento interno dos diferentes elementos do sistema. A modelagem da comunicação é baseada na **transferência** e não na forma de seu desenvolvimento. A propriedade característica do nível TLM é a abstração da comunicação em termos de interfaces que implementam o conjunto de métodos [MOU03].

5.2.1 Transação

Uma transação é a unidade fundamental da descrição no sistema TLM [PAS02]. Uma transação envolve a transferência de informação através da plataforma de comunicação.

Conceito geral de uma transação:

A transação surgiu como uma abstração para descrever as relações entre os diferentes comportamentos que implementam as transformações dos dados descritos nos sistemas complexos. Esta representação está associada a uma mudança de estado, resultado de uma transformação. A transformação apresenta as seguintes quatro características: **atômica** (ocorre ou não ocorre), **consistente** (a transformação é correta), **isolada** (não apresenta anomalias de concorrência) e **durável** (assim que for feita uma mudança, este estado é mantido).

Estas características constituem as propriedades **ACID** (atomicidade, consistência, isolamento e durabilidade) que identificam uma transação.

O conceito de transação pode ser ilustrado como um caso de uma operação bancária. Um depósito é uma transação devido a seu caráter *atômico*, isto é, só apresenta duas possibilidades, ou se deposita ou não se deposita o dinheiro. Não existem estados intermediários. É *consistente* se o dinheiro depositado corresponde ao mesmo valor incrementado na conta. A operação é *isolada* por ser possível realizar vários depósitos de forma concorrente e por serem independentes entre si. Finalmente, esta operação é *durável*, uma vez que, finalizada a transação, o resultado se reflete em um balance.

Transação em um SoC:

Dois módulos devem participar de um evento ou de uma transação: o módulo mestre e o módulo escravo. Quando o mestre inicia a transação, os módulos escravos podem somente reagir de forma passiva. Os mestres são os processadores, os controladores de acesso direto à memória (DMA) ou blocos ASIC, já os escravos são memórias, co-processadores, módulos de E/S.

Uma transação é realizada, por exemplo, quando um componente mestre requisita a leitura de um dado na memória. Para isto, é realizada uma **transação de leitura**, especificando o endereço de memória onde se encontra o dado a ser lido. Outro caso pode ser uma **transação de escrita** por parte do *software* embarcado (embutido) nos

registradores. Outros exemplos incluem **transações de atualização** e **transações de autenticação** quando implementados em serviços de segurança.

Uma transação é modelada como uma sequência de eventos que garantem a correta operação do *SoC*. Uma interrupção pode ser considerada como um evento de transação. A transação inclui a transferência de dados de informação e controle. Isto é, uma coleção de sinais que especifica o fluxo da informação [PAS02]. Ao modelar a comunicação em termos de transações, a quantidade de informação processada durante a simulação é minimizada, diminuindo o tempo empregado e otimizando a operação de verificação e exploração do espaço de soluções [OGA 03].

5.2.2 Sub-níveis TLM

TLM faz referência a um nível de abstração formado por um conjunto de subníveis que variam o grau de detalhe das descrições funcional e temporal do sistema³⁶ [DON04]. De acordo com a precisão temporal, a modelagem da comunicação e do processamento do sistema pode ser dividida em: *não temporizadas* (NT), de *tempo aproximado* (TA) e *temporizadas com precisão de ciclos de relógio* (TC).

Considerando a ortogonalidade entre o projeto da plataforma de processamento e a de comunicação, a Figura 5.1 apresenta as possíveis combinações de detalhamento da modelagem do processamento e da comunicação do sistema [BLA04].

A modelagem ***não temporizada*** (NT) representa o processamento do sistema como um conjunto de *tarefas, funções e processos*. Estes elementos se comunicam pela troca de estruturas de dados. O sistema é descrito como uma *hierarquia de tarefas* que seguem uma ordem de execução. Não inclui nenhuma informação temporal. A modelagem ***tempo aproximado*** (TA) modela o processamento do sistema como um conjunto de *componentes* que implementam as tarefas, funções e processos do sistema.

³⁶ Sistema constituído pelas plataformas de processamento e de comunicação.

Figura 5.1 Níveis de modelagem do sistema

Os componentes se comunicam de forma sincronizada. O sincronismo é atingido pela ocorrência de *eventos*. Na modelagem **temporizada por ciclos** (TC), os componentes do sistema se comunicam segundo um protocolo de comunicação. O processamento do sistema é realizado pela execução de um conjunto de instruções. O sistema realiza as operações de processamento e comunicação de acordo com uma base de tempo.

Segundo a terminologia empregada em [GAJ04][BLA04], as 9 possíveis combinações da figura 5.1 originam 6 diferentes subníveis de descrição³⁷ conforme à granularidade³⁸ da transação: SAM, RTL, TLM1 (montagem do componente), TLM2 (arbitragem), TLM3 (funcionalidade) e TLM4 (processamento com precisão de ciclo). A Figura 5.2 apresenta parte da simulação de um sistema. Nesta figura são ilustrados os subníveis TLM. O sistema é constituído por um conjunto de núcleos IP: IP1, IP2 e IP3. Na simulação é exemplificada a transação de autenticação. Esta transação é implementada em um sistema que oferece serviços de segurança e consiste na verificação da veracidade do certificado digital recebido. Neste sistema o componente IP1 recebe uma mensagem

³⁷ Os três subníveis não indicados na figura 5.1 não apresentaram nenhuma utilidade até hoje.

³⁸ Faz referência ao conjunto de sinais que constituem uma transação ou evento.

que corresponde a um certificado digital do componente que será autenticado. A autenticação envolve os componentes IP2 e IP3. O IP2 envia a informação necessária para que o IP3 verifique a integridade da mensagem recebida por IP1. Finalmente, a transação de autenticação se completa quando o resultado do processamento da informação pelo componente IP3 é comunicado a IP1.

Figura 5.2 Exemplo de comunicação de um sistema

O nível TLM1 é empregado nas primeiras etapas do projeto do sistema, quando os detalhes exatos de sincronismo da estrutura de comunicação não afetam as decisões do projeto. Neste nível, todos os detalhes envolvidos na transferência de informação são encapsulados na forma de uma transação. No exemplo da Figura 5.2, está representada uma transação de *autenticação* que envolve três módulos IP. O nível TLM2 modela a transação pelos componentes que nela participam. O TLM3 descreve a transação em função da ocorrência dos eventos de comunicação. No exemplo da Figura 5.2, são utilizados os eventos de *request*, *acknowledge* e *grant*. O nível TLM4 utiliza um sinal periódico, o sinal *clk* no exemplo da figura, como base temporal para o sincronismo da comunicação.

Em cada um dos sub-níveis TLM, a granularidade da comunicação pode mudar, empregando estruturas de transmissão de informação com diferentes graus de

complexidade que compreendem desde a forma de mensagens, pacotes, *flits*, até simples *bits*. Estes níveis são utilizados dentro do fluxo de projeto da plataforma de comunicação de acordo com o grau de detalhe funcional e temporal necessários em cada etapa.

O estudo de caso de avaliação de desempenho de uma rede *intrachip* foi desenvolvido a partir de um modelo de uma rede descrita em TLM4 (denominada *Hermes_Temp*), adaptado de um modelo TLM não temporizado de uma rede existente (denominada *Hermes*). Nesta modelagem, os mecanismos de transporte de informação (fios na implementação física) são abstraídos na forma de sinais ou variáveis, além de existir uma seqüência de eventos para realizar a transferência de dados, o que constitui o protocolo de comunicação.

5.3 SYSTEMC

SystemC é uma biblioteca de classes C++ que permite a modelagem dos componentes dos sistemas *hardware* em um alto nível de abstração. É também uma metodologia através da qual é possível descrever e validar as características de diferentes tipos de sistemas eletrônicos em diferentes níveis de abstração.

SystemC contém um conjunto de ferramentas de modelagem e sincronização da comunicação de um projeto descrito no nível TLM: *interfaces*, *canais* e *eventos*.

Uma interface é definida como um objeto funcional que consiste em um conjunto de métodos³⁹. Um canal é uma estrutura que contém uma ou mais interfaces de um porto, o que permite ao módulo (e conseqüentemente ao seu processo) acessar a tal (tais) interface(s). O porto somente pode acessar os canais pelas interfaces e unicamente tem acesso ao canal através dos métodos que a interface define.

A Figura 5.3 apresenta as convenções usadas para a representação gráfica dos elementos empregados para a modelagem do sistema.

³⁹ Um método é um procedimento, uma declaração de uma tarefa ou uma função.

Figura 5.3 Representação gráfica das ferramentas de modelagem

Há dois tipos de canais: os primitivos e os hierárquicos [CAL03]. Um canal **primitivo** não apresenta uma estrutura, não contém processos e não pode acessar diretamente outros canais primitivos. Canais **hierárquicos** são módulos (têm estrutura) que contêm outros módulos e processos, com capacidade de acessar diretamente outros canais.

SystemC define os canais de comunicação e permite que o usuário defina as características dos canais de alto nível para desenvolver seus projetos. Estes canais satisfazem as necessidades de velocidade de simulação para o desenvolvimento de *software* embutido.

Características adicionais que favorecem o uso de *SystemC* devem-se ao fato de estar vinculada a uma organização responsável pela evolução e suporte da linguagem, chamada *Open SystemC Initiative*. Esta organização disponibiliza uma lista de suporte de alcance mundial. Adicionalmente, *SystemC* foi escolhida por grandes empresas como base para o desenvolvimento de ferramentas de modelagem, validação e síntese de sistemas. Exemplos destas são *Synopsys Inc.* e *CoWare Inc.* Além disso, a captura e a validação do sistema pode ser feita mediante uso de ferramentas gratuitas.

5.4 CONSIDERAÇÕES

Neste capítulo foram apresentados os principais conceitos do nível TLM (*Transaction Level Model*). TLM é empregado para modelar sistemas de uma forma mais abstrata que no nível RTL (*Register Transfer Level*). Em TLM a comunicação entre os diferentes

componentes (blocos) do sistema é separada da descrição do comportamento de cada um destes blocos. Segundo a granularidade da transação o modelo pode ser descrito em quatro diferentes subníveis: TLM1, TLM2, TLM3 e TLM4. Para o estudo de caso do presente trabalho foi desenvolvido um modelo da rede *intrachip Hermes*⁴⁰ no subnível TLM4. Atualmente, pretende-se estabelecer uma definição mais clara que permita a padronização deste nível de descrição.

⁴⁰ Modelo de rede *intrachip* desenvolvido em TLM1

Capítulo 6

DESCRIÇÃO DO SISTEMA EXPERIMENTAL

O objetivo do capítulo é apresentar a metodologia para realizar a estimativa de desempenho de uma rede intrachip (NoC) a partir de seu modelo em TLM (transaction level modeling) temporizado. O capítulo está dividido em seis partes. A primeira parte apresenta o sistema utilizado na avaliação de desempenho da rede intrachip. Descreve as características dos geradores de tráfego e o modelo da rede Hermes_Temp desenvolvidos. A segunda parte apresenta os requisitos de comunicação do projeto. A terceira parte descreve o conjunto de métricas adotadas para a avaliação de desempenho. Na quarta parte é definido o conjunto de simulações realizadas. A quinta parte apresenta a forma como as métricas são quantificadas e na sexta parte é realizada uma discussão das informações aqui apresentadas.

A Figura 6.1 apresenta o fluxo geral do projeto da plataforma de comunicação. É uma copia da Figura 3.3 apresentada no capítulo 3 e aparece aqui novamente para facilitar a leitura do texto.

Uma vez estabelecidos os requisitos de comunicação do projeto e escolhida uma rede *intrachip* como plataforma de comunicação, a tarefa do projetista consiste em encontrar a configuração da rede que melhor se ajuste às necessidades do projeto, de acordo com os requisitos do projeto e otimizando alguma métrica de custo ou desempenho. A tarefa do projeto da rede envolve a avaliação das diferentes instâncias (configurações) da rede, geralmente disponibilizada na forma de um núcleo IP próprio ou de terceiros.

A metodologia proposta para este trabalho consiste na avaliação de desempenho da rede a partir de um modelo em alto nível de abstração (TLM - *SystemC*). Esta metodologia é baseada na simulação do comportamento da rede sob diferentes condições de tráfego e na coleta de dados, o que permite calcular um conjunto de métricas definidas pelo projetista.

Figura 6.1 Projeto da plataforma de comunicação

6.1 SISTEMA PARA A AVALIAÇÃO DE DESEMPENHO

A Figura 6.2 abaixo apresenta o sistema empregado no desenvolvimento do trabalho experimental. O sistema é constituído pelo conjunto de blocos mestres⁴¹ que se comunicam com os blocos escravos através de uma rede *intrachip*. O processo de comunicação é acompanhado e avaliado através do emprego de um conjunto de monitores. Estes monitores calculam métricas a partir das quais é feita a estimativa do

⁴¹ Os blocos mestres são simulados empregando geradores de tráfego.

desempenho da rede para as diferentes condições de tráfego estabelecidas pelo projetista durante as primeiras etapas do fluxo do projeto da plataforma de comunicação.

Figura 6.2 Sistema empregado para a avaliação do desempenho.

6.1.1 Geradores de tráfego

Os geradores de tráfego são módulos utilizados para simular o comportamento dos blocos mestres. Estes componentes estabelecem diferentes condições de tráfego baseados em modelos que representam a geração de informação em um sistema de computação. A informação pode ser transmitida sobre a forma de mensagens, pacotes, palavras, rajadas ou *bits*. A granularidade escolhida para a transmissão de informação da rede no presente trabalho foi o pacote.

Para garantir o desempenho da rede *intrachip* seria necessário simular todas as situações de tráfego do sistema durante sua execução. Com o objetivo de limitar o esforço exigido na cobertura de todo o espaço de simulação, o projetista deve criar cenários que representam um conjunto de condições de tráfego da rede. Estes cenários

devem ser significativos para fins de avaliação de desempenho⁴². No trabalho são empregados geradores de tráfego **paramétricos e pseudo-aleatórios**. Nos geradores paramétricos o projetista controla a intensidade e outras características do tráfego. Nos geradores pseudo-aleatórios as características do tráfego são estabelecidas por distribuições de probabilidade a fim de obter uma grande diversidade de situações (vide seção 3.3 do projeto da plataforma de comunicação).

As condições de tráfego são estabelecidas pelas seguintes características: tipo de dispositivo (mestre ou escravo), padrão de acesso a outros dispositivos, tipo de acesso (por exemplo leitura ou escrita), intervalo de tempo entre as requisições de comunicação por um mesmo componente, tamanho e granularidade da informação a ser transmitida e finalmente o *Hurst* (parâmetro de medição de autosimilaridade) [GOM98], [LAH01], [ARR05].

Neste trabalho são utilizados três dos parâmetros supracitados para modelar as condições de tráfego: intervalo de tempo entre as requisições de comunicação por um mesmo componente, tamanho da informação a ser transmitida e padrão de acesso a outros dispositivos. Estes parâmetros são denominados no presente trabalho por, respectivamente, tempo de geração de pacotes, largura dos pacotes e destino da comunicação.

A modificação (paramétrica ou aleatória) destes três parâmetros estabelece as diferentes condições de tráfego.

Tempo de geração de pacotes:

Figura 6.3 Representação do tempo de geração de pacotes.

⁴² Envolve o conceito intuitivo de cobertura (qualidade de exploração do espaço de soluções), mencionado no capítulo 3.

O parâmetro t na Figura 6.3 representa o intervalo de tempo entre pacotes consecutivos de uma mesma mensagem gerados por um mesmo componente mestre ligado à rede. Na modelagem paramétrica são empregados três valores de tempo (25, 100 e 150 ciclos). Para sua modelagem aleatória é empregada a distribuição exponencial. A unidade utilizada no tempo de geração de pacotes é ciclos de relógio.

Largura dos pacotes:

Figura 6.4 Representação da largura dos pacotes.

O pacote que circula pela rede é formado por um cabeçalho, por uma carga útil e por um terminador. O cabeçalho é constituído pela informação de endereçamento do componente mestre e componente escravo, além da informação de controle que corresponde ao tamanho da carga útil. A carga útil é constituída pela informação gerada pelo componente mestre. Finalmente, o terminador é constituído por mecanismos de detecção de erros.

A largura do pacote representa o número de *flits* da carga útil (L na Figura 6.4), sendo que *flits* é a unidade utilizada em sua medida de largura. Na modelagem paramétrica são empregados três valores de L (5 *flits*, 20 *flits* e 100 *flits*).

O processo de *Poisson* é empregado na modelagem aleatória da largura do pacote e utiliza uma média de 25 *flits*.

Destino do pacote:

O destino do pacote é o endereço do componente escravo da comunicação dentro da rede.

Figura 6.5 Destino do pacote.

A Figura 6.5 apresenta um exemplo de uma rede 4x4 onde o componente mestre $B2$ se comunica com os componentes escravos $B1$, $B5$, $B6$ e $B11$. Na modelagem paramétrica são estabelecidos dois cenários: *hot spot* e *transposto*.

No cenário *hot spot*, o componente mestre possui um destino para se comunicar com maior frequência. No cenário *transposto*, o componente mestre ligado a um roteador localizado na posição (i,j) da rede se comunica unicamente com o componente ligado à posição (j,i) da mesma. Portanto, no *hot spot*, o componente mestre além de se comunicar com o nodo preferencial, pode se comunicar também com outros nodos ligados à rede (escravos múltiplos). Por outro lado, no *transposto*, o alvo da comunicação é sempre o mesmo (escravo único).

Estes cenários são empregados na literatura para modelar de forma paramétrica a localidade na comunicação, onde para cada componente mestre existe um componente escravo correspondente que apresenta uma probabilidade maior de se comunicar [MAG06].

Na modelagem aleatória do destino da comunicação, os componentes mestres se comunicam uniformemente com os outros nodos da rede. Este modelo fornece aos componentes ligados à rede a mesma probabilidade de ser escravo da comunicação [MAG06].

O conjunto de valores que representam o destino do pacote são adimensionais.

Neste trabalho são adotadas três combinações dos parâmetros supracitados: (i) largura do pacote e destino do pacote constante e tempo de geração variável, (ii) largura do pacote e tempo de geração constante e o destino do pacote variável e (iii) largura do pacote variável e tempo de geração e destino do pacote constante. As três combinações são apresentadas na Figura 6.6.

Figura 6.6 Três combinações dos parâmetros do gerador de tráfego: (a) largura do pacote e destino do pacote constante e tempo de geração variável; (b) largura do pacote e tempo de geração constante e o destino do pacote variável; (c) largura do pacote variável e tempo de geração e destino do pacote constante

6.1.2 Blocos escravos

Estes módulos recebem os *flits* enviados pelos módulos mestre na rede. O cabeçalho do pacote é armazenado temporalmente até confirmar o correto recebimento da carga útil. O bloco escravo notifica à rede que o pacote foi aceito.

6.2 REDE INTRACHIP

Atualmente existem aproximadamente 17 redes *intrachip* desenvolvidas no mundo [MOR05]. Alguns dos modelos se encontram disponibilizados por grupos de investigação (rede *AEthernet* [GOO05], *Hermes* [MEL05]). Para nosso estudo de caso é empregada a rede *Hermes_Temp*, um modelo em *SystemC* temporizado criado ao longo deste trabalho a partir de um modelo em *SystemC* não temporizado da rede *Hermes* disponível em [FEN05]. Esta rede foi escolhida devido ao suporte oferecido pelo grupo de investigação que a criou e pela sua ampla documentação.

6.2.1 REDE HERMES

Hermes é uma rede de comunicação *intrachip* parametrizável, projetada pelo grupo de investigação GAPH da Pontifícia Universidade Católica do Rio Grande do Sul. Esta rede está modelada em VHDL-RTL e *SystemC* TLM não temporizado [MOE05]. *Hermes* é uma rede de interconexão para sistemas digitais multiprocessadores. Esta rede foi estruturada em camadas que encapsulam funções equivalentes àquelas definidas para os níveis hierárquicos do modelo de referência OSI (*Open System Interconnection*), um padrão internacional de organização de redes de computadores proposto pela ISO (*International Organization for Standardization*). Cada camada da estrutura implementa uma função da rede na forma de *hardware* e *software*. Estas camadas se comunicam através de um conjunto de regras que constituem o protocolo de comunicação.

Figura 6.7 Camadas do modelo OSI.

A Figura 6.7 apresenta as sete camadas do modelo OSI. A rede *Hermes* implementa as quatro primeiras camadas: física, enlace, rede e transporte.

A camada *física* fornece o meio mecânico e elétrico para interconectar os diferentes componentes. Na rede *Hermes* esta camada é representada pela conexão entre os roteadores (vide Figura 6.8), que realiza a transferência de dados através de um enlace (portos e correspondentes sinais de transmissão e recepção de informação). A largura do barramento empregado é escolhida em concordância com os recursos disponíveis (*buffers*, interconexões, área). Em *Hermes* a largura do barramento de dados (N bits) foi escolhida de forma a reduzir os requisitos de roteamento e o tamanho das filas. Este parâmetro é parametrizável em função de requisitos de desempenho.

Figura 6.8 Interconexão entre roteadores

A camada *enlace* encarrega-se da comunicação de um grupo de *bits* empregando a camada *física*. O objetivo é o estabelecimento da conexão lógica confiável entre os roteadores. Na rede *Hermes* está representada pela adoção do protocolo *handshake* com o fim de detectar erros e realizar uma transmissão correta da informação.

A camada *rede* realiza a comunicação de pacotes e é responsável pelo empacotamento das mensagens, roteamento dos pacotes entre a origem e o destino, controle de congestionamento e a contagem de pacotes transferidos.

A camada *transporte* estabelece uma conexão fim-a-fim entre a origem e o destino da mensagem. Isto inclui o controle de fluxo, segmentação e remontagem de pacotes. Na

rede *Hermes*, essa camada possui apenas a característica de conexão fim-a-fim, não implementando as demais funcionalidades atribuídas a ela.

Hermes é uma rede direta ortogonal que implementa a topologia grelha (*mesh*). O roteador proposto para a topologia grelha também pode ser usado para construir topologias do tipo *Torus*, Hipercubo e similares. Entretanto, construir estas topologias implica em modificações nas conexões dos roteadores, entre si e com seus nodos terminais, e no algoritmo de roteamento. Cada nodo de roteamento possui um conjunto de portas bi-direcionais com *buffers*, conectados aos outros roteadores e a um nodo terminal. A Figura 6.9 apresenta a topologia de uma rede *Hermes* 3x3 (9 roteadores e 9 IPs).

Figura 6.9 Topologia de uma rede *Hermes* 3x3

A comunicação entre os componentes da rede é baseada na transmissão de pacotes, cada pacote sendo composto de um cabeçalho e de sua carga útil. Contrário à Figura 6.4, o pacote utilizado na rede *Hermes* não tem terminador. A *NoC* permite que cada componente de processamento envie pacotes para qualquer outro componente de processamento conectado à rede *intrachip*.

Roteador

O roteador é o módulo principal da rede. É encarregado de realizar a transferência de mensagens entre os núcleos de *hardware*. O roteador da rede *Hermes* possui uma lógica de controle e cinco portas bidirecionais (*East, West, North, South, e Local*) como indica a Figura 6.10. Cada porto tem um *buffer* de entrada (FIFO, circular e parametrizável) para armazenar temporariamente a informação. O porto local estabelece a comunicação entre o roteador e o nodo terminal. Os demais portos estão conectados aos roteadores vizinhos.

Figura 6.10 Roteador da NoC *Hermes*

A lógica de controle da rota é implementada pela lógica de arbitragem e pelo algoritmo *packet-switching*. *Hermes* implementa o modo de comutação *wormhole*⁴³, onde um pacote é transmitido entre os roteadores na forma de *flits*. Somente o *flit* cabeçalho contém a informação do roteamento. Os *flits* restantes que compõem o pacote, devem seguir o mesmo caminho reservado para o *flit* cabeçalho.

O pacote é constituído pelo cabeçalho e carga útil. O cabeçalho contém os endereços do componente mestre e escravo da comunicação (coordenadas XY da rede), seguido do número de *flits* que o pacote contém (cabeçalho). A carga útil é composta pela informação gerada pelo componente mestre da rede (vide Figura 6.11).

Figura 6.11 Pacote transmitido pela rede

⁴³ Para maior informação referir-se ao capítulo 4.

À medida que os pacotes atingem os portos do roteador, estes são armazenados no *buffer* de cada entrada e a respectiva solicitação de transmissão é feita. O árbitro, que na rede *Hermes* implementa o algoritmo dinâmico rotativo *round robin*⁴⁴, concede um dos pedidos de comunicação e solicita o estabelecimento da conexão. Na etapa de roteamento, o endereço do roteador atual é comparado com o do roteador destino o que permite escolher o porto apropriado. Uma vez que o roteamento da rede *Hermes* é implementado através do algoritmo XY, este realiza primeiro o roteamento completo na direção horizontal (X) e depois na direção vertical (Y). Se o porto escolhido não está disponível, o pacote é bloqueado até que uma conexão possa ser estabelecida, afetando o desempenho de toda a rede.

Quadro resumido das características da rede *intrachip Hermes*

A Tabela 6.1 apresenta o resumo das características da rede *Hermes*.

Tabela 6.1 Quadro resumo das características da rede *Hermes*

Nível Físico	Caminhos bidirecionais de n bits com suporte a protocolo <i>handshake</i>
Nível Enlace	Protocolo assíncrono, técnica de <i>handshake</i>
Nível Rede	Comunicação ponto-a-ponto, utilizando comutação por pacotes
Nível Transporte e superiores	Não implementados – suporte a interface OCP
Pacote	1 <i>flit</i> com destino 1 <i>flit</i> com tamanho do <i>payload</i> (# de <i>flits</i>) $\leq 2^{n-1}$ <i>flits</i> de <i>payload</i> (n é o # de bits por <i>flit</i>)
Comutação	Comutação por pacotes
Repasse de Pacotes	<i>Wormhole</i>
Roteamento	Distribuído, Determinístico/Parcial Adaptativo, Mínimo
Topologia	<i>Mesh</i> (2x2 na prototipagem, <i>nxm</i> na simulação)
Filas	Na entrada de cada porto, com tamanho parametrizável.

⁴⁴ Uma vez atendida a requisição de comunicação, o núcleo atendido terá a menor prioridade no próximo atendimento.

6.2.2 REDE *HERMES_TEMP*

Hermes_Temp é um modelo temporizado de alto nível (TLM- *SystemC*) da rede *Hermes*. O modelo é descrito no subnível TLM4 (vide capítulo 5). A característica deste tipo de descrição é a modelagem do processamento do sistema com tempo aproximado e a modelagem da comunicação com precisão de ciclos. Este modelo requereu a inclusão de um sinal de relógio que sincroniza a ocorrência dos eventos de comunicação na rede⁴⁵.

Hermes é constituído por um conjunto de módulos. Estes módulos possuem processos concorrentes que descrevem a funcionalidade do sistema. Os processos se comunicam entre si através de canais ou eventos.

Na modelagem de *Hermes_Temp*, inicialmente foram identificados os módulos que compõem a rede *Hermes*, e em seguida, a sua respectiva funcionalidade foi estudada.

A Figura 6.12 apresenta o conjunto de máquinas de estado finitas (FSM) que descrevem os processos envolvidos na comutação de um pacote, definida como a transferência de um pacote de informação de um dos portos de entrada a um dos portos de saída de um roteador da rede. As máquinas de estado (a) e (b) correspondem, respectivamente, ao processo de espera de chegada de novos *flits* ao roteador e ao processo de identificação dos *flits* do pacote (*flit* destino: *CBC*, *flit* tamanho do pacote: *TAM* e *flits* de carga útil: *UTI*). Estes processos são implementados nos módulos correspondentes às portas de entrada do roteador.

As máquinas de estado (c) e (d) correspondem, respectivamente, ao processo de arbitragem e ao processo de roteamento. Estes processos são implementados no canal interno do roteador junto com a lógica de controle do roteador.

⁴⁵ A comunicação da rede é definida como a transferência de pacotes desde o componente mestre até o componente escravo.

Figura 6.12 Processos da rede envolvidos na comutação de um pacote

Ao longo da execução do primeiro processo (espera de chegada de novos *flits* ao roteador) são ativados sinais, que por sua vez, iniciam outros processos dentro do roteador, isto é, o conjunto de processos operam concomitantemente. A seguir são descritos os processos envolvidos na transfrença de informação.

Processo de espera de chegada de novos flits ao roteador

O processo completo é composto por cinco estados. Este processo é encarregado de receber e armazenar os *flits* que chegam aos portos de entrada do roteador em seus respectivos *buffers*.

Inicialmente o roteador se encontra aguardando a chegada de um novo *flit*. Uma vez que o *flit* atinge o roteador, é verificado o estado da fila. O *flit* aguarda por espaço disponível para ser armazenado. Terminado o processo de armazenagem, o estado da fila é atualizado e mantido em espera para a chegada de outro *flit*.

Processo de identificação dos flits

O processo como um todo é composto por cinco estados. Este processo é encarregado de identificar cada um dos *flits* que compõem o pacote transmitido na rede. O pacote está composto por um *flit* destino, que contém os endereços do componente mestre e do componente escravo; um *flit* tamanho, que indica o número de *flits* da carga útil e o conjunto de *flits* da carga útil, que correspondem à informação.

No primeiro estado, o roteador espera pela petição de seu uso. Este processo é ativado quando o primeiro *flit* do pacote atinge o roteador e é armazenado no *buffer* do porto de entrada, que ocorre no estado S2 do processo de espera de chegada de novos *flits* ao roteador. O primeiro *flit* é identificado como o destino do pacote (*flit* CBÇ). Este estado é mantido até o roteamento do *flit*. Uma vez que o *flit* CBÇ é roteado com sucesso, o *flit* tamanho (*flit* TAM) é identificado e roteado. Os *flits* que conformam a carga útil do pacote (*flits* UTI) são posteriormente também identificados e roteados.

Processo de arbitragem

Este processo é composto por oito estados. O processo é encarregado da concessão do uso do roteador. O processo de arbitragem do roteador implementa o algoritmo *Round Robin*, que concede em cada ciclo a prioridade de comunicação a um dos portos de entrada do roteador.

Em um primeiro momento, o roteador espera pela petição de roteamento de um *flit*. Este processo é ativado quando é identificado o *flit* CBÇ do pacote, no estado S1 do processo de identificação dos *flits*. Posteriormente, é realizada a avaliação de prioridade do uso do roteador, adquirindo um dos estados do S2 ao S6. Por último, o processo informa o desfecho da arbitragem e espera uma nova petição de comunicação.

Processo de roteamento

Composto por sete estados, este processo é encarregado da escolha do porto de saída, utilizado para transferir o pacote. Neste processo é implementado o algoritmo de roteamento, que define o caminho a ser percorrido pelo pacote.

Inicialmente, o roteador espera pela concessão de seu uso. Este processo é ativado no momento da escolha de um porto de entrada do roteador durante o estado S1 do processo de arbitragem. Segundo a informação contida no *flit* CBC, o algoritmo de roteamento se encarrega de selecionar a saída adequada para o pacote. O roteador consulta a disponibilidade desse porto de saída, seja do porto de entrada do roteador vizinho escolhido, seja do componente ligado ao porto local do roteador. Uma vez estabelecida a conexão, o roteador está capacitado a enviar o *flit* CBC do pacote. Confirmado seu correto recebimento, o *flit* TAM e os *flits* UTI podem ser enviados. Finalizado o envio do pacote, o roteador libera a porta de saída.

Modelo de comunicação da rede *HERMES_TEMP*

Com base na descrição dos processos da rede *Hermes*, foi desenvolvido o modelo de comunicação da rede *Hermes_Temp*. A comunicação na rede é definida neste modelo como a transferência da seqüência de *flits* de um pacote completo do mestre ao escravo. O modelo compreende a seqüência de eventos ou ações que ocorrem na transferência de um pacote.

A rede *intrachip* permite a transferência da informação na forma de mensagens devidamente compostas por um conjunto de pacotes entre um módulo mestre e um módulo escravo através de uma seqüência de comutações.

Uma comutação é definida como o envio de um pacote de informação do porto ou do *buffer* de entrada de um roteador até um de seus portos de saída (vide Figura 6.13). O número de comutações simultâneas que ocorrem na rede pode ser igual ao número de roteadores que a constituem.

Figura 6.13 Comutação da informação na rede.

A Figura 6.14 apresenta os intervalos de tempo associados à comutação de um pacote em um roteador da rede.

Figura 6.14 Intervalos de tempo de uma comutação.

A comutação compreende os seguintes eventos:

1. **Pedido de comutação a um roteador:** Este evento é originado por um mestre ou por um outro roteador. Um roteador recebe um sinal de pedido de comutação disparada pela chegada de um cabeçalho a um dos seus portos de entrada.
2. **Verificação do espaço no *buffer* do porto de entrada do roteador:** É verificado o estado atual do *buffer* e este informa se é possível ou não receber o *flit*.

3. **Armazenamento do *flit* no *buffer* de entrada:** Quando existe espaço disponível no *buffer* do porto de entrada, este armazena o *flit* que chegou ao roteador enquanto dispara um sinal para que o processo de arbitragem seja ativado. Concomitante a sua ativação, o espaço no *buffer* é verificado (passo 2) para os outros *flits* que compõem o pacote.
4. **Arbitragem:** Este processo otorga a prioridade de comunicação para um dos portos de entrada do roteador. A arbitragem é concedida segundo o algoritmo *Round Robin*. Uma vez atendida a requisição de comunicação de um porto, este terá a menor prioridade no próximo atendimento.
5. **Roteamento:** Este processo define qual será o porto de saída utilizado para a transferência do pacote (*North*, *South*, *Local*, *East* e *West*). O roteamento é definido por um algoritmo.
6. **Transferência do *flit* CBC:** O *flit* CBC espera o estabelecimento da conexão do roteador com o porto de entrada do roteador vizinho ou do porto ligado ao componente escravo (porto *Local*) para realizar a sua transferência.
7. **Transferência do *flit* TAM:** O *flit* TAM espera a disponibilidade de espaço no *buffer* do porto de entrada do roteador vizinho ou do porto ligado ao componente escravo (porto *Local*) para realizar a sua transferência.
8. **Transferência dos *flits* UTI:** Os *flits* que constituem a carga útil do pacote esperam a disponibilidade de espaço no *buffer* do porto de entrada do roteador vizinho ou do porto ligado ao componente escravo (porto *Local*), para realizar sua transferência. Uma vez terminada a transferência destes *flits*, a conexão estabelecida é interrompida, liberando o porto de saída que estava sendo utilizado. Desta maneira a comutação é finalizada.

A Figura 6.14 ressalta quatro instantes de tempo: E1, E2, E3 e E4. Estes instantes identificam: a petição de transferência de um pacote (E1), a concessão de prioridade máxima de comunicação a um porto de entrada do roteador (E2), a escolha do porto de saída do roteador utilizado na comutação (E3) e o final da comutação (E4).

Os instantes 1 a 8 sinalizam a ocorrência dos eventos relacionados com a comutação de um pacote. Os seguintes intervalos de tempo são definidos:

T_{ve} : Tempo empregado na verificação de espaço no *buffer* do porto de entrada.

T_a : Tempo empregado na armazenagem do pacote no *buffer* do porto de entrada.

T_{la} : Tempo gasto entre a armazenagem do *flit* cabeçalho no *buffer* de entrada e a permissão de uso do roteador (Latência de arbitragem).

T_{lr} : Tempo de execução do algoritmo de roteamento e a habilitação do porto de saída selecionado (Latência de roteamento).

T_h : Tempo empregado na transmissão do *flit* cabeçalho.

T_t : Tempo empregado na transmissão do *flit* tamanho.

T_{ec} : Tempo empregado na transmissão dos *flits* que formam a carga útil do pacote.

O tempo empregado para realizar uma comutação ($T_{comutação}$) é a somatória dos intervalos de tempo delimitados por estes 8 eventos. Vide Equação 6.1.

$$T_{comutação} = \sum T_i \quad \text{Equação 6.1}$$

A comunicação dentro da rede *Hermes* é definida como a transferência de um pacote de informação desde o componente mestre até o componente escravo. Este processo é denominado transação.

Uma transação é definida neste trabalho como o conjunto de comutações que tiveram lugar desde a geração do pacote até atingir o seu destino (vide Figura 6.15). Para completar a transação foram necessárias as comutações do pacote pelos roteadores 01, 11 e 21.

Figura 6.15 Roteamento de informação na rede.

Um exemplo de uma transação que percorreu três roteadores até atingir o seu destino é apresentada na Figura 6.16

Figura 6.16 Transação composta de três comutações.

O início da comunicação é disparada por um sinal de controle (sinal *haveNewFlit*) enviado pelo mestre ao seu roteador local. A transferência é finalizada pelo envio de um sinal de controle (sinal *flitAccepted*) enviado pelo escravo ao mestre. O caminho entre mestre e escravo pode percorrer um ou mais roteadores da rede, cada um realizando a

comutação da informação entre um de seus portos de entrada para um de seus portos de saída.

O tempo empregado para realizar uma transação ($T_{\text{transação}}$) corresponde ao tempo gasto na travessia do caminho do pacote ($T_{\text{travessia}}$), desde sua injeção (T_{IN}) na rede até atingir o destino (T_{OUT}). Vide Equação 6.2).

$$T_{\text{transação}} = T_{\text{OUT}} - T_{\text{IN}}$$

Equação 6.2

Os tempos envolvidos dependem do número de roteadores percorridos, assim como da disponibilidade dos recursos da rede. Os intervalos de tempo envolvidos no modelo da comunicação são apresentados na Figura 6.17. Entre duas transações consecutivas existe um intervalo de tempo chamado *tempo entre transações*.

Figura 6.17 Diagrama de tempos associadas à comunicação dentro da NoC.

As transações não têm necessariamente a mesma quantidade de comutações. O número de comutações é determinado pelo algoritmo de roteamento e as condições de tráfego da rede.

Criação do modelo da rede *HERMES_TEMP*

Uma vez identificados os componentes do modelo *Hermes* que necessitavam ser sincronizados, os arquivos correspondentes ao conjunto de declarações de classe, métodos e atributos dos componentes (arquivos *.h*) foram modificados. Os arquivos

modificados foram os correspondentes ao conjunto de blocos (*INDOOR*, *OUTDOOR*) que constituem os portos de entrada e saída de cada roteador (cinco portos por roteador: *Local*, *North*, *East*, *South* e *West*). Nestes arquivos foi acrescentado um porto de entrada (CLK) que permite a ligação do sinal de controle de sincronismo (relógio). Este sinal de controle passou a formar parte da lista de sensibilidade dos processos descritos nestes dois tipos de blocos.

Modificados os arquivos de declaração de classe dos portos do roteador, são então identificados os arquivos que implementam as classes definidas nestes blocos (arquivos *.cpp*). Nestes arquivos são determinados os oito eventos do modelo de comutação da Figura 6.14, pertencentes aos processos implementados nos componentes da rede: processo de espera de chegada de novos *flits* ao roteador, processo de identificação dos *flits* do pacote, processo de arbitragem e processo de roteamento.

As mudanças feitas nos componentes que constituem o roteador são apresentadas na Figura 6.18.

Figura 6.18 Um dos portos do roteador da rede *Hermes_Temp*.

Ao identificar cada um dos oito eventos (pedido de comutação a um roteador, verificação do espaço no *buffer* do porto de entrada do roteador, armazenamento do *flit* no *buffer* de entrada, arbitragem, roteamento, transferência do *flit* destino, transferência do *flit* tamanho e transferência dos *flits* de carga útil) dentro do modelo não temporizado da rede, o código foi acrescido de uma sentença dependente do sinal de sincronismo (relógio).

O canal *INTRAROUTERCHL* na Figura 6.18, por sua vez, é um módulo que implementa a funcionalidade das interfases *outTointernallf* e *inFrominternallf*. A interface *outTointernallf* define as tarefas que identificam cada um dos *flits* do pacote (CBÇ, TAM, UTI). A interface *inFrominternallf*, por sua vez, define as tarefas que estabelecem a conexão com os outros roteadores da rede ou o componente escravo da comunicação.

O conjunto de tarefas implementadas no canal é sincronizado, de forma a implementar o modelo de comutação apresentado na Figura 6.13. O modelo temporizado do canal envolve a criação de eventos de sincronismo, incorporados ao longo do código. O conjunto de eventos determina o instante de tempo em que deve ser realizada uma das tarefas descritas nas interfases. Os eventos são ativados por ocorrências de ações notificadas imediatamente dentro do canal pelo método *notify()*.

A partir do momento de sua ativação, cada um dos oito eventos do modelo de comutação é anotado pelos monitores ligados à rede. Esta informação é coletada para realizar a posterior quantificação das métricas. Finalmente a avaliação de desempenho pelo projetista é baseada nos valores das métricas calculadas.

A *Hermes_Temp* é uma rede paramétrica que oferece quatro diferentes tipos de topologias (*Grelha*, *Toróide*, *Anel Cordal* e *Circular Entrelaçada*⁴⁶), tamanho dos *buffers* de entrada dos roteadores e quatro algoritmos de roteamento.

⁴⁶ Criada ao longo deste trabalho.

6.3 REQUISITOS DE COMUNICAÇÃO

O conjunto de requisitos de comunicação é derivado das especificações do sistema, no princípio do fluxo do projeto. Estes requisitos constituem o conjunto de objetivos do projeto da rede. A definição dos requisitos de comunicação permite a escolha de métricas que garantam uma adequada avaliação.

Em [GOO05] é descrito um exemplo de apresentação dos requisitos de comunicação. Estes requisitos são expressos como uma lista de conexões entre os diferentes componentes de processamento do sistema. Cada conexão especifica os seguintes parâmetros: largura de banda mínima a ser utilizada, latência máxima permitida, tamanho da rajada e classe de serviço (melhor esforço ou vazão garantida). Na seção 7.1, os requisitos de comunicação utilizados no estudo de caso são expressos sob a forma de padrões de qualidade. Estes estabelecem os parâmetros que definem a qualidade dos resultados das estimativas de desempenho obtidas a partir das simulações.

6.4 MÉTRICAS DE DESEMPENHO

As métricas de desempenho são critérios usados para expressar de forma quantitativa o desempenho de um sistema. As métricas são escolhidas segundo as especificações e requisitos do sistema, já que as informações medidas devem fornecer critérios que permitam realizar uma escolha da melhor instância (configuração definida pelos seus parâmetros) da estrutura de comunicação. Por exemplo, em dispositivos portáteis, a potência média consumida pelo sistema resulta ser um valor crítico, já em aplicações de processamento de vídeo, os parâmetros de desempenho de *throughput* (vazão) e latência são mais importantes [MOR05].

6.4.1 Classificação das métricas

As métricas são baseadas nos serviços fornecidos pelo sistema. Podem ser classificadas em três categorias principais: serviço correto, serviço incorreto e serviço não executado.

Quando é avaliado o serviço correto do sistema, parâmetros como a velocidade, confiabilidade e disponibilidade podem ajudar a descrever o sistema. O estudo do serviço incorreto gera informação dos erros e a suas possíveis fontes, os parâmetros informam sobre a qualidade e a potência do sistema. Por último, o estudo do serviço não executado fornece dados como o tempo de reparo, tempo entre erros e tempo de permanência em falha.

O presente trabalho pretende realizar um estudo do serviço correto do sistema. Na seguinte seção é descrito o conjunto de métricas utilizadas neste trabalho para avaliar o desempenho da rede.

6.4.2 Métricas adotadas

Foi adotado um conjunto de seis (6) métricas de desempenho para avaliar as diferentes configurações da *NoC*. Estas métricas surgem das leis operacionais [RAJ91].

1. Duração média das transações

Esta métrica representa a média do número de ciclos de relógio necessários para transferir um *flit* através da *NoC*, e é dada pela Equação 6.3.

$$Duração_Média = \frac{\sum (tempo_transações)_i}{\sum (Flits_transmitidos)_i} \quad \text{Equação 6.3}$$

Sendo o numerador, a somatória dos tempos correspondentes a todas as transações de cada mestre, e o denominador, o número total de *flits* transferidos ao longo da simulação.

A unidade utilizada na duração média das transações é *ciclos/flits*.

2. Vazão das transações

A vazão (*throughput*) de transações representa o número total de transações realizadas em toda a rede durante a simulação e é dada pela Equação 6.4.

$$Vazão_transações = \frac{\sum (número_transações)_i}{ciclos_simulados} \quad \text{Equação 6.4}$$

A unidade utilizada na vazão das transações é *transações/ciclos*.

3. Largura de banda

Esta métrica representa a quantidade total de *flits* transmitidos em toda a rede ao longo da simulação e é dada pela Equação 6.5.

$$Largura_banda = \frac{\sum (Flits_transferidos)_i}{ciclos_simulados} \quad \text{Equação 6.5}$$

A unidade utilizada na largura de banda é *flits/ciclos*.

4. Tempo de espera médio por roteador

Representa o tempo médio decorrido entre um pacote chegar a um roteador e sua transferência ser realizada. Esta métrica é dada pela Equação 6.6.

$$Tempo_espera_medio = \frac{\sum (Tempo_espera_comutação)_i}{Número_transferências} \quad \text{Equação 6.6}$$

A unidade utilizada no tempo de espera médio por roteador é ciclos/ transferência.

5. Nível de utilização por roteador

Esta métrica representa a participação, em termos de ocupação, de cada roteador na transmissão da informação durante o tempo da simulação e é dada pela Equação 6.7.

$$Utilização_chave = \frac{\sum (Flits_transferidos_por_roteador)_i}{ciclos_simulados} \quad \text{Equação 6.7}$$

A unidade utilizada no nível de utilização por roteador é *flits/ciclo*.

6. Dispersão:

Esta métrica representa a relação entre a porcentagem de pacotes transmitidos que empregaram o caminho mínimo ótimo (menor número de roteadores) e a porcentagem de pacotes transmitidos que empregaram um caminho diferente e é dada pela Equação 6.8. A métrica é usada sempre que seja possível determinar o caminho mínimo percorrido. O caminho depende da natureza do roteamento implementado nos roteadores: Adaptativa ou determinística.

$$Dispersão = \frac{(Pacotes\ caminho\ mínimo)\%}{(Pacotes\ não\ caminho\ mínimo)\%} \quad \text{Equação 6.8}$$

O conjunto de valores que representam a dispersão do pacote são adimensionais.

6.5 DEFINIÇÃO DO CONJUNTO DE SIMULAÇÕES

No trabalho são empregadas um conjunto de configurações da rede *Hermes_Temp*. A rede é constituída por 16 roteadores⁴⁷. As configurações surgem da variação da topologia (*Grelha*, *Toróide*, *Anel Cordal* e *Circular Entrelaçado*⁴⁸) e do algoritmo de roteamento (*XYPuro*, *West First*, *XYAdaptativo Mínimo*, *North Last*, *Anel* e *Circular Entrelaçado*).

- *XYPuro*: Prioriza o atingimento da coordenada X para depois atingir a Y.
- *West First*: Prioriza o uso da porta *West*. Uma vez escolhida outra direção, nunca escolhe esta porta novamente.
- *XYAdaptativo Mínimo*: Prioriza as portas mais próximas do destino do pacote.
- *North Last*: Prioriza qualquer direção que não seja a porta *North*. Uma vez escolhida a porta *North*, esta direção não é mais abandonada.
- *Anel*: Segue o caminho do anel e somente utiliza o caminho central se seu destino está além de um quarto do caminho.
- *Circular entrelaçado*: Prioriza as portas laterais *West* ou *East*.

Nestas configurações, o tamanho dos *buffers* de cada roteador armazena 4 *flits*.

As condições de tráfego surgem das diferentes configurações dos geradores de tráfego, que por sua vez surgem do estabelecimento (paramétrico ou pseudo-aleatório) do *alvo da comunicação*, *tamanho dos pacotes* e *tempo de geração de pacotes*.

A largura dos enlaces da rede *Hermes_Temp* é de 16 bits. O tráfego da rede pode ser paramétrico ou pseudo-aleatório.

⁴⁷ Este tamanho foi escolhido pela sua facilidade de análise.

⁴⁸ Esta topologia foi desenvolvida ao longo do trabalho.

Geradores paramétricos

Nos geradores paramétricos, as características são manipuladas pelo projetista. A manipulação da característica *alvo da comunicação* estabeleceu dois cenários: o cenário *hot spot* e o cenário *transposto*. Estes foram discutidos na seção 6.1.1. O cenário *hot spot* representa o desempenho da rede sob as piores condições de tráfego. O cenário *transposto* permite observar os efeitos do mapeamento dos nodos de processamento à plataforma de comunicação e a influência da configuração do algoritmo de roteamento.

Geradores pseudo-aleatórios

Nos geradores pseudo-aleatórios, os nodos de processamento se comunicam uniformemente com os outros nodos de processamento ligados à rede. Cada mestre cria pacotes com uma distribuição específica do *tamanho do pacote* e do *tempo entre geração de pacotes*. Foi estabelecido o cenário *pseudo-aleatório*.

Nas simulações foi modificado um parâmetro por vez⁴⁹. Por exemplo, quando o *tamanho do pacote* estava sendo mudado (nas simulações S6,S22,S38 da Tabela 6.2), o tempo entre transações permaneceu constante (100 ns). Por sua vez, quando o tempo entre transações era mudado (simulações S54,S71,S87 da Tabela 6.2), o tamanho dos *flits* permaneceu constante (5 *flits*).

A Tabela 6.2 apresenta o conjunto de 277 simulações realizadas.

⁴⁹ A mudança de mais de um parâmetro simultaneamente aumenta a complexidade da análise dos resultados.

Tabela 6.2 Conjunto de simulações realizadas

Topologia		GRELHA / TOROIDE																ANEL CORDAL				CIRCULAR ENTRELACADA			
Algoritmo de roteamento		XYPURO				WEST FIRST				XYADAPTATIVO MINIMO				NORTH LAST				ANEL				CIRCULAR ENTRELACADO			
		Tamanho dos pacotes		Tempo entre transações		Tamanho dos pacotes		Tempo entre transações		Tamanho dos pacotes		Tempo entre transações		Tamanho dos pacotes		Tempo entre transações		Tamanho dos pacotes		Tempo entre transações		Tamanho dos pacotes		Tempo entre transações	
		5	20	100	V.A	25	100	150	V.A	5	20	100	V.A	25	100	150	V.A	5	20	100	V.A	25	100	150	V.A
ALVODA COMUNICAÇÃO	Hot spot	00	51	517	535	549	566	582																	
		01	52	518	534	550	567	583																	
		02	53	519	535	551	568	584																	
		03	54	520	536	552	569	585																	
		10	55	521	537	553	570	586																	
		11	56	522	538	554	571	587																	
		12	57	523	539	555	572	588																	
		13	58	524	540	556	573	589																	
		20	59	525	541	557	574	590																	
		21	510	526	542	558	559	591	598																
		22	511	527	543	560	576	592																	
		23	512	528	544	561	577	593																	
		30	513	529	545	562	578	594																	
		31	514	530	546	563	579	595																	
		32	515	531	547	564	580	596																	
		33	516	532	548	565	581	597																	
	Trasposto																								
	Aleatório																								

6.6 QUANTIFICAÇÃO DAS MÉTRICAS

Os blocos monitores, representados na Figura 6.2, anotaram os valores de interesse que permitiram calcular o conjunto das seis métricas definidas na seção 6.3 e posteriormente realizar a avaliação de desempenho das instâncias da rede *Hermes_Temp*.

O resultado da simulação corresponde aos eventos que ocorreram em 800000⁵⁰ ciclos de relógio. Como resultado é gerado um arquivo, em formato texto, que apresenta uma listagem dos eventos de comunicação que ocorreram na rede (ciclo a ciclo). Foram geradas, em média, 10 linhas por ciclo simulado.

A complexidade da manipulação desta quantidade de informação levou ao desenvolvimento de ferramentas de captura e análise de dados. Foram desenvolvidas duas ferramentas: *Apolo*⁵¹ e *Atenea*⁵².

⁵⁰ Número de ciclos escolhido após realizar um conjunto de simulações que indicou que este número de ciclos era apropriado pela resolução dos resultados.

⁵¹ Deus grego das profecias. Prevê o que acontecerá no futuro.

⁵² Deusa grega das artes e da guerra.

Apolo é uma ferramenta desenvolvida em C++. Esta realiza a leitura do arquivo de simulação, calculando e apresentando em formato texto o conjunto das métricas de desempenho (vide Figura 6.19). Como resultado da análise do arquivo, a ferramenta fornece outras informações adicionais.

A segunda ferramenta, *Atenea*, foi desenvolvida em *Visual Basic*. Esta integra um conjunto de programas que baseados no arquivo intermediário gerado por *Apolo*, apresenta em forma de tabelas as métricas obtidas, representa a distribuição da latência dos pacotes e fornece estatísticas do comportamento da rede. Estas informações podem ser convertidas em forma de gráficos para sua posterior análise.

Para cada condição de tráfego pseudo-aleatório foram realizadas 5 simulações. Cada uma destas simulações foi feita sob as mesmas características de tráfego, sendo unicamente variada a semente do gerador pseudo-aleatório. Esta condição foi aplicada para determinar o intervalo de confiança das métricas obtidas. A confiança definida foi de 95%.

A Figura 6.19 apresenta o fluxo de captura e análise de dados obtidos a partir da simulação do modelo *Hermes_Temp*.

Figura 6.19 Fluxo de captura e análise da avaliação de desempenho

A Figura 6.20 apresenta o formato dos arquivos gerados nas diferentes etapas de simulação e análise do comportamento da rede *Hermes_Temp*.

[illegible]

(b) Arquivo gerado pela ferramenta *Apolo*

Classe	Nível de utilização (%)			
0-20	11,14	0,61	0,61	0,61
21-40	2,66	10,34	1,19	1,19
41-60	0,00	1,74	12,88	2,67
61-80	0,00	0,00	0,00	10,34
81-100	0,00	0,00	0,00	0,00
101-120	2,66	0,00	0,00	0,00
121-140	0,00	0,00	0,00	0,00
141-160	1,14	1,14	1,14	2,67
161-180	0,00	0,00	0,00	0,00

(d) Gráficos gerados a partir das tabelas da ferramenta *Atenea*

Figura 6.20 Arquivos gerados nas diferentes etapas da simulação e análise da rede

O conjunto de ferramentas pode ser reaproveitado para modelos que empregam um sistema similar de comunicação (apresentado na seção 6.1.2). O modelo deve permitir a identificação dos eventos de comunicação.

Em contraste com estudos prévios, que utilizavam métricas globais para descrever o comportamento da rede, neste trabalho as ferramentas desenvolvidas, *Apolo* e *Atenea*, fornecem informações que permitem elucidar o que ocorre também no interior da rede. A avaliação de desempenho não é baseada somente na quantificação da latência e vazão (informações globais da rede), como vem sendo feito até agora. Além destas informações é possível conhecer o comportamento ao interior da rede e informações dos portos de entrada da rede. Esta situação é apresentada na Figura 6.21.

Figura 6.21 Avaliação na rede *intrachip*.

As informações fornecidas pela ferramenta *Apolo* podem ser classificadas segundo a Tabela 6.3. A informação ressaltada na Tabela 6.3 corresponde às métricas utilizadas até agora nos estudos de desempenho que utilizam modelos de alto nível da rede *intrachip*.

A Tabela 6.4 apresenta o tempo médio gasto na simulação da rede, além da filtragem e apresentação dos dados.

Tabela 6.3 Informações fornecidas por *Apolo*

<i>Informações globais</i>	<i>Informações por porto de entrada da rede</i>	<i>Informações ao interior da rede</i>
Número de Pacotes que tráfegaram pela rede	Número de pacotes que entraram na rede por cada porto de entrada	Número de comutações em cada roteador da rede
Número de <i>flits</i> que tráfegaram pela rede	Número de <i>flits</i> que entraram na rede por cada porto de entrada	<i>Tempo de espera médio dos flits para serem comutados em cada roteador da rede (ciclos)</i>
<i>Duração média das transações na rede (ciclos/flit)</i>	Duração média das transações que iniciam nesse porto (ciclos/flit)	<i>Nível de utilização de cada roteador (percentagem)</i>
<i>Vazão das transações na rede (transações/ciclo)</i>	Vazão das transações que iniciam nesse porto (transações/ciclo)	Utilização de cada porto de entrada (5) dentro de cada roteador (percentagem)
<i>Largura de banda (flits/ciclo)</i>	Dispersão dos flits que ingressam nesse porto	Utilização de cada porto de saída (5) dentro de cada roteador (percentagem)
Tempo de atendimento médio (ciclos)		
<i>Dispersão dos flits na rede</i>		

Tabela 6.4 Tempo médio gasto na simulação e filtragem

<i>Padrão de tráfego</i>	<i>Tempo medio de simulação (segundos)</i>	<i>Tempo medio de filtragem e apresentação de dados (segundos)</i>
Transposto	250	45
<i>Hot Spot</i>	230	35
Pseudo-aleatório	900	180

6.7 VALIDAÇÃO

Para realizar a validação do modelo de *Hermes_Temp* foram replicados os experimentos realizados por [TED05] e os resultados foram comparados.

Em [TED05] foi realizada a avaliação de desempenho do modelo RTL (do inglês *Register Transfer Level*) da rede *intrachip Hermes*. Esta descrição foi posteriormente implementada sobre um dispositivo lógico programável (FPGA).

No cenário utilizado para a validação do modelo, cada componente mestre envia 1000 pacotes de 50 flits segundo o padrão de tráfego complemento. No padrão complemento, o nodo de processamento ligado ao roteador com coordenadas $a_{n-1}, a_{n-2}, \dots, a_1, a_0$, no formato binário, se comunica com o nodo de coordenadas $\overline{a_{n-1}}, \overline{a_{n-2}}, \dots, \overline{a_1}, \overline{a_0}$. Sendo $\overline{a_{n-1}}$ o inverso de a_{n-1} .

A Figura 6.22 apresenta o padrão de tráfego complemento sobre a topologia grelha 4x4.

Figura 6.22 Padrão de tráfego complemento.

Foram utilizados dois algoritmos de roteamento: *XYPuro* e *WestFirst*. Foram realizadas dez simulações, onde cada uma das taxas de injeção de pacotes de cada nodo de processamento tiveram valores de 10, 20, 30, 40, 50 e 60% da carga máxima da rede.

Para atingir este valor, o tempo de geração de pacotes mudou.

A Figura 6.23 apresenta os resultados obtidos da simulação do tráfego complemento sobre a rede *Hermes* no trabalho de [TED05].

Figura 6.23 Resultado da simulação de *Hermes* apresentado em [TED05].

Em contrapartida, a Figura 6.24 apresenta os resultados utilizados na validação do modelo da rede *intrachip Hermes_Temp*.

Figura 6.24 Resultados da simulação do modelo *Hermes_Temp*.

A comparação dos resultados evidencia a diferença entre a ordem de grandeza do tempo da latência. Isto é explicável pelo grau de detalhamento de cada um dos modelos. A rede *Hermes* de [TED05] é RTL e *Hermes_Temp* é TLM4. A abstração TLM perde em

precisão dos resultados. Por outro lado, o gráfico descrito pelos dois modelos é semelhante, isto é, seguem a mesma tendência e os dois resultados apresentam a rede grelha que implementa o algoritmo de roteamento XY, como a configuração que atinge os menores tempos de latência quando comparada com a configuração que implementa o algoritmo *West First*.

Baseado neste resultado é possível afirmar que apesar da precisão dos resultados ser necessariamente baixa em relação ao real desempenho da estrutura, devido ao alto nível de abstração do modelo, os resultados são fieis⁵³.

6.8 CONSIDERAÇÕES

O capítulo apresentou a metodologia empregada para realizar a avaliação de desempenho de uma rede *intrachip* a partir de seu modelo em TLM. A metodologia inclui a identificação dos requisitos de comunicação do projeto, a modelagem do ambiente de simulação (modelagem do *SoC*), a definição das métricas de desempenho, a definição do conjunto de simulações, a quantificação das métricas e análise dos dados obtidos.

Seguindo estas fases, o projetista pode avaliar as diferentes configurações da rede e escolher a rede *intrachip* que além de satisfazer os requisitos de comunicação otimize alguma das métricas de custo ou desempenho.

Neste capítulo foi também descrito o modelo da rede *Hermes_Temp*, criada ao longo deste trabalho. A modelagem de *Hermes_Temp* envolveu a adoção de um modelo de comunicação. Neste modelo foram incluídos os diferentes eventos da transferência de

⁵³ A fidelidade permite que possam ser induzidas as mesmas conclusões a partir dos resultados obtidos das simulações em diferentes níveis de abstração.

um pacote do porto de entrada de um roteador até um dos seus portos de saída que o ligam com outro roteador ou com o componente escravo.

A avaliação das diferentes configurações da *Hermes_Temp* é realizada utilizando a quantificação de seis métricas: duração média das transações, vazão das transações, largura de banda, tempo de espera médio por roteador, nível de utilização por roteador e dispersão.

No seguinte capítulo são apresentados os resultados obtidos a partir dos dados gerados nas simulações.

Capítulo 7

AVALIAÇÃO DE DESEMPENHO

O objetivo deste capítulo é apresentar a avaliação de desempenho da rede intrachip. São apresentados os experimentos e os resultados obtidos a partir da análise do conjunto de seis métricas definidas no capítulo 6. São definidos os parâmetros de qualidade utilizados na avaliação de desempenho. O capítulo está dividido em cinco partes. A primeira parte apresenta os padrões de qualidade utilizados para avaliar o desempenho das configurações da rede. As três seções seguintes apresentam os resultados e a avaliação das simulações de 3 padrões de tráfego: transposto, hot spot e pseudo-aleatório. Na quinta parte é apresentado um resumo dos resultados obtidos.

7.1 PADRÕES DE QUALIDADE

Para realizar a avaliação de desempenho da rede é necessária a definição de um conjunto de parâmetros que definam a qualidade dos resultados das estimativas de desempenho obtidas a partir das simulações. Este conjunto de parâmetros é denominado padrão de qualidade.

O padrão de qualidade é utilizado pelo projetista como referência para avaliar a qualidade da rede baseado nos resultados da simulação, isto é, julgar se os resultados são bons ou não.

Para estabelecer o padrão de qualidade foi determinado o valor ótimo de cada uma das seis métricas definidas no capítulo 6. Baseado nestes valores foram definidos diferentes intervalos de qualidade que permitem classificar os resultados obtidos em três categorias: *Bom (B)*, *Aceitável (A)* e *Insatisfatório (I)*.

A seguir são discutidos os padrões de qualidade estabelecidos para cada uma das seis métricas e são definidos seus respectivos intervalos de qualidade.

O estabelecimento do padrão de qualidade é constituído por parâmetros passíveis de mudança segundo o critério do projetista e as exigências do projeto. O padrão é necessário para a avaliação de desempenho, já que este fornece os intervalos que permitem avaliar os resultados das simulações. Nesta seção é apresentado um padrão de qualidade imaginário. A pretensão é ilustrar o procedimento utilizado na avaliação dos resultados. É importante ressaltar que para fins práticos, o padrão de qualidade pode ser modificado sem alterar a metodologia aqui proposta.

1. Duração média das transações (DT)

A duração média das transações é determinada pelo tempo médio gasto para uma transação ser completada. Ver Equação 6.1.

O tempo gasto para completar uma transação depende do número de roteadores que o pacote percorre para atingir o seu destino. Para definir o padrão de qualidade desta métrica, foi definido o caminho ótimo percorrido por cada pacote gerado pelos nodos de processamento ligados à rede e o mínimo tempo gasto para completar a comunicação.

O caminho ótimo é expresso em termo do número mínimo de roteadores percorridos por um pacote desde o roteador ligado ao componente mestre que originou o pacote até o roteador ligado ao componente escravo destino da comunicação. Para determinar o número de ciclos mínimo gasto por cada um dos pacotes é utilizado o modelo de comunicação descrito na seção 6.1.2.

A Figura 7.1 apresenta os eventos envolvidos no modelo de comunicação desenvolvido no capítulo 6 para a comutação de um pacote de 5 *flits* de carga útil. Na Figura 7.1 ARM inclui os três primeiros eventos do modelo de comunicação: Pedido de comutação a um roteador; Verificação do espaço no *buffer* do porto de entrada do roteador e Armazenamento do *flit* no *buffer* de entrada. ARB representa a arbitragem, ROT representa o roteamento do pacote e CBÇ, TAM e UTI representam, respectivamente, a transferência dos *flits* destino, tamanho e de carga útil.

ARM	ARB	ROT	CBÇ	TAM	UTI1	UTI2	UTI3	UTI4	UTIS
-----	-----	-----	-----	-----	------	------	------	------	------

Figura 7.1 Comutação de um pacote de 5 *flits* de carga útil

Para encontrar o valor ótimo, foram consideradas as melhores condições de tráfego para um pacote atingir o seu destino. Isto é, o *flit* que chega ao roteador encontra sempre espaço disponível no *buffer* do porto de entrada do roteador, o arbitro concede o uso do roteador ao pacote que está sendo encaminhado e ao realizar o roteamento o *buffer* do roteador vizinho ou do nodo de processamento (no caso do *flit* se encontrar no roteador que liga ao nodo de processamento destino) tem espaço disponível. A Figura 7.2 apresenta o modelo da transação ótima dos pacotes de 5 *flits* UTI que trafegam pela rede e percorrem até sete roteadores. Cada fila corresponde ao modelo de comutação em um roteador.

Figura 7.2 Modelo da comunicação para completar uma transação (até sete comutações)

A Tabela 7.1 resume o número mínimo de ciclos utilizados para os pacotes atingir o seu destino.

Tabela 7.1 Número mínimo de ciclos segundo o número de roteadores percorridos

Número de roteadores percorridos	Número de ciclos mínimo (5 <i>flits</i>)	Número de ciclos mínimo (20 <i>flits</i>)	Número de ciclos mínimo (100 <i>flits</i>)
1	10	25	105
2	13	28	108
3	16	31	111
4	19	34	114
5	22	37	117
6	25	40	120
7	28	43	123

O modelo de comunicação se sobrepõe conforme os *flits* que conformam o pacote avançam pela rede. Seja o roteador atual r , este pode transferir o *flit* cabeçalho do pacote (CBÇ) ao roteador vizinho $(r+1)$ somente se o *buffer* do porto de entrada deste roteador tem espaço disponível para armazenar o *flit*. Caso contrário, o roteador r permanece na espera desta condição. Esta característica permite que enquanto prossegue a comunicação dos *flits* restantes que conformam o pacote, *flits* tamanho (TAM) e *flits* de carga útil (UTI) no roteador r , a comunicação no roteador seguinte $(r+1)$ se desenvolve. Quando o *flit* CBÇ é transferido ao roteador vizinho, é liberado o espaço que o *flit* estava ocupando no *buffer* do porto de entrada do roteador r . Esta condição permite que enquanto que o *flit* TAM é enviado, os *flits* UTI sejam armazenados no *buffer*, evitando a geração de uma latência adicional causada pelo armazenamento dos outros *flits* no buffer de entrada.

Os intervalos de qualidade que permitem avaliar a rede são definidos na Tabela 7.2 da seguinte forma:

Tabela 7.2 Intervalo de qualidade da Duração Média das transações (DT)

	Percentagem do incremento máximo do valor DT respeito ao valor ótimo
<i>Bom (B)</i>	Menor a 40
<i>Aceitável (A)</i>	40-60
<i>Insatisfatório (I)</i>	Maior a 60

A largura dos intervalos foi escolhida arbitrariamente. Pertencem ao intervalo *bom* os valores da duração média das transações más próximos ao valor ótimo.

2. Vazão das transações (VT)

A vazão das transações é definida como o número de transações completadas ao longo do tempo de simulação (ver Equação 6.2). A vazão ótima resulta do maior número de transações completadas no menor tempo possível.

A vazão depende de dois parâmetros: a quantidade de transações depende do número de componentes geradores de transações, isto é do número de mestres do sistema, e o número de ciclos envolvidos na transação. O tempo gasto em completar uma transação inclui o tempo gasto até atingir o destino da comunicação e o tempo entre pacotes consecutivos.

Como resultado destas considerações o valor da vazão ótima é definida na Equação 7.1.

$$Vazão = \frac{\text{Número de Componentes mestre do sistema}}{\text{Duração da transação ótima} + \text{Tempo de geração de pacotes}} \quad \text{Equação 7.1}$$

Os intervalos de qualidade que permitem avaliar a rede são definidos Tabela 7.3 da seguinte forma:

Tabela 7.3 Intervalo de qualidade da vazão

	Percentagem da diminuição do valor da vazão (VT) respeito ao valor ótimo
Bom (B)	Menor a 30
Aceitável (A)	30-50
Insatisfatório (I)	Maior a 50

Um menor valor de vazão indica a dificuldade da rede na comunicação de informação. Os intervalos foram estabelecidos de forma arbitrária. Estes critérios pretendem ilustrar a definição dos intervalos de qualidade e sua posterior utilização na avaliação de desempenho. Estes intervalos são mais restritos que os definidos para a duração das transações. Esta restrição é resultado da natureza crítica do parâmetro da vazão.

3. Largura de Banda (LB)

A largura de banda foi definida no capítulo 6 como o número de *flits* enviados ao longo do tempo de simulação (ver Equação 6.3). A largura de banda ótima resulta da transmissão do maior número de *flits* no menor tempo possível.

A largura de banda depende da quantidade de *flits* dos pacotes gerados pelos componentes mestre ligados à rede e do tempo gasto por estes *flits* para atingirem o seu destino.

O valor da largura de banda ótima pode ser calculado utilizando a expressão da Equação 7.2.

$$\text{Largura de banda} = \frac{\text{Número de Componentes mestre do sistema}}{\text{Duração da transação ótima} + \text{Tempo de geração de pacotes}} \times \text{Número de flits} \quad \text{Equação 7.2}$$

Quando o tamanho dos pacotes gerados pelos componentes mestre é igual, a expressão da Equação 7.2 se reduz à Equação 7.3.

$$\text{Largura de banda} = \text{Vazão_ótima} \times \text{Número de flits do pacote} \quad \text{Equação 7.3}$$

O número de *flits* do pacote, parâmetro da equação, representa a soma dos *flits* CBC, TAM e o número total de *flits* UTI. Os intervalos de qualidade que permitem avaliar a rede são definidos na Tabela 7.4.

A largura dos intervalos é igual à apresentada para a métrica da vazão. Estes intervalos foram escolhidos arbitrariamente e somente pretendem ilustrar um exemplo de definição do padrão de qualidade. Pode ser modificado segundo os requisitos do projeto.

Tabela 7.4 Intervalo de qualidade da largura de banda

	Percentagem da diminuição do valor da LB respeito ao valor ótimo
<i>Bom (B)</i>	Menor a 30
<i>Aceitável (A)</i>	30-50
<i>Insatisfatório (I)</i>	Maior a 50

4. Tempo de espera médio (TE)

Esta métrica corresponde ao tempo médio que um pacote espera para ganhar a permissão de uso de um roteador da rede. Esta métrica é definida no capítulo 6 pela Equação 6.2.

O valor desta métrica está diretamente relacionada à política de arbitragem implementado no roteador da rede. Na rede *Hermes_Temp* é utilizada a política *round-robin* para conceder o uso do roteador.

Uma vez ganha a prioridade de comunicação por um porto do roteador, o porto que é atendido obtém a menor prioridade de comunicação no ciclo seguinte de arbitragem. Isto é, uma vez atendido este porto terá que esperar que os outros portos do roteador sejam atendidos antes de poder ganhar de novo a permissão para o uso do roteador.

O algoritmo *round-robin* outorga o uso do roteador aos portos segundo a seguinte ordem: NORTH, SOUTH, EAST, WEST e LOCAL. Se um porto do roteador foi atendido, por exemplo o NORTH, este terá que esperar que a prioridade de comunicação seja outorgada aos portos SOUTH, EAST, WEST e LOCAL para voltar a ganhar a prioridade de comunicação. O porto pode ou não fazer uso dessa prioridade, isto é, utilizar ou não o roteador.

A Figura 7.3 apresenta a seqüência ótima de comutação de um pacote. Neste cenário o pacote teria que esperar dois ciclos para utilizar o roteador. O primeiro ciclo (ARM) é utilizado no armazenamento do *flit* destino (CBÇ) e na solicitação de uso do roteador e o segundo ciclo corresponde ao tempo ótimo gasto pela política de arbitragem decidir qual é o porto do roteador que ganhará a prioridade de se comunicar.

1	2	3	4	5	6	7	8	9	10
ARM	ARB	ROT	CBÇ	TAM	UTI1	UTI2	UTI3	UTI4	UTIS

Figura 7.3 Ciclos envolvidos na comutação

Outro cenário é quando o porto termina de ser atendido e solicita o uso de novo ao roteador. A prioridade de comunicação é outorgada aos outros portos do roteador, mas estes não utilizam o roteador. A Figura 7.4 apresenta as alterações nos tempos de comutação de um pacote que solicita a comutação pelo porto NORTH do roteador. Este porto foi recentemente atendido. O pacote do porto NORTH terá que esperar que a arbitragem outorgue de novo o uso do roteador. Se os outros quatro portos não utilizam o roteador, o porto terá que esperar até o ciclo seis (6) para se comunicar.

1	2	3	4	5	6	7	8	9	10	11	12	13	14
ARM	ARB south	ARB east	ARB west	ARB local	ARB north	ROT	CBÇ	TAM	UTI1	UTI2	UTI3	UTI4	UTIS

 ARB

Figura 7.4 Aumento da latência de arbitragem na comutação de um pacote

A Equação 7.4 quantifica o tempo de espera máximo de um porto ($TE_{\text{máximo}}$). Este valor expressa o tempo que o porto NORTH teria que esperar para ganhar a prioridade da comunicação se os outros portos utilizarem o roteador.

$$TE_{\text{máximo}} = \text{Ciclo}(\text{ARM}) + \text{Ciclo}(\text{ARB})_{\text{máximo}} \quad \text{Equação 7.4}$$

Onde o ciclo $\text{ARB}_{\text{máximo}}$ envolve o uso dos quatro portos do roteador. Ver Equação 7.5.

$$TE_{máximo} = Tempo_ARM + \sum_{i=1}^4 (tempo\ de\ comutação)_i$$

Equação 7.5

Os intervalos de qualidade que permitem avaliar o tempo de espera são definidos da seguinte forma:

Bom (B): A este intervalo correspondem os valores do tempo de espera (TE) menores ao tempo correspondente a uma ronda completa de *round-robin* quando nenhum porto utilizou o roteador (6 ciclos).

Aceitável (A): Este intervalo é definido para os valores do tempo de espera quando em uma ronda completa da arbitragem *round-robin*, só um dos portos utilizou o roteador.

Insatisfatório (I): A este intervalo pertencem os valores com um tempo de espera cujo tempo equivale ao uso do roteador por mais de um dos portos de entrada.

Figura 7.5 Intervalos de qualidade para TE

A Figura 7.5 apresenta de forma gráfica os intervalos de qualidade definidos. Este ilustra o tempo de espera que o pacote do porto *North*, que foi atendido recentemente, deve esperar para ser roteado. A avaliação de *bom* (B) é outorgada se nenhum dos outros portos de entrada utilizam a prioridade de comunicação concedida pelo algoritmo de arbitragem *round robin* até que o porto *North* ganhe a prioridade. A avaliação de *aceitável* (A) é outorgada se somente um dos outros portos de entrada utiliza a prioridade de comunicação concedida pelo algoritmo de arbitragem *round robin* antes que o porto *North* ganhe a prioridade. Na figura o porto West utilizou a prioridade de comunicação.

Finalmente, a avaliação de *insatisfatório* (I) é outorgada se mais de um dos outros portos de entrada utilizam a prioridade de comunicação concedida pelo algoritmo de arbitragem *round robin* antes que o porto *North* ganhe a prioridade. Na figura, os portos *West* e *Local* utilizaram a prioridade.

O tempo TE depende do número de *flits* da carga útil. A Tabela 7.5 apresenta os intervalos definidos para os TE correspondentes aos quatro valores de carga útil: 5, 20, 100 e 600 *flits*.

Tabela 7.5 Intervalos de TE definidos para os quatro tamanhos de pacote

Número de <i>flits</i> de carga útil	BOM (B) ciclos	ACEITÁVEL (A) ciclos	INSATISFATORIO (I) ciclos
5	$TM \leq 6$	$6 < TM \leq 14$	$TM > 14$
20	$TM \leq 6$	$6 < TM \leq 29$	$TM > 29$
100	$TM \leq 6$	$6 < TM \leq 109$	$TM > 109$
600	$TM \leq 6$	$6 < TM \leq 609$	$TM > 609$

Um menor tempo de espera do pacote para ser roteado indica uma menor contenção na rede. A definição dos intervalos de qualidade utilizou critérios arbitrários. Estes somente pretendem ilustrar um exemplo. Sua modificação depende das características do projeto e do projetista.

5. *Nível de utilização por roteador (NU)*

Esta métrica expressa a percentagem de pacotes comutados por cada roteador em relação ao número total de pacotes que trafegaram pela rede. A métrica foi definida no capítulo 6 (ver Equação 6.4).

Idealmente, os roteadores deveriam ser igualmente utilizados. Esta característica apresenta vantagens na distribuição da potência consumida pela rede [BEN05]. A Equação 7.6 expressa o nível de utilização ótimo ($NU_{ótimo}$).

$$NU_{\text{ótimo}} = \frac{1}{\text{Número de roteadores}} \times 100$$

Equação 7.6

A homogeneidade do sistema segundo o nível de utilização está relacionado ao algoritmo de roteamento implementado na rede.

Os intervalos de qualidade que permitem avaliar a rede são definidos na Tabela 7.6 da seguinte forma:

Tabela 7.6 Intervalo de qualidade do nível de utilização

	Percentagem da distancia entre o valor NU respeito ao valor ótimo
Bom (B)	Menor a 30
Aceitável (A)	30 – 50
Insatisfatório (I)	Maior a 50

A Figura 7.6 ilustra os intervalos de qualidade definidos para o nível de utilização por roteador. Os valores do nível de utilização podem ser maiores ou menores ao valor ótimo. Por este motivo, os intervalos de qualidade são expressos em termos de distancia. A distancia é definida como a diferenca entre o valor de NU respeito ao valor ótimo.

Figura 7.6 Intervalos de qualidade definidos para NU

Um valor do nível de utilização próximo ao valor ótimo indica que a rede utiliza homogeneamente seus recursos. O intervalo foi definido de forma arbitrária. Este pode ser alterado conforme os requisitos do projeto e o critério do projetista.

6. Dispersão (D)

Esta métrica expressa a percentagem de pacotes que percorreram o número mínimo de roteadores para atingir o seu destino. (ver Equação 6.6). O número mínimo de roteadores pode ser encontrado de forma determinística.

A condição ótima é que todos os pacotes utilizam o mínimo de roteadores para atingir o seu destino, isto é, o valor da dispersão é igual a 100%. O valor desta métrica está estreitamente ligado ao algoritmo de roteamento implementado na rede.

Os intervalos de qualidade que permitem avaliar a rede são definidos na Tabela 7.7 da seguinte forma:

Tabela 7.7 Intervalo de qualidade da dispersão

	<i>Valor da Dispersão</i>
<i>Bom (B)</i>	Maior a 70%
<i>Aceitável (A)</i>	50-70%
<i>Insatisfatório (I)</i>	Menor a 50%

Uma maior dispersão indica que a maioria dos pacotes que trafegam pela rede percorre o caminho ótimo. Estes intervalos foram arbitrariamente definidos. Somente pretende ilustrar um exemplo da definição de intervalo para a métrica de dispersão.

DISCUSSÃO

Os padrões de qualidade são parâmetros, isto é, são estabelecidos pelo projetista. Os valores definidos nesta seção foram escolhidos arbitrariamente para os fins de avaliação neste trabalho. O projetista pode mudar os valores segundo os requisitos de seu projeto e a metodologia de avaliação não muda. O importante é que o avaliador de desempenho tenha um conjunto de definições das características de qualidade que lhe permitam determinar um critério para decidir o que é *bom* e o que é ruim.

As seções seguintes apresentam os diferentes padrões de tráfego que descrevem a comunicação na rede. São apresentados três padrões de tráfego: transposto, *hot spot* e pseudo-aleatório. Estes padrões definem o destino da comunicação dos componentes dentro da rede. Os três padrões de tráfego aqui utilizados não podem ser considerados suficientes para avaliar todas as situações que ocorrem em uma aplicação real. Os três padrões procuram apenas maximizar a cobertura do espaço possível de tráfego em uma rede *intrachip*.

Idealmente, o projetista deve utilizar quatro tipos de tráfego: (i) Tráfego específico da aplicação, (ii) Tráfego parâ métrico, (iii) Tráfego aleatório e (iv) Tráfego de condições extremas, também denominado *corner cases*. Cada um destes tipos de tráfego oferece um nível diferente de cobertura. Como consequência, estas condições não são excludentes mas sim complementares no estudo de desempenho da rede. Estes quatro tipos de tráfego foram discutidos na seção 3.3.

Em contraste aos estudos precedentes sobre a avaliação de desempenho a partir de modelos de alto nível, neste trabalho são fornecidas métricas e informações que permitem conhecer o comportamento ao interior da rede. Como foi apresentado no capítulo 6, nosso estudo inclui três tipos de informações: *informações globais*, *informações dos portos de entrada da rede* e *informações do comportamento ao interior da rede*. Estas informações permitem que o projetista detecte pontos de contenção dentro da rede e facilita o projeto de redes heterogêneas.

As ferramentas desenvolvidas neste trabalho *Apolo* e *Atenea*, (vide seção 6.5) calculam o conjunto das seis métricas definidas no capítulo anterior e entregam um conjunto de estatísticas que descrevem o comportamento tanto da rede como dos roteadores e o comportamento dos componentes mestres.

As seis métricas definidas na seção 6.3.1 podem ser classificadas em métricas externas e métricas internas. As métricas externas descrevem o comportamento da rede como

uma entidade só, enquanto que as métricas internas, desmembram a rede e descrevem o comportamento dos componentes que a constituem. As métricas externas são: *Duração média das transações*, *vazão das transações*, *Largura de banda* e *Dispersão*. As métricas internas são: *Tempo de espera médio por roteador* e *Nível de utilização por roteador*.

7.2 PADRÃO TRANSPOSTO

No padrão de tráfego transposto, cada nodo de processamento ligado a um roteador localizado na coordenada (i,j) da rede, envia informação unicamente ao nodo de processamento ligado ao roteador localizado na coordenada (j,i) . Neste padrão os componentes da diagonal não geram informação, isto é os nodos de processamento ligados aos roteadores: R00,R11, R22 e R33. A Figura 7.7 apresenta o padrão transposto sobre as quatro topologias: *Grelha*, *Toróide*, *Anel Cordal* e *Circular Entrelaçada*.

Os nodos de processamento ligados à rede são constituídos por componentes mestres (por exemplo, processadores) e componentes escravos (por exemplo, memórias). Esta característica permite que cada um dos nodos possa se comportar como mestre, quando envia informação, ou escravo, quando recebe informação de outro nodo de processamento.

O padrão de tráfego transposto é utilizado na literatura no estudo de desempenho de redes [HU04] [MON06]. É reconhecido como um teste padrão (*benchmark*) que permite comparar o desempenho de diferentes configurações da rede.

Figura 7.7 Padrão de tráfego transposto sobre (a) topologia *Grelha*, (b) topologia *Toróide*, (c) topologia *Anel Cordal* e (d) topologia *Circular Entrelaçada*.

7.2.1 PADRÕES DE QUALIDADE PARA O PADRÃO TRANSPOSTO

Nesta seção são apresentados os critérios estabelecidos para a avaliação de desempenho do padrão transposto. Estes valores são obtidos segundo as definições dos intervalos de qualidade da seção anterior.

1. Duração média das transações

No padrão de tráfego transposto, o conjunto de nodos de processamento ligados à rede pode ser dividido segundo o número de roteadores que os pacotes por eles gerados têm que utilizar para chegar no seu destino final. A Tabela 7.8 apresenta o número mínimo de roteadores percorridos segundo a origem do pacote transmitido dentro da rede.

Tabela 7.8 Duração ótima das transações

Topologia	Roteadores origem	Número mínimo de roteadores percorridos	Número mínimo de ciclos gastos			Média ponderada		
			5 flits	20 flits	100 flits	5 flits	20 flits	100 flits
<i>Grelha</i>	R10, R01, R12, R21, R23, R32	3	16	31	111	20	35	115
	R20, R02, R13, R31	5	22	37	117			
	R30, R03	7	28	43	123			
<i>Toróide</i>	R10, R01, R12, R21, R23, R32, R03, R30	3	16	31	111	18	33	113
	R20, R02, R13, R31	5	22	37	117			
Circular	R03, R30	3	16	31	111	18,5	33,5	113,5
	R01, R10, R02, R20, R12, R21, R13, R31, R23, R32	4	19	34	114			
<i>Circular Entrelaçada</i>	R01, R10, R12, R21, R23, R32	3	16	31	111	18	33	113
	R02, R20, R13, R31	4	19	34	114			
	R30, R03	5	22	37	117			

Na tabela, as configurações que obtêm a média ponderada mais baixa são as correspondentes às topologias *Toróide* e *Circular Entrelaçada*. Entretanto, a duração mínima mais alta sempre corresponde à topologia *gralha*.

A avaliação da rede segundo os intervalos de qualidade discutidos na seção 7.1 são apresentados na Tabela 7.9.

Tabela 7.9 Valores ótimos da configuração transposta

Métrica	Número de roteadores percorridos	Duração média das transações (ciclos)	Vazão das transações (transações/1000 ciclos)	Largura de Banda (Flits/1000 ciclos)	Tempo de espera médio (Ciclos/transferência)	Nível de utilização (flits/ciclo)	Dispersão
Intervalo de qualidade							
Valor ótimo	2	13	VT = 94	LB = 658	TE = 2	NU = 6,25%	D = 100%
<i>Bom</i>	2	$DT \leq 19$	$VT \geq 67$	$LB \geq 460$	$TE \leq 6$	$4,3\% \leq NU \leq 8,12\%$	$D \geq 70\%$
	3	$DT \leq 22,4$					
	4	$DT \leq 26,6$					
	5	$DT \leq 30,8$					
	7	$DT \leq 39,2$					
<i>Aceitável</i>	2	$19 < DT \leq 20,8$	$48 \leq VT < 67$	$329 \leq LB < 460$	$6 < TE \leq 14$	$(3,13\% \leq NU < 4,3\%)$ <i>e</i> $(8,12\% < NU \leq 9,3\%)$	$50\% \leq D < 70\%$
	3	$22,4 < DT \leq 25,6$					
	4	$26,6 < DT \leq 30,4$					
	5	$30,8 < DT \leq 35,2$					
	7	$39,2 < DT \leq 44,8$					
Insatisfatório	2	$DT > 20,8$	$VT < 48$	$LB < 329$	$TE > 14$	$(NU < 3,13\%)$ <i>e</i> $(NU > 9,3\%)$	$D < 50\%$
	3	$DT > 25,6$					
	4	$DT > 30,4$					
	5	$DT > 35,2$					
	7	$DT > 44,8$					

7.2.2 RESULTADOS DO PADRÃO TRANSPOSTO

O padrão transposto foi aplicado nas diferentes configurações da rede. As configurações resultam da combinação dos seguintes parâmetros da rede:

- Topologia (*Grelha*, *Toróide*, *Anel Cordal* e *Circular Entrelaçado*).
- Algoritmo de roteamento (*XY*, *West first - WF*, *XY adaptativo - XY_A*, *North last-NL*, *Anel - AN*, *Circular entrelaçado - CE*).

A combinação destes parâmetros originou dez configurações da rede *Hermes_Temp*. As características destas configurações são apresentadas na Tabela 7.10.

Tabela 7.10 Características do conjunto de configurações da rede *Hermes_Temp*

Configuração	Topologia	Algoritmo de roteamento
C1	<i>Grelha</i>	XY
C2		WF
C3		XY_A
C4		NL
C5	<i>Toróide</i>	XY
C6		WF
C7		XY_A
C8		NL
C9	<i>Anel Cordal</i>	AN
C10	<i>Circular Entrelaçada</i>	CE

Ao longo das simulações o tamanho do *buffer* permaneceu constante. Este armazena 4 *flits*. Nesta seção são apresentados os resultados obtidos das simulações do padrão transposto quando o número de *flits* de carga útil que compõem os pacotes que trafegam pela rede é de 5 *flits*. Como resultado da mudança do tempo entre a geração dos pacotes, quatro cenários de tráfego são gerados: 25, 100, 150 ciclos e a forma aleatória aplicando a distribuição exponencial.

Duração média

A Tabela 7.11 apresenta os resultados do valor médio e do desvio padrão respeito a duração das transações obtidas para as dez configurações da rede. No cenário exponencial o intervalo de confiança é 95%.

Tabela 7.11 Duração média das dez configurações da rede *Hermes Temp*

Tempo entre pacotes	25 ciclos		100 ciclos		150 ciclos		Exponencial		
	<i>Med</i>	<i>Desvio</i>	<i>Med</i>	<i>Desvio</i>	<i>Med</i>	<i>Desvio</i>	<i>Mínimo</i>	<i>Med</i>	<i>Máximo</i>
Configuração									
C1	53,2	47,2	45,5	42,6	30,1	37	18,38	21,6	24,81
C2	32,6	50,5	38,6	43,6	28,2	39,1	12,79	16,5	20,2
C3	58,2	51,8	46,6	49,7	45,2	40,9	22,91	31,4	39,88
C4	84,2	49,9	54,3	47,5	47,3	44,2	11,11	17,1	23,08
C5	44,1	48,5	35,1	44,4	33,9	43,5	15,53	20,4	25,26
C6	34,2	53,5	39,8	48,7	39,1	47,9	12,64	17,4	22,15
C7	46,6	34,7	36,9	34,8	35,6	31	17,08	21,8	26,51
C8	78,4	50,1	48,3	48,1	47,5	42,3	27,58	33,2	38,81
C9	33,2	10,6	29,1	7,4	20,3	8,3	13,63	16,3	18,96
C10	40,1	23	33,4	31,4	29,4	8,3	16,35	19,8	23,24

Dos resultados apresentados, as configurações que implementam a topologia *Toróide* C5 e C7 quando comparadas com as configurações de topologia *Grelha* C1 e C3 atingem melhores resultados, diminuindo, respectivamente, 23,03% e 18,73% em relação à duração média. A menor duração média das transações (16,3 ciclos) é atingida com a topologia *Anel Cordal* no cenário exponencial, diminuindo 50,9% em relação ao pior valor alcançado pela configuração C8. O pior tempo de transações teve lugar nas configurações que implementam o algoritmo de roteamento *north last*.

A Figura 7.8 apresenta os valores da duração média para cada uma das dez configurações da rede (C1 – C10). As configurações que implementam o algoritmo de roteamento *north last* (C4 e C8) apresentam os maiores valores de duração média,

enquanto que as topologias *Anel Cordal* e *Circular Entrelaçada*⁵⁴, obtiveram os melhores resultados para os quatro cenários (tempo entre a geração dos pacotes de 25, 100, 150 ciclos e de forma aleatória aplicando a distribuição exponencial).

A diminuição da duração média nas configurações que implementam a topologia *Toróide* (C5 – C8) quando comparadas com as configurações com topologia *Grelha* (C1-C4), é causada pelas rotas alternativas que minimizam o número de roteadores percorridos pelos pacotes que trafegam na rede. Por exemplo os pacotes gerados pelos componentes de processamento ligados aos roteadores R03 e R30 diminuem de 7 a 3 o número de roteadores percorridos. Os novos caminhos aliviam a contenção de informação apresentada nas configurações com topologia *Grelha*. Os maiores valores da duração média são registrados no primeiro cenário, quando o tempo entre pacotes é de 25 ciclos. Isto deve-se à maior quantidade de informação gerada no tempo de simulação, aumentando a contenção na rede. Em contraste, o menor tempo aconteceu quando os geradores de tráfego são pseudo-aleatórios.

Figura 7.8 Duração média das transações

⁵⁴ Criada neste trabalho

A Figura 7.8 inclui os intervalos definidos pelos padrões de qualidade. Os valores destes intervalos podem ser alterados segundo os requisitos do projeto sem mudar a metodologia de avaliação de desempenho.

A partir dos resultados podemos concluir que segundo os padrões de qualidade para a duração das transações, as melhores configurações da rede para um tráfego que segue o padrão transposto são C2, C6, C9 e C10, pois nestes casos o valor da duração média foi sempre *bom* (B) ou *aceitável* (A).

Se quisermos que a duração média seja semelhante para todos os cenários (25, 100, 150 e exponencial), as configurações indicadas são C2, C9 e C10, pois apresentam o menor desvio médio.

Vazão

O maior valor da vazão é atingido pela configuração C9. Esta configuração realiza 82 transações cada 1000 ciclos. A Figura 7.9 apresenta os valores obtidos das simulações correspondentes à vazão.

Figura 7.9 Vazão das transações

A figura mostra que segundo os padrões de qualidade para a vazão, as melhores configurações são C6, C9 e C10. As vazões destas configurações se encontram dentro dos intervalos *bom* (B) ou *aceitável* (A) definidos pelo padrão de qualidade.

As piores condições de vazão foram para as configurações C4 e C8. A baixa vazão é consequência de uma contenção de informação dentro da rede. Isto significa que o número de pacotes que conseguem atingir o seu destino é reduzido.

A figura evidencia a importância da escolha do algoritmo de roteamento na vazão. Nas configurações com topologia *Grelha* e *Toróide*, por exemplo, no cenário exponencial a vazão diminuiu 84% quando foram implementados os algoritmos XY_AM ou WF. Para os cenários paramétricos (25, 100 e 150 ciclos) a redução máxima da vazão foi 22,41%, portanto inferior ao cenário exponencial.

Largura de banda

Nos quatro cenários, como o número de *flits* dos pacotes que trafegam pela rede é fixo (7 *flits*: *flit* CBÇ, *flit* TAM e 5 *flits* de UTI), a largura de banda conserva o padrão exibido pela vazão de transações.

Quando o número de *flits* do pacote foi mudado utilizando a distribuição de Poisson, a largura de banda diminuiu 32% com respeito ao valor de 5 *flits*.

Tempo de espera médio

O tempo de espera médio por roteador (TE) para as configurações que implementam a topologia *Grelha* (C1-C4) são apresentadas na Figura 7.10. Estes valores pertencem ao segundo cenário (tempo entre geração dos pacotes é de 100 ciclos). A configuração C3 apresenta o maior tempo de espera (23,8 ciclos) no roteador R20.

Na configuração C1 os roteadores que pertencem à periferia da rede apresentam os maiores tempos de espera. Este fenômeno está ligado à natureza do padrão de tráfego transposto e ao algoritmo de roteamento XY que implicam em uma contenção da comunicação nos roteadores localizados nas colunas da periferia da rede.

A distribuição destes tempos é diferente para cada uma das configurações. Isto mostra a dependência dos valores do tempo de espera com o algoritmo de roteamento implementado pelo roteador.

Figura 7.10 Tempo de espera médio (TE) por roteador para as configurações com topologia *Grelha*

As diferentes configurações que implementam a topologia *Grelha* não conseguem atingir uma avaliação *bom* (B) ou *aceitável* (A) para o roteador R30.

A Figura 7.11 apresenta o tempo de espera médio por roteador para as configurações que implementam a topologia *Toróide*. Segundo os resultados obtidos, se conserva o ponto de contenção no roteador R30. O tempo de espera aumentou neste ponto, alcançando o valor de 35,04 ciclos.

Nas configurações com topologia *Toróide*, quando comparadas com as da topologia *Grelha*, os pontos críticos se conservam. Os tempos de espera (exceto em R30) diminuíram 23% em média. A redução mais notória é apresentada na configuração C5 com respeito à C1, ambas implementando o algoritmo de roteamento XY.

Figura 7.11 Tempo de espera médio (TE) por roteador para topologia *Toróide*

Por outro lado, os tempos atingidos pela topologia *Anel Cordal*, apresentados na Figura 7.12, são 71,42% menores em relação às configurações discutidas anteriormente. A configuração C9 eliminou os pontos de contenção e atingiu uma distribuição mais homogênea dos tempos de espera.

Figura 7.12 Tempo de espera médio (TE) por roteador para a topologia *Anel Cordal*

A Figura 7.13 apresenta o tempo de espera médio da configuração C10. Esta implementa a topologia *Circular Entrelaçada*. A configuração C10 conseguiu que o tempo médio dos roteadores da rede estivesse na categoria de *bom* (B) ou *aceitável* (A).

Porém a distribuição de tempos seja menos homogênea que os apresentados na topologia *Anel Cordal*, a configuração C10 atingiu valores de tempo menores na maioria dos roteadores.

Figura 7.13 Tempo de espera médio (TE) por roteador para a topologia *Circular Entrelaçada*

A métrica TE referente aos roteadores da rede permite ao projetista observar o impacto do algoritmo de roteamento na rede e determinar, no caso de uma rede heterogênea⁵⁵, quais roteadores deveria evitar para otimizar a duração da transação.

Nível de utilização:

A Tabela 7.12 apresenta o desvio médio dos valores da utilização dos roteadores dentro da rede.

Tabela 7.12 Desvio médio do nível de utilização dos roteadores

Configuração	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
Media	4,6	4,9	5,1	3,2	5,8	5,2	4,8	3,8	5,6	6,1
Desvio médio	3,27	5,43	4,22	11,6	4,07	7,78	5,32	26,4	2,03	1,12

⁵⁵ Uma rede em que os roteadores não são necessariamente iguais.

A métrica de nível de utilização permitiu observar que as configurações C1 a C8 que utilizaram os algoritmos de roteamento XY e XY_A apresentaram uma distribuição de utilização mais homogênea que as que utilizaram os algoritmos WF e NL.

As configurações mais homogêneas são C9 e C10.

Dispersão:

A dispersão indicou que nas configurações C1 e C5 todos os pacotes utilizaram o caminho mínimo. Para as configurações que implementam os algoritmos WF, XY_A e NL a dispersão se conserva constante nas duas topologias: *Grelha* e *Toróide*. Os valores são 86%, 84% e 82% respectivamente. Já nas configurações C9 e C10, que implementam as topologias *Anel Cordal* e *Circular Entrelaçada*, os algoritmos de roteamento implementados obrigam os pacotes a percorrerem o caminho mínimo, fazendo com que a dispersão seja de 100%.

7.2.3 DISCUSSÃO

Usando os valores das métricas para o padrão transposto e os parâmetros de qualidade estabelecidos, foram modificados a topologia e o algoritmo de roteamento da configuração C1. A rede heterogênea resultante é apresentada na Figura 7.14.

Uma tal modificação faz com que a duração média das transações diminua até 43% e a vazão aumente até 52% nos quatro cenários (25, 100, 150 ciclos e exponencial). O tempo de espera nos roteadores da periferia sofreu uma redução de máxima 32,5%.

Estes resultados ilustram o alcance de nossa metodologia de avaliação de desempenho. A partir do conjunto de simulações das diferentes configurações foram obtidas informações pelas ferramentas F1 e F2 (apresentadas no capítulo 6) que permitiram

calcular os valores do conjunto de métricas. Estes valores, junto com os padrões de qualidade definidos, permitiram que o projetista julgasse os resultados e propusesse uma modificação da rede que permitiu melhorar os resultados e atingir valores *bom* (B) dos padrões de qualidade.

Figura 7.14 Configuração C1 modificada

A escolha de uma configuração resulta dos compromissos e requisitos do projeto. O projetista deve avaliar e escolher a que melhor se adapte a suas necessidades. Neste processo, o critério do projetista é utilizado para organizar as prioridades do projeto desenvolvido e escolher uma configuração.

7.3 PADRÃO *HOT SPOT*

No padrão de tráfego *hot spot*, cada componente mestre ligado à rede possui um destino para se comunicar com maior frequência. Neste padrão, o componente mestre, além de se comunicar com o nodo preferencial, pode se comunicar também com outros nodos ligados à rede. No padrão de tráfego transposto o nodo preferencial de todos os componentes mestre ligados à rede é o mesmo.

Este padrão de tráfego é utilizado para simular um sistema que possui um componente de processamento central. O padrão de tráfego *hot spot* é utilizado na literatura no estudo de desempenho de redes como *benchmark* [HU04] [MON06].

A Figura 7.15 apresenta um exemplo de padrão *hot spot*. A espessura das linhas indica a frequência da comunicação. O componente ligado ao roteador R10 se comunica com maior frequência com o nodo ligado ao roteador R21. No padrão *hot spot*, todos os componentes da rede têm este nodo de processamento como seu destino preferencial.

Figura 7.15 Padrão de tráfego *hot spot* sobre (a) topologia Grelha, (b) topologia Toróide, (c) topologia Anel Cordal e (d) topologia Circular Entrelaçada.

Os nodos de processamento desta configuração podem ser componentes mestre ou componentes escravo.

Nesta seção são discutidos os resultados obtidos quando o destino preferencial é o nodo de processamento ligado ao roteador R21. Foram utilizadas as mesmas 10 configurações e os quatro cenários de carga (25, 100, 150 ciclos e a forma aleatória aplicando a distribuição exponencial) discutidos na seção 7.2. Ao longo das simulações o tamanho do *buffer* permaneceu constante. Este armazena 4 *flits*. São apresentados os resultados obtidos das simulações do padrão *hot spot* quando o número de *flits* de carga útil que compõem os pacotes que trafegam pela rede é de 5 *flits*.

7.3.1 PADRÕES DE QUALIDADE PARA O PADRÃO *HOT SPOT*

Os critérios de avaliação de qualidade definidos para o padrão *hot spot* são os seguintes:

1. Duração média das transações

A Tabela 7.13 apresenta os valores ótimos de duração das transações segundo a origem do pacote transmitido e o número de roteadores que deve percorrer para atingir o seu destino.

Os intervalos de qualidade que permitem avaliar a rede que implementa o padrão de tráfego *hot spot* são apresentados na Tabela 7.14.

Tabela 7.13 Duração ótima das transações

Topologia	Roteadores origem	Número mínimo de roteadores percorridos	Número mínimo de ciclos gastos		
			5 flits	20 flits	100 flits
<i>Grelha</i>	R11, R20, R22, R31	2	13	28	108
	R01, R10, R12, R23, R30, R32	3	16	31	111
	R00, R02, R13, R33	4	19	34	114
	R03	5	22	37	117
<i>Toróide</i>	R11, R20, R22, R31	2	13	28	108
	R01, R10, R12, R23, R30, R32	3	16	31	111
	R00, R02, R13, R33	4	19	34	114
	R03	5	22	37	117
Circular	R01, R20, R22	2	13	28	108
	R00, R02, R13, R23	3	16	31	111
	R03, R12, R30, R33	4	19	34	114
	R10, R11, R31, R32	5	22	37	117
<i>Circular Entrelaçada</i>	R13, R20, R22, R23	2	13	28	108
	R11, R12, R30, R31	3	16	31	111
	R03, R10, R32, R33	4	19	34	114
	R00, R01, R02	5	22	37	117

Tabela 7.14 Intervalos de qualidade para o padrão *hot spot*

Métrica	Número de roteadores percorridos	Duração média das transações (ciclos)	Vazão das transações (transações/1000 ciclos)	Largura de Banda (Flits/1000 ciclos)	Tempo de espera médio (Ciclos/ transferência)	Nível de utilização (flits/ciclo)	Dispersão
Intervalo de qualidade							
Valor ótimo	2	13	$VT = 123$	$LB = 861$	$TE = 2$	$NU = 6,25\%$	$D = 100\%$
<i>Bom</i>	2	$DT \leq 19$	$VT \geq 86$	$LB \geq 602$	$TE \leq 6$	$4,3\% \leq NU \leq 8,12\%$	$D \geq 70\%$
	3	$DT \leq 22,4$					
	4	$DT \leq 26,6$					
	5	$DT \leq 30,8$					
<i>Aceitável</i>	2	$19 < DT \leq 20,8$	$61 \leq VT < 86$	$430 \leq LB < 602$	$6 < TE \leq 14$	$(3,13\% \leq NU < 4,3\%)$ <i>e</i> $(8,12\% < NU \leq 9,3\%)$	$50\% \leq D < 70\%$
	3	$22,4 < DT \leq 25,6$					
	4	$26,6 < DT \leq 30,4$					
	5	$30,8 < DT \leq 35,2$					
Insatisfatório	2	$DT > 20,8$	$VT < 61$	$LB < 430$	$TE > 14$	$(NU < 3,13\%)$ <i>e</i> $(NU > 9,3\%)$	$D < 50\%$
	3	$DT > 25,6$					
	4	$DT > 30,4$					
	5	$DT > 35,2$					

7.3.2 RESULTADOS DO PADRÃO *HOT SPOT*

Duração média

Os valores da duração média das transações para as dez configurações da rede *Hermes_Temp* são apresentadas na Tabela 7.15.

Tabela 7.15 Duração média para o padrão *hot spot*

Tempo entre pacotes	25 ciclos		100 ciclos		150 ciclos		Exponencial		
Configuração	<i>Med</i>	<i>Desvio</i>	<i>Med</i>	<i>Desvio</i>	<i>Med</i>	<i>Desvio</i>	<i>Mínimo</i>	<i>Med</i>	<i>Máximo</i>
C1	94,5	348,5	156	130,6	101	285	56,2	63,2	70,2
C2	122	407,5	193	160,6	157	322,5	61,58	68,3	75,01
C3	103	288,5	177	245,7	150	383,9	66,82	76,5	86,17
C4	188	334	202	449	199	262,9	66,96	73,5	80,032
C5	99,9	284,3	186	376,3	147	226,5	80,98	87,6	94,21
C6	145	269,7	191	357,4	188	277,7	88,85	98,4	107,94
C7	161	949,2	194	355,4	189	310,6	89,11	100	110,88
C8	140	293,7	206	414,1	119	284,2	69,43	79,5	89,57
C9	99,3	66,8	91,5	84,3	90,2	62,7	26,63	33,2	39,767
C10	154	166,6	134	81,9	120	19	48,57	51,7	54,82

As informações resultantes mostram um melhor desempenho para as configurações que implementam a topologia gelha quando comparada com as configurações com topologia *Toróide*. A duração média diminui em média 18,2%. A menor duração média das transações (33,2 ciclos) é atingida com a topologia *Anel Cordal* no cenário exponencial. O pior tempo de transações teve lugar na configuração C8.

A Figura 7.16 apresenta a duração média das transações para as diferentes configurações. O alto valor da duração média das transações mostra a intensidade do tráfego do padrão *hot spot*. Neste padrão são gerados pontos de contenção de comunicação, consequência da comunicação dos componentes com um alvo preferencial.

Os tempos obtidos resultaram insatisfatórios (I) segundo o padrão de qualidade estabelecido. Como esperado, uma rede homogênea não é a melhor opção para implementar este tipo de tráfego.

Figura 7.16 Duração média das transações

A Tabela 7.16 apresenta os valores de duração média para as dez configurações quando o alvo preferencial de comunicação era mudado. Estes valores refletem o efeito do mapeamento dos componentes de processamento na rede. Os resultados mostram a intensidade do padrão de tráfego *hot spot* e a dependência da duração média segundo a topologia e algoritmo de roteamento. As configurações C9 e C10, que implementam a topologia *Anel Cordal* e *Circular Entrelaçada*, respectivamente, apresentaram a menor mudança na duração média das transações. Isto é consequência da alta homogeneidade destas configurações

Os valores da duração média da configuração C3, uma rede *intrachip* com topologia *Grelha* e algoritmo de roteamento *XY_Adaptativo*, são os mais variáveis. O valor da duração média aumentou até 30% do valor mínimo (168,4) desta configuração.

Tabela 7.16 Duração media das transações das dez configurações para diferentes localizações do *hot spot*

Localização do <i>hotspot</i>	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
R00	187,8	189,4	212,3	236,6	194,9	198,7	198,6	241,0	90,7	135,2
R01	179,9	156,7	198,6	220,2	172,8	191,0	191,1	241,0	91,6	135,9
R02	178,8	157,2	201,2	238,3	189,6	165,3	189,4	235,8	91,5	134,2
R03	189,6	187,2	215,6	243,1	195,4	196,1	196,1	248,6	90,8	133,3
R10	170,8	171,9	180,2	237,6	194,3	170,4	198,5	250,4	92,1	134,9
R11	159,7	158,2	166,7	222,6	176,5	186,2	197,4	252,4	91,6	133,3
R12	149,2	159,0	176,4	243,5	179,5	179,5	202,4	220,1	91,9	135,2
R13	157,9	171,1	168,4	241,6	160,2	155,2	201,5	247,0	93,0	134,0
R20	160,5	181,4	178,4	234,8	165,3	184,2	199,3	238,8	90,9	134,0
R21	155,5	193,1	177,3	202,0	185,5	191,0	194,0	258,4	91,5	134,2
R22	155,0	186,4	176,3	212,3	176,2	192,6	195,7	243,2	91,6	134,7
R23	162,7	183,0	187,2	241,8	184,5	187,4	189,4	247,2	90,9	134,0
R30	193,9	213,0	189,2	212,7	187,3	158,2	187,5	284,4	90,3	135,2
R31	184,4	208,5	205,1	214,4	176,8	169,4	198,4	249,8	91,3	136,0
R32	180,4	205,5	218,4	216,5	169,4	165,9	198,5	255,4	93,1	134,3
R33	190,1	202,7	212,3	219,9	165,3	178,5	193,3	230,7	92,1	134,3
Desvio médio	14,9	19,0	17,6	13,6	11,2	13,7	4,5	13,9	0,8	0,8

Vazão das transações e Largura de banda

A Figura 7.17 apresenta a vazão das transações para as diferentes configurações. Pelo tráfego intenso dentro da rede, o valor da vazão caiu 34% quando comparada com o padrão de tráfego transposto. Os melhores valores da vazão correspondem ao cenário com tempo entre pacotes exponencial (53 transações a cada 1000 ciclos).

Figura 7.17 Vazão das transações

Os resultados obtidos resultaram insatisfatórios (I) segundo os intervalos de qualidade definidos. Os piores valores foram atingidos pela configuração *Anel Cordal* C9.

Quando o comprimento dos pacotes foi modelado aleatoriamente (pelo processo de Poisson) a largura de banda aumentou 2,5%.

Tempo de espera médio

A Figura 7.18 apresenta o tempo de espera médio dos roteadores das configurações que implementam a topologia *Grelha*.

Figura 7.18 Tempo de espera médio (TE) para os roteadores com topologia *Grelha*

A figura apresenta dois pontos de contenção em R00 e R10 na configuração C4. As distribuições são diferentes segundo o algoritmo de roteamento. A melhor configuração para a topologia *Grelha* é a C2 que supera em média 25,3% dos resultados alcançados pelas configurações C1, C3 e C4.

A Figura 7.19 apresenta o tempo de espera médio para as configurações que implementam a topologia *Toróide*. Os tempos superaram em 38,2% os tempos atingidos com a topologia *Grelha*. O número de pontos que apresentou contenção aumentou (10 pontos).

Figura 7.19 Tempo de espera médio (TE) para os roteadores com topologia *Toróide*

Os tempos de espera médio nos roteadores da configuração C9, que implementa a topologia *Anel Cordal*, são 53% menores do que os atingidos com a topologia *Toróide*. Existem quatro pontos de contenção. Ver Figura 7.20.

Figura 7.20 Tempo de espera médio (TE) para os roteadores com topologia *Anel Cordal*

Os tempos atingidos pela configuração C10 alcançam 80,2 ciclos, embora alguns dos roteadores diminuam o tempo de espera médio em 20,2% quando comparados com a C9. Ver Figura 7.21.

Figura 7.21 Tempo de espera médio (TE) para os roteadores com topologia anel entrelaçada

Nível de utilização:

A Tabela 7.17 apresenta o desvio médio dos valores de utilização dos roteadores dentro da rede.

Tabela 7.17 Desvio médio do nível de utilização dos roteadores

Configuração	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
Media	4,2	5,1	4,8	3,1	5,1	5,3	3,4	3,5	6,2	5,7
Desvio médio	4,88	8,98	8,54	28,3	4,92	11,3	23,4	13,2	4,06	4,34

A menor dispersão pertence à configuração C10. Os roteadores pertencentes à vizinhança de R21 são os mais utilizados na rede. Isto é consequência das características do padrão de tráfego.

Dispersão:

A dispersão indicou que nas configurações C1, C5, C9 e C10 todos os pacotes utilizaram o caminho mínimo. Para as configurações que implementam os algoritmos WF, XY_A e NL os valores da dispersão são 78%, 70% e 88%. A característica do algoritmo de roteamento de procurar os caminhos mais curtos contribuiu para a contenção de informação dentro da rede. Um alto valor na métrica de dispersão indica que a maioria dos pacotes transmitidos utilizou o caminho que utiliza o menor número de roteadores. Este valor deve ser confrontado com a métrica de duração média da transação e a métrica do tempo de espera médio por roteador. Obrigar ao pacote a percorrer o caminho mínimo pode contribuir à gerar contenção dentro da rede.

7.4 PADRÃO PSEUDO-ALEATÓRIO

No padrão de tráfego pseudo-aleatório cada componente mestre ligado à rede se comunica de forma uniforme com os outros nodos de processamento. Isto é, cada componente ligado à rede tem a mesma probabilidade de ser o alvo da comunicação de algum componente mestre ligado à rede. Os nodos de processamento ligados à rede podem ser mestres ou escravos.

Este padrão de tráfego é utilizado para simular as condições de tráfego de sistemas em que a comunicação não é previsível [HU04].

A Figura 7.22 apresenta a característica do padrão pseudo-aleatório. As linhas pontilhadas representam a probabilidade do componente ligado a R12 se comunicar com outro componente ligado à rede. No padrão de tráfego pseudo-aleatório cada mestre pode se comunicar com qualquer escravo ligado à rede com a mesma probabilidade.

Figura 7.22 Padrão de tráfego pseudo-aleatório sobre (a) topologia *Grelha* e (b) topologia *Toróide*, (c) topologia *Anel Cordal* e (d) topologia *Circular Entrelaçada*.

Foram utilizadas as 10 configurações e os quatro cenários de carga (25, 100, 150 ciclos e a forma aleatória aplicando a distribuição exponencial) discutidos na seção 7.2. O tamanho do *buffer* permaneceu constante (4 *flits*). Nesta seção são apresentados os resultados obtidos das simulações do padrão pseudo-aleatório quando o número de *flits* de carga útil que compõem os pacotes que trafegam pela rede é de 5 *flits*.

A Tabela 7.18 apresenta o valor ótimo da duração média segundo o número máximo de roteadores que um pacote pode percorrer para as quatro topologias implementadas.

Tabela 7.18 Duração ótima das transações

Topologia	Número máximo de roteadores percorridos	Número mínimo de ciclos gastos		
		5 <i>flits</i>	20 <i>flits</i>	100 <i>flits</i>
<i>Grelha</i>	7	28	43	123
<i>Toróide</i>	7	28	43	123
Circular	5	22	37	117
<i>Circular Entrelaçada</i>	5	22	37	117

Os intervalos de qualidade que permitem avaliar a rede que implementa este tráfego são apresentados na Tabela 7.19.

Tabela 7.19 Valores ótimos da configuração aleatoria

Métrica	Número de roteadores percorridos	Duração média das transações (ciclos)	Vazão das transações (transações/1000 ciclos)	Largura de Banda (Flits/1000 ciclos)	Tempo de espera médio (Ciclos/transferência)	Nível de utilização (flits/ciclo)	Dispersão
Intervalo de qualidade							
Valor ótimo	2	13	VT = 123	LB=861	TE = 2	NU = 6,25%	D = 100%
<i>Bom</i>	2	$DT \leq 19$	$VT \geq 86$	$LB \geq 602$	$TE \leq 6$	$4,3\% \leq NU \leq 8,12\%$	$D \geq 70\%$
	3	$DT \leq 22,4$					
	4	$DT \leq 26,6$					
	5	$DT \leq 30,8$					
<i>Aceitável</i>	2	$19 < DT \leq 20,8$	$61 \leq VT < 86$	$430 \leq LB < 602$	$6 < TE \leq 14$	$(3,13\% \leq NU < 4,3\%)$ e $(8,12\% < NU \leq 9,3\%)$	$50\% \leq D < 70\%$
	3	$22,4 < DT \leq 25,6$					
	4	$26,6 < DT \leq 30,4$					
	5	$30,8 < DT \leq 35,2$					
<i>Insatisfatório</i>	2	$DT > 20,8$	$VT < 61$	$LB < 430$	$TE > 14$	$(NU < 3,13\%)$ e $(NU > 9,3\%)$	$D < 50\%$
	3	$DT > 25,6$					
	4	$DT > 30,4$					
	5	$DT > 35,2$					

7.4.1 RESULTADOS DO PADRÃO PSEUDO-ALEATÓRIO

Duração média

Os valores da duração média das transações para as dez configurações da rede *Hermes_Temp* que implementam o tráfego pseudo-aleatório são apresentados na Tabela 7.20.

Tabela 7.20 Duração média para o padrão pseudo-aleatório

Tempo entre pacotes	25 ciclos		100 ciclos		150 ciclos	
Configuração	<i>Med</i>	<i>Desvio</i>	<i>Med</i>	<i>Desvio</i>	<i>Med</i>	<i>Desvio</i>
C1	35,7	9,0	101	67,8	43,2	43,0
C2	22,1	6,8	98,4	82,3	77,7	47,6
C3	32,5	19,2	98	79,4	51,8	40,1
C4	47,4	25,1	135	109,7	78,6	52,5
C5	90,9	82,5	106	49,4	93,8	78,5
C6	86,5	31,5	100	54,1	90,3	85,4
C7	72,7	117,3	67,2	106,7	55,6	36,3
C8	85,5	130,9	99,9	109,9	43	50,6
C9	35,7	53,2	50,4	46,6	45	13,7
C10	37,6	17,4	44,7	26,7	40,1	11,8

O menor valor da duração média é atingido pela configuração C2 no cenário 25 ciclos. Somente as configurações C9 e C10 conseguiram estar dentro dos intervalos *bom* (B) e *aceitável* (A) definidos no padrão de qualidade. Os tempos atingidos diminuiram 27% em relação aos tempos atingidos no padrão de tráfego *hot spot*. A Figura 7.23 apresenta os valores da duração média das transações para os diferentes cenários (25, 100, 150 ciclos e exponencial).

Figura 7.23 Duração média das transações

Vazão

O maior valor da vazão é atingido pela configuração C1 no cenário 25 ciclos. Esta configuração realiza 78 transações a cada 1000 ciclos. Esta mesma configuração apresentou os piores valores nos outros dos cenários (100 e 150 ciclos). A Figura 7.24 apresenta os valores obtidos para a vazão.

Figura 7.24 Vazão das transações

Em geral, nenhuma configuração atingiu um valor de vazão dentro da categoria *bom* (B) definida no padrão de qualidade. Configurações como a C6, C7, C9 e C10 obtiveram os valores mais homogêneos nos três cenários (25, 100 e 150 ciclos).

Largura de banda

Nos três cenários, a largura de banda conserva o padrão exibido pela vazão de transações.

Quando o número de *flits* do pacote foi mudado utilizando a distribuição de Poisson, a largura de banda diminuiu 32% com respeito ao valor de 5 *flits*.

A Figura 7.25 apresenta o tempo de espera médio dos roteadores das configurações que implementam a topologia *Grelha*. Na figura pode ser observado um padrão de tempo de espera por coluna da rede que faz com que aumente o tempo nos roteadores da coluna (por exemplo, do R00 ao R03). Este padrão é mantido para todas as configurações.

Figura 7.25 Tempo de espera médio (TE) para os roteadores com topologia *Grelha*

A Figura 7.26 apresenta o tempo de espera médio para as configurações com topologia *Toróide*. Nestas configurações a duração das transações aumentou 23% quando comparadas com a topologia *Grelha*.

Figura 7.26 Tempo de espera médio (TE) para os roteadores com topologia *Toróide*

O tempo de espera médio da configuração C9 é apresentado na Figura 7.27. Nesta configuração existe um ponto de contenção em R32. O tempo de espera médio foi 35% menor do que os tempos atingidos com as topologias *Toróide* e *Grelha*.

Figura 7.27 Tempo de espera médio (TE) para os roteadores com topologia *Anel Cordal*

A Figura 7.28 apresenta o tempo de espera médio da configuração que implementa a topologia *Circular Entrelaçada*. Esta atingiu os menores resultados. Os tempos de espera nos roteadores são mais homogêneos que nas outras configurações. O tempo de espera médio máximo alcança 34,3 ciclos.

Figura 7.28 Tempo de espera médio (TE) para os roteadores com topologia *Circular Entrelaçada*

Nível de utilização:

A Tabela 7.21 apresenta o desvio médio dos valores da utilização dos roteadores dentro da rede.

Tabela 7.21 Desvio médio do nível de utilização dos roteadores

Configuração	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
Media	5,3	4,7	4,8	4,1	5,2	4,7	5,1	3,4	5,4	6,2
Desvio médio	5,26	7,21	8,34	6,67	6,39	7,05	8,2	5,83	4,35	3,21

O nível de utilização mais homogêneo, indicado na Tabela 7.21 com o menor desvio médio, corresponde à configuração C10.

Dispersão:

A dispersão indicou que nas configurações C1, C5, C9 e C10 todos os pacotes utilizaram o caminho mínimo. Para as configurações que implementam os algoritmos WF, XY_A e NL os valores da dispersão são 84%, 90% e 83%. A característica do algoritmo de roteamento de procurar os caminhos mais curtos contribuiu para a contenção de informação dentro da rede.

7.5 RESUMO

A Tabela 7.22 apresenta um resumo dos melhores e piores valores obtidos em cada métrica com os diferentes padrões de tráfego. Nem sempre os melhores valores são os mesmos para todas as métricas.

Tabela 7.22 Melhores e piores configurações

<i>Padrão de tráfego</i>	<i>Tempo entre pacotes (Ciclos)</i>	<i>Duração média</i>		<i>Vazão das transações</i>		<i>Largura de banda</i>		<i>Tempo de espera médio</i>		<i>Nível de utilização</i>		<i>Dispersão</i>	
		M	P	M	P	M	P	M	P	M	P	M	P
Transposto	25	C2	C4	C9	C4	C9	C4	C10	C4	C10	C8	C9,C10	C4,C8
	100	C9	C4	C9	C4	C9	C4	C9	C4	C10	C8	C9,C10	C4
	150	C9	C8	C9	C4	C9	C4	C10	C3	C9	C4	C9,C10	C7
	Exp.	C9	C8	C9	C4	C9	C4	C10	C4	C10	C8	C9,C10	C7
Hot spot	25	C1	C4	C10	C9	C10	C9	C10	C6	C10	C8	C9,C10	C3
	100	C9	C8	C10	C9	C10	C9	C9	C10	C9	C4	C9,C10	C3
	150	C9	C4	C10	C9	C10	C9	C10	C7	C10	C4	C9,C10	C6
	Exp.	C9	C7	C10	C9	C10	C9	C10	C8	C10	C4	C9,C10	C7
Aleatório	25	C2	C5	C1,C2	C8	C1,C2	C8	C2	C8	C10	C2	C9,C10	C4
	100	C10	C4	C7	C3	C7	C3	C10	C8	C10	C3	C9,C10	C4
	150	C10	C5	C10	C8	C10	C8	C10	C8	C10	C5	C9,C10	C4

M: Melhor valor

P: Pior Valor

A duração média decresce conforme aumenta o tempo entre transações e decresce o tamanho dos pacotes. Isto deve-se ao menor número de conflitos entre elementos que competem por um roteador para intervalos mais longos entre transações. No caso do aumento do tamanho do pacote o efeito é contrário, pois maiores tamanhos de pacotes originam maiores tempos em completar uma transação. A vazão das transações está ligada à efetividade do algoritmo de roteamento em evitar gerar pontos de contenção. Os algoritmos que impõem ao pacote o caminho mínimo restringem o tráfego dentro da rede. Os pontos de contenção diminuem a vazão.

7.6 EXEMPLO DE ESCOLHA DE CONFIGURAÇÃO

Nesta seção é apresentado um exemplo de como o projetista pode utilizar o conjunto de métricas para escolher uma configuração da rede *intrachip*. São utilizados os resultados obtidos a partir da simulação das dez configurações possíveis descritas na Tabela 7.10.

São recopilados os valores das seis métricas aqui definidas quando as dez configurações implementam os três padrões de tráfego: *Transposto*, *hot spot* e *aleatório*.

A Tabela 7.23 e Tabela 7.24 apresentam os quatro cenários definidos: 25, 100, 150 ciclos e exponencial. Estes cenários expressam o tempo entre a geração dos pacotes dos componentes mestre ligados à rede.

Para ilustrar a escolha de uma configuração será admitindo a seguinte especificação⁵⁶:

- A duração de uma transação não pode ser maior de 160 ciclos.
- A vazão de dados nunca pode ser inferior a 16 ciclos.
- O tempo de espera de cada roteador deve ser mínimo.
- Os roteadores devem ser igualmente utilizados.
- A desviação da utilização dos roteadores deve ser inferior a 10.

Para escolher a configuração da rede *intrachip* que satisfaça estes requisitos, cada item da especificação e comparado com os resultados das tabelas. As configurações que satisfazem a condição da especificação geram um subconjunto de possíveis soluções. Finalmente, do subconjunto de soluções é escolhida a configuração da rede *intrachip*.

Da análise da primeira condição são geradas a Tabela 7.25 e Tabela 7.26. As regiões sombreadas e com os números ressaltados em negrita constituem o subconjunto de possíveis soluções. Por outra parte, as regiones somente sombreadas representam os

⁵⁶ Estes dados são inventados e são utilizados sómente com fins didácticos.

valores que não cumprem com a primeira condição, portanto, as configurações que geram estes valores são descartadas do subconjunto de possíveis condições.

No subconjunto de soluções resultado da análise da primeira condição inclui as configurações C1, C9 e C10.

Aplicando a segunda condição são geradas a Tabela 7.27 e Tabela 7.28. As convenções estabelecidas na primeira análise são iguais.

No subconjunto de soluções resultado da análise da primeira condição inclui as configurações C1 e C10.

Finalmente são analisadas a terceira e quarta condição. Estas exigem a comparação entre estas duas soluções. Da análise são geradas a Tabela 7.29 e Tabela 7.30.

A configuração escolhida é a C10. Esta configuração corresponde a topologia *Anel Cordal*. Os resultados obtidos a partir da simulação desta topologia sob os três padrões de tráfego satisfazem as condições impostas pela especificação.

Este exemplo ilustra como com ajuda das métricas pode ser diminuído o espaço de soluções até encontrar a configuração que satisfaça um conjunto de requisitos ou especificações.

Tabela 7.23 Resumo dos resultados das dez configurações que implementam os três padrões de tráfego para os cenários 25 e 100 ciclos

		<i>Transposto</i>						<i>Hot Spot</i>						<i>Aleatório</i>					
		<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>
<i>Cenário 25 ciclos</i>	<i>C1</i>	53	42	294	11	4,3	100	95	19	133	66	6,3	100	36	77	539	7,4	7,3	100
	<i>C2</i>	33	43	301	10	6,4	86	122	21	147	74	12	88	22	77	539	7,2	10	78
	<i>C3</i>	58	38	266	12	3,8	85	103	18	126	73	13	76	32	64	448	10	9,8	80
	<i>C4</i>	84	3	21	15	17	83	188	20	140	113	20	78	47	47	329	17	4,4	76
	<i>C5</i>	44	47	329	7,9	3,2	100	100	22	154	127	7,9	100	91	22	154	31	5,3	100
	<i>C6</i>	34	57	399	10	6,2	86	145	21	147	148	9,9	87	86	32	224	27	6,2	82
	<i>C7</i>	47	48	336	7,9	7,8	85	161	22	154	146	18	80	73	50	350	12	7,6	90
	<i>C8</i>	78	4	28	15	16	83	140	20	140	109	22	78	85	15	105	45	8,2	84
	<i>C9</i>	33	66	462	8,6	2,3	100	99	12	84	27	6,1	100	36	65	455	8,6	6,3	100
	<i>C10</i>	40	52	364	7,9	1,1	100	154	28	196	26	5,3	100	38	56	392	9,1	4,4	100
<i>Cenário 100 ciclos</i>	<i>C1</i>	45	40	280	13	3,3	100	156	17	119	71	4,9	100	100	16	112	25	5,3	100
	<i>C2</i>	39	41	287	10	5,4	86	193	19	133	79	9	78	98	16	112	26	7,2	84
	<i>C3</i>	47	37	259	13	4,2	84	177	16	112	75	8,5	70	98	13	91	24	8,3	90
	<i>C4</i>	54	2	14	16	12	81	202	19	133	116	28	88	135	21	147	26	6,7	83
	<i>C5</i>	35	45	315	9	4,1	100	186	20	140	128	4,9	100	106	15	105	26	6,4	100
	<i>C6</i>	40	58	406	10	7,8	86	191	20	140	148	11	78	100	42	294	24	7,1	84
	<i>C7</i>	37	49	343	9,9	5,3	84	194	20	140	155	23	71	67	48	336	24	8,2	90
	<i>C8</i>	48	3	21	16	26	82	206	18	126	115	13	88	100	33	231	35	5,8	93
	<i>C9</i>	29	64	448	9	2	100	92	9	63	32	4,1	100	50	30	210	22	4,4	100
	<i>C10</i>	33	50	350	9	1,1	100	134	25	175	29	4,3	100	45	35	245	14	3,2	100

Tabela 7.24 Resumo dos resultados das dez configurações que implementam os três padrões de tráfego para os cenários 150 ciclos e exponencial

		<i>Transposto</i>						<i>Hot Spot</i>						<i>Aleatório</i>					
		<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>
<i>Cenário 150 ciclos</i>	<i>C1</i>	30	39	273	13	8,2	100	101	14	98	84	8,6	100	43	12	84	31	6,2	100
	<i>C2</i>	28	40	280	11	6,3	90	157	17	119	83	10	90	78	14	98	28	8,5	86
	<i>C3</i>	45	36	252	18	3,2	92	150	14	98	79	14	90	52	12	84	28	9,6	88
	<i>C4</i>	47	2	14	14	19	88	199	16	112	127	30	89	79	31	217	23	8,6	92
	<i>C5</i>	34	44	308	9,4	7,2	100	147	18	126	134	8,4	100	94	17	119	23	10	100
	<i>C6</i>	39	56	392	13	14	87	188	18	126	156	9,4	87	90	30	210	19	9,9	86
	<i>C7</i>	36	52	364	10	8,7	86	189	19	133	158	15	89	56	38	266	26	10	88
	<i>C8</i>	48	3	21	17	19	91	119	16	112	121	24	88	43	11	77	43	6,3	92
	<i>C9</i>	20	62	434	10	2,1	100	90	6	42	46	7,8	100	45	29	203	22	4,7	100
	<i>C10</i>	29	49	343	9,3	2,6	100	120	23	161	34	5,3	100	40	41	287	9,4	3,6	100
<i>Cenário Exponencial</i>	<i>C1</i>	22	73	511	10	8,1	100	63	42	294	43	6,3	100						
	<i>C2</i>	16	76	532	9,7	9,6	92	68	44	308	40	7,5	88						
	<i>C3</i>	31	12	84	16	6,8	94	77	45	315	40	8,6	93						
	<i>C4</i>	17	2	14	18	20	88	73	41	287	43	20	96						
	<i>C5</i>	20	77	539	8,6	8,4	100	88	46	322	39	7,8	100						
	<i>C6</i>	17	82	574	11	9,5	87	98	44	308	40	8,6	89						
	<i>C7</i>	22	16	112	15	8,4	86	100	47	329	37	11	86						
	<i>C8</i>	33	3	21	17	23	900	80	44	308	60	19	91						
	<i>C9</i>	16	87	609	7,5	5,2	100	33	37	259	22	6,3	100						
	<i>C10</i>	20	83	581	5,2	3	100	52	53	371	10	5,1	100						

Tabela 7.25 Subconjunto de soluções gerado a partir da análise da primeira condição nos cenários 25 e 100 ciclos

		<i>Transposto</i>						<i>Hot Spot</i>						<i>Aleatório</i>					
		<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>
<i>Cenário 25 ciclos</i>	<i>C1</i>	53	42	294	11	4,3	100	95	19	133	66	6,3	100	36	77	539	7,4	7,3	100
	<i>C2</i>	33	43	301	10	6,4	86	122	21	147	74	12	88	22	77	539	7,2	10	78
	<i>C3</i>	58	38	266	12	3,8	85	103	18	126	73	13	76	32	64	448	10	9,8	80
	<i>C4</i>	84	3	21	15	17	83	188	20	140	113	20	78	47	47	329	17	4,4	76
	<i>C5</i>	44	47	329	7,9	3,2	100	100	22	154	127	7,9	100	91	22	154	31	5,3	100
	<i>C6</i>	34	57	399	10	6,2	86	145	21	147	148	9,9	87	86	32	224	27	6,2	82
	<i>C7</i>	47	48	336	7,9	7,8	85	161	22	154	146	18	80	73	50	350	12	7,6	90
	<i>C8</i>	78	4	28	15	16	83	140	20	140	109	22	78	85	15	105	45	8,2	84
	<i>C9</i>	33	66	462	8,6	2,3	100	99	12	84	27	6,1	100	36	65	455	8,6	6,3	100
	<i>C10</i>	40	52	364	7,9	1,1	100	154	28	196	26	5,3	100	38	56	392	9,1	4,4	100
<i>Cenário 100 ciclos</i>	<i>C1</i>	45	40	280	13	3,3	100	156	17	119	71	4,9	100	100	16	112	25	5,3	100
	<i>C2</i>	39	41	287	10	5,4	86	193	19	133	79	9	78	98	16	112	26	7,2	84
	<i>C3</i>	47	37	259	13	4,2	84	177	16	112	75	8,5	70	98	13	91	24	8,3	90
	<i>C4</i>	54	2	14	16	12	81	202	19	133	116	28	88	135	21	147	26	6,7	83
	<i>C5</i>	35	45	315	9	4,1	100	186	20	140	128	4,9	100	106	15	105	26	6,4	100
	<i>C6</i>	40	58	406	10	7,8	86	191	20	140	148	11	78	100	42	294	24	7,1	84
	<i>C7</i>	37	49	343	9,9	5,3	84	194	20	140	155	23	71	67	48	336	24	8,2	90
	<i>C8</i>	48	3	21	16	26	82	206	18	126	115	13	88	100	33	231	35	5,8	93
	<i>C9</i>	29	64	448	9	2	100	92	9	63	32	4,1	100	50	30	210	22	4,4	100
	<i>C10</i>	33	50	350	9	1,1	100	134	25	175	29	4,3	100	45	35	245	14	3,2	100

Tabela 7.26 Subconjunto de soluções gerado a partir da análise da primeira condição nos cenários 25 e 100 ciclos

		<i>Transposto</i>						<i>Hot Spot</i>						<i>Aleatório</i>					
		<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>
<i>Cenário 150 ciclos</i>	<i>C1</i>	30	39	273	13	8,2	100	101	14	98	84	8,6	100	43	12	84	31	6,2	100
	<i>C2</i>	28	40	280	11	6,3	90	157	17	119	83	10	90	78	14	98	28	8,5	86
	<i>C3</i>	45	36	252	18	3,2	92	150	14	98	79	14	90	52	12	84	28	9,6	88
	<i>C4</i>	47	2	14	14	19	88	199	16	112	127	30	89	79	31	217	23	8,6	92
	<i>C5</i>	34	44	308	9,4	7,2	100	147	18	126	134	8,4	100	94	17	119	23	10	100
	<i>C6</i>	39	56	392	13	14	87	188	18	126	156	9,4	87	90	30	210	19	9,9	86
	<i>C7</i>	36	52	364	10	8,7	86	189	19	133	158	15	89	56	38	266	26	10	88
	<i>C8</i>	48	3	21	17	19	91	119	16	112	121	24	88	43	11	77	43	6,3	92
	<i>C9</i>	20	62	434	10	2,1	100	90	6	42	46	7,8	100	45	29	203	22	4,7	100
	<i>C10</i>	29	49	343	9,3	2,6	100	120	23	161	34	5,3	100	40	41	287	9,4	3,6	100
<i>Cenário Exponencial</i>	<i>C1</i>	22	73	511	10	8,1	100	63	42	294	43	6,3	100						
	<i>C2</i>	16	76	532	9,7	9,6	92	68	44	308	40	7,5	88						
	<i>C3</i>	31	12	84	16	6,8	94	77	45	315	40	8,6	93						
	<i>C4</i>	17	2	14	18	20	88	73	41	287	43	20	96						
	<i>C5</i>	20	77	539	8,6	8,4	100	88	46	322	39	7,8	100						
	<i>C6</i>	17	82	574	11	9,5	87	98	44	308	40	8,6	89						
	<i>C7</i>	22	16	112	15	8,4	86	100	47	329	37	11	86						
	<i>C8</i>	33	3	21	17	23	900	80	44	308	60	19	91						
	<i>C9</i>	16	87	609	7,5	5,2	100	33	37	259	22	6,3	100						
	<i>C10</i>	20	83	581	5,2	3	100	52	53	371	10	5,1	100						

Tabela 7.27 Subconjunto de soluções gerado a partir da análise da segunda condição nos cenários 25 e 100 ciclos

		<i>Transposto</i>						<i>Hot Spot</i>						<i>Aleatório</i>					
		<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>
<i>Cenário 25 ciclos</i>	<i>C1</i>	53	42	294	11	4,3	100	95	19	133	66	6,3	100	36	77	539	7,4	7,3	100
	<i>C2</i>	33	43	301	10	6,4	86	122	21	147	74	12	88	22	77	539	7,2	10	78
	<i>C3</i>	58	38	266	12	3,8	85	103	18	126	73	13	76	32	64	448	10	9,8	80
	<i>C4</i>	84	3	21	15	17	83	188	20	140	113	20	78	47	47	329	17	4,4	76
	<i>C5</i>	44	47	329	7,9	3,2	100	100	22	154	127	7,9	100	91	22	154	31	5,3	100
	<i>C6</i>	34	57	399	10	6,2	86	145	21	147	148	9,9	87	86	32	224	27	6,2	82
	<i>C7</i>	47	48	336	7,9	7,8	85	161	22	154	146	18	80	73	50	350	12	7,6	90
	<i>C8</i>	78	4	28	15	16	83	140	20	140	109	22	78	85	15	105	45	8,2	84
	<i>C9</i>	33	66	462	8,6	2,3	100	99	12	84	27	6,1	100	36	65	455	8,6	6,3	100
	<i>C10</i>	40	52	364	7,9	1,1	100	154	28	196	26	5,3	100	38	56	392	9,1	4,4	100
<i>Cenário 100 ciclos</i>	<i>C1</i>	45	40	280	13	3,3	100	156	17	119	71	4,9	100	100	16	112	25	5,3	100
	<i>C2</i>	39	41	287	10	5,4	86	193	19	133	79	9	78	98	16	112	26	7,2	84
	<i>C3</i>	47	37	259	13	4,2	84	177	16	112	75	8,5	70	98	13	91	24	8,3	90
	<i>C4</i>	54	2	14	16	12	81	202	19	133	116	28	88	135	21	147	26	6,7	83
	<i>C5</i>	35	45	315	9	4,1	100	186	20	140	128	4,9	100	106	15	105	26	6,4	100
	<i>C6</i>	40	58	406	10	7,8	86	191	20	140	148	11	78	100	42	294	24	7,1	84
	<i>C7</i>	37	49	343	9,9	5,3	84	194	20	140	155	23	71	67	48	336	24	8,2	90
	<i>C8</i>	48	3	21	16	26	82	206	18	126	115	13	88	100	33	231	35	5,8	93
	<i>C9</i>	29	64	448	9	2	100	92	9	63	32	4,1	100	50	30	210	22	4,4	100
	<i>C10</i>	33	50	350	9	1,1	100	134	25	175	29	4,3	100	45	35	245	14	3,2	100

Tabela 7.28 Subconjunto de soluções gerado a partir da análise da segunda condição nos cenários 150 ciclos e exponencial

		<i>Transposto</i>						<i>Hot Spot</i>						<i>Aleatório</i>					
		<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>
		<i>C1</i>	<i>C2</i>	<i>C3</i>	<i>C4</i>	<i>C5</i>	<i>C6</i>	<i>C7</i>	<i>C8</i>	<i>C9</i>	<i>C10</i>	<i>C1</i>	<i>C2</i>	<i>C3</i>	<i>C4</i>	<i>C5</i>	<i>C6</i>	<i>C7</i>	<i>C8</i>
<i>Cenário 150 ciclos</i>	<i>C1</i>	30	39	273	13	8,2	100	101	14	98	84	8,6	100	43	12	84	31	6,2	100
	<i>C2</i>	28	40	280	11	6,3	90	157	17	119	83	10	90	78	14	98	28	8,5	86
	<i>C3</i>	45	36	252	18	3,2	92	150	14	98	79	14	90	52	12	84	28	9,6	88
	<i>C4</i>	47	2	14	14	19	88	199	16	112	127	30	89	79	31	217	23	8,6	92
	<i>C5</i>	34	44	308	9,4	7,2	100	147	18	126	134	8,4	100	94	17	119	23	10	100
	<i>C6</i>	39	56	392	13	14	87	188	18	126	156	9,4	87	90	30	210	19	9,9	86
	<i>C7</i>	36	52	364	10	8,7	86	189	19	133	158	15	89	56	38	266	26	10	88
	<i>C8</i>	48	3	21	17	19	91	119	16	112	121	24	88	43	11	77	43	6,3	92
	<i>C9</i>	20	62	434	10	2,1	100	90	6	42	46	7,8	100	45	29	203	22	4,7	100
	<i>C10</i>	29	49	343	9,3	2,6	100	120	23	161	34	5,3	100	40	41	287	9,4	3,6	100
<i>Cenário Exponencial</i>	<i>C1</i>	22	73	511	10	8,1	100	63	42	294	43	6,3	100						
	<i>C2</i>	16	76	532	9,7	9,6	92	68	44	308	40	7,5	88						
	<i>C3</i>	31	12	84	16	6,8	94	77	45	315	40	8,6	93						
	<i>C4</i>	17	2	14	18	20	88	73	41	287	43	20	96						
	<i>C5</i>	20	77	539	8,6	8,4	100	88	46	322	39	7,8	100						
	<i>C6</i>	17	82	574	11	9,5	87	98	44	308	40	8,6	89						
	<i>C7</i>	22	16	112	15	8,4	86	100	47	329	37	11	86						
	<i>C8</i>	33	3	21	17	23	900	80	44	308	60	19	91						
	<i>C9</i>	16	87	609	7,5	5,2	100	33	37	259	22	6,3	100						
	<i>C10</i>	20	83	581	5,2	3	100	52	53	371	10	5,1	100						

Tabela 7.29 Subconjunto de soluções gerado a partir da análise da terceira condição nos cenários 25 e 100 ciclos

		<i>Transposto</i>						<i>Hot Spot</i>						<i>Aleatório</i>					
		<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>
<i>Cenário 25 ciclos</i>	<i>C1</i>	53	42	294	11	4,3	100	95	19	133	66	6,3	100	36	77	539	7,4	7,3	100
	<i>C2</i>	33	43	301	10	6,4	86	122	21	147	74	12	88	22	77	539	7,2	10	78
	<i>C3</i>	58	38	266	12	3,8	85	103	18	126	73	13	76	32	64	448	10	9,8	80
	<i>C4</i>	84	3	21	15	17	83	188	20	140	113	20	78	47	47	329	17	4,4	76
	<i>C5</i>	44	47	329	7,9	3,2	100	100	22	154	127	7,9	100	91	22	154	31	5,3	100
	<i>C6</i>	34	57	399	10	6,2	86	145	21	147	148	9,9	87	86	32	224	27	6,2	82
	<i>C7</i>	47	48	336	7,9	7,8	85	161	22	154	146	18	80	73	50	350	12	7,6	90
	<i>C8</i>	78	4	28	15	16	83	140	20	140	109	22	78	85	15	105	45	8,2	84
	<i>C9</i>	33	66	462	8,6	2,3	100	99	12	84	27	6,1	100	36	65	455	8,6	6,3	100
	<i>C10</i>	40	52	364	7,9	1,1	100	154	28	196	26	5,3	100	38	56	392	9,1	4,4	100
<i>Cenário 100 ciclos</i>	<i>C1</i>	45	40	280	13	3,3	100	156	17	119	71	4,9	100	100	16	112	25	5,3	100
	<i>C2</i>	39	41	287	10	5,4	86	193	19	133	79	9	78	98	16	112	26	7,2	84
	<i>C3</i>	47	37	259	13	4,2	84	177	16	112	75	8,5	70	98	13	91	24	8,3	90
	<i>C4</i>	54	2	14	16	12	81	202	19	133	116	28	88	135	21	147	26	6,7	83
	<i>C5</i>	35	45	315	9	4,1	100	186	20	140	128	4,9	100	106	15	105	26	6,4	100
	<i>C6</i>	40	58	406	10	7,8	86	191	20	140	148	11	78	100	42	294	24	7,1	84
	<i>C7</i>	37	49	343	9,9	5,3	84	194	20	140	155	23	71	67	48	336	24	8,2	90
	<i>C8</i>	48	3	21	16	26	82	206	18	126	115	13	88	100	33	231	35	5,8	93
	<i>C9</i>	29	64	448	9	2	100	92	9	63	32	4,1	100	50	30	210	22	4,4	100
	<i>C10</i>	33	50	350	9	1,1	100	134	25	175	29	4,3	100	45	35	245	14	3,2	100

Tabela 7.30 Subconjunto de soluções gerado a partir da análise da terceira condição nos cenários 150 ciclos e exponencial

		<i>Transposto</i>						<i>Hot Spot</i>						<i>Aleatório</i>					
		<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>	<i>M1</i>	<i>M2</i>	<i>M3</i>	<i>M4</i>	<i>M5</i>	<i>M6</i>
<i>Cenário 150 ciclos</i>	<i>C1</i>	30	39	273	13	8,2	100	101	14	98	84	8,6	100	43	12	84	31	6,2	100
	<i>C2</i>	28	40	280	11	6,3	90	157	17	119	83	10	90	78	14	98	28	8,5	86
	<i>C3</i>	45	36	252	18	3,2	92	150	14	98	79	14	90	52	12	84	28	9,6	88
	<i>C4</i>	47	2	14	14	19	88	199	16	112	127	30	89	79	31	217	23	8,6	92
	<i>C5</i>	34	44	308	9,4	7,2	100	147	18	126	134	8,4	100	94	17	119	23	10	100
	<i>C6</i>	39	56	392	13	14	87	188	18	126	156	9,4	87	90	30	210	19	9,9	86
	<i>C7</i>	36	52	364	10	8,7	86	189	19	133	158	15	89	56	38	266	26	10	88
	<i>C8</i>	48	3	21	17	19	91	119	16	112	121	24	88	43	11	77	43	6,3	92
	<i>C9</i>	20	62	434	10	2,1	100	90	6	42	46	7,8	100	45	29	203	22	4,7	100
	<i>C10</i>	29	49	343	9,3	2,6	100	120	23	161	34	5,3	100	40	41	287	9,4	3,6	100
<i>Cenário Exponencial</i>	<i>C1</i>	22	73	511	10	8,1	100	63	42	294	43	6,3	100						
	<i>C2</i>	16	76	532	9,7	9,6	92	68	44	308	40	7,5	88						
	<i>C3</i>	31	12	84	16	6,8	94	77	45	315	40	8,6	93						
	<i>C4</i>	17	2	14	18	20	88	73	41	287	43	20	96						
	<i>C5</i>	20	77	539	8,6	8,4	100	88	46	322	39	7,8	100						
	<i>C6</i>	17	82	574	11	9,5	87	98	44	308	40	8,6	89						
	<i>C7</i>	22	16	112	15	8,4	86	100	47	329	37	11	86						
	<i>C8</i>	33	3	21	17	23	900	80	44	308	60	19	91						
	<i>C9</i>	16	87	609	7,5	5,2	100	33	37	259	22	6,3	100						
	<i>C10</i>	20	83	581	5,2	3	100	52	53	371	10	5,1	100						

Capítulo 8

CONCLUSÕES

Neste capítulo é apresentado um resumo das principais contribuições do trabalho aqui desenvolvido, além de suas possíveis implicações e desdobramentos no desenvolvimento de futuros trabalhos.

Nos últimos anos, a estrutura de comunicação rede *intrachip* ganhou importância e destaque devido aos requisitos de comunicação das novas aplicações. Surge então, uma série de desafios para o projetista: Quais das possíveis configurações da rede satisfazem as especificações? Como escolher uma dessas configurações? Qual se adapta melhor ao projeto?

Frente a estes desafios, o presente trabalho propõe uma metodologia de avaliação de desempenho da rede *intrachip* a partir de seu modelo em alto nível de abstração TLM. As vantagens desta metodologia residem na possibilidade de fazer uma rápida previsão do desempenho, permitindo escolher uma configuração da rede durante as primeiras etapas do fluxo do projeto do sistema. Esta previsão e escolha permitirão a redução do tempo e custo de desenvolvimento do projeto. Apesar da precisão dos resultados ser necessariamente baixa em relação ao real desempenho da estrutura (devido ao alto nível de abstração do modelo), nossa opinião é que a fidelidade das estimativas permitirá obter resultados confiáveis que compensarão a falta de precisão.

A metodologia inclui: a identificação dos requisitos de comunicação do projeto a partir do modelo do *SoC*, a modelagem da rede *intrachip*, a definição das métricas de desempenho, a definição do conjunto de simulações, a quantificação das métricas e análise dos dados obtidos.

Seguindo estas etapas, o projetista pode avaliar diferentes configurações e escolher a rede *intrachip* que satisfaça os requisitos de comunicação e otimize alguma das métricas de custo ou desempenho.

Como estudo de caso, foi realizada a avaliação de dez diferentes configurações da rede *intrachip Hermes_Temp*. Esta rede, modelada em TLM (*Transaction Level Modelling*) temporizado, foi desenvolvida a partir do modelo não temporizado da rede *intrachip Hermes*. Foram acrescentadas três topologias: *Toróide*, *Anel cordal* e *Circular Entrelaçada* e quatro algoritmos de roteamento: XY, XY adaptativo para a topologia toróide, Anel cordal e Entrelaçado.

Para avaliar o comportamento da *Hermes_Temp*, foram feitas simulações de um SoC constituído de componentes mestre, componentes escravo e da rede *Hermes_Temp*. Os componentes mestre do SoC foram representados utilizando-se diferentes geradores de tráfego. Estes geradores surgem da variação aleatória e/ou paramétrica de três características: destino do pacote, largura do pacote e tempo de geração entre pacotes.

A variação destes três parâmetros origina três padrões de tráfego: Transposto, *Hot spot* e aleatório. Devido à quantidade de informação obtida a partir das simulações, foram desenvolvidas duas ferramentas: *Apolo* e *Atenea*. Estas ferramentas de aquisição e análise de dados permitem calcular métricas de desempenho. As métricas aqui adotadas são: duração média das transações, vazão das transações, largura de banda, tempo de espera médio por roteador, nível de utilização por roteador e dispersão.

Em contraste com os estudos de avaliação de desempenho em alto nível existentes na literatura, este trabalho inclui métricas de desempenho que permitem elucidar o que ocorre no interior da rede. Esta característica permite que o projetista possa detectar pontos de contenção dentro da rede e facilita o projeto de redes heterogêneas.

A qualidade dos resultados obtidos foi definida de acordo com um padrão. Este padrão ilustrou como as informações geradas podem ser utilizadas na escolha de uma

configuração da rede. O padrão pode ser alterado de acordo com os requisitos do projeto e o critério do projetista. A mudança do padrão de qualidade não altera a metodologia proposta.

Adicionalmente, ao longo deste trabalho, foi criada uma nova topologia denominada *Circular entrelaçada*. A criação desta topologia foi motivada pelos avanços nas tecnologias de integração e desenvolvimento das técnicas de projeto de circuitos 3D [BAL06],[FUJ06]. A topologia Circular entrelaçada é apresentada na Figura 8.1. Esta é baseada em dois anéis que interconectam, respectivamente, os roteadores pares e os roteadores ímpares. O algoritmo de roteamento foi descrito de tal forma que é priorizado o encaminhamento por cada um destes anéis.

Figura 8.1 Topologia *Circular Entrelaçada* desde (a) representada em dois dimensões e (b) na sua construção Tridimensional

Esta estrutura é mais homogênea que as propostas anteriores e apresentou um bom desempenho. A sua implementação posterior estará condicionada a uma futura análise de custo.

Além disso, foi evidenciada a capacidade das ferramentas de análise na avaliação de estruturas de diferentes naturezas.

Em etapas futuras, pretendemos aumentar o conjunto de métricas, a fim de expandir a capacidade de avaliação para outros serviços da rede, tais como, serviço incorreto, não executado e serviços de segurança.

Planejamos também avaliar o desempenho da rede usando modelos em outros níveis de abstração TLM e modelos analíticos. O estudo de caso inclui padrões de tráfego paramétricos, aleatórios e reais.

BIBLIOGRAFIA

- [BEI03] R. Beidas, J. Zhu. "Performance efficiency of context-flow System-on-Chip platform". In ICCAD 2003: 356-362
- [BEN01] L. Benini, G. De Micheli, "Powering networks on chips", IEEE/ACM ISSS Symposium, 2001.
- [BEN02] Benini, L. De Micheli, G. "Networks on chips: a new SoC paradigm". Computer, Volume: 35(1), Jan. 2002, pp. 70-78.
- [BLA04] D. Black, J. Donovan. "SystemC: From the ground up". Kluwer Academic Publishers. New York, 2004.
- [BOL04] E. Bolotin, I. Cidon, R. Ginosar, A. Kolodny. "QNoC: QoS architecture and design process for network on chip". Journal of system architecture. 2004.
- [CHI04] D. Ching, P. Schaumont, I. Verbauwhede. "Integrated modelling and generation of reconfigurable NoC". In Proceedings, 18th IPDPS. 2004
- [GUE99] P. Guerrier, A. Greiner. "A Scalable Architecture for System-On-Chip Interconnections". In: SAME99, 1999.
- [GUE00] P. Guerrier, A. Greiner, "A generic architecture for on-chip packet-switched interconnections", in: Design Automation and Test in Europe (DATE'00), March 2000, pp. 250–256.
- [HEN00] A. Hemani, A. Jantsch, S. Kumar, A. Postula, J. Oberg, M. Millberg, D. Lindqvist, "Network on chip: An architecture for billion transistor era", In Proceeding of the IEEE NorChip Conference, November 2000.
- [HEN04] J. Henkely, W. Wolfz, S. Chakradhary. "On-chip networks: A scalable, communication-centric embedded system design paradigm". In: Proceedings of the 17th International Conference on VLSI Design (VLSID'04).
- [HOR01] M. Horowitz, R. Ho, K. Mai, "The future of wires". In Proceedings of the IEEE, April 2001, pp. 490– 504.
- [LAH00] K. Lahiri, A. Raghunathan, S. Dev. "Performance Analysis of Systems With Multi-Channel Communication Architectures". In: Proc. ICCAD 2000
- [LIN03] M. Link, J. Kim, N. Vijaykrishnan, R. Das. "Network-on-Chip (NoC) Architectures: A Resource-Constrained Perspective". International Symposium on Low Power Electronics and Design (ISLPED'03), Seoul, Korea, August, 2003.
- [MOR03] F. Moraes, A. Mello, L. Möller, L. Ost, N. Calazans. "A Low Area Overhead Packet-switched Network on Chip: Architecture and Prototyping". 2003
- [MOR05] F. Moraes, N. Calazans, A. Mello, L. Möller, L. Ost. "HERMES: an infrastructure for low area overhead packet-switching networks on chip". In: Integration, the VLSI journal 2005.
- [RAJ91] J. Raj. "The Art of Computer System Performance Analysis: Techniques for Experimental Design, Measurement, Simulation and Modeling", John Wiley & Sons, INC., 1991.

Outras Referências

- [ABD97] H. Ben-Abdallah, I. Lee. "The integrated specification and analysis of functional, temporal and resource requirements". Requirements engineering 1997. Proceedings of the third IEEE international symposium on. 1997.
- [AGO03] G. Agosta, F. Bruschi, D. Sciuto. "Static analysis of transaction-level models". In: DAC 2003, 2003, Anaheim, California, USA.

- [AHO06] T. Ahonen, H. Bin, J. Nurmi. "Network-on-Chip Generation and Optimization Tool". NoC Workshop. DATE06.2006
- [AMB] ARM. "AMBA home page". Online: www.arm.com/products/solutions/AMBAHomePage.html.
- [AUG05] I. Augé, F. Pétrot, F. Dnet, P. Gomez. "Platform-Based design from parallel C specifications". IEEE transactions on computer-aided design of integrated circuits and systems. 2005.
- [ANJ04] K. Anjo, Y. Yamada, M. Koibuchi. "Black-Bus: A New Data-Transfer technique using local address on Networks-on-Chips". Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS'04).
- [BAL06] J. Balachandran, S. Brebels, G. Carchon, M. Kuijk, W. Raedt, B. Nauwelaers, E. Beyne. "Extended Global Interconnect Architecture for Nano-CMOS technologies". NoC Workshop. DATE06.2006
- [BAN04] N. Banerjee, P. Vellanki, K. Chatha. "A Power and Performance Model for Network-on-Chip Architectures". In Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'04).
- [BER04] D. Bertozzi, L. Benini. "XPIPES: A Network-on-Chip architecture for gigascale Systems-on-Chip". In IEEE circuits and systems magazine. 2004.
- [BER05] D. Bertozzi, A. Jalabert, S. Murali, R. TAMhankar, S. Stergiou, L. Benini, G.D. Micheli. "NoC synthesis flow for customized domain specific multiprocessor System-on-Chip". In IEEE Computer Society. 2005.
- [BHO05] P. Bhojwani, R. mahapatra, E. Jung, T. Chen. "A heuristic peak power onstrained design for Network-on-Chip (NoC) based multimode systems". VLSID 2005.
- [BIP] Projeto Fênix Brazil-ip, <http://brazilip.cin.ufpe.br/fenix/>
- [CAL03] M. Caldari, M. Conti, M. Coppola, s. Curaba, L. Pieralisi, C. Turchetti. "Transaction-Level Models for AMBA Bus Architecture using SystemC 2.0". In: Design, automation and test in europe. 2003.
- [CHA99] H. Chang, L. Cooke, M. Hunt, G. Martin, A. McNelly, L. Tood."Surviving the SoC revolution: a guide to platform-based designed". Kluwer Academic Publishers Group. 1999.
- [COC02] A. Cochrane. "SystemC, transaction level modeling simplifies capture". online: <http://www.commsdesign.com/showArticle.jhtml?articleID=16500866>. 2002.
- [COP06] M. Coppola, C. Pistrutto, R. Locatelli, A. Scndurra. "STNoCTM: An Evolution towards MPSoC Era". NoC Workshop. DATE06.2006
- [COR] IBM. "CoreConnect Bus Architecture". Online: www.ibm.com/chips/products/coreconnect.
- [DAL01] W. Dally, B. Towles, "Route packets, not wires: On-chip interconnection networks". In Proceeding of the 38th Design Automation Conference, June 2001, pp. 684–689.
- [DON04] A. Donlín. "Transaction Level Modeling: Flows and Use Models". CODES ISS'04. September 2004. Suecia.
- [DUA02] J. Duato, J. Fernandez, J. Garcia. "Improving the performance of real-time communication services on high-speed LANs under toolology changes". In: Proceedings of the 27th Annual IEEE Conference on Local Computer Networks (LCN.02).
- [DAY83] Day, J.D.; Zimmermman, H. "The OSI reference model". Proceedings of IEEE, Volume: 71, Dec. 1983, pp. 1334-1340.
- [EIS04] N. Easley, L. Peh. "High-level power analysis fo on-chip networks". Cases04. 2004.
- [EMA95] E. Emam, N. Madhavji. "Measuring the sucesso f requirements engineering processes". ISRE IEEE 1995.
- [FEN05] online: <http://toledo.inf.pucrs.br/~gaph/Projects/Fenix/Fenix.html>
- [FUJ06] S. Fujita, K. Abe, ^a Nomura, T. Lee. "Novel performance of three-dimensional (3D) On-chip Crossbar Bus using non-silicon-transistors". NoC Workshop. DATE06.2006
- [GAJ94] D. Gajski, F. Vahid, S. Narayan, J. Gong. "Specification and design of embedded systems". Prentice Hall 1994.
- [GER 05] A. Gerstlauer, D. Shin, R. Dömer, D. Gajski. " System-level communication Modeling for Network-on-Chip Synthesis". ASO DAC. 2005.
- [GOM98] M. Gomez, V. Santonja. "Self-similarity in I/O workload: analysis and modeling" Workload Characterization: Methodology and Case Studies, 1998.

- [GOO05] K. Goossens, J. Dielissen, O. Prakash, S. González, A. Rădulescu, E. Rijpkema. “*A Design Flow for Application-Specific Networks on Chip with Guaranteed Performance to Accelerate SoC design and verification*”. Proceedings of the design, Automation and Test in Europe Conference and Exhibition (DATE’05). 2005.
- [GUP97] S. Gupta, Y. Zorian. “*Introducing Core-Based System Design*”. In: IEEE Design and Test of Computers, vol. 14-(4), October-December 1997, pp. 15-25.
- [GUP03] S. Gupta, N. Dutt, A. Nicolau. “*SPARK : A High-Level Synthesis Framework For Applying Parallelizing Compiler Transformations*”. In: International Conference on VLSI Design, VLSI’03, 2003, 6 p.
- [GRU04] M. Grünwald, J. Niemann, M. Portmann, U. Rückert. “*A mapping strategy for resource-efficient network processing on multiprocessor SoCs*”. Proceedings of the Design, Automation and Test in Europe Conference and Exhibition DATE’04.
- [GSE] Grupo de Sistemas Eletrônicos Integrados e Software Aplicado. www.lme.usp.br/gseis/index.html.
- [HAD03] A. haddad. “*Efect of the Traffic Bursts in the Network Queue*”. Tesis de maestrado. Rice University. Texas, 2003.
- [HAV02] Haverinen, A., Leclercq, M., Weyrich, N. e Wingard, D. “*SystemC based SoC Communication Modeling.2002*”.
- [HU03] J. Hu, R. Marculescu. “*Exploiting the Routing Flexibility for Energy/Performance aware mapping of regular NoC Architectures*”. Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE’03).
- [HU04] J. Hu, R. Marculescu. “*Energy-Aware communication and task scheduling for Networks-on-Chip architectures under Real-Time constraints*”. Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE’04).
- [JAL04] A. Jalabert, S. Murali, L. Benini, G. De Micheli. “*XpipesCompiler: A tool for instantiating application specific Networks on Chip*”. Proceedings of the Design, Automation and Test in Europe Conference and Exhibition DATE’04.
- [JER05] A. Jerraya. W. Wolf. “*Multiprocessor Systems-on-Chips*”. Morgan Kaufmann publishers.2005.
- [JIN03] R. Jindal, K. Jain. “Recycle RTL test benches to verify Transaction level IP model”. In: <http://www.commsdesign.com/showArticle.jhtml?articleID=16501130>. 2003.
- [KED02] H. Keding. “*Transaction Level Modeling (TLM) with systemC*”. In: SNUG Paris. 2002.
- [KOG 03] T. Kogel, M. Doerper, A. Wieferink, R. Leupers, G. Ascheid, H. Meyr, S. Goossens. “*A modular simulation Framework for architectural exploration of On-Chip interconnection Networks*”. CODES+ ISS’03. California, USA. 2003.
- [KOG04] T. Kogel, H. Meyr. “*Heterogeneous MP-SoC: The solution to Energy-Efficient signal processing*”. Proceedings of the DAC 2004. San Diego, California, USA.
- [KRO02] A. Kroll. “*Transaction Level Modeling (TLM) with SystemC and CoCentric System Studio*”. SNUG Boston. 2002.
- [KUL04] G. Dattatreya, S. Kulkarni “*Performance of Communication Networks Fielding Bursty Data Traffic*”. Annual Review of Communications, 2004.
- [KUM02] S. Kumar, A. Jantsch, J. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, A. Hemani. “*A Network on chip architecture and design methodology*”. Proceedings of the IEEE computer society annual symposium on VLSI. 2002
- [LAH99] K. Lahiri, A. Raghunathan, S. Dey, “*Fast Performance Analysis of Bus-Based System-on-Chip Communication Architectures*”, in Proc. Int. Conf. Computer-Aided Design, pp 566-572, Nov. 1999.
- [LAH01] K. Lahiri, A. Raghunathan, S. Dey, “*Evaluation of the Traffic-Performance Characteristics of System-on-Chip Communication Architectures*”, The 14th International conference on VLSI design, 2001.
- [LEI 03] T. Lei, S. Kumar. “*Algorithms and tools for Network on Chip based system design*”. Proceedings of the 16 Symposium on Integrated Circuits and Systems Design (SBCCI’03).
- [LIA00] J. Liang, S. Swaminathan, R. Tessier, “*aSOC: A scalable, single-chip communications architecture*”, in: IEEE International Conference on Parallel Architectures and Compilation Techniques, October 2000, pp. 37-46.
- [MÄÄ06] S. Määttä, J. Nurmi. “*SystemC simulation Model of a Flexible Network-on-Chip*”. NoC Workshop. DATE06.2006.

- [MAD 03] J. Madsen, S. Mahadevan, K. Virk, M. Gonzalez. "Network-on-Chip Modeling for System-level multiprocessor simulation". Proceedings of the 24th IEEE International Real-Time Systems Symposium (RTSS'03).
- [MAR06] P. Martin, J. Leclerc. "Methodology, benefits and tools for NoC topology exploration" NoC Workshop. DATE06.2006.
- [MEL05] A. Mello, L. Moller, N. Calazans, F. Moraes. "MultiNoC: A multiprocessing system enabled by a Network on chip". Design automation and test in europe (DATE05). 2005.
- [MIL04] M. Millberg, E. Nilsson, R. Thid, S. Kumar, A. Jantsch. "The Nostrum Backbone - a Communication Protocol Stack for Networks on Chip". Proceedings of the 17th International Conference on VLSI Design (VLSID'04).
- [MIS04] P. Mishra, N. Dutt. "Functional validation of programmable architectures". Proceedings of the EUROMICRO Systems on digital design. DSD04. IEEE. 2004.
- [MOU03] I. Moussa, T. Grellier, G. Nguyen. "Exploring SW performance using SoC Transaction-level Modeling". In: Design, automation and test in europe. 2003.
- [MUK02] S. Mukherjee, P. Bannon, S. Lang, A. Spink, D. Webb, "The Alpha 21364 network architecture", IEEE Micro, 22(1), 2002.
- [MUR04] S. Murali, G. De Micheli. "Bandwidth-Constrained Mapping of Cores onto NoC Architectures". Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'04).
- [NIL04] E. Nilsson, J. Öberg. "Reducing power and latency in 2-D Mesh NoCs using Globally Pseudochronous Locally Synchronous Clocking". CODES ISSS 2004. Sweden
- [NUR02] J. Nurmi, I. Saastamoinen, D. Sigüenza. "IP Node for Future System-on-Chip Designs". Proceedings of the First IEEE International Workshop on Electronic Design, Test and Applications (DELTA.02).
- [NUR03] J. Nurmi, I. Saastamoinen, M. Alho. "Buffer Implementation for Proteo Networks-on-Chip". In: International Symposium on Circuits and Systems (ISCAS'03), May 2003.
- [OCP] OCP. "OCP home Page". Online: www.ocpip.org.
- [OGA03] O. Ogawa, S. Noyer, P. Chauvet, K. Shinohara, Y. Watanabe, H. Niizuma, T. Sasaki, Y. Takai. "A practical approach for Bus architecture optimization at transaction level". Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'03). 2003.
- [PAL04] G. Palermo, C. Silvano. "PIRATE: A Framework for Power/Performance Exploration of Network-on-Chip Architectures".
- [PAN03] P. Pande, C. Grecu, A. Ivanov, R. Saleh, "Design of a Switch for Network on Chip Applications," IEEE International Symposium on Circuits and Systems, ISCAS 2003, Vol. V, pp. 217-220, 2003.
- [PAN04] P. Pande, C. Grecu, M. Jones, A. Ivanov, R. Saleh. "Evaluation of MP-SoC Interconnect architectures: a Case Study". Proceedings of the 4th workshop on System-on-Chip for Real-time applications IWSOC04.2004 IEEE.
- [PES04] S. Pestana, E. Rijpkema, A. Radulescu, K. Goossens, O. Gangwal. "Cost-performance trade-offs in networks on chip: A simulation-based approach". DATE04.2004
- [POL03] F. Poletti, D. Bertozzi, L. Benini, A. Bogliolo, "Performance Analysis of Arbitration Policies for SoC Communication Architectures", Design Automation for Embedded System, 8, 189 210, 2003.
- [PAS02] S. Pasricha. "Transaction level modeling of SoC with SystemC 2.0". online: <http://www.ics.uci.edu/~sudeep/publications/SnugPDF2002.pdf>. 2002.
- [RHE04] C. Rhee, H. Jeong, S. Ha. "Many-to-Many Core-Switch Mapping in 2-D Mesh NoC Architectures". In Proceedings of the IEEE International Conference on Computer Design ICCD04. 2004
- [RIJ03] E. Rijpkema, K. Goossens, A. Radulescu, J. Dielissen, J. van Meerbergen, P. Wielage, E. Waterlander, "Trade offs in the design of a router with both guaranteed and best-effort services for networks on chip Design", Design Automation and Test in Europe (DATE) Conference, pp.350-355, 2003.
- [ROM05] E. Romero. "Técnicas para a verificação funcional eficiente de uma implementação RTL da camada banda base do protocolo bluetooth". 2005. Dissertação Mestrado em Engenharia Elétrica. São Paulo - Universidade de São Paulo.

- [SHE05] H. Shen, L. Ma, H. Zhang. "CRPG: a configurable random test-program generator for microprocessors" ISCAS IEEE 2005.
- [SUN03] Y. Sun, S. Kumar, A. Jantsh. "Simulation and evaluation for a Network on Chip architecture using Ns-2".
- [SRI05] K. Srinivasan, K. Chatha. *"ISIS: A Genetic algorithm based technique for custom on-chip interconnection network synthesis"*. In Proceedings of the 18th international conference on VLSI design. VLSI. 2005.
- [SYS02] Synopsys, Inc; CoWare, Inc; Fujitsu Microelectronics, Inc; Motorola; ST Microelectronics; Cadence Design Systems, Inc. *"Functional Specification for SystemC 2.0, version 2.0-Q"*. Capturado em: <http://www.systemc.org>, 2002.
- [TED05] L. Tedesco. "Uma proposta para a geração de tráfego e avaliação de desempenho para NoCs". Dissertação de mestrado em ciências da computação. Pontifícia Universidade Católica do Rio Grande do Sul. 2005.
- [THI03] R. Thid, M. Millberg, A. Jantsh. *"Evaluating NoC communication backbones with simulation"*. 2003
- [VAR02] G. Varatkar, R. Marculescu. *"Traffic Analysis for on-Chip Networks design for multimedia applications"*. IEEE DAC 2002.
- [WAS01] A. Wassal, M. Hasan. *"Low-power system-level design of VLSI packet switching fabrics"*, IEEE Transactions on Computer Aided Design, pp.723-738, June 2001.
- [WIN01] Wingard, D. *"MicroNetwork-based integration for SoCs"*. In: Design Automation Conference, 2001, pp. 673-677
- [WOL04] W. Wolf, J. Xu, J. Henkel, S. Chakradhar, T. Lv, *"A Case Study in Networks-on-Chip Design for Embedded Video"*, Design, Automation and Test in Europe, March, 2004.
- [ZEF04] C. Zeferino, M. Kreutz, A. Susin. "RASoC: A Router soft-core for Networks-on-chip". Proceedings of the design, automation and test in Europe conference and exhibition designers forum. DATE04.2004.
- [ZHA98] Y. Zhang, R. Chen, W. Ye, M. Irvin, *"System level interconnect power modeling"*, ASIC Conference 1998. Proceedings. Eleventh Annual IEEE International, pp.289-293, 1998.